**NB 2.0 AAU Design Specification  
Tracker Retrospective Detection**

**V1.2**

|  |  |  |
| --- | --- | --- |
| **Approved by:** |  |  |
|  | *Software Engineering Manager*  *CJ Bourn* | *Date* |

Table of contents

[1 Document Change History 4](#_Toc21533486)

[2 Overview 5](#_Toc21533487)

[2.1 Document Scope 6](#_Toc21533488)

[2.2 Document Management 6](#_Toc21533489)

[2.3 References 6](#_Toc21533490)

[2.3.1 Internal Autoliv References 6](#_Toc21533491)

[2.3.2 External References 6](#_Toc21533492)

[2.4 Requirements 6](#_Toc21533493)

[2.4.1 Derived Requirements 7](#_Toc21533494)

[2.5 Architecture 7](#_Toc21533495)

[2.6 Definitions 7](#_Toc21533496)

[2.7 Abbreviations 7](#_Toc21533497)

[2.8 Source Files 7](#_Toc21533498)

[3 details 8](#_Toc21533499)

[3.1 Retrospective Detection parameter and Waveform Updates 8](#_Toc21533500)

[3.2 Retrospective Detection Operation 8](#_Toc21533501)

[3.3 Memory Usage 9](#_Toc21533502)

[3.3.1 Scratch (Re-usable) Memory 10](#_Toc21533503)

[4 Application Programming Interface 11](#_Toc21533504)

[4.1 Function Definitions 11](#_Toc21533505)

[4.2 Function Definitions 11](#_Toc21533506)

[4.2.1 Macros 11](#_Toc21533507)

[4.2.2 RetroDetect\_MatchElem\_t structure type 11](#_Toc21533508)

[4.2.3 RetroDetect\_Matches\_t structure type 12](#_Toc21533509)

[4.2.4 RetroDetect\_GlobalDeltaVec\_t structure type 12](#_Toc21533510)

[4.2.5 RetroDetect\_ScratchMem\_t structure type 12](#_Toc21533511)

[4.2.6 RetroDetect\_State\_t structure type 13](#_Toc21533512)

[4.2.7 Error Codes 13](#_Toc21533513)

[4.2.8 Tracker\_RetroDetectInit 13](#_Toc21533514)

[4.2.9 Tracker\_RetroDetect 14](#_Toc21533515)

[5 Unit Test Considerations 15](#_Toc21533516)

[5.1 Environment 15](#_Toc21533517)

[5.2 Use Case Details 15](#_Toc21533518)

# Document Change History

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Author** | **Description** |
| 0.1 | 12/14/2017 | Oliver McDonald | Initial Draft |
| 0.2 | 2/2/2018 | Chaitra Alase G | Added unit test cases |
| 1.0 | 2/26/2018 | CJ Bourn | Remove watermarks and release |
| 1.1 | 10/09/2018 | Chaitra Alase G | Added NB MathLib – Qsort document link in Reference section. Updated the PTC Links with GIT links and Autoliv SharePoint Links with Veoneer SharePoint Links. Updated Structure Types. Updated Function definitions. Updated Error Code. Updated Test cases. |

# Overview

This document is a Design Specification that covers the RSP Tracker Retrospective Detection AAU for the NB 2.0 sensor. This AAU is utilized as part of the RSP Tracker layer.

The Tracker Retrospective Detection AAU is tasked with building clusters of qualified paired CW and FM detections records and assembling such lists for use in the initiation of tracks in the tracker layer. It is a key function within the tracker layer and is used in specifically in assisting track initiation. Figure 1 identifies the key inputs/outputs and processing performed by the Tracker Retrospective Detection AAU.



Figure Retrospective Detection Overview

## Document Scope

This document describes the NB 2.0 software component suitable for performing Tracker Retrospective Detection as part of RSP Tracking Layer.

The low-level mechanics of the Tracker Retrospective Detection process are beyond the scope of this document; for that, the reader is referred to the Tracker Requirements document [INT5].

## Document Management

The document source is available in [GIT](http://us02-bitb01.corp.int:7990/projects/NB20/repos/doc_platform/browse/Designs/RSP/Tracker/NB%202.0%20RSP%20Tracker%20Retrospective%20Detection%20Design%20Spec.docx) and the released versions of this document are available on the [NB Sharepoint page](https://veoneer.sharepoint.com/sites/aelnb/Project%20Documents/Forms/AllItems.aspx?RootFolder=%2Fsites%2Faelnb%2FProject%20Documents%2FAS151%20-%20NB%202%2E0&FolderCTID=0x0120001580D516F8C9D94C87B80E97AA31BAAE&View=%7B3D69C494-612A-4A09-B701-1B468D62EFF6%7D).

## References

This section lists all document references within this specification.

### Internal Autoliv References

| **Ref #** | **Document** | **Location** |
| --- | --- | --- |
| 1. NB 2.0 Source Code Format and Coding Conventions | | [GIT](http://us02-bitb01.corp.int:7990/projects/NB20/repos/doc_platform/browse/Methods/NBM_2.0-007%20Source%20Code%20Format%20and%20Coding%20Conventions.docx) |
| 1. NB 2.0 RSP Software Architecture Specification | | [GIT](http://us02-bitb01.corp.int:7990/projects/NB20/repos/doc_platform/browse/Architecture/NB%202.0%20Radar%20Signal%20Processor%20Software%20Architecture%20Specification.docx) |
| 1. Eiger Reference manual (Eiger\_6M\_Cut2.1\_RM\_Rev3.pdf) | | [Sharepoint](https://veoneer.sharepoint.com/:b:/s/aelnb/EVTz4GIOLO5AnNBH5wr1S8wBFyACKSKNuTgzigiXmUbrFA) |
| 1. NB 2.0 RSP Process Manager design spec | | [GIT](http://us02-bitb01.corp.int:7990/projects/NB20/repos/doc_platform/browse/Designs/RSP/Process%20Manager/NB%202.0%20RSP%20Process%20Manager%20Design%20Specifications.docx) |
| 1. SW\_16\_Tracker\_Requirements | | [Sharepoint](https://veoneer.sharepoint.com/sites/aelnb/NB%2020%20Requirements/ADD%20Library/SW16_Tracker_Requirements.pdf) |
| 1. NB 2.0 RSP Tracker Manager AAU Design Spec | | [GIT](http://us02-bitb01.corp.int:7990/projects/NB20/repos/doc_platform/browse/Designs/RSP/Tracker/NB%202.0%20RSP%20Tracker%20Manager%20Design%20Spec.docx) |
| 1. NB 2.0 MathLib QSORT Design Specification | | [GIT](http://us02-bitb01.corp.int:7990/projects/NB20/repos/doc_platform/browse/Designs/MathLib/NB%202.0%20MathLib%20QSORT%20Design%20Specification.docx) |

### External References

| **Ref #** | **Document** | **Location** |
| --- | --- | --- |
| 1. <Document Title> | | <short URL> |

## Requirements

Requirements for this module are available in SW\_16\_Tracker\_Requirements document [INT5].

|  |  |
| --- | --- |
| **DOORS ID** | **Summary** |
|  |  |

### Derived Requirements

Following are the derived requirements uncovered by this design. These requirements will have to be added to DOORS by Requirements Engineers. Requirements that are struck out are no longer in effect. They are still listed here to preserve traceability.

|  |  |
| --- | --- |
| **ID** | **Summary** |
|  | |

## Architecture

This AAU is described in the NB2.0 RSP Software Architecture Specification [INT2].

## Definitions

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Channel | CW/SFM data stream corresponding to a single receiver/transmitter pair |
| DMEM | Eiger microcontroller high-speed *data* memory associated with a single processing core. [INT3] |

## Abbreviations

|  |  |
| --- | --- |
| **Item** | **Definition** |
| AAU | Atomic Architectural Unit |
| ADD | Algorithm Description Document |
| API | Application Programming Interface |
| CW | Continuous Wave |
| FFT | Fast Fourier Transform |
| IQ | In-phase and Quadrature |
| NB | Narrow Band |
| UT | Unit Test |

## Source Files

In accordance with the naming convention NAM\_004 in [INT1], the unique short name of the Tracker Retropsective Detection AAU will be RetroDetect in upper CamelCase and retroDetect in lower camelCase. The source code will be contained in Tracker\_**RetroDetect.c** and **RetroDetect.h** and other files as needed, located in folder …/Software/**SigProc**/**Tracker**.

# details

Figure 2 depicts the interaction between this AAU and other RSP components.



Figure Retrospective Detection Component Interactions

## Retrospective Detection parameter and Waveform Updates

The retrospective detection AAU will obtain its parameters such as tracker parameters and waveform segment parameters as pointers passed to its function argument from the tracker manager. These parameters are obtained by execution of a notional Tracker\_Initialize function i.e. Tracker\_Init() which is spelled out in the Tracker Manager AAU Design Specifications [INT6].

## Retrospective Detection Operation

Figure 3 provides a notional overview of the Retrospective Detection AAU as it operates within the context of tracking.



Figure Tracker Retrospective Detection Context and Sequence

A few key details about the processing overseen by this AAU.

* The basis for the retrospective detection process are historical detection records for both CW and SFM detections for a total of a specified integer number of radar sweeps as determined by a point.
* The number of sweeps for Retrospective Detection is obtained from the Tracker Parameters data structure that is populated by the Tracker Manager and passed as a pointer to the Tracker Retrospective Detection AAU.
* The Retrospective Detection AAU creates a global sorted list of potential candidates from the match list. Refer to NB 2.0 MathLib QSORT Design Specification for sorting operation [INT7].
* The Retrospective Detection AAU produces qualified detection clusters that are used in Track Initiation AAU.

## Memory Usage

As per the Scratch Memory Management section, this AAU will utilize 13088 (RETRODETECT\_REQ\_SCRATCH\_BYTES) bytes of scratch memory.

### Scratch (Re-usable) Memory

| **Structure** | **Memory Requirement (Bytes)** |
| --- | --- |
| RetroDetect\_ScratchMem\_t | 13088 |

# Application Programming Interface

## Function Definitions

| **Function** | **Description** |
| --- | --- |
| Tracker\_RetroDetectInit | This function initializes retrospective detection AAU |
| Tracker\_RetroDetect | Generates cluster information based on detections. This function provides qualified clusters of detections to be further examined in track initiation as possible new tracks. |

## Function Definitions

### Macros

| **Macros Name** | **Value** | **Notes/Definition** |
| --- | --- | --- |
| RETRODETECT\_REQ\_  SCRATCH\_BYTES | sizeof(struct RetroDetect\_  ScratchMem\_t) | Required scratch memory in bytes for retrospective detection AAU. |
| RETRODETECT\_MAX\_  ADDED\_MATCHES\_PER\_DET | 8 | Number of new match detection duplicates allowed |
| RETRODETECT\_  MAX\_DELTAVEC | (RETRODETECT\_MAX\_  ADDED\_MATCHES\_PER\_DET \* MAX\_FM\_DET) | duplicates \* max det (30) |
| RETRODETECT\_  MAX\_MATCH | ((RETRODETECT\_MAX\_  ADDED\_MATCHES\_PER\_DET + 3) \* MAX\_FM\_DET) | (duplicates \* max det) +  extra 90 detections matches (3 hist buffer \* max det) |

### RetroDetect\_MatchElem\_t structure type

This structure used to store a match detection score and FmIdx position

| **Member Name** | **Type** | **Notes/Definition** |
| --- | --- | --- |
| Score | float\_t | Match detection score |
| Range | float\_t | Match detection range |
| Velocity | float\_t | Match detection Velocity |
| DeltaR | float\_t | Match detection element DeltaR |
| DeltaC | float\_t | Match detection element DeltaC |
| Cidx | uint16\_t | Match detection element Cidx |
| Nidx | uint16\_t | Match detection element Nidx |
| HistIdx | uint8\_t | Match detection element HistIdx |

### RetroDetect\_Matches\_t structure type

This structure used to store viable match detections for retrospective processing. It is sized to the maximum possible number of detections, but the number of entries in the list is based on the Num member.

| **Member Name** | **Type** | **Notes/Definition** |
| --- | --- | --- |
| Match | RetroDetect\_MatchElem\_t  [RETRODETECT\_MAX\_MATCH] | Match detection structure |
| NewDetectNum | uint16\_t | Number of new non-broadside detections of interest |
| Num | uint16\_t | Number of total non-broadside matches of interest |

### RetroDetect\_GlobalDeltaVec\_t structure type

This structure used to store the sorted score array of the potential candidates (along with supporting information). This sorted list is used to create the detection clusters.

| **Member Name** | **Type** | **Notes/Definition** |
| --- | --- | --- |
| ScoreVal | float\_t [RETRODETECT\_MAX\_DELTAVEC] | Score value |
| SortedScoreVal | float\_t [RETRODETECT\_MAX\_DELTAVEC] | Sorted score value |
| Nidx | uint16\_t [RETRODETECT\_MAX\_DELTAVEC] | Number of indices |
| SortedIdx | uint16\_t [RETRODETECT\_MAX\_DELTAVEC] | Sorted index |
| SortScratchMem | uint16\_t [RETRODETECT\_MAX\_DELTAVEC] | Scratch memory used for Sorting |
| DeltaVectNum | uint16\_t [RETRODETECT\_MAX\_DELTAVEC] | Number of delta vector entries |
| CandAssigned | bool [RETRODETECT\_MAX\_DELTAVEC] | Candidate assigned |

### RetroDetect\_ScratchMem\_t structure type

This structure defines the usage of scratch memory to support retrospective processing.

| **Member Name** | **Type** | **Notes/Definition** |
| --- | --- | --- |
| Matches | RetroDetect\_MatchElem\_t | Match detection structure |
| GlobalDeltaVec | RetroDetect\_GlobalDeltaVec\_t | Global delta vector |
| Assigned | uint16\_t[MAX\_FM\_DET]  [MAX\_DETECTION\_HISTORY\_SIZE] | Assigned detection. |

### RetroDetect\_State\_t structure type

structure encapsulating RetroDetect state.

| **Member Name** | **Type** | **Notes/Definition** |
| --- | --- | --- |
| ScratchMemPtr | RetroDetect\_ScratchMem\_t\* | pointer to Tracker Retro scratch memory data |
| W0Range | float\_t | “Range weighting” applied to determine the best detectionn score. |
| W0Bearing | float\_t | “Bearing weighting” applied to determine the best detectionn score. |
| W0Vel | float\_t | “Velocity weighting” applied to determine the best detectionn score. |
| W0Cent | float\_t | “Centroid weighting” applied to determine the best detectionn score. |
| W0RangeCw | float\_t | “CW Range weighting” applied to determine the best detectionn score. |
| W0VelCw | float\_t | “CW velocity weighting” applied to determine the best detectionn score. |
| Init | bool | Flag to determine AAU state initialization. |

### Error Codes

| **Code** | **Notes/Definition** |
| --- | --- |
| NB\_OK | Indicates successful operation |
| RETRODETCT\_E\_ARG | Invalid input argument detected |
| RETRODETECT\_E\_CONFIG | Necessary initialization not completed |
| RETRODETECT\_E\_OUTOFMEMORY | Insufficient scratch memory provided |

### Tracker\_RetroDetectInit

This function initializes retrospective detection AAU.

Prototype:

PUBLIC\_API (Nb\_Status\_t) Tracker\_RetroDetectInit()

Parameters:

| **Name** | **Purpose** |
| --- | --- |
| State | Pointer to RetroDetect state  (Type: RetroDetect\_State\_t\*) |
| WP | Pointer to CW & FM waveform Segment Parameters  (Type: RspCmn\_WaveformInfo\_t\*) |
| TP | Pointer to Tracker Parameters  (Type: RspCmn\_TrackerBasicParams\_t\*) |
| MemCfg | Pointer to scratch memory configuration  (Type: RspCmn\_ScratchMemDesc\_t\*) |

Returns:

One of the error codes listed in section 4.2.6.

### Tracker\_RetroDetect

Caches detection parameters for use in target and clutter detection. This function should be called as part of initialization process and in response to a mode change.

Prototype:

PUBLIC\_API (Nb\_Status\_t) Tracker\_RetroDetect()

Parameters:

| **Name** | **Purpose** |
| --- | --- |
| State | Pointer to RetroDetect state  (Type: RetroDetect\_State\_t\*) |
| WaveformInfo | Pointer to CW & FM Segment Parameters  (Type: RspCmn\_WaveformInfo\_t\*) |
| TP | Pointer to Tracker Parameters  (Type: RspCmn\_TrackerBasicParams\_t\*) |
| DetectHistory | Pointer to CW and FM Detection History Buffer  (Type: RspCmn\_TrackerDetectionHist\_t\*) |
| DetectCluster | Pointer to Candidate Cluster  (Type: RspCmn\_TrackClusterArray\_t\*) |

Returns:

One of the error codes listed in section 4.2.6.

# Unit Test Considerations

## Environment

All unit tests will be implemented within the Visual Studio environment and run automatically with the nightly builds. No other equipment is required to run the unit tests.

## Use Case Details

|  |  |  |
| --- | --- | --- |
| Test Case ID | Tracker\_RetroDetect\_BadParams | |
| Purpose | Demonstrate the ability of Tracker\_RetroDetect() function to detect invalid input parameters. | |
| Requirement | [INT5] | |
| Setup | NA | |
| Test Preconditions | NA | |
| Description | Provide invalid input parameters to Tracker\_RetroDetect() function and check for error code RETRODETECT\_E\_ARG reflecting invalid input arguments are detected. | |
| Input(s) | Scenario 1:  State == NULL  WaveformInfo = valid pointer  TP = valid pointer  DetectHistory = valid pointer  DetectCluster = valid pointer  VehData = valid pointer | Scenario 2:  State == valid pointer  WaveformInfo = NULL  TP = valid pointer  DetectHistory = valid pointer  DetectCluster = valid pointer  VehData = valid pointer |
| Scenario 3:  State == valid pointer  WaveformInfo = valid pointer  TP = NULL  DetectHistory = valid pointer  DetectCluster = valid pointer  VehData = valid pointer | Scenario 4:  State == valid pointer  WaveformInfo = valid pointer  TP = valid pointer  DetectHistory = NULL  DetectCluster = valid pointer  VehData = valid pointer |
| Scenario 5:  State == valid pointer  WaveformInfo = valid pointer  TP = valid pointer  DetectHistory = valid pointer  DetectCluster = NULL  VehData = valid pointer | Scenario 6:  State == valid pointer  WaveformInfo = valid pointer  TP = valid pointer  DetectHistory = valid pointer  DetectCluster = valid pointer  VehData = NULL |
| Scenario 7:  State == valid pointer  WaveformInfo = valid pointer  TP = valid pointer  DetectHistory = valid pointer  DetectCluster = valid pointer  VehData = valid pointer  State->Init == false | |
| Expected Output(s) | Return value RETRODETECT\_E\_ARG | |

|  |  |
| --- | --- |
| Test Case ID | Tracker\_RetroDetect\_Success1 |
| Purpose | Demonstrate successful operation of Tracker\_RetroDetect() function. |
| Requirement | [INT5] |
| Setup | NA |
| Test Preconditions | NA |
| Description | Provide valid input parameters to Tracker\_RetroDetect() function and call the function from 1 to 200 frames for 20171025\_FIT\_JARI\_NCAP\_FE data file. Compare the output of each frame with the previously computed results. |
| Input(s) | Data files:  20171025\_FIT\_JARI\_NCAP\_FE/VehicleTestData-TrackerCwDetHistBeforeRetro.csv  20171025\_FIT\_JARI\_NCAP\_FE/VehicleTestData-TrackerFmDetHistBeforeRetro.csv  20171025\_FIT\_JARI\_NCAP\_FE/VehicleTestData-TrackerDetClusterData.csv |
| Expected Output(s) | For each frame:   * Returned code is NB\_OK, indicating proper operation. * All of the following quantities match the previously computed results: * TrackClusterArray.TrackCandidCw * TrackClusterArray.TrackCandidFm |

|  |  |
| --- | --- |
| Test Case ID | Tracker\_RetroDetect\_Success2 |
| Purpose | Demonstrate successful operation of Tracker\_RetroDetect() function. |
| Requirement | [INT5] |
| Setup | NA |
| Test Preconditions | NA |
| Description | Provide valid input parameters to Tracker\_RetroDetect() function and call the function from 1 to 100 frames for NCAP-004\_Trial1\_REC\_19Apr2019\_08-54-41 data file. Compare the output of each frame with the previously computed results. |
| Input(s) | Data files:  NCAP-004\_Trial1\_REC\_19Apr2019\_08-54-41/VehicleTestData-TrackerCwDetHistBeforeRetro.csv  NCAP-004\_Trial1\_REC\_19Apr2019\_08-54-41/VehicleTestData-TrackerFmDetHistBeforeRetro.csv  NCAP-004\_Trial1\_REC\_19Apr2019\_08-54-41/VehicleTestData-TrackerDetClusterData.csv |
| Expected Output(s) | For each frame:   * Returned code is NB\_OK, indicating proper operation. * All of the following quantities match the previously computed results: * TrackClusterArray.TrackCandidCw * TrackClusterArray.TrackCandidFm |

|  |  |
| --- | --- |
| Test Case ID | Tracker\_RetroDetect\_Success3 |
| Purpose | Demonstrate successful operation of Tracker\_RetroDetect() function. |
| Requirement | [INT5] |
| Setup | NA |
| Test Preconditions | NA |
| Description | Provide valid input parameters to Tracker\_RetroDetect() function and call the function from 1 to 100 frames for REC\_12Feb2019\_17-05-51\_TIMED\_8673577000 data file. Compare the output of each frame with the previously computed results. |
| Input(s) | Data files:  REC\_12Feb2019\_17-05-51\_TIMED\_8673577000/VehicleTestData-TrackerCwDetHistBeforeRetro.csv  REC\_12Feb2019\_17-05-51\_TIMED\_8673577000/VehicleTestData-TrackerFmDetHistBeforeRetro.csv  REC\_12Feb2019\_17-05-51\_TIMED\_8673577000/VehicleTestData-TrackerDetClusterData.csv |
| Expected Output(s) | For each frame:   * Returned code is NB\_OK, indicating proper operation. * All of the following quantities match the previously computed results: * TrackClusterArray.TrackCandidCw * TrackClusterArray.TrackCandidFm |

|  |  |
| --- | --- |
| Test Case ID | Tracker\_RetroDetect\_Success4 |
| Purpose | Demonstrate successful operation of Tracker\_RetroDetect() function. |
| Requirement | [INT5] |
| Setup | NA |
| Test Preconditions | NA |
| Description | Provide valid input parameters to Tracker\_RetroDetect() function and call the function from 500 to 550 frames for REC\_11Dec2018\_13-42-21\_TIMED\_2819484000 data file. Compare the output of each frame with the previously computed results. |
| Input(s) | Data files:  REC\_11Dec2018\_13-42-21\_TIMED\_2819484000/VehicleTestData-TrackerCwDetHistBeforeRetro.csv  REC\_11Dec2018\_13-42-21\_TIMED\_2819484000/VehicleTestData-TrackerFmDetHistBeforeRetro.csv  REC\_11Dec2018\_13-42-21\_TIMED\_2819484000/VehicleTestData-TrackerDetClusterData.csv |
| Expected Output(s) | For each frame:   * Returned code is NB\_OK, indicating proper operation. * All of the following quantities match the previously computed results: * TrackClusterArray.TrackCandidCw * TrackClusterArray.TrackCandidFm |

|  |  |
| --- | --- |
| Test Case ID | Tracker\_RetroDetect\_Success4 |
| Purpose | Demonstrate successful operation of Tracker\_RetroDetect() function. |
| Requirement | [INT5] |
| Setup | NA |
| Test Preconditions | NA |
| Description | Provide valid input parameters to Tracker\_RetroDetect() function and call the function from 110 to 160 frames for REC\_18Jan2019\_14-29-34\_TIMED\_3483751000 data file. Compare the output of each frame with the previously computed results. |
| Input(s) | Data files:  REC\_18Jan2019\_14-29-34\_TIMED\_3483751000/VehicleTestData-TrackerCwDetHistBeforeRetro.csv  REC\_18Jan2019\_14-29-34\_TIMED\_3483751000/VehicleTestData-TrackerFmDetHistBeforeRetro.csv  REC\_18Jan2019\_14-29-34\_TIMED\_3483751000/VehicleTestData-TrackerDetClusterData.csv |
| Expected Output(s) | For each frame:   * Returned code is NB\_OK, indicating proper operation. * All of the following quantities match the previously computed results: * TrackClusterArray.TrackCandidCw * TrackClusterArray.TrackCandidFm |