visual\_odometry.h:

visual\_odometry类用于实现视觉里程计功能。

original\_image和new\_image存储原始图像和新图像。

horizontalFOV和verticalFOV存储相机的水平和垂直视场角。

height和width存储图像的高度和宽度。

pt\_original和pt\_new存储特征点在原始图像和新图像中的位置。

odom\_init函数用于初始化视觉里程计，包括计算视场角和转换图像为灰度图像。

KeyPointsToPoints函数将关键点转换为2D坐标点。

removePointsIfExceedThreshold函数用于清除超出图像范围的点。

drawPointsOnImage函数将特征点绘制在图像上。

LK\_run函数使用LK光流法计算特征点在新图像中的位置。

visual\_odometry.cpp:

odom\_init函数实现了视觉里程计的初始化，包括计算视场角和转换图像为灰度图像。

KeyPointsToPoints函数将关键点转换为2D坐标点。

removePointsIfExceedThreshold函数用于清除超出图像范围的点。

drawPointsOnImage函数将特征点绘制在图像上。

LK\_run函数实现了LK光流法，计算特征点在新图像中的位置。

ICP.h:

ICP类用于实现ICP算法。

original\_depth\_image和new\_depth\_image存储原始深度图像和新深度图像。

horizontalFOV和verticalFOV存储相机的水平和垂直视场角。

height和width存储图像的高度和宽度。

original\_depths和new\_depths存储原始深度和新深度。

original\_points和new\_points存储原始点和新点。

R和t存储旋转矩阵和平移向量。

depthToPointCloud函数将深度图像转换为点云。

ICP\_init函数用于初始化ICP算法，包括计算视场角和设置深度图像。

ICP\_run函数运行ICP算法，包括深度图像转换为点云和姿态估计。

pose\_estimation\_3d3d函数实现了3D点云的姿态估计。

ICP.cpp:

depthToPointCloud函数将深度图像转换为点云。

ICP\_init函数用于初始化ICP算法，包括计算视场角和设置深度图像。

ICP\_run函数运行ICP算法，包括深度图像转换为点云和姿态估计。

pose\_estimation\_3d3d函数实现了3D点云的姿态估计。

main.cpp:

main函数是程序的入口点。

在main函数中，首先初始化相机和视觉里程计。

然后，循环读取彩色图像和深度图像，并进行视觉里程计和ICP算法。

最后，显示特征点和相机位姿。