传感器联合标定与感知融合（标定篇）

一）背景扯皮

为啥把这两个放在一起呢?两个貌似没有关联的板块其实有很多的共性，如空间共性，时间共性等。

先介绍一下标定。传感器标定可以分为两个部分，内参标定和外参标定。内参标定是决定传感器内部的映射关系，比如摄像头的焦距、畸变系数等，而外参是决定传感器和车辆坐标系的转换关系，如旋转平移等。为啥会有联合标定呢？各种传感器得出的数据格式，摄像头获取的是 RGB 的图像信息流，激光雷达获取的是 3D 点云距离信息；GPS-IMU 给出的是车身位置姿态信息；毫米波雷达给出的是 2D 反射图，所以要取得一个最佳的衡量值，来使得传感器之间的数据性能达到最优。

标注，不管是 targetless 还是 target 的标注，都需要纹理特征比较明显的场景，如 targetless 需要的是空旷的、有树木、路灯、交通标识牌、车道线等纹理信息明确的，而 target 虽然受场地约束，但标定参考物纹理特征也比较明显，如棋盘格之类的。

单一传感器的标定会在后续的章节中详细介绍，联合标定按照传感器类型不同，可以分为：相机到相机、相机到激光、相机到毫米波、相机到 IMU 的标定。这里仅就相机到相机、相机到激光的外参进行标定（相对车辆后轴中心的坐标）。

相机到相机：一般智能驾驶会采用多个相机，长距相机一般视野比较小，用来检测远处场景，短距相机的视野比较大，一般用来车道线检测和近处检测等。常规的做法一般是将同一视野范围内的图像进行融合（和融合相关了）判断，检测同一目标是否进行对齐。在融合图像中，一般选择距离适中的目标进行对齐（如 50 米以外），这样重合后能够得到较高的精度。如果出现重影或者错位，那么就认为是存在误差，当误差大于一定范围（范围根据经验定义）时，标定失败，需要重新进行标定。（注：近处物体因为相机畸变，存在水平错位，且距离越近，错位量越大。纵向不受畸变影响。）

相机到激光：相机和激光的标定是目前经常采用方法是投影法，即将产生的点云数据投影到图像内（数据融合来了），然后寻找其中具有明显边缘的物体

和标记物，检查其轮廓对齐情况，一般算法包括语义分割和点云分割等。如果在 50 米以内的目标，点云边缘和图像边缘能够重合，则可以证明标注结果的精度很高，如果出现错位，则标定结果存在误差。当误差大于一定范围时，该外参不可用。

二）技术扯皮

关于传感器联合标定，网上有很多内容可以去参考，这里没啥可以写的，就简单的总结一下。

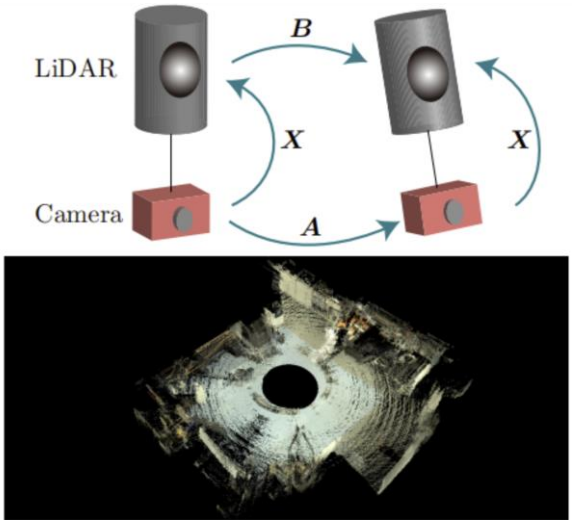


图 1 基于传感器融合里程计的联合标定

上图是日本东京大学发的一篇激光和摄像头联合标定的方法。虽然也是基于手眼标定法，但其基于传感器融合里程计的摄像头运动估计思想，先分别获取摄像头和激光的初始外参，然后利用初始外参和点云数据，重新计算带尺度的相机运动，并根据这些运动重新计算外部参数，并反复进行相机运动和外部参数的交互，直到估计收敛为止。

图 1 上中，假设 A 和 B 分别为两个固定传感器观察到的位置和方向变化，X 为两个传感器之间未知的相对位置和方向。进而得到表达式 AX=XB，利用该表

达式求解 X，得到传感器之间的外部参数。然后根据噪声对传感器的影响，在标定方法中采用卡尔曼滤波对传感器的偏差进行补偿。

从图 1 可以看出，激光雷达的姿态发生了变化，针对这种变化，一般采用 ICP 来进行姿态的求解。而相机的运动，一般采用特征匹配。

然后两种方式进行结合进行外部参数的估计。大概的流程图如图 2。

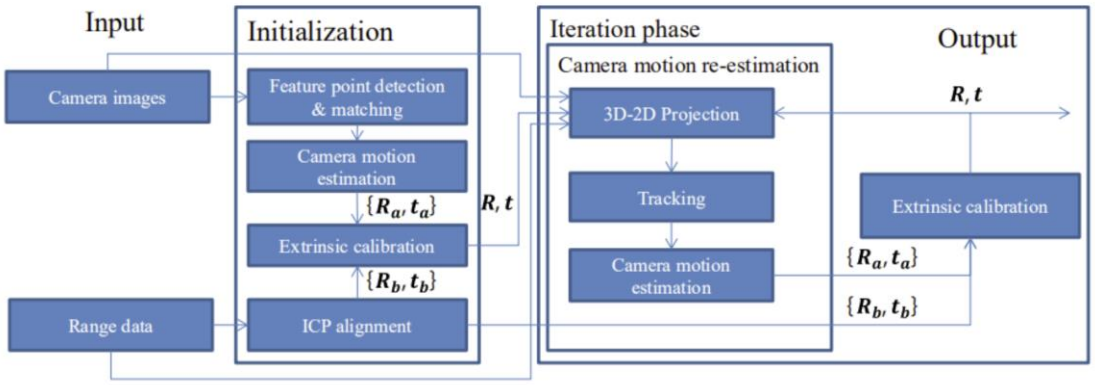


图 2 基于传感器融合里程计联合标定流程图

手眼标定法的经典解法一般是先求解旋转矩阵，然后再估计平移向量。如下公式，假设相机与激光测量的第 i 个位置和姿态变化分别为 4*4 的矩阵 Ai、Bi，两个传感器之间外部参数为 4*4 的矩阵 X，可以建立$AiX = XBi$，公式建模有：

$$R_\alpha^i R = RR_b^i \tag{1}$$

$$R_\alpha^i t + t_\alpha^i = Rt_b^i + t \tag{2}$$

假设k_α^i和k_b^i为旋转矩阵R_α^i和R_b^i的旋转轴，当公式 1 成立时，下面的公式成立：

$$k_\alpha^i = Rk_b^i \tag{3}$$

由于无法计算相机帧间平移运动的绝对尺度，公式 2 用比例因子S_i写成如下：

$$R_\alpha^i t + S^i t_\alpha^i = Rt_b^i + t \tag{4}$$

利用k_α^i和k_b^i系数的 SVD 分解，对 R 进行线性求解。这里解决旋转问题，需要两个以上的位置和姿态变化，并且变换序列中必须包含不同方向的旋转。在非线性优化中，R 通过最小化从式 1 中导出代价函数赖进行优化。

$$R = \arg\min_R \sum_i |R_\alpha^i R - RR_b^i| \tag{5}$$

因为相机存在 scale 问题（啥意思呢，就是说，逻辑分辨率<和软件算法相关，方图>和物理分辨率<和晶体管相关，圆图>之间的缩放关系），上述算法存在不稳定因素，所以激光雷达数据就需要其发挥作用。见图 3，输入是世界

坐标系中的点云，在相机 1、2 的位置拍摄的两个相机图像以及用于将相机 1 定位到世界坐标系的外参。

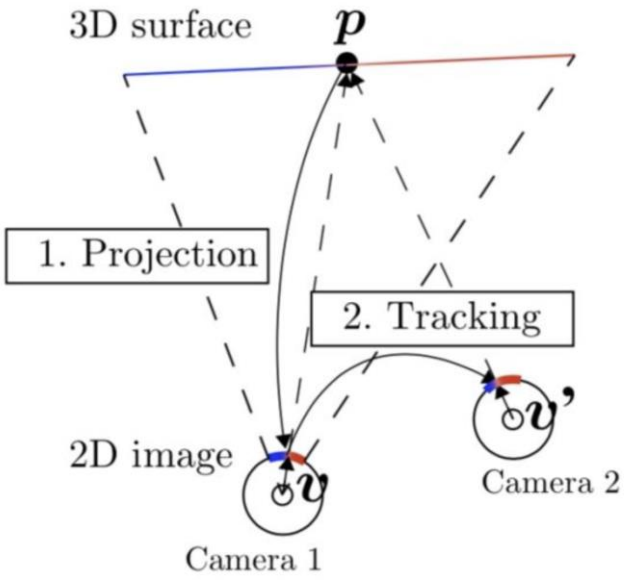


图 3 2D-3D 之间的转换关系

首先使用投影函数将点云中的一个点 p 投影到相机 1 的图像上，其 2D-3D 对应关系可以如下，其中 V 是从相机 1 的中心点到相应像素的矢量方向。

$$v = Proj(Rp + t) \tag{6}$$

将公式 6 带入到公式 5 中，那么就有

$$R_a^i, t_a^i = \arg\min_{Ra,ta} \sum_j |v_j' \times Proj(R_a(Rp_j + t) + ta)| \tag{7}$$

咦，公式 7 很熟悉，那么公式 7 的初始解，就可以通过经典 P3P 求得。

在相机位置和相机姿态发生转换后，使用相机和激光雷达的运动再次优化外部参数，每次迭代中，R 和 t 都是线性和非线性求解。在非线性优化中，R 由公式 5 进行优化，t 由下面公式进行优化：

$$t = \arg\min_t \sum_i |(R_a^i t + t_a^i) - (Rt_b^i + t))| \tag{8}$$

有啥问题没？因为运动估计和外部参数估计都存在一定的误差，且依赖于被测环境和运动次数（路面颠簸啥的），很难获得精确的收敛条件。所以就需要考虑收敛估计的运动。首先考虑一下外参误差对相机定位的影响。如下图展示。

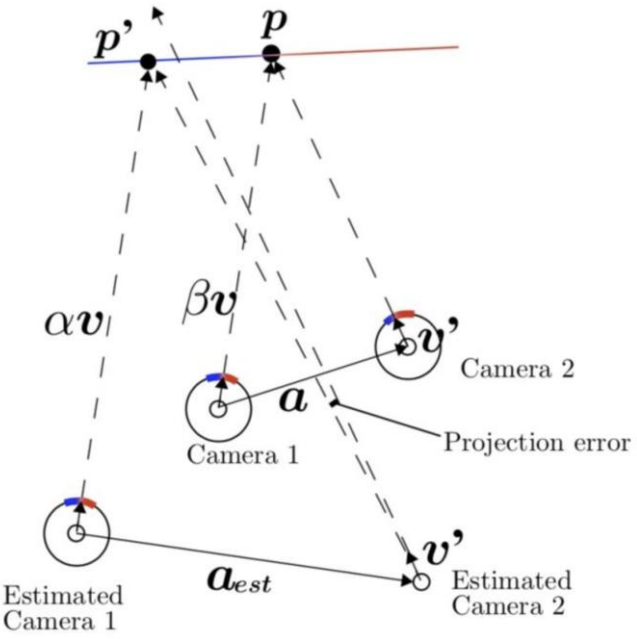


图 4 外参误差与相机定位之间的转换关系

看图说话，1）相机实际运动 a 越小，投影误差也就越小。2）$(\alpha-\beta)$越小，投影误差越小。这就要求，标定的时候相机运动要小，标定周围的环境深度纹理要小。

日本东京大学是将激光点云投影到图像上，利用图像轮廓进行标定，而斯坦福大学(Automatic Online Calibration of Cameras and Lasers)方法更简单，直接采用点云和图像匹配赖进行标定的。斯坦福采用的是在线修正标定的“漂移”，如下图 5，精确的标定是的图中绿色点和红色边缘匹配（通过逆距离变换 IDT，inverse distance transform）。

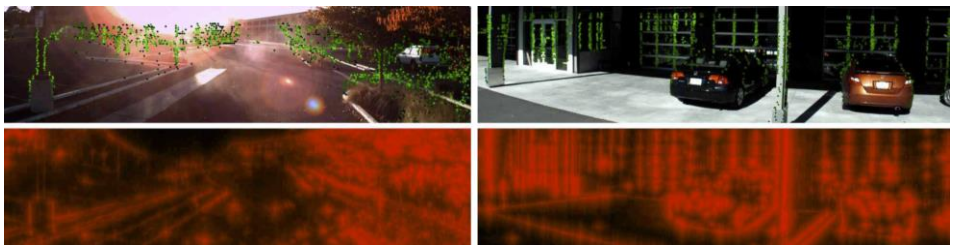


图 5 斯坦福点云图像匹配法

咦，问题来了，目标函数咋写？文中这么定义。

$$J_C = \sum_{f=n-w}^{n} \sum_{p=1}^{|X^f|} X_p^f \cdot D_{i,j}^f$$

其中 W 是图像大小，f 是帧号，(i,j)是图像中像素的位置，而 p 是点云的 3-D 点，X 表示激光雷达点云数据，D 是图像做过 IDT 的结果。

图 6 显示的是标定后的结果，最上面是标定好的，中间是出现漂移的，下面是重新标定的效果。

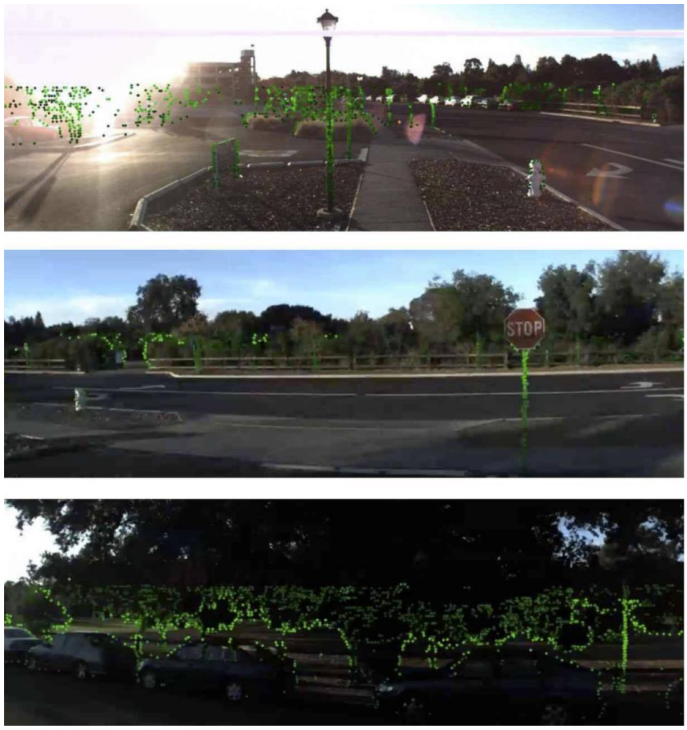


图 6 斯坦福激光相机联合标定结果展示

简单吧，而且效果也就一般，为了提升标定效果，密歇根大学< Automatic Targetless Extrinsic Calibration of a 3D Lidar and Camera by Maximizing Mutual Information> 在斯坦福的基础上做了一些复杂操作（看完发现基本上没啥基础）。密歇根主要是为了求解两个传感器之间的转换关系。如图 7（盗图），求 R，T。

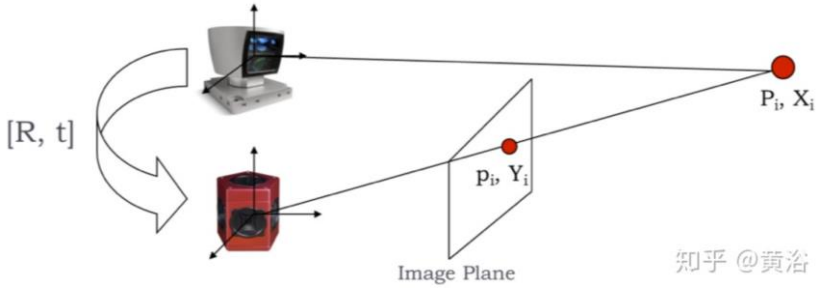


图 7

文中定义了 Mutual Information(MI)目标函数是一个熵：

$$\mathrm{MI}(X,Y) = \mathrm{H}(X) + \mathrm{H}(Y) - \mathrm{H}(X,Y), \quad (1)$$

where $\mathrm{H}(X)$ and $\mathrm{H}(Y)$ are the entropies of random variables X and Y, respectively, and $\mathrm{H}(X,Y)$ is the joint entropy of the two random variables:

$$\mathrm{H}(X) = -\sum_{x\in X} p_X(x)\log p_X(x), \quad (2)$$

$$\mathrm{H}(Y) = -\sum_{y\in Y} p_Y(y)\log p_Y(y), \quad (3)$$

$$\mathrm{H}(X,Y) = -\sum_{x\in X}\sum_{y\in Y} p_{XY}(x,y)\log p_{XY}(x,y). \quad (4)$$

文中采用梯度法进行求解：

Algorithm 1 Automatic Calibration by maximization of MI

Input: 3D Point cloud $\{\mathbf{P}_i;\ i = 1, 2, \cdots, n\}$, Reflectivity $\{X_i;\ i = 1, 2, \cdots, n\}$, Image $\{I\}$, Initial guess $\{\mathbf{\Theta}_0\}$.
Output: Estimated parameter $\{\hat{\mathbf{\Theta}}\}$.
while $(\|\mathbf{\Theta}_{k+1} - \mathbf{\Theta}_k\| > THRESHOLD)$ **do**
  $\mathbf{\Theta}_k \rightarrow [\mathrm{R}\,|\,\mathbf{t}]$
  for $i = 1 \rightarrow n$ **do**
    $\tilde{\mathbf{p}}_i = \mathrm{K}[\mathrm{R}\,|\,\mathbf{t}]\tilde{\mathbf{P}}_i$
    $Y_i = I(\mathbf{p}_i)$
  end for
  Calculate the joint histogram: $\mathrm{Hist}(X,Y)$.
  Calculate the kernel density estimate of the joint distribution: $p(X,Y;\mathbf{\Theta}_k)$.
  Calculate the MI: $\mathrm{MI}(X,Y;\mathbf{\Theta}_k)$.
  Calculate the gradient: $\mathbf{G}_k = \nabla\,\mathrm{MI}(X,Y;\mathbf{\Theta}_k)$.
  Calculate the step size γ_k.
  $\mathbf{\Theta}_{k+1} = \mathbf{\Theta}_k + \gamma_k \frac{\mathbf{G}_k}{\|\mathbf{G}_k\|}$.
end while

完事了，很无聊，就是把斯坦福的目标函数改成熵函数，并进行求解。效果提升咋样，文中说很牛逼。不过澳大利亚悉尼大学的文章对上面的方法进行了改进<Automatic Calibration of Lidar and Camera Images using Normalized Mutual Information>,内容大同小异，就是增加了一个 Gradient Orientation Measure（GOM）来进行计算图像和激光点云梯度相关测度。

$$GOM = \frac{\sum_{j=1}^{n} |g(1,j)\cdot g(2,j)|}{\sum_{j=1}^{n} ||g(1,j)||\,||g(2,j)||}$$

悉尼大学还采用了一种创新方法，就是将图像进行变换为柱面图像，然后将点云投影到柱面图像中去。如图 8 所示。

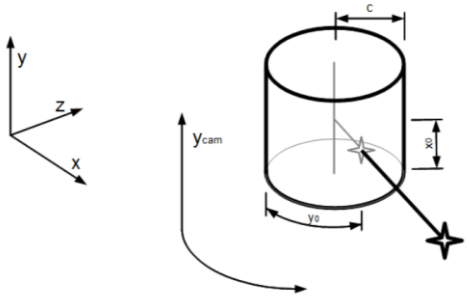


图 8 柱面图系

投影公式可以表示为：

$$x_{cam} = x_0 - \frac{cx}{z} + \Delta x_{cam} \tag{6}$$

$$y_{cam} = y_0 - \frac{cy}{z} + \Delta y_{cam} \tag{7}$$

$$x_{cam} = x_0 - c\arctan(\frac{-y}{x}) + \Delta x_{cam} \tag{8}$$

$$y_{cam} = y_0 - \frac{cz}{\sqrt{x^2 + y^2}} + \Delta y_{cam} \tag{9}$$

where
$\mathbf{x}_{cam}$, $\mathbf{y}_{cam}$ are the x and y position of the point in the image.
x, y, z are the coordinates of points in the environment.
c is the principle distance of the model.
$\mathbf{x}_0$, $\mathbf{y}_0$ are the location of the principle point in the image.
$\Delta\mathbf{x}$, $\Delta\mathbf{y}$ are the correction terms used to account for several imperfections in the camera.

投影完之后，就是求解点云梯度的方法了。如下：

Let
$\boldsymbol{r^i(t)}$ be the position of particle i at time t
$\boldsymbol{v^i(t)}$ be the velocity of particle i at time t
$\boldsymbol{p_n^{i,L}}$ be the local best of the ith particle for the nth dimension
$\boldsymbol{p_n^g}$ be the global best for the nth dimension
$\boldsymbol{n \in 1, 2, ...N}$
$\boldsymbol{t}$ is the time
$\boldsymbol{\Delta t}$ is the time step
$\boldsymbol{c_1}$ and $\boldsymbol{c_2}$ are the cognitive and social factor constants
$\boldsymbol{\phi_1}$ and $\boldsymbol{\phi_2}$ are two statistically independent random variables uniformly distributed between 0 and 1
$\boldsymbol{w}$ is the inertial factor

for *each itertaion l* **do**
  if $f(r^i(l+1)) > f(p^{i,L}(l))$ **then**
    $p^{i,L}(l+1) = r^i$
  end
  if $f(r^i(l+1)) > f(p^g(l))$ **then**
    $p^g(l+1) = r^i$
  end
  $v_n^i(t+\Delta t) =$
  $wv_n^i(t) + c_1\phi_1[p_n^{i,L} - x_n^i(t)]\Delta t + c_2\phi_2[p_n^g - x_n^i(t)]\Delta t$
  $r_n^i(t+\Delta t) = r_n^i(t) + \Delta t v_n^i(t)$
end

根据给出的公式，可以看出，所谓的标定就是求解 GOM 最大的过程。

上面讲的很多都是利用激光点云向摄像头的一个投影或者映射，其实也有很多基于深度学习的方法来进行传感器的标定。为啥采用深度学习呢？图像、点云都是可以用 CNN 来进行特征提取的，而联合标定就是依赖图像和点云来进行坐标系变换参数的求解，所以就有采用 CNN 来回归传感器坐标系转换参数。这里比较有名的就是 CalibNet。其实还有一篇 RegNet（与何大神貌似没啥关系，大神有一篇 RegNetX，有时也称为 RegNet，用来做检测的）。

RegNet<RegNet: Multimodal Sensor Registration Using Deep Neural Networks>应该是第一个使用 CNN 来推断多传感器的 6 自由度外参标定方法。其实这篇文章的最初思路来源于两张图像的配准（采用雅可比矩阵学习网络的方法来进行，有兴趣的话，可以自行翻阅）。RegNet 将标定的三个步骤（特征提取、特征匹配和全局回归）映射到单个实时 CNN 模型中，在训练中，随机对系统进行调整，可以使得 RegNet 来推断出点云投影到相机的深度测量与 RGB 图像之间的对应关系，并最终回归标定外参数。另外，通过迭代执行多个 CNN，在不同程序的 decalibration 数据上进行训练。如下图解释。

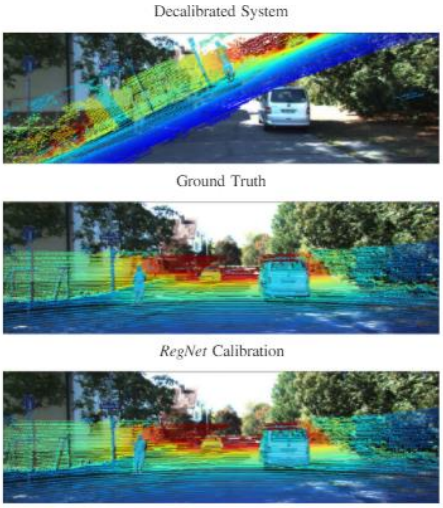


Fig. 1: *RegNet* is able to correct even large decalibrations such as depicted in the top image. The inputs for the deep neural network are an RGB image and a projected depth map. *RegNet* is able to establish correspondences between the two modalities which enables it to estimate a 6 DOF extrinsic calibration.

具体呢，先将传感器的坐标系中的点 x 转换为世界坐标系中的点 y，定义一个仿射变换矩阵 H，则有$y = Hx$。估计变换矩阵 H 的任务就称为外标定。估计变换矩阵 H 的任务称为外标定。应用深度学习，需要重新定义外标定的问题，在给定初始标定 Hinit 和基础事实标定 Hgt 的情况下，确定失标定矩阵 φdecalib，其定义如下：

$$\phi_{\text{decalib}} = \begin{bmatrix} & \boldsymbol{R} & & \boldsymbol{t} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

然后可以随机改变 Hinit 以获得大量的训练数据。为了能够建立标定过程可观测的对应关系，用 Hinit 和摄像头内参数矩阵 P 将激光雷达点投影在摄像头图像平面上，即

$$z_c \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \boldsymbol{P}\ \boldsymbol{H}_{\text{init}}\ \boldsymbol{x}\ .$$

在每个像素（u，v），如果没有投射的激光雷达点，则存储投影点的逆深度值（摄像头坐标）zc 或者为零。 由于相比图像像素的数量大多数常见的激光雷达传感器仅提供少量测量数据，因此深度图像非常稀疏。为了对付这种稀疏性，在输入深度图使用最大值池化（Max Pooling）对投影的激光雷达深度点上采样。

叨叨了这么多，RegNet 全貌改出来现身了。如下图。

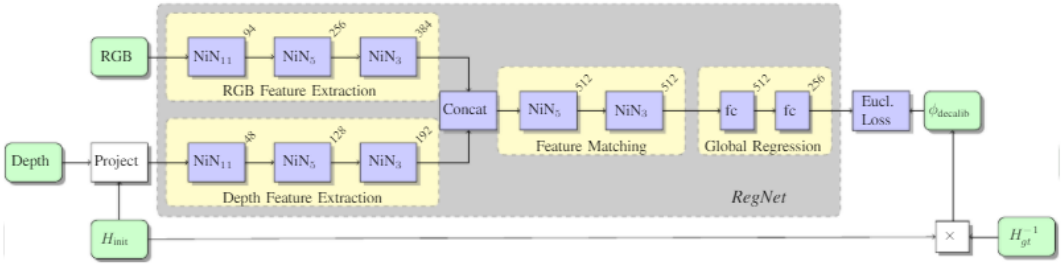
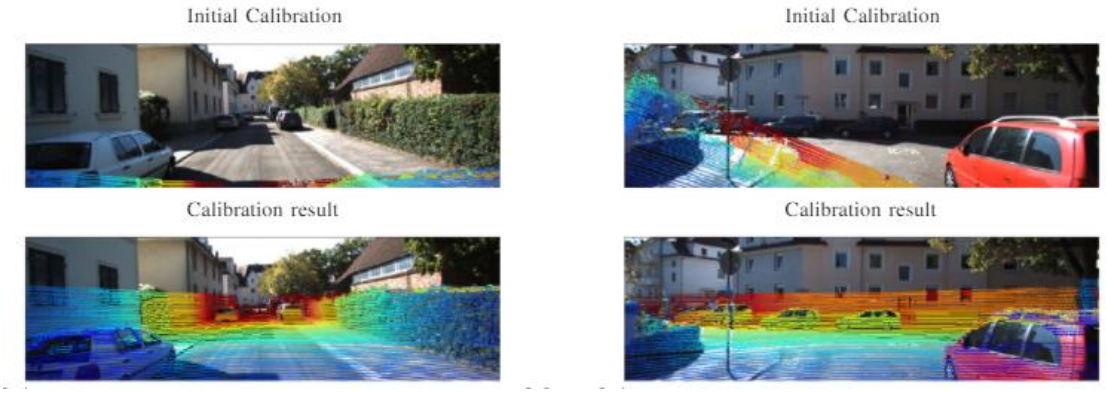


Fig. 2: Our method estimates the calibration between a depth and an RGB sensor. The depth points are projected on the RGB image using an initial calibration $\boldsymbol{H}_{\text{init}}$. In the first and second part of the network we use NiN blocks to extract rich features for matching. The kernel size k of the first convolutional layer of the NiN block is displayed by the indices. The number of feature channels is shown in the top right corner of each layer module. The final part regresses the decalibration by gathering global information using two fully connected layers. During training ϕ_{decalib} is randomly permutated resulting in different projections of the depth points.

使用初始标定 Hinit 将深度点投影在 RGB 图像上。在 CNN 网络的第一和第二部分，使用 NiN（Network in Network）块来提取丰富的特征以进行匹配，其中索引显示 NiN 块的第一卷积层的核大小 k。 特征通道的数量显示在每个模块的右上角。 CNN 网络最后一部分通过使用两个全连接层收集全局信息来对失标定进行回归。（注：NiN 块由一个 k×k 卷积，然后是几个 1×1 卷积组成。）在训练期间，失标定矩阵会被随机排列，形成深度点的不同投影数据。

Fig. 3: We deviate the initial calibration up to 20° in rotation and up to 1.5 m in translation from the ground truth calibration. This might result in projections of the LiDAR points where most of the points are outside the image area and it is therefore difficult to establish correspondences with the RGB image.

如上图，深度点的投影随给定的初始标定值而强烈地变化。当初始校准从标定的基础事实（GT）旋转偏离 20°平移偏离 1.5 米的时候，可能导致激光雷达点云的投影的大多数点在图像区域之外，难以与 RGB 图像建立对应关系。即使在这些情况下，训练的 CNN 网络仍然能够改进标定。使用新的估计标定参数可以再次投影深度点，从而产生更多供相关计算的深度点。然后，该步骤多次迭代即可。



哔哔完 RegNet，再来哔哔 CalibNet。这个网络深受 RegNet 的影响。

CalibNet 是一个自监督的深度网络，能够实时的自动估计激光与相机之间的 6 自由度刚体转换关系。这篇文章有一个特点就是不直接回归标定参数，但却可以训练网络去预测标定参数（好神奇呀），以最大化输入图像和点云的几何与光度一致性。二话不说，直接贴图。这张图就是 CalibNet 的流程图。其中 a 是来自待标定相机的 RGB 图像；b 是点云作为输入，并输出最佳对齐两个输入的 6 自由度刚体变换 T；c 是错误标定设置的彩色点云输出；d 是使用 CalibNet 网络标定后的结果图。

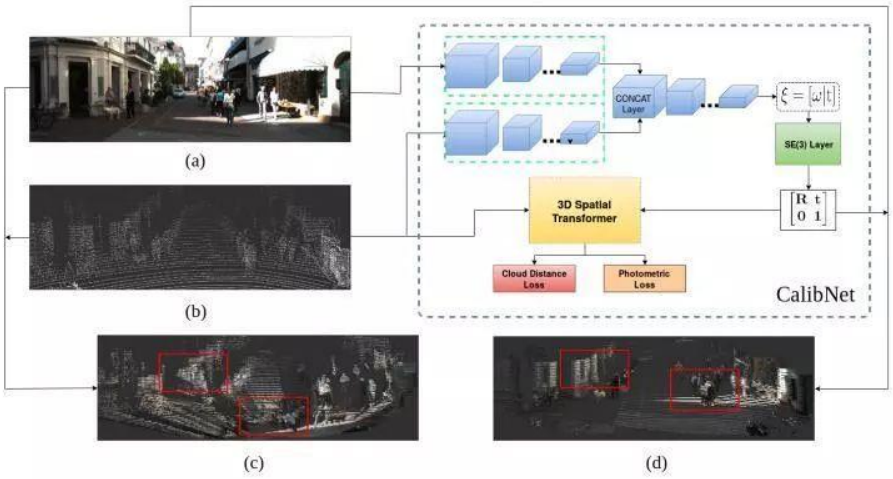
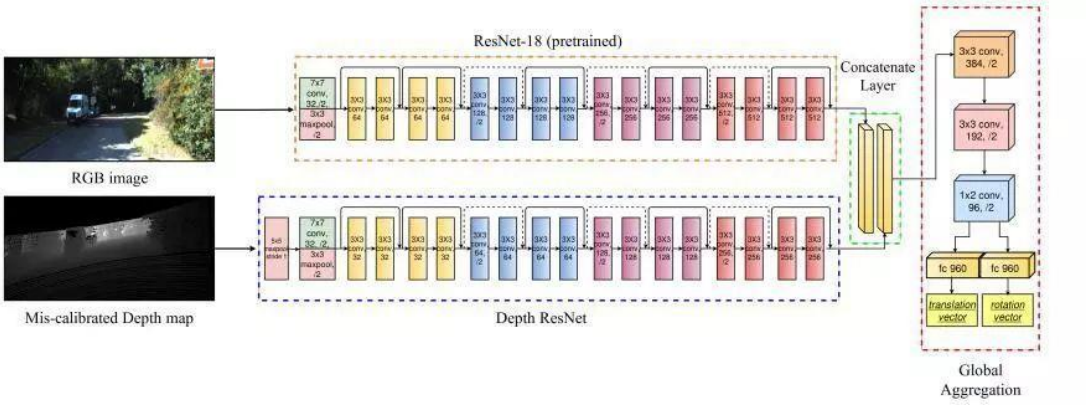


Fig. 1: CalibNet estimates the extrinsic calibration parameters between a 3D LiDAR and a 2D camera. It takes as input an RGB image (a) from a calibrated camera, a raw LiDAR point cloud (b), and outputs a 6-DoF rigid-body transformation T that *best* aligns the two inputs. (c) shows the colorized point cloud output for a mis-calibrated setup, and (d) shows the output after calibration using our network. As shown, using the mis-calibrated point cloud to recover a colorized 3D map of the world results in an incoherent reconstruction. Notice how the 3D structures highlighted in (c) using red rectangles fail to project to their 2D counterparts. However, using the extrinsic calibration parameters predicted by CalibNet produces more consistent and accurate reconstructions (d), even for large initial mis-calibrations.

从上图可以看出，该网络是将 RGB 图像、相应的误标定(mis-calibration)的激光点云（为啥有个误标定呢，时间同步、空间同步）和相机标定矩阵 K 作为输入。先将点云转换为稀疏的深度图，然后将点云投影到图像平面即可。由于误标定的存在，将误标定的点投影到图像平面上会导致稀疏深度图与图像很不一致。为了减少这一现象，采用 Calibnet 将 RGB 输入图像和稀疏深度图标准化为±1 的范围，然后使用 5*5 最大池化窗将系数深度图进行预处理,以创建半密度深度图。



从上面的网络结构图中可以看到其网络是由两个不对称分支组成。针对 RGB 图像，采用的 ResNet18 来作为 backbone，针对点云数据，先输入到 Depth ResNet 中，从结构图上看与 ResNet18 相类似，但每一个阶段的卷积核数量减少一半，上面 RegNet 介绍了 Depth ResNet 网络在特征提取上有优势。文章对于 RGB 输入使用预训练权重进行相关参数学习，但是对于深度图的参数，就需要从头开始学习，所以深度图的卷积核数量在每个阶段都会减少。最后将两个

分支的输出沿通道维级联，并通过一些列附加的全卷积曾，来进行全局特征聚合。每次卷积后，BN 都会被应用，将输出流分离成旋转和平移，以获取旋转和平移矩阵上的不同。该网络中有一个特殊层，涉及到李代数的相关知识，自行翻阅相关资料。其主要是将标定参数转换为李代数 SE(3)下的元素：变换矩阵 T、通过预测的变换矩阵 T，然后结合 3D 空间转化层，变换为输入的深度图。

该网络的损失函数有两部分组成：光度损失、点云损失。废话了一整篇的光度，这里大概提一下就是光强弱的大小。光度损失，顾名思义就是在利用预测值 T 转化深度图后，我们在预测的深度图和纠正的深度图之间检查了像素级误差（每一个像素值都用反射率(**depth intensity**)编码），其定义为：

$$\mathcal{L}_{photo} = \frac{1}{2}\sum_{1}^{N}\left(D_{gt} - KT\pi^{-1}[D_{miscalib}]\right)^2$$

点云损失主要指：尝试度量尺度最小化未校准的变换点和目标点云之间的 3D-3D 点距离。由于并不知道点集之间的相关性，因此考虑多种距离测量方法：

a) Chamfer 距离

$$d_{CD}(S_1, S_2) = \sum_{x \in S_1}\min_{y \in S_2}\|x-y\|_2^2 + \sum_{y \in S_2}\min_{x \in S_1}\|x-y\|_2^2$$

b) 推土机距离（Earth Mover's Distance）：

$$d_{EMD}(S_1, S_2) = \min_{\phi: S_1 \to S_2}\sum_{x \in S1}\|x - \phi(x)\|_2$$

where $\phi : S_1 \to S_2$ is a bijection.

c) 质心 ICP 距离

$$d_{icp}(S_1, S_2) = \frac{1}{2}\sum_{1}^{N}\sum_{1}^{i}\left\|X^i_{exp} - (RX^i_{miscalib} + t)\right\|^2 \quad (5)$$

最后，整个损失函数定义为：

$$\mathcal{L}_{final} = \alpha_{ph}(\mathcal{L}_{photo}) + \beta_{dist}(d(S_1, S_2))$$

下图是 CalibNet 标定的一些结果。第一行显示输入的 RGB 图像，第二行显示投影到图像上的相应的误标定的激光雷达点云。第三行显示使用网络预测变换投影的激光雷达点云，最后一行显示相应的基础事实结果。第二行中的红色框表示未对齐，而在第三行中，红色框表示标定后的正确对齐。

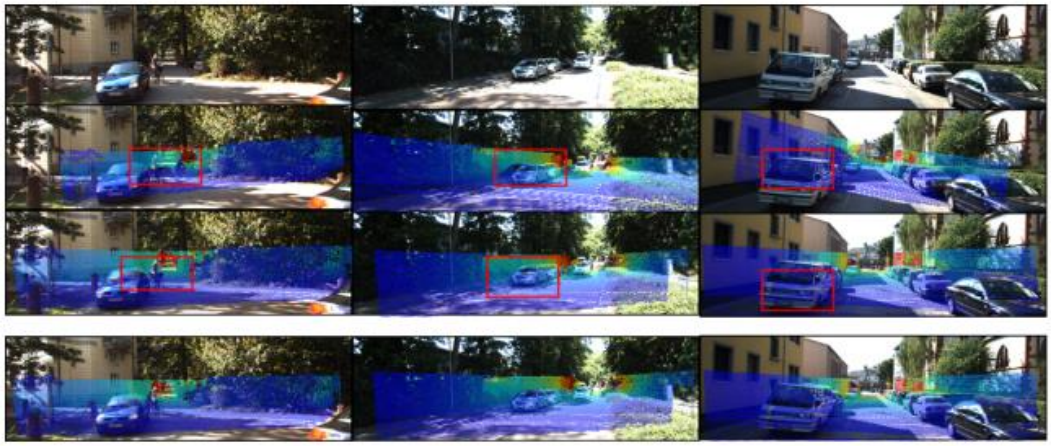


Fig. 4: Examples of calibration results for different scenes from the KITTI dataset. The first row shows the input RGB images, and the second row shows the corresponding mis-calibrated LiDAR point cloud projected onto the images. The third row shows LiDAR point clouds projected using the network's predicted transforms, and the last row shows the corresponding ground truth results. The red rectangles in the second row indicate the mis-alignment, and in the third row they denote the proper alignment after calibration.