DAFTAR GAMBAR

halaman

DAFTAR TABEL

DAFTAR ARTI LAMBANG DAN SINGKATAN

Lambang/Singkatan	Arti dan Keterangan
ALPR	*Automatic License Plate Recognition*. Sistem pengenalan plat nomor kendaraan yang dapat diimplementasikan menggunakan beberapa cara, misalnya menggunakan algoritma deteksi objek.
GPIO	*General-Purpose Input-Output*. Pada komputer Raspberry Pi merupakan pin yang dapat diprogramkan sebagai pin masukan (*input*) atau keluaran (*output*) sinyal.
CSI-2	*Camera Serial Interface 2*. Spesifikasi yang mendefinisikan *interface* sebuah kamera dengan host (misalnya dengan komputer Raspberry Pi)
d.s.	Data sintesis. Data untuk pelatihan algoritma detektsi objek yang di-*generate* menggunakan komputer.

BAB I
PENDAHULUAN

1.1. Latar Belakang

Lahan kosong yang dikhususkan untuk memarkirkan kendaraan biasa disebut sebagai *parking lot* atau *car park* (lahan parkir), dan biasa digunakan oleh sektor publik atau swasta (Börklü & Sadık A. Kalyon, 2017). Salah satu bagian dari sebuah lahan parkir adalah sistem penghalang, yang biasa disebut sebagai *parking gate*, *boom gate* atau *boom barrier* yang berfungsi sebagai pengendali perpindahan dan pengaman kendaraan yang keluar dan masuk dari lahan parkir tersebut (Chaudhuri, 2018). Sistem penghalang pada sebuah lahan parkir dapat dibangun untuk dioperasikan secara manual (membutuhkan tenaga manusia) atau otomatis (menggunakan bantuan komputer atau mikrokontroler). Sistem penghalang pada lahan parkir gedung Departemen Teknik Elektro, Fakultas Teknik, Universitas Hasanuddin, dilakukan secara otomatis; dua pembaca RFID – satu untuk masuk, dan yang lainnya untuk keluar – terhubung melalui komputer Raspberry Pi yang mengontrol mekanisme palang (Djamaluddin, 2021). Pada sistem palang saat ini, secara *default* penghalang berada dalam posisi horizontal, sehingga menutup pintu masuk tempat parkir. Pembaca RFID secara aktif menunggu untuk membaca kartu RFID. Ketika kartu dibaca, identitas (ID) diperiksa terhadap *database* yang disimpan di penyimpanan komputer. Jika ID cocok dengan entri dalam *database*, penghalang dinaikkan, memungkinkan kendaraan untuk melewatinya. Jika tidak, penghalang akan tetap dalam posisi horizontal.

Sistem RFID yang digunakan saat ini memiliki beberapa kelemahan. Pengguna tempat parkir diharuskan mengeluarkan dan secara manual menempelkan kartu RFID mereka ke pembaca, yang dapat mengurangi kenyamanan pengguna. Misalnya, pengemudi mobil diharuskan membuka jendela saat hujan agar dapat mengetuk kartunya untuk mengakses tempat parkir. Pemeliharaan mungkin juga perlu sering dilakukan, karena lingkungan dapat memengaruhi kualitas pembaca kartu;

penggantian komponen yang lengkap diperlukan untuk salah satu pembaca, karena *pin header*-nya telah berkarat.

Pada tugas akhir ini, dibuat rancang bangun sistem baru yang berupaya menghilangkan kelemahan tersebut dengan menambahkan fungsionalitas yang didasarkan pada algoritma *Automatic License Plate Recognition* (ALPR). Dengan itu, pengguna tidak perlu lagi secara manual mengidentifikasi diri selain dengan menghentikan kendaraan mereka di depan palang.

Melihat peningkatan efisiensi yang dapat dilakukan pada sistem lahan parkir gedung Departemen Teknik Elektro, Fakultas Teknik, Universitas Hasanuddin dengan menggantikan beberapa bagian pada sistemnya, peneliti ingin mengajukan penelitian mengenai sistem deteksi plat kendaraan untuk stimulasi buka-tutup *gate* pada lahan parkir gedung Departemen Teknik Elektro, Fakultas Teknik, Universitas Hasanuddin, yang juga dinamakan sebagai sistem *smart gate*. Penelitian ini berjudul *"Rancang Bangun Sistem Deteksi Plat Kendaraaan untuk Stimulasi Buka-Tutup Gate Lahan Parkir Departemen Teknik Elektro, Fakultas Teknik, Universitas Hasanuddin"*.

1.2. Rumusan Masalah

Berdasarkan latar belakang tersebut, maka dapat dirumuskan masalah sebagai berikut:

1. Bagaimana pengenalan plat kendaraan bermotor secara otomatis (*automatic license plate recognition*, ALPR) dapat diimplementasikan?
2. Bagaimana mengintegrasikan kerja detektor plat nomor kendaraan secara otomatis (ALPR) dengan mekanisme sistem buka-tutup palang parkir?
3. Bagaimana kinerja mekanisme sistem buka tutup palang parkir berbasis sistem deteksi plat?

1.3. Tujuan Penelitian

Tujuan yang ingin dicapai dalam penelitian ini:

1. Membangun implementasi program yang dapat digunakan untuk mengenal plat kendaraan bermotor.

2. Membangun prototipe sistem di mana program deteksi plat nomor kendaraan bermotor diintegrasikan dengan mekanisme buka-tutup palang parkir.
3. Menguji kinerja sistem yang telah dirancang.

1.4. Batasan Masalah

Untuk membuat penulisan tugas akhir lebih terarah, maka tugas akhir ini diberikan beberapa batasan masalah:

1. Sistem yang dibangun adalah bersifat prototipe.
2. Sistem kendali berbasis komputer Raspberry Pi.
3. Jenis plat kendaraan yang dideteksi berwarna putih dengan latar belakang hitam.
4. Pada pengujian sistem, waktu malam hari diabaikan.

1.5. Sistematika Penulisan

Penyusunan tugas akhir ini memiliki sistematika penulisan sebagai berikut:

BAB 1 – PENDAHULUAN

Bab ini berisi uraian tentang latar belakang alasan penelitian ini dilakukan, rumusan masalah, tujuan penelitian, batasan masalah, metode penelitian, dan sistematika penelitian yang digunakan pada penelitian.

BAB 2 – TINJAUAN PUSTAKA

Bab ini berisi teori dasar dan penelitian lain yang terkait pada penelitian yang dilakukan. Pertama terdapat definisi atau penjelasan tentang sistem deteksi plat kendaraan serta pengaplikasiannya pada sistem perparkiran. Lalu terdapat penjelasan tentang komponen-komponen elektronik yang akan digunakan pada penelitian ini.

BAB 3 – PERANCANGAN SISTEM

Bab ini membahas tentang metode penelitian yang digunakan. Pertama terdapat penjelasan tentang rancangan umum untuk penelitian, lalu lokasi dan tahapan penelitian yang dilakukan. Pada tahapan penelitian ini akan dijelaskan desain perangkat keras (*hardware*), desain perangkat lunak (*software*) serta desain pengujian yang digunakan pada penelitian.

BAB 4 – HASIL PENGUJIAN DAN PEMBAHASAN

Pada bab ini akan dibahas hasil uji coba yang dilakukan. Pada bab ini pula akan ditunjukkan beberapa tabel hasil uji coba pengambilan data pada prototipe.

BAB 5 – PENUTUP

Bab ini berisi kesimpulan dari pembahasan permasalahan dan saran-saran untuk perbaikan dan penyempurnaan tugas akhir ini.

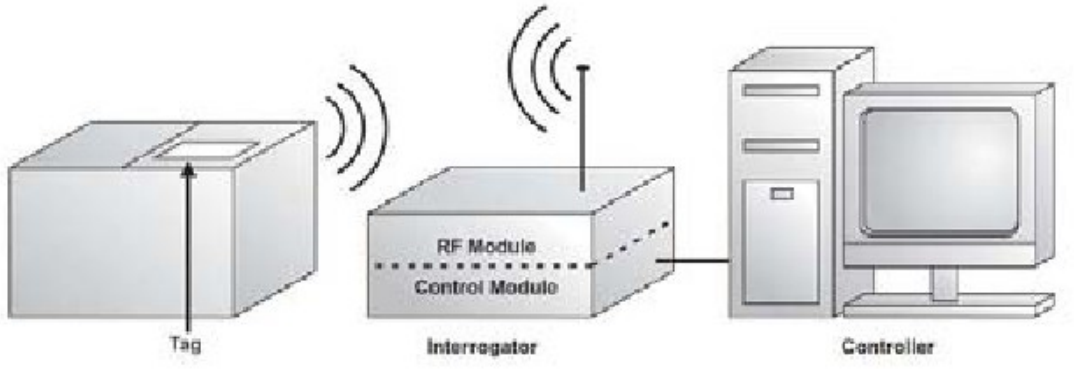
BAB II
TINJAUAN PUSTAKA

2.1. Sistem Stimulasi Palang Parkir

Menurut cara pengguna berinteraksi dengan sistem palang parkir untuk mengakses lahan parkir, sistem palang parkir dapat dibagi menjadi tiga jenis: semi-otomatis, di mana mekanisme buka-tutup palang parkir memerlukan picu dari tenaga pengguna palang parkir; otomatis, di mana mekanisme buka-tutup palang parkir dilakukan oleh operator pengelola lahan parkir melalui perangkat komputer, sehingga tenaga pengguna parkir tidak diperlukan untuk menstimulasi mekanisme palang; dan *full-automatic*, di mana stimulasi mekanisme buka-tutup palang parkir sepenuhnya dilakukan oleh komputer, tanpa memerlukan tenaga manusia (Prasetyo, 2020). Dua jenis teknologi yang dapat digunakan untuk kendali akses sebuah gerbang adalah *Radio Frequency Identification* (RFID) dan *Automatic License Plate Recognition* (ALPR) (Mohandes et al., 2016).

2.1.1. *Radio Frequency Identification* (RFID)

Teknologi RFID menggunakan komunikasi gelombang elektromagnetik dengan tujuan identifikasi melalui penggunaan peranti yang dinamakan *RFID tag (Malawakkang, 2019)*. Pada umumnya, sistem RFID bekerja dengan menggunakan satu peranti yang simpel pada satu titik, yang biasa disebut sebagai *tag* atau *transponder*, dan peranti yang lebih kompleks, yang biasa disebut sebagai *reader* (pembaca) atau *interrogator* pada titik lainnya. *RFID tag* terdiri atas antena, dan *microchip* yang berfungsi sebagai bagian penyimpanan data. Pembaca RFID tersusun atas antena untuk keperluan komunikasi dengan *tag*, dan modul kendali untuk komunikasi pembaca dengan komputer, yang biasanya digunakan untuk memproses data yang dikirimkan oleh pembaca (González, 2013).



Gambar 2.1: Rangkaian sederhana sebuah sistem RFID.

Sebuah sistem RFID dapat bersifat *read-only*, di mana data ditransfer dalam satu arah dari *tag* ke pembaca, atau *read-write*, di mana data ditransfer dari dan ke *tag* dan pembaca. Sebuah *tag* juga dapat berjenis aktif, di mana *tag* tersebut membutuhkan catu daya untuk dapat secara aktif memancarkan sinyal (Malawakkang, 2019), dan dapat pula berjenis pasif, di mana *tag* tidak memerlukan catu daya dan mendapat daya yang dibutuhkan dari medan elektromagnetik yang dipancarkan oleh pembaca (Tanim, 2016). Sistem RFID dapat beroperasi dalam frekuensi yang bermacam-macam. Terdapat empat *range* frekuensi yang biasanya digunakan pada teknologi RFID (González, 2013): frekuensi rendah (*low frequency*, LF) yang beroperasi dengan frekuensi 125 KHz dan 134.2 KHz, frekuensi tinggi (*high frequency*, HF) yang beroperasi pada frekuensi 13.56 MHz, frekuensi ultra tinggi (*ultra high frequency*, UHF) yang beroperasi pada frekuensi 433 MHz untuk *tag* aktif dan 840–960 MHz untuk *tag* pasif, dan frekuensi gelombang mikro (*microwave*) yang beroperasi pada frekuensi 2.45 GHz dan 5.8 GHz.

2.1.2. *Automatic License Plate Recognition* (ALPR)

Automatic License Plate Recognition (ALPR), dengan nama lain *Automatic Number Plate Recognition* (ANPR), *Car Plate Recognition* (CPR) dan *Automatic Vehicle Identification* (AVI) (Lubna et al., 2021) merupakan teknologi yang mengekstraksi informasi plat nomor dari sebuah atau deretan citra (Du et al., 2013). Sebuah sistem (ALPR) pada umumnya bekerja dengan mengambil sebuah citra sebagai masukan, dan apabila citra yang diterima mengandung citra sebuah

kendaraan, maka sistem akan menghasilkan isi dari plat nomor kendaraan tersebut sebagai keluaran. Kerja sistem ALPR dapat dibagi menjadi tiga tahap: ekstraksi plat nomor kendaraan; segmentasi dan ekstraksi karakter pada plat nomor yang diekstraksi; dan pengenalan karakter yang telah diekstraksi (Shashirangana et al., 2021).

2.1.1.1. Deteksi Objek

Deteksi objek adalah penerapan bidang pembelajaran mesin (*machine learning*) untuk tujuan penglihatan komputer, di mana deteksi objek yang memiliki kelas-kelas (*classes*) tertentu (misalnya manusia, bangunan, kendaraan) dilakukan. Dalam deteksi objek, algoritma ditugaskan untuk menghasilkan keluaran berupa daftar objek yang terdeteksi pada suatu citra. Posisi dan skala objek yang terdeteksi oleh algoritma kemudian ditunjukkan dengan menggambar kotak pembatas (*bounding boxes*) ke masing-masing objek yang terdeteksi (Russakovsky et al., 2015). Arsitektur deteksi objek umumnya terdiri dari tiga bagian: *backbone*, *neck*, dan *head*. *Backbone* detektor objek berfungsi sebagai bagian yang mengekstrak fitur-fitur yang dimiliki oleh citra (Yan et al., 2021); *neck* menggabungkan fitur yang diekstraksi di lapisan sebelumnya; dan *head* melakukan prediksi kelas objek yang terdeteksi dan kotak pembatas (Bochkovskiy et al., 2020).

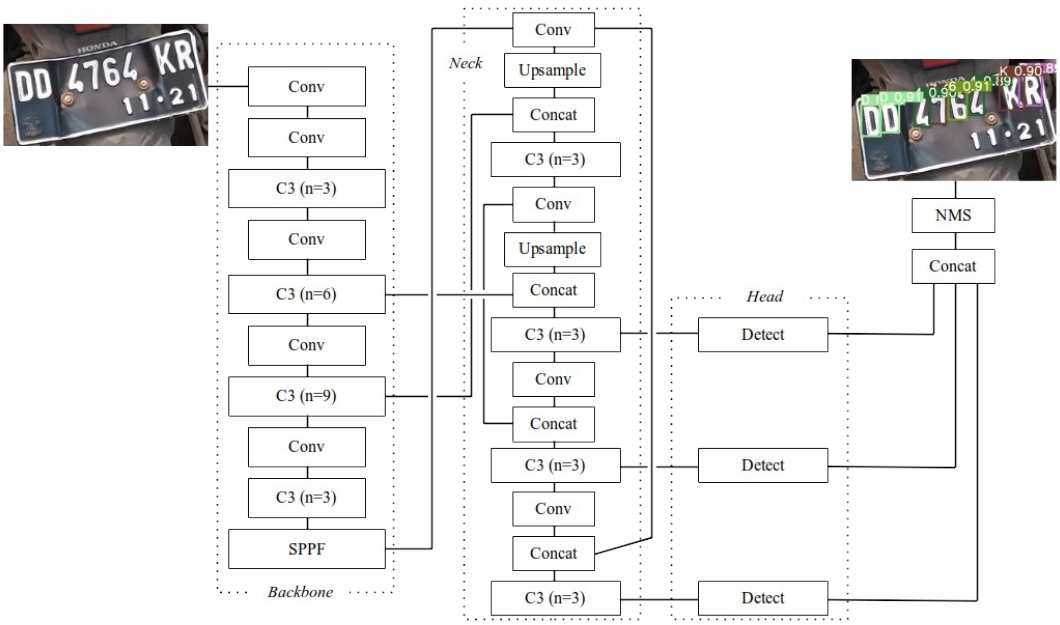
Sebagian besar metode statistika yang digunakan untuk fungsi deteksi plat nomor telah digantikan oleh pengaplikasian *deep learning* karena tingkat akurasinya yang tinggi dalam memecahkan masalah deteksi objek. Segementasi karakter juga dapat digunakan menggunakan *deep learning*, namun banyak proses ALPR tidak melalui proses segmentasi karakter secara eksplisit, dan lebih memilih segmentasi karakter secara implisit pada tahap selanjutnya, sehingga mengurangi jumlah parameter dan biaya komputasi yang diperlukan. Penggunaan *deep learning* memungkinkan *neural network* untuk langsung diberikan data piksel mentah dan menghasilkan keluaran sebagai mendeteksi ciri-ciri khas citra masukan dan sebagai *classifier* citra. Salah satu contoh sistem ALPR seperti ini adalah yang penggunaan metode deteksi objek *You Only Look Once* (YOLO) untuk kepentingan segmentasi dan pengenalan karakter (Shashirangana et al., 2021).

2.1.1.2. YOLO-Fastest

YOLO-Fastest (You Only Look Once – Fastest) (dog-qiuqiu, 2021) adalah arsitektur jaringan saraf konvolusional yang dikembangkan untuk tujuan deteksi objek. YOLO-Fastest (Ma et al., 2018) menggunakan ShuffleNet v2 sebagai *backbone*nya, sedangkan *neck* serta *head*-nya diambil dari YOLOv3, dengan modifikasi pada bagian *head*, di mana *head* dibuat agar tidak perlu mendeteksi objek-objek yang sangat kecil pada citra yang diolah (qiuqiuqiu, n.d.).

2.1.1.3. YOLOv5

YOLOv5 (*You Only Look Once, version 5*) adalah arsitektur jaringan saraf konvolusional yang diimplementasikan untuk tujuan deteksi objek. Arsitekturnya dikembangkan oleh Ultralytics, dan merupakan pengembangan dari YOLOv3 versi PyTorch yang juga dikembangkan oleh perusahaan yang sama. Awalnya, arsitekturnya akan diberi nama YOLOv4, namun, sudah ada arsitektur pendeteksi objek lain dengan nama yang sama, yang dikembangkan dari versi YOLOv3 dengan framework Darknet. Untuk menghindari tabrakan versi, nama YOLOv5 kemudian digunakan (Solawetz, 2020).



Gambar 2.2: Arsitektur YOLOv5

Arsitektur YOLOv5 hadir dengan beberapa versi: YOLOv5n, YOLOv5s,

YOLOv5m, YOLOv5l dan YOLOv5x. Perbedaan utama antara versi ini terletak pada jumlah modul ekstraksi fitur dan jumlah kernel di lapisan konvolusi (Yan et al., 2021). Pada **Gambar. 2.2.** aristektur YOLOv5 dapat dilihat.

Variabel *n* pada **Gambar. 2.2.**, beserta variabel yang dinamakan *depth multiple* atau *depth gain* (g_d) digunakan untuk menentukan jumlah modul C3 yang digunakan arsitektur. Hubungan antara *n*, g_d, dan banyaknya modul C3 yang digunakan (n_{new}) adalah:

$$n_{new} = \begin{cases} 1 & if \quad n \leq 1 \\ \max(\lfloor n \times g_d \rceil, 1) \end{cases} \tag{1}$$

Variabel yang disebut *width multiple* atau *width gain* (g_w) digunakan untuk menentukan jumlah kernel yang digunakan arsitektur saat melakukan konvolusi. Jumlah kernel ini secara tidak langsung ditentukan dengan mengubah jumlah *channels* yang ingin di-*output* lapisan Conv (D_{out}):

$$D_{out} \leftarrow \lceil \frac{D_{out} \times g_w}{8} \rceil \times 8 \tag{2}$$

2.2. Kendali Motor Penggerak Palang Parkir

Kendali motor sebuah palang penggerak palang parkir dilakukan melalui sebuah *ccontrol board* yang dimiliki oleh palang parkir. Pada *control board* tersebut terdapat pin UP, DOWN dan STOP, yang pada skematiknya terhubung dengan pin COM dan bersifat seperti *push button.* Palang akan bergerak ke posisi vertikal apabila COM dan UP dalam kondisi *short*, horizontal apabila COM dan DOWN dalam kondisi *short*, dan apabila palang sedang bergerak, palang akan berhenti apabila COM dan STOP dibuat ke kondisi *short* (*Barrier Gate Manual*, n.d.).

2.3. Raspberry Pi

Raspberry Pi merupakan sebuah *board* komputer kecil berukuran kartu kredit yang pertama kali diperkenalkan pada tahun 2012. Hampir semua komponen pada Raspberry Pi, termasuk CPU, GPU, perangkat keras audio dan *chip* memorinya dibangun sebagai satu komponen (Maksimović, et al., 2014). Pada Raspberry Pi 4 Model B, digunakan prosesor *quad core* 64-bit ARM Cortex A72, yang *clock speed*-nya sebesar 1.5GHz. SD Card berfungsi sebagai *hard drive* komputer, dan terdapat

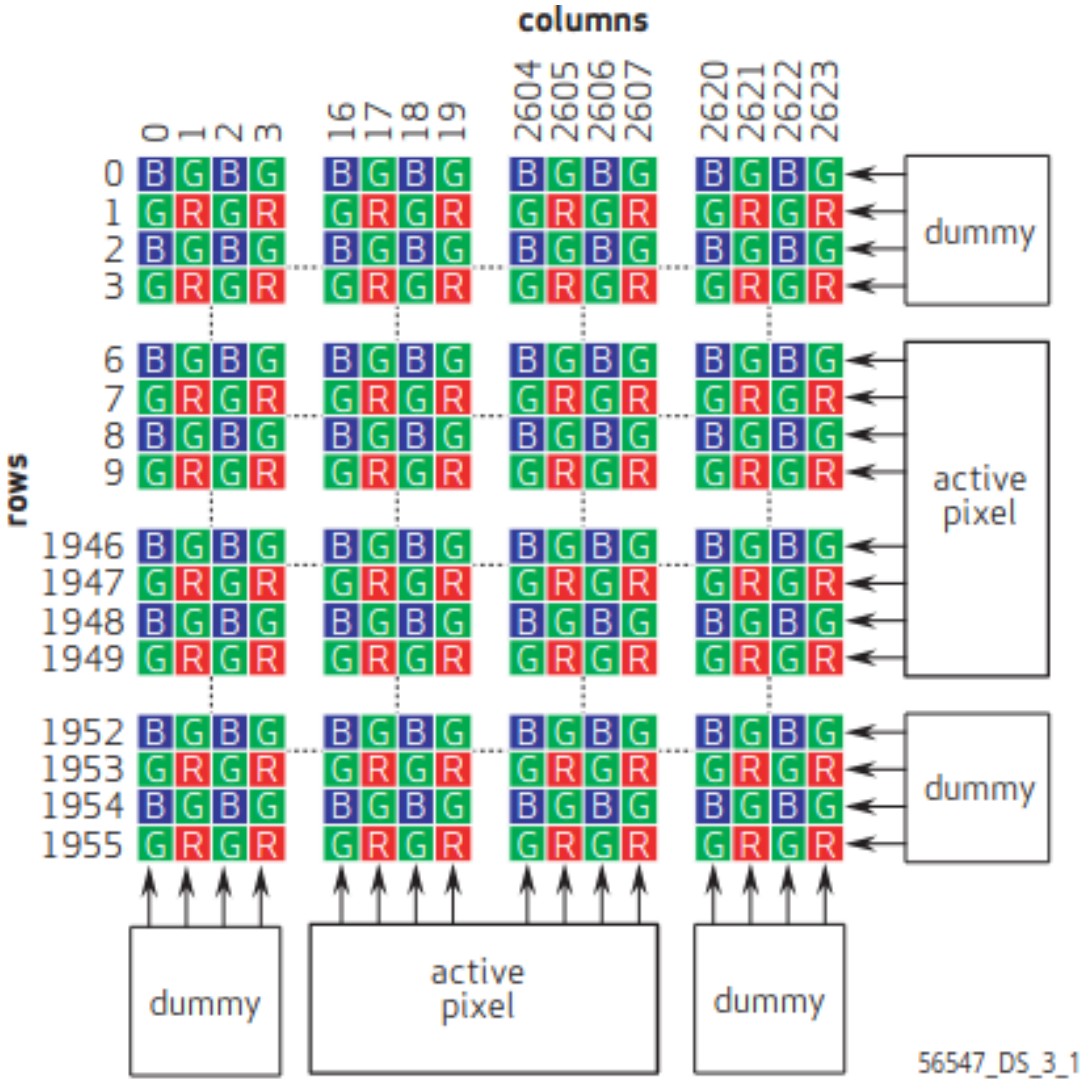
pula, 2 buah port USB2, 2 buah port USB3, serta port kamera dan *display*. Raspberry pi 4 Model juga memiliki *40-pin GPIO header (DATASHEET – Raspberry Pi 4 Model B*, 2019). Sistem operasi yang biasa digunakan adalah sebuah distribusi Linux yang dinamakan Raspberry Pi OS, yang sebelumnya juga dikenal sebagai Raspbian (Upton, n.d.). Spesifikasi Raspberry Pi 4 Model B dapat dilihat pada **Tabel 2.1.**.

Tabel 2.1: Spesifikasi Raspberry Pi 4 Model B (*DATASHEET – Raspberry Pi 4 Model B*, 2019)

Parameter	**Nilai**
Prosesor	Quad core 64-bit ARM-Cortex A72 @ 1.5GHz
RAM	Opsi RAM 1, 2, dan 4GB LPDDR4
Antarmuka	28 pin GPIO untuk *user* 2 port USB2 2 port USB3 2 port micro-HDMI 1 port *display 2-lane* MIPI DSI 1 port *camera 2-lane* MIPI CSI Media penyimpanan SD Card
Perangkat lunak	*Linux software stack* dengan sistem operasi Raspberry Pi OS yang berbasiskan pada distribusi Debian

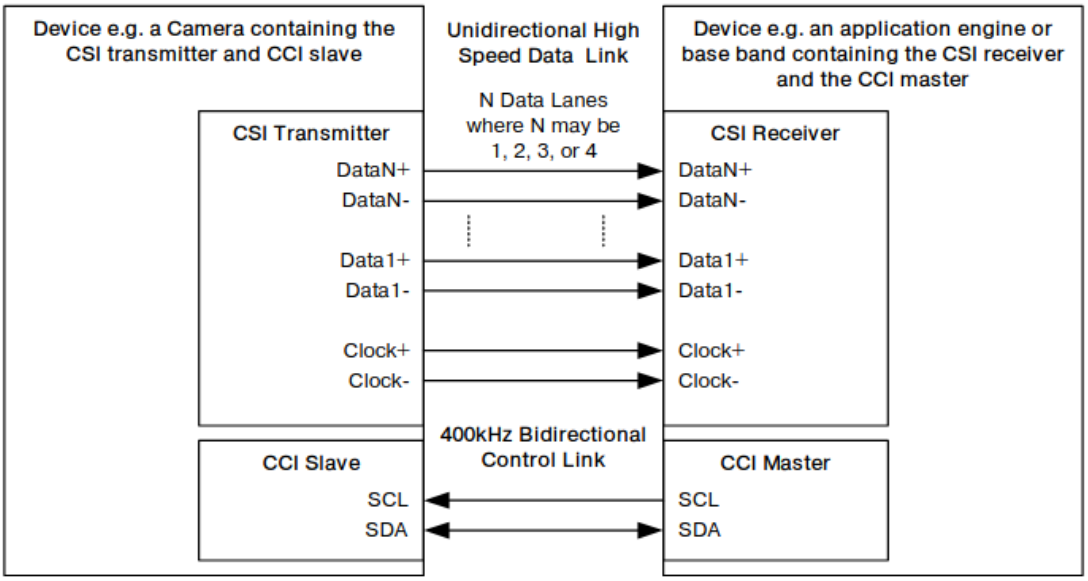
2.4. Sensor Citra OV5647

OV5647 merupakan sensor citra dengan resolusi 5 megapiksel. Resolusi 5 megapiksel tersebut didapatkan dari piksel aktif yang dimiliki *array* sensor-sensor piksel yang dilapisi filter warna pada perangkat (**Gambar 2.3.**), di mana *array* tersebut tersusun atas 1956 baris dan 2624 kolom (5,132,544 piksel). Filter-filter warna tersebut disusun dalam bentuk pola Bayer, di mana setengah dari tiap-tiap *array* terdiri atas filter hijau, seperempatnya terdiri atas filter merah, dan sisa seperempatnya lagi terdiri atas filter biru. Dari 5,132,544 piksel yang dimiliki oleh sensor, 2592x1944 (5,038,848) piksel merupakan piksel aktif yang menghasilkan keluaran. Piksel-piksel lainnya digunakan oleh sensor untuk kalibrasi tingkat kehitaman citra *(Color CMOS QSXGA (5 Megapixel) Image Sensor with OmniBSITM Technology*, n.d.).



Gambar 2.3: Layout susunan sensor-sensor piksel pada sensor citra OV5647 (*Color CMOS QSXGA (5 Megapixel) Image Sensor with OmniBSITM Technology*, n.d.).

Sensor OV5647 dihubungkan ke komputer Raspberry Pi 4 Model B dengan *interface* yang menggunakan spesifikasi CSI-2 (*Camera Serial Interface Type 2*) *(Raspberry Pi Documentation - Raspberry Pi OS*, n.d.), yang ilustrasinya dapat dilihat pada **Gambar 2.4.**. CSI-2 menspesifikasikan sensor agar terdiri atas CSI *transmitter* dan CCI (*Camera Control Interface*) *slave*, dan perangkat yang disambungkan ke sensor (misalnya komputer Raspberry Pi) agar memiliki CSI *receiver* dan CCI *master*.



Gambar 2.4: *Interface* kamera dengan perangkat yang disambung ke kamera menggunakan spesifikasi CSI-2 (*MIPI Alliance Standard for Camera Serial Interface 2 (CSI-2) Version 1.00 – 29 November 2005*, n.d.).

Pada Raspberry Pi, sensor OV5647 memiliki dua *data lanes (DATASHEET – Raspberry Pi 4 Model B*, 2019) yang arah transmisi datanya satu arah (*unidirectional*), sedangkan bagian CCI perangkat mengikuti spesifikasi I2C, di mana data mampu ditransimiskan secara *bidirectional*, namun juga hanya dapat ditransmisikan dalam satu arah pada satu waktu (*half duplex*) *(MIPI Alliance Standard for Camera Serial Interface 2 (CSI-2) Version 1.00 – 29 November 2005*, n.d.). Piksel-piksel citra yang didapatkan oleh *array* sensor ditransmisikan melalui kedua *data lanes* yang menyambungkan CSI *transmitter* dengan CSI *receiver*, sedangkan bagian CCI berfungsi sebagai fasilitator komunikasi komputer dengan sensor citra, misalnya sebagai jalur pengiriman perintah mengambil citra dari komputer ke sensor.

2.5. Sensor Ultrasonik JSN-SR04T

Sensor ultrasonik (*ultrasonic sensor*), yang kadang juga dikenal sebagai *ultrasonic transducer*, bekerja dengan memancarkan gelombang dengan frekuensi ultrasonik (frekuensi >20kHz), yang kemudian akan dipantulkan kembali ke sensor setelah melalui suatu benda. Sensor ultrasonik dapat digunakan untuk pengukuran

jarak suatu titik terhadap benda lain, dan mungkin dapat dipilih karena biaya yang diperlukan oleh sensor lain untuk keperluan hal yang sama, seperti sensor laser, memerlukan biaya yang lebih mahal (Al-Mahturi & R. Rahim, 2016).

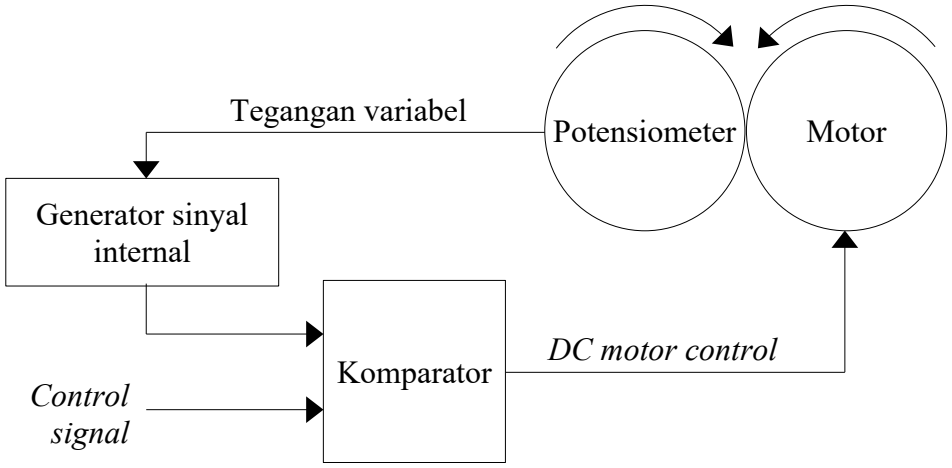
Pada sensor ultrasonik JSN-SR04T, modul *transmitter* memancarkan gelombang ultrasonik dengan frekuensi 40kHz setelah sensor menerima sebuah sinyal *trigger* (misalkan dari mikrokontroler seperti Arduino, atau dari komputer seperti Raspberry Pi). Setelah gelombang yang dipancarkan terpantulkan oleh suatu halangan, modul *receiver* sensor akan menerima pantulan gelombang tersebut, di mana frekuensi gelombang pantulan (atau *echo*) tersebut besarnya sama dengan gelombang yang dipancarkan setelah sensor di-*trigger*, yaitu 40kHz. Waktu perjalanan gelombang dari *transmitter* ke *receiver* adalah dua kali jarak antara sensor dan halangan yang memantulkan gelombang ultrasonik yang dipancarkan. Ketika sensor memancarkan gelombang ultrasonik, sensor juga menghasilkan sinyal TTL *high*, yang mengindikasikan bahwa sensor telah memulai penghitungan waktu perjalanan gelombang. Sinyal TTL dikembalikan menjadi *low*, menghentikan penghitungan waktu perjalanan gelombang pada sensor, setelah *receiver* menerima pantulan gelombang (Dswilan et al., 2021). Apabila dimisalkan waktu perjalanan gelombang tersebut adalah t , dan gelombang suara merambat dengan kecepatan c , maka jarak d antara sensor dengan halangan didepannya adalah:

$$d = \frac{t \times c}{2}$$ (Al-Mahturi & R. Rahim, 2016).

2.6. Motor Servo

Motor servo adalah jenis motor yang dapat digunakan apabila dibutuhkan penentuan posisi sudut (*angular positioning*) yang tepat. Sebuah motor servo dapat dianggap sebagai motor dc (yang dengan sendirinya tidak memiliki fungsi penentuan posisi sudut) yang memiliki pengendali (*controller*)nya sendiri. Rangkaian pengendali tersebut berfungsi membandingkan sebuah posisi sudut yang nilainya ditentukan oleh sinyal pengendali (*control signal*), dengan posisi motor pada saat itu, yang biasanya ditentukan menggunakan potensiometer yang ikut berputar saat motor berputar. Sinyal pengendali itu sendiri juga biasanya merupakan sinyal digital yang

diatur variasinya mengugnakan metode *pulse-width modulation* (PWM), sedangkan potensiometer pada rangkaian servo berfungsi sebagai pembagi tegangan (*voltage divider*), di mana tegangan dari pin tengah potensiometer digunakan pada generator sinyal internal servo untuk menghasilkan sebuah sinyal digital. Pengendali kemudian membandingkan sinyal tersebut dengan sinyal pengendali, lalu motor dc diputar untuk menyamakan kedua sinyal (Pinckney, 2006). Bagan yang mengilustrasikan prinsip kerja servo ini dapat dilihat pada **Gambar 2.5.**.



Gambar 2.5: Prinsip kerja motor servo (Pinckney, 2006).

Metode *pulse-width modulation* digunakan pada sinyal pengendali. Pada metode PWM, digunakan sinyal digital yang waktu *high*-nya dalam satu periode sinyal dibuat bervariasi, sehingga tegangan rata-rata yang dimiliki sinyal dapat digunakan untuk mengatur daya yang diberikan pada sebuah komponen. Rasio waktu *high* sebuah sinyal digital dalam satu periode dengan periode sinyal tersebut disebut sebagai *duty cycle* sebuah sinyal (Pinckney, 2006).

Pada servo Tower Pro SG90 dan Tower Pro MG995, sinyal pengendali yang digunakan memiliki periode 20ms, sehingga sinyal tersebut memiliki frekuensi 50Hz. Waktu high sinyal pengendali dalam satu periode digunakan servo untuk mengatur posisi servo, di mana posisi netral servo (sudut 0°) dicapai apabila waktu high sinyal pengendali 1.5ms, sudut 90° pada SG90 dan 60° pada MG995 pada

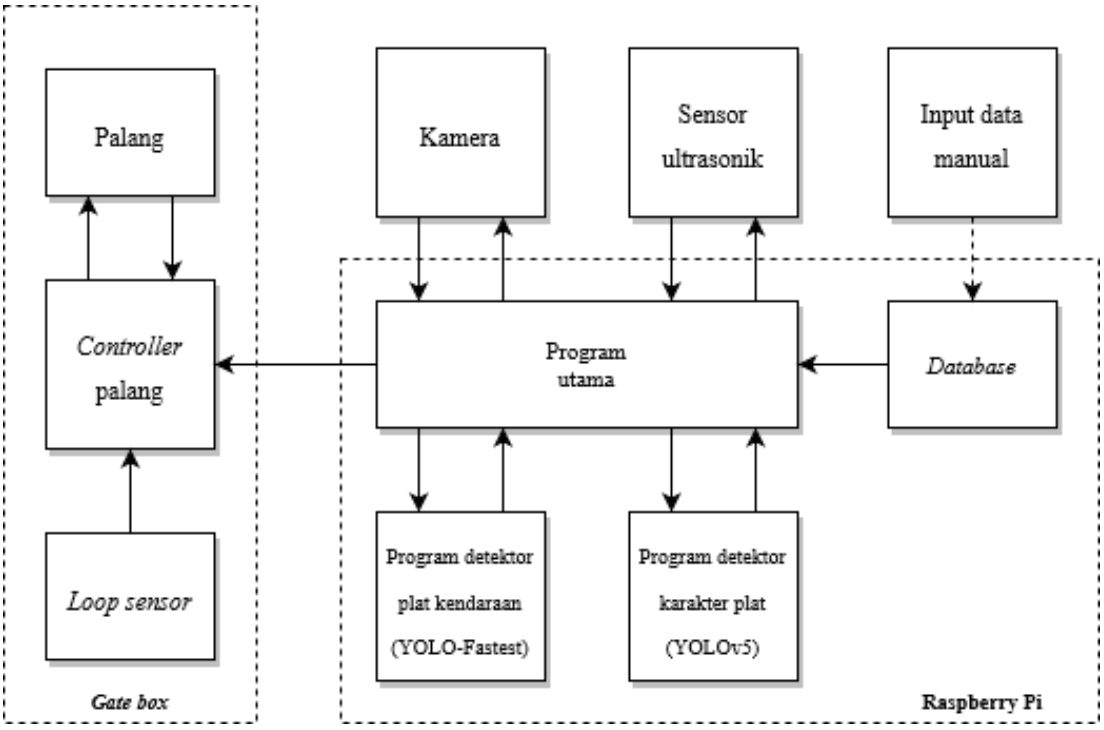
waktu *high* 2ms, dan sudut -90° pada SG90 dan -60° pada MG995 pada waktu *high* 1ms (*31150-MP - MG995 High Speed Servo Actuator*, n.d.; *Servo Motor SG90*, n.d.).

2.7. Penelitian Terkait

D. Djamaluddin (Djamaluddin, 2021) menjelaskan bahwa sistem palang yang saat ini digunakan di Departemen Teknik Elektro, Fakultas Teknik, Universitas Hasanuddin menggunakan sistem RFID, di mana pengguna diharuskan untuk menempelkan kartu pada card reader yang ada di jalur keluar dan masuk lahan parkir. Lubna, N. Mufti, dan S. A. A. Shah (Lubna et al., 2021) menyebutkan bahwa penggunaan sistem ALPR memiliki kelebihan di mana sistem tidak memerlukan bagian *transponder* yang dipasang ke kendaraan pengguna, seperti pada sistem RFID UHF. Apabila dibandingkan dengan sistem ALPR, sistem RFID memerlukan biaya tambahan untuk keperluan pembuatan *tag* baru tiap penambahan entri baru pada *database* dilakukan. Penggunaan sistem ALPR juga tidak memerlukan pengguna untuk menempelkan kartu yang dibawa untuk keperluan identifikasi diri.

BAB III
PERANCANGAN SISTEM

3.1. Rancangan Umum

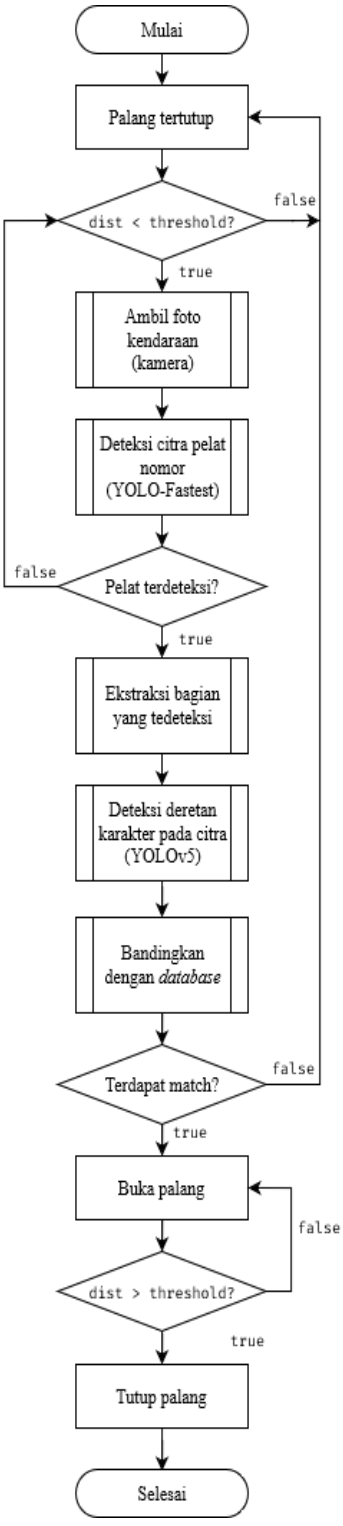


Gambar 3.1: Blok diagram kerja sistem *smart gate*.

Pada tugas akhir ini, dirancang sebuah sistem deteksi plat kendaraan untuk stimulasi buka-tutup palang lahan parkir yang diimplementasikan dengan menggunakan teknologi deteksi objek pada bidang studi kecerdasaan buatan, yang kemudian dinamakan sebagai sistem *smart gate*.

Seperti yang dapat dilihat pada blok diagram sistem pada **Gambar 3.1.**, sistem dapat dianggap terdiri atas dua bagian: *gate box* yang memiliki *controller*. *Controller* pada kotak *gate* berfungsi menerima status *loop sensor* palang (ada atau tidaknya kendaraan yang melintas) dan juga berfungsi menggerakkan palang. *Controller* ini pula yang menerima perintah dari luar (misalnya mikrokontroler atau komputer) untuk menggerakkan dan menghentikan gerak palang, sehingga gerakan palang dapat dikendalikan melalui perintah yang dikirim ke *controller* ini. Perangkat komputer

Raspberry Pi pada sistem *smart gate* ini terhubung dengan kamera dan sensor ultrasonik, dan berisi program utama sistem, *database* sistem, serta algoritma-algoritma deteksi objek yang dibutuhkan. *Database* sistem diisi secara manual oleh *progammer*. Pada program utama di komputer Raspberry Pi, diprogram perintah ke sensor-sensor dan algoritma-algoritma deteksi objek, data yang kemudian diterima dari sensor dan algoritma tersebut, serta data dari *database.* Program utama kemudian memproses data yang diterima, setelah itu dihasilkan keluaran yang diteruskan ke *controller* palang.



Gambar 3.2: *Flowchart* kerja sistem smart gate.

Seperti yang dapat dilihat pada diagram alir kerja sistem pada **Gambar 3.2.**, alur kerja sistem ini dimulai dengan kondisi palang tertutup. Apabila jarak yang dibaca oleh sensor ultrasonik lebih kecil dari ambang batas (*threshold*) tertentu, kamera akan mengambil citra kendaraan yang menyebabkan pencapaian nilai ambang batas tersebut. Menggunakan bantuan arsitektur deteksi objek YOLO-Fastest, lokasi plat nomor pada citra yang telah diambil oleh kamera kemudian dideteksi, dan citra dipotong (di-*crop*) sehingga menyisakan bagian plat nomor yang terdeteksi itu sendiri. Setelah proses ekstraksi tersebut selesai, YOLOv5 kemudian digunakan untuk membaca plat nomor yang telah diekstraksi. Apabila deretan karakter yang terbaca oleh metode AI yang digunakan tersimpan dalam sebuah *database*, maka palang akan dibuka. Apabila tidak, maka palang tidak akan dibuka. Palang kemudian kembali ditutup setelah kendaraan meninggalkan lokasi, yang ditandai dengan sensor ultrasonik kembali membaca nilai yang lebih besar dari ambang batas yang telah ditentukan.

3.2. Perancangan Perangkat Keras Sistem

Pada tahap perancangan perangkat keras ini, dibangun lapangan parkir pada prototipe *smart gate,* yang terdiri atas tiga miniatur tempat parkir kendaraan, di mana masing-masing tempat parkir dibatasi oleh *gate* yang penggeraknya berupa motor servo. Sensor ultrasonik JSN-SR04T digunakan untuk sensor jarak yang menyimulasikan akan adanya kendaraan yang berhenti di depan tempat parkir. Satu buah sensor citra OV5647 digunakan untuk mengambil citra. Seluruh komponen elektronik yang digunakan dihubungkan ke komputer Raspberry Pi 4 Model B, yang berfungsi sebagai pengendali komponen elektronik pada prototipe *smart gate*. Pada **Tabel 3.1.** berikut dapat dilihat bahan-bahan yang digunakan untuk membangun perangkat keras prototipe *smart gate*.

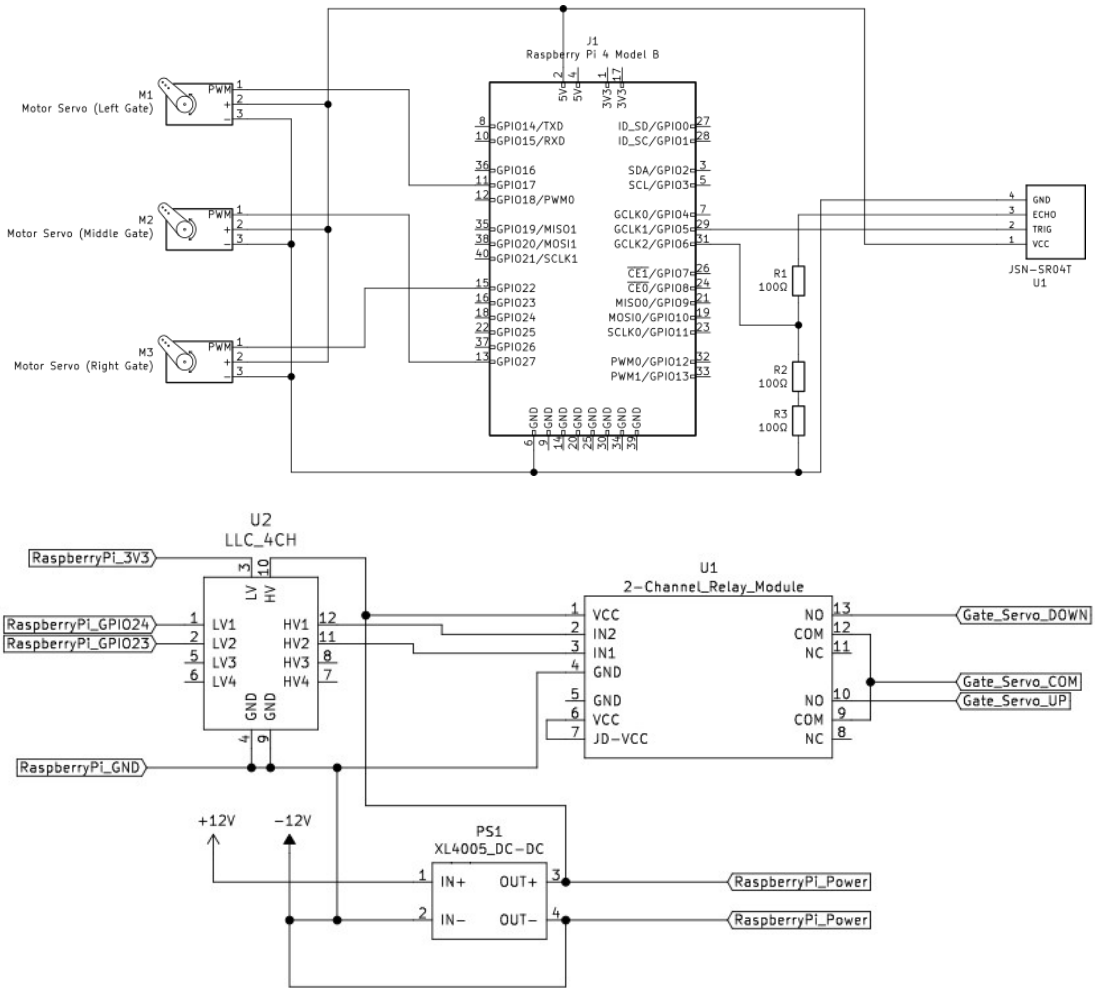
Tabel 3.1: Bahan yang digunakan

No.	**Bahan**	**Keterangan**
1.	Raspberry Pi 4	Sebagai komputer sistem *smart gate*.

No.	Bahan	Keterangan
2.	Motor servo	Sebagai mekanisme *gate*.
3.	Sensor citra OV5647	Untuk mengambil citra plat nomor kendaraan pengguna sistem.
4.	Sensor ultrasonik JSN-SR04T	Sebagai bagian dari mekanisme *gate*, yang menentukan apakah *gate* dapat ditutup setelah pengguna meninggalkan lokasi *gate*.
5.	Gabus	Sebagai bahan untuk pembangunan miniatur tempat parkir.

3.2.1. Perancangan Sistem Elektronik

Skematik sistem *smart gate* dibuat menggunakan perangkat lunak KiCad, dan dapat dilihat pada **Gambar 3.3.**, yang secara berurutan menampilkan skematik prototipe sistem, dan skematik sistem pada lahan parkir gedung Departemen Teknik Elektro, Universitas Hasanuddin. Pin PWM ketiga motor servo, yang masing-masing merepresentasikan *gate* kiri, tengah, dan kanan, secara berurutan disambung ke pin GPIO17, GPIO27 dan GPIO22 pada komputer Raspberry Pi. Pin TRIG pada sensor ultrasonik JSN-SR04T disambung ke pin GPIO5 pada Raspberry Pi, dan *voltage divider* digunakan pada sambungan pin ECHO ke pin GPIO6. *Voltage divider* ini digunakan untuk *step-down* tegangan 5V antara pin ECHO dan GND (pin 8 pada Raspberry Pi) ke 3.3V antara pin GPIO 6 dengan GND. Tiga resistor 100Ω digunakan untuk rangkaian *voltage divider* tersebut.



Gambar 3.3: Atas: skematik prototipe sistem smart gate. Bawah: skematik sistem smart gate saat diuji di tempat parkir gedung

Selain perangkat motor servo dan sensor ultrasonik yang disambung ke pin-pin GPIO pada Raspberry Pi 4, terdapat pula sensor citra OV5647 yang disambung ke *slot* kabel CSI-2 yang terdapat pada komputer Raspberry Pi 4. Sensor citra dan sensor ultrasonik pada sistem disimpan berdampingan.

3.3. Perancangan Perangkat Lunak Sistem

Pada tahap ini, dilakukan pembangunan program yang dibutuhkan agar alur kerja sistem seperti yang telah dijelaskan pada bagian **3.1.** dapat dilakukan. Perancangan perangkat lunak sistem dapat dibagi menjadi beberapa tahap: pembuatan model untuk metode AI yang berfungsi sebagai detektor lokasi plat nomor pada citra dan pembaca plat nomor tersebut, pembuatan program yang

menangani pergerakan ketiga motor servo yang digunakan sebagai palang, pembuatan program yang menangani kerja sensor ultrasonik untuk deteksi akan adanya kendaraan, dan pembuatan *script* yang menghubungkan kerja sensor ultrasonik, sensor citra, dan mekanisme motor servo.

Pembuatan model untuk metode AI itu sendiri terdiri dari beberapa tahap. Pembangunan *dataset* pertama-tama dilakukan dengan mengumpulkan berbagai foto plat nomor sepeda motor. Foto-foto yang didapatkan kemudian dianotasi. Anotasi dilakukan sebanyak dua kali: yang pertama untuk membangun *dataset* yang digunakan untuk melatih arsitektur YOLOv5 agar dapat mendeteksi lokasi plat nomor kendaraan pada citra, dan yang kedua untuk membangun *dataset* yang digunakan untuk melatih arsitektur YOLOv5 agar dapat mendeteksi deretan karakter yang dimiliki plat nomor yang telah dideteksi. Proses pelatihan kemudian dilakukan menggunakan kedua *dataset* tersebut, sehingga dihasilkan dua model yang berfungsi sebagai detektor lokasi plat nomor dan sebagai pembaca plat nomor. Pelatihan digunakan menggunakan bantuan Kaggle, yang menyediakan modul GPU Tesla P100-PCIE-16GB tanpa biaya untuk kepentingan pelatihan metode AI.

3.3.1. Pembangunan *Dataset*

Pembangunan *dataset* untuk detektor objek-objek yang diperlukan pada penelitian ini dapat dibagi menjadi beberapa tahap: pengumpulan data berupa citra yang berisi objek-objek yang ingin dideteksi; anotasi atau pelabelan data yang dimiliki, di mana tiap-tiap citra yang didapatkan diberi label berupa *bounding boxes* yang melingkupi objek-objek yang ingin dideteksi, serta pembagian *dataset* yang telah dianotasi ke dalam kategori *train* dan *test*; dan augmentasi data, di mana citra-citra *dataset* dalam kategori *train* dimanipulasi sehingga data yang dimiliki untuk pelatihan lebih banyak jumlahnya. Augmentasi citra diimplementasikan menggunakan bahasa pemrograman Python 3 dengan bantuan *library* `albumentations`, dan anotasi *dataset* dilakukan menggunakan bantuan perangkat lunak cvat.

3.3.1.1. Anotasi Data

Anotasi data tiap-tiap citra pada sebuah *dataset* disimpan sebagai *text file*, di mana tiap-tiap citra pada *dataset* memiliki satu *text file*-nya sendiri, yang menyimpan label objek-objek pada citra tersebut. Anotasi data pada YOLOv5 memiliki bentuk:

$$class, x, y, w, h$$

di mana $class$ adalah *class* sebuah objek, x dan y adalah titik tengah *bounding box*, dan w dan h adalah panjang dan lebar *bounding box* tersebut. Nilai variabel x, y, w, dan h merupakan *floating point* yang berkisar antara 0 hingga 1, sedangkan $class$ memiliki tipe data *integer*.

3.3.1.2. Pembangunan *Dataset* untuk Detektor Plat Nomor

Dataset yang digunakan untuk pembangunan model detektor plat nomor dibuat dari kumpulan citra sepeda motor yang memiliki plat nomor berwarna hitam. Untuk penelitian ini, dikumpulkan 311 citra yang telah dianotasi, yang kemudian dibagi menjadi kumpulan *train* dan *test*. Dari 311 citra yang dimiliki, 248 diantaranya dipilih secara acak untuk dimasukkan pada kategori *train*, dan 63 sisanya dimasukkan pada kategori *test*. Pada 248 citra *train* tersebut, diaplikasikan 6 augmentasi berbeda untuk tiap-tiap citra, sehingga untuk tiap-tiap citra pada *dataset* dihasilkan 6 citra baru yang merupakan augmentasi dari satu citra tersebut. Pada **Tabel 3.2.** dapat dilihat macam-macam augmentasi yang dilakukan pada terhadap *dataset* yang dimiliki.

Tabel 3.2: Enam augmentasi yang diaplikasikan pada satu citra.

No.	Augmentasi yang dilakukan
1.	Ubah *perspective* citra secara acak. Ubah ukuran citra ke 512 × 512 piksel.
2.	Ubah *brightness* citra secara acak. Ubah *perspective* citra secara acak. Ubah ukuran citra ke 512 × 512 piksel.
3.	Blur citra. Ubah *perspective* citra secara acak.

No.	Augmentasi yang dilakukan
	Ubah ukuran citra ke 512 × 512 piksel.
4.	Ubah ukuran citra ke 512 × 512 piksel. Ubah *perspective* citra secara acak. Tambahkan *cutout* pada bagian-bagian acak pada citra.
5.	Ubah *perspective* citra secara acak. *Flip* citra secara horizontal. Ubah ukuran citra ke 512 × 512 piksel.
6.	Ubah ukuran citra ke 512 × 512 piksel.

Setelah augmentasi tiap-tiap citra selesai dilakukan, dimiliki 1488 citra plat nomor dengan ukuran 512 × 512 piksel dengan variasi pada sudut pandang dan *brightness*.

3.3.1.3. Pembangunan *Dataset* untuk Detektor Karakter Plat Nomor

Dataset yang digunakan untuk pembangunan model detektor karakter plat nomor dibuat dari kumpulan citra plat nomor berwarna hitam. Untuk penelitian ini, dikumpulkan 1025 citra plat nomor yang telah dianotasi, yang kemudian dibagi menjadi kumpulan *train* dan *test*. Dari 1025 citra yang dimiliki, 799 diantaranya dipilih secara acak untuk dimasukkan pada kategori *train*, dan 226 sisanya dimasukkan pada kategori *test*. Pada 799 citra *train* yang dimiliki, augmentasi data dilakukan, pertama-tama dengan memutar tiap-tiap citra sebanyak 30° dan -30°, sehingga akhirnya dimiliki 2397 citra pada kategori *train*. Pada 2397 citra tersebut, diaplikasikan 6 augmentasi berbeda untuk tiap-tiap citra, sehingga untuk tiap-tiap citra pada *dataset* dihasilkan 6 citra baru yang merupakan augmentasi dari satu citra tersebut. Pada **Tabel 3.3.** dapat dilihat berbagai augmentasi yang dilakukan terhadap *dataset* yang dimiliki.

Tabel 3.3: Enam augmentasi yang diaplikasikan pada satu citra.

No.	Augmentasi yang dilakukan
1.	Ubah *perspective* citra secara acak. Ubah ukuran citra ke 256 × 256 piksel.
2.	Ubah *brightness* citra secara acak.

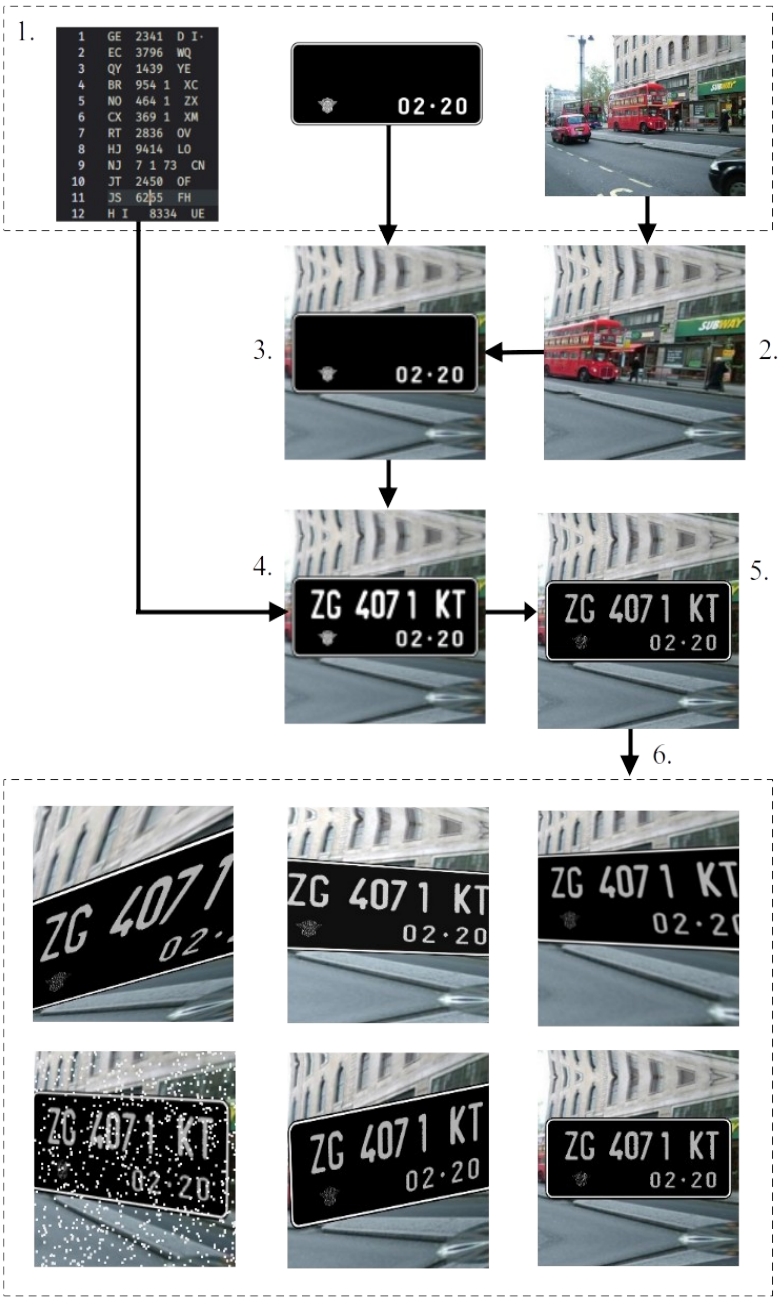
No.	Augmentasi yang dilakukan
	Ubah *perspective* citra secara acak. Ubah ukuran citra ke 256 × 256 piksel.
3.	Blur citra. Ubah *perspective* citra secara acak. Ubah ukuran citra ke 256 × 256 piksel.
4.	Ubah ukuran citra ke 256 × 256 piksel. Ubah *perspective* citra secara acak. Tambahkan *cutout* pada bagian-bagian acak pada citra.
5.	Ubah *perspective* citra secara acak. Ubah ukuran citra ke 256 × 256 piksel.
6.	Ubah ukuran citra ke 256 × 256 piksel.

Setelah augmentasi tiap-tiap citra selesai dilakukan, dimiliki 14,382 citra plat nomor dengan ukuran 256 × 256 piksel dengan variasi pada sudut pandang, *brightness*, dan kemiringan.

3.3.1.3.1. Pembangunan Dataset Sintesis

Selain *dataset* yang disusun dari foto-foto plat nomor kendaraan di dunia nyata, dibangun pula *dataset* sintesis (*synthtetic dataset*) untuk membantu proses *learning* arsitektur. Tahap ini diperlukan karena label untuk beberapa karakter pada *dataset* yang dibangun dari data dunia nyata lebih sedikit banyaknya dibandingkan label untuk karakter lain.

Proses *dataset generation* ini dimulai dari pembuatan: 10,000 *strings* yang memiliki pola `huruf-huruf-angka-angka-angka-angka-huruf-huruf`—sesuai dengan pola plat nomor kendaraan di Indonesia—di mana nilai tiap-tiap huruf dan angka pada *strings* tersebut di-*generate* secara acak; citra plat nomor hitam kosong, di mana 10,000 *strings* yang telah di-*generate* tadi akan ditempelkan (di-*paste*) ke citra plat nomor kosong tersebut, menggunakan *font face* yang telah di-*trace* dari citra plat nomor yang diambil di dunia nyata; dan *background* untuk citra plat nomor sintesis yang telah dibuat. Alur kerja *synthetic dataset generation* ini dapat dilihat pada **Gambar 3.4.**.



Gambar 3.4: Alur kerja *synthetic data generation*: (1) Pembuatan *strings*, citra plat kosong dan *background*, (2) *background* dibuat agar memiliki panjang dan lebar yang sama, (3) penempelan citra plat pada *background*, (4) penempelan *string* pada *background*, (5) aplikasi filter *emboss* dan *spread* pada citra,(6) augmentasi data sintesis yang dihasilkan, mengikuti **Tabel 3.3.**

Kumpulan *background* yang digunakan untuk *dataset generation* ini diambil dari koleksi *Stanford Background Dataset (Gould et al., 2009)*, yang berisi 715 citra berukuran sekitar 320 × 240 piksel. *Dataset* ini didesain untuk keperluan *semantic segmentation*, sehingga selain kumpulan citra, terdapat pula label yang menandai berbagai objek yang terdapat pada masing-masing citra. Pada penelitian ini hanya diperlukan citra-citra pada *dataset*, dan kumpulan labelnya tidak digunakan. Masing-masing citra kemudian diubah ukurannya agar memiliki panjang dan lebar yang sama.

Looping terhadap 10,000 *strings* yang dimiliki kemudian dilakukan. Pada tiap *loop*, diambil satu dari 715 citra *background* secara acak, dan *string* pada tiap-tiap *loop* kemudian ditempelkan ke citra *background* yang terpilih. Anotasi tiap-tiap karakter pada citra juga dilakukan pada tahap ini. Setelah *loop* ini selesai, didapatkan 10,000 citra plat nomor sintesis beserta labelnya masing-masing. Filter *emboss* dan *spread* pada perangkat lunak GIMP kemudian diaplikasikan pada tiap-tiap data sintesis ini. *Emboss* digunakan untuk memberikan efek tiga dimensi pada citra, dan *spread* untuk menambahkan *noise* pada citra. Setelah filter-filter tersebut diaplikasikan, dilakukan augmentasi yang langkah-langkahnya sama seperti pada **Tabel 3.3.**. Hasil akhir dari *dataset generation* ini adalah sebuah *dataset* yang berisi 60,000 citra plat nomor sintesis, dengan ukuran 256 × 256 piksel dan variasi pada sudut pandang dan *brightness*. Seluruh citra pada *dataset* ini ditambahkan ke 14,382 citra *train* yang dimiliki, sehingga secara keseluruhan, dimiliki 74,382 citra untuk *training* detektor karakter plat nomor.

3.3.2. Pelatihan Arsitektur Deteksi Objek

Setelah *dataset* yang diperlukan selesai dibangun, pelatihan dilakukan menggunakan bantuan Kaggle, yang menyediakan modul GPU Tesla P100-PCIE-16GB untuk kepentingan pelatihan metode AI. Pada pelatihan detektor objek YOLO-Fastest, ukuran batch yang digunakan adalah 16, jumlah *epoch* yang dilalui disetel ke 150. Pada pelatihan detektor karakter plat nomor, digunakan arsitektur YOLOv5s. *Batch size* yang digunakan adalah 16, dan banyak *epoch* yang dilalui adalah 100 untuk detektor plat nomor, dan 10 untuk detektor karakter. *Hyperparameter* fliplr dan

flipud, yang menentukan probabilitas YOLOv5 agar mem-*flip* citra yang digunakan untuk *training* secara horizontal dan vertikal, dibuat nol. Nilai fliplr yang digunakan saat pelatihan detektor plat nomor adalah 0.5, yang merupakan nilai *default* dari *hyperparameter* tersebut. Untuk *hyperparameters* yang lainnya digunakan nilai *default*.

Setelah pelatihan detektor objek selesai dilakukan, model yang dihasilkan dikonversi ke bentuk model yang kompatibel dengan *framework* ncnn.

3.3.3. Pemrograman Komponen Elektronik Sistem

Komponen-komponen yang digunakan pada sistem *smart-gate* ini adalah sensor ultrasonik, tiga motor servo dan satu sensor citra. Untuk komunikasi terhadap sensor ultrasonik dan ketiga motor servo melalui pin-pin GPIO, digunakan bahas pemrograman Python 3 dengan bantuan *library* RPi.GPIO. Pengambilan citra oleh kamera dilakukan menggunakan perintah `raspistill`, yang merupakan perintah bawaan dari Raspberry Pi:

```
raspistill -o 'path/to/save_directory'
```

Mekanisme *gate* dibuat sebagai modul Pythonnya sendiri, yang disimpan dengan nama berkas `gate.py`. Bagian terpenting modul ini adalah tiga fungsi untuk masing-masing motor servo. Untuk dapat mengirim sinyal PWM, tiap-tiap pin yang disambungkan ke pin PWM motor servo di-*set* ke mode PWM:

```
servo = GPIO.PWM(pin, freq)
```

di mana pin adalah nomor pin GPIO yang disambung ke pin PWM motor servo (17, 27 dan 22 untuk *gate* kiri, tengah dan kanan), dan freq adalah frekuensi motor servo (50Hz untuk motor servo Tower Pro SG90 dan Tower Pro MG995). Pengaturan sudut motor servo dilakukan dengan mengubah *duty cycle* sinyal PWM yang dikirim menggunakan metode `ChangeDutyCycle()`. Hubungan antara sudut yang diinginkan (angle) dengan *duty cycle* servo adalah:

```
servo.ChangeDutyCycle(2+(angle/18))
```

Untuk sensor ultrasonik, pemancaran gelombang suara dilakukan dalam rentang waktu 0.01ms. Metode time.time() kemudian digunakan untuk menghitung waktu rambat gelombang dari sensor hingga saat gelombang kembali ke sensor:

```
GPIO_TRIGGER = 5; GPIO_ECHO = 6

GPIO.setup(GPIO_TRIGGER, GPIO.OUT)
GPIO.setup(GPIO_ECHO, GPIO.IN)

GPIO.output(GPIO_TRIGGER, True)
time.sleep(0.00001) # pancarkan gelombang suara selama 0.01ms
GPIO.output(GPIO_TRIGGER, False)

startTime = time.time()
stopTime = time.time()

while GPIO.input(GPIO_ECHO) == 0:
    startTime = time.time()
while GPIO.input(GPIO_ECHO) == 1:
    stopTime = time.time()
# hitung waktu rambat gelombang
distance = ((stopTime - startTime) * 34300) / 2
```

Nilai *c* yang digunakan pada kalkulasi waktu rambat sensor adalah 34300cm/s.

3.3.4. Penyatuan Seluruh Proses Menjadi Satu Sistem

Setelah kerja detektor-detektor objek, sensor ultrasonik, sensor citra dan mekanisme servo selesai diprogram, kerja tiap-tiap proses disatukan menjadi sebuah sistem *smart gate*. Perintah yang menjalankan fungsi detektor dan pembacaan karakter disimpan pada sebuah *script* bash dengan nama `readphoto.sh`, yang dijalankan melalui perintah `bash readphoto.sh`. Mekanisme palang dibuat sebagai modul Pythonnya sendiri, yang disimpan sebagai berkas dengan nama `gate.py`. Penyatuan kedua proses tersebut, beserta kerja sensor ultrasonik dan sensor citra, dilakukan seperti berikut:

```
import RPi.GPIO as GPIO
import time
from tools.gate_miniature import *
from tools.ultrasonic_sensor import *
from tools.check_database import *

if __name__ == '__main__':
  print("SmartGate starting...")
  path = "/home/pi/Programs/smartgate/detected_chars.txt"
  plate = ""
  try:
    while True:
      dist_entrance = distance(5, 6) # distance for entrance vehicle
sensor
      print ("Measured Distance = %.1f cm" % dist_entrance)
      time.sleep(1)
      if dist_entrance < 140:
        import subprocess
        import glob
        cmd_takepic = "raspistill -t 500 -o ./tmp/cam.jpg -q 10 -th
none -x none"
        print(f"Photo taken at {dist_entrance}")
        subprocess.run(cmd_takepic.split(), capture_output=True)
        print("Photo taken")
        print("Processing...")

        cmd_process = "bash analyse_photo.sh"
        processing = subprocess.run(cmd_process.split(),
capture_output=True, cwd="/home/pi/Programs/smartgate")
        files_list = glob.glob(path)
        if len(files_list) == 0:
          print("No licence plate detected.")
          #beep(5)
          continue
        for filename in glob.glob(path):
          with open(filename, 'r') as f:
            for line in f:
              plate = line
```

```
        plate_exists, gate_no = check_existence(plate)
        if plate_exists:
          print(f"Plate {plate} accepted.")
          if(gate_no == 3):
            barrier_left(1)
            open_gate = "left"
          if(gate_no == 2):
            barrier_mid(1)
            open_gate = "middle"
          if(gate_no == 1):
            barrier_right(1)
            open_gate = "right"
        else:
          print(f"Plate {plate} rejected.")
          #beep(5)
          #close_gate()
          continue
      else:
        continue

      while True:
        dist_mid_sensor = distance(5, 6)
        time.sleep(1)
        print(f"After gate opened: {dist_mid_sensor}")
        if dist_mid_sensor > 140:
          # close gate after vehicle leaves for 50 seconds
          barrier_left(0); barrier_mid(0); barrier_right(0)
          #close_gate()
          break

  # Exit by pressing Ctrl-C
  except KeyboardInterrupt:
    clean_gate()
    print("SmartGate stopped manually.")
```

Pada *script* Python 3 di atas, kerja sensor ultrasonik diprogram sebagai fungsi distance(). Perintah raspistill yang digunakan untuk mengambil citra kendaraan

dijalankan melalui fungsi `subprocess.run()`, yang menerima *list* berisi deretan karakter yang dimiliki perintah yang ingin dijalankan sebagai masukan. Sebagai contoh, perintah `ls -la` dijalankan melalui `subprocess.run()` sebagai `subprocess.run(['ls', '-la'])`, maka dari itu perintah-perintah yang dijalankan pada disimpan sebagai *string* yang kemudian di-*split* menjadi *list* menggunakan metode `split()`. `barrier_left(0)`, `barrier_mid(0)` dan `barrier_right(0)` masing-masing merupakan fungsi yang di-*import* dari modul gate.py. Program juga akan terus berjalan, dan hanya akan dihentikan apabila terdapat *exception* berupa `KeyboardInterrupt`, yang terjadi apabila pengguna menekan tombol *interrupt* pada keyboard (misalnya `Ctrl-C`). Pada penelitian ini, *script* Python di atas disimpan dengan nama berkas `smartgate.py`.

3.4. Perancangan Pengujian

Pengujian dilakukan pada Laboratorium *Cognitive Social Robotics and Advanced Artificial Intelligence Research Centre* yang bertempat di Departemen Elektro, Fakultas Teknik, Universitas Hasanuddin, dan dilakukan melalui dua tahap: uji prototipe alat, dan uji rancang bangun di palang lahan parkir Departemen.

3.4.1. Rancangan Pengujian Prototipe

Pada tahap pertama pengujian ini, digunakan 21 foto plat nomor kendaraan yang dicetak ke kertas A4, yang ditaruh di depan sensor ultrasonik prototipe dengan sudut 30° dari kamera. Variabel-variabel yang diobservasi pada pengujian adalah deretan karakter yang terbaca serta waktu yang diambil sistem dari saat sistem merespon terhadap *obstacle* di depan sensor ultrasonik hingga saat salah satu dari ketiga servo diputar.

Pada tahap ini pula perbandingan kinerja sistem ketika data sintesis tidak digunakan, dengan sistem yang menggunakan data sintesis dibandingkan. Variabel yang dibahas pada tahap ini adalah waktu dan akurasi sistem yang dihasilkan.

3.4.2. Rancangan Pengujian pada Lahan Parkir Departemen

Pada tahap kedua pengujian ini, dipasang sensor ultrasonik dan perangkat lunak sistem ke komputer sistem palang di lahan parkir Departemen Teknik Elektro, di mana sistem diuji dengan menaruh kamera dan sensor ultrasonik sistem di tengah palang, sehingga kamera berada pada posisi bersudut 90° dari bagian depan kendaraan. 3 mobil dan 3 sepeda motor digunakan pada tahap kedua pengujian ini, dan diambil pada waktu dan kondisi cuaca yang berbeda-beda. Pada tahap kedua ini diobservasi deretan karakter yang terbaca dan waktu yang diambil sistem dari saat sistem merespon terhadap *obstacle* di depan sensor ultrasonik hingga saat palang digerakkan.

3.5. Lokasi Penelitian

Penelitian tugas akhir ini dilaksanakan di Laboratorium *Cognitive Social Robotics and Advanced Artificial Intelligence Research Centre* yang bertempat di Departemen Elektro, Fakultas Teknik, Universitas Hasanuddin.

BAB IV
HASIL PENGUJIAN DAN PEMBAHASAN

Pada bab ini dibahas hasil pengujian prototipe, yang dilakukan di dalam ruangan laboratorium, dan pengujian ketika prototipe diaplikasikan di lahan parkir Departemen Teknik Elektro, Fakultas Teknik, Universitas Hasanuddin.

4.1. Pengujian Prototipe Sistem

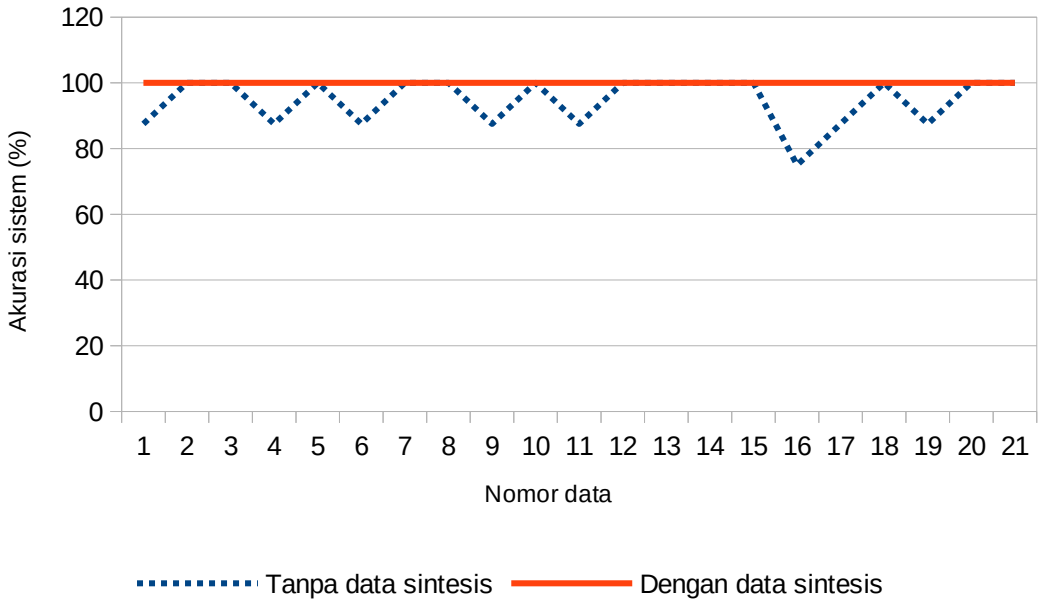
Tabel 4.1: Hasil pengujian prototipe sistem.

No.	Karakter pelat nomor	Hasil deteksi		Akurasi		Waktu (s)	
		Tanpa d.s.	Dengan d.s.	Tanpa d.s.	Dengan d.s.	Tanpa d.s.	Dengan d.s.
1.	DW4029CD	DW4029D	DW4029CD	87.5%	100%	5.02	4.99
2.	DW4365AW	DW4365AW	DW4365AW	100%	100%	5.08	5.04
3.	DD4723QB	DD4723QB	DD4723QB	100%	100%	4.97	4.97
4.	DD5741DO	DD5741D	DD5741DO	87.5%	100%	5.05	5.03
5.	DD6511QQ	DD6511QQ	DD6511QQ	100%	100%	4.99	5.04
6.	DD2057RO	DD2057R	DD2057RO	87.5%	100%	5.1	5.01
7.	DD4032SV	DD4032SV	DD4032SV	100%	100%	5.02	5.05
8.	DD4634VT	DD4634VT	DD4634VT	100%	100%	5.03	4.97
9.	DD2430RP	DD2430PP	DD2430RP	87.5%	100%	5.09	5
10.	DD5226UB	DD5226UB	DD5226UB	100%	100%	5.03	4.98
11.	DW2900AO	DW2900A	DW2900AO	87.5%	100%	5.1	5
12.	DD2174QR	DD2174QR	DD2174QR	100%	100%	4.99	5.07
13.	DP5350BF	DP5350BF	DP5350BF	100%	100%	4.98	5.08
14.	DD6267TI	DD6267TI	DD6267TI	100%	100%	5	5.04
15.	DD3170XP	DD3170XP	DD3170XP	100%	100%	4.97	5
16.	DD5882MV	DD5882W	DD5882MV	75%	100%	5.2	5.02
17.	DP2803FG	DP2803F	DP2803FG	87.5%	100%	5.2	5.06
18.	DP5396CK	DP5396CK	DP5396CK	100%	100%	5	5.13
19.	DD4496OS	DD4496BS	DD4496OS	87.5%	100%	5.1	4.99
20.	DD2923YF	DD2923YF	DD2923YF	100%	100%	5	5.02
21.	DP3544UN	DP3544UN	DP3544UN	100%	100%	4.99	5.02
			Rata-rata	95%	100%	5.04	5.03

Pada **Tabel 4.1.** di atas, ditampilkan hasil kerja sistem ketika data sintesis (d.s.) digunakan, dan ketika d.s. tidak digunakan. Waktu yang dibutuhkan sistem untuk inferensi (mendeteksi, melokalisasi, dan membaca) pelat nomor hingga membuka palang dengan memutar servo juga dapat dilihat. Akurasi hasil deteksi dihitung dengan membandingkan banyaknya deretan karakter yang terbaca dengan benar

dengan deretan karakter yang seharusnya dihasilkan. Sebagai contoh, pada data no. 19 pada **Tabel 4.1.**, sistem yang tidak menggunakan d.s. membaca tujuh dari delapan karakter dengan benar. Akurasi sistem kemudian dihitung sebagai berikut:

$$\frac{7}{8} * 100\% = 87.5\% \quad .$$



Gambar 4.1: Perbandingan akurasi sistem apabila menggunakan model yang dilatih dengan dan tanpa data sintesis

Dari tabel dan grafik data yang dikumpulkan, dapat dilihat bahwa penggunaan data sintesis pada *training data* untuk pembacaan pelat nomor membantu meningkatkan akurasi kinerja sistem dalam pembacaan pelat nomor kendaraan. Untuk model yang dilatih tanpa menggunakan data sintesis, sistem kadang-kadang gagal mendeteksi keberadaan karakter pada pelat nomor, dan apabila keberadaan karakter terdeteksi, sistem kadang-kadang salah membaca karakter tersebut. Sistem ini memiliki akurasi rata-rata 95%. Penggunaan data sintesis terlihat dapat membantu sistem memitigasi kesalahan baca pelat nomor yang terdeteksi, meningkatkan akurasi rata-rata sistem menjadi 100%. Penggunaan data sintesis juga tidak memiliki pengaruh yang terhadap waktu yang dibutuhkan sistem untuk

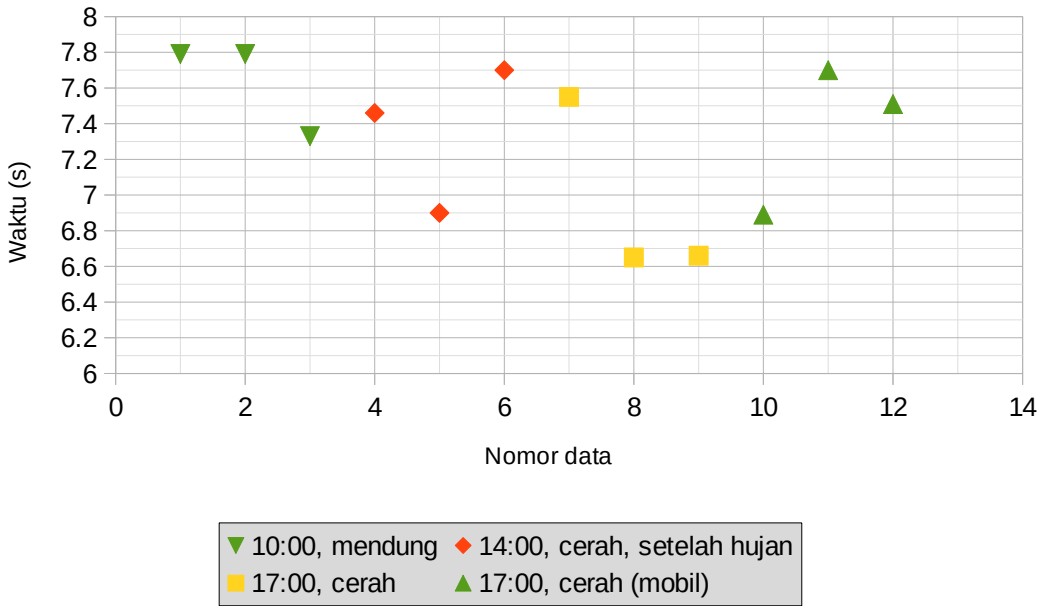
mendeteksi dan membaca pelat nomor; sistem yang *training data*-nya tidak menggunakan data sintesis membutuhkan rata-rata waktu 5.04 detik untuk dapat mendeteksi dan membaca pelat nomor, sedangkan apabila data sintesis digunakan, waktu yang dibutuhkan sistem memiliki rata-rata 5.03 detik.

4.2. Pengujian Sistem pada Lahan Parkir Departemen

Pada tahap ini, pengujian penggunaan plat nomor sepeda motor dilakukan pada tiga waktu berbeda: pukul 10:00 WITA dalam keadaan cuaca mendung, pukul 14:00 WITA saat cuaca cerah, setelah hujan, dan pukul 17:00 WITA dalam keadaan cuaca cerah. Pengujian plat nomor mobil juga dilakukan pada pukul 17:00 WITA sore dalam keadaan cuaca cerah. Data sintesis digunakan pada tahap pengujian ini.

Tabel 4.2: Hasil pengujian pada lahan parkir Departemen.

No.	Karakter plat nomor	Hasil deteksi	Akurasi	Waktu	Waktu, cuaca	Jenis kendaraan
1.	DD6325MG	DD6325MG	100%	7.79	10:00 WITA, mendung	Sepeda motor
2.	DD4764KR	DD4764KR	100%	7.79		
3.	DD2717RO	DD2717RO	100%	7.33		
4.	DD4764KR	DD4764KR	100%	7.46	14:00 WITA, mendung	
5.	DD5449VQ	DD5449VQ	100%	6.9		
6.	DD6325MG	DD6325MG	100%	7.7		
7.	DD6031QJ	DD6031QJ	100%	7.55	17:00 WITA, cerah	
8.	DD6325MG	DD6325MG	100%	6.65		
9.	DD4764KR	DD4764KR	100%	6.66		
10.	DD1642RD	DD1642RD	100%	6.89		Mobil
11.	DD1274AG	DD1274AG	100%	7.7		
12.	DD1923RG	DD1923RG	100%	7.51		
Rata-rata			100%	7.32		



Gambar 4.2: Grafik waktu kerja sistem pada lahan parkir Departemen.

Berdasarkan data yang didapatkan selama pengujian, dapat dilihat bahwa sistem membutuhkan waktu antara 6.6 hingga 7.8 detik untuk mengolah citra dan membuka palang. Perbedaan waktu dan cuaca pengujian tidak memiliki dampak terhadap performa sistem, yang dapat dilihat dari sistem yang membutuhkan waktu yang lebih lama pada pengujian dengan satu pelat nomor, namun lebih cepat pada pengujian dengan pelat nomor lain, meski kedua pengujian tersebut dilakukan pada waktu dan cuaca yang sama.

BAB V
PENUTUP

5.1. Kesimpulan

1. Implementasi fungsi pengenalan plat kendaraan bermotor secara otomatis (*automatic license plate recognition*, ALPR) dapat dilakukan dengan memanfaatkan kombinasi algoritma deteksi objek YOLO-Fastest dan YOLOv5, di mana algoritma YOLO-Fastest digunakan sebagai detektor pelat nomor kendaraan, dan algoritma YOLOv5 digunakan untuk membaca pelat nomor yang telah dideteksi. Gabungan data dari dunia nyata dan data sintesis juga digunakan untuk melatih model detektor karakter pelat nomor kendaraan.
2. Fungsi ALPR yang telah dibangun dapat diintegrasikan dengan mekanisme buka-tutup palang parkir dengan menggunakan komputer Raspberry Pi 4, sensor ultrasonik JSN-SR04T, dan kamera OV5647. Sensor ultrasonik digunakan untuk mendeteksi adanya kendaraan yang berhenti di depan palang parkir; kamera digunakan untuk mengambil citra kendaraan yang berhenti; dan komputer Raspberry Pi 4 digunakan sebagai *controller* di mana program yang menjalankan fungsi-fungsi sensor dan fungsi ALPR disimpan dan dijalankan.
3. Penggunaan data sintesis membantu meningkatkan akurasi sistem ALPR dalam pembacaan pelat nomor sebanyak 5%. Waktu yang dibutuhkan sistem dengan dan tanpa bantuan data sintesis untuk mendeteksi plat nomor kendaraan juga relatif sama, yaitu sekitar 5 detik. Perbedaan waktu dan cuaca pengujian (dengan batasan masalah di mana waktu malam diabaikan) tidak memiliki dampak terhadap performa sistem, yang dapat dilihat dari sistem yang membutuhkan waktu yang lebih lama pada pengujian dengan satu pelat nomor, namun lebih cepat pada pengujian dengan pelat nomor lain, meski kedua pengujian tersebut dilakukan pada waktu dan cuaca yang sama.

5.2. Saran

1. Sistem stimulasi palang parkir berbasis ALPR yang telah dipresentasikan disarankan untuk diintegrasikan ke sistem palang parkir gedung Departemen Teknik Elektro, Universitas Hasanuddin. Apabila sistem tidak menemukan pelat nomor yang dibaca pada *database* sistem, maka sistem me-*revert* ke penggunaan fungsionalitas RFID untuk memverifikasi identitas pengguna lahan parkir.
2. Ketika prototipe diimplementasikan pada palang parkir, disarankan agar sistem dilengkapi penutup yang dapat mencegah air hujan dari mengenai alat, untuk mencegah kerusakan pada alat yang diakibatkan oleh terkena air.
3. Pengujian sistem tugas akhir ini dilakukan dengan kamera yang dipasang pada posisi tegak lurus terhadap bagian depan kendaraan. Penelitian yang lebih lanjut diperlukan mengenai kinerja sistem apabila kamera dipasang pada sudut tertentu dari bagian depan kendaraan.
4. Pengujian sistem tugas akhir ini dilakukan dengan sensor ultrasonik yang memiliki *threshold* 3 m, di mana kamera akan mengambil citra dan memasukkannya ke algoritma deteksi objek setelah *threshold* tersebut dicapai. Penelitian yang lebih lanjut diperlukan mengenai kinerja sistem apabila *threshold* tersebut ditambah atau dikurangi.

DAFTAR PUSTAKA

31150-MP - MG995 High Speed Servo Actuator. (n.d.). Marlin P. Jones & Assoc., Inc. https://www.mpja.com/download/31150mp.pdf

Al-Mahturi, A., & R. Rahim. (2016). Ultrasonic Sensor for Distance Measurement. *Progress in Process Tomography & Instrumentation System*, *24*, 9–14.

Barrier Gate Manual. (n.d.). OneTech. https://qdigital.mx/content/Barreras/Barreras tradicional/Manual de barrera tradicional- tarjeta 2.0.pdf

Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). YOLOv4: Optimal Speed and Accuracy of Object Detection. *ArXiv*, *abs/2004.10934*.

Börklü, H. R., & Sadık A. Kalyon. (2017). A Design Study of an Innovative Barrier System for Personal Parking Lots. *Gazi University Journal of Science Part A: Engineering and Innovation*, *4*(4), 113–123.

Chaudhuri, M. (2018). Boom barrier to jaywalking. *The Telegraph India*. http://www.telegraphindia.com/west-bengal/boom-barrier-to-jaywalking/cid/1410167

Color CMOS QSXGA (5 megapixel) image sensor with OmniBSITM technology. (n.d.). OmniVision Technologies, Inc. https://cdn.sparkfun.com/datasheets/Dev/RaspberryPi/ov5647_full.pdf

DATASHEET – Raspberry Pi 4 Model B. (2019). raspberrypi.org. https://datasheets.raspberrypi.org/rpi4/raspberry-pi-4- datasheet.pdf.

Djamaluddin, D. (2021). Sosialisasi Penggunaan Parkir Cerdas Pada Departemen Teknik Elektro Universitas Hasanuddin. *Jurnal_Tepat*, *4*(1), 94–100. https://doi.org/10.25042/jurnal_tepat.v4i1.164

dog-qiuqiu. (2021). *dog-qiuqiu/Yolo-FastestV2: V0.2* (V0.2) [Computer software]. Zenodo. https://doi.org/10.5281/zenodo.5181503

Dswilan, S., Harmadi, & Marzuki. (2021). Flood monitoring system using ultrasonic sensor SN-SR04T and SIM 900A. *Journal of Physics: Conference Series*, *1876*(1). https://doi.org/10.1088/1742- 6596/1876/1/012003

Du, S., Ibrahim, M., Shehata, M., & Badawy, W. (2013). Automatic License Plate Recognition (ALPR): A State-of-the-Art Review. *IEEE Transactions on Circuits and Systems for Video Technology*, *23*(2), 311–325. https://doi.org/10.1109/TCSVT.2012.2203741

González, G. Z. (2013). *Radio Frequency Identification (RFID) Tags and Reader Antennas Based on Conjugate Matching and Metamaterial Concepts*.

Gould, S., Fulton, R., & Koller, D. (2009). Decomposing a scene into geometric and semantically consistent regions. *2009 IEEE 12th International Conference on Computer Vision*, 1–8. https://doi.org/10.1109/ICCV.2009.5459211

Lubna, Mufti, N., & Shah, S. A. A. (2021). Automatic Number Plate Recognition:A Detailed Survey of Relevant Algorithms. *Sensors (Basel, Switzerland)*, *21*(9), 3028. PubMed. https://doi.org/10.3390/s21093028

Ma, N., Zhang, X., Zheng, H.-T., & Sun, J. (2018). Shufflenet v2: Practical guidelines for efficient cnn architecture design. *Proceedings of the European Conference on Computer Vision (ECCV)*, 116–131.

Maksimović, M., V. Vujović, Milošević, V., & B. Perišić. (2014). Raspberry Pi as Internet of Things Hardware: Performances and Constraints. *International Conference on Electrical, Electronic and Computing Engineering (IcETRAN)*.

Malawakkang, M. N. (2019). *ATM Beras dengan Sistem Aktifasi RFID* [Undergraduate]. Universitas Hasanuddin.

MIPI Alliance Standard for Camera Serial Interface 2 (CSI-2) Version 1.00 – 29 November 2005. (n.d.). MIPI Alliance.

Mohandes, M., Deriche, M., Ahmadi, H., Kousa, M., & Balghonaim, A. (2016). An Intelligent System for Vehicle Access Control using RFID and ALPR Technologies. *Arabian Journal for Science and Engineering*, *41*. https://doi.org/10.1007/s13369-016-2136-0

Pinckney, N. (2006). Pulse-width modulation for microcontroller servo control. *IEEE Potentials*, *25*(1), 27–29. https://doi.org/10.1109/MP.2006.1635026

Prasetyo, M. L. (2020). *Autentikasi Biometrik Berbasis Face Recognition Menggunakan Metode Convolutional Neural Network untuk Simulasi Barrier Gate System* [Undergraduate, UIN Sunan Ampel]. http://digilib.uinsby.ac.id/43177/3/Mohammad%20Langgeng%20Prasetyo_H76216062.pdf

qiuqiuqiu. (n.d.). *Yolo-FastestV2: Gèng kuài, gèng qīng, yídòng duān kě dá 300FPS, cānshù liàng jǐn 250k [Yolo-FastestV2: Faster, lighter, up to 300FPS on mobile, only 250k parameters]*. https://zhuanlan.zhihu.com/p/400474142

Raspberry Pi Documentation—Raspberry Pi OS. (n.d.). Raspberrypi.org. https://www.raspberrypi.org/documentation/

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, *115*(3), 211–252. https://doi.org/10.1007/s11263-015-0816-y

Servo Motor SG90. (n.d.). http://www.ee.ic.ac.uk/pcheung/teaching/DE1_EE/stores/sg90_datasheet.pdf

Shashirangana, J., Padmasiri, H., Meedeniya, D., & Perera, C. (2021). Automated License Plate Recognition: A Survey on Methods and Techniques. *IEEE Access*, *9*, 11203–11225. https://doi.org/10.1109/ACCESS.2020.3047929

Solawetz, J. (2020). YOLOv5 New Version—Improvements And Evaluation. *Roboflow Blog*. https://blog.roboflow.com/yolov5- improvements-and-evaluation/

Tanim, M. M. Z. (2016). *How does passive RFID works, briefly explained.* https://doi.org/10.13140/RG.2.2.12361.34402

Upton, E. (n.d.). *8GB Raspberry Pi 4 on sale now at $75," Raspberry Pi Blog, 2020*. https://www.raspberrypi.org/blog/8gb- raspberry-pi-4-on-sale-now-at-75/

Yan, B., Fan, P., Lei, X., Liu, Z., & Yang, F. (2021). A Real-Time Apple Targets Detection Method for Picking Robot Based on Improved YOLOv5. *Remote Sensing*, *13*(9). https://doi.org/10.3390/rs13091619

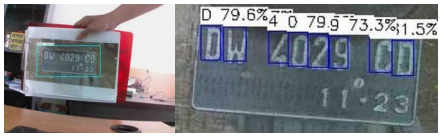
LAMPIRAN

HASIL PENGUJIAN SISTEM

Pada lampiran berikut ditunjukkan citra-citra hasil deteksi sistem.

A.1. Pengujian Prototipe Sistem

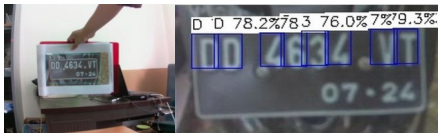
Pada gambar berikut ditunjukkan hasil deteksi prototipe sistem yang diambil pada kondisi laboratorium.



DW4029CD (4.99s)

DD4032SV (5.05s)

DW4365AW (5.04s)



DD4634VT (4.97s)

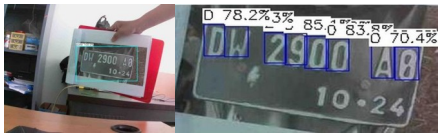
DD4723QB (4.97s)

DD2430RP (5.00s)

DD5741DO (5.03s)

DD5226UB (4.98s)

DD6511QQ (5.04s)

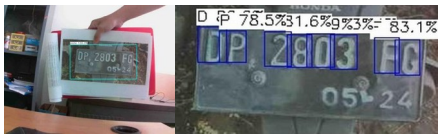


DW2900AO (5.00s)

DD2057RO (5.01s)

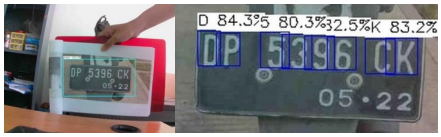
DD2174QR (5.07s)

DP5350BF (5.08s)

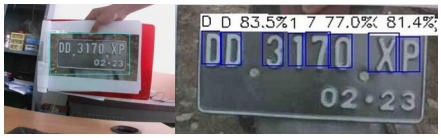


DP2803FG (5.06s)

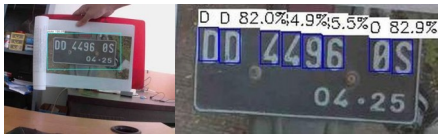
DD6267TI (5.04s)



DP5396CK (5.13s)

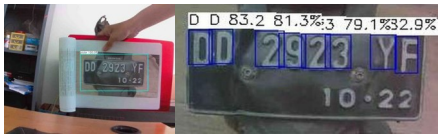


DD3170XP (5.00s)



DD4496OS (4.99s)

DD5882MV (5.02s)



DD2923YF (5.02s)

DP3544UN (5.02s)

Gambar A.1: Hasil pengujian prototipe sistem

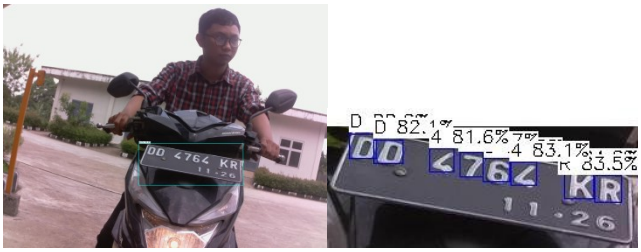
A.2. Pengujian Sistem pada Lahan Parkir Departemen

Pada tahap ini, pengujian penggunaan plat nomor sepeda motor dilakukan pada tiga waktu berbeda: pukul 10:00 WITA dalam keadaan cuaca mendung, pukul 14:00 WITA saat cuaca cerah, setelah hujan, dan pukul 17:00 WITA dalam keadaan cuaca cerah. Pengujian plat nomor mobil juga dilakukan pada pukul 17:00 WITA sore dalam keadaan cuaca cerah.

A.2.1. Pengujian Pada Plat Sepeda Motor

Pukul 10:00, cuaca mendung.

DD6325MG (7.79s)

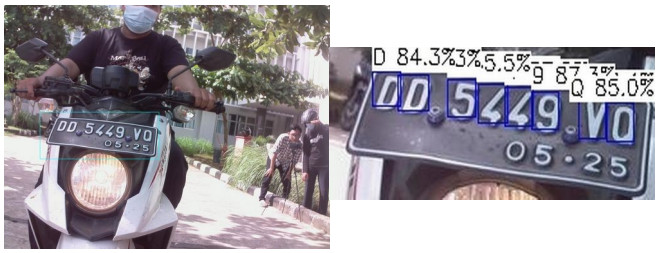


DD4764KR (7.79s)

DD2717RO (7.33s)

Pukul 14:00, cuaca cerah, setelah hujan.

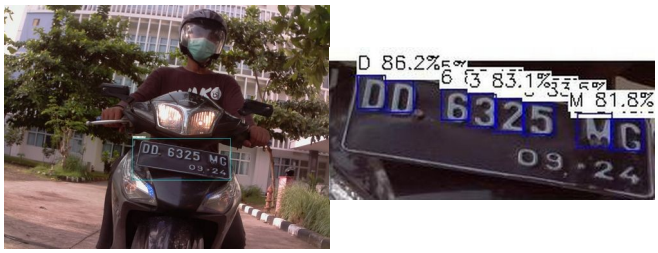
DD4764KR (7.46s)



DD5449VQ (6.90s)

DD6325MG (7.70s)

Pukul 17:00, cuaca cerah.



DD6325MG (6.65s)

DD6031QJ (7.55s)

DD6325MG (6.66s)

Gambar A.2: Hasil pengujian pada plat nomor sepeda motor asli

A.2.2. Pengujian Pada Plat Mobil

Pengujian plat nomor mobil dilakukan pada pukul 17:00 WITA dalam keadaan cuaca cerah.

DD1642RD (6.89s)

DD1274AG (7.7s)

DD1923RG (7.51s)

Gambar A.3: Hasil pengujian pada plat nomor mobil asli