Implemetasi Algoritma *You Only Look Once* (YOLO) menggunakan Web Camera untuk Mendeteksi Objek Statis dan Dinamis

Implementation of You Only Look Once (YOLO) Algorithm using Web Camera for Static dan Dinamic Object Detection

Jeremia Zophie[1], Hendri Himawan Triharminto[2]

[1,2,3,4] Departemen Elekronika, Akademi Angkatan Udara, Yogyakarta
hhtriharminto@aau.ac.id

Abstrak — Teknologi kamera telah mengalami perkembangan yang pesat. Saat ini kamera bukan hanya digunakan untuk mengambil citra atau membuat video secara utuh. Akan tetapi, kamera dapat digunakan untuk mendeteksi objek yang diinginkan. Pengembangan deteksi objek yang dilakukan menerapkan algoritma You Only Look Once (YOLO). Algoritma ini digunakan karena dijalankan pada dua framework yaitu Darknet dan Darkflow. Algoritma YOLO merupakan algoritma deep learning untuk mendeteksi objek yang menggunakan pendekatan berbeda dari algoritma lain, yaitu menerapkan sebuah jaringan syaraf tunggal pada keseluruhan citra. Pengembangan deteksi objek dilakukan pada interface desktop dengan bahasa C dan Phyton. Untuk menguji kehandalan dari algoritma YOLO, maka algoritma akan diujikan pada dua skenario. Pertama pengujian secara langsung yang dilakukan pada webcam yang telah diimplementasikan YOLO dan yang kedua secara tidak langsung dimana objek citra berupa citra JPG dan video mp4. Kedua skenario tersebut divariasikan kembali berdasarkan jarak dan intensitas cahaya. Hasil dari percobaan ini pengujian YOLO diukur menggunakan confusion matrix dan mendapatkan hasil akurasi sebesar 69,5652%, kesalahan klasifikasi sebesar 26,0869%, ketelitian sebesar 94,1176%, dan kekhususan sebesar 50%.

Kata Kunci — Webcam, Deteksi objek, YOLO

Abstract — *Nowadays, camera technology has been developed very rapidly. Camera does not merely use to capture image or video, but it is used to identify an object. The research implements You Only Look Once (YOLO) algorithm. This algorithm is run by two unique frameworks, i.e. Darknet and Darkflow. This algorithm is one type of deep learning algorithm that applies a single neural network to the whole picture for object detection. The development also accommodated by an interface application developed in C and Phyton. Experiment of the research are conducted using two different scenarios. The first scenario is online and second is offline. Online method uses the YOLO algorithm on an webcam and the object detection will occur through on the stream, while in the offline method, the algorithm will process a file which JPG for a picture and mp4 for a video. The distance and light intensity is vary for both of the scenarios. The results are detetermined by using confusion matrix. The result shows that this algorithm achieved 69.5652% for accuracy, 26,0869% for classification error, 26,0869% for precision, and 50% for exclusiveness.*

Keywords— Webcam, Deteksi objek, YOLO

I. PENDAHULUAN

Perkembangan teknologi menuntut manusia untuk menciptakan hal-hal baru yang bermanfaat untuk kehidupan sehari-hari. Salah satu teknologi yang telah mengalami perkembangan yang pesat adalah *computer vision* dengan menggunakan kamera. Saat ini aplikasi *computer vision* sangat beragam pada berbagai aktivitas kehidupan sehari-hari, seperti alat tilang otomatis dan melacak target. Dengan teknologi yang berkembang sekarang ini, kamera tidak hanya berbentuk kamera digital pada biasanya. Namun, kamera sekarang dapat melekat pada benda elektronik apa saja, seperti pada laptop yang bernama *web camera* (webcam).

Agardapatmelakukanpengenalanobjek,maka kamera harus mengaplikasikan pengolahan citra. Beberapa algoritma pengolahan citra untuk mendeteksi objek yaitu Deep Learning [1][2]. Deep learning terbagi menjad dua kategori yaitu *region-based method* dan *end-to-end method*. Pada *region-based method* melakukan pendekatan dengan membentuk *bounding box* dengan berbagai macam metode seperti *sliding window*[3], *selective search* [4], dan fitur dari tiap objek area akan diekstrasi dengan Convolutional Neural Network (CNN)[5][6][7]. CNN menerapkan lapisan konvolusi di dalam jaringan untuk gambar tertentu, mengidentifikasi fitur yang memiliki kesamaan spasial dari pola yang ditemukan dalam citra. Kelemahan dari CNN adalah menghasilkan performa yang kurang bagus pada objek yang kecil.

Pendekatan kedua untuk *end-to-end method* dalam deteksi objek adalah You Only Look Once (YOLO) [8][9], yang membagi citra menjadi grid[10]. Masing-masing gridcell bertindak sebagai acuan dari objek yang akan diindentifikasi yang telah ditentukan sebelumnya. Saat proses pelatihan, klasifikasi keluaran akan dibandingkan pada setiap *bounding box* untuk satu kali iterasi. Hal tesebut membuat algoritma YOLO mempunyai komputasi yang cukup cepat.

Pada penelitian akan menerapkan algoritma YOLO dengan menggunakan Bahasa Python untuk penyeleksian citra dan Bahasa C untuk pembuatan aplikasi. YOLO adalah salah satu algoritma yang dapat mendeteksi objek dengan kecepatan proses dan tingkat akurasi yang tinggi. Algoritma ini digunakan karena dijalankan di dua framework yaitu Darknet dan Darkflow dan didukung oleh Graphics Processing Unit (GPU).

II. LANDASAN TEORI

Algoritma YOLO merupakan algoritma *deep learning* dengan menerapkan sebuah jaringan syaraf tunggal pada keseluruhan citra. YOLO mendeteksi sebuah objek dalam beberapa tahap yaitu [10]:

A. Membagi citra dalam region/grid berukuran sxs

Grid-grid tersebut bertanggung jawab untuk mendeteksi objek. Pada tiap grid juga akan diprediksi *bounding box* beserta nilainya. Nilai ini menunjukkan kepastian *bounding box* tersebut berisi objek dan seberapa akurat prediksinya. Nilai confidence diperoleh melalui persamaan berikut.

$$(class) = \Pr(class) x I 0 U \Pr edTruth \quad (1)$$

dimana $(class)$ adalah probabilitas objek yang muncul dalam suatu region dan $Truth$ adalah rasio tumpang tindih *Intersection Over Union* (IOU) antara *bounding box* prediksi dan *bounding box ground truth*. Pred adalah luas area dalam *bounding box* prediksi, Truth adalah area dalam *ground truth*. Makin besar nilai IOU, maka makin tinggi tingkat akurasi pendeteksiannya

B. Tiap bounding box pembatas memiliki 5 nilai informasi.

Tiap *boundingbox* akanmemilikinilaiinformasi *x*, *y*, *w*, *h*, dan *c*. Nilai *x* dan *y* adalah koordinat titik tengah *bounding box* yang terprediksi, nilai *w* dan *h* adalah rasio ukuran lebar dan tinggi relatif terhadap grid, dan c adalah nilai confidence *bounding box* tersebut.

C. Tiap grid akan memprediksi nilai class probabilitas jika diprediksi terdapat objek di dalam-nya.

$$\Pr(class \mid object)x\Pr(object)xIOU\ \Pr edTruth = \Pr(class)xIOU\ \Pr edTruth \quad (2)$$

Sehingga menghasilkan nilai *confidence* kelas secara spesifik pada tiap *bounding box*. Nilai ini menunjukkan *class probability* yang muncul pada *bounding box* dan seberapa akurat *bounding box* memprediksi sesuai dengan objek.

Pada langkah identifikasi objek, citra yang dimasukan akan dibagi dengan ukuran 52x52. Setiap sel akan memprediksi objek yang ada di dalam sel tersebut. Objek yang diprediksi dalam setiap sel akan diberi kotak yang disebut *bounding box*. *Bounding box* ini akan menghitung seberapa persen objek yang di dalam mirip dengan data yang telah dilatih. Berdasarkan hasil data yang telah dilatih tersebut, maka akan didapatkan nilai *confidence* dari objek yang dideteksi. Di sisi lain, citra dibagi berdasarkan warna citra yang diperoleh. Kemudian dari kedua sisi ini digabungkan, dari sekian banyak *bounding box* yang dihasilkan dan pewarnaan berdasarkan objek yang diprediksi, maka dari seluruh hasil prediksi tersebut hanya yang melampaui threshold saja yang digunakan. Jika terdapat duplikasi atau tumpeng tindih antara *bounding box*, maka akan diterapkan metode Non-Max Suppresion (NMS), yaitu penilaian berdasarkan persentase nilai *confidence*, berperan untuk menghilangkan duplikat tersebut.

III. METODE PENELITIAN

A. Flowchart Metode Penelitian.

Gambar 1 adalah susunan alur diagram penelitian yang dilakukan pada penelitian dengan terbagi menjadi 4 tahapan.

Gambar 1 memperlihatkan dengan tahapan awal melakukan studi literatur untuk dapat merumuskan masalah yang ada. Masalah yang dihadapi dalam penelitian ini adalah mengimplementasikan algoritma YOLO pada webcam untuk mendeteksi objek dan menguji akurasi dari algoritma YOLO pada deteksi objek. Objek yang akan dideteksi pada penelitian ini adalah kendaraan bermotor yang berada di lingkungan Akademi Angkatan Udara. Tahap selanjutnya adalah perancangan dimana pada tahap ini akan dilakukan pemrograman terhadap webcam dengan bahasa python yang menggunakan

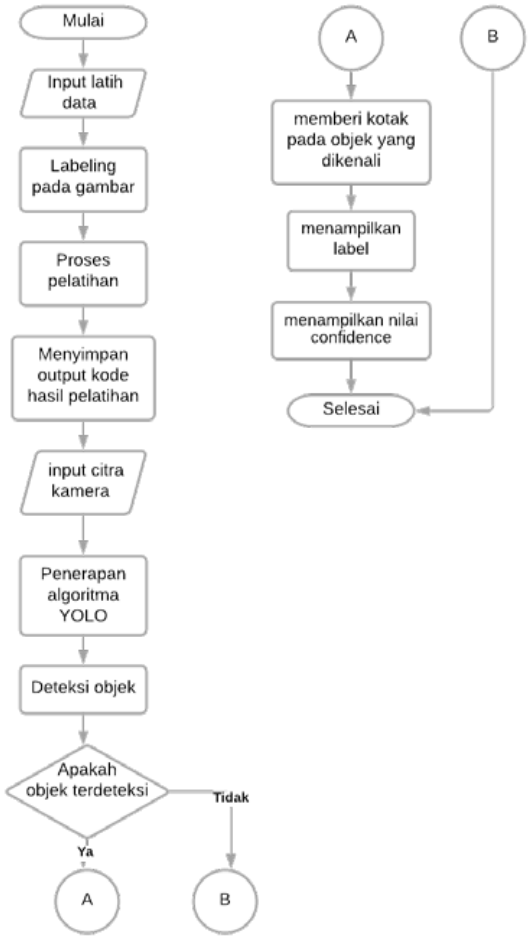
aplikasi Microsoft Visual Studio Code dengan Anaconda. Setelah proses perancangan selesai, tahapan berikutnya akan dilakukan pengujian terhadap algoritma YOLO yang sudah diimplementasikan pada hadrware. Dalam perancangan ini akan menggunakan sampel gambar sebagai dataset untuk awal mula dari pendeteksian objek. Gambar disini diseleksi terlebih dahulu dengan aplikasi LabelImg guna menentukan objek mana yang akan dideteksi. Nama gambar ini akan masuk ke dalam file berbentuk .txt guna memudahkan pembacaan pada pendeteksian objek nantinya. Dataset kemudian dilatih dengan dimasukkan dalam algoritma YOLO untuk memastikan gambar mana yang akan dideteksi. Dataset disini memerlukan banyak gambar karena semakin banyak data yang dilatih, maka semakin akurat pendeteksian objek pada gambar tersebut. Pada bagian akhir akan ditarik kesimpulan serta saran penelitian lanjutan yang dilakukan.

Gambar 1. Diagram alir penelitian

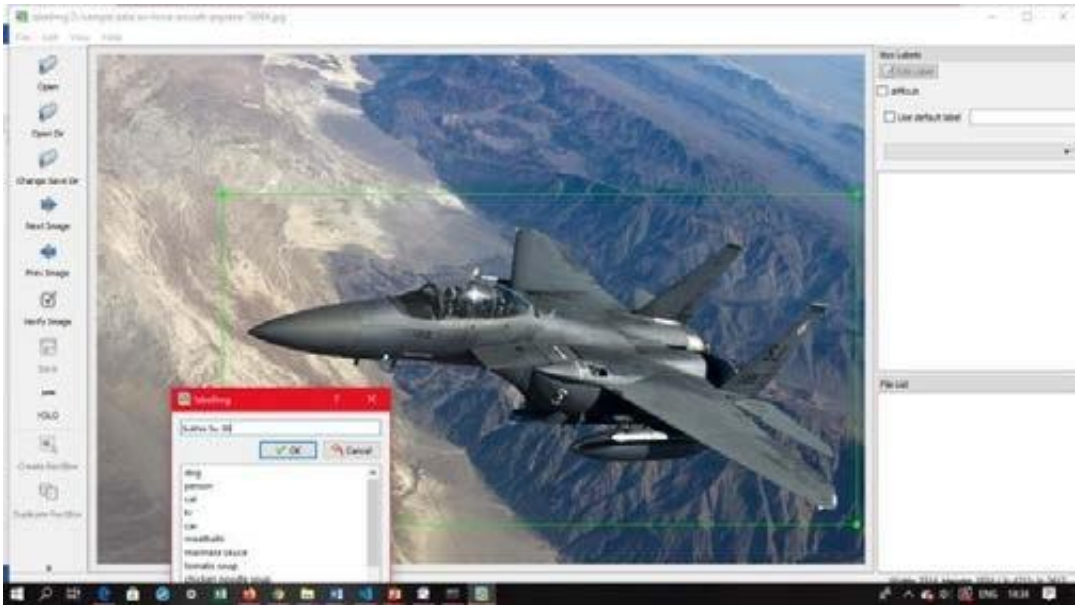
B. Blok Diagram Implementasi YOLO

Pada gambar 2 menjelaskan tentang alur secara umum proses pengolahan citra dalam pendeteksian objek.

Gambar 2 menjelaskan proses diawali dengan input latih data. Pada tahap ini, gambar-gambar dikumpulkan melalui internet, foto, maupun sumber lainnya untuk mendapatkan data yang banyak agar menampilkan persentase yang mendekati sempurna. Pada perancangan ini akan memakai 100 data per masing-masing objek. Setelah gambar dikumpulkan, maka proses selanjutnya adalah melakukan labeling pada Gambar. Gambar yang telah disimpan dalam folder akan dimasukkan satu persatu ke dalam aplikasi LabelImg yang akan melakukan penyeleksian pada objek yang akan dideteksi nantinya. Objek yang diseleksi akan diberi label dan mengubah format pendeteksian menjadi YOLO. Pada gambar 3 terdapat file bernama .xml dimana file tersebut merupakan hasil dari labeling dan terdapat file .text dimana file tersebut berisi nama dari objek yang telah dikumpulkan.



Gambar 2. Proses dalam Mendeteksi Objek



Gambar 3. Melakukan labeling pada gambar

Gambar 3 menjelaskan tentang pelabelan atau penamaan kepada objek yang diseleksi tersebut. Gambar yang telah diseleksi disimpan dengan kode-kode penyeleksian dengan format .xml. File classes merupakan file yang berisi nama objek yang telah dideteksi pada proses labeling.

Setelah data yang dikumpulkan telah memasuki proses labeling, maka data tersebut akan diubah menjadi format file .xml yang gambar disini diubah menjadi kode-kode yang dapat dibaca oleh Bahasa Python. Dengan memasukkan kode gambar tersebut ke dalam program training YOLO yang akan dijadikan sebagai dasar pendeteksian. Metode YOLO akan membagi suatu citra atau gambar masukan dalam suatu grid berukuran SxS. Setiap sel bertugas untuk memprediksi objek yang ada di dalam *bounding box* beserta confidence yang merupakan nilai probabilitas keberadaan suatu objek tersebut. Bounding box disini memprediksi gambar berdasarkan nilai RGB yang sama. Kemudian, setelah *bounding box* dipetakan berdasarkan nilai confidence yang dihasilkan, YOLO akan memprediksi kelas dari objek yang terdapat dalam *bounding box* tersebut dengan probabilitasnya. Setelah melalui metode YOLO, gambar akan dikenali apakah ada objek yang terdeteksi. Pendeteksian ini didasarkan pada data training yang dimasukkan pada tahap awal pemrosesan. Setelah melalukan pendeteksian jika ada objek yang terdeteksi maka akan dilakukan pemberian *bounding box* kepada objek yang terdeteksi tersebut dan pemberian label nama objek beserta nilai confidence yang mana nilai confidence ini berdasarkan berapa persentase objek yang terdeteksi tersebut. Sebaliknya jika tidak terdeteksi objek pada gambar masukan, maka gambar akan tetap dan tidak ada proses pelabelan maupun pemetaan dikarenakan nilai confidence bernilai 0 dan tidak ada *bounding box*.

IV. HASIL DAN PEMBAHASAN

A. Pengujian secara tidak Langsung

Paada pengujian tidak langsung, menggunakan data yang telah diambil dan disimpan dalam bentuk jpg ataupun mp4.

1) Objek pada Citra hanya satu

Pada citra dengan jumlah objek satu, citra yang digunakan berupa citra motor dan mobil yang dideteksi menggunakan bobot yang berbeda yaitu 1000, 2000, 3000 dan 4000 seperti pada gambar berikut ini.


```
Loading weights from yolov3_custom_1000.weights...
 seen 64, trained: 64 K-images (1 Kilo-batches_64)
Done! Loaded 107 layers from weights-file
 Detection layer: 82 - type = 28
 Detection layer: 94 - type = 28
 Detection layer: 106 - type = 28
DATA/DATA (40).JPG: Predicted in 2365.673000 milli-seconds.
mobil: 85%
```

Gambar 4. Penerapan YOLO dengan bobot 1000 iterasi

Gambar 4 adalah contoh pengujian dengan bobot 1000 iterasi. Perbedaan yang dihasilkan akibat perbedaan bobot adalah akurasi dan posisi *bounding box*. Pada iterasi 1000 mempunyai akurasi sebesar 85% akurasi dan *bounding box* yang dihasilkan tidak tepat sesuai objek pada citra. Sedangkan pada bobot 2000 iterasi, hasil akurasi mencapai 95% dan *bounding box* yang hampir sebesar objek pada citra. Pada iterasi 3000, hasil akurasi mencapai 100% akurasi, akan tetapi *bounding box* belum sesuai dengan besarnya objek pada citra. Dan yang terakhir ketika bobot iterasi 4000, akurasi mencapai 100% dan *bounding box* sesuai dengan objek mobil pada citra.

2) Objek pada Citra lebih dari satu

Objek pada citra berjumlah 2 objek yaitu mobil dan motor. Tujuan dari pengujian ini adalah untuk mendeteksi objek yang terdeteksi pada citra apabila berjumlah lebih dari 1. Iterasi yang digunakan dalam pengujian ini memakai iterasi yang terbaik yaitu 4000 iterasi. Dalam skenario pengujian algoritma, pada citra terdapat objek yang saling bertumpukan seperti pada gambar 5.

Gambar 5. Penerapan YOLO dengan objek lebih dari satu dan saling bertumpuk

Dari gambar diatas dapat dilihat bahwa hasil pendeteksian pada motor yang tertumpuk tidak dapat terdeteksi dengan baik sehingga kedua objek dianggap sebagai satu objek.

3) Intensitas Cahaya.

Pada pengujian ini intensitas cahaya menjadi parameter dalam menilai akurasi dari algoritma YOLO. Skenario dari citra terbagi menjadi 2 kategori yaitu antara gambar dengan intensitas terang dan gelap. Citra dengan pencahayaan yang terang dapat terlihat pada gambar berikut ini.

Gambar 6. Citra dengan Pencahayaan Terang

Dari hasil pengujian, pada intensitas cahaya yang terang memiliki nilai akurasi 100%. Sedangkan pada intensitas cahaya yang gelap terdapat objek yang tidak dapat terdeteksi dengan baik seperti pada gambar 7.

Gambar 7. Citra dengan Pencahayaan Gelap

Pada Gambar 7 terlihat bahwa algoritma YOLO tidak dapat mendeteksi mobil yang berwarna hitam pada intensitas pencahayaan yang gelap.

4) Resolusi Citra.

Resolusi citra menentukan ketajaman citra pada objek yang dideteksi. Jika gambar yang diambil memiliki pixel yang kurang maka gambar akan menjadi blur atau tidak jelas. Pada percobaan ini menerapkan algoritma YOLO dalam mendeteksi objek dalam gambar yang mempunyai pixel 299x169 dan pixel 1280x853 dalam setiap iterasi 1000, 2000, 3000, dan 4000. Contoh pengujian dari citra dengan resolusi yang berbeda dapat dilihat pada gambar 8.

a b

Gambar 8. Citra dengan Resolusi yang Berbeda

Gambar 8a mempunya resolusi yang lebih rendah dari gambar 8b. Pada bobot iterasi 1000, citra yang mempunyai resolusi lebih rendah menghasilkan tingkat akurasi yang lebih buruk dari citra yang mempunyai resolusi lebih tinggi. Dengan bobot iterasi 2000 dan 3000 juga menghasilkan akurasi citra yang kurang optimal pada citra dengan resolusi rendah. Hasil akurasi yang sama dapat diperoleh apabila bobot iterasi 4000 dimana seluruh bagian objek dapat terdeteksi dengan baik.

Hasil akhir dari akurasi dari keseluruhan pengujian menggunakan *confusion matrix*. Dari total 20 data pengujian dapat diidentifikasi harapan prediksi dan hasil prediksinya.

TABEL I
HASIL PERHITUNGAN

	1	0
1	16	1
0	5	1

Pada tabel I diatas dapat dilihat bahwa hasil pendeteksiannya jika bernilai 1 maka benar dan sebaliknya jika bernilai 0 berarti salah. Definisi salah disini adalah tidak terprediksi ataupun kesalahan prediksi dan nilai 1 dapat diprediksi. Dari tabel I dapat dilihat hasil perhitungannya jika bernilai 1 maka disebut *true positive* yang banyaknya 16. Jika bernilai 0 dan 1 disebut *false positive* dengan jumlah 1. Jika bernilai 0 dan 1 disebut *false negative* dengan jumlah 5 dan nilai 0 disebut *true negative* yang berjumlah 1. Dari hasil tersebut dapat dihitung akurasi, kesalahan klasifikasi, ketelitian, dan kekhususan gambar seperti dibawah ini.

a) *Akurasi (all correct / all) = TP + TN / TP + TN + FP + FN (16+1 / 16+1+5+1) x 100% = 69,5652%*
b) *Kesalahan Klasifikasi (all incorrect / all) = FP + FN / TP + TN + FP + FN (1+5 / 16+1+5+1) x 100% = 26,0869%*
c) *Ketelitian (true positives / predicted positives) = TP / TP + FP (16 / 16+1) x 100% = 94,1176%*
d) *Kekhususan (true negatives / all actual negatives) =TN / TN + FP (1 / 1+1) x 100% = 50%*

Dari hasil perhitungan dapat terlihat bahwa akurasi hanya mencapai 69,5652% dimana hampir mencapai 30% masih terdapat kesalahan pendeteksian. Kesalahan klasifikasi dalam pelabelan nama objek sekitar 26%. Ketelitian dalam pendeteksian yaitu 94%, dan kekhususan dalam *bounding box* adalah 50%.

B. Pengujian secara langsung.

Pengujian ini diambil melalui kamera *webcam* secara *real-time* atau langsung dengan objek mobil di depan Viratama III dan Viratama IV di Akademi Angkatan Udara. Gambar 9 merupakan hasil dari pengujian secara langsung yang diambil dari kamera *webcam*.

Gambar 9. Hasil Pendeteksian Secara Langsung

Gambar 9 menjelaskan bahwa terdapat perbedaan persentase objek diam dan bergerak. Pada objek diam nilainya lebih tinggi karena objek tersebut dapat dengan mudah diidentifikasi oleh algoritma YOLO sedangkan yang bergerak memerlukan perhitungan dan penyesuaian dengan pergerakan yang dilakukan oleh objek sehingga mengurangi persentase akurasi pendeteksian.

V. KESIMPULAN DAN SARAN

Berdasarkan *confusion matrix* dengan uji coba 20 gambar yang diprediksi, menghasilkan akurasi mencapai 69,5652%, kesalahan klasifikasi mencapai 26,0869%, ketelitian dalam pendeteksian mencapai 94,1176%, dan kekhususan *bounding box* mencapai 50%. Intensitas cahaya berpengaruh dalam pendeteksian objek karena dapat mempengaruhi nilai matrix RGB pada citra. Pada deteksi objek menggunakan YOLO bergantung pada kontras warna dimana setiap warna akan diambil nilai matrix terbesar dan akan dibandingkan dengan data pelatihan. Tingkat kesalahan juga diakibatkan data latih yang kurang begitu banyak dan beberapa objek mempengaruhi pendeteksian seperti objek yang memiliki kesamaan dengan objek yang akan dideteksi. Hal lain yang mempengaruhi pada pendeteksian adalah jarak dan intensitas cahaya pada objek gelap sehingga masih terjadi kesalahan klasifikasi. Tingkat ketelitian cukup tinggi karena algoritma YOLO dapat membedakan objek dengan objek lain. Tingkat kekhususan *bounding box* setengah karena menyesuaikan pada objek yang dideteksi.

Untuk penelitian lanjutan, dataset dapat lebih diperbanyak dengan berbagai macam bentuk objek agar nilai akurasi lebih meningkat. Dengan banyaknya dataset yang dilatihkan membuat ketelitian dalam pendeteksian semakin tinggi dan tingkat kesalahan klasifikasi semakin kecil. Selain itu, deteksi objek dapat dikembangkan untuk mendeteksi kecepatan kendaraan bermotor dengan memanfaatkan disparitas kamera.

UCAPAN TERIMA KASIH

Terima kasih atas kepada Akademi Angkatan Udara atas dukungan penelitian sebagai bagian dari penelitian Dosen dan Taruna Akademi Angkatan Udara.

REFERENSI

[1] S. Schneider, G. W. Taylor, and S. C. Kremer, "Deep Learning Object Detection Methods for Ecological Camera Trap Data," *ArXiv180310842 Cs*, Mar. 2018, Accessed: Nov. 09, 2021. [Online]. Available: http://arxiv.org/abs/1803.10842

[2] F. Nobis, M. Geisslinger, M. Weber, J. Betz, and M. Lienkamp,"A Deep Learning-based Radar and Camera Sensor Fusion Architecture for Object Detection," *ArXiv200507431 Cs*, May 2020, Accessed: Nov. 09, 2021. [Online]. Available: http://arxiv.org/abs/2005.07431

[3] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully Convolutional One-Stage Object Detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South), Oct. 2019, pp. 9626–9635. doi: 10.1109/ICCV.2019.00972.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 1, pp. 142–158, Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[5] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra R-CNN: Towards Balanced Learning for Object Detection," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA, Jun. 2019, pp. 821–830. doi: 10.1109/CVPR.2019.00091.

[6] X. Xie, G. Cheng, J. Wang, X. Yao, and J. Han, "Oriented R-CNN for Object Detection," p. 10.

[7] H. Jiang and E. Learned-Miller, "Face Detection with the Faster R-CNN," *ArXiv160603473 Cs*, Jun. 2016, Accessed: Nov. 09, 2021. [Online]. Available: http://arxiv.org/abs/1606.03473

[8] J. Pedoeem and R. Huang, "YOLO-LITE: A Real-Time Object Detection Algorithm Optimized for Non-GPU Computers," *ArXiv181105588 Cs*, Nov. 2018, Accessed: Dec. 30, 2021. [Online]. Available: http://arxiv.org/abs/1811.05588

[9] W. Fang, L. Wang, and P. Ren, "Tinier-YOLO: A Real-Time Object Detection Method for Constrained Environments," *IEEE Access*, vol. 8, pp. 1935–1944, 2020, doi: 10.1109/ACCESS.2019.2961959.

[10] J. Redmon and A. Farhadi, "YOLO9000: Better, Faster, Stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI, Jul. 2017, pp. 6517–6525. doi: 10.1109/CVPR.2017.690.