实验三: 机器人定位中的直线拟合与提取

1 实验介绍

机器人定位是一项极其重要的任务，其目的在于实时计算机器人所在的位置和方向。本次实验的目标是实现机器人定位中的直线拟合和提取算法，主要有以下两个子任务：

1）实现极坐标下的直线拟合。即输入为一系列笛卡尔坐标 (x, y)，输出为极坐标参数 (r, α)。要求在 MATLAB 中编程实现。

2）利用拆分-合并算法 (Split and Merge)，将获得离散点集划分为一系列线段。要求在 MATLAB 中编程实现。

2 任务一：直线拟合（60 分）

通过激光雷达对场景范围扫描，以描述场景的二维切片。扫描完成后返回一组带有坐标 (x, y) 的点。现在需要把这组无序的点拟合到直线上。

采用极坐标参数 (r, α) 表示一条直线，如直线方程 (1) 所定义，该方程适用于直线上各点的笛卡尔坐标 (x, y)

$$xcos\alpha + ysin\alpha = r \tag{1}$$

其中，$-\pi < \alpha <= \pi$ 表示直线与原点的最短连线与 x 轴所成的角。该连线的长度即为 r。

如图1所示，设目标直线的极坐标参数为 (r, α)，则每一个离散点 (x_i, y_i) 与目标直线的距离 D 的计算公式为：

$$D((r, \alpha)(x_i, y_i)) = r - x_i cos\alpha - y_i sin\alpha \tag{2}$$

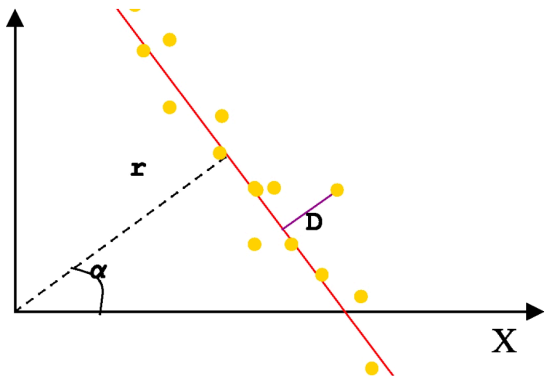

图 1: 极坐标下的直线拟合

离散点集与目标直线的误差即为：

$$S(r,\alpha)=\sum_i (r-x_i cos\alpha - y_i sin\alpha)^2 \tag{3}$$

通过最小化该误差，即可实现对直线的最佳拟合。令 $\frac{\partial S}{\partial \alpha}=0$，可得 α 的计算公式如下：

$$\begin{aligned} \alpha &= \frac{\tan^{-1}(\frac{\text{num}}{\text{denom}})}{2} \\ \text{num} &= -2\sum_i (x_i-x_c)(y_i-y_c) \\ \text{denom} &= \sum_i (y_i-y_c)^2-(x_i-x_c)^2 \end{aligned} \tag{4}$$

其中 (x_c, y_c) 为所有离散点的质心。

任务：令 $\frac{\partial S}{\partial r}=0$，推导 r 的表达式，并根据式 (4) 补全 **fitLine.m** 中的代码。（提示：反正切 $\tan^{-1}()$ 直接使用 MATLAB 中的`atan2()`函数）

验证：在 MATLAB 中运行 **test/testLineFitting.m** ，对实现的直线拟合算法进行验证。如果代码正确无误，则所有测试用例都应通过。

3 任务二：直线提取（40 分）

如图2所示，按照 Split-and-Merge 算法，利用递归的方式将离散点集划分为多条直线。

Algorithm 1: Split-and-Merge

Data: Set S consisting of all N points, a distance threshold $d > 0$
Result: L, a list of sets of points each resembling a line
$L \leftarrow (S), i \leftarrow 1$;
while $i \leq len(L)$ **do**
  fit a line (r, α) to the set L_i;
  detect the point $P \in L_i$ with the maximum distance D to the line (r, α);
  if $D < d$ **then**
    $i \leftarrow i+1$
  else
    split L_i at P into S_1 and S_2;
    $L_i \leftarrow S_1$; $L_{i+1} \leftarrow S_2$;
  end
end
Merge collinear sets in L;

图 2: SAM 算法（每次递归会在距离误差最大的点对当前拟合的直线进行分割）

任务：在 **extractLines.m** 中编辑函数 **findSplitPosInD** 来找到分割线的索引 splitPos。如果当前直线不应拆分，则函数的返回值置为-1。只要直线拟合正确，算法就会执行拆分和合并，并提取每个线段的端点。（提示：要考虑到距离最大值点出现在直线的两端的情况，此时直接作为分割点则会分割出空直线。传入的参数d有正负之分，分别表示位于直线的两侧）

验证：在 MATLAB 中运行 **test/testLineExtraction.m** ，对实现的直线提取算法进行验证。其中测试用例 1-4 为 **30 分**。测试用例 5-6 为 **10 分**。（当进行验证时，你可能会发现部分测试样例无法通过，仔细观察结果图像，你应该会发现部分直线并没有提取完全。这是因为在拆分时，我们只考虑了距离误差最大的点，这可能导致部分不在任何直线上的离散点成为距离误差最大的点，使得提取的直线被离散点提前分割。因此，你需要改进算法，使其能够考虑到连续性问题，即考虑距离误差次大的点和

距离误差最大的点是否相邻，使所有测试样例都能通过。需注意距离误差最大的点和次大的点位于直线同侧才能进行拆分。）

4 需提交内容

1. 实验报告

```
(1). 题目
(2). 姓名-学号-班级
(3). 实验内容分析：比如对实验内容的理解、关键点、思路
(4). 实验过程分析：比如每一步的解析
(5). 实验结果分析
(6). 遇到的问题和心得
```

2. 实验源代码