Sensor Fusion Networks for 3D Object Detection

Aniket Gujarathi[1], Miguel Rogel[1]

***Abstract*— The scope of this project is to analyze sensor fusion networks for the task of 3D object detection. Specifically, we evaluate the performance of CenterFusion [1], a middle-fusion approach, on the NuScenes mini-dataset. Centerfusion [1] uses a camera based primary object detection head to obtain initial predictions. It then fuses image features with radar data through frustrum association in the intermediate stage to provide refined bounding boxes and velocity information. However, Centerfusion's [1] overall performance may be limited by the camera capabilities, as the fusion depends on the quality of the detections made by the primary detectors. We investigate these limitations by training CenterFusion on foggy data generated by a Foggy-CycleGAN. Finally, to overcome the limitations of CenterFusion [1], we propose addition of lidar in the primary head.**

I. INTRODUCTION

Object Detection is a critical module of the perception system in robotics, especially in autonomous driving. It is the task of localizing and classifying objects in the surroundings based on the data acquired by onboard sensors. Autonomous vehicles are generally equipped with a wide array of sensors such as cameras, lidars, radars, and ultrasonic sensors. These complement each other by sensing non-redundant types of real word information. For example, cameras provide rich information of the surrounding, but lack depth information. Furthermore, cameras do not work well in ill-lighted conditions like night-times, fog, among others. Lidars, on the other hand, provide dense depth information, but do not work well in adverse weather conditions such as snow. Radars provide accurate velocity information and work well in harsh weather, but provide sparse information. In this project, we aim to investigate sensor-fusion methods to fuse the information from different sensors to overcome the blind-spots of each.

Sensor fusion is the task of fusing the data from multiple sensors such that they complement each other. Using multiple sensors helps in increasing the robustness and precision of the system, but also introduces new challenges in the perception pipeline. For the task of object detection, the sensors should be aligned properly, to avoid any blind-spots, and the extrinsic inter-sensor calibration should be accurate, to fuse information without errors. Another issue is that the sensors provide data in different formats. For example, camera outputs data in the form of a multi-dimensional array depending on the illumination information, lidar provides output in the form of dense depth point-clouds, and radars provide data in the form of sparse point-clouds with multiple information fields. Thus, fusing data in different formats requires appropriate association between the corresponding representations. For this project, we use a learning-based approach called CenterFusion [1] as our baseline to fuse camera and radar data for object detection.

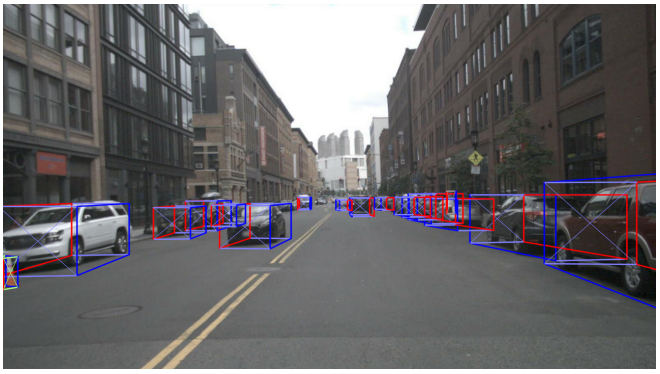


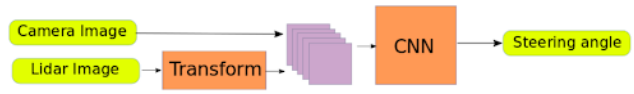
Fig. 1: 3D Object Detection on nuScenes dataset using CenterFusion

CenterFusion is a middle fusion approach that makes use of camera and radar data to perform 3D object detection and object velocity estimation. On a higher level, the network consists of two detector heads and a frustrum association module. The first detector head consists of camera based object detector that outputs preliminary 3D bounding boxes. In the intermediate stage, the radar data is associated with these preliminary bounding boxes. Finally, the camera and this associated radar data are fused and fed into the secondary detector head which refines the bounding boxes and provides accurate object velocities. As this method consists of camera based primary head, it also suffers from the drawbacks of camera only methods such as inability to perform under bad lighting conditions, harsh weather, and innacurate depth estimation. To prove this theory, we augment our dataset to generate foggy images using a cycle GAN. We train the network on this modified dataset and observe the decline in the performance. To overcome these drawbacks, we propose the integration of lidar data in the preliminary head. We use [2] as the replacement for the primary head, whilst keeping the radar association module.

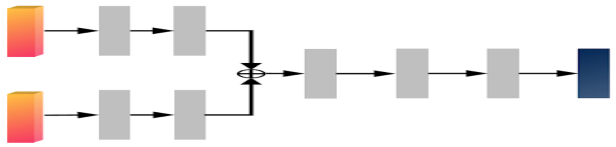
The rest of the report is organized as follows: II explains the related work in the domain of object detection. III goes through the methodology of our approach in detail. IV demonstrate our results on the task of 3D object detection on nuscenes mini dataset and foggy nuscenes dataset. V concludes the report and provides guidelines for future work.

*This work is a part of the course project for AER1515

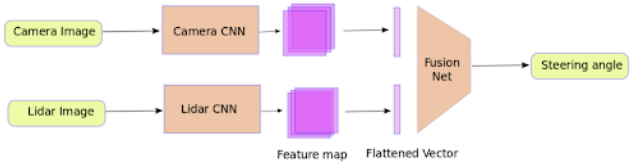
[1]Aniket Gujarathi and [1]Miguel Rogel are with the University of Toronto Institute of Aerospace Studies, Toronto, Ontario.



(a) Early Fusion architecture



(b) Middle Fusion architecture



(c) Late Fusion architecture

Fig. 2: Examples of multi-modal sensor fusion architectures. (a) In Early Fusion, unprocessed sensor data is passed early through a single network. (b) The Middle Fusion approach fuses data at an intermediate stage. (c) Sensor data is fused at the decision stage in Late Fusion.

II. Related work

A. Single Modality Object Detection

3D object detection is a well studied problem with solutions ranging from single modality to multiple modalities. Several single modality solutions exist which use only cameras, lidars or radars for object detections. One of the benefit of using a single modality for object detection is low data throughput that results in less compute required and closer to real-time operation. However, such methods are generally limited by the performances of the sensors used. 3D RCNN [3] is a camera based 3D detector that uses Fast-RCNN based detector with an additional 3D regression head. CenterNet [4] takes an image as input and outputs a keypoint heatmap to detect the center point of the object in the image. The height, width, depth, and orientation are then regressed by separate heads directly from the detected center points.

Similarly, lidar-only detection methods also exist for object detection. CenterPoint [5] uses a center-based 3D object detection pipeline to produce a bird's eye view heatmap and regression outputs. SECOND [6] uses voxels to encode Lidar data and improves inference time over previous approaches by leveraging sparse 3D CNNs to process the voxels. A Region Proposal Network is then used to compute detections from the processed voxel features.

B. Multiple Modality Object Detection

Multi-modal sensor fusion (2) is typically categorized into three methods: early fusion, middle fusion and late fusion.

1) Early Fusion Approaches: Early Fusion involves combining raw sensor data from multiple sources early in the network. Early Fusion networks are able to learn joint representations using a single model, but may be affected by spatial or temporal misalignment of the data, which can occur frequently due to innacurate calibrations. Radarnet [7] uses Lidar and Radar data at an early fusing stage. The voxel-based data for each modality, concatenated along the channel dimension, is passed to a BEV detection backbone. Radar data is then used again at a Late fusion stage to improve object detection and velocity predictions [7].

2) Late Fusion Approaches: In late fusion, the data from different sensors is combined at the final (decision level) stage. An independently trained network for each sensor is typically used. While Late Fusion methods provide the flexibility to switch between pre-trained networks for different sensor modalities, they are bottle-necked by the slower sensor network and may not be able to learn rich joint representations of the sensor data.

3) Middle Fusion Approaches: Middle fusion is a compromise between early and late fusion that extracts features from different modalities separately and combines them at an intermediate stage, enabling the network to learn joint representations and increasing modality flexibility. MV3D [8] fuses lidar BEV and front view pointclouds with RGB image features at the end of a 3D region proposal stage. The 3d proposals are generated from the lidar's BEV in the first stage abd projected to all the views. MV3D jointly fuses the features in the second stage for 3d bounding box regression and classification [8].

III. Method

A. Dataset Generation

Multiple datasets have been created to push research in autonomous driving. Some of the major datasets include Kitti [9], nuScenes [10], and Waymo [11]. All these datasets consist of a car equipped with multiple onboard sensors driving in different scenarios. Diverse datasets such as these are required to generate efficient and generalizable solutions for autonomous driving. For this project we use the nuScenes mini dataset.

1) nuScenes Data: The nuScenes Dataset is a publicly available large scale dataset released by Motional. The complete dataset consists of 1000 driving scenes in Boston and Singapore. Each scene is 20 seconds long and consists of diverse driving maneuvers. For computer vision tasks, ground truth annotations are provided for 23 object classes with accurate 3D bounding boxes at a frequency of 2 Hz. The sensor suite in [10] consists of 6 cameras, 1 lidar, 5 radars, GPS and IMU, as shown in Fig. 3.

For this project, we use the nuScenes mini-dataset consisting of 10 scenes from the complete dataset. A mini-dataset is useful to explore the data without the need to download the entire dataset and in case of scarce compute resources.

2) Foggy nuScenes: Although the nuscenes dataset is collected in diverse lightning conditions like night-time and daytime, it does not cover all types of environmental variations. As the majority of the self-driving datasets are generated at specific locations around the world, it is difficult to capture all types of variations in weather like snow, fog, rain, etc,

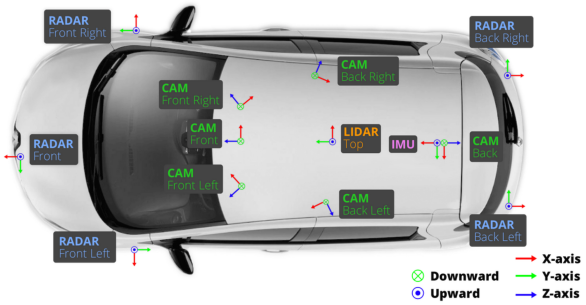


Fig. 3: nuScnenes sensor placement [10]

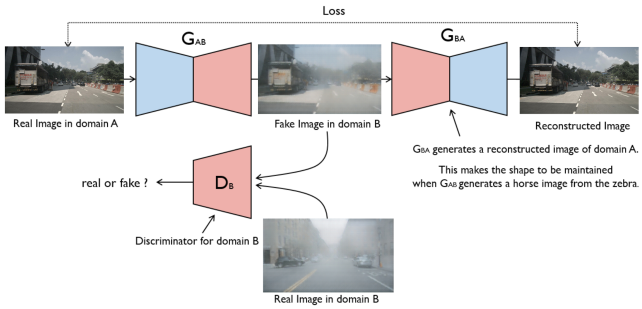


Fig. 4: CycleGAN Structure. A cycle GAN consists of two generators $G_{AB} : A \rightarrow B$ and $G_{BA} : B \rightarrow A$. G_{AB} generates outputs that look more like the images in domain B, and G_{BA} generates outputs that look more like the original domain A.

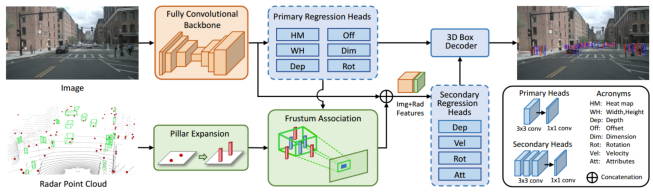


Fig. 5: CenterFusion Architecture [1]

leading to a lack of datasets with a wide variety in weather conditions. For robust and weather invariant models, there is a dire need of datasets capturing data in every condition. However, achieving this is a difficult task because of the effort, resources, and data storage required.

To include variations in data during training, many methods use simulators such as the GTA-V videogame or Carla [12]. Although simulators allow limitless data generation capacity, there still exists a gap between real world data and simulated data. Another option to generate data is the use of GANs [13]. GANs are generative models that generate data similar to the training set. However, generating paired images to train GANs require considerable effort due to the need to collect identical image pairs in different conditions. In situations where paired images are not present in the dataset, the concept of a CycleGAN [14] can be employed.

For this project, we use the Foggy-CycleGAN [15] to generate foggy images of the nuScenes dataset. An image a from domain A is translated to an image b in domain B by the generator G_{AB}, and the result $\hat{b}$ is converted back to the original domain to give the output $\hat{a}$. The final result $\hat{a}$ is supposed to resemble a. The cycle consistency loss used here is:

$$L_{cyc} = E_a(||G_{BA}(G_{AB}(a)) - a||_1)$$

The reverse process from domain B to domain A makes use of the backward cycle consistency loss. The loss function for the Foggy cycleGAN is:

$$\begin{aligned} L_{cycleGAN} = E_b[logD_{BA}(b)] + E_a[logD_{BA}(a)] \\ + E_a[logD_{AB}(a)] + E_b[logD_{AB}(b)] \\ + E_a(||G_{BA}(G_{AB}(a)) - a||_1) \\ + E_b(||G_{AB}(G_{BA}(b)) - b||_1) \end{aligned}$$

To generate our foggy images on nuScenes data, we make use of transfer learning, where we use the pretrained weights from [15] and retrain the network on nuscenes data. This data augmentation not only increases the size of the dataset but also incorporates diversity in the data.

B. CenterFusion

We use [1] as our baseline for sensor fusion. CenterFusion is a middle fusion approach that fuses camera and radar data. It consists of two detector heads and an intermediate association mechanism. The first stage consists of a primary detector head that takes camera data as input and outputs preliminary bounding boxes. [1] uses CenterNet [4] for the primary head with Deep Layer Aggregation [16] as the backbone. The extracted image features from the backbone are fed into a regression head to output the image size, depth, center offset, and the rotation.

In the intermediate stage, the radar data is accurately associated with the primary bounding boxes using frustrum based association. Given the 2D primary bounding boxes and depth information, a frustrum is created, the radar data is associated with the frustrum. In case of multiple radar points, the closest point to the center is taken as the radar point to be associated with the object. Radars generally do not give accurate elevation (height) information. To address this issue [1] use a method of pillar expansion, where each radar point is extended vertically to a fixed height and the pillar is associated with the frustrum.

After the radar association, depth and velocity information from the radar points is used to generate heat-maps which are fused as extra channels with the image. This joint representation is then passed to a secondary decoder and regression head to output refined 3D bounding boxes and the detection attributes namely the class scores, bounding box dimensions, object velocity, and pose.

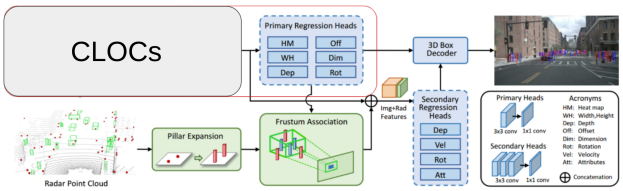


Fig. 7: Camera, Lidar, Radar fusion architecture [1]

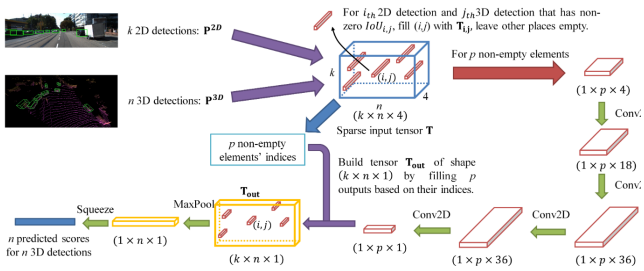


Fig. 6: CLOCS Architecture

C. CLOCs

CLOCS [2] is a late fusion approach for camera and lidar based architectures. CLOCs computes 3D bounding boxes as follows: First, individual 2D and 3D detection candidates are converted into a sparse tensor set of joint detection candidates, based on the IoU values between the 2D and projected 3D detections. Then, a 2D CNN is used to process the non-empty elements in the sparse input tensor. This CNN is a set of 1 x 1 2D Convolution Layers. The processed tensor is then used to refine the initial 3d detection candidates by regressing their prediction scores [2]. The modularity of CLOCS allows us to easily insert lidar data into the architecture of CenterFusion [1], as the 2D and 3D detections can come from arbitrary networks.

D. Camera, Lidar, Radar Fusion

To overcome the drawbacks in middle-fusion approaches relying on a monocular sensor primary detector, we propose to replace the first detector head in CenterFusion [1] with CLOCs [2] as our baseline . The preliminary bounding boxes in the original CenterFusion architecture [1] are improved by fusing Lidar 3D detections obtained through SECOND [6] with 2D detections obtained with YOLOv5 [17]. These detections are fused with CLOCs [2] to obtain higher quality features to be used by CenterFusions Primary Regression heads. The envisoned architecture is shown in Figure 7.

IV. Results

A. Data Generation

As explained in Sec. III, we use the nuScenes mini-dataset for training our fusion networks. We carefully choose dataset scenes that provide variety in lighting and traffic conditions to better generalize our network. We use scenes 61, 553, 655, 757, and 1100 as our training set, scenes 0103, 0916 as the validation set and scenes 1077 and 796 as our test set.

TABLE I: Foggy-nuScenes Dataset Examples

TABLE II: Qualitative Comparison between clear nuScenes and Foggy-nuScenes data

For Foggy-nuScenes, we train the Foggy-cycleGAN on the same train-val-test split while using the pretrained weights as a starting point. The reference code used for Foggy-cycleGAN training can be found at [1]. Fig. I shows some examples of the foggy dataset.

B. CenterFusion results

We train CenterFusion on our augmented nuscenes dataset. The dataset consists of 1660 camera images and correspoding radar pointclouds. We augment the camera images using the Foggy-cycleGAN to double the images, but keep the radar data same as radar data is unaffected due to fog. We train the network on Google Colab Pro for 30 epochs. We use batch size of 8, learning rate of $2.5e^{-4}$. As radar data is sparse, we use 6 sweeps of radar data as a single pointcloud such that it is easier to associate. The preliminary detector head follows CenterNet [4] and uses DLA [16] as the backbone. Note that we freeze the backbone while training centerfusion.

Fig. II show some qualitative results on nuScenes and Foggy-nuScenes data. We can see that due to the fog (right image), centerfusion is not able to detect all the objects in the scene, whereas the left image shows better detection results. Table III shows the average precision scores for the same and we can see a dip in the performance of CenterFusion on the Foggy data. This proves our theory that a sensor fusion approach with camera only preliminary head is limited to the performance of the sensor.

C. CLOCs results

Although CLOCs is originally trained on KITTI dataset, we tried training it on the nuScenes dataset. We made use of the scripts provided by [2] to convert our data to the KITTI format. Unfortunately, we weren't able to properly train CLOCs on the nuScenes data. This may be because of missing properties in the conversion of nuScenes data

[1]https://github.com/ghaiszaher/Foggy-CycleGAN
[2]https://github.com/nutonomy/nuscenes-devkit

Weather	AP score height
Clear	0.446
Foggy	0.141

TABLE III: Average Precision results with CenterFusion on clear nuScenes and Foggy-nuScenes datasets

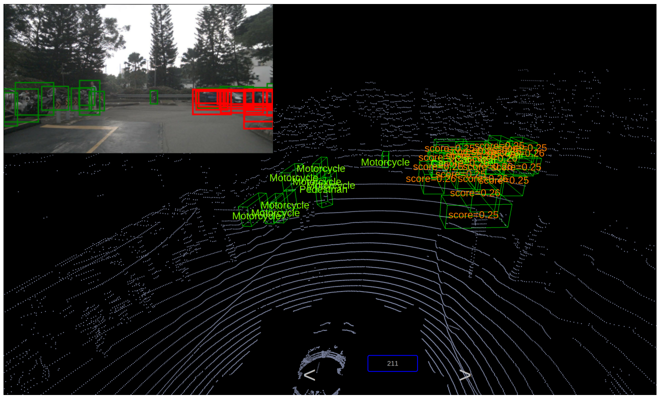


Fig. 8: Qualitative Results of CLOCs on nuScenes data

to KITTI format, or a bug in the CLOCs implementation. Nonetheless, we show our qualitative prediction results in Fig. 8.

V. Conclusion and Future Work

In this project, we analyzed CenterFusion's performance on 3D object detection. We trained CenterFusion on the nuScenes mini dataset, and compared results of CenterFusion on real nuScenes images and foggy images generated through Foggy-CycleGAN. Our results show that even though CenterFusion incorporates radar data, the middle-fusion architecture depends heavily on the primary camera based detection heads, as we see in the performance decrease on foggy images. We also begun work on camera, lidar and radar fusion using CLOCs and CenterFusion, but weren't able to obtain the desired results as CLOCs was primarily tested on the KITTI dataset. Future work includes attempting to fix CLOCs to work on nuScenes data and compare performance in harsh weather conditions between radar and lidar 3D detections in our proposed fusion network.

References

[1] R. Nabati and H. Qi, "Centerfusion: Center-based radar and camera fusion for 3d object detection," 01 2021, pp. 1526–1535.

[2] S. Pang, D. D. Morris, and H. Radha, "Clocs: Camera-lidar object candidates fusion for 3d object detection," *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10 386–10 393, 2020.

[3] A. Kundu, Y. Li, and J. M. Rehg, "3d-rcnn: Instance-level 3d object reconstruction via render-and-compare," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3559–3568.

[4] K. Duan, S. Bai, L. Xie, H. Qi, Q. Huang, and Q. Tian, "Centernet: Keypoint triplets for object detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 6568–6577.

[5] T. Yin, X. Zhou, and P. Krähenbühl, "Center-based 3d object detection and tracking," *CVPR*, 2021.

[6] Y. Yan, Y. Mao, and B. Li, "Second: Sparsely embedded convolutional detection," *Sensors*, vol. 18, no. 10, p. 3337, 2018.

[7] B. Yang, R. Guo, M. Liang, S. Casas, and R. Urtasun, "Radarnet: Exploiting radar for robust perception of dynamic objects," in *European Conference on Computer Vision*. Springer, 2020, pp. 496–512.

[8] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 1907–1915.

[9] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[10] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *CVPR*, 2020.

[11] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine, V. Vasudevan, W. Han, J. Ngiam, H. Zhao, A. Timofeev, S. Ettinger, M. Krivokon, A. Gao, A. Joshi, Y. Zhang, J. Shlens, Z. Chen, and D. Anguelov, "Scalability in perception for autonomous driving: Waymo open dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[12] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[13] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27. Curran Associates, Inc., 2014.

[14] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Computer Vision (ICCV), 2017 IEEE International Conference on*, 2017.

[15] G. Zaher and A. Hajdu, "Simulating weather conditions on digital images," in *PhD. Thesis University of Debrecen Faculty of Informatics*, 2020.

[16] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018.

[17] G. Jocher, A. Chaurasia, A. Stoken, J. Borovec, NanoCode012, Y. Kwon, K. Michael, TaoXie, J. Fang, imyhxy, Lorna, Yifu), C. Wong, A. V, D. Montes, Z. Wang, C. Fati, J. Nadar, Laughing, UnglvKitDe, V. Sonck, tkianai, yxNONG, P. Skalski, A. Hogan, D. Nair, M. Strobel, and M. Jain, "ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation," Nov. 2022. [Online]. Available: https://doi.org/10.5281/zenodo.7347926