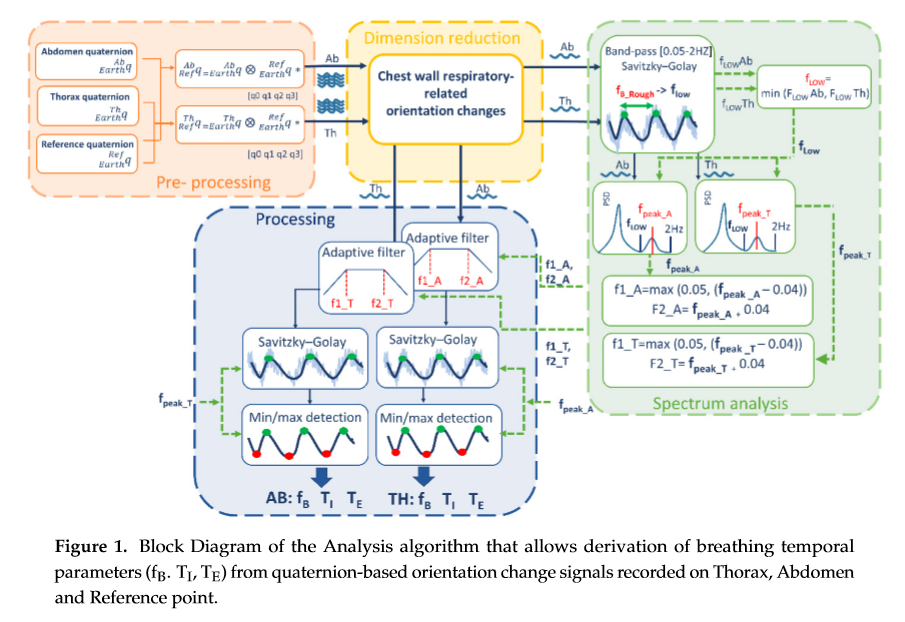
**Analysis algorithm (MATLAB)**

A signal processing procedure was designed to extract the breathing frequency starting from quaternions representing the change of orientation of each unit relative to the earth frame (, ).

The block diagram of the signal processing part is presented in Figure 1. The algorithm is divided into 4 main blocks: (i) pre-processing, (ii) DR, (iii) spectrum analysis, and (iv) processing.



All the steps mentioned regard the abdomen component, but the same procedure is also performed for the thoracic one.

**Pre-processing**

Pre-processing block includes the preliminary steps that leads to chest-wall respiratory-related orientation change signals.

Window selection:

F(i,:)=round(ginput(1));i=i+1; %inizio sessione da tenere, ginput(n) allows you to identify the coordinates of n points.

%To choose a point, move your cursor to the desired location and press either a mouse button or a key on the keyboard.

%Press the Return key to stop before all n points are selected. MATLAB® returns the coordinates of your selected points.

F(i,:)=round(ginput(1));i=i+1; %fine sessione da tenere

tenere

add\_pose\_w=mean(add(F(1,1):F(2,1),:)); %dati addome raw intervallo da tenere

ref\_pose\_w=mean(ref(F(1,1):F(2,1),:));% dati riferimento raw intervallo da tenere

For cycles (creation of an array with length equal to add, ref):

Add\_pose=[Add\_pose; add\_pose\_w]; %values are all equal to add\_pose\_w (mean of the values between F(1,1) and F(2,1))

Ref\_pose=[Ref\_pose; ref\_pose\_w];

Position ZERO computation (quatmultiply: quaternion products, quatconj: quaternion conjugate):

ref\_pose=quatmultiply(ref,quatconj(Ref\_pose)); %calcolo posizione ZERO del riferimento

add\_pose=quatmultiply(add,quatconj(Add\_pose)); %calcolo posizione ZERO dell'addome

Window selection:

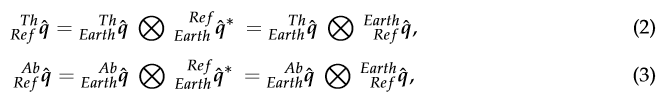
G(i,:)=round(ginput(1));i=i+1; %inizio sessione da tenere

G(i,:)=round(ginput(1));i=i+1; %fine sessione da tenere

Add\_Ok=add\_pose(G(1,1):G(2,1),:); %dati addome raw intervallo da tenere

Ref\_Ok=ref\_pose(G(1,1):G(2,1),:);% dati riferimento e raw intervallo da tenere

The orientations changes of thoracic and abdominal units were referred to the reference unit frame (that in turn represents orientation changes of trunk) applying Equations (2) and (3) respectively:



Compound quaternion:

%calcolo compound quaternion: Addome rispetto a Ref e Torace rispetto a Ref

a1=quatmultiply(Add\_Ok,quatconj(Ref\_Ok));

interp\_A=movmean(a1,97,'includenan'); %returns an array of local 97-point mean values, where each mean is calculated over a sliding window of length 97 across neighboring elements of A. 'includenan' includes all NaN values in the calculation

a1=a1-interp\_A; %subtraction mean value interp\_A obtained with movmean

**Dimension reduction**

Dimension-reduction block takes the quaternions obtained from Equations (2) and (3), that are composed by 4 components each, and provides as output 2 single-component signals (1 for the abdomen and 1 for the thorax) representing chest-wall respiratory-related orientation change signals. These two signals represent the input of the power spectrum block and of the processing block. To reduce dimension from 4 components to 1, PCA-based fusion of the quaternion components is used.

PCA computation:

[coeff\_add,score,latent,tsquared,explainedA,mu] = pca(a1);

FuseA\_1=a1\*coeff\_add(:,1);

[coeff\_tor,score,latent,tsquared,explainedT,mu] = pca(t1);

principal component coefficients (loadings), each column of coeff\_add contains coefficients for one principal component, and the columns are in descending order of component variance.

a1 is n x p (3380 x 4), while coeff\_add is p x p (4 x 4).

FuseA\_1 is the first principal component, which accounts for the largest variance in the data.

**Spectrum analysis**

Spectrum Analysis block includes a set of steps needed to optimize the subsequent processing phase. The two signals representing chest-wall (abdominal and thoracic) respiratory-related orientation obtained downstream of the dimension-reduction block underwent the following steps:

1. Low-frequency threshold selection

* Application of the Savitzky-Golay FIR filter (smoothing) to facilitate subsequent maxima points identification:

EstimSmoothA=sgolayfilt(FuseA\_1, 3,SgolayWindowPCA); % smoothing. Applies a Savitzky-Golay finite impulse response (FIR) smoothing filter of polynomial order 3 and frame length SgolayWindowPCA=31 to the data in vector FuseA\_1

* Maxima detection:

diff=max(EstimSmoothA)-min(EstimSmoothA); %signal amplitude evaluation in order to define a threshold for the maxima identification

thr=diff\*5/100; %threshold 5% of the maximum variation in signal amplitude [M,I] =findpeaks(EstimSmoothA,'MinPeakDistance',6,'MinPeakProminence',thr) %signal peaks; returns a vector with the local maxima (M) of the input signal vector and indices at which the peaks occur.

%Only those peaks that have a relative importance of at least 'MinPeakProminence'(Prominence: how much the peak stands out due to itsintrinsic height and its location relative to other peaks).

%When you specify a value for 'MinPeakDistance', the algorithm chooses the tallest peak in the signal and ignores all peaks within 'MinPeakDistance' of it.

* The rough estimate of fB (fStim)was done by identifying maxima points of the signal and computing the fB, breath by breath, as reciprocal of the temporal distance between consecutive maxima points. Then, the mean (fStimMean) and the standard deviation (fStimstd) of the fB over the entire trial were computed.

for i=1:length(M)-1

intraPicco=(I(i+1)-I(i))/fDispo; %interpeak distance is the estimated time of the duration of every breath. fDispo=10

fStim=1/intraPicco; %the frequency is the inverse of the interpeak time breath by breath

fStimVec=[fStimVec fStim];

end

fStimMean=mean(fStimVec); % the estimated breathing frequency is the mean of the computed frequencies breath by breath

fStimstd=std(fStimVec); %estimated frequency standard deviation

* Low-frequency threshold:

lowThresoldA=max(0.05,(fStimMean-fStimstd));%computation of the low-frequency threshold from the estimated frequency, the spectrum peak will be identified starting from this frequency

The procedure is repeated also for the thoracic component, and then the low-frequency threshold is computed as the minimum between the one derived from the abdomen component and the one derived from the thoracic component:

lowThresold=min([lowThresoldA lowThresoldT]);

2. PSD estimate (Welch's method, Hamming window size: 300 samples, overlapping: 50 samples):

[pxxA,fA] = pwelch(FuseA\_1,300,50,512,10); %PCA\_1 abdomen spectrum (fA is the nomralized frequency vector)

Identification of the spectrum frequency corresponding to the breathing rate by lookingfor the local peak of the PSD within the window [fLOW, 2 Hz] (the use of a low threshold supports the selection of the PSD peak linked to the breathing rate and avoid selecting wrong peaks, often related to low-frequency oscillation artifacts):

Signal=-FuseA\_1; **(?)**

Peaks in the [lowThresold, 2] interval (find return positions):

in = find(fA>lowThresold)-1;

fi=find(fA>2);

[M,I] = findpeaks(pxxA(in(1):fi(1)));

Maximum of the peaks and its position:

[bFMax, BFI]= max(M); %max value and index that corresponds to the maximum value of M

bFSpettro=fA(I(BFI)+in(1)-1); %fpeak is the value of fA at the max position

3. Adaptive band-pass filter settings (centered in fpeak), identification of the cut-off frequencies:

fL: f1=max(0.05,bFSpettro-0.4);

fU: f2=bFSpettro+0.4;

**Processing**

Processing block includes all the steps needed to extract breathing frequency and temporal

parameters from the signals obtained downstream of the dimension-reduction block. Chest-wall

respiratory-related orientation change signals (abdominal and thoracic) underwent the following steps:

1. Adaptive band-pass filter. The signals were band-pass filtered (first-order IIR Butterworth filter filtfilt), with fU(f2) and fL(f1) cut-off frequency points determined within the spectrum analysis block.

%low pass filter

ft\_pl = f2;

fDispo=10;

Wn\_pl= ft\_pl/(fDispo/2);

[b,a] = butter(1,Wn\_pl,'low'); %returns the transfer function coefficients of an first-order lowpass digital Butterworth filter with normalized cutoff frequency Wn\_pl

LowFilt=filtfilt(b,a,Signal); %performs zero-phase digital filtering by processing the input data, Signal

%high pass filter (elimino continua)

ft\_ph=f1;

Wn\_ph= ft\_ph/(fDispo/2);

[b,a] = butter(1,Wn\_ph,'high'); %returns the transfer function coefficients of an first-order highpass digital Butterworth filter with normalized cutoff frequency Wn\_ph

HighFilt=filtfilt(b,a,LowFilt); %performs zero-phase digital filtering by processing the input data, LowFilt

1. Smoothing. Filtered signals were furtherly smoothed (third-order Savitzky–Golay FIR filter)

to simplify subsequent identification of maxima and minima points. The level of smoothing

(window length) was automatically selected based on fpeak, i.e., increasing window length for

decreasing fpeak. Relation between optimal window length values and fpeak values has been

determined empirically.

Parameters selection (for 2. Smoothing and 3. Minima and Maxima detecion)

if bFSpettro\*60<12

perc=15;

distance=35; %min peak distance di 35 frames corrisponde ad una freq respiratoria di 17 resp/min (siamo conservativi)

SgolayWindow=15;

end

if (bFSpettro\*60>12&&bFSpettro\*60<20)

perc=8;

distance=20; %min peak distance di 20 frames corrisponde ad una freq respiratoria di 30 resp/min (siamo conservativi)

SgolayWindow=11;

end

if (bFSpettro\*60>20&&bFSpettro\*60<40)

perc=5;

distance=9; %min peak distance di 12 frames corrisponde ad una freq respiratoria di 50 resp/min (siamo conservativi)

SgolayWindow=9;

end

if (bFSpettro\*60>40&&bFSpettro<59)

perc=4;

distance=7; %min peak distance di 8 frames corrisponde ad una freq respiratoria di 75 resp/min (siamo conservativi)

SgolayWindow=7;

end

if bFSpettro\*60>59

perc=3;

distance=3; %min peak distance di 5 frames corrisponde ad una freq respiratoria di 120 resp/min (siamo conservativi)

SgolayWindow=5;

end

Smoothing:

SmoothSmoothA=sgolayfilt(HighFilt, 3,SgolayWindow);

1. Minima and maxima points detection. A set of optimized parameters (i.e., minimum peak

distance (MPD) and minimum prominence threshold (MPT)) was automatically selected based

on fpeak to optimize recognition of minima and maxima points of the smoothed signals. Optimal

MPD and MPT values depending on fpeak were experimentally determined.

Diff=max(SmoothSmoothA)-min(SmoothSmoothA);

thr=Diff\*perc/100;

[Maxima,MaxIdx] = findpeaks(SmoothSmoothA,'MinPeakDistance',distance,'MinPeakProminence',thr);

DataInv = 1.01\*max(SmoothSmoothA)-SmoothSmoothA; %data inversion, minima become maxima, detectable with findpeaks

MinFindIdx=[];

MinIdx=[];

M=[];

I=[];

Minima=[];

for i=1:length(Maxima)-1

[M I]=max(DataInv(MaxIdx(i):MaxIdx(i+1)));

Minimum=SmoothSmoothA(I+MaxIdx(i)-1);

[minima,MinFindIdx] = findpeaks(DataInv(MaxIdx(i):MaxIdx(i+1)));

MinFindIdx=MinFindIdx+MaxIdx(i)-1;

SelectedMin=max(MinFindIdx);

SelectedMinValue=SmoothSmoothA(SelectedMin);

thr2=2\*Diff/100;

if (abs(SelectedMinValue-Minimum))<abs(thr2)

MinIdx=[MinIdx SelectedMin];

minima=SelectedMinValue;

Minima=[Minima minima];

else

MinIdx=[MinIdx I+MaxIdx(i)];

Minima=[Minima Minimum];

end

end

1. Breathing frequency extraction. Breath by breath, inspiratory time (TI) was computed as the

temporal distance between a minimum point (mi) and the consecutive maximum point (Mi).

Expiratory time (Te) was computed as the temporal distance between the maximum point (Mi)

and the consecutive minimum point (mi); total time (TTOT) was computed as TTOT = TI + TE

[s], duty cycle (DC) was computed as (TI / TTOT) ⋅ 100 [%] and breathing frequency was computed

as 60 / (TTOT) [breaths/minute]. A mean value for each of the above-mentioned parameter was

computed for each trial (~3 min).

for i=1:1:length(MinIdx)

te=(MinIdx(i)- MaxIdx(i))/fDispo;

ti=(MaxIdx(i+1)-MinIdx(i))/fDispo;

Ti=[Ti ti];

Te=[Te te];

ti\_te=ti/te;

Ti\_Te=[Ti\_Te ti/te];

end

for i=1:1:length(MinIdx)-1

ttot=(MinIdx(i+1)-MinIdx(i))/fDispo;

bf=1/ttot\*60;

T=[T; ttot];

bF=[bF; bf];

end

%paramters' mean values

Tmean\_PCA=mean(T);

Timean\_PCA=mean(Ti);

Temean\_PCA=mean(Te);

bFmean\_PCA=mean(bF);

bFSpettro\_PCA=bFSpettro\*60;

Ti\_Te\_PCA\_A=mean(Ti\_Te);

duty\_PCA\_A=mean(Ti(1:end-1)./T');

%dev. standard

Tmean\_sd=std(T);

Timean\_sd=std(Ti);

Temean\_sd=std(Te);

bFmean\_sd=std(bF);

bFSpettro\_sd=bFSpettro\*60;

Ti\_Te\_sd\_A=std(Ti\_Te);

duty\_sd\_A=std(Ti(1:end-1)./T');

PCA\_A=[bFmean\_PCA Timean\_PCA Temean\_PCA Ti\_Te\_PCA\_A duty\_PCA\_A];

SD\_A=[bFmean\_sd Timean\_sd Temean\_sd Ti\_Te\_sd\_A duty\_sd\_A];

The whole procedure is performed also for the thoracic component. Afterwards, the two components are summed and the results are computed even for the total component.