IMPLEMENTASI *OPTICAL CHARACTER RECOGNITION* PADA SISTEM PAGAR PINTAR BERBASIS *INTERNET OF THINGS* DAN *COMPUTER VISION*

Rifki Firmansyah[1*], Abdullah Hanif[2], Irwansyah[3]

[1,2,3]Politeknik Negeri Samarinda
E-mail : rifkifirmansyah132@gmail.com[*]

ABSTRAK

Perkembangan teknologi Internet of Things (IoT) dan Computer Vision telah mendorong inovasi dalam sistem keamanan otomatis, salah satunya pada sistem pengendalian akses kendaraan. Penelitian ini bertujuan untuk mengimplementasikan teknologi Optical Character Recognition (OCR) pada sistem pagar pintar guna mengotomatiskan pembukaan dan penutupan pagar berdasarkan hasil pengenalan plat nomor kendaraan secara real-time. Sistem ini memanfaatkan modul ESP32-CAM sebagai perangkat akuisisi citra, Python sebagai pemroses OCR menggunakan pustaka Tesseract OCR, serta motor stepper dengan driver ULN2003 sebagai aktuator penggerak pagar. Metode pengembangan sistem menggunakan model Waterfall, meliputi analisis kebutuhan, perancangan, implementasi, pengujian, dan pemeliharaan. Selain itu, sistem juga menggunakan database lokal untuk mencatat riwayat plat nomor kendaraan yang terverifikasi. Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi dan membaca plat nomor kendaraan dengan tingkat keberhasilan rata-rata XX%, serta membuka pagar dengan waktu respon rata-rata XX detik setelah plat nomor dikenali. Berdasarkan hasil tersebut, sistem pagar pintar berbasis OCR ini dapat meningkatkan efisiensi dan keamanan proses akses kendaraan tanpa memerlukan interaksi manual.

Kata Kunci : *IoT, OCR, ESP32-CAM, Smart Gate, Computer Vision*

PENDAHULUAN

Perkembangan teknologi informasi dan komunikasi pada era digital telah mendorong transformasi berbagai sektor, termasuk bidang keamanan dan otomasi. Sistem keamanan konvensional yang masih mengandalkan interaksi manual secara bertahap mulai digantikan oleh sistem cerdas yang mampu bekerja secara otomatis, cepat, dan efisien. Salah satu implementasi yang berkembang adalah sistem pagar pintar (*smart gate system*) yang memanfaatkan teknologi *computer vision* dan *Internet of Things* (IoT) untuk mengendalikan akses kendaraan secara otomatis. Penerapan sistem ini tidak hanya meningkatkan efisiensi proses keluar-masuk kendaraan, tetapi juga memperkuat aspek keamanan melalui proses identifikasi kendaraan secara otomatis tanpa memerlukan intervensi manusia. Integrasi teknologi *computer vision*, *Optical Character Recognition* (OCR), dan IoT memungkinkan sistem melakukan pengenalan plat nomor kendaraan, pencatatan data, serta pengendalian aktuator secara *real-time* sehingga sangat sesuai diterapkan pada kawasan perumahan, gedung perkantoran, kawasan industri, maupun area parkir dengan tingkat mobilitas kendaraan yang tinggi.

Internet of Things merupakan konsep yang menghubungkan berbagai perangkat fisik melalui jaringan komunikasi sehingga mampu mengumpulkan, mengirimkan, dan memproses data secara otomatis. Menurut Prawiyogi dan Anwar (2023), IoT merupakan infrastruktur jaringan global yang menghubungkan objek fisik dan virtual melalui teknologi komunikasi dengan arsitektur yang terdiri atas *perception layer*, *network layer*, dan *application layer*. Arsitektur tersebut memungkinkan sensor dan aktuator melakukan interaksi dengan lingkungan fisik, mengirimkan data melalui jaringan komunikasi, kemudian diproses pada lapisan aplikasi untuk menghasilkan keputusan secara otomatis. Implementasi IoT memberikan berbagai keuntungan berupa otomatisasi, efisiensi operasional, serta kemampuan pemantauan secara *real-time*, meskipun masih menghadapi tantangan berupa keamanan data, interoperabilitas perangkat, serta pengelolaan jaringan dalam skala besar (Prawiyogi & Anwar, 2023).

Dalam sistem pagar pintar, teknologi *computer vision* berperan sebagai kemampuan komputer untuk memperoleh, mengolah, dan memahami informasi visual dari citra maupun video sehingga mampu mengenali objek tertentu secara otomatis. Irfansyah, Ruseno, Yani, dan Santoso (2025) menjelaskan bahwa *computer vision* merupakan salah satu cabang kecerdasan buatan yang banyak dimanfaatkan pada sistem keamanan modern karena mampu melakukan deteksi objek, pengenalan wajah, serta analisis kondisi lingkungan secara otomatis. Secara umum, proses *computer vision* meliputi akuisisi citra, *pre-processing*, ekstraksi fitur, klasifikasi objek, hingga interpretasi hasil yang selanjutnya digunakan sebagai dasar pengambilan keputusan. Perkembangan metode *deep learning* juga telah meningkatkan kemampuan sistem dalam mengenali objek secara lebih akurat dibandingkan metode konvensional, walaupun tetap dipengaruhi oleh kualitas data, kebutuhan komputasi, serta kondisi lingkungan seperti pencahayaan dan sudut pengambilan gambar (Irfansyah et al., 2025).

Salah satu implementasi penting dari *computer vision* adalah teknologi *Optical Character Recognition* (OCR), yaitu metode yang digunakan untuk mengenali karakter teks dari suatu citra dan mengubahnya menjadi data digital yang dapat diproses oleh komputer. OCR bekerja melalui tahapan akuisisi citra, *pre-processing*, segmentasi karakter, ekstraksi fitur, proses pengenalan karakter, hingga *post-processing*. Akurasi OCR dipengaruhi oleh resolusi kamera, jarak pengambilan gambar, pencahayaan, kualitas objek, serta metode yang digunakan pada proses pengolahan citra. Zulhijar, Basri, dan Marlina (2025) menyatakan bahwa kombinasi teknik *pre-*

processing yang tepat dengan model OCR yang sesuai mampu menghasilkan tingkat akurasi tinggi pada kondisi ideal, namun performanya masih mengalami penurunan ketika kondisi pencahayaan kurang baik maupun jarak pengambilan gambar semakin jauh.

Penerapan OCR pada sistem pengenalan plat nomor kendaraan atau *Automatic Number Plate Recognition* (ANPR) telah banyak dikembangkan sebagai solusi otomatisasi pengendalian akses kendaraan. Sistem ANPR bekerja melalui tahapan akuisisi citra kendaraan, deteksi lokasi plat nomor, segmentasi karakter, pengenalan karakter menggunakan OCR, kemudian mengintegrasikan hasil pengenalan ke dalam sistem kontrol maupun basis data. Penelitian Hanif, Nasrullah, dan Setyawan (2023) menunjukkan bahwa tingkat keberhasilan OCR dalam mengenali plat nomor kendaraan mencapai 93,3% pada jarak 150 cm, sebesar 80% pada jarak 200 cm, dan menurun menjadi 46,6% pada jarak 250 cm. Hasil tersebut menunjukkan bahwa jarak pengambilan gambar memiliki pengaruh signifikan terhadap keberhasilan sistem dalam mengenali karakter plat nomor kendaraan.

Berbagai penelitian terdahulu telah mengembangkan sistem deteksi plat nomor maupun sistem pagar otomatis berbasis IoT dengan pendekatan yang berbeda. Veri Arinal et al. (2024) mengembangkan sistem otomatisasi palang pintu berbasis *machine learning* menggunakan *Cascade Classifier* dan EasyOCR. Penelitian tersebut berhasil mengotomatisasi pembukaan palang berdasarkan data plat nomor yang tersimpan pada basis data, namun sistem masih memiliki keterbatasan pada kondisi pencahayaan rendah, hujan, serta variasi bentuk plat nomor terbaru. Selain itu, antarmuka yang digunakan masih berbasis *command line* sehingga kurang praktis bagi pengguna. Shafin Adinata et al. (2024) mengembangkan sistem monitoring pagar otomatis berbasis IoT menggunakan ESP32, sensor gerak, dan platform *ThingsBoard* sehingga mampu melakukan pemantauan dan pengendalian secara *real-time*. Sistem tersebut berhasil memberikan notifikasi terhadap aktivitas mencurigakan serta memungkinkan pengendalian akses melalui jaringan internet, meskipun masih memiliki ketergantungan terhadap kualitas koneksi jaringan.

Penelitian lain juga menunjukkan bahwa pengembangan metode OCR terus mengalami peningkatan. Nehru et al. (2024) menerapkan metode *Canny Filter* pada OCR untuk identifikasi tanda nomor kendaraan bermotor dan memperoleh akurasi rata-rata sebesar 73,75% pada siang hari serta 67,50% pada malam hari. Sementara itu, Zakiyamani et al. (2022) mengembangkan sistem deteksi dan pengenalan plat nomor menggunakan OpenCV, WpodNet, dan *Convolutional Neural Network* (CNN) berbasis Python yang menghasilkan tingkat akurasi mencapai 96% dari 40 citra kendaraan yang diuji. Walaupun memiliki tingkat akurasi yang tinggi, sistem tersebut masih dipengaruhi oleh kondisi pencahayaan, sudut pengambilan gambar, serta belum menjelaskan kemampuan integrasi secara *real-time* dengan sistem IoT.

Pada sisi perangkat keras, modul ESP32-CAM menjadi salah satu mikrokontroler yang banyak digunakan karena telah dilengkapi kamera, konektivitas Wi-Fi, serta kemampuan pemrosesan data yang memadai untuk aplikasi IoT berbasis *computer vision*. Arrahma dan Mukhaiyar (2023) menyatakan bahwa ESP32-CAM mampu melakukan *video streaming* melalui *web browser* dan mendeteksi objek hingga jarak sekitar 150 cm. Selain memiliki ukuran yang ringkas dan biaya implementasi yang relatif rendah, ESP32-CAM juga mendukung pengembangan sistem keamanan berbasis pengolahan citra. Namun demikian, keterbatasan memori, kemampuan komputasi, konsumsi daya, serta kondisi pencahayaan tetap menjadi faktor yang memengaruhi performa sistem (Herwandi, Ramadhan, Sunggono, & Ferawati, 2025).

Agar hasil identifikasi kendaraan dapat disimpan dan dimanfaatkan dalam proses pengendalian akses, sistem memerlukan dukungan basis data sebagai media penyimpanan

informasi kendaraan. Menurut Adisa dan Nasution (2023), *Database Management System* (DBMS) memungkinkan penyimpanan data secara terstruktur, mengurangi redundansi melalui normalisasi, menjaga integritas data, serta mendukung proses pencarian menggunakan *Structured Query Language* (SQL). Integrasi basis data pada sistem pagar pintar memungkinkan proses verifikasi kendaraan dilakukan secara otomatis berdasarkan data yang telah tersimpan sehingga meningkatkan efisiensi sekaligus keamanan sistem.

Pengembangan perangkat lunak pada sistem ini memanfaatkan Arduino IDE sebagai lingkungan pemrograman mikrokontroler. Arduino IDE merupakan perangkat lunak *open-source* yang menyediakan berbagai pustaka sehingga memudahkan integrasi sensor, aktuator, serta modul komunikasi pada sistem IoT. Kamal, Firdayanti, Tyas, Buckhari, dan Pattasang (2023) menjelaskan bahwa Arduino IDE memiliki antarmuka sederhana dan mendukung proses pengembangan prototipe secara cepat, sedangkan Jurnal *Teknologika* (2025) menyebutkan bahwa platform tersebut sangat sesuai digunakan untuk mengembangkan sistem otomasi berbasis sensor dan aktuator meskipun memiliki keterbatasan untuk implementasi sistem berskala besar.

Berdasarkan berbagai penelitian terdahulu, dapat diketahui bahwa sistem deteksi plat nomor kendaraan telah menunjukkan performa yang baik, namun masih memiliki beberapa keterbatasan, seperti sensitivitas terhadap kondisi pencahayaan, sudut pengambilan gambar, jarak kamera, serta belum optimalnya integrasi antara sistem OCR, IoT, dan mekanisme pengendalian pagar secara otomatis. Selain itu, sebagian besar penelitian masih menggunakan kamera eksternal atau perangkat pemrosesan terpisah sehingga biaya implementasi relatif lebih tinggi. Oleh karena itu, penelitian ini mengembangkan sistem pagar pintar berbasis *computer vision*, OCR, dan IoT dengan memanfaatkan ESP32-CAM sebagai perangkat akuisisi citra sekaligus mikrokontroler, motor stepper sebagai aktuator pembuka pagar, serta basis data untuk proses verifikasi kendaraan secara otomatis. Sistem yang dikembangkan diharapkan mampu mengenali plat nomor kendaraan secara akurat, membuka pagar secara otomatis berdasarkan hasil identifikasi OCR, serta melakukan pencatatan data kendaraan secara *real-time* sehingga dapat meningkatkan efektivitas, efisiensi, dan keamanan sistem pengendalian akses kendaraan.

METODE

Penelitian ini menggunakan metode pengembangan perangkat lunak *Waterfall* karena tahapan pengembangan sistem telah ditentukan secara jelas sejak awal sehingga sesuai untuk pembangunan prototipe sistem pagar pintar berbasis *Optical Character Recognition* (OCR), *Internet of Things* (IoT), dan *Computer Vision*. Penelitian dilaksanakan pada Mei–Oktober 2025 di rumah penulis yang berlokasi di Loa Janan Ulu, Kabupaten Kutai Kartanegara, Kalimantan Timur. Lokasi tersebut dipilih karena menyediakan lingkungan yang memadai untuk proses perancangan, perakitan perangkat keras, pengembangan perangkat lunak, serta pengujian prototipe secara langsung.

Objek penelitian berupa prototipe sistem pagar pintar yang mampu melakukan pengendalian akses kendaraan secara otomatis berdasarkan hasil pengenalan plat nomor kendaraan. Sistem memanfaatkan teknologi OCR untuk mengenali karakter pada plat nomor, *Computer Vision* untuk proses akuisisi dan pengolahan citra, serta IoT sebagai media komunikasi antarperangkat. Prototipe terdiri atas modul ESP32-CAM sebagai perangkat utama sekaligus kamera pengambil citra, aktuator berupa motor stepper sebagai penggerak pagar otomatis, basis data lokal untuk menyimpan data kendaraan, serta miniatur pagar sebagai media simulasi sistem.

OCR yang digunakan memanfaatkan *Tesseract OCR* untuk mengekstraksi karakter huruf dan angka dari citra plat nomor, kemudian hasil pengenalan dibandingkan dengan data kendaraan yang telah tersimpan pada basis data.

Perangkat keras yang digunakan meliputi ESP32-CAM dengan modul USB to TTL CH340 dan kamera OV2640, motor stepper, miniatur pagar, *case* ESP32-CAM, papan PCB, kabel jumper, catu daya 5 V, serta komputer sebagai perangkat pemrosesan. Perangkat lunak yang digunakan terdiri atas Arduino IDE sebagai lingkungan pemrograman mikrokontroler, *Tesseract OCR* sebagai mesin pengenalan karakter, dan basis data lokal sebagai media penyimpanan informasi kendaraan. ESP32-CAM berfungsi mengambil citra plat nomor kendaraan melalui jaringan Wi-Fi, sedangkan Arduino IDE digunakan untuk mengembangkan dan mengunggah program pengendali ke mikrokontroler.

Tahapan penelitian mengikuti alur metode *Waterfall* yang terdiri atas analisis kebutuhan, perancangan sistem, implementasi, pengujian, dan pemeliharaan. Tahap analisis kebutuhan dilakukan untuk mengidentifikasi kebutuhan fungsional dan nonfungsional sistem, meliputi proses pembacaan plat nomor kendaraan menggunakan OCR, mekanisme komunikasi IoT, serta integrasi antara kamera, mikrokontroler, basis data, dan aktuator. Tahap perancangan meliputi penyusunan arsitektur sistem, diagram blok, *flowchart*, dan desain prototipe pagar otomatis. Pada tahap implementasi dilakukan pengembangan perangkat lunak menggunakan Arduino IDE, integrasi ESP32-CAM dengan modul OCR, pembangunan basis data lokal, serta perakitan seluruh komponen perangkat keras sesuai rancangan. Tahap pengujian bertujuan mengevaluasi keberhasilan fungsi setiap komponen sistem, meliputi kemampuan kamera menangkap citra, keberhasilan OCR mengenali karakter plat nomor, proses verifikasi terhadap basis data, respons aktuator dalam membuka dan menutup pagar, serta kestabilan komunikasi sistem. Tahap pemeliharaan dilakukan melalui perbaikan kesalahan yang ditemukan selama pengujian dan penyempurnaan sistem agar memiliki performa yang lebih baik.

Arsitektur sistem diawali ketika kamera pada ESP32-CAM menangkap citra kendaraan dari arah depan. Citra tersebut diproses menggunakan OCR untuk mengekstraksi karakter huruf dan angka pada plat nomor kendaraan. Hasil ekstraksi kemudian dibandingkan dengan data yang tersimpan pada basis data lokal. Apabila nomor kendaraan terdaftar, sistem mengirimkan sinyal kendali kepada motor stepper sehingga prototipe pagar terbuka secara otomatis selama dua menit untuk memberikan akses kepada kendaraan. Sebaliknya, apabila nomor kendaraan tidak ditemukan pada basis data, sistem mempertahankan kondisi pagar tetap tertutup dan kembali menunggu proses akuisisi citra berikutnya. Seluruh proses berlangsung secara otomatis tanpa memerlukan intervensi operator sehingga menghasilkan sistem pengendalian akses kendaraan yang lebih efisien dan aman.

HASIL PENELITIAN DAN PEMBAHASAN

Hasil Penelitian

Penelitian ini berhasil mengimplementasikan sistem pagar pintar berbasis *Computer Vision, Optical Character Recognition* (OCR), dan *Internet of Things* (IoT) yang terdiri atas tiga subsistem utama, yaitu modul akuisisi citra menggunakan ESP32-CAM, modul pengolahan citra dan OCR berbasis Python, serta modul aktuator menggunakan motor stepper. Sistem bekerja secara terintegrasi mulai dari proses pengambilan citra plat nomor kendaraan, ekstraksi karakter

menggunakan OCR, verifikasi data terhadap *database* lokal, hingga pemberian perintah untuk membuka atau menutup pagar secara otomatis.

Tahap awal implementasi dilakukan dengan menganalisis kebutuhan sistem yang meliputi perangkat keras dan perangkat lunak. Perangkat keras yang digunakan terdiri atas ESP32-CAM sebagai kamera sekaligus mikrokontroler, motor stepper sebagai aktuator, serta miniatur pagar sebagai media implementasi. Sementara itu, perangkat lunak terdiri atas Python, OpenCV, *Tesseract OCR*, Arduino IDE, dan *database* lokal. Kombinasi teknologi tersebut memungkinkan proses identifikasi kendaraan dilakukan secara otomatis tanpa memerlukan intervensi pengguna.

Implementasi Program OCR

Program OCR dikembangkan menggunakan bahasa pemrograman Python dengan memanfaatkan pustaka *OpenCV*, *NumPy*, *Requests*, *PyTesseract*, dan *Regular Expression*. Program diawali dengan pengambilan citra dari ESP32-CAM melalui protokol HTTP, kemudian dilakukan proses *pre-processing* berupa konversi citra menjadi *grayscale*, *bilateral filtering*, deteksi tepi menggunakan metode *Canny Edge Detection*, pencarian kontur, pemotongan (*cropping*) area plat nomor, *adaptive thresholding*, hingga proses pengenalan karakter menggunakan *Tesseract OCR*. Hasil pembacaan kemudian dibersihkan menggunakan *regular expression* sehingga hanya menghasilkan kombinasi huruf kapital dan angka sesuai format plat nomor kendaraan Indonesia. Program berjalan secara kontinu dengan interval pengambilan gambar setiap 10 detik sehingga sistem mampu melakukan identifikasi kendaraan secara *real-time*.

```
import requests
import cv2
import numpy as np
import time
import pytesseract
import re

#
================================================================================
# PENTING: GANTI DENGAN IP ADDRESS ESP32-CAM ANDA YANG MUNCUL DI
SERIAL MONITOR
#
================================================================================
ESP32_CAM_URL = "http://10.142.56.69/capture" # Pastikan IP ini sudah benar

# Arahkan ke lokasi instalasi Tesseract Anda
pytesseract.pytesseract.tesseract_cmd = r'C:\Program Files\Tesseract-OCR\tesseract.exe'

def find_and_crop_plate(image):
    """
    Mendeteksi area plat nomor dalam gambar menggunakan deteksi kontur,
    lalu meng-crop gambar untuk menyisakan area plat saja.
    """
```

```
    if image is None:
        return None

    # Pra-pemrosesan khusus untuk deteksi kontur
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    # Filter ini mengurangi noise sambil menjaga tepi tetap tajam, bagus untuk deteksi bentuk
    bfilter = cv2.bilateralFilter(gray, 11, 17, 17)
    edged = cv2.Canny(bfilter, 30, 200)

    # Temukan kontur pada gambar yang sudah dideteksi tepinya
    contours, _ = cv2.findContours(edged.copy(), cv2.RETR_TREE,
cv2.CHAIN_APPROX_SIMPLE)

    # Urutkan kontur berdasarkan area dari terbesar ke terkecil, dan ambil 10 teratas
    contours = sorted(contours, key=cv2.contourArea, reverse=True)[:10]

    location = None
    # Loop pada kontur yang sudah diurutkan
    for contour in contours:
        # Aproksimasi (perkirakan) bentuk kontur menjadi poligon
        peri = cv2.arcLength(contour, True)
        approx = cv2.approxPolyDP(contour, 0.018 * peri, True)

        # Jika poligon yang dihasilkan memiliki 4 titik (segiempat), kita anggap itu kandidat plat
        if len(approx) == 4:
            location = approx
            break

    if location is None:
        print("Bentuk plat nomor (segiempat) tidak dapat ditemukan.")
        return None

    # Buat mask hitam seukuran gambar asli
    mask = np.zeros(gray.shape, np.uint8)
    # Gambar kontur yang ditemukan (sebagai area putih) pada mask
    new_image = cv2.drawContours(mask, [location], 0, 255, -1)
    # Terapkan mask ke gambar asli untuk mengisolasi plat
    new_image = cv2.bitwise_and(image, image, mask=mask)

    # Crop gambar berdasarkan area mask putih
    (x, y) = np.where(mask == 255)
    (topx, topy) = (np.min(x), np.min(y))
    (bottomx, bottomy) = (np.max(x), np.max(y))
    cropped = image[topx:bottomx+1, topy:bottomy+1]
```

```
    print("Plat nomor berhasil ditemukan dan di-crop.")
    return cropped

def preprocess_image_adaptive(image):
    """
    Fungsi pra-pemrosesan menggunakan Adaptive Thresholding.
    """
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    blur = cv2.GaussianBlur(gray, (5, 5), 0)
    adaptive_thresh = cv2.adaptiveThreshold(blur, 255,
cv2.ADAPTIVE_THRESH_GAUSSIAN_C, cv2.THRESH_BINARY_INV, 11, 2)
    kernel = np.ones((2,2),np.uint8)
    opening = cv2.morphologyEx(adaptive_thresh, cv2.MORPH_OPEN, kernel)
    return opening

def clean_text(text):
    """Membersihkan hasil OCR."""
    return re.sub(r'[^A-Z0-9\s]', '', text).strip()

def get_plate_number():
    """Fungsi utama untuk mengambil gambar, meng-crop, dan melakukan OCR."""
    print("Mencoba mengambil gambar dari ESP32-CAM...")
    try:
        response = requests.get(ESP32_CAM_URL, timeout=10)

        if response.status_code == 200:
            print("Gambar berhasil diterima. Memproses...")
            np_arr = np.frombuffer(response.content, np.uint8)
            img = cv2.imdecode(np_arr, cv2.IMREAD_COLOR)

            if img is None:
                print("Gagal men-decode gambar.")
                return None

            cv2.imwrite("received_image.jpg", img)

            # --- LANGKAH BARU: CROP PLAT NOMOR ---
            cropped_plate = find_and_crop_plate(img)

            if cropped_plate is None:
                return None

            # Simpan gambar yang di-crop untuk debugging
            cv2.imwrite("cropped_plate_image.jpg", cropped_plate)
```

```
            # Lanjutkan proses hanya dengan gambar yang sudah di-crop
            processed_img = preprocess_image_adaptive(cropped_plate)
            cv2.imwrite("processed_for_ocr.jpg", processed_img)

            custom_config = r'--psm 6 -c
tessedit_char_whitelist=ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789'
            text = pytesseract.image_to_string(processed_img, config=custom_config)
            cleaned_text = clean_text(text)

            print(f"Hasil Mentah OCR: '{text.strip()}' -> Hasil Bersih: '{cleaned_text}'")
            return cleaned_text
        else:
            print(f"Gagal mengambil gambar. Status HTTP: {response.status_code}")
            return None

    except requests.exceptions.RequestException as e:
        print(f"Error saat menghubungi ESP32-CAM: {e}")
        return None

if __name__ == '__main__':

    while True:
        plate = get_plate_number()

        if plate and len(plate) > 4:
            print("\n========================================")
            print(f"==> PLAT NOMOR TERDETEKSI: {plate}")
            print("========================================\n")
        else:
            print("==> Tidak ada plat nomor yang terdeteksi dengan jelas.\n")

        print("Menunggu 10 detik sebelum siklus berikutnya...")
        time.sleep(10)
```

Implementasi Program ESP32-CAM

ESP32-CAM dikonfigurasi sebagai *web server* yang menyediakan layanan pengambilan citra melalui jaringan Wi-Fi menggunakan protokol HTTP pada endpoint */capture*. Kamera dikonfigurasi menggunakan resolusi VGA (640 × 480 piksel) dengan format JPEG sehingga menghasilkan kualitas citra yang cukup baik sekaligus menjaga kecepatan transmisi data. Setelah berhasil terhubung ke jaringan Wi-Fi, ESP32-CAM mengirimkan alamat IP yang digunakan oleh program Python untuk memperoleh citra kendaraan secara otomatis. Proses pengiriman gambar dilakukan menggunakan *frame buffer* sehingga konsumsi memori tetap efisien dan sistem mampu memberikan respons secara berkelanjutan.

```
#include "esp_camera.h"
#include <WiFi.h>
#include "soc/soc.h"
```

```
#include "soc/rtc_cntl_reg.h"

const char* ssid = "TaKaSiMa";
const char* password = "gkmungkin";

// Definisikan pin untuk model AI-Thinker ESP32-CAM
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22

// Deklarasi Web Server
WiFiServer server(80);

void handleCapture(WiFiClient client);
void serverTask(void *pvParameters);

void setup() {
  WRITE_PERI_REG(RTC_CNTL_BROWN_OUT_REG, 0); // Disable brownout detector

  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  // Konfigurasi kamera
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
```

```
config.pin_d5 = Y7_GPIO_NUM;
config.pin_d6 = Y8_GPIO_NUM;
config.pin_d7 = Y9_GPIO_NUM;
config.pin_xclk = XCLK_GPIO_NUM;
config.pin_pclk = PCLK_GPIO_NUM;
config.pin_vsync = VSYNC_GPIO_NUM;
config.pin_href = HREF_GPIO_NUM;
config.pin_sccb_sda = SIOD_GPIO_NUM;
config.pin_sccb_scl = SIOC_GPIO_NUM;
config.pin_pwdn = PWDN_GPIO_NUM;
config.pin_reset = RESET_GPIO_NUM;
config.xclk_freq_hz = 20000000;
config.pixel_format = PIXFORMAT_JPEG;

// Frame size: Pilih resolusi yang tidak terlalu besar agar pemrosesan cepat
// Pilihan: FRAMESIZE_VGA (640x480) atau FRAMESIZE_SVGA (800x600)
config.frame_size = FRAMESIZE_VGA;
config.jpeg_quality = 12; // 0-63, lebih rendah = kualitas lebih tinggi
config.fb_count = 1;

// Inisialisasi kamera
esp_err_t err = esp_camera_init(&config);
if (err != ESP_OK) {
  Serial.printf("Inisialisasi kamera gagal dengan error 0x%x", err);
  return;
}

// Koneksi ke WiFi
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
  delay(500);
  Serial.print(".");
}
Serial.println("");
Serial.println("WiFi terhubung");

// Mulai server dan cetak IP
server.begin();
Serial.print("Server kamera siap! Gunakan 'http://");
Serial.print(WiFi.localIP());
Serial.println("/capture' untuk mengambil gambar.");

// Buat task untuk server agar tidak memblokir loop utama
xTaskCreatePinnedToCore(
  serverTask,   // Fungsi yang akan dijalankan
```

```
    "serverTask", // Nama task
    10000,        // Ukuran stack
    NULL,         // Parameter
    1,            // Prioritas
    NULL,         // Handle task
    0             // Core 0
  );
}

void loop() {
  // Tidak ada yang perlu dilakukan di sini karena server berjalan di task terpisah
  delay(10000);
}

// Fungsi untuk menangani permintaan HTTP untuk gambar
void handleCapture(WiFiClient client) {
  camera_fb_t * fb = NULL;

  // Mengambil frame dari kamera
  fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("Gagal mengambil frame");
    client.println("HTTP/1.1 500 Internal Server Error");
    client.println("Content-Type: text/plain");
    client.println();
    client.println("Gagal mengambil frame dari kamera");
    return;
  }

  // Mengirim header HTTP
  client.println("HTTP/1.1 200 OK");
  client.println("Content-Type: image/jpeg");
  client.println("Content-Disposition: inline; filename=capture.jpg");
  client.println("Content-Length: " + String(fb->len));
  client.println("Connection: close");
  client.println();

  // Mengirim data gambar
  client.write(fb->buf, fb->len);

  // Mengembalikan frame buffer agar memori bisa digunakan kembali
  esp_camera_fb_return(fb);
}

// Task untuk Web Server
```

```
void serverTask(void *pvParameters) {
  (void)pvParameters; // Mencegah warning unused parameter
  while (1) {
    WiFiClient client = server.available();
    if (client) {
      String currentLine = "";
      bool isFirstLine = true;
      while (client.connected()) {
        if (client.available()) {
          char c = client.read();
          if (c == '\n') {
            if (currentLine.length() == 0) {
              // Jika kita mendapat baris kosong, berarti ini adalah akhir dari header HTTP
              // kita tidak perlu melakukan apa-apa lagi selain menunggu client disconnect.
              break;
            } else {
              // Jika ini baris pertama, periksa apakah ini permintaan yang kita inginkan
              if (isFirstLine) {
                if (currentLine.startsWith("GET /capture")) {
                  handleCapture(client);
                }
                isFirstLine = false;
              }
              currentLine = ""; // Bersihkan baris untuk baris berikutnya
            }
          } else if (c != '\r') {
            currentLine += c;
          }
        }
      }
      client.stop();
    }
    delay(1); // Beri kesempatan task lain untuk berjalan
  }
}
```

Implementasi Program Motor Stepper

Motor stepper dikendalikan menggunakan Arduino Uno dengan bantuan *driver* ULN2003. Program dikembangkan menggunakan Arduino IDE dengan pustaka *Stepper.h*. Komunikasi antara komputer dan Arduino dilakukan melalui komunikasi serial, di mana sistem OCR mengirimkan perintah berupa teks **OPEN** atau **CLOSE** sesuai hasil verifikasi plat nomor kendaraan. Ketika menerima perintah **OPEN**, motor stepper berputar searah jarum jam sebanyak satu putaran penuh untuk membuka pagar. Sebaliknya, ketika menerima perintah **CLOSE**, motor berputar berlawanan arah jarum jam untuk menutup pagar kembali. Implementasi ini menunjukkan bahwa komunikasi antara sistem OCR dan aktuator berlangsung dengan baik.

```
#include <Stepper.h>

// Jumlah langkah per satu putaran penuh motor stepper 28BYJ-48
const int stepsPerRevolution = 2048;

// Inisialisasi pin ULN2003 (IN1, IN2, IN3, IN4)
Stepper myStepper(stepsPerRevolution, 8, 10, 9, 11);

String command = "";

void setup() {
  Serial.begin(9600);
  myStepper.setSpeed(15); // Kecepatan putaran (RPM)
  Serial.println("Sistem Pagar Otomatis Siap");
}

void loop() {
  // Periksa apakah ada data masuk dari serial
  if (Serial.available() > 0) {
    command = Serial.readStringUntil('\n');
    command.trim();

    if (command == "OPEN") {
      Serial.println("Membuka pagar...");
      // Putar searah jarum jam
      myStepper.step(stepsPerRevolution);
      delay(1000);
      Serial.println("Pagar terbuka");
    }
    else if (command == "CLOSE") {
      Serial.println("Menutup pagar...");
      // Putar berlawanan arah jarum jam
      myStepper.step(-stepsPerRevolution);
      delay(1000);
      Serial.println("Pagar tertutup");
    }
    else {
      Serial.println("Perintah tidak dikenal. Gunakan OPEN atau CLOSE.");
    }
  }
}
```

Pengujian Sistem OCR

Pengujian dilakukan menggunakan kamera ESP32-CAM dengan objek berupa plat nomor kendaraan bertuliskan **KT 2911 RK**. Tahapan pengujian dimulai dari pengambilan citra

menggunakan kamera, proses *cropping* area plat nomor, *pre-processing*, pengenalan karakter menggunakan OCR, hingga verifikasi hasil pembacaan. Berdasarkan hasil pengujian, sistem berhasil mengenali karakter pada plat nomor kendaraan dengan benar pada jarak pengambilan gambar 45 cm maupun 65 cm.

Pada tahap pengambilan citra, ESP32-CAM berhasil mengirimkan gambar ke komputer dan program Python menampilkan pesan bahwa gambar berhasil diterima untuk diproses. Selanjutnya sistem melakukan deteksi kontur sehingga area plat nomor berhasil dipisahkan dari latar belakang kendaraan melalui proses *cropping*. Tahap berikutnya dilakukan *adaptive thresholding* sehingga karakter pada plat memiliki kontras yang lebih tinggi terhadap latar belakang. Setelah proses tersebut selesai, *Tesseract OCR* berhasil mengenali karakter plat nomor dengan hasil **KT2911RK**, kemudian dilakukan proses pembersihan sehingga menghasilkan keluaran akhir **KT 2911 RK.**

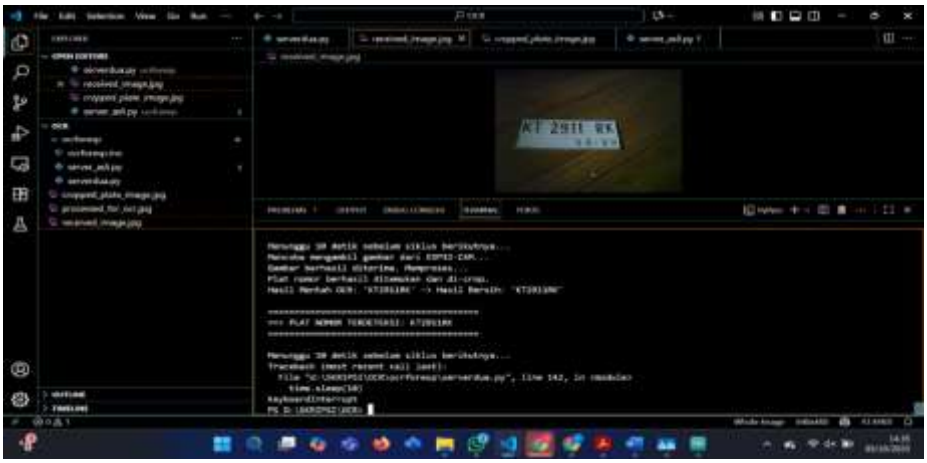
Pengujian juga menunjukkan bahwa komunikasi antara ESP32-CAM dan komputer berlangsung stabil selama proses identifikasi berlangsung. Seluruh tahapan mulai dari pengambilan citra, *cropping*, *pre-processing*, OCR, hingga pembersihan karakter dapat dilakukan secara otomatis tanpa memerlukan campur tangan pengguna.



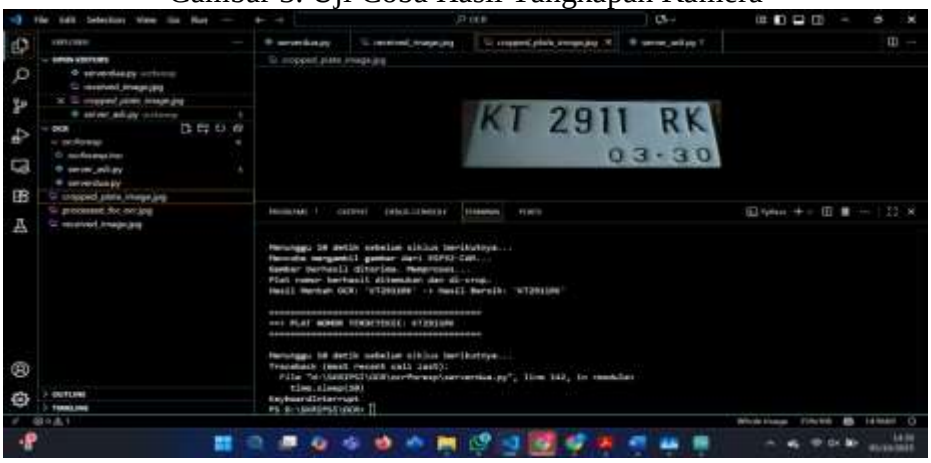
Gambar 1. Membaca Plat Nomor Kendaraan



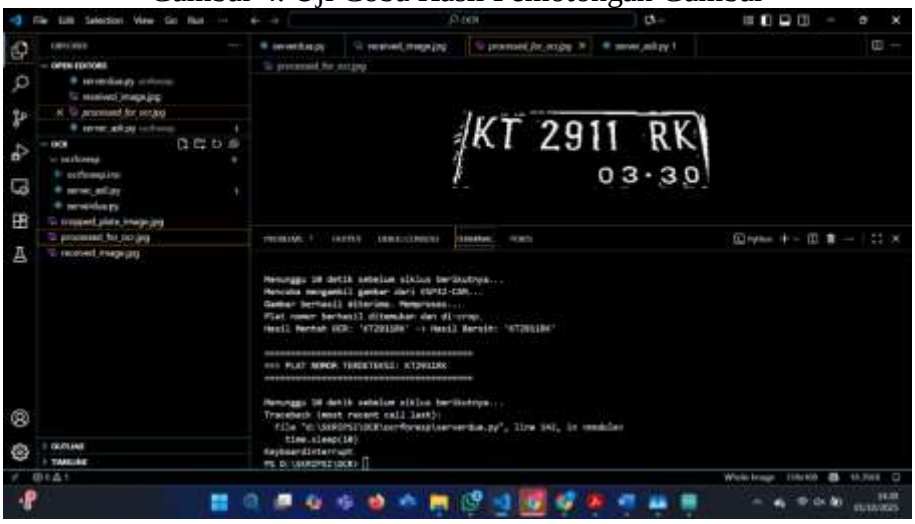
Gambar 2. Uji Coba Pemrosesan Citra



Gambar 3. Uji Coba Hasil Tangkapan Kamera



Gambar 4. Uji Coba Hasil Pemotongan Gambar



Gambar 5. Uji Coba Hasil Pengolahan Gambar

Pengujian Perangkat Keras

Pengujian perangkat keras dilakukan terhadap ESP32-CAM dan aktuator motor stepper. Pengujian ESP32-CAM dilakukan pada jarak sekitar 45 cm dari objek plat nomor kendaraan. Hasil

pengujian menunjukkan bahwa kamera mampu menghasilkan citra dengan kualitas yang cukup baik sehingga karakter pada plat nomor dapat dikenali oleh sistem OCR. Selain itu, komunikasi melalui jaringan Wi-Fi berjalan dengan baik sehingga citra dapat dikirimkan ke komputer tanpa mengalami kegagalan pengiriman.



Gambar 6. Jarak Uji Coba ESP32 Cam

Pengujian aktuator dilakukan menggunakan Arduino Uno, *driver* ULN2003, dan motor stepper yang terhubung dengan miniatur pagar. Berdasarkan hasil pengujian, motor stepper mampu menjalankan perintah membuka dan menutup pagar sesuai sinyal yang dikirimkan oleh sistem OCR melalui komunikasi serial. Ketika plat nomor kendaraan berhasil dikenali dan ditemukan pada *database*, sistem mengirimkan perintah **OPEN** sehingga motor stepper membuka pagar secara otomatis. Sebaliknya, apabila nomor kendaraan tidak ditemukan pada *database*, sistem tidak mengirimkan perintah pembukaan sehingga pagar tetap berada dalam kondisi tertutup.



Gambar 7. Uji Coba Pagar

Hasil implementasi menunjukkan bahwa integrasi antara ESP32-CAM, Python, OpenCV, *Tesseract OCR*, *database* lokal, dan motor stepper mampu membentuk sebuah sistem pagar pintar yang bekerja secara otomatis. Sistem diawali dengan proses akuisisi citra menggunakan ESP32-CAM yang berfungsi sebagai sensor visual sekaligus perangkat IoT. Citra yang diperoleh kemudian diproses menggunakan OpenCV melalui tahapan *grayscale conversion*, *bilateral filtering*, *Canny Edge Detection*, pencarian kontur, *cropping*, dan *adaptive thresholding*. Tahapan tersebut menghasilkan citra plat nomor yang memiliki kontras tinggi sehingga memudahkan proses pengenalan karakter oleh *Tesseract OCR*.

Penggunaan metode *contour detection* sebelum proses OCR terbukti meningkatkan kualitas pengenalan karakter karena sistem hanya memproses area plat nomor kendaraan. Selain mengurangi gangguan dari latar belakang, proses *cropping* juga mempercepat proses OCR karena area yang dianalisis menjadi lebih kecil. Selanjutnya, penggunaan *adaptive thresholding* menghasilkan pemisahan karakter dengan latar belakang yang lebih baik dibandingkan citra asli sehingga keluaran OCR menjadi lebih akurat.

Berdasarkan hasil pengujian, sistem mampu mengenali plat nomor kendaraan **KT 2911 RK** dengan benar pada pengujian jarak 45 cm maupun 65 cm. Hasil tersebut menunjukkan bahwa kombinasi ESP32-CAM dan *Tesseract OCR* mampu bekerja dengan baik pada kondisi pencahayaan normal serta sudut pengambilan gambar dari arah depan. Setelah karakter berhasil dikenali, sistem melakukan proses pembersihan menggunakan *regular expression* sehingga karakter yang tidak diperlukan dapat dihilangkan dan menghasilkan format plat nomor yang sesuai dengan data pada *database*.

Integrasi antara hasil OCR dan *database* lokal memungkinkan proses verifikasi kendaraan dilakukan secara otomatis. Apabila hasil OCR sesuai dengan data yang tersimpan, sistem mengirimkan perintah melalui komunikasi serial kepada Arduino Uno untuk mengaktifkan motor stepper sehingga pagar terbuka. Sebaliknya, apabila nomor kendaraan tidak ditemukan pada *database*, sistem mempertahankan kondisi pagar tetap tertutup. Mekanisme tersebut menunjukkan bahwa proses autentikasi kendaraan dapat dilakukan secara otomatis tanpa memerlukan operator.

Dari sisi perangkat keras, ESP32-CAM mampu menjalankan fungsi sebagai kamera jaringan (*network camera*) dengan baik melalui layanan HTTP. Pengiriman citra menuju komputer berlangsung stabil sehingga proses OCR dapat dilakukan secara berkala. Selain itu, motor stepper yang dikendalikan menggunakan Arduino Uno dan *driver* ULN2003 mampu menjalankan mekanisme buka-tutup pagar sesuai perintah yang diterima. Hal ini menunjukkan bahwa komunikasi antara sistem *Computer Vision* dengan perangkat aktuator telah berhasil diimplementasikan secara terpadu.

Pembahasan

Hasil penelitian menunjukkan bahwa implementasi *Optical Character Recognition* (OCR) pada sistem pagar pintar berbasis *Internet of Things* (IoT) dan *Computer Vision* telah menghasilkan sebuah rancangan sistem yang mengintegrasikan perangkat keras dan perangkat lunak untuk mendukung proses kontrol akses kendaraan secara otomatis. Sistem yang dikembangkan memanfaatkan modul ESP32-CAM sebagai perangkat akuisisi citra untuk menangkap gambar plat nomor kendaraan, kemudian citra diproses menggunakan metode OCR untuk mengekstraksi karakter huruf dan angka sebelum dilakukan proses verifikasi terhadap *database* lokal. Integrasi tersebut dilengkapi dengan aktuator berupa motor *stepper* yang dirancang sebagai mekanisme pembuka dan penutup pagar otomatis ketika plat nomor kendaraan teridentifikasi sebagai kendaraan yang memiliki hak akses. Berdasarkan hasil implementasi dan pengujian terhadap masing-masing komponen, proses akuisisi citra, pengolahan gambar, deteksi area plat nomor, serta ekstraksi karakter menggunakan OCR telah mampu berjalan dengan baik sehingga menghasilkan pembacaan plat nomor yang sesuai dengan objek pengujian. Meskipun demikian, integrasi keseluruhan sistem belum dapat beroperasi secara optimal karena masih terdapat kendala pada proses komunikasi data antara perangkat keras dan perangkat lunak, sehingga mekanisme verifikasi data dan pengendalian aktuator belum dapat direalisasikan secara otomatis sesuai dengan rancangan sistem. Kondisi tersebut menunjukkan bahwa keberhasilan implementasi sistem pagar pintar tidak hanya dipengaruhi oleh akurasi proses OCR, tetapi juga

oleh sinkronisasi komunikasi antarkomponen yang digunakan. Oleh karena itu, diperlukan pengembangan lebih lanjut pada mekanisme integrasi sistem, terutama pada komunikasi data antara ESP32-CAM, aplikasi pemrosesan OCR, *database*, dan aktuator agar sistem mampu bekerja secara *real-time* dan andal. Secara keseluruhan, hasil penelitian ini menunjukkan bahwa penerapan teknologi OCR dan *Computer Vision* memiliki potensi yang baik untuk dikembangkan sebagai sistem kontrol akses kendaraan otomatis berbasis IoT, sehingga dapat menjadi dasar pengembangan penelitian selanjutnya dengan tingkat integrasi, keandalan, dan akurasi sistem yang lebih tinggi.

KESIMPULAN

Berdasarkan hasil penelitian mengenai implementasi *Optical Character Recognition* (OCR) pada sistem pagar pintar berbasis *Internet of Things* (IoT) dan *Computer Vision*, dapat disimpulkan bahwa penelitian ini berhasil menghasilkan rancangan sistem pagar pintar yang memanfaatkan teknologi OCR untuk mengenali plat nomor kendaraan sebagai media kontrol akses otomatis. Sistem dirancang dengan mengintegrasikan modul ESP32-CAM sebagai perangkat akuisisi citra, *database* sebagai media penyimpanan data kendaraan, serta aktuator sebagai mekanisme pembuka dan penutup pagar dalam satu arsitektur yang mendukung konsep IoT dan *Computer Vision*. Hasil implementasi menunjukkan bahwa teknologi OCR memiliki potensi untuk digunakan dalam proses identifikasi kendaraan secara otomatis sehingga mampu mendukung peningkatan efisiensi dan keamanan sistem kontrol akses kendaraan. Prototipe yang dikembangkan telah mampu merepresentasikan konsep kerja sistem secara keseluruhan, mulai dari proses pengambilan citra, pengenalan karakter plat nomor, verifikasi data, hingga mekanisme pengendalian pagar. Meskipun demikian, implementasi dan pengujian sistem secara menyeluruh masih memerlukan penyempurnaan, khususnya pada aspek integrasi antara perangkat keras dan perangkat lunak agar seluruh komponen dapat bekerja secara optimal. Oleh karena itu, hasil penelitian ini dapat dijadikan sebagai dasar dalam pengembangan sistem pagar pintar berbasis OCR, IoT, dan *Computer Vision* yang lebih andal, akurat, serta siap diterapkan pada lingkungan nyata di masa mendatang.

UCAPAN TERIMA KASIH

Penulis mengucapkan puji dan syukur kepada Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga penelitian yang berjudul *Implementasi Optical Character Recognition pada Sistem Pagar Pintar Berbasis Internet of Things dan Computer Vision* dapat diselesaikan dengan baik. Penulis juga menyampaikan terima kasih kepada seluruh pihak yang telah memberikan dukungan, bimbingan, serta bantuan selama proses penelitian dan penyusunan artikel ini. Ucapan terima kasih secara khusus disampaikan kepada dosen pembimbing atas arahan, masukan, dan motivasi yang diberikan selama pelaksanaan penelitian. Penulis juga mengucapkan terima kasih kepada keluarga dan rekan-rekan yang telah memberikan dukungan moral maupun material sehingga penelitian ini dapat terselesaikan. Penulis berharap hasil penelitian ini dapat memberikan manfaat bagi pengembangan ilmu pengetahuan, khususnya pada bidang *Internet of Things*, *Computer Vision*, dan *Optical Character Recognition*, serta menjadi referensi bagi penelitian selanjutnya.

DAFTAR PUSTAKA

Adinata, S., Irwansyah, I., & Utomo, K. B. (2024). Rancang bangun *monitoring* pagar otomatis berbasis *Internet of Things* dengan kontrol dan keamanan. *JIP (Jurnal Informatika Polinema), 11*(1), 83–91.

Adisa, Y., & Nasution, M. I. P. (2023). Konsep dan peran sistem manajemen basis data relasional pada sistem informasi manajemen. *MASIP: Jurnal Manajemen Administrasi Bisnis dan Publik Terapan, 1*(3), 76–83.

Arinal, V., Nuari, F. A., Sanip, W., Taufik, M., & Sarikah, D. (2024). Implementasi alat deteksi plat nomor kendaraan untuk otomatisasi palang pintu pada lingkungan perumahan RT 05/05 Gondrong dengan *machine learning*. *Kohesi: Jurnal Multidisiplin Saintek, 2*(10), 91–112.

Arrahma, S. A., & Mukhaiyar, R. (2023). Pengujian ESP32-CAM berbasis mikrokontroler ESP32. *JTEIN: Jurnal Teknik Elektro Indonesia, 4*(1), 60–66.

Hanif, A. R., Nasrullah, E., & Setyawan, F. X. A. (2023). Deteksi karakter plat nomor kendaraan dengan menggunakan metode *optical character recognition* (OCR). *Jurnal Informatika dan Teknik Elektro Terapan, 11*(1), 109–118.

Herwandi, A., Ramadhan, A. A., Sunggono, N. T., & Ferawati. (2025). Analisis kinerja ESP32-CAM dalam mendeteksi objek. *Bit-Tech (Binary Digital–Technology), 7*(2).

Irfansyah, D., Ruseno, N., Yani, A., & Santoso, G. (2025). Penerapan *computer vision* dalam sistem keamanan berbasis *artificial intelligence* (AI), robotik, dan otomatisasi. *Jurnal PASTI: Jurnal Publikasi Artikel Sistem Teknologi Informasi, 1*(1), 26–33.

Kamal, A., Firdayanti, U. M., Tyas, U. M., Buckhari, A. A., & Pattasang, P. (2023). Implementasi aplikasi Arduino IDE pada mata kuliah sistem digital. *TEKNOS: Jurnal Pendidikan dan Teknologi, 1*(1), 1–9.

Nehru, N., Hais, Y. R., & Inonu, A. C. D. (2024). Implementasi *Canny filter optical character recognition* (OCR) untuk identifikasi tanda nomor kendaraan bermotor (TNKB). *Jurnal Engineering, 6*(1), 1–9.

Prawiyogi, A. G., & Anwar, A. S. (2023). Perkembangan *Internet of Things* (IoT) pada sektor energi: *Systematic literature review*. *Jurnal MENTARI: Manajemen, Pendidikan dan Teknologi Informasi, 1*(2), 187–197.

Romdoni, M. R., Safitri, L., Huda, D. N., Mubarok, S., & Supriyadi, N. (2024). Pengendalian gerbang otomatis berjadwal dengan IoT dan Android. *Jurnal Pendidikan dan Teknologi Indonesia*.

Zakiyamani, M., Cahyani, T. I., Riana, D., & Hardianti, S. (2022). Deteksi dan pengenalan plat karakter nomor kendaraan menggunakan OpenCV dan *deep learning* berbasis Python. *Journal of Information Technology and Computer Science (INTECOMS), 5*(1), 56–64.