

# Laporan Praktikum 6 Kontrol Cerdas

Nama : Kevin Bayu Pradana  
NIM : 224308009  
Kelas : TKA - 6A  
Akun Github (Tautan) : <https://github.com/Kevinbyu84>  
Student Lab Assistant : Rizky Putri Ramadhani

## 1. Judul Percobaan

Week 6: Canny Edge Detection & Lane Detection with Instance Segmentation

## 2. Tujuan Percobaan

Tujuan dari praktikum "Canny Edge Detection & Lane Detection with Instance Segmentation", mahasiswa diharapkan mampu:

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

## 3. Landasan Teori

- **Canny Edge Detection** adalah algoritma deteksi tepi yang dikembangkan oleh John Canny pada tahun 1986. Metode ini digunakan untuk mendeteksi tepi dalam citra dengan meminimalkan kesalahan deteksi dan menghasilkan representasi yang akurat dari tepi dalam gambar. Algoritma Canny terdiri dari beberapa langkah utama, yaitu: (1) penghalusan dengan Gaussian filter, (2) perhitungan gradien intensitas, (3) non-maximum suppression, dan (4) hysteresis thresholding. Metode ini banyak digunakan dalam pemrosesan citra, termasuk dalam segmentasi objek dan deteksi fitur.
- **Instance Segmentation** adalah teknik dalam computer vision yang tidak hanya membedakan antara kelas objek dalam suatu gambar tetapi juga mengisolasi setiap instansi objek dalam kelas yang sama. Teknik ini lebih kompleks dibandingkan semantic segmentation karena memerlukan deteksi batas objek secara individual. Model seperti Mask R-CNN dan YOLOv8-seg telah diterapkan secara luas untuk instance segmentation karena kemampuannya dalam memberikan segmentasi yang akurat dan efisien.
- **Rail Segmentation** adalah pendekatan dalam computer vision yang digunakan untuk mendeteksi dan mengidentifikasi rel kereta api dalam citra atau video. Teknik ini sangat penting dalam sistem pemantauan infrastruktur perkeretaapian untuk mendeteksi kerusakan atau gangguan. Metode yang digunakan dalam rail segmentation biasanya berbasis deep learning, seperti CNN dan model berbasis segmentasi instance seperti YOLOv8-seg atau DeepLabV3+.
- **YOLOv8-seg** adalah varian dari YOLOv8 yang dirancang khusus untuk instance segmentation. Model ini mampu melakukan deteksi objek sekaligus memberikan mask segmentasi untuk setiap objek yang terdeteksi. YOLOv8-seg lebih cepat dan lebih akurat dibandingkan model terdahulu karena menggunakan arsitektur yang lebih efisien dan teknik post-processing yang dioptimalkan. Model ini cocok untuk aplikasi seperti rail segmentation dan analisis video berbasis deep learning.
- **Konversi video ke gambar** adalah proses ekstraksi frame dari video untuk digunakan dalam berbagai aplikasi seperti pelatihan model machine learning, analisis gerak, dan pemrosesan citra.

Proses ini dapat dilakukan menggunakan pustaka seperti OpenCV, FFmpeg, dan ImageIO. Konversi ini penting dalam instance segmentation dan object detection karena memungkinkan pembuatan dataset dari sumber video.

- **Roboflow** adalah platform yang menyediakan alat untuk manajemen, augmentasi, dan pelatihan dataset dalam proyek computer vision. Roboflow memungkinkan pengguna untuk mengunggah dataset, menerapkan preprocessing, dan mengonversi dataset ke berbagai format yang kompatibel dengan model deep learning seperti YOLO, TensorFlow, dan PyTorch. Dengan fitur AutoML dan pelatihan berbasis cloud, Roboflow telah menjadi alat yang populer dalam penelitian dan pengembangan model computer vision.

## 4. Analisis dan Diskusi

### Analisis Hasil:

- **Seberapa Baik Deteksi Jalur dengan Instance Segmentation dibandingkan Canny?**  
Instance segmentation, seperti yang diterapkan dalam **YOLOv8-seg**, secara umum lebih unggul dibandingkan **Canny Edge Detection** dalam mendeteksi jalur rel karena mampu membedakan antara berbagai objek dalam gambar dan menghasilkan segmentasi yang lebih akurat. Sementara **Canny** hanya mendeteksi tepi tanpa memahami konteks objek, YOLOv8-seg dapat mengenali rel sebagai suatu entitas dengan segmentasi yang lebih presisi. Namun, Canny lebih cepat dan ringan dibandingkan model deep learning seperti YOLOv8-seg, sehingga lebih cocok untuk aplikasi real-time dengan keterbatasan komputasi.
- **Apakah Kombinasi Kedua Metode Dapat Meningkatkan Akurasi?**  
Ya, kombinasi **Canny Edge Detection** dan **Instance Segmentation** berpotensi meningkatkan akurasi. Canny dapat digunakan sebagai tahap pre-processing untuk menyoroti fitur tepi rel sebelum digunakan sebagai input bagi model YOLOv8-seg. Teknik ini dapat membantu meningkatkan kontras dan mengurangi noise pada input, sehingga model deep learning dapat lebih fokus pada fitur yang relevan. Pendekatan ini juga dapat digunakan untuk validasi hasil segmentasi, dengan membandingkan hasil YOLOv8-seg dengan hasil Canny untuk mengurangi kesalahan deteksi.
- **Apa Dampak Perubahan Parameter Canny (Thresholds) terhadap Hasil Deteksi?**  
Canny Edge Detection menggunakan dua parameter threshold:  
Lower threshold: Jika nilai gradien piksel lebih rendah dari threshold ini, piksel diabaikan.  
Upper threshold: Jika nilai gradien piksel lebih tinggi dari threshold ini, piksel dianggap sebagai tepi yang kuat.  
Jika threshold terlalu rendah, terlalu banyak tepi yang terdeteksi, menghasilkan noise dan artefak yang tidak diinginkan. Sebaliknya, jika threshold terlalu tinggi, banyak tepi penting bisa hilang, sehingga rel mungkin tidak terdeteksi dengan baik. Pengaturan threshold yang optimal bergantung pada kondisi pencahayaan dan kontras gambar.

### Diskusi:

- **Kapan Lebih Baik Menggunakan Canny Edge Detection dibanding Instance Segmentation?**  
Jika komputasi terbatas: Canny lebih cepat dan ringan, cocok untuk sistem dengan keterbatasan hardware.  
Jika lingkungan pencahayaan konstan: Canny bekerja dengan baik pada gambar dengan kontras tinggi tanpa banyak noise.  
Jika kebutuhan hanya deteksi tepi: Misalnya, dalam aplikasi pemrosesan gambar sederhana tanpa perlu memahami konteks objek.
- **Bagaimana Cara Meningkatkan Deteksi Jalur dengan Tuning Parameter YOLOv8-seg?**  
Menggunakan dataset berkualitas tinggi: Dataset harus memiliki anotasi yang akurat dengan variasi kondisi pencahayaan, perspektif, dan lingkungan.  
Augmentasi data: Menambahkan augmentasi seperti brightness adjustment, rotation, dan blur dapat meningkatkan ketahanan model terhadap kondisi yang berbeda.  
Fine-tuning model: Menggunakan transfer learning dengan dataset khusus rail segmentation dapat meningkatkan akurasi model YOLOv8-seg.
- **Bagaimana Metode Ini Dapat Diterapkan dalam Sistem Navigasi Kereta Otomatis?**

Deteksi dan Segmentasi Jalur: Menggunakan instance segmentation untuk mengidentifikasi jalur rel secara real-time.

Peringatan Dini terhadap Hambatan: Mendeteksi objek di atas rel seperti batu, manusia, atau kendaraan untuk mencegah kecelakaan.

Navigasi Otomatis: Informasi dari YOLOv8-seg dapat dikombinasikan dengan sensor lain (misalnya LiDAR, IMU) untuk menentukan posisi dan arah pergerakan kereta otomatis.

Maintenance Prediktif: Menganalisis kondisi rel dari hasil segmentasi untuk mendeteksi kerusakan sebelum terjadi kegagalan.

## 5. Assignment

- **Ubah parameter Canny Edge Detection dan bandingkan hasilnya.**

Parameter Threshold 1: (50, 150)

- Deteksi tepi lebih sensitif terhadap perubahan warna dan tekstur.
- Lebih banyak noise terdeteksi, termasuk bagian yang tidak relevan seperti bayangan atau kerikil di sekitar rel.
- Rel tetap terlihat, tetapi ada banyak gangguan yang bisa menghambat segmentasi.

Parameter Threshold 2: (200, 250)

- Hanya tepi yang sangat jelas yang terdeteksi.
- Noise berkurang secara signifikan, tetapi beberapa bagian rel mungkin hilang dari deteksi.
- Jika pencahayaan kurang baik, jalur rel bisa saja tidak terdeteksi dengan sempurna.

- **Modifikasi model YOLOv8-seg agar hanya mendeteksi jalur rel.**

Percobaan membuat dataset baru **rail-dataset.pt** untuk mendeteksi jalur rel, dengan langkah-langkah berikut:

Pengambilan Data:

- Menggunakan video sebagai sumber dataset dan mengekstrak **1150 frame**.
- Frame diambil secara berkala untuk mencakup variasi sudut dan pencahayaan.

Anotasi Dataset:

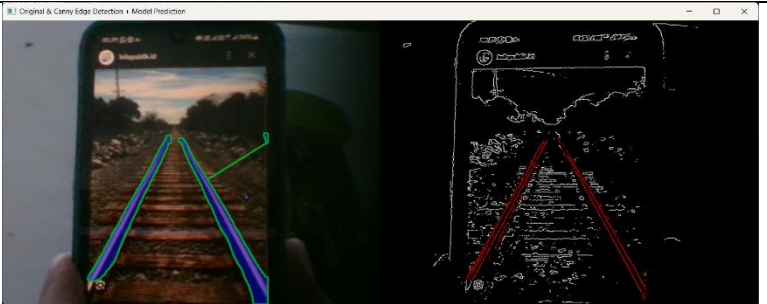
- Menggunakan **Roboflow** untuk memberi label **jalur rel** dalam setiap frame.
- Memastikan bahwa dataset memiliki **beragam kondisi** (siang/malam, basah/kering, dan berbagai perspektif).

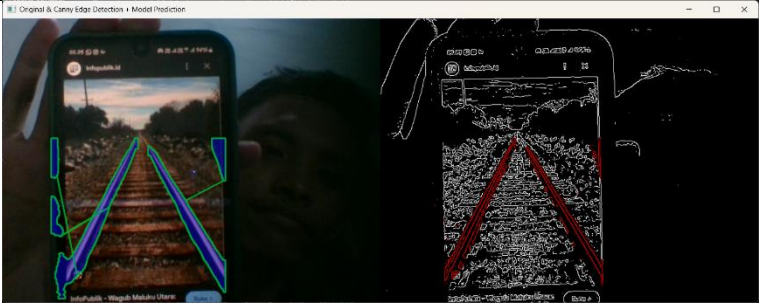
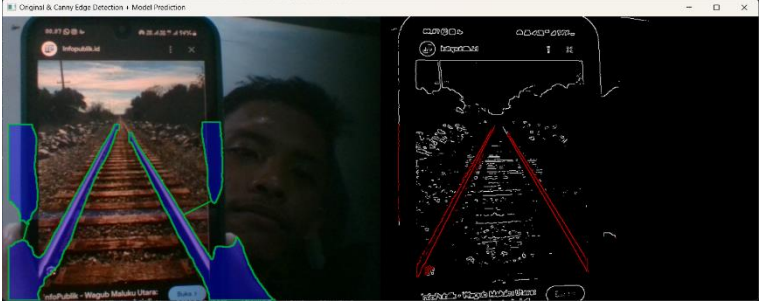
**Upload hasil eksperimen ke GitHub dan buat laporan analisis hasil.**

Hasil eksperimen telah diunggah ke GitHub berserta hasil pengamatan, analisis dan diskusi beserta laporan praktikum.

## 6. Data dan Output Hasil Pengamatan

Data dan hasil yang diperoleh selama percobaan.

No	Variabel	Hasil Pengamatan
1	Hasil Modifikasi model dataset YOLOv8-seg ( <b>rail-dataset.pt</b> )	

No	Variabel	Hasil Pengamatan
2	Hasil Pengujian Canny Edge Detection Parameter Threshold 1: (50, 150)	
3	Hasil Pengujian Canny Edge Detection Parameter Threshold 2: (200, 250)	

## 7. Kesimpulan

- Canny Edge Detection mampu mendeteksi tepi jalur rel dengan cepat, tetapi rentan terhadap noise dan tidak dapat membedakan objek berdasarkan konteks.
- Instance Segmentation dengan YOLOv8-seg lebih akurat dalam mendeteksi jalur rel karena mampu mengenali bentuk keseluruhan rel dan membedakannya dari objek lain di sekitar.
- Pemilihan parameter yang optimal bergantung pada kondisi pencahayaan dan kualitas gambar yang digunakan.

## 8. Saran

- Menambahkan lebih banyak variasi kondisi lingkungan (misalnya hujan, kabut, perubahan sudut kamera) untuk meningkatkan generalisasi model (dataset).
- Mencoba **Adaptive Canny Thresholding** agar threshold dapat menyesuaikan dengan kontras gambar secara otomatis.

## 9. Daftar Pustaka

1. Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.
2. Jocher, G., et al. (2023). YOLOv8 Documentation. Ultralytics.
3. Roboflow. (2023). Roboflow: Computer Vision for Everyone. Retrieved from <https://roboflow.com>.