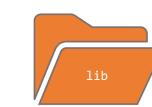
# Structure of RIGA code

benchmark\_registration\_

3dmatch.sh download\_data.sh → Execution of Python

scripts!





- calc\_fine\_matching\_loss(...)
- calc\_fine\_matching\_recall(...)
  def calc\_recall(...) def forward(...)

## tester.py

- b def \_load\_pretrain(...)

trainer.py

- def \_toau\_pretriam(...)

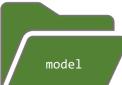
  def \_get\_lr(...)

  def stats\_dict(self)

  def stats\_meter(self)

  def inference\_one\_batch(...)

  def inference\_one\_epoch(...)
- def train(self)
  def eval(self)



- class RIGA(nn.Module)

- def forward(...)
- LearnableLogOptimalTransport(nn.Module)

  b def \_\_init\_\_(...)

  def
  - def
     log\_sinkhorn\_normalization(...)
     def forward(...)
     def \_\_repr\_(...)
    class NodeCorrespondenceSelector(...)
     def \_\_init\_\_(...)
     def forward(...)

benchmark.py

def extract\_corresponding\_trajectors(...)
def write\_trajectory(...)
def read\_pairs(...)

evaluate\_registration\_c2f.py

def extract\_correspondence(...)

def benchmark\_registration(...)

def rotation error(R1, R2)

def evaluate\_registration(...)
def benchmark(...)

def read trajectory(...) def read\_trajectory\_info(...)

modules.py

- - class crossAttention(nn.Module)

    > def \_init\_(...)

    > def forward(...)

    class PPFTrans(nn.Module)

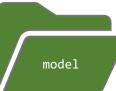
    > def \_init\_(...)

    > def forward(...)

# utils.py

- def setup\_seed(seed)
  def get\_correspondences(...)
  def matching\_descriptors(...)
- def sinkhorn(...)
  def interpolate(weights, points)
  def soft\_assignment(...)
  def get\_geometric\_structure\_embeddings(...

- class AverageMeter(object)
  class Timer(object)
  class Logger(object)



## ppftransformer.py

- def MLP(channels: list, do\_bn=True)
- class PointNetfeat(nn.Module)
  > def \_\_init\_\_(...)
  > def forward(...)
  > def attention(...)
- class MultiHeadAttention(nn.Module
- class CrossAttention(nn.Module)

# 



- def mutual\_selection(score\_mat)
  def get\_inlier\_ratio\_correspondence(.
- def get\_inlier\_ratio(...)
  def ransac\_pose\_estimation(...)
- ransac\_pose\_estimation\_corre
  def get\_scene\_split(...)
- def write\_est\_trajectory(...)

## evaluate\_registration\_c2 f\_rotated.py

def extract\_correspondence(...)
def benchmark\_registration(...)

functions...

# evaluate\_modelnet\_c2f.py

def benchmark\_evaluation(desc)

def benchmark\_evaluation(desc)

evaluate\_modelnet.py

# evaluate\_registration\_ro

def benchmark\_registration(...)

## def benchmark\_registration(...)

evaluate\_registration.py