InflationLayer没有自身的栅格地图要维护，直接在主地图上进行操作，它根据膨胀参数设置用来膨胀的“参考矩阵”，并在主地图上从障碍物出发，不断传播更新，完成对整个地图障碍的膨胀，等效于完成一个由“将机器人视为一个点”到“考虑机器人本身的足迹”的转变过程，防止因为忽视了足迹而碰上障碍。



#### 初始化 InflationLayer::onInitialize

这个函数没什么特别的，最重要的工作是调用了类内的matchSize函数。

#### 参考矩阵 InflationLayer::computeCaches & computeCosts

这里称cached\_distances\_和cached\_costs\_为参考矩阵，因为它们是后续膨胀计算的参照物。

最终计算出距离障碍物不同距离的点的代价值，这是在updatecosts中计算出的。

#### 更新膨胀地图边界 InflationLayer::updateBounds

need\_reinflation\_默认false，更新bound这里和其他两层的主要区别是，膨胀层在传入的bound的值的基础上，通过inflation\_radius\_再次扩张。

#### 更新膨胀地图代价 InflationLayer::updateCosts

用指针master\_array指向主地图，并获取主地图的尺寸，确认seen\_数组被正确设置。

#### “传播” InflationLayer::enquene

这个函数通过调用distanceLookup函数，在我们的参考矩阵cached\_distances\_上，找到当前cell与最近的障碍物cell的距离，当距离在阈值内，将当前cell加入到inflation\_cells\_相应键（表示距离）下。