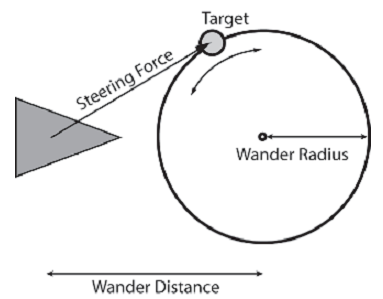
# 2.3 操控行为——Wander（徘徊）

当创建智能体的行为时，你经常会发现wander行为很有用。它产生一个操控力，使智能体在环境中随机走动。

一个幼稚的做法是每帧都计算出一个随机的驱动力。但这会产生抖动，不能达到持久的转弯（事实上，一个好的随机函数，Perlin噪声，可以产生光滑转弯，但是CPU的开销会很大。当然当你没有其他办法时，这仍然是个办法，Perlin噪声有很多应用程序）。

Reynolds的解决方案是在交通工具的前端凸出个圆圈，目标被限制在该圆圈上，然后我们移向目标。每帧给目标添加一个随机的位移，随着时间的推移，沿着圆周移来移去，以创建一个没有抖动的往复运动。利用不同的圆圈尺寸、到交通工具的距离、每帧的随机位移的大小，这个方法可以产生所有范围的随机运动，从非常光滑的波状式转弯到狂野的Strictly Ballroom式旋转，再到以脚尖立地的旋转。下图可帮助你更好地理解这个方法。



下面一步一步讲解代码。首先是wander使用的3个成员变量：

private float \_wanderJitter; // 每秒加到目标的随机位移的最大值

private float \_wanderRadius; // wander圆半径

private float \_wanderDistance; // wander圆凸出在智能体前面的距离

接下来是方法本身：

//--------------------------- Wander -------------------------------------

//

// 这个行为让智能体在环境中随机行走

//------------------------------------------------------------------------

public Vector2 Wander()

{

float JitterThisTimeSlice = \_wanderJitter \* \_vehicle.TimeElapsed;

// 首先，在目标位置上加一个随机向量

\_wanderTarget += new Vector2((((float)Global.random .NextDouble())\*2.0f-1.0f) \* JitterThisTimeSlice,(((float)Global.random .NextDouble())\*2.0f-1.0f) \* JitterThisTimeSlice);

// 将这个向量重新投影到单位圆周上

if(\_wanderTarget!=Vector2 .Zero )

\_wanderTarget.Normalize();

// 然后将这个向量的长度等于wander圆的半径

\_wanderTarget \*= \_wanderRadius;

// 移动目标到智能体前面WanderDist的位置

Vector2 targetLocal = \_wanderTarget + new Vector2(\_wanderDistance, 0);

// 对三角形的顶点进行变换

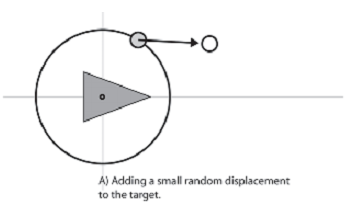
transformMatrix = Matrix.CreateScale (\_vehicle.BRadius )\*Matrix.CreateRotationZ((float)(Math.Atan2(\_vehicle.Heading.Y, \_vehicle.Heading.X)))\* Matrix.CreateTranslation(new Vector3(\_vehicle.Pos, 0));

targetWorld = Vector2.Transform (targetLocal,transformMatrix);

return targetWorld - \_vehicle.Pos;

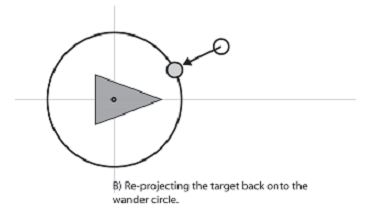
}

\_wanderTarget是一个点，被限制半径为\_wanderRadius的圆圈上，以交通工具为中心，我们每帧都给wander目标位置添加一个小的随机的位移，如图A所示。



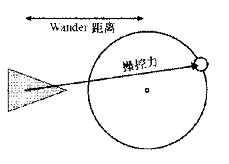
**A 给目标添加一个小的随机位移**

下一步，通过把归一化向量再乘上wander圈的半径，就可以把新的目标重新投影到wander圆上，如图B所示。



**B重新把目标投影到wander圆**

最后，新的目标移到交通工具的前面距离等于\_wanderDistance的位置，投影到世界空间。然后，计算向量移动到这个位置所需的操控力，如图C所示。



C 将目标投影到交通工具前面

看一下这个行为的示例程序。绿色的圆是限定的wander圈，红色的点是目标。这个示例程序允许调整wander圈的尺寸、抖动的数量、wander距离。所以你可以观察到它们不同的作用。注意wander距离和这个方法返同的操控力的角度变化之间的关系。当wander圆离交通工具很远时，这个方法使角度发生小变化，因此交通工具只能小转弯。当圆被移近交通工具时，它就可以大转弯了。

如果想让智能体在三维空间（像空中飞船在它的领地上巡逻）wander，你所要做的是限制wander圆为1个球体，而不是圆周。