

Кватернионы

Представление: $q = [w, x, y, z] = w + x\mathbf{i} + y\mathbf{j} + z\mathbf{k} = w + (x, y, z)$

Поворот трехмерного вектора: $q\mathbf{v}q^{-1}$ – поворот вектора \mathbf{v} , представленного в виде $\mathbf{v} = [0, \mathbf{v}_x, \mathbf{v}_y, \mathbf{v}_z]$ на угол θ вокруг единичной оси \mathbf{r} , где

$$q = \cos\left(\frac{\theta}{2}\right) + \mathbf{r} \sin\left(\frac{\theta}{2}\right)$$

Умножение: $ab = [a_1 \ a_2 \ a_3 \ a_4][b_1 \ b_2 \ b_3 \ b_4] = \begin{bmatrix} a_1b_1 - a_2b_2 - a_3b_3 - a_4b_4 \\ a_1b_2 + a_2b_1 + a_3b_4 - a_4b_3 \\ a_1b_3 - a_2b_4 + a_3b_1 + a_4b_2 \\ a_1b_4 + a_2b_3 - a_3b_2 + a_4b_1 \end{bmatrix}^T$

Обратный кватернион: $q^{-1} = \frac{q^*}{|q|^2}, q^* = [w, -x, -y, -z]$

Преобразование кватернионов в углы Эйлера

$$q = [q_1, q_2, q_3, q_4]$$

$$\psi = \text{atan2}(2(q_1q_4 + q_2q_3), 1 - 2(q_3^2 + q_4^2))$$

$$\theta = \text{asin}(2(q_1q_3 - q_4q_2))$$

$$\phi = \text{atan2}(2(q_1q_2 + q_3q_4), 1 - 2(q_2^2 + q_3^2))$$



Фильтр Маджвика

- ▶ **Входные данные:**

- ▶ Вектор угловых скоростей от гироскопа
- ▶ Вектор ускорений от акселерометра
- ▶ Вектор магнитной индукции от магнитометра

- ▶ **Выходные данные:**

- ▶ Кватернион, определяющий ориентацию в пространстве



$q_{est,t} = \hat{q}_{est,t-1} + \dot{q}_{est,t} \Delta t$	Текущее расчетное значение ориентации
$\dot{q}_{est,t} = \dot{q}_{g,t} - \beta \dot{\hat{q}}_{e,t}$	Расчетное значение скорости изменения ориентации
$\dot{q}_{g,t} = \frac{1}{2} \hat{q}_{est,t-1} g_{c,t}$	Скорость изменения ориентации по показаниям гироскопа
$\dot{\hat{q}}_{e,t} = \frac{\nabla f(\hat{q}_{est,t-1}, \hat{a}_t, \hat{b}_t, \hat{m}_t)}{\ \nabla f\ }$	Направление ошибки скорости изменения ориентации
$b_t = [0, \sqrt{h_x^2 + h_y^2}, 0, h_z]$	Относительное магнитное поле Земли
$h_t = \hat{q}_{est,t-1} m_t \hat{q}_{est,t-1}^*$	Направление магнитного поля в земных координатах



$$g_{c,t} = g_t - g_{b,t}$$

Скомпенсированные показания гироскопа

$$g_{b,t} = \zeta \sum_t g_{e,t} \Delta t$$

Смещение показаний гироскопа

$$g_{e,t} = 2\hat{q}_{est,t-1}^* \dot{\hat{q}}_{e,t}$$


Угловая погрешность показаний гироскопа

$$g_t, a_t, m_t$$

Показания гироскопа, акселерометра и магнитометра, представленные в виде кватернионов

$$\beta, \zeta$$

Настраиваемые коэффициенты усиления



Вычисление $\nabla f = J^T f$

$$\mathbf{f}_g({}^S_E\hat{\mathbf{q}}, {}^S\hat{\mathbf{a}}) = \begin{bmatrix} 2(q_2q_4 - q_1q_3) - a_x \\ 2(q_1q_2 + q_3q_4) - a_y \\ 2(\frac{1}{2} - q_2^2 - q_3^2) - a_z \end{bmatrix}$$

$$\mathbf{J}_g({}^S_E\hat{\mathbf{q}}) = \begin{bmatrix} -2q_3 & 2q_4 & -2q_1 & 2q_2 \\ 2q_2 & 2q_1 & 2q_4 & 2q_3 \\ 0 & -4q_2 & -4q_3 & 0 \end{bmatrix}$$

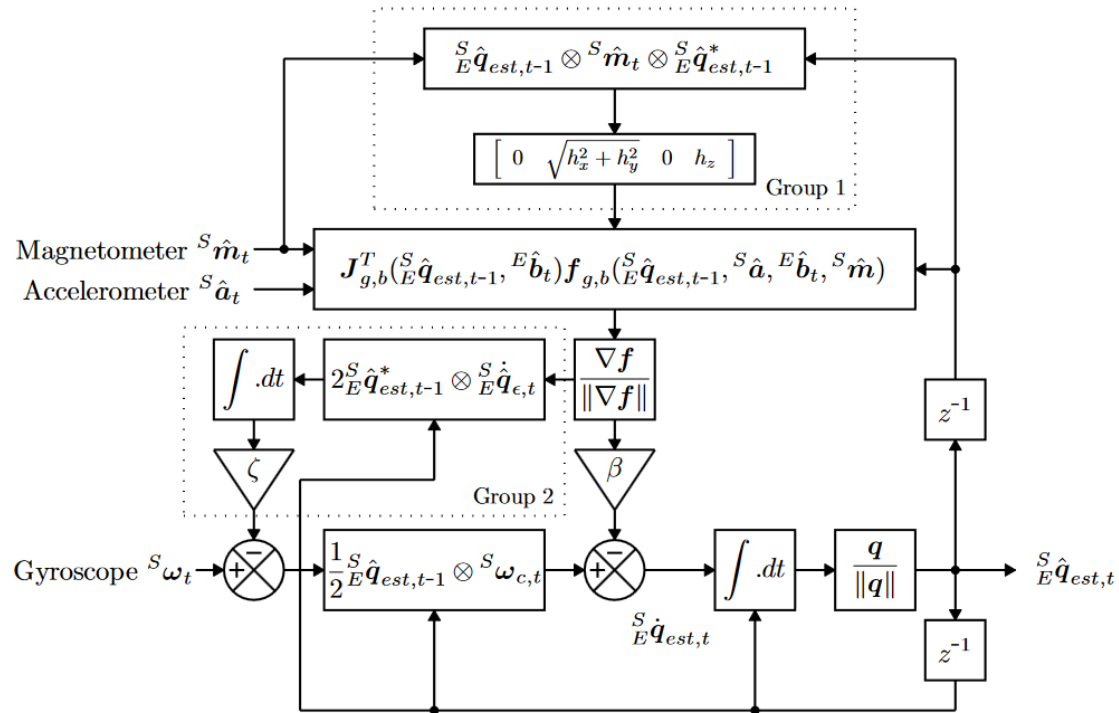
$$\mathbf{f}_b({}^S_E\hat{\mathbf{q}}, {}^E\hat{\mathbf{b}}, {}^S\hat{\mathbf{m}}) = \begin{bmatrix} 2b_x(0.5 - q_3^2 - q_4^2) + 2b_z(q_2q_4 - q_1q_3) - m_x \\ 2b_x(q_2q_3 - q_1q_4) + 2b_z(q_1q_2 + q_3q_4) - m_y \\ 2b_x(q_1q_3 + q_2q_4) + 2b_z(0.5 - q_2^2 - q_3^2) - m_z \end{bmatrix}$$

$$\mathbf{J}_b({}^S_E\hat{\mathbf{q}}, {}^E\hat{\mathbf{b}}) = \begin{bmatrix} -2b_zq_3 & 2b_zq_4 & -4b_xq_3 - 2b_zq_1 \\ -2b_xq_4 + 2b_zq_2 & 2b_xq_3 + 2b_zq_1 & 2b_xq_2 + 2b_zq_4 \\ 2b_xq_3 & 2b_xq_4 - 4b_zq_2 & 2b_xq_1 - 4b_zq_3 \\ & -4b_xq_4 + 2b_zq_2 \\ & -2b_xq_1 + 2b_zq_3 \\ & 2b_xq_2 \end{bmatrix}$$

$$\mathbf{f}_{g,b}({}^S_E\hat{\mathbf{q}}, {}^S\hat{\mathbf{a}}, {}^E\hat{\mathbf{b}}, {}^S\hat{\mathbf{m}}) = \begin{bmatrix} \mathbf{f}_g({}^S_E\hat{\mathbf{q}}, {}^S\hat{\mathbf{a}}) \\ \mathbf{f}_b({}^S_E\hat{\mathbf{q}}, {}^E\hat{\mathbf{b}}, {}^S\hat{\mathbf{m}}) \end{bmatrix}$$

$$\mathbf{J}_{g,b}({}^S_E\hat{\mathbf{q}}, {}^E\hat{\mathbf{b}}) = \begin{bmatrix} \mathbf{J}_g^T({}^S_E\hat{\mathbf{q}}) \\ \mathbf{J}_b^T({}^S_E\hat{\mathbf{q}}, {}^E\hat{\mathbf{b}}) \end{bmatrix}$$





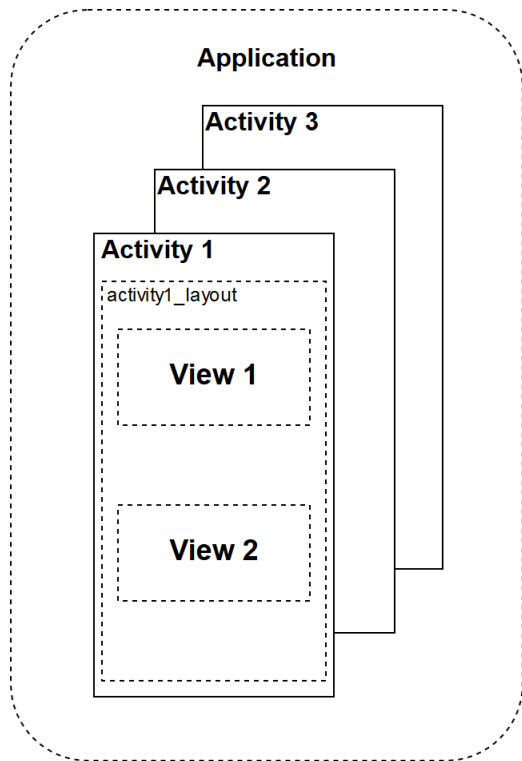
Разработка под Android



Языки	Java, Kotlin, C++	C#
Платформы	Win, Linux, macOS	Win, macOS



Структура Android приложения



- ▶ Логика взаимодействия с UI находится в наследниках класса `Activity`
- ▶ Базовые элементы UI (ввод/вывод текста, вывод изображений, кнопки и т.д.) представлены наследниками `View`
- ▶ Макет `Activity` (расположение на экране различных `View`) находится в `layout` файле
- ▶ В один момент времени активно только одно `Activity`, но между различными `Activity` можно переключаться

Сбор данных датчиков

- ▶ Создать класс, реализующий интерфейс `SensorEventListener`
- ▶ Подписать данный класс на получение данных от определенного датчика при помощи класса `SensorManager` и метода `registerListener`
- ▶ После окончания работы с датчиками вызвать `unregisterListener`



```
▶ public class MainActivity extends Activity implements SensorEventListener {
    private TextView accelerometerTextView;

    private SensorManager sensorManager;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        accelerometerTextView = (TextView)findViewById(R.id.accelerometerTextView);
        sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);

        sensorManager.registerListener(this,
            sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
            SensorManager.SENSOR_DELAY_UI);
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        sensorManager.unregisterListener(this);
    }

    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
    }

    @Override
    public void onSensorChanged(SensorEvent e) {
        String text = String.format(Locale.ENGLISH, "Accelerometer, m/s^2\nX: %f\nY: %f\nZ: %f",
            e.values[0], e.values[1], e.values[2]);
        accelerometerTextView.setText(text);
    }
}
```



Задание

- ▶ Написать приложение под Android для сбора данных акселерометра/гироскопа/магнитометра
- ▶ Реализовать фильтр Маджвика (на любом языке/платформе) и обработать им собранные данные



Литература

- ▶ Статья Маджвика про данный фильтр: http://x-io.co.uk/res/doc/madgwick_internal_report.pdf
- ▶ Перевод данной статьи: <https://habr.com/en/post/255661/>
- ▶ Кватернионы и вращение пространства: https://en.wikipedia.org/wiki/Quaternions_and_spatial_rotation
- ▶ Документация по Android API: <https://developer.android.com/>

