УДК 621.391

# ВЕКТОРНЫЙ ФИЛЬТР КАЛЬМАНА

*Ахапкина А.М.[1], курсант гр. 033701*

*Белорусский государственный университет информатики и радиоэлектроники[1]*

*г. Минск, Республика Беларусь*

*Фильченкова Т.М. – магистр. технических наук*

**Аннотация.** В данной статье приведены теоретические сведения и практические результаты выполнения векторного фильтра Калмана, который позволяет строить оценки подвижной модели и тем самым определять ее местоположение в пространстве относительно трех осей.

**Ключевые слова.** оценка, фильтр Калмана, марковская модель, шум, оси, GPS, матрица

Актуально определение местонахождения подвижных объектов (машин, самолетов, людей) как для частных, так и для государственных структур. Определение местоположения включает в себя три этапа: непосредственное определение местоположение, передача данных и управления, обработка данных.

Самым определяющим этапом является непосредственное определение местоположения движущегося объекта, ведь если на начальном этапе определить его неверно – последующие этапы будут бессмысленны. Так же не менее важным этапов является передача данных и управления подвижными объектами, который не должен затрачивать много времени.

Во многим системах GPS используется Калмановский подход, имеющий наименьшую погрешность.

В нем используется алгоритм оценки сразу по трем пространственным осям, где процедуры выполняются независимо от каждой:

$$\begin{cases} \hat{x}_k = u\left(z_k, \hat{x}_{k-1}\right) \\ \hat{y}_k = u\left(z_k, \hat{y}_{k-1}\right) \\ \hat{z}_k = u\left(z_k, \hat{z}_{k-1}\right) \end{cases} \qquad k = 1, 2, 3, \ldots \tag{1}$$

где u(○) – калмановский алгоритм построения текущей оценки.

Так как все три координаты прадставляют собой точку в трехмерном пространстве, то изменение положение объекта удобнее представить векторно-матричной марковской моделью:

$$\begin{bmatrix} x_k \\ y_k \\ z_k \end{bmatrix} = \begin{bmatrix} r_x & 0 & 0 \\ 0 & r_y & 0 \\ 0 & 0 & r_z \end{bmatrix} \cdot \begin{bmatrix} x_{k-1} \\ y_{k-1} \\ z_{k-1} \end{bmatrix} + \begin{bmatrix} \xi_{xk} \\ \xi_{yk} \\ \xi_{zk} \end{bmatrix} \tag{2}$$

или, в таких очевидных обозначениях:

$$\overline{x}_k = R \cdot \overline{x}_{k-1} + \overline{\xi}_k \tag{3}$$

Если предположить, что все случайные составляющие подчиняются нормальному закону распределения с нулевым средним и некоторыми дисперсиями. Например, для

$$M\left\{\overline{\xi}_k \cdot \overline{\xi}_k^T\right\} = M\left\{\begin{bmatrix}\xi_{xk}, \xi_{yk}, \xi_{zk}\end{bmatrix} \cdot \begin{bmatrix}\xi_{xk} \\ \xi_{yk} \\ \xi_{zk}\end{bmatrix}\right\} = \begin{bmatrix}\sigma_{\xi x}^2 & 0 & 0 \\ 0 & \sigma_{\xi y}^2 & 0 \\ 0 & 0 & \sigma_{\xi z}^2\end{bmatrix} = V_\xi \tag{4}$$

Модель наблюдения, соответственно, определяется вектором z:

$$\begin{bmatrix}z_{xk} \\ z_{yk} \\ z_{zk}\end{bmatrix} = \begin{bmatrix}x_k \\ y_k \\ z_k\end{bmatrix} + \begin{bmatrix}n_{xk} \\ n_{yk} \\ n_{zk}\end{bmatrix} \tag{5}$$

или, в векторном виде:

$$\overline{z}_k = \overline{x}_k + \overline{n}_k \tag{6}$$

Здесь дисперсии шумов будут образовывать диагональную матрицу:

$$M\left\{\overline{n}_k \cdot \overline{n}_k^T\right\} = M\left\{\begin{bmatrix}n_{xk} \\ n_{yk} \\ n_{zk}\end{bmatrix} \cdot \begin{bmatrix}n_{xk}, n_{yk}, n_{zk}\end{bmatrix}\right\} = \begin{bmatrix}\sigma_{nx}^2 & 0 & 0 \\ 0 & \sigma_{ny}^2 & 0 \\ 0 & 0 & \sigma_{nz}^2\end{bmatrix} = V_k \tag{7}$$

Имеем векторную марсковскую модель изменения объекта в пространстве и вектор наблюдений текущего местоположения объекта (рисунок 1):

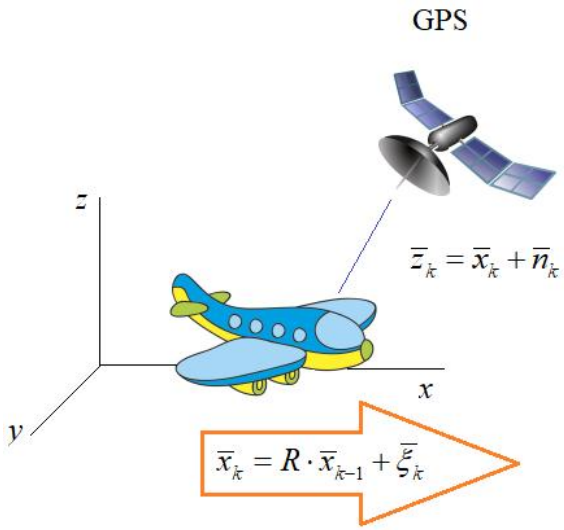


Рисунок 1 – Векторная марковская модель изменения местоположения самолета

Когда на вход приемника приходит самое первое наблюдение, то лучшая оценка – это значение самого наблюдения:

$$\hat{\bar{x}}_1 = z_1 \tag{8}$$

с дисперсией ошибок оценивания:

$$P = V_1 \tag{9}$$

В следующий момент времени модель (самолет) немного меняет свое местоположение и на вход GPS-приемника приходит очередное наблюдение:

$$\bar{z}_2 = \bar{x}_2 + \bar{n}_2 \tag{10}$$

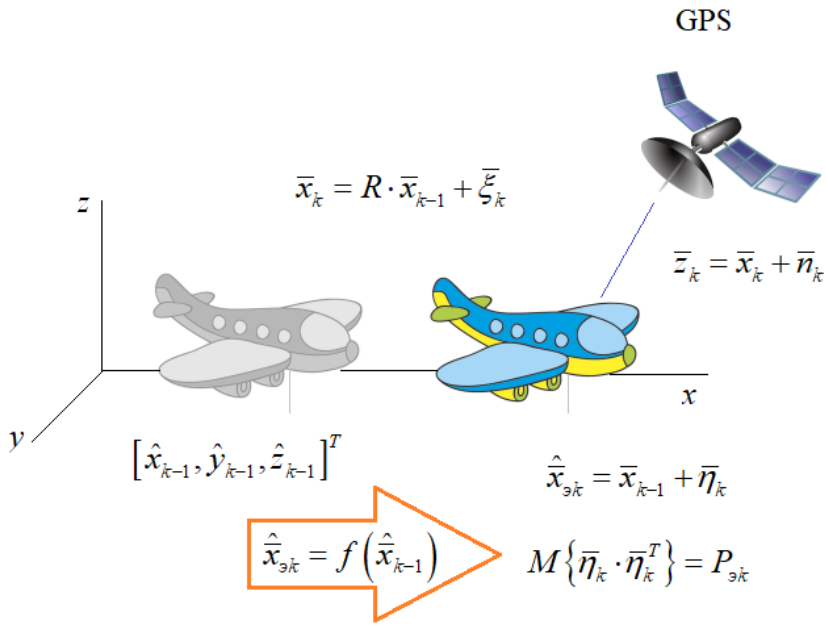


Рисунок 2 – Векторная марковская модель (самолет) в следующий момент времени

В данной ситуации при построении оценки текущего местоположения воспользуемся двумя наблюдениями. Первое – вектор $\vec{z_2}$, а второе наблюдение берется как прогноз текущего положения из предыдущей оценки $\vec{x_1}$, используя марковскую модель движения:

$$\hat{\bar{x}}_{э2} = R \cdot \hat{\bar{x}}_1 \tag{11}$$

Затем вычисляется дисперсия ошибок прогноза, которая равна:

$$P_{э2} = M\left\{\left(\bar{x}_2 - R\hat{\bar{x}}_1\right)^2\right\} = M\left\{\left(R\bar{x}_1 + \bar{\xi}_2 - R\hat{\bar{x}}_1\right)^2\right\} =$$
$$= RM\left\{\left(\bar{x}_1 - \hat{\bar{x}}_1\right)^2\right\} R^T + V_\xi = R \cdot P_1 \cdot R^T + V_\xi \tag{12}$$

В результате имеются все необходимые данные для построения оценок координат на текущем шаге с помощью векторного фильтра Калмана. Используя формулы скалярного фильтра:

$$P_k = \frac{P_{эk} \cdot \sigma_k^2}{P_{эk} + \sigma_k^2}, \qquad \hat{x}_k = \hat{x}_{эk} + \frac{P_k}{\sigma_k^2} \cdot (z_k - \hat{x}_{эk}) \tag{13}$$

Обобщенно для векторно-матричного случая:

$$P_k = P_{эk} \cdot V_k \cdot \left(P_{эk} + V_k\right)^{-1}, \qquad \hat{\bar{x}}_k = \hat{\bar{x}}_{эk} + P_k \cdot V_k^{-1} \cdot \left(\bar{z}_k - \hat{\bar{x}}_{эk}\right) \tag{14}$$

В данном случаи формально преобразовано скалярное выражение в векторное и получено возможность строить векторы оценок. В этот состоит одно из достоинств векторов и матриц: оно позволяют легко обобщать задачи на многомерный случай.

На втором шаге имеются следующие вычисления:

$$P_2 = P_{э2} \cdot V_2 \cdot \left(P_{э2} + V_2\right)^{-1}, \qquad \hat{\overline{x}}_2 = \hat{\overline{x}}_{э2} + P_2 \cdot V_2^{-1} \cdot \left(\overline{z}_2 - \hat{\overline{x}}_{э2}\right) \tag{15}$$

Вычисления на последующих шагах вычисляются рекуррентно по формулам (15).

Для реализации векторного фильтра Калмана на Python необходимо учесть, что при условии некоррелированности и шумов и независимости перемещений по каждой из координат, получено тни независимых канала оценивания параметров x,y,z:

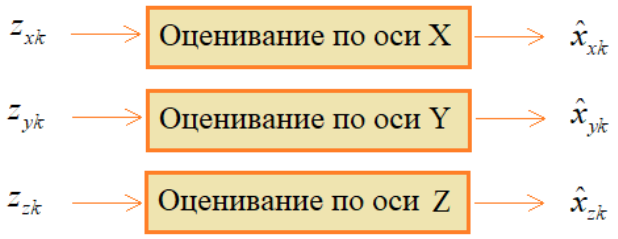


Рисунок 3 – Независимые каналы оценивания параметров

Однако, если модифицировать модель и учесть в ней скорости движения объекта по каждой координат, то получится взаимосвязь между наблюдаемыми скоростями и соответствующей координаты:

$$\begin{bmatrix} x_i \\ y_i \\ z_i \\ v_{xi} \\ v_{yi} \\ v_{zi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & r_x & 0 & 0 \\ 0 & 0 & 0 & 0 & r_y & 0 \\ 0 & 0 & 0 & 0 & 0 & r_z \end{bmatrix} \cdot \begin{bmatrix} x_{i-1} \\ y_{i-1} \\ z_{i-1} \\ v_{xi-1} \\ v_{yi-1} \\ v_{zi-1} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ \xi_{vxi} \\ \xi_{vyi} \\ \xi_{vzi} \end{bmatrix} \tag{16}$$

Причем, векторный фильтр Каламана автоматически наилучшим образом скомбинирует эти набюлдения для вычисления оптимальной выходной оценки в соответствии с описанной моделью перемещения. И это очень удобно, ведь не нужно решать систему линейных уравнений, находить оптимальные коэффициенты, все аккуратно расписывать. Все это делается автоматически при вычислении обратных матриц в фильтре Калмана.

Далее, для второго варианта матрицы дисперсий порождающего шума и шума наблюдений, будут выглядеть следующим образом:

$$V_\xi = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_\xi^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_\xi^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_\xi^2 \end{bmatrix} \quad V = \begin{bmatrix} \sigma^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma_v^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \sigma_v^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_v^2 \end{bmatrix} \tag{17}$$

Весь остальной алгоритм оценивания будет прежним:

$$P_k = P_{зk} \cdot V_k \cdot \left(P_{зk} + V_k\right)^{-1}, \qquad \hat{\bar{x}}_k = \hat{\bar{x}}_{зk} + P_k \cdot V_k^{-1} \cdot \left(\bar{z}_k - \hat{\bar{x}}_{зk}\right) \tag{18}$$

```
N = 100          # число наблюдений
dNoise = 1       # дисперсия шума
dSignal = 5      # дисперсия сигнала
r = 0.99         # коэффициент корреляции в модели движения
en = 0.1         # дисперсия СВ в модели движения

M = 3            # размерность вектора координат положения объекта
```

Рисунок 4 – Задание параметров для построения оценок

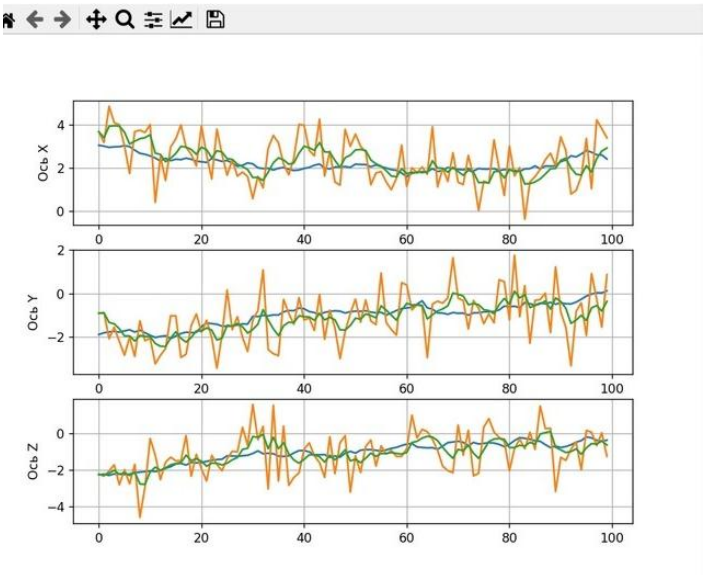


Рисунок 5 – Результат выполнения программы

**Список использованных источников:**

1. Mohamed Laaraiedh «Implementation of Kalman Filter with Python Language» // paper, 2012
2. Бутковский, О.А «Предельные теоремы для марковских процессов» // диссертация, 2013.
3.Пучков, А.Ю «Разработка методов фильтрации в постановке Р. Калмана в условиях случайной дискретизации сигналов» // диссертация, 2019.