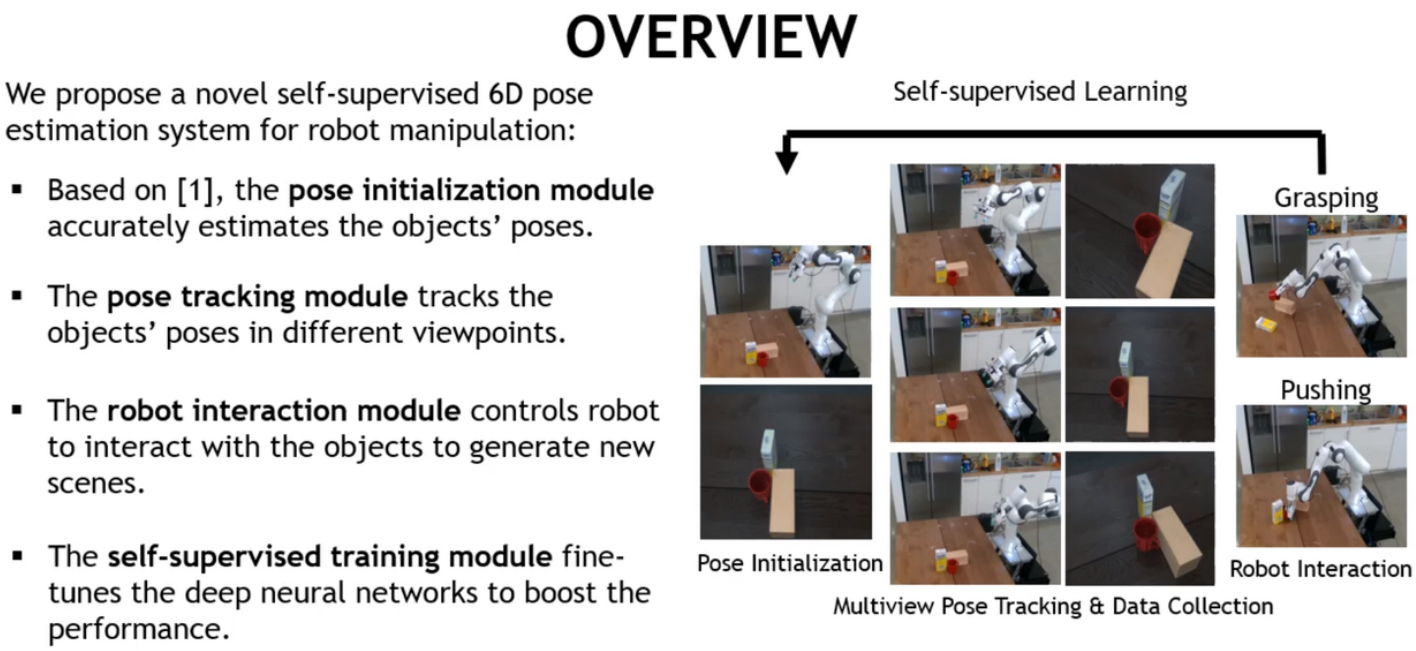
Self-supervised 6d object pose estimation for robot manipulation

由于真实的6d标记数据是 time consuming and expensive，文中提出了一个自监督的标记系统。self-supervised 6d object pose estimation sysem 包含pose initialization module，pose tracking module，robot interaction module，和self-supervised training module四个模块。

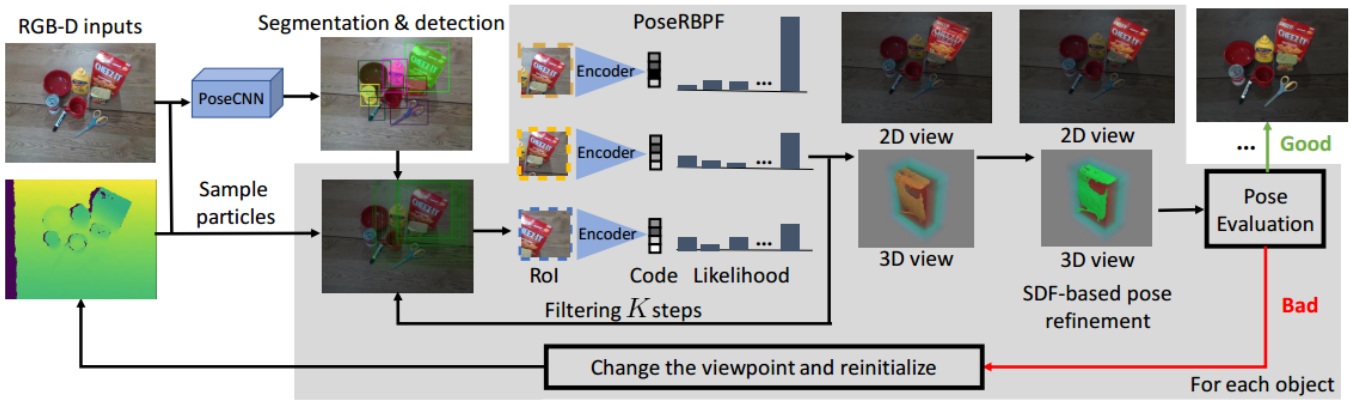


[1] Deng, Xinke, et al. "PoseRBPF: A Rao-Blackwellized Particle Filter for 6D Object Pose Tracking." In Robotics: Science and Systems (RSS) 2019

##实现细节 在该系统中，摄像头固定在机械臂末端，方便采集不同视角的图片。文中使用的相机为Intel RealSense D415。在采集开始时，机械臂自动运动到物体上方，以较高的elevation angle去拍摄物体，避免遮挡。采集的数据为RGB-D图像。

The Pose Initialization Module

初始姿态估计流程：image-->mask-->pose-->refine-->varify

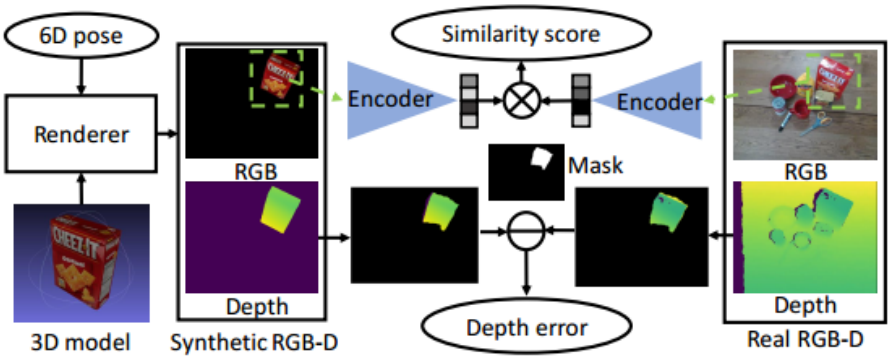


对于采集到的RGB-D图像，首先需要对目标物体物体进行分割和识别。文中使用PoseCNN方法，可以实现像素级别的分割，而且相对YOLO、SSD可以获取更精确的物体中心坐标。 在获取物体的mask后，使用 PoseRGPF估计R和T，然后使用

Signed Distance Function(SDF)优化R和T.

$$(\hat{\mathbf{t}}, \hat{\mathbf{R}}) = \arg\min_{\mathbf{t},\mathbf{R}} \sum_{\mathbf{p}_i \in \mathbf{P}_{\text{obj}}} |\mathbf{SDF}_{\text{obj}}(\mathbf{p}_i, \mathbf{t}, \mathbf{R})| + \lambda \frac{1}{2} \|\mathbf{t} - \bar{\mathbf{t}}\|^2$$

在得到R，T之后，结合物体模型生成RGBD渲染图片，生成的渲染图与真实图片对比，在RGB和depth空间分别计算误差，如果误差过大，认为pose 估计失败，重新估计初始位姿。在计算过程中，先获取真实照片的mask（姿态预估计的时候得到），然后对渲染图和真实图的mask部分做了encoder，提取特征，最后对特征向量做了cosin距离的计算，作为估计姿态的误差。encoder部分可以使用CNN的特征层，比如vgg16, alexNet去掉分类层。



The Pose Tracking Module

在采集初始姿态后，移动机械臂采集不同视角的图片。这里作者没有直接利用机械臂的变换矩阵去计算新位姿，而是利用变换矩阵去计算particles，然后用PoseRBPF框架 去估计位姿。

The Robot Interaction Module

在采集完一个场景后，利用机械臂去push和grasp物品来扰乱物品，从而开始新场景数据的采集。这样就实现了数据的全自动采集。

The Self-Supervised Training Module

在数据采集之初， PoseRBPF 框架(包括PoseCNN 和encoder)只使用仿真数据训练（因为没有真实数据此时）。这种情况下，PoseRBPF 框架的分割和估计的效果都不会很好，因此作者参考curriculum learning idea，设计了一套引导训练的方法。首先使用只包含单个物品的简单场景，去采集、标记真实数据。然后使用这些数据去fine-tuning网络，然后用新的网络去采集

cluttered scene多个物体的数据，继续采集、训练过程，提成网络性能。

Model	Synth.	+20% Real	+40% Real	+60% Real	+80% Real	+100% Real
master_chef_can	69.3	88.7	92.8	91.9	91.6	**93.9**
cracker_box	84.7	92.8	93.0	**93.4**	93.0	**93.4**
sugar_box	83.0	92.0	92.0	92.4	92.5	**92.6**
tomato_soup_can	83.6	90.2	90.5	90.8	91.2	**91.4**
mustard_bottle	83.9	92.5	93.3	93.3	93.9	**94.2**
tuna_fish_can	42.3	90.1	90.2	**91.7**	**91.7**	91.6
pudding_box	61.6	85.7	84.6	85.9	**87.1**	87.0
gelatin_box	66.6	83.6	83.2	83.7	82.0	**84.6**
potted_meat_can	62.9	84.1	85.4	86.4	86.6	**88.6**
banana	79.8	87.3	88.2	89.0	89.0	**89.3**
pitcher_base	51.5	86.3	84.3	88.0	**89.7**	89.6
bleach_cleanser	57.9	89.3	92.1	93.3	90.2	**93.4**
bowl	69.8	90.4	92.5	93.2	94.5	**95.4**
mug	69.2	90.3	90.9	91.4	**92.0**	91.0
power_drill	66.1	84.4	87.3	88.0	87.5	**88.5**
wood_block	64.2	82.6	80.2	85.1	86.0	86.1
scissors	36.3	71.9	75.8	77.4	77.5	**78.9**
large_marker	55.5	73.8	75.6	76.2	75.4	**77.1**
extra_large_clamp	15.5	76.3	76.0	**79.1**	77.1	**79.1**
foam_brick	12.2	86.5	86.7	87.6	86.6	**88.9**
MEAN	60.8	85.9	86.7	87.9	87.8	**88.7**

从表中可以看到，只用仿真数据，网络的表现不是很好，在加入真实数据后，性能得到了巨大的提升，20%以上。这里使用的指标为F1，是分割和分类的metric。pose的metric为ADD和ADD-S。

Experiments

作者选取了YCB中的20个物品，在12个robot hours内，共采集了497 scenes, 6541 张RGB-D 图像 和 22,851 实体，其中训练集265 scenes, 3590 张图像，测试集232 scenes, 2951张图像。 这个采集效率应该是比较高的，作为对比YCB-Video dataset 仅有92个scenes，而 LabelFusion dataset 是138个scenes。 在这个过程中，人类很少去干预采集过程，除了：放置、更换物品到工作台上；当系统无法获取初始位姿时，rearrange物品。 然后做了一些抓取实验来验证数据采集、训练的效果。加入真实数据后，抓取成功率提升很大40%。

Grasp Trials	Success rate [%]	Avg. initialization time per object [s]	Avg. grasp duration [s]
synthetic data only	46.70	10.04 (std: 15.98)	21.27
synthetic + real data	**86.70**	**4.48** (std: 5.71)	**15.47**

思考

1. 单纯的算法无法完全保证数据的准确性，但是通过增加fine-tuning和验证算法可以有效改善。
2. 相机固定在机械臂上拍摄，可能存在物体姿态的分布偏移。在正常无序摆放过程中，可能很多姿态不会出现（物体平衡问题），但是在摆放过程可能会出现，导致数据的不均衡。

Reference

Deng X, Xiang Y, Mousavian A, et al. Self-supervised 6d object pose estimation for robot manipulation[J]. arXiv preprint arXiv:1909.10159, 2019. BibTex: @article{deng2019self, title={Self-supervised 6d object pose estimation for robot manipulation}, author={Deng, Xinke and Xiang, Yu and Mousavian, Arsalan and Eppner, Clemens and Bretl, Timothy and Fox, Dieter}, journal={arXiv preprint arXiv:1909.10159}, year={2019} }