

# Laporan Praktikum Kontrol Cerdas

Nama : Moh.Farhan Baihaqi

NIM : 224308037

Kelas : TKA-6B

Akun Github (Tautan) : <https://github.com/farhanbaihaqi21>

Student Lab Assistant :Mbak Dwi

1. Judul Percobaan : *Canny Edge Detection & Lane Detection with Instance Segmentation*

2. Tujuan Percobaan :

- 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.

3. Landasan Teori :

*Machine Learning* merupakan sistem yang mampu belajar sendiri peran manusia dalam pengambilan keputusan dengan mengambil keputusan secara otomatis, belajar dari data, dan menentukan pola tanpa harus berulang kali diprogram oleh manusia (Prasetyo & Dewayanto., 2024). Untuk bisa menggunakan *machine learning* harus memiliki data. Data yang digunakan dibagi menjadi dua data training dan data testing. Data training digunakan untuk melatih algoritma, sedangkan data testing digunakan untuk mengetahui performa yang telah dilatih sebelumnya ketika menemukan data baru yang belum pernah diberikan dalam data training (Yudianto & Fatta, 2020). Machine Learning memiliki beberapa jenis, seperti, supervised learning, unsupervised learning dan reinforcement learning. *Deep learning* merupakan pembelajaran mesin yang berkaitan dengan algoritma

dimana cara kerja dari algoritma ini meniru struktur dan fungsi otak yang disebut jaringan saraf tiruan (Rochmawati et al., 2021). *Deep learning* sangat berguna ketika mencoba untuk memahami pola dari data yang tidak terstruktur. Jaringan saraf kompleks dalam deep learning dirancang untuk meniru cara kerja otak manusia sehingga komputer dapat dilatih untuk menangani abstraksi dan masalah yang kurang terdefinisi dengan baik (Windiawan & Suharso, 2021).

Pengolahan citra merupakan proses yang melibatkan analisis dan pengolahan citra dengan mempertimbangkan persepsi visual. Metode canny edge Detection merupakan salah satu teknik dalam pengolahan citra yang memiliki pengaruh besar karena kemampuannya dalam mendeteksi tepi dengan efektif. Teknik ini memanfaatkan serangkaian algoritma yang terdiri dari beberapa langkah, termasuk penghalusan gambar untuk mengurangi noise, perhitungan gradien untuk menentukan kekuatan dan arah tepi, penekanan non-maksimum untuk mempertajam tepi, serta penghubungan tepi dengan histeresis untuk mengidentifikasi tepi yang kuat dan lemah. Hasil dari proses ini adalah garis-garis tepi yang jelas yang selanjutnya dapat diinterpretasikan untuk mengenali karakteristik tertentu (Nugroho et al., 2025).

#### 4. Analisis dan Diskusi :

##### -Analisis

Pada praktikum minggu keenam, percobaan yang dilakukan adalah penerapan Canny Edge Detection dan deteksi jalur menggunakan Instance Segmentation untuk mengidentifikasi jalur kereta api. Hasilnya menunjukkan bahwa metode Instance Segmentation memiliki kinerja yang lebih baik dibandingkan dengan Canny Edge Detection dalam hal akurasi dan kemampuan membedakan objek. Instance Segmentation dapat memisahkan dan mengklasifikasikan setiap elemen dalam gambar, termasuk rel kereta, kendaraan, dan rintangan lainnya. Teknik ini tidak hanya mendeteksi tepi, tetapi juga memprediksi bentuk (mask) dari setiap instance secara terpisah. Kemampuan Instance Segmentation untuk mengenali jalur tetap konsisten meskipun menghadapi kondisi lingkungan yang rumit, seperti bayangan,

variasi pencahayaan, dan tekstur jalan yang tidak merata. Di sisi lain, Canny Edge Detection hanya bergantung pada perubahan gradien intensitas dalam citra, yang sering kali menghasilkan deteksi palsu (false positive) ketika terdapat noise atau tekstur yang kompleks pada permukaan jalur. Instance Segmentation dapat mengatasi keterbatasan ini dengan mendeteksi fitur objek secara keseluruhan, bukan hanya tepi, sehingga memberikan hasil deteksi yang lebih akurat dan stabil. Kombinasi antara Canny Edge Detection dan Instance Segmentation dapat meningkatkan akurasi dalam deteksi jalur. Metode Canny efektif dalam mengekstraksi tepi dengan tingkat presisi yang tinggi dan membedakan antara area tepi yang kuat dan lemah, sementara Instance Segmentation memungkinkan sistem untuk mengelompokkan dan membedakan jalur dari elemen lain di sekitarnya. Sinergi ini menghasilkan sistem yang lebih adaptif dan akurat, karena hasil deteksi tepi dari metode Canny diverifikasi oleh Instance Segmentation. Dengan pendekatan ini, sistem dapat mendeteksi jalur dengan lebih stabil dan akurat, bahkan dalam kondisi lingkungan yang menantang seperti pencahayaan rendah, bayangan, atau permukaan jalur yang tidak rata. Proses deteksi dimulai dengan penerapan metode Canny untuk mengekstraksi tepi jalur, diikuti oleh Instance Segmentation untuk memisahkan dan mengenali jalur dengan lebih tepat. Kombinasi ini juga meningkatkan kecepatan dan efisiensi dalam pengambilan keputusan, karena Instance Segmentation dapat memperbaiki hasil deteksi yang kurang akurat dari metode Canny.

Perubahan parameter pada metode Canny Edge Detection, terutama nilai ambang batas (threshold), memiliki pengaruh yang signifikan terhadap hasil deteksi tepi dan jalur. Jika nilai ambang batas terlalu tinggi, algoritma Canny hanya akan mendeteksi tepi yang sangat kuat, sehingga banyak tepi lemah yang terabaikan, mengakibatkan jalur menjadi terputus atau tidak terdeteksi sepenuhnya. Sebaliknya, jika ambang batas terlalu rendah, algoritma Canny akan mendeteksi terlalu banyak tepi, termasuk noise atau artefak dalam gambar, yang menyebabkan hasil deteksi menjadi tidak stabil dan menghasilkan banyak false positive. Oleh karena itu,

pemilihan nilai ambang batas yang tepat sangat penting untuk mencapai hasil deteksi yang optimal. Penyesuaian parameter yang tepat pada metode Canny memungkinkan sistem untuk lebih efektif membedakan antara tepi yang relevan dan noise, sehingga meningkatkan ketepatan dan stabilitas hasil deteksi jalur. Kombinasi metode Canny dengan Instance Segmentation memberikan keuntungan tambahan, karena Instance Segmentation dapat memperbaiki deteksi tepi yang terputus atau salah dari hasil metode Canny, menciptakan sistem deteksi jalur yang lebih akurat dan adaptif terhadap perubahan kondisi lingkungan.

#### -Diskusi

Pada praktikum minggu keenam, dapat disimpulkan bahwa metode Canny Edge Detection lebih sesuai digunakan dalam situasi yang memerlukan deteksi tepi dengan kecepatan tinggi dan sumber daya komputasi yang terbatas. Metode ini sangat efektif untuk mendeteksi struktur tepi yang jelas dan memiliki kontras tinggi, seperti jalur lurus atau pola garis yang teratur pada permukaan jalan atau rel kereta. Karena algoritma Canny menghitung perubahan gradien intensitas dalam citra, metode ini sangat berguna pada gambar dengan latar belakang yang sederhana dan minim noise. Selain itu, Canny Edge Detection lebih ringan secara komputasi dibandingkan dengan Instance Segmentation, sehingga dapat diterapkan pada sistem real-time dengan keterbatasan perangkat keras atau pada sistem yang membutuhkan daya rendah. Namun, dalam lingkungan yang kompleks, seperti perubahan pencahayaan, bayangan, atau tekstur jalan yang tidak konsisten, kinerja metode Canny dapat menurun karena algoritma ini sensitif terhadap noise dan tekstur yang rumit.

Peningkatan akurasi dalam deteksi jalur menggunakan YOLOv8-seg dapat dicapai melalui penyesuaian parameter yang tepat. Parameter seperti learning rate, batch size, dan jumlah epochs sangat berpengaruh terhadap kinerja model. Pengaturan learning rate yang terlalu tinggi dapat menyulitkan model untuk mencapai konvergensi, sedangkan learning rate yang terlalu rendah dapat memperlambat proses pelatihan. Batch size juga perlu disesuaikan untuk menjaga

keseimbangan antara kecepatan pelatihan dan stabilitas konvergensi model. Selain itu, penyesuaian pada parameter confidence threshold dan IoU (Intersection over Union) threshold dapat meningkatkan ketelitian dalam mendeteksi jalur dengan menyaring prediksi yang kurang akurat atau memiliki tingkat kepercayaan rendah. Akurasi juga dapat ditingkatkan dengan memperbesar ukuran dataset pelatihan, melakukan augmentasi data seperti rotasi dan flip, serta menerapkan regularisasi untuk mengurangi overfitting pada model. Fine-tuning pada arsitektur YOLOv8-seg, seperti menyesuaikan jumlah layer atau konfigurasi anchor box, juga dapat meningkatkan sensitivitas model dalam mengenali pola jalur yang kompleks.

Penerapan metode ini dalam sistem navigasi kereta otomatis dapat meningkatkan keandalan dan ketepatan dalam pengenalan jalur, mendukung sistem pengendalian kereta yang lebih aman dan efisien. Kombinasi antara Canny Edge Detection dan Instance Segmentation memungkinkan sistem untuk mendeteksi jalur secara real-time dengan kecepatan dan akurasi tinggi. Canny Edge Detection dapat memberikan respons cepat dalam mengenali tepi jalur, sementara Instance Segmentation dapat memperbaiki hasil deteksi dengan membedakan jalur dari elemen lain di sekitarnya, seperti kendaraan atau rintangan. Dalam sistem navigasi kereta otomatis, hasil deteksi jalur ini dapat digunakan untuk mengatur kecepatan kereta, menghindari tabrakan, dan memastikan kereta tetap berada di jalur yang benar. Selain itu, sistem dapat dilengkapi dengan algoritma pemrosesan citra tambahan untuk mengenali sinyal atau rambu di sepanjang jalur, sehingga memungkinkan sistem untuk menyesuaikan kecepatan atau melakukan pengereman secara otomatis. Dengan menggabungkan metode deteksi jalur berbasis Canny dan Instance Segmentation, sistem navigasi kereta otomatis dapat beroperasi dengan lebih andal dalam berbagai kondisi lingkungan, meningkatkan keselamatan dan efisiensi operasional.


## 5. Assignment :

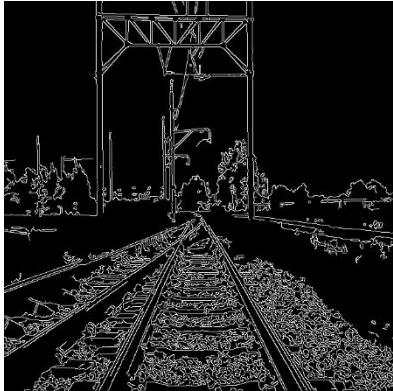

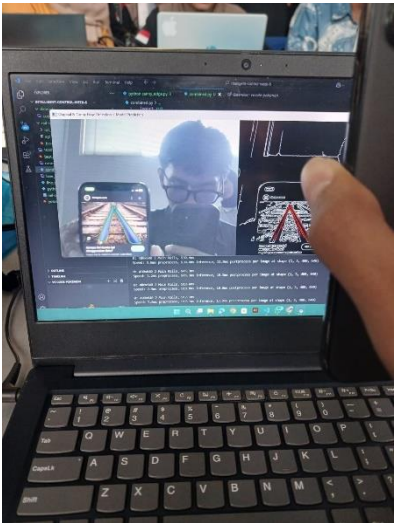
Berdasarkan praktikum keenam yang telah dilakukan, kode program ini dirancang untuk menciptakan sistem yang dapat melakukan deteksi tepi menggunakan metode Canny Edge Detection dan menggabungkannya dengan deteksi objek atau segmentasi menggunakan model YOLO dari Ultralytics. Sistem ini dapat menyoroti objek yang terdeteksi dalam video yang diambil secara real-time dari kamera. Program dimulai dengan memuat model YOLO dari path yang telah ditentukan menggunakan `YOLO(MODEL_PATH)`, memastikan bahwa model tersebut tersedia sebelum melanjutkan ke pemrosesan video dengan OpenCV. Setelah kamera diakses melalui `cv2.VideoCapture(0)`, setiap frame yang ditangkap dikonversi menjadi skala abu-abu dengan `cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)` untuk melakukan deteksi tepi menggunakan metode Canny Edge Detection, yang menghasilkan gambar biner dengan tepi putih di latar belakang hitam melalui `cv2.Canny(gray, low_threshold, high_threshold)`. Secara bersamaan, frame asli dikonversi ke format RGB dengan `cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)` dan kemudian diproses oleh model YOLO untuk melakukan prediksi objek atau segmentasi, di mana hasil prediksi pertama disimpan dalam `predictions = results[0]`. Jika model menggunakan segmentasi, program akan menggambar area objek dengan poligon biru transparan menggunakan `cv2.fillPoly(overlay, [mask], (255, 0, 0))`. Jika model hanya mendeteksi objek dalam bentuk bounding box, objek akan dilingkari dengan kotak hijau menggunakan `cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)`, dan label objek akan ditampilkan jika tersedia menggunakan `cv2.putText()`. Selanjutnya, masking digunakan untuk membedakan tepi objek yang terdeteksi oleh YOLO dari tepi yang dideteksi oleh metode Canny, di mana tepi dalam area objek yang dikenali oleh YOLO akan diwarnai merah, sementara tepi lainnya tetap berwarna putih dengan `edges_bgr[np.where((mask_model == 255) & (edges == 255))] = [0, 0, 255]`. Kedua hasil, yaitu frame dengan anotasi YOLO dan hasil deteksi tepi, ditampilkan secara berdampingan menggunakan `np.hstack()` untuk memberikan tampilan visual yang lebih jelas mengenai perbedaan antara tepi umum dan tepi objek yang dikenali.

Program ini berjalan dalam loop hingga pengguna menekan tombol 'q', yang akan memicu `cv2.waitKey(1) & 0xFF == ord('q')` untuk keluar. Setelah itu, kamera akan dilepaskan dengan `cap.release()` dan semua jendela yang terbuka akan ditutup dengan `cv2.destroyAllWindows()` untuk mencegah kebocoran memori. Dengan menggabungkan metode deteksi tepi tradisional dan teknologi deep learning untuk segmentasi atau deteksi objek, program ini dapat diterapkan dalam berbagai aplikasi seperti deteksi rel, pengawasan lalu lintas, dan lainnya.

6. Data dan Output Hasil :




- Data Sebelum Modifikasi

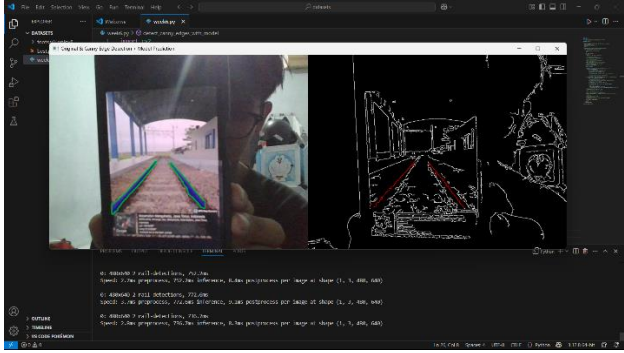
No.	Variabel	Hasil Pengamatan
1.	Foto	

2.	Canny Edge Detection	 <p>The image shows the result of Canny edge detection applied to a photograph of a railway track. The track rails and the overhead power line structure are highlighted as white lines on a black background, showing the detected edges of the scene.</p>
3.	Lane Detection dengan Instance Segmentation (YOLOv8-seg)	 <p>The image displays a railway track with lane detection and instance segmentation results. Four bounding boxes are overlaid on the track rails, each labeled with a confidence score: "Branch Rail 0.92", "Branch Rail 0.95", "Main Rail 0.93", and "Main Rail 0.92". The boxes are colored blue and green, indicating the detected segments of the track.</p>
4.	Menggabungkan Canny Edge Detection dengan Lane Detection	 <p>The image shows a laptop screen displaying a video feed. The video feed is processed with Canny edge detection and lane detection. The results are overlaid on the video, showing the detected edges and lane segments. A hand is visible pointing at the screen, indicating the results of the combined detection process.</p>



- Data Sesudah Modifikasi

No.	Variabel	Hasil Pengamatan
1.	Foto	 A photograph of a railway track stretching into the distance under a cloudy sky. The tracks are made of metal rails on gravel bed, with overhead power lines and greenery on the sides.
2.	Canny Edge Detection	 The result of Canny edge detection applied to the railway track photo. The image is black with white lines highlighting the edges of the tracks, power lines, and surrounding vegetation.
3.	Lane Detection dengan Instance Segmentation (YOLOv8-seg)	 The result of lane detection using YOLOv8-seg. The original photo is shown with blue bounding boxes around the railway tracks. Text labels 'rail-detection 0.96' are visible above the bounding boxes, indicating the confidence score for the detection.

4.	Menggabungkan Canny Edge Detection dengan Lane Detection	
----	--	--

## 7. Kesimpulan

Berdasarkan praktikum dan analisis yang telah dilakukan, maka dapat diambil kesimpulan yaitu:

- Program berhasil mengintegrasikan metode Canny Edge Detection dengan model YOLO untuk mendeteksi tepi sekaligus mengenali dan menyoroti objek dalam video secara real-time.
- Model YOLO menunjukkan kemampuan yang baik dalam mendeteksi objek, baik dalam bentuk bounding box maupun segmentasi, yang kemudian ditandai dengan warna dan anotasi yang sesuai.
- Teknik masking digunakan untuk membedakan tepi objek yang terdeteksi oleh YOLO dari tepi umum yang dihasilkan oleh Canny Edge Detection, sehingga memungkinkan visualisasi yang lebih jelas.
- Dengan melakukan pelatihan pada dataset yang sesuai, pengguna dapat meningkatkan kinerja model YOLOv8 untuk mendeteksi objek spesifik dalam konteks tertentu, sehingga meningkatkan akurasi dan efektivitas dalam aplikasi dunia nyata. Pelatihan yang tepat pada dataset yang relevan memungkinkan model untuk belajar mengenali pola dan fitur yang penting, menjadikannya lebih adaptif terhadap berbagai kondisi dan tantangan dalam pengenalan objek.

#### 8. Saran :

Untuk meningkatkan performa dan akurasi, program dapat dioptimalkan dengan memanfaatkan GPU, sehingga prediksi YOLO dapat berjalan lebih cepat dan responsif. Selain itu, penerapan preprocessing tambahan seperti Gaussian Blur sebelum Canny Edge Detection dapat membantu mengurangi noise yang tidak diinginkan. Untuk meningkatkan keterbacaan visual, transparansi overlay warna dapat disesuaikan agar objek yang terdeteksi lebih jelas tanpa menutupi detail penting dalam gambar. Menambahkan fitur untuk menyimpan hasil deteksi dalam bentuk gambar atau video juga dapat menjadi nilai tambah untuk analisis lebih lanjut dalam berbagai aplikasi di dunia nyata.

#### 9. Daftar Pustaka :

Nugroho, C. W., Nurtanio, I., & Jalil, A. (2025). Penentuan Kualitas Kopra Berbasis

Citra Kontur Menggunakan Metode Canny Edge Detection: Determination of Copra Quality Based on Contour Image Using the Canny Edge Detection Method. *MALCOM: Indonesian Journal of Machine Learning and Computer Science*, 5(1), 436–450. <https://doi.org/10.57152/malcom.v5i1.1823>

Prasetyo, S., & Dewayanto, T. (n.d.). *PENERAPAN MACHINE LEARNING, DEEP LEARNING, DAN DATA MINING DALAM DETEKSI KECURANGAN LAPORAN KEUANGAN - A SYSTEMATIC LITERATURE REVIEW*.

Rochmawati, N., Hidayati, H. B., Yamasari, Y., Tjahyaningtijas, H. P. A., Yustanti, W., & Prihanto, A. (2021). Analisa Learning Rate dan Batch Size pada Klasifikasi Covid Menggunakan Deep Learning dengan Optimizer Adam. *Journal of Information Engineering and Educational Technology*, 5(2), 44–48. <https://doi.org/10.26740/jieet.v5n2.p44-48>

Winarno, E. (2011). *Aplikasi Deteksi Tepi pada Realtime Video menggunakan Algoritma Canny Detection*. 16.

Yudianto, M. R. A., & Fatta, H. A. (2020). *ANALISIS PENGARUH TINGKAT AKURASI KLASIFIKASI CITRA WAYANG DENGAN ALGORITMA CONVOLUTIONAL NEURAL NETWORK*.