本周的周报想写一下对这篇关于实时生成抓取的合成方法的RSS论文理解。以及自己的一些想法。

Paper

Closing the Loop for Robotic Grasping: A Real-time, Generative Grasp Synthesis Approach

论文概述

这篇论文主要讲述了作者提出了一种新的实时生成抓取的方法GG-CNN。这种来自深度图像的一对一映射通过避免抓取候选的离散采样和较长的计算时间，克服了当前深度学习抓取技术的局限性：这种轻量级的GG-CNN方法允许物体在移动的过程中或者非传统静态状态下精准地抓取物体。在真实的测试中，对于不常见的物品抓取成功率达到了83%，对于家用的物品成功率达到88%，在动态杂乱的环境中抓取物品成功率达到81%。

论文工作

抓取位姿定义

定义一次抓取grasp，作者将其定义为 $g = (p, \phi, \omega, q)$，其中：

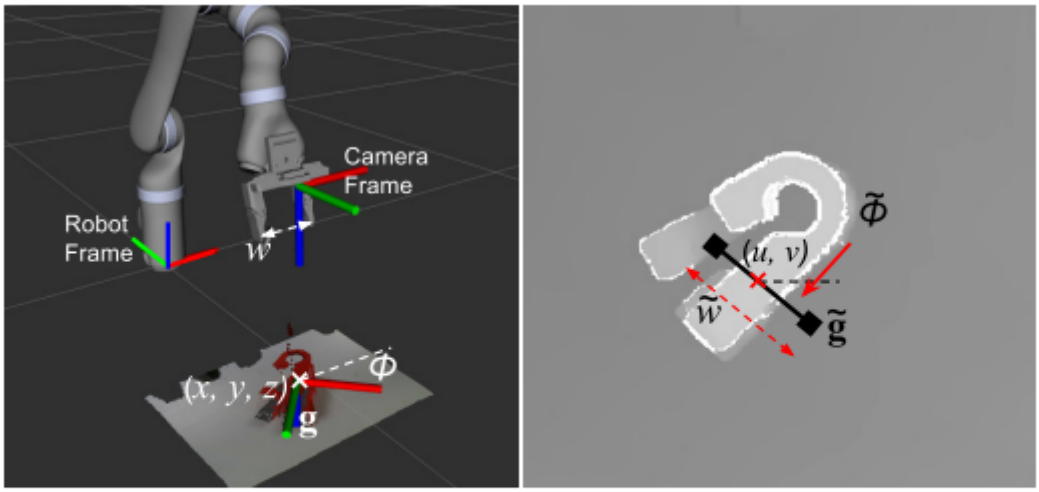


Fig. 2. Left: A grasp $\mathbf{g}$ is defined by its Cartesian position (x, y, z), rotation around the z-axis ϕ and gripper width w required for a successful grasp. Right: In the depth image the grasp pose $\tilde{\mathbf{g}}$ is defined by its centre pixel (u, v), its rotation $\tilde{\phi}$ around the image axis and perceived width $\tilde{w}$.

左图是3D图，右侧是深度图。

个人理解 1

在我看来，作者在深度图上花心思是有以下几个原因：

- 对于闭环的抓取位姿控制，可以利用深度图像不断地反馈给机器当前的图像，机器根据反馈能够更好的调整当前位姿。

- 这种深度图，从人为的角度去考虑，可以更好的展现出抓取一个物体时，从哪里入手，对于右图来讲很容易从视觉图上判断哪里是方便机器去抓取的。
- 深度图对于移动的物体，不需要精确的照相机对物品的抓取位置进行校准。

论文中作者特色工作

作者要利用深度图去选择抓取的位姿，因此在深度图中，一次抓取被定义为：$\widetilde{g} = (s, \widetilde{\phi}, \widetilde{\omega}, q)$

其中对于每一项的解释是：

- s = (u, v) is the centre point in image coordinates
- $\widetilde{\phi}$ is the rotation in the camera's reference frame
- $\widetilde{\omega}$ is the grasp width in image coordinates

对于 g 和 $\widetilde{g}$ ，有公式 $g = t_{RC}(t_{CI}(\widetilde{g}))$ 可以进行相互转换。g 是真实世界下的抓取位姿，$\widetilde{g}$ 是深度图中的抓取位姿。t_{RC} 是从camera frame转换为world/robot frame，t_{CI} 是从2D深度图转换到3D的camera frame。基于以上的定义，作者给出了一个集合grasp map：$G = (\Phi, W, Q) \in \mathbb{R}^{3\times H\times W}$ 其中 Φ, W, Q 是针对于每一个像素来讲的。

为了计算深度图中每一个像素点的 $\widetilde{g}$，作者定义了function M (from depth to grasp map in the image coordinates)，$M(I) = G$，从 G 中，我们可以计算出来最好的grasp coordinate：$\widetilde{g}^* = \underset{Q}{max}\, G$，从而计算出来真实世界中，最好的抓取坐标。

作者后文做的所有工作就是利用神经网络去表达复杂的方程 M。

变量说明

- Q is an image which describe the quality of a grasp executed at each point (u, v). The value is a scalar in the range $[0, 1]$
- Φ is an image which describe the angle of a grasp to be executed at each point. $[-\pi/2, \pi/2]$
- W is an image which describe the gripper width of a grasp to be executed at each point. $[0, 150]$

数据集说明

Training Dataset

作者所用的数据集是Cornell Grasping Dataset（我在Kaggle上已经下载下来了，下周准备利用自己改进的网络去重新跑一遍），数据集中包含了885个RGB深度图像（都是真实物体的），并且伴有5110 human-labelled positive 和 2909 negative 个抓取。

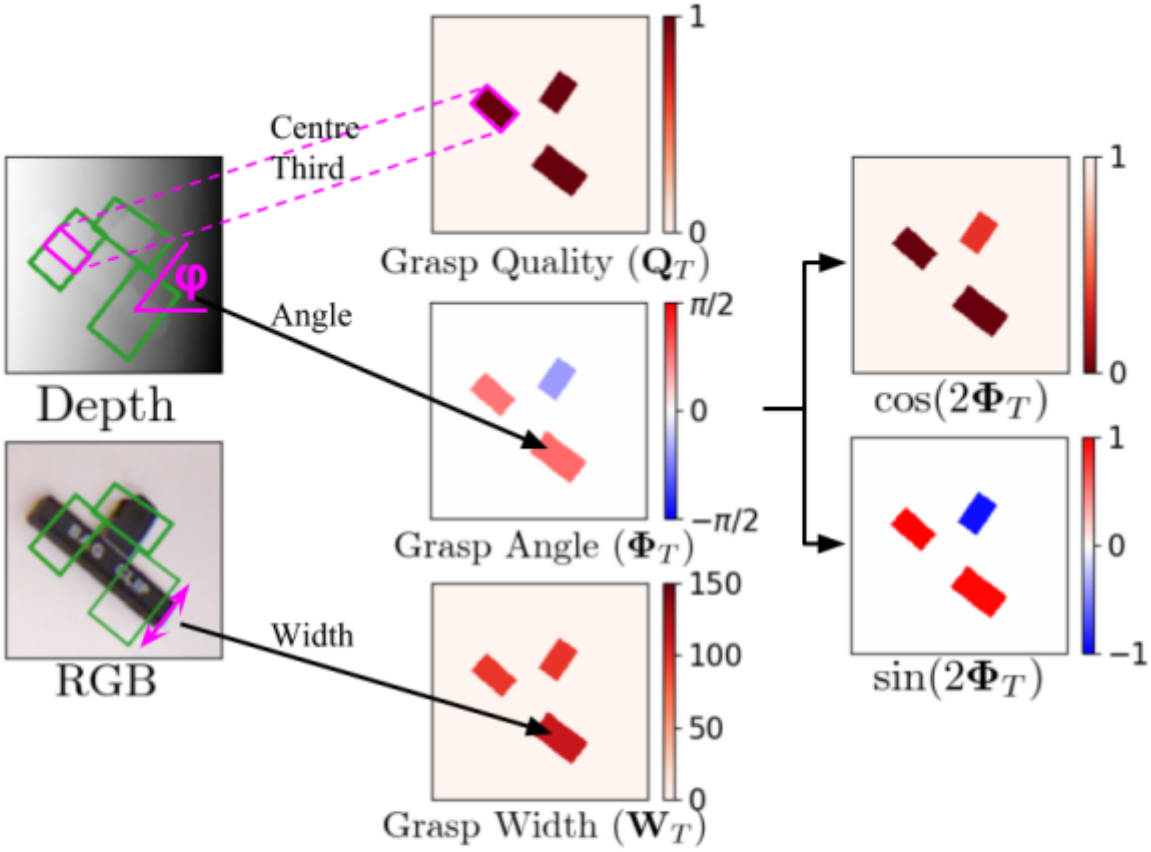


Fig. 3. Generation of training data used to train our GG-CNN. Left: The cropped and rotated depth and RGB images from the Cornell Grasping Dataset [17], with the ground-truth positive grasp rectangles representing antipodal grasps shown in green. The RGB image is for illustration and is not used by our system. Right: From the ground-truth grasps, we generate the Grasp Quality ($\mathbf{Q}_T$), Grasp Angle ($\mathbf{\Phi}_T$) and Grasp Width ($\mathbf{W}_T$) images to train our network. The angle is further decomposed into $\cos(2\mathbf{\Phi}_T)$ and $\sin(2\mathbf{\Phi}_T)$ for training as described in Section IV-B.

个人理解 2 + 下周工作

数据集是做深度学习任务中非常重要的一个部分，5110和2909的数据在做CV任务时是非常少的，但是作者在小型数据集上能够得到比较高的准确率一定是得益于网络架构非常优秀（虽然作者做了**数据增强**，对数据集进行crop, zoom rotations的操作，但是数据量还是相对较小），但是在我去github上下载论文的代码时发现论文中构建的神经网络较为庞大，参数量较多（最终的参数量为62420），因此我打算采用**知识蒸馏**的方法压缩模型，使student model能够覆盖teacher model的解空间并且使模型规模变小。

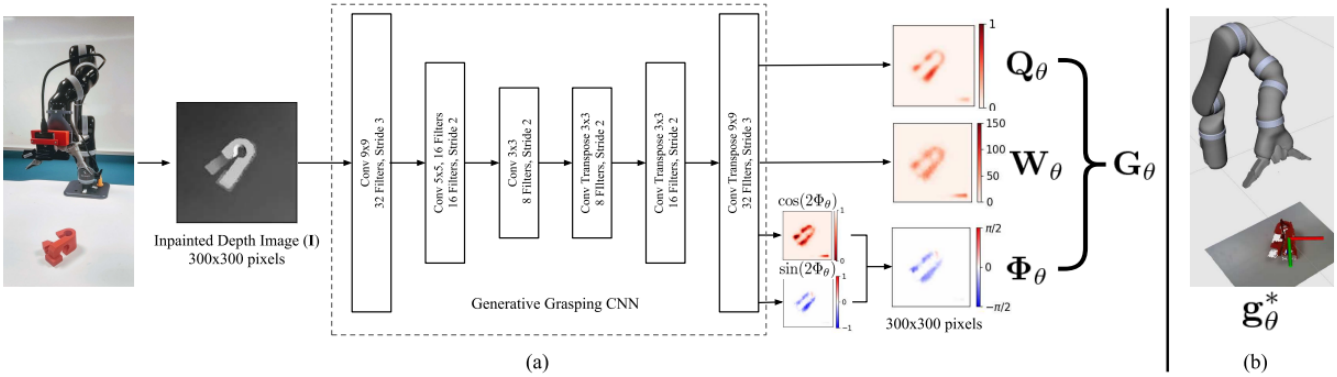


Fig. 4. (a) The Generative Grasping CNN (GG-CNN) takes an inpainted depth image ($\mathbf{I}$), and directly generates a grasp pose for every pixel (the *grasp map* $\mathbf{G}_\theta$), comprising the grasp quality $\mathbf{Q}_\theta$, grasp width $\mathbf{W}_\theta$ and grasp angle $\mathbf{\Phi}_\theta$. (b) From the combined network output, we can compute the best grasp point to reach for, $\mathbf{g}^*_\theta$.

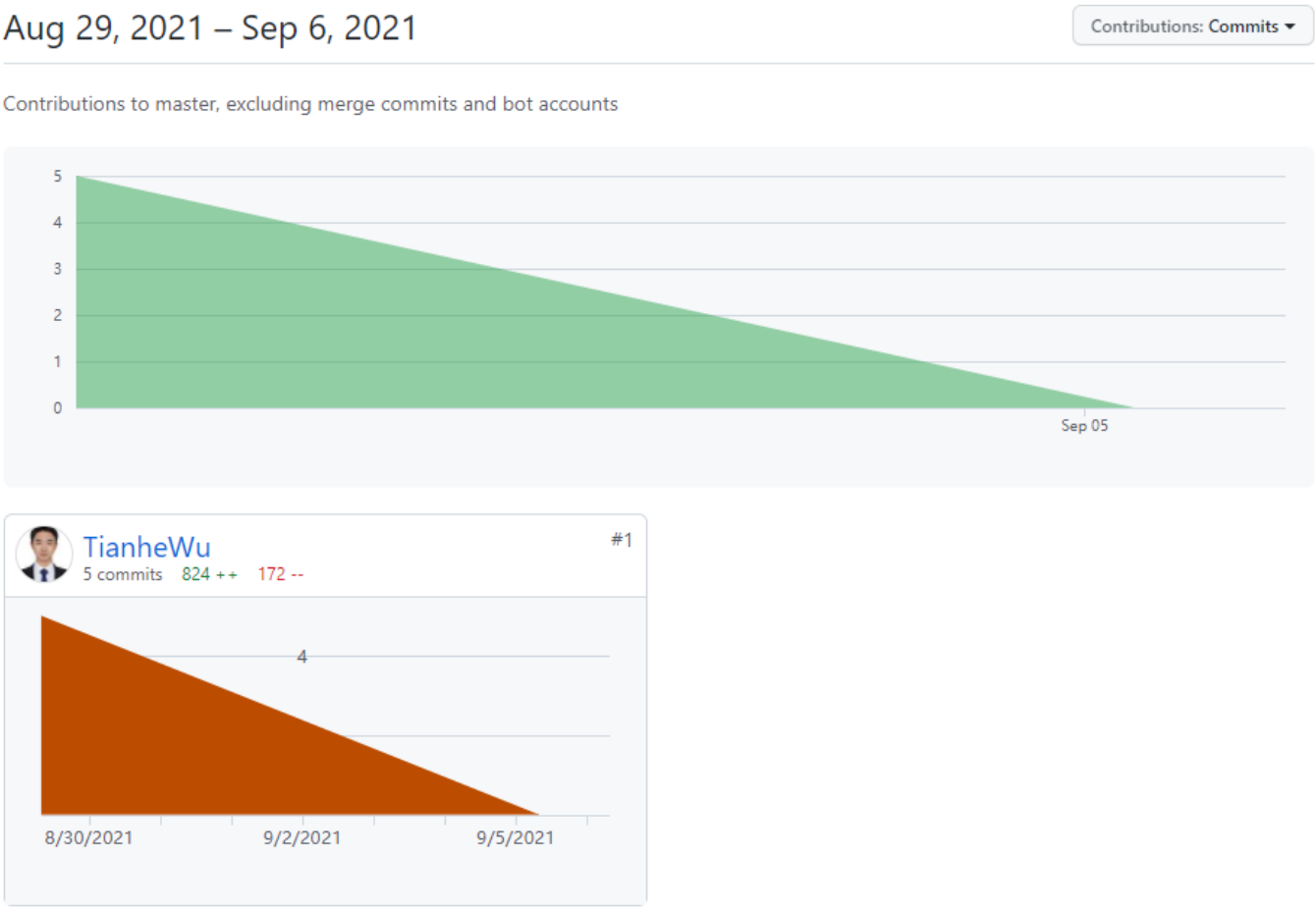
上图是作者构建的神经网络架构图，下图是对应在pytorch中搭建的网络。

```
def __init__(self, input_channels=1):
    super().__init__()
    self.conv1 = nn.Conv2d(input_channels, filter_sizes[0], kernel_sizes[0]
    self.conv2 = nn.Conv2d(filter_sizes[0], filter_sizes[1], kernel_sizes[1
    self.conv3 = nn.Conv2d(filter_sizes[1], filter_sizes[2], kernel_sizes[2
    self.convt1 = nn.ConvTranspose2d(filter_sizes[2], filter_sizes[3], kerne
    self.convt2 = nn.ConvTranspose2d(filter_sizes[3], filter_sizes[4], kerne
    self.convt3 = nn.ConvTranspose2d(filter_sizes[4], filter_sizes[5], kerne

    self.pos_output = nn.Conv2d(filter_sizes[5], 1, kernel_size=2)
    self.cos_output = nn.Conv2d(filter_sizes[5], 1, kernel_size=2)
    self.sin_output = nn.Conv2d(filter_sizes[5], 1, kernel_size=2)
    self.width_output = nn.Conv2d(filter_sizes[5], 1, kernel_size=2)

    for m in self.modules():
        if isinstance(m, (nn.Conv2d, nn.ConvTranspose2d)):
            nn.init.xavier_uniform_(m.weight, gain=1)
```

下周我会对GG-CNN中的model进行改进。并利用新模型重新训练测试，与作者的测试结果进行比较。

实习内容 Scrapy 爬虫

本周我也学习了scrapy爬虫框架对新浪和搜狐新闻进行爬取。最大的收获是掌握了一个新的爬虫技术，之前只会写简单的爬虫，对动态渲染的页面中的html内容是爬取不下来的，但是Scrapy爬虫框架可以爬取任意想要的页面html内容。虽然框架复杂，但是爬取效果非常好。



上图是coding代码量内容。

其实，学会爬虫对做深度学习也是有一定帮助的，有一些数据集很分散，需要我们人为手动寻找。但是还是要把重心放在掌握基础理论知识，多思考，多去想论文中有哪些可以创新的地方。