A121 Breathing Reference Application User Guide

User Guide

A121 Breathing Reference Application User Guide

User Guide

Author: Acconeer AB

Version:a121-v1.8.1

Acconeer AB November 8, 2024

Contents

1 Acconeer SDK Documentation Overview

To better understand what SDK document to use, a summary of the documents are shown in the table below.

Table 1: SDK document overview.

Name	Description	When to use
	RSS API documentation (html)	
rss_api	The complete C API documentation.	- RSS application implementation - Understanding RSS API functions
	User guides (PDF)	
A121 Assembly Test	Describes the Acconeer assembly test functionality.	- Bring-up of HW/SW - Production test implementation
A121 Breathing Reference Application	Describes the functionality of the Breathing Reference Application.	- Working with the Breathing Reference Application
A121 Distance Detector	Describes usage and algorithms of the Distance Detector.	- Working with the Distance Detector
A121 SW Integration	Describes how to implement each integration function needed to use the Acconeer sensor.	- SW implementation of custom HW integration
A121 Presence Detector	Describes usage and algorithms of the Presence Detector.	- Working with the Presence Detector
A121 Smart Presence Reference Application	Describes the functionality of the Smart Presence Reference Application.	- Working with the Smart Presence Reference Application
A121 Sparse IQ Service	Describes usage of the Sparse IQ Service.	- Working with the Sparse IQ Service
A121 Tank Level Reference Application	Describes the functionality of the Tank Level Reference Application.	- Working with the Tank Level Reference Application
A121 Touchless Button Reference Application	Describes the functionality of the Touchless Button Reference Application.	- Working with the Touchless Button Reference Application
A121 Parking Reference Application	Describes the functionality of the Parking Reference Application.	- Working with the Parking Reference Application
A121 STM32CubeIDE	Describes the flow of taking an Acconeer SDK and integrate into STM32CubeIDE.	- Using STM32CubeIDE
A121 Raspberry Pi Software	Describes how to develop for Raspberry Pi.	- Working with Raspberry Pi
A121 Ripple	Describes how to develop for Ripple.	- Working with Ripple on Raspberry Pi
XM125 Software	Describes how to develop for XM125.	- Working with XM125
XM126 Software	Describes how to develop for XM126.	- Working with XM126
I2C Distance Detector	Describes the functionality of the I2C Distance Detector Application.	- Working with the I2C Distance Detector Application
I2C Presence Detector	Describes the functionality of the I2C Presence Detector Application.	- Working with the I2C Presence Detector Application
I2C Breathing Reference Application	Describes the functionality of the I2C Breathing Reference Application.	- Working with the I2C Breathing Reference Application
	Handbook (PDF)	
Handbook	Describes different aspects of the Acconeer offer, for example radar principles and how to configure	- To understand the Acconeer sensor - Use case evaluation
	Readme (txt)	
README	Various target specific information and links	- After SDK download

2 Breathing Reference Application

This reference application shows how the breathing rate of a stationary person can be estimated using the A121 sensor.

The algorithm consists of the following key concepts:

- Determine the distance to the person using the presence algorithm.
- Form a time series, reflecting the breathing motion over time.
- Remove irrelevant signal content by applying a bandpass filter to the time series.
- Estimate the breathing rate by calculating the power spectrum of the time series.

Each concept is explained in more detail in the following sections.

2.1 Determine distance to person

The algorithm utilize the presence algorithm to determine the distance to the person, located somewhere in the measurement range. The measurement range is defined through the reference application configuration parameters *start_m* and *end_m*.

The presence processor is run for a configurable duration of time, determined by *distance_determination_duration*. If no presence is detected during this period, a new measurement period is initiated. This procedure is repeated until a person is detected.

When presence is detected, the corresponding distance estimates (outputted by the presence algorithm) are passed through a lowpass filter. The final output of the filter, after the configured duration has elapsed, is used as the distance to the person.

Once the distance to the person has been determined, a sub-segment of the measured range is analyzed to estimate the breathing rate. The segment is centered around the estimated distance and the width is determined by the parameter *num_distances_to_analyze*. The value of the parameter should be determined by evaluating the specific use case at hand. A larger number will yield more distances being fed through the algorithm, providing more information, but also increase processing and memory usage. Also, a too large number can potentially result in distances containing no breathing being fed to the algorithm, introducing more noise to estimation process.

The usage of the presence processor can be disabled through the user parameter *use_presence_processor*. If disabled, the full measurement range is analyzed. In this case, it is important to narrow the measured range to the interval where the breathing motion is present.

2.2 Form time series

Once the segment to be analyzed has been identified, a FIFO buffer is used to store the time series, characterizing the breathing motion at each distance in the segment.

The concept for estimating the breathing motion utilize processing, similar to what is described in the phase tracking example. For details, see the phase tracking documentation.

The sparse IQ data service produce complex data samples where the amplitude corresponds to the amount of measured energy and the phase to the relative timing of the transmitted and returning pulse. A displacement of the reflecting object results in a change of this relative phase. The difference in phase between two consecutive measurements can therefor be converted to the corresponding relative change in distance to the reflecting object. The algorithm takes advantage of this by cumulating the relative changes in phase and thereby track the motion of the chest of the breathing person.

The result is stored in the previously mentioned FIFO buffer. The length of the buffer depends on the selected time series length (*time_series_length_s*), frame rate (*frame_rate*). The number of buffers is determined by the number of distance to be analyzed (*num_distances_to_analyze*).

2.3 Bandpass filter

The purpose of the bandpass filter is to remove irrelevant content in the signal before further processing. When configuring the application, the user specifies the lowest and highest anticipated breathing rates through the configuration parameters *lowest_breathing_rate* and *highest_breathing_rate*. These values are used when defining the parameters of the bandpass filter.

After filtering, low frequency components, including bias, and high frequency components are suppressed, resulting in a more easily processed time series.

2.4 Estimate power spectrum

The breathing rate is estimated by identifying the peak location in the Power Spectral Density (PSD) of the time series.

As the frequency bins of the PSD are discrete, peak interpolation is utilized to further improve the estimation accuracy.

The PSD is not calculated at each time step as the majority of the FIFO buffer consists of the same data it did during the previous time step. Instead, the PSD is analyzed once half of the buffer contains new data, e.g., if the time series length is 20 s, there will be 10 s between evaluations of the PSD.

2.5 Application states

The application utilize a state variable with the following states to track the status of the algorithm.

- *NO_PRESENCE_DETECTED*: The algorithm did not detect any presence. If no presence is found, the algorithm initiates a new search.
- *INTRA_PRESENCE_DETECTED*: Intra presence has been detected. Intra presence corresponds to a fast or large motion. If detected, the breathing analysis is paused. Once the intra presence is no longer detected, the distance to the person is again estimated as it might have changed due to the movement.
- *DETERMINE_DISTANCE_ESTIMATE*: Determining the distance to the person using the presence processor.
- *ESTIMATE_BREATHING_RATE*: Estimating the breathing rate in the segment where a person has been located.

The state is returned as a part of the reference application result, *app_state*

2.6 Calibration hints

This section outlines a number of recommendations when calibrating the reference application.

- Use the presence processor to determine the distance to the person by setting *use_presence_processor* to True.
- Set *num_distances_to_analyze* so that the majority of the peak of the presence score is included in the breathing analysis.
- Use a duration between 5-10 s for the *distance_determination_duration*.
- Adjust the intra presence threshold to minimize false triggers when breathing normally, but detecting anticipated movements. Anticipated movements refers to motions that does not originate from regular breathing, such as a person changing position.
- Select as high *profile* as possible, while avoiding interference with the direct leakage. A larger starting point (*start_m*) allows for a higher profile.
- Adjust *sweeps_per_frame* to get good performance of the presence processor. The default value works well in general, but might have to be increased when measuring at longer distances.
- Once the sweeps per frame has been set, increase *hwaas* to achieve better SNR.
- If needed, the *frame_rate* can be lowered from the default of 20 Hz to reduce the memory and power consumption. A suitable value for an embedded application is 5-10Hz.

Use the predefined presets as a starting point and then tweak if necessary.

2.7 Practical considerations

This section outlines a number of practical considerations when getting started with the breathing reference application.

- Start with one of the recommended presets, and then tune parameters if necessary.
- If there is a need to change the dynamics of the presence processor, do the tuning in the presence algorithm GUI as there is more visual feedback related to the presence algorithm. Once new parameter values has been determined, transfer them to the breathing reference application.
- When running the breathing reference application, aim the sensor towards the chest and stomach of the person for best performance.
- Use a lens when measuring at distances greater than 1 meter.

2.8 Tests

This section presents results from testing the algorithm in various scenarios.

Test setup

The tests were performed with an adult sitting, an adult lying down and an infant laying down. The data collection and processing was done with the breathing reference application, available in Exploration tool. The following pictures illustrates the setup.

Configuration

The presets, available in the Exploration tool, were used when testing. In the case when the person is lying down at 2 meters(presented below), the end point of the sitting preset was extended to 2.5 m.

Results

The results from the testing are reported in the following table.

Table 2: Breathing reference application test results.

Case	Distance to person (m)	Actual rate (bpm)	Estimate rate (bpm)
Adult sitting	1.0	15.9	15.0
Adult lying down	1.0	18.3	18.0
Adult lying down	2.0	8.7	9.0
Infant lying down	0.5	18.4	18.0

3 Memory

3.1 Flash

The reference application compiled from ref_app_breathing_main.c on the XM125 module requires around 90 kB.

3.2 RAM

The RAM can be divided into three categories, static RAM, heap, and stack. Below is a table for approximate RAM for an application compiled from ref_app_breathing_main.c.

RAM	**Size (kB)**	
Preset	*Sitting*	*Infant*
Static	1	1
Heap	15	15
Stack	3	3
Total	19	19

4 Power Consumption

Average current	**Current (mA)**	
Preset	*Sitting*	*Infant*
	10	6.5

5 Disclaimer

The information herein is believed to be correct as of the date issued. Acconeer AB (“Acconeer”) will not be responsible for damages of any nature resulting from the use or reliance upon the information contained herein. Acconeer makes no warranties, expressed or implied, of merchantability or fitness for a particular purpose or course of performance or usage of trade. Therefore, it is the user’s responsibility to thoroughly test the product in their particular application to determine its performance, efficacy and safety. Users should obtain the latest relevant information before placing orders.

Unless Acconeer has explicitly designated an individual Acconeer product as meeting the requirement of a particular industry standard, Acconeer is not responsible for any failure to meet such industry standard requirements.

Unless explicitly stated herein this document Acconeer has not performed any regulatory conformity test. It is the user’s responsibility to assure that necessary regulatory conditions are met and approvals have been obtained when using the product. Regardless of whether the product has passed any conformity test, this document does not constitute any regulatory approval of the user’s product or application using Acconeer’s product.

Nothing contained herein is to be considered as permission or a recommendation to infringe any patent or any other intellectual property right. No license, express or implied, to any intellectual property right is granted by Acconeer herein.

Acconeer reserves the right to at any time correct, change, amend, enhance, modify, and improve this document and/or Acconeer products without notice.

This document supersedes and replaces all information supplied prior to the publication hereof.