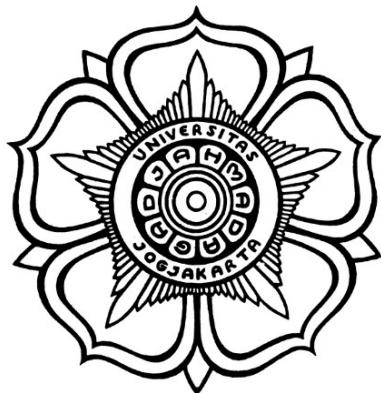


USULAN PENELITIAN S2

PERBAIKAN PENGENALAN GESTUR TANGAN TERHADAP KONDISI LINGKUNGAN BERINTENSITAS CAHAYA RENDAH MENGGUNAKAN RETINEX



ANTHONIUS ADI NUGROHO
19/448690/PPA/05773

PROGRAM MAGISTER ILMU KOMPUTER
DEPARTEMEN ILMU KOMPUTER DAN ELEKTRONIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS GADJAH MADA
YOGYAKARTA

2020

HALAMAN PENGESAHAN USULAN PENELITIAN S2

Judul Tesis : Perbaikan Pengenalan Gestur Tangan Terhadap Kondisi Lingkungan Berintensitas Cahaya Rendah Menggunakan Retinex

Nama Mahasiswa : Anthonius Adi N

NIM : 19/448690/PPA/05773

Proposal telah diuji pada tanggal _____ dan sudah diperbaiki sesuai saran penguji dan sudah disetujui para penguji.

Nama Penguji

Tanda Tangan

1. .

1.

2. .

2.

Yogyakarta,2020

Mengetahui,
Pembimbing

Pengusul



Dr. Raden Sumiharto, S.Si., M.Kom.
NIP. 197706252005011001

Anthonius Adi N.
NIM. 19/448690/PPA/05773

DAFTAR ISI

HALAMAN PENGESAHAN USULAN PENELITIAN S2	ii
DAFTAR ISI	iii
DAFTAR TABEL	vi
DAFTAR GAMBAR	vii
INTISARI	ix
I PENDAHULUAN	1
1.1. Latar Belakang Masalah	1
1.2. Rumusan Masalah	4
1.3. Batasan Masalah	4
1.4. Tujuan Penelitian	4
1.5. Manfaat Penelitian	4
II TINJAUAN PUSTAKA	6
III LANDASAN TEORI	10
3.1. Computer Vision	10
3.2. Citra Digital	10
3.2.1. Citra RGB	11
3.3. Operasi Morfologi	11
3.3.1. Erosi	11
3.3.2. Dilasi	12
3.3.3. <i>Opening</i>	12
3.3.4. <i>Closing</i>	12
3.4. <i>Hand Gesture Recognition</i>	12
3.4.1. <i>American Sign Language(ASL)</i>	12
3.5. Retinex(Single Scale Retinex)	13
3.5.1. Multiscale Retinex(MSR)	14
3.5.2. Multiscale Retinex Color Restoration(MSRCR)	15
3.6. MobileNet	15
3.6.1. Depthwise Separable Convolution	16
3.7. MobileNetV2	17
3.8. Convolutional Neural Network(CNN)	18
3.8.1. Local Receptive Fields	19
3.8.2. Shared Weight	21
3.8.3. Konvolusi	21
3.8.4. Fungsi Aktivasi	22
3.9. Evaluasi	23
3.9.1. Confussion Matrix	23

3.9.2. Average Precicion(AP)	23
IV METODOLOGI PENELITIAN	26
4.1. Alat dan Bahan	26
4.1.1. Alat	26
4.1.2. Bahan	26
4.2. Prosedur Kerja	27
4.2.1. Analisis dan Perancangan Sistem	27
4.2.2. Pra-proses	29
4.2.3. Pengumpulan Data	33
4.3. Proses Pelatihan	34
4.3.1. Pelatihan Gesture Recognition	34
4.3.2. Pelatihan Object Detection	35
4.4. Pengujian dan Evaluasi	36
4.4.1. Evaluasi Deteksi Tangan	36
4.4.2. Evaluasi Pengenalan Gestur Tangan	37
4.4.3. Pengujian PSNR	37
4.4.4. Pengujian Deteksi Tangan Menggunakan Retinex	37
4.4.5. Pengujian Pengenalan Gestur Tangan Menggunakan Retinex	39
4.4.6. Pengujian Sistem Keseluruhan	42
V IMPLEMENTASI	43
5.1. Data Collection	43
5.1.1. Implementasi Proses Akuisisi Citra	43
5.1.2. Implementasi Data Augmentasi	45
5.1.3. Pembuatan file CSV	47
5.2. Implementasi Data Labelling	48
5.3. Implementasi Pembuatan TFRecord file	49
5.4. Konfigurasi <i>Pipeline Object Detection</i>	52
5.5. Implementasi Training	54
5.5.1. Object Detection	54
5.5.2. Pengenalan Gestur	55
5.6. Implementasi Retinex	56
5.7. Evaluasi Model	57
5.7.1. PSNR	57
5.7.2. mAP	57
5.7.3. Kfold Cross Validation	58
5.8. Implementasi Pengujian	59

5.8.1. Pengujian Pengenalan Gestur Tangan	59
5.8.2. Pengujian Deteksi Objek	61
5.8.3. Pengujian Sistem Secara Keseluruhan	63
VI HASIL DAN PEMBAHASAN	66
6.1. Evaluasi Deteksi Tangan	66
6.2. Evaluasi Pengenalan Gestur Tangan	67
6.3. Pengujian SNR	67
6.4. Pengujian Deteksi Tangan	72
6.5. Pengujian Pengenalan Gestur Tangan	78
6.6. Pengujian Sistem Keseluruhan	93
VII KESIMPULAN DAN SARAN	94
7.1. Kesimpulan	94
7.2. Saran	94
DAFTAR PUSTAKA	95

DAFTAR TABEL

2.1	Tinjauan Pustaka	8
2.2	Lanjutan Tabel	9
3.1	Blok Arsitektur <i>Bottleneck MobileNetV2</i> (Sandler et al., 2018) . . .	17
4.1	Pengujian Deteksi Tangan	39
4.2	Pengujian Pengenalan Gestur Tangan	41
5.1	Akurasi 5-fold Cross Validation	58
6.1	Evaluasi mAP Model Object Detection	67
6.2	Parameter Nilai Retinex	67
6.3	Hasil Pengujian PSNR	69
6.4	Perbandingan Nilai Deteksi	70
6.5	Waktu Komputasi Retinex	70
6.6	Deteksi Subjek 1 Tanpa Retinex	74
6.7	Deteksi Subjek 1 Retinex	74
6.8	Perbandingan Deteksi Tangan Subjek_1	74
6.9	Deteksi Subjek 2 Tanpa Retinex	75
6.10	Deteksi Subjek 2 Retinex	75
6.11	Perbandingan Deteksi Tangan Subjek_2	75
6.12	Deteksi Subjek 3 Tanpa Retinex	76
6.13	Deteksi Subjek 3 Retinex	76
6.14	Perbandingan Deteksi Tangan Subjek_3	76
6.15	Perbandingan Pengenalan Gestur Tangan subjek_1	85
6.16	Perbandingan Pengenalan Gestur Tangan subjek_1	89
6.17	Perbandingan Pengenalan Gestur Tangan subjek_3	93

DAFTAR GAMBAR

3.1	Koordinat Citra Digital (Sarifudin.,2015)	10
3.2	Citra RGB	11
3.3	American Sign Language (Barczak et al., 2011)	13
3.4	Convolutional standard dan depthwise separable (Howard et al., 2017)	17
3.5	Struktur Dasar Pada <i>MobileNetV2</i> (Google AI Blog., 2018)	18
3.6	Ilustrasi Citra 28x28 Piksel (Nielsen., 2015)	19
3.7	Ilustrasi <i>local receptive fields</i> (Nielsen., 2015)	19
3.8	Ilustrasi Pergesaran <i>local receptive fields</i> (Nielsen., 2015)	20
3.9	Ilustrasi Pergeseran dengan Stride = 1 (Nielsen., 2015)	20
3.10	Confusion Matrix (Leonard., 2017)	23
3.11	Ilustrasi IoU(Hui., 2018)	24
3.12	Metrik Evaluasi COCO (Chen et al., 2015)	25
4.1	Alur Kegiatan Penelitian	27
4.2	Rancangan Sistem Deteksi Objek	28
4.3	Rancangan Sistem Pengenalan Gestur	28
4.4	Rancangan Sistem Keseluruhan	28
4.5	Ilustrasi Perbaikan Kontras Menggunakan Retinex (a) Citra Asli; (b) Citra Hasil Perbaikan(Petro et al., 2014)	33
4.6	Skema Pengambilan Dataset (Barczak et al., 2011)	34
4.7	Transfer Learning Pelatihan Gestur	35
4.8	Arsitektur Mobilenet V2 (Sandler et al., 2018)	36
4.9	Ilustrasi Pengujian Deteksi Tangan	38
4.10	Skema Kegiatan Pengujian Deteksi Tangan	38
4.11	Skema Kegiatan Pengujian Pengenalan Gestur Tangan	40
4.12	Ilustrasi Pengujian Pengenalan Gestur Tangan	40
5.1	Source code akuisisi citra	44
5.2	Implementasi Augmentasi	45
5.3	Source Code Augmentasi Citra	46
5.4	Source Code Pembuatan file CSV	47
5.5	Hasil File CSV	47
5.6	Proses Pelabelan Citra	48
5.7	hasil anotasi citra	48
5.8	Source Code Pengubahan XML ke CSV	49
5.9	Source Code Pembentukan TFRecord	51

5.10 konfigurasi pipeline <i>Object Detection</i>	53
5.11 Source Code Pelatihan <i>Object Detection</i>	54
5.12 Source Code Pelatihan <i>Object Detection</i>	55
5.13 Source Code Pelatihan <i>Pengenalan Gestur</i>	56
5.14 Source Code Retinex	57
5.15 Visualisasi IOU pada Citra	58
5.16 Source Code Pengenalan Gestur	60
5.17 Source Code Deteksi Objek	62
5.18 Source Code Sistem Keseluruhan	65
6.1 Perbandingan Hasil Citra Retinex Parameter Petro, 2014.	68
6.2 Perbandingan Hasil Citra Retinex Parameter Variasi 1.	68
6.3 Perbandingan Hasil Citra Retinex Parameter Variasi 2.	69
6.4 Variasi Sigma Terhadap Nilai PSNR	71
6.5 Variasi Sigma Terhadap Nilai Deteksi.	72
6.6 Subjek_1 (a: Subjek 2 tanpa Retinex, b: subjek 3 tanpa Retinex) . .	73
6.7 Subjek_1 (a: Subjek 2 dengan Retinex, b: subjek 3 dengan Retinex) .	73
6.8 Subjek_1 (a: Grafik tanpa Retinex, b: Grafik dengan Retinex) . . .	77
6.9 Subjek_2 (a: Grafik tanpa Retinex, b: Grafik dengan Retinex) . . .	77
6.10 Subjek_3 (a:Grafik tanpa Retinex, b: Grafik dengan Retinex) . . .	77
6.11 Subjek_1 (a: Kondisi 133 lux, b: Kondisi 77 lux, c:Kondisi 133 lux)	78

INTISARI

PERBAIKAN PENGENALAN GESTUR TANGAN TERHADAP KONDISI LINGKUNGAN BERINTENSITAS CAHAYA RENDAH MENGGUNAKAN RETINEX

Oleh

ANTHONIUS ADI NUGROHO

19/448690/PPA/05773

Bahasa isyarat adalah salah satu bentuk komunikasi non-verbal antar manusia yang memiliki makna tersendiri. *American Sign Language*(ASL) merupakan bentuk dari bahasa isyarat tangan yang digunakan oleh penyandang disabilitas untuk berkomunikasi satu sama lain. Penggunaan bahasa isyarat membantu penyelesaian hak atas informasi yang diberikan. Perkembangan ilmu yang sangat maju membantu seseorang dapat memahami suatu bahasa isyarat tanpa harus mempelajari hal tersebut. Salah satu bagian disiplin ilmu tersebut adalah pengenalan gestur, dimana di dalamnya terdapat bahasa isyarat yang dapat diterjemahkan dengan bantuan kamera.

Kamera digunakan sebagai alat yang mengimitasi mata manusia dalam mengenali sebuah gestur. Proses pengenalan bahasa isyarat memiliki beberapa hal penting di dalamnya. Intensitas cahaya merupakan salah satu faktor penting dalam pengambilan sebuah citra yang ditangkap oleh kamera, dimana merupakan suatu tantangan tersendiri ketika informasi citra tidak dapat ditangkap dengan jelas yang menyebabkan penurunan performa. Permasalahan ini dapat diatasi dengan menggunakan teori *Retinex* yang diambil dari kata retina dan cortex. *Retinex* mampu meningkatkan tingkat kecerahan citra dengan cara layaknya mata manusia yang dapat melihat walaupun dalam ruangan minim cahaya. Citra yang telah dilakukan perbaikan kontras akan dilakukan deteksi tangan kemudian akan dikenali bahasa isyarat tersebut menggunakan teknik *Convolution Neural Network*.

Kata kunci - Retinex, Hand Gesture Recognition, Object Detection

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Disabilitas adalah kelompok masyarakat yang memiliki keterbatasan yang dapat menghambat partisipasi dan peran serta mereka dalam kehidupan bermasyarakat. Penyandang disabilitas memiliki berbagai kategori, yaitu disabilitas fisik, intelektual, mental dan sensorik. menurut data dari kemenkes tahun 2015, presen-tase 3 teratas penyandang disabilitas di provinsi indonesia adalah 6.36% kesulitan melihat, 3.76% kesulitan berjalan dan 3.35% kesulitan mendengar (Kemenkes., 2018). Pemerintah Indonesia telah menandatangani konvensi tentang Hak-Hak Penyandang Disabilitas pada tanggal 30 Maret 2007 di New York. Adanya penan-datanganan tersebut menunjukan bangsa indonesia menghormati, melindungi, me-menuhi dan memajukan hak-hak penyandang disabilitas. Untuk itu perlu adanya dukungan dari masyarakat dalam mewujudkannya.

Bagi penyandang disabilitas, mereka memiliki hambatan akses dalam me-lakukan aktivitas sehari hari. Dengan adanya perkembangan pengetahuan dan tek-nologi, mereka mulai terbantu dan dapat melakukan aktivitas layaknya masyarakat pada umumnya. Mulai banyak penyandang disabilitas yang melakukan mobilitas tinggi dengan kursi roda, mengakses informasi dengan adanya penerjemah bahasa. Dengan adanya hal tersebut, penyandang disabilitas mendapat tempat dan peranan yang sama dengan masyarakat lainnya.

Teknologi yang semakin dewasa membuat mobilitas penyandang disabili-tas menjadi lebih tinggi. Kursi roda mungkin dapat digunakan untuk membantu penyandang bergerak dari suatu tempat ke tempat lain, namun bagi beberapa pe-nyandang disabilitas tertentu yang tidak memiliki kemampuan normal pada kondisi tangan atau lumpuh sebagian tidak dapat menggunakan kursi roda tersebut. Alhasil perlu adanya orang lain untuk membantu mennggerakan kursi roda tersebut.

Teknologi komputer dan robotika saat ini memiliki peranan penting dalam membantu sebuah permasalahan dari mulai kegiatan industri hingga kegiatan ma-syarakat. (Posada-Gómez et al., 2007) Membuat kursi roda pintar dengan kontrol

gestur tangan, namun pada penelitian tersebut memiliki kelemahan terhadap cahaya. Intensitas cahaya yang cenderung rendah membuat sistem tidak mampu mendeksi kontrol dari gestur tangan, sehingga hanya dapat digunakan dalam keadaan cahaya yang cenderung terang. Penelitian tersebut menggunakan teknologi pemrosesan citra yang dikombinasikan dengan elektronika dan mekanika untuk pergerakan kursi roda.

Penelitian yang menggunakan variasi cahaya juga dilakukan oleh (Saputra., 2016) dengan variasi metode *Retinex* dan variasi intensitas cahaya 439,75 lux, 273,25 lux, 150 lux dan 9 lux. Penelitian yang dilakukan untuk meningkatkan akurasi deteksi wajah menggunakan *adaboost haar-like*. Hasil pengujian pada penelitian ini menghasilkan nilai akurasi dengan rata-rata tertinggi dari 10 kali uji 96,67% pada kondisi 439,75 lux, 90,59% pada kondisi 273,25 lux, 42,29% pada kondisi 150 lux dan 0% pada kondisi 9 lux.

Pemrosesan citra memiliki beberapa hal fundamental permasalahan diantaranya adalah proses perbaikan citra. Perbaikan citra digunakan untuk memperbaiki sebuah citra yang bermasalah agar informasi citra terlihat lebih jelas secara visual maupun perhitungan. Penggunaan *image processing* sangat dibutuhkan untuk membantu menyelesaikan permasalahan dalam kehidupan sehari hari. Aplikasi dari implementasi *image processing* beberapa diantaranya adalah pengenalan dan deteksi pada sebuah objek. Untuk menyelesaikan permasalahan tersebut perlu membuat sistem yang tahan terhadap kondisi cahaya berintensitas rendah.

Pengenalan dan deteksi sebuah objek memiliki ruang lingkup yang sangat luas untuk dikembangkan, pada penelitian ini berfokus pada gesture recognition. Dalam pemrosesan citra algoritme pada gesture recognition dapat diimplementasikan pada komunikasi non verbal ataupun suatu gerakan yang dapat membantu seseorang menyelesaikan permasalahan. Pola pada gesture dapat dikenali oleh seorang dengan cara melihat gesture tersebut, hal yang sama terjadi pada kamera yang mengadopsi apa yang dilakukan mata manusia untuk mengenali sebuah objek. *Hand gesture* salah satu contoh implementasi *image processing* yang dapat dikembangkan untuk membantu seseorang menyelesaikan permasalahan dengan meng-

gunakan gestur tangan.

Salah satu faktor yang dapat menurunkan kualitas suatu citra yaitu pencahayaan dari sebuah citra, proses pengambilan citra dalam intensitas rendah akan menghasilkan citra yang buruk (Saputra., 2016). Implementasi algoritme untuk meningkatkan kualitas citra terhadap kondisi cahaya merupakan permasalahan yang menarik untuk diteliti. Kondisi cahaya pada image processing adalah sesuatu topik yang menantang dalam permasalahan *image processing* dan *computer vision* untuk meningkatkan visibilitas maupun kualitas yang lebih baik dari suatu citra. Beberapa penelitian terdahulu telah melakukan peningkatan algoritme pada kondisi cahaya yang minim untuk suatu citra (Loh et al., 2019). Berkembangnya algoritme dalam ruang lingkup kondisi cahaya yang minim bertujuan mendapatkan kualitas kontras yang lebih jelas untuk dapat dilakukan komputasi lebih lanjut.

Retinex merupakan metode yang diusulkan oleh Land dengan memodelkan pencahayaan dan persepsi warna berdasarkan penglihatan mata manusia. Mata manusia dapat membedakan sebuah objek sekalipun dalam kondisi intensitas cahaya yang rendah. Metode *Retinex* terus mengalami berbagai pengembangan dari *Single Scale Retinex* hingga *Multiscale Retinex* berupaya untuk memperoleh keseimbangan kontras dalam pencahayaan berintensitas rendah(Saputra., 2016).

Peningkatan kontras menjadi salah satu solusi untuk meningkatkan visibilitas dari sebuah citra. Penelitian ini menggunakan algoritme *Multiscale Retinex Color Restoration* untuk meningkatkan kontras serta *object detection* untuk melakukan segmentasi pada suatu citra, kemudian untuk melakukan pengenalan sebuah gesture yang telah dilakukan perbaikan kontras, akan dilanjutkan dengan metode *Convolutional Neural Network*.

Implementasi dari peningkatan kontras untuk pengenalan gestur tangan ini diharapkan dapat dibawa untuk menyelesaikan permasalahan pada sistem yang bergantung pada cahaya. Beberapa sistem yang dimaksud seperti kursi roda pintar, ataupun sistem lainnya seperti pengenalan wajah dan pengenalan gestur maupun pendekripsi suatu objek yang menggunakan kamera.

1.2. Rumusan Masalah

Berdasarkan rumusan masalah yang telah dijelaskan sebelumnya, maka rumusan masalah pada penelitian ini adalah nilai akurasi dari deteksi wajah dengan intensitas cahaya 150 lux hanya mampu 42,39%, sehingga deteksi dan pengenalan sebuah objek sangat dipengaruhi oleh cahaya. Intensitas cahaya yang semakin rendah menyebabkan penurunan akurasi suatu deteksi dan pengenalan. Hal tersebut dapat berdampak pada performa sebuah sistem yang semakin menurun.

1.3. Batasan Masalah

Penelitian ini memiliki batasan masalah yang bertujuan untuk tidak memperluas pokok bahasan. Batasan masalah yang digunakan dalam penelitian ini adalah sebagai berikut:

1. Acuan dataset menggunakan *American Sign Language*, dengan sepuluh klasifikasi yaitu angka 0 hingga angka 9.
2. Data yang digunakan menggunakan dataset private dan dataset public.
3. Proses pengujian akan dilakukan dengan 3 subjek dengan warna kulit yang berbeda.
4. Penurunan intensitas cahaya dikurangi sebesar 50% lux sebelumnya hingga nilai lux kurang dari 50 lux.

1.4. Tujuan Penelitian

Penelitian yang dilakukan bertujuan untuk mengatasi permasalahan sistem yang bergantung pada pencahaayaan dengan mengimplementasikan algoritme *Retinex* pada sebuah sistem deteksi dan pengenalan gestur tangan, dengan tingkat intensitas cahaya yang bervariasi.

1.5. Manfaat Penelitian

Penelitian ini diharapkan dapat memberi manfaat berupa:

1. Memberikan fitur tambahan untuk sistem yang memiliki permasalahan dengan pencahayaan dalam intensitas cahaya rendah.
2. Meningkatkan akurasi pengenalan gestur tangan dan deteksi objek pada kondisi cahaya berintensitas rendah.

BAB II

TINJAUAN PUSTAKA

Penelitian untuk menghasilkan algoritme deteksi dan pengenalan gestur sudah banyak dilakukan, sehingga banyak metode yang memiliki tingkat keberhasilan tinggi. Namun pada pemrosesan citra tingkat keberhasilan tidak hanya didukung pada algoritme deteksi dan pengenalan saja, namun kualitas pada citra tersebut harus memiliki kualitas yang bagus pula sehingga dapat menghasilkan akurasi yang tinggi(Saputra., 2016).

Salah satu hal yang mempengaruhi kualitas tersebut adalah kondisi cahaya yang mengarah pada objek, sehingga mempengaruhi hasil yang ditangkap oleh kamera. Peningkatan kontras sangat dibutuhkan untuk meningkatkan kualitas image dalam kondisi cahaya yang minim. Salah satu pendekatan yang populer dipakai adalah Retinex. (Tanaka et al., 2019) mengimplementasikan Retinex pada prepossessing untuk meningkatkan kontras citra. Percobaan yang dilakukan dengan membandingkan hasil segmentasi citra asli dan citra dengan proses preprocessing. Hasil yang didapat secara kualitatif terlihat setelah dilakukan segmentasi Gaussian Mixture Model. Hasil segmentasi objek dengan citra preprocessing mendapatkan foreground yang jelas daripada citra asli. Namun pada citra dengan Retinex peningkatan kontras berubah menghasilkan warna yang tidak natural.

Peningkatan pencahayaan pada citra dapat dilakukan dengan banyak metode. Selain Retinex, metode yang paling populer adalah *Histogram Equalization*. (Srinivasan, 2016) Melakukan perbandaingan antara algoritme *Retinex* dan *Histogram Equalization*. Pada penelitian tersebut *Retinex* yang di implementasikan adalah *Single Scale Retinex* (SSR) dan *Multiscale Retinex* (MSR). Kemudian *Histogram Equalization* yang diimplementasikan adalah BBHE, DSIE dan RLBHE Kedua algoritme tersebut diuji menggunakan citra yang sama kemudian dilakukan perbandingan.

Pada citra keluaran *Histogram Equalization* (BBHE dan DSIE), citra mengalami meningkatkan kontras dengan keterbatasan beberapa fitur yang tidak terlihat, Keluaran RLBHE mereduksi kualitas dari piksel. Kemudian dengan citra keluaran

Retinex, citra mengalami peningkatan kontras dengan fitur yang terlihat lebih jelas daripada citra *Histogram Equalization*.

Algoritme Retinex telah mengalami pengembangan untuk meningkatkan kualitas citra pada kondisi lingkungan tertentu. Saputra ditahun 2016 melakukan perbandingan variasi Retinex untuk peningkatan deteksi wajah yang dilakukan pada kondisi ruangan berintensitas rendah. Algoritme *Adaptive Multiscale Retinex* (AMSR), *Multiscale Retinex Color Restoration*(MSRCR) di implementasikan pada 4 kondisi cahaya. Hasil peningkatan maksimal yang didapatkan pada MSRCR dapat meningkatkan 1,46 kali dan ASMR mampu meningkatkan 1,11 kali. Namun peningkatan tersebut terjadi pada parameter intensitas 273,25 lux(Saputra., 2016).

Pengembangan MSR juga dilakukan oleh (Shen et al., 2017) dengan menambah layer preprocessing/post processing pada proses konvolusi citra pada CNN menjadi MSR-Net. Kemudian dibandingkan dengan MSRCR dan beberapa metode lainnya. Hasil MSR-Net mengalami peningkatan kontras citra dengan warna natural dibandingkan MSRCR dan beberapa metode lainnya.

Pengenalan sebuah gestur adalah yang menentukan hasil akhir dari sistem. CNN merupakan salah satu algoritme yang sering dijadikan solusi untuk proses klasifikasi dalam *machine learning*. Beberapa penelitian menggunakan CNN dengan beberapa variasi parameter. Penelitian (Yingxin et al., 2017) menggunakan CNN untuk mengenali sebuah gestur tangan dengan dataset *Cambridge hand gesture datasets* (CHGD). Pada penelitian ini dilakukan parameter illuminasi cahaya pada beberapa kondisi. Pengenalan gestur menghasilkan angka presentase yang tinggi sebesar 94.1%. Proses dalam preprocessing menggunakan *canny edge* untuk menghilangkan efek illuminasi. *Canny edge* sangat membantu untuk kondisi illuminasi cahaya dibandingkan dengan algoritme CNN saja yang menggunakan citra asli di dapat hasil 70.0%.

Pada citra gelap informasi atau fitur fitur penting dari sebuah citra akan tersembunyi. Informasi dalam sebuah citra penting untuk merepresentasikan sebuah citra itu sendiri. Untuk mendapatkan informasi tersebut pada penelitian (Loh et al., 2019) berfokus pada perbaikan citra dengan tujuan memperoleh fitur untuk men-

dukung sistem visi otomatis dimana sebuah citra memiliki kontras dan pencahayaan yang rendah. Dalam penelitian ini memodelkan sebuah citra dengan cahaya rendah sebagai distribusi peningkatan fungsi lokal menggunakan proses gaussian yang dilatih pada saat runtime menggunakan data referensi yang dihasilkan dari sebuah CNN. CNN sendiri dilatih menggunakan dengan data yang sangat besar berdasarkan statistik pencahayaan. Sehingga proses pembelajaran dapat mempelajari hubungan antara fitur dengan piksel. Dengan demikian refrensi yang dihasilkan melatih gaussian proses untuk melakukan representasi fitur dengan benar.

Dasar-dasar penelitian sebelumnya yang menjadi tinjauan pustaka pada penelitian ini dirangkum dalam Tabel 2.1.

Tabel 2.1 Tinjauan Pustaka

No	Nama	Penelitian	Metode	Hasil
1	(Loh et al., 2019)	<i>Low-light image enhancement using Gaussian Process for features retrieval</i>	<i>Gaussian process and Convolutional Neural Network</i>	Citra dengan kontras yang sangat rendah dapat di perbaiki menggunakan gaussian prosses dan CNN untuk memperoleh detail informasi dari sebuah objek
2	(Tanaka et al., 2019)	<i>Retinex Foreground Segmentation for Low Light Environments</i>	<i>Retinex dan Gaussian Mixture Model</i>	Secara kualitatif citra yang dihasilkan setelah melalui <i>preprocessing</i> algoritme Retinex dapat meningkatkan pencahayaan dari sebuah citra
3	(Yingxin et al., 2017)	<i>A Robust Hand Gesture Recognition Method via Convolutional Neural Network</i>	<i>Edge detection dan Convolutional Neural Network</i>	Pengenalan gestur tangan mendapatkan hasil 94.1% digabungkan dengan proses deteksi tepi.

Tabel 2.2 Lanjutan Tabel

4	(Shen et al., 2017)	<i>MSR-Net: Low-light Image Enhancement using Deep Convolutional Network</i>	MSR-net	Hasil dari implementasi algoritme MSR-Net di bandingkan dengan MSRCR dan beberapa algoritme lain mendapatkan kontras yang lebih tinggi dengan warna yang natural dibandingkan algoritme lain.
5	(Saputra., 2016)	Perbandingan Varian Metode Multiscale Retinex Untuk Peningkatan Akurasi Deteksi Wajah Adaboost HAAR-like	Variasi metode Multiscale Retinex	Kondisi 439,75 lux MSRCR meningkatkan akurasi 1,31 kali dan AMSR hanya 1,11 kali. Kondisi 273,25 lux MSRCR 1,46 kali dan AMSR 1,31 kali. Kondisi 150 lux MSRCR 1,38 kali dan AMSR 0,97 kali. Kondisi 9 lux kedua algoritme tidak dapat mendeteksi wajah sebuah citra.
6	(Srinivasan, 2014)	Perbandingan antara <i>Retinex</i> dan <i>Histogram Equalization</i>	SSR, MSR, BBHE, DSIHE, RLBHE	Citra keluaran <i>Retinex</i> memiliki <i>output</i> kontras yang baik tanpa menghilangkan fitur pada citra.
7	(Arabi, 2019)	Mendeteksi peralatan konstruksi yang diaplikasikan pada embedded system dan PC	<i>SSD MobileNet</i>	Untuk semua device mAP > 90%
8	(Huang et al., 2019)	<i>Hand Gesture Recognition with Skin Detection and Deep Learning Method</i> <i>Hand Gesture Recognition with Skin Detection and Deep Learning Method</i>	<i>Skin detection</i> dan <i>Convolutional Neural Network</i>	Menghasilkan akurasi 98,41%

BAB III

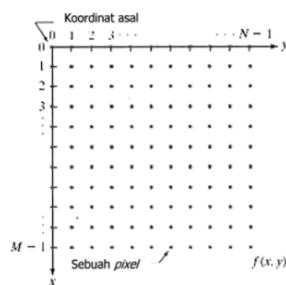
LANDASAN TEORI

3.1. Computer Vision

Computer vision adalah teknologi yang membuat computer dapat melihat layaknya mata manusia. Penggunaan computer vision tanpa kita sadari telah di implementasikan dalam membantu menyelesaikan persoalan sehari hari. Contoh implementasi *computer vision* antara lain *face recognition*, *object classification*, *medical imaging*, *gesture recognition*, *video surveillance*, *3D reconstruction*. Pada pengolahan citra, sebuah citra memiliki fitur-fitur dari penting yang digunakan sebagai informasi saat pengolahan. Namun untuk mendapatkan suatu hasil, beberapa tahap proses harus dilakukan seperti *preprocessing*, ekstraksi ciri, *post-processing* dan sebagainya.

3.2. Citra Digital

Citra dapat didefinisikan sebagai fungsi $f(x, y)$ berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial dan amplitudo f di titik koordinat (x, y) dinamakan tingkat keabuan dari suatu citra. Apabila nilai (x, y) dan f secara keseluruhan berhingga dan bernilai diskrit maka citra tersebut adalah citra digital. Citra digital dalam bentuk matrik dapat dilihat pada Persamaan 3.1 dan posisi koordinat citra digital dapat dilihat pada gambar 3.1(Sarifudin.,2015).



Gambar 3.1 Koordinat Citra Digital (Sarifudin.,2015)

$$f(x, y) \begin{bmatrix} f(0, 0) & f(0, 1) & \dots & f(0, n-1) \\ f(1, 0) & f(1, 1) & \dots & f(1, n-1) \\ \dots & \dots & \dots & \dots \\ f(m-1, 0) & f(m-1, 1) & \dots & f(m-1, n-1) \end{bmatrix} \quad (3.1)$$

3.2.1. Citra RGB

Citra RGB disebut juga sebagai citra berwarna, citra ini menyajikan tiga layer warna yaitu Red, Green dan Blue. Setiap piksel dari citra RGB merupakan gabungan dari variasi nilai intensitas tiga warna dasar yaitu merah(R), hijau(G), biru(B). Tiga warna tersebut dikodekan dengan 8 bit,dengan total ketiganya $3 \times 8 = 24$ bit. Sehingga variasi warna sebanyak $2^{24} = 16.777.216$ variasi warna (Sarifudin., 2015). Contoh citra RGB dapat dilihat pada Gambar 3.2.



Gambar 3.2 Citra RGB

3.3. Operasi Morfologi

3.3.1. Erosi

Operasi erosi adalah operasi penipisan objek yang terdapat pada citra biner. Operasi erosi dilakukan dengan cara mengurangi piksel pada kontur dari objek citra sesuai dengan kernel. Operasi erosi dinotasikan pada Persamaan 3.6 (Hidayatullah., 2017).

$$A \ominus B = A^c \oplus B^c \quad (3.2)$$

3.3.2. Dilasi

Operasi dilasi adalah operasi penebalan objek yang terdapat pada citra biner. Operasi ini dilakukan dengan menambah piksel pada kontur dari objek sesuai dengan kernel. Operasi ini berguna untuk menghaluskan citra dan menutupi lubang-lubang yang kosong. Operasi dilasi dinotasikan pada Persamaan 3.7 (Hidayatullah., 2017).

$$A \oplus B = t \in Z^2 : t = a + b, a \in A, b \in B \quad (3.3)$$

3.3.3. Opening

Operasi *opening* merupakan operasi yang biasa digunakan untuk memperhalus kontur citra serta menghilangkan lubang-lubang kecil pada citra. Operasi ini terdiri dari 2 dua tahap yaitu erosi kemudian dilanjutkan dilasi. Operasi erosi berguna untuk menghilangkan noise pada citra karena struktur latar depan yang berukuran kecil tereliminasi, sedangkan dilasi digunakan untuk menebalan citra. Operasi opening dinotasikan dalam Persamaan 3.8(Hidayatullah., 2017).

$$A \bullet B = (A \ominus B) \oplus B \quad (3.4)$$

3.3.4. Closing

Operasi *closing* merupakan kebalikan dari operasi *opening*. Terdiri dari dua tahap yaitu operasi dilasi kemudian dilanjutkan dengan operasi erosi. Kegunaan operasi ini adalah untuk menutupi lubang yang kosong pada citra dengan menggunakan dilasi kemudian dilakukan erosi citra untuk menipiskan suatu citra. Operasi closing dinotasikan pada Persamaan 3.9 (Hidayatullah., 2017).

$$A \bullet B = (A \oplus B) \ominus B \quad (3.5)$$

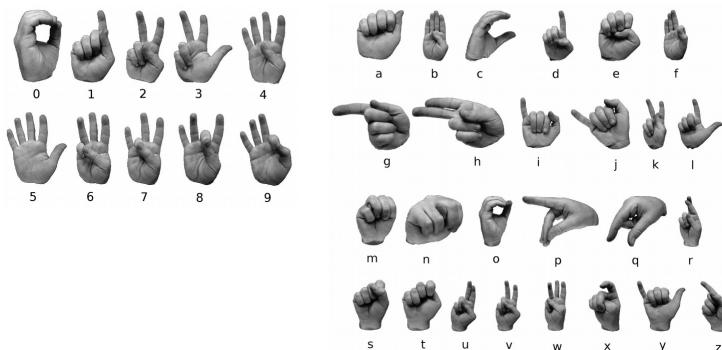
3.4. Hand Gesture Recognition

3.4.1. American Sign Language(ASL)

Bahasa isyarat merupakan media komunikasi utama bagi kaum difabel khusunya penyandang tunarungu di dunia. Setiap negara mempunyai bahasa isyarat

masing masing. Dengan adanya bahasa isyarat, penyandang tunarungu dapat melakukan komunikasi antar sesama penderita maupun berkomunikasi dengan setiap orang. Bahasa isyarat mulai banyak digunakan pada siaran televisi, seperti acara debat ataupun channel berita dengan dibantu penerjemah bahasa isyarat. Dengan begitu memberi mereka hak yang sama untuk mendapatkan informasi.

Salah satu bahasa isyarat yang digunakan adalah *American Sign Language*(ASL). *American Sign Language* menjadi salah satu alat bantu pembelajaran komunikasi untuk penggunanya. penggunaan bahasa isyarat dilakukan dengan gerakan gestur tangan yang memiliki makna untuk setiap pose. Pada penelitian ini, bahasa isyarat yang digunakan untuk menguji sistem adalah *American Sign Language*. Pada gestur ASL, mempunyai beberapa bentuk gestur tangan huruf dan angka seperti pada Gambar 3.4.



Gambar 3.3 American Sign Language (Barczak et al., 2011)

3.5. Retinex(Single Scale Retinex)

Algoritme *Retinex* merupakan algoritme yang berusaha untuk mempertahankan ketetapan warna dimana warna suatu objek yang dilihat dalam keadaan pencahayaan yang berbeda(Aribowo et al., 2009). Algoritme *Retinex* ini sering juga disebut dengan *Single Scale Retinex* (SSR) karena hanya memiliki satu kanal. Menurut Land sebuah citra terbentuk sebagai perkalian reflektansi dan iluminasi. Berdasarkan teori *Retinex* dapat dituliskan secara matematis seperti Persamaan 3.10 berikut(Parihar & Singh., 2018)(Petro et al., 2014):

$$I(x, y) = L(x, y) \cdot R(x, y) \quad (3.6)$$

$$\log I(x, y) = \log L(x, y) + \log R(x, y) \quad (3.7)$$

$$\log R(x, y) = \log I(x, y) - \log L(x, y) \quad (3.8)$$

dimana

$$L(x, y) = F(x, y) * I(x, y) \quad (3.9)$$

$$F(x, y) = \frac{1}{2\pi\sigma^2} e^{-(\frac{x^2+y^2}{2\sigma^2})} \quad (3.10)$$

$$\int F(x, y) dx dy = 1 \quad (3.11)$$

sehingga

$$\log R = \log I(x, y) - \log[F(x, y) * I(x, y)] \quad (3.12)$$

Keterangan :

$I(x, y)$: representasi dari kumpulansinyal dari citra asli.

$R(x, y)$: representasi dari komponen reflektansi dari objek.

$L(x, y)$: representasi komponen pencahayaan yang memenuhi Persamaan 3.12.

$F(x, y)$: *gaussian function* yang memenuhi Persamaan 3.13.

Bentuk logaritmik pada Persamaan 3.11 digunakan untuk memisahkan antara komponen pencahayaan dan komponen reflektansinya. Pemisahan ini dilakukan untuk mendapatkan informasi dari citra tersebut sehingga hasil *Single Scale Retinex* dapat dilihat pada Persamaan 3.17, dimana $\log R = R_{SSRi}$ adalah hasil *Single Scale Retinex* pada kanal i (Parihar & Singh., 2018)(Petro et al., 2014).

$$R_{SSRi}(x, y) = \log I_i(x, y) - \log[F(x, y) * I_i(x, y)] \quad (3.13)$$

3.5.1. Multiscale Retinex(MSR)

Multiscale Retinex merupakan pengembangan dari *Single Scale Retinex* dengan menggabungkan beberapa scale yang berbeda dengan pembobotan tertentu, dimana bobot tersebut jika dijumlahkan menghasilkan nilai 1. Sehingga nilai sangat mempengaruhi detail warna dari sebuah citra. Bentuk matematis dari *multiscale Retinex* dapat dilihat pada Persamaan 3.15 dan 3.16 berikut(Petro et al., 2014):

$$R_{MSRi} = \sum_{n=1}^N \omega_n R_{SSRi} = \omega_n [\log I(x, y) - \log(F_n(x, y) * I_i(x, y))] \quad (3.14)$$

Keterangan :

R_{MSR_i} : hasil *Multiscale Retinex* pada kanal i

ω_n : bobot pada skala n

Pada SSR dapat menyediakan kompresi rentang dinamis dan penampakan warna, namun keduanya tidak secara bersamaan. Oleh karena itu *Multiscale Retinex* menggabungkan kompresi rentang dinamis dari *Single Scale Retinex* dengan penampakan warna(Aribowo et al., 2009).

3.5.2. Multiscale Retinex Color Restoration(MSRCR)

MSRCR merupakan pengembangan dari metode MSR yang mampu memperbaiki kualitas citra yang berhubungan dengan pencahayaan yaitu dengan mempertahankan *color constancy*. *Color constancy* berfungsi untuk mempertahankan komposisi warna suatu citra tetap terlihat sama walaupun kondisi pencahayaan yang berbeda-beda (Saputra., 2016). Perhitungan MSRCR dapat dilihat pada Persamaan 3.19, 3.20 dan 3.21(Petro et al., 2014).

$$R_{MSRCR_i}(x, y) = C_i(x, y) R_{MSR_i}(x, y) \quad (3.15)$$

$$C_i(x, y) = \beta \log[\alpha I'_i(x, y)] \quad (3.16)$$

$$I'(x, y) = \frac{I_i(x, y)}{\sum_{i=0}^S I_i(x, y)} \quad (3.17)$$

Keterangan :

$i = 3$ kanal warna RGB

α = konstanta kontrol ketidaklinieran

β = konstanta gain

3.6. MobileNet

MobileNet merupakan salah satu arsitektur CNN yang dibangun oleh *Google* dan dikhususkan untuk *mobile device* karena proses komputasi yang ringan. Perbedaan arsitektur *MobileNet* dan CNN pada umumnya terdapat pada penggunaan layer konvolusi dengan ketebalan sesuai input dari citra. Arsitektur SSD *MobileNet* menggunakan *Depthwise layer* dan *Pointwise layer*.

3.6.1. Depthwise Separable Convolution

Depthwise separable convolution merupakan kunci utama untuk membangun arsitektur *neural network*. Ide utamanya adalah mengganti operator konvolusi secara menyeluruh dengan memisahkan konvolusi menjadi dua lapisan terpisah. Layer pertama adalah *depthwise convolution*, layer ini memiliki *light-weight filter* dengan menggunakan *single convolutional filter* untuk setiap input channel. Layer kedua menggunakan 1x1 convolution yang disebut dengan *pointwise convolution* dimana layer ini membentuk fitur baru melalui perhitungan kombinasi linear dari input channel.

Pada konvolusi biasa memiliki input citra $D_F \times D_F \times M$ fitur map pada F dan menghasilkan $D_G \times D_G \times N$ fitur map pada G , dimana D_F adalah tinggi dan lebar dari fitur map. M merupakan jumlah input channel(input depth). D_G merupakan tinggi dan lebar keluaran fitur map dan N adalah keluaran channel dari fitur map(*output depth*). Kemudian kernel pada konvolusi biasa $D_K \times D_K \times M \times N$, dimana D_K adalah dimensi kernel, M jumlah input channel dan N jumlah keluaran channel. Total cost dari konvolusi biasa memiliki ketergantungan pada M pada Persamaan 3.22.

$$D_K \bullet D_K \bullet M \bullet N \bullet D_F \bullet D_F \quad (3.18)$$

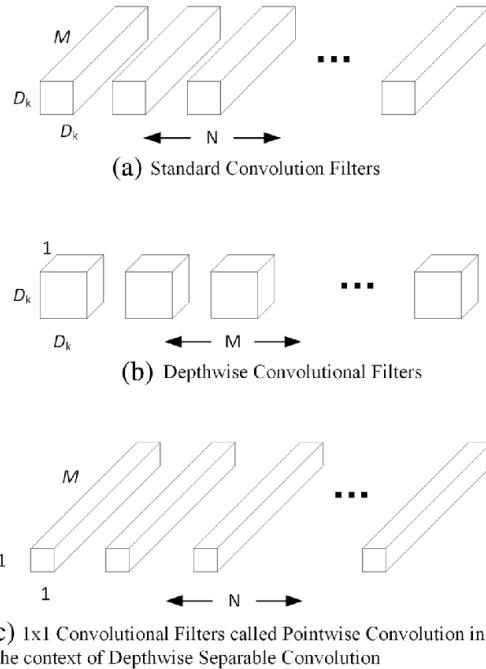
Depthwise separable convolution terbentuk dari layer *depthwise convolution* dan *pointwise convolution*. *Depthwise convolution* sangat efisien daripada konvolusi biasa, namun pada layer *depthwise* tersebut tidak menghasilkan fitur baru, oleh karena itu digunakan tambahan layer *pointwise convolution* untuk menghitung linier kombinasi dari keluaran *depthwise convolution* menggunakan 1x1 convolution , sehingga menghasilkan fitur baru. Total cost dari *depthwise convolution* pada Persamaan 3.23.

$$D_K \bullet D_K \bullet M \bullet D_F \bullet D_F \quad (3.19)$$

Kombinasi kedua layer tersebut dinamakan *depthwise separable convolution* yang memiliki total cost Persamaan 3.24 :

$$D_K \bullet D_K \bullet M \bullet N \bullet D_F \bullet D_F + M \bullet N \bullet D_F \bullet D_F \quad (3.20)$$

MobileNet menggunakan 3×3 *depthwise separable convolution* yang mana mereduksi komputasi 8 hingga 9 kali lebih cepat dari konvolusi biasa. Perbedaan konvolusi biasa dan *depthwise separable convolution* digambarkan pada Gambar 3.5(Howard et al., 2017).



Gambar 3.4 Convolutional standard dan depthwise separable (Howard et al., 2017)

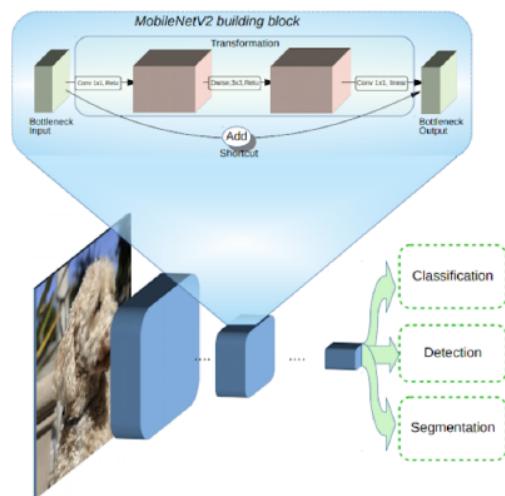
3.7. MobileNetV2

Perbedaan *MobileNet* versi kedua merupakan pengembangan dari *MobileNet* versi pertama. Pada *MobileNet* memiliki modifikasi arsitektur linier *bottleneck* dan *shortcut* koneksi antar *bottleneck* seperti Gambar 3.6. Sehingga arsitektur dalam *MobileNetV2* menjadi *bottleneck depth-separable convolution residuals*. Arsitektur *bottleneck* dituliskan pada Tabel 3.1.

Tabel 3.1 Blok Arsitektur *Bottleneck* *MobileNetV2* (Sandler et al., 2018)

Input	Operator	Output
$h \times w \times k$	1×1 Conv2D, ReLU6	$h \times w \times (tk)$
$h \times w \times tk$	3×3 dwise s=s, ReLU6	$\frac{h}{s} \times \frac{w}{s} \times (tk)$
$\frac{h}{s} \times \frac{w}{s} \times tk$	linear 1×1 Conv2D	$\frac{h}{s} \times \frac{w}{s} \times k'$

Tabel 3.1 menunjukkan blok arsitektur *bottleneck* *MobileNetV2* terdiri dari *fully convolution* dengan 32 filter diikuti 19 layer *residual bottleneck*. Kernel yang digunakan menggunakan ukuran 3x3 (standard untuk modern network) dan menggunakan dropout dan normalisasi selama pelatihan(Sandler et al., 2018).



Gambar 3.5 Struktur Dasar Pada *MobileNetV2* (Google AI Blog., 2018)

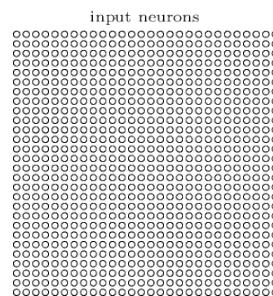
Pada bagian *bottleneck* terdapat *input* dan *output*, sedangkan layer di bagian dalam mengenkapsulasi kemampuan model untuk merubah input dari piksel ke gambar. Tambahan *shortcut* antar *bottleneck* memungkinkan saat melakukan *training* menjadi lebih cepat dengan peningkatan akurasi yang lebih baik(Google AI Blog., 2018).

3.8. Convolutional Neural Network(CNN)

Convolutional Neural Network (CNN) merupakan jaringan syaraf tiruan yang memiliki arsitektur khusus yang dapat bekerja dengan baik dalam penggunaannya pada pemrosesan data berbentuk larik, seperti data runtun waktu dan citra digital. Dalam arsitekturnya CNN menggunakan operasi matematika konvolusi pada layer jaringan. Operasi konvolusi antara a dan b disimbolkan dengan $(a \times b)$. *Convolutional neural network* pada penerapannya menggunakan tiga ide dasar yaitu local receptive fields, shared weights, dan pooling(Nielsen., 2015).

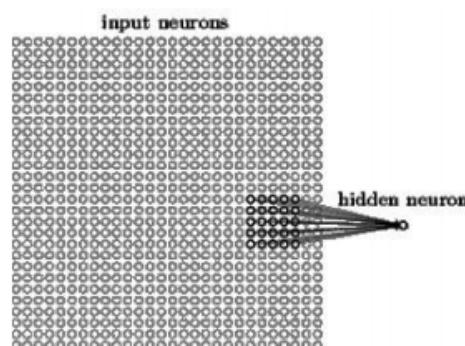
3.8.1. Local Receptive Fields

Pada jaringan syaraf tiruan dengan *fully-connected*, input dari jaringan digambarkan dengan garis vertikal dari kumpulan neuron. Pada CNN, input digambarkan sebagai persegi dengan ukuran $o \times o$ sesuai dengan ukuran input yang diberikan. Misalkan terdapat input dengan ukuran 28×28 neuron, yang berkesesuaian dengan 28×28 intensitas piksel pada citra seperti terlihat pada Gambar 3.7.



Gambar 3.6 Ilustrasi Citra 28x28 Piksel (Nielsen., 2015)

Seperti jaringan syaraf tiruan pada umumnya, setiap neuron dari input terhubung dengan layer dari hidden neuron. Sedikit berbeda, pada CNN neuron input tidak terhubung secara *fully-connected* dengan setiap hidden neuron, tetapi hanya region lokal kecil dari input terhubung dengan sebuah hidden neuron (Nielsen., 2015). Pada Gambar 3.8 terlihat bahwa hanya sebagian kecil region yang *localized* dari input neuron yang terhubung dengan hidden neuron.

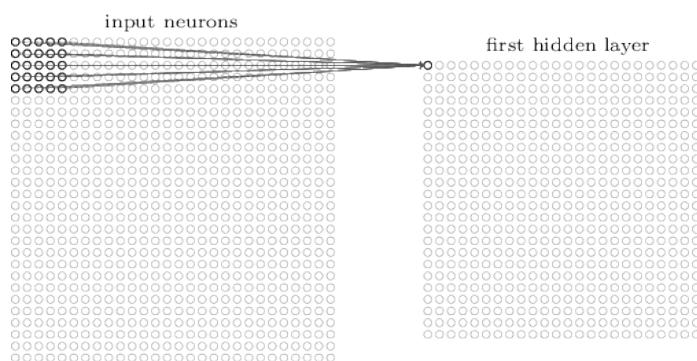


Gambar 3.7 Ilustrasi local receptive fields (Nielsen., 2015)

Pada Gambar 3.9 setiap neuron pada hidden layer terhubung dengan region berukuran 5×5 piksel pada neuron input. Region pada input layer inilah yang dina-

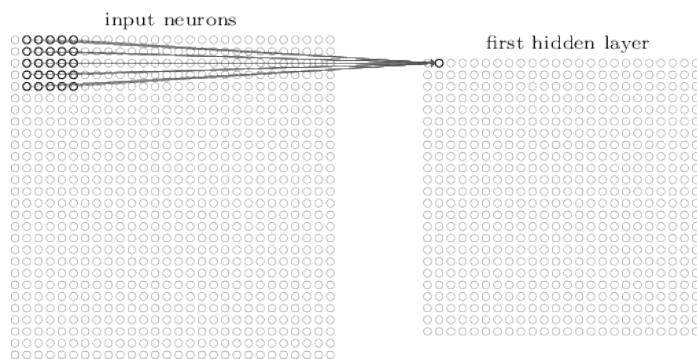
makan dengan *local receptive fields* dan setiap koneksi memiliki bobot yang akan disesuaikan seiring dengan proses pelatihan. Selain itu hidden neuron juga memiliki dan mempelajari bias secara keseluruhan. Oleh karena itu, setiap hidden neuron dilatih untuk menganalisa masing-masing local receptive fields yang bersesuaian.

Tahap selanjutnya, local receptive fields akan digeser (slide) sepanjang ukuran citra dari posisi paling kiri atas hingga kanan bawah. Setiap local receptive fields akan memiliki pasangan hidden neuron yang berbeda pada hidden layer. Ilustrasi proses pergeseran *local receptive fields* terdapat pada Gambar 3.9 dan Gambar 3.10 berikut.



Gambar 3.8 Ilustrasi Pergeseran *local receptive fields* (Nielsen., 2015)

Selanjutnya *local receptive fields* akan digeser sebanyak satu piksel ke kanan seperti pada Gambar 3.8 dan terhubung dengan hidden neuron yang berbeda dengan *local receptive fields* sebelumnya.



Gambar 3.9 Ilustrasi Pergeseran dengan Stride = 1 (Nielsen., 2015)

Pergeseran piksel dengan stride = 1 di ilustrasikan pada Gambar 3.9. Proses ini berlangsung hingga *local receptive fields* berada pada posisi piksel paling kanan

bawah. Sehingga setelah proses slide dilakukan akan terbentuk hidden layer dengan ukuran sesuai dengan ukuran local receptive fields dan panjang pergeseran (stride) yang digunakan (Nielsen, 2015). Maka jika citra dengan ukuran 28×28 piksel dan digunakan *local receptive fields* dengan ukuran 5×5 , dan digeser dengan stride = 1, maka akan terbentuk hidden layer dengan ukuran 24×24 neuron.

3.8.2. Shared Weight

Pada CNN, sebuah neuron pada hidden layer yang terhubung dengan 5×5 neuron pada input layer (sesuai dengan ukuran *local receptive fields* yang digunakan). Hal ini menunjukkan bahwa neuron tersebut memiliki sebuah bias dan matriks bobot dengan ukuran 5×5 yang menghubungkan 5×5 neuron inputnya. Matriks bobot ini, dalam CNN, disebut sebagai kernel. Matriks bobot pada jaringan syaraf tiruan biasa sedikit berbeda dengan matriks bobot pada CNN, yaitu nilai matriks bobot pada CNN bernilai sama untuk setiap 24×24 neuron pada hidden layer. Hal tersebut menunjukkan bahwa semua neuron pada hidden layer akan mendeteksi fitur yang sama dengan lokasi pada input citra yang berbeda (Nielsen., 2015).

3.8.3. Konvolusi

Lapisan yang pertama kali akan dilewati oleh data masukan adalah lapisan konvolusi, bertujuan untuk memperoleh feature map yang merepresentasikan masukan. Parameter yang digunakan untuk menentukan ukuran feature map keluaran berupa filter, ukuran dari filter, besarnya langkah pergeseran filter pada operasi konvolusi atau biasa disebut stride, dan ukuran padding. Cara menghitung ukuran dari keluaran operasi konvolusi ditunjukkan pada Persamaan 3.25 dan 3.26 berikut(Nielsen., 2015):

$$H_0 = \frac{H - F + 2P}{S} + 1 \quad (3.21)$$

$$W_0 = \frac{W_i - F + 2P}{S} + 1 \quad (3.22)$$

Keterangan :

H_0 = tinggi fitur map keluaran

W_0 = lebar fitur map keluaran

H_i = tinggi fitur map masukan

W_i = lebar fitur map masukan

F = ukuran filter

P = ukuran padding

S = ukuran stride

Nilai semua elemen pada persamaan tersebut harus merupakan bilangan bulat karena akan merepresentasikan suatu ukuran feature map. Apabila terdapat salah satu nilai yang bukan merupakan bilangan bulat, nilai tersebut harus dibulatkan dengan melakukan pembulatan kebawah. Adapun cara menghitung nilai keluaran dari proses konvolusi dapat dilihat pada Persamaan 3.27 berikut(Nielsen., 2015):

$$O_{mn} = \sum I_{i,j}^k \cdot F_{i,j} \quad (3.23)$$

Keterangan :

O_{mn} = elemen matriks keluaran pada baris ke-m kolom ke -n.

$I_{i,j}^k$ = elemen matriks masukan bagian ke-k pada baris ke-i kolom ke-j.

$F_{i,j}$ = elemen matriks filter pada baris ke-i kolom ke -j.

3.8.4. Fungsi Aktivasi

Salah satu faktor signifikan mempengaruhi kinerja algoritme Convolutional Neural Network adalah penerapan fungsi aktivasi dalam jaringan. Fungsi ini membantu menyelesaikan permasalah-permasalahan yang bersifat non-trivial dalam suatu jaringan dengan cara mengambil sebuah nilai dan melakukan operasi matematika. Fungsi aktivasi ini diletakkan di perhitungan akhir dari keluaran feature map atau setelah layer konvolusi dan subsampling layer. Fungsi aktivasi yang sering digunakan adalah Rectified Linear Unit (ReLU) karena fungsi ini lebih cepat daripada fungsi aktivasi non-linear lainnya seperti sigmoid atau tanh. Fungsi ReLU dapat dilihat pada Persamaan 3.28 dan 3.29(Nielsen., 2015):

$$f(x) = ReLU(x) = \max(0, x) \quad (3.24)$$

$$f(x) = 0 \text{ jika } x \leq 0 \text{ atau } x \text{ jika } x > 0 \quad (3.25)$$

3.9. Evaluasi

3.9.1. Confusion Matrix

Pengukuran evaluasi dari pengujian dapat dilakukan menggunakan *confusion matrix*. tabel *confusion matrix* merupakan tabel klasifikasi yang bersifat prediktif seperti ditunjukkan pada Gambar 3.11. Evaluasi menggunakan confusion matrix dapat digunakan untuk mengukur nilai akurasi dengan Persaman 3.30.

		Predicted values	
		Negative	Positive
Actual values	Negative	TN	FP
	Positive	FN	TP

Gambar 3.10 Confusion Matrix (Leonard., 2017)

$$\text{akurasi} = \frac{TP}{TP + FP} \quad (3.26)$$

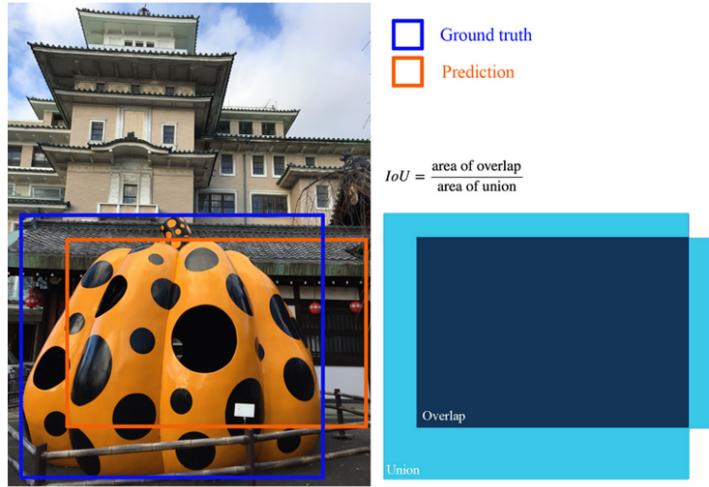
Dengan :

TP = True Positif, objek berupa gestur tangan dan terkenali

FP = False Positif, objek berupa gestur tangan tapi tidak terkenali

3.9.2. Average Precicion(AP)

Average Precicion adalah metrik populer untuk melakukan evaluasi pada objek detektor seperti *Faster-RCNN*, *SSD*, dan lainnya. AP menghitung nilai rata rata nilai presisi untuk nilai recall 0 hingga 1 (Hui., 2018). Evaluasi deteksi menggunakan konsep IoU(*intersection over union*) IoU menghitung interseksi kedua bounding boxes dimana terdiri dari ground truth dan prediksi dari bounding box digambarkan pada Gambar 3.12 dan perhitungan IoU pada Persamaan 3.31.



Gambar 3.11 Ilustrasi IoU(Hui., 2018)

$$IoU = \frac{\text{area_of_overlap}}{\text{area_of_union}} \quad (3.27)$$

IoU yang digunakan untuk memprediksi suatu bounding box, apabila nilai IoU melebihi *threshold* yang ditentukan maka akan terdeteksi sebuah objek. Sebaliknya jika IoU kurang dari batas yang ditentukan maka menandakan tidak terdeteksi apapun pada citra.

Terdapat 4 kategori dalam penentuan deteksi yaitu, FN(*False Negative*) terdapat objek namun tidak mendekksi apapun. FP(*False Positive*) tidak terdapat objek namun hasilnya terdeteksi. TP(*True Positive*) terdapat objek dan terdeteksi. TN(*True Negative*) akan terjadi jika tidak ada objek dan tidak terdeteksi apapun. Berdasarkan nilai yang didapat maka diketahui presisi dan recall menggunakan Persamaan 3.32 dan Persamaan 3.33 (Hui., 2018).

$$\text{presisi} = \frac{TP}{TP + FP} \quad (3.28)$$

$$\text{recall} = \frac{TP}{TP + FN} \quad (3.29)$$

Metrik untuk mengukur detektor dari *object detection* menggunakan mAP (*mean average precision*). Nilai mAP dan AP dalam COCO dataset tidak memiliki perbedaan, dimana mAP adalah rata rata dari setiap AP yang memiliki nilai IoU sebagai

threshold. Total metrik evaluasi pada COCO memiliki 12 metrik seperti pada Gambar 3.13 (Chen et al., 2015).

```

Average Precision (AP) :
    AP                                % AP at IoU=.50:.05:.95 (primary challenge metric)
    APIoU=.50                      % AP at IoU=.50 (PASCAL VOC metric)
    APIoU=.75                      % AP at IoU=.75 (strict metric)

AP Across Scales:
    APsmall                         % AP for small objects: area < 322
    APmedium                        % AP for medium objects: 322 < area < 962
    APlarge                          % AP for large objects: area > 962

Average Recall (AR) :
    ARmax=1                         % AR given 1 detection per image
    ARmax=10                        % AR given 10 detections per image
    ARmax=100                       % AR given 100 detections per image

AR Across Scales:
    ARsmall                          % AR for small objects: area < 322
    ARmedium                         % AR for medium objects: 322 < area < 962
    ARlarge                           % AR for large objects: area > 962

```

Gambar 3.12 Metrik Evaluasi COCO (Chen et al., 2015)

BAB IV

METODOLOGI PENELITIAN

4.1. Alat dan Bahan

4.1.1. Alat

Penelitian ini menggunakan beberapa peralatan yang digunakan untuk membantu kegiatan dari mulai pengumpulan data hingga pengujian sistem, beberapa peralatan tersebut adalah sebagai berikut :

1. PC/Laptop dengan spesifikasi processor Intel (R) Core i5-8300H CPU @2,4 GHz, GPU NVIDIA 1050, RAM 8 GB, sistem operasi Linux 64 bit.
2. Webcam Logitech C270
3. Dimmer
4. Kain
5. Lux Meter

Laptop merupakan peralatan yang paling utama yang digunakan untuk melakukan proses pengolahan citra dengan webcam sebagai alat untuk menangkap citra. Lux meter digunakan untuk mengukur intensitas cahaya saat pengambilan data dan pengujian data dengan menurunkan intensitas cahaya yang diatur menggunakan dimmer. Penggunaan kain bersifat opsional untuk latar belakang yang bervariasi.

4.1.2. Bahan

Bahan yang digunakan untuk melakukan penelitian ini berupa dataset, diantaranya sebagai berikut:

1. Dataset gestur tangan ASL
2. Dataset gambar tangan

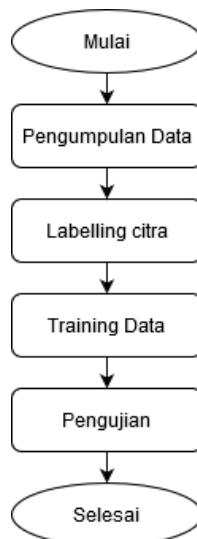
Dataset gestur tangan ASL merupakan dataset dari *Massey University* yang digunakan untuk melatih model pengenalan gestur tangan, sementara dataset gambar

tangan merupakan dataset untuk melakukan pelatihan deteksi tangan. Dataset gambar tangan diambil dari 3 subjek yang diambil gambar menggunakan webcam.

4.2. Prosedur Kerja

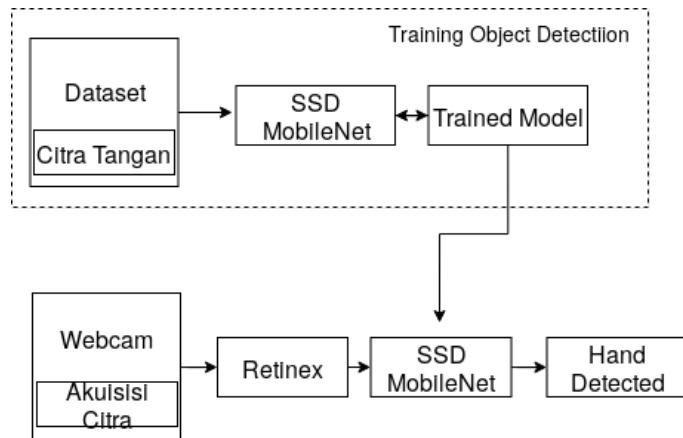
4.2.1. Analisis dan Perancangan Sistem

Alur penelitian yang akan dilakukan dalam penelitian ini memiliki beberapa tahapan diantaranya pengambilan dataset gestur, training data dan tahap paling akhir adalah pengujian. Alur kegiatan penelitian yang akan dilakukan secara garis besar ditunjukkan pada Gambar 4.1.

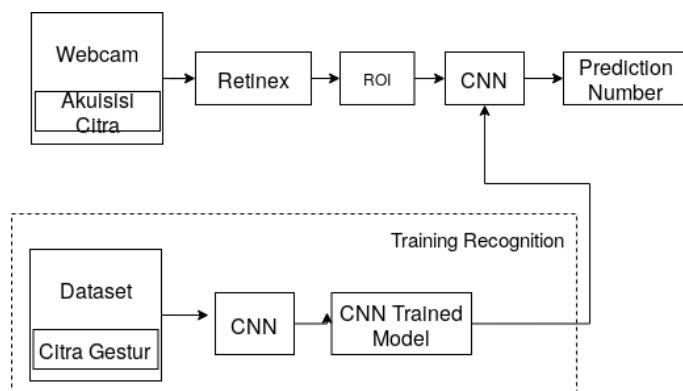


Gambar 4.1 Alur Kegiatan Penelitian

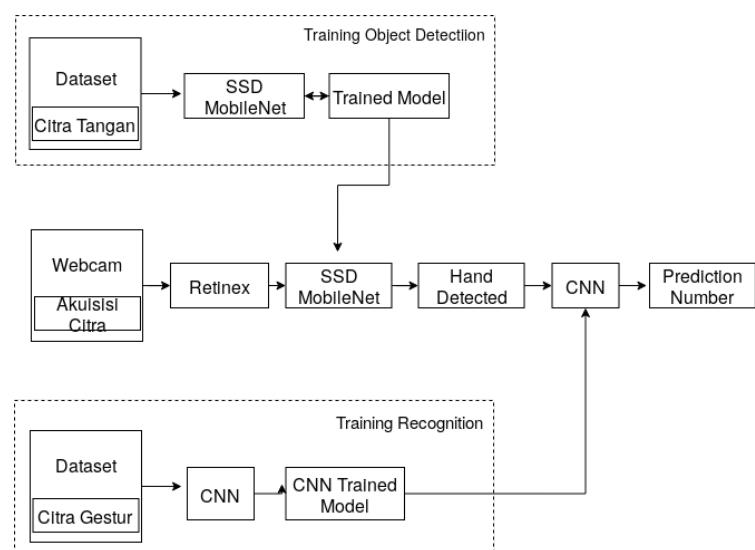
Rancangan pengujian sistem ditunjukkan pada Gambar 4.2 yang dimulai dengan *input* dari webcam secara *realtime*, sehingga setiap frame akan di proses pada tahap berikutnya. Setiap frame dengan citra RGB akan dilewatkan pada algoritme *Retinex* untuk dilakukan perbaikan kualitas citra dengan harapan meningkatkan kontras pada sebuah citra. Citra yang mengalami peningkatan kontras kemudian sistem akan melakukan deteksi tangan dengan *SSD MobileNet* yang kemudian citra tangan tersebut akan diambil sebagai ROI. Hasil segmentasi akan diklasifikasikan dengan model CNN yang telah di training sebelumnya. Keluaran dari sistem ini akan menampilkan klasifikasi angka yang terdeteksi.



Gambar 4.2 Rancangan Sistem Deteksi Objek



Gambar 4.3 Rancangan Sistem Pengenalan Gestur



Gambar 4.4 Rancangan Sistem Keseluruhan

Penelitian ini memiliki 3 bagian yaitu deteksi objek yang di gambarkan pada gambar 4.2, pengenalan gestur pada gambar 4.3 dan sistem keseluruhan pada gambar 4.4. Peranan Retinex pada penelitian ini merupakan solusi untuk mengatasi permasalahan intensitas cahaya, sehingga pada dataset yang dilatih tidak perlu menggunakan variasi intensitas yang rendah. Penggunaan SSD dengan arsitektur *MobileNet* bertujuan untuk mempercepat waktu komputasi supaya tidak terlalu berat dengan harapan lebih cepat. *Convolutional Neural Network* sendiri digunakan untuk melakukan ekstraksi fitur dari sebuah citra. Ekstraksi fitur ini sangat sering digunakan dalam pemrosesan sebuah citra untuk memperoleh informasi.

4.2.2. Pra-proses

Tahap pra-proses pada penelitian ini terdapat pada proses *enhancement* yang dilakukan oleh *Retinex*. Citra dengan intensitas cahaya rendah akan dilakukan proses perbaikan dahulu sebelum dilakukan proses deteksi maupun pengenalan. *Input* citra untuk proses *Retinex* menggunakan 3 kanal(R,G,B).

Proses pertama yang dilakukan adalah membuat filter *gaussian* menggunakan Persamaan 3.14 dengan permisalan $\sigma = 10$.

$$\begin{bmatrix} \frac{1}{2\pi(10)^2} \times \exp^{-\frac{(-1)^2+1^2}{2(10)^2}} & \frac{1}{2\pi(10)^2} \times \exp^{-\frac{0^2+1^2}{2(10)^2}} & \frac{1}{2\pi(10)^2} \times \exp^{-\frac{1^2+1^2}{2(10)^2}} \\ \frac{1}{2\pi(10)^2} \times \exp^{-\frac{(-1)^2+0^2}{2(10)^2}} & \frac{1}{2\pi(10)^2} \times \exp^{-\frac{0^2+0^2}{2(10)^2}} & \frac{1}{2\pi(10)^2} \times \exp^{-\frac{1^2+0^2}{2(10)^2}} \\ \frac{1}{2\pi(10)^2} \times \exp^{-\frac{(-1)^2+(-1)^2}{2(10)^2}} & \frac{1}{2\pi(10)^2} \times \exp^{-\frac{0^2+(-1)^2}{2(10)^2}} & \frac{1}{2\pi(10)^2} \times \exp^{-\frac{1^2+(-1)^2}{2(10)^2}} \end{bmatrix}$$

Filter *gaussian* yang dihasilkan menggunakan $\sigma=10$ adalah sebagai berikut.

$$\begin{bmatrix} 0,001575 & 0,001583 & 0,001575 \\ 0,001583 & 0,001591 & 0,001583 \\ 0,001575 & 0,001583 & 0,001575 \end{bmatrix}$$

Berdasarkan Persamaan 3.15, integral dari filter *gaussian* harus sama dengan 1. Integral dari filter tersebut dapat dicari dengan menjumlahkan semua elemen. Pada kasus ini nilai integral atau jumlahan dari elemen adalah 0.0142288. Apabila hasil dari integral tersebut tidak sama dengan 1 maka akan dikalikan menggunakan formula berikut.

$$new_a_{(x,y)} = a_{(x,y)} \times \frac{1}{total}$$

$$new_a_{(-1,1)} = 0,001575 \times \frac{1}{0,142288} = 0,110740$$

$$new_a_{(-1,0)} = 0,001583 \times \frac{1}{0,142288} = 0,111295$$

Perhitungan nilai filter *gaussian* dilakukan untuk setiap distribusi kernel sehingga membentuk filter dengan nilai baru. Nilai filter yang baru menghasilkan hasil integral sama dengan 1.

$$\begin{bmatrix} 0,110740 & 0,111295 & 0,110740 \\ 0,111295 & 0,111853 & 0,111295 \\ 0,110740 & 0,111295 & 0,110740 \end{bmatrix}$$

Filter *gaussian* ini akan dikonvolusikan dengan input citra, contoh citra yang akan dikonvolusikan sebagai berikut.

$$I_{(x,y)} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 120 & 125 & 124 & 126 & 0 \\ 0 & 126 & 135 & 146 & 189 & 0 \\ 0 & 170 & 187 & 200 & 210 & 0 \\ 0 & 189 & 188 & 197 & 221 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} * F_{(x,y)} = \begin{bmatrix} 0,110740 & 0,111295 & 0,110740 \\ 0,111295 & 0,111853 & 0,111295 \\ 0,110740 & 0,111295 & 0,110740 \end{bmatrix}$$

Perhitungan konvolusi untuk mendapatkan dimensi yang sama dengan citra *input* digunakan *zero padding*. Hasil konvolusi di ilustrasikan pada perhitungan berikut.

$$O_{(1,1)} = (0*0,110740) + (0*0,111295) + (0*0,110740) + (0*0,111295) + (120*0,111853) + (125*0,111295) + (0*0,110740) + (126*0,111295) + (135*0,110740) = 56,307698$$

$$O_{(2,1)} = (0*0,110740) + (0*0,111295) + (0*0,110740) + (120*0,111295) + (125*0,111853) + (124*0,111295) + (126*0,110740) + (135*0,111295) + (146*0,110740) = 86,284315$$

Konvolusi dilakukan dengan pergeseran 1 piksel sehingga menghasilkan citra dengan dimensi yang sama. Hasil konvolusi ($O_{(x,y)}$) merupakan nilai dari Persamaan 3.13 yang mendapatkan nilai sebagai berikut.

$$O_{(x,y)} = \begin{bmatrix} 56,307698 & 86,284315 & 93,934305 & 65,097310 \\ 95,945406 & 148,09183 & 160,21034 & 110,66494 \\ 110,65489 & 170,91206 & 185,90262 & 129,36380 \\ 81,692776 & 125,77508 & 133,77841 & 92,165220 \end{bmatrix}$$

$$\log O_{(x,y)} = \begin{bmatrix} 1,750568 & 1,935932 & 1,972825 & 1,813563 \\ 1,982024 & 2,170531 & 2,204691 & 2,044011 \\ 2,043971 & 2,232773 & 2,269286 & 2,111813 \\ 1,912183 & 2,099594 & 2,126386 & 1,964568 \end{bmatrix}$$

Nilai akhir dari *Single Scale Retinex* adalah pengurangan antara hasil logaritmik citra asli dengan logaritmik citra hasil konvolusi dimana dituliskan dalam Persamaan 3.17. Berikut adalah hasil *Single Scale Retinex* ($\sigma=10$).

$$R_{(SSR(\sigma=10))} = \begin{bmatrix} 0,328612 & 0,160977 & 0,120596 & 0,286806 \\ 0,118345 & -0,040198 & -0,04033 & 0,232450 \\ 0,186477 & 0,039068 & 0,031743 & 0,210405 \\ 0,364278 & 0,174563 & 0,168079 & 0,379824 \end{bmatrix}$$

Perhitungan *Multiscale Retinex* jumlahan dari setiap *Single Scale Retinex*. *Single Scale Retinex* dengan $\sigma=65$ dan $\sigma=180$ didapatkan nilai sebagai berikut.

$$R_{(SSR(\sigma=65))} = \begin{bmatrix} 0,329257 & 0,161283 & 0,120802 & 0,287442 \\ 0,118596 & -0,040252 & -0,040368 & 0,232871 \\ 0,186859 & 0,039126 & 0,031776 & 0,210871 \\ 0,364991 & 0,174929 & 0,168434 & 0,380586 \end{bmatrix}$$

$$R_{(SSR(\sigma=180))} = \begin{bmatrix} 0,329271 & 0,161289 & 0,120806 & 0,287455 \\ 0,118601 & -0,040253 & -0,040369 & 0,232879 \\ 0,186867 & 0,0391275 & 0,0317764 & 0,210880 \\ 0,365006 & 0,1749365 & 0,1684419 & 0,380602 \end{bmatrix}$$

Multiscale Retinex memiliki bobot pada setiap skala, dimana jika bobot dijumlahkan = 1. Bobot tersebut dikalikan dengan citra untuk masing masing skala, sehingga mendapatkan nilai berikut dengan masing masing bobot $\omega_n = 0.3, 0.3, 0.4$.

$$R_{(SSR(\sigma=10)) * 0,3} = \begin{bmatrix} 0,098583 & 0,048293 & 0,036178 & 0,086041 \\ 0,035503 & -0,012059 & -0,012101 & 0,069735 \\ 0,055943 & 0,011720 & 0,009523 & 0,063121 \\ 0,109283 & 0,052368 & 0,050423 & 0,113947 \end{bmatrix}$$

$$R_{(SSR(\sigma=65)) \times 0,3} = \begin{bmatrix} 0,098777 & 0,048385 & 0,036240 & 0,086232 \\ 0,035578 & -0,012075 & -0,012110 & 0,069861 \\ 0,056057 & 0,011737 & 0,009532 & 0,063261 \\ 0,109497 & 0,052478 & 0,050530 & 0,114175 \end{bmatrix}$$

$$R_{(SSR(\sigma=180)) \times 0,4} = \begin{bmatrix} 0,131708 & 0,064515 & 0,048322 & 0,114982 \\ 0,047440 & -0,016101 & -0,016147 & 0,093151 \\ 0,074746 & 0,015651 & 0,012710 & 0,084352 \\ 0,146002 & 0,069974 & 0,067376 & 0,152240 \end{bmatrix}$$

Setiap nilai *Single Scale Retinex* dijumlahkan seperti pada Persamaan 3.18, kemudian hasil dari penjumlahan akan dilakukan normalisasi dengan hasil akhir yang dibulatkan.

$$R_{(MSR)} = \begin{bmatrix} 0,329069 & 0,161194 & 0,120742 & 0,287256 \\ 0,118523 & -0,040236 & -0,040360 & 0,232748 \\ 0,186748 & 0,039109 & 0,031766 & 0,210735 \\ 0,364783 & 0,174822 & 0,168331 & 0,380363 \end{bmatrix}$$

$$\text{Normalisasi} = \frac{(ABS(Nilai_{(x,y)}) - Nilai_min)}{(Nilai_max - Nilai_min)} \times 255$$

$$R_{(MSR)} = \begin{bmatrix} 224 & 122 & 98 & 198 \\ 96 & 0 & 1 & 165 \\ 138 & 48 & 44 & 152 \\ 246 & 130 & 126 & 255 \end{bmatrix}$$

Gambar 4.3 menunjukkan ilustrasi peningkatan kualitas citra dengan intensitas cahaya rendah dikenai algoritme *Retinex* sehingga menghasilkan citra dengan pengkatan kontras.



Gambar 4.5 Ilustrasi Perbaikan Kontras Menggunakan Retinex (a) Citra Asli; (b) Citra Hasil Perbaikan(Petro et al., 2014)

4.2.3. Pengumpulan Data

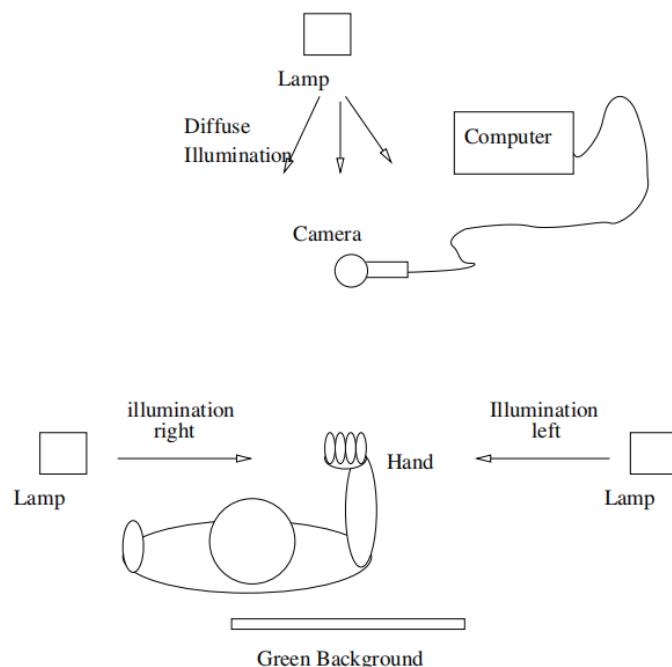
Data pelatihan dibagi menjadi dua, yaitu dataset citra gestur tangan yang mengacu pada ASL dan dataset citra tangan biasa untuk *object detection*. Kedua dataset ini merupakan dua hal yang berbeda tujuan. Dataset ASL digunakan untuk model pengenalan gestur ASL dan dataset citra tangan digunakan untuk model deteksi tangan.

Dataset gestur tangan ASL menggunakan dataset public dari *Massey University*. Dataset tersebut berisi 2425 citra yang dibagi dalam 36 kelas, yaitu 26 untuk huruf A-Z dan 10 kelas untuk angka 0-9. Pengambilan dataset dilakukan oleh 5 individu dengan variasi cahaya yang berbeda(Barczak et al., 2011). Acuan dalam penelitian ini menggunakan dataset angka 0 sampai 9, dengan citra yang sudah di segmentasi. Dataset ini akan dilatih dalam CNN untuk pengenalan gestur tangan. Pengambilan dataset *Massey University* dilakukan menggunakan *green screen* sehingga mudah untuk disegmentasi, setup pengambilan dataset dilakukan dengan skema seperti Gambar 4.4 (Barczak et al., 2011).

Pengambilan dataset berikutnya berbeda dengan dataset ASL, dataset berikutnya diambil menggunakan webcam untuk keperluan deteksi tangan, citra diambil dari 10 orang dengan random pose. Dataset ini digunakan untuk pelatihan pada *object detection*. Pengambilan dataset dilakukan masing masing 100 capture untuk setiap orang, sehingga pada dataset ini diperoleh 1000 citra.

Proses pengumpulan dataset tidak menghiraukan nilai intensitas cahaya ka-

rena permasalahan pencahayaan sudah diatasi pada algoritme *Retinex*, namun akan tetap diukur kondisi intensitas pada lingkungan tersebut menggunakan lux meter. Penggunaan dataset ASL dari *Massey University* tidak memiliki keterangan nilai intesitas cahaya, namun berdasarkan citra tersebut cenderung memiliki nilai intensitas yang tinggi, dimana dapat dilihat pada isi dataset. Tujuan dari dataset yang dilatih hanya untuk membuat model klasifikasi dan deteksi tanpa memperhitungkan intensitas cahaya rendah.



Gambar 4.6 Skema Pengambilan Dataset (Barczak et al., 2011)

4.3. Proses Pelatihan

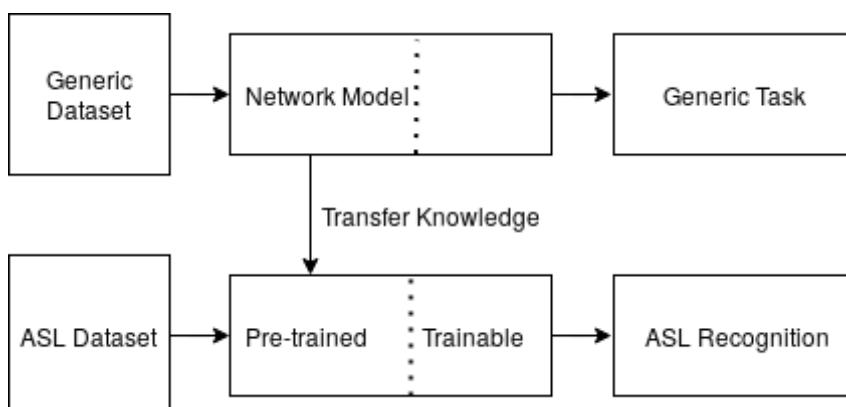
Proses pelatihan dalam penelitian ini dibagi menjadi dua bagian, yaitu pelatihan object detection dan pelatihan gesture recognition.

4.3.1. Pelatihan Gesture Recognition

Dataset pada pelatihan *gesture recognition* menggunakan dataset dari *Massey Univertisy* yang di labeli sesuai acuan ASL dari angka 0 hingga 9. Jumlah dataset adalah 700 citra kemudian dibagi yang akan dibagi dalam *train* dan *test* meng-

gunakan *k-fold cross validation* dengan $k = 5$. Proses pelatihan untuk pengenalan gestur tangan menggunakan teknik *transfer learning* yang memiliki kelebihan pada waktu pelatihan dan penggunaan dataset yang cenderung sedikit. *Pre-trained* model yang digunakan adalah *MobileNetV2*. *Pre-trained* model memiliki pengetahuan dari dataset yang telah dilatih sebelumnya, pada *MobileNetV2* telah dilatih menggunakan dataset dari *ImageNet*, kemudian pengetahuan tersebut akan di pindahkan ke model baru dan dilatih sesuai dataset baru, sehingga memiliki keluaran dalam bentuk klasifikasi atau tugas sesuai yang diinginkan.

Masukan citra memiliki dimensi 224x224 piksel yang terdiri 3 channel RGB, sehingga node *input* berjumlah 50176 dengan range 0-255 untuk setiap piksel. Arsitektur jaringan pada pelatihan ini dilakukan dengan mengganti bagian blok klasifikasi dari *MobileNetV2* menjadi klasifikasi dataset ASL. Arsitektur transfer learning dapat dilihat pada Gambar 4.5.



Gambar 4.7 Transfer Learning Pelatihan Gestur

4.3.2. Pelatihan Object Detection

Proses pelatihan pada *object detection* menggunakan *pre-trained model SSD MobilenetV2 COCO* yang sudah di latih sebelumnya dengan dataset COCO. Dataset yang baru akan dilatih menggunakan *pre-trained* model, sehingga dengan pengetahuan yang sudah ada dilatih agar menghasilkan model baru sesuai dengan *output* yang diinginkan. Dataset yang akan dilatih sejumlah 1000 citra yang telah dilabeli secara manual dengan pembagian 80% *training* dan 20% *testing*. Arsitektur yang digunakan dalam pelatihan *object detection* dapat dilihat pada Gambar 4.6.

Input	Operator	<i>t</i>	<i>c</i>	<i>n</i>	<i>s</i>
$224^2 \times 3$	conv2d	-	32	1	2
$112^2 \times 32$	bottleneck	1	16	1	1
$112^2 \times 16$	bottleneck	6	24	2	2
$56^2 \times 24$	bottleneck	6	32	3	2
$28^2 \times 32$	bottleneck	6	64	4	2
$14^2 \times 64$	bottleneck	6	96	3	1
$14^2 \times 96$	bottleneck	6	160	3	2
$7^2 \times 160$	bottleneck	6	320	1	1
$7^2 \times 320$	conv2d 1x1	-	1280	1	1
$7^2 \times 1280$	avgpool 7x7	-	-	1	-
$1 \times 1 \times 1280$	conv2d 1x1	-	k	-	-

Gambar 4.8 Arsitektur Mobilenet V2 (Sandler et al., 2018)

4.4. Pengujian dan Evaluasi

Pengujian dan evaluasi adalah tahap untuk mengukur performa dari sebuah sistem. Pengujian ini akan dibagi menjadi 6 tahap, yaitu evaluasi deteksi tangan, evaluasi pengenalan gestur, pengujian deteksi tangan menggunakan *Retinex*, pengujian pengenalan gestur tangan menggunakan *Retinex*, pengujian SNR dan pengujian keseluruhan sistem. Setiap pengujian yang menggunakan *Retinex* dilakukan hal yang sama, yaitu dengan menurunkan intensitas cahaya.

4.4.1. Evaluasi Deteksi Tangan

Evaluasi model deteksi tangan dilakukan setelah *training* menggunakan metrik *mean average precision* (mAP). *Mean average precision* merupakan metrik yang populer untuk mengevaluasi model dari detektor. *Mean average precision* merupakan rata rata dari *Average Precision* (AP) pada setiap nilai IoU. Sesuai dengan acuan COCO dataset untuk mengevaluasi model pada *object detection* dengan menghitung mAP dari IoU=0,5 hingga 0,95 dengan pertambahan 0,05.

4.4.2. Evaluasi Pengenalan Gestur Tangan

Evaluasi model CNN untuk pengenalan gestur tangan didapatkan saat *training* dan *testing* menggunakan *k-fold cross validation* dengan $k = 5$, dimana dilakukan validasi sebanyak 5 kali dengan skenario pengujian yang berbeda pada proses *training* dan *testing*. *Training accuracy* dan *testing accuracy* didapatkan pada setiap *fold*. Model dengan akurasi terdinggi akan digunakan sebagai model untuk pengenalan gestur tangan.

4.4.3. Pengujian PSNR

SNR(*Signal to Noise Ratio*) adalah ukuran untuk mengukur kualitas citra terhadap citra yang dilakukan perbaikan. Citra hasil perbaikan dibandingkan dengan citra asli untuk mendapatkan nilai PSNR nya. Nilai PSNR yang tinggi mengindikasikan kualitas citra yang semakin baik karena rasio sinyal terhadap metode juga tinggi, sebaliknya nilai SNR yang rendah berarti kualitas citra yang dihasilkan semakin buruk atau semakin kecil dalam peningkatan kualitas citra. Pengujian PSNR ini dilakukan untuk menentukan nilai dari parameter *retinex* yang akan dipakai pada sistem. Formula PSNR ditunjukkan pada persamaan xx (CV NOTE., 2019).

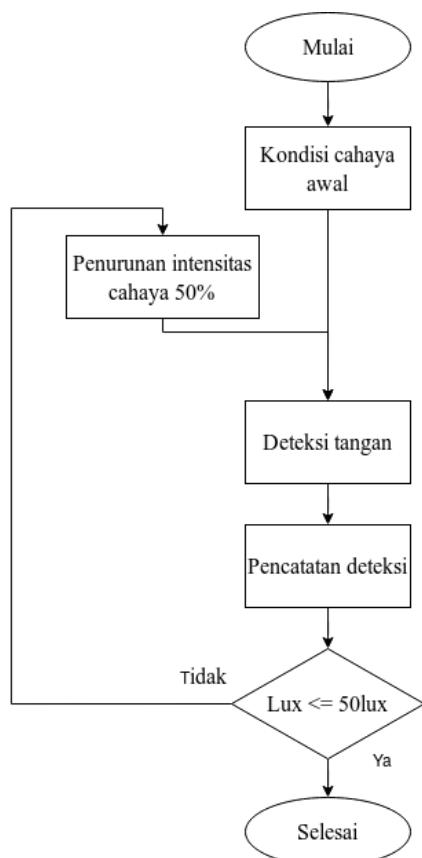
$$PSNR = 20 \log_{10} \left(\frac{MAX_f}{\sqrt{MSE}} \right) \quad (4.1)$$

4.4.4. Pengujian Deteksi Tangan Menggunakan Retinex

Tujuan pengujian deteksi tangan dilakukan untuk menguji sistem dalam melakukan deteksi tangan terkait dengan penurunan intensitas cahaya. Ilustrasi pengujian deteksi tangan digambarkan pada Gambar 4.7. Pengujian dilakukan dengan kondisi cahaya awal sesuai yang terjadi pada ruangan. Pada kondisi awal tersebut akan dilakukan deteksi tangan, setiap hasil dari deteksi tersebut berupa *true positive* atau *false positive*. Hasil deteksi akan dicatat dalam bentuk tabel seperti pada Tabel 4.1.



Gambar 4.9 Ilustrasi Pengujian Deteksi Tangan



Gambar 4.10 Skema Kegiatan Pengujian Deteksi Tangan

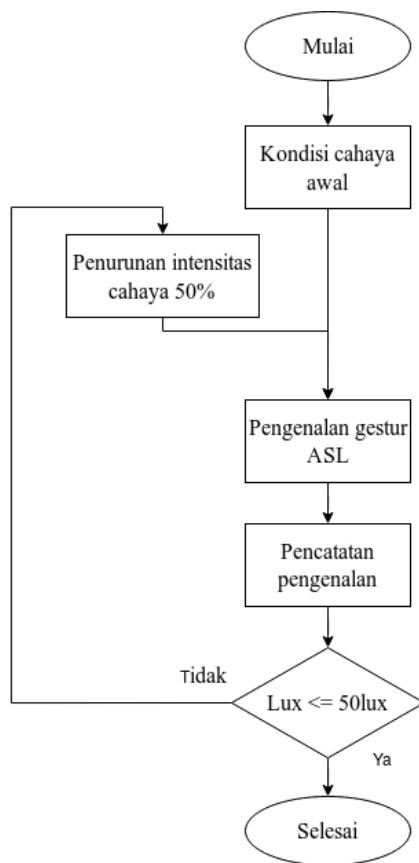
Pada pengujian ini alur kegiatan yang dilakukan saat pengujian deteksi tangan dapat dilihat pada Gambar 4.8. Pengujian deteksi tangan dilakukan secara manual oleh 3 subjek yang berbeda, setiap subjek akan dicatat apakah sistem mampu mendeteksi tangan atau tidak pada intensitas cahaya pada saat itu selama 10 kali, kemudian dilakukan penurunan intensitas sebesar 50% untuk kondisi selanjutnya dan dilakukan hal yang sama. Penurunan ini dilakukan hingga nilai lux yang terukur bernilai kurang dari atau sama dengan 50lux. Evaluasi pengujian dilakukan dengan *confussion matrix* menggunakan data yang dicatat pada Tabel 4.1, sehingga dapat diperoleh nilai akurasi untuk setiap kondisi lux dari hasil pengujian.

Tabel 4.1 Pengujian Deteksi Tangan

Nilai Lux(X)	Hasil Deteksi Subjek_1			Hasil Deteksi Subjek_2			Hasil Deteksi Subjek_3		
X									
X=X*50%									
X=X*50%									
X=X*50%									
...									
X=X ≤ 50Lux									

4.4.5. Pengujian Pengenalan Gestur Tangan Menggunakan Retinex

Pengujian tahap kedua yaitu pengenalan gestur tangan dengan *retinex*. Tujuan dari pengujian ini untuk mengukur performa sistem dalam melakukan pengenalan gestur tangan dengan acuan ASL sesuai dengan dataset yang telah dilatih. Skema kegiatan pengujian pengenalan gestur tangan dapat dilihat pada Gambar 4.9. Pengujian dilakukan dengan kondisi cahaya awal sesuai yang terjadi pada ruangan. Pada kondisi awal tersebut akan dilakukan pengenalan gestur tangan, setiap hasil dari pengenalan tersebut berupa *true positive* atau *false positive*. Nilai *true positive* terjadi apabila *input* gestur tangan dari webcam sesuai dengan klasifikasi gestur tangan ASL yang sebenarnya. Nilai *false positive* terjadi apabila *input* gestur tangan tidak sesuai dengan klasifikasi gestur ASL yang sebenarnya. Hasil pengenalan akan dicatat dalam bentuk tabel seperti pada Tabel 4.1.



Gambar 4.11 Skema Kegiatan Pengujian Pengenalan Gestur Tangan



Gambar 4.12 Ilustrasi Pengujian Pengenalan Gestur Tangan

Ilustrasi pengujian ini digambarkan pada Gambar 4.10. Pengujian pengenalan gestur tangan dilakukan secara manual dan terpisah dari deteksi tangan, tahap ini memiliki perlakuan sama dengan tahap sebelumnya, yaitu menggunakan 3 subjek yang berbeda, kemudian dilakukan pengenalan gestur tangan sebanyak 10 kali

untuk setiap klasifikasi. Total citra yang didapat dalam satu kondisi lux oleh satu subjek adalah 100 citra. Tahap berikutnya dilakukan penurunan intensitas cahaya, dengan mengatur cahaya lampu di ruangan uji sebesar 50% lux menggunakan dimmer. Pengujian dilakukan hingga batas lux kurang dari atau sama dengan 50 lux. Evaluasi yang dilakukan dengan *confussion matrix* menggunakan data yang diperoleh pada Tabel 4.2. Hasil evaluasi pengujian akan diperoleh nilai akurasi untuk setiap kondisi lux.

Tabel 4.2 Pengujian Pengenalan Gestur Tangan

4.4.6. Pengujian Sistem Keseluruhan

Pengujian tahap terakhir adalah pengujian keseluruhan sistem yang dilakukan secara manual, dengan menggabungkan *retinex*, deteksi tangan dan pengenalan gestur tangan dalam satu program utuh. Pengenalan akan terjadi apabila sebuah tangan dideteksi terlebih dahulu, jika tidak terdeteksi maka tidak akan dilakukan pengenalan. Nilai yang didapatkan dari hasil pengujian ini berupa *true positive* dan *false positive*. *True positive* didapatkan ketika sebuah *input* citra gestur dari webcam menghasilkan klasifikasi ASL yang sama dengan gestur ASL yang sebenarnya. Nilai *false positive* terjadi apabila *input* citra gestur dari webcam menghasilkan klasifikasi yang tidak sesuai dengan gestur ASL yang sebenarnya.

Penurunan intensitas dilakukan sama seperti pengujian sebelumnya. Tabel pengujian yang digunakan mengacu pada Tabel 4.2. Evaluasi dari akurasi sistem keseluruhan dapat diperoleh menggunakan *confusion matrix* pada data yang dicatat pada tabel. Nilai akurasi dihitung untuk setiap kondisi lux yang tercatat pada tabel pengujian.

BAB V

IMPLEMENTASI

Tahap implementasi pada penelitian ini menggunakan memiliki beberapa tahap yang terbagi menjadi empat yaitu *data collection*, *data labelling*, *training*, *testing*. Data collection pada penelitian ini adalah proses pengumpulan dataset yang dilakukan dengan melakukan akuisisi citra menggunakan webcam dan pengunduhan dataset *public*. Data yang di dapatkan dari proses akuisisi citra akan dilakukan *preprocessing* terlebih dahulu sebelum dilakukan proses labelling sesuai dengan klasifikasi. Tahap *training* digunakan untuk ekstraksi fitur dataset yang disimpan dalam sebuah model. Model yang terbentuk akan dilakukan proses pengujian pada tahap *testing*.

5.1. Data Collection

Data yang digunakan dalam penelitian ini adalah data yang diambil dari proses akuisisi citra dan dataset *American Sign Language* dari *Massey University*. Dataset yang diambil dari proses akuisisi citra sebanyak 1000 citra tangan yang diambil dari 10 orang. Selanjutnya untuk dataset *American Sign Language* akan dilakukan preprocessing terlebih dahulu sebelum dijadikan format csv yang berisi nama dan lokasi citra.

5.1.1. Implementasi Proses Akuisisi Citra

Tahap akuisisi citra dilakukan bertujuan mendapatkan dataset untuk deteksi objek tangan sehingga dataset yang diambil adalah pose tangan dari subjek. Setiap subjek akan diambil sebanyak 100 citra sehingga menghasilkan 1000 citra. Pada program yang dibuat akan melakukan penyimpanan citra pada folder tertentu dengan setiap nama masing masing subjek. Gambar 5.1 menunjukan kode implementasi saat proses akuisisi citra berlangsung.

```

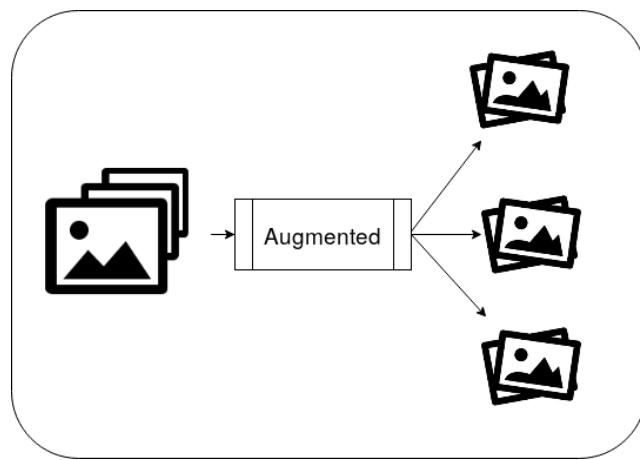
1  if interrupt & 0xFF == 27:
2      break
3  if interrupt & 0xFF == ord('t'):
4      cv2.imwrite(directory+'thomas/'+'thom'+str(count['
tangan_thomas'])+'.jpg',img)
5  if interrupt & 0xFF == ord('m'):
6      cv2.imwrite(directory+'meisy/'+'meis'+str(count['
tangan_meisy'])+'.jpg',img)
7  if interrupt & 0xFF == ord('o'):
8      cv2.imwrite(directory+'tongam/'+'tong'+str(count['
tangan_tongam'])+'.jpg',img)
9  if interrupt & 0xFF == ord('c'):
10     cv2.imwrite(directory+'cbnw/'+'cbn'+str(count['tangan_cbnw
'])+'.jpg',img)
11  if interrupt & 0xFF == ord('a'):
12     cv2.imwrite(directory+'aries/'+'ar'+str(count['
tangan_aries'])+'.jpg',img)
13  if interrupt & 0xFF == ord('h'):
14     cv2.imwrite(directory+'hendrik/'+'hen'+str(count['
tangan_hendrik'])+'.jpg',img)
15  if interrupt & 0xFF == ord('v'):
16     cv2.imwrite(directory+'viany/'+'via'+str(count['
tangan_viany'])+'.jpg',img)
17  if interrupt & 0xFF == ord('e'):
18     cv2.imwrite(directory+'edwin/'+'edw'+str(count['
tangan_edwin'])+'.jpg',img)
19  if interrupt & 0xFF == ord('n'):
20     cv2.imwrite(directory+'novi/'+'nov'+str(count['tangan_novi
'])+'.jpg',img)
21  if interrupt & 0xFF == ord('s'):
22     cv2.imwrite(directory+'silfany/'+'sil'+str(count['
tangan_silfany'])+'.jpg',img)

```

Gambar 5.1 Source code akuisisi citra

5.1.2. Implementasi Data Augmentasi

Proses augmentasi data digunakan untuk menduplikat citra dengan variasi yang berbeda. Pada proses augmentasi ini diterapkan untuk dataset *American Sign Language* karena hanya memiliki 700 citra untuk 10 klasifikasi. Sebelum dilakukan augmentasi, citra pada dataset *American Sign Language* akan dilakukan *resize* pada ukuran (224x224). Augmentasi yang dilakukan berupa variasi intensitas cahaya, intensitas warna, operasi morfologi, *filtering* dan rotasi.



Gambar 5.2 Implementasi Augmentasi

Citra dataset dengan total 700 citra akan diaugmentasi 17 kali dengan beberapa filter dan transformasi yang berbeda-beda untuk setiap citra. Citra hasil augmentasi sejumlah 11900 citra kemudian ditambah dengan 700 citra dataset, sehingga total citra yang akan di training menjadi 12600 untuk 10 klasifikasi.

```

1 def add_light(image, gamma=1.0, nama=name):
2     invGamma = 1.0 / gamma
3     table = np.array([(i / 255.0) ** invGamma) * 255 for i in np.
arange(0, 256)]).astype("uint8")
4     image=cv2.LUT(image, table)
5     if gamma>=1:
6         cv2.imwrite(Folder_name + "/light-"+str(gamma)+nama+
Extension, image)
7     else:
8         cv2.imwrite(Folder_name + "/dark-"+ str(gamma)+nama +
Extension, image)
9 def erosion_image(image,shift,nama=name):
10    kernel = np.ones((shift,shift),np.uint8)
11    image = cv2.erode(image,kernel,iterations = 1)
12    cv2.imwrite(Folder_name + "/Erosion-"+'*'+str(shift)+nama +
Extension, image)
13 def rotate_image(image,deg,nama=name):
14    rows, cols,c = image.shape
15    M = cv2.getRotationMatrix2D((cols/2,rows/2), deg, 1)
16    image = cv2.warpAffine(image, M, (cols, rows))
17    cv2.imwrite(Folder_name + "/Rotate-" + str(deg)+nama +
Extension, image)
18 def hue_image(image,saturation,nama=name):
19    image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
20    v = image[:, :, 2]
21    v = np.where(v <= 255 + saturation, v - saturation, 255)
22    image[:, :, 2] = v
23    image = cv2.cvtColor(image, cv2.COLOR_HSV2BGR)
24    cv2.imwrite(Folder_name + "/hue-" + str(saturation)+nama +
Extension, image)
25 def dilation_image(image,shift,nama=name):
26    kernel = np.ones((shift, shift), np.uint8)
27    image = cv2.dilate(image,kernel,iterations = 1)
28    cv2.imwrite(Folder_name + "/Dilation-"+'*'+ str(shift)+nama +
Extension, image)

```

Gambar 5.3 Source Code Augmentasi Citra

5.1.3. Pembuatan file CSV

Dataset dari pengenalan gestur akan dibuat file CSV yang berisi label dan lokasi dari setiap citra dalam dataset. Fungsi tocsv berikut adalah implementasi dari pembuatan file CSV

```

1 def to_csv(pth, dirname):
2     for i in os.listdir(path):
3         new_path=path+i
4         for img in os.walk(new_path):
5             for n in img[2]:
6                 data=new_path+"/"+n
7                 label = i
8                 my_data.append(data)
9                 my_label.append(label)
10                with open (dirname+'.csv','w') as csvfile:
11                    writer = csv.writer(csvfile)
12                    writer.writerow(['data', 'label'])
13                    for index_data in range(len(my_data)):
14                        print(my_data[index_data])
15                        print(*my_label[index_data])
16                        writer.writerow([my_data[index_data] , my_label[
index_data]])

```

Gambar 5.4 Source Code Pembuatan file CSV

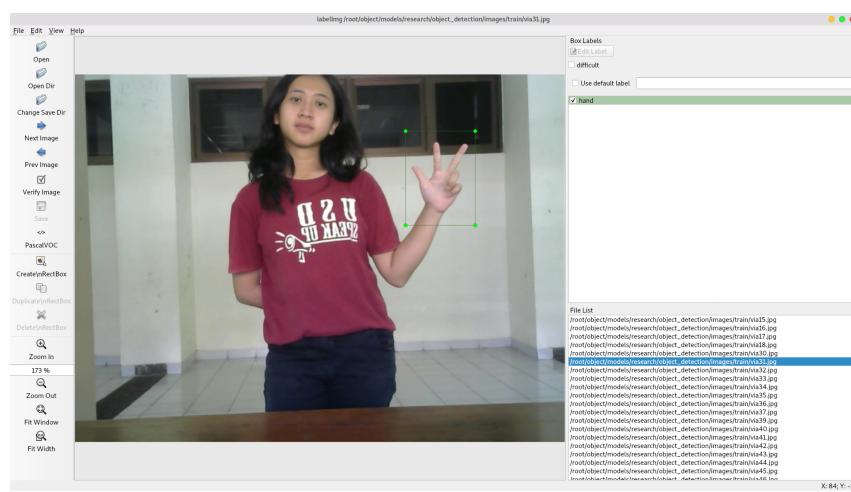
Hasil dari implementasi csv tersebut akan digunakan untuk refrensi input dalam proses pelatihan pengenalan gestur. Berikut adalah hasil dari pembuatan file csv dataset.

1	data	label
2	/home/m448690/ASL4/augmented_data/1/Dilation-*5me7satu.jpg	1
3	/home/m448690/ASL4/augmented_data/1/Erosion-*6e37satu.jpg	1
4	/home/m448690/ASL4/augmented_data/1/dark-0.4hands5_1_dif_seg_2_cropped.jpg	1
5	/home/m448690/ASL4/augmented_data/1/Rotate-25hand1_1_top_seg_4_cropped.jpg	1
6	/home/m448690/ASL4/augmented_data/1/hue-170hands5_1_bot_seg_3_cropped.jpg	1
7	/home/m448690/ASL4/augmented_data/1/hue-50hand1_1_bot_seg_5_cropped.jpg	1
8	/home/m448690/ASL4/augmented_data/1/Rotate-15e19satu.jpg	1
9	/home/m448690/ASL4/augmented_data/1/Erosion-*3hand2_1_bot_seg_1_cropped.jpg	1
10	/home/m448690/ASL4/augmented_data/1/Dilation-*5hand4_1_bot_seg_1_cropped.jpg	1
11	/home/m448690/ASL4/augmented_data/1/me6.jpg	1
12	/home/m448690/ASL4/augmented_data/1/light-2.0e17satu.jpg	1

Gambar 5.5 Hasil File CSV

5.2. Implementasi Data Labelling

Pelabelan sebuah citra dilakukan untuk memberikan pengenalan suatu objek yang ada dalam sebuah citra, pada proses pelabelan menyimpan anotasi yang berisi informasi gambar yang disimpan dalam file XML yang berformat PASCAL VOC. Setiap data dilakukan pelabelan sesuai dengan nama objek yang mau di deteksi dalam hal ini adalah tangan yang diberi label *hand*. Proses pelabelan ditunjukkan pada gambar xxxx.



Gambar 5.6 Proses Pelabelan Citra

Hasil citra terlabel dijadikan refensi dalam informasi XML atau anotasi dari citra tertentu. berikut adalah contoh hasil anotasi dari salah satu citra.

```
<annotation>
  <folder>train</folder>
  <filename>via31.jpg</filename>
  <path>/root/object/models/research/object_detection/images/train/via31.jpg</path>
  <source>
    <database>Unknown</database>
  </source>
  <size>
    <width>640</width>
    <height>480</height>
    <depth>3</depth>
  </size>
  <segmented>0</segmented>
  <object>
    <name>hand</name>
    <pose>Unspecified</pose>
    <truncated>0</truncated>
    <difficult>0</difficult>
    <bndbox>
      <xmin>432</xmin>
      <ymin>74</ymin>
      <xmax>523</xmax>
      <ymax>197</ymax>
    </bndbox>
  </object>
</annotation>
```

Gambar 5.7 hasil anotasi citra

5.3. Implementasi Pembuatan TFRecord file

Proses selanjutnya adalah mengubah informasi XML ke CSV dengan tujuan membuat format *Tensorflow* dataset atau *TFRecord*. Berikut program konversi file dataset.

```

1 def xml_to_csv(path):
2     xml_list = []
3     for xml_file in glob.glob(path + '/*.xml'):
4         tree = ET.parse(xml_file)
5         root = tree.getroot()
6         for member in root.findall('object'):
7             value = (root.find('filename').text,
8                     int(root.find('size')[0].text),
9                     int(root.find('size')[1].text),
10                    member[0].text,
11                    int(member[4][0].text),
12                    int(member[4][1].text),
13                    int(member[4][2].text),
14                    int(member[4][3].text)
15                   )
16             xml_list.append(value)
17     column_name = ['filename', 'width', 'height', 'class', 'xmin',
18                    'ymin', 'xmax', 'ymax']
19     xml_df = pd.DataFrame(xml_list, columns=column_name)
20     return xml_df
21
22 def main():
23     for folder in ['train','test']:
24         image_path = os.path.join(os.getcwd(), ('images/' + folder
25         ))
26         xml_df = xml_to_csv(image_path)
27         xml_df.to_csv(('images/' + folder + '_labels.csv'), index=
None)
28         print('Successfully converted xml to csv.')

```

Gambar 5.8 Source Code Pengubahan XML ke CSV

```

1 def class_text_to_int(row_label):
2     if row_label == 'hand':
3         return 1
4     else:
5         None
6 def split(df, group):
7     data = namedtuple('data', ['filename', 'object'])
8     gb = df.groupby(group)
9     return [data(filename, gb.get_group(x)) for filename, x in zip
10        (gb.groups.keys(), gb.groups)]
11 def create_tf_example(group, path):
12     with tf.gfile.GFile(os.path.join(path, '{}'.format(group.
13         filename)), 'rb') as fid:
14         encoded_jpg = fid.read()
15         encoded_jpg_io = io.BytesIO(encoded_jpg)
16         image = Image.open(encoded_jpg_io)
17         width, height = image.size
18         filename = group.filename.encode('utf8')
19         image_format = b'jpg'
20         xmins = []
21         xmaxs = []
22         ymins = []
23         ymaxs = []
24         classes_text = []
25         classes = []
26         for index, row in group.object.iterrows():
27             xmins.append(row['xmin'] / width)
28             xmaxs.append(row['xmax'] / width)
29             ymins.append(row['ymin'] / height)
30             ymaxs.append(row['ymax'] / height)
31             classes_text.append(row['class'].encode('utf8'))
32             classes.append(class_text_to_int(row['class']))

```

```

31     tf_example = tf.train.Example(features=tf.train.Features(
32         feature={
33             'image/height': dataset_util.int64_feature(height),
34             'image/width': dataset_util.int64_feature(width),
35             'image/filename': dataset_util.bytes_feature(filename),
36             'image/source_id': dataset_util.bytes_feature(filename),
37             'image/encoded': dataset_util.bytes_feature(encoded_jpg),
38             'image/format': dataset_util.bytes_feature(image_format),
39             'image/object/bbox/xmin': dataset_util.float_list_feature(
40                 xmins),
41                 'image/object/bbox/xmax': dataset_util.float_list_feature(
42                 xmaxs),
43                 'image/object/bbox/ymin': dataset_util.float_list_feature(
44                 ymins),
45                 'image/object/bbox/ymax': dataset_util.float_list_feature(
46                 ymaxs),
47                 'image/object/class/text': dataset_util.bytes_list_feature(
48                 classes_text),
49                 'image/object/class/label': dataset_util.
50                     int64_list_feature(classes),
51             })))
52
53     return tf_example
54
55 def main(_):
56     writer = tf.python_io.TFRecordWriter(FLAGS.output_path)
57     path = os.path.join(os.getcwd(), FLAGS.image_dir)
58     examples = pd.read_csv(FLAGS.csv_input)
59     grouped = split(examples, 'filename')
60
61     for group in grouped:
62         tf_example = create_tf_example(group, path)
63         writer.write(tf_example.SerializeToString())
64
65     writer.close()
66
67     output_path = os.path.join(os.getcwd(), FLAGS.output_path)
68     print('Successfully created the TFRecords: {}'.format(
69         output_path))
70
71 if __name__ == '__main__':
72     tf.app.run()

```

Gambar 5.9 Source Code Pembentukan TFRecord

5.4. Konfigurasi Pipeline Object Detection

Proses pelatihan Tensorflow Object Detection API menggunakan konfigurasi pipeline, beberapa konfigurasi tersebut diantaranya model, training_config, eval_config, train_input_config, eval_input_config. Dataset yang berformat *TFRecord* akan di *lewatkan sebagai input* dan model pada *ssd mobilenet v2* akan digunakan sebagai *pretrained* model. Berikut adalah konfigurasi yang digunakan

```

1 train_input_reader: {
2     label_map_path: "/home/m448690/object/models/research/
3         object_detection/training/label_map.pbtxt"
4     tf_record_input_reader {
5         input_path: "/home/m448690/object/models/research/
6             object_detection/train.record"
7     }
8 }
9 eval_config: {
10     metrics_set: "coco_detection_metrics"
11     use_moving_averages: false
12     batch_size:12
13 }
14 eval_input_reader: {
15     label_map_path: "/home/m448690/object/models/research/
16         object_detection/training/label_map.pbtxt"
17     shuffle: false
18     num_epochs: 3
19     tf_record_input_reader {
20         input_path: "/home/m448690/object/models/research/
21             object_detection/test.record"
22     }
23 }
```

```

20 train_config: {
21   fine_tune_checkpoint_version: V2
22   fine_tune_checkpoint: "/home/m448690/object/models/research/
23     object_detection/ssd_mobilenet_v2_fpnlite_640x640_coco17_tpu-8/
24     checkpoint/ckpt-0"
25   fine_tune_checkpoint_type: "detection"
26   batch_size: 12
27   sync_replicas: true
28   startup_delay_steps: 0
29   replicas_to_aggregate: 8
30   num_steps: 30000
31   data_augmentation_options {
32     random_horizontal_flip {
33     }
34     data_augmentation_options {
35       random_crop_image {
36     }
37     optimizer {
38       momentum_optimizer: {
39         learning_rate: {
40           cosine_decay_learning_rate {
41             learning_rate_base: .08
42             total_steps: 30000
43             warmup_learning_rate: .026666
44             warmup_steps: 1000
45           }
46         }
47         momentum_optimizer_value: 0.9
48       }
49       use_moving_average: false
50     }
51     max_number_of_boxes: 100
52     unpad_groundtruth_tensors: false
53   }

```

Gambar 5.10 konfigurasi pipeline *Object Detection*

5.5. Implementasi Training

5.5.1. Object Detection

Tahap pelatihan dalam *object detection* menggunakan arsitektur mobilenet v2, proses pelatihan berlangsung dengan menggunakan parameter pada konfigurasi pipeline. berikut adalah program untuk melakukan pelatihan.

```

1 def main(unused_argv):
2     flags.mark_flag_as_required('model_dir')
3     flags.mark_flag_as_required('pipeline_config_path')
4     config = tf.estimator.RunConfig(model_dir=FLAGS.model_dir)
5     train_and_eval_dict = model_lib.create_estimator_and_inputs(
6         run_config=config,
7         pipeline_config_path=FLAGS.pipeline_config_path,
8         train_steps=FLAGS.num_train_steps,
9         sample_1_of_n_eval_examples=FLAGS.
10            sample_1_of_n_eval_examples,
11            sample_1_of_n_eval_on_train_examples=(
12                FLAGS.sample_1_of_n_eval_on_train_examples))
13     estimator = train_and_eval_dict['estimator']
14     train_input_fn = train_and_eval_dict['train_input_fn']
15     eval_input_fns = train_and_eval_dict['eval_input_fns']
16     eval_on_train_input_fn = train_and_eval_dict['
17         eval_on_train_input_fn']
18     predict_input_fn = train_and_eval_dict['predict_input_fn']
19     train_steps = train_and_eval_dict['train_steps']
20     if FLAGS.checkpoint_dir:
21         if FLAGS.eval_training_data:
22             name = 'training_data'
23             input_fn = eval_on_train_input_fn
24         else:
25             name = 'validation_data'
26             input_fn = eval_input_fns[0]
```

Gambar 5.11 Source Code Pelatihan *Object Detection*

```

25   if FLAGS.run_once:
26     estimator.evaluate(input_fn, steps=None, checkpoint_path=tf.
27       train.latest_checkpoint(FLAGS.checkpoint_dir))
28   else:
29     model_lib.continuous_eval(estimator, FLAGS.checkpoint_dir,
30       input_fn,
31       train_steps, name, FLAGS.
32       max_eval_retries)
33   else:
34     train_spec, eval_specs = model_lib.create_train_and_eval_specs
35     (
36       train_input_fn,
37       eval_input_fns,
38       eval_on_train_input_fn,
39       predict_input_fn,
40       train_steps,
41       eval_on_train_data=False)
42     tf.estimator.train_and_evaluate(estimator, train_spec,
43       eval_specs[0])

```

Gambar 5.12 Source Code Pelatihan *Object Detection*

5.5.2. Pengenalan Gestur

Pada proses pelatihan gestur menggunakan *k-fold cross validation* dengan $k=5$. Arsitektur yang digunakan adalah Mobilenet V2, pada pelatihan ini *pretrained* model akan di *load* menjadi base model dan jumlah *classifier* diganti sesuai dengan objek yang akan di klasifikasikan dan menambahkan *hidden layer*. Proses tersebut ditunjukan pada program berikut.

```

1 base_model_path = "/home/448690/ASL4/mobilenetv2.h5"
2 train_path='/home/m448690/ASL4/augmented_data/'
3 in_img=(224,224,3)
4 colname=["data", "label"]
5 train_data = pd.read_csv('train_augmented.csv', dtype=str)
6 Y = train_data['label']
7 X = train_data['data']
8 kf = KFold(n_splits = 5,shuffle=True)
9 for train_index, val_index in kf.split(X):
10     training_data = train_data.iloc[train_index]
11     validation_data = train_data.iloc[val_index]
12     base_model = tf.keras.applications.MobileNetV2(input_shape=
13         in_img ,include_top=False, weights="mobilenetv2.h5")
14     for layer in (base_model.layers):
15         layer.trainable=True
16     x=base_model.output
17     x=GlobalAveragePooling2D()(x)
18     x=Flatten()(x)
19     x=Dense(1024,activation='relu')(x)
20     x=Dense(812,activation='relu')(x)
21     preds=Dense(10,activation='softmax')(x)
22     model=Model(inputs=base_model.input,outputs=preds)
model.summary()

```

Gambar 5.13 Source Code Pelatihan *Pengenalan Gestur*

5.6. Implementasi Retinex

Proses retinex yang digunakan adalah Multiscale Retinex dengan nilai sigma 15, 80, 250. Multiscale Retinex merupakan hasil penggabungan tiga Single-scale Retinex. Implementasi program Retinex dapat dilihat pada gambar 5.14. Baris 1 merupakan fungsi Singlescale Retinex, kemudian baris 4 fungsi dari Multiscale Retinex yang didalamnya akan memanggil fungsi baris nomor 1 sebanyak tiga kali sesuai dengan input sigma yang diberikan.

```

1 def singleScaleRetinex(img, sigma):
2     retinex = np.log10(img) - np.log10(cv2.GaussianBlur(img, (0,
0), sigma))
3     return retinex
4 def MSR(img, sigma_list):
5     try:
6         img = np.float64(img) + 1.0
7         retinex = np.zeros_like(img)
8         for sigma in sigma_list:
9             retinex += singleScaleRetinex(img, sigma)
10        retinex = retinex / len(sigma_list)
11        for i in range(retinex.shape[2]):
12            retinex[:, :, i] = (retinex[:, :, i] - np.min(retinex[:, :, i])) / (np.max(retinex[:, :, i]) - np.min(retinex[:, :, i])) * 255
13        retinex = np.uint8(np.minimum(np.maximum(retinex, 0), 255))
14    )
15    return retinex
16 except:
17     print("exit")
18     return retinex

```

Gambar 5.14 Source Code Retinex

5.7. Evaluasi Model

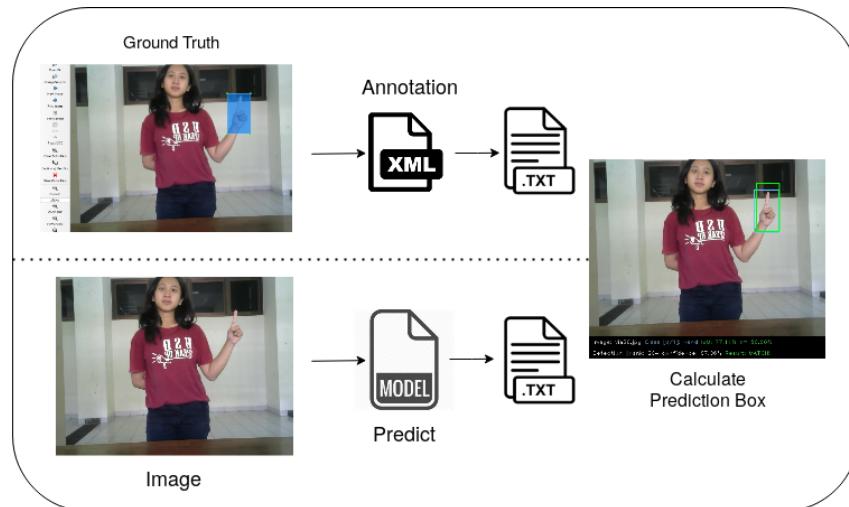
5.7.1. PSNR

Pengujian PSNR dilakukan untuk mengevaluasi perbaikan citra yang dikenai Retinex dengan parameter yang digunakan. Proses evaluasi ini membutuhkan 2 input citra, yaitu citra asli dan citra yang dikenai Retinex. Implementasi PSNR ditunjukkan pada gambar xx.

5.7.2. mAP

Model dari pelatihan object detection dilakukan evaluasi menggunakan mAP yang mengacu pada metrik evaluasi coco dengan mAP@IOU[0.5:0.05:0.95]. Per-

hitungan nilai AP(*Average Precision*) berdasarkan pada hasil interseksi antara box ground truth dan box prediksi. Visualisasi evaluasi model deteksi ditunjukkan pada gambar 5.11.



Gambar 5.15 Visualisasi IOU pada Citra

Proses pengujian mAP harus mengekstrak informasi koordinat anotasi pada dataset menjadi file txt sebagai *ground truth*. Proses yang sama dilakukan untuk citra testing, namun nilai koordinat diperoleh dari hasil prediksi model. Hasil dari prediksi akan disimpan koordinatnya dalam file txt sebagai *prediction box*. Implementasi mAP ditunjukkan pada gambar xx.

5.7.3. Kfold Cross Validation

Model CNN yang dilatih menggunakan Kfold Cross Validation dengan $k = 5$, dari hasil seluruh fold dipilih model dengan nilai akurasi terbesar. Berdasarkan tabel 5.1 maka model yang akan digunakan untuk pengujian adalah model dengan dengan nilai akurasi 0.9531863 pada fold empat. Hasil dari proses pelatihan ditunjukan pada Tabel 5.1.

Tabel 5.1 Akurasi 5-fold Cross Validation

K=1	K=2	K=3	K=4	K=5
0.9490	0.9399	0.9436	0.9531	0.9375

5.8. Implementasi Pengujian

Implementasi pengujian pada rancangan penelitian ini meliputi pengenalan gestur tangan dan deteksi objek yang masing masing menggunakan retinex. Setelah dilakukan pengujian, peneliti melakukan pengujian yang sama namun tanpa menggunakan retinex, kemudian membandingkan hasil keduanya untuk dianalisa.

5.8.1. Pengujian Pengenalan Gestur Tangan

Pengujian pengenalan gestur tangan dilakukan seperti rancangan sistem, pada tahap ini video akan di tulis ulang untuk dianalisis ketika menggunakan Retinex dan saat sistem berjalan tanpa Retinex. Keluaran dari pengenalan gestur ini menampilkan angka yang akan dikenali oleh sistem. Implementasi program disajikan pada gambar 5.16.

```

1 fourcc = cv2.VideoWriter_fourcc(*'mp4v')
2 out = cv2.VideoWriter('dela/no_retinex/ALL/ALL_Dela25lux.mp4',
   fourcc, 25.0, (640, 480))
3 json_file = open("mobile_4.json", "r")
4 model_json = json_file.read()
5 json_file.close()
6 loaded_model = model_from_json(model_json)
7 loaded_model.load_weights("mobile_4.h5")
8 cap = cv2.VideoCapture("/root/pengujian/video_pengujian/
   ASL_Dela25lux.mp4")
9 categories = {0: 'ZERO', 1: 'ONE', 2: 'TWO', 3: 'THREE', 4: 'FOUR',
   5: 'FIVE'}
10 labels_dict = {'0':0,'1':1,'2':2,'3':3,'4':4,'5':5,'6':6,'7':7,'8':
   8,'9':9}
11 global mulai
12 start= True
13 while start:
```

```

14     try:
15         _, fr = cap.read()
16         frame = rescale_frame(fr, percent=100)
17         x1 = int(0.5*frame.shape[1])+80
18         y1 = 10
19         x2 = frame.shape[1]-10
20         y2 = int(0.5*frame.shape[1])-80
21         cv2.rectangle(frame, (x1-1, y1-1), (x2+1, y2+1), (255,0,0)
22 ,1)
23         roi = frame[y1:y2, x1:x2]
24         roi = cv2.resize(roi, (224, 224))
25         test_image = (roi)
26         result = loaded_model.predict(test_image.reshape(1, 224,
27 224, 3))
28         res = str(np.argmax(result))
29         cv2.putText(frame,res ,(20, 120), cv2.FONT_HERSHEY_PLAIN,
30 3, (20,10,255) , 2)
31         out.write(frame)
32         interrupt = cv2.waitKey(10)
33         if interrupt & 0xFF == 27:
34             break
35     except:
36         selesai=time.time()
37         print("waktu selesai",selesai)
38         print("done")
39         waktukomp= selesai - mulai
40         print(" waktu komputasi ",waktukomp)
41         out.release()
42         cap.release()
43         cv2.destroyAllWindows()
44         start=False

```

Gambar 5.16 Source Code Pengenalan Gestur

5.8.2. Pengujian Deteksi Objek

Proses pengujian deteksi objek dilakukan hal yang sama seperti pengujian gestur. Keluaran dari deteksi objek ini menampilkan *bounding box* dari objek yang terdeteksi. Gambar 5.17 merupakan implementasi program deteksi objek. Line 6 pada program dilakukan untuk melakukan *load* label untuk objek, dimana `label_map.pbtxt` berisi *hand* dengan id sama dengan 1. Line 8 digunakan untuk memuat model hasil pelatihan deteksi objek yang disimpan dengan nama `saved_model.pb`.

```

1 fourcc = cv2.VideoWriter_fourcc(*'mp4v')
2 out = cv2.VideoWriter('saya/retinex/OD/od133luxlux.mp4',fourcc
   ,25.0,(640,480))
3 def load_model(model_path):
4     return tf.saved_model.load(model_path)
5 cap = cv2.VideoCapture("/root/pengujian/video_pengujian/
   odsaya_133lux.mp4")
6 PATH_TO_LABELS = '/root/object/models/research/object_detection/
   training/label_map.pbtxt'
7 category_index = label_map_util.
   create_category_index_from_labelmap(PATH_TO_LABELS,
   use_display_name=True)
8 model_name = '/root/object/models/research/object_detection/
   inference_graph/saved_model'
9 detection_model = load_model(model_name)
10 print(detection_model.signatures['serving_default'].inputs)
```

```

11 while start:
12     try:
13         r, image_np = cap.read()
14         ret=retinex.MSR(image_np ,[15, 80 ,250])
15         output_dict = run_inference_for_single_image(
16             detection_model, ret)
17         vis_util.visualize_boxes_and_labels_on_image_array(
18             ret,
19             output_dict['detection_boxes'],
20             output_dict['detection_classes'],
21             output_dict['detection_scores'],
22             category_index,
23             instance_masks=output_dict.get('
24             detection_masks_reframed', None),
25             use_normalized_coordinates=True,
26             max_boxes_to_draw=1,
27             min_score_thresh=.3,
28             agnostic_mode=False,
29             line_thickness=5)
30         for index, score in enumerate(output_dict['
31             detection_scores']):
32             if score < 0.3:
33                 continue
34             else:
35                 label = category_index[output_dict['
36                 detection_classes'][index]]['name']
37                 if(label):
38                     counting+=1
39                     if(counting%20==0):
40                         value=label+"detected"
41                         tmp=1
42                         if(tmp>=1 and tmp<=5):
43                             highlight(ret,value)
44                             tmp+=1
45                         if tmp==5:
46                             tmp=counting=0
47                             out.write(ret)

```

Gambar 5.17 Source Code Deteksi Objek

5.8.3. Pengujian Sistem Secara Keseluruhan

Pengujian ketiga merupakan gabungan antara deteksi tangan dan pengenalan gestur. Keluaran dari program ini akan menampilkan objek yang terdeteksi kemudian objek tersebut akan dikenali gesturnya. Pengujian ini dilakukan sama seperti sebelumnya. Implementasi program ditampilkan pada gambar 5.18

```

1 fourcc = cv2.VideoWriter_fourcc(*'mp4v')
2 out = cv2.VideoWriter('dela/with_retinex/ALL/ALL_dela15lux.mp4',
3                         fourcc, 25.0, (640, 480))
4 categories = {0: 'ZERO', 1: 'ONE', 2: 'TWO', 3: 'THREE', 4: 'FOUR',
5                 , 5: 'FIVE'}
6 labels_dict = {'0':0,'1':1,'2':2,'3':3,'4':4,'5':5,'6':6,'7':7,'8':
7 :8,'9':9}
8 json_file = open("mobile_4.json", "r")
9 model_json = json_file.read()
10 json_file.close()
11 loaded_model = model_from_json(model_json)
12 loaded_model.load_weights("mobile_4.h5")
13 print("Loaded model from disk")
14 def load_model(model_path):
15     model = tf.saved_model.load(model_path)
16     return model
17 cap = cv2.VideoCapture("/root/pengujian/video_pengujian/
18 ASL_Dela15lux.mp4")
19 PATH_TO_LABELS = '/root/object/models/research/object_detection/
20 training/label_map.pbtxt'
21 category_index = label_map_util.
22     create_category_index_from_labelmap(PATH_TO_LABELS,
23     use_display_name=True)
24 model_name = '/root/object/models/research/object_detection/
25 inference_graph/saved_model'
26 detection_model = load_model(model_name)
```

```

62 def run_inference(model, cap):
63     start= True
64     while start:
65         try:
66             ret, image_np = cap.read()
67             ret = image_np
68             ret=retinex.MSR(image_np ,[10, 80 ,250])
69             output_dict = run_inference_for_single_image(model,
70             ret)
71             vis_util.visualize_boxes_and_labels_on_image_array(
72                 ret,
73                 output_dict['detection_boxes'],
74                 output_dict['detection_classes'],
75                 output_dict['detection_scores'],
76                 category_index,
77                 instance_masks=output_dict.get('
78                 detection_masks_reframed', None),
79                 use_normalized_coordinates=True,
80                 max_boxes_to_draw=1,
81                 min_score_thresh=.2,
82                 agnostic_mode=False,
83                 line_thickness=5)
84             for index, score in enumerate(output_dict['
85                 detection_scores']):
86                 if score < 0.2:
87                     label=""
88                     continue
89                 else:
90                     label = category_index[output_dict['
91                 detection_classes'][index]]['name']
92                     ymin, xmin, ymax, xmax = output_dict['
93                 detection_boxes'][index]
94                     koordinat.append((label, int(xmin *
95                     image_width), int(ymin * image_height),int(xmax * image_width),
96                     int(ymax * image_height)))

```

```
90         ymin = (int(ymin * image_height))
91         xmin = (int(xmin * image_width))
92         xmax = (int(xmax * image_width))
93         ymax = (int(ymax * image_height))
94         roi = ret[ymin:ymax, xmin:xmax].copy()
95         new = cv2.resize(roi, (224, 224))
96         result = loaded_model.predict(new.reshape(1,
97                                         224, 224, 3))
98
99         res = 'Number Prediction : ' + str(np.argmax(
100            result))
101
102         cv2.putText(ret, res, (21, 300), cv2.
103 FONT_HERSHEY_PLAIN, 2, (20, 19, 255), 2)
104
105         out.write(ret)
106
107         if cv2.waitKey(25) & 0xFF == ord('q'):
108             cap.release()
109             cv2.destroyAllWindows()
110             break
```

Gambar 5.18 Source Code Sistem Keseluruhan

BAB VI

HASIL DAN PEMBAHASAN

Pada bab ini akan dibahas hasil evaluasi terhadap model dan pengujian sistem. Evaluasi terkait model diantaranya evaluasi deteksi tangan, evaluasi pengenalan gestur tangan dan pengujian SNR. Pengujian terhadap subjek diantaranya pengujian deteksi tangan menggunakan retinex, pengujian pengenalan destur tangan menggunakan Retinex serta pengujinan sistem secara keseluruhan.

Evaluasi terkait model digunakan untuk menilai suatu model yang akan digunakan dalam pengujian terhadap subjek. Model deteksi tangan akan dievaluasi menggunakan mAP, sedangkan model pengenalan gestur akan dievaluasi dengan *k-fold cross-validation*, kemudian untuk mengevaluasi Retinex menggunakan perbandingan SNR.

Pengujian sistem dibagi menjadi 3 yaitu pengujian deteksi tangan, pengujian pengenalan gestur tangan dan pengujian sistem keseluruhan. Setiap pengujian tersebut akan dibandingkan ketika menggunakan retinex dan tanpa menggunakan Retinex. Pengujian dilakukan oleh 3 subjek yang berbeda di luar dari dataset.

6.1. Evaluasi Deteksi Tangan

Pengujian mAP dari dilakukan dengan skema pada gambar 6.1. Hasil annotasi setiap citra dataset maupun citra prediksi akan diekstrak menjadi format file txt dimana berisi nilai koordinat dari citra. Citra yang akan dilakukan prediksi adalah citra dataset testing yang berjumlah 200 citra. Nilai mAP merupakan hasil rata rata dari setiap *Average Precicion*, dimana variasi IOU 0,5 hingga 0,95 dengan 0,05 setiap stepnya. Hasil pengujian mAP ditunjukan pada tabel 6.1 dengan nilai mAP sebesar 50,62. Hasil dari pelatihan deteksi objek ini akan digunakan sebagai model pada sistem. Penelitian yang dilakukan dengan dataset COCO memperoleh hasil mAP sebesar 22,1.

Tabel 6.1 Evaluasi mAP Model Object Detection

Mean Average Precicion(mAP) = 50,62									
@0,5	@0,55	@0,60	@0,65	@0,70	@0,75	@0,80	@0,85	@0,90	@0,95
76,58	76,58	74,72	73,34	63,85	46,3	29,12	19,76	3,64	0,02

6.2. Evaluasi Pengenalan Gestur Tangan

6.3. Pengujian SNR

Pengujian SNR dilakukan dengan membandingkan 2 algoritma Retinex dengan citra asli. Citra yang akan digunakan pada pengujian SNR diambil dari 3 kondisi citra yang mewakili intensitas yang digunakan. Nilai parameter Retinex yang akan digunakan merujuk dalam pengujian ini merujuk pada penelitian yang dilakukan oleh (Petro, 2014) dan 2 variasi nilai parameter yang ditentukan oleh peneliti. Nilai yang digunakan untuk mengisi parameter Retinex ditunjukkan pada tabel 6.2.

Tabel 6.2 Parameter Nilai Retinex

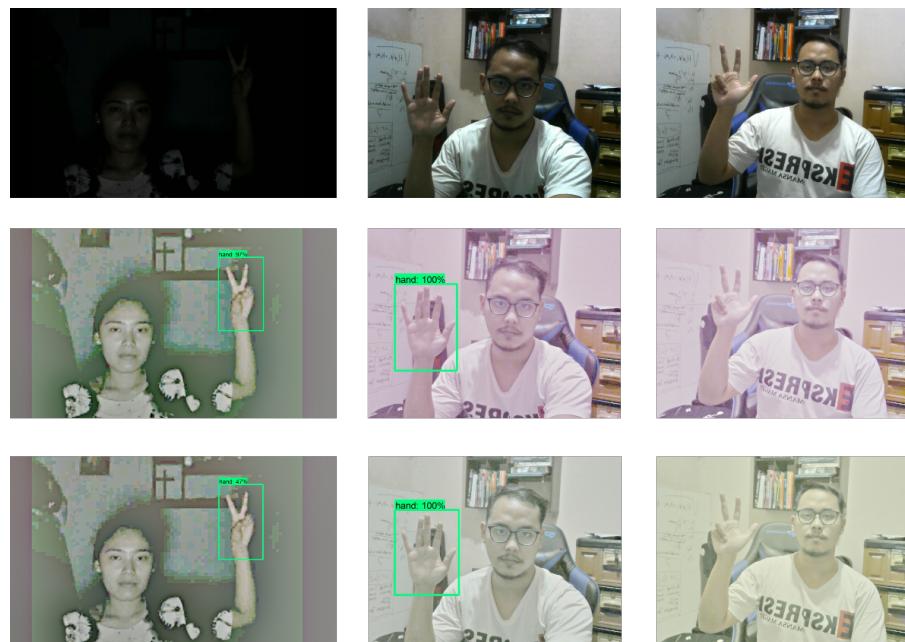
Parameter	σ_1	σ_2	σ_3	α	β
Petro, 2014	15	80	250	125	46
Variasi 1	10	60	180	100	30
Variasi 2	30	100	210	140	60

Citra yang dilakukan operasi Retinex akan dilanjutkan dengan operasi deteksi tangan, kemudian dilakukan perbandingan dengan waktu komputasi, hasil deteksi dan nilai PSNR untuk menentukan Retinex yang akan dipakai.

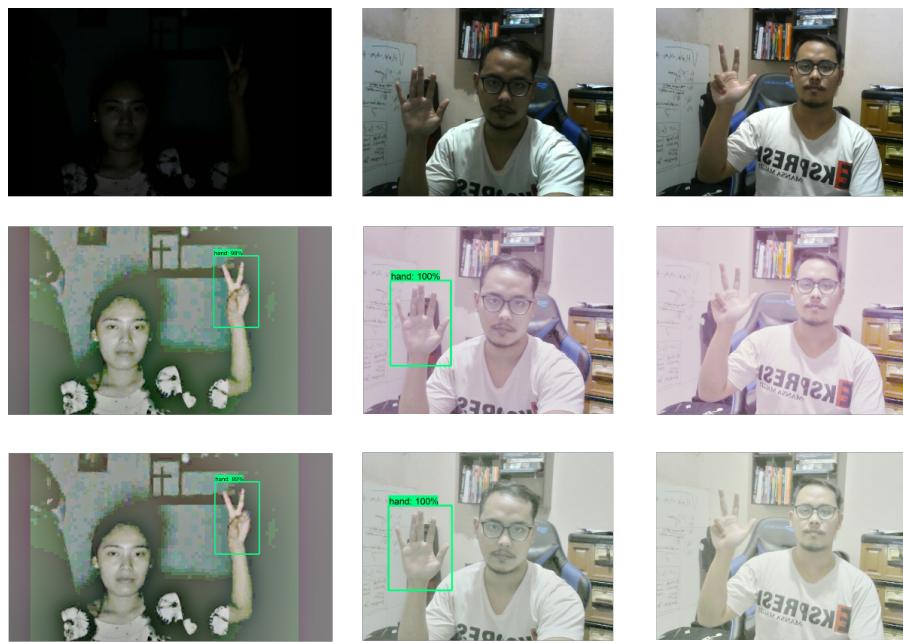
Hasil perbandingan citra Retinex ditunjukkan pada gambar 6.1, 6.2, dan 6.3. Secara visual citra asli setelah dilakukan operasi Retinex terjadi peningkatan terhadap kecerahan, sehingga detail dari citra lebih dapat terlihat daripada citra asli. Citra asli terdapat pada baris pertama, kemudian citra *Multiscale Retinex* pada baris kedua dan citra *Multiscale Retinex Color Restoration* pada baris ketiga. Seluruh citra yang dikenai Retinex telah dilakukan proses deteksi tangan.



Gambar 6.1 Perbandingan Hasil Citra Retinex Parameter Petro, 2014.



Gambar 6.2 Perbandingan Hasil Citra Retinex Parameter Variasi 1.



Gambar 6.3 Perbandingan Hasil Citra Retinex Parameter Variasi 2.

Peneliti meninjau kedua algoritma Retinex terkait dengan waktu komputasi dan akurasi dalam melakukan deteksi tangan dan nilai PSNR. Tujuan dari peninjauan ini dilakukan untuk memilih algoritma Retinex yang efektif. Hasil Tabel 6.3 merupakan pengujian PSNR, dari data tersebut untuk citra1 dimana kondisi citra sangat gelap MSR lebih unggul akan tetapi nilai yang dihasilkan tidak terlalu signifikan antara kedua metode. Citra2 dan citra 3 memiliki selisih 0.94 dan 0.575, dimana MSRCR lebih unggul pada kondisi tersebut untuk parameter Petro. Kondisi cahaya yang semakin terang meningkatkan nilai PSNR pada MSRCR untuk ketiga variasi parameter.

Tabel 6.3 Hasil Pengujian PSNR

Petro, 2014			Variasi 1		Variasi 2	
	MSR	MSRCR	MSR	MSRCR	MSR	MSRCR
Citra1	6,692	6,688	6,163	6,11	6,657	6,19
Citra2	8,23	8,805	8,497	9,367	8,75	9,41
Citra3	9,472	10,412	9,7	10,79	9,69	10,462

Tabel 6.4 Perbandingan Nilai Deteksi

Petro, 2014			Variasi 1		Variasi 2	
	MSR	MSRCR	MSR	MSRCR	MSR	MSRCR
Citra1	98%	88%	97%	47%	98%	89%
Citra2	100%	100%	100%	100%	100%	100%
Citra3	32%	0%	0%	0%	0%	0%

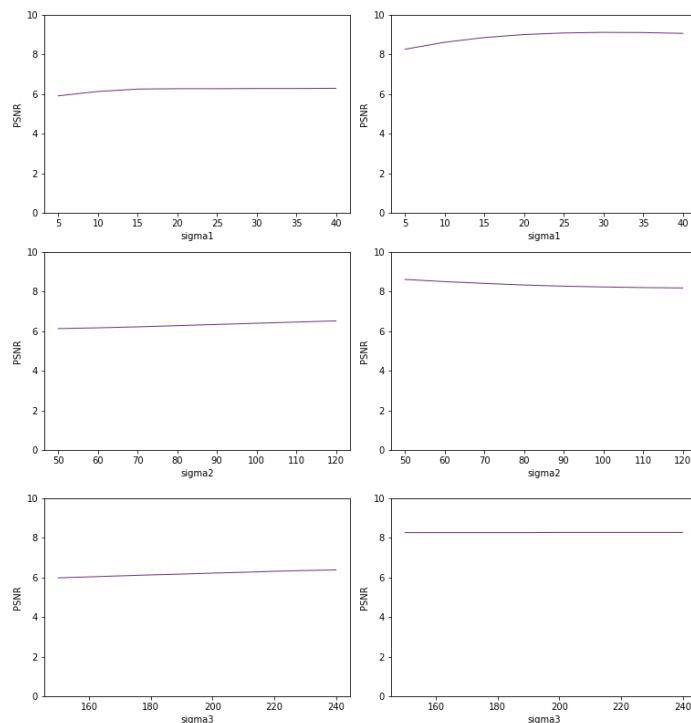
Tabel 6.4 merupakan nilai *confidence* atau nilai deteksi dari hasil deteksi ketiga citra. Hasil tersebut menunjukkan bahwa citra yang dikenai MSR memiliki nilai yang lebih tinggi untuk ketiga parameter. Kondisi citra pertama dan kedua dapat dikenali dengan baik oleh semua parameter, namun MSR memiliki nilai yang lebih besar. Kondisi citra ketiga tidak dapat di deteksi oleh MSRCR pada semua variasi parameter, namun MSR dapat mendeteksi dengan nilai 32% pada parameter Petro. Tabel 6.5 menunjukkan waktu komputasi dari suatu Retinex dalam memproses satu citra. Waktu komputasi MSR lebih cepat dibandingkan dengan MSRCR pada citra kondisi gelap, namun semakin terang intensitas citra nilai MSRCR semakin lebih cepat dibandingkan MSR. Selisih waktu komputasi citra tidak terlalu jauh pada setiap variasi parameter yang diujikan. Selisih tertinggi pada parameter variasi 1 citra gelap yang memiliki selisih 1,28 detik.

Tabel 6.5 Waktu Komputasi Retinex

Petro, 2014			Variasi 1		Variasi 2	
	MSR	MSRCR	MSR	MSRCR	MSR	MSRCR
Citra1	10,257s	11,121s	13,2s	14,48s	17s	17,05s
Citra2	7,172s	7,234s	6,41s	6,14s	7,75s	7,357s
Citra3	6,795s	7,367s	6,2s	6,46s	7,366	7,37s

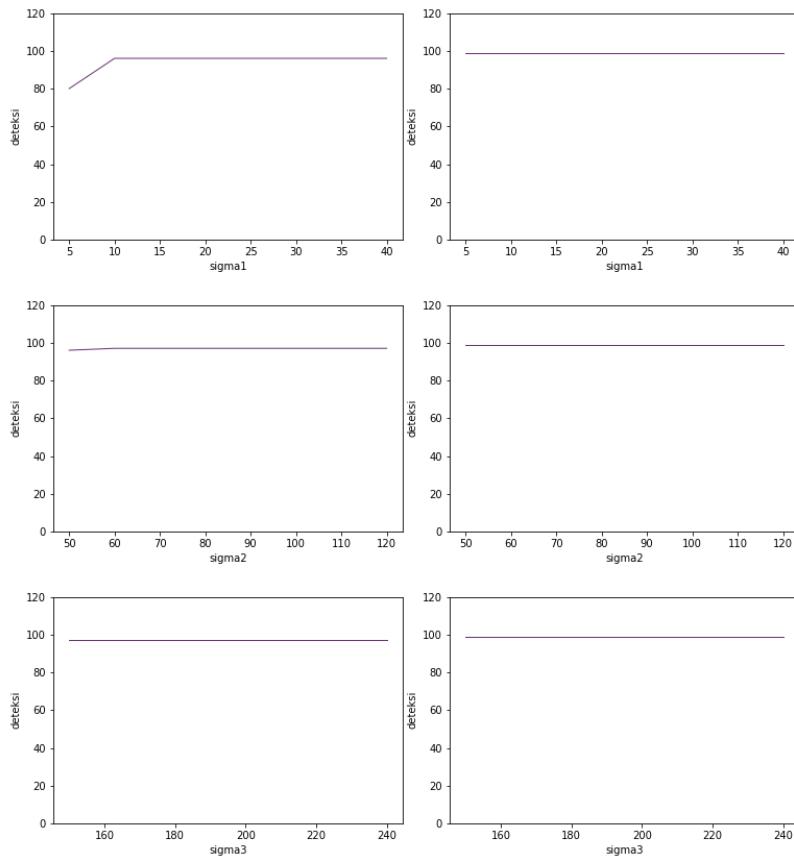
Berdasarkan hasil ketiga tabel perbandingan tersebut dengan mempertimbangkan antara PSNR, waktu komputasi dan nilai deteksi pada setiap variasi parameter, maka penelitian ini menggunakan MSR sebagai Retinex yang dipakai dengan nilai parameter dari Petro, dimana mampu mendeteksi objek dari ketiga citra.

Nilai PSNR yang tinggi belum tentu menentukan citra dapat diproses dengan baik oleh proses berikutnya, dalam hal ini adalah deteksi tangan.



Gambar 6.4 Variasi Sigma Terhadap Nilai PSNR

Peneliti melakukan variasi pengujian nilai parameter sigma 1,sigma 2, sigma 3 pada parameter MSR terhadap citra gelap dan citra sedang. Hasil dari variasi nilai parameter terlihat pada gambar 6.4 untuk nilai psnr dan gambar 6.5 untuk nilai deteksi. Variasi nilai yang diberikan untuk sigma 1 [5:40:5], sigma 2 [50:120:10], sigma 3 [150:240:10]. Hasil yang di dapat cenderung stabil dan tidak memiliki nilai puncak. Pada variasi yang diberikan, selisih nilai PSNR tidak memiliki peningkatan yang signifikan dan nilai deteksi cenderung stabil diatas nilai 80. Nilai PSNR citra 1 pada variasi yang diberikan dalam kisaran 6 dan citra 2 dalam kisaran 8 hingga 9,5. Pada variasi yang diberikan, sigma 1 memiliki pengaruh yang besar pada nilai deteksi. Sigma 1 dengan nilai 5 menghasilkan nilai deteksi 80, seiring nilai sigma 1 ditingkatkan nilai deteksi stabil diatas 90.



Gambar 6.5 Variasi Sigma Terhadap Nilai Deteksi.

6.4. Pengujian Deteksi Tangan

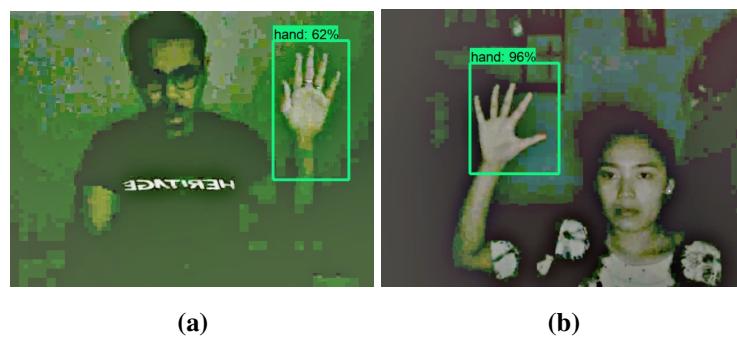
Pengujian dilakukan oleh 3 subjek dengan penurunan intensitas cahaya yang sama. Data pada tabel 6.6, 6.7, 6.8, dilakukan oleh subjek pertama, 6.9, 6.10, 6.11, dilakukan oleh subjek kedua, 6.12, 6.13 ,6.14, dilakukan oleh subjek subjek ketiga. Hasil dari ketiga subjek di plot pada grafik gambar 6.6, 6.7, 6.8.

Kondisi 0 lux tanpa Retinex pada subjek pertama menghasilkan 40%, subjek kedua 0% dan subjek ketiga 50%, sementara ketika sistem menggunakan Retinex menghasilkan 70% pada subjek pertama, 70% pada subjek kedua dan 100% untuk subjek ketiga. Keadaan pada intensitas 0 lux ini memiliki perbedaan nilai yang sangat jauh, pada citra gelap terlalu susah untuk menemukan dan mengenali objek yang ada. Retinex mampu memberikan perbaikan citra sehingga kontras pada citra meningkat. Peningkatan kontras pada citra dapat membantu sistem mendeteksi

keberadaan sebuah objek, terlihat pada gambar 6.6 dan 6.7 menunjukan perubahan sebelum dan sesudah dikenai Retinex.



Gambar 6.6 Subjek_1 (a: Subjek 2 tanpa Retinex, b: subjek 3 tanpa Retinex)



Gambar 6.7 Subjek_1 (a: Subjek 2 dengan Retinex, b: subjek 3 dengan Retinex)

Pengujian dengan Retinex menunjukan pada kondisi 0 lux dan 15 lux mendapat nilai 100%. Pada kondisi intensitas rendah Retinex mampu bekerja dengan baik yang ditunjukan dengan tingkat keberhasilan deteksi. Pengaruh Retinex sangat besar pada kondisi intensitas rendah. Kondisi intensitas diatas 15 lux nilai deteksi yang menggunakan retinex cenderung menurun. Berdasarkan tabel 6.7 pada kondisi intensitas diatas 15 lux sistem masih lebih baik mendeteksi tanpa menggunakan Retinex. Semakin menurun intensitas cahaya sistem tanpa Retinex mengalami penurunan performa yang mengakibatkan tidak mampu mendeteksi objek tangan yang ada, hal ini ditunjukan di grafik perbandingan pada gambar 6.8, 6.9, 6.10.

Tabel 6.6 Deteksi Subjek 1 Tanpa Retinex

Nilai Lux	Hasil Deteksi Subjek_1(Dheo)									
0 Lux	F	T	F	T	F	T	F	T	F	F
15 Lux	T	T	T	T	T	T	T	T	T	T
20 Lux	T	T	T	T	T	T	T	T	T	T
25 Lux	T	T	T	T	T	T	T	T	T	T
33 Lux	T	T	T	T	T	T	F	T	T	T
77 Lux	T	T	T	T	T	T	T	T	T	T
133 Lux	T	T	T	T	T	T	T	T	F	T

Tabel 6.7 Deteksi Subjek 1 Retinex

Nilai Lux	Hasil Deteksi Subjek_1(Dheo)									
0 Lux	F	T	T	T	F	T	T	T	F	T
15 Lux	T	T	T	T	T	T	T	T	T	T
20 Lux	T	T	T	T	T	T	T	T	T	T
25 Lux	T	T	T	T	T	T	T	T	T	T
33 Lux	T	T	T	T	T	T	T	T	T	T
77 Lux	T	T	T	T	T	T	T	T	F	T
133 Lux	F	T	T	F	T	T	T	T	F	T

Tabel 6.8 Perbandingan Deteksi Tangan Subjek_1

Kondisi lux	Tanpa Retinex	Retinex
0 lux	40%	70%
15 lux	100%	100%
20 lux	100%	100%
25 lux	100%	100%
33 lux	90%	100%
77 lux	100%	90%
133 lux	90%	70%

Tabel 6.9 Deteksi Subjek 2 Tanpa Retinex

Nilai Lux	Hasil Deteksi Subjek_2(Saya)									
0 Lux	F	F	F	F	F	F	F	F	F	F
15 Lux	T	T	T	T	T	T	T	T	T	T
20 Lux	T	T	T	T	T	T	T	T	T	T
25 Lux	T	T	T	T	T	T	T	T	T	T
33 Lux	T	T	T	T	T	T	T	T	T	T
77 Lux	T	T	T	T	T	T	T	T	T	T
133 Lux	T	T	T	T	T	T	T	T	T	T

Tabel 6.10 Deteksi Subjek 2 Retinex

Nilai Lux	Hasil Deteksi Subjek_2(Saya)									
0 Lux	T	F	T	F	T	T	F	T	T	T
15 Lux	T	T	T	T	T	T	T	T	T	T
20 Lux	F	T	T	F	T	T	T	T	F	T
25 Lux	T	T	T	T	T	T	F	T	T	T
33 Lux	T	T	T	T	T	T	T	T	T	T
77 Lux	T	T	T	T	F	T	F	T	T	T
133 Lux	T	T	T	T	T	T	T	T	T	T

Tabel 6.11 Perbandingan Deteksi Tangan Subjek_2

Kondisi lux	Tanpa Retinex	Retinex
0 lux	0%	70%
15 lux	100%	100%
20 lux	100%	70%
25 lux	100%	90%
33 lux	100%	100%
77 lux	100%	80%
133 lux	100%	100%

Tabel 6.12 Deteksi Subjek 3 Tanpa Retinex

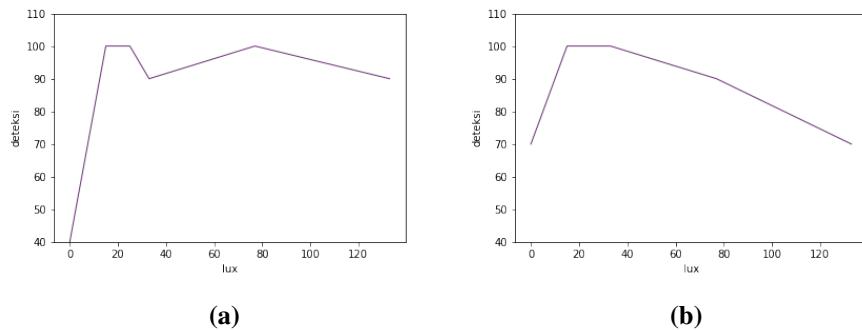
Nilai Lux	Hasil Deteksi Subjek_1(Dela)									
0 Lux	F	T	F	F	T	F	T	T	F	T
15 Lux	T	T	T	T	T	T	F	F	T	T
20 Lux	T	T	T	T	T	T	T	T	T	T
25 Lux	T	T	T	T	T	T	T	T	T	T
33 Lux	T	T	T	T	T	T	T	T	T	T
77 Lux	T	T	T	T	T	T	T	T	T	T
133 Lux	T	T	T	T	T	F	T	T	T	T

Tabel 6.13 Deteksi Subjek 3 Retinex

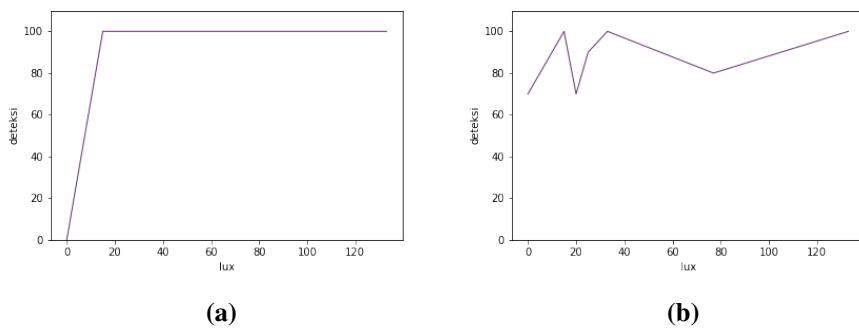
Nilai Lux	Hasil Deteksi Subjek_1(Dela)									
0 Lux	T	T	T	T	T	T	T	T	T	T
15 Lux	T	T	T	T	T	T	T	T	T	T
20 Lux	T	T	T	T	T	T	F	T	T	T
25 Lux	T	T	T	T	T	T	F	T	T	T
33 Lux	T	T	F	T	T	T	T	T	T	T
77 Lux	T	T	T	T	T	T	T	T	T	T
133 Lux	T	T	T	F	T	T	F	F	T	T

Tabel 6.14 Perbandingan Deteksi Tangan Subjek_3

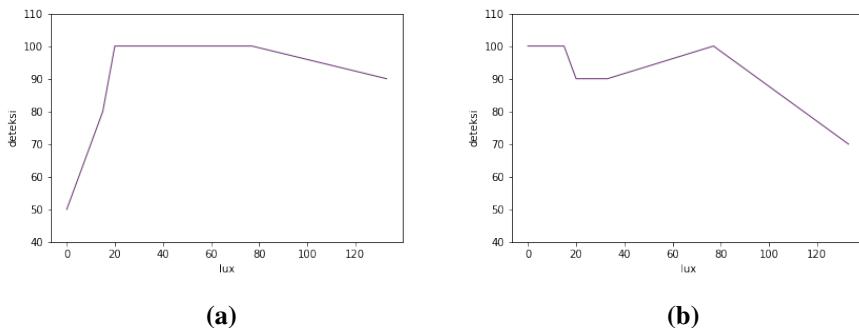
Kondisi lux	Tanpa Retinex	Retinex
0 lux	50%	100%
15 lux	80%	100%
20 lux	100%	90%
25 lux	100%	90%
33 lux	100%	90%
77 lux	100%	100%
133 lux	90%	70%



Gambar 6.8 Subjek_1 (a: Grafik tanpa Retinex, b: Grafik dengan Retinex)



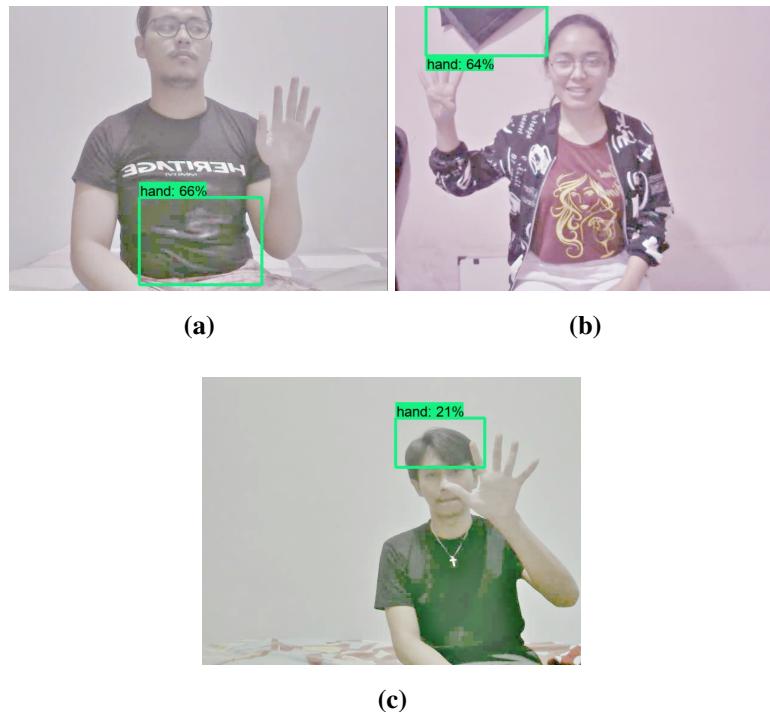
Gambar 6.9 Subjek_2 (a: Grafik tanpa Retinex, b: Grafik dengan Retinex)



Gambar 6.10 Subjek_3 (a:Grafik tanpa Retinex, b: Grafik dengan Retinex)

Berdasarkan ketiga percobaan pengaruh Retinex terlihat pada kondisi lux kurang dari sama dengan 15 lux. Pemberian Retinex pada intensitas yang tinggi menyebabkan *false positive* pada deteksi. Hal tersebut terjadi pada beberapa data dengan intensitas tinggi. Terlihat pada gambar 6.11, semakin terang kondisi cahaya jika diberikan Retinex menyebabkan nilai piksel yang sudah terang akan semakin terang sehingga piksel tersebut dapat berubah nilai menyebabkan objek yang bukan

tangan dikenali sebagai tangan. *False positive* sering muncul pada kondisi intensitas diatas 33 lux.



Gambar 6.11 Subjek_1 (a: Kondisi 133 lux, b: Kondisi 77 lux, c:Kondisi 133 lux)

6.5. Pengujian Pengenalan Gestur Tangan

77 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	0	10	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	0	10	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	0	10	0
Angka 9	0	0	0	0	0	0	0	0	0	10

133 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	0	10	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	0	10	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	0	10	0
Angka 9	0	0	0	0	0	0	0	0	0	10

WITH RETINEX

0 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	0	9
Angka 0	0	0	0	0	0	0	0	0	0	0
Angka 1	0	0	0	0	0	0	0	0	0	0
Angka 2	0	0	0	0	0	0	0	0	0	0
Angka 3	0	0	0	0	0	0	0	0	0	0
Angka 4	0	0	0	0	0	0	0	0	0	0
Angka 5	0	0	0	0	0	0	0	0	0	0
Angka 6	0	0	0	0	0	0	0	0	0	0
Angka 7	0	0	0	0	0	0	0	0	0	0
Angka 8	0	0	0	0	0	0	0	0	0	0
Angka 9	0	0	0	0	0	0	0	0	0	0

15 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	8	2	0	0	0	0	0	0	0
Angka 2	0	0	10	0	0	0	0	0	0	0
Angka 3	0	0	2	7	1	0	0	0	0	0
Angka 4	0	0	0	0	7	0	3	0	0	1
Angka 5	0	0	0	0	10	0	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	2	0	0	0	6	2	0	0
Angka 8	0	0	2	0	0	0	0	7	1	0
Angka 9	0	0	3	2	1	0	4	0	0	0

33 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	6	4	0	0	0	0	0	0	0
Angka 2	0	0	10	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	5	5	0	0	0	0
Angka 5	0	0	0	0	0	10	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	3	0	0	0	0	10	0	0
Angka 8	0	0	2	0	0	0	0	1	7	0
Angka 9	0	0	2	2	0	0	6	0	0	4

77 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	8	2	0	0	0	0	0	0	0
Angka 2	0	0	10	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	5	0	5	0	0	0
Angka 5	0	0	0	0	1	9	0	0	0	0
Angka 6	0	0	1	0	0	0	9	0	0	0
Angka 7	0	0	2	0	0	0	0	8	0	0
Angka 8	0	0	0	0	0	0	0	0	10	0
Angka 9	0	0	4	2	0	0	0	0	0	4

133 lux	Pengenalan Subjek_1(dheo)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	0	10	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	1	1	8	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	0	10	0
Angka 9	0	0	4	0	0	0	2	0	0	4

Tabel 6.15 Perbandingan Pengenalan Gestur Tangan subjek_1

Kondisi lux	Tanpa Retinex	Retinex
0 lux		
15 lux	55%	23%
20 lux	69%	53%
25 lux	68%	42%
33 lux	70%	51%
77 lux	70%	77%
133 lux	75%	90%

subjek 2 saya NO RETINEX

0 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	7	3	0	0	0	0	0	0	0	0
Angka 1	8	2	0	0	0	0	0	0	0	0
Angka 2	5	5	0	0	0	0	0	0	0	0
Angka 3	5	3	0	0	0	0	0	0	0	2
Angka 4	1	3	0	0	0	0	0	0	0	6
Angka 5	1	1	0	0	0	0	0	0	0	8
Angka 6	5	3	0	0	0	0	0	0	0	2
Angka 7	6	3	0	0	0	0	0	0	0	1
Angka 8	5	2	0	0	0	0	0	0	0	3
Angka 9	2	3	0	0	0	0	0	0	0	5

15 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	1	7	0	0	0	0	2	0	0
Angka 2	0	0	0	0	0	0	4	6	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	1	9	0	0	0	0
Angka 6	0	0	0	0	4	0	6	0	0	0
Angka 7	0	0	0	0	0	0	0	0	10	0
Angka 8	0	0	0	0	0	0	0	0	10	0
Angka 9	0	0	0	1	0	2	0	2	0	5

20 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	10	0	0	0
Angka 2	0	0	0	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	1	9	0	0	0	0
Angka 6	0	0	0	0	1	0	9	0	0	0
Angka 7	0	0	0	0	1	0	1	8	0	0
Angka 8	0	0	0	0	0	0	0	9	1	0
Angka 9	0	0	0	0	2	3	0	0	0	5

25 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	9	1	0	0	0	0	0	0	0
Angka 2	0	0	2	0	0	0	6	7	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	0	4	6	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	8	2	0
Angka 9	0	0	0	2	2	0	0	0	0	6

33 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	9	0	0	0	0	0	0	1	0
Angka 2	0	0	2	0	0	5	0	3	0	0
Angka 3	0	0	0	8	1	0	0	1	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	4	6	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	10	0	0
Angka 9	0	0	0	0	1	0	0	0	1	8

77 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	9	0	0	0	0	0	0	1	0
Angka 2	0	0	2	0	1	0	3	4	0	0
Angka 3	0	0	0	9	0	0	0	1	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	4	6	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	9	1	0
Angka 9	0	0	0	0	1	0	0	0	1	8

133 lux	Pengenalan Subjek_2(saya)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	9	1	0	0	0	0	0	0	0
Angka 2	0	0	4	0	0	0	6	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	6	4	0	0	0	0
Angka 6	0	0	0	0	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	10	0	0
Angka 9	0	0	0	0	0	0	0	0	0	10

Retinex

Tabel 6.16 Perbandingan Pengenalan Gestur Tangan subjek_1

Kondisi lux	Tanpa Retinex	Retinex
0 lux	14%	
15 lux	61%	%
20 lux	72%	%
25 lux	73%	%
33 lux	73%	%
77 lux	75%	%
133 lux	77%	%

0 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	1	9	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	10	0	0	0	0	0	9	0	0
Angka 3	0	0	2	0	0	0	1	0	2	5
Angka 4	0	0	0	0	6	0	4	0	0	0
Angka 5	0	0	0	0	5	0	0	0	0	5
Angka 6	0	0	0	0	4	0	4	0	1	1
Angka 7	0	0	0	0	2	0	5	0	1	2
Angka 8	0	0	1	0	3	0	2	0	0	4
Angka 9	0	0	1	0	7	0	0	0	0	2

15 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	9	1	0	0	0	0	0	0	0
Angka 2	0	0	0	1	0	0	0	9	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	0	10	0	0	0	0
Angka 6	0	0	0	0	5	0	0	5	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	10	0	0
Angka 9	0	0	0	3	0	0	0	7	0	0

20 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	0	0	1	0	0	0	9	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	2	8	0	0	0	0
Angka 6	0	0	0	0	5	0	2	3	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	10	0	0
Angka 9	0	0	0	1	8	0	0	1	0	0

25 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	0	0	0	0	0	2	8	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	4	6	0	0	0	0
Angka 6	0	0	0	0	6	0	2	2	0	0
Angka 7	0	0	0	0	0	0	0	10	0	0
Angka 8	0	0	0	0	0	0	0	10	0	0
Angka 9	0	0	0	3	7	0	0	0	0	0

33 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	8	0	0	0	0	2	0	0	0
Angka 2	0	0	2	0	0	0	1	3	4	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	0	10	0	0	0	0
Angka 6	0	0	0	1	0	0	8	1	0	0
Angka 7	0	0	0	0	0	0	0	8	2	0
Angka 8	0	0	0	0	0	0	0	10	0	0
Angka 9	0	0	0	0	8	0	1	0	1	0

77 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	2	0	0	0
Angka 2	0	2	8	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	10	0	0	0	0	0
Angka 5	0	0	0	0	7	3	0	0	0	0
Angka 6	0	0	0	1	0	0	10	0	0	0
Angka 7	0	0	1	1	1	0	1	6	0	0
Angka 8	0	0	0	0	0	0	1	8	1	0
Angka 9	0	0	0	1	5	0	0	2	0	2

133 lux	Pengenalan Subjek_3(dela)									
Klasifikasi	0	1	2	3	4	5	6	7	8	9
Angka 0	10	0	0	0	0	0	0	0	0	0
Angka 1	0	10	0	0	0	0	0	0	0	0
Angka 2	0	2	8	0	0	0	0	0	0	0
Angka 3	0	0	0	10	0	0	0	0	0	0
Angka 4	0	0	0	0	6	0	4	0	0	0
Angka 5	0	0	0	0	6	4	0	0	0	0
Angka 6	0	0	0	1	0	0	10	0	0	0
Angka 7	0	0	0	0	0	0	7	3	0	0
Angka 8	0	0	0	0	0	0	1	7	2	0
Angka 9	0	0	0	1	7	0	2	0	0	0

Tabel 6.17 Perbandingan Pengenalan Gestur Tangan subjek_3

Kondisi lux	Tanpa Retinex	Retinex
0 lux	23%	%
15 lux	59%	%
20 lux	60%	%
25 lux	58%	%
33 lux	66%	%
77 lux	70%	%
133 lux	63%	%

6.6. Pengujian Sistem Keseluruhan

dfafa

BAB VII

KESIMPULAN DAN SARAN

7.1. Kesimpulan

7.2. Saran

DAFTAR PUSTAKA

- Aribowo, E., Yustina, E., Studi, P., Informatika, T., Teknologi, F., Universitas, I. & Dahlan, A., 2009, *Implementasi Metode Retinex Untuk Pencerahan Citra*, Jurnal Informatika, 3, 2, 323–330.
- Loh, Y.P., Liang, X. & Chan, C.S., 2019, *Low-light image enhancement using Gaussian Process for features retrieval*, Signal Processing: Image Communication, 74, 175–190. <https://doi.org/10.1016/j.image.2019.02.001>.
- Saputra, L.K.P., 2016, *Perbandingan Varian Metode Multiscale Retinex Untuk Peningkatan Akurasi Deteksi Wajah Adaboost HAAR-like*, Jurnal Teknik Informatika dan Sistem Informasi, 2, 1, 89–98.
- Shen, L., Yue, Z., Feng, F., Chen, Q., Liu, S. & Ma, J., 2017, *MSR-net: Low-light Image Enhancement Using Deep Convolutional Network*, , , January. <http://arxiv.org/abs/1711.02488>.
- Tanaka, Y., Yamashita, Y., Nishikawa, K., Yamaguchi, T. & Nishitani, T., 2019, *Retinex Foreground Segmentation for Low Light Environments*, 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC 2018 - Proceedings, , November, 285–290.
- Yingxin, X., Jinghua, L., Lichun, W. & Dehui, K., 2017, *A Robust Hand Gesture Recognition Method via Convolutional Neural Network*, Proceedings - 2016 International Conference on Digital Home, ICDH 2016, 64–67.
- Madenda,Sarifudin., 2015, *Pengolahan & Video Digital*, Erlangga, Jakarta[BUKU].
- Nielsen, M., 2015, *Neural Networks and Deep Learning*, Determination Press., [Daring]. tersedia di <http://neuralnetworksanddeeplearning.com>.
- Rinaldi, Munir., 2004, *Pengolahan Citra Digital*, Bandung, Informatika[BUKU].
- Hidayatullah, Priyanto,. 2017, *Pengolahan Citra Digital Teori dan Aplikasi Nyata*, Bandung, Informatika[BUKU].
- Barczak, A.L.C., Reyes, N.H., Abastillas, M., Piccio, A. & Susnjak, T., 2011, *A New 2D Static Hand Gesture Colour Image Dataset for ASL Gestures*, Res. Lett. Inf. Math. Sci., 15, 12-20. <http://iims.massey.ac.nz/research/letters/>.
- Arabi, S., Haghigat, A. & Sharma, A., 2019, *A deep learning based solution for construction equipment detection: from development to deployment* , , April.
- Srinivasan, R., 2016, *Implementing Histogram Equalization and Retinex*, , , July.
- Huang, H., Chong, Y., Nie, C. & Pan, S., 2019, *Hand Gesture Recognition with Skin Detection and Deep Learning Method Hand Gesture Recognition with Skin Detection and Deep Learning Method*, J. Phys.: Conf. Ser. 1213 022001.

- Posada-Gómez, Rubén & Sanchez Medel, Luis & Alor-Hernández, Giner & Martinez Sibaja, Albino & Aguilar-Laserre, A. & Leija-Salas, L.. 2007. *A Hands Gesture System Of Control For An Intelligent Wheelchair*. 68 - 71. 10.1109/ICE-EE.2007.4344975.
- Kemenkes, 2018, *Indonesia Inklusi dan Ramah Disabilitas*[Daring], tersedia di <https://www.kemkes.go.id/resources/download/pusdatin/infodatin/infodatin-disabilitas.pdf>.
- Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. & Adam, H., 2017, *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications*, <http://arxiv.org/abs/1704.04861>.
- Sandler, M., Howard, A., Zhu, M., Zhmoginov, A. & Chen, L.C., 2018, *MobileNetV2: Inverted Residuals and Linear Bottlenecks*, Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 45104520.
- Google AI Blog., 2018, *MobileNetV2: The Next Generation of On-Device Computer Vision Networks*[Daring], tersedia di <ai.googleblog.com/2018/04/mobilenetv2-next-generation-of-on.html?m=1>.
- Leonard, L.C., 2017, *Web-Based Behavioral Modeling for Continuous User Authentication (CUA)*, 1 edisi, Elsevier Inc., [Daring]. tersedia di DOI:10.1016/bs.adcom.2016.12.001.
- Hui, Jonathan., 2018, *mAP (mean Average Precision) for Object Detection*[Daring], tersedia di www.medium.com/@jonathan_hui/map-mean-average-precision-for-object-detection-45c121a31173.
- Chen, X., Fang, H., Lin, T.Y., Vedantam, R., Gupta, S., Dollár, P & Zitnick, C.L., 2015, *Microsoft COCO Captions: Data Collection and Evaluation Server*, arXiv:1504.00325.
- Petro., Ana-Belen & Sbert., Catalina & Morel., Jean-Michel., 2014. *Multiscale Retinex*. Image Processing On Line, 4. 71-88, DOI:10.5201/ipol.2014.107.
- X. He., T. Wang., Y. Jia., Y. Wang., Z. Xie and D. Xie., 2016, *Studying fidelity issues in image enhancement by means of multi-scale retinex with color restoration*, 2016 3rd International Conference on Systems and Informatics (ICSAI), Shanghai, pp. 536-540, DOI: 10.1109/ICSAI.2016.7811013.
- A. S. Parihar and K. Singh., *A study on Retinex based method for image enhancement*, 2018, 2nd International Conference on Inventive Systems and Control (ICISC), Coimbatore, pp. 619-624, DOI: 10.1109/ICISC.2018.8398874.
- CV NOTE., 2019, *PSNR and SSIM Metric: Python Implementation* [Daring],tersedia di<https://cvnote.ddlee.cc/2019/09/12/psnr-ssim-python>.