BAB III

ANALISIS DAN PERANCANGAN SISTEM

3.1 Metode Penelitian

Dalam penelitian ini, peneliti membuat metode penelitian terdiri dari empat (4) tahap, yaitu, studi pustaka, perancangan sistem, implementasi sistem, dan pengujian sistem. Tahap studi pustaka adalah tahap untuk peneliti melakukan pencarian referensi terkait masalah yang ingin diteliti. Tahap perancangan sistem adalah tahap untuk peneliti merancang sistem yang diharapkan dapat memecahkan masalah penelitian. Tahap implementasi sistem adalah tahap untuk peneliti mengimplementasikan sistem yang sudah dibuat. Tahap pengujian sistem adalah tahap untuk peneliti melakukan pengujian terkait dengan sistem yang telah dibuat.

Penulis melakukan studi pustaka dengan membaca, mempelajari, dan mengambil poin-poin penting dari penelitian terkait terdahulu, selain itu peneliti juga melakukan studi pustaka melalui website. Perancangan sistem yang penulis usulkan akan dibahas pada sub-bab Bab III. Hasil dan analisis dari implementasi sistem akan dibahas di Bab IV.

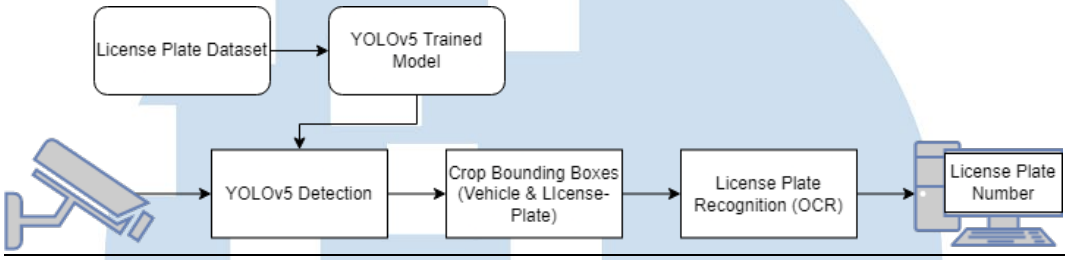
3.2 Perancangan Sistem Secara Umum

Pada penelitian ini, sistem akan dibagi menjadi dua (2) bagian, yaitu, *Vehicle Detection* dan *Plate Number Recognition*.

Pada bagian *Vehicle Detection* sistem akan mendeteksi jika ada kendaraan atau tidak. Menggunakan model YOLOv5 yang sudah di *training* menggunakan *custom dataset*, dataset ini memiliki 350 gambar berisi kendaraan (khususnya mobil berplat nomor) dan plat nomor kendaraan sudah diberi keterangan dalam format YOLOv5 PyTorch. Dataset ini memiliki 2 *class*, yaitu, *vehicle* dan *license-plate*.

Pada bagian *Plate Number Recognition* sistem akan mendeteksi plat nomor pada kendaraan menggunakan model YOLOv5 yang sudah di *training* menggunakan *custom dataset* tadi, setelah sistem mengetahui *bounding box*

tempat plat nomor berada pada sebuah kendaraan selanjutnya algoritma OCR (*Optical Character Recognition*) akan memproses *bounding box* plat nomor yang sudah ditentukan sebelumnya, kemudian mengubahnya menjadi teks berupa karakter-karakter yang terdapat pada plat nomor kendaraan tersebut.



Gambar 3.1 Rancangan Umum

3.3 Perancangan Sistem Vehicle Detection

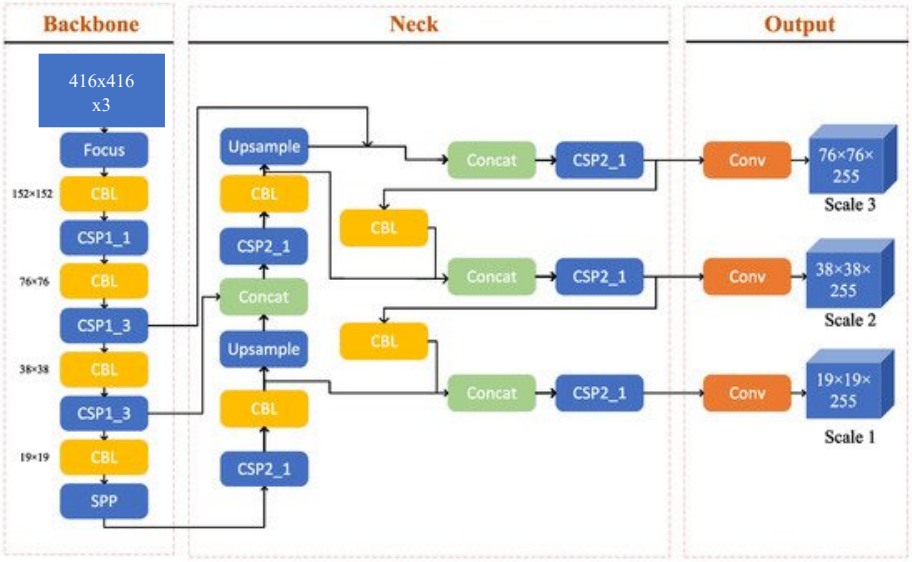
Sistem *Vehicle Detection* akan disimulasikan di platform *Google Collab*, karena keterbatasan akses dan kualitas yang kurang baik terhadap kamera CCTV JasaMarga selaku pengelola jalan tol, maka peneliti akan mensimulasikan tangkapan kamera CCTV yang berada di gerbang tol, dan gambar yang disimulasikan juga merepresentasikan resolusi dari kamera CCTV yang digunakan oleh sistem E-TLE. Sistem *Vehicle Detection* yang peneliti buat akan menggunakan model YOLOv5 yang di-*train* menggunakan *custom dataset*, dataset yang peneliti gunakan adalah License Plates - v3 yang tersedia di website roboflow.ai. Dataset ini terdiri dari 350 gambar yang sudah diberi anotasi berdasarkan format YOLOv5 PyTorch, 245 gambar akan digunakan untuk *training* dan 70 gambar akan digunakan untuk *testing*. Beberapa pre-processing yang diterapkan pada setiap gambar yang ada di dataset, yaitu, orientasi otomatis data piksel, mengubah ukuran gambar menjadi 416x416 piksel, dan tidak ada teknik augmentasi gambar diterapkan pada gambar di dataset.

Gambar-gambar yang terdapat di dataset didominasi gabungan dari gambar tampak depan dan tampak belakang dari kendaraan, kedua perspektif gambar ini akan membantu model untuk mempelajari bentuk mobil secara keseluruhan dan juga membantu model untuk mempelajari posisi plat nomor kendaraan di bagian belakang kendaraan. Hal ini memungkinkan sistem untuk mendeteksi kendaraan

dan plat nomor dari sisi depan dan belakang kendaraan, sehingga nantinya ketika sistem ingin digunakan untuk mendeteksi kendaraan dan plat nomor dari sisi belakang, model tidak perlu di *train* lagi dengan dataset khusus yang memiliki gambar kendaraan dari sisi belakang.

Gambar 3.2 Contoh gambar dataset

Dataset yang digunakan memiliki tipe kendaraan yang mirip dengan tip kendaraan yang ada di Indonesia namun dengan plat nomor luar negeri, walaupun plat nomor tidak sama namun memiliki beberapa karakteristik yang mirip dengan plat nomor di Indonesia seperti lokasi plat nomor, warna, dan bentuk plat nomor kendaraan yang ada di dataset.



Gambar 3.3 Arsitektur YOLOv5

Arsitektur YOLOv5 terdiri dari 3 bagian yaitu; *backbone, neck,* dan *output*. *Backbone* merupakan sebuah *convolutional neural network* yang mengekstrak fitur dengan ukuran berbeda dari gambar input dengan beberapa konvolusi dan *pooling*. Ada 4 *layer* dari *feature map* yang terdapat di *backbone* yang terdiri dari; 152 x 152 piksel, 76 x 76 piksel, 38 x 38 piksel, dan 19 x 19 piksel. *Neck* menggabungkan fitur-fitur dari tingkat yang berbeda untuk mendapatkan lebih banyak informasi kontekstual dan mengurangi kemungkinan *information loss*. Dalam prosesnya, *feature pyramid* dari FPN dan PAN digunakan, FPN akan menyampaikan fitur semantik yang kuat dari *feature map* teratas ke *feature map* yang lebih rendah dan PAN akan menyampaikan fitur lokalisasi dari *feature map* yang lebih rendah ke *feature map* yang lebih tinggi. Dari proses tersebut dihasilkan tiga *feature map* baru dengan ukuran 76 x 76 x 255, 38 x 38 x 255, dan 19 x 19 x 255, dimana 255 mengindikasikan jumlah *channel*. Semakin kecil *feature map*, semakin besar area gambar yang sesuai dengan setiap *grid unit* di *feature map*. Hal ini menunjukkan bahwa cocok untuk mendeteksi objek besar dari *feature map* 19 x 19 x 255, sedangkan *feature map* 76 x 76 x 255 cocok untuk mendeteksi objek kecil. Dari *feature map* baru ini, bagian *output* akan melakukan deteksi dan klasifikasi objek.

Sistem dirancang menggunakan bahasa pemrograman *Python* karena bahasa pemrograman *Python* adalah bahasa yang *simple* dan konsisten, memiliki akses ke

berbagai macam *library* dan *framework* untuk *Artificial Intelligence* (AI) dan *machine* learning (ML) yang dibutuhkan peneliti untuk membuat sistem ini, Bahasa pemrograman *python* juga memiliki *readability* yang baik sehingga peneliti dapat dengan mudah untuk membaca dan mengerti kode yang peneliti buat karena *python* dibuat semirip mungkin dengan Bahasa Inggris. *Python* juga merupakan bahasa pemrograman yang fleksibel dan *platform independence*, juga memiliki komunitas yang luas.

Library yang peneliti gunakan dalam membuat sistem *Vehicle Detection* adalah torch, IPython.display, utils.google_utils, roboflow, dan yaml. *Library* torch digunakan oleh peneliti untuk mengembangkan dan melatih model baru dari YOLOv5. IPython.display digunakan oleh peneliti untuk menampilkan gambar pada output di setiap *cell* ketika diperlukan. *Library u*tils.google_utils digunakan oleh peneliti untuk men-*download* model YOLOv5 yang sudah dilatih menggunakan *custom dataset* yang disimpan di *Google Drive* peneliti. *Library* roboflow digunakan oleh peneliti untuk men-*download custom dataset* dari website roboflow.ai. *Library* yaml digunakan oleh peneliti untuk mengakses file konfigurasi yaml yang digunakan untuk memberitahu sistem dimana letak file yaml dari dataset berada.

Peneliti melakukan *training* pada YOLOv5 menggunakan *custom dataset* dengan konfigurasi parameter sebagai berikut:

- --img: 416 (ukuran gambar dari dataset 416x416)
- --batch: 32 (jumlah gambar yang digunakan dalam satu iterasi)
- --epochs: 300 (jumlah berapa kali dataset melalui *neural network*)
- --cache (memasukkan gambar ke dalam *cache* agar mengurangi waktu *training*)

Setelah model selesai di *training*, peneliti menggunakan model tersebut untuk melakukan deteksi objek berupa kendaraan mobil berplat nomor yang sesuai dengan regulasi plat nomor di Indonesia. Dalam keadaan optimal, sistem akan membentuk 2 buah *bounding box* yang terdiri dari *vehicle* dan *license-plate*.

Nantinya, model akan dievaluasi menggunakan *test* set yang terdiri dari 70 gambar. Metrik yang umum digunakan untuk mengukur akurasi deteksi dari algoritma deteksi objek adalah *mean Average Precision* (mAP), dengan nilai antara 0 dan 1. Semakin tinggi skornya, semakin akurat modelnya. Lalu, ada metrik *Precision* yang memberi tahu kita seberapa tepat model dalam menentukan prediksinya, metrik *precision* dapat dihitung dengan menggunakan rumus $precision = \frac{TP}{TP+FP}$. *True-positive* (TP) mengacu pada situasi ketika model memprediksi objek dengan benar. Sebaliknya, *false-positive* (FP) mengacu pada prediksi objek yang salah. Kemudian, penulis menggunakan metrik *recall* yang mengukur seberapa baik model dapat menemukan semua hal positif dalam dataset, metrik *recall* dapat dihitung dengan menggunakan rumus $recall = \frac{TP}{TP+FN}$. *False-negative* (FN) berarti ketika model gagal memprediksi objek apa pun. Itu semua bergantung pada kesehatan data dan anotasi *ground truth* yang tepat yang digunakan model untuk proses *training*.

Setelah *bounding box* ditentukan, sistem akan memotong gambar menjadi 2 bagian berbeda, bagian pertama berisi gambar dari kendaraan yang terdeteksi secara keseluruhan dan bagian kedua terdiri dari hanya plat nomor dari kendaraan yang terdeteksi, kedua bagian ini disimpan dalam sebuah file berformat jpg yang selanjutnya akan diteruskan ke sistem *Plate Number Recognition* yang akan memproses gambar berupa plat nomor kendaraan menjadi teks yang dapat di yang dapat dibaca oleh komputer untuk diedit, dihitung, dan dianalisis.

3.4 Perancangan Sistem Plate Number Recognition

Sistem *Plate Number Recognition* dirancang oleh peneliti menggunakan OCR (*Optical Character Recognition*), lebih tepatnya menggunakan *package* easyOCR dan dengan bantuan *library* PyTorch sistem akan lebih cepat dalam mendeteksi karakter pada plat nomor kendaraan.

Input yang akan diberikan kepada *package* easyOCR ini berupa gambar plat nomor dari kendaraan yang sudah dipotong sebelumnya dan gambar tidak diberi *pre-processing* apapun, easyOCR dapat membaca teks dengan mudah karena sudah menggunakan CRAFT (*Character-Region Awareness For Text Detection*)

detection model yang secara efektif mendeteksi area teks dengan menjelajahi setiap wilayah karakter dan afinitas antar karakter pada sebuah gambar.

Beberapa konfigurasi parameter yang peneliti gunakan dalam menggunakan *package* easyOCR untuk memaksimalkan hasil akhir dari deteksi plat nomor kendaraan, yaitu:

- detail: 0 (menghilangkan detail tentang koordinat *bounding box*)
- paragraph: True (menganggap bahwa tulisan yang dideteksi adalah satu paragraph)
- allowlist: "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789" (*whitelist* dari karakter-karakter yang bisa dideteksi oleh sistem)

Dengan menggunakan parameter-parameter diatas, hasil akhir dari deteksi plat nomor kendaraan akan lebih mudah untuk dibaca dan diolah.