输入：全局路径，代价地图

1. 机器人位置信息，假设为pose=[x0, y0, z0]=[0,0,0]
2. 当前速度vel=[vx0,vy0,vz0]=[0,0,0]
3. 目标点goalpose
4. Dwb参数设置：

自身速度限制 vel\_x=[0.0, 0.3]，vel\_y=[0.0, 0.0]，vel\_theta=[0.0, 0.6]，

speed\_xy=[0.0, 0.3]，

电机性能限制 acc\_lim\_x=0.5 ，acc\_lim\_y=0.0，acc\_lim\_theta=1.0 ，

样本数 vx\_samples=20，vy\_samples=5，vtheta\_samples=20

sim\_time=1.7，linear\_granularity=0.05，angular\_granularity=0.025

Step1: 计算速度样本

(1)在自身和电机限制下得到速度范围：cmd\_vel=[0, 0.3]∩[-0.5\*1.7,0.5\*1.7]=[0,0.3]

(2)在时间1.7s内划分步长，一般计算选择：

x方向上的速度步长=0.3/（20-1）=3/190，则x方向的20个速度样本为[0, 3/190, ..., 0.3]

同理：y始终为0，theta方向的速度样本为[0, 6/190, ..., 0.6]

1. 组成多种速度样本，随机选择一种，假设为[vx,vy,vth]=[12/190, 0 ,21/190]

Step2：选择速度样本计算局部移动位置，生成局部规划路径并对该路径进行评分

1. 计算时间步长

dt=min(1.7/[12/190\*1.7/0.05],1.7/[21/190\*1.7/0.025])=min(1.7/2,1.7/8)=0.2125

[]代表向上取整

1. 计算每一段dt时间的速度

因为当前速度0<采样速度12/190，所以x方向的速度cmd\_vx=min(12/190, 0+0.5\*0.2125)=12/190

（第二段dt的速度，因为当前速度=采样速度，所以x方向的速度=max(12/190,12/190-0.5\*0.2125)=12/190,所以会按当前速度不变)

y一直为0，theta同理

1. 计算第一步dt时间移动后的位置

X1=x0+(cmd\_vx\*cos(theta)-cmd\_vy\*sin(theta))\*dt

Y1=y0+(cmd\_vx\*sin(theta)+cmd\_vy\*cos(theta))\*dt

Theta1=theta0+cmd\_vtheta\*dt

1. 得到新的位置后，继续计算新的采样速度(2)，然后又得到新的位置(3)，最后就会得到局部的一些位置点生成的轨迹
2. 对轨迹进行评分

Step3：将所有速度样本进行step2操作，选择评分最好的一组速度样本

输出：最优的速度样本[v,w]

后续：将其发送给底盘左右两轮速度[vl,vr]=[v+wd,v-wd],d是两轮距离的一半

算法流程简述：

1. 根据平移速度最为目标速度得到速度样本，速度样本个数取决于速度样本数
2. 将每一条速度样本作为目标速度去计算评分，在计算评分之前需要进行3操作
3. 得到一个目标速度，并且有一个sim\_time,就会有一条局部路径，但是不是简单的直线运动，将局部路径划分多步，这个与粒度有关，每一小步有个速度，而且速度向平移速度靠拢，如果角速度不设置为0，则始终是走弧线的，但是需要转弯，所以必须有角速度，所以机器人实际上走的是弧线，每一步走的路径连起来就是局部路径
4. 对局部路径评分，每一个速度样本都会有一个评分，选择最优的路径
5. 将最优的best对应的第2步中的速度发步出来

超调原因分析：

1. 机械原因：轮子品质、机械加工与安装多多少少存在误差。（比如，买轮毂大小一样的车轮，两个电机减速箱也一样，两个轮子处于同一根轴，轮子也相互平行）
2. 算法的问题：在选择速度样本时，有可能得到存在角速度样本，此时移动偏离直线，此时下一段局部最优采样速度可能会进行矫正，重复该步骤，机器人移动也就是s型，也就会存在超调现象。
3. 定位原因：不明确当前位置与方向，靠近障碍物边缘移动更加安全？
4. 评分函数的影响？如果只算最短距离，不看障碍物是否会直线移动

参数详情：

min\_vel\_x: 0.0 用于设置X轴方向上的最小速度（单位为m/s）。

min\_vel\_y: 0.0 用于设置Y轴方向上的最小速度（单位为m/s）。

max\_vel\_x: 0.26 用于设置X轴方向上的最大速度（单位为m/s）。

max\_vel\_y: 0.0 用于设置Y轴方向上的最大速度（单位为m/s）。

max\_vel\_theta: 1.0 用于设置最大角速度（单位为弧度/秒即rad/s）。

min\_speed\_xy: 0.0 用于设置最小平移速度（单位为m/s）。

max\_speed\_xy: 0.26 用于设置最大平移速度（单位为m/s）。

min\_speed\_theta: 0.0 用于设置最小角速度（单位为rad/s）。

acc\_lim\_x: 2.5 用于设置X轴方向上的最大加速度（单位为m/s2）。

acc\_lim\_y: 0.0 用于设置Y轴方向上的最大加速度（单位为m/s2）。

acc\_lim\_theta: 3.2 用于设置最大旋转加速度（单位为rad/s2）。

decel\_lim\_x: -2.5 用于设置X轴方向上的最大减速度（单位为m/s2）。

decel\_lim\_y: 0.0 用于设置Y轴方向上的最大减速度（单位为m/s2）。

decel\_lim\_theta: -3.2 用于设置最大旋转（角度）减速度（单位为rad/s2）。

vx\_samples: 20 用于设置X速度方向上的速度样本数

vy\_samples: 5 用于设置Y速度方向上的速度样本数。

vtheta\_samples: 20 用于设置角度方向上的速度样本数。

sim\_time: 1.7 用于设置向前仿真的时间（单位为秒）。

linear\_granularity: 0.05 用于设置线性粒度，即向前投射的线性距离。

angular\_granularity: 0.025 用于设置角度粒度，即向前投射的角度距离。

transform\_tolerance: 0.2 用于设置TF坐标变换的容差（单位为秒）

xy\_goal\_tolerance: 0.25 用于设置满足目标完成标准的容差（单位为m）。

trans\_stopped\_velocity: 0.25 用于设置停止速度（单位为弧度/秒），当机器人速度低于该速度时会被认为满足容差而处于停止状态。

BaseObstacle.scale: 0.02

PathAlign.scale: 32.0 用于设置对齐路径评分插件的加权比例，会覆盖本地默认值。

GoalAlign.scale: 24.0 用于设置目标位姿对齐评分插件的加权比例，会覆盖本地默认值。

PathAlign.forward\_point\_distance: 0.1 用于设置机器人前方点的距离以用于计算角度变化。

GoalAlign.forward\_point\_distance: 0.1

PathDist.scale: 32.0 用于设置路径距离评分插件的加权比例，会覆盖本地默认值。

GoalDist.scale: 24.0 用于设置目标位姿距离评分插件的加权比例，会覆盖本地默认值。

RotateToGoal.scale: 32.0

RotateToGoal.slowing\_factor: 5.0 用于设置旋转到目标位姿时减慢机器人运动的因子。

RotateToGoal.lookahead\_time: -1.0 如果>0，则为等待碰撞的时间量。

调整参数方案：

1.加速度限制（acc\_lim\_x，acc\_lim\_y，acc\_lim\_th）非常的重要，如果不知道机器人的加速度，可以尽量的往大的设置，因为如果设置太小了，往往会出现机器人往前跑断断续续的，转弯转过头（看似加速度太大了，实际是加速度太小，以至于机器人想把机器人掰回来而掰不及），从而导致反复的震荡。（目前机器人没出现该问题）

2.sim\_time局部路径模拟的时间不用太长也不用太短，一般一秒左右就差不多了，如果太多会容易导致偏离全局的路径，特别启动的时候会转较大的半径，如果想启动的时候基本原地旋转摆正机器人的方向和全局路径的方向一致，那么就把模拟的时间设置短点。如果太小的仿真时间也不好，容易导致频繁的路径规划消耗资源，甚至也会出现震荡的可能。

3.PathAlign.scale 32刻画局部路径和全局路径的贴合程度，该权重参数越大说明越贴合，越小说明不用那么贴合。

4.GoalAlign.scale 24达到局部目标点的权重参数，也用来控制速度。权重如果设置为0表示要求完全到达目标点，这会导致机器人走动缓慢和震荡的现象，因为要求达到目标点的精度太高，所对机器人的控制比较苛刻

仿真解析：  
1.通过仿真打印评价分数以及发行速度，可以确定dwb算法没有问题，与理论一致

1. 可以尝试调整速度与加速度，使其符合电机标准
2. 仿真发现控制频率影响机器人移动稳定性
3. 样本数可以更改为合适的，减少计算复杂度
4. 控制频率5hz表示每秒发送5条速度指令给底盘，设置电机最高频率即可，频率越高计算能力要求越高
5. 粒度是用来计算局部规划的步长，一般加速度够大会达到样本速度，所以不会有什么改变，主要是用来检查是否存在障碍物
6. 代价图参数：inflflation 、resolution等

综上，dwb算法与理论一致，没有太大问题，速度和加速度根据电机设置。粒度和sim\_time协调设置计算时间步长。评分感觉没太大影响。样本数影响不是很大，只是用来寻找最优速度，样本数越多样本速度越多。其余的与超调没啥关系。所以主要是全局路径规划，而我们自己设置的是理想直线规划，那不是nav2上的全局路径规划。如果全局规划输入是一条直线，在没有障碍物影响下，该算法可以让他走直线。打印评分发现，在没有障碍物影响下，RotateToGoal、Oscillation、BaseObstacle这三个评分都为0，影响都在GoalAlign、PathAlign、PathDist、GoalDist这四个评分上

sim\_time是做局部规划，controller\_frequency是发布控制速度，5hz表示1秒发布5条速度，也就是说在机器人运动过程中，每0.2秒更新一次局部规划，同时每0.2秒更新一次发布速度。先做局部规划，找出最优轨迹，得到轨迹对应的速度，将其作为移动速度发布给机器人，每0.2s更新速度，计算速度用的轨迹时间还是sim\_time。