Mobile Sensing: Spring 2015
Exercise: 5

Due on 9th April 2015 by 17:45 PM.

Instructions: All course participants are requested to submit their exercise solutions electronically to the instructors (samuli.hemminki at cs.helsinki.fi and teemu.pulkkinen at cs.helsinki.fi), as well as to the course lecturer (petteri.nurmi at cs.helsinki.fi) by the due date (latest before the exercise session). In all the exercises, do not just give the answer, but also the derivation how you obtained it. Participants are encouraged to write computer programs to derive solutions to some of the given problems.

Ex 1. Load accelerometer data collected during a tram ride.

a) Estimate and eliminate gravity using mean-filtering based methods presented on the lectures: Mizell (30sec), Nericell (10sec)

b) Estimate and eliminate gravity using an opportunistic method, where near-stationary periods are detected by comparing a fixed threshold value [1] with the variance of accelerometer's L2-norm: $\sqrt{x^2 + y^2 + z^2}$

c) For each gravity estimation result, calculate horizontal and vertical projections of linear acceleration.

While this exercise was quite simple in itself, the 50% overlapping time windowing could get a bit tricky. Projecting the data onto horizontal and vertical frames was also generally problematic. The main idea with the projections were to first project the data along vertical direction (defined by gravity), and then using subtraction from total linear acceleration to define horizontal component. Below is example code in MATLAB for Ex 1 a)-c). Note that 1 a) requires function windownizeByTime. Example code in MATLAB below:

```
%% Ex.1 (a)
```

[1] E.g., variance is ≤ 0.002, over 0.75s frame

```
%tram = webread('http://www.cs.helsinki.fi/u/shemmink/MobileSensing/w5e1_accelerometer.csv'); % F
tram = csvread('w5e1_accelerometer.csv');

% Resample data (optional)
timestep       = median(diff(tram(:,1)));
timestamps_rs  = tram(1,1):timestep:tram(end,1);
tramTS         = timeseries(tram(:,2:4),tram(:,1));
tramRS         = resample(tramTS,timestamps_rs);
tram_rs(:,2:4) = tramRS.Data;
tram_rs(:,1)   = tramRS.Time;

% Construct data frames
overlap = 50;   % Overlap percentage
winLen  = 30;   % Window length (seconds)

divider = 100/(100-overlap);
windows = windownizeByTime(tram_rs,winLen/divider,tram_rs(1,1));

% PreAllocate
[sampleNr,dims] = size(tram_rs(:,2:4));

Gmiz = zeros(sampleNr,dims);
Gner = zeros(sampleNr,dims);

% Loop through the data windows
idx = 1;
for i=1:length(windows)

    % Handle first cases where i<divider
    if i<divider
        k = i;
    else
        k = divider;
    end

    % Concatenate data for overlap
    data = [];
    for j=1:k
       data = [data;windows{i-(j-1)}(:,2:4)];
    end

    % Estimate gravity
    dataL = length(data);
    Gmiz(idx:idx+dataL-1,:) = ones(dataL,1) * mean(data);
    Gner(idx:idx+dataL-1,:) = ones(dataL,1) * median(data);

    % Update index tracker
    idx = idx+length(windows{i-(k-1)});
```

```
end

%% Ex.1 (b)

TH_ACCELVAR = 0.002;

winLen      = round(0.75/timestep); % Samples per window on avrg
inputLen    = length(tram_rs);
frameNr     = floor(inputLen/winLen);
frameRange  = 1:frameNr*winLen;
frames      = reshape(tram_rs(frameRange,2:4),[winLen,frameNr,3]);

frameMean = squeeze(mean(frames));
frameVar  = squeeze(var(frames));
frameNr   = length(frameMean);

% PreAllocate
Gopp = ones(sampleNr,1) * frameMean(1,:);

% Gravity estimation
for i=1:frameNr
    if max(frameVar(i,:)) < TH_ACCELVAR
        Gopp(i*winLen:end,:) = ones(length(i*winLen:sampleNr),1) * frameMean(i,:);
    end
end

% Some plotting on the way to make sure everything is as expected..
clf; plot(tram_rs(:,2:4)); hold on ;
plot(Gmiz,'black','linewidth',2);
plot(Gner,'magenta','linewidth',2);
plot(Gopp,'cyan','linewidth',2);

%% Ex.1 (c)

% PreAllocate
Lmiz = zeros(sampleNr,dims);
Lner = zeros(sampleNr,dims);
Lopp = zeros(sampleNr,dims);

Vmiz = zeros(sampleNr,dims);
Vner = zeros(sampleNr,dims);
Vopp = zeros(sampleNr,dims);

Hmiz = zeros(sampleNr,dims);
Hner = zeros(sampleNr,dims);
Hopp = zeros(sampleNr,dims);

% Project onto vertical/horizontal planes
for i=1:length(tram_rs(:,1))
```

```
    % Remove gravity: Data - G
    Lmiz(i,:) = tram_rs(i,2:4) - Gmiz(i,:);
    Lner(i,:) = tram_rs(i,2:4) - Gner(i,:);
    Lopp(i,:) = tram_rs(i,2:4) - Gopp(i,:);

    % Project data to direction of gravity (vertical)
    Vmiz(i,:) = (dot(Lmiz(i,:),Gmiz(i,:))/dot(Gmiz(i,:),Gmiz(i,:))) * Gmiz(i,:);
    Vner(i,:) = (dot(Lner(i,:),Gner(i,:))/dot(Gner(i,:),Gner(i,:))) * Gner(i,:);
    Vopp(i,:) = (dot(Lopp(i,:),Gopp(i,:))/dot(Gopp(i,:),Gopp(i,:))) * Gopp(i,:);

    % Horizontal Acc. = Total Lin.Acc. - Vertical Acc.
    Hmiz(i,:) = Lmiz(i,:)-Vmiz(i,:);
    Hner(i,:) = Lner(i,:)-Vner(i,:);
    Hopp(i,:) = Lopp(i,:)-Vopp(i,:);

end

% Finally, som plotting to compare results
figure;
subplot(3,1,1); plot(Hmiz);
subplot(3,1,2); plot(Hner);
subplot(3,1,3); plot(Hopp);

figure;
subplot(3,1,1); plot(Vmiz);
subplot(3,1,2); plot(Vner);
subplot(3,1,3); plot(Vopp);
```

Ex 2. Load accelerometer data collected during walking [2]

a) Calculate L2-norm of the signal and perform mean filtering using frames of 10 samples.

b) Calculate the number of steps using peak detection with a predefined threshold (you can visually examine the data to find a suitable threshold).

c) Load the walking data used last week for activity recognition. Calculate steps using same method as above in b). How well does it perform? Can you reason why?

Matlab: *findpeaks*, Python: See page *DetectPeaks*

Mean filtering in Ex.2 could be done simply by taking mean of 10-sample windows. In the subtask c), the data is collected from trouser pocket and hence each other step (for the leg with the device) produces higher acceleration peak. Example code in MATLAB below:

```
%% Ex. 2 (a)
```

[2] Note: modified +80% amplitude from original for this exercise

```
walk = load('w5e2_walking.csv');

% L2-norm & mean-filtering with 10 samples
walkL2      = sqrt(sum(walk.^2,2));

winLen      = 10;
inputLen    = length(walkL2);
frameNr     = floor(inputLen/winLen);
frameRange  = 1:frameNr*winLen;
frames      = reshape(walkL2(frameRange),[winLen,frameNr,1]);

walkF = mean(frames);

% Peak detection
minpeak = 12;
[val,loc] = findpeaks(walkF,'MinPeakHeight',minpeak);
clf; plot(walkF); hold on ; plot(loc,val,'r*');

%% Ex.2 (b)

walk2 = load('../Week4/w4e2_walking.csv');

walk2L2 = sqrt(sum(walk2(:,2:4).^2,2));

winLen      = 10;
inputLen    = length(walk2L2);
frameNr     = floor(inputLen/winLen);
frameRange  = 1:frameNr*winLen;
frames      = reshape(walk2L2(frameRange),[winLen,frameNr,1]);

[val,loc] = findpeaks(walk2L2,'MinPeakHeight',minpeak);
clf; plot(walk2F); hold on ; plot(loc,val,'r*');
```