CS 8790: Solution to assignment 6

Chanmann Lim

October 27, 2014

Report:

Suppose we have a rover deployed in a remote 2D environment and we would like to move the rover to a targeted position (in the 2D environment) by sending control input commanding the rover to move some distances relative to its current position. However various levels of uncertainties (in the remote environment or in some errors in the design of the rover) might affect the prediction of the location of the rover as it is traversing and to re-adjust our estimate we have the sensor that can observe the rover and provides independent and consistent (unbiased and conservative covariance) observation of its current position then we go through control input and observation loop until the rover reaches the desired location.

In maintaining the estimate of the state of the rover, the control inputs and the observations have to be treated differently. When the control input is sent the best estimate of the new location of the rover is just the sum of the rover's current position and the relative changes in position moreover the covariance of the two measurements also get added together.

$$\begin{aligned} x_{new} &= x + z \\ P_{new} &= P + R \end{aligned}$$

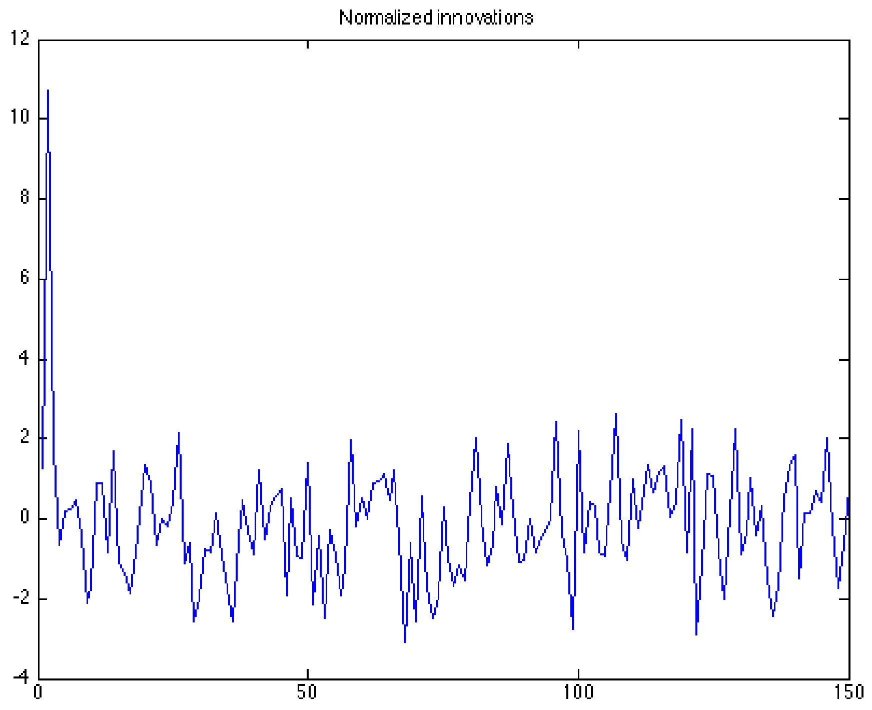
This reduces the value of information about our prediction and only until the new observation coming in can the estimate be fused using the Kalman filter to produce the new result which will lower the uncertainty introduced by prior control inputs.

$$\begin{aligned} P_{new} &= P - WSW^T \\ x_{new} &= x + W(z - Hx) \\ v &= S^{-\frac{1}{2}}(z - Hx) \end{aligned}$$

where

$$\begin{aligned} S &= HPH^T + R \\ W &= PH^T S^{-1} \end{aligned}$$

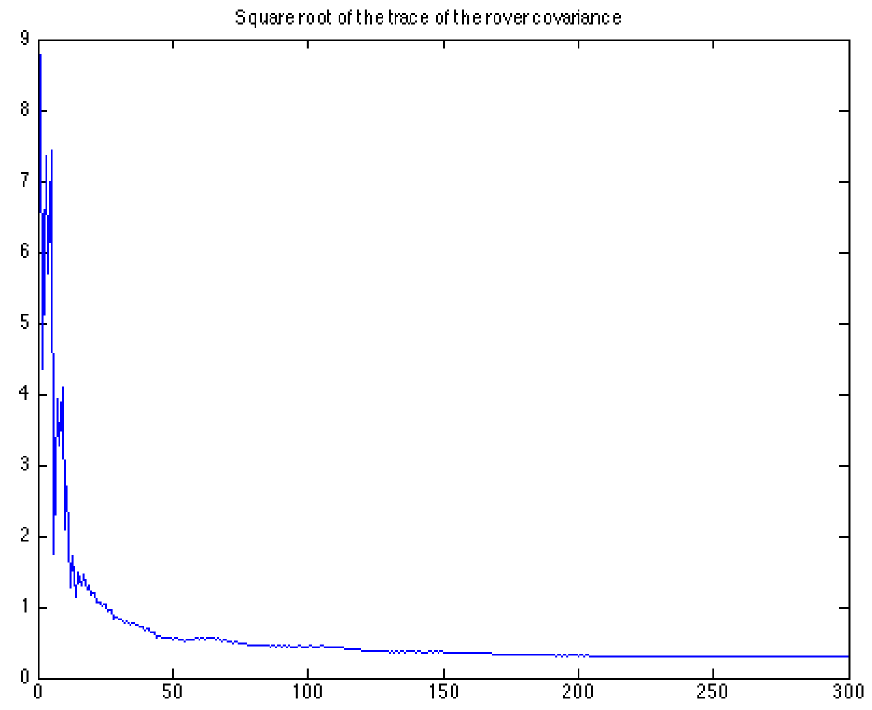
1. The plot of the normalized innovations (for the observation that provides 2 innovations, the first is being chosen for plotting):



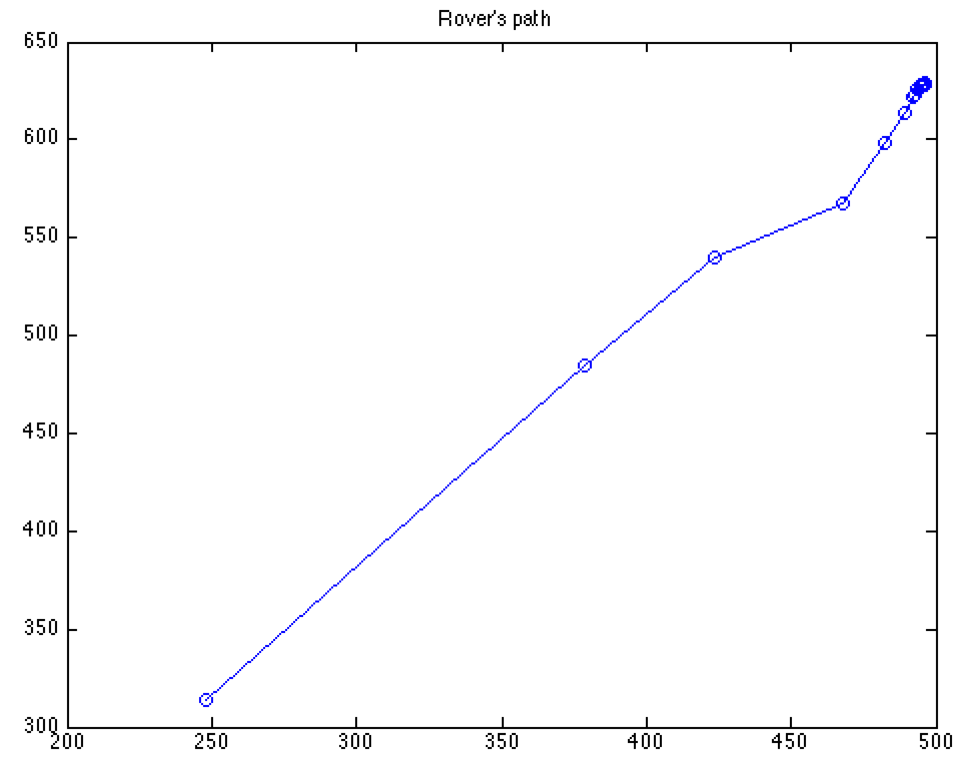
2. The square root of the trace of the rover covariance matrix is the distance from the rover estimated position and the true position calculated by taking the square root of sum of the diagonal values of its covariance matrix P

$$square\_root\_trace = \sqrt{sum(diag(P))}$$

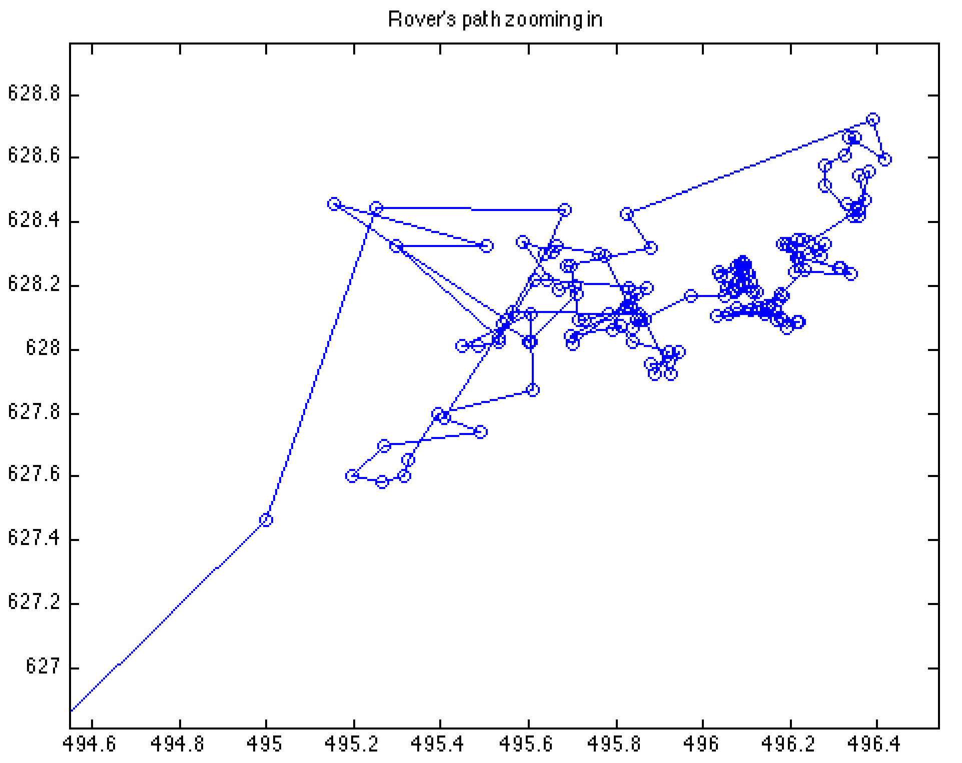
and the following is the plot of the square root of the trace of the rover covariance matrix for both when sending the control input and obtaining the observation :



3. The rover's path position estimates after each control input:



The rover moved a few fairly large distances at the beginning then started taking many granular steps to an exact target on the upper-right corner of the above plot and zooming in to the location will reveal the rover's maneuvers.



4. The observations with innovation size less than 1 is 61.33%.

5. The rover's final estimated position is [495.868838, 628.111014] and the standard deviation can be obtained by taking the square root of diagonal elements of the covariance P of the final position.

$$x_{final} = \begin{bmatrix} 495.868838 \\ 628.111014 \end{bmatrix}$$
$$standard\_deviation = \begin{bmatrix} 0.216657 \\ 0.216158 \end{bmatrix}$$

$$final\_estimate = \begin{pmatrix} 495.868838 & 0.216657 \\ 628.111014 & 0.216158 \end{pmatrix}$$

Appendix:

assignment_6.m

```matlab
% initialize filter
x = zeros(2,1);
P = zeros(2,2);
result = report();

% open data file
fid = fopen('RoverData.txt');
while ~feof(fid)
    tline = fgetl(fid);
    data = sscanf(tline, '%f');

    if ~isempty(data)
        if (data(1) == 0)
            [x, P] = process_control_input(data, x, P);
            result = result.add_rover_path_data(x);
        else
            [z, R, H] = get_observation(data);
            [x, P, v] = update(x, P, z, R, H);
            result = result.add_data(v, x);
        end
        result = result.add_rover_trace_data(P);
    end
end
% close file
fclose(fid);

result.plot();
result.print_innovation_size_percentage();

% print final position
template = 'The final position of the rover =\n%14f, %14f\n%14f, %14f\n';
standard_deviation = sqrt(diag(P));
fprintf(template, x(1), standard_deviation(1), x(2), standard_deviation(2));
```

get_observation.m

```matlab
function [ z, R, H ] = get_observation( data )
%GET_OBSERVATION get observation from data
%   data - column vector containing observation data

    dimension = data(1);
    if dimension == 1
        z = data(2);
        R = data(3);
        H = data(4:5)';
    elseif dimension == 2
        z = data(2:3);
        R = [data(4:5)'; data(5:6)'];
        H = [data(7:8)'; data(9:10)'];
    end
end
```

update.m

```matlab
function [ x, P, v ] = update( x, P, z, R, H )
%UPDATE - update sensor estimate
%   incorporate information from new observation (z, R, H)

    S = (H * P * H') + R;
    W = (P * H') / S;
    P = P - (W * S * W');
    innovation = (z - H * x);
    x = x + W * innovation;
    v = sqrtm(S) \ innovation;
end
```

report.m

```matlab
classdef report
```

```
    properties
        normalized_innovations;
        traces;
        rover_path;
        innovation_sizes;
    end

    methods
        function obj = report()
            obj.normalized_innovations = [];
            obj.traces = [];
            obj.rover_path = [];
            obj.innovation_sizes = [];
        end

        function obj = add_rover_path_data(obj, x)
        % x - mean estimate
            obj.rover_path(:,end + 1) = x;
        end

        function obj = add_data(obj, v, x)
        % v - normalized innovation vector
        % x - mean estimate
            obj.normalized_innovations(end + 1) = v(1);
            obj.innovation_sizes(end + 1) = v'*v / length(x);
        end

        function obj = add_rover_trace_data(obj, P)
        % P - covariance
            obj.traces(end + 1) = sqrt(sum(diag(P)));
        end

        function plot(obj)
            % plot: normalized innovations
            figure
            plot(obj.normalized_innovations)
            title('Normalized innovations')

            % plot: traces
            figure
            plot(obj.traces)
            title('Square root of the trace of the rover covariance')

            % plot: rover path
            figure
            plot(obj.rover_path(1,:), obj.rover_path(2,:), '-o')
            title('Rover''s path')
        end

        function print_innovation_size_percentage(obj)
            template = 'The observations with innovation size less than 1 = %.2f%%\n';
            percent = sum(obj.innovation_sizes < 1) / length(obj.innovation_sizes);
            fprintf(template, percent * 100);
        end
    end
end
```