BAB II
TINJAUAN PUSTAKA DAN DASAR TEORI

2.1. Tinjauan Pusataka

Tinjauan pustaka diambil dari beberapa penelitian sebelumnya yang terkait dengan *marker* ArUco, MobileNetV2, dan MobileViTV2. Proses indentifikasi *marker* ArUco sebagai alat bantu penentuan lokasi untuk tuna netra sangat penting peranannya. Model MobileNetV2 memungkinkan bisa mengidentifikasi gambar *marker* ArUco pada fitur lokal dengan sangat baik, sementara MobileViTV2 ini memberikan kerangka kerja untuk menangani konteks global dari fitur gambar *marker* ArUco dengan tetap menjaga kopleksitas dan mempertahankan model yang ringan saat digunakan. Integrasi kedua metode ini diharapkan mampu memberikan kemampuan yang lebih komprehensif tentang meningkatkan keberhasilan deteksi *marker* ArUco pada kondisi lingkungan yang menantang dan memberikan panduan terhadap tuna netra dalam bernavigasi dalam ruangan. Pembahasan ekplorasi teori, penelitian terkait, dan aplikasi praktis yang membahas konsep, keunggulan dan integrasi dari model MobilNetV2 dan MobileViTV2, membuka jalan untuk pemahaman yang lebih mendalam dan penerapan yang tepat terkait alat bantu navigasi dalam ruangan.

Kemajuan yang signifikan dibuat dalam penggunaan *deep learning* dan *computer vision* dalam penelitian medis, seperti gambar biomedis (SAKÇİ dkk., 2022), prediksi kanker (Matsui & Crowley, 2023), deteksi objek (Schmitt-Koopmann dkk., 2022), dan pengenalan objek (Mondal dkk., 2022). *Smartphone* menjadi sangat penting karena memiliki berbagai kemampuan antara lain kekuatan pemrosesan, kamera yang terintegrasi, dan berbagai sensor. Teknologi dan kemampuan telepon pintar yang muncul dan berkembang ini memungkinkan para peneliti membangun aplikasi baru untuk membantu tunanetra mengidentifikasi objek (Manjari dkk., 2020b; Mantoro & Zamzami, 2022) dan bernavigasi di dalam ruangan dengan aman (Al-Madani dkk., 2019; Simões dkk., 2020). Peneliti bertujuan untuk meningkatkan kualitas penelitian ini dengan meningkatkan akurasi

dan meminimalkan waktu eksekusi agar sesuai untuk penggunaan secara *real-time*. Di bagian ini, peneliti menyajikan beberapa sistem navigasi umum yang menggunakan *marker* dan *deep learning*.

Marker persegi adalah tag berbentuk persegi dengan batas hitam tebal dan bagian dalam untuk mewakili kode. Bagian pada wilayah area dalam berisi gambar atau kode biner yang direpresentasikan sebagai kisi-kisi wilayah hitam dan putih. Pada penelitian (T Pivoňka dkk., 2023) meneiliti sebuah sistem lokalilasi visual untuk *docking* otomatis berbasis penanda fidusial. Sistem ini dirancang untuk *docking* sebuah robot logistik mobile di bawah rak penyimpanan dalam aplikasi industri. Masalah utama yang diatasi dalam artikel ini adalah kebutuhan akan kekokohan tinggi, akurasi yang dapat digunakan, dan pemeliharaan yang mudah dalam proses *docking* dengan biaya rendah. Solusi yang diusulkan terdiri dari dua bagian utama: alat kalibrasi dan sistem lokalilasi. Sistem lokalilasi menghitung posisi robot relatif terhadap lokasi *docking* menggunakan *solver* Perspektif-n-Titik untuk penanda yang terdeteksi. Algoritma yang dipilih, Struktur-dari-Gerakan (SfM), umumnya digunakan dalam robotika untuk rekonstruksi skenario 3D dan tugas simultan lokalilasi dan pemetaan (SLAM). SfM memungkinkan rekonstruksi titik 3D dan posisi kamera dari beberapa sudut pandang dengan mendeteksi fitur visual dan menemukan korespondensi antara gambar. Sistem yang diusulkan telah diuji untuk presisi dan kekokohan dalam berbagai kondisi kerja. Alat kalibrasi mencapai akurasi rata-rata sekitar 1mm untuk 10+ gambar, yang dianggap cukup untuk penggunaan praktis. Keandalan sistem lokalilasi dievaluasi dalam kondisi pencahayaan yang berbeda dan gerakan kamera, menunjukkan kemampuannya untuk bekerja dengan kamera yang bergerak. Khan dkk. mengembangkan sistem generik untuk navigasi menggunakan *marker* AR ToolKit (Khan dkk., 2019). Itu menggunakan kamera *smartphone* untuk mengenali *marker* yang ditempatkan di langit-langit gedung. Pengguna memilih tujuan mereka dan kemudian aplikasi menentukan jalur terpendek ke tujuan berdasarkan *marker* terdekat pertama. Umpan balik audio diberikan untuk memandu tunanetra mencapai titik akhir mereka. Namun, sistem ini juga menghadapi beberapa keterbatasan. Sulit bagi tunanetra yang memegang *smartphone* untuk mengarahkan kamera, mendeteksi *marker* di

langit-langit gedung. Sistem diuji hanya oleh orang dengan penglihatan normal yang ditutup matanya. Penelitian Lee dkk. mengusulkan sistem navigasi dalam ruangan menggunakan *marker* dan *augmented reality* (Lee & Kim, 2020). Itu melakukan lokalisasi hybrid dengan menggunakan gambar *marker* serta data *Inertial Measurement Unit* (IMU) dari *smartphone*. Pertama, peta dalam ruangan disiapkan untuk mencatat posisi tempat-tempat dalam ruangan. Kemudian, *marker* dibuat dan dicetak untuk tempat-tempat terdaftar ini. Sistem navigasi digunakan untuk membantu pengguna berhasil mencapai tujuan mereka. Namun, tidak dapat mendeteksi *marker* di lingkungan yang ramai, karena *marker* dipasang di lantai. Selain itu, gagal mengidentifikasi *marker* dari jarak jauh.

Penelitian (Yang dkk., 2023) menjawab dalam upaya meningkatkan diagnosis efisien dari hematologic malignancies menggunakan gambar mikroskopis sumsum tulang. Pendekatan yang digunakan melibatkan penggunaan MultiPathGAN untuk meningkatkan kualitas gambar dengan stain normalization dan MobileViTv2 sebagai model *deep learning* untuk klasifikasi penyakit. Hasilnya menunjukkan bahwa model MobileViTv2 mampu mencapai akurasi tinggi dalam diagnosis, dengan akurasi rata-rata sebesar 94,28% dan akurasi tes pada tingkat pasien mencapai 96,72%, sambil tetap memiliki ukuran model yang relatif kecil dan membutuhkan sumber daya komputasi yang efisien. Penelitian yang dilakukan oleh Dash dkk. mengusulkan sistem *Augmented reality* (AR) untuk membantu anak belajar huruf (Dash dkk., 2018). Penelitian selanjutnya dari (Lv dkk., 2023) untuk mengatasi tantangan segmentasi cacat permukaan aluminium secara *online* dengan cepat dan tepat, yang sering kali diakibatkan oleh penuaan peralatan, kondisi produksi yang sulit, dan parameter proses yang tidak memadai. Untuk mencapai tujuan tersebut, penelitian menggunakan pendekatan segmentasi berbasis *Convolutional Neural Network* (CNN) dengan model MobileViTv2 yang ringan dan efisien, serta dilengkapi dengan mekanisme fusion attention (FA) untuk meningkatkan fokus pada karakteristik kritis dalam citra. Hasil eksperimen menunjukkan bahwa model yang diusulkan mampu mencapai *mean Intersection over Union* (mIoU) sebesar 87,01%, dengan kecepatan segmentasi sebesar 61,67 fps, dan ukuran model yang dihasilkan sebesar 16,23 MB, menunjukkan efisiensi

dalam penggunaan sumber daya. Dalam perbandingan dengan model-model lain, hasil pengujian menunjukkan bahwa model yang diusulkan memiliki potensi untuk meningkatkan akurasi meskipun memiliki struktur model yang sederhana, sehingga memberikan kontribusi penting dalam pengembangan sistem segmentasi cacat permukaan aluminium yang efisien dan dapat diimplementasikan secara *online*, dengan potensi untuk meningkatkan efisiensi produksi dan kualitas produk aluminium. Penelitian berikutnya dari Elgendy dkk. Yang menerapkan model CNN untuk mendeteksi *marker* yang disajikan dalam sebuah adegan. Kemudian, objek virtual ditampilkan di bagian atas *marker* yang terdeteksi. Untuk merendernya dengan benar, mereka harus meletakkannya di depan kamera dengan posisi dan orientasi yang benar. Sistem ini mencapai akurasi tinggi dalam identifikasi *marker*. Namun, gagal mendeteksi *marker* dari jarak jauh. Penelitian selanjutnya dari Elgendy dkk. mengusulkan suatu sistem untuk membantu tunanetra dalam navigasi dalam ruangan menggunakan *marker* (Elgendy dkk., 2021; Elgendy, Guzsvinecz, dkk., 2019). Langkah identifikasi didefinisikan ulang sebagai masalah klasifikasi, dan CNN digunakan untuk mengidentifikasi *marker*. Beberapa teknik CV digunakan untuk memilih kandidat, dan kemudian diumpankan ke model CNN untuk mengklasifikasikan apakah mereka adalah *marker* atau bukan. Sistem ini membantu tunanetra bernavigasi di dalam ruangan menggunakan *marker*. Itu mencapai akurasi tinggi. Penggunaan teknik CV untuk menyeleksi calon *marker* membutuhkan waktu proses yang perlu diminimalkan. Bazi et al. mengusulkan sistem navigasi untuk membantu tunanetra mengenali beberapa objek dalam gambar menggunakan SVM convolutional multi-label (Bazi dkk., 2019). Ini menggunakan kamera portabel yang dipasang pada pelindung ringan yang dikenakan oleh pengguna untuk mengambil gambar dan mengirimkannya melalui kabel USB ke unit pemrosesan laptop. Untuk mengidentifikasi objek, satu set SVM linier digunakan sebagai filter di setiap lapisan konvolusional untuk menghasilkan satu set peta fitur baru. Akhirnya, output diumpankan lagi ke pengklasifikasi SVM linier untuk melaksanakan tugas klasifikasi. Namun, ukuran dan berat unit pemroses sekali lagi menjadi masalah besar bagi tunanetra. Juga, gagal mendeteksi *marker* dari jarak yang lebih jauh. Penelitian berikutnya dari Kayukawa dkk. mengusulkan

sistem penghindaran tabrakan untuk tunanetra menggunakan kamera yang terintegrasi ke dalam koper (Kayukawa dkk., 2019). Itu menggunakan gambar kedalaman untuk menentukan risiko tabrakan dengan orang buta menggunakan model CNN untuk mendeteksi objek sementara YOLOv2 digunakan untuk mendeteksi pejalan kaki menggunakan pendeteksi RGB. Sistem ini mendeteksi individu secara efisien. Namun, waktu eksekusi perlu diminimalkan untuk penggunaan real-time. Penelitian berikutnya dari (M. Liu dkk., 2020) yaitu tentang mengembangkan metode deteksi objek kecil berdasarkan YOLOv3. Struktur CNN darknet telah dimodifikasi dengan meningkatkan operasi konvolusi di awal untuk meningkatkan kinerja. Metode yang diusulkan meningkatkan kinerja pendeteksian objek kecil. Namun, itu tidak cocok untuk penggunaan real-time oleh *smartphone*.

Penelitian menggunakan MobileNetV2 dari (Sutaji & Yıldız, 2022) membahas penggunaan model *Convolutional Neural Network* (CNN) untuk deteksi otomatis dan klasifikasi penyakit tanaman pada daun. Penelitian ini mengusulkan sebuah ensemble dari model CNN MobileNetV2 dan Xception untuk meningkatkan kinerja deteksi penyakit tanaman. Penelitian tersebut menyoroti akurasi yang lebih tinggi yang dicapai oleh model ensemble yang diusulkan dibandingkan dengan model individual dan model CNN terkini lainnya, di sejumlah dataset. Model yang diusulkan juga menunjukkan jumlah parameter dan ukuran file yang lebih kecil, menjadikannya cocok untuk integrasi ke dalam perangkat *online*. Implikasi dari penelitian ini sangat signifikan untuk industri pertanian, karena memberikan wawasan tentang perkembangan terbaru dalam diagnosis penyakit tanaman berbantuan komputer. Penelitian selanjutnya datang dari (Islam dkk., 2023) mengimplementasikan sistem deteksi objek otomatis dan deskripsi lingkungan sekitar untuk individu tunanetra dan lanjut usia. Masalah utama yang diatasi adalah kebutuhan akan teknologi asistif untuk membantu individu tunanetra dalam deteksi objek dan navigasi. Solusi yang diusulkan memanfaatkan teknik SSDLite MobileNetV2 dan dataset open-source untuk mengembangkan sistem deteksi objek otomatis dan narasi lingkungan. Setup hardware melibatkan penggunaan Raspberry Pi 4B dan kamera Pi, sementara implementasi perangkat lunak menggunakan teknik *deep learning* dan modul teks-ke-suara. Eksperimen pilot dilakukan untuk

mengevaluasi fungsionalitas prototipe. Pada penelitian tersebut menghasilkan tingkat akurasi deteksi sebesar 88,89%, presisi sebesar 0,892, dan recall sebesar 0,89. Secara keseluruhan, karya ini menyajikan solusi yang efisien biaya untuk meningkatkan kemandirian dan keselamatan bagi individu tunanetra dalam deteksi objek dan navigasi di sekitar mereka.

Dari tinjaun pustaka tentang penelitian dengan tema MobileNetV2, MobileViTV2 dan Aruco *Marker* masih terdapat kurangnya eksplorasi mendalam mengenai bagaimana model MobileNetV2 terintegrasi dengan MobileViTV2 yang dapat mempengaruhi pengambilan akurasi deteksi terhadap *marker* AruCo. Perlu adanya penelitian lanjutan yang membahas secara rinci bagaimana aspek ektraksi fitur lokal yang sangat baik dilakukan MobileNetV2 dengan mempertahankan konseptual global dengan menggunakan MobileViTV2 untuk meningkatkan tingkat akurasi dan keceptatan deteksi *marker*. Belum adanya penelitian yang fokus pada implementasi praktis dari model terintegrasi ini dalam konteks *Marker* AruCo. Penelitian ini dapat mengidentifikasi tantangan dan keberhasilan implementasi model MobileNetV2 terintegrasi dengan MobileViTV2 dalam skala yang lebih luas, menggambarkan konteks-konteks yang optimal untuk penerapannya.

Penelitian juga dapat memfokuskan pada pengembangan model yang lebih terperinci dan kontekstual dengan mengintegrasikan *multi query attention* untuk menggabungkan antara fitur lokal dan kontekstual global pada setiap tahapan sampling. Dapat dimaksimalkan untuk memberikan rekomendasi perbaikan performa kemampuan deteksi pada lingkungan ekstrim dengan parameter sudut pembacaan, intensitas cahaya, jarak dan gerakan. Melalui penelitian lebih lanjut, diharapkan dapat mengisi celah pengetahuan ini dan memberikan kontribusi yang signifikan untuk pengembangan teori dan praktik keberlanjutan pada bidang navigasi dalam ruangan.

2.2. Landasan Teori

Citra atau gambar dapat diartikan sebagai fungsi matematika dua dimensi yang dinyatakan dengan notasi $f(x, y)$, di mana x dan y merupakan koordinat pada bidang datar. Nilai fungsi f pada setiap pasangan koordinat (x, y) disebut dengan

tingkat intensitas keabuan (*grey level*). Citra digital terdiri dari elemen-elemen yang berhingga, masing-masing memiliki lokasi dan nilai tertentu. Elemen-elemen ini disebut sebagai picture element, image element, atau lebih umum dikenal sebagai piksel (pels) (Fajar, 2013). Citra digital merujuk pada gambar yang dapat diolah menggunakan komputer untuk memperoleh informasi tertentu. Informasi ini berupa nilai-nilai numerik yang direpresentasikan dalam bentuk piksel. Sebagai contoh, sebuah citra berukuran 150x150 piksel dapat disajikan dalam bentuk angka, seperti yang ditunjukkan pada Gambar 2.1. Citra yang awalnya merupakan representasi visual dari objek, ketika diolah oleh komputer, dapat dikonversi menjadi matriks angka yang merepresentasikan citra tersebut.

Gambar 2. 1 Citra objek beserta matriks

Akuisisi citra digital merupakan serangkaian langkah untuk memperoleh citra digital dari sumber aslinya, seperti kamera digital, pemindai (*scanner*), atau sensor lainnya. Salah satu model citra digital yang paling sederhana adalah citra biner atau citra hitam-putih, yang hanya menggunakan dua nilai piksel, yaitu 0 untuk hitam dan 1 untuk putih. Model citra biner ini cocok untuk menggambarkan citra yang hanya memiliki dua tingkat intensitas. Dalam sistem pencitraan, sensor-sensor disusun sedemikian rupa hingga membentuk bidang dua dimensi, di mana nilai $f(x, y)$ sebanding dengan energi yang dipancarkan oleh sumber cahaya. Akibatnya, nilai $f(x, y)$ harus bernilai positif dan terbatas (tidak boleh nol atau tak hingga).

$$0 < f(x, y) < \infty \tag{2.1}$$

Fungsi $f(x, y)$ dapat dipisahkan menjadi dua komponen utama, yaitu:

1. Intensitas cahaya yang berasal dari sumber cahaya, dilambangkan dengan $i(x, y)$, di mana nilainya berada pada rentang $0 \leq i(x, y) < \infty$.

2. Derajat reflektansi objek, yang menunjukkan kemampuan objek memantulkan cahaya, dilambangkan dengan $r(x, y)$, dengan nilai yang berada dalam rentang $0 \leq r(x, y) \leq 1$.

Besar fungsi $f(x, y)$ merupakan hasil perkalian dari kedua komponen tersebut, sehingga dapat dinyatakan dengan persamaan berikut:

$$f(x, y) = i(x, y) \cdot r(x, y) \tag{2.2}$$

Dengan demikian, nilai fungsi $f(x, y)$ menggambarkan kombinasi antara jumlah cahaya yang diterima dan kemampuan objek memantulkannya dengan $0 < i(x, y) < \infty$ dan $0 < r(x, y) < 1$.

Dalam pengolahan citra digital, tetangga sebuah piksel merujuk pada sekelompok piksel yang berada di sekitar piksel tertentu, yang umumnya digunakan dalam operasi seperti filter spasial dan deteksi tepi. Area tetangga ini didefinisikan oleh ukuran tertentu, yang disebut sebagai "ukuran kernel" atau "ukuran jendela," dan dapat berbentuk kotak atau lingkaran, tergantung pada kebutuhan aplikasi. Penggunaan tetangga piksel sangat penting karena memungkinkan pelaksanaan operasi seperti filtering, konvolusi, dan deteksi tepi, di mana informasi dari piksel-piksel tetangga digunakan untuk memodifikasi nilai piksel pusat guna mencapai efek yang diinginkan, seperti penghalusan gambar atau penajaman tepi.

Beberapa operasi dalam pengolahan citra membutuhkan informasi dari tetangga piksel untuk menghasilkan output yang akurat. Sebuah piksel p pada koordinat (x, y) memiliki 4 tetangga horizontal dan vertikal, dengan koordinat $(x + 1, y), (x - 1, y), (x, y + 1)$, dan $(x, y - 1)$. Kumpulan piksel ini disebut sebagai 4-tetangga dari p, yang dinotasikan dengan $N_4(p)$. Selain itu, piksel p juga memiliki 4 tetangga diagonal, yang disebut sebagai $N_D(p)$, dengan koordinat $(x + 1, y + 1), (x + 1, y - 1), (x - 1, y + 1)$, dan $(x - 1, y - 1)$. Jika himpunan

$N_D(p)$ dan $N_4(p)$ digabung, maka membentuk 8-tetangga dari p, yang dinotasikan sebagai $N_8(p)$ (Gonzalez & Woods, 2002).

$x-1, y-1)$	$(x-1, y)$	$(x-1, y+1)$
$(x, y-1)$	p	$(x, y+1)$
$(x+1, y-1)$	$(x+1, y)$	$(x+1, y+1)$

Gambar 2. 2 Tetangga sebuah piksel

2.2.1. Ukuran Jarak antar piksel

Jarak antar piksel dalam citra digital adalah ukuran fisik atau spasial yang mengukur jarak antara dua piksel bersebelahan dalam citra tersebut. Jarak ini dinyatakan dalam satuan panjang, seperti milimeter (mm), sentimeter (cm), atau piksel, dan sangat berpengaruh terhadap resolusi serta kualitas gambar yang dihasilkan. Semakin kecil jarak antar piksel, semakin tinggi resolusi citra, sehingga detail yang ditampilkan semakin tajam. Jarak antar piksel pada citra digital ditentukan oleh dua faktor utama: ukuran fisik sensor atau perangkat pengambilan gambar dan jumlah piksel yang digunakan untuk menangkap citra tersebut.

Resolusi citra mengacu pada jumlah piksel yang terdapat dalam citra secara horizontal dan vertikal. Sebagai contoh, citra dengan resolusi 1920x1080 memiliki 1920 piksel dalam lebar dan 1080 piksel dalam tinggi. Semakin tinggi resolusi sebuah citra, semakin banyak piksel yang terkandung di dalamnya, sehingga jarak antar piksel menjadi lebih kecil. Faktor kedua adalah ukuran fisik citra, yaitu ukuran nyata dari citra ketika dicetak atau ditampilkan pada media fisik. Jarak antar piksel dipengaruhi oleh kombinasi antara ukuran fisik citra dan resolusinya. Sebagai contoh, jika citra beresolusi 1920x1080 dicetak pada media berukuran 10x5 cm, jarak antar piksel akan lebih kecil dibandingkan jika citra yang sama dicetak pada media berukuran 20x10 cm.

Citra digital memiliki batasan pada resolusi maksimumnya, yang bergantung pada perangkat atau sensor yang digunakan untuk menangkap citra tersebut. Batasan ini sangat penting dalam berbagai aplikasi, terutama dalam bidang seperti analisis citra medis atau ilmu pengetahuan, di mana ukuran piksel berperan dalam

menentukan akurasi pengukuran atau analisis. Untuk memastikan akurasi yang memadai, pemilihan ukuran piksel menjadi krusial. Ada tiga metode umum yang sering digunakan untuk mengukur jarak d antara dua titik dengan koordinat (x_1, y_1) dan (x_2, y_2) (G. Li dkk., 2022), yaitu:

$$d_{Ecludiean} = \sqrt{(x2 - x1)^2 + (y2 - y1)^2} \tag{2.3}$$

Jarak Euclidean, seperti dinyatakan dalam rumus (2.3), digunakan untuk menghitung jarak antara dua titik dalam ruang Euclidean. Dalam ruang ini, jarak antara dua titik diukur sebagai garis lurus yang menghubungkan kedua titik tersebut. Rumus Euclidean didasarkan pada teorema Pythagoras dalam geometri, di mana sisi miring segitiga siku-siku merepresentasikan jarak antara dua titik dalam ruang dua dimensi. Teorema Pythagoras juga dapat diperluas ke dimensi yang lebih tinggi, sehingga memungkinkan perhitungan jarak antara dua titik dalam ruang Euclidean berdimensi lebih tinggi. Hal ini menjadikan jarak Euclidean sebagai metode umum dan fleksibel untuk mengukur jarak antar titik dalam berbagai aplikasi, termasuk pengolahan citra dan analisis data multidimensi.

Persamaan Euclidean digunakan secara luas dalam berbagai bidang, termasuk matematika, fisika, ilmu komputer, serta berbagai aplikasi ilmiah dan teknis lainnya. Misalnya, dalam analisis data, persamaan Euclidean digunakan untuk menghitung jarak antara titik-titik data dalam ruang fitur berdimensi-n. Dalam ruang fitur ini, setiap dimensi mewakili satu atribut atau variabel, dan jarak Euclidean memberikan ukuran kesamaan atau perbedaan antara dua data berdasarkan posisi mereka dalam ruang tersebut. Pendekatan ini banyak diaplikasikan dalam teknik seperti clustering, klasifikasi, dan machine learning, di mana jarak antar data mempengaruhi hasil analisis atau model yang dibangun.

$$d_{Manhattan} = |x_2 - x_1| + |y_2 - y_1| \tag{2.4}$$

Persamaan *City-Block* (Persamaan 2.4), yang juga dikenal sebagai jarak Manhattan, adalah metode untuk mengukur jarak antara dua titik dalam ruang dua dimensi dengan menghitung perbedaan absolut antara koordinat x dan y dari kedua

titik tersebut. Dalam metode ini, jarak dihitung dengan menjumlahkan jarak horizontal (sumbu x) dan jarak vertikal (sumbu y) antara dua titik, tanpa memperbolehkan pergerakan secara diagonal, seperti pada metode Euclidean. Jika kedua titik digambarkan pada bidang kartesian, jarak yang dihitung adalah seolah-olah seseorang bergerak melalui jalan-jalan dalam grid, hanya mengikuti arah horizontal dan vertikal. Dengan kata lain, jarak yang ditempuh adalah sepanjang sumbu x dan sumbu y secara bergantian. Metode ini sering digunakan dalam berbagai aplikasi yang melibatkan ruang yang berbentuk grid, seperti pada perencanaan rute di kota atau dalam algoritma pencarian jalur di sistem koordinat ortogonal.

Persamaan *Chess board* (Persamaan 2.5), juga dikenal sebagai jarak *Chebyshev* atau jarak infinity, merupakan metode untuk mengukur jarak antara dua titik dalam ruang dengan mempertimbangkan perbedaan absolut dari setiap koordinat, kemudian mengambil nilai maksimum dari perbedaan tersebut. Jarak *Chebyshev* mengukur jarak berdasarkan jumlah langkah minimum yang diperlukan untuk bergerak dari satu titik ke titik lain pada papan catur, di mana pergerakan dapat dilakukan baik secara horizontal, vertikal, atau diagonal dalam langkah yang sama.

Persamaan ini cocok digunakan dalam konteks di mana pergerakan melibatkan pergeseran ke salah satu arah (sumbu x atau y) dengan langkah terbesar. Secara matematis, jarak *Chebyshev* antara dua titik (x_1, y_1) dan (x_2, y_2) dinyatakan sebagai:

$$d_{Chebyshev} = \max(\quad |\, x_2 - x_1 \,|, |\, y_2 - y_1 \,|\quad) \tag{2.5}$$

Metode ini sering digunakan dalam simulasi permainan seperti catur, di mana pergerakan raja dapat dilakukan dalam satu langkah ke segala arah, termasuk secara diagonal, atau dalam situasi lain yang memerlukan pengukuran langkah minimum di sepanjang arah horizontal maupun vertikal.

2.2.2. Transformasi citra digital

Transformasi intensitas dalam citra digital mengacu pada proses perubahan tingkat kecerahan dan kontras citra untuk meningkatkan kualitas, menyoroti fitur penting, atau mempermudah visualisasi. Proses ini dapat digunakan untuk

memperbaiki citra sehingga informasi yang diinginkan lebih mudah diakses, sementara bagian citra yang tidak relevan tidak perlu diolah. Peningkatan kualitas citra melalui transformasi intensitas dilakukan dengan mengubah nilai intensitas setiap piksel tanpa mengubah posisi piksel tersebut. Transformasi ini dilakukan dengan menerapkan fungsi yang disebut fungsi transformasi skala keabuan atau *Grey Scale Transformation Function* (GST).

Salah satu contoh transformasi intensitas citra adalah konversi citra berwarna (*Red, Green, Blue*) menjadi citra abu-abu (*grayscale*). Nilai abu-abu diperoleh dengan menghitung rata-rata dari elemen warna merah, hijau, dan biru di setiap piksel. Perhitungan ini dinyatakan secara sistematis menggunakan Persamaan 2.6.

$$f_o(x,y) = \frac{f_i^R(x,y) + f_i^G(x,y) + f_i^B(x,y)}{3} \tag{2.6}$$

Konversi citra RGB (*Red, Green, Blue*) ke citra biner melibatkan proses mengubah setiap piksel dalam citra RGB menjadi nilai biner, yang hanya memiliki dua kemungkinan nilai piksel, yaitu hitam atau putih. Proses ini dikenal sebagai binarisasi. Pada proses ini, operasi *threshold* (T) digunakan untuk menentukan ambang batas tertentu yang memisahkan piksel menjadi hitam atau putih, tergantung pada intensitas warnanya. Piksel dengan intensitas di bawah ambang batas T akan diubah menjadi hitam, sedangkan piksel dengan intensitas di atas ambang tersebut akan diubah menjadi putih. Transformasi ini bertujuan untuk menyederhanakan citra dan mempermudah analisis dengan hanya mempertahankan informasi penting dalam bentuk dua tingkat warna, hitam dan putih.

Transformasi citra abu-abu ke citra biner melibatkan proses mengubah setiap piksel bernilai angka dalam citra abu-abu menjadi citra biner dengan dua nilai piksel, yaitu hitam dan putih. Tujuan dari transformasi ini adalah untuk menghasilkan citra biner yang menampilkan objek atau fitur dengan lebih jelas melalui kontras yang tegas antara hitam dan putih. Proses ini dilakukan menggunakan teknik binarisasi yang melibatkan pemilihan ambang batas (*threshold*) tertentu. Ambang ini digunakan untuk memisahkan piksel menjadi

hitam atau putih berdasarkan tingkat keabuan. Secara matematis, transformasi ini dapat dinyatakan sebagai berikut (Persamaan 2.7):

$$f_B\,(x,y) = \begin{cases} 1, \text{jika } f_{\text{gray}}(x,y) > T \\ 0, \text{jika } f_{\text{gray}}(x,y) \leq T \end{cases} \tag{2.7}$$

Penentuan nilai *threshold* pada konversi citra biner merupakan proses yang krusial, karena ambang yang dipilih akan sangat memengaruhi hasil akhir dari citra biner. Nilai *threshold* yang ideal adalah nilai yang mampu memisahkan objek atau fitur yang ingin ditonjolkan dari latar belakang secara efektif. Dengan *threshold* yang tepat, objek dapat tampil dengan jelas, sedangkan area yang tidak penting dapat dihilangkan, sehingga analisis atau pemrosesan citra lebih mudah dilakukan. Pemilihan *threshold* yang kurang tepat dapat menyebabkan hilangnya detail penting atau munculnya *noise* dalam citra biner.

Ada beberapa metode yang dapat digunakan untuk menentukan nilai *threshold* secara otomatis berdasarkan karakteristik citra. Dua metode umum yang sering digunakan adalah:

1. Metode Otsu: Metode ini mencari nilai *threshold* yang meminimalkan varians intra-kelas atau memaksimalkan varians inter-kelas dari histogram citra. Metode Otsu secara otomatis membagi citra menjadi dua kelompok piksel, yaitu objek dan latar belakang, dengan cara mengidentifikasi ambang yang optimal sehingga perbedaan antara kedua kelompok tersebut maksimal.
2. Metode *Mean* (Rata-rata): Metode ini menggunakan rata-rata intensitas piksel pada citra sebagai nilai *threshold*. Piksel yang memiliki intensitas di atas rata-rata dianggap sebagai bagian dari objek, sedangkan piksel dengan intensitas di bawah rata-rata dianggap sebagai latar belakang.
3. Metode *Adaptive Thresholding* adalah teknik yang membagi citra menjadi beberapa blok kecil dan menghitung nilai *threshold* yang berbeda untuk setiap blok berdasarkan intensitas lokal di dalamnya. Metode ini sangat efektif dalam situasi di mana pencahayaan tidak merata di seluruh citra, atau

terdapat variasi keabuan yang signifikan. Dengan menyesuaikan nilai *threshold* untuk setiap area kecil dalam citra, metode ini memungkinkan deteksi fitur yang lebih akurat, terutama dalam kasus di mana metode *thresholding* global tidak efektif. *Adaptive thresholding* sering digunakan dalam aplikasi yang memerlukan pengolahan citra dengan variasi pencahayaan yang kompleks, seperti dalam analisis citra medis atau pengenalan objek pada lingkungan dengan cahaya yang berubah-ubah.

Histogram citra dapat digunakan sebagai alat untuk menentukan nilai ambang batas (*threshold*) dalam proses transformasi citra abu-abu menjadi citra biner. Pada histogram citra, sisi sebelah kiri umumnya mewakili objek yang lebih terang di atas latar belakang yang gelap. Piksel-piksel yang membentuk objek dan latar belakang ini dikelompokkan ke dalam dua mode yang dominan. Dari dua mode tersebut, nilai ambang batas dapat diekstrak untuk membedakan objek dari latar belakang. Pada titik (x, y), jika intensitas piksel $f(x, y)$ lebih besar dari nilai ambang T (yaitu $f(x, y) > T$), maka piksel tersebut diklasifikasikan sebagai titik objek, sedangkan selain itu dikategorikan sebagai titik latar belakang.

Proses klasifikasi menggunakan nilai ambang melibatkan tiga kondisi utama. Pertama, suatu titik (x, y) akan dianggap sebagai bagian dari objek pertama jika intensitas piksel tersebut memenuhi syarat $(T_1 < f(x, y) \leq T_2)$. Kedua, jika intensitas piksel $f(x, y)$ lebih besar dari T_2, maka titik tersebut diklasifikasikan sebagai bagian dari objek lain. Ketiga, jika intensitas piksel $f(x, y)$, kurang dari atau sama dengan T_1, maka titik tersebut dikategorikan sebagai bagian dari latar belakang (Gonzalez & Woods, 2002).

2.2.3. Segmentasi Citra

Segmentasi citra adalah proses membagi atau mempartisi citra digital menjadi beberapa segmen atau bagian yang memiliki karakteristik visual atau sifat-sifat tertentu. Tujuan utama dari segmentasi citra adalah untuk menyederhanakan atau merepresentasikan citra dalam bentuk yang lebih mudah dipahami atau diinterpretasi. Dengan demikian, segmentasi citra mempermudah dalam

mengidentifikasi dan memisahkan objek atau area tertentu dalam citra, sehingga analisis lebih terfokus pada bagian-bagian penting dari citra tersebut. Beberapa tujuan umum dari segmentasi citra meliputi pemisahan objek dari latar belakang, identifikasi area tertentu yang memiliki fitur serupa, dan pengelompokan piksel berdasarkan karakteristik seperti warna, tekstur, atau intensitas. Beberapa tujuan umum segmentasi citra meliputi:

1. Pengenalan Objek: Memisahkan dan mengidentifikasi objek atau area tertentu dalam citra, seperti mendeteksi wajah manusia, kendaraan, atau objek lain.
2. Analisis Medis: Dalam bidang medis, segmentasi digunakan untuk mendeteksi dan memisahkan struktur anatomi tertentu dalam gambar medis, seperti organ atau jaringan.
3. Pengolahan Citra untuk *Computer Vision*: Segmentasi adalah langkah penting dalam pemrosesan citra untuk aplikasi *computer vision*, membantu dalam ekstraksi fitur dan pengenalan pola.
4. Klasifikasi dan Pengenalan: Segmentasi dapat digunakan sebagai langkah awal dalam proses klasifikasi dan pengenalan objek pada tingkat yang lebih tinggi.

Ada berbagai metode untuk melakukan segmentasi citra, seperti metode berbasis warna, tekstur, intensitas piksel, serta metode berbasis objek. Pemilihan metode tergantung pada karakteristik citra dan tujuan segmentasi. Dua kategori algoritma yang umum digunakan dalam segmentasi citra adalah berbasis diskontinuitas dan berbasis similaritas (Gonzalez & Woods, 2002).

Deteksi tepi adalah proses untuk mengidentifikasi batas antara dua objek yang memiliki perbedaan warna atau intensitas yang signifikan dalam citra digital, sehingga terbentuk garis atau batas antara objek dan latar belakang. Pada pengolahan citra, tepi objek ditandai dengan perubahan nilai keabuan yang jelas antara satu piksel dengan piksel di sebelahnya. Operasi deteksi tepi bertujuan menemukan perubahan intensitas yang nyata dalam citra, dan dilakukan melalui operator deteksi tepi yang bekerja dengan memodifikasi nilai keabuan piksel berdasarkan nilai dari piksel-piksel di sekitarnya. Terdapat dua jenis utama operator

deteksi tepi, yaitu operator berbasis gradien (turunan pertama), seperti Sobel, Prewitt, dan Roberts, serta operator berbasis turunan kedua, seperti *Laplacian of Gaussian* (LoG). Deteksi tepi dilakukan dengan menghitung selisih intensitas antara dua piksel bertetangga, sehingga menghasilkan nilai gradien citra. Persamaan gradien turunan pertama ini didefinisikan sebagai vektor yang menunjukkan perubahan intensitas di setiap titik dalam citra (Gonzalez & Woods, 2002).

$$G[f(x,y)] = [G_x, G_y] = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right] \tag{2.8}$$

Gradien memiliki dua sifat penting, yaitu:

1. Vektor $G[f(x,y)]$ menunjukkan arah di mana fungsi $f(x,y)$ mengalami penambahan laju maksimum.
2. Besar gradien merepresentasikan nilai penambahan laju maksimum dari fungsi $f(x,y)$ per satuan jarak dalam arah gradien G.

Besar gradien dapat dihitung menggunakan Persamaan 2.9 sebagai berikut:

$$\mathrm{G}[f(x,y)] = \sqrt{G_x^2 + \mathrm{G}_y^2} \tag{2.9}$$

Dalam pengolahan citra, untuk penyederhanaan, besar gradien sering kali dihitung dengan menggunakan pendekatan Persamaan 2.10:

$$\mathrm{G}[f(x,y)] = |\mathrm{G}_x| + |\mathrm{G}_y| \tag{2.10}$$

Di sini:

- $G[f(x,y)]$ menunjukkan arah penambahan laju maksimum dari fungsi $f(x,y)$,
- G_x dan G_y adalah turunan parsial fungsi $f(x,y)$ terhadap x dan y, yang masing-masing menggambarkan perubahan intensitas piksel dalam arah horizontal dan vertikal.

Gradien G_x merepresentasikan besarnya gradien yang sama dengan penambahan laju maksimum dari fungsi $f(x,y)$ per satuan jarak dalam arah gradien G. Salah satu metode segmentasi citra yang populer adalah kontur aktif (Kass dkk., 1980). Kontur aktif, atau sering disebut *snakes*, menggunakan model kurva tertutup yang bergerak secara melebar atau menyempit mengikuti objek yang ingin disegmentasi. Kurva ini bergerak dengan cara meminimalkan energi total yang terdiri dari energi

internal dan energi eksternal. Energi eksternal dipengaruhi oleh karakteristik objek, seperti garis atau tepi objek, yang mengarahkan kurva untuk mengikuti batas-batas objek.

Energi eksternal ini diformulasikan dalam Persamaan 2.11 sebagai:

$$E = \int_0^1 \mathrm{E}_{int}\big(Y(\mathrm{s})\big)ds + \int_0^1 \mathrm{E}_{ext}\big(Y(\mathrm{s})\big)ds \tag{2.11}$$

Di mana:

- E adalah energi yang mempengaruhi pergerakan kontur aktif,
- E_{int} adalah energi internal yang dipengaruhi oleh bentuk objek,
- E_{ext} adalah energi eksternal yang mengarahkan kurva melebar atau menyempit menuju objek,
- $Y(s)$ adalah kurva dalam ruang dua dimensi.

Energi internal, yang mengontrol kelancaran kurva, dihitung menggunakan Persamaan 2.12:

$$E_{\mathrm{int}} = \frac{\alpha(s)|\boldsymbol{Y_s}(s)|^2 + \beta(s)|\boldsymbol{Y_{ss}}(s)|^2}{2} \tag{2.12}$$

Di mana:

- $\alpha(s)$ dan $\beta(s)$ adalah fungsi yang mengontrol bagaimana kurva bergerak dalam ruang.

Sedangkan energi eksternal, yang mendorong kurva menuju fitur citra seperti tepi, dinyatakan dengan Persamaan 2.13:

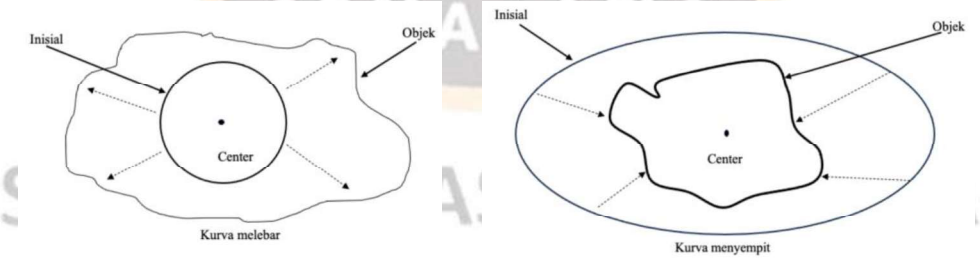
$$E_{\mathrm{ext}} = \left|\nabla G\big(\boldsymbol{Y}(s)\big)\right|^2 \tag{2.13}$$

Dalam hal ini, G adalah citra yang akan disegmentasi, dan ∇G adalah gradien dari citra tersebut. Segmentasi kontur aktif bekerja dengan menghubungkan titik-titik pada kurva secara berurutan, yang kemudian dikendalikan oleh garis lurus. Penentuan titik-titik awal (kontur inisial) harus cukup mendekati objek sehingga kurva dapat mendekati atau menyentuh tepi objek secara melebar atau menyempit sesuai pengaruh energi eksternal hingga mencapai garis tepi objek.

Perumpamaan yang diberikan menyiratkan bahwa fungsi $\alpha(s)$ dan $\beta(s)$ berperan dalam mengontrol pergerakan kurva dalam dua mode yang berbeda:

1. Pergerakan seperti membran: Dalam mode ini, kurva akan cenderung untuk bergerak dengan cara meregang atau merapatkan diri pada dimensi tertentu, menyerupai sifat fisik dari sebuah membran. Ketika $\alpha(s)$ mendominasi, kurva lebih sensitif terhadap perbedaan panjang di antara titik-titik, sehingga menghasilkan pergerakan yang lebih fleksibel atau elastis. Akibatnya, kurva akan lebih mudah berubah bentuk dan mengikuti kontur yang lebih sederhana.
2. Pergerakan seperti plat yang tipis: Dalam mode ini, kurva cenderung bergerak dengan cara menjaga kelenturan dan kelengkungannya, seperti sebuah plat tipis yang cenderung mempertahankan bentuk datarnya. Jika $\beta(s)$ mendominasi, kurva akan lebih stabil dan tahan terhadap perubahan bentuk yang tajam. Hal ini berarti kurva akan lebih kaku dan cenderung mengikuti kontur yang lebih halus dan kompleks.

Kedua parameter, $\alpha(s)$ dan $\beta(s)$, berfungsi sebagai bagian dari model matematis yang menggambarkan bagaimana kurva bereaksi terhadap gaya eksternal atau kondisi lingkungan tertentu. Dalam konteks matematika atau fisika, perumpamaan ini sering digunakan untuk mendeskripsikan perilaku elastis atau deformasi suatu benda, di mana kurva digunakan untuk merepresentasikan bentuk benda tersebut dalam ruang. Model ini mirip dengan konsep elastisitas, di mana kurva dapat meregang atau menyempit sesuai dengan pengaruh gaya eksternal.



(a) (b)

Gambar 2.3 Ilustrasi segmentasi kontur

Pada Gambar 2.3, diperlihatkan ilustrasi segmentasi kontur aktif dengan kurva inisial yang bergerak melebar atau menyempit mengikuti kondisi objek dalam citra. Gambar 2.3a menunjukkan situasi di mana kurva inisial melebar karena objek yang lebih besar dibandingkan dengan kurva inisialnya, sehingga kurva harus menyesuaikan untuk mencakup seluruh objek. Sebaliknya, pada Gambar 2.3b, kurva menyempit karena objek lebih kecil daripada kurva inisial, sehingga kurva menyesuaikan dengan mengecil agar sesuai dengan batas objek tersebut. Proses ini mencerminkan bagaimana kurva mengikuti bentuk dan ukuran objek melalui pengaruh parameter-parameter yang ada.

Langkah-langkah dalam segmentasi kontur aktif dapat dilakukan sebagai berikut:

1. Inisialisasi. Langkah pertama adalah menempatkan kurva awal di dekat objek yang akan disegmentasi. Proses inisialisasi ini bisa dilakukan secara manual atau otomatis, tergantung pada metode yang digunakan.
2. Evolusi Kontur. Setelah kurva awal diinisialisasi, kontur mulai bergerak berdasarkan energi internal dan eksternal. Pergerakan ini terjadi melalui serangkaian iterasi, di mana posisi titik kontrol pada kurva disesuaikan secara bertahap, mengikuti bentuk objek yang diinginkan.
3. Konvergensi. Proses evolusi kontur berlangsung hingga kurva konvergen ke batas objek. Konvergensi terjadi ketika kurva mencapai batas objek atau ketika energi total mencapai nilai minimal yang stabil, menandakan bahwa kurva telah menemukan posisi optimal.
4. Hasil Segmentasi. Hasil dari algoritma segmentasi kontur aktif adalah kurva yang menggambarkan batas objek dalam citra. Informasi ini kemudian dapat digunakan untuk memisahkan objek dari latar belakang secara otomatis, memungkinkan analisis citra yang lebih fokus dan akurat.

Segmentasi unggul dibandingkan deteksi objek dalam kondisi pencahayaan rendah karena mampu mengenali objek secara lebih detail. Penenlitian Cho et al. menunjukkan bahwa segmentasi semantik tetap efektif pada gambar malam hari dengan membedakan objek dari latar belakang meski dalam kondisi gelap (Cho dkk., 2020). Dalam pengenalan lampu lalu lintas, segmentasi meningkatkan akurasi

deteksi hingga lebih dari 12% dibandingkan metode deteksi objek biasa (Masaki dkk., 2021). Pendekatan berbasis piksel ini memungkinkan analisis yang lebih mendalam pada pencahayaan minim.

2.2.4. Video

Video adalah representasi dinamis dari gambar statis. Sebuah video alternatif dapat didefinisikan sebagai urutan gambar statis. Setiap gambar disebut sebagai bingkai. Bingkai akan terus diproyeksikan dengan kecepatan tetap ke layar, menciptakan efek dinamis yang mengubah jumlah kehadiran visual. Ada dua jenis video: analog dan digital. Karena video adalah rangkaian gambar statis, dapat memprosesnya serupa dengan cara menangani foto. Gambar adalah kumpulan piksel dalam beberapa ruang warna. Piksel adalah nama yang diberikan untuk sampel ini. Piksel adalah komponen terkecil dari sebuah gambar (Salim, 2020). Dimana sebuah video dapat direpresentasikan dengan $N \times C \times H \times W$. N merepresentasikan jumlah frame per detik. C merepresantisakan Channel, H dan W mewakili panjang dan lebar dari sebuah frame.

2.2.5. *Object Detection*

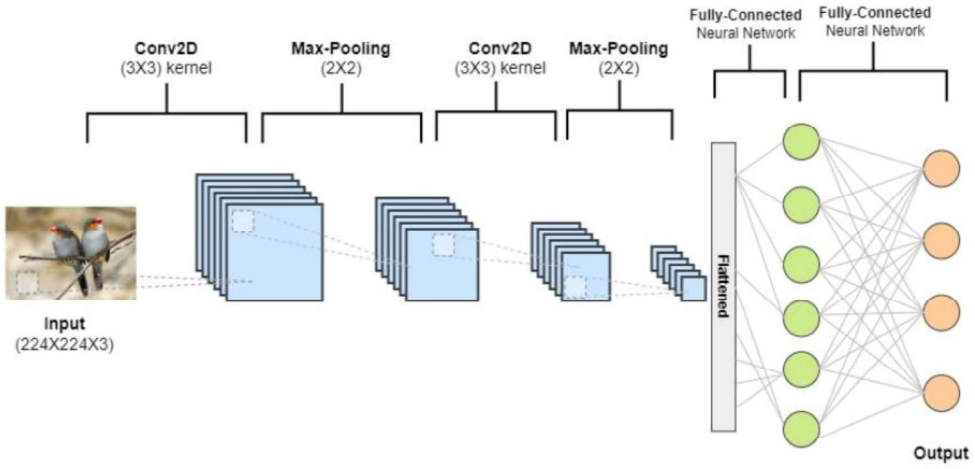
Deteksi item mengidentifikasi keberadaan, ruang lingkup, dan penempatan objek dalam gambar. Deteksi objek mendeteksi kelas objek yang ada dalam database yang dipelajari. Identifikasi item adalah langkah pertama dalam deteksi objek. Ini mungkin dianggap sebagai mengenali dua kelas objek, satu mewakili kelas benda dan yang lainnya mewakili kelas non-objek. Deteksi lunak dan deteksi keras adalah dua jenis deteksi objek. Deteksi lunak dapat dengan mudah mengidentifikasi keberadaan suatu objek, tetapi deteksi keras dapat mendeteksi keberadaan dan penempatan objek dalam suatu gambar (Jalled & Voronkov, 2016).

Identifikasi item sering dilakukan dengan mencari setiap bagian dari gambar untuk menentukan bagian yang cocok secara fotometrik atau geometris dengan objek target dalam database pelatihan. Ini dilakukan dengan memindai *template* item di seluruh gambar penuh pada berbagai posisi, skala, dan rotasi, dengan deteksi yang terjadi jika kemiripan antara *template* dan gambar cukup tinggi. Korelasi

antara *template* dan wilayah gambar dapat digunakan untuk menentukan kemiripannya (Jalled & Voronkov, 2016).

Cara pendeteksian objek pada video sama dengan teknik pendeteksian objek pada foto. Video terdiri dari beberapa gambar atau bingkai. Itu dapat dibagi menjadi banyak bingkai dalam satu film, dengan setiap bingkai melakukan prosedur deteksi objek. Bingkai yang diproses kemudian dipasang kembali untuk membuat video lengkap. *Graphical Processing Unit* (GPU) dapat digunakan untuk memproses deteksi objek dalam frame film (Salim, 2020).

2.2.6. *Convolution Neural Network* (CNN)



Gambar 2.4 Contoh arsitektur CNN (Saha, n.d.)

CNN, subbidang penting dari pembelajaran mendalam (Wang Chen; Liu, Bing; Zhou, Xiaoqing; Zhang, Liang; Zheng, Jin; Bai, Xiao, 2021), sering diterapkan pada pemrosesan gambar dan video. CNN berbeda dari jaringan saraf konvensional karena tahap pembelajaran fiturnya terdiri dari satu atau lebih lapisan konvolusional dan penggabungan. Setiap neuron di lapisan konvolusional tidak terhubung ke setiap neuron di lapisan berikutnya, sehingga mengurangi beban komputasi dalam melatih jaringan saraf konvensional yang terhubung penuh.

$$h_j^{(n)} = \sum_{k=1}^{K} h_k^{n-1} \otimes W_{kj}^{(n)} + b_{kj}^{(n)} \qquad (2.14)$$

Dimana $\otimes$ adalah konvolusi 2 dimensi. Keluaran peta fitur ke-j pada lapisan tersembunyi ke-n diwakili oleh $h_j^{(n)}$. h_k^{n-1} adalah saluran ke-k di lapisan tersembunyi ke-$(n-1)$. $w_{kj}^{(n)}$ menunjukkan bobot saluran ke-k dalam filter ke-j pada lapisan ke-n. $b_{kj}^{(n)}$ adalah istilah bias terkait. Lapisan berikutnya menghasilkan kernel konvolusi sebanyak peta fitur. Semakin besar nilai peta fitur yang diperoleh setelah konvolusi, semakin besar kemiripan antara gambar di wilayah tersebut dan fitur kernel.

MobileNetV2 adalah arsitektur jaringan saraf konvolusional (CNN) yang dirancang khusus untuk tugas-tugas visi komputer pada perangkat mobile dengan sumber daya terbatas. Arsitektur ini dikembangkan oleh Google dan dirilis sebagai bagian dari MobileNetV2 (Sandler dkk., 2018). Berikut merupakan bagian penting yang ditawarkan pada MobileNetV2 :

a. *Depthwise Separable Convolution*: MobileNetV2 menggunakan konsep "*depthwise separable convolution*" untuk mengurangi jumlah parameter dan komputasi. Pada konvolusi biasa, setiap filter berinteraksi dengan seluruh kedalaman input. Namun, dalam *depthwise separable convolution*, konvolusi dibagi menjadi dua langkah terpisah: *depthwise convolution* (beroperasi pada setiap saluran secara terpisah) dan *pointwise convolution* (konvolusi 1x1 untuk menggabungkan saluran tersebut). Pendekatan ini membantu mengurangi beban komputasi dengan mempertahankan representasi fitur yang penting.

   Operasi konvolusi standar mengambil tensor input L_i dengan dimensi $h_i \times w_i \times d_i$(tinggi, lebar dan kedalaman) dan mereapkan kernel konvolusi $K \in R^{k \times k \times d_i \times d_j}$ untuk menghasilkan tensor output L_j dengan dimensi $h_i \times w_i \times d_j$. Biaya komputasi dari lapisan hampir sama dengan konvolusi standar, dengan besar biaya :

$$h_i \cdot w_i \cdot d_i\left(k^2 + d_j\right) \tag{2.15}$$

   dimana :

- $h_i \times w_i$ mewakili dimensi spasial dari tensor input.
- d_i adalah jumlah saluran input atau kedalaman dari tensor input.
- d_j adalah jumlah saluran output atau kedalaman dari tensor output.
- $k \times k$ adalah ukuran kernel konvolusi

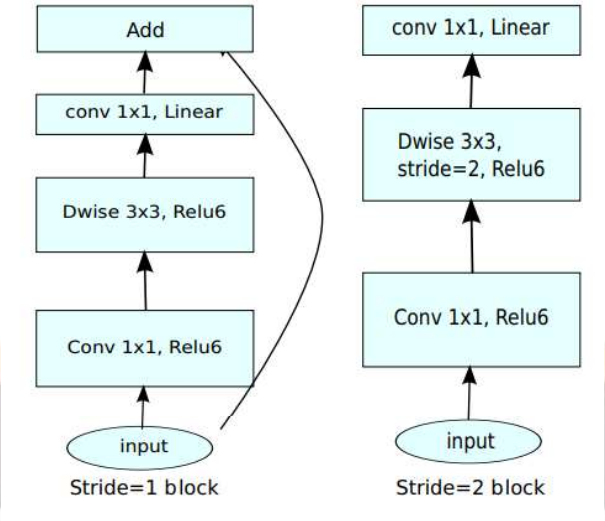
MobilNetV2 menggunakan $k = 3$ (3×3 *depthwise separable convolutions*) jadi biaya komputasi akan menjadi 8 sampai 9 kali lebih kecil dari konvolusi standar dengan sedikit mengurangi akurasi (Andreetto dkk., 2017).

b. *Linear Bottleneck:* MobileNetV2 menggunakan struktur blok linear bottleneck, yang mencakup tiga lapisan konvolusi: expansi (lapisan 1x1 untuk menambah dimensi fitur), depthwise (konvolusi depthwise untuk menangkap informasi spasial), dan projection (konvolusi 1x1 untuk mengurangi dimensi kembali). Blok ini membantu dalam mengurangi dimensi fitur sebelumnya tanpa kehilangan informasi penting.
c. *Inverted Residuals with Linear Bottlenecks:* Konsep "inverted residuals" digunakan untuk membuat MobileNetV2 lebih efisien. Ini melibatkan penggunaan shortcut connections dengan bottleneck linear, yang membantu dalam propagasi gradien dan membantu menjaga informasi fitur yang penting selama pelatihan.
d. *Squeeze-and-Excitation (SE) Blocks:* MobileNetV2 memasukkan blok "*squeeze-and-excitation*" untuk menyesuaikan bobot fitur. Blok ini membantu model untuk secara adaptif menekankan fitur yang lebih penting dan mengurangi pengaruh fitur yang kurang relevan.
e. *Global Depthwise Separable Convolution:* Untuk meningkatkan efisiensi, MobileNetV2 menggunakan konvolusi depthwise separable global pada lapisan terakhir. Hal ini membantu dalam mengurangi dimensi fitur secara global sebelum masuk ke lapisan fully connected.

2.2.7. Vision Transformer

MobileViTV2 adalah variasi dari model Vision Transformer (ViT), yang dirancang khusus untuk perangkat *online* dan *edge devices*. Ini menyertakan

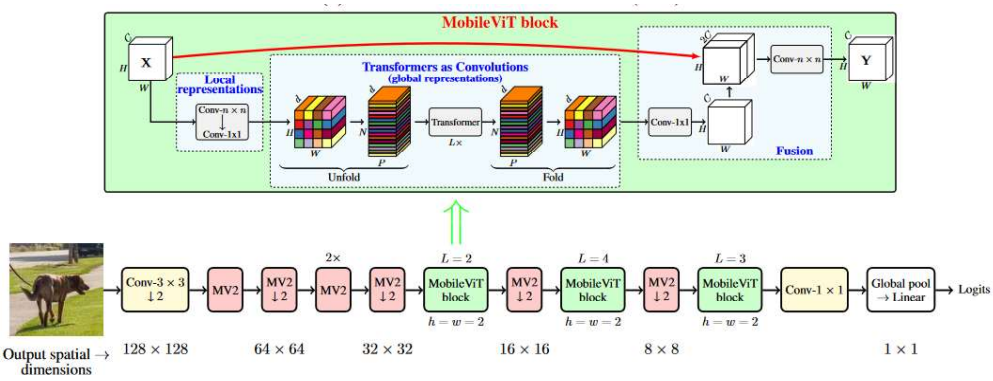
rangkaian konvolusional, khususnya BiT (Big Transfer), untuk mengekstrak fitur yang berfungsi sebagai "token" awal bagi Transformer.



Gambar 2.5 Arsitektur MobileNetV2

MobileViT menerima gambar sebagai masukan. Ini terdiri dari dua blok utama: ekstraktor fitur dan detektor. Saat gambar baru masuk, ekstraktor fitur menggunakannya sebagai input untuk mengekstrak fitur yang disematkan pada skala yang berbeda. Kemudian, fitur ini dimasukkan ke dalam dua cabang detektor untuk mendapatkan kotak pembatas dan informasi kelas. Ekstraktor fitur secara hierarki mengekstraksi fitur dari piksel yang berasal dari lapisan input. Ini menggunakan filter 3×3 yang melewati seluruh struktur dan penyatuan lapisan secara maksimal untuk mengurangi dimensi input. Detektor menggunakan struktur konvolusional 1×1 untuk menganalisis hasil yang dihasilkan untuk memprediksi posisi dan kelas objek yang terdeteksi pada citra masukan. Diberi gambar untuk model ini, hasil akhirnya adalah daftar kotak pembatas bersama dengan informasi kelas yang dikenali. Pada awalnya, dimensi gambar masukan akan dikurangi dengan menggunakan stride untuk mencari fitur. Kemudian, fitur-fitur tersebut diekstraksi dengan melalui beberapa lapisan konvolusional, yang membuat klasifikasi deteksi. Terakhir, output diberikan sebagai peta fitur, yang merepresentasikan prediksi kelas jaringan. Keluaran ini diubah menjadi kotak pembatas dengan ID kelas. Selama pelatihan, model asli MobileViT menggunakan

fungsi *lost* yang sama dengan yang digunakan oleh MobileViT, yang terdiri dari empat parameter: (1) posisi bingkai prediksi (x, y); (2) ukuran bingkai prediksi (w, h); (3) klasifikasi prediksi; (4) keyakinan prediksi.



Gambar 2.6 Arsitektur asli Mobile ViT

Model ini dirancang dengan fokus pada perangkat *online* dan *edge*, menunjukkan penekanan pada efisiensi dan kecocokan untuk platform dengan sumber daya komputasi yang terbatas. Penggunaan rangkaian konvolusional, terinspirasi oleh arsitektur seperti BiT, menunjukkan upaya untuk mencapai keseimbangan antara kelebihan arsitektur konvolusional dan berbasis transformer. Dengan fokus model ini pada perangkat *online* dan edge, MobileViTV2 memungkinkan cocok untuk berbagai aplikasi visi komputer dalam situasi di mana sumber daya komputasi terbatas, seperti pada *smartphone* atau perangkat *edge*.

Berikut merupakan langkah-langkah kunci dari MobileViT :

a. Lapisan Konvolusi
   - Tensor input $X \in \mathbb{R}^{H \times W \times C}$ melewati lapisan konvolusi standar $N \times N$, yang menangkap informasi spasial lokal.
   - Lapisan konvolusi satu titik (1×1) kemudian diterapkan untuk menghasilkan $X_L \in \mathbb{R}^{H \times W \times d}$, di mana $d > C$. Langkah ini memproyeksikan tensor ke ruang dimensi yang lebih tinggi.

b. Ekstraksi dan *Flattening Patch*

   X_L diperluas menjadi N patch yang tidak tumpang tindih dan terpipih $X_U \in$

$\mathbb{R}^{P\times N\times d}$, dimana $P = wh$ adalah jumlah total patch, $N = \frac{HW}{P}$, dan h dan w adalah tinggi dan lebar masing-masing patch.

c. Transformer dan *Flattening patch*

Transformer selanjutnya diterapkan pada X_U untuk mengkodekan hubungan antar patch. Setiap patch p diproses melalui $X_G(p) = Transformer\left(X_U(p)\right), 1 \leq p \leq P$

d. Operasi fold, $X_G \in \mathbb{R}^{P\times N\times d}$ kemudian dilipat untuk mendapatkan $X_F \in \mathbb{R}^{H\times W\times d}$

e. Proyeksi dan *Concatenation*

X_F diproyeksikan ke dalam ruang dimensi yang lebih rendah (C dimensi) oleh konvolusi titik demi titik.

Output akhir X_F digabungkan dengan input asli X menggunakan operasi penggabungan.

f. Pemeliharaan urutan spatial dan *patch*

Berbeda dengan struktur umum ViT, MobileViT mempertahankan baik urutan patch maupun urutan spasial piksel dalam setiap *patch* selama langkah kedua, menjaga informasi spasial.

g. Pengkodean Informasi Lokal dan Global

X_U mengkodekan informasi lokal, menggabungkan informasi piksel dalam setiap patch.

X_G mengkodekan hubungan antar *patch*, menangkap informasi global melalui operasi transformer.

h. Desain Ringan

Model ini dirancang ringan karena operasi dot-product dipilih secara selektif berdasarkan posisi piksel, memungkinkan pengkodean informasi global yang efisien.

2.2.8. Indeks Efisiensi Navigasi

Indeks Efisiensi navigasi (IEN) (Ganz dkk., 2012) digunakan untuk mengevaluasi kinerja navigasi dari hasil eksekusi sistem. IEN didefinisikan sebagai

rasio jarak yang sebenarnya ditempuh oleh subjek dibandingkan dengan jarak optimal dari suatu sumber ke tujuan tertentu. Jalur optimal mewakili rute paling efisien antara titik awal dan akhir. Untuk menghitung IEN, digunakan variabel jarak yang sebenarnya ditempuh oleh subjek dan membaginya dengan jarak optimal. Formula IEN sebagai berikut:

$$IEN = \frac{1}{N} \sum_{i=1}^{N} \frac{L_A(S_i)}{L_O(S_i)} \tag{2.16}$$

di mana:

- N adalah jumlah sub-jalur,
- S_i mewakili setiap sub-jalur,
- L_A adalah panjang sebenarnya yang ditempuh di sub-jalur S_i,
- L_O adalah panjang optimal dari sub-jalur S_i.

IEN berfungsi sebagai ukuran kuantitatif untuk menilai seberapa efisien suatu sistem navigasi membimbing subjek dari satu titik ke titik lainnya dengan membandingkan jalur sebenarnya dengan jalur optimal. IEN yang lebih tinggi menunjukkan efisiensi navigasi yang lebih baik, karena menandakan bahwa jalur subjek secara dekat sejalan dengan jalur optimal.

2.2.9. Klasifikasi

Deteksi objek merupakan salah satu topik penting dalam pengolahan citra digital dan visi komputer. Namun, proses deteksi objek dapat menjadi lebih kompleks ketika dihadapkan pada kondisi yang kurang ideal, seperti pencahayaan yang buruk, sudut pengambilan gambar yang tidak optimal, serta adanya noise (Xie et al., 2021; , Hameed et al., 2022). Dalam situasi demikian, teknik segmentasi citra dapat menjadi pilihan yang tepat untuk meningkatkan akurasi dan keandalan proses deteksi objek (Jalilian et al., 2021). Tugas klasifikasi dalam pembelajaran mesin bertujuan untuk mengembangkan algoritma yang dapat secara otomatis mengenali pola dan hubungan dalam data untuk mengelompokkan atau memberi label pada kejadian ke dalam kategori yang telah ditentukan. Proses pelatihan model dilakukan

dengan menggunakan data yang sudah dilabeli, di mana algoritma belajar untuk menghubungkan ciri-ciri input dengan kelas output yang sesuai. Umumnya, teknik ini melibatkan pembelajaran yang diawasi, di mana model belajar dari contoh-contoh sebelumnya untuk membuat prediksi pada data baru yang belum pernah dilihat sebelumnya. Ada beragam algoritma klasifikasi yang dapat digunakan, mulai dari metode konvensional seperti *decision tree*, *support vector machine*, dan regresi logistik, hingga teknik yang lebih modern seperti jaringan syaraf dan metode ansambel seperti *random forest*.

Proses klasifikasi berfokus pada penyesuaian parameter model agar mengurangi kesenjangan antara label kelas yang diprediksi dan label kelas yang sebenarnya. Konsep generalisasi menjadi kunci dalam teori klasifikasi, di mana model yang terlatih harus dapat menghasilkan hasil yang baik pada data baru yang belum pernah dilihat sebelumnya, menunjukkan kemampuannya untuk mengidentifikasi pola di luar dataset pelatihan. Metrik evaluasi seperti akurasi, presisi, recall, dan skor F1 digunakan untuk memberikan penilaian kuantitatif terhadap kinerja algoritma klasifikasi. Fondasi teoritis dalam pembelajaran mesin menyediakan kerangka kerja untuk menangani berbagai masalah dunia nyata, termasuk di antaranya diagnosis medis, pengenalan gambar, filter spam, dan deteksi penipuan.

Pengukuran hasil performa dari deteksi objek menggunakan analisis *confusion matrix* merupakan metode yang efektif karena mampu memberikan penilaian komprehensif terhadap kinerja model. *Confusion matrix* mencakup empat elemen utama yang digunakan untuk mengevaluasi hasil prediksi:

1. *True Positive* (TP): Objek yang terdeteksi secara benar sebagai objek. Ini adalah kasus di mana model berhasil mendeteksi objek yang ada dengan benar.
2. *True Negative* (TN): Latar belakang yang diidentifikasi secara benar sebagai bukan objek. Ini adalah kondisi ketika model dengan tepat mengabaikan area yang bukan merupakan objek.
3. *False Positive* (FP): Latar belakang yang salah diidentifikasi sebagai objek.

Ini menunjukkan bahwa model salah mendeteksi sesuatu yang bukan objek sebagai objek.

4. *False Negative* (FN): Objek yang tidak terdeteksi atau diidentifikasi sebagai latar belakang. Ini adalah kondisi di mana model gagal mendeteksi objek yang seharusnya dikenali.

Berikut adalah rumus untuk metrik evaluasi akurasi, presisi, recall, dan skor F1.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.17}$$

Akurasi adalah pengukuran performa model klasifikasi yang menunjukkan seberapa baik model mengklasifikasikan data dengan benar. Pengukuran ini sering digunakan dalam analisis menggunakan confusion matrix untuk menilai seberapa tepat prediksi model secara keseluruhan (persamaan 2.17).

$$Recall = \frac{TP}{TP + FN} \tag{2.18}$$

Recall adalah ukuran yang menunjukkan seberapa handal model dalam mendeteksi data yang berlabel "positif" dengan benar. Recall dihitung sebagai proporsi dari jumlah data yang terdeteksi sebagai "positif" terhadap keseluruhan data yang memang berlabel "positif". Ini memberikan gambaran tentang sensitivitas model dalam mendeteksi data positif (Persamaan 2.18). Recall sangat penting dalam situasi di mana prioritasnya adalah meminimalkan kesalahan dalam mengabaikan data positif.

$$Precission = \frac{TP}{TP + FP} \tag{2.19}$$

Precision adalah ukuran yang menunjukkan seberapa akurat model dalam memprediksi data berlabel "positif". Dengan kata lain, precision mengukur proporsi prediksi positif yang benar-benar positif dari semua prediksi yang dianggap positif oleh model. *Precission* penting untuk memahami seberapa sering prediksi positif model tepat sasaran, terutama ketika kesalahan prediksi positif (false positives) perlu diminimalkan.

$$F1 - score = \frac{2TP}{(2TP + FP + FN)} \quad (2.20)$$

F1-Score adalah kombinasi antara recall dan presisi yang digabungkan ke dalam satu metrik tunggal secara harmonik. *F1-Score* memberikan keseimbangan antara kedua metrik tersebut, terutama ketika terdapat trade-off antara presisi dan recall. Nilai *F1-Score* dihitung menggunakan rata-rata harmonik dari presisi dan recall, yang memberikan gambaran lebih menyeluruh mengenai kinerja model, terutama dalam situasi di mana penting untuk menyeimbangkan antara prediksi positif yang benar dan sensitivitas deteksi. Persamaan 2.20 digunakan untuk menghitung nilai rata-rata presisi dan recall dalam *F1-Score*.

Loss Function digunakan untuk merangkum performa model deteksi objek dengan tujuan memahami seberapa baik hasil akhir model. Model menunjukkan performa yang baik jika nilai *loss* semakin menurun selama proses *training*, yang menandakan bahwa model semakin akurat dalam melakukan prediksi.

Salah satu metode evaluasi yang umum digunakan dalam algoritma deteksi objek adalah *Intersection over Union* (IoU). IoU adalah metrik yang digunakan untuk mengukur tingkat tumpang tindih antara dua objek atau area, yaitu antara prediksi model dan *ground truth* (hasil yang diharapkan). IoU sering digunakan dalam bidang *computer vision* dan pemrosesan citra digital untuk mengevaluasi sejauh mana hasil deteksi objek sesuai dengan target, terutama dalam tugas seperti segmentasi objek dan deteksi objek. Gambar 2.5 menunjukkan ilustrasi evaluasi *Intersection Over Union* (IoU), di mana nilai IoU yang lebih tinggi menunjukkan deteksi yang lebih akurat.

(a) (b)

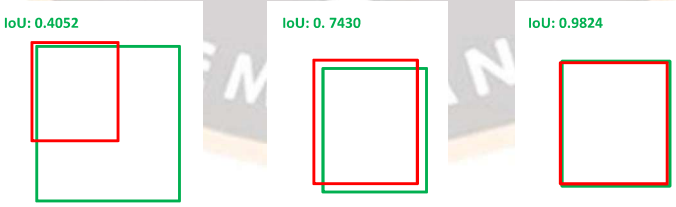
Gambar 2. 7 (a) *Area of intersection* (b) *Area of union*

IoU dihitung sebagai rasio antara luas perpotongan (*intersection*) dari kedua wilayah tersebut dengan luas gabungan (*union*) dari kedua wilayah. Formula untuk menghitung IoU dapat dinyatakan sebagai berikut (Persamaan 2.21, (Hidayatullah, 2021):

$$IoU = \frac{|\text{Luas Perpotongan}|}{|\text{Luas Gabungan}|} \tag{2.21}$$

Nilai IoU yang lebih tinggi menunjukkan tumpang tindih yang lebih besar antara wilayah referensi dan hasil deteksi, menandakan performa deteksi objek yang lebih baik.

IoU memberikan nilai dalam rentang 0 hingga 1, di mana 0 berarti tidak ada tumpang tindih antara wilayah referensi dan hasil deteksi, sementara 1 menunjukkan bahwa wilayah tersebut bertumpang tindih sepenuhnya. Semakin tinggi nilai IoU, semakin akurat deteksi atau segmentasi objek tersebut. Nilai ambang batas IoU yang digunakan untuk menentukan apakah hasil deteksi dianggap benar atau cocok dengan referensi dapat bervariasi tergantung pada aplikasi dan kebutuhan spesifik. Gambar 2.6 menunjukkan klasifikasi hasil evaluasi kinerja dengan menggunakan teknik IoU. Evaluasi ini membagi hasil menjadi tiga kelompok, yaitu *poor, good,* dan *excellent*, berdasarkan nilai IoU (Matei dkk., 2022). Klasifikasi ini membantu dalam menilai kualitas deteksi objek, di mana nilai IoU yang lebih tinggi mencerminkan hasil yang lebih baik.



Poor *Good* *Excellent*

Gambar 2. 8 Hasil evaluasi menggunakan IoU

2.2.10. *Mobile Indoor Navigation*

Mobile Indoor Navigation adalah teknologi yang terus berkembang, memungkinkan pengguna perangkat seluler untuk menavigasi di dalam ruangan

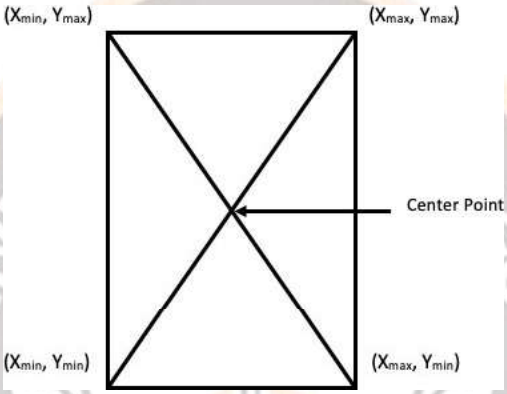
dengan memanfaatkan kamera dan algoritma penglihatan komputer. Teknologi ini menggunakan metode seperti *Simultaneous Localization and Mapping* (SLAM) untuk memetakan dan melacak posisi pengguna secara *real-time*, serta *Visual-Inertial Odometry* (VIO) yang menggabungkan data visual dengan sensor inersial untuk melacak pergerakan dengan lebih akurat (Pooja dkk., 2020; Usenko dkk., 2020). Selain itu, kemajuan dalam *deep learning* dan *artificial intelligence* (AI) telah memungkinkan pengenalan objek yang lebih baik, membantu aplikasi navigasi mengenali elemen-elemen di dalam ruangan, seperti pintu atau tanda. *Augmented Reality* (AR) juga memperkaya pengalaman navigasi dengan menampilkan petunjuk arah langsung di layar ponsel, menjadikan navigasi lebih interaktif dan intuitif. Penggunaan *cloud computing* dan *edge computing* membantu meningkatkan kecepatan dan akurasi pemrosesan data, sementara kombinasi dengan teknologi lain, seperti *Bluetooth Low Energy* (BLE) beacons atau Wi-Fi *positioning*, memperkuat kemampuan navigasi di area yang sulit dijangkau oleh kamera (C. Liu dkk., 2023). Teknologi ini telah diterapkan dalam berbagai sektor, termasuk bandara, rumah sakit, museum, dan pabrik, untuk memberikan navigasi yang lebih efisien dan akurat meskipun tetap ada batasan seperti jangkauan sinyal dan tempat ekstream.

2.2.11. *ArUco Marker Generator*

Jarak antara kamera dan objek mempengaruhi jumlah piksel dalam citra yang dihasilkan, termasuk kondisi pencahayaan saat pengambilan gambar. Faktor-faktor tersebut dapat menyebabkan kemampuan navigasi dengan optimal. Oleh karena itu, diperlukan titik referensi sebagai acuan untuk menghitung piksel dalam citra, sehingga hasil pengukuran luas tidak dipengaruhi oleh jarak kamera dengan objek, dan ukuran objek tetap konsisten.

Pada proses navigasi, citra yang diambil oleh kamera, baik dari perangkat seluler atau robot, akan diproses terlebih dahulu melalui deteksi tepi untuk memisahkan *marker* dari latar belakang ruangan. Dengan deteksi tepi ini, dapat diidentifikasi koordinat tiap sudut *marker*, dan kemudian titik tengah (*center point*) *marker* ditentukan sebagai acuan posisi. Lebar dan tinggi piksel *marker*

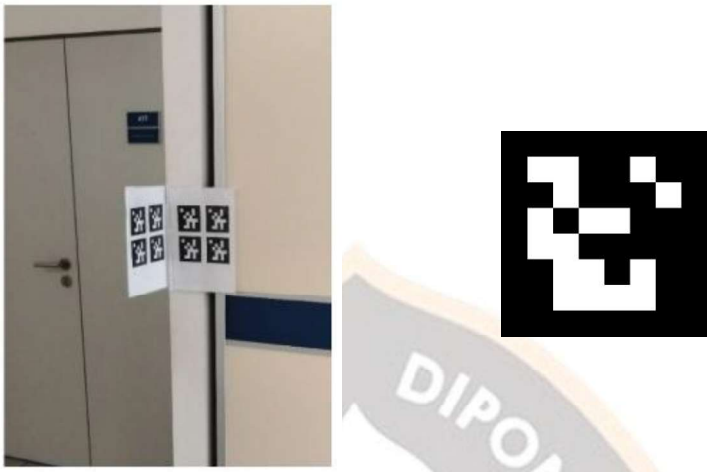
dihitung, dan jumlah piksel tersebut dikonversi ke satuan jarak (misalnya sentimeter), dengan titik referensi digunakan sebagai patokan. Hal ini memastikan bahwa *marker* tetap terdeteksi dengan baik, meskipun jarak atau sudut pandang kamera berubah. Gambar 2.9 menggambarkan deteksi *marker* beserta koordinat tiap sudutnya, yang digunakan untuk menentukan batas-batas *marker* dan membantu navigasi dalam ruangan secara optimal.



Gambar 2. 9 Koordinat sudut objek

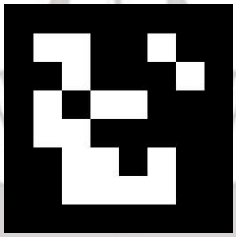
Titik referensi sebagai acuan dalam navigasi dalam ruangan adalah sebuah *marker* dengan ukuran tertentu, yang harus selalu muncul dalam satu frame bersama objek target yang dideteksi. ArUco *marker* ini tidak hanya berfungsi sebagai acuan referensi tetapi juga sebagai alat kalibrasi untuk memastikan bahwa jarak antara kamera dengan objek tidak mempengaruhi akurasi deteksi posisi atau ukuran objek.

Deteksi tepi dari citra yang diambil membantu mengidentifikasi batas objek dan membedakannya dari latar belakang. Proses ini juga berfungsi untuk menentukan koordinat di tiap sudut objek, yang penting untuk memahami bentuk geometris objek dalam navigasi. Koordinat sudut ini hanya dapat diidentifikasi jika objek memiliki sudut-sudut yang jelas, sehingga setelah deteksi tepi, citra diubah menjadi polyline untuk menonjolkan koordinat sudut-sudut tersebut. Gambar 2.10 menunjukkan contoh penggunaan ArUco *marker* sebagai titik referensi dalam sistem navigasi.



Gambar 2. 10 Titik referensi *ArUco marker*

ArUco *marker* digunakan sebagai acuan referensi yang krusial, karena koordinat batas objek yang terdeteksi tidak cukup untuk menghitung jarak atau area secara akurat tanpa adanya *marker* sebagai patokan. *Marker* ini memungkinkan sistem navigasi dalam ruangan untuk menentukan posisi objek secara konsisten, meskipun ada perubahan jarak atau sudut pandang kamera, sehingga navigasi dapat dilakukan dengan akurat dan andal.



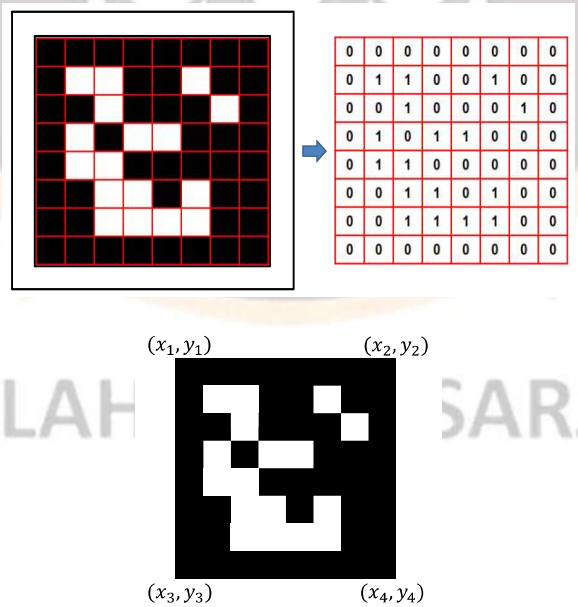
Gambar 2. 11 *ArUco Marker* 6x6 id 3

Titik referensi berupa ArUco *Marker* adalah objek berbentuk persegi yang memiliki ukuran luas yang sudah diketahui. Hal ini memastikan bahwa saat pengambilan citra dilakukan dari jarak yang berbeda, ukuran luas objek tetap tidak terpengaruh. ArUco *Marker* digunakan sebagai alat ukur referensi dan untuk kalibrasi piksel citra serta kamera yang digunakan. *Marker* ini terdiri dari kisi-kisi biner hitam dan putih, yang dikelilingi oleh batas berwarna hitam. Penggunaan latar belakang berwarna putih saat akuisisi citra membuat kontras *marker* lebih jelas, sehingga citra yang dihasilkan lebih tajam dan mudah diidentifikasi.

Pada Gambar 2.11, ditunjukkan contoh ArUco *Marker* yang digunakan dalam penelitian ini, dengan ukuran 6x6 dan ID 3. *Marker* ini tidak hanya berfungsi sebagai titik referensi untuk kalibrasi, tetapi juga memastikan bahwa ukuran objek yang terdeteksi tetap akurat meskipun ada perubahan jarak kamera, sehingga hasil deteksi tetap konsisten dan andal.

Identifikasi ArUco *Marker* memerlukan beberapa tahapan komputasi yang kompleks (Romero-Ramirez dkk., 2018). Langkah pertama adalah melakukan segmentasi kontur dengan metode ambang adaptif untuk memisahkan *marker* dari latar belakang. Langkah kedua melibatkan ekstraksi kontur dan pendekatan poligonal untuk memastikan bahwa bentuk persegi panjang *marker* tetap terjaga. Langkah ketiga adalah menghapus informasi yang tidak relevan sehingga hanya *marker* yang diperlukan yang dipertahankan untuk analisis lebih lanjut.

Setelah itu, setiap piksel pada ArUco *Marker* dibinarisasi, sehingga citra *marker* dibagi menjadi grid yang terdiri dari elemen-elemen biner berupa angka 0 atau 1 (Tocci dkk., 2021). Proses ini memastikan bahwa ArUco *Marker* dapat dibaca dengan akurat oleh sistem komputer. Gambar 2.12 menunjukkan sistem identifikasi ArUco *Marker* dengan 4 sudut beserta koordinatnya, yang digunakan sebagai referensi untuk menentukan posisi dan orientasi *marker* dalam ruang tiga dimensi.



Gambar 2. 12 (a) Binerisasi *ArUco Marker* dalam bentuk grid, (b) *ArUco Marker* dalam sistem 4 sudut koordinat

Berbagai penelitian telah menunjukkan keterkaitan antara penggunaan ArUco *marker* dengan algoritma *deep learning*, salah satunya adalah penelitian yang menggabungkan *Convolutional Neural Network* (CNN) dengan ArUco *Marker* untuk meningkatkan ketepatan pendaratan *Unmanned Aerial Vehicle* (UAV) atau drone (B. Li dkk., 2020). Dalam skenario ini, drone dapat mendarat dengan sempurna pada titik homebase jika ada penanda unik, seperti ArUco *Marker*, yang dikenali oleh kamera. Namun, tantangan muncul ketika penanda tersebut tertutup sebagian, sehingga kamera drone tidak dapat mengenali *marker* secara akurat.

Untuk mengatasi masalah ini, pendekatan yang memadukan CNN dengan algoritma deteksi objek MoNetViT (integrasi MobileNet dengan MobileViT) digunakan. Dengan kombinasi ini, sistem dapat mengenali dan memprediksi posisi *marker* bahkan jika sebagian *marker* tidak terlihat, memungkinkan drone untuk mendarat dengan tepat pada lokasi yang telah ditentukan. Pendekatan ini meningkatkan ketahanan sistem terhadap kendala visual dan memastikan keakuratan pendaratan drone meskipun kondisi lingkungan tidak ideal.

Penelitian mengenai ArUco *Marker* juga berkembang di sektor robotik, terutama dalam mendeteksi dan memperkirakan posisi objek menggunakan kamera RGB-D. Salah satu pendekatan yang signifikan adalah kombinasi antara *deep learning* DRBox-v2 dengan ArUco *Marker*, yang menjadi fokus penelitian oleh Jamzuri dan rekan-rekannya. Dalam penelitian ini, hasil deteksi objek dari DRBox-v2 diskalakan dan diaplikasikan pada citra depth untuk memperkirakan posisi dan keberadaan objek secara akurat. Penelitian ini menghasilkan *Average Precision* (AP) sebesar 0,740 untuk akurasi deteksi, sementara estimasi posisi objek menunjukkan kesalahan rata-rata sebesar 13,36 mm, dan kesalahan orientasi rata-rata sebesar 0,75 derajat (Jamzuri dkk., 2023). Hasil ini menunjukkan bahwa kombinasi metode *deep learning* dengan ArUco *Marker* dapat secara efektif digunakan untuk meningkatkan akurasi dalam deteksi dan pengukuran posisi objek dalam aplikasi robotik.

2.2.12. *Mean Square Error* (MSE)

Pengukuran akurasi model prediksi sering kali dilakukan menggunakan metrik *Mean Square Error* (MSE), terutama dalam analisis regresi dan machine learning. MSE mengukur rata-rata perbedaan kuadrat antara nilai yang diprediksi dan nilai sebenarnya (observasi). Metrik ini banyak digunakan dalam berbagai bidang, termasuk statistik, ekonomi, *machine learning*, dan teknik, untuk menilai kualitas prediksi dan membandingkan kinerja berbagai model.

Dalam konteks pengolahan citra digital, MSE digunakan untuk mengukur tingkat kesalahan atau deviasi antara citra yang dihasilkan oleh suatu proses pengolahan dengan citra asli atau citra referensi. MSE membantu mengevaluasi seberapa dekat citra hasil pengolahan dengan citra yang diharapkan, sehingga menjadi ukuran yang penting untuk menilai efektivitas algoritma atau metode pengolahan citra yang digunakan (Sara dkk., 2019; Tagi dkk., 2022). Persamaan untuk menghitung MSE ditunjukkan pada Persamaan 2.25(Luo dkk., 2021).

$$\mathrm{MSE} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i}=1}^{\mathrm{n}}(\mathrm{y_i} - \widehat{\mathrm{y_i}})^2 \tag{2.22}$$

Dengan:

- MSE = *Mean Square Error*
- y_i = nilai observasi aktual (nyata)
- $\hat{y}_i$ = nilai prediksi
- i = indeks atau urutan data
- n = total jumlah data

Korelasi adalah konsep dalam statistik yang mengukur tingkat hubungan atau keterkaitan antara dua variabel dalam suatu dataset. Korelasi mengidentifikasi sejauh mana perubahan pada satu variabel berkaitan dengan perubahan pada variabel lainnya. Dalam analisis statistik, korelasi sering digunakan untuk memahami hubungan antara dua atau lebih variabel, memberikan wawasan

tentang apakah variabel-variabel tersebut cenderung bergerak secara bersama-sama (positif) atau justru bergerak berlawanan (negatif).

Dua ukuran korelasi yang paling umum digunakan adalah:

1. Korelasi Pearson (*Pearson's correlation*): Ukuran ini mengukur hubungan linear antara dua variabel numerik. Korelasi Pearson memberikan nilai antara -1 dan 1, dengan interpretasi sebagai berikut:
    - 1 menunjukkan korelasi positif sempurna, yang berarti jika satu variabel naik, variabel lain juga naik dalam hubungan linear yang sempurna.
    - -1 menunjukkan korelasi negatif sempurna, yang berarti jika satu variabel naik, variabel lain turun dalam hubungan linear yang sempurna.
    - 0 menunjukkan tidak ada hubungan linear antara kedua variabel.
2. Korelasi Spearman (*Spearman's rank correlation*): Ukuran ini mengukur hubungan monotone antara dua variabel. Korelasi Spearman tidak mengharuskan hubungan linear, melainkan mengukur kesesuaian urutan atau peringkat relatif antara nilai dalam dua variabel. Nilai korelasi Spearman juga berkisar antara -1 dan 1, dengan interpretasi serupa dengan korelasi Pearson.

    Persamaan untuk menghitung korelasi dapat ditunjukkan pada Persamaan 2.23.

$$r_{xy} = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{\sqrt{(n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2)(n\sum_{i=1}^{n} y_i^2 - (\sum_{i=1}^{n} y_i)^2)}} \quad (2.23)$$

    Dengan:

    - r_{xy} = korelasi Pearson antara variabel x dan y
    - x, y = data atau variabel yang diukur korelasinya
    - n = jumlah data
    - i = indeks data ke-i