



中国研究生创新实践系列大赛
“华为杯”第十六届中国研究生
数学建模竞赛

学 校 清华大学

参赛队号 C19100030017

1.孙万虎

队员姓名 2.易梦云

3.艾浩

中国研究生创新实践系列大赛

“华为杯”第十六届中国研究生

数学建模竞赛

题 目 基于单目视觉的视频图像信息分析

摘 要：

视频图像信息分析是计算机视觉研究中非常重要的一部分，它的目的是依据已知的图像信息，通过数学模型提取出用户感兴趣的物体信息，如物体的大小、距离、速度等。本文研究了在单目视觉下的单幅图像、平面视频、立体视频下的距离信息分析技术，通过射影几何、透视投影、灭点检测、相机位姿估计、RANSAC、SafeUAV 等算法，完成对典型应用场景下视频图像信息分析。

针对任务一：单幅图像下的距离信息分析。单幅图像可以通过求解灭点利用透视投影的交比不变性得到结果，也可以通过灭点求解相机内参，得到**单应矩阵**进行求解。因此，灭点的求解对于单幅图像的距离分析至关重要。本文分别采用了基于**霍夫变换**的灭点检测、基于**Gabor 小波变换**的灭点检测和**线性回归法**进行单幅图像下的灭点求解，并选取出最优的检测结果。基于求解出的灭点，本文分别利用了两种不同的方法进行求解并进行交叉验证。其中，直接几何法利用射影变换中的**交比不变性**直接求解物体的距离，而间接几何法可以通过**EPnP 相机位姿估计**算法求解单应矩阵，从而得到像素坐标系与真实坐标系之间的对应关系。

针对任务二：平面视频下（高速上拍摄）的距离信息分析。相机在拍摄的过程中难免会存在一些晃动或者偏移，这也就造成了视频中的抖动。为了去除视频抖动影响，根据原视频产生经过运动补偿过的视频序列，我们采用随机抽样一致 **RANSAC** 算法，利用帧间的信息来计算帧的全局运动参数，进而去除视频的抖动，从而得到运动稳定的视频图片序列。与任务一不同，任务二中的视频中含有的真实尺度信息较少，很难得到足够多的像素坐标与真实坐标对应关系进而求解单应矩阵。因此我们通过寻找几何约束对实际问题进行建模，利用**交比不变性**得到对物体距离和尺度的估计。

针对任务三：平面视频下（高铁上拍摄）的距离信息分析。与任务二相同，我们用 **RANSAC** 算法消除视频中的抖动。高铁上拍摄的视频由于缺乏明显的尺度信息，通过广泛的资料查找，我们挖掘出了视频中隐含的部分物体的真实尺度。进而利用地平线上灭点的性质，通过几何约束对实际问题进行建模，利用**交比不变性**得到对物体距离和尺度的估计。

针对任务四：立体视频下的信息分析，本文采取了基于**深度学习**的无人机飞行高度估计算法 **SafeUAV** 对视频中的深度和无人机飞行高度进行估计。**SafeUAV** 算法对无人机拍摄视频中的深度信息进行预测，进而得到图像中各个像素点的深度图。基于 **SafeUAV** 算法

提供的深度预测，以及灭点求解出相机内参，我们进一步通过层次分解和射影几何方法求解单应矩阵，将等高度的单位距离网格可视化，从而得到了立体视频下的物体距离和尺度估计。

最后，本文对所建立的模型给出了评价和改进的方向。

关键词：交比不变性，相机位姿估计，霍夫变换，单应矩阵，RANSAC 算法，深度学习，SafeUAV 算法

目录

1 问题重述	4
1.1 问题背景	4
1.2 问题提出	4
2 问题分析	4
3 模型假设	5
4 符号说明	5
5 模型的建立与求解	6
5.1 任务一：单幅图像下的距离信息分析	6
5.1.1 问题分析	6
5.1.2 直接几何量测：透视投影法	6
5.1.2.1 交比不变性	6
5.1.2.2 灭点检测算法结果	9
5.1.2.3 图片一求解	10
5.1.2.3 图片二求解	12
5.1.2.4 图片三求解	14
5.1.2.4 图片四求解	16
5.1.3 间接几何量测：单应矩阵求解	17
5.1.4 模型求解结果分析	21
5.2 任务二：平面视频下的距离信息分析	21
5.2.1 问题分析	21
5.2.2 汽车运动模型建立	21
5.2.3 RANSAC 视频去抖动	21
5.2.4 模型求解	23
5.3 任务三：平面视频下的距离信息分析	25
5.3.1 问题分析	25
5.3.2 模型建立	25
5.3.3 模型求解结果	28
5.4 任务四：立体视频下的距离信息分析	28
5.4.1 问题分析	28
5.4.2 SafeUAV 算法求解深度	28
5.4.3 单应矩阵求解	30
5.4.4 模型求解结果	30
6 模型的评价与改进	32
6.1 模型的评价	32
6.2 模型的改进	32
7 参考文献	33
8 附录	33
相关程序代码	33

1 问题重述

1.1 问题背景

从图像或视频序列中提取物体的大小、距离、速度等信息是视觉情报分析工作的重要内容之一，在移动机器人、无人驾驶、计算机视觉、无人机侦察等领域，更是存在着大量的应用需求。随着双目、多目视觉系统和 3D 视觉系统的蓬勃发展，视频图像信息分析中的深度信息变容易被感知。但在条件有限的情况下，我们只能获取普通的图像或视频，对其进行信息分析需要综合考虑各种因素。

得益于计算机视觉领域的蓬勃发展，单目视觉系统的几何信息分析已有成熟的方法和算法。对于单幅图像而言，由于成像过程中深度信息的缺失，其几何信息分析需要一定的先验度量信息。基于先验度量，可以通过直接几何法、间接几何法实现几何量测。对于多幅图像或视频而言，可以通过立体视觉原理恢复场景的三维结构，实现几何信息分析。

1.2 问题提出

问题一：实验对象是单幅图片。构造数学模型，对其图片中的物体距离、高度以及拍摄者的位置进行求解。

问题二：实验对象是高速行驶的车上拍摄的后视镜视频，设计有效的模型求解视频中车辆的相对距离、相对行驶速度。

问题三：实验对象是高铁上拍摄的视频，设计有效的模型求解视频中桥面距水面的高度、距高铁轨道的距离以及水面宽度，估算拍摄时高铁的行驶速度。

问题四：实验对象是无人机航拍视频，设计有效的模型求解建筑物的高度、面积，并测算无人机的飞行高度和速度。

2 问题分析

本文研究了视频中静态背景和动态背景下运动目标检测技术，当摄像机发生晃动或偏移下的视频去抖动算法，选出包含显著前景目标的视频帧标号的方法，多摄像机不同视角拍摄视频的前景目标提取以及视频异常事件检测算法。

对于任务一：单幅图像下的距离信息分析。图像视频序列中的距离测量可以通过直接几何法和间接几何法进行求解。直接几何法利用透视投影的交比不变性，通过求解出灭点和灭线从而计算出结果。间接几何法通过求解单应矩阵得到真实坐标系与像素坐标系的变换关系，从而计算出结果。对于单幅图像而言，可以通过一组相互垂直方向的灭点求解相机内参，再依据相机内参求解单应矩阵。因此，灭点的求解对于单幅图像的距离分析至关重要。其中，现有的灭点检测算法有基于霍夫变换的灭点检测、基于 Gabor 小波变换的灭点检测和几何法。本文分别采用上述算法进行单幅图像下的灭点求解，并选取出最优的检测结果。通过求解出的灭点，利用直接几何法和间接几何法两种方法对图像中的距离信息分别进行求解，并对实验结果作对比及分析。

针对任务二：平面视频下（高速上拍摄）的距离信息分析。相机在移动拍摄的过程中难免会存在一些晃动或者偏移，这也就造成了视频中的相机抖动。视频去抖动的过程就是要根据原视频产生经过运动补偿过的视频序列，这样就可以将不需要的抖动除去。这种方法关键就在于怎样利用帧间的信息来得出当前帧的全局运动参数。本文采用 RANSAC 算法利用帧间的信息来计算帧的全局运动参数，进而去除视频的抖动，从而得到稳定的图片拍摄序列。再通过几何约束和交比不变性对图像中的距离信息进行求解。

针对任务三：平面视频下（高铁上拍摄）的距离信息分析。同样，我们采用 RANSAC 算法消除任务三视频中的抖动影响。但高铁上的拍摄视频由于没有很好的尺度参照物，因此，本文利用灭点和地平线性质通过几何约束对大桥和河面进行物理建模，并选取其中几帧进行运算，最终得到一个的结果。

针对任务四：立体视频下的距离分析。立体视频下由于缺乏深度信息，因此需要对深度进行建模求解。我们采用了基于深度学习的 SafeUAV 算法对深度信息进行估计，得到了建筑物的高度信息，并给出了视频对应的深度图像。通过估计出的建筑物高度，我们对，最终求解出。其次，我们利用近似同一地点在不同高度层上的标志物，计算基于多层的不同视角间计算单应性矩阵，然后通过单应性变换获得目标在不同高度层的定位信息，

最后，本文对所建立的模型给出了评价和改进的方向。

3 模型假设

假设 1：视频和图像中出现的普通小型轿车车长为标准车长 4.3m，轴距为标准轴距 2.6 米。

假设 2：视频和图像中双向两车道公路宽度符合标准，即每条机动车道宽度为 3.75 米。

假设 3：任务 1 图 2 中可跨越对向车行道分界线（黄色虚线）符合国家标准 [1]，即线长 4 米，间隔 6 米。

假设 4：任务 1 图 3 中的车道中人行横道预告标识线（白色菱形图案）符合国家标准，即对角线长度分别为 3 米和 1.5 米。

假设 5：任务 2 视频中的高速公路上的车行道分界线（白色虚线）符合国家标准，即线长 6 米，间隔 9 米。

假设 6：任务 2 视频中的高速公路上的所有车辆均处于匀速行驶状态。

假设 7：任务 3 视频中的高铁处于匀速行驶状态。

假设 8：任务 3 中高铁行驶铁轨与水平面平行，且相机的拍摄方向与铁轨方向垂直。

假设 9：任务 3 视频中移动通信基站单管塔严格按照设计图纸建设，符合 30m 灯杆景观塔设计标准，即铁塔高度为 30m，塔尖高 5 米，总高 35 米。

假设 10：任务 4 视频中拍摄用无人机在近似固定高度匀速绕圆周飞行。

4 符号说明

符号	说明
V	灭点
$\frac{H}{R}$	交比
f_x	相机在 x 轴上的尺度因子
f_y	相机在 y 轴上的尺度因子
u_0	相机主点在 x 轴上的像素坐标
v_0	相机主点在 y 轴上的像素坐标
M_1	相机内参矩阵
M_2	相机外参矩阵
R	相机旋转矩阵
T	相机平移矩阵

v	速度
s	距离
N	帧
R	相机帧率

5 模型的建立与求解

5.1 任务一：单幅图像下的距离信息分析

5.1.1 问题分析

距离信息分析就是利用单幅图像中图形和我们有限的已知信息，将我们所感兴趣的物体相对位置信息从图像中提取出来。对于单幅图像而言，由于成像过程中深度信息的缺失，其几何信息分析需要一定的先验度量信息。因此，拍摄图像中通常包括水平面平行线（道路、地砖）、竖直面平行线（楼房），已知信息通常包括车辆轴距、道宽、标线长度等等。

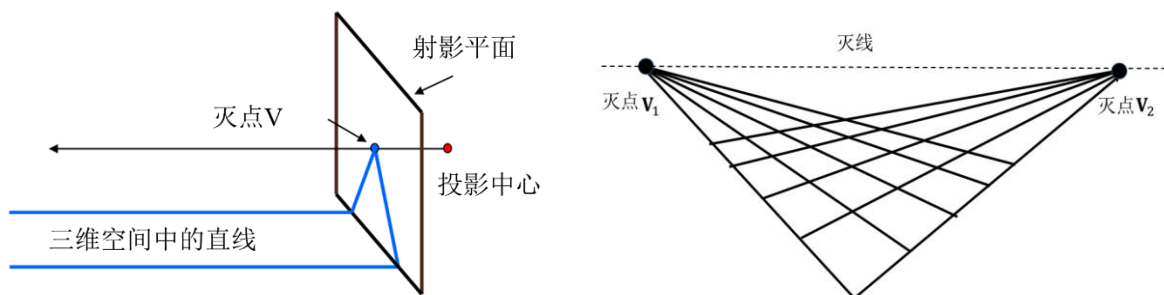
图像视频序列中的距离测量可以通过直接几何法和间接几何法进行求解。直接几何法利用透视投影的交比不变性，通过求解出灭点和灭线从而计算出结果。间接几何法通过求解单应矩阵得到真实坐标系与像素坐标系的变换关系，从而计算出结果。本文分别采用直接几何量测法和间接几何量测法进行单幅图像下的距离信息分析实验，并对实验结果进行对比及分析。

5.1.2 直接几何量测：透视投影法

5.1.2.1 交比不变性

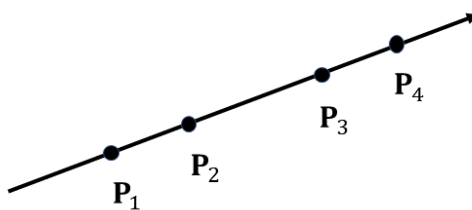
根据射影几何原理，将三维空间中的点投影到特定的射影平面上时，三维空间中的平行直线在射影平面上会相交在一个射影点上，即为射影平面中的灭点（Vanishing Points）。射影平面上的灭点对应三维空间上的无穷远点。

灭点只和直线的方向相关，三维空间中一组平行的直线可以确定射影平面中的一个灭点。一个平面上所有平行线组确定的灭点共线，称为灭线，特别的，地平面对应的灭线即地平线。



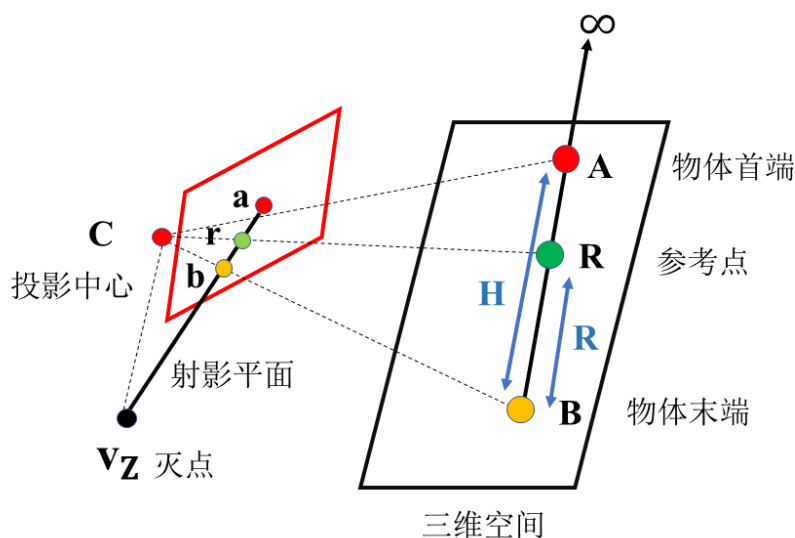
根据射影变换和透视投影的性质，图像中四个共线点的交比（Cross Ratio）与真实世界的交比相等。因此，在真实场景下，可以通过交比的不变性来确定图像中的距离信息。

在图像中：



$$\text{Cross Ratio} = \frac{\|P_3 - P_1\| \|P_4 - P_2\|}{\|P_3 - P_2\| \|P_4 - P_1\|}$$

在三维空间中：



$$\text{Cross Ratio} = \frac{\|A - B\| \|\infty - R\|}{\|R - B\| \|\infty - A\|} = \frac{H}{R}$$

$$\text{Cross Ratio} = \frac{\|a - b\| \|v_z - r\|}{\|r - b\| \|v_z - a\|} = \frac{H}{R}$$

因此，根据交比不变性，可以通过求解图像中的灭点来求解物体的长度、距离信息。

5.1.2.2 常见的灭点检测算法

灭点检测算法[2]的根本在于对三维空间平行线段在图像中的投影线段的检测。实现平行直线段从背景中清晰、完整及快速的提取，并通过线段求解灭点是灭点检测算法的目的。灭点的正确检测对后续的灭线求解以及图像物体距离计算有重要的作用。

灭点检测算法主要包括基于霍夫变换的灭点检测法、基于纹理信息的灭点滤波法，和线性回归法。

(1) 基于霍夫变换的灭点检测

基于霍夫变换的灭点检测首先检测图像中的直线。对于每一个可能的直线对，计算直线交点的方向。并且他们的角度能够积累到一个参量中。通过选择包含最高数量的参量来获得占据主导的灭点。但是，霍夫变换对于角度参量量化的水平十分的敏感，因此结果中可能会出现多重的检测。

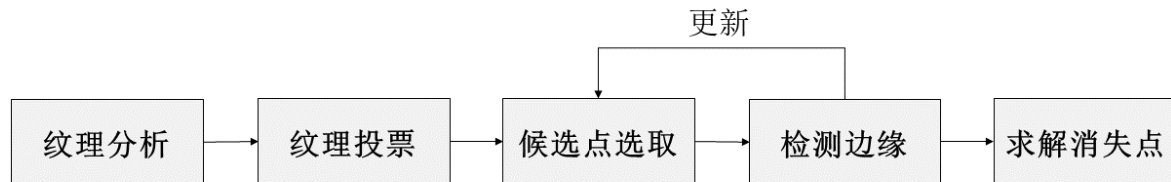


霍夫变换通过计算霍夫矩阵，把使每个像素坐标点经过变换转换得到极坐标系的极半

径和极角，并对转换后的参量进行累计。当一个峰值点出现时候，说明有直线存在。检测出图像中的直线后通过选择包含最高数量的参量获得灭点。

(2) 基于 Gabor 小波的道路消失点检测

针对公路上的灭点检测，可以通过基于 Gabor 小波的道路消失点检测算法[3]进行检测。



首先进行纹理分析，使用 Gabor 小波在多个尺度上进行纹理分析，对纹理不显著光滑区域进行舍弃。之后计算每个点与纹理信息的关系和得分，通过纹理投票的方法，选取候选点。根据候选点寻找比较显著的边缘，然后依据这个边缘对候选点进行更新。对于更新后的新候选点，再寻找另一个边缘，从而得到道路的直线边缘以及道路消失点。

(3) 线性回归法

在需要处理的图像数量有限的情况下，可以通过手工标注穷举出三维空间中平行直线上的点，根据点求解回归方程，确定直线在空间中的表达式，求出平行直线组的交点，即为灭点。

对于图 1 到图 4，我们选取了三维空间中的多个平行直线组，如图 1 和图四中的地砖连线，图 2、图 3 中的线作为平行线组。

根据选取的像素点坐标，将同一直线上的像素点进行线性拟合，得到各条直线的函数表达式。直线组的交点即为灭点。

为了减少手工标注的误差，我们通过数据拟合出直线的表达式。以图 1 为例，我们选取了 50 个像素点对直线 L1-L11 进行线性回归，得到的结果如下：

$$y = p_1 \cdot x + p_2$$

(x,y)	L1	L2	L3	L4	L5	p1	p2
L6	(783.5,1369.5)	(861.5,1393.5)	(951.5,1415.5)	(1261.5,1499.5)	-	0.274	1156
L7	(861.5,1393.5)	-	(1085.5,1281.5)	(1355.5,1337.5)	(1810.5,1437)	0.214	1049
L8	(951.5,1415.5)	(1115.5,1171.5)	-	(1425.5,1221.5)	(1815.5,1293.1)	0.1718	978
L9	-	-	(1263.5,1105.5)	(1475.5,1137.5)	(1815.5,1188.9)	0.1457	922.6
L10	(1147.5,1473.5)	(1263.5,1039.5)	(1319.5,1047.5)	(1517.5,1071.5)	-	0.1226	885.4
L11	(1261.5,1499.5)	(1311.5,995.5)	(1367.5,1001.5)	(1547.5,1019.5)	-	0.09924	863.9
p1	0.7837	-0.8833	-0.9961	-1.672	-43.49		
p2	1985	2155	2363	3607	80170		

注：表中的 - 表示像素点由于遮挡等原因无法准确选取，我们选取了其他的像素点作为补充。为了直观显示，部分像素坐标点未在表中给出。

	p1**	p2**	SSE	R-Square	RMSE
L1	(-0.792,0.7754)	(1976, 1993)	3.23	1	1.038
L2	(-0.9025,0.8642)	(2133, 2177)	4.861	0.9999	1.559
L3	(-1.005, -0.987)	(2352, 2374)	2.962	1	0.9936
L4	(-1.7, -1.644)	(3567, 3647)	23.08	0.9999	2.042
L5	(-48.42, -38.56)	(71200, 89130)	516.2	0.9986	16.07
L6	(0.2539,0.2942)	(1135, 1176)	19.1	0.9984	2.523
L7	(0.2057,0.2223)	(1037, 1060)	48	0.9985	2.829
L8	(0.1639,0.1797)	(966.6,989.5)	22.85	0.9984	2.138

L9	(0.1371,0.1542)	(909.3,935.9)	8.077	0.9982	1.421
L10	(0.1167,0.1284)	(876.6,894.2)	11.48	0.9977	1.383
L11	(0.09048,0.108)	(850.5,877.3)	21.36	0.9923	1.887

其中，L1-L5 为三维空间中的平行直线组，L6-L11 为与该平行直线组相互垂直的另一个平行直线组，通过求解每一个直线组的交点，可以得到灭点坐标。

由于穷举标注和数据拟合过程中存在误差，每个直线组并非只有一个交点，我们将灭点坐标与直线组中每条直线的距离作为误差函数，通过最小化误差函数进行灭点求解。

设灭点 v_0 的坐标 (x_0, y_0)













直线方程：





$$y_i = p_{i1} \cdot x_i + p_{i2}$$

误差函数：

$$\sum_i \frac{p_{i1} \cdot x_i + p_{i2} - y_i}{\sqrt{p_{i1}^2 + 1}}$$

5.1.2.2 灭点检测算法结果

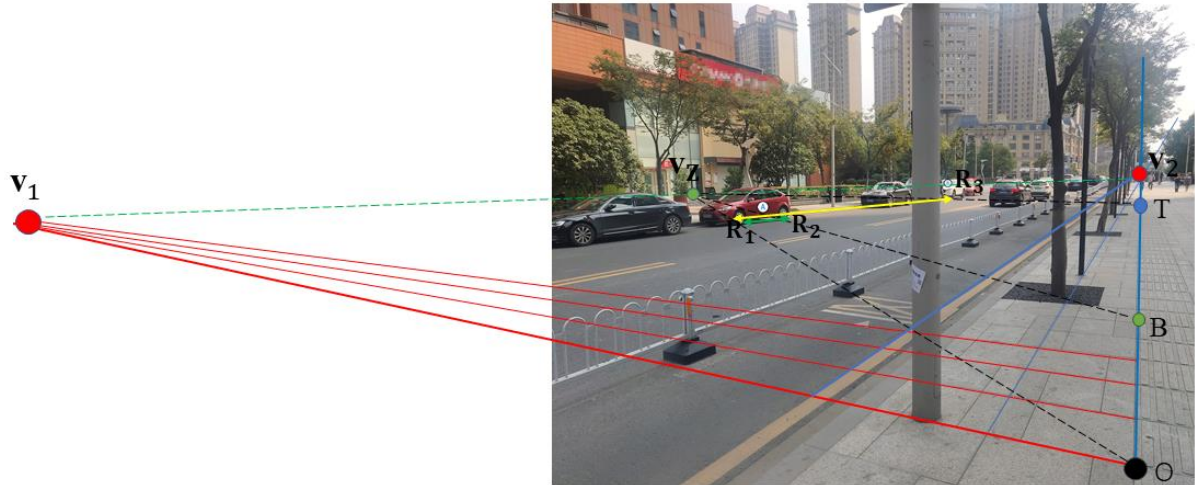
	原图	霍夫变换法	Gabor 小波法	线性回归法
图 1				
灭点		(1836.5,531.8)	(1816.4,527.3)	(1829.6,539.2)
				
灭点		(416.8,800.6)	(337.5,795)	(332.1,786.4)
				
灭点		求解错误	求解错误	(-1953,409)

				
灭点		(663.6,1271.9)	求解错误	(664.9,1271.1)

通过对比分析可以看出，基于霍夫变换的灭点检测依赖于检测出的直线对的好坏，容易出现错误判断。基于 Gabor 小波变换的道路消失点检测只能检测出在图像中的灭点，对于图像外的消失点不能检测出来。基于线性回归法的灭点检测依赖于对像素点的手工标注，存在一定的手工误差，但结果相对来说更准确。

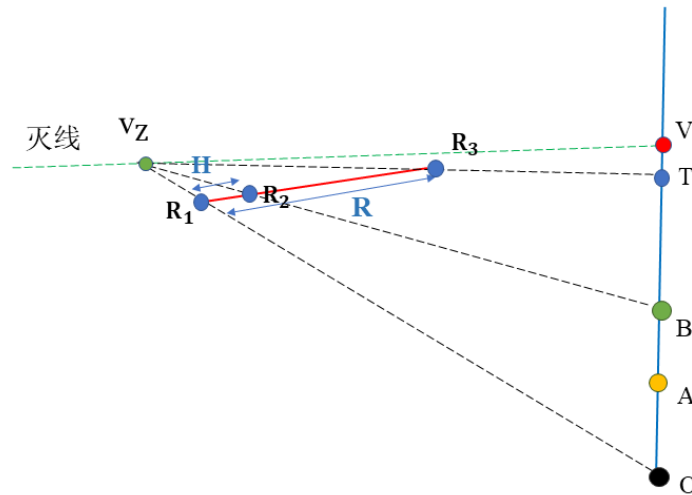
5.1.2.3 图片一求解

(1) 红色车辆 A 车头和白色车辆 B 车头之间的距离



基于霍夫变换法、Gabor 小波法和穷举法，我们得到了三组灭点，根据分析结果，我们选取 V_1, V_2 作为灭点并构造出灭线（图中绿色虚线）。

简化图如下所示：



其中，点 R_1 为红色车辆前轮位置，点 R_3 为白色车辆前轮位置。由于两车车型相似，因此可认为两车车头之间的距离等于两车前轮之间的距离。即为点 R_1 和 R_3 在空间中的直线距离。

由于两辆车车型相差不大，不失一般性的认为两车前车轮之间的距离即为车头之间的距离。基于假设 1，红色车辆 A 的轴距为 2600mm。从灭线上取一个灭点 V_z ，连接灭点与红色车辆的轮胎中心点，与蓝色直线交于 O 点和 B 点。

图中点的像素坐标分别为：

点	O	A	V	B	T
像素坐标	(1813,1439)	(1818,1187)	(1829,539)	(1821,985)	(1829,618)

根据灭线性质，灭线上的所有点都为灭点，从灭点连出的任意两条直线平行。则有：

$$R_1R_2 \parallel OB, V_zO \parallel V_zB, R_1O \parallel R_2B$$

根据平行四边形原理， $\|O-B\|=\|R_1-R_2\|=2600\text{mm}$ ， $\|R_1-R_3\|=\|O-T\|$

又根据交比性质，图像中的交比与三维空间中的交比相等：

$$\frac{\|o-t\|\|v-b\|}{\|b-t\|\|v-o\|} = \frac{\|O-T\|\|\infty-B\|}{\|B-T\|\|\infty-O\|} = \frac{H}{R}$$

$$\|O-T\| = (\|O-T\| - \|O-B\|) \frac{\|o-t\|\|v-b\|}{\|b-t\|\|v-o\|}$$

计算得出 OT 两点的真实距离为 26.544m，也即两车车头之间的距离为 26.544m

为了验证计算结果的准确性，我们利用交比不变性对 OA 两点间的地砖长度进行求解。依据交比不变性：

$$\frac{\|o-b\|\|v-a\|}{\|a-b\|\|v-o\|} = \frac{\|O-B\|\|\infty-A\|}{\|A-B\|\|\infty-O\|} = \frac{H}{R}$$

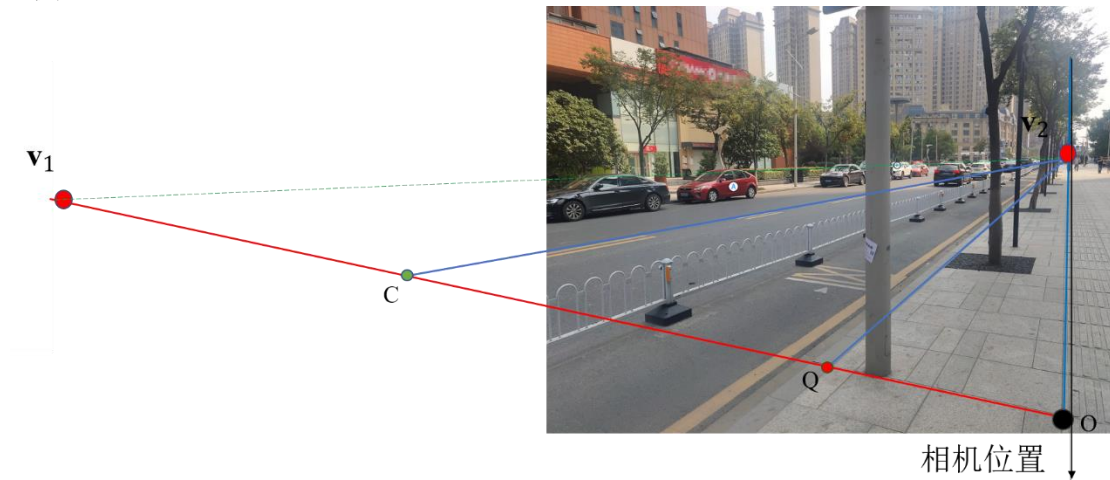
由上文知 $\|O-B\|=\|R_1-R_2\|=2600\text{mm}$ ， $\|A-B\|=\|O-B\|-\|O-A\|$ =轴距-地砖长度

根据交比性质，算出 AB 之间的地砖长度为 99.3cm，结合日常生活经验，真实的地砖长度大致为 100cm，与结果接近。因此，我们认为上文中求解的结果较为可信。

(2) 拍照者距马路左侧边界的距离

如图所示，相机位置几乎与 O 点重合，偏差在 5cm 左右。因此，我们可以将拍照

者离马路左侧的距离近似为 O 点到马路左侧的距离。即为马路宽度与相机位置到马路右侧距离的和。



利用交比不变性：

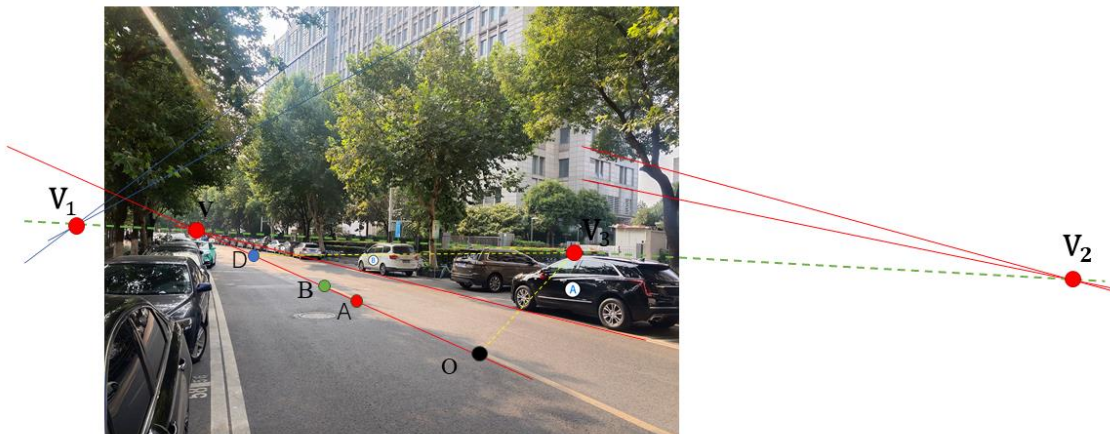
$$\frac{\|o-c\| \|v_1-q\|}{\|q-c\| \|v_1-o\|} = \frac{\|O-C\| \|\infty-Q\|}{\|Q-C\| \|\infty-O\|} = \frac{H}{R}$$

求出 OC 为 9m。由于地砖大小为 1m，可以得到与人行道边沿距离 OQ=2.25m，因此相机距离边沿距离为 $2 \times (OC - OQ) + OQ = 15.75m$ 。

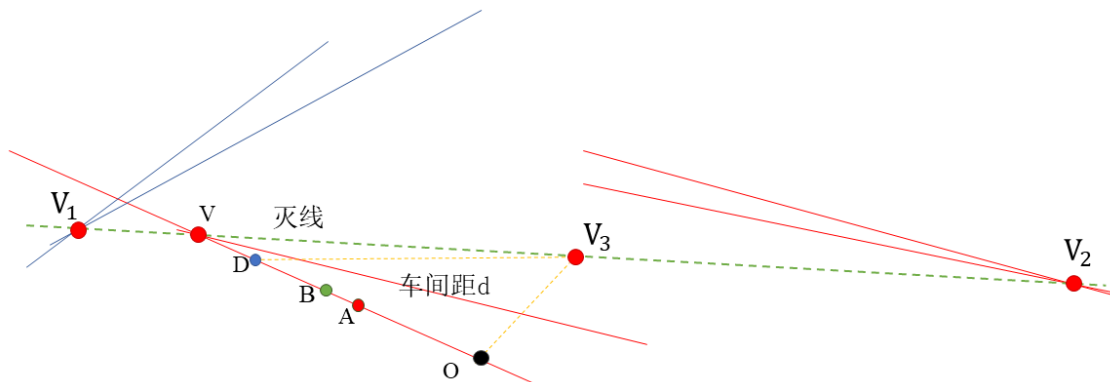
5.1.2.3 图片二求解

(1) 黑色车辆 A 车头和灰色车辆 C 车尾之间的距离

基于霍夫变换法、Gabor 小波法和穷举法，我们得到了三组灭点，根据分析结果，我们选取作为灭点并构造出灭线（图中绿色虚线）。



简化图如下所示：



我们将 O 点作为原点, 从 O 点做直线连接黑色车辆的车前轮, 并与灭线交于点 V_3 , 连接灭点 V_3 与灰色车辆的车后轮, 并交与直线 OV 交于 D 点。A、B 为参考坐标点。

图中点的像素坐标分别为:

点	O	A	B	V	D
像素坐标	(1293,1207)	(883,1028)	(771,979)	(332,786)	(493,856)

基于假设 3, 标准的可跨越对向车行道分界线长度为 4m, 间隔为 6m。即 $\|\mathbf{O}-\mathbf{B}\|=1000$ cm $\|\mathbf{A}-\mathbf{B}\|=400$ cm。

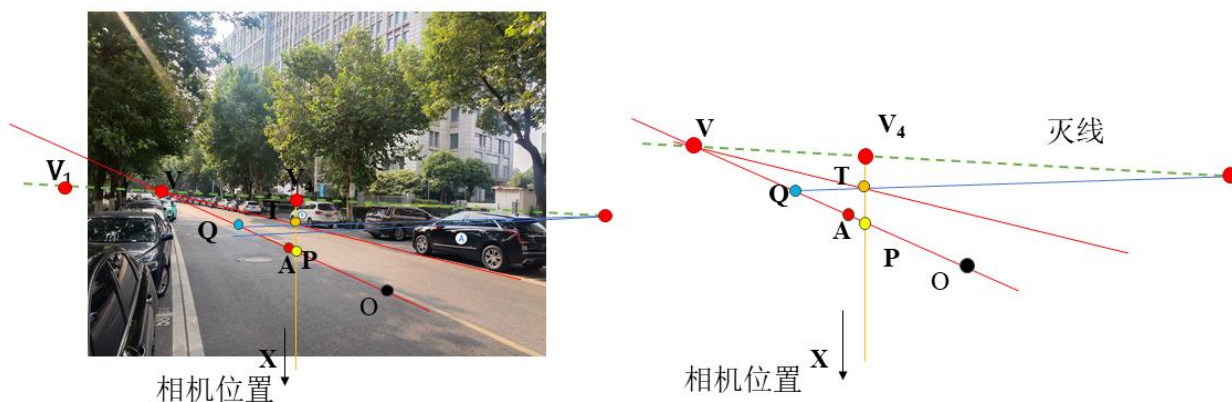
根据交比性质，图像中的交比与三维空间中的交比相等：

$$\frac{\|o-d\|\|v-a\|}{\|a-d\|\|v-o\|} = \frac{\|\mathbf{O}-\mathbf{D}\|\|\infty-\mathbf{A}\|}{\|\mathbf{A}-\mathbf{D}\|\|\infty-\mathbf{O}\|} = \frac{H}{R}$$

$$\|\mathbf{O}-\mathbf{D}\| = (\|\mathbf{O}-\mathbf{D}\| - \|\mathbf{O}-\mathbf{A}\|) \frac{\|o-d\|\|v-a\|}{\|a-d\|\|v-o\|}$$

因此，得到 OD 点对应车长为 40.28m。由于题目所求为黑色车辆车头和灰色车辆车尾之间的距离，还需要减去黑色车辆车头到车轮距离和灰色车尾到车轮距离。根据假设条件 1，普通家用轿车的车长为 4.3m，轴距为 2.6m，差为 1.7m。因此，最终得到的距离为： $40.28 - 1.7 = 38.58\text{m}$ 。

(2) 拍照者距白色车辆 B 车头的距离



垂直于照片底边的黄线即相机到 B 车连线，与 OV 交于点 P。马路右侧的花坛边线与直线 OV 垂直，利用灭线，找到白色车辆车轮 T 在 OV 线上的垂直投影 Q。根据直角三角形勾股定理，则有：

$$TQ \perp OV, TP^2 = PQ^2 + TQ^2$$

点	O	A	P	V	V ₄	T	Q
像素坐标	(1293,1207)	(883,1028)	(938,1052)	(332,786)	(968,829)	(938,921)	(642,923)

根据交比性质，图像中的交比与三维空间中的交比相等：

$$\frac{\|o-q\|\|v-a\|}{\|a-q\|\|v-o\|} = \frac{\|\mathbf{O}-\mathbf{Q}\|\|\infty-\mathbf{A}\|}{\|\mathbf{A}-\mathbf{Q}\|\|\infty-\mathbf{O}\|} = \frac{H}{R}$$

$$\|\mathbf{O}-\mathbf{Q}\| = (\|\mathbf{O}-\mathbf{Q}\| - \|\mathbf{O}-\mathbf{A}\|) \frac{\|o-q\|\|v-a\|}{\|a-q\|\|v-o\|}$$

因此，得到 OQ=16.87m

$$\frac{\|o-q\|\|v-p\|}{\|p-q\|\|v-o\|} = \frac{\|\mathbf{O}-\mathbf{Q}\|\|\infty-\mathbf{P}\|}{\|\mathbf{P}-\mathbf{Q}\|\|\infty-\mathbf{O}\|} = \frac{H}{R}$$

$$\|\mathbf{P}-\mathbf{Q}\| = (\|\mathbf{O}-\mathbf{Q}\| - \|\mathbf{O}-\mathbf{P}\|) \frac{\|p-q\|\|v-o\|}{\|o-q\|\|v-p\|}$$

因此，得到 PQ=12.14m

根据假设 2，车道宽度为 3.75m，又由于 $TQ \perp OV, PT^2 = PQ^2 + TQ^2$ ，故有：

$$\text{所以 } PT = \sqrt{12.14^2 + 3.75^2} = 12.71\text{m}$$

又有相机位置 X、点 P、T、V₄ 四点共线，利用交比不变性：

$$\frac{\|x-t\|\|v_4-p\|}{\|p-t\|\|v_4-x\|} = \frac{\|\mathbf{X}-\mathbf{T}\|\|\infty-\mathbf{P}\|}{\|\mathbf{P}-\mathbf{T}\|\|\infty-\mathbf{X}\|} = \frac{H}{R}$$

由于相机位置 X 在照片中的坐标相当于 (938, +∞)，即像素坐标系中相当于 y 处于无穷远的位置。因此，可做如下近似：

$$\|\mathbf{X}-\mathbf{T}\| = \|\mathbf{P}-\mathbf{T}\| \frac{\|x-t\|\|v_4-p\|}{\|p-t\|\|v_4-x\|} \approx \|\mathbf{P}-\mathbf{T}\| \frac{\|v_4-p\|}{\|p-t\|}$$

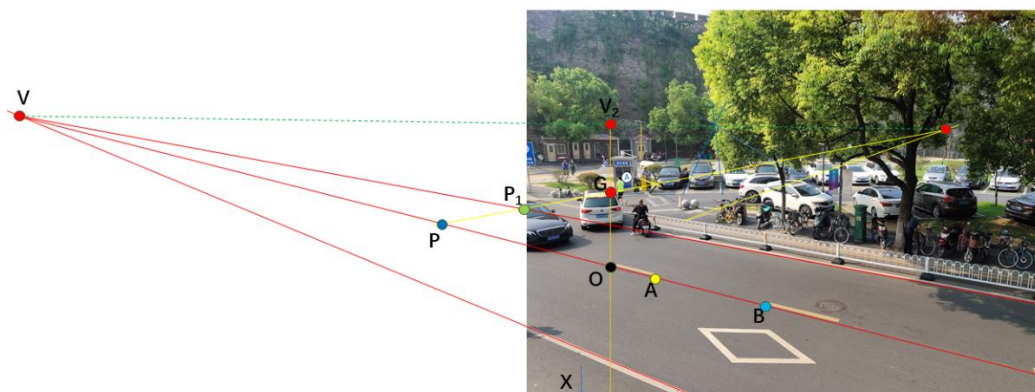
所以，拍照者距白色车辆的距离约 21.73m。

5.1.2.4 图片三求解

(1) 拍照者距离地面的高度

以远处门为参照，简易房门高度为 2 米，地面到灭线 138 pixel，门高 58pixel。由于与相机底片平行的平面上的点的相对距离关系不变，所以灭线相对地面高 $2 \times 138 / 58 = 4.76$ 米。由灭线性性质，相机高度应与灭线对应高度，即相机高度为 4.76 米。

(2) 拍照者距岗亭 A 的距离



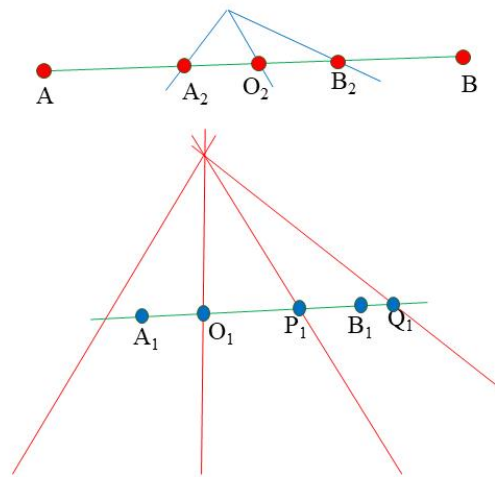
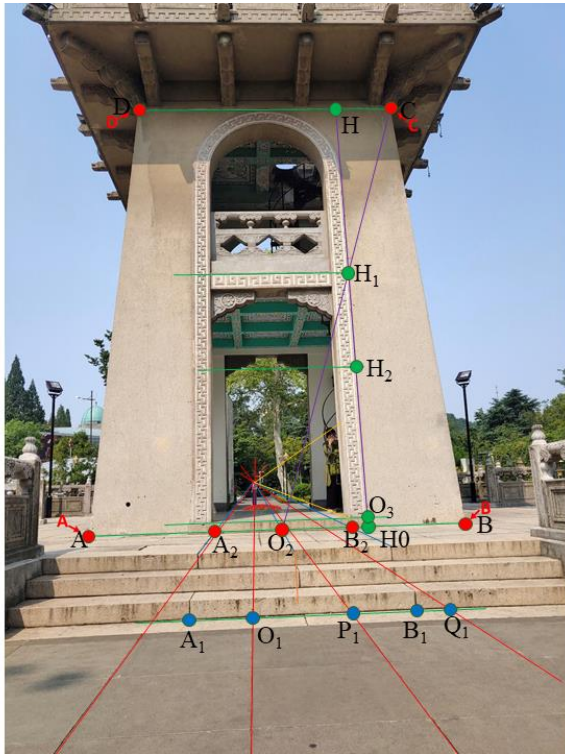
$$\|\mathbf{X}-\mathbf{G}\|=\|\mathbf{O}-\mathbf{G}\|\frac{\|v_2-o\|}{\|o-g\|}$$

所以， $XG=29.94m$ 。

G 与岗亭很近，可以认为相机到岗亭的水平距离为 28.94 米，考虑到相机高度，实际距离约 29.33 米。

5.1.2.4 图片四求解

(1) AB 的长度



点	A ₁	O ₁	P ₁	B ₁	Q ₁
像素坐标	(492,1643)	(660,1636)	(925,1623)	(1095,1613)	(1185,1611)
点	A	A ₂	O ₂	B ₂	B
像素坐标	(221,1420)	(558,1409)	(739,1402)	(923,1396)	(1227,1384)
点	H ₀	O ₃	H ₂	H ₁	H
像素坐标	(967,1395)	(965,1369)	(934,966)	(915,717)	(882,282)

首先求台阶砖长度，记 P_1B_1 实际长为 a ， O_1A_1 实际长为 b 。则有：

$$\frac{O_1B_1 \cdot P_1Q_1}{O_1Q_1 \cdot P_1B_1} = \frac{(80+a) \cdot 80}{160 \cdot a}$$

$$\frac{O_1Q_1 \cdot P_1A_1}{O_1P_1 \cdot Q_1A_1} = \frac{(80+b) \cdot 160}{(160+b) \cdot 80}$$

求出 $a=52.14cm$ ， $b=49.93cm$ ，因此，所以台阶砖块 $A_1B_1=a+b+80=182cm$ ，约为 182cm。

连线 AB，与平行直线组分别交于 A_2, O_2, B_2 ，其中 O_2 近似为 A_2 和 B_2 中点处，故认为： $A_2O_2=B_2O_2=91cm$

记 AA_2 实际长度 c ， BB_2 实际长度 d

$$\frac{AO_2 \cdot A_2B_2}{AB_2 \cdot A_2O_2} = \frac{(c+91) \cdot 182}{(c+182) \cdot 91}$$

$$\frac{A_2B_2 \cdot BO_2}{B_2O_2 \cdot A_2B} = \frac{182 \cdot (d+91)}{91 \cdot (d+182)}$$

求出 $c=173.5\text{cm}$, $d=147.1\text{cm}$, 因此所以 $AB=173.5+147.1+182=502.6\text{cm}$ 。

(2) AB 到 CD 的距离

根据假设 1, 图片中出现的人身高约 1.7 米, 观察到门框纹路有周期性, 我们认为每周等长。由于人的身高共约 8.5 个周期, 因此估算出门框纹路每周期高度为 20cm。

作辅助线, H_0H , 与 AB 交于 H_0 , 与 CD 交于 H 。已知 O_3H_2 共 13 格, H_2H_1 共 9 格
记 HH_1 实际长度为 h , O_3H_0 实际长度为 j , 有:

$$\frac{HH_2 \cdot H_1O_3}{HO_3 \cdot H_2H_2} = \frac{(h+180) \cdot 440}{(h+440) \cdot 180}$$

$$\frac{H_1O_3 \cdot H_2H_0}{H_2O_3 \cdot H_1H_0} = \frac{440 \cdot (260+j)}{(440+j) \cdot 260}$$

求出 $h=357.7\text{cm}$, $j=15.6\text{cm}$, 总高度 $=h+440+j=813.3\text{cm}$

(3) CD 的长度

做辅助线, 得到 CH_1 和 H_1O_2 正好共线, 因此有, 三角形 $O_2H_0H_1$ 和 CHH_1 相似。

假设 O_2H_0 长度为 c , 有:

$$\frac{A_2B_2 \cdot H_0O_2}{A_2H_0 \cdot O_2B_2} = \frac{182 \cdot c}{(c+91) \cdot 91}$$

得到 $c=112.5\text{cm}$ 。又有 BH_0 实际长度 125.6cm, 故所以我们可以得到 CH 实际长度 $c \cdot 357.7/455.6=88.3\text{cm}$

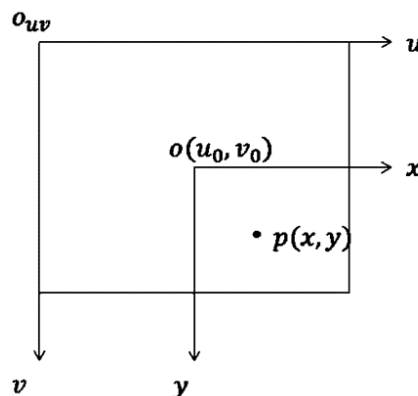
由于 $ABCD$ 为等腰梯形, HH_0 垂直于上下底边, 所以 $CD=AB-2(BH_0-CH)$, 因此得到 CD 长度 428cm。

5.1.3 间接几何量测：单应矩阵求解

5.1.3.1 模型建立

像素坐标系、图像坐标系、相机坐标系、世界坐标系。

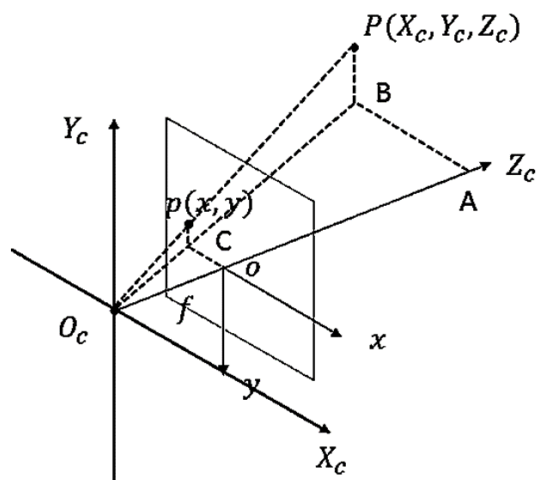
(1) 从像素坐标系到图像坐标系



$$\begin{cases} u = \frac{x}{dx} + u_0 \\ v = \frac{y}{dy} + v_0 \end{cases}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

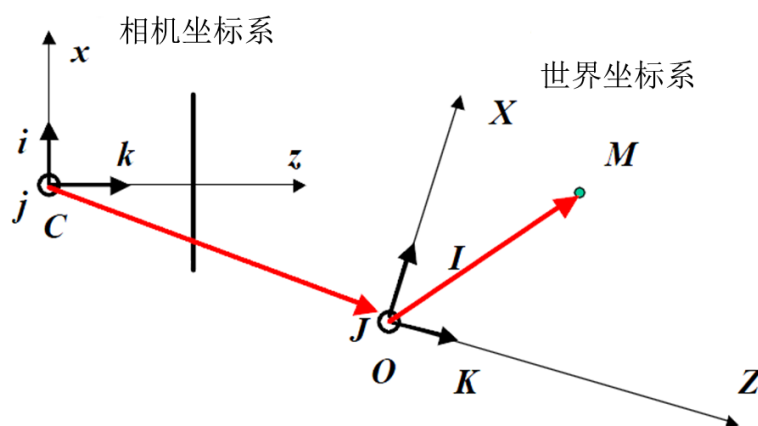
(2) 从图像坐标系到相机坐标系



$$x = f \frac{X_c}{Z_c}, \quad y = f \frac{Y_c}{Z_c}$$

$$Z_c \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

(3) 从相机坐标系到真实坐标系



$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}_3^T & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

因此，可以得到从世界坐标系到像素坐标系的转化关系：

$$\begin{aligned} Z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} &= \begin{bmatrix} \frac{1}{dx} & 0 & u_0 \\ 0 & \frac{1}{dy} & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ \vec{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \\ &= \begin{bmatrix} f_x & 0 & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ \vec{0} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \mathbf{M}_1 \mathbf{M}_2 \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \end{aligned}$$

其中， \mathbf{M}_1 相机内参矩阵， \mathbf{M}_2 为相机外参矩阵。

因此，只要求出相机的内参矩阵和外参矩阵，就能得出从像素坐标系和真实坐标系的对应关系。[4]

5.1.3.2 内参矩阵求解

一般情况下，相机内参矩阵包含 5 个内部参数，即 f_x, f_y, u_0, v_0, s 。[4]

$$\mathbf{M}_1 = \begin{bmatrix} f_x & s & u_0 & 0 \\ 0 & f_y & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

其中， f_x, f_y 分别为相机在坐标轴上的尺度因子； u_0, v_0 为相机的主点； s 为相机畸变因子。对于大多数相机而言，图像坐标轴相互垂直，且坐标轴上的尺度因子 f_x, f_y 相等，因此相机内参矩阵可以简化为三参数模型。不失一般性的，我们将相机的主点 u_0, v_0 取在像素坐标系的中心点，因此，相机内参矩阵可以简化为对 $f = f_x = f_y$ 的求解。

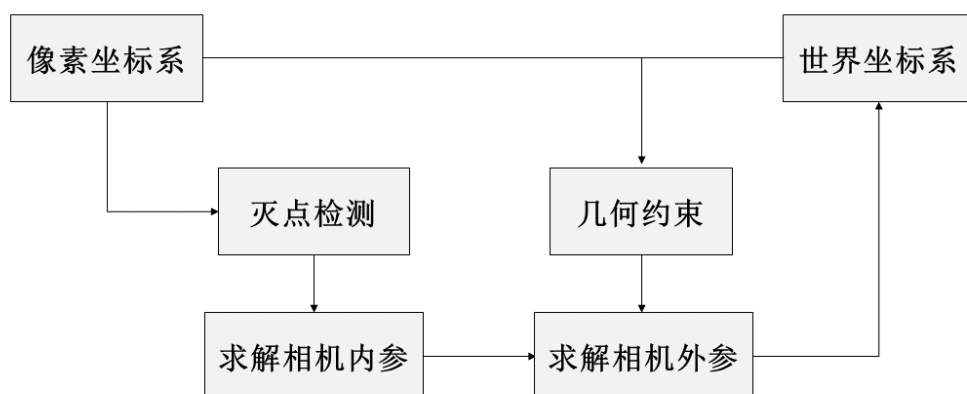
单幅图像缺少多幅图像之间的约束，标定过程仅依靠图像中所包含的几何信息。当前基于单幅图像的自标定方法多利用场景中三个垂直方向的灭点实现对相机内参的求解。

根据拉盖尔定理，3 个互相垂直的灭点所构成的三角形的垂心即为主点，可以据此计算焦距。当只有一组互相垂直的灭点时，可以假定主点位于图像的中心。因此，只需要通过一组互相垂直的灭点，即可求出相机内参矩阵。

5.1.3.2 外参矩阵求解

相机外参矩阵可以通过 PnP 算法求解。PnP 算法(Perspective-n-Point)是求解 3D 到 2D 点对运动的方法：即给出 n 个 3D 空间点时，如何求解相机的位姿。

典型的 PnP 问题求解方式有很多种，例如 P3P，直接线性变换(DLT)，EPnP(Efficient PnP), UPnP。



5.1.3.3 模型求解结果

由于只有单目相机拍摄的一张图片，不能由二维图像准确恢复出三维的信息。因此，我们采用在世界坐标系中等间隔采样，把采样点利用计算出来的投影矩阵投影到二维平面中，找到投影出的像素坐标与所求点的像素坐标最接近的点作为候选点，再依据几何约束，从候选点集中选出最优的点作为所求点的三维坐标，最后，求出所求点之间的欧氏距离即可。由于候选点密集生成，像素坐标接近的候选点可能三维坐标相差较远，因此会给最终求解结果带来较大误差。



由于缺少深度信息，通过 EPnP 算法求解出的单应矩阵存在一定误差，从而影响到候选点的选取。同时由于候选点密集生成，基于像素坐标得到的最优候选点可能于三维空间相差很远，算法不能胜任所有的任务。根据算法求出的部分结果如下：

任务一：

(1)红色车辆 A 车头和白色车辆 B 车头之间的距离：23.08m

(2)拍照者距马路左侧边界的距离：16m

任务二：

(2) 拍照者距白色车辆 B 车头的距离：18.38m

任务四：

(1) AB 的长度：6.20m

5.1.4 模型求解结果分析

图	任务	直接几何法	间接几何法
图片 1	AB 之间的距离	26.544m	23.08m
	拍照者离马路边界的距离	15.75m	16m
图片 2	AC 之间的距离	28.58m	-
	拍照者距白色车辆车头的距离	21.73m	18.38m
图片 3	相机高度	4.76m	-
	拍照者距离岗亭的距离	28.94m	-
图片 4	AB 的长度	5.026m	6.20m
	CD 的长度	4.28m	-
	建筑物的高度	8.131m	-

基于交比不变性的直接测量法的优点是算法准确性较高，可以通过几何建模求出结果。缺点是对于每一幅图片都需要基于几何关系建模，当需要处理的图片较多时，计算量会很大。

基于 EPnP 相机位姿估计的间接测量法，优点是算法计算量小，实时性高，简单且容易实现。缺点是需要有足够多的已知像素坐标与真实坐标的对应点，若对应点不够多（小于 4 个），算法无法求解结果；缺乏深度信息，无法直接从像素坐标重建出空间坐标；相机内参求解的误差会累计，造成结果的偏差。

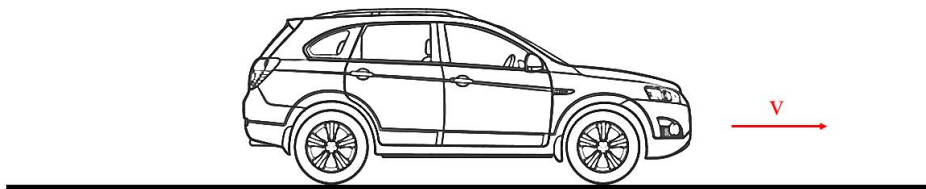
5.2 任务二：平面视频下的距离信息分析

5.2.1 问题分析

5.2.2 汽车运动模型建立

根据假设 6，我们认为任务二中所有车辆为匀速直线行驶状态。因此有：

$$s = v \cdot t$$



5.2.3 RANSAC 视频去抖动

相机在移动拍摄的过程中难免会存在一些晃动或者偏移，这也就造成了视频中的相机抖动。为了去除视频的抖动影响，需要对原视频进行运动补偿。通过查阅相关资料[5]，我们采用随机抽样一致算法 RANSAC 算法计算全局运动参数，进而去除视频的抖动，从而

得到稳定的图片拍摄序列。再对图像进行估计。

RANSAC 视频去抖动算法的具体步骤如下：

1.块匹配

首先用块匹配计算相邻两帧的运动向量集。每得到一帧图像，采用 SSD 或 SAD 算法把它和前一帧图像做块匹配，从而得到当前图像的一个比较稠密的运动向量场。

2.全局运动参数计算

根据前一步得到的运动向量场，采用 RANSAC 计算当前帧图像的全局运动参数。

3.运动参量滤波和图像运动补偿

通过滤波可以去除由抖动产生的运动参数在时间域上的高频分量，从而计算出相对平滑的运动轨迹和图像运动补偿，最后通过平移和旋转等图像变换就得到无抖动的结果图像。

5.2.3.1 块匹配

用绝对差和（SAD）对当前帧的 16×16 的块在前一帧搜索匹配块，搜索范围为 $[-15,15] \times [-15,15]$

上述块匹配算法找出了当前帧和前一帧的对应点集 $\left\{ \left((x_i, y_i)^T ; (x'_i, y'_i)^T \right) \right\}$,

$i = 1, 2, \dots, n$ 。假设每帧图像的运动包括伸缩变换，旋转变换和平移变换，则图像全局运动方程为：

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = s \times \begin{bmatrix} \cos \alpha & -\sin \alpha \\ \sin \alpha & \cos \alpha \end{bmatrix} \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix}, i = 1, 2, \dots, n$$

其中 s 表示图像的伸缩， α 表示图像绕中心逆时针旋转角度， $\begin{bmatrix} dx \\ dy \end{bmatrix}$ 表示图像中心的位移。

上式也可写为：

$$\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} x'_i & -y'_i & 1 & 0 \\ y'_i & x'_i & 0 & 1 \end{bmatrix} \times \begin{bmatrix} a \\ b \\ dx \\ dy \end{bmatrix}, i = 1, 2, \dots, n$$

其中， $s = \sqrt{a^2 + b^2}$ ， $\alpha = \tan^{-1}(b/a)$ 。

在上式中存在 a, b, dx, dy 四个未知量，可以通过用最小二乘法求解。

5.2.3.2全局运动参数计算

在图像的块匹配结果中大部分的点都有相同的运动趋势，但也有一些“坏点”的运动向量异常，当坏点数目较多或偏差较大时，最小二乘法的结果可能会有较大偏差。

RANSAC 算法对坏点有很好的鲁棒性，即使坏点的数目接近一半，RANSAC算法仍然可以得出正确的结果。由于RANSAC 算法是对数据进行多次随机取样，每次随机取出尽可能少但充分多个数据来确定模型参数，再根据已确定的模型对所有数据进行划分，因此可以减少坏点的影响。经过多次随机取样试验后，RANSAC 算法找出落在误差范围内最多的点的集合，用此集合来做最优化，最终确定模型的参数。

本文视频去抖动中求运动参数时采用的RANSAC算法描述如下：

假定块匹配的结果是在某一时刻前后两帧图像一系列对应点对的集合

$P = \left\{ \left((x_i, y_i)^T ; (x'_i, y'_i)^T \right) \right\}$ ， $i = 1, 2, \dots, n$ 。在 P 中随机取两对对应点构成 P 的子集合

$S_1 = \left\{ \left((x_i, y_i)^T ; (x'_i, y'_i)^T \right), \left((x_j, y_j)^T ; (x'_j, y'_j)^T \right) \right\}$ 。其中 i, j 为两个随机数，由 S_1 和给定的误

差范围 T ，可以得到 P 的子集 S_1^* 如下：

$$S_1^* = \left\{ \left((x_i, y_i)^T; (x'_i, y'_i)^T \right) \left\| \begin{bmatrix} x_i \\ y_i \end{bmatrix} - \begin{bmatrix} a & -b \\ b & a \end{bmatrix} \begin{bmatrix} x'_i \\ y'_i \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix} \right\| \leq T \right\}$$

其中，误差范围 T 是一个经验值，在系统中取 $T=5$ 。把 S_1^* 叫做 S_1 的一致集。

将以上过程重复 K 次，并记下元素个数最多的 S_k^* 。根据 S_k^* 中的元素通过最小二乘法可以得到最终结果。

5.2.3.3 运动参数滤波

滤除运动参量时间函数的高频分量，采用如下FIR滤波器：

$$h(t) = \begin{cases} \frac{\sin\left(\frac{2\pi t}{N-1}\right)}{\frac{2\pi t}{N-1}} & -\frac{N-1}{2} \leq t \leq \frac{N-1}{2} \\ 0 & \text{其他} \end{cases}$$

5.2.3.4 图像运动补偿

知道了图像振动或偏移的运动分量，就可以对图像做运动补偿，以消除图像的抖动。对原始图像上的每一个像素点 $[x, y]^T$ ，运动补偿后的位置 $[x', y']^T$ 为：

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos(\alpha'_t - \alpha_t) & -\sin(\alpha'_t - \alpha_t) \\ \sin(\alpha'_t - \alpha_t) & \cos(\alpha'_t - \alpha_t) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dx'_t - dx_t \\ dy'_t - dy_t \end{bmatrix}$$

5.2.4 模型求解

首先根据匀速直线运动模型：

$$s = v \cdot t = \frac{v \cdot N}{R}$$

因此，计算出别克英朗车速为：

$$v = \frac{R}{N} \cdot s$$

根据假设3，白色车行道分界线长为6米，间隔9米。因此通过观察汽车后视镜内出现白色车行道分界线的帧数，我们得到了如下数据：

其中 N 表示出现白色分界线的帧数， s 表示车在该段时间内行驶的距离。

N(frame)	3	21	39	57	75	93	110	128	146	164
s (m)	0	15	30	45	60	75	90	105	120	135
N(frame)	182	200	217	234	252	269	287	305	322	
s (m)	150	165	180	195	210	225	240	255	270	

s 与 N 存在线性关系，因此将数据用matlab进行线性回归，回归方程为：

$$s = p_1 \cdot N + p_2$$

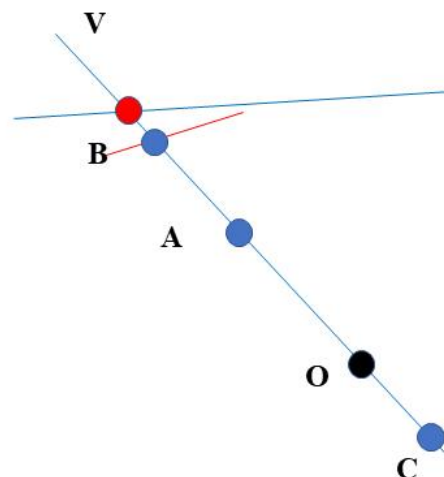
结果为：

$$p_1^{**} = 0.8464 \quad (0.8439, 0.8489)$$

$$p_2^{**} = -3.279 \quad (-3.758, -2.799)$$

也即在95%的显著性水平下，车速为：25.317~25.467m/s之间。(即91.14~91.68km/h)

(1) 与红色车辆的距离



图像中的对应的像素坐标为：

点	O	A	B	V	C
像素坐标	(377,468)	(319,406)	(281,364)	(269,352)	(399,493)

图中 O、A、B、V、C 五点共线，根据交比不变性有：

$$\frac{\|o-b\| \|v-a\|}{\|v-o\| \|a-b\|} = \frac{\|O-B\| \|\infty-A\|}{\|A-B\| \|\infty-O\|} = \frac{H}{R}$$

$$\frac{\|a-c\| \|v-o\|}{\|a-o\| \|v-c\|} = \frac{\|A-C\| \|\infty-O\|}{\|A-O\| \|\infty-C\|} = \frac{H}{R}$$

根据假设 3，车道线长为 6 米，间隔 9 米。因此，求出 $OB=43.23m$ ， $OC=0.921m$
下图为视频第 74 帧与 75 帧的图片。



可以看到第 75 帧时，白色车道分隔标线的一端出现在副驾驶车窗视野内，另一端出现在后视镜内。车窗边的白色标线可以看出，此时当线段点出现在车窗视野，另一端正好出现在后视镜内。根据经验，出现在副驾驶视野时位置大约为右前轮下。由于右前轮到车尾的距离通常为车长加轴距的一半，根据查阅的别克英朗 2016 款参数配置[6]，我们得到距离为： $(4587+2640)/2=3613.5\text{mm}$ 。

因此，当白线端点正好出现在后视镜时，端点 O 处距离别克英朗车尾距离为 $6-3.6135=2.3865\text{m}$

因此，别克英朗车于红色车辆车头的距离为 $2.3865+0.921+43.23=46.54\text{m}$ 。

(2) 超越第一辆白色车辆时两车的速度差异

由假设条件，白色车辆的车型为普通紧凑型轿车，根据假设 1，车辆的轴距为 2.6 米。根据假设，车辆均为匀速直线行驶状态，通过计算别克英朗车经过白车的帧数信息，即可求出相对车速。



根据视频，第 107 帧时经过白色车辆后车轮，第 121 帧时经过白色车辆前车轮。

$$v = \frac{R}{N} \cdot s$$

所以相对车速为：

$$v = \frac{30}{14} \times 2.6 = 5.57\text{m/s} = 20.06\text{ km/h}$$

5.3 任务三：平面视频下的距离信息分析

5.3.1 问题分析

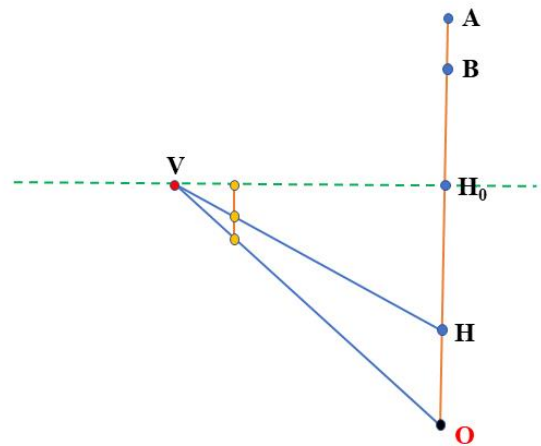
任务三中平面视频下的距离分析，由于高铁上拍摄的视频由于缺乏明显的尺度信息，通过广泛的资料查找，我们挖掘出了视频中隐含的部分物体的真实尺度。进而通过几何约束对实际问题进行建模，利用交比不变性得到对物体距离和尺度的估计。

5.3.2 模型建立

通过查阅相关资料[7]，我们得到了视频中通信基站铁塔的相关设计参数：塔高为 30m，塔尖高度为 5m，具体参数如下图所示。



(1)铁塔的几何模型



如图所示，点 O、A、B、H 四点共线，利用交比不变性：

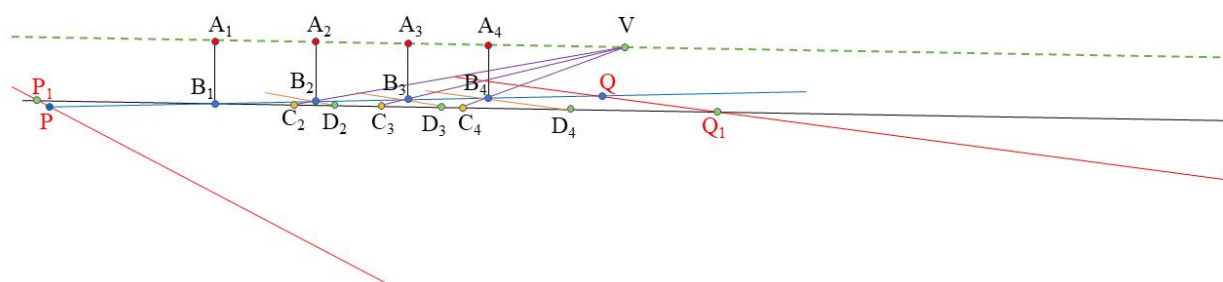
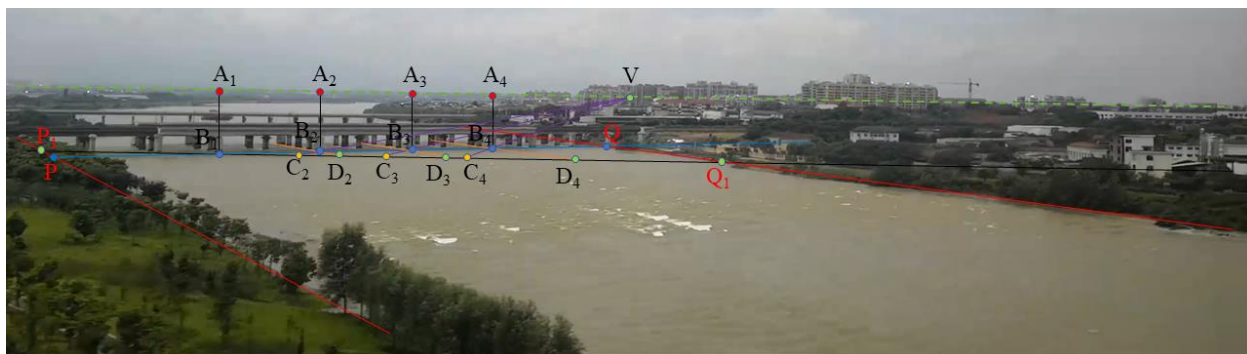
$$\frac{H}{R} = \frac{\|a-h\| \|b-o\|}{\|a-b\| \|o-h\|} = \frac{\|A-H\| \|B-O\|}{\|A-B\| \|O-H\|}$$

同理有：

$$\frac{H}{R} = \frac{\|a-h_0\| \|b-o\|}{\|a-b\| \|o-h_0\|} = \frac{\|A-H_0\| \|B-O\|}{\|A-B\| \|O-H_0\|}$$

(2) 大桥的几何模型

我们选取视频第 142 帧，根据穷举法求解并画出灭线。利用桥墩延长线与其灭线的交点，我们可以得到如下所示的几何模型：

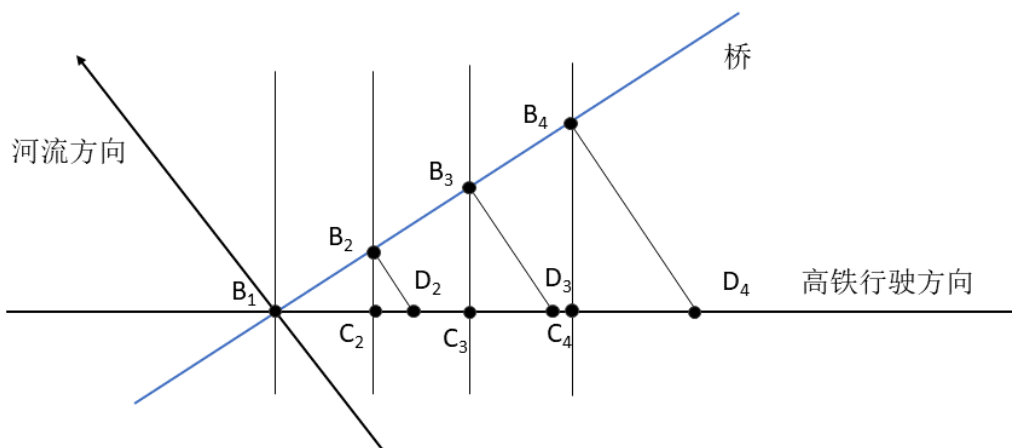


由于相机中心的灭点 V 引出的线都平行于相机的视线，而与相机底片平行的线，投影后长度比例不会发生变化

所以 P_1Q_1 对应为桥真实长度。因此我们可以得到如下图所示的几何关系：

$$\frac{B_1C_2}{A_1B_1} = \frac{C_2C_3}{A_1B_1} = \frac{C_3C_4}{A_1B_1} = \frac{\text{桥墩投影间距}}{\text{相机高度}}$$

设桥与高铁线夹角为 θ



$$\cos^2 \theta = \frac{B_1C_i}{B_1B_i}$$

可以求出河宽 $PQ = P_1Q_1 \cdot \cos \theta$

(3) 高铁的几何模型

由假设 7，视频中高铁处于匀速直线运动状态。因此，求高铁速度，只需要计算出相机中高铁经过已知的真实距离段所花费的时间即可。

设 B_4 在高铁线上的投影位置 X (该位置在照片上的像素值无穷大), 利用交比不变性可以求解出 B_4X 的长度, 从而得到 P, Q 与高铁的距离。

$$PX_1 = B_4X - P_1C_4 \times \tan\theta$$

$$QX_1 = B_4X + Q_1C_4 \times \tan\theta$$

5.3.3 模型求解结果

像素坐标如下:

点	A	B	H	O	H_0	A_1
像素坐标	(1096,421)	(1094,490)	(1087,851)	(1084,982)	(1091,652)	(329,616)
点	A_2	A_3	A_4	B_1	B_2	B_3
像素坐标	(484,618)	(625,620)	(750,622)	(329,710)	(484,708)	(625,705)
点	B_4	V	C_2	C_3	C_4	D_2
像素坐标	(750,703)	(961,626)	(455,712)	(587,714)	(712,716)	(516,713)
点	D_3	D_4	P_1	Q_1		
像素坐标	(678,715)	(866,718)	(54,706)	(1079,722)		

OP 实际长度 $h=7.14$ 米

相机高度 $h_0=19.11$

桥墩在高铁线上的投影间距=26.02 米

P_1Q_1 的长度=208.4 米

桥与高铁线夹角 $\cos\theta=0.842$, $\theta=31.65^\circ$

河宽 $PQ=P_1Q_1 \times \cos\theta=175.5$ 米

高铁速度= $P_1Q_1 / (219-138)=77.2\text{m/s}=278\text{km/h}$

C_4B_4 实际距离= $26.02 \times \tan\theta=16.67$ 米

B_4X 实际距离=307 米

$P_1C_4=133.8$ 米, $Q_1C_4=74.6$ 米

P_1 距高铁的距离 $PX_1=307-133.8 \times \tan\theta=221$ 米

Q_1 距高铁的距离 $QX_1=307+74.6 \times \tan\theta=355$ 米

故桥距离高铁铁路最近 221 米, 最远 355 米

5.4 任务四：立体视频下的距离信息分析

5.4.1 问题分析

立体视频中由于相机拍摄角度不平行于地面, 且由于距离地面高度过高, 参照物的选取变得更为困难。此时几何量测法不适用。

本文使用基于深度学习的 SafeUAV 算法估算无人机的飞行高度、计算物体深度信息。再进一步利用射影几何的知识, 利用灭点计算相机内参, 并将单应矩阵进行分解。通过编写重建单位距离网格的算法, 将等高度的单位距离网格可视化, 对立体视频进行分析并估计物体尺寸和相机位置。

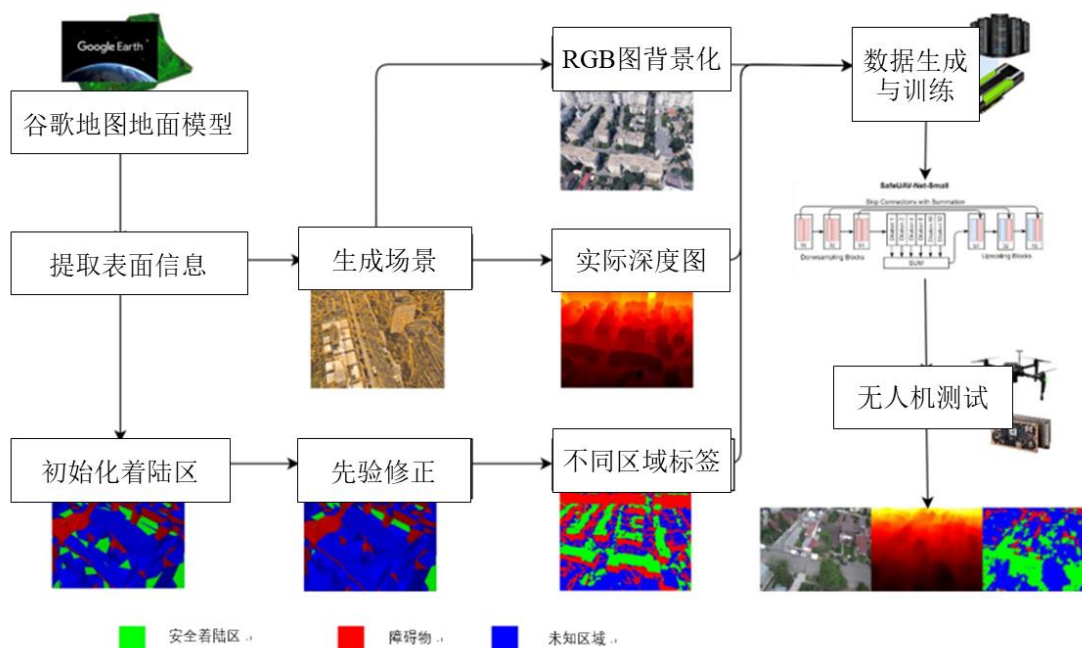
5.4.2 SafeUAV 算法求解深度

SafeUAV 算法[8]是为了给无人机提供能够安全着陆而提出的基于深度学习的算法。其基本原理就是利用合成的深度图片数据来训练网络, 让网络能够直接从无人机拍摄的图片中就能获得深度信息, 从而测算无人机飞行的高度。而且此算法还将地图进行了分类, 分为了可着陆区域和不可着陆区域, 训练好模型之后就可以直接来预测无人机可以着陆的位置。

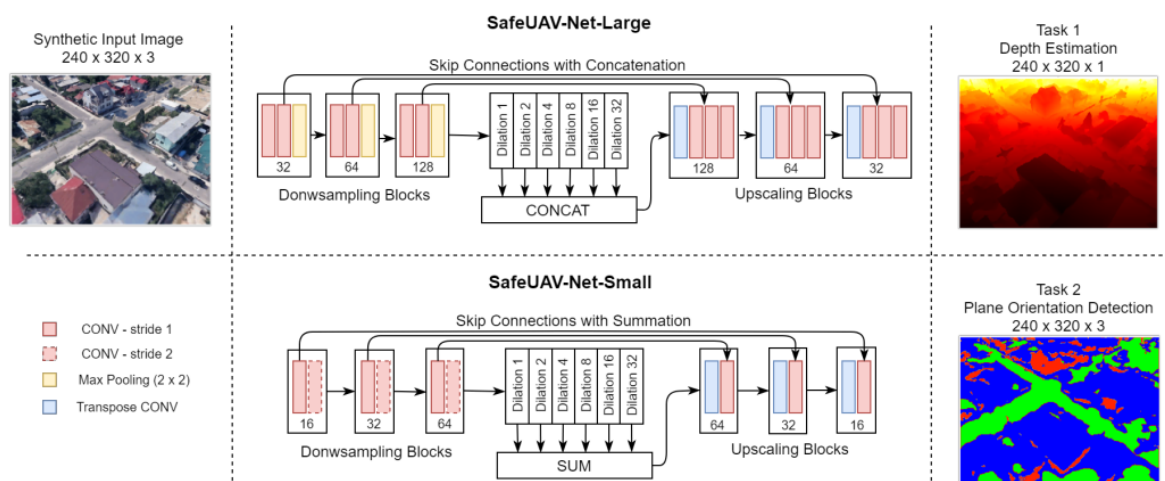
任务四立体视频下的距离分析为无人机视角，为了测算无人机的高度以及速度，图像的深度信息至关重要。因此，我们采用 SafeUAV 算法和已经在大型数据集上训练好的模型[9]，对无人机拍摄视频中的深度信息进行预测，进而得到图像中各个像素点距离无人机的实际距离。

5.4.2.1 SafeUAV 算法原理

SafeUAV 算法原理如下图所示：



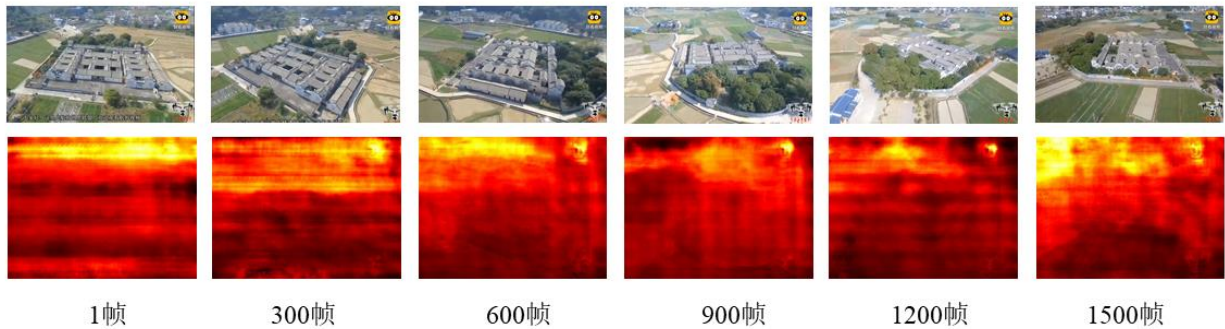
网络框架：



SafeUAV 算法提出了两种方案，第一种是将提取的特征进行拼接，进而预测结果，第二种方案是将卷积层提取的特征进行求和来特征融合，并分别实现。网络是类似于 UNet 的全卷积网络，loss 函数采用多分类交叉熵。

5.4.2.2 输出结果

将任务四中的视频输入网络，输出得到如下的深度图，无人机的飞行高度约为 65m。



5.4.3 单应矩阵求解

通过 SafeUAV 算法，我们能估计出无人机飞行时的高度。基于此，我们通过计算单应矩阵来获得像素坐标系与世界坐标系的对应转换关系。

首先需要对视频进行预处理，去除掉黑边并将视频拉伸至 1280×720 大小。

(1) 相机内参计算

由上文可知，相机内参可以简化为三参数模型，其中 f 可以通过一组垂直方向上的灭点进行约束求解。因此，我们选取第 340 帧，找出两个灭点坐标分别为 $(-348, -101)$, $(1662, -87)$ ，计算得到 $f = 1276$ 。

(2) 按变换层次求解单应矩阵

参考平面和图像平面之间的映射是一种射影变换，可分解为一串变换链的复合。平面单应矩阵可作如下分解：

$$\mathbf{H} = \mathbf{H}_s \mathbf{H}_A \mathbf{H}_p = \begin{bmatrix} \mathbf{sR} & t \\ \mathbf{0}^T & 1 \end{bmatrix} \begin{bmatrix} 1/\beta & -\alpha/\beta & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_1 & l_2 & l_3 \end{bmatrix}$$

其中， $\mathbf{H}_s, \mathbf{H}_A, \mathbf{H}_p$ 分别为相似变换、仿射变换和射影变换。其中， \mathbf{H}_p 与灭线相关，

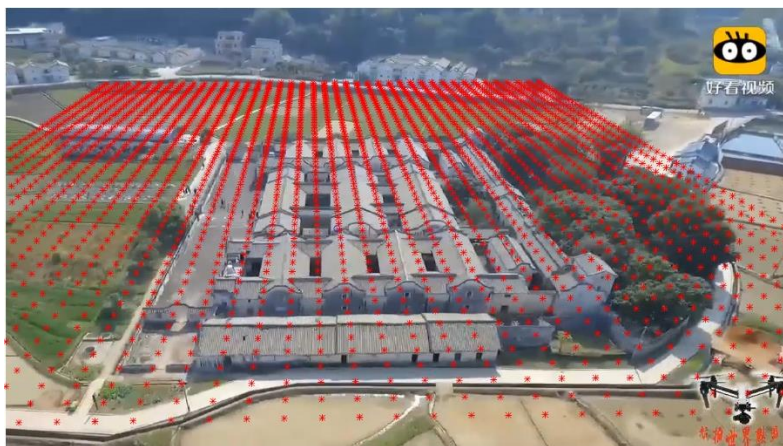
\mathbf{H}_A 影响仿射性质而与灭线无关； \mathbf{H}_s 为一般的相似变换，它不影响仿射及射影性质。

当单应矩阵恢复到相似变换层次，校正后的图像与实际对象只相差一个比例因子。按变换层次求解单应矩阵的方法并没有完全恢复单应矩阵，通常只应用于距离的量算。

由于我们已通过 SafeUAV 算法估算得到了无人机飞行的高度信息，因此，我们可以将单应矩阵分层进行求解，同时通过生成可视化图像估计求解误差并进行修正。

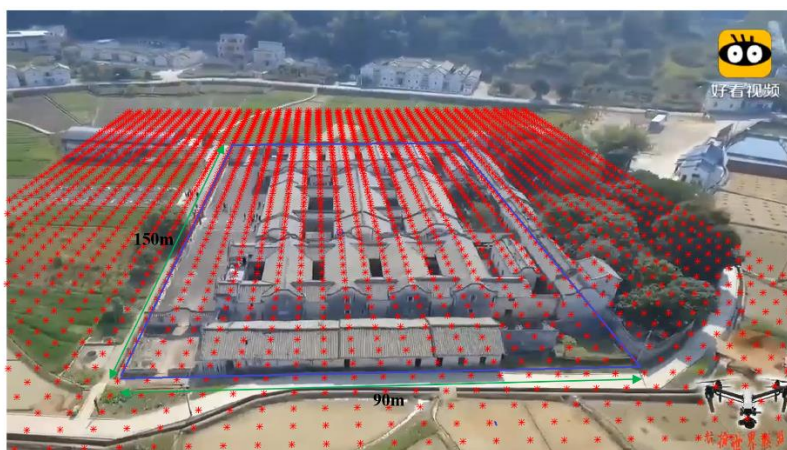
5.4.4 模型求解结果

为了方便参数修正，我们使用拍摄方向与屋脊方向基本一致的 539 帧。调整参数到生成网格与照片中参照物（道路，房屋）基本重合。



我们知道在水平面上的点在照片上有一一对应关系，所以当重合时，设定参数应与实际拍摄时的参数相同。即我们可以通过参考网格点的准确度反推出设定参数的置信度，从而对参数进行修正。

当飞行高度设定为 65 米时：

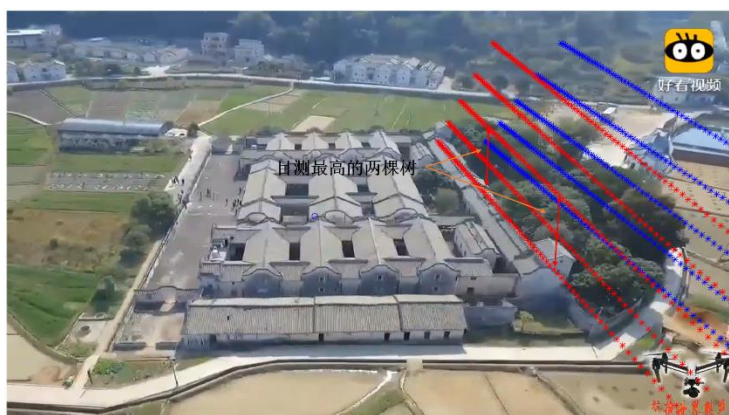


图中红点的间隔为 5 米。此时可以看到网格图与外墙轮廓基本重合，同时通过此网格可以推出房门宽度约为 0.9~1 米，符合通常的门宽。因此推断，我们设定的参数符合实际。

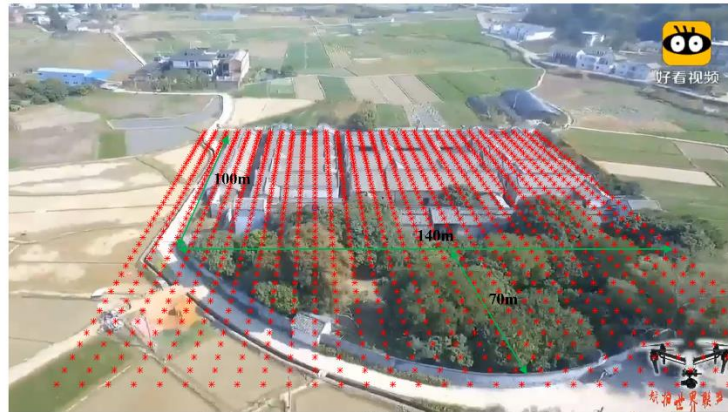
通过网格，我们可以估算得到道路宽度和长度。道路宽度最大约 5 米，最窄部分为 2 米。在上图中，庄园面向相机的边长约为 90 米，另一边长约为 150 米。

通过设定高度 Z 生成网格与屋脊比对，得到高度为 6 米时基本吻合，因此推断庄园内大部分房屋高 6 米，后方有两个三层建筑，高度约 9 米。

在视频中通过观察，得到后花园内最高的树为靠近内院墙的两棵，在后花园作出等高线，观察发现树高约为 10 米。



为了消除误差，我们同时选取 921 帧进行结果比对。设定飞行高度为 80 米后得到了吻合的网格。同时得到庄园长 140 米，宽 100 米，与之前结果偏差不大。庄园后花园近似为半圆，半径为 75 米。



因此，可以得到道路总长 $2 \times (100 + 150) \times 75 \times 3.14 \approx 735$ 米。

基于以上结果，得到无人机飞行轨迹约为半径 200 米的圆，经历半圆的帧数为 539~1477 帧，时间为 31 秒，得到无人机飞行速度为 20m/s。

综上所述，无人机飞行高度在 65~80m 之间，飞行速度约为 20m/s。

6 模型的评价与改进

6.1 模型的评价

针对任务一的单幅图像信息分析，本文运用了多个模型来解决。其中单幅图像中的灭点提供的几何约束对距离信息度量极为关键。本文采取了霍夫变换、基于 Gabor 小波的消失点检测、穷举法等算法求解图像中的灭点，并比较了算法结果。基于灭点提供的几何约束，我们采取了两种不同的方法求解。直接几何法利用了射影变换中的交比不变性来求解，间接几何法可以通过相机 EPnP 位姿估计算法进行求解。

针对平面视频下的信息分析，本文首先利用 RANSAC 算法去除视频中的抖动，从而得到运动稳定的视频。通过广泛的资料查找，我们挖掘出了视频中隐含的部分物体的真实尺度。进而通过几何约束对实际问题进行建模，利用射影变换中的交比不变性得到对物体距离和尺度的估计。

针对立体视频下的信息分析，本文采取了基于深度学习的无人机飞行高度估计算法 SafeUAV 对视频中的深度和无人机飞行高度进行估计。SafeUAV 算法对无人机拍摄视频中的深度信息进行预测，进而得到图像中各个像素点的深度图。基于 SafeUAV 算法提供的深度预测，我们进一步通过层次分解求解单应矩阵，从而得到了立体视频下的物体距离和尺度估计。

本文提出算法模型可以有效的利用图片中的隐含信息，在仅含少量先验信息的情况下，对图片和视频中的距离、尺度信息进行高效的估算。

6.2 模型的改进

任务一中的 EPnP 算法求解相机位姿时存在误差，这是由于图像中已知的真实尺度过少，模型的数据不够，后续可以通过挖掘更多的尺度信息增加更多的数据，从而得到更为准确的结果；任务二和任务三中，只利用了几何法求解，没有利用视频序列前后帧的关键点求解相机位姿，后续可以对视频进行单目三角化从而求解视频中的距离、尺度信息。通过两种方法交叉验证结果。任务四中，利用的是已经在大规模数据集上训练过的模型，后

续可以用自己的数据集重新训练网络，进一步优化深度计算结果。

7 参考文献

- [1] GB 51038 – 2015.城市道路交通标志和标线设置规范[S]. 北京：中国计划出版社，131-146，2015.
- [2] Bazin J C , Seo Y , Demonceaux C , et al. Globally Optimal Line Clustering and Vanishing Point Estimation in Manhattan World[C]// Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on. IEEE, 2012.
- [3] smallflyingpig, 基于消失点检测的道路检测.<https://blog.csdn.net/smallflyingpig/article/details/70550715>,2019.9.21
- [4] 刘学军, 王美珍, 甄艳等: 单幅图像几何量测研究进展[J], 《武汉大学学报》(信息科学版), 36(8): pp941 - 947.
- [5] 赵文华, 姚天翔, 叶秀清, 顾伟康. RANSAC 算法在视频去抖动中的应用[J]. 电路与系统学报, 2005, (04): 91-94.
- [6] 汽车之家, 上汽通用别克-英朗参数配置, <https://car.autohome.com.cn/config/series/982-8625.html#pvareaid=3454437>, 2019.9.21.
- [7] 高扬明. 移动通信基站单管塔水平位移的设计[J]. 电信技术, 2015, 7(12):59-61.
- [8] Marcu A, Costea D, Licaret V, et al. SafeUAV: Learning to estimate depth and safe landing areas for UAVs from synthetic data[C]//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 0-0.
- [9] Pîrvu Mihai Cristian, Depth-HVN-Estimation-Aerial, <https://gitlab.com/mihaicristianpîrvu/Depth-HVN-Estimation-Aerial>.2019.9.22

8 附录

相关程序代码

霍夫变换：（Matlab）

```
img = imread('图 4.png');
subplot(221), imshow(img), title('original image');
img_gray = rgb2gray(img);
prewittSample = uint8(filter2(fspecial('prewitt'),img_gray));%图像锐化
prewittSample = medfilt2(prewittSample,[3,3]) ;
subplot(222), imshow(prewittSample), title('edge enhance');
img_gray = img_gray+uint8(prewittSample);
% imshow(addSample)
% the canny edge of image
img_gray = medfilt2(img_gray,[3,3]) ;
BW = edge(img_gray,'canny');%canny 边缘检测
subplot(223), imshow(BW), title('image edge');
% the theta and rho of transformed space
[H,Theta,Rho] = hough(BW);
subplot(224),
imshow(H,[],'XData',Theta,'YData',Rho,'InitialMagnification','fit'),...
    title('rho\_theta space and peaks');
```

```

xlabel('\theta'), ylabel('\rho');
% axis on, axis normal, hold on;
% label the top 5 intersections
P = houghpeaks(H,5,'threshold',ceil(0.3*max(H(:))));
x = Theta(P(:,2));
y = Rho(P(:,1));
plot(x,y,'*','color','r');
lines = houghlines(BW,Theta,Rho,P,'FillGap',50,'MinLength',10); %直线检测
figure, imshow(img), hold on
max_len = 0;
for k = 1:length(lines)
    xy = [lines(k).point1; lines(k).point2];
    plot(xy(:,1),xy(:,2),'LineWidth',1,'Color','g');
end
% f=getframe(gcf);
% imwrite(f, strcat('D:\Pictures\1_1.png'), 'jpg');%

```

EPnP 算法求解单应矩阵：（Python）

```

# -*- coding: utf-8 -*-
"""
Created on Thu Sep 19 22:48:55 2019

@author: Administrator
"""
import cv2
import numpy as np
import math
import matplotlib.pyplot as plt

def img_world(xy,K_in,RT):
    xy=np.array([xy[0],xy[1],1]).reshape(3,1)
    rt=np.dot(K_in,RT)
    world_coord=np.dot(np.linalg.pinv(rt),xy)
    world_coord_list=[world_coord[0]/world_coord[-1],world_coord[1]/world_coord[-1],world_coord[2]/world_coord[-1]]
    return world_coord_list

def com_rt(p3d,p2d,k_ycbvideo):
    # retval, rot, trans, inliers = cv2.solvePnP(Ransac(p3d, p2d, k_ycbvideo,
    None, flags=cv2.SOLVEPNP_EPnP)
    dist_coefs = np.array([0, 0, 0, -0], dtype=np.double)
    found, rvec, tvec = cv2.solvePnP(p3d, p2d, k_ycbvideo, dist_coefs)
    R = cv2.Rodrigues(rvec)[0] # convert to rotation matrix
    T = tvec.reshape(-1, 1)

```

```

    rt = np.concatenate((R, T), 1)
#    print(rt)
    return rt

def com_distance(xy1,xy2):

    w1=xy1[0]
    w2=xy2[0]
    d=math.sqrt(pow(w1[0]-w2[0],2)+pow(w1[1]-w2[1],2)+pow(w1[2]-w2[2],2))
    return d

def perspect(k,rt,w):
    l=w.shape[0]
    a=np.array([1]*l).reshape(1,1)
    _w=np.c_[w,a]
    _uv=np.dot(np.dot(k,rt),_w.T)
    uv=np.zeros((1,2))
    _uv=_uv.T
    uv[:,0]=_uv[:,0]/_uv[:,-1]
    uv[:,1]=_uv[:,1]/_uv[:,-1]
    return uv

def Visual2dImg(uv,p2d,img):
    #plot scatter figure by uv coordinates
    x,y=uv[:,0],uv[:,1]
    x1,y1=p2d[:,0],p2d[:,1]
    fig = plt.figure(figsize=(20,20))
    ax1 = fig.add_subplot(111)
    ax1.set_title('Scatter Plot')
    plt.xlabel('X')
    plt.ylabel('Y')
    ax1.scatter(x,y,c = 'r',marker = 'o')
    ax1.scatter(x1,y1,c = 'g',marker = 'o')
    ax1.imshow(img)
    #
    plt.show()

def Vis_2(uv,img):
    #plot scatter figure by uv coordinates
    x,y=uv[:,0],uv[:,1]
    fig = plt.figure(figsize=(20,20))
    ax1 = fig.add_subplot(111)
    ax1.set_title('Scatter Plot')
    #

```

```

plt.xlabel('X')
plt.ylabel('Y')
ax1.scatter(x,y,c = 'r',marker = 'o')
ax1.imshow(img)
#
plt.show()

def com_world(x,y,h,alpha):
    x1,x2=x
    y1,y2=y
    w1=np.linspace(x1,x2,int(abs(x2-x1)/alpha))
    w2=np.linspace(y1,y2,int(abs(y2-y1)/alpha))

    lx=w1.shape[0]
    ly=w2.shape[0]

    w1=w1.reshape(1,lx)
    w2=w2.reshape(1,ly)
    _w1=w1.repeat(ly,axis=0).reshape(lx,ly,1)
    _w2=w2.repeat(lx,axis=0).T.reshape(lx,ly,1)

    world_coord=np.concatenate((_w1,_w2),axis=-1)
    world_coord=world_coord.flatten().reshape(ly*lx,2)
    cc=np.array([h]*lx*ly).reshape(lx*ly,1)
#    print(np.c_[world_coord,cc].shape)
    return np.c_[world_coord,cc]

def get_xs(yy,x,y,alpha):
    h=0
    world_coord=com_world(x,y,h,alpha)
    world_new=world_coord[np.where(world_coord[:,1]==yy)]

    return world_new

def comput_dis(k,rt,A,world_a):
    delta=40
#    world_a=com_world(xa,ya,0,100)
    uv_a=perspect(k,rt,world_a)

    x1,y1=A

    res_a=[]

```



```

for i in range(uv_a.shape[0]):
    co=uv_a[i]
    dist=math.sqrt(pow(co[0]-x1,2)+pow(co[1]-y1,2))
    if dist<=delta:
        res_a.append(world_a[i])
    else:
        continue

return res_a

def select_point(w,a):
    x1,y1=a
    _dist=[]
    for co in w:
        dist=math.sqrt(pow(co[0]-x1,2)+pow(co[1]-y1,2))
        _dist.append(dist)

    index_min=_dist.index(min(_dist))
    pt=w[index_min]

#    print(pt)
    return pt

k_ycbvideo = np.array([[1450, 0.00000000e+00, 1000],
                        [0.00000000e+00, 1450, 750],
                        [0.00000000e+00, 0.00000000e+00, 1.00000000e+00]])

_p3d=np.array([[0,0,0],[0,1600,0],[1600,1600,0],[800,0,0]],dtype=np.double)
p3d=np.zeros((4,3),dtype=np.float)
p3d[:,-1]=_p3d[:,-1]*1.25
p3d[:,0]=_p3d[:,0]*1.25
p3d[:,1]=_p3d[:,1]*1.25

p2d=np.array([[1811,1437],[1004,1264],
              [1313,994],
              [1815.5,1188.9]],dtype=np.double)
rt=com_rt(p3d,p2d,k_ycbvideo)

img_uv=perspect(k_ycbvideo,rt,p3d)
img=cv2.imread("D:\\Mathematical_modeling\\pictures\\1.png")
img= cv2.cvtColor(img, cv2.COLOR_BGR2RGB)

```

```

##A
a=[585.6767,688.5931]
##B
b=[1251.8,609.6516]
yy=1800
w_a=get_xs(yy,[0,4000],[0,1800],100)

#print(w_a)
r_a=comput_dis(k_ycbvideo,rt,a,w_a)
r_b=comput_dis(k_ycbvideo,rt,b,w_a)
pt_a=select_point(r_a,a).reshape(1,3)
pt_b=select_point(r_b,a).reshape(1,3)
d=com_distance(pt_a,pt_b)
uv_a=perspect(k_ycbvideo,rt,pt_a)
uv_b=perspect(k_ycbvideo,rt,pt_b)
print(pt_a)
print(uv_b)

uv_w_a=perspect(k_ycbvideo,rt,w_a)
Vis_2(uv_w_a,img)

print("The distance of AB is {}".format(d))
Visual2dImg(uv_a,uv_b,img)

#uv_a=perspect(k_ycbvideo,rt,wa)
#Vis_2(uv_a,img)

```

计算相机内参：（Matlab）

```

size=[1280,720];
V1=[-348,-101];
V2=[1662,-87];

O=size/2;
O1=O;
O2=O;
O1(1)=O(1);
O1(2)=V1(2)-(V1(2)-V2(2))*(O(1)-V1(1))/(V2(1)-V1(1));
V101=sqrt((O1(1)-V1(1))^2+(O1(2)-V1(2))^2);
OO2=(O(2)-O1(2))*(O(1)-V1(1))/V101;

O2(1)=O1(1)+OO2*(V1(2)-O1(2))/V101;
O2(2)=O(2)+OO2*(O(1)-V1(1))/V101;

V202=sqrt((V2(1)-O2(1))^2+(V2(2)-O2(2))^2);

```

```
V102=sqrt((V1(1)-O2(1))^2+(V1(2)-O2(2))^2);
```

```
h=sqrt(V202*V102-O02^2);
```

单位距离网格重建：（Matlab）

```
h=1276.2;
H=65;
L=200;
size=[1280,720];
O=[0,0];
O1=size/2;
%光心修正
O1(1)=O1(1)-100;
HL=H^2+L^2;
%相机旋转修正
b=-0.00;
a=sqrt(1-b^2);

figure
img=imread('539.png');
imshow(img);
hold on
plot(O1(1),O1(2),'ob');
%平面高度
z=0;
for x=-100:5:100
    for y=-200:5:100
        %坐标旋转
        d=-0.03;
        c=sqrt(1-d^2);
        A=[x*c-y*d,y*c+x*d];
        %比例系数 k
        k=(h*sqrt(HL)*H)/((HL-L*A(2))*(H-z));
        x1=A(1)*k;
        y1=(A(2)-z*L/H)*H*k/sqrt(HL);
        A1=[x1,y1];
        x2=O1(1)+x1*a-y1*b;
        y2=O1(2)+y1*a+x1*b;
        plot(x2,y2,'*r');
    end
end
```