

前方视觉检测感知

目录

前方视觉检测感知.....	1
一、 双目立体算法.....	2
二、 障碍物检测与交通信号灯算法.....	5
三、 地面标线检测算法.....	6
四、 交通标志牌检测.....	7
五、 障碍物与深度信息融合.....	8
六、 工程编译与运行.....	9
参考文献：.....	10

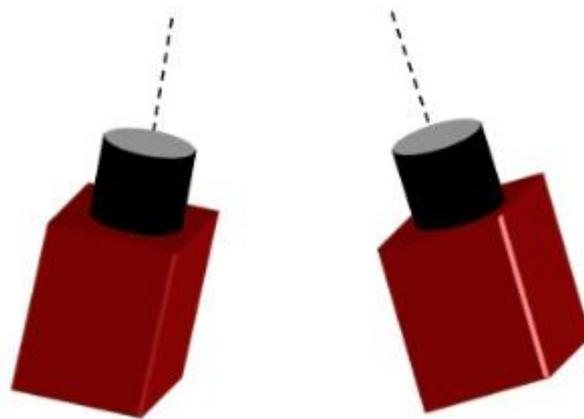
前方视觉检测感知主要处理基于 camera 的检测任务，主要包括了障碍物的检测，地面标线的检测，交通标志牌的检测，交通信号灯的检测和双目立体视觉算法。工程代码位于项目目录的 src/detection/front_vision/。该工程具有相对的独立性，工程将处理后的结果通过共享内存发出，但不依赖项目的其他工程。

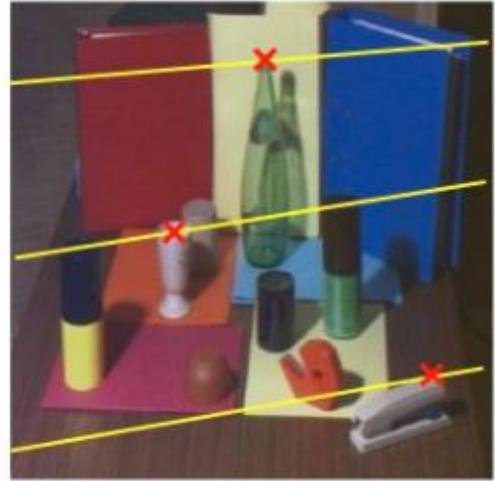
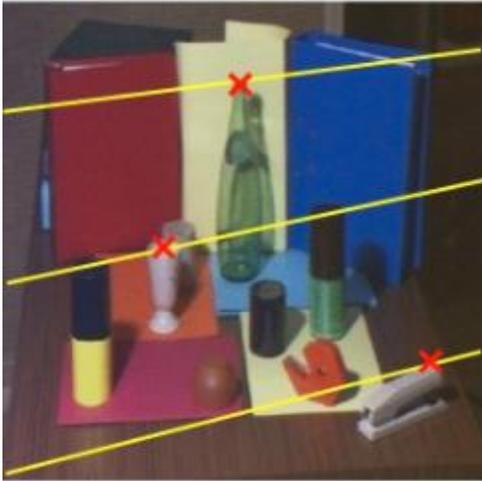
一、双目立体算法

基于 image 的检测算法，只能将检测到的障碍物以像素坐标系下的 bounding box 描述障碍物的位置，而自动驾驶对障碍物的感知信息要求在三维空间下。为了将像素坐标的 bbox 转换到三维空间。采用双目立体算法，计算出图像像素的深度信息，再将图像与双目 depth 图配准，通过像素之间的一一映射关系，可将 image 图像的 bbox 映射到 depth 图以获得三维空间信息。

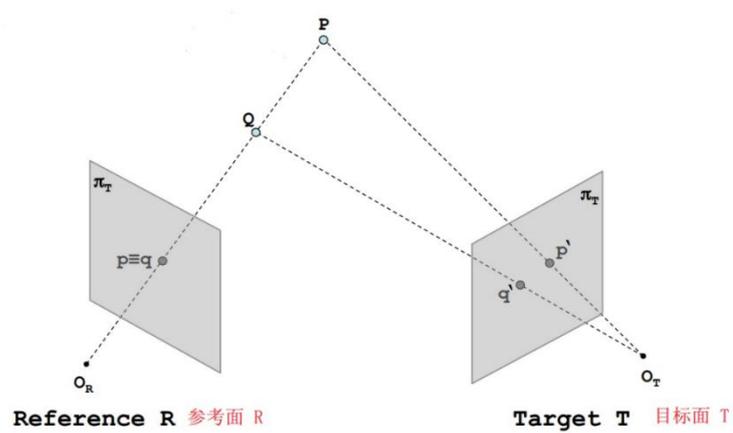
双目算法的原理：

一个双目系统包括两个看向同一方向的摄像机，通常通过人工安装的位置如下图

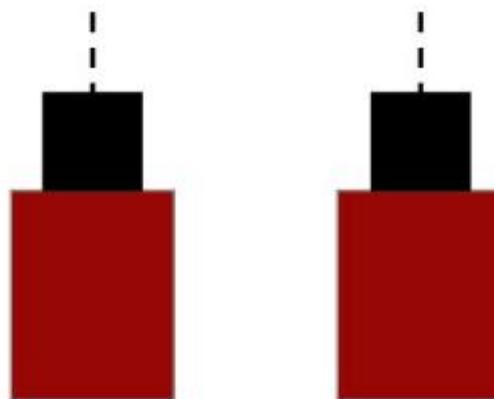


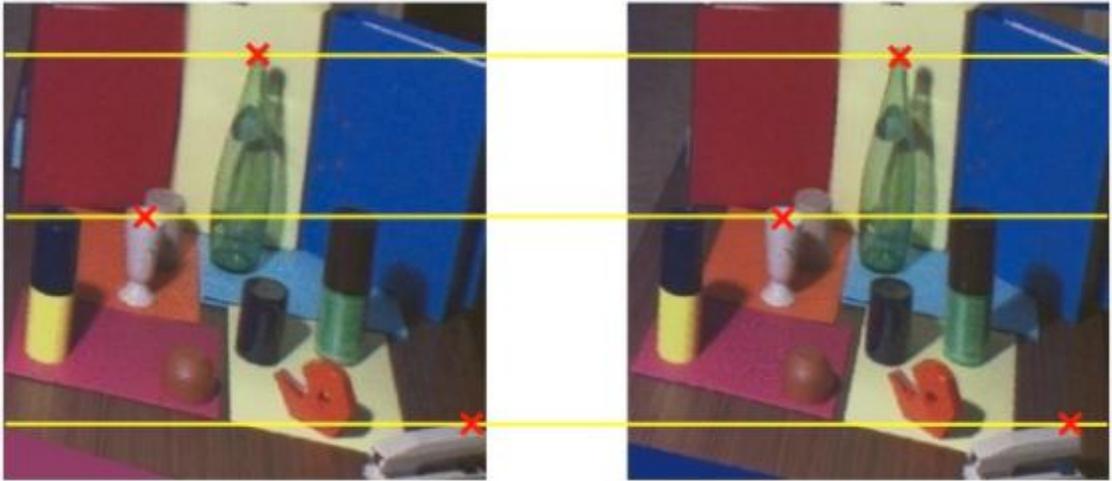


两路摄像头的焦距不能落到同一个平面上，其光路图如下图，这种存在一定偏角的相机安装方式将导致立体匹配的计算量激增和无法用三角形相似来计算出深度信息。

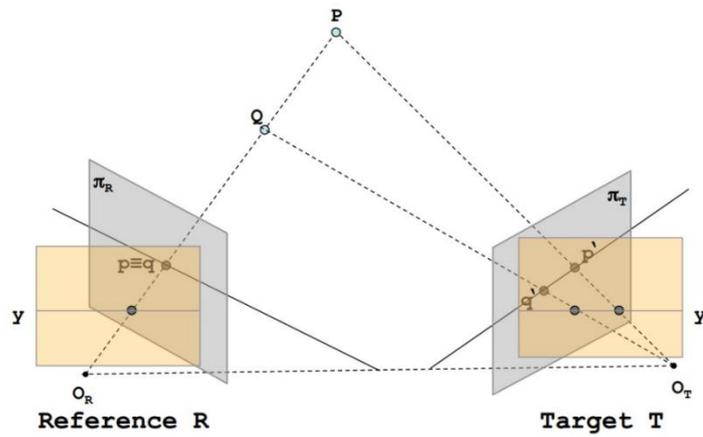


理想情况下，双目系统左右相机的焦平面位于同一平面，并且焦点在水平方向上是对齐的。如下图所示：





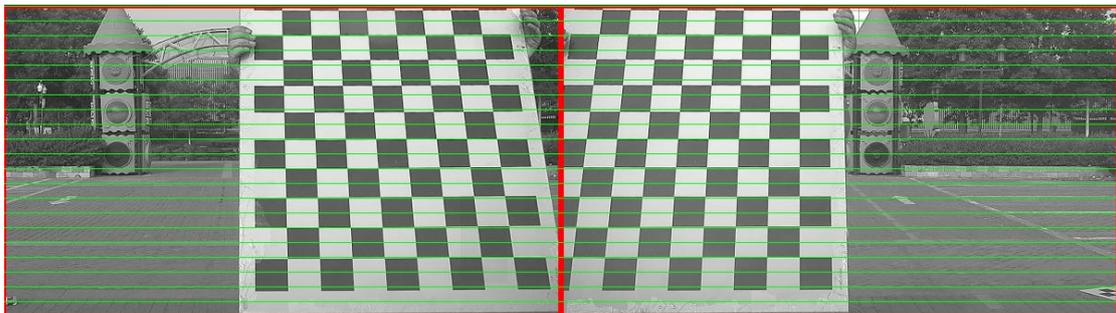
其光路图如下：浅黄色的焦平面是理想情况下，相机的焦平面。



为了消除安装相机直接的偏角，需要对左右相机做极线几何，将左右图像重新 remap 到新的像素位置以实现极线几何，同时需要再裁剪出左右相机最佳匹配的像素区域。工程参考

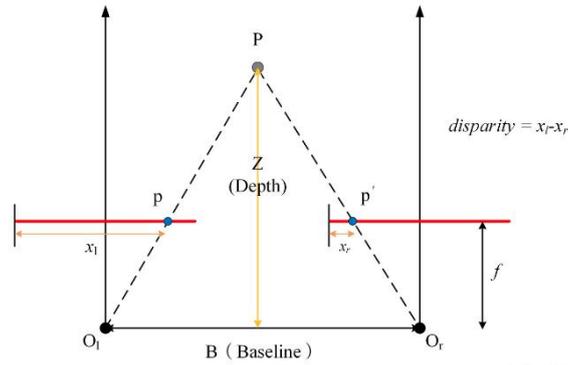
Zhang Z.等人提出的方法^[1]对左右 images 做极线几何，对左右图像做了极线几何的效果

如下图



当两个摄像头极线几何后，左右图像的物体在水平位置上是对齐的。此时，根据可以在同一水平线上匹配左右图像的特征点得到视差图，本工程参考由学者 Hirschmüller 在 2005 年所提出的 sgm 方法^[2]。

根据视差图，利用三角测距的方法计算出个像素的深度信息，如下图：

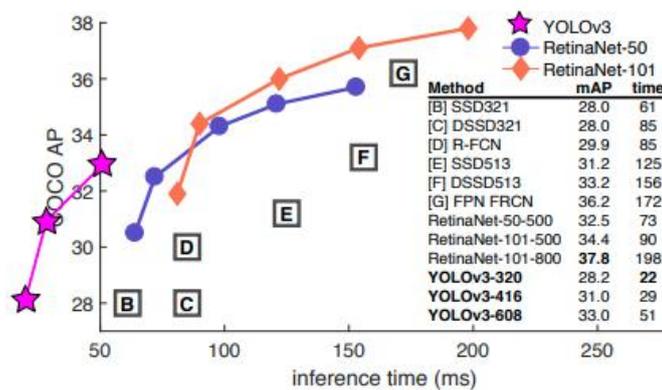


通过三角形 PO_1O_r 和 Ppp' 相似可以得到

$$Z = \frac{B \cdot f}{x_1 - x_r} = \frac{B \cdot d}{d}$$

二、障碍物检测与交通信号灯算法

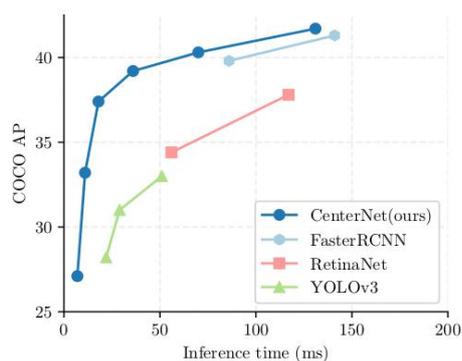
障碍物检测的目标是通过输入图像，能给出在图像上定位障碍物的位置以及包裹障碍物的 bbox，本工程采用 Joseph Redmon 提出的 yolo-v3^[8] 算法对目标进行检测，yolo-v3 是在 yolo-v2 的基础上提出的，他在算法精度和速度都有提高，如下图所示：



本工程基于 yolo-v3 设计了障碍物，并将训练的模型部署到 nvidia AGX xavier 上，在 nvidia AGX xavier 上测试的运行速度可达到 20fps 左右。

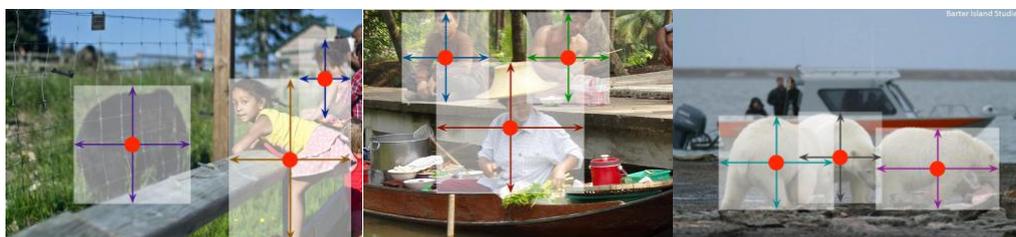
三、地面标线检测算法

地面标线检测算法，主要是能够在 image 上定位出地面标线的位置和包围地面标线的 bbox，地面标线检测是一个典型的图像检测任务，本工程采用基于深度学习的方法，进行对障碍物检测，主要方法参考得克萨斯大学奥斯汀分校 Xingyi Zhou 在 2019 年提出的方法 CenterNet^[3]。CenterNet 不论在精度和速度上都表现的比较优秀，如下图与其他主流算法的对比：



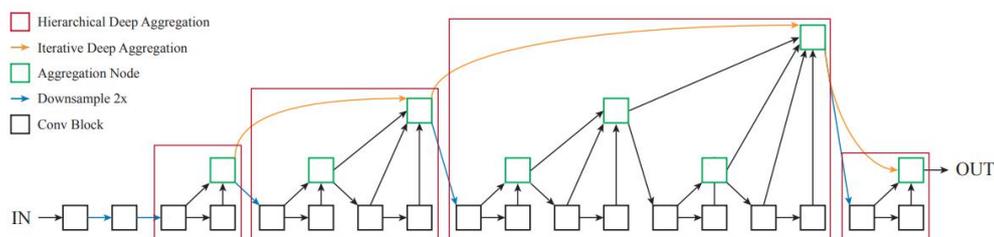
CenterNet 提出了一种基于中心点的目标检测算法，将每个目标映射成一个高斯

核热图：



基于高斯核中心，预测目标的长和宽。详细说明请阅读参考文献[3]。CenterNet 提出了一种新颖目标检测框架和 Loss 函数，为了实现目标检测，本工程采用 Yu, Fisher 等人提出的

《deep layer Aggregation》[4]方法作为网络的 backbone,本网络可以在网络深度方法对信息进行聚合,以达到对图像低频信息和高频信息的聚合,有利与大目标和小目标的检测效果。网络结构如下图所示,详细内容请阅读参考文献[4]。



结合 centerNet 和 DLA 的模型训练和测试实在 python 环境下实现,在 TITAN RTX P0 的帧率可达到 100fps 以上。

算法工程化部署部分,本工程采用 TensorRT 定点化技术,将模型转换到 float16 的 TensorRT 的系列化模型,并且编写了基于 C++ 代码的前处理和后处理过程,为了加速整个 pipeline 的速度,本工程还对后处理采用 cuda 加速处理。在部署到 NVIDIA AGX xavier 上,在满载,单独运行检测模型时帧率可达到 50fps 以上。

四、交通标志牌检测

交通标志牌检测属于一个典型的小目标检测任务,CenterNet^[3]在 heat_map 的 loss 函数上正好也使用了 focal loss 处理了目标不均衡的问题。由于数据集标注的问题,本工程将可检测的目标类别分为三大类别(指示标志、禁止标志、警告标志)。

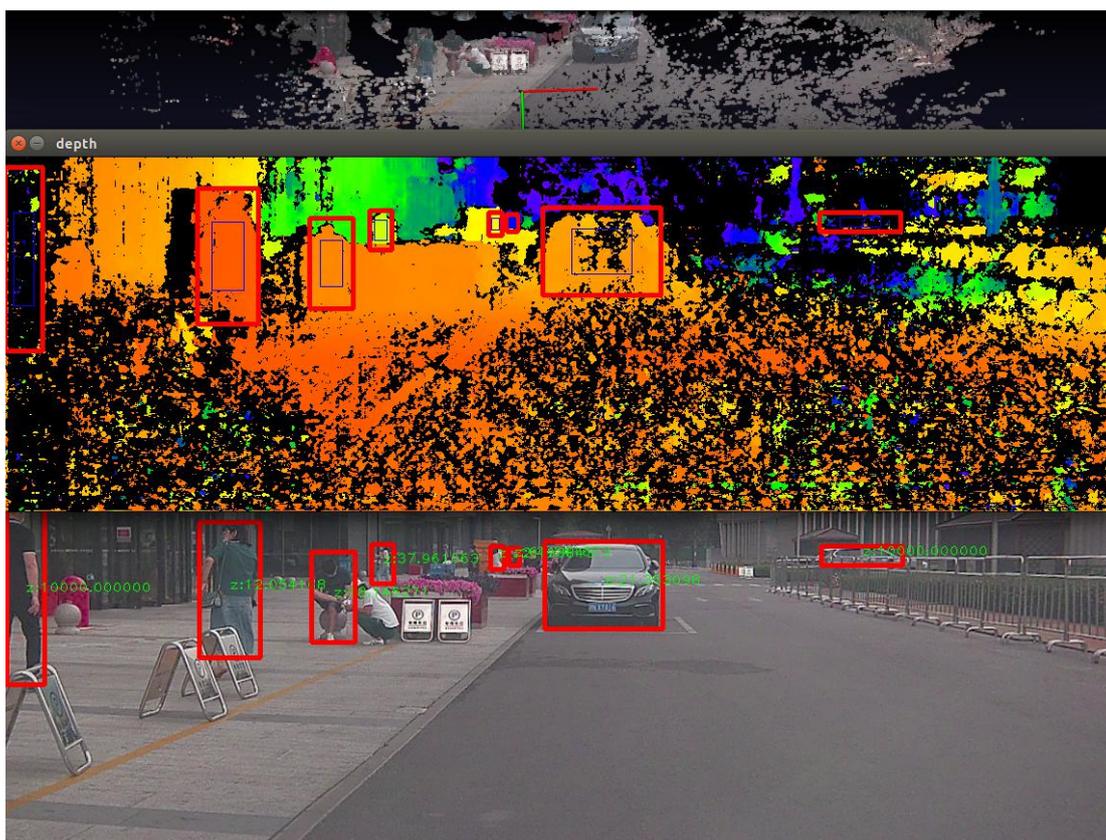
交通标注牌的检测与地面标线采用相同的神经网络对交通标志牌进行检测,详细请参考地面标线算法实现。在部署到 NVIDIA AGX xavier 上,在满载,单独运行检测模型时帧率可达到 50fps 以上。

五、障碍物与深度信息融合

为了获取障碍物在三维空间的信息,需要将 image 上检测到的障碍物映射到深度图上,这样就可以根据深度信息获得障碍物的三维空间信息。

双目算法和障碍物检测算法运行耗时不一致,想要将双目算法得到的深度图与障碍物检测的目标在时间上配准,本工程还实现了对双目算法和障碍物检测的缓存机制和同步机制,缓存机制采用数据结构队列对带处理的图像进行缓存,同步机制才用匹配时间最近的方法。

为了让障碍物检测的 bbox 与深度图对齐,本工程将双目的左侧 image 做一次极线几何后再将其喂给深度学习网络推理。最后将 image 检测到了 bbox 直接映射到深度图,如下图所示:



通过截取 bbox 中心点的 $1/2$ 作为计算障碍物深度信息的 ROI, 通过求 ROI 的均

值作为障碍物的三维空间位置。

六、工程编译与运行

工程依赖 : glog , yaml-cpp, cuda, tensorrt, opencv, protobuf, boost

工编编译 :

Build

1. cd source dir
2. mkdir build & cmake .. & make -j12

运行工程 :

Run

3. cd source dir
4. sh run.sh

配置工程 : 工程需要的配置选项都在 prams 目录下的 yaml 文件。主要的配置文件是

front_vision.yaml 文件 :

```
obstacle_detector_config: "obstacle_detection.yaml"
mark_ground_detector_config: "mark_ground_detection.yaml"
cctsd_db_detector_config: "cctsd_db_detection.yaml"
stereo_vision_config: "stereo.yaml"
left_camera_params: "left.yaml"
used_loop_fetch: true
mono: false
crop_top: 200
crop_bottom: 570
hz: 30.0
vis_mark_ground: false
vis_pointcloud: true
vis_obstacle: false
fusion_depth_and_obstacle: true
topic_obstacle: "vision_obstacles"
topic_mark_ground: "vision_mark_ground"
topic_cctsd_db: "vision_cctsd_db"
```

默认情况是全部打开状态。输出的结果—共享内存的方式发出 ,共享内存的名字依次为 :

障碍物 : "vision_obstacles" , 地面标线:"vision_mark_ground" , 交通标志牌 :

"vision_cctsd". 发送的格式如 proto 文件定义 :

```
package iv.vision;
message Bbox3D
{
    optional string category = 1;
    optional uint64 class_id = 2;
    optional double x = 3;
    optional double y = 4;
    optional double z = 5;
    /*
    `----->x
    |
    |
    |
    v
    y
    */
    optional uint64 top_left_x = 6;
    optional uint64 top_left_y = 7;
    optional uint64 bottom_right_x = 8;
    optional uint64 bottom_right_y = 9;
}

message ObstacleInfo
{
    repeated Bbox3D bbox_3d = 1;
    optional uint64 time = 2;
}
```

参考文献

- [1]. Zhang Z . Determining the Epipolar Geometry and its Uncertainty: A Review[J]. International Journal of Computer Vision, 1998, 27(2):161-195.
- [2]. Heiko Hirschmüller. Hirschmüller, H.: Stereo processing by semiglobal matching and mutual information. IEEE PAMI 30(2), 328-341[J]. IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008, 30(2):328-341.
- [3]. Zhou X , Wang D , Krhenbühl, Philipp. Objects as Points[J]. 2019.
- [4]. Yu F , Wang D , Shelhamer E , et al. Deep Layer Aggregation[J]. arXiv, 2017.

- [5]. Cisse M , Dauphin Y N , et al. mixup: Beyond Empirical Risk Minimization[J]. 2017.
- [6]. Hendrycks D , Mu N , Cubuk E D , et al. AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty[J]. 2019.
- [7]. Howard A , Sandler M , Chu G , et al. Searching for MobileNetV3[J]. 2019.
- [8]. Redmon J , Farhadi A . YOLOv3: An Incremental Improvement[J]. 2018.