AP_InertialNav.cpp

惯导系统通过积分加速度计的读数来获得速度与位置信息，随时间增长会产生积累误差。InertialNav 中的函数用于修正速度和位置。加速度-速度-位置构成三阶系统，通过将位置误差 _position_error 积分后反馈到各物理量，形成三个反馈回路，实现闭环控制。

主要函数为 update(),check_gps() 和 check_baro()，整体结构如图所示，虚线左侧由 update() 实现，虚线右侧由 check_gps() 和 check_baro() 实现：

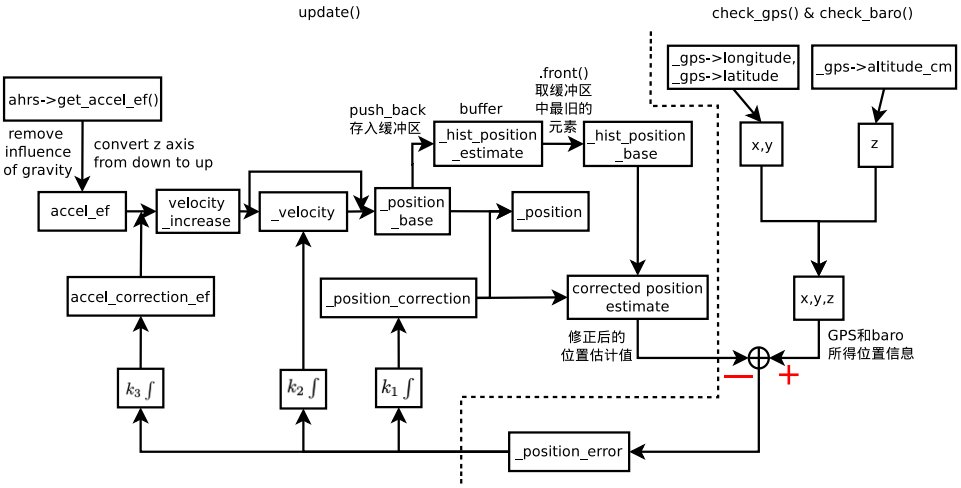

图 1: InertialNav

修正具体分为两个通道：

1.horizontal：使用 GPS 信息修正水平位置；

2.vertical：　使用 Barometer 信息修正高度。

主要步骤：

1.init()

初始化。调用 update_gains() 函数，使用互补滤波时间常数计算控制增益 _k1_xy,_k2_xy,_k3_xy 和 _k1_z,_k2_z,_k3_z。

```
1        _k1_xy = 3 / _time_constant_xy;
2        _k2_xy = 3 / (_time_constant_xy*_time_constant_xy);
3        _k3_xy = 1 / (_time_constant_xy*_time_constant_xy*
             _time_constant_xy);
4
5        _k1_z = 3 / _time_constant_z;
6        _k2_z = 3 / (_time_constant_z*_time_constant_z);
7        _k3_z = 1 / (_time_constant_z*_time_constant_z*
             _time_constant_z);
```

从增益的形式来看，系统采用了扩张状态观测器。GPS 和 baro 所得位置信息与修正后的位置估计值的差值 (即 _position_error) 为观测误差，增益为 _k3 的状态变量 (即 accel_correction_ef) 是扩张变量。状态观测器的特征多项式为：

$$\lambda^3 + 3\lambda^2 + 3\lambda + 1 = (\lambda + 1)^3 \tag{1}$$

其满足 Hurwitz 条件，故能够保证观测误差的快速收敛。从而，通过状态反馈 (将 accel_correction_ef 补偿到 accel_ef)，实现了对位置和速度的修正。

2.update(float dt)

如果 GPS 和气压计有新数据可用，使用最新信息更新速度和位置。

函数调用关系：

update()-> check_baro()

check_gps()-> correct_with_gps()

correct_with_baro()

correct_with_gps() 和 correct_with_baro() 用于计算 _position_error。注意到代码中传递给 check_gps() 调用的 correct_with_baro() 的参数是 _gps->altitude_cm-_home_alt，而 check_baro() 调用 correct_with_baro() 的语句被 if(0) 跳过，即实际使用了 gps 而不是 baro 来获得高度。update() 实现图1中的其余部分。

结合流程图分析代码：

1) 调用函数 check_baro() 和 check_gps() 获取位置信息，并计算加速度计所得位置估计值的误差 _position_error：

```
1    // check if new baro readings have arrived and use them to
         correct vertical accelerometer offsets.
2    check_baro();
3
4    // check if new gps readings have arrived and use them to
         correct position estimates
5    check_gps();
```

2) 使用 ahrs 获得加速度计读数，然后进行处理：去除重力加速度的影响，然后修改坐标系定义，将 z 轴由向下改为向上：

```
1    Vector3f accel_ef = _ahrs->get_accel_ef();  ///
         accelerometer values in the earth frame
2
3    // remove influence of gravity
4    accel_ef.z += GRAVITY_MSS;
5    accel_ef *= 100;
6
7    //Convert North-East-Down to North-East-Up
8    accel_ef.z = -accel_ef.z;
```

3) 建立反馈回路，将位置误差 _position_error 乘以相应的系数反馈到加速度、速度和位置中：

```
float tmp = _k3_xy * dt;
accel_correction_ef.x += _position_error.x * tmp;
accel_correction_ef.y += _position_error.y * tmp;
accel_correction_ef.z += _position_error.z * _k3_z  * dt;

tmp = _k2_xy * dt;
_velocity.x += _position_error.x * tmp;
_velocity.y += _position_error.y * tmp;
_velocity.z += _position_error.z * _k2_z  * dt;

tmp = _k1_xy * dt;
_position_correction.x += _position_error.x * tmp;
_position_correction.y += _position_error.y * tmp;
_position_correction.z += _position_error.z * _k1_z  * dt;
```

$$
\begin{aligned}
a_{correction\_ef} &= \int_0^t \varepsilon_{position} \cdot k_1 \cdot d\tau \\
v_{correction} &= \int_0^t \varepsilon_{position} \cdot k_2 \cdot d\tau \\
position_{correction} &= \int_0^t \varepsilon_{position} \cdot k_3 \cdot d\tau
\end{aligned} \tag{2}
$$

$\varepsilon_{position}$ 即 _position_error。k_1 是反馈到加速度的增益系数，k_2 是反馈到速度的增益系数，k_3 是反馈到位置的增益系数。注意每个回路的水平通道增益 (k_xy) 和高度通道增益 (k_z) 不同。

4) 将时间间隔 dt 内的运动视为匀加速运动，计算速度增量 velocity_increase，更新（未修正的）位置估计值 _position_base 和速度 _velocity：

```
// calculate velocity increase adding new acceleration from
    accelerometers
const Vector3f &velocity_increase = (accel_ef +
    accel_correction_ef) * dt;

// calculate new estimate of position
_position_base += (_velocity + velocity_increase*0.5) * dt;

// update the corrected position estimate
_position = _position_base + _position_correction;

// calculate new velocity
_velocity += velocity_increase;
```

即

$$\begin{aligned} \Delta v &= (a_{ef} + a_{correction\_ef})dt \\ \Delta s &= (v + \frac{1}{2}\Delta v)t \\ v &= v + \Delta v \end{aligned} \tag{3}$$

式中 Δv 为 velocity_increase，Δs 为 _position_base 的增量。

5) 将得到的位置估计值存入 buffer _hist_position_estimate_x/y/z。这样做的原因是，由于 GPS 信息的传输存在延迟，其数据落后于加速度计得到的数据。将加速度计得到的实时数据存入 buffer，buffer 的容量即延迟时间内的采样数据量，这样，与 GPS 数据进行比较求位置误差时，使用 buffer 中最旧的数据即可消除延迟的影响。

```
1     // store 3rd order estimate (i.e. estimated vertical
          position) for future use
2     _hist_position_estimate_z.push_back(_position_base.z);  ///
          push_back: add an item to the end of the buffer.buffer
          大小: 15
3
4     // store 3rd order estimate (i.e. horizontal position) for
          future use at 10hz
5     _historic_xy_counter++;
6     if( _historic_xy_counter >=
          AP_INTERTIALNAV_SAVE_POS_AFTER_ITERATIONS ) {    ///10,函
          数调用应该是100hz, 因此调用10次存一次数据
7         _historic_xy_counter = 0;
8         _hist_position_estimate_x.push_back(_position_base.x);
              //buffer大小: 5
9         _hist_position_estimate_y.push_back(_position_base.y);
10    }
```

2015.8.4

方酉