COMPLETE
Journal of Computer, Electronic, and Telecommunication
ISSN 2723-4371, E-ISSN 2723-5912
doi.org/10.52435/complete.v3i2.263

Analisa Performa Modul ESP32 Sebagai Perangkat untuk Sistem Pengenalan Objek

Alifio Noerifanza[1*]

[1*] Teknik Telekomunikasi, Institut Teknologi Telkom Jakarta, Jakarta; alifionsz@gmail.com

Abstrak: Pengenalan objek atau Object Recognition adalah salah satu perkembangan teknologi AI yg penting dan sangat berguna untuk kehidupan sehari-hari. Dapat digunakan sebagai alat keamanan dan investigasi kejahatan, serta system dalam robot otonom. Contohnya adalah penggunaannya dalam robot penyortir sampah, Kamera pengawas penggunaan masker di China, dan mobil self-driving milik Testla yang menggunakan Object Recognition untuk mengenali lampu merah, obstruksi jalanan serta pejalan kaki. Pengenalan objek membutuhkan sebuah kamera yang dapat menangkap gambar yang akan diproses algoritma. Dikarenakan biaya yang cukup murah serta dapat mengirimkan data menggunakan LAN (Local Area Network), modul kamera ESP32 dipilih sebagai kandidat yang berpotensi dalam kegiatan Object Recognition. ESP32 adalah serangkaian sistem modul kamera yang murah dan berdaya rendah yang terintegrasi dengan mikrokontroler,Wi-Fi (wireless fidelity) dan Bluetooth dual-model dalam satu board[3]. Modul ESP32 yang digunakan dalam kegiatan ini menggunakan kamera OV-2460.

Kata Kunci: ESP32, object recognition, kamera.

1. Pendahuluan

Pengenalan objek atau Object Recognition adalah salah satu perkembangan teknologi AI yg penting dan sangat berguna untuk kehidupan sehari-hari. Dapat digunakan sebagai alat keamanan dan investigasi kejahatan, serta system dalam robot otonom. Contohnya adalah penggunaannya dalam robot penyortir sampah, Kamera pengawas penggunaan masker di China, dan mobil self-driving milik Testla yang menggunakan *Object Recognition* untuk mengenali lampu merah, obstruksi jalanan serta pejalan kaki. Pengenalan objek membutuhkan sebuah kamera yang dapat menangkap gambar yang akan diproses algoritma. Dikarenakan biaya yang cukup murah serta dapat mengirimkan data menggunakan LAN (Local Area Network), modul kamera ESP32 dipilih sebagai kandidat yang berpotensi dalam kegiatan Object Recognition. ESP32 adalah serangkaian sistem modul kamera yang murah dan berdaya rendah yang terintegrasi dengan mikrokontroler,Wi-Fi (wireless fidelity) dan Bluetooth dual-model dalam satu board[1],[5]. Modul ESP32 yang digunakan dalam kegiatan ini menggunakan kamera OV-2460.

Untuk dapat melakukan kegiatan deteksi objek diperlukan sebuah modul kamera yang akan mengambil gambar. Modul ESP32 akan diprogram untuk mengirimkan gambar secara terus menerus sebagai web server LAN. Program Python digunakan untuk mengambil gambar dari web server menggunakan library Urllib dan algoritma Yolov3 untuk memproses gambar dengan library OpenCV. Python dipilih dikarenakan Python adalah bahasa pemrograman yang sering digunakan dalam lingkup data processing dan machine learning [4], [6]. Library OpenCV adalah salah satu library computer vision yang menjadi salah satu bidang dari ilmu artificial intelligence dan machine learning yang membuat sebuah komputer dapat memproses informasi dari gambar, video, atau input visual lainnya[2], [4]. Algoritma YoloV3 digunakan untuk mendeteksi Yolov3 dipilih sebagai

algoritma deteksi objek dikarenakan kecepatan deteksi, kemudahan instalasi dan kelengkapan dataset yang tersedia[3]. Yolo (You Only Look Once) adalah algoritma pengenalan objek yang dapat mendeteksi secara cepat. YoloV3 adalah versi pengembangan dari Yolo yang menggunakan model CNN Darknet-53. Cara kerja darknet-53 adalah menggunakan sistem Convolutional Neural Network yang memotong gambar menjadi kotak-kotak kecil yang memiliki dimensi yang sama. Setiap kotak berfungsi untuk deteksi dan lokalisasi objek[3], [9]–[12]. Saat gambar diproses oleh CNN, filter tersebut akan membaca setiap pixel atau titik yang ada didalam area kerja filter tersebut diikuti dengan Fungsi Aktivasi Non-Linear yang akan menghasilkan peta fitur dari gambar tersebut. Peta fitur ini akan menghasilkan Output dalam 3 parameter[7], [8]. Gambar dalam objek tersebut kemudian akan dibandingkan dengan data dalam dataset yang telah ada untuk menentukan kemungkinan objek tersebut adalah apa. Darknet-53 memiliki 53 Convolution Layer sehingga deteksi menjadi semakin akurat tanpa membuang kekuatan processing secara berlebih[3], [10]-[12]. YoloV3 dipilih karena memiliki besar file yang kecil dibandingkan versi terbaru dan memiliki akurasi yang cukup. YoloV3 yang digunakan sudah membawa pretrained dataset COCO yang berisi Weights dari 80 benda yang dapat dideteksi[13]. Arduino Nano adalah mikrokontroller Open source yang berbasis chip ATmega328P yang didesain oleh Arduino.cc[14], [15]. mikrokontroller ini memiliki profile yang kecil dan dapat digunakan sebagai media untuk melakukan programming ESP32 dengan menyambungkan ESP32 dengan Arduino melalui Pin konektor[14], [15]. Setelah ESP-32 selesai diprogram, kamera OV-2460 yang terpasang di ESP-32 kemudian menangkap dengan resolusi 1600×1200pixel tetapi terbatas sampai 15 frame per detik. Untuk mempercepat pengiriman data, resolusi yang akan digunakan adalah 800x600 pixel.

Tujuan proyek ini adalah untuk menguji potensi kamera ESP32 untuk kepentingan pengambilan gambar secara remote dan mengidentifikasi faktor-faktor yang dapat mempengaruhi performa kecepatan pengiriman gambar dan proses pengenalan objek dalam gambar yang ditangkap oleh ESP32 mengunakan algoritma *Object detection* Yolov3.

2. Metode Penelitian

Gambar 1. Full flowchart kegiatan

Pengujian akan dilakukan dalam 4 tahap besar yaitu tahap studi literatur yang mencakup studi literatur dan panduan yang berkaitan dengan ESP32 dan algoritma pengenalan objek berbasis Python baik dalam bentuk toturial, panduan, jurnal atau *research paper*. Tahap programming dilakukan untuk menyiapkan alat ESP32 untuk bekerja sebagai kamera yang akan melakukan deteksi. ESP32 akan mengambil gambar secara periodik dan menjadikan ESP-32 Web Server yang terkoneksi dengan LAN sebagai wadah dimana Program Python dapat mengambil gambar yang telah direkam ESP-32 secara Wireless. Dikarenakan sebuah media diperlukan untuk menjembatani pemrograman ESP32, Arduino Nano dipilih dikarenakan harganya yang murah, ketersediaan yang banyak dan kemudahan dalam pemrograman. Programming ESP32 dilakukan menggunakan Arduino IDE dan mikrokontroler Arduino sebagai relay. Tahap programming Python dilakukan setelah ESP32 bekerja dengan baik. Dengan menggunakan aplikasi Spyder untuk memudahkan programming. Python digunakan sebagai program yang menjalankan algoritma YoloV3 dan mengambil gambar dari web server ESP32 untuk diproses. Tahap pengujian mencakup pengambilan dan pemprosesan data untuk mendapatkan hasil performa ESP32 sebagai alat Object Detection serta analisa terkait hal-hal yang mempengaruhi performanya. Pengambilan data dilakukan secara indoor dalam 2 kondisi pencahayaan, siang hari dan malam hari. Perhitungan frame pertama dimulai dari perubahan frame pertama setelah stopwatch dinyalakan. Pengujian dilakukan selama 1 menit dari perubahan frame pertama setelah stopwatch dinyalakan. Data terakhir adalah rame yang terakhir keluar dalam durasi 1 menit tersebut.

Pengujian dilakukan di kamar penulis secara indoor. Dalam tahap ini kecepatan proses Object Detection dan kehandalannya diobservasi. Akan disajikan sample berupa barang sehari-hari kepada kamera. Kamera akan merekam dengan cara menangkap gambar secara kontinyu dan menguploadnya ke web server ESP32 kemudian komputer akan memprosesnya menggunakan program Python yang menggunakan Library GetUrl untuk mengambil gambar dari server dan OpenCV untuk memproses dan menyajikan hasil deteksi objek. Pengambilan data dilakukan dalam 5 skenario yang memiliki kondisi dan jumlah sampel yang berbeda-beda dan dapat dilihat pada table berikut:

Tabel 1. Skenario pengujian

Skenario	Lokasi	Pencahayaan	Pergerakan	Resolusi	Jumlah sampel
1	Indoor	Malam	ya	800 x 600	3
2	Indoor	Siang	ya	800 x 600	3
3	Indoor	Siang	tidak	800 x 600	3
4	Indoor	Siang	tidak	1280 x 1024	10
5	Indoor	Siang	ya	1280 x 1024	10

3.2. Metode Analisa

Analisa dilakukan dengan metode kombinasi dan dilakukan pada tahap empat setelah data telah didapatkan. Parameter yang akan diperhatikan dalam proses Analisa untuk menilai performa ESP32 adalah:

1. Kejelasan Gambar Sampel

Sampel yang terlihat didefinisikan sebagai sampel yang cukup jelas untuk dilihat secara kasat mata. Berikut contohnya berupa tas laptop:

Gambar 2. Gambar tas yang ditangkap, hasil jelas dan terlihat

Dapat dilihat bahwa gambar tas laptop yang ditangkap dengan resolusi 1280px dapat dikenali dengan mudah. Sedangkan sampel yang tidak jelas adalah yang *blurry* atau sulit dikenali entah dikarenakan background, pencahayaan, atau ukuran benda. Berikut contohnya berupa Garpu:

Gambar 3. Gambar Garpu yang ditangkap.

Dapat dilihat walaupun ditangkap dengan resolusi yang sama, karena ukuran kecil dan background tidak contrast hasil terlihat blur dan benda sulit dikenali.

2. Ukuran benda Sample

Objek dikatakan kecil jika memiliki Lebar kurang dari 6cm. Objek dinyatakan sedang jika memiliki lebar 6cm sampai 22cm. Objek dinyatakan besar jika memiliki lebar diatas 22cm.

Gambar 4. Contoh Ukuran Sampel Kecil dan Sedang.

Ukuran dari sampel yang akan dideteksi akan berpengaruh pada hasil deteksi karena algoritma Object Detection membutuhkan gambar yang jelas untuk deteksi yang akurat. Untuk skenario 1 sampai 3 semua sampel berukuran besar, sedangkan untuk skenario 4 dan 5 ukuran objek terbagi tiga jenis yaitu Besar, Sedang, dan Kecil. Berikut table objeknya:

Table 2. Ukuran objek sample

Ukuran	Nama Sampel
Kecil	Garpu
Sedang	Botol, gelas, hp, mouse, remote, buku
Besar	Kursi, tas, jam

3. Kondisi Pencahayaan

Untuk memudahkan Analisa, dibuatlah 2 kategori untuk menentukan performa ESP32 dalam redup dan terang. Jika sampel diambil dari dari jam 6.30 sampai jam 17.30 maka termasuk kategori sampel siang dan jika sampel diambil dari jam 17.30 sampai 6.30 maka termasuk kategori malam.

4. Kondisi Pergerakan Kamera

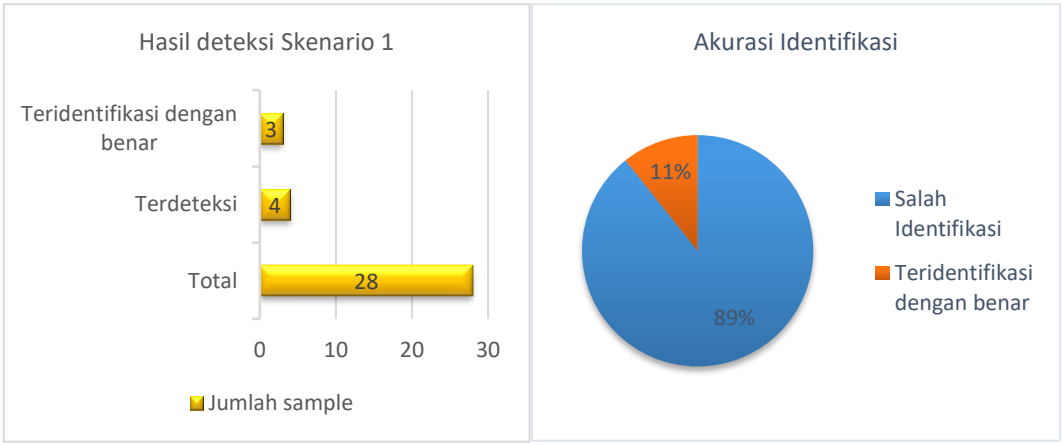
Kondisi pergerakan kamera dalam gambar mungkin tidak terlihat jelas dalam hasil pengujian dikarenakan system ESP32 mengambil gambar secara periodik dan menguploadnya ke web server tetapi penulis berharap untuk mendapatkan hasil tentang hubungan pergerakan kamera dan akurasi serta kecepatan deteksi objek.

5. Setting Resolusi ESP32

Setting Resolusi terbagi dua dikarenakan terjadi perubahan resolusi menambah hasil pengujian. Untuk 2 test pertama yang terdapat hanya 3 sampel dilakukan dengan setting hiRes 800x600px. Untuk test selanjutnya digunakan setting hiRes 1280x1024px untuk mengecek performa deteksi.

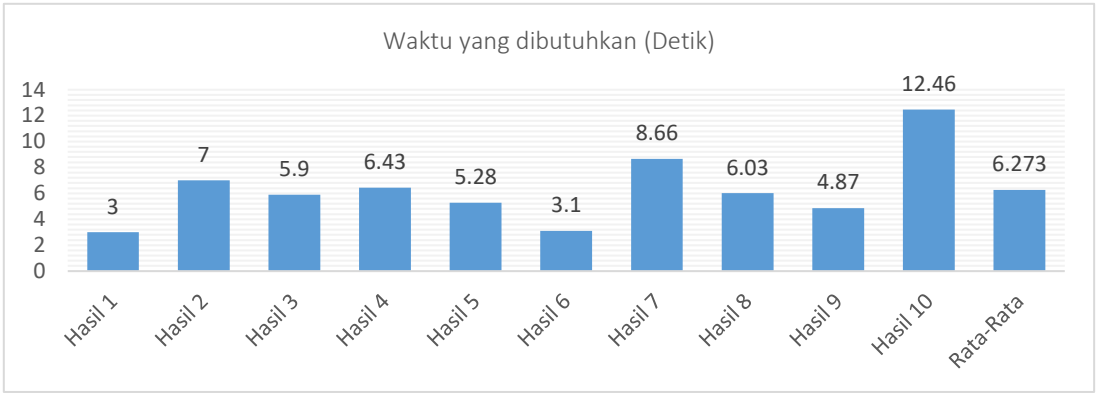
3. Hasil Pengujian

Skenario 1 dilakukan dalam kondisi malam dengan pergerakan. Jumlah sampel ada 3 dan dilakukan dengan resolusi 800px. Semua sampel berukuran besar. Berikut adalah grafik hasil deteksi:



Gambar 5. Grafik hasil deteksi dan presentasi akurasi deteksi skenario dua.

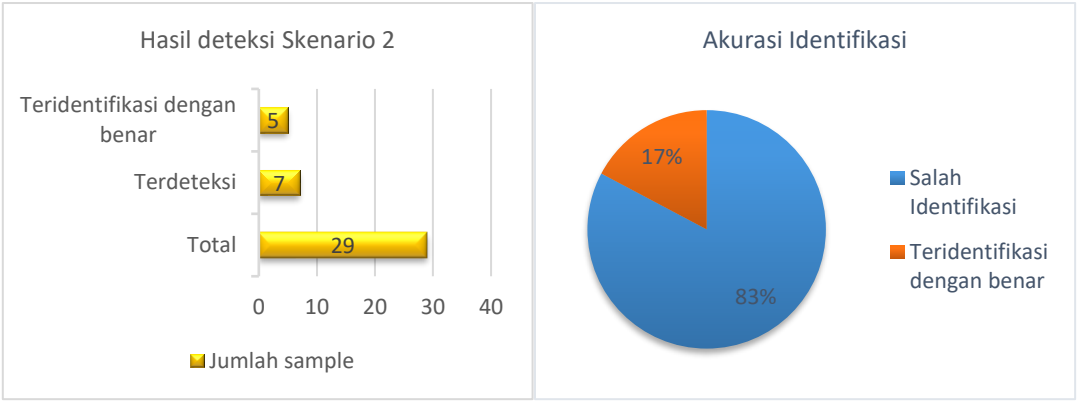
Skenario satu mempunyai total 28 jumlah sample yang muncul dalam 1 menit, dimana terdeteksi 4 objek sampel dan hanya 3 yang teridentifikasi dengan benar. Dari semua sampel yang terdeteksi hanya 11% yang teridentifikasi dengan benar dan 89% yang salah identifikasi.



Gambar 6. Grafik waktu yang dibutuhkan untuk deteksi dalam skenario dua.

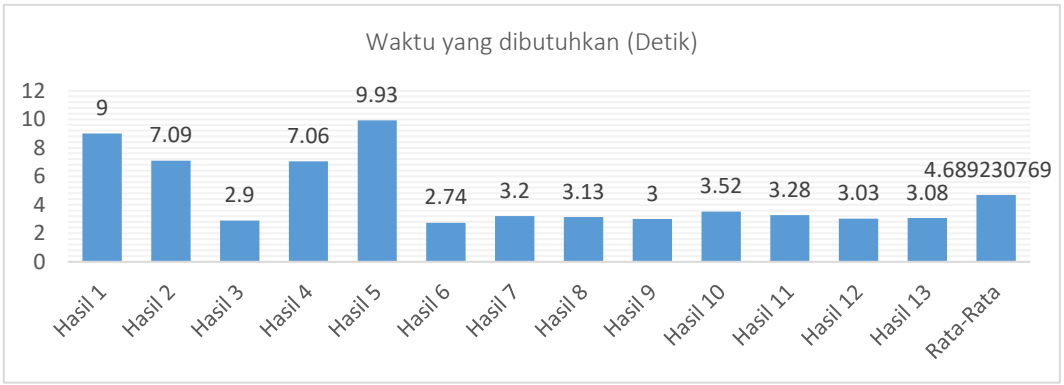
Dalam skenario satu terdapat 10 frame hasil deteksi dimana rata-rata waktu yang dibutuhkan untuk deteksi dan identifikasi adalah 6,273 detik. Dapat dilihat bahwa kecepatan deteksi lambat dan akurasinya sangat rendah.

Skenario 2 dilakukan siang hari dengan pergerakan dengan resolusi 800px, sampel ada 3 dan semua sampel berukuran besar. Berikut adalah grafik hasil deteksi skenario dua:



Gambar 7. Grafik hasil deteksi dan presentase akurasi identifikasi skenario dua.

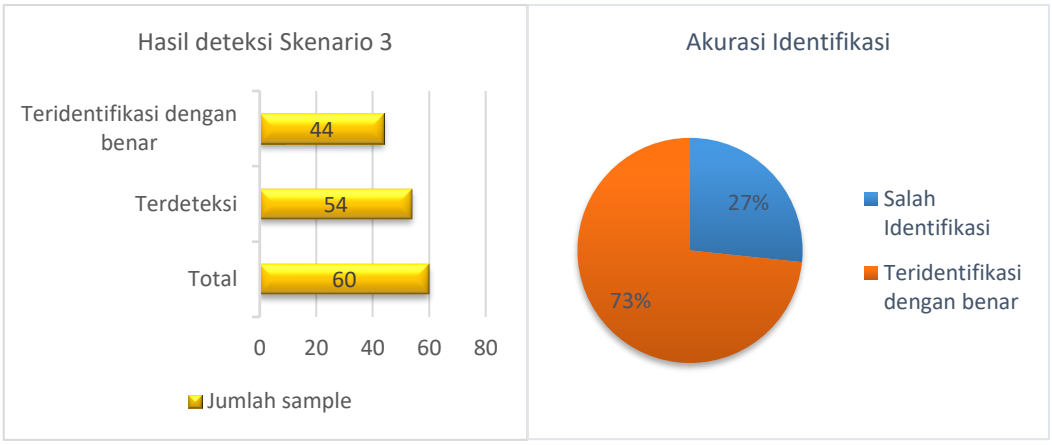
Skenario dua mempunyai total 29 jumlah sample yang muncul dalam 1 menit, dimana terdeteksi 7 objek sampel dan hanya 5 yang teridentifikasi dengan benar. Dari semua sampel yang terdeteksi hanya 17% yang teridentifikasi dengan benar dan 83% yang salah identifikasi. Terjadi peningkatan jumlah deteksi dibandingkan skenario sebelumnya.



Gambar 8. Grafik waktu yang dibutuhkan untuk deteksi dalam skenario dua.

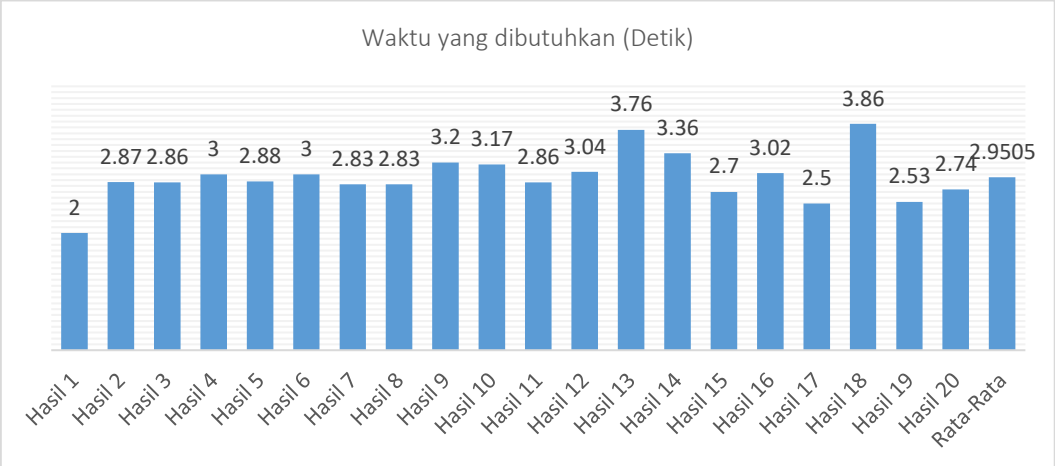
Dalam skenario dua terdapat 13 frame hasil deteksi dimana rata-rata waktu yang dibutuhkan untuk deteksi dan identifikasi adalah 4,689 detik. Dapat dilihat bahwa jika dibandingkan dengan skenario sebelumnya yang dilakukan pada malam hari, kecepatan dan akurasi deteksi meningkat.

Skenario 3 Dilakukan Siang hari dengan menyalakan Lampu kamar sebagai bantu tambahan pencahayaan dan kamera tidak bergerak. Resolusi yang dipakai adalah 800px dan semua sampel berukuran besar. Berikut adalah grafik hasil deteksi skenario tiga:



Gambar 9. Grafik hasil identifikasi skenario tiga.

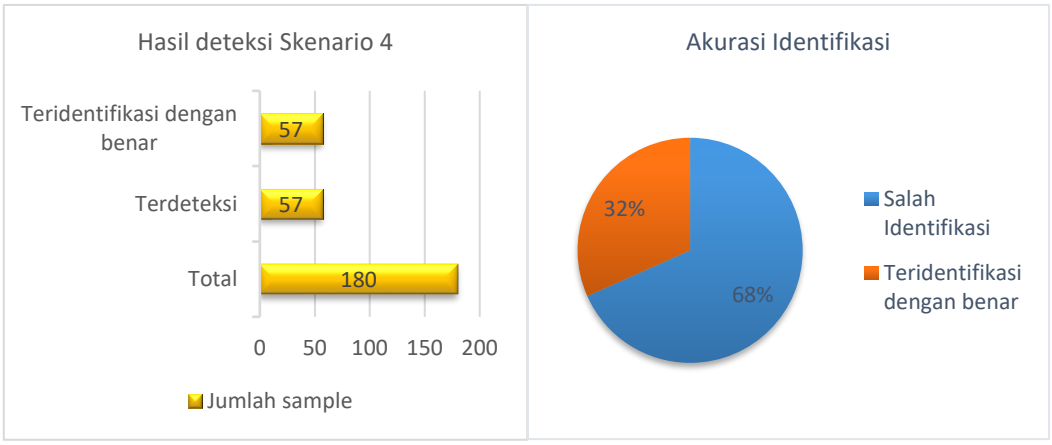
Skenario tiga mempunyai total 60 jumlah sample yang muncul dalam 1 menit, dimana terdeteksi 54 objek sampel dan hanya 44 yang teridentifikasi dengan benar. Dari semua sampel yang terdeteksi 73% yang teridentifikasi dengan benar dan hanya 27% yang salah identifikasi.



Gambar 10. Grafik waktu yang dibutuhkan untuk deteksi dalam skenario tiga.

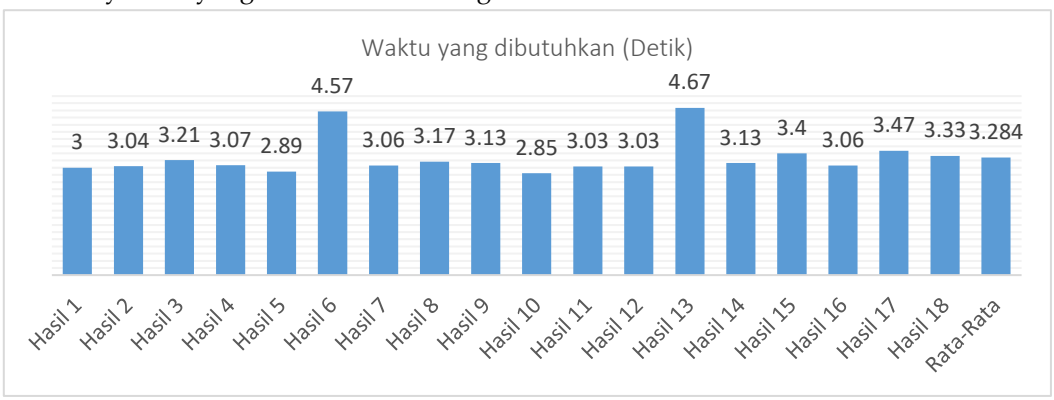
Dalam skenario tiga terdapat 20 frame hasil deteksi dimana rata-rata waktu yang dibutuhkan untuk deteksi dan identifikasi adalah 2,95 detik. Jika dibandingkan dengan skenario-skenario sebelumnya terlihat bahwa pencahayaan dan pergerakan kamera berpengaruh pada kecepatan identifikasi objek.

Skenario 4 adalah deteksi banyak objek dengan resolusi 1280px dan dilakukan di siang hari dengan sampel 10 objek dengan tas dalam kondisi tertidur. Kamera tidak bergerak dengan ukuran sampel campuran. Berikut adalah hasilnya:



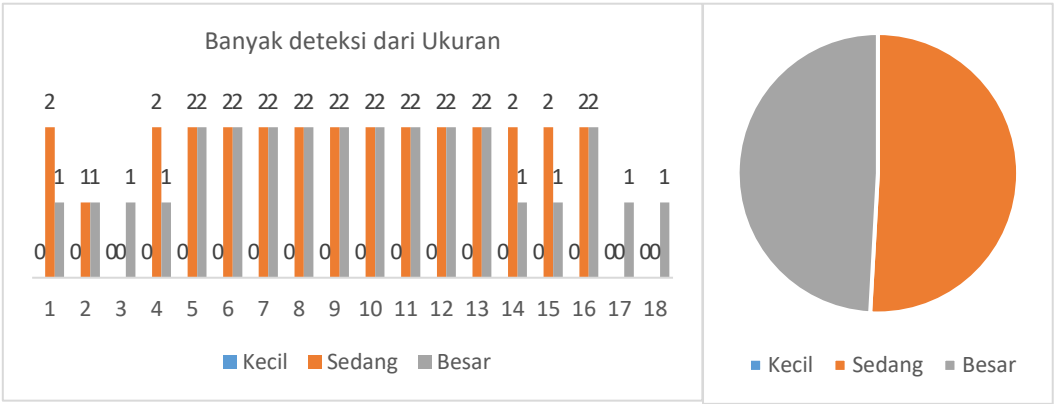
Gambar 11. Grafik hasil deteksi dan presentase akurasi identifikasi skenario empat.

Skenario empat mempunyai total 180 jumlah sample yang muncul dalam 1 menit, dimana terdeteksi 57 objek sampel dan semua yang teridentifikasi dengan benar. Dari semua sampel yang terdeteksi hanya 32% yang teridentifikasi dengan benar dan 68% salah identifikasi.



Gambar 12. Grafik waktu yang dibutuhkan untuk deteksi dalam skenario empat

Dalam skenario empat terdapat 18 frame hasil deteksi dimana rata-rata waktu yang dibutuhkan untuk deteksi dan identifikasi adalah 3,28 detik.

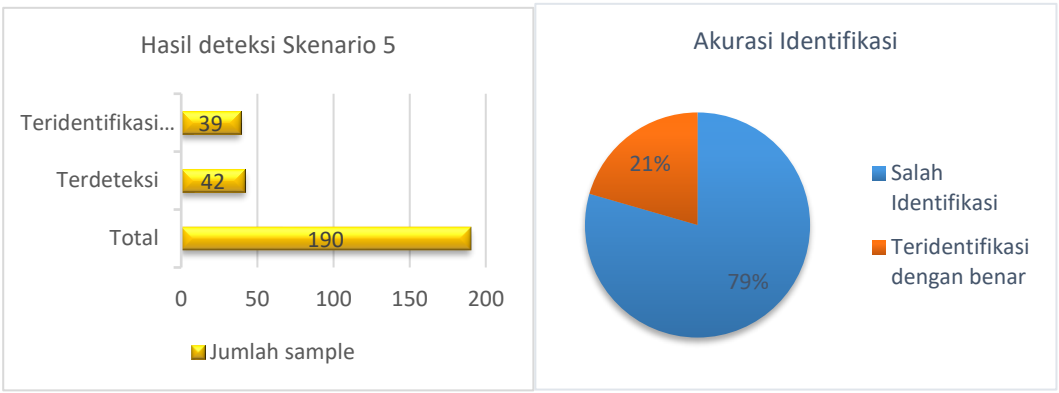


Gambar 13. Grafik deteksi dan presentase deteksi skenario empat berdasarkan ukuran.

Dari grafik diatas dapat dilihat bahwa mayoritas objek yang terdeteksi adalah objek berukuran sedang. Objek berukuran sedang dan objek berukuran besar terdeteksi dengan presentase hampir

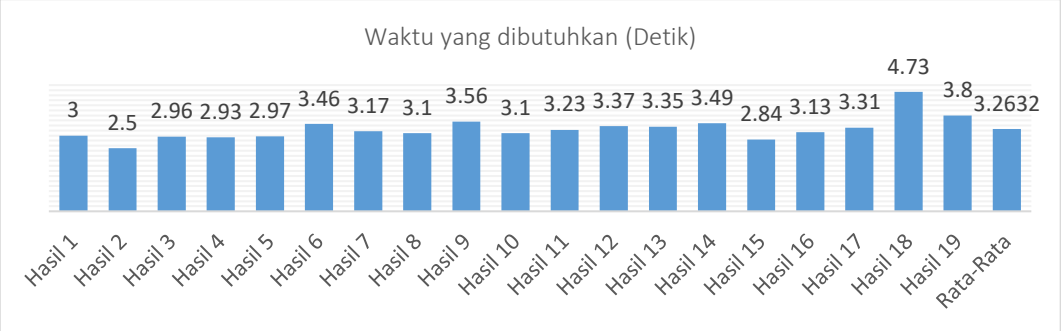
setara sedangkan objek berukuran kecil tidak terdeteksi sama sekali. Dalam skenario 4 tas dalam kondisi tertidur dan tidak terdeteksi oleh algoritma.

Skenario 5 adalah deteksi banyak objek dengan resolusi 1280px dan dilakukan di siang hari dengan sampel 10 objek, tas dalam kondisi berdiri dengan kamera bergerak dan ukuran sampel campuran. Berikut adalah hasil deteksi skenario 5:



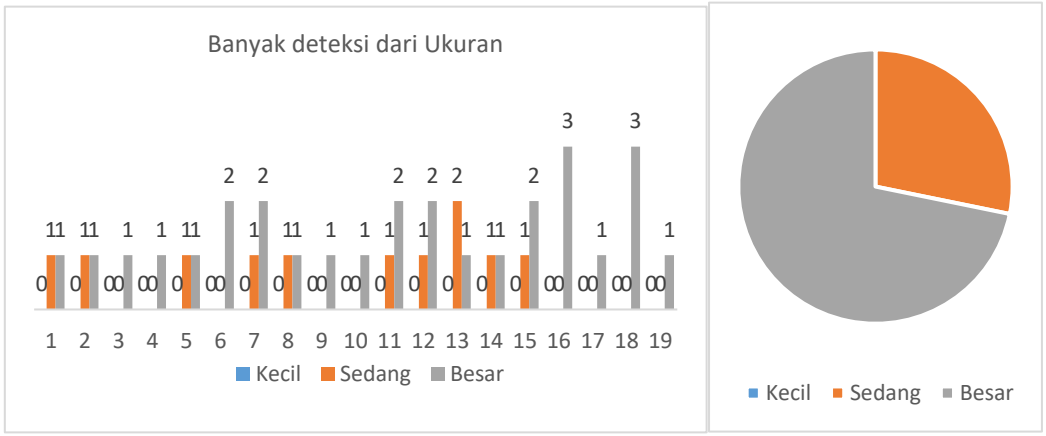
Gambar 14. Grafik hasil deteksi dan presentase akurasi identifikasi skenario lima.

Skenario lima mempunyai total 190 jumlah sample yang muncul dalam 1 menit, dimana terdeteksi 42 objek sampel dan hanya 39 yang teridentifikasi dengan benar. Dari semua sampel yang terdeteksi hanya 21% yang teridentifikasi dengan benar dan 79% salah identifikasi.



Gambar 15. Grafik waktu yang dibutuhkan untuk deteksi dalam skenario lima.

Dalam skenario lima terdapat 19 frame hasil deteksi dimana rata-rata waktu yang dibutuhkan untuk deteksi dan identifikasi adalah 3,26 detik.



Gambar 16. Grafik hasil deteksi dan presentase deteksi skenario lima berdasarkan ukuran.

Dari grafik diatas dapat dilihat bahwa mayoritas objek yang terdeteksi adalah objek berukuran besar. Objek berukuran sedang menjadi minoritas sedangkan objek berukuran kecil tidak terdeteksi sama sekali. Dalam skenario 5 tas dalam kondisi berdiri dan terdeteksi oleh algoritma.

4. Pembahasan

Berdasarkan hasil proyek akhir dapat disimpulkan bahwa parameter dibawah ini dapat mempengaruhi hasil deteksi ESP32:

1. Pencahayaan mempengaruhi kecepatan dan akurasi deteksi dikarenakan gambar menjadi terlalu gelap untuk algoritma deteksi objek mengenali objek. Walaupun secara kasat mata objek dapat terlihat tetapi tidak terdeteksi atau salah identifikasi oleh algoritma tersebut contohnya terdapat dalam gambar A1.

2. Pergerakan kamera mempengaruhi akurasi deteksi dikarenakan gambar yang diambil bisa blur sehingga menyebabkan gambar lebih sulit untuk diproses. Gambar juga dapat menjadi sangat blurry sehingga tidak jelas dan tidak dapat dikenali contohnya terdapat dalam gambar A2.

3. Resolusi gambar resolusi gambar mempengarui kejelasan objek yang akan dideteksi. Objek lebih sulit terdeteksi dan diidentifikasi dalam resolusi 800px dibandingkan dengan resolusi 1280px. resolusi kecil juga dapat menyebabkan objek berukuran lebih kecil tidak dapat terdeteksi dikarenakan tidak jelas objeknya.

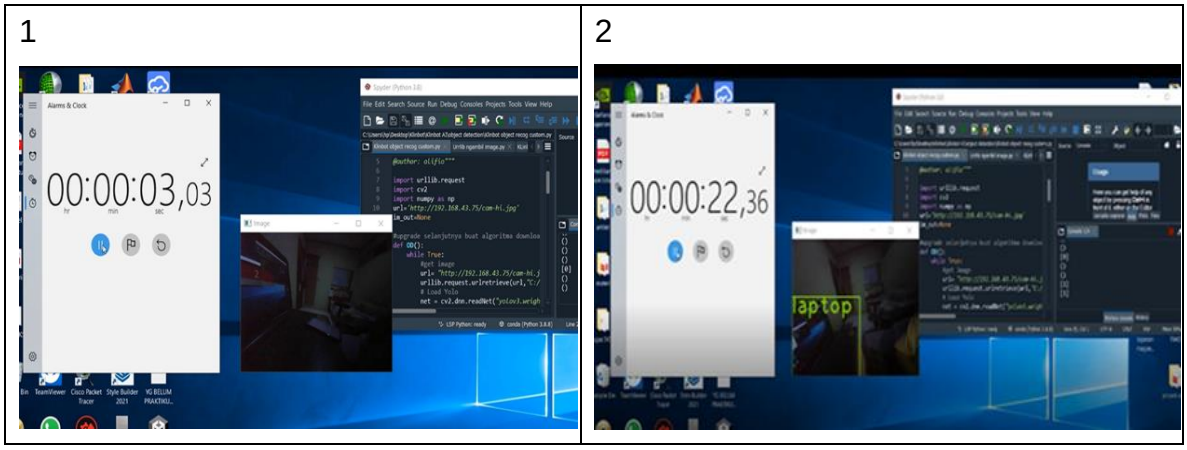
4. Ukuran objek mempengaruhi kemungkinan objek tersebut terdeteksi dan teridentifikasi dengan akurat dikarenakan resolusi gambar yang terbatas. Objek berukuran besar seperti kursi, Laptop dan tas lebih mudah dideteksi daripada objek berukuran lebih kecil seperti garpu, remote.

5. Kejelasan objek mempengaruhi deteksi dikarenakan ada beberapa objek yang dapat menyerupai objek lain. Tas yang tertidur dapat diartikan sebagai objek lain yang tidak ada dalam dataset sehingga tidak terdeteksi dan layar laptop yang keyboardnya tidak terlihat dapat salah identifikasi menjadi layar TV contohnya tertera dalam gambar A3.

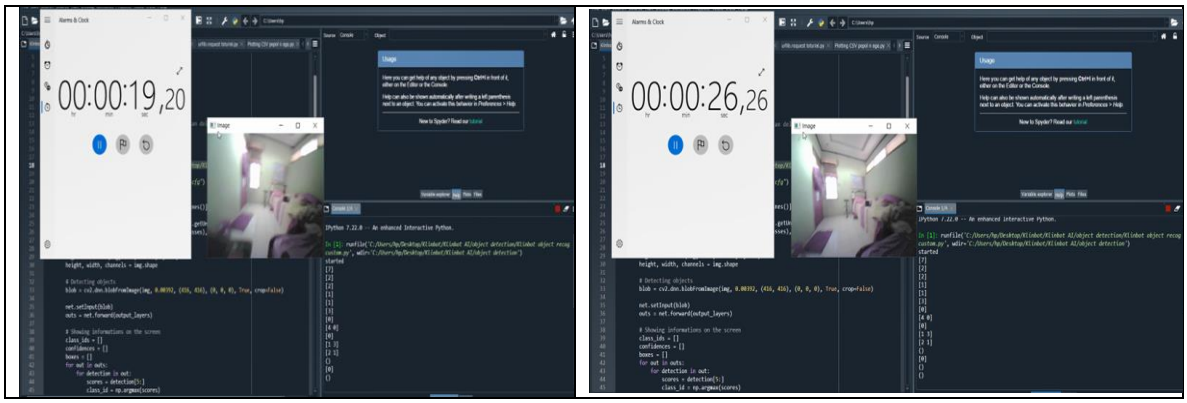
5. Kesimpulan

ESP32 dapat digunakan sebagai alat deteksi objek hanya jika kamera diam, dalam pencahayaan yang cukup dan hanya bertugas mendeteksi objek berukuran besar. Kecepatan deteksi adalah 3 sampai 4 detik per frame. Untuk menunjang performa deteksi yang baik disarankan menggunakan resolusi tinggi. Ketidaktersediaan kamera ESP32 yang mempunyai resolusi tinggi membuat gambar hasil tangkapan mempunyai resolusi terbatas sehingga mengganggu akurasi. Diharapkan kedepannya pengujian deteksi objek dilakukan dengan kamera lain yang mempunyai sambungan USB atau kabel ke computer untuk mengurangi latency serta menggunakan kamera yang lebih baik. Algoritma deteksi objek diharapkan menggunakan dataset yang dibuat lebih spesifik untuk meningkatkan kemungkinan deteksi objek.

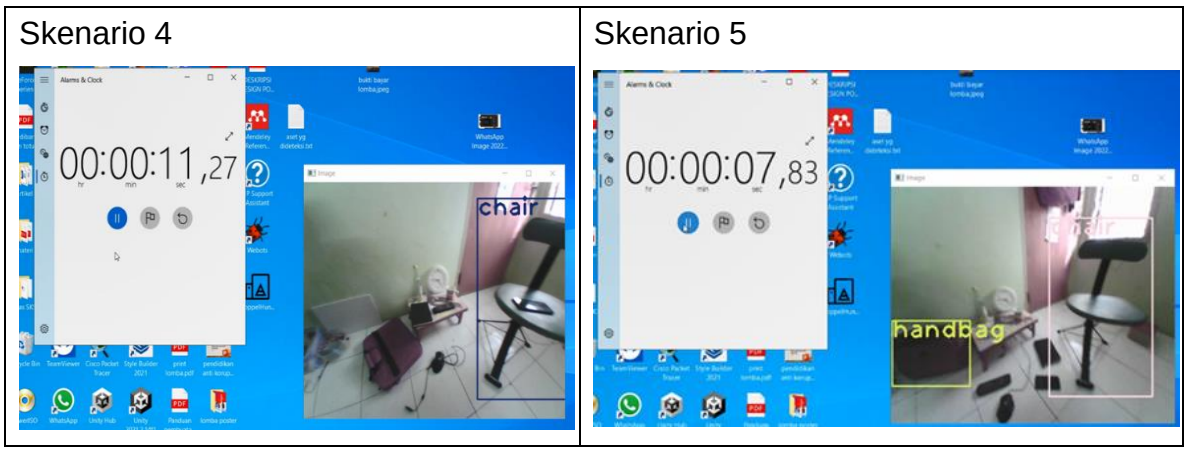
Appendix



Appendix A1. Contoh hasil skenario 1 dimana objek dapat dilihat dengan mata tetapi tidak terdeteksi.



Appendix A2. Contoh hasil skenario 2, terlihat hasil blur karena terdapat pergerakan



Appendix A. Hasil skenario 4 dibandingkan dengan hasil scenario 5

Referensi

1. H. Fitri dan dan Ivan Finiel Hotmartua Bagariang, "PEMANFAATAN ESP32-CAM UNTUK MENGUKUR KETINGGIAN AIR MENGGUNAKAN METODE IMAGE PROCESSING," *Seminar Nasional Terapan Riset Inovatif (SENTRINOV) Ke-6 ISAS Publishing Series: Engineering and Science*, vol. 6, no. 1, 2020.
2. V. Wiley dan T. Lucas, "Computer Vision and Image Processing: A Paper Review," *International Journalof Artificial Intelligence Research*, vol. 2, no. 1, hlm. 22, Jun 2018, doi: 10.29099/ijair.v2i1.42.

3. Z. Z. Jin dan Y. F. Zheng, "Research on application of improved YOLO V3 algorithm in road target detection," dalam *Journal of Physics: Conference Series*, Okt 2020, vol. 1654, no. 1. doi: 10.1088/1742-6596/1654/1/012060.
4. S. N. Srivatsa, "Object Detection using Deep Learning with OpenCV and Python," *International Research Journal of Engineering and Technology*, 2021, [Daring]. Available: www.irjet.net
5. R. B. Salikhov, V. K. Abdrakhmanov, dan I. N. Safargalin, "Internet of things (IoT) security alarms on ESP32-CAM," dalam *Journal of Physics: Conference Series*, Nov 2021, vol. 2096, no. 1. doi: 10.1088/1742-6596/2096/1/012109.
6. A. Sharma, F. Khan, D. Sharma, S. Gupta, dan F. Y. Student, "Python: The Programming Language of Future," 2020.
7. T. Bezdan dan N. Bačanin Džakula, "Convolutional Neural Network Layers and Architectures," Mei 2019,hlm. 445–451. doi: 10.15308/sinteza-2019-445-451.
8. J. Li, A. Hassani, S. Walton, dan H. Shi, "ConvMLP: Hierarchical Convolutional MLPs for Vision," Sep2021, [Daring]. Available: http://arxiv.org/abs/2109.04454
9. J. Redmon dan A. Farhadi, "YOLOv3: An Incremental Improvement." [Daring]. Available: https://pjreddie.com/yolo/.
10. Q. Aini, N. Lutfiani, H. Kusumah, dan M. S. Zahran, "DETEKSI DAN PENGENALAN OBJEK DENGAN MODEL MACHINE LEARNING: MODEL YOLO," 2021.
11. A. Vidyavani, K. Dheeraj, M. Rama Mohan Reddy, dan K. N. Kumar, "Object detection method based onYOLOv3 using deep learning networks," *International Journal of Innovative Technology and Exploring Engineering*, vol. 9, no. 1, hlm. 1414–1417, Nov 2019, doi: 10.35940/ijitee.A4121.119119.
12. S. Wang, "Research towards Yolo-Series Algorithms: Comparison and Analysis of Object Detection Models for Real-Time UAV Applications," dalam *Journal of Physics: Conference Series*, Jun 2021, vol. 1948, no. 1. doi: 10.1088/1742-6596/1948/1/012021.