|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

**Calculating a XYZ-pos for a vector with rotation in XYZ**

[Reply](http://xboxforums.create.msdn.com/forums/AddPost.aspx?ReplyToPostID=188097&Quote=False) [Quote](http://xboxforums.create.msdn.com/forums/AddPost.aspx?ReplyToPostID=188097&Quote=True)

childImage.Position = parent.Position + parent.Scale \* Position\_Relative;

childImage.Scale = parent.Scale \* Size\_Relative;

if (parent.Rotation != Vector3.Zero)

{

childImage.Rotation = parent.Rotation + Rotation\_Relative; //This doesn't work yet

Vector3 posDiff = childImage.Position - parent.Position;

if (posDiff != Vector3.Zero)

{

Vector3 vectorRotation = Vector3.Zero;

float vectorLength = posDiff.Length();

vectorRotation.X = ((float)Math.Asin(posDiff.X / vectorLength) + parent.RotateX);

vectorRotation.Y = (float)Math.Asin(posDiff.Y / vectorLength) + parent.RotateY;

vectorRotation.Z = ((float)Math.Asin(posDiff.Z / vectorLength) + parent.RotateZ);

Matrix calcPos = Matrix.CreateFromYawPitchRoll(vectorRotation.X, vectorRotation.Y, vectorRotation.Z);

childImage.Position = calcPos.Forward \* vectorLength + parent.Position;

}

}

childImage.Scale = parent.Scale \* Size\_Relative;

childImage.QuatRotation = parent.QuatRotation \* Rotation\_Relative;

Vector3 vectorRotation = Vector3.Zero;

Matrix calcPos = Matrix.CreateFromQuaternion(parent.QuatRotation);

childImage.X = parent.X + calcPos.Right.X \* Position\_Relative.X \* parent.Scale.X;

childImage.Y = parent.Y + calcPos.Down.Y \* Position\_Relative.Y \* parent.Scale.Y;

childImage.Z = parent.Z + calcPos.Backward.Z \* Position\_Relative.Z \* parent.Scale.Z;