Table of Contents


```
function [xhat, meas] = SuvarnaFilter(calAcc, calGyr, calMag)

% FILTERTEMPLATE  Filter template
%
% This is a template function for how to collect and filter data
% sent from a smartphone live.  Calibration data for the
% accelerometer, gyroscope and magnetometer assumed available as
% structs with fields m (mean) and R (variance).
%
% The function returns xhat as an array of structs comprising t
% (timestamp), x (state), and P (state covariance) for each
% timestamp, and meas an array of structs comprising t (timestamp),
% acc (accelerometer measurements), gyr (gyroscope measurements),
% mag (magnetometer measurements), and orint (orientation quaternions
% from the phone).  Measurements not availabe are marked with NaNs.
%
% As you implement your own orientation estimate, it will be
% visualized in a simple illustration.  If the orientation estimate
% is checked in the Sensor Fusion app, it will be displayed in a
% separate view.
%
% Note that it is not necessary to provide inputs (calAcc, calGyr,
 calMag).
```

Setup necessary infrastructure

```
  import('com.liu.sensordata.*');  % Used to receive data.
```

Filter settings

```
  t0 = [];  % Initial time (initialize on first data received)
  nx = 4;   % Assuming that you use q as state variable.
  % Add your filter settings here.
%   Ra = [0.0001   -0.0000   -0.0001;
%    -0.0000    0.0001    0.0002;
%    -0.0001    0.0002    0.0412];
    Ra = [0.1362    0.0711   -0.0002
          0.0711    0.0790    0.0014
        - 0.0002    0.0014    0.0775];

%   Rg = 1.0e-05 *[0.0817    0.0017   -0.0026;
```

```
%                  0.0017    0.1054    0.0008;
%                 -0.0026    0.0008    0.0610];
    Rg = [0.0043   -0.0007   -0.0049
         -0.0007    0.0015    0.0131
         -0.0049    0.0131    0.6355];
  Rm = [31.4348  -14.5106   -1.5389
       -14.5106  104.8704   -0.9395
       -1.5389   -0.9395    4.1759];
  g0 = [0.0939 -0.1733 9.8425]';
  acc_err_margin = norm(g0)*0.2;
  m0 = [0 24.8054 -22.5217]';
   L = norm(m0);
%   mag_err_margin = norm(m0)*0.1;
  % Current filter state.
  x = [1; 0; 0 ;0];
  P = eye(nx, nx);
  alpha = 0.01;

  % Saved filter states.
  xhat = struct('t', zeros(1, 0),...
                'x', zeros(nx, 0),...
                'P', zeros(nx, nx, 0));

  meas = struct('t', zeros(1, 0),...
                'acc', zeros(3, 0),...
                'gyr', zeros(3, 0),...
                'mag', zeros(3, 0),...
                'orient', zeros(4, 0));
  try
```

Create data link

```
    server = StreamSensorDataReader(3400);
    % Makes sure to resources are returned.
    sentinel = onCleanup(@() server.stop());

    server.start();  % Start data reception.

    % Used for visualization.
    figure(1);
    subplot(1, 2, 1);
    ownView = OrientationView('Own filter', gca);  % Used for
 visualization.
    googleView = [];
    counter = 0;  % Used to throttle the displayed frame rate.
```

Filter loop

```
    while server.status()  % Repeat while data is available
      % Get the next measurement set, assume all measurements
      % within the next 5 ms are concurrent (suitable for sampling
      % in 100Hz).
      data = server.getNext(5);
```

```
        if isnan(data(1))  % No new data received
          continue;        % Skips the rest of the look
        end
        t = data(1)/1000;  % Extract current time

        if isempty(t0)  % Initialize t0
          t0 = t;
        end

        acc = data(1, 2:4)';
        if ~any(isnan(acc))  % Acc measurements are available.
%           if abs(norm(acc) - norm(g0))< acc_err_margin
%               [x, P] = mu_g(x, P, acc, Ra, g0);
%               [x, P] = mu_normalizeQ(x, P);
%               ownView.setAccDist(0);
%           else
%               ownView.setAccDist(1);
%           end
      end
        gyr = data(1, 5:7)';
        if ~any(isnan(gyr))  % Gyro measurements are available.
           [x, P] = tu_qw( x, P, gyr, 0.01, Rg);
           [x,P] = mu_normalizeQ(x,P);
        end

        mag = data(1, 8:10)';
        if ~any(isnan(mag))  % Mag measurements are available.
          L = (1-alpha)*L+alpha*norm(mag);
          if abs(L-norm(mag))< L*0.2
              [x, P] = mu_m(x, P, mag, Rm, m0);
              [x, P] = mu_normalizeQ(x, P);
              ownView.setMagDist(0);
          else
              ownView.setMagDist(1);
          end
        end

        orientation = data(1, 18:21)';  % Google's orientation estimate.

        % Visualize result
        if rem(counter, 10) == 0
          setOrientation(ownView, x(1:4));
          title(ownView, 'OWN', 'FontSize', 16);
          if ~any(isnan(orientation))
            if isempty(googleView)
              subplot(1, 2, 2);
              % Used for visualization.
              googleView = OrientationView('Google filter', gca);
            end
            setOrientation(googleView, orientation);
            title(googleView, 'GOOGLE', 'FontSize', 16);
          end
        end
```

```
        counter = counter + 1;

        % Save estimates
        xhat.x(:, end+1) = x;
        xhat.P(:, :, end+1) = P;
        xhat.t(end+1) = t - t0;

        meas.t(end+1) = t - t0;
        meas.acc(:, end+1) = acc;
        meas.gyr(:, end+1) = gyr;
        meas.mag(:, end+1) = mag;
        meas.orient(:, end+1) = orientation;
      end

    catch e
      fprintf(['Unsuccessful connecting to client!\n' ...
        'Make sure to start streaming from the phone *after*'...
              'running this function!']);
    end

end
```

Published with MATLAB® R2019b