Deteksi Halangan Menggunakan Metode Stereo R-CNN pada Mobil Otonom

Adrian Aryaputra Firmansyah, Rusdhianto Effendie, dan Ari Santoso
Departemen Teknik Elektro, Institut Teknologi Sepuluh Nopember (ITS)
e-mail: ditto@ee.its.ac.id

***Abstrak*—Mobil otonom merupakan inovasi yang muncul akibat perkembangan teknologi komputasi yang semakin mumpuni. Mobil otonom beroperasi dengan meminimalkan campur tangan manusia sehingga dapat mengurangi kecelakaan lalu lintas yang disebabkan oleh kesalahan pengemudi. Mobil otonom memerlukan beberapa sensor agar dapat bekerja dan melakukan navigasi. Sensor yang umumnya digunakan adalah LIDAR. Namun LIDAR memiliki keterbatasan pada rentang deteksi serta adanya komponen bergerak yang membutuhkan perawatan lebih, rentan terhadap perubahan cuaca, serta harganya yang relatif mahal dibandingkan dengan alternatif lain seperti kamera stereo. Algoritma Stereo Regional Convolutional Neural Network (Stereo R-CNN) akan diterapkan pada kamera stereo untuk membuat bounding box tiga dimensi dengan memproses perbedaan sudut pandang pada gambar di kedua kamera. Metode ini dapat mendeteksi bounding box 3D dari objek pada gambar. Algoritma ini dapat mendeteksi objek dengan jarak deteksi yang lebih jauh dibandingkan dengan LIDAR, namun saat ini metode ini masih kurang feasible untuk dapat diterapkan sebagai pengganti LIDAR sebagai sensor utama karena kecepatan deteksinya yang masih lambat di kisaran 0.81 detik (1,2 frame per detik).**

***Kata Kunci*—Stereo Camera, R-CNN, Kendaraan Otonom, Deteksi Halangan, 3D Bounding Box**

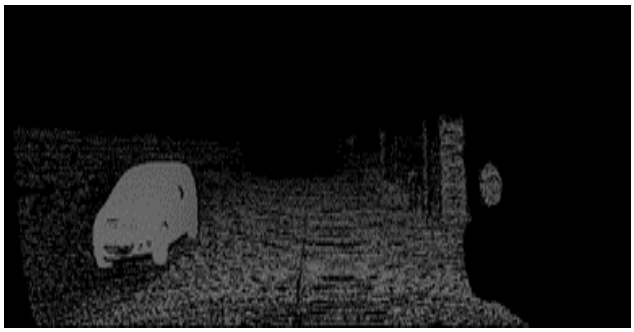
I. PENDAHULUAN

DETEKSI objek 3D merupakan dasar dari persepsi visual, prediksi gerakan, dan perencanaan untuk kendaraan otonom. Saat ini, sebagian besar metode deteksi objek 3D bergantung pada LIDAR sebagai sensor utama untuk menyediakan data jarak atau peta kedalaman. Namun, LIDAR memiliki kelemahan seperti harganya yang relatif mahal, jarak deteksi yang relatif dekat, dan informasi yang dihasilkan terbatas, yakni berjumlah 32 hingga 64 baris, dibandingkan dengan kamera yang memiliki informasi sejumlah ribuan piksel. Penelitian ini akan membahas megenai alternatif deteksi jarak selain menggunakan Lidar yakni dengan menggunakan kamera stereo dengan metode deteksi menggunakan R-CNN untuk mengklasifikasikan sekaligus menentukan *bounding box* tiga dimensi dari halangan yang terdeteksi. Hasil penelitian ini diharapkan dapat menjadi alternatif pengganti LiDAR sebagai sensor utama deteksi halangan.

Dalam penelitian ini, kami secara bersamaan mendeteksi dan mengkorelasikan objek untuk gambar kiri dan kanan menggunakan arsitektur Stereo R-CNN yang diusulkan. Arsitektur jaringan syaraf tiruan yang diusulkan dapat ditinjau pada **Error! Reference source not found.**, yang dapat dibagi menjadi tiga bagian utama. Yang pertama adalah modul Stereo RPN yang menampilkan proposal RoI kiri dan kanan yang sesuai. Setelah menerapkan RoI Align pada peta fitur kiri dan kanan, dilakukan penggabungan fitur RoI untuk mengklasifikasikan kategori objek, melakukan regresi kotak stereo 2D, dan mencari sudut pandang. Keluaran dari segmen ini membentuk batasan untuk estimasi kotak 3D, di mana kami merumuskan hubungan proyeksi antara sudut *bounding box* 3D dengan *bounding box* 2D dan *keypoint*.

Gambar 1. Sampel Data Training (kiri).

Gambar 2.Sampel Data Training (kanan).



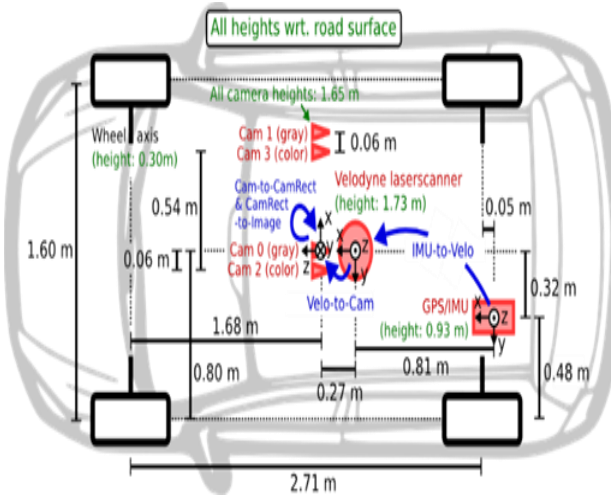
Gambar 3. Visualisasi data yang dihasilkan LiDAR.

II. PERANCANGAN ARSITEKTUR

Bagian ini akan menjelaskan mengenai sistem yang akan dibangun seperti perancangan data *training*, serta arsitektur yang digunakan dalam penelitian ini.

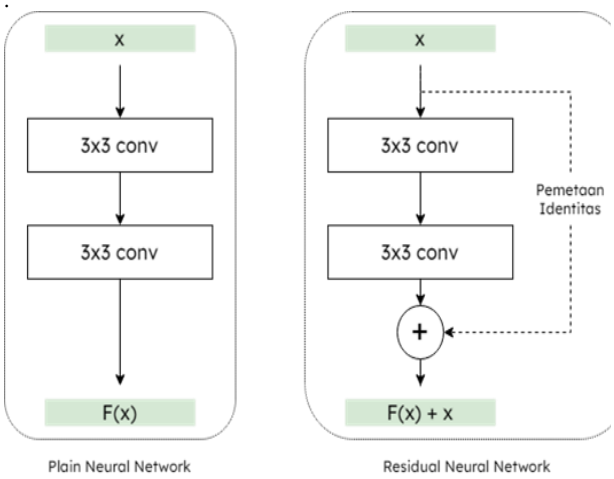
A. Data Masukan (Training Data)

Data masukan yang digunakan sebagai data *training* algoritma Stereo R-CNN adalah data gambar dari kamera



Gambar 4. Pengaturan tata letak yang digunakan untuk pengambilan dataset KITTI Stereo 2015 yang digunakan pada penelitian ini.

.



Gambar 5. Perbedaan antara RNN dengan NN konvensional.

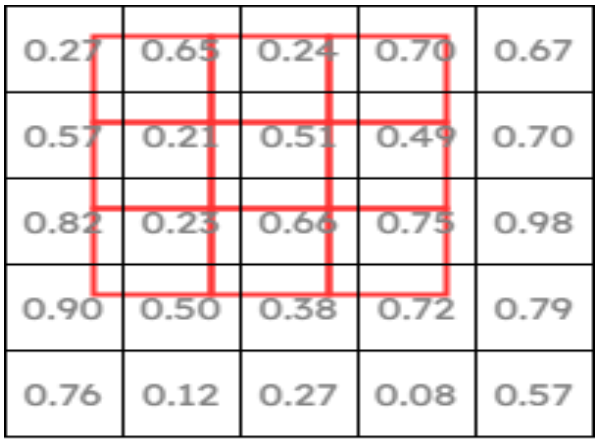
stereo, serta data yang dihasilkan oleh LIDAR. Ketiga data tersebut divisualisasikan pada Gambar, Gambar , dan Gambar .

Data masukan yang digunakan berasal dari dataset KITTI 3D Object Detection Evaluation 2017. *Dataset* ini terdiri dari 7481 data *training* dan 7518 adegan tes beserta dengan 14936 data point cloud yang dihasilkan oleh LIDAR, dengan total 80,256 objek berlabel. Dataset ini diambil menggunakan kendaraan Volkswagen Passat B6 yang telah dimodifikasi seperti pada **Error! Reference source not found.** sehingga dapat menampung sensor yang dibutuhkan. Data direkam menggunakan komputer delapan inti Intel Core-i7 yang dilengkapi dengan sistem RAID, menjalankan Ubuntu Linux, dan database realtime. Sensor yang digunakan untuk pengambilan data ini adalah sebagai berikut:

1 buah *Inertial Navigation System* (IMU): OXTS RT 3003
1 buah Laserscanner (LIDAR): Velodyne HDL-64E
2 buah kamera *grayscale*, 1.4 MP: FLIR FL2-14S3M-C
2 buah kamera berwarna, 1.4 MP: FLIR FL2-14S3C-C
4 buah lensa *Varifocal*, 4-8 mm: Edmund Optics NT59-917

Lidar berputar dengan frekuensi 10 putaran per detik, dan menangkap sekitar 100 ribu poin per siklus putarannya. Resolusi vertikal LIDAR adalah 64 poin. Kamera dipasang kira-kira sejajar dengan bidang tanah. Gambar kamera dipangkas ke ukuran 1382 x 512 piksel menggunakan mode format 7 libdc. Setelah dilakukan filter dan pemotongan, gambar menjadi sedikit lebih kecil. Kamera dipicu untuk menangkap gambar dengan frekuensi 10 *frame* per detik oleh

Gambar 7. Perbedaan penugasan target untuk regresi dan klasifikasi.



Gambar 8. Visualisasi ROI Align yang memangkas patch yang tepinya tidak sesuai dengan batasan cell.

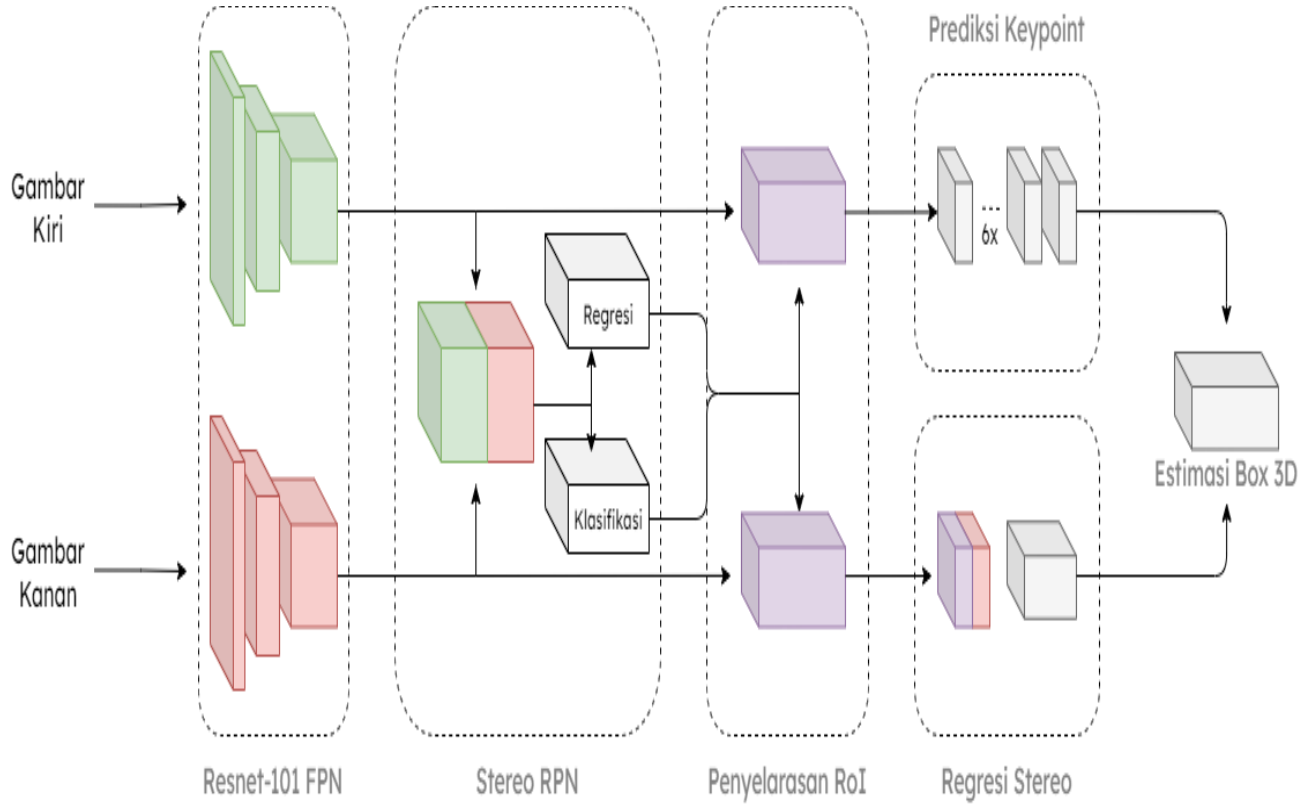
Lidar (saat menghadap ke depan) dengan periode yang disesuaikan secara dinamis (periode maksimum: 2 ms).

B. Arsitektur Stereo R-CNN

Secara garis besar, arsitektur jaringan Stereo R-CNN yang diusulkan terbagi menjadi beberapa tahap, dimulai dengan tahap deteksi *feature* menggunakan *Residual Neural Network Feature Pyramid Network* (ResNet-101 FPN), kemudian dilanjutkan dengan pencarian wilayah yang memiliki fitur-fitur khusus di kedua gambar menggunakan *Stereo Region Proposal Network* (Stereo RPN), lalu dilakukan regresi stereo pada hasil Stereo RPN sehingga dihasilkan peta kedalaman (*depth map*) yang selanjutnya akan diestimasi menjadi *bounding box* tiga dimensi. Visualisasi arsitektur Stereo R-CNN yang diusulkan dapat dilihat pada **Error! Reference source not found.**. Algoritma Stereo R-CNN ini berusaha mempelajari fitur dari suatu objek tiga dimensi. Algoritma ini akan mendeteksi dan mengasosiasikan objek tiga dimensi secara simultan.

Feature Pyramid Network merupakan suatu metode pembelajaran dimana algoritma mempelajari tiap fitur mulai dari fitur yang sederhana, dan terus meningkat hingga ke fitur yang lebih rumit. Perbedaan antara *residual neural network* dengan *neural network* konvensional dapat dilihat pada **Error! Reference source not found.**. Koneksi pintas antar *layer* ini menghasilkan kemampuan untuk melatih jaringan yang lebih dalam dari apa yang sebelumnya mungkin

Pemetaan identitas pada RNN tidak memiliki parameter apa pun dan hanya ada untuk menambahkan output dari lapisan sebelumnya ke lapisan di depan. Namun, terkadang x dan F (x) tidak akan memiliki dimensi yang sama. Perlu



Gambar 6. Visualisasi Arsitektur Stereo R-CNN yang Diusulkan.

diingat bahwa operasi konvolusi biasanya mengecilkan resolusi spasial suatu gambar, Karena itu, pemetaan identitas perlu dikalikan dengan nilai proyeksi linear W agar dimensi dari pemetaan identitas sesuai dengan residual dan memungkinkan input x dan F(x) untuk digabungkan sebagai input ke lapisan berikutnya. Secara matematis, fungsi dari residual neural network dapat dituliskan sebagai Persamaan **Error! Reference source not found.**.

Pada penelitian ini, ResNet-101 FPN akan digunakan untuk mengekstraksi fitur-fitur *sparse* yang ada pada gambar kiri dan gambar kanan. Setelah melalui ekstraksi fitur di ResNet-101 FPN, gambar akan dikurangi dimensinya menggunakan *layer* konvolusi 3×3.

Menurut S. Ren, et. all (2017), Region Proposal Network (RPN) adalah detektor objek berbasis *sliding window* [1]. Output dari RPN adalah sekelompok kotak / *proposal* yang selanjutnya akan diperiksa oleh *classifier* dan *regressor* akan adanya objek di dalamnya. Pada penelitian ini, RPN yang digunakan diubah agar dapat menerima output dari FPN dengan mengevaluasi *anchor* pada beberapa skala di *feature map*. Kunci utama dari kemampuan algoritma ini mendeteksi dan mengasosiasikan objek secara simultan adalah penentuan kotak *ground-truth* (GT) yang berbeda untuk pengklasifikasi objek dan untuk regresi kotak stereo.

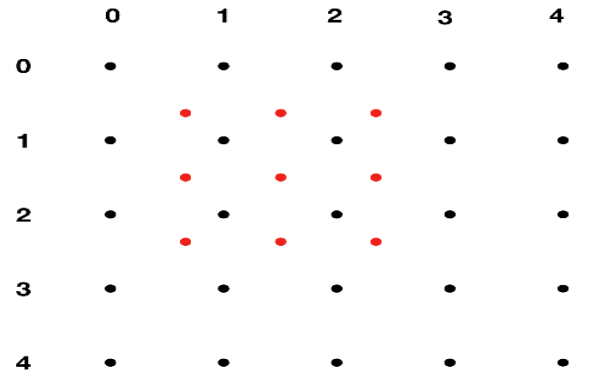
Pada Gambar dapat dilihat bahwa gabungan kotak GT kiri dan kanan (selanjutnya disebut sebagai kotak *union* GT) ditetapkan sebagai target untuk klasifikasi objek. Sebuah *anchor* diberi label positif jika rasio *Intersection-over-Union* (IoU) dengan salah satu kotak union GT di atas 0.7, dan label negatif jika IoU dengan salah satu kotak union di bawah 0.3. Selanjutnya, offset titik tengah *anchor* yang terkandung dalam kotak *union* GT gabungan dapat dihitung.

Penyelarasan RoI atau *RoI alignment* merupakan suatu teknik yang didasarkan pada interpolasi bilinear untuk memangkas *patch* dengan mulus dari peta fitur gambar penuh berdasarkan hasil RPN, dan kemudian mengubah ukuran *patch* yang dipangkas menjadi ukuran spasial yang diinginkan [2].

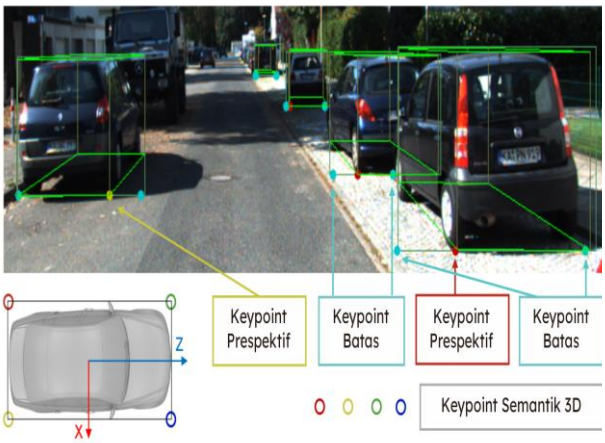
RoI alignment dapat menggunakan peta fitur (misalnya *array* 2D) dan *bounding box* hasil dari RPN sebagai masukannya. Pada penelitian ini, penyelasaran RoI akan menggunakan *bounding box* hasil dari RPN sebagai masukannya. Cara kerja dari penyelarasan RoI ini sendiri dapat divisualisasikan seperti pada Gambar . Dengan peta fitur 5x5 dan kotak pembatas merah, *RoI* Align menghasilkan peta fitur 3x3 yang dihitung dari nilai yang dibatasi dalam kotak merah. Lalu, bagaimana kita harus berurusan dengan nilai-nilai yang sebagian dibagikan oleh sel-sel di kotak merah, karena tepi hitam tidak selalu selaras dengan tepi merah? Akan lebih mudah apabila sel-sel dalam kotak hitam dan merah diumpamakan sebagai titik dalam *array* 2-D, seperti yang ditunjukkan di Gambar: Nilai interpolasi dari titik merah dihitung berdasarkan pada nilai dari empat titik hitam terdekat, yang merupakan masalah interpolasi bilinear.

Seperti yang dapat kita lihat pada Gambar, sel-sel dalam peta fitur output direpresentasikan sebagai titik-titik yang berjarak sama dalam ruang 2-D (titik merah), jadi pertama-tama kita perlu menghitung koordinatnya. Dan karena koordinat *bounding box* relatif, kita perlu mengubahnya menjadi koordinat absolut dimana *img_width* dan *img_height* merupakan lebar dan tinggi dari peta fitur, dan *height* dan *width* merupakan ukuran bounding box RPN yang diinginkan.

```
1  y_coord = linspace(y_min, y_max, height) *  \
```



Gambar 9. Titik hitam adalah sel di peta fitur, sementara titik merah adalah sel hasil RPN.



Gambar 10. Ilustrasi keypoint semantik 3D, keypoint perspektif 2D, dan keypoint batas.

```
            (img_height - 1)
x_coord = linspace(x_min, x_max, width) * \
            (img_width - 1)
```

Karena koordinat dari titik merah berupa pecahan, kita dapat mencari koordinat dari keempat titik hitam yang bertetangga dengan titik merah menggunakan *ceil* & *floor*.

```
y_low, y_high = floor(y), ceil(y)
x_low, x_high = floor(x), ceil(x)
```

Selanjutnya, kita dapat mencari nilai dari sel RPN menggunakan interpolasi bilinier. Secara keseluruhan, algoritma untuk penyelarasan ROI dapat dituliskan dalam bentuk *pseudocode* sebagai berikut:

```
for y in \
linspace(y_min, y_max, height)*(img_height - 1):
    for x in \
    linspace(x_min,  x_max,  width)*(img_width  -
1):

      y_l, y_h = floor(y), ceil(y)
      x_l, x_h = floor(x), ceil(x)

      a = image[y_l, x_l]
      b = image[y_l, x_h]
      c = image[y_h, x_l]
      d = image[y_h, x_h]

      y_weight = y - y_l
      x_weight = x - x_l

      val = a * (1 - x_weight) \
              * (1 - y_weight) + \
            b * x_weight * (1 - y_weight) + \
            c * y_weight * (1 - x_weight) + \
            d * x_weight * y_weight

      feature_map.append(val)

feature_map.reshape(height, width)
```

Tabel 1.
Hardware yang digunakan pada proses training

vCPU	8 core 16 thread allocation from Intel Xeon
GPU	Nvidia Tesla V100 – 16GB VRAM
RAM	30 GB
Disk	300 GB SSD Network Storage
OS	Ubuntu 18.04 LTS untuk Google Cloud Platform

Tabel 2.
Kriteria tingkat kesulitan

	Mudah / Easy	Normal / Moderate	Sulit / Hard
Tinggi Minimal	40 piksel	25 piksel	25 piksel
Tingkat oklusi	Terlihat sepenuhnya	Tertutup sebagian	Sulit terlihat
Maksimal terpotong	15%	30%	50%

Tabel 3.
Hardware yang digunakan pada proses deteksi

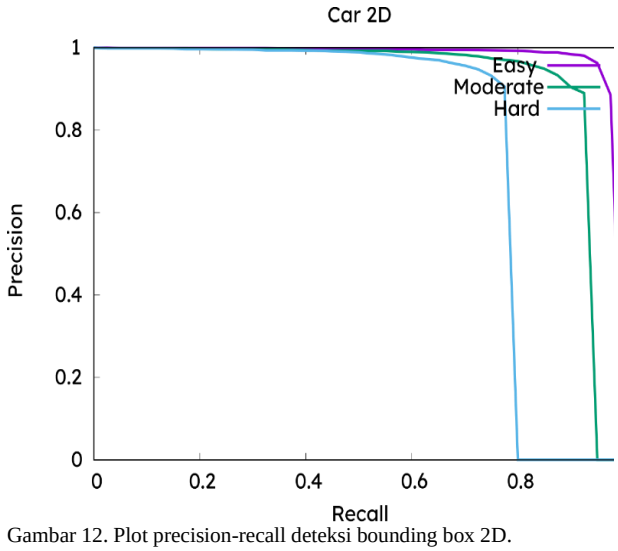
CPU	6 core 12 thread Ryzen 5 3600
GPU	Nvidia GTX 1660Ti – 6GB VRAM
RAM	16 GB
Disk	128 GB SSD NVMe
OS	Elementary OS 5.1.2 Hera

Menurut P. Li, X. Chen. et.al (2019), selain *bounding box* stereo dan sudut pandang, sudut-sudut kotak 3D yang diproyeksikan di tengah kotak dapat memberikan batasan yang lebih ketat untuk estimasi kotak 3D [3]. Seperti yang ditunjukkan Gambar , empat *keypoint* semantik 3D didefinisikan untuk menunjukkan empat sudut di bagian bawah *bounding box* 3D. Hanya ada satu *keypoint* semantik 3D yang dapat diproyeksikan ke kotak tengah (alih-alih tepi kiri atau kanan). Proyeksi *keypoint* semantik ini didefinisikan sebagai *keypoint* perspektif.

Keypoint pada algoritma ini diprediksi seperti pada metode Mask R-CNN [2]. Perlu diperhatikan bahwa pada algoritma ini, hanya peta fitur kiri yang digunakan untuk prediksi keypoint. Selanjutnya, keluaran dari peta fitur 14×14 RoI dijadikan sebgai masukan ke enam lapisan konvolusi 3×3 secara berurutan seperti yang diilustrasikan pada **Error! Reference source not found.**, masing-masing diikuti oleh lapisan fungsi aktivasi *Rectified Linear Unit* (ReLU). Langkah terakhir, lapisan dekonvolusi 2×2 digunakan untuk mengubah dimensi keluaran menjadi 28×28. Karena informasi tambahan hanya diberikan oleh koordinat u dari *keypoint*, Untuk menghemat komputasi, *channel* ketinggian dalam output $6 \times 28 \times 28$ dijumlahkan untuk

```
uncert: -10.7367, -9.4778, -4.8259, -4.8202, -6.0453, -0.4388
[epoch 12][iter 6474/6478] loss: -35.5933, lr: 1.00e-04
                    fg/bg=(8/504), time cost: 0.798754
                    rpn_cls: 0.0000, rpn_box_left_right: 0.0000, rcnn_cls: 0.0011, rcnn_box_l
uncert: -10.7377, -9.4780, -4.8262, -4.8204, -6.0454, -0.4387
[epoch 12][iter 6475/6478] loss: -34.6875, lr: 1.00e-04
                    fg/bg=(12/500), time cost: 0.797885
                    rpn_cls: 0.0000, rpn_box_left_right: 0.0000, rcnn_cls: 0.0043, rcnn_box_l
uncert: -10.7386, -9.4784, -4.8265, -4.8206, -6.0456, -0.4388
[epoch 12][iter 6476/6478] loss: -33.2985, lr: 1.00e-04
                    fg/bg=(37/475), time cost: 0.807711
                    rpn_cls: 0.0000, rpn_box_left_right: 0.0000, rcnn_cls: 0.0056, rcnn_box_l
uncert: -10.7396, -9.4787, -4.8267, -4.8209, -6.0458, -0.4388
[epoch 12][iter 6477/6478] loss: -33.9783, lr: 1.00e-04
                    fg/bg=(18/494), time cost: 0.801024
                    rpn_cls: 0.0000, rpn_box_left_right: 0.0000, rcnn_cls: 0.0059, rcnn_box_l
save model: models_stereo/stereo_rcnn_12_6477.pth
time 8.4766
(env_stereo) adrian16@gcp-stereo-rcnn:~/Stereo-RCNN$
```

Gambar 11. Hasil tangkapan layar proses training.



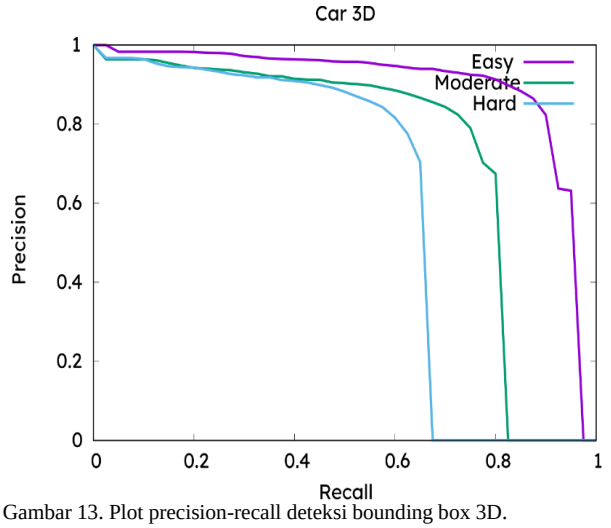
Gambar 12. Plot precision-recall deteksi bounding box 2D.

menghasilkan prediksi 6 × 28. Akibatnya, setiap kolom dalam fitur RoI akan dikumpulkan dan berkontribusi pada prediksi *keypoint*, algoritma ini dapat dituliskan dalam bentuk *pseudocode* sebagai berikut:

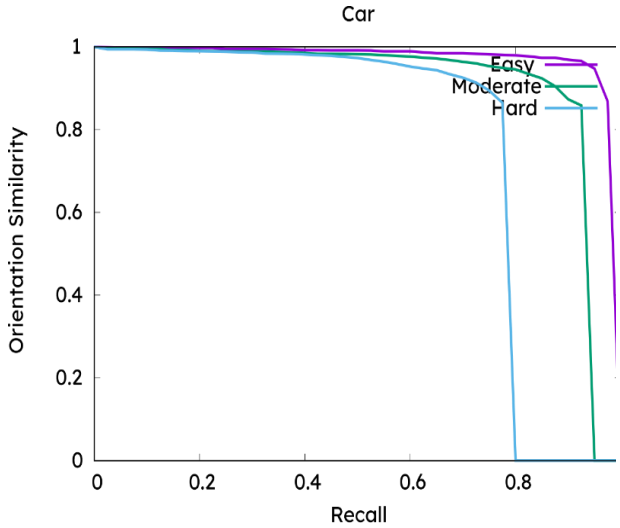
```
# 6 * cfg.KPTS_GRID * cfg.KPTS_GRID
kpts_pred = kpts_class(roi_feat_dense)
# 6 * cfg.KPTS_GRID
kpts_pred_all = kpts_pred.sum(height)
```

Dari enam *channel* yang dihasilkan pada prediksi keypoint, empat *channel* pertama mewakili probabilitas dari empat titik kunci semantik yang diproyeksikan ke lokasi u yang sesuai sementara dua *channel* lainnya mewakili probabilitas masing-masing u terletak di batas kiri dan kanan masing-masing. Perhatikan bahwa hanya satu dari empat keypoint 3D yang terletak di tengah kotak 2D, sehingga fungsi aktivasi *softmax* diterapkan pada output 4 × 28 untuk mendorong *keypoint* semantik eksklusif diproyeksikan ke satu lokasi. Strategi ini menghindari kemungkinan kebingungan jenis perspektif keypoint (sesuai dengan setiap *keypoint* semantik). Untuk titik kunci batas kiri dan kanan, fungsi aktivasi *softmax* juga diterapkan pada output 1 × 28 di masing-masing *keypoint*.

```
kpts_pred = kpts_pred_all[:3] \
    .view(-1, 4*cfg.KPTS_GRID)
kpts_prob = softmax(kpts_pred,1)
# kpts_prob.size = 4 * cfg.KPTS_GRID
```



Gambar 13. Plot precision-recall deteksi bounding box 3D.



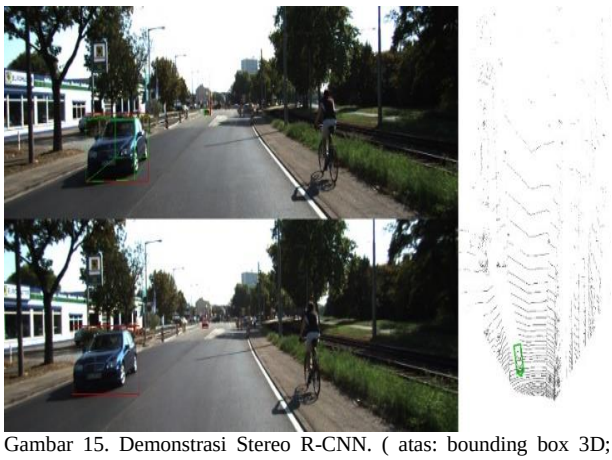
Gambar 14. Plot precision-recall deteksi orientasi mobil.

```

left_border_pred = kpts_pred_all[4] \
    .view(-1, cfg.KPTS_GRID)
left_border_prob = softmax(left_border_pred,1)
# left_border_prob.size = 1 * cfg.KPTS_GRID
right_border_pred = kpts_pred_all[5] \
    .view(-1, cfg.KPTS_GRID)
right_border_prob = softmax(left_border_pred,1)
# left_border_prob.size = 1 * cfg.KPTS_GRID
```

Selama proses training, *cross-entropy loss* pada hasil keluaran *softmax* 4 × 28 untuk prediksi keypoint perspektif diminimalkan. Hanya satu lokasi dalam keluaran 4 × 28 yang dilabeli sebagai target *keypoint* perspektif. Pada metode ini, kasus di mana tidak ada titik kunci semantik 3D yang diproyeksikan secara nyata di tengah kotak diabaikan. Pada dua keypoint batas, *cross-entropy loss* diminimalkan secara independen.

Pada segmen estimasi *bounding box* 3D ini, informasi dari segmen prediksi *keypoint*, serta informasi *bounding box* 2D yang telah dicari sebelumnya digunakan untuk mengestimasi secara kasar *bounding box* 3D. *State* dari *bounding box* 3D dapat direpresentasikan dengan persamaan $\bar{x} = \{x, y, z, \theta\}$, yang menunjukkan posisi pusat dan orientasi horizontal masing-masing *bounding box* 3D. Dengan *bounding box* 2D kiri dan kanan, sudut pandang perspektif, dan dimensi yang



Gambar 15. Demonstrasi Stereo R-CNN. (atas: bounding box 3D; bawah : klasifikasi & bounding box 2D; kanan: bird eye's view).

Gambar 16. Demonstrasi Stereo R-CNN dengan gambar mobil yang terpotong sebagian.

diregresikan, kotak 3D dapat dicari dengan meminimalkan *error* proyeksi ulang kotak 2D dan *keypoint*.

III. PENGUJIAN DAN ANSLISIS SISTEM

Pada bab ini akan membahas tentang pengujian algoritma Stereo R-CNN yang telah dirancang pada bab sebelumnya, serta pada bab ini juga akan dilakukan analisis data dari pengujian yang dilakukan. Semua data yang digunakan untuk pengujian diambil secara langsung dari dataset KITTI.

A. Pelatihan Neural Network

Pelatihan Neural Network dilakukan menggunakan Google Cloud Computing Platform (GCP) dengan hardware komputasi seperti pada Tabel 1. Training neural network dilakukan dengan input dataset sejumlah 7481 untuk masing-masing gambar kamera kiri, gambar kamera kanan, label, dan data lidar. Dari 7481 dataset tersebut, dilakukan filter pada dataset yang tidak memiliki objek deteksi sehingga hanya 6477 data yang digunakan untuk melakukan training. Training dilakukan sebanyak 12 epoch dengan kecepatan komputasi training sebesar 0.8 detik untuk setiap datanya. Lama waktu training dari algoritma ini menggunakan hardware seperti pada Tabel 1 adalah 84766 detik atau 23 jam 32 menit 46 detik.

Hasil dari proses training adalah bobot dari neural network yang disimpan ke dalam file di folder ./models_stereo dengan nama stereo_rcnn_{jumlahEpoch}_{jumlahData}.pth yang dalam kasus ini bernama stereo_rcnn_12_6477.pth seperti pada Gambar.

Gambar 17. Demonstrasi Stereo R-CNN dengan beberapa kendaraan tidak terdeteksi.

Tabel 4.
Hubungan antara jumlah objek pada gambar dengan lama waktu pemrosesan

Jumlah objek pada gambar	waktu pemrosesan (detik)		
	Rata-Rata	Terlama	Tercepat
0	0,808243	0,883845	0.723242
1	0,810192	0,893267	0,732364
2	0,809334	0,882736	0,722135
3	0,810075	0,882453	0,731351
4	0,816123	0,903240	0,723710
5	0,814956	0,893234	0,730354

B. Pengujian Neural Network

Hasil dari training neural network di tes menggunakan data testing yang terdapat pada dataset KITTI. Selanjutnya dilakukan plot nilai *Precision* dan *Recall* dari hasil tes tersebut. Kinerja deteksi objek 3D dievaluasi menggunakan kriteria PASCAL. Tingkat kesulitan deteksi dibagi menjadi tiga yakni Mudah (*easy*), Normal (*moderate*), dan Sulit (*hard*) yang didefinisikan seperti pada Tabel 2.

Data *precision-recall* dari hasil *testing* deteksi bounding box 2D, bounding box 3D, dan orientasi mobil dapat dilihat pada Gambar, Gambar, dan Gambar.

C. Demonstrasi Algoritma Stereo R-CNN

Pada demonstrasi algoritma Stereo R-CNN ini akan ditampilkan beberapa data kualitatif dari 437 data potongan video yang dihasilkan. Pada **Error! Reference source not found.** dapat dilihat bahwa algoritma dapat mendeteksi posisi dan orientasi mobil dengan cukup akurat. Algoritma ini juga mampu mendeteksi mobil kedua yang terletak di luar rentang deteksi lidar. Pada **Error! Reference source not found.** juga terlihat batasan dari algoritma ini, yakni algoritma ini hanya dapat mendeteksi objek yang sudah di *training* sebelumnya sehingga pengendara sepeda pada **Error! Reference source not found.** tidak terdeteksi.

Pada **Error! Reference source not found.** ditampilkan hasil yang kurang baik. Hal ini disebabkan karena prediksi orientasi objek yang tidak sesuai akibat mobil yang ada pada gambar terpotong sebagian. Sementara itu, pada Gambar ditampilkan trem yang tidak terdeteksi oleh algoritma Stereo R-CNN ini karena

objek trem tidak termasuk dalam data training.

D. Kecepatan Deteksi Algoritma Stereo R-CNN

Proses deteksi dilakukan dengan hardware yang berbeda dengan proses training dan testing data. Pada proses deteksi, digunakan komputer konvensional dengan spesifikasi hardware seperti pada Tabel 3.

Dilakukan uji coba deteksi objek dengan jumlah objek dalam satu frame yang bervariasi antara nol hingga empat objek. Uji coba ini dilakukan menggunakan 437 data gambar yang berasal dari data demonstrasi algoritma stereo r-cnn. Dari data pada Tabel 4 terlihat bahwa lama dari waktu pemrosesan tidak berkorelasi dengan jumlah objek yang dideteksi untuk objek berjumlah nol hingga empat pada satu gambar. Namun, tidak menutup kemungkinan bahwa waktu deteksi terpengaruh jumlah objek.

IV. KESIMPULAN

Setelah melakukan serangkaian ujicoba algoritma Stereo R-CNN, dapat disimpulkan bahwa algoritma dapat melakukan deteksi bounding box 3D dan prediksi jarak objek (mobil) pada gambar stereo yakni : (1)Hasil deteksi bounding box dan prediksi jarak objek masih memiliki error yang berkorelasi dengan jarak dan ukuran objek. Hal ini ditunjukkan dengan error yang semakin meningkat; (2)Pada frame tertentu terdapat kesalahan deteksi false positive, terutama pada objek dengan ukuran lebih kecil dari 25 piksel dan objek yang tertutupi objek lain atau terpotong lebih dari 50%; (3)Kecepatan deteksi algoritma ini berada pada kisaran 0.81 detik dengan deviasi sebesar 0.1 detik. Kecepatan deteksi ini bergantung pada spesifikasi hardware yang digunakan.

DAFTAR PUSTAKA

[1] S. Ren, K. He, R. G. Jian, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, 2017, doi: 10.2307/j.ctt1d98bxx.10.

[2] K. He, G. Gkioxari, P. Dollar, and R. Girshick, “Mask R-CNN,” *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2017-October, pp. 2980–2988, 2017, doi: 10.1109/ICCV.2017.322.

[3] P. Li, X. Chen, and S. Shen, “Stereo r-cnn based 3d object detection for autonomous driving,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 7636–7644, 2019, doi: 10.1109/CVPR.2019.00783.