PREDIKSI JALUR JALAN BERBASIS KALMAN FILTER PADA SISTEM *ADAS (ADVANCE DRIVING ASSISTANCE SYSTEM)* MENGGUNAKAN *HOUGH TRANSFORM* DENGAN KAMERA BERSUDUT PANDANG TERBATAS

Andreas Edi Widyartono*, Yohanes Agung Purwoko, dan I Nyoman Kesawa
Mesin Otomotif, Politeknik Astra, Cikarang, Bekasi, Indonesia
E-mail : edi.widyartono@polytechnic.astra.ac.id*

Abstract*--Road lane detection is an important component in intelligent driver assistance systems for autonomous vehicles. However, challenges arise when the camera is mounted on the upper windshield of the vehicle with a limited viewing angle, especially when the vehicle turns. This limitation causes the region of interest (ROI) area to become narrow, or even the lane is not detected at all. This study proposes a combination approach between the Hough Transform Probabilistic (HoughLinesP)-based line detection method and lane position prediction using the Kalman Filter. When the lane is only partially detected or not detected due to the limited camera angle, the Kalman Filter is used to predict the lane position based on the movement of previous points. Experimental results show that this approach is able to maintain stable and accurate lane position estimation even though there is a loss of some visual information due to the limited camera view. This method improves the reliability of the lane detection system in dynamic environmental conditions and narrow viewing angles.

Keywords: Road Lane Detection, Region of Interest, Kalman Filter

Abstrak*--Deteksi jalur jalan *(road lane detection)* merupakan komponen penting dalam sistem bantuan pengemudi cerdas kendaraan otonom. Namun, tantangan muncul ketika kamera dipasang pada kaca depan bagian atas kendaraan dengan sudut pandang terbatas, terutama saat kendaraan berbelok. Keterbatasan ini menyebabkan area *region of interest (ROI)* menjadi sempit, bahkan jalur tidak terdeteksi sama sekali. Penelitian ini mengusulkan pendekatan kombinasi antara metode deteksi garis berbasis Hough Transform Probabilistik (HoughLinesP) dan prediksi posisi jalur menggunakan Kalman Filter. Ketika jalur hanya sebagian terdeteksi atau tidak terdeteksi karena keterbatasan sudut kamera, Kalman Filter digunakan untuk memprediksi posisi jalur berdasarkan pergerakan titik-titik sebelumnya. Hasil eksperimen menunjukkan bahwa pendekatan ini mampu mempertahankan estimasi posisi jalur secara stabil dan akurat meskipun terjadi kehilangan sebagian informasi visual akibat keterbatasan pandangan kamera. Metode ini meningkatkan keandalan sistem deteksi jalur dalam kondisi lingkungan dinamis dan sudut pandang sempit.

Kata Kunci* : *Road Lane Detection, Region of Interest, Kalman Filter

I. PENDAHULUAN

Kondisi jalan yang semakin ramai dengan banyaknya jumkah kendaraan yang melintas dapat mengakibatkan lalu lintas semakin padat dan mempengaruhi waktu tempuh menjadi lebih lama. Kondisi ini dapat menurunkan tingkat konsentrasi pengemudi karena faktor kelelahan dan salah satunya mengakibatkan kendaraan keluar jalur[1]. Antisipasi kejadian yang tidak diinginkan sudah dilakukan oleh produsen kendaraan, salah satunya adalah sistem peringatan kepada pengemudi berupa notifikasi[2] bahkan tindakan misalnya koreksi kemudi supaya kendaraan kembali ke jalur yang seharusnya[3]. Sistem peringatan ini bekerja dengan cara mendeteksi jalur jalan (berupa garis/marka jalan) melalui kamera yang terpasang di kendaraan. Deteksi jalur jalan *(lane detection)* merupakan salah satu komponen utama dalam sistem bantuan pengemudi cerdas (ADAS) dan kendaraan otonom. Teknologi ini memungkinkan kendaraan untuk mengenali batas jalur secara real-time[4] sehingga dapat mempertahankan posisi yang aman di dalam lintasan[5][6] Salah satu metode yang umum digunakan dalam deteksi jalur adalah transformasi *Hough probabilistik (HoughLinesP)*, yang mampu mengenali garis-garis lurus dari citra biner hasil pemrosesan awal.[7][8]

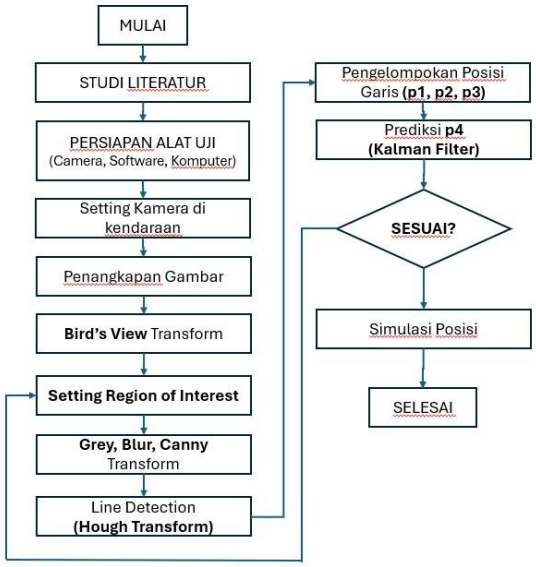
Efektivitas metode ini bergantung pada posisi dan sudut pandang kamera yang digunakan. Dalam banyak kasus, kamera dipasang di bagian atas kaca depan kendaraan untuk alasan praktis dan keamanan tetapi posisi ini memiliki keterbatasan sudut pandang, terutama saat kendaraan berbelok atau melintasi jalan dengan kelokan tajam[9]. Hal ini menyebabkan sebagian atau seluruh jalur tidak terdeteksi karena berada di luar area pandang kamera, sehingga mengurangi keakuratan dan keandalan sistem.[10]

Untuk mengatasi masalah ini, diperlukan mekanisme prediksi yang dapat memperkirakan posisi jalur meskipun tidak seluruh garis terlihat oleh kamera. Salah satu metode prediksi yang efektif

adalah Kalman Filter[11][12], yang mampu memperkirakan keadaan sistem berdasarkan pengamatan sebelumnya. Dengan menggabungkan hasil deteksi jalur melalui *HoughLinesP* dan prediksi dari Kalman Filter[13], sistem diharapkan dapat mempertahankan estimasi posisi jalur berikutnya berdasarkan data posisi sebelumnya.[14]

II. METODOLOGI PENELITIAN

Penelitian ini terdiri dari beberapa tahapan utama yang dirancang untuk mendeteksi jalur jalan secara andal meskipun terdapat keterbatasan sudut pandang kamera. Adapun tahapan metodologi sebagai berikut:



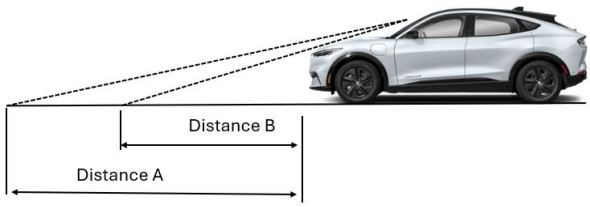
Gambar 4. Alur Penelitian

1. Pemasangan kamera (pengambilan gambar)

Gambar diperoleh dari kamera yang dipasang di kaca depan bagian atas kendaraan, posisi yang umum digunakan pada sistem ADAS[15][16]. Kamera ini merekam kondisi jalan di depan kendaraan, termasuk jalur jalan *(lane marking)*.

Alat yang digunakan adalah :

- Kendaraan jenis Sedan (Honda Brio 2016)
- Kamera Webcam Logitech C270 (0,9 mp)
- Laptop MSI GL65 9SC 028ID Intel Core i7 9750H RAM 8GB SSD 512GB
- Interpreter Pycharm
- Software OpenCV-Python



Gambar 2. *Layout* kamera di kendaraan

Gambar 2 menunjukkan layout kamera yang hasil tangkapan gambarnya mempunyai korelasi jarak dengan kendaraan, misalnya: jarak Distance A dan Distance B mempunyai letak yang berbeda dalam piksel.

Gambar 3. Lokasi kamera di kaca depan kendaraan

Gambar 3 adalah lokasi kamera yang terpasang di kaca depan kendaraan yang sering digunakan oleh pembuat mobil untuk menempatkan kamera. Lokasi ini dinilai tepat karena mempunyai sudut pandang yang luas.

Gambar 4. Hasil Pandangan dari Kamera

Gambar 4 menunjukkan pandangan dari kamera yang terpasang di kaca depan kendaraan dalam kondisi awal dan dan dilakukan penyesuaian ukuran menjadi 420X360.

2. Pemilihan area yang menjadi fokus

Tidak semua area akan dideteksi,[17] dalam hal ini adalah area yang akan memberi dampak secara langsung sesuai arah pergerakan kendaraan.[18][19] Dengan menggunakan Code dibawah ini, maka akan didapatkan gambar dengan posisi x dan y.

```
import cv2
import matplotlib.pyplot as plt

img = cv2.imread("Tech_Bitung_lurus.jpg")

plt.imshow(img)
plt.show()
```

Gambar 5. Area yang berpengaruh secara langsung

Gambar 5 menunjukkan gambar dan posisi tiap titik dalam piksel.

Dengan mengetahui koordinat setiap pandangan, maka dapat ditandai dengan beberapa titik sebagai batasan atau fokus pada area yang paling berdampak[20]. Salah satu cara untuk memudahkan dalam hal ini adalah transformasi pandangan menjadi *Bird's View* sehingga area yang tidak berpengaruh menjadi tidak nampak.

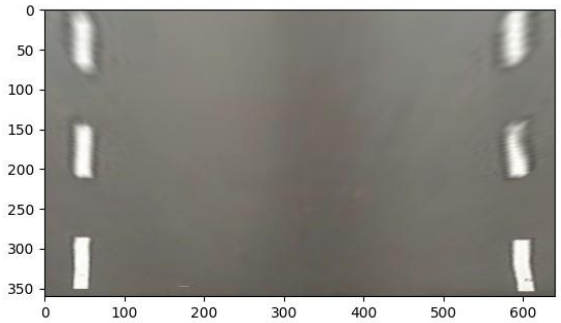
3. Transformasi pandangan menjadi *Bird's View*

Code dibawah ini adalah untuk transformasi ke dalam pandangan *Bird's View*

```
pts1 = np.float32
([[10,360],[630,360],[355,210],[220,210]])

pts2 = np.float32
([[10,360],[630,360],[630, 100],[0,100]])

matrix = cv2.getPerspectiveTransform(pts1,
pts2)
image = cv2.warpPerspective(img, matrix, (640,
360))
```



Gambar 6. Area transformasi *Bird's View*

Gambar 6 menunjukkan pandangan hasil transformasi, apabila dari gambar tersebut dideteksi dengan Houghline transform (pendeteksi garis) dengan Code terlampir di bawah ini akan didapatkan koordinat garis sebagai berikut :

Code :

```
gray = cv2.cvtColor(image,
cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (5,5),0)
canny = cv2.Canny(blur, 50, 150)
lines = cv2.HoughLinesP
(canny, 2, np.pi/180, 10, 10, 1)
if lines is not None:
    for line in lines:
        x1, y1, x2, y2 = line.reshape(4)
        cv2.line(image, (x1, y1), (x2, y2), (0, 255, 0), 5)
    print(lines)
```

Hasil :

```
[[[ 38 206  38 194]]

 [[ 55 339  55 321]]

 [[589 324 589 327]]

 [[ 58 205  58 201]]

.....(sebagian dihapus karena terlalu banyak)

 [[ 39 149  39 151]]]
```

Dimana :

Angka pertama, kedua, ketiga, dan keempat adalah [X1, Y1, X2, Y2] atau (x,y) awal dan akhir dari sebuah garis.

Hasil garis yang terdeteksi adalah secara acak, tidak sesuai urutan tertentu dan tidak dapat digunakan sebagai koordinat untuk diprediksi dengan Kalman Filter. Solusinya adalah dengan membagi menjadi 3 bagian dan koordinat garis yang terjadi akan terbagi dalam 3 kelompok untuk selanjutnya dibuat rata-rata dari tiap kelompok sehingga diharapkan akan terjadi deret sebagai berikut :

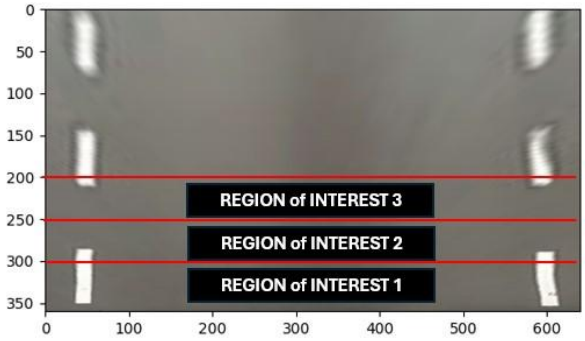
P1 = Posisi 1 (X1, Y1)
P2 = Posisi 2 (X2, Y2)
P3 = Posisi 3 (X3, Y3)
.....
P4 = Posisi 4 (X4, Y4) – Prediksi dengan Kalman Filter

4. Pembagian Gambar menjadi beberapa *Region of Interest*

Code untuk membagi region/area masing-masing adalah sebagai berikut :

```
def region_of_interest_01(image):
    polygons = np.array
```

```
([[(0,360),(640, 360), (640, 300), (0, 300)]])
    mask = np.zeros_like(image)
    cv2.fillPoly(mask, polygons, 255)
    masked_image_01 = cv2.bitwise_and(image,
mask)
    return masked_image_01
```

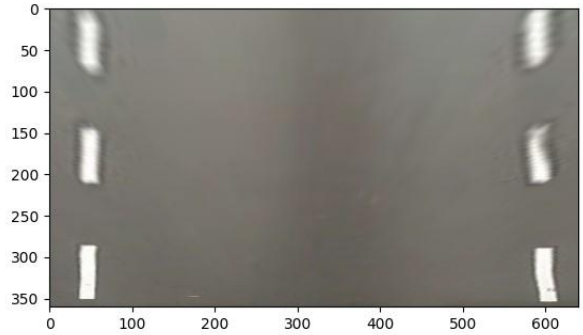


Gambar 7. Pembagian menjadi 3 *Region of Interest*

Gambar 7 menunjukkan gambar yang terbagi dalam beberapa area dan menyisakan area yang berguna untuk pengujian dari sistem yang dirancang, sehingga diharapkan akan terdeteksi deret posisi garis di setiap *Region of Interest* yang diperlukan dalam prediksi posisi garis berikutnya.

III. HASIL DAN PEMBAHASAN

Pengujian dilakukan dengan melibatkan perangkat yang telah dipersiapkan sebelumnya dalam satu kesatuan sistem, menggunakan gambar sebagai berikut:



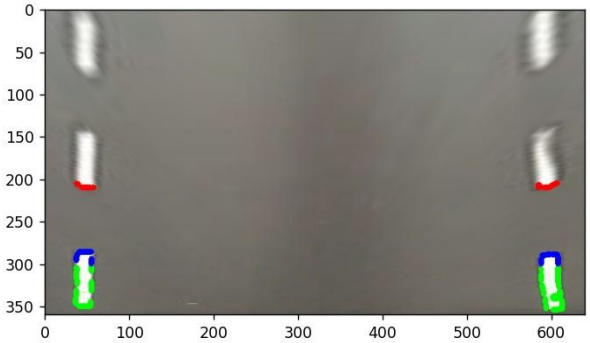
Gambar 8. Gambar untuk uji coba 1

Gambar 8 adalah gambar untuk uji coba 1 yang merupakan lintasan lurus.

Deteksi garis dalam 3 Region of Interest
Dengan menggunakan Code dibawah ini dan diterapkan pada masing-masing *Region of Interest,*

```
def lines_01(croped_image_01):
    left_fit_A = []
    right_fit_A = []
    xLA_list = []
    yLA_list = []
    xRA_list = []
    yRA_list = []
    linesA = cv2.HoughLinesP(croped_image_01, 2,
np.pi / 180, 10, 10, 1)
    if linesA is not None:
        for line in linesA:
            x1, y1, x2, y2 = line.reshape(4)
            x1, y1, x2, y2 = int(x1), int(y1), int(x2),
int(y2)
            cv2.line(image, (x1, y1), (x2, y2), (0, 255, 0),
5)
            fit_A = (x1+x2)/2
            if fit_A<250:
                left_fit_A.append([x1, y1, x2, y2])
                xLA_list.append((x1+x2)/2)
                yLA_list.append((y1+y2)/2)
            if fit_A>350:
                right_fit_A.append([x1, y1, x2, y2])
                xRA_list.append((x1+x2)/2)
                yRA_list.append((y1+y2)/2)
    XL1 = int(np.mean(xLA_list))
    YL1 = int(np.mean(yLA_list))
    XR1 = int(np.mean(xRA_list))
    YR1 = int(np.mean(yRA_list))
    xy1_01_left = [(x1, y1) for x1, y1, x2, y2, in
left_fit_A]
    xy1_01_right = [(x1, y1) for x1, y1, x2, y2 in
right_fit_A]
    return XL1, YL1, XR1, YR1, xy1_01_left,
xy1_01_right
```

maka akan didapatkan hasil sebagai berikut:



Gambar 9. Hasil uji coba 1

Dari Gambar 9 terlihat ada 3 area yang terdeteksi, masing-masing dengan warna yang berbeda dan menunjukkan 3 kelompok berbeda yang dapat digunakan dalam melakukan prediksi dengan Kalman Filter.
Hasil koordinar garis dari Gambar 9 dan dipisahkan hanya (x1, y1) menggunakan Code :

```
xy1_01_left = [(x1, y1) for x1, y1, x2, y2, in
left_fit_A]
   xy1_01_right = [(x1, y1) for x1, y1, x2, y2 in
right_fit_A]
```

adalah sebagai berikut :
[(54, 350), (55, 309), (37, 309), (55, 332), (40, 350), (37, 338), (38, 324), (37, 304), (36, 347)]
[(588, 312), (608, 309), (593, 341), (608, 318), (610, 332), (591, 332), (613, 352), (609, 327), (605, 340), (609, 353), (602, 354), (602, 341), (608, 308), (589, 325), (610, 341), (605, 337), (609, 338), (602, 340), (588, 306), (613, 346), (612, 343)]
Selanjutnya data tersebut akan digunakan sebagai posisi awal dalam Kalman Filter.

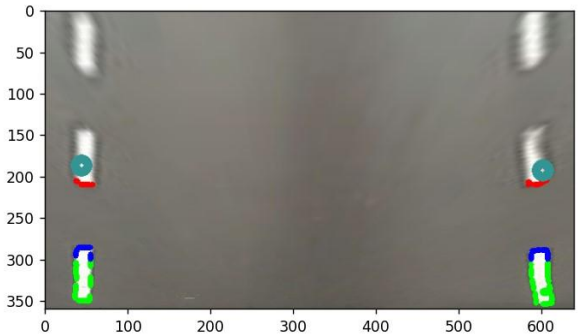
Prediksi menggunakan Kalman Filter
Setelah mengetahui koordinat garis sebelumnya, maka posisi selanjutnya dapat diprediksi.

```
def next_pos(data):
   positions = []
   for sub in data:
      for item in sub:
         positions.append(item)

   kalman = cv2.KalmanFilter(4, 2)  # 4 state vars: x,
y, dx, dy | 2 measurements: x, y
   kalman.measurementMatrix = np.array([[1, 0, 0,
0],
                                        [0, 1, 0, 0]], np.float32)
   kalman.transitionMatrix = np.array([[1, 0, 1, 0],
                                       [0, 1, 0, 1],
                                       [0, 0, 1, 0],
                                       [0, 0, 0, 1]], np.float32)
   kalman.processNoiseCov = np.eye(4,
dtype=np.float32) * 0.03
   kalman.statePre = np.array([[positions[0][0]],
                               [positions[0][1]],
                               [0],
                               [0]], np.float32)
   for pos in positions:
      measured = np.array([[np.float32(pos[0])],
[np.float32(pos[1])]])
      kalman.correct(measured)
      prediction = kalman.predict()
      x = prediction[0].item()
      y = prediction[1].item()
   print("x_prediction", x)
   print("y_prediction", y)
   cv2.circle(image, (int(x), int(y)), 8, (50, 150, 150),
10)
   return prediction
```

hasilnya :
x_prediction 44.111671447753906
y_prediction 187.95159912109375
x_prediction 602.7269897460938
y_prediction 193.03851318359375

Kemudian digambarkan seperti di bawah ini :

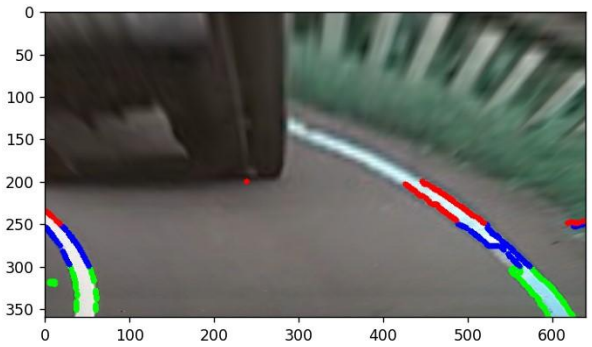


Gambar 10. Hasil uji coba 1 (Kalman Filter)

Gambar 10 terdapat lingkaran yang menunjukkan prediksi posisi berikutnya berdasarkan posisi dari koordinat garis yang terdeteksi.
Uji coba selanjutnya dengan menggunakan gambar sebagai berikut :


Gambar 11. Gambar uji coba 2

Gambar 11 merupakan gambar uji coba 2 dengan lintasan yang tidak lurus. Dengan menggunakan Code yang sama, maka hasilnya adalah sama dengan hasil uji coba 1, yaitu terlihat 3 bagian/area yang terdeteksi dan menunjukkan 3 kelompok garis dan bisa digunakan dalam prediksi garis (posisi) selanjutnya menggunakan Kalman Filter.



Gambar 12. Hasil uji coba 2

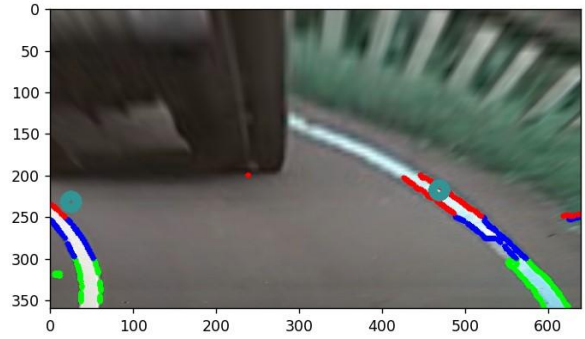
Gambar 12 menunjukkan hasil deteksi garis dari 3 Region Of Interest yang berbeda. Hasil koordinar garis dari Gambar 12 dan dipisahkan hanya (x1, y1) menggunakan Code :

```
xy1_01_left = [(x1, y1) for x1, y1, x2, y2, in
left_fit_A]
   xy1_01_right = [(x1, y1) for x1, y1, x2, y2 in
right_fit_A]
```

[(61, 335), (32, 306), (37, 352), (54, 303), (37, 327), (56, 310), (60, 345), (57, 313), (5, 320), (33, 308), (38, 332), (5, 318), (10, 321), (36, 324), (52, 301), (29, 300), (60, 333), (55, 306), (36, 318)]
[(590, 316), (558, 311), (580, 308), (586, 343), (578, 333), (556, 302), (582, 337), (552, 307), (583, 310), (621, 355), (551, 305), (588, 346), (571, 324), (619, 352), (562, 300), (559, 302), (606, 333), (576, 332), (595, 320), (562, 308), (594, 356), (612, 342), (577, 304), (579, 333), (565, 319), (585, 343), (618, 350), (553, 302), (576, 303), (593, 354)]

Hasil prediksi adalah sebagai berikut :
x_prediction 25.643701553344727
y_prediction 232.5470428466797
x_prediction 469.1783142089844
y_prediction 218.18069458007812

Kemudian digambarkan sebagai berikut :



Gambar 13. Hasil uji coba 2 (Kalman Filter)

Gambar 13 memperlihatkan hasil prediksi dari Kalman Filter tidak begitu presisi saat jalanan berbelok (terlihat dari posisi lingkaran yang tidak pada posisi seharusnya, secara visual garis yang terdeteksi tidak seimbang antara bagian kiri dan bagian kanan.

IV. KESIMPULAN

Penggunaan beberapa *Region of Interest*, garis-garis yang terdeteksi dapat dikelompokkan menjadi suatu deret posisi yang yang lebih terarah sesuai dengan arah pergerakan kendaraan.

Hasil prediksi pada jalanan yang lurus lebih presisi dari pada di jalanan yang berbelok, dalam hal ini kemungkinan disebabkan lingkup pandang yang lebih terbatas dan kualitas gambar *(image)* yang kurang mendukung karena pada saat transformasi gambar akan berubah bentuk.

Diperlukan penelitian lebih lanjut menggunakan peralatan yaitu kamera dengan posisi lebih luas dan semakin mendekati 90 derajat *(bird's view)* maka kualitasnya akan semakin baik dan perlakuan data awal, misalnya diurutkan terlebih dahulu sebelum diprediksi dengan Kalman Filter.

V. DAFTAR PUSTAKA

[1] N. Gyulyev, O. Prasolenko, E. Litomin, and D. Zinchenko, "Study of the influence of road congestion on the fatigue level of a sanguine driver," *Technology audit and production reserves*, vol. 6, no. 2(50), pp. 31–35, Nov. 2019, doi: 10.15587/2312-8372.2019.191275.

[2] Z. Luo, Y. Bi, Q. Ye, Y. Li, and S. Wang, "A Novel Machine Vision-Based Collision Risk Warning Method for Unsignalized Intersections on Arterial Roads," *Electronics (Switzerland)*, vol. 14, no. 6, Mar. 2025, doi: 10.3390/electronics14061098.

[3] M. M. Narkhede and N. B. Chopade, "Review of advanced driver assistance systems and their applications for collision avoidance in urban driving scenario," in *Machine Learning and Big Data Analytics (Proceedings of International Conference on Machine Learning and Big Data Analytics (ICMLBDA) 2021)*, 2022, pp. 253–267.

[4] Q. Cao, Z. Zhao, Q. Zeng, Z. Wang, and K. Long, "Real-Time Vehicle Trajectory Prediction for Traffic Conflict Detection at Unsignalized Intersections," *J Adv Transp*, vol. 2021, 2021, doi: 10.1155/2021/8453726.

[5] D. GHEORGHE, "The Need of Advanced Driver-Assistance System's Development based on an Analysis of Road Accidents," *Informatica Economica*, vol. 27, no. 3/2023, pp. 17–28, Sep.

2023, doi: 10.24818/issn14531305/27.3.2023.02.

[6] S. A. Useche, M. Faus, and F. Alonso, "'Cyclist at 12 o'clock!': a systematic review of in-vehicle advanced driver assistance systems (ADAS) for preventing car-rider crashes," 2024, *Frontiers Media SA*. doi: 10.3389/fpubh.2024.1335209.

[7] D. Cao *et al.*, "Research on Apple Detection and Tracking Count in Complex Scenes Based on the Improved YOLOv7-Tiny-PDE," *Agriculture (Switzerland)*, vol. 15, no. 5, Mar. 2025, doi: 10.3390/agriculture15050483.

[8] I. Kunina, E. I. Panfilova, O. S. Shipitko, and I. A. Kunina, "Fast Hough Transform-Based Road Markings Detection For Autonomous Vehicle." [Online]. Available: https://www.researchgate.net/publication/345309351

[9] Q. Huang and J. Liu, "Practical limitations of lane detection algorithm based on Hough transform in challenging scenarios," *Int J Adv Robot Syst*, vol. 18, no. 2, 2021, doi: 10.1177/17298814211008752.

[10] Y. Zhang, P. Gong, S. Ji, and Q. Xu, "Real-time lane detection method based on region of interest," in *2022 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS)*, 2022, pp. 1188–1192.

[11] Y. Zhang, Q. Xiao, X. Liu, Y. Wei, and J. Xue, "Method for reconstructing safety and arming motion process by integrating Kalman filter and KCF," *Sci Rep*, vol. 15, no. 1, Dec. 2025, doi: 10.1038/s41598-025-92957-y.

[12] A. Salhi, F. Ghozzi, and A. Fakhfakh, "Estimation for motion in tracking and detection objects with Kalman filter," in *Dynamic Data Assimilation-Beating the Uncertainties*, IntechOpen, 2020.

[13] S. A. Elsagheer Mohamed, K. A. Alshalfan, M. A. Al-Hagery, and M. T. Ben Othman, "Safe Driving Distance and Speed for Collision Avoidance in Connected Vehicles," *Sensors*, vol. 22, no. 18, Sep. 2022, doi: 10.3390/s22187051.

[14] H. Wei, Y. Huang, F. Hu, B. Zhao, Z. Guo, and R. Zhang, "Motion estimation using region-level segmentation and extended kalman filter for autonomous driving," *Remote Sens (Basel)*, vol. 13, no. 9, May 2021, doi: 10.3390/rs13091828.

[15] P. Shi, Z. Liu, X. Dong, and A. Yang, "CL-fusionBEV: 3D object detection method with camera-LiDAR fusion in Bird's Eye View," *Complex and Intelligent Systems*, Dec. 2024, doi: 10.1007/s40747-024-01567-0.

[16] M. Jakubec, M. Cingel, E. Lieskovská, and M. Drliciak, "Integrating Neural Networks for Automated Video Analysis of Traffic Flow Routing and Composition at Intersections," *Sustainability (Switzerland)*, vol. 17, no. 5, Mar. 2025, doi: 10.3390/su17052150.

[17] I. C. Chang *et al.*, "An Effective YOLO-Based Proactive Blind Spot Warning System for Motorcycles," *Electronics (Switzerland)*, vol. 12, no. 15, Aug. 2023, doi: 10.3390/electronics12153310.

[18] E. Casanova, D. Guffanti, and L. Hidalgo, "Technological Advancements in Human Navigation for the Visually Impaired: A Systematic Review," Apr. 01, 2025, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/s25072213.

[19] A. H. Pratomo, W. Kaswidjanti, and S. Mu'arifah, "IMPLEMENTASI ALGORITMA REGION OF INTEREST (ROI) UNTUK MENINGKATKAN PERFORMA ALGORITMA DETEKSI DAN KLASIFIKASI KENDARAAN IMPLEMENTATION OF REGION OF INTEREST (ROI) ALGORITHM TO IMPROVE CAR DETECTION AND CLASSIFICATION ALGORITHM," vol. 7, no. 1, pp. 155–162, 2020, doi: 10.25126/jtiik.202071718.

[20] L. C. Sousa *et al.*, "Obstacle Avoidance Technique for Mobile Robots at Autonomous Human-Robot Collaborative Warehouse Environments," *Sensors*, vol. 25, no. 8, Apr. 2025, doi: 10.3390/s2508238