E-ISSN : 2988-1986
https://ejournal.warunayama.org/kohesi

Kohesi: Jurnal Multidisiplin Saintek
Volume 01, No. 06 2023, pp. 81-90

PERBANDINGAN ALGORITMA PENDETEKSI PENYUSUP PADA SISTEM KEAMANAN RUMAH BERBASIS INTERNET OF THINGS

Didit Karyadi[1], Hustinawaty[2]
Program Studi Sistem Informasi Bisnis, Fakultas Ilmu Komputer, Universitas Gunadarma
Email: diditkaryadi@gmail.com[1], hustina@staff.gunadarma.ac.id[2]

Abstrak

Studi empiris menyatakan lingkungan menjadi faktor utama yang berpengaruh pada pola-pola kejahatan sehingga closed circuit television (CCTV) menjadi pilihan untuk mengurangi risiko kejahatan. Namun, CCTV kurang efektif karena membutuhkan high bandwidth & storage serta tidak dapat memberikan notifikasi. Oleh karena itu, muncul teknologi bernama internet of things (IoT) sehingga CCTV atau webcam bisa bekerjasama dengan sensor untuk mendeteksi kehadiran penyusup dan memberikan notifikasi. Penelitian ini mengusulkan sistem yang mendeteksi adanya penyusup dan mengirimkan notifikasi kepada pemilik rumah tanpa terikat waktu dan tempat. Sistem ini biasanya disebut sebagai smart home security system. Penelitian ini bertujuan untuk mengetahui perbandingan algoritma pendeteksi penyusup pada sistem kemanan rumah berbasis IoT. Metode penelitian ini menggunakan *internet of things* (IoT) pada *smart home* atau *home security* dengan membandingkan akurasi dan waktu proses algoritma HoG+SVM dan Yolo V3. Hasil implementasi sistem mendapatkan hasil bahwa pendeteksi penyusup paling akurat yaitu algoritma Yolo V3 dengan akurasi 99% dan waktu proses 14,852 detik. Waktu proses ini bisa dipercepat dengan menggunakan Graphics Processing Unit (GPU) yang lebih tinggi spesifikasinya.

Kata Kunci—Yolo, Keamanan Rumah, Deteksi Penyusup.

Abstract

Empirical studies suggest that environment is the main factor influencing crime patterns so that closed circuit television (CCTV) becomes the choice to reduce the risk of crime. However, CCTV is less effective because it requires high bandwidth & storage and cannot provide notifications. Therefore, a technology called the Internet of Things (IoT) appears so that CCTV or webcam can work together with sensors to detect the presence of intruders and provide notifications. This study proposes a system that detects intruders and sends notifications to homeowners regardless of time and place. This system is usually referred to as a smart home security system. The author compares the accuracy and processing time of the HoG + SVM algorithm and Yolo V3. The results of the implementation of the system get the result that the most accurate intruder detector is the Yolo V3 algorithm with an accuracy of 99% and a processing time of 14.852 seconds. This processing time can be accelerated by using a Graphics Processing Unit (GPU) with higher specifications.

Keywords—Yolo, Home Security, Intruder Detection.

Prefix DOI : 10.3785/kjst.v1i6.347

Pendahuluan

Kamera pengawas atau yang sering disebut CCTV (*Closed Circuit Television*) sudah sering ditemui di gedung-gedung perkantoran, bank, pusat perbelanjaan bahkan juga digunakan oleh toko-toko skala kecil hingga menengah maupun di rumah-rumah kalangan menengah atas. Hal tersebut secara tidak langsung menunjukkan semakin meningkatnya kekhawatiran dan kewaspadaan masyarakat terhadap tindak pencurian. Penggunaan kamera pengawas selama ini sebagian besar digunakan sebagai bukti-bukti kejahatan ataupun sebagai referensi bagi penegak hukum untuk mengenali pelaku sehingga dapat menggali informasi lebih lanjut untuk menangkap pelaku namun hanya dengan CCTV saja akan kurang efektif dalam mencegah kejahatan karena hanya memantau secara terus menerus, akan tetapi tidak memberikan peringatan atau reaksi awal ketika menangkap suatu objek yang mencurigakan. Disamping itu, CCTV membutuhkan *high bandwidth* dan *storage* karena akan mentransmisikan data dalam jumlah yang besar (M. A. Hossain & Song, 2016)

Saat ini beberapa studi dari akademisi, industri, dan pemerintah telah mencoba untuk menghubungkan semua hal di dunia ke dalam internet untuk menyediakan sistem yang terintegrasi dengan mulus demi meningkatkan kinerja dalam transmisi informasi, yang disebut sebagai *Internet of Things* (IoT). Pengembangan dan pertumbuhan dari IoT sangat cepat dan telah terdapat berbagai macam jenis aplikasi – aplikasi IoT yang sangat membantu dan berkontribusi untuk kehidupan manusia sehari – hari menjadi lebih baik. IoT telah diterapkan secara luas pada aplikasi kehidupan sosial seperti *smart grid, intelligent transportation, smart security*, dan *smart home* (Jing et al., 2014). Teknologi – teknologi baru terus bermunculan pada IoT yang dapat meningkatkan jumlah sensor dan kompleksitas dari *smart homes*. Salah satu pemanfaatan perangkat dalam IoT adalah dengan *passive infrared* (PIR) *sensors* yang dapat melakukan mendeteksi perpindahan atau pergerakan suatu objek dengan informasi yang sederhana (Ahvar et al., 2016). Selain itu, pemanfaatan kamera untuk menangkap gambar atau objek dapat dikombinasikan dengan PIR *motion* untuk mendeteksi orang yang tidak berwenang di dalam rumah sehingga dapat memberikan notifikasi keamanan kepada pemilik rumah. Hal tersebut dapat menjadi kombinasi yang baik dalam pengembangan *home security* yang dapat meningkatkan keamanan dan kenyamanan yang lebih hemat energi (Hossain Jewel et al., 2017).

Beberapa penelitian mengenai *monitoring system* telah dilakukan. Salah satunya yaitu metode *human detection* dengan *video surveillance system.* Sistem ini dapat mendeteksi manusia dengan menggunakan fitur ekstraksi *histogram of gradient* (HoG) dan klasifikasi dengan menggunakan algoritma *support vector machine* (SVM). Metode ini hanya mendapatkan akurasi kurang lebih sekitar 89% (Seemanthini & Manjunath, 2018). Selain itu, metode HoG dan SVM telah diimplementasikan pada sebuah *home security system* yang memiliki kemampuan pendeteksian manusia dengan menggunakan Raspberry Pi 3, *webcam*, PIR *sensor*, dan *buzzer*. Hasil dari sistem dan metode tersebut dapat mendeteksi kehadiran penyusup dengan rata – rata akurasi sebesar 90% dan rata – rata waktu proses sekitar 2 detik (Surantha & Wicaksono, 2019).

Selain itu terdapat metode atau algoritma bernama *You Only Look Once* (YOLO) yang biasanya dapat digunakan untuk mendeteksi pejalan kaki dan sistem pelacakan. Algoritma YOLO sudah ada hingga YOLO V3 yang lebih akurat dan cepat dibandingkan versi sebelumnya. Pada beberapa percobaan, YOLO V3 dapat mendeteksi, melacak pejalan kaki, manusia dalam permainan bola basket dan / atau berbagai objek lainnya dengan sukses pada setiap video *frames*. YOLO V3 lebih akurat dan cepat dalam dibandingkan dengan versi sebelumnya. YOLO V3 dapat mendeteksi berbagai jenis objek seperti manusia, mobil, dan

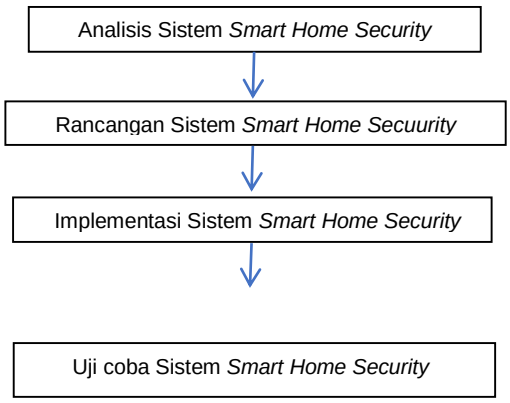
Prefix DOI : 10.3785/kjst.v1i6.347

objek – objek lainnya yang tingkat akurasinya lebih besar dari 90% (Lee & Seo, 2019; Qu et al., 2019; Yoon et al., 2019)

Berdasarkan masalah di atas, penulis ingin mengembangkan sistem yang dapat memberikan pemberitahuan kepada pemilik rumah kapanpun dan dimanapun dengan lebih mudah dan nyaman melalui *smartphone* ketika mendeteksi adanya penyusup. Selain itu, penulis ingin melihat keterkaitan antara akurasi dengan waktu yang dibutuhkan untuk melakukan proses pendeteksian manusia. Oleh karena itu, tujuan penelitian ini untuk membandingkan akurasi dan waktu proses beberapa algoritma yang dapat digunakan untuk mengembangkan sistem yang efektif berdasarkan akurasi atau waktu prosesnya.

Metode

2.1 Tahap Penelitian



Gambar 1. Diagram Tahap Penelitian

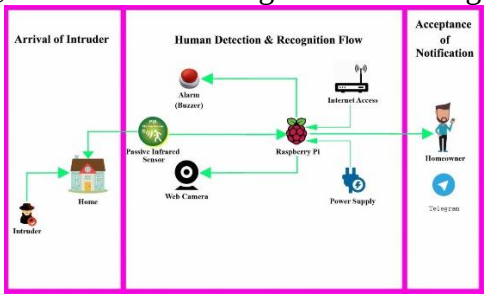
Gambar 1 adalah langkah-langkah yang dilalui dalam penelitian ini, langkah pertama penulis melakukan analisis sistem *smart home*, setelah itu penulis membuat rancangan sistem *smart home*, kemudian setelah rancangan sistem terbentuk penulis melakukan implementasi dari rancangan tersebut, langkah terakhir sistem *smart home* melakukan uji coba sistem *smart home* untuk mengetahui apakah sistem *smart home* dapat bekerja sesuai dengan harapan.

2.2 Perancangan Sistem Smart Home Security

Perancangan sistem *Smart Home Security* terdiri dari Arsitektur Sistem, Alur Kerja Sistem, Arsitektur Perangkat keras, Pengambilan Data dan Desain Ruangan

2.2.1 Arsitektur Sistem

Dalam penelitian ini, arsitektur sistem digambarkan sebagai berikut:



Gambar 2. Arsitektur Sistem

Prefix DOI : 10.3785/kjst.v1i6.347

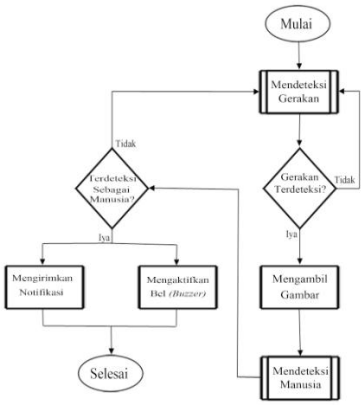
Gambar 2. adalah perancangan arsitektur sistem dibagi ke dalam tiga wilayah proses yaitu ketika kedatangan dari arrival of intruder, human detection & recognition flow, dan acceptance of notification. Penulis mengasumsikan penyusup akan masuk dari titik dimana penyusup memungkinkan untuk masuk kerumah seperti pintu, jendela, loteng.

Kemudian, proses selanjutnya terjadi pada human detection & recognition flow. Passive Infrared sensor akan selalu siap untuk memantau atau mengontrol status akses masuk rumah apakah terdeteksi gerak atau tidak setelah sistem diaktifkan oleh pemilik rumah. Passive Infrared sensor akan terus menerus melakukan pemantauan. Namun, ketika status sensor terdeteksi gerakan, maka sistem akan menggunakan web camera untuk menangkap gambar pada wilayah pendeteksian.

Sistem selanjutnya akan memproses gambar yang telah ditangkap oleh web camera menggunakan metode YoloV3 untuk human detection. Jika gambar yang telah ditangkap tidak terdeteksi sebagai manusia, maka sistem akan kembali ke proses awal untuk mendeteksi status Passive Infrared sensor apakah terdeteksi gerakan atau tidak. Sebaliknya, jika gambar yang telah ditangkap terdeteksi sebagai manusia, maka sistem akan mengirimkan notifikasi kepada pemilik rumah dan mengaktifkan alarm.

2.2.2 Alur Kerja Sistem

Dalam penelitian ini, alur kerja dari sistem digambarkan sebagai berikut:

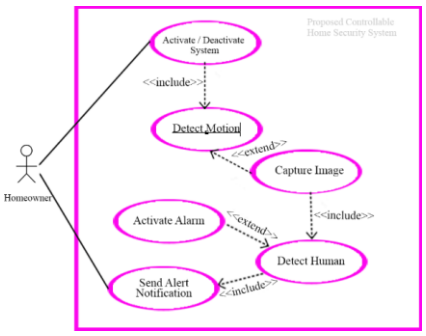


Gambar 3. Diagram Alur Kerja Sistem

Gambar 3 adalah alur kerja sistem dimulai ketika pemilik rumah mulai mengaktifkan sistem dengan meyambungkan perangkat ke power supply dan melalui aplikasi pada smartphone pemilik rumah. Kemudian Passive Infrared sensor akan aktif dan akan memantau status akses masuk ke dalam rumah apakah terdeteksi gerakan atau tidak. Jika Passive Infrared sensor mendeteksi adanya gerakan, maka web camera akan menangkap gambar dari objek tersebut dan diproses lebih lanjut. Namun jika sensor tidak medeteksi gerakan, maka Passive Infrared sensor akan terus memantau status akses masuk dari pintu.

Selanjutnya, objek yang telah ditangkap oleh web camera akan dideteksi apakah objek tersebut adalah manusia atau bukan melalui metode YoloV3. Jika objek tersebut tidak terdeteksi sebagai manusia, maka sistem akan kembali ke alur kerja awal yaitu memantau status akses masuk penyusup dengan passive infrared sensor. Namun, jika objek tersebut terdeteksi sebagai manusia, maka sistem akan mengirimkan notifikasi ke smartphone pemilik rumah dan alarm yang berupa buzzer. Alur kerja dari sistem jika digambarkan pada use case diagram akan seperti pada gambar 4.

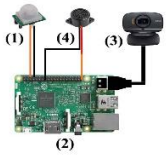
Prefix DOI : 10.3785/kjst.v1i6.347



Gambar 3. Use Case Diagram

2.2.3 Arsitektur Perangkat Keras

Dalam penelitian ini, terdapat beberapa perangkat dalam arsitektur perangkat keras yang digambarkan sebagai berikut:

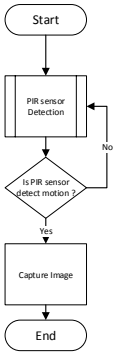


Gambar 4. Arsitektur Perangkat Keras

Passive Infrared (PIR) sensor terkoneksi dengan pin **input** pada raspberry sensor ini mendeteksi gerakan dengan memancarkan secara terus menerus infrared apabila infrared mendeteksi gerakan maka akan men trigger webcam untuk mengambil foto, foto dikirim dari webcam untuk diproses ke Raspbery.

2.2.4 Pengambilan Data & Desain Ruangan

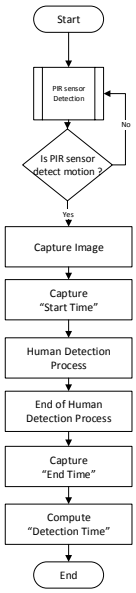
Dalam penelitian ini, penulis melakukan uji coba dengan menggunakan arsitektur smart home security system yang telah diusulkan. Sumber data yang diambil terdiri atas akurasi dan waktu proses human detection. Pengambilan data dilakukan sebanyak 100 kali pada objek manusia dan 100 kali pada objek bukan manusia. Data – data gambar objek manusia dan bukan manusia diambil dari webcam yang aktif setelah ada pemicu dari Passive Infra Red (PIR) sensor ketika pintu terbuka. Proses pengambilan data berupa gambar objek manusia dan manusia ditunjukkan pada gambar 6 berikut



Gambar 6. Alur Pengambilan Data

Prefix DOI : 10.3785/kjst.v1i6.347

Kemudian untuk pengambilan data waktu proses deteksi dilakukan dengan mengambil dua parameter. Proses pengambilan parameter yang digunakan untuk menghitung waktu proses deteksi ditunjukkan pada gambar 7 berikut.



Gambar 7. Alur Pengambilan Parameter Waktu Proses

2.3 Implementasi Sistem

Pada penelitian ini diimplementasikan dengan menggunakan Raspberry Pi 4 B yang terhubung dengan webcam Logitech C525, passive infrared sensor, dan alarm atau buzzer. Fungsi dari Raspberry Pi 3 adalah melakukan segala proses komputasi. Untuk melakukan proses komputasi tersebut, Raspberry Pi 4 B memiliki spesifikasi detail yang terdapat pada Tabel 2.

Tabel 1. Spesifikasi Raspberry Pi 4 B

Operating System	:	Raspbian
Processor	:	Quad core 64-bit ARM-Cortex A72 running at 1.5GHz
RAM	:	4 Gigabyte LPDDR4 RAM
GPU	:	VideoCore VI 3D Graphics
WLAN	:	2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN
Bluetooth	:	Bluetooth 5.0 with BLE
Ethernet	:	Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
GPIO	:	40 Pin
Display Output	:	HDMI ports supporting dual displays up to 4Kp60 resolution
USB	:	2x USB2 ports, 2x USB3 ports
Interface	:	CSI, DSI, 3.5 mm audio jack
Storage	:	Micro SD (16 GB)
Power Supply	:	5V/2.5A DC

Prefix DOI : 10.3785/kjst.v1i6.347

2.4 Implementasi Penggunaan *Dataset*

2.4.1 HoG + SVM

```
import os
import cv2
import env
import shutil
import numpy as np
import imutils as imutils

hog = cv2.HOGDescriptor()
hog.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
```

Gambar 8. Source Code Penggunaan Dataset Pada HoG + SVM (1)

Penelitian ini menggunakan sebuah dataset training yang telah disediakan oleh library OpenCV. Cara menggunakannya yaitu dengan memanggil suatu fungsi tertentu. Pada gambar 8, dapat dilihat bahwa penulis menggunakan library OpenCV dengan perintah "import cv2" untuk penggunaan dataset training. Kemudian "cv2.HOGDescriptor()" adalah fungsi dari library OpenCV untuk membuat HoG descriptor dan detector dengan beberapa default parameters.

```
image = imutils.resize(image, width=min(800, image.shape[1]))
orig = image.copy()

# Detect people in the image
(rects, weights) = hog.detectMultiScale(image, winStride=(4, 4), padding=(4, 4), scale=1.07)


# Draw the original bounding boxes
for (x, y, w, h) in rects:
    cv2.rectangle(orig, (x, y), (x + w, y + h), (0, 0, 255), 2)

# Apply non-maxima suppression to the bounding boxes using a
# fairly large overlap threshold to try to maintain overlapping
# boxes that are still people
rects = np.array([[x, y, x + w, y + h] for (x, y, w, h) in rects])
pick = non_max_suppression(rects, probs=None, overlapThresh=0.65)

# Draw the final bounding boxes
for (xA, yA, xB, yB) in pick:
    cv2.rectangle(image, (xA, yA), (xB, yB), (0, 255, 0), 2)
```

Gambar 9. *Source Code* Penggunaan *Dataset* Pada HoG + SVM (2)

Fungsi "imutils.resize(image, width=min(800, image.shape[1]))" digunakan untuk melakukan *resize* objek gambar sesuai yang dalam hal ini menjadi 800 *width*. Pada fungsi hog.detectMultiScale(image, winStride=(4, 4), padding=(4, 4), scale=1.07) digunakan untuk mendeteksi apakah ada manusia di dalam objek gambar. Selanjutnya fungsi non_max_suppression(rects, probs=None, overlapThresh=0.65) digunakan untuk menerapkan *non-maxima suppression* untuk *bounding box* yang mengalami *overlap* terlalu besar dari *threshold* sehingga dapat menjaga *bounding box* yang *overlap* tetap berada disekitar objek yang terdeteksi sebagai manusia.

2.4.2 YOLOV3

```
labelsPath = os.path.join(DATA_YOLO , "coco.names")
LABELS = open(labelsPath).read().strip().split("\n")

# initialize a list of colors to represent each possible class label
np.random.seed(42)
COLORS = np.random.randint(0, 255, size=(len(LABELS), 3),
    dtype="uint8")

# derive the paths to the YOLO weights and model configuration
weightsPath = os.path.join(DATA_YOLO, "yolov3.weights")
configPath = os.path.join(DATA_YOLO, "yolov3.cfg")

# load our YOLO object detector trained on COCO dataset
net = cv2.dnn.readNetFromDarknet(configPath, weightsPath)

imagePaths = list(paths.list_images(YOLO))
# load our input image and grab its spatial dimensions
for imagePath in imagePaths:

    #Tolerance of object
    tolerance = 0.8
    threshold = 0.4
```

Gambar 10. *Source Code* Penggunaan *Dataset* Pada YOLOV3 (1)

Pada penelitian ini, penulis menggunakan *dataset* yang sudah dilakukan *training* oleh penemu algoritma YOLOV3 tersebut. *Dataset* tersebut tersedia pada *website* https://pjreddie.com/darknet/yolo/ dan penulis menggunakan *dataset* yang bernama YOLOV3-608 pada *website* tersebut. Kemudian penulis menggunakan fungsi

Prefix DOI : 10.3785/kjst.v1i6.347

os.path.join(DATA_YOLO, ”coco.names”) untuk menentukan pengklasifikasian jenis objek gambar yang dideteksi. Jenis objek terdapat pada *file* yang bernama “coco.names”.

Gambar 11. *Source Code* Penggunaan *Dataset* Pada YOLOV3 (2)

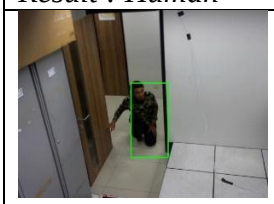
Pada fungsi “cv2.dnn.NMSBoxes (boxes, confidences, tolerance, threshold)” digunakan untuk menerapkan *non-maxima suppression* sehingga dapat menekan *overlapping bounding boxes*. Kemudian pada fungsi “if len(idxs) > 0:” merupakan pengulangan pada indeks – indeks yang sudah ada sebelumnya untuk memastikan apakah ada paling tidak satu objek terdeteksi.

HASIL DAN PEMBAHASAN

3.1 Hasil Uji Coba

3.1.1 Hasil *Human Detection* Algoritma HoG + SVM

Berikut gambar - gambar yang merupakan beberapa contoh dari hasil pendeteksian dengan algoritma HoG + SVM pada objek manusia atau objek bukan manusia:

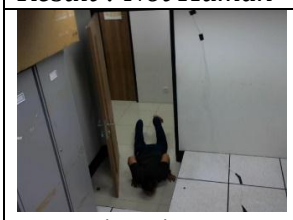
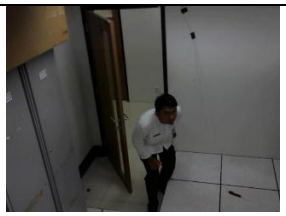
 *Detection time : 3.532s* *Result : Human*	 *Detection time : 1.891s* *Result : Human*
 *Detection time : 1.888s* *Result : Human*	 *Detection time : 1.609s* *Result : Human*

Gambar 12. Hasil Deteksi Gambar Objek Manusia : HoG + SVM

Pada Gambar 12 merupakan beberapa contoh hasil deteksi gambar objek manusia dengan menggunakan algoritma HoG + SVM. Berdasarkan gambar tersebut, algoritma ini mampu mendeteksi manusia dengan berbagai posisi yaitu berdiri, jongkok, menghadap ke belakang dan pose – pose lainnya. Algoritma ini dapat mendeteksi manusia dengan baik jika objek yang dideteksi memiliki postur tubuh manusia secara lengkap karena algoritma ini menggunakan gradien atau arah tepi sebagai dasar pendeteksian sehingga akan menghasilkan hasil akhir deteksi sebagai “Human”. Gambar – gambar tersebut merupakan hasil pemprosesan

Prefix DOI : 10.3785/kjst.v1i6.347

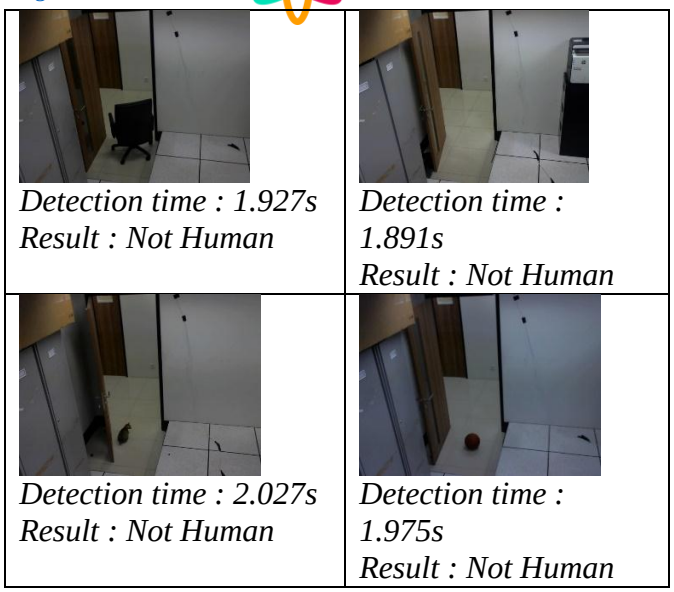
algoritma HoG + SVM yang terdeteksi sebagai manusia (*true positives*). Total *true positives* (TP) pada algoritma ini yaitu 90 dari 100 kali percobaan.

 *Detection time : 1.601s* *Result : Not Human*	 *Detection time : 1.621s* *Result : Not Human*
 *Detection time : 2.115s* *Result : Not Human*	 *Detection time : 1.860s* *Result : Not Human*

Gambar 13. Hasil Deteksi Gambar Objek Manusia : HoG + SVM

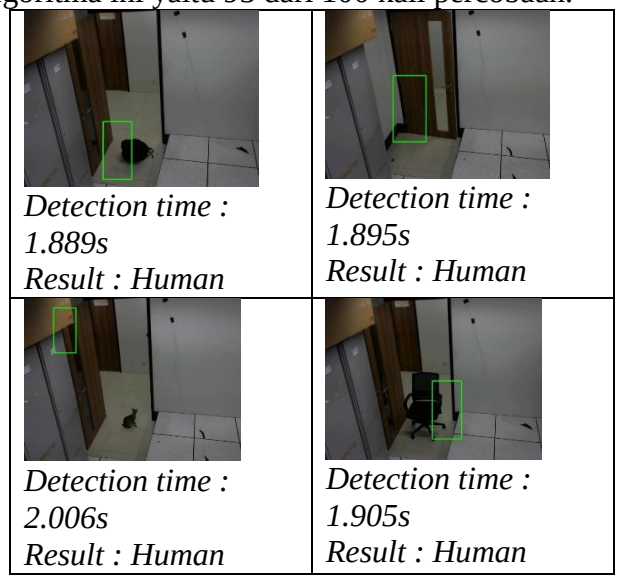
Pada Gambar 13 merupakan beberapa contoh hasil deteksi gambar objek manusia dengan menggunakan algoritma HoG + SVM. Berdasarkan gambar tersebut, algoritma ini menghasilkan hasil deteksi yang salah terhadap objek manusia pada pose tengkurap atau tiarap. Selain itu, pada objek yang dideteksi tidak penuh atau terpotong posturnya walau hanya sedikit akan menghasilkan hasil akhir deteksi sebagai "Not Human" karena algoritma ini menggunakan gradien atau arah tepi sebagai dasar pendeteksiannya. Gambar – gambar tersebut merupakan hasil pemprosesan algoritma HoG + SVM yang tidak terdeteksi sebagai manusia (false positives). Total false positives (FP) pada algoritma ini yaitu 10 dari 100 kali percobaan.

Prefix DOI : 10.3785/kjst.v1i6.347



Gambar 14. Hasil Deteksi Gambar Objek Bukan Manusia : HoG + SVM

Pada Gambar 14. merupakan beberapa contoh gambar objek hasil deteksi dengan algoritma HoG + bukan manusia menggunakan gambar tersebut, SVM. Berdasarkan berhasil algoritma ini baik objek yang mendeteksi dengan Hal ini dapat bukan manusia. algoritma ini terjadi karena gradient atau arah menggunakan tepi sebagai dasar pendeteksian dan *data set* yang ada hanya mempelajari postur atau objek manusia saja sehingga ketika ada objek yang bukan manusia maka akan menghasilkan akhir deteksi sebagai "Not Human". Gambar – gambar tersebut merupakan hasil pemprosesan algoritma HoG + SVM yang terdeteksi sebagai bukan manusia (*true negatives*). Total *true negatives* (TN) pada algoritma ini yaitu 93 dari 100 kali percobaan.



Gambar 15. Hasil Deteksi Gambar Objek Bukan Manusia : HoG + SVM

Pada Gambar 15 merupakan beberapa contoh hasil deteksi gambar objek bukan manusia dengan menggunakan algoritma HoG + SVM. Berdasarkan gambar tersebut, algoritma ini menghasilkan hasil deteksi yang salah terhadap objek bukan manusia pada beberapa objek. Hal ini terjadi karena algoritma ini mendeteksi objek dengan gradien atau arah tepi sebagai dasar pendeteksian. Dalam beberapa objek bukan manusia yang terdeteksi sebagai manusia ini karena gradien atau arah tepi yang terdeteksi mirip dengan yang ada pada *data set* sehingga akan menghasilkan hasil akhir deteksi sebagai "Human". Gambar – gambar tersebut merupakan hasil pemprosesan algoritma HoG + SVM yang terdeteksi sebagai manusia (*false negatives*). Total *false negatives* (FN) pada algoritma ini yaitu 7 dari 100 kali percobaan.

Prefix DOI : 10.3785/kjst.v1i6.347

3.1.2 Hasil Human *Detection* Algoritma Yolo

Berikut gambar - gambar yang merupakan beberapa contoh dari hasil pendeteksian dengan algoritma Yolo V3 pada objek manusia atau objek bukan manusia sebagai berikut:

Detection time : 15.419s
Result : Human

Detection time : 14.835s
Result : Human

Detection time : 14.789s
Result : Human

Detection time : 14.914s
Result : Human

Gambar 16. Hasil Deteksi Gambar Objek Manusia : Yolo V3

Pada Gambar 16 merupakan beberapa contoh hasil deteksi gambar objek manusia dengan menggunakan algoritma Yolo V3. Berdasarkan gambar tersebut, algoritma ini dapat mendeteksi berbagai pose objek manusia pada berbagai jarak dengan sangat akurat sehingga menghasilkan hasil akhir deteksi sebagai "Human". Hal ini terjadi karena algoritma ini melakukan pengenalan dan pendeteksian objek dengan jaringan saraf tunggal (*single neural network*) yang memprediksi kotak – kotak pembatas dan probabilitas kelas secara langsung pada satu langkah evaluasi. Gambar – gambar tersebut merupakan hasil pemprosesan algoritma Yolo V3 yang terdeteksi sebagai manusia (*true positives*). Total *true positives* (TP) pada algoritma ini yaitu 97 dari 100 kali percobaan.

Detection time : 14.744s
Result : Not Human

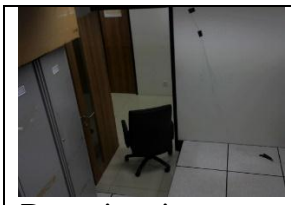
Detection time : 14.837s
Result : Not Human

Gambar 17. Hasil Deteksi Gambar Objek Manusia : Yolo V3

Pada Gambar 17 merupakan beberapa contoh hasil deteksi gambar objek manusia dengan menggunakan algoritma Yolo V3. Berdasarkan gambar tersebut, algoritma ini hanya tidak tepat mendeteksi objek manusia pada pose tengkurap atau tiarap sehingga menghasilkan hasil akhir deteksi sebagai "Not Human". Hal ini terjadi karena pose tengkurap atau tiarap ini terlihat atau mendekati seperti hewan atau objek bukan manusia pada *data set* algoritma ini. Gambar – gambar tersebut merupakan hasil pemprosesan algoritma Yolo V3 yang tidak

Prefix DOI : 10.3785/kjst.v1i6.347

terdeteksi sebagai manusia (*false positives*). Total *false positives* (FP) pada algoritma ini yaitu 3 dari 100 kali percobaan.

 *Detection time : 15.223s* *Result : Not Human*	 *Detection time : 15.723s* *Result : Not Human*
 *Detection time : 15.223s* *Result : Not Human*	 *Detection time : 15.530s* *Result : Not Human*

Gambar 18. Hasil Deteksi Gambar Objek Bukan Manusia : Yolo V3

Pada Gambar 18 merupakan beberapa contoh hasil deteksi gambar objek bukan manusia dengan menggunakan algoritma Yolo V3. Berdasarkan gambar tersebut, algoritma ini dapat mendeteksi berbagai objek bukan manusia pada berbagai jarak dengan sangat akurat sehingga menghasilkan hasil akhir deteksi sebagai "Not Human". Hal ini terjadi karena algoritma ini melakukan pengenalan dan pendeteksian objek dengan jaringan saraf tunggal (*single neural network*) yang memprediksi kotak – kotak pembatas dan probabilitas kelas secara langsung pada satu langkah evaluasi. Gambar – gambar tersebut merupakan hasil pemprosesan algoritma Yolo V3 yang terdeteksi sebagai bukan manusia (*True Negatives*). Total *true negatives* (TN) pada algoritma ini yaitu 100 dari 100 kali percobaan. Sehingga tidak ada *false negatives* (FN) atau berjumlah 0 kali FN pada algoritma ini.

3.2 Hasil Pengiriman Notifikasi

Proses yang dilakukan setelah pendeteksian menggunakan 2 algoritma yaitu dengan mengirimkan notifikasi ke telegram dan email pengguna. Email dan telegram dapat dibuka dari *smartphone* pemilik rumah. Berikut ini penjelasan detail mengenai pengiriman notifikasi ke pemilik rumah melalui telegram dan email.

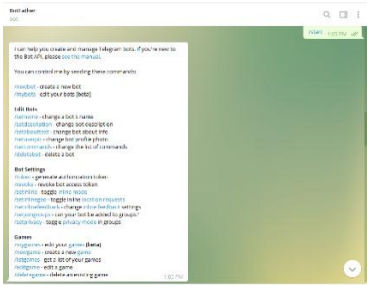
3.2.1 Pengiriman Notifikasi Melalui Telegram

Sebelum mengirimkan notifikasi ke telegram, terlebih dahulu mendaftarkan dan membuat Bot API secara resmi pada akun @BotFather telegram. Gambar 19 merupakan akun @BotFather untuk membuat Bot API.

Gambar 19. Akun @BotFather Pada Telegram

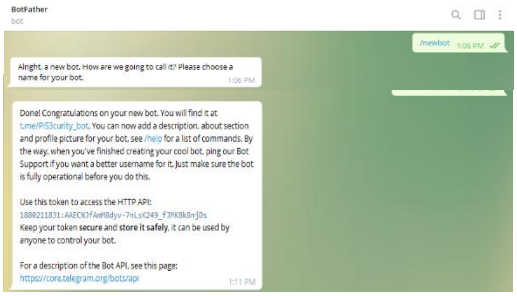
Prefix DOI : 10.3785/kjst.v1i6.347

Kemudian ketik dan kirim kata "/start" untuk menampilkan menu serta fitur pada Bot telegram. Halaman awal untuk memulai Bot dan menu serta fitur pada telegram terdapat pada Gambar 20 berikut.



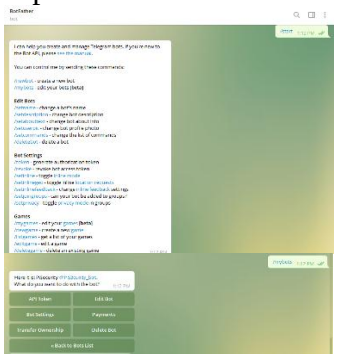
Gambar 20. Halaman Awal & Fitur @BotFather Telegram

Selanjutnya ketik dan kirim kata "/newbot" untuk membuat Bot baru pada telegram. Setelah itu ketik nama Bot yang diinginkan dan jika nama Bot yang diinginkan telah dipakai, maka dapat memasukkan nama Bot baru yang belum terpakai. Pada sistem ini, penulis menggunakan nama Bot "PiSecurity". Untuk lebih jelasnya, dapat dilihat pada Gambar 21 berikut.



Gambar 21. Nama *Bot & Token* Pada @BotFather Telegram

Jika sudah berhasil, maka akan mendapatkan pemberitahuan bahwa akun Bot telah berhasil dibuat pada telegram. Selain itu, terdapat link Bot yang sudah dibuat dan juga token untuk dapat mengakses API. Token tersebut digunakan sebagai identitas agar sistem dapat mengirimkan notifikasi ke telegram pemilik rumah.



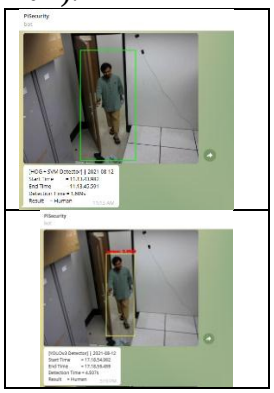
Gambar 22. Pengecekan Bot Pada Telegram

Selain itu, kita juga dapat mengecek Bot yang sudah kita miliki dengan mengetik dan mengirimkan kata "/mybots". Setelah mengirimkan kata tersebut, maka akun BotFather dari telegram akan mengirimkan balasan yang berupa nama Bot yang sudah dibuat.

Prefix DOI : 10.3785/kjst.v1i6.347

Gambar 23. Contoh Pop Up Notifikasi Pada Smartphone

Gambar 23 merupakan *pop up* notifikasi yang dikirimkan ke telegram dan email melalui *smartphone* pemilik rumah. *Pop up* notifikasi akan dikirimkan apabila Passive Infrared sensor ter-*trigger* dan proses deteksi setiap algoritma telah selesai dilakukan. Isi dari notifikasi yang dikirimkan berupa foto yang ditangkap *webcam* dan *log* yang menampilkan jenis algoritma yang digunakan untuk mendeteksi objek, tanggal, waktu awal deteksi, waktu selesai deteksi, waktu proses deteksi dan hasil deteksi (*result*) apakah objek yang terdeteksi manusia (*human*) atau bukan manusia (*not human*).



Gambar 24. Hasil Pengiriman Notifikasi ke Telegram

Gambar 24 merupakan hasil pengiriman notifikasi ke telegram. Pengiriman notifikasi dilakukan setelah proses pendeteksian selesai dan gambar yang dikirim merupakan semua objek baik yang terdeteksi sebagai manusia dan juga yang terdeteksi bukan manusia. Hal tersebut dibedakan dengan pemberian tanda "*human*" atau "*not human*" pada *log result*. Selain itu, terdapat keterangan lain berupa waktu awal deteksi dan waktu akhir deteksi sehingga dapat menentukan waktu proses deteksi dari suatu algoritma. Kemudian terdapat jenis algoritma yang digunakan dan *bounding box* sebagai petunjuk lokasi objek yang terdeteksi sebagai manusia.

3.2.2 Pengiriman Notifikasi Melalui Email

Kemudian untuk mengirimkan notifikasi melaui email, cara yang dilakukan hanya dengan membuat atau mendaftar akun email beserta password seperti pada umumnya. Dalam hal ini, penulis menggunakan akun email google atau berdomain @gmail.com. Kemudian lakukan pengaturan "Akses aplikasi yang kurang aman" (*Less secure app access*) menjadi "aktif" (on) seperti pada Gambar 25 berikut.

Prefix DOI : 10.3785/kjst.v1i6.347

Gambar 25. Pengaturan Akses Aplikasi Pada Gmail

3.3 Skenario Pengujian & Evaluasi

Dalam pengujian akurasi human detection akan diambil 100 kali atau lebih percobaan yang terdiri dari beberapa kondisi dari objek manusia yang berdiri, menghadap belakang, menghadap samping, atau kondisi lainnya. Selain itu, penulis juga akan mencoba melakukan pendeteksian objek yang bukan manusia seperti anjing, kucing, ayam atau hewan lainnya sebanyak 100 kali atau lebih sebagai pelengkap.

Dalam penelitian ini, evaluasi yang dilakukan untuk mengukur persentase akurasi yaitu dengan menggunakan rumus accuracy paradox. Pengukuran dilakukan dengan pendeteksian objek manusia yang dideteksi sebagai manusia dengan benar dan tidak benar. Selain itu, pengukuran juga dilakukan dengan pendeteksian objek bukan manusia yang dideteksi sebagai bukan manusia dengan benar dan tidak benar. Oleh karena itu akan didapatkan rumus persentase akurasi sebagai berikut:

$$A = \frac{(TP+TN)}{(TP+TN+FP+FN)} \text{ x } 100\%$$

Penjelasan simbol – simbol dalam rumus yaitu:
Accuracy (*A*) = persentase akurasi.
True Positives (*TP*) = jumlah pendeteksian yang benar terhadap objek manusia yang diuji.
False Positives (*FP*) = jumlah pendeteksian yang salah terhadap objek manusia yang diuji.
True Negatives (*TN*) = jumlah pendeteksian yang benar terhadap objek bukan manusia yang diuji.
False Negatives (*FN*) = jumlah pendeteksian yang salah terhadap objek bukan manusia yang diuji.

Kemudian evaluasi juga dilakukan untuk mengukur waktu proses dari masing – masing algoritma pendeteksian manusia. Pengukuran waktu proses diambil dari waktu mulai pendeteksian yaitu setelah objek ditangkap oleh webcam hingga waktu pendeteksian objek selesai dilakukan yaitu ketika objek dinyatakan sebagai manusia atau bukan manusia. Oleh karena itu akan didapatkan rumus penghitungan waktu proses pendeteksian manusia sebagai berikut:

$$DT = ET - ST$$

Detection Time (*DT*) = waktu proses yang dibutuhkan untuk satu kali pendeteksian objek gambar.
Start Time (*ST*) = waktu mulai pendeteksian objek yaitu setelah objek gambar ditangkap *webcam*.
End Time (*ET*) = waktu selesai pendeteksian objek yaitu setelah objek gambar telah disimpulkan sebagai manusia atau bukan manusia.

3.4 Hasil Perbandingan Akurasi dan Waktu Proses Deteksi

Berdasarkan hasil pengujian algoritma yang telah dilakukan, terdapat dua data utama yang menjadi perhatian. Dalam penelitian ini data utama yang menjadi perhatian yaitu data akurasi (A) dan waktu proses (DT) hasil deteksi dari kedua algoritma yang telah dilakukan

Prefix DOI : 10.3785/kjst.v1i6.347

pengujian. Dari 100 kali pengujian, pada masing- masing objek manusia dan bukan manusia serta pada kedua algoritma didapatkan hasil sebagai berikut:

Tabel 2. *Confusion Matrix*

Total Sample = 200		**Prediction : HoG + SVM** ---------- Negative	Positive
Actual	Negative Positive	TN = 93 TP = 90	FN = 7 FP = 10
Accuracy	= (TP + TN) / (TP+FP+FN+TN) = (90+93) / (90+10+7+93) **= 92 %**		
Total Sample = 200		**Prediction : Yolo V3** ---------- Negative	Positive
Actual	Negative Positive	TN = 100 TP = 97	FN = 0 FP =3
Accuracy	= (TP + TN) / (TP+FP+FN+TN) = (97 + 100) / (97+3+0+100) **= 99 %**		

Berdasarkan Tabel 2. dapat dilihat bahwa algoritma HoG + SVM menghasilkan rata – rata akurasi sebesar 92 % dan waktu proses deteksi sekitar 1,622 detik. Kemudian untuk algoritma Yolo V3 menghasilkan rata – rata akurasi sebesar 99% dan waktu proses deteksi sekitar 14,852 detik. Terdapat faktor - faktor yang mempengaruhi tingkat akurasi pada masing – masing algoritma sehingga menghasilkan pendeteksian yang berbeda – beda pada satu gambar yang sama, faktor – faktor tersebut yaitu:

Dataset yang digunakan pada masing – masing algoritma berbeda karena sudah disediakan pada *library* OpenCV dan sumber - sumber lainnya yang ada di literatur.

Algoritma YOLOV3 menggunakan CNNs, sedangkan HoG + SVM tidak menggunakannya sehingga akurasi YOLOV3 lebih baik karena mengkonsumsi GPU (*Graphics Processing Unit*) lebih tinggi sehingga proses deteksinya lebih lambat jika dijalankan pada perangkat yang memiliki spesifikasi GPU yang rendah.

Tabel 3. Perbandingan Spesifikasi Raspberry Pi 4 B dengan Laptop

No	*Items*	Raspberry Pi 4 B	Asus VivoBook X407UF
1	OS	Raspbian	Windows 10 Pro (64-bit)

Prefix DOI : 10.3785/kjst.v1i6.347

2	*Processor*	*Quad core 64-bit ARM-Cortex A72 running at 1.5GHz*	Intel® Core™ i3-7020U CPU @ 2.30GHz
3	RAM	4 Gigabyte LPDDR4 RAM	8 GB DDR4
4	GPU	*VideoCore VI 3D Graphics*	Nvidia Geforce mx130
5	*Storage*	16 GB Micro SD	1 TB HDD

Namun, pengujian yang dilakukan pada Laptop yang memiliki perbedaan spesifikasi dengan Raspberry Pi 4 B seperti yang tertera pada Tabel 4.3, didapatkan hasil pada Tabel 4.4 bahwa waktu proses deteksi tercepat yaitu Algoritma HoG + SVM dan Yolo V3 berbeda tipis yaitu 1,401 detik untuk HoG + SVM dan 1,405 detik untuk Yolo V3. Hal menarik terjadi dalam pendeteksian menggunakan Yolo V3 dengan menggunakan Laptop. Terjadi perbedaan waktu proses yang sangat signifikan dari sebelumnya rata – rata waktu proses deteksi sekitar 14,852 detik dengan menggunakan Raspberry Pi 4 B menjadi rata – rata sekitar 1,405 detik.

Tabel 4. Perbandingan Akurasi dan Waktu Proses Deteksi

No	Works	Methods	Accuracy	Detection Times (in Raspberry Pi)	Detection Times (in Laptop)	Total Sample
1	Surantha, N., & Wicaksono, W. R	HoG + SVM	92%	1,952s	1,401s	200
2	*This work*	Yolo V3	99%	14,852s	1,405s	200

Hal ini dapat terjadi karena Yolo V3 sangat kompleks dalam melakukan pendeteksian objek sehingga menghasilkan akurasi pendeteksian objek yang tinggi yaitu rata – rata 99%. Dengan kondisi tersebut, algoritma Yolo V3 membutuhkan spesifikasi yang lebih baik terutama pada unit pemrosesan grafis atau graphics processing unit (GPU) yang semakin tinggi jika ingin mendapatkan waktu proses deteksi yang lebih cepat. Hal ini dikuatkan dengan hasil penelitian dari (Redmon & Farhadi, 2018) bahwa dataset yang telah dilakukan training menggunakan Darknet-53 model akan mencapai pengukuran floating point terbesar per detik. Ini artinya struktur network lebih baik menggunakan GPU sehingga menjadikannya lebih efisien untuk mengevaluasi dan dengan demikian akan menghasilkan hasil yang lebih cepat.

Selain itu, dalam penelitian (S. Hossain & Lee, 2019) menyebutkan bahwa algoritma – algoritma pendeteksian objek seperti Yolo, SSD, dan R-CNN akan menghasilkan kinerja yang lebih baik dan efisien dengan menggunakan GPU yang memiliki kinerja yang lebih baik pula. Kemudian pada percobaan yang dilakukan oleh penulis dapat dilihat dengan perbandingan spesifikasi pada Tabel 4.3. Pada Raspberry Pi 4 B hanya menggunakan GPU standar yaitu VideoCore VI 3D Graphics yang menyebabkan waktu proses pendeteksian dengan menggunakan algoritma Yolo V3 menjadi lambat yaitu rata – rata 14,852 detik. Namun, waktu proses pendeteksian pada Laptop menjadi meningkat menjadi lebih cepat yaitu rata rata 1,405 detik karena menggunakan GPU Nvidia Geforce mx130.

Kesimpulan

Prefix DOI : 10.3785/kjst.v1i6.347

Dalam penelitian ini ada beberapa langkah yang dilakukan penulis, langkah pertama penulis menganalisis sistem smart home, setelah itu penulis membuat rancangan sistem smart home, kemudian setelah rancangan sistem terbentuk penulis mengimplementasikan rancangan tersebut, langkah terakhir adalah menguji sistem rumah pintar. rumah.

Sistem keamanan rumah pintar dapat digunakan untuk mendeteksi keberadaan penyusup dan mengirimkan peringatan atau pemberitahuan melalui telegram dan email. Sistem yang penulis usulkan sederhana, ekonomis dan efektif karena hanya terdiri dari Raspberry Pi 4 B, webcam, sensor infra merah pasif dan buzzer. Algoritma Yolo dan HoG yang digunakan masing-masing memiliki rata-rata akurasi sebesar 99% dan 92%, serta rata-rata waktu pemrosesan sebesar 14,852 detik dan 2 detik. Algoritma Yolo V3 perlu dimodifikasi lebih lanjut dan diterapkan pada perangkat yang memiliki Graphic Processing Unit yang lebih baik untuk mendapatkan waktu proses pendeteksian yang lebih cepat dan optimal.

Daftar Pustaka

Ahvar, E., Lee, G. M., Han, S. N., Crespi, N., & Khan, I. (2016). Sensor network-based and user-friendly user location discovery for future smart homes. *Sensors (Switzerland)*, *16*(7), 1–18. https://doi.org/10.3390/s16070969

Guo, H., Ren, J., Zhang, D., Zhang, Y., & Hu, J. (2018). A scalable and manageable IoT architecture based on transparent computing. *Journal of Parallel and Distributed Computing*, *118*, 5–13. https://doi.org/10.1016/j.jpdc.2017.07.003

Hossain Jewel, Md. K., Mostakim, Md. N., Rahman, M. K., Ali, Md. S., Dobir Hossain, S., Hossain, Md. K., & Ghosh, H. K. (2017). Design and Development of a Versatile and Intelligent Home Security System. *International Journal of Engineering and Manufacturing*, *7*(4), 60–72. https://doi.org/10.5815/ijem.2017.04.06

Hossain, M. A., & Song, B. (2016). Efficient Resource Management for Cloud-enabled Video Surveillance over Next Generation Network. *Mobile Networks and Applications*, 806–807. https://doi.org/10.1007/s11036-016-0699-3

Hossain, S., & Lee, D. J. (2019). Deep Learning-Based Real-Time Multiple-Object Detection and Tracking from Aerial Imagery via a Flying Robot with GPU-Based Embedded Devices. *Sensors*, *19*(15), 1–25. https://doi.org/10.3390/s19153371

Jing, Q., Vasilakos, A. V., Wan, J., Lu, J., & Qiu, D. (2014). Security of the Internet of Things: Perspectives and challenges. *Wireless Networks*, *20*(8), 2481–2501. https://doi.org/10.1007/s11276-014-0761-7

Lee, J. H., & Seo, C. J. (2019). Deep Learning based Pedestrian Detection and Tracking System using Unmanned Aerial Vehicle and Prediction Method. *International Journal of Innovative Technology and Exploring Engineering*, *8*(8), 794–799.

Qu, H., Yuan, T., Sheng, Z., & Zhang, Y. (2019). A Pedestrian Detection Method Based on YOLOv3 Model and Image Enhanced by Retinex. *Proceedings - 2018 11th International*

Prefix DOI : 10.3785/kjst.v1i6.347

Congress on Image and Signal Processing, BioMedical Engineering and Informatics, CISP-BMEI 2018, 2016, 1–5. https://doi.org/10.1109/CISP-BMEI.2018.8633119

Rahman, Md. W., Harun-Ar-Rashid, M., Islam, R., & Rahman, Dr. M. M. (2018). Embodiment of IOT based Smart Home Security. *International Journal for Research in Applied Science & Engineering Technology (IJRASET)*, *6*(September), 4–14. https://doi.org/10.22214/ijraset.com/fileserve.php

Ray, P. P. (2016). A survey on Internet of Things architectures. *Journal of King Saud University - Computer and Information Sciences*, *30*(3), 291–319. https://doi.org/10.1016/j.jksuci.2016.10.003

Redmon, J., & Farhadi, A. (2018). YOLOv3: An Incremental Improvement. *arXiv*, 1–8.

Seemanthini, K., & Manjunath, S. S. (2018). Human Detection and Tracking using HOG for Action Recognition. *Procedia Computer Science*, *132*(Iccids), 1317–1326. https://doi.org/10.1016/j.procs.2018.05.048

Surantha, N., & Wicaksono, W. R. (2019). An IoT based house intruder detection and alert system using histogram of oriented gradients. *Journal of Computer Science*, *15*(8), 1108–1122. https://doi.org/10.3844/jcssp.2019.1108.1122

Yoon, Y., Hwang, H., Choi, Y., Joo, M., Oh, H., Park, I., Lee, K. H., & Hwang, J. H. (2019). Analyzing Basketball Movements and Pass Relationships Using Realtime Object Tracking Techniques Based on Deep Learning. *IEEE Access*, *7*, 56564–56576. https://doi.org/10.1109/ACCESS.2019.2913953

Prefix DOI : 10.3785/kjst.v1i6.347