

Laporan Praktikum 6 Kontrol Cerdas

Nama : M. Reza Arimurti

NIM 214308017

Kelas : TKA – 6A

Akun Github (Tautan) : <https://github.com/jaezr1>

Student Lab Assistant : Muhammad Fatih Anfa Adhima

1. Judul Percobaan

Judul : Canny Edge Detection & Lane Detection with Instance Segmentation

Tujuan Percobaan

Pada praktikum ini, mahasiswa akan dapat:

- Memahami konsep Canny Edge Detection sebagai metode dasar deteksi tepi.
- Menggunakan Instance Segmentation untuk deteksi jalur rel kereta (Lane Detection).
- Menggunakan dataset Rail Segmentation dari Kaggle untuk eksperimen.
- Menggabungkan metode Canny Edge Detection dengan Instance Segmentation untuk meningkatkan deteksi jalur.

2. Landasan Teori

Deteksi Tepi Canny (Canny Edge Detection) merupakan salah satu teknik penting dalam pemrosesan citra yang bertujuan untuk mengenali batas atau kontur objek dalam sebuah gambar. Beberapa metode seperti Sobel, Prewitt, dan Roberts telah lama digunakan untuk keperluan ini. Namun, Canny Edge Detection dianggap lebih unggul karena kemampuannya mendeteksi tepi dengan presisi tinggi dan menghasilkan gambar yang minim gangguan atau noise (Luh Putu Risma Noviana dkk., 2024). Keunggulan utama dari metode ini terletak pada kemampuannya dalam melakukan good detection, yaitu memaksimalkan rasio sinyal terhadap noise (SNR), sehingga hampir semua tepi dapat dikenali dengan baik. Selain itu, metode ini memiliki good location, yang berarti dapat mendeteksi tepi sedekat mungkin dengan lokasi sebenarnya, meningkatkan akurasi deteksi (semakin tinggi nilai Loc, maka kualitas deteksi semakin baik). Keunggulan lain dari metode ini adalah one response to single edge, yang menjamin bahwa setiap tepi yang terdeteksi merupakan hasil yang valid, tanpa duplikasi atau kesalahan deteksi (Perangin-angin dkk., 2019).

Dalam praktikum ini, juga digunakan teknik Instance Segmentation untuk mendeteksi jalur rel kereta (Lane Detection). Instance Segmentation adalah proses pelabelan objek dalam gambar yang biasanya dilakukan dengan bantuan perangkat seperti Roboflow (Wulanningrum dkk., 2024). Model pelatihan menggunakan sekitar 2000 citra rel kereta api dengan berbagai variasi lintasan, seperti lurus, berbelok, jalur tunggal (single track), dan ganda (double track). Setiap epoch dalam pelatihan melibatkan input data ke dalam model, penyesuaian parameter, dan pembaruan bobot untuk meningkatkan akurasi dalam mendeteksi serta melakukan segmentasi rel. Tujuan dari proses ini adalah untuk mengurangi kesalahan dalam membedakan rel dengan objek lain berdasarkan label pada dataset pelatihan. Teknik data augmentation juga diterapkan guna memperluas kemampuan model dalam mengenali pola-pola yang bervariasi (Husain dkk., 2024).

Algoritma yang digunakan dirancang agar tahan terhadap oklusi atau gangguan visual kompleks, sehingga cocok diterapkan pada lingkungan kendaraan otonom (Autonomous Vehicle/AV). Algoritma ini telah diterapkan ke berbagai arsitektur jaringan klasifikasi seperti ENet, SegNet, dan ResNet38, yang semuanya dikhususkan untuk pengenalan gambar. Hasil penelitian menunjukkan bahwa arsitektur ResNet38 memberikan akurasi tertinggi secara rata-rata, meskipun membutuhkan memori yang lebih besar dibandingkan metode lain (Darwin & Saputro, 2021).

3. Analisis dan Diskusi

Analisis:

1. Seberapa baik deteksi jalur dengan Instance Segmentation dibandingkan Canny?

Deteksi jalur menggunakan **Instance Segmentation** umumnya lebih akurat dibandingkan dengan **Canny Edge Detection** karena mampu mengenali objek secara menyeluruh dan memberikan segmentasi berbasis instance, bukan hanya mendeteksi tepi. Instance Segmentation dapat membedakan jalur dari objek lain dalam gambar, bahkan dalam kondisi pencahayaan yang buruk atau latar belakang yang kompleks. Sebaliknya, Canny Edge Detection hanya mendeteksi perubahan intensitas piksel, sehingga rentan terhadap gangguan seperti bayangan, noise, atau tekstur yang menyerupai tepi jalur..

2. Apakah kombinasi kedua metode dapat meningkatkan akurasi?

Kombinasi **Canny Edge Detection** dan **Instance Segmentation** berpotensi meningkatkan akurasi deteksi jalur. Canny dapat digunakan sebagai langkah awal untuk menyoroti tepi jalur secara kasar, sementara Instance Segmentation dapat mengoreksi hasil deteksi dengan memahami konteks gambar secara lebih mendalam. Integrasi ini dapat mengurangi kesalahan deteksi, terutama dalam kondisi lingkungan yang menantang seperti persimpangan yang rumit atau jalur yang tertutup sebagian..

3. Apa dampak perubahan parameter Canny (thresholds) terhadap hasil deteksi

Perubahan **parameter threshold pada Canny Edge Detection** sangat berpengaruh terhadap hasil deteksi. Jika nilai threshold terlalu rendah, terlalu banyak tepi yang tidak relevan akan terdeteksi, sehingga meningkatkan noise dalam gambar. Sebaliknya, jika threshold terlalu tinggi, beberapa bagian jalur yang lebih samar mungkin tidak terdeteksi, menyebabkan hilangnya informasi penting. Oleh karena itu, pemilihan threshold yang optimal sangat krusial agar Canny Edge Detection dapat memberikan hasil yang lebih akurat, terutama saat dikombinasikan dengan Instance Segmentation.

Diskusi:

Kapan lebih baik menggunakan Canny Edge Detection dibanding Instance Segmentation?

Canny Edge Detection lebih cocok digunakan dalam kondisi di mana perhitungan cepat dan efisiensi komputasi menjadi prioritas, seperti pada perangkat dengan daya pemrosesan terbatas atau sistem real-time yang memerlukan deteksi tepi sederhana tanpa memerlukan segmentasi berbasis objek. Metode ini juga efektif dalam lingkungan dengan kontras tinggi, di mana tepi jalur sangat jelas terlihat, seperti pada citra dengan pencahayaan baik dan latar belakang yang tidak terlalu kompleks. Namun, jika gambar memiliki banyak gangguan atau pencahayaan yang tidak merata, metode ini bisa menghasilkan banyak deteksi yang keliru.

Bagaimana cara meningkatkan deteksi jalur dengan tuning parameter YOLOv8-seg?

Untuk meningkatkan **deteksi jalur menggunakan YOLOv8-seg**, tuning parameter menjadi langkah

penting. Beberapa aspek yang dapat disesuaikan termasuk **confidence threshold**, yang menentukan ambang batas kepercayaan model dalam mendeteksi suatu objek, serta IoU (Intersection over Union) threshold, yang mengontrol tingkat tumpang tindih yang diperbolehkan antara prediksi dan ground truth. Selain itu, penyesuaian arsitektur model seperti jumlah lapisan konvolusi dan ukuran anchor box dapat membantu model lebih responsif terhadap variasi bentuk dan ukuran jalur rel. Teknik augmentasi data, seperti rotasi, pemotongan, dan perubahan pencahayaan, juga dapat meningkatkan kemampuan generalisasi model terhadap berbagai kondisi lingkungan.

Bagaimana metode ini dapat diterapkan dalam sistem navigasi kereta otomatis?

Penerapan metode ini dalam **sistem navigasi kereta otomatis** dapat meningkatkan keamanan dan efisiensi operasional. Instance Segmentation dapat membantu dalam **mendeteksi dan mengenali jalur rel dengan lebih presisi**, memungkinkan sistem untuk memahami struktur lintasan dengan lebih baik, termasuk jalur bercabang, perlintasan sebidang, dan area dengan rintangan. Dengan kombinasi Canny Edge Detection untuk deteksi cepat dan Instance Segmentation untuk pemetaan detail, sistem navigasi dapat mengurangi risiko kesalahan dalam pengambilan keputusan, seperti menghindari tabrakan atau memastikan kereta tetap berada di jalur yang benar. Integrasi metode ini dengan sensor lain seperti LiDAR dan GPS juga dapat meningkatkan akurasi sistem navigasi secara keseluruhan.

4. Assignment



Pada praktikum keenam, dikembangkan sebuah sistem yang menggabungkan deteksi tepi menggunakan metode **Canny Edge Detection** dengan deteksi atau segmentasi objek melalui model **YOLO** dari Ultralytics. Sistem ini dirancang untuk mendeteksi objek secara real-time melalui video yang ditangkap dari kamera. Proses dimulai dengan memuat model YOLO dari direktori tertentu menggunakan fungsi `YOLO(MODEL_PATH)`, guna memastikan model tersedia sebelum dilanjutkan ke pemrosesan video dengan OpenCV. Kamera diakses melalui `cv2.VideoCapture(0)`, dan setiap frame yang diambil diubah ke format grayscale menggunakan `cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)` untuk memungkinkan penerapan Canny Edge Detection. Proses ini menghasilkan citra biner dengan tepi putih di atas latar belakang hitam melalui `cv2.Canny(gray, low_threshold, high_threshold)`.


Secara bersamaan, frame asli diubah ke format RGB melalui `cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)` dan kemudian dianalisis menggunakan model YOLO untuk melakukan prediksi objek atau segmentasi. Hasil pertama dari prediksi disimpan dalam `predictions = results[0]`. Bila model dilatih untuk segmentasi, objek ditandai dengan poligon transparan berwarna biru menggunakan `cv2.fillPoly()`. Jika model hanya mendeteksi objek menggunakan kotak pembatas (*bounding box*), maka kotak hijau akan digambar di sekitar objek menggunakan `cv2.rectangle()`, dan label objek ditampilkan bila tersedia menggunakan `cv2.putText()`.

Selanjutnya, sistem menerapkan masking untuk membedakan tepi-tepi yang berasal dari objek yang dikenali YOLO dan yang hanya hasil dari Canny Edge Detection. Tepi di dalam area objek yang terdeteksi oleh YOLO akan diberi warna merah, sedangkan tepi lainnya tetap berwarna putih, dilakukan melalui ekspresi `edges_bgr[np.where((mask_model == 255) & (edges == 255))] = [0, 0, 255]`. Hasil dari prediksi YOLO dan deteksi tepi ditampilkan secara berdampingan menggunakan `np.hstack()`, sehingga perbedaan antara tepi umum dan tepi dari objek yang terdeteksi dapat terlihat dengan jelas. Program akan terus berjalan hingga pengguna menekan tombol 'q', yang akan dihentikan dengan `cv2.waitKey(1) & 0xFF == ord('q')`. Setelah itu, kamera akan dilepas melalui `cap.release()` dan seluruh jendela ditutup dengan `cv2.destroyAllWindows()` untuk mencegah penggunaan memori yang tidak perlu. Gabungan antara metode deteksi tepi klasik dan teknologi deep learning ini memungkinkan sistem untuk diterapkan pada berbagai keperluan, seperti deteksi jalur rel, sistem pemantauan lalu lintas, dan aplikasi lainnya.

5. Data dan Output Hasil Pengamatan

Sajikan data dan hasil yang diperoleh selama percobaan. Gunakan tabel untuk menyajikan data jika diperlukan.

No	Variabel	Hasil Pengamatan
1	Canny Edge Detection	 The image shows the result of a Canny edge detection algorithm applied to a photograph of a railway track. The background is black, and the edges of the tracks, rails, and surrounding infrastructure are highlighted in white. The tracks run from the bottom center towards the horizon, where they appear to cross or merge. The surrounding area is filled with a dense network of white lines representing various edges in the scene.
2	Lane Detection dengan Instance Segmentation (YOLOv8-seg)	 The image shows the result of a lane detection and instance segmentation algorithm (YOLOv8-seg) applied to a photograph of a railway track. The original color image is shown, but with cyan-colored bounding boxes overlaid on the tracks. Two labels are present: 'Main Rail 0.80' on the left track and 'Main Rail 0.62' on the right track. The bounding boxes are rectangular and follow the length of the tracks, indicating the model's confidence in the detection.

3	Menggabungkan Canny Edge Detection dengan Lane Detection	
---	--	--

6. Kesimpulan

1. Integrasi metode **Canny Edge Detection** dengan model **YOLO**, memungkinkan deteksi tepi sekaligus identifikasi dan penyorotan objek dalam video real-time.
2. Model **YOLO** mampu mengenali objek dengan baik, baik dalam bentuk **bounding box** maupun **segmentasi**, yang kemudian ditandai menggunakan warna dan anotasi yang sesuai. Selain itu, teknik **masking** diterapkan untuk membedakan tepi objek yang dikenali YOLO dengan tepi umum hasil Canny Edge Detection, sehingga menghasilkan visualisasi yang lebih jelas.

7. Saran

Untuk meningkatkan performa dan akurasi, program dapat dioptimalkan dengan **pemanfaatan GPU**, sehingga prediksi YOLO dapat berjalan lebih cepat dan responsif. Selain itu, **preprocessing tambahan** seperti penerapan **Gaussian Blur** sebelum **Canny Edge Detection** dapat membantu mengurangi noise yang tidak diinginkan, sehingga deteksi tepi menjadi lebih akurat.

8. Daftar Pustaka

- Darwin, S., & Saputro, N. (2021). SURVEI APLIKASI SEGMENTASI CITRA UNTUK AUTONOMOUS VEHICLE.
- Husain, B. H., Osawa, T., Astiti, S. P. C., Frianto, D., Nandika, M. R., & Agustine, D. (2024). Deteksi Lubang Jalan Secara Otomatis dari Rekaman Drone Menggunakan Model Machine Learning Berbasis YOLOv5 Instance Segmentation di Kota Pekanbaru, Provinsi Riau, Indonesia. *Jurnal Zona*, <https://doi.org/10.52364/zona.v8i2.126> 8(2), 137–146.
- Luh Putu Risma Noviana, I Putu Eka Indrawan, & Gde Iwan Setiawan. (2024). ANALYSIS OF CANNY EDGE DETECTION METHOD FOR FACIAL RECOGNITION IN DIGITAL IMAGE PROCESSING. *Jurnal Manajemen dan Teknologi Informasi*, 15(2), 29–34. <https://doi.org/10.59819/jmti.v15i2.4107>
- Perangin-angin, R., Harianja, E. J. G., & No, J. H. T. (2019). COMPARISON DETECTION EDGE LINES ALGORITMA CANNY DAN SOBEL. 2.
- Wulanningrum, R., Handayani, A. N., & Wibawa, A. P. (2024). Perbandingan Instance Segmentation Image Pada Yolo8. *Jurnal Teknologi Informasi dan Ilmu Komputer*, <https://doi.org/10.25126/jtiik.1148288>

