```cpp
void TestEnvControllableCamera::TickImplORBIT(float horionzalDeltaAxis, float verticalDeltaAxis, float offsetDelta, GameWorld* inWorld) {
    AddPitchInputImpl(verticalDeltaAxis);
    AddYawInputImpl(horionzalDeltaAxis);
    mViewTargetOffset -= offsetDelta * TranslateScale.z;

    // controllable unit handle position
    TRSQuaternionAType focusRotation = GetEntityRotT(inWorld, Focuson);
    Double3 focusPosition = GetEntityPosT(inWorld, Focuson) + focusRotation.Double3Rotate(Double3(FocusonOffset));

    TRSQuaternionAType cameraRotation = TRSQuaternionAType::EulerToQuaternion64(mRotationEuler);

    Double3 EUNNormal;
    TransformSystemG::CartesianCoordinateTransform_ToWGS84(focusPosition.x, focusPosition.y, focusPosition.z, EUNNormal);
    TRSQuaternionAType worldRotation = FFSWGS84SystemG::CreateSphereSpaceWorldRotation(EUNNormal);
    TRSQuaternionAType finalRotation = cameraRotation * worldRotation;

    mViewTargetOffset = std::clamp(mViewTargetOffset, ViewTargetOffsetMinMax.x, ViewTargetOffsetMinMax.y);
    Double3 finalPosition = focusPosition + finalRotation.Double3Rotate(Double3(0, 0, -mViewTargetOffset));

    sys->SetWorldTranslationT(comp.Write(), finalPosition);
    sys->SetWorldRotationT(comp.Write(), finalRotation);
  }
```