BAB I
PENDAHULUAN

1.1 Latar belakang

Bencana alam adalah peristiwa yang tidak dapat diprediksi dan berdampak buruk terhadap aktivitas masyarakat. Pada tahun 2023, tercatat 173 bencana alam yang terjadi di seluruh Indonesia. Tidak jarang bencana alam dapat menyebabkan korban jiwa dan kerugian secara finansial. Untuk meminimalisir korban jiwa, petugas SAR (*Search and Rescue)* akan melakukan proses evakuasi terhadap kota atau permukian yang terdampak bencana alam. Namun, proses evakuasi seringkali menghadapi hambatan, seperti akses menuju lokasi bencana alam tidak dapat dilalui. Hal tersebut tentunya akan berdampak negatif terhadap proses evakuasi, bahkan dampak terburuknya dapat menambah korban jiwa. Mengingat hal tersebut, diperlukan inovasi besar dalam hal robotika untuk menggantikan pekerjaan tim SAR, khususnya pada tempat-tempat yang beresiko untuk dilalui. Penerapan teknologi robotika dapat mendorong terciptanya sebuah robot SAR yang dapat bergerak melewati rintangan dan mengidentifikasi korban secara otonom (*autonomous*) [1]. Robot SAR perlu untuk dilengkapi dengan kemampuan pengolahan citra (*image processing*) untuk memaksimalkan kinerja dari Robot. Kemampuan tersebut dapat membantu tim untuk mengidentifikasi korban di daerah yang sulit terjangkau oleh manusia, seperti reruntuhan bangunan [2].

Pengolahan citra (*image processing*) merupakan kumpulan teknik komputasi yang digunakan untuk menganalisis, meningkatkan, dan rekonstruksi gambar. Terdapat beberapa jenis pengolahan citra yang umumnya dapat diterapkan pada robot SAR, salah satunya adalah *object detection.* Pengenalan objek (*object detection*) merupakan teknologi dalam *computer vision* yang berfokus pada pendeteksian keberadaan objek melalui gambar maupun video. Pengimpelemtasian CNN (*Convolutional Neural Networks*) dan perkembangan GPU (*Graphics Processing Unit*) mendongkrak perkembangan algoritma *object detection,* seperti : R-CNN (*Region-Convolutional Neural Networks*)*, Fast* R-CNN, *Faster* R-CNN, SSD (*Single Shot Detector*), dan YOLO (*You Only Look Once*) [3]. Pemilihan algoritma *object detection* perlu memperhatikan faktor kecepatan, akurasi, dan ruang [4]. Umumnya, algoritma yang sering digunakan untuk *object detection* adalah *Faster* R-CNN dan YOLO [5] . Perbedaan utama dari kedua algoritma tersebut berada pada cara kerjanya. *Faster* R-CNN merupakan *two-stage detector*, artinya bagian *region proposal* dan *object detection stage* akan dilakukan terpisah [5,

6]. Berbeda dengan YOLO merupakan *one-stage detector,* artinya dapat melakukan pendeteksian dan klasifikasi objek dalam satu kali iterasi [5, 6]. Hal tersebut mengakibatkan akurasi yang dihasilkan oleh *Faster* R-CNN akan lebih tinggi dibandingkan dengan YOLO. Namun, kecepatan dari YOLO akan lebih tinggi dari pada *Faster* R-CNN yang memerlukan waktu komputasi lebih lama. Penerapan algoritma YOLO akan jauh lebih optimal bila diimplementasikan ke dalam robot SAR.

YOLO menjadi salah satu algoritma yang umumnya digunakan dalam *object detection* dengan kelebihan : arsitektur jaringan yang ringan, penggabungan metode dan fitur yang efektif, dan pendeteksian yang akurat [7]. Algoritma YOLO berkembang cukup pesat, hal tersebut ditandai dengan hadirnya berbagai versi YOLO dari tahun ke tahun [8]. Berikut perbandingan versi YOLO dari V1 hingga V8 pada tabel 1.1.

Tabel 1.1 Perbandingan Versi YOLO

Versi	Framework	Backbone	Kerugian	Keuntungan
V1	Darknet	Darknet-24	Hanya dapat mendeteksi 2 objek dalam satu *grid*.	Arsitektur yang sederhana dengan satu jaringan *convolutional.*
V2	Darknet	Darknet-24	Memiliki kesulitan dalam mendeteksi objek berukuran kecil.	Terdapat *batch norm* dan *anchor boxes*. Dapat mendeteksi > 9000 kategori.
V3	Darknet	Darkent-53	Model memiliki ukuran yang jauh lebih berat dibandingkan dengan V2, akibat kompleksitas model.	Terdapat *multi-scale prediction* dan *spatial pyramid pooling* yang dapat meningkatkan akurasi pendeteksian objek berukuran kecil.
V4	Darknet	CSPDarknet-53	Model sulit untuk di implementasi dan memerlukan banyak sumber daya dalam proses *training.*	Terpapat *mish activation*, *path aggregation network* yang membuat data *augmentation* menjadi lebih baik.
V5	PyTorch	Modified CSPv7	Rentan mendeteksi *false detection* pada objek yang berukuran kecil.	Mudah untuk diimplementasikan karena dibuat dalam Pytorch.

V6	PyTorch	EfficientRep	Berfokus pada kecepatan komputasi, akan tetapi penurunan akurasi.	Menghadirkan mekanisme penentuan *loss* yang baru, seperti : VFL (*Varifocal Loss*), DFL (*Distribution Focal Loss*), dan SLoU (*Skew Intersection Over Union*).
V7	PyTorch	RepConvN	Penurunan kecepatan komputasi. Hanya optimal menggunakan GPU.	Penerapan arsitektur E-ELAN (*Extend-Efficient Layer Aggregation Networks*) yang berdampak pada peningkatan akurasi.
V8	PyTorch	ResNet-50	Memerlukan lebih banyak sumber daya komputasi dalam mentraining model.	Penerapan *anchor-free* yang dapat mengurangi jumlah *bounding box* sehingga mempercepat proses komputasi

Pada versi V1 hingga V4 umumnya tidak diimplementasikan pada perangkat karena konfigurasi yang sulit [8]. Hal tersebut disebabkan karena versi V1 dan V4 dibangun pada *framework* Darknet yang berbasis C. Umumnya, developer akan menggunakan versi YOLO yang dibangun pada *framework* Pytorch karena memiliki infrastruktur dan fasilitas yang lebih baik, dalam pengembangan model *object detection* [8]. Algoritma YOLO memiliki beberapa versi yang umumnya digunakan, yaitu YOLOv5 dan YOLOv7 [7]. Model YOLOv5 merupakan hasil perkembangan dari YOLOv3 yang dibangun di kerangka kerja PyTorch yang dinilai efektif untuk *real time object detection.* Akan tetapi, YOLOv5 memiliki beberapa kekurangan dalam hal pendeteksian objek berukuran kecil. Juga, YOLOv5 mengalami penurunan performa dalam pendeteksian objek yang mengalami perubahan pose [7]. YOLOv7 merupakan algoritma yang menerapkan *Trainable Bag of Freebies* (TBoF), dimana mengkombinasikan 3 jenis algoritma *object detection* (SSD, YOLOv3, dan RetinaNet). Sehingga akurasi yang dihasilkan mengalami peningkatan yang pesat. Akan tetapi, memerlukan lebih banyak sumber daya komputasi yang menyebabkan penurunan kecepatan komputasi [7]. YOLOv8 merupakan versi terbaru dari model YOLO yang terbit pada tahun 2023, yang bertujuan menggabungkan beberapa metode *object detection* yang dinilai efektif dalam *real time processing* [7]. Hal tersebut didukung dengan hadirnya fitur *anchor-free* model yang berfungsi memisahkan *decoupled head* untuk menangani deteksi, klasifikasi, dan regresi secara independen [9]. Selain itu, penerapan *anchor-free* model

membuat model dapat mendeteksi pusat objek langsung tanpa perlu menggunakan *offset* dari *anchor box* [9]. Penerapan *anchor-free* model dinilai dapat meningkatkan akurasi dan kecepatan proses komputasi. Algoritma YOLO dinilai cocok untuk diimplementasikan ke dalam robot SAR yang membutuhkan tingkat akurasi yang tinggi dan kecepatan dalam memproses data. Namun, tidak semua versi YOLO optimal untuk diimplementasi. Maka dari itu, akan dilakukan pengujian terlebih dahulu mengenai akurasi dan performa dari YOLOv5 dan YOLOv8 yang memiliki keunggulan dalam *real time processing*.