附录：

工作任务书

1. 工作范围

本项目的研究目标为完成以下四个算子的开发，与51个PCL API的分析。

序号	算子名称	框架	可供参考的开源实现
1	MaxRoiPool	ONNX	https://github.com/onnx/onnx/blob/master/docs/Operators.md
2	ThresholdedRelu	ONNX	
3	Celu	ONNX	
4	CornerNet网络后处理	ONNX	https://github.com/princeton-vl/CornerNet

PCL API算子分析需求（开源参考链接 https://pointclouds.org/documentation/group__features.html）：

序号	API名称
1	class pcl::ShapeContext3DEstimation< PointInT， PointNT， PointOutT >
2	class pcl::BOARDLocalReferenceFrameEstimation< PointInT， PointNT， PointOutT >
3	class pcl::BoundaryEstimation< PointInT， PointNT， PointOutT >
4	class pcl::BRISK2DEstimation< PointInT， PointOutT， KeypointT， IntensityT >
5	class pcl::CRHEstimation< PointInT， PointNT， PointOutT >
6	class pcl::CVFHEstimation< PointInT， PointNT， PointOutT >
7	class pcl::DifferenceOfNormalsEstimation< PointInT， PointNT， PointOutT >
8	class pcl::ESFEstimation< PointInT， PointOutT >
9	class pcl::Feature< PointInT， PointOutT >
10	class pcl::FeatureWithLocalReferenceFrames< PointInT， PointRFT >
11	class pcl::FLARELocalReferenceFrameEstimation< PointInT， PointNT， PointOutT， SignedDistanceT >
12	class pcl::FPFHEstimation< PointInT， PointNT， PointOutT >
13	class pcl::GASDEstimation< PointInT， PointOutT >
14	class pcl::GASDColorEstimation< PointInT， PointOutT >
15	class pcl::GFPFHEstimation< PointInT， PointLT， PointOutT >
16	class pcl::GRSDEstimation< PointInT， PointNT， PointOutT >

17	class pcl::IntensityGradientEstimation< PointInT， PointNT， PointOutT， IntensitySelectorT >
18	class pcl::IntensitySpinEstimation< PointInT， PointOutT >
19	class pcl::MomentInvariantsEstimation< PointInT， PointOutT >
20	class pcl::Narf
21	class pcl::NormalEstimation< PointInT， PointOutT >
22	class pcl::OURCVFHEstimation< PointInT， PointNT， PointOutT >
23	class pcl::PFHEstimation< PointInT， PointNT， PointOutT >
24	class pcl::PrincipalCurvaturesEstimation< PointInT， PointNT， PointOutT >
25	class pcl::RangeImageBorderExtractor
26	class pcl::RIFTEstimation< PointInT， GradientT， PointOutT >
27	class pcl::RSDEstimation< PointInT， PointNT， PointOutT >
28	class pcl::SHOTEstimation< PointInT， PointNT， PointOutT， PointRFT >
29	class pcl::SHOTColorEstimation< PointInT， PointNT， PointOutT， PointRFT >
30	class pcl::SHOTLocalReferenceFrameEstimation< PointInT， PointOutT >
31	class pcl::SpinImageEstimation< PointInT， PointNT， PointOutT >
32	class pcl::UniqueShapeContext< PointInT， PointOutT， PointRFT >
33	class pcl::VFHEstimation< PointInT， PointNT， PointOutT >
34	bool pcl::computePointNormal (const pcl::PointCloud< PointT > &cloud， Eigen::Vector4f &plane_parameters， float &curvature)
35	void pcl::flipNormalTowardsViewpoint (const PointT &point， float vp_x， float vp_y， float vp_z， Eigen::Matrix< Scalar， 4， 1 > &normal)
36	bool pcl::flipNormalTowardsNormalsMean (pcl::PointCloud< PointNT > const &normal_cloud， pcl::Indices const &normal_indices， Eigen::Vector3f &normal)
37	PCL_EXPORTS bool pcl::computePairFeatures (const Eigen::Vector4f &p1， const Eigen::Vector4f &n1， const Eigen::Vector4f &p2， const Eigen::Vector4f &n2， float &f1， float &f2， float &f3， float &f4)
38	void pcl::getFeaturePointCloud (const std::vector< Eigen::MatrixXf， Eigen::aligned_allocator< Eigen::MatrixXf > > &histograms2D， PointCloud< Histogram< N > > &histogramsPC)

39	Eigen::MatrixXf pcl::computeRSD (const pcl::PointCloud< PointInT > &surface, const pcl::PointCloud< PointNT > &normals, const pcl::Indices &indices, double max_dist, int nr_subdiv, double plane_radius, PointOutT &radii, bool compute_histogram=false)
40	class pcl::octree::OctreePointCloud< PointT, LeafContainerT, BranchContainerT, OctreeT >
41	class pcl::octree::OctreePointCloudAdjacency< PointT, LeafContainerT, BranchContainerT >
42	class pcl::octree::OctreePointCloudChangeDetector< PointT, LeafContainerT, BranchContainerT >
43	class pcl::octree::OctreePointCloudDensity< PointT, LeafContainerT, BranchContainerT >
44	class pcl::octree::OctreePointCloudOccupancy< PointT, LeafContainerT, BranchContainerT >
45	class pcl::octree::OctreePointCloudPointVector< PointT, LeafContainerT, BranchContainerT, OctreeT >
46	class pcl::octree::OctreePointCloudSinglePoint< PointT, LeafContainerT, BranchContainerT, OctreeT >
47	class pcl::octree::OctreePointCloudVoxelCentroid< PointT, LeafContainerT, BranchContainerT >
48	class pcl::octree::OctreePointCloudSearch< PointT, LeafContainerT, BranchContainerT >
49	class pcl::octree::OctreeDepthFirstIterator< OctreeT >
50	class pcl::octree::OctreeBreadthFirstIterator< OctreeT >
51	class pcl::octree::OctreeFixedDepthIterator< OctreeT >

2. 定义

不涉及

3. 工作计划

3.1 乙方应在华南理工大学（“工作地点”），按照下表的各阶段开展协议工作。各阶段工作的详细计划、应交付的阶段性成果及验收标准如下所示：

Content 合作内容	Deliverables 交付件描述
MaxRoiPool、ThresholdedRelu、Celu、CornerNet网络后处理算子开发	1、IR原型定义代码 2、算子信息库代码 3、插件代码 4、算子实现代码 5、测试用例代码（包括UT、BBIT）

	6、测试报告 7、算子说明文档 8、算子设计文档，包括算法描述（分析过程）、性能设计等。
PCL API算子分析需求表中51个算子分析	1、算子分析文档

阶段	起始时间	结束时间	工作内容 Contents	工作目标 Objective	输出 Output
第一阶段	T	T+3个月	1.ThresholdedRelu、Celu算子开发 2.PCL API算子分析需求表中序号1到15的API分析	ThresholdedRelu、Celu算子开发、测试完成 PCL API算子分析需求表中序号1到15的API分析完成	ThresholdedRelu、Celu算子交付件： 1、IR原型定义代码 2、算子信息库代码 3、插件代码 4、算子实现代码 5、测试用例代码（包括UT、BBIT） 6、测试报告 7、算子说明文档 8、算子设计文档，包括算法描述（分析过程）、性能设计等。 PCL 库 API 算子分析交付件： PCL API算子分析需求表中序号1到15的API算子分析文档
第二阶段	T+3	T+6.5个月	1.MaxRoiPool、CornerNet网络后处理算子开发 2.PCL API算子分析需求表中序号16到51的API分析	MaxRoiPool、CornerNet网络后处理算子开发、测试完成 PCL API算子分析需求表中序号16到51的API分析完成	MaxRoiPool、CornerNet网络后处理算子交付件： 1、IR原型定义代码 2、算子信息库代码 3、插件代码 4、算子实现代码 5、测试用例代码（包括UT、BBIT） 6、测试报告 7、算子说明文档 8、算子设计文档，包括算法描述（分析

					过程）、性能设计等。 PCL 库 API 算子分析交付件： PCL API算子分析需求表中序号16到51的API算子分析文档

序号	交付件	验收标准	验收方法
1	MaxRoiPool、ThresholdedRelu、Celu、CornerNet网络后处理 算子交付件	1. **算子精度标准**：与该算子在 CPU 或 GPU 上已实现的开源代码相比，精度要求默认 FP16 为双千分之一，FP32 为双万分之一。 双千分之一：相对误差高于千分之一的点不多于总数据的千分之一； 双万分之一：相对误差高于万分之一的点不多于总数据的万分之一。 2. **运行性能标准**： 达标：1/Tmdc >= 90%*1/Tgpu 未达标：1/Tmdc < 90%*1/Tgpu 其中： Tmdc：在使用两个MDC610 AIcore情况下，单个算子单次运行耗时(ms) Tgpu：在使用GPU(JETSON AGX XAVIER)的情况下，现有算子经过TensorRT优化后，单个算子单次运行耗时 (ms) 若GPU(JETSON AGX XAVIER)上的TensorRT算子库中无优化版本，则使用原有算子开源实现。在此情况下，上述达标系数调整为100%。 3. **算子泛化能力**： 随机遍历不同输入shape下的算子精度达标。shape的各个维度大小在[1, 1024]之间。后处理算子输入的shape范围由模型的输入范围决定。 4. **单算子编译性能**：参考标准为 ms 级，最大不超过 200ms。 **编程规范**：满足甲方静态检查(pylint)和安全规范要求。	精度及性能： 将所开发的算子转换为om模型，在MDC610上使用两个AICORE进行模型推理，对比原算子在GPU上的运行情况，测试精度及性能是否达标。 泛化能力： 随机输入不同shape的数据，在MDC610上使用两个AICORE运行。 编译性能： 乙方提供编译性能数据，甲方复核。 编程规范： 甲方使用工具对算子实现代码、插件代码、测试用例进行静态检查和安全规范检查，并通过MDC专家组复核。 文档验收方法： 甲方验收小组对算子说明、设计文档进行审核。
2	PCL 库 API 算子分析交付件	1. **算子分析文档**	文档验收方法： 甲方验收小组对算子分析文档进行审核。