扩展卡尔曼滤波仿真

刘玉彬

23 November 2020

Contents

1 问题介绍

本任务中的观测模型输入为 GPS 全局绝对观测，运动模型的输入为固定的速度和角速度。运动模型的输入在 $robot\_control()$ 函数中已定义，v 包含 v_x，v_y 和角速度 ψ，不需要修改。机器人的状态量为三维，起始位姿的描述为 $(x, y, \theta) = [0, 0, 0]^T$。

在给定的 $prepare()$ 函数中，本程序首先根据机器人速度和运动模型计算定位真值。然后在速度上添加噪声，计算有误差的里程计定位。然后在定位真值上添加噪声，形成 GPS 观测。噪声在文件开始已给定，该函数不需要修改。最后结果的图像中，包括定位真值、仅里程计定位的值、GPS 观测的值与 EKF 算法预测的值，仅里程计定位的值是在速度上叠加误差后在上一次里程计定位的基础上迭代生成的。

考虑到工具的熟练程度，本次实验利用 $python$ 语言实现，并最终利用 $matplotlib$ 库完成了测量值，估计值与机器人轨迹的绘制。在报告中简述了运动模型和观测模型的构建过程，实现算法的过程与最后的结果分析。可以看出，相比于仅里程计定位造成的累计误差逐渐增加，扩展卡尔曼滤波算法实现的定位与真值非常接近。

2 扩展卡尔曼滤波算法简介

根据观测信息和机器人的运动模型，我们可以用马尔科夫定位方法增量地对机器人进行定位，例如扩展卡尔曼滤波 (Extended Kalman Filter)。对运动模型和观测模型，我们使用一阶近似，有：

$$\mathbf{x}_k \approx \gamma(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k) + \left.\frac{\partial \gamma}{\partial \mathbf{x}_{k-1}}\right|_{\hat{\mathbf{x}}_{k-1}} (\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}) + \mathbf{w_k} \tag{1}$$

$$\mathbf{z}_k \approx \phi(\check{\mathbf{x}}_k) + \left.\frac{\partial \phi}{\partial \mathbf{x}_k}\right|_{\check{\mathbf{x}}_k} (\mathbf{x}_k - \check{\mathbf{x}}_k) + \mathbf{n}_k \tag{2}$$

其中，$\mathbf{x}_k$ 表示机器人的状态，$\check{\mathbf{x}}_k$ 是对机器人状态的先验 (*a priori*) 估计，$\hat{\mathbf{x}}_k$ 是对机器人状态的后验 (*a posteriori*) 估计。同样地，我们有传感器的观测值 $\mathbf{z}_k$。假设所有的状态和控制噪声均满足高斯分布：

$$\mathbf{w}_k \sim N(\mu_k, \mathbf{R}_k) \qquad \mathbf{n}_k \sim N(\mu'_k, \mathbf{Q}_k)$$

利用 $\mathbf{F}$ 和 $\mathbf{H}$ 表示函数 γ 和 ϕ 的雅克比矩阵 (Jacobian matrix)，我们可以推导出扩展卡尔曼滤波的公式如下所示：

$$\check{\mathbf{x}}_k = \gamma(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k) \tag{3}$$

$$\check{\mathbf{P}}_k = \mathbf{F}\hat{\mathbf{P}}_{k-1}\mathbf{F}^T + \mathbf{R}_k \tag{4}$$

$$\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{z}_k - \phi(\check{\mathbf{x}}_k)) \tag{5}$$

$$\hat{\mathbf{P}}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\check{\mathbf{P}}_k \tag{6}$$

其中，$\mathbf{P}_k$ 是协方差矩阵，用于表述状态估计的不确定性。$\mathbf{K}_k$ 是卡尔曼增益:

$$\mathbf{K}_k = \check{\mathbf{P}}_k\mathbf{H}^T(\mathbf{H}\check{\mathbf{P}}_k\mathbf{H}^T + \mathbf{Q}_k)^{-1} \tag{7}$$

扩展卡尔曼滤波算法示于 Algorithm1

Algorithm 1: EKF Algorithm

Input: Position $\mathbf{x}_{k-1}$, Covariance matrix $\mathbf{P}_{k-1}$, Observation $\mathbf{z}_k$

Output: *A posterior* estimation $\hat{\mathbf{x}}_k$ and new covariance matrix $\hat{\mathbf{P}}_k$

1 $\check{\mathbf{x}}_k = \gamma(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_k)$

2 $\check{\mathbf{P}}_k = \mathbf{F}\hat{\mathbf{P}}_{k-1}\mathbf{F}^T + \mathbf{R}_k$

3 $\mathbf{K}_k = \check{\mathbf{P}}_k\mathbf{H}^T(\mathbf{H}\check{\mathbf{P}}_k\mathbf{H}^T + \mathbf{Q}_k)^{-1}$

4 $\hat{\mathbf{x}}_k = \check{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{z}_k - \phi(\check{\mathbf{x}}_k))$

5 $\hat{\mathbf{P}}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H})\check{\mathbf{P}}_k$

3 运动模型与观测模型

由上述内容可以看出，实现卡尔曼滤波必须构建机器人的运动模型与观测模型，这两部分在程序的 $do\_motion()$ 和 $do\_observation()$ 函数中实现。此外，$jacob\_f()$ 和 $jacob\_h()$ 计算了两个函数的雅克比矩阵 $\mathbf{F}$ 与 $\mathbf{H}$。

3.1 运动模型

考虑卡尔曼滤波算法中的运动模型，我们有：

$$\mathbf{x}_k = \gamma(\mathbf{x}_{k-1}, \mathbf{v}_k) + \mathbf{w}_k \quad \mathbf{w}_k \sim N(\mu_k, \mathbf{R}_k) \tag{8}$$

在里程计模型中，将上一时刻卡尔曼滤波估计值作为上一时刻位置 $\mathbf{x}_{k-1}$ 的输入，速度变量 $\mathbf{v}_k$ 已经给定，我们有：

$$\begin{bmatrix} x_k \\ y_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ \theta_{k-1} \end{bmatrix} + dt \times \begin{bmatrix} \cos\theta_{k-1} & -\sin\theta_{k-1} & 0 \\ \sin\theta_{k-1} & \cos\theta_{k-1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_\theta \end{bmatrix} \tag{9}$$

其中，速度向量 $\mathbf{v}_k$ 叠加了一个协方差矩阵为 $\mathbf{U}$ 的正态分布：

$$\mathbf{v}_k \sim N(v_k, \mathbf{U}) \tag{10}$$

因此，公式9分别对位置变量 $\mathbf{x}_{k-1}$ 和速度变量 $\mathbf{v}_k$ 求偏导，可以得到运动模型的雅克比矩阵 $\mathbf{F}$ 和用于将控制空间噪声转换到状态空间的矩阵 $\mathbf{V}_k$：

$$\mathbf{F} = \left.\frac{\partial \gamma}{\partial \mathbf{x}_{k-1}}\right|_{\mathbf{x}_{k-1}} = \begin{bmatrix} 1 & 0 & -v_x\sin\theta_{k-1} - v_y\cos\theta_{k-1} \\ 0 & 1 & v_x\cos\theta_{k-1} - v_y\sin\theta_{k-1} \\ 0 & 0 & 1 \end{bmatrix} \tag{11}$$

$$\mathbf{V} = \left.\frac{\partial \gamma}{\partial \mathbf{v}_k}\right|_{\mathbf{v}_k} = \begin{bmatrix} \cos\theta_{k-1} & -\sin\theta_{k-1} & 0 \\ \sin\theta_{k-1} & \cos\theta_{k-1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{12}$$

基于此，我们有：

$$\mathbf{R}_k = \mathbf{V}_k \mathbf{U} \mathbf{V}_k^T \tag{13}$$

3.2 观测模型

观测模型是对当前的先验位姿进行一个观测的假设，之后在滤波器中根据实际的 GPS 测量对先验位姿进行修正。考虑 GPS 为全局绝对观测，我们将观测模型如下定义：

$$\mathbf{z}_k = \mathbf{I}\mathbf{x}_k + \mathbf{n}_k \quad \mathbf{n}_k \sim N(\mu_k, \mathbf{Q}_k) \tag{14}$$

其中，$\mathbf{n}_k$ 为叠加的噪声，其协方差矩阵为 $\mathbf{Q}_k$，$\mathbf{I}$ 为单位矩阵。

显然，该模型为一个线性模型，因此有：

$$\mathbf{H} = \mathbf{I} \tag{15}$$

4 实验结果与分析

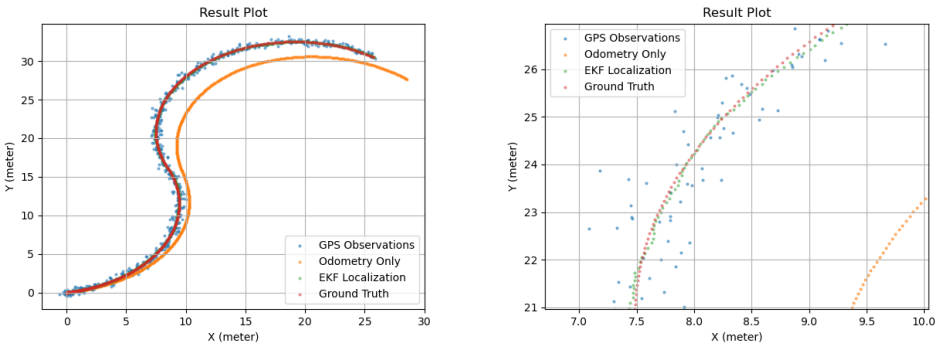
本次实验利用 *python* 完成，使用 *numpy* 进行矩阵的运算并利用 *matplotlib* 完成结果的绘制。函数功能在 *ekf_alg.py* 中实现，包括两个模型的定义、状态初始化、卡尔曼滤波与结果绘制和误差计算。在主函数中，实现了一个 $60s$ 的机器人定位估计，其中每一时刻的速度已经给定，差分时间 $dt = 0.1s$。实验的结果，包括每一帧的各个估计量与观测量与真实值均存于文件 *data.mat* 中。

机器人的运动轨迹示于图4中，可以看出，相比于仅里程计定位造成的累计误差逐渐增加，扩展卡尔曼滤波算法实现的定位与真值非常接近。在多次试验后，测得的仅里程计估计的误差与卡尔曼滤波后的误差示于表1中。误差的计算为求取估计点 (x_e, y_e) 到实际点 (x_t, y_t) 距离的平均值：

$$Error = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_e - x_t)^2 + (y_e - y_t)^2} \tag{16}$$

实验	1	2	3	4	5
仅里程计估计的误差 (meter)	0.322	1.684	0.604	0.998	0.344
卡尔曼滤波的误差 (meter)	0.047	0.056	0.054	0.059	0.053

Table 1: 实验误差



(a) 完整图 (b) 部分放大图

Figure 1: 实验结果