# 16.2 世界/本地空间中的拾取射线

现在，我们已经知道了如何计算观察空间中的拾取射线，但是它的用途非常有限，因为只有当物体也在观察空间中时，我们才能使用该射线进行相交测试。我们知道，观察矩阵可以将几何体从世界空间变换到观察空间，所以它的逆矩阵可以将几何体从观察空间变换回世界空间。设**r**v(*t*) = **q** + *t***u**为观察空间中的拾取射线，**V**为观察矩阵，则世界空间中的拾取射线为：

**r**w(*t*) = **qV**-1+ *t***uV**-1 = **q**w + *t***u**w

注意，射线起点**q**是按照“点的方式”来变换的（即，**q**w = 1），而射线方向**u**是按照“向量的方式”来变换的（即，**u**w =0）。

世界空间中的拾取射线可以与世界空间中的物体进行相交测试。不过，在大多数情况下，一个物体中的几何体都是相对于该物体自身的局部空间来定义的。所以，我们必须把射线变换到物体的局部空间后再进行射线与物体之间的相交测试。设**W**为物体的世界矩阵，矩阵**W**-1可以将几何体从世界空间变换到物体的局部空间。则局部空间中的拾取射线为：

**r**L(*t*) = **q**W**W**-1+ *t***u**W**W**-1

通常，场景中的每个物体都有它自身的局部空间。 所以，必须把射线变换到每个物体的局部空间后再进行相交测试。

有些读者可能会想：是否可以把网格变换到世界空间，然后在世界空间中进行相交测试呢？方法可行，但不可取。因为这样的计算量会非常大。一个网格可能会包含几千个顶点，如果把些顶点逐个变换到世界空间，那代价会非常大。在效率上，这种方式远不如将一条射线变换到物体的局部空间更为高效。

下面的代码示范了如何将一条拾取射线从观察空间变换到一个物体的局部空间：

// 将设置转换到网格的本地空间

XMMATRIXV = mCam.View();

XMMATRIXinvView = XMMatrixInverse(&XMMatrixDeterminant(V),V);

XMMATRIXW = XMLoadFloat4x4(&mMeshWorld);

XMMATRIXinvWorld = XMMatrixInverse(&XMMatrixDeterminant(W),W);

XMMATRIXtoLocal = XMMatrixMultiply(invView,invWorld);

rayOrigin = XMVector3TransformCoord(rayOrigin,toLocal);

rayDir = XMVector3TransformNormal(rayDir,toLocal);

// 规范化射线方向用于相交测试

rayDir = XMVector3Normalize(rayDir);

**XMVec3TransformCoord**和**XMVec3TransformNormal**函数都是以3D向量作为参数。但是请注意，**XMVec3TransformCoord**函数将向量的第4个分量视为1；而**XMVec3TransformNormal**函数将向量的第4个分量视为0。所以，我们使用**XMVec3TransformCoord**函数变换点；使用**XMVec3TransformNormal**函数变换向量。