NMPB08 – PROPAGATION

Reference Manual

Van Maercke Dirk
CSTB Grenoble
20/09/2012

This rest of this file was generated automatically with Doxygen 1.7.4

Table of Contents

NMPB08 Propagation Model (as published in NF S 31-133:2011)

Public functions defined in the PropagationNMPB08 library

Create and delete a calculation context

- NMPB08_CreatePath
- NMPB08_DeletePath
- NMPB08_CreatePathEx
- NMPB08_SetOption
- NMPB08_GetNbFrequencies
- NMPB08_GetFrequencies

Define the propagation path

- NMPB08_ClearPath
- NMPB08_ExtendPath
- NMPB08_ExtendPathExt
- NMPB08_SetSourceHeight
- NMPB08_SetReceiverHeight

Calculate propagation attenuation

- NMPB08_DoCalculation
- NMPB08_GetAttH
- NMPB08_GetAttF

Calculate long term averaged noise levels

- NMPB08_GetFavorableConditionProbability
- NMPB08_CalculateLeqLT
- NMPB08_SumLevels
- NMPB08_Leq_LT

Namespace Index

Namespace List

Here is a list of all namespaces with brief descriptions:

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Namespace Documentation

CalculPropagationNMPB Namespace Reference

Classes

- class Attenuation

Class used to sound attenuation calculations along a given path, with a specified frequency. Functions

- double SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- double SoundLevelForPath (double soundLevel_h, double soundLevel_f, double favourableProbability)
  Calculates long-term sound level for the given the sound levels (homogeneous and favorable) for a path.
- double GetFavorableConditionProbability (Position3D const *sourcePos, Position3D const *receiverPos, int nbAngles, double const *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- void CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)
  Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Function Documentation

void CalculPropagationNMPB::CalculateLeqLT (int *nbFreq*, double const **Lw*, double const **attH*, double const **attF*, double *fcp*, double **LeqH*, double **LeqF*, double **LeqLT*)

Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Parameters:

nbFreq	The frequencies number (user data)
Lw	The sound power levels of the source, in each frequency band (user data)
attH	Attenuations due to the propagation between source and receiver in homogeneous conditions, in each frequency band (user data)
attF	Attenuations due to the propagation between source and receiver in downward-refraction conditions, in each frequency band (user data)
fcp	Probability of occurrence of downward-refraction conditions over a long-term period in a given direction, p in [0, 1] (user data)
LeqH	Sound levels due to source Si at point R in homogeneous conditions, in each frequency band (calculated in this function)
LeqF	Sound levels due to source Si at point R in downward-refraction conditions, in each frequency band (calculated in this function)
LeqLT	Long-term sound levels due to source Si at point R, in each given frequency band (calculated in this function)

Definition at line 170 of file CalculPropagation.cpp.

double CalculPropagationNMPB::GetFavorableConditionProbability (Position3D const *sourcePos, Position3D const *receiverPos, int nbAngles, double const *fcpAngles, double angleNorth)

Calculates the favorable conditions probability for the (SR) direction.

Parameters:

sourcePos	The source position 3D
receiverPos	The receiver position 3D
nbAngles	The number of angle probabilities (should be 18)
fcpAngles	The favorable conditions probabilities for the angles 20, 40, 60, ..., 360
angleNorth	The north direction (Ox,ON)

Returns:

the favorable conditions probability for the (SR) direction

Exceptions:

ERRAngle	

Definition at line 103 of file CalculPropagation.cpp.

double CalculPropagationNMPB::SoundLevelForPath (double soundLevel_h, double soundLevel_f, double favourableProbability)

Calculates long-term sound level for the given the sound levels (homogeneous and favorable) for a path.

p.18 - § 5.2.3 Formula (7)

Parameters:

soundLevel_h	The sound level in homogeneous conditions
soundLevel_f	The sound level in downward-refraction conditions
favourableProbability	The average occurrence of downward-refraction conditions in the direction of the path

Returns:

The calculated sound level

Exceptions:

ERRProbability	

Definition at line 75 of file CalculPropagation.cpp.

double CalculPropagationNMPB::SumLevels (int *n*, double const **levels*)

Calculates sound levels sum.

Used for § 5.2.4 Formula (8) and § 5.2.5 Formula (9)

Parameters:

n	the sound levels number
levels	The sound levels to sum

Returns:

the sum

Definition at line 47 of file CalculPropagation.cpp.

DiffractionNMPB Namespace Reference

Classes

- class Diffraction

Class used to calculate diffraction attenuation. Functions

- double CurveRayLength (double distMN, double curvatureRadius)
  Calculates the circular ray length for MN.
- double PathDifference (Position2D const *source2D, Position2D const *receiver2D, vector< ProfilePointNMPB * > screenItems, bool favourableConditions)
  Calculation of the path difference for the screen elements.
- double PathDifference (Position2D const *source2D, Position2D const *receiver2D, ProfilePointNMPB *reflectionItem)
  Calculation of the path difference for a reflection element.
- double SidePathDifference (Position3D const *source3D, Position3D const *receiver3D, vector< ProfilePointNMPB * > screenItems, double &totalDiffDist)
  Calculation of the path difference for the side diffractions.

Function Documentation

double DiffractionNMPB::CurveRayLength (double *distMN*, double *curvatureRadius*)

Calculates the circular ray length for MN.

Parameters:

distMN	The MN distance

curvatureRadius	The radius of curvature

Returns:

the circular ray length

Definition at line 27 of file Diffraction.cpp.

double DiffractionNMPB::PathDifference (Position2D const *source2D, Position2D const *receiver2D, vector< ProfilePointNMPB * >screenItems, boolfavourableConditions)

Calculation of the path difference for the screen elements.

p.47-49 - § 9.4.3

Parameters:

source2D	: source coordinates
receiver2D	: receiver coordinates
screenItems	: vector containing the terrain elements (terrain elements with diffracted indicated on the path, ie convex hull)
favourableConditions	: true if calculation must be in favorable conditions, false for homogeneous conditions

Returns:

the path difference

Definition at line 53 of file Diffraction.cpp.

double DiffractionNMPB::PathDifference (Position2D const *source2D, Position2D const *receiver2D, ProfilePointNMPB *reflectionItem)

Calculation of the path difference for a reflection element.

p.53 - § 9.5.2 formula (44)

Parameters:

source2D	: source coordinates
receiver2D	: receiver coordinates
reflectionItem	The reflection item

Returns:

the path difference

Definition at line 206 of file Diffraction.cpp.

double DiffractionNMPB::SidePathDifference (Position3D const *source3D, Position3D const *receiver3D, vector< ProfilePointNMPB * >screenItems, double &totalDiffDist)

Calculation of the path difference for the side diffractions.

p.35-36 - § 8.2.3 and p.47-49 - § 9.4.3 and p.51 - § 9.4.5

Parameters:

source3D	: source coordinates
receiver3D	: receiver coordinates
screenItems	: vector containing the terrain elements
totalDiffDist	: distance between the first and the last diffraction (calculated in this function)

Returns:

the path difference

Definition at line 239 of file Diffraction.cpp.

ElementaryPathNMPB Namespace Reference

Functions

- void ConvexHull (vector< ProfilePointNMPB * > &pathItems, int n1, int n2, int level)
  Finds the convex hull of the screenItems vector.
- void SetElementaryPath (PropagationPath *path)
  Sets the elementary path for the given path (calculates convex hull and plane positions)

Function Documentation

void ElementaryPathNMPB::ConvexHull (vector< ProfilePointNMPB * > &*pathItems*, int*n1*, int*n2*, int*level*)

Finds the convex hull of the screenItems vector.

p.47-49 - § 9.4.3

Parameters:

pathItems	The points to find the convex hull
n1	The first element place in the vector
n2	The last element place in the vector
level	The convex hull level

Definition at line 26 of file ElementaryPath.cpp.

void ElementaryPathNMPB::SetElementaryPath (PropagationPath *path)

Sets the elementary path for the given path (calculates convex hull and plane positions)

Parameters:

path	The Propagation path

Exceptions:

ERRNoPoint	
ERROnePoint	

Definition at line 106 of file ElementaryPath.cpp.

EmbankmentNMPB Namespace Reference

Classes

- class Embankment

Class used to calculate the embankment attenuation.

GroundEffectNMPB Namespace Reference

Classes

- class MeanPlane
- *Class used to calculate the mean plane.* class GroundEffect

Class used to calculate ground effects. Enumerations

- enum GroundCalculationType { Asol = 1, DeltaSol_SO = 2, DeltaSol_OR = 3 }

Ground Calculation Type used to know when using _Gpath or _correctedGpath in the ground attenuation calculation.

Enumeration Type Documentation

enum GroundEffectNMPB::GroundCalculationType

Ground Calculation Type used to know when using _Gpath or _correctedGpath in the ground attenuation calculation.

Enumerator:

Asol
DeltaSol_SO
DeltaSol_OR

Definition at line 24 of file GroundEffect.h.

Class Documentation

CalculPropagationNMPB::Attenuation Class Reference

Class used to sound attenuation calculations along a given path, with a specified frequency.

`#include <CalculPropagation.h>`

Public Member Functions

- Attenuation (PropagationPath *path, double frequency)
  Initialization of Attenuation class with the path and the frequency.
- Attenuation (PropagationPath *path, double frequency, double favourableProbability)
  Initialization of Attenuation class with the path, the frequency and the favorable probability.
- double DivergenceAttenuationCalculation (double dist)

Attenuation calculation following geometric divergence.

- double AtmosphericAbsorptionCalculation (double dist, double freq)
  Atmospheric absorption.
- void BoundaryAttenuationCalculation (int nbSideDiffractions)
  Calculates border attenuation, in homogeneous and favorable conditions.
- void AttenuationCalculation ()
  Calculates total attenuation along the propagation path, in homogeneous and favorable conditions.
- void FillFrequencyMap (map< int, double > newMap)
  Fills the _attenuationCoeffByFrequencyMap map with values of another map.
- double getHomogeneousAttenuation ()
  Gets attenuation along the path in homogeneous conditions, for the given frequency.
- double getFavorableAttenuation ()
  Gets attenuation along the path in favorable conditions, for the given frequency.
- double getDivergenceAttenuation ()
  Gets divergence attenuation.
- double getAtmosphericAbsorption ()
  Gets atmospheric absorption.
- double getGroundAttenuation_h ()
  Gets ground attenuation in homogeneous conditions.
- double getGroundAttenuation_f ()
  Gets ground attenuation in favorable conditions.
- double getDiffractionAttenuation_h ()
  Gets diffraction attenuation in homogeneous conditions.
- double getDiffractionAttenuation_f ()
  Gets diffraction attenuation in favorable conditions.
- double getEmbankmentAttenuation ()
  Gets embankment attenuation.
- double getBoundaryAttenuation_h ()
  Gets border attenuation in homogeneous conditions.
- double getBoundaryAttenuation_f ()
  Gets border attenuation in favorable conditions.

Detailed Description

Class used to sound attenuation calculations along a given path, with a specified frequency.

Definition at line 102 of file CalculPropagation.h.

Constructor & Destructor Documentation

CalculPropagationNMPB::Attenuation::Attenuation (PropagationPath *path, double frequency)

Initialization of Attenuation class with the path and the frequency.

Parameters:

path	The path where the attenuation will be calculated
frequency	The frequency to calculation sound attenuation

Definition at line 194 of file CalculPropagation.cpp.

CalculPropagationNMPB::Attenuation::Attenuation (PropagationPath *path, double*frequency*, double*favourableProbability*)

Initialization of Attenuation class with the path, the frequency and the favorable probability.

Parameters:

path	The path where the attenuation will be calculated
frequency	The frequency to calculation sound attenuation
favourableProbability	The favorable probability

Definition at line 216 of file CalculPropagation.cpp.

Member Function Documentation

double CalculPropagationNMPB::Attenuation::AtmosphericAbsorptionCalculation (double*dist*, double*freq*)

Atmospheric absorption.

Method to calculate atmospheric absorption attenuation § 9.2, formula (21), p. 37

Parameters:

dist	: direct distance between source and receptor
freq	: third octave bands center frequency

Returns:

the calculated absorption attenuation.

Method to calculate atmospheric absorption attenuation § 9.2, formula (21), p. 37

Parameters:

dist	: direct distance between source and receptor
freq	: octave bands center frequency

Returns:

the calculated absorption attenuation.

Definition at line 325 of file CalculPropagation.cpp.

void CalculPropagationNMPB::Attenuation::AttenuationCalculation ()

Calculates total attenuation along the propagation path, in homogeneous and favorable conditions.

Method to calculate total attenuation along the propagation path, in homogeneous and favorable conditions § 5.2.1, formula (4), p.17 and § 5.2.2, formula (6), p.17

Exceptions:

ERRSideDiff	

Definition at line 540 of file CalculPropagation.cpp.

void CalculPropagationNMPB::Attenuation::BoundaryAttenuationCalculation (int*nbSideDiffractions*)

Calculates border attenuation, in homogeneous and favorable conditions.

Method to calculate border attenuation, in homogeneous and favorable conditions § 5.2.1 and 5.2.2, p.17-18

Parameters:

nbSideDiffractions	The number of side diffractions in the path

Exceptions:

ERRNoPoint,ERROnePoint,ERRAttCoeffFrequency	

Definition at line 404 of file CalculPropagation.cpp.

double CalculPropagationNMPB::Attenuation::DivergenceAttenuationCalculation (double*dist*)

Attenuation calculation following geometric divergence.

Method for attenuation calculation following geometric divergence § 9.1, formula (20), p. 37

Parameters:

dist	: Direct distance between source and receiver.

Returns:

the calculated attenuation.

Definition at line 240 of file CalculPropagation.cpp.

void CalculPropagationNMPB::Attenuation::FillFrequencyMap (map< int, double >*newMap*)

Fills the _attenuationCoeffByFrequencyMap map with values of another map.

Parameters:

newMap	The map to get values

Definition at line 272 of file CalculPropagation.cpp.

double CalculPropagationNMPB::Attenuation::getAtmosphericAbsorption () `[inline]`

Gets atmospheric absorption.

Returns:

atmospheric absorption

Definition at line 217 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getBoundaryAttenuation_f () `[inline]`

Gets border attenuation in favorable conditions.

Returns:
border attenuation in favorable conditions
Definition at line 287 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getBoundaryAttenuation_h () [inline]

Gets border attenuation in homogeneous conditions.

Returns:
border attenuation in homogeneous conditions
Definition at line 277 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getDiffractionAttenuation_f () [inline]

Gets diffraction attenuation in favorable conditions.

Returns:
diffraction attenuation in favorable conditions
Definition at line 257 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getDiffractionAttenuation_h () [inline]

Gets diffraction attenuation in homogeneous conditions.

Returns:
diffraction attenuation in homogeneous conditions
Definition at line 247 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getDivergenceAttenuation () [inline]

Gets divergence attenuation.

Returns:
divergence attenuation
Definition at line 207 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getEmbankmentAttenuation () [inline]

Gets embankment attenuation.

Returns:
embankment attenuation
Definition at line 267 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getFavorableAttenuation () [inline]

Gets attenuation along the path in favorable conditions, for the given frequency.

Returns:
attenuation along the path in favorable conditions, for the given frequency
Definition at line 197 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getGroundAttenuation_f () [inline]

Gets ground attenuation in favorable conditions.

Returns:
ground attenuation in favorable conditions
Definition at line 237 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getGroundAttenuation_h () [inline]

Gets ground attenuation in homogeneous conditions.

Returns:
ground attenuation in homogeneous conditions
Definition at line 227 of file CalculPropagation.h.

double CalculPropagationNMPB::Attenuation::getHomogeneousAttenuation () [inline]

Gets attenuation along the path in homogeneous conditions, for the given frequency.

Returns:
attenuation along the path in homogeneous conditions, for the given frequency
Definition at line 187 of file CalculPropagation.h.

The documentation for this class was generated from the following files:

- CalculPropagation.h
- CalculPropagation.cpp

DiffractionNMPB::Diffraction Class Reference

Class used to calculate diffraction attenuation.

`#include <Diffraction.h>`

Public Member Functions

- Diffraction ()
  Initialization of the Diffraction Class.
- Diffraction (vector< ProfilePointNMPB * > terrainItems, double freq, int freqPos, bool withEmbankment)
  Initialization of the Diffraction Class.
- Diffraction (vector< ProfilePointNMPB * > terrainItems, double freq, int freqPos, int nbSideDiffractions)
  Initialization of the Diffraction Class in side diffraction case.
- void ReflectionAttenuation (vector< ProfilePointNMPB * > &terrainItems, int freqPos)
  Calculates the reflection attenuation.
- void CalculAttenuationDiffraction (bool withCh=true)
  Calculates attenuation due to diffraction.
- double CalculDiffractionPure (Position2D const *source2D, Position2D const *receiver2D, vector< ProfilePointNMPB * > terrainItems, bool favourableConditions, bool totalPath, bool withCh)
  Calculation of the pure diffraction.
- double CalculDiffractionPure (Position2D const *source2D, Position2D const *receiver2D, ProfilePointNMPB *reflectionItem)
  Calculation of the retro diffraction in reflection case.
- double Get_pathDifferenceSR_h ()
  Gets path difference for the SR path, in homogeneous conditions (used to know if diffraction must be calculated)
- double Get_pathDifferenceSR_f ()
  Gets path difference for the SR path, in favorable conditions (used to know if diffraction must be calculated)
- double Get_diffractionAttenuation_h ()
  Gets diffraction attenuation (Adif), in homogeneous conditions.
- double Get_diffractionAttenuation_f ()
  Gets diffraction attenuation (Adif), in favorable conditions.
- double Get_DeltaDifSR_h ()
  Gets attenuation due to pure diffraction, in homogeneous conditions.
- double Get_DeltaDifSR_f ()
  Gets attenuation due to pure diffraction, in favorable conditions.
- double Get_DeltaSolSO_h ()
  Gets attenuation from the ground effect source side, weighted by the diffraction source side, in homogeneous conditions.
- double Get_DeltaSolSO_f ()
  Gets attenuation from the ground effect source side, weighted by the diffraction source side, in favorable conditions.
- double Get_DeltaSolOR_h ()
  Gets attenuation from the ground effect receiver side, weighted by the diffraction receiver side, in homogeneous conditions.
- double Get_DeltaSolOR_f ()
  Gets attenuation from the ground effect receiver side, weighted by the diffraction receiver side, in favorable conditions.
- double Get_DeltaTalusSO_h ()

Gets attenuation due to embankment in homogeneous conditions.

- double Get_DeltaTalusSO_f ()
  Gets attenuation due to embankment in favorable conditions.
- double Get_AttsolSO_h ()
  Gets attenuation due to the ground effect for the SO path in homogeneous conditions.
- double Get_AttsolSO_f ()
  Gets attenuation due to the ground effect for the SO path in favorable conditions.
- double Get_absorptionAttenuation ()
  Gets attenuation due reflection absorption.

Protected Member Functions

- double SoundAbsorption (ProfilePointNMPB *pointNMBP, int posRef, int freqPos)
  Calculates the sound absorption due to reflection.
- void MeanPlanesDataCalculation (bool withEmbankment)
  Calculates data of the 2 mean planes (SO) and (OR), and embankment.
- double CalculDeltaSol (double aSol, double deltaDif1, double deltaDif2)
  Calculation of the ground attenuation with diffraction, source side or receiver side.
- void ClearData ()
  Clear data.

Protected Attributes

- ProfilePointNMPB * _source
  Source.
- ProfilePointNMPB * _receiver
  Receiver.
- vector< ProfilePointNMPB * > _terrainItems
  Terrain items.
- double _freq
  Median frequency fm.
- ProfilePointNMPB * _firstScreen
  the first used screen after source
- ProfilePointNMPB * _lastScreen
  the last used screen before receiver
- MeanPlane _meanPlaneSO
  the mean plane between source and first screen
- MeanPlane _meanPlaneOR
  the mean plane between last screen and receiver
- double _wavelength
  wavelength (lambda)
- double _totalDiffDist
  total distance between diffraction closest to the source and diffraction closest to the receptor (e)
- double _h0
  higher diffraction edge for the 2 mean planes (source side and receiver side)
- double _pathDifferenceSR_h
  path differences for the SR path (in homogeneous conditions)
- double _pathDifferenceSR_f

path differences for the SR path (in favorable conditions)

- double _Adif_h
  diffraction attenuation (Adif), in homogeneous conditions
- double _Adif_f
  diffraction attenuation (Adif), in favorable conditions
- double _DeltaDifSR_h
  attenuation due to pure diffraction between source S et receiver R, in homogeneous conditions
- double _DeltaDifSR_f
  attenuation due to pure diffraction between source S et receiver R, in favorable conditions
- double _DeltaSolSO_h
  attenuation due to source ground effect, in homogeneous conditions
- double _DeltaSolSO_f
  attenuation due to source ground effect, in favorable conditions
- double _AttTalusSO
  attenuation due to embankment in source side
- Position2D _embankmentSourceImage
  the embankment source image position
- double _DeltaTalusSO_h
  attenuation due to embankment in source side, in homogeneous conditions
- double _DeltaTalusSO_f
  attenuation due to embankment in source side, in favorable conditions
- double _DeltaSolOR_h
  attenuation due to receiver ground effect, in homogeneous conditions
- double _DeltaSolOR_f
  attenuation due to receiver ground effect, in favorable conditions
- double _aSolSO_h
  Attenuation due to the ground effect for the SO path in homogeneous conditions.
- double _aSolSO_f
  Attenuation due to the ground effect for the SO path in favorable conditions.
- double _absorptionAttenuation
  absorption attenuation due to vertical screens

Detailed Description

Class used to calculate diffraction attenuation.

Class used to calculate diffraction attenuation with intermediates values and functions ; Contains ground attenuation ; uses PathDifference p. 44-51 - § 9.4

Definition at line 31 of file Diffraction.h.

Constructor & Destructor Documentation

DiffractionNMPB::Diffraction::Diffraction () [inline]

Initialization of the Diffraction Class.

Definition at line 38 of file Diffraction.h.

DiffractionNMPB::Diffraction::Diffraction (vector< ProfilePointNMPB * >*terrainItems*, double*freq*, int*freqPos*, bool*withEmbankment*)

Initialization of the Diffraction Class.

Parameters:

terrainItems	The terrain items
freq	Center frequency
freqPos	The frequency position in the spectrum array
withEmbankment	True if embankment must be checked

Initialization of the Diffraction Class with source, receiver, screens and frequency (containing mean planes calculation, SR path difference calculation and reflection attenuation)

Exceptions:

ERRFrequency	

section 9.4 p.44-50

Parameters:

terrainItems	The terrain items
freq	Center frequency
freqPos	The frequency position in the spectrum array
withEmbankment	True if embankment must be checked

Initialization of the Diffraction Class with source, receiver, screens and frequency (containing mean planes calculation, SR path difference calculation and reflection attenuation)

Exceptions:

ERRFrequency	

Definition at line 369 of file Diffraction.cpp.

DiffractionNMPB::Diffraction::Diffraction (vector< ProfilePointNMPB * >*terrainItems*, double*freq*, int*freqPos*, int*nbSideDiffractions*)

Initialization of the Diffraction Class in side diffraction case.

Parameters:

terrainItems	the terrain items
freq	center frequency
freqPos	The frequency position in the spectrum array
nbSideDiffractions	the side diffractions number

Initialization of the Diffraction Class in side diffraction case with source, receiver, screens and frequency (containing SR path difference calculation, reflection attenuation and _DeltaDifSR_h calculation)

Exceptions:

ERRFrequency	section 9.4.5 p.51

Parameters:

terrainItems	the terrain items

freq	center frequency
freqPos	The frequency position in the spectrum array
nbSideDiffractions	the side diffractions number

Initialization of the Diffraction Class in side diffraction case with source, receiver, screens and frequency (containing SR path difference calculation, reflection attenuation and _DeltaDifSR_h calculation)

Exceptions:

ERRFrequency	

Definition at line 425 of file Diffraction.cpp.

Member Function Documentation

void DiffractionNMPB::Diffraction::CalculAttenuationDiffraction (bool*withCh* = `true`)

Calculates attenuation due to diffraction.

Calculates attenuation due to diffraction, with ground effects and embankment p.49 - § 9.4.4 formula (37)

Parameters:

withCh	if true, enables the calculation Ch, the correction term for low height obstacles, if false, Ch is taken equal to 1

Calculates attenuation due to diffraction, with ground effects and embankment p.49 - § 9.4.4 formula (37)

Definition at line 796 of file Diffraction.cpp.

double DiffractionNMPB::Diffraction::CalculDeltaSol (double*aSol*, double*deltaDif1*, double*deltaDif2*) `[protected]`

Calculation of the ground attenuation with diffraction, source side or receiver side.

Parameters:

aSol	: ground attenuation (source side or receiver side)
deltaDif1	: diffraction attenuation during the image path (S'R or SR')
deltaDif2	: diffraction attenuation during the real path (SR)

Returns:

Definition at line 769 of file Diffraction.cpp.

double DiffractionNMPB::Diffraction::CalculDiffractionPure (Position2D const **source2D*, Position2D const **receiver2D*, ProfilePointNMPB **reflectionItem*)

Calculation of the retro diffraction in reflection case.

p.53 - § 9.5.2 Formula (45)

Parameters:

source2D	The source coordinates

receiver2D	The receiver coordinates
reflectionItem	The vertical reflection screen

Returns:

the calculated retro diffraction

Definition at line 738 of file Diffraction.cpp.

double DiffractionNMPB::Diffraction::CalculDiffractionPure (Position2D const *source2D, Position2D const *receiver2D, vector< ProfilePointNMPB * >screenItems, boolfavourableConditions, booltotalPath, boolwithCh)

Calculation of the pure diffraction.

p.46 - § 9.4.2 Formula (31)

Parameters:

source2D	The source coordinates
receiver2D	The receiver coordinates
screenItems	The screen items
favourableConditions	: true if conditions are favorable (false if homogeneous)
totalPath	: true if the calculation is on the total path
withCh	: true if Ch must be calculated (false if Ch = 1)

Returns:

the calculated pure diffraction

Definition at line 666 of file Diffraction.cpp.

void DiffractionNMPB::Diffraction::ClearData () [inline, protected]

Clear data.

Definition at line 451 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_absorptionAttenuation () [inline]

Gets attenuation due reflection absorption.

Returns:

attenuation due to reflection absorption

Definition at line 288 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_AttsolSO_f () [inline]

Gets attenuation due to the ground effect for the SO path in favorable conditions.

Returns:

attenuation due to the ground effect for the SO path in favorable conditions

Definition at line 278 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_AttsolSO_h () [inline]

Gets attenuation due to the ground effect for the SO path in homogeneous conditions.

Returns:
attenuation due to the ground effect for the SO path in homogeneous conditions

Definition at line 268 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaDifSR_f () [inline]

Gets attenuation due to pure diffraction, in favorable conditions.

Returns:
attenuation due to pure diffraction, in favorable conditions

Definition at line 198 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaDifSR_h () [inline]

Gets attenuation due to pure diffraction, in homogeneous conditions.

Returns:
attenuation due to pure diffraction, in homogeneous conditions

Definition at line 188 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaSolOR_f () [inline]

Gets attenuation from the ground effect receiver side, weighted by the diffraction receiver side, in favorable conditions.

Returns:
attenuation from the ground effect receiver side, weighted by the diffraction receiver side, in favorable conditions

Definition at line 238 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaSolOR_h () [inline]

Gets attenuation from the ground effect receiver side, weighted by the diffraction receiver side, in homogeneous conditions.

Returns:
attenuation from the ground effect receiver side, weighted by the diffraction receiver side, in homogeneous conditions

Definition at line 228 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaSolSO_f () [inline]

Gets attenuation from the ground effect source side, weighted by the diffraction source side, in favorable conditions.

Returns:

attenuation from the ground effect source side, weighted by the diffraction source side, in favorable conditions

Definition at line 218 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaSolSO_h () [inline]

Gets attenuation from the ground effect source side, weighted by the diffraction source side, in homogeneous conditions.

Returns:

attenuation from the ground effect source side, weighted by the diffraction source side, in homogeneous conditions

Definition at line 208 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaTalusSO_f () [inline]

Gets attenuation due to embankment in favorable conditions.

Returns:

attenuation due to embankment for the SO path in favorable conditions

Definition at line 258 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_DeltaTalusSO_h () [inline]

Gets attenuation due to embankment in homogeneous conditions.

Returns:

attenuation due to embankment for the SO path in homogeneous conditions

Definition at line 248 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_diffractionAttenuation_f () [inline]

Gets diffraction attenuation (Adif), in favorable conditions.

Returns:

diffraction attenuation (Adif), in favorable conditions

Definition at line 178 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_diffractionAttenuation_h () [inline]

Gets diffraction attenuation (Adif), in homogeneous conditions.

Returns:
diffraction attenuation (Adif), in homogeneous conditions
Definition at line 168 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_pathDifferenceSR_f () [inline]

Gets path difference for the SR path, in favorable conditions (used to know if diffraction must be calculated)

Returns:
the path difference for the SR path in favorable conditions
Definition at line 158 of file Diffraction.h.

double DiffractionNMPB::Diffraction::Get_pathDifferenceSR_h () [inline]

Gets path difference for the SR path, in homogeneous conditions (used to know if diffraction must be calculated)

Returns:
the path difference for the SR path in homogeneous conditions
Definition at line 148 of file Diffraction.h.

void DiffractionNMPB::Diffraction::MeanPlanesDataCalculation (bool*withEmbankment*) [protected]

Calculates data of the 2 mean planes (SO) and (OR), and embankment.

Parameters:

withEmbankment	True if embankment calculation must be done

Definition at line 569 of file Diffraction.cpp.

void DiffractionNMPB::Diffraction::ReflectionAttenuation (vector< ProfilePointNMPB * > &*terrainItems*, int*freqPos*)

Calculates the reflection attenuation.

Parameters:

terrainItems	The terrain items
freqPos	The frequency position in the spectrum array

Definition at line 474 of file Diffraction.cpp.

double DiffractionNMPB::Diffraction::SoundAbsorption (ProfilePointNMPB **pointNMPB*, int*posRef*, int*freqPos*) [protected]

Calculates the sound absorption due to reflection.

Parameters:

pointNMPB	The reflection screen
posRef	The reflection position in the elements vector
freqPos	The frequency position in the spectrum array

Returns:

the calculated absorption

Exceptions:

ERRScreenAbsorption	

Definition at line 502 of file Diffraction.cpp.

Member Data Documentation

double DiffractionNMPB::Diffraction::_absorptionAttenuation [protected]

absorption attenuation due to vertical screens

Definition at line 410 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_Adif_f [protected]

diffraction attenuation (Adif), in favorable conditions

Definition at line 357 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_Adif_h [protected]

diffraction attenuation (Adif), in homogeneous conditions

Definition at line 353 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_aSolSO_f [protected]

Attenuation due to the ground effect for the SO path in favorable conditions.

Definition at line 405 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_aSolSO_h [protected]

Attenuation due to the ground effect for the SO path in homogeneous conditions.

Definition at line 401 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_AttTalusSO [protected]

attenuation due to embankment in source side
Definition at line 377 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaDifSR_f [protected]

attenuation due to pure diffraction between source S et receiver R, in favorable conditions
Definition at line 365 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaDifSR_h [protected]

attenuation due to pure diffraction between source S et receiver R, in homogeneous conditions
Definition at line 361 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaSolOR_f [protected]

attenuation due to receiver ground effect, in favorable conditions
Definition at line 397 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaSolOR_h [protected]

attenuation due to receiver ground effect, in homogeneous conditions
Definition at line 393 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaSolSO_f [protected]

attenuation due to source ground effect, in favorable conditions
Definition at line 373 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaSolSO_h [protected]

attenuation due to source ground effect, in homogeneous conditions
Definition at line 369 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaTalusSO_f [protected]

attenuation due to embankment in source side, in favorable conditions
Definition at line 389 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_DeltaTalusSO_h [protected]

attenuation due to embankment in source side, in homogeneous conditions

Definition at line 385 of file Diffraction.h.

Position2D DiffractionNMPB::Diffraction::_embankmentSourceImage [protected]

the embankment source image position
Definition at line 381 of file Diffraction.h.

ProfilePointNMPB* DiffractionNMPB::Diffraction::_firstScreen [protected]

the first used screen after source
Definition at line 314 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_freq [protected]

Median frequency fm.
Definition at line 309 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_h0 [protected]

higher diffraction edge for the 2 mean planes (source side and receiver side)
Definition at line 340 of file Diffraction.h.

ProfilePointNMPB* DiffractionNMPB::Diffraction::_lastScreen [protected]

the last used screen before receiver
Definition at line 318 of file Diffraction.h.

MeanPlane DiffractionNMPB::Diffraction::_meanPlaneOR [protected]

the mean plane between last screen and receiver
Definition at line 327 of file Diffraction.h.

MeanPlane DiffractionNMPB::Diffraction::_meanPlaneSO [protected]

the mean plane between source and first screen
Definition at line 323 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_pathDifferenceSR_f [protected]

path differences for the SR path (in favorable conditions)
Definition at line 349 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_pathDifferenceSR_h [protected]

path differences for the SR path (in homogeneous conditions)
Definition at line 345 of file Diffraction.h.

ProfilePointNMPB* DiffractionNMPB::Diffraction::_receiver [protected]

Receiver.
Definition at line 301 of file Diffraction.h.

ProfilePointNMPB* DiffractionNMPB::Diffraction::_source [protected]

Source.
Definition at line 297 of file Diffraction.h.

vector<ProfilePointNMPB*> DiffractionNMPB::Diffraction::_terrainItems [protected]

Terrain items.
Definition at line 305 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_totalDiffDist [protected]

total distance between diffraction closest to the source and diffraction closest to the receptor (e)
Definition at line 336 of file Diffraction.h.

double DiffractionNMPB::Diffraction::_wavelength [protected]

wavelength (lambda)
Definition at line 332 of file Diffraction.h.

The documentation for this class was generated from the following files:

- SousCalculs/Diffraction.h
- SousCalculs/Diffraction.cpp

EmbankmentNMPB::Embankment Class Reference

Class used to calculate the embankment attenuation.
`#include <Embankment.h>`

Public Member Functions

- Embankment (vector< ProfilePointNMPB * > terrainItems, double freq)
  Initialization of the Embankment class and embankment attenuation calculation.
- ~Embankment (void)
  Embankment Destructor.
- double GetEmbankmentAttenuation ()
  Gets the calculated embankment attenuation.
- Position2D * GetSourceImage ()
  Gets the calculated source image 2D Position.

Protected Member Functions

- void FillPosInPerpendicularPlane (Position2D *posToFill, Position2D const *posToRead, double cosTheta)
  Calculates the 2D position in the perpendicular plane to road.
- bool DataInPerpendicularPlane (vector< ProfilePointNMPB * > terrainItems)
  Calculates the 2D position of needed items in the perpendicular plane to road and check data.
- bool CheckData (vector< ProfilePointNMPB * > terrainItems)
  Checks the data.
- bool LineCoeff (Position2D const *pos1, Position2D const *pos2, double &aCoef, double &bCoef)
  *Calculates line coefficients : z = a*d + b (or d = b if line vertical) Warning The 2 positions must be different.*
- void SourceImage (double aCoef, double bCoef, bool verticalSlope)
  *Calculates the source image by the line z = a * d + b (or d = b if vertical)*
- bool LinesIntersection (double a1, double b1, bool vert1, double a2, double b2, bool vert2, Position2D *Ipoint)
  Calculates the lines intersection.
- double ThetaAngle ()
  Calculates the Theta angle in the formula (28) p.43.
- double HalfLength (double cosAngleTheta, double freq, double SimageI, double IR)
  Calculates half-length e.
- void EmbankmentCalculation (Position2D const *intersect, double freq)
  Calculates embankment attenuation (in _embankmentAttenuation)

Protected Attributes

- Position2D * _source
  the source position (S)
- Position2D * _receiver
  the receiver position (R)
- Position2D * _platformEnd
  the platform end
- Position2D * _O1
  the foot of the bank position
- Position2D * _O2
  the top of the bank position
- Position2D * _sourceImage
  the source image position (S")
- double _angleAlpha
  the (O1x,O1O2) angle

- double _Gembankment
  the slope impedance
- double _embankmentAttenuation
  the embankment attenuation

Detailed Description

Class used to calculate the embankment attenuation.

p. 43-44 - § 9.3.5

Definition at line 25 of file Embankment.h.

Constructor & Destructor Documentation

EmbankmentNMPB::Embankment::Embankment (vector< ProfilePointNMPB * >*terrainItems*, double*freq*)

Initialization of the Embankment class and embankment attenuation calculation.

Parameters:

terrainItems	the terrain elements
freq	the frequency

Exceptions:

ERRFrequency	

Definition at line 23 of file Embankment.cpp.

EmbankmentNMPB::Embankment::~Embankment (void) `[inline]`

Embankment Destructor.

Definition at line 43 of file Embankment.h.

Member Function Documentation

bool EmbankmentNMPB::Embankment::CheckData (vector< ProfilePointNMPB * >*terrainItems*) `[protected]`

Checks the data.

Checking according § 9.3.5 p. 43

Parameters:

terrainItems	The terrain items

Returns:

true if embankment can be calculated

Definition at line 256 of file Embankment.cpp.

bool EmbankmentNMPB::Embankment::DataInPerpendicularPlane (vector< ProfilePointNMPB * >*terrainItems*) [protected]

Calculates the 2D position of needed items in the perpendicular plane to road and check data.

Parameters:

terrainItems	The terrain items

Returns:

True if the data are OK

Exceptions:

ERREmbankment	

Definition at line 165 of file Embankment.cpp.

void EmbankmentNMPB::Embankment::EmbankmentCalculation (Position2D const **intersect*, double*freq*) [protected]

Calculates embankment attenuation (in _embankmentAttenuation)

Parameters:

intersect	The intersection point (I)
freq	The frequency

Definition at line 81 of file Embankment.cpp.

void EmbankmentNMPB::Embankment::FillPosInPerpendicularPlane (Position2D **posToFill*, Position2D const **posToRead*, double*cosTheta*) [protected]

Calculates the 2D position in the perpendicular plane to road.

Parameters:

posToFill	The position the fill
posToRead	The position to read
cosTheta	cos(Theta)

Definition at line 144 of file Embankment.cpp.

double EmbankmentNMPB::Embankment::GetEmbankmentAttenuation () [inline]

Gets the calculated embankment attenuation.

Returns:

the embankment attenuation

Definition at line 58 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::GetSourceImage () [inline]

Gets the calculated source image 2D Position.

Returns:

the source image 2D Position

Definition at line 68 of file Embankment.h.

double EmbankmentNMPB::Embankment::HalfLength (double *cosAngleTheta*, double *freq*, double *SimageI*, double *IR*) [protected]

Calculates half-length e.

p. 43 - § 9.3.5 formula (28)

Parameters:

cosAngleTheta	The theta cosines (must be != 0)
freq	The frequency (must be > 0)
SimageI	The distance between Simage and I
IR	The distance between Receiver and I

Returns:

the half-length e

Definition at line 456 of file Embankment.cpp.

bool EmbankmentNMPB::Embankment::LineCoeff (Position2D const **pos1*, Position2D const **pos2*, double *&aCoef*, double *&bCoef*) [protected]

Calculates line coefficients : z = a*d + b (or d = b if line vertical) Warning The 2 positions must be different.

Parameters:

pos1	The first line point
pos2	The second line pont
aCoef	The a coefficient to fill
bCoef	The b coefficient to fill

Returns:

true if the line is vertical

Definition at line 316 of file Embankment.cpp.

bool EmbankmentNMPB::Embankment::LinesIntersection (double *a1*, double *b1*, bool *vert1*, double *a2*, double *b2*, bool *vert2*, Position2D **Ipoint*) [protected]

Calculates the lines intersection.

Parameters:

a1	Parameter a1 of the first line
b1	Parameter b1 of the first line
vert1	True if the first line is vertical
a2	Parameter a2 of the second line
b2	Parameter b2 of the second line

vert2	True if the second line is vertical
Ipoint	The intersection point to fill

Returns:

true if there is an only intersection

Definition at line 383 of file Embankment.cpp.

void EmbankmentNMPB::Embankment::SourceImage (double*aCoef*, double*bCoef*, bool*verticalSlope*) [protected]

Calculates the source image by the line z = a * d + b (or d = b if vertical)

Parameters:

aCoef	The a coefficient
bCoef	The b coefficient
verticalSlope	true if vertical line

Definition at line 344 of file Embankment.cpp.

double EmbankmentNMPB::Embankment::ThetaAngle () [protected]

Calculates the Theta angle in the formula (28) p.43.

Returns:

the angle

Definition at line 431 of file Embankment.cpp.

Member Data Documentation

double EmbankmentNMPB::Embankment::_angleAlpha [protected]

the (O1x,O1O2) angle

Definition at line 101 of file Embankment.h.

double EmbankmentNMPB::Embankment::_embankmentAttenuation [protected]

the embankment attenuation

Definition at line 109 of file Embankment.h.

double EmbankmentNMPB::Embankment::_Gembankment [protected]

the slope impedance

Definition at line 105 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::_O1 [protected]

the foot of the bank position
Definition at line 89 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::_O2 [protected]

the top of the bank position
Definition at line 93 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::_platformEnd [protected]

the platform end
Definition at line 85 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::_receiver [protected]

the receiver position (R)
Definition at line 81 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::_source [protected]

the source position (S)
Definition at line 77 of file Embankment.h.

Position2D* EmbankmentNMPB::Embankment::_sourceImage [protected]

the source image position (S")
Definition at line 97 of file Embankment.h.

The documentation for this class was generated from the following files:

- SousCalculs/Embankment.h
- SousCalculs/Embankment.cpp

ExtensionNMPB Struct Reference

Extension for path elements.
`#include <pathdefNMPB.h>`

Public Attributes

- ExtensionTypeNMPB type
  the extension type
- double height
  the height
- double * alphaArray
  absorption coefficients of the surface, for each frequency
- double cosTheta
  the Theta cosines where Theta is the angle between the perpendicular to the road and the propagation path

---

Detailed Description

Extension for path elements.

Definition at line 140 of file pathdefNMPB.h.

Member Data Documentation

double* ExtensionNMPB::alphaArray

absorption coefficients of the surface, for each frequency

Definition at line 153 of file pathdefNMPB.h.

double ExtensionNMPB::cosTheta

the Theta cosines where Theta is the angle between the perpendicular to the road and the propagation path

Definition at line 157 of file pathdefNMPB.h.

double ExtensionNMPB::height

the height

Definition at line 149 of file pathdefNMPB.h.

ExtensionTypeNMPB ExtensionNMPB::type

the extension type

Definition at line 145 of file pathdefNMPB.h.

The documentation for this struct was generated from the following file:

- pathdefNMPB.h

GroundEffectNMPB::GroundEffect Class Reference

Class used to calculate ground effects.
`#include <GroundEffect.h>`

Public Member Functions

- GroundEffect ()
  Initialization of the GroundEffect class.
- GroundEffect (double dp, double zEqS, double zEqR, double Gpath, double Gsource, double freq, GroundCalculationType groundCalculationType)
  Initialization of the GroundEffect class.
- double AttenuationCalculationH ()
  Calculates homogeneous attenuation.
- double AttenuationCalculationF ()
  Calculates favorable attenuation.
- double get_AsolH ()
  Gets ground effect attenuation in homogeneous conditions.
- double get_AsolF ()
  Gets ground effect attenuation in favorable conditions.
- double Get_zEqSource ()
  Gets equivalent height of S measured perpendicular to the mean ground plane.
- double Get_zEqReceiver ()
  Gets equivalent height of R measured perpendicular to the mean ground plane.
- double Get_zEqSource_f ()
  Gets equivalent height of S measured perpendicular to the mean ground plane in favorable conditions.
- double Get_zEqReceiver_f ()
  Gets equivalent height of R measured perpendicular to the mean ground plane in favorable conditions.
- double get_EquivalentGpath ()
  Gets equivalent ground coefficient along a propagation path.

Protected Member Functions

- void CorrectedGroundCoeffCalculation ()
  Calculates corrected ground sound absorption.
- double WparamCalculation (double Gw)
  Calculates the w parameter.
- void CfCalculation ()
  Calculates the Cf parameter.
- void KfreqCalculation ()
  Calculates the k parameter.
- double AttenuationCalculation (double z1, double z2, double cf)
  : Calculates attenuation

Protected Attributes

- double _dp
  distance between source and receiver on the mean plane
- double _zEqS

S equivalent height in homogeneous conditions.

- double _zEqR
  R equivalent height in homogeneous conditions.
- double _zEqS_f
  S equivalent height in favorable conditions.
- double _zEqR_f
  R equivalent height in favorable conditions.
- double _freq
  center frequency fm
- double _Gpath
  ground coefficient
- double _Gsource
  source ground coefficient
- double _correctedGpath
  corrected ground coefficient (with d and z)
- GroundCalculationType _groundCalculationType
  ground Calculation Type (Enum) used to know when using _Gpath or _correctedGpath
- double _kFreq
  k parameter depending on the frequency fm
- double _cfH
  Cf parameter depending on the distance and w, for homogeneous or favorable conditions.
- double _cfF
- double _AsolH
  ground effect attenuation in homogeneous conditions
- double _AsolF
  ground effect attenuation in favorable conditions

Detailed Description

Class used to calculate ground effects.

p. 38-44 - § 9.3

Definition at line 247 of file GroundEffect.h.

Constructor & Destructor Documentation

GroundEffectNMPB::GroundEffect::GroundEffect () [inline]

Initialization of the GroundEffect class.

Definition at line 254 of file GroundEffect.h.

GroundEffectNMPB::GroundEffect::GroundEffect (double*dp*, double*zEqS*, double*zEqR*, double*Gpath*, double*Gsource*, double*freq*, GroundCalculationType*groundCalculationType*) [inline]

Initialization of the GroundEffect class.

Parameters:

dp	: distance between source and receiver on the mean plane
zEqS	: S equivalent height
zEqR	: R equivalent height
Gpath	: ground coefficient
Gsource	: source ground coefficient
freq	: center frequency
groundCalculationType	: ground calculation type (Asol, DeltaSol_SO or DeltaSol_OR)

GroundEffect Initialization with SR mean plane distance, equivalent heights, and center frequency

Definition at line 293 of file GroundEffect.h.

Member Function Documentation

double GroundEffectNMPB::GroundEffect::AttenuationCalculation (double *z1*, double *z2*, double *cf*) [protected]

: Calculates attenuation

Parameters:

z1	: source equivalent height
z2	: receiver equivalent height
cf	: Cf parameter

Returns:

: calculated attenuation

Attenuation calculation depending on distances, heights, frequencies and ground type p. 41 - § 9.3.3 formula (23) before calling max

Exceptions:

ERRDivZero,ERRSqrtNegative	

Definition at line 306 of file GroundEffect.cpp.

double GroundEffectNMPB::GroundEffect::AttenuationCalculationF ()

Calculates favorable attenuation.

Returns:

Asol,H : favorable attenuation

Attenuation calculation in favorable propagation conditions, depending on distances, heights, frequency and ground type p. 42 - § 9.3.4 using formula (23) and (27)

Exceptions:

ERRDivZero	

Returns:

Asol,F : favorable attenuation

Attenuation calculation in favorable propagation conditions, depending on distances, heights, frequency and ground type p. 42 - § 9.3.4 using formula (23) and (27)

Exceptions:

ERRDivZero	

Definition at line 387 of file GroundEffect.cpp.

double GroundEffectNMPB::GroundEffect::AttenuationCalculationH ()

Calculates homogeneous attenuation.

Returns:

Asol,H : homogeneous attenuation

Attenuation calculation in homogeneous propagation conditions, depending on distances, heights, frequency and ground type p. 41 - § 9.3.3 formula (23)

Definition at line 344 of file GroundEffect.cpp.

void GroundEffectNMPB::GroundEffect::CfCalculation () [protected]

Calculates the Cf parameter.

Cf parameter calculation : used in the ground attenuation calculation p. 41 - § 9.3.3 formula (24)

Exceptions:

ERRDivZero	

Definition at line 236 of file GroundEffect.cpp.

void GroundEffectNMPB::GroundEffect::CorrectedGroundCoeffCalculation () [protected]

Calculates corrected ground sound absorption.

Calculates Gpath' : correct Gpath depending on distances p. 39 - § 9.3.2 formula (22)

Definition at line 188 of file GroundEffect.cpp.

double GroundEffectNMPB::GroundEffect::get_AsolF () [inline]

Gets ground effect attenuation in favorable conditions.

Returns:

ground effect attenuation in favorable conditions

Definition at line 353 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::get_AsolH () [inline]

Gets ground effect attenuation in homogeneous conditions.

Returns:

ground effect attenuation in homogeneous conditions

Definition at line 344 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::get_EquivalentGpath () [inline]

Gets equivalent ground coefficient along a propagation path.

Returns:

equivalent ground coefficient along a propagation path

Definition at line 403 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::Get_zEqReceiver () [inline]

Gets equivalent height of R measured perpendicular to the mean ground plane.

Returns:

equivalent height of R measured perpendicular to the mean ground plane

Definition at line 373 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::Get_zEqReceiver_f () [inline]

Gets equivalent height of R measured perpendicular to the mean ground plane in favorable conditions.

Returns:

equivalent height of R measured perpendicular to the mean ground plane in favorable conditions

Definition at line 393 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::Get_zEqSource () [inline]

Gets equivalent height of S measured perpendicular to the mean ground plane.

Returns:

equivalent height of S measured perpendicular to the mean ground plane

Definition at line 363 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::Get_zEqSource_f () [inline]

Gets equivalent height of S measured perpendicular to the mean ground plane in favorable conditions.

Returns:

equivalent height of S measured perpendicular to the mean ground plane in favorable conditions

Definition at line 383 of file GroundEffect.h.

void GroundEffectNMPB::GroundEffect::KfreqCalculation () [protected]

Calculates the k parameter.

k parameter calculation : used in the ground attenuation calculation p. 41 - § 9.3.3 formula (23)

Definition at line 285 of file GroundEffect.cpp.

double GroundEffectNMPB::GroundEffect::WparamCalculation (double *Gw*) [protected]

Calculates the w parameter.

w parameter calculation : used in Cf calculation for ground attenuation p. 41 - § 9.3.3 formula (25)

Parameters:

Gw	: Gpath or corrected Gpath

Exceptions:

ERRDivZero	

Definition at line 211 of file GroundEffect.cpp.

Member Data Documentation

double GroundEffectNMPB::GroundEffect::_AsolF [protected]

ground effect attenuation in favorable conditions

Definition at line 466 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_AsolH [protected]

ground effect attenuation in homogeneous conditions

Definition at line 462 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_cfF [protected]

Definition at line 457 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_cfH [protected]

Cf parameter depending on the distance and w, for homogeneous or favorable conditions.

Definition at line 457 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_correctedGpath [protected]

corrected ground coefficient (with d and z)
Definition at line 444 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_dp [protected]

distance between source and receiver on the mean plane
Definition at line 412 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_freq [protected]

center frequency fm
Definition at line 432 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_Gpath [protected]

ground coefficient
Definition at line 436 of file GroundEffect.h.

GroundCalculationType GroundEffectNMPB::GroundEffect::_groundCalculationType [protected]

ground Calculation Type (Enum) used to know when using _Gpath or _correctedGpath
Definition at line 448 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_Gsource [protected]

source ground coefficient
Definition at line 440 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_kFreq [protected]

k parameter depending on the frequency fm
Definition at line 453 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_zEqR [protected]

R equivalent height in homogeneous conditions.
Definition at line 420 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_zEqR_f [protected]

R equivalent height in favorable conditions.

Definition at line 428 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_zEqS [protected]

S equivalent height in homogeneous conditions.

Definition at line 416 of file GroundEffect.h.

double GroundEffectNMPB::GroundEffect::_zEqS_f [protected]

S equivalent height in favorable conditions.

Definition at line 424 of file GroundEffect.h.

The documentation for this class was generated from the following files:

- SousCalculs/GroundEffect.h
- SousCalculs/GroundEffect.cpp

GroundEffectNMPB::MeanPlane Class Reference

Class used to calculate the mean plane.

`#include <GroundEffect.h>`

Public Member Functions

- MeanPlane ()
  Initialization of the MeanPlane class.
- MeanPlane (vector< ProfilePointNMPB * > terrainItems)
  Initialization of the MeanPlane class with given arguments, and data calculation.
- double Get_dp ()
  Gets distance between source and receiver on the mean plane.
- double Get_zEqSource ()
  Gets Source equivalent height.
- double Get_zEqReceiver ()
  Gets Receiver equivalent height.
- double Get_Gpath ()
  Gets ground coefficient.
- Position2D Get_sourceImage ()
  Gets the source image by the mean plane.
- Position2D Get_receiverImage ()
  Gets the receiver image by the mean plane.
- double Get_aCoeff ()
  Gets the "a" coefficient for the mean plane line equation z = ad + b.

- double Get_bCoeff ()
  Gets the "b" coefficient for the mean plane line equation z = ad + b.
- double Get_Gsource ()
  Gets the impedance source.

Protected Member Functions

- void FillLineCoefficients ()
  Calculates the mean plane coefficients (a and b : z = ad + b)
- Position2D MeanPlaneProjection (Position2D const *point)
  Calculates the mean plane projection of a point.
- void FillData ()
  Calculates distance between source and receiver on the mean plane, S image, R image, S equivalent height, R equivalent height and the ground coefficient.
- void CalculateData ()
  Calculates the mean plane coefficients and then the equivalent heights and the distance on the mean plane.

Protected Attributes

- ProfilePointNMPB * _source
  Source.
- ProfilePointNMPB * _receiver
  Receiver.
- vector< ProfilePointNMPB * > _terrainItems
  terrain items
- vector< Position2D > _pointsList
  items coordinates list
- double _aCoeff
  the mean plane a coefficient (a and b : z = ad + b)
- double _bCoeff
  the mean plane b coefficient (a and b : z = ad + b)
- double _dp
  distance between source and receiver on the mean plane
- double _zEqS
  S equivalent height.
- double _zEqR
  R equivalent height.
- double _Gpath
  ground coefficient
- Position2D _imageS
  source image by the mean plane
- Position2D _imageR
  receiver image by the mean plane

Detailed Description

Class used to calculate the mean plane.

p. 69 - Annexe E

Definition at line 37 of file GroundEffect.h.

Constructor & Destructor Documentation

GroundEffectNMPB::MeanPlane::MeanPlane () [inline]

Initialization of the MeanPlane class.

Definition at line 43 of file GroundEffect.h.

GroundEffectNMPB::MeanPlane::MeanPlane (vector< ProfilePointNMPB * >*terrainItems*)

Initialization of the MeanPlane class with given arguments, and data calculation.

Parameters:

terrainItems	The terrain item vector

Definition at line 21 of file GroundEffect.cpp.

Member Function Documentation

void GroundEffectNMPB::MeanPlane::CalculateData () [inline, protected]

Calculates the mean plane coefficients and then the equivalent heights and the distance on the mean plane.

Definition at line 233 of file GroundEffect.h.

void GroundEffectNMPB::MeanPlane::FillData () [protected]

Calculates distance between source and receiver on the mean plane, S image, R image, S equivalent height, R equivalent height and the ground coefficient.

p. 38 - § 9.3.1 ; p. 39 - § 9.3.2

Definition at line 109 of file GroundEffect.cpp.

void GroundEffectNMPB::MeanPlane::FillLineCoefficients () [protected]

Calculates the mean plane coefficients (a and b : z = ad + b)

p. 69 - Annexe E

Definition at line 49 of file GroundEffect.cpp.

double GroundEffectNMPB::MeanPlane::Get_aCoeff () [inline]

Gets the "a" coefficient for the mean plane line equation z = ad + b.

Returns:
the "a" coefficient for the mean plane line equation z = ad + b
Definition at line 125 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::Get_bCoeff () [inline]

Gets the "b" coefficient for the mean plane line equation z = ad + b.

Returns:
the "b" coefficient for the mean plane line equation z = ad + b
Definition at line 135 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::Get_dp () [inline]

Gets distance between source and receiver on the mean plane.

Returns:
the distance between source and receiver on the mean plane
Definition at line 65 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::Get_Gpath () [inline]

Gets ground coefficient.

Returns:
the ground coefficient
Definition at line 95 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::Get_Gsource () [inline]

Gets the impedance source.

Returns:
the impedance source
Definition at line 145 of file GroundEffect.h.

Position2D GroundEffectNMPB::MeanPlane::Get_receiverImage () [inline]

Gets the receiver image by the mean plane.

Returns:
the receiver image by the mean plane
Definition at line 115 of file GroundEffect.h.

Position2D GroundEffectNMPB::MeanPlane::Get_sourceImage () [inline]

Gets the source image by the mean plane.

Returns:
the source image by the mean plane
Definition at line 105 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::Get_zEqReceiver () [inline]

Gets Receiver equivalent height.

Returns:
the Receiver equivalent height
Definition at line 85 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::Get_zEqSource () [inline]

Gets Source equivalent height.

Returns:
the Source equivalent height
Definition at line 75 of file GroundEffect.h.

Position2D GroundEffectNMPB::MeanPlane::MeanPlaneProjection (Position2D const *point) [protected]

Calculates the mean plane projection of a point.

Parameters:

point	The 2D point to be projected

Returns:
The projected 2D point
Definition at line 93 of file GroundEffect.cpp.

Member Data Documentation

double GroundEffectNMPB::MeanPlane::_aCoeff [protected]

the mean plane a coefficient (a and b : z = ad + b)
Definition at line 175 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::_bCoeff [protected]

the mean plane b coefficient (a and b : z = ad + b)
Definition at line 179 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::_dp [protected]

distance between source and receiver on the mean plane
Definition at line 183 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::_Gpath [protected]

ground coefficient
Definition at line 195 of file GroundEffect.h.

Position2D GroundEffectNMPB::MeanPlane::_imageR [protected]

receiver image by the mean plane
Definition at line 203 of file GroundEffect.h.

Position2D GroundEffectNMPB::MeanPlane::_imageS [protected]

source image by the mean plane
Definition at line 199 of file GroundEffect.h.

vector<Position2D> GroundEffectNMPB::MeanPlane::_pointsList [protected]

items coordinates list
Definition at line 171 of file GroundEffect.h.

ProfilePointNMPB* GroundEffectNMPB::MeanPlane::_receiver [protected]

Receiver.
Definition at line 163 of file GroundEffect.h.

ProfilePointNMPB* GroundEffectNMPB::MeanPlane::_source [protected]

Source.
Definition at line 159 of file GroundEffect.h.

vector<ProfilePointNMPB*> GroundEffectNMPB::MeanPlane::_terrainItems [protected]

terrain items

Definition at line 167 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::_zEqR [protected]

R equivalent height.

Definition at line 191 of file GroundEffect.h.

double GroundEffectNMPB::MeanPlane::_zEqS [protected]

S equivalent height.

Definition at line 187 of file GroundEffect.h.

The documentation for this class was generated from the following files:

- SousCalculs/GroundEffect.h
- SousCalculs/GroundEffect.cpp

Position2D Struct Reference

2D point coordinates in the vertical plane containing Source and Receptor

```
#include <pathdefNMPB.h>
```

Public Attributes

- double d
- double z

Detailed Description

2D point coordinates in the vertical plane containing Source and Receptor

Definition at line 171 of file pathdefNMPB.h.

Member Data Documentation

double Position2D::d

Definition at line 173 of file pathdefNMPB.h.

double Position2D::z

Definition at line 173 of file pathdefNMPB.h.

The documentation for this struct was generated from the following file:

- pathdefNMPB.h

Position3D Struct Reference

3D point coordinates

`#include <pathdefNMPB.h>`

Public Attributes

- double x
- double y
- double z

Detailed Description

3D point coordinates

Definition at line 163 of file pathdefNMPB.h.

Member Data Documentation

double Position3D::x

Definition at line 165 of file pathdefNMPB.h.

double Position3D::y

Definition at line 165 of file pathdefNMPB.h.

double Position3D::z

Definition at line 165 of file pathdefNMPB.h.

The documentation for this struct was generated from the following file:

- pathdefNMPB.h

ProfilePointNMPB Struct Reference

Profile point Structure.
`#include <PathStructures.h>`

Public Member Functions

- ProfilePointNMPB ()
  ProfilePointNMPB initialization.
- ~ProfilePointNMPB (void)
  ProfilePointNMPB destructor.
- void Copy (ProfilePointNMPB const *pointToCopy)
  Copy ProfilePointNMPB data in this.
- Position2D * position2D_withHeight (void)
  Gets the 2D coordinates with the local height added to z.
- Position3D * position3D_withHeight (void)
  Gets the 3D coordinates with the local height added to z.

Public Attributes

- Position3D * position3D
  3D coordinates (x, y, z)
- Position2D * position2D
  2D coordinates (d : cumulated distance, z)
- double impedance
  impedance value before the point
- bool isDiff
  To know if there is diffraction on this point.
- double height
  The local height.
- double h_ray
  To know height of the point to [SR] (or another ray of the convex hull) : < 0 if the ray is under the point.
- ExtensionNMPB * ext
  Extension for path elements.

Detailed Description

Profile point Structure.

Definition at line 49 of file PathStructures.h.

Constructor & Destructor Documentation

ProfilePointNMPB::ProfilePointNMPB () [inline]

ProfilePointNMPB initialization.

Definition at line 83 of file PathStructures.h.

ProfilePointNMPB::~ProfilePointNMPB (void) [inline]

ProfilePointNMPB destructor.

Definition at line 107 of file PathStructures.h.

Member Function Documentation

void ProfilePointNMPB::Copy (ProfilePointNMPB const *pointToCopy) [inline]

Copy ProfilePointNMPB data in this.

Parameters:

pointToCopy	The ProfilePointNMPB to copy data

Definition at line 119 of file PathStructures.h.

Position2D* ProfilePointNMPB::position2D_withHeight (void) [inline]

Gets the 2D coordinates with the local height added to z.

Returns:

the position2D coordinates with height

Definition at line 138 of file PathStructures.h.

Position3D* ProfilePointNMPB::position3D_withHeight (void) [inline]

Gets the 3D coordinates with the local height added to z.

Returns:

the position3D coordinates with height

Definition at line 151 of file PathStructures.h.

Member Data Documentation

ExtensionNMPB* ProfilePointNMPB::ext

Extension for path elements.

Definition at line 78 of file PathStructures.h.

double ProfilePointNMPB::h_ray

To know height of the point to [SR] (or another ray of the convex hull) : < 0 if the ray is under the point.
Definition at line 74 of file PathStructures.h.

double ProfilePointNMPB::height

The local height.
Definition at line 70 of file PathStructures.h.

double ProfilePointNMPB::impedance

impedance value before the point
Definition at line 62 of file PathStructures.h.

bool ProfilePointNMPB::isDiff

To know if there is diffraction on this point.
Definition at line 66 of file PathStructures.h.

Position2D* ProfilePointNMPB::position2D

2D coordinates (d : cumulated distance, z)
Definition at line 58 of file PathStructures.h.

Position3D* ProfilePointNMPB::position3D

3D coordinates (x, y, z)
Definition at line 54 of file PathStructures.h.

The documentation for this struct was generated from the following file:

- PathStructures.h

PropagationPath Struct Reference

Structure for the propagation path.
`#include <PathStructures.h>`

Public Member Functions

- PropagationPath (void)

PropagationPath constructor.

- ~PropagationPath (void)
  PropagationPath destructor.
- void ExtendPath (Position3D const *point3D, double g)
  adds a new terrain item to the terrainItems list
- void ExtendPathExt (Position3D const *point3D, double g, ExtensionNMPB const *ext)
  Adds a new terrain item to the terrainItems list, with extension data.
- bool SetSourceHeight (double h)
  Sets source height.
- bool SetReceiverHeight (double h)
  Sets receiver height.
- ProfilePointNMPB * GetSource ()
  Gets Source.
- ProfilePointNMPB * GetReceiver ()
  Gets Receiver.
- void SetFrequencies (int nbFreq, double const *freq)
  Set frequencies.
- int GetNbFrequencies ()
  Gets frequencies number.
- double const * GetFrequencies ()
  Gets frequencies.
- int GetFrequencyPosition (double freq)
  Search a frequency in the frequencies array.
- void SetOption (Option option, bool on_off)
  Set an option for the path.
- bool GetOption (Option option)
  Get the value of the option.
- void ClearPath ()
  Clears the path items list.

Public Attributes

- vector< ProfilePointNMPB * > pathPoints
  the terrain items list
- double distSR
  distance between S and R
- vector< double > frequencies
  the frequencies list
- double * favorableAttenuations
  the calculated favorable attenuations
- double * homogeneousAttenuations
  the calculated homogeneous attenuations

Detailed Description

Structure for the propagation path.

Definition at line 164 of file PathStructures.h.

Constructor & Destructor Documentation

PropagationPath::PropagationPath (void) [inline]

PropagationPath constructor.

Definition at line 190 of file PathStructures.h.

PropagationPath::~PropagationPath (void) [inline]

PropagationPath destructor.

Definition at line 203 of file PathStructures.h.

Member Function Documentation

void PropagationPath::ClearPath () [inline]

Clears the path items list.

Definition at line 483 of file PathStructures.h.

void PropagationPath::ExtendPath (Position3D const *point3D, double g) [inline]

adds a new terrain item to the terrainItems list

Parameters:

point3D	The 3D coordinates
g	The ground impedance before the point

Definition at line 221 of file PathStructures.h.

void PropagationPath::ExtendPathExt (Position3D const *point3D, double g, ExtensionNMPB const *ext) [inline]

Adds a new terrain item to the terrainItems list, with extension data.

Parameters:

point3D	The 3D coordinates
g	The ground impedance before the point
ext	The extension data

Definition at line 247 of file PathStructures.h.

double const* PropagationPath::GetFrequencies () [inline]

Gets frequencies.

Returns:
the frequencies array
Definition at line 415 of file PathStructures.h.

int PropagationPath::GetFrequencyPosition (double*freq*) [inline]

Search a frequency in the frequencies array.

Parameters:

freq	The searched frequency

Returns:
the frequency position in the array (-1 if not found)
Definition at line 434 of file PathStructures.h.

int PropagationPath::GetNbFrequencies () [inline]

Gets frequencies number.

Returns:
the number of frequencies
Definition at line 405 of file PathStructures.h.

bool PropagationPath::GetOption (Option*option*) [inline]

Get the value of the option.

Parameters:

option	The option to check

Returns:
true if the option is selected
Definition at line 475 of file PathStructures.h.

ProfilePointNMPB* PropagationPath::GetReceiver () [inline]

Gets Receiver.

Returns:
the receiver

Definition at line 371 of file PathStructures.h.

ProfilePointNMPB* PropagationPath::GetSource () [inline]

Gets Source.

Returns:
the source
Definition at line 356 of file PathStructures.h.

void PropagationPath::SetFrequencies (int *nbFreq*, double const **freq*) [inline]

Set frequencies.

Parameters:

nbFreq	The frequencies number
freq	The frequencies array

Definition at line 388 of file PathStructures.h.

void PropagationPath::SetOption (Option *option*, bool *on_off*) [inline]

Set an option for the path.

Parameters:

option	The option to set
on_off	True if the option must be set to true

Definition at line 456 of file PathStructures.h.

bool PropagationPath::SetReceiverHeight (double *h*) [inline]

Sets receiver height.

Parameters:

h	The receiver height

Returns:
true if all OK
Exceptions:

ERRNoPoint	

Definition at line 335 of file PathStructures.h.

bool PropagationPath::SetSourceHeight (double *h*) [inline]

Sets source height.

Parameters:

h	The source height

Returns:

true if all OK

Exceptions:

ERRNoPoint	

Definition at line 310 of file PathStructures.h.

Member Data Documentation

double PropagationPath::distSR

distance between S and R

Definition at line 173 of file PathStructures.h.

double* PropagationPath::favorableAttenuations

the calculated favorable attenuations

Definition at line 181 of file PathStructures.h.

vector<double> PropagationPath::frequencies

the frequencies list

Definition at line 177 of file PathStructures.h.

double* PropagationPath::homogeneousAttenuations

the calculated homogeneous attenuations

Definition at line 185 of file PathStructures.h.

vector<ProfilePointNMPB*> PropagationPath::pathPoints

the terrain items list

Definition at line 169 of file PathStructures.h.

The documentation for this struct was generated from the following file:

- PathStructures.h

File Documentation

CalculPropagation.cpp File Reference

General Calculations for noise propagation with the NMPB 2008 method.

```
#include "CalculPropagation.h"
#include "SousCalculs/Diffraction.h"
#include <math.h>
#include <stdio.h>
#include <assert.h>
#include "../test_mem/safe_new.h"
```

Namespaces

- namespace CalculPropagationNMPB

Functions

- double CalculPropagationNMPB::SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- double CalculPropagationNMPB::SoundLevelForPath (double soundLevel_h, double soundLevel_f, double favourableProbability)
  Calculates long-term sound level for the given the sound levels (homogeneous and favorable) for a path.
- double CalculPropagationNMPB::GetFavorableConditionProbability (Position3D const *sourcePos, Position3D const *receiverPos, int nbAngles, double const *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- void CalculPropagationNMPB::CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)
  Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Detailed Description

General Calculations for noise propagation with the NMPB 2008 method.

Author:
CSTB

Version:
1.0

Definition in file CalculPropagation.cpp.

CalculPropagation.h File Reference

General Calculations for noise propagation with the NMPB 2008 method.

```
#include "PathStructures.h"
#include <vector>
#include <math.h>
```

```
#include <map>
#include <stdlib.h>
#include <string.h>
```

Classes

- class CalculPropagationNMPB::Attenuation

Class used to sound attenuation calculations along a given path, with a specified frequency. Namespaces

- namespace CalculPropagationNMPB

Defines

- #define Attenuation _Local_PROPAN8_Attenuation_

Functions

- double CalculPropagationNMPB::SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- double CalculPropagationNMPB::SoundLevelForPath (double soundLevel_h, double soundLevel_f, double favourableProbability)
  Calculates long-term sound level for the given the sound levels (homogeneous and favorable) for a path.
- double CalculPropagationNMPB::GetFavorableConditionProbability (Position3D const *sourcePos, Position3D const *receiverPos, int nbAngles, double const *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- void CalculPropagationNMPB::CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)
  Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Detailed Description

General Calculations for noise propagation with the NMPB 2008 method.

Author:
CSTB

Version:
1.0

Definition in file CalculPropagation.h.

Define Documentation

#define Attenuation _Local_PROPAN8_Attenuation_

Definition at line 24 of file CalculPropagation.h.

Doxyfile.dox File Reference

Linux/PropagationNMPB08.h File Reference

`#include "pathdefNMPB.h"`

Defines

- #define _COMPILE_NMPB extern "C"

Typedefs

- typedef void * PathID
  the path reference

Functions

- _COMPILE_NMPB PathID NMPB08_CreatePath ()
  Creates path calculator with default frequency range.
- _COMPILE_NMPB PathID NMPB08_CreatePathEx (int nbFreq, double const *freq)
  Creates path calculator with user defined frequency range.
- _COMPILE_NMPB bool NMPB08_SetOption (PathID, Option option, bool on_off)
  Set an option for the path.
- _COMPILE_NMPB int NMPB08_GetNbFrequencies (PathID)
  Gets the frequencies number.
- _COMPILE_NMPB double const * NMPB08_GetFrequencies (PathID)
  Gets the frequencies array.
- _COMPILE_NMPB bool NMPB08_DeletePath (PathID)
  Delete the path calculator.
- _COMPILE_NMPB bool NMPB08_ClearPath (PathID)
  clear the path profile
- _COMPILE_NMPB bool NMPB08_ExtendPath (PathID, Position3D const *point3D, double g)
  Add a segment to the path profile.
- _COMPILE_NMPB bool NMPB08_ExtendPathExt (PathID, Position3D const *point3D, double g, ExtensionNMPB const *ext)
  Add a segment to the path profile with extension data.
- _COMPILE_NMPB int NMPB08_SetSourceHeight (PathID, double h)
  Set the source height.
- _COMPILE_NMPB int NMPB08_SetReceiverHeight (PathID, double h)
  Set the receiver height.
- _COMPILE_NMPB int NMPB08_DoCalculation (PathID)
  Do the propagation calculation.
- _COMPILE_NMPB double const * NMPB08_GetAttF (PathID)
  Get path attenuations under favorable conditions.
- _COMPILE_NMPB double const * NMPB08_GetAttH (PathID)
  Get path attenuations under homogeneous conditions.
- _COMPILE_NMPB double NMPB08_Leq_LT (double soundLevel_h, double soundLevel_f, double p)

Calculates long-term sound level for the given the sound levels (homogeneous and favorable)

- _COMPILE_NMPB double NMPB08_SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- _COMPILE_NMPB double NMPB08_GetFavorableConditionProbability (Position3D const *source, Position3D const *receiver, int nbAngles, double const *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- _COMPILE_NMPB int NMPB08_CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)
  Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Define Documentation

#define _COMPILE_NMPB extern "C"

Definition at line 42 of file PropagationNMPB08.h.

Typedef Documentation

typedef void* PathID

the path reference

Definition at line 50 of file PropagationNMPB08.h.

Function Documentation

_COMPILE_NMPB int NMPB08_CalculateLeqLT (int *nbFreq*, double const **Lw*, double const **attH*, double const **attF*, double *fcp*, double **LeqH*, double **LeqF*, double **LeqLT*)

Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Parameters:

nbFreq	The frequencies number (user data)
Lw	The sound power levels of the source, in each frequency band (user data)
attH	Attenuations due to the propagation between source and receiver in homogeneous conditions, in each frequency band (user data)
attF	Attenuations due to the propagation between source and receiver in downward-refraction conditions, in each frequency band (user data)
fcp	Probability of occurrence of downward-refraction conditions over a long-term period in a given direction, p in [0, 1] (user data)
LeqH	Sound levels due to source Si at point R in homogeneous conditions, in each frequency band (calculated in this function)
LeqF	Sound levels due to source Si at point R in downward-refraction conditions, in each frequency band (calculated in this function)
LeqLT	Long-term sound levels due to source Si at point R, in each given frequency

	band (calculated in this function)

Returns:

the exception number (0 if no exception)

Definition at line 682 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ClearPath (PathID *path*)

clear the path profile

Parameters:

path	The path to clear

Returns:

true if all OK

Definition at line 214 of file PropagationNMPB08.cpp.

_COMPILE_NMPB PathID NMPB08_CreatePath ()

Creates path calculator with default frequency range.

Returns:

the new Propagation Path

Definition at line 41 of file PropagationNMPB08.cpp.

_COMPILE_NMPB PathID NMPB08_CreatePathEx (int *nbFreq*, double const **freq*)

Creates path calculator with user defined frequency range.

Parameters:

nbFreq	: frequencies number
freq	: frequencies values

Returns:

the new Propagation Path

Definition at line 74 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_DeletePath (PathID *path*)

Delete the path calculator.

Parameters:

path	The path to delete

Returns:

true if all OK

Definition at line 186 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_DoCalculation (PathID *path*)

Do the propagation calculation.

Parameters:

path	The path to do propagation calculation

Returns:

the exception number (0 if no exception)

Definition at line 398 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ExtendPath (PathID *path*, Position3D const **point3D*, double *g*)

Add a segment to the path profile.

Parameters:

path	The path to add segment
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point

Returns:

true if all OK

Definition at line 247 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ExtendPathExt (PathID *path*, Position3D const **point3D*, double *g*, ExtensionNMPB const **ext*)

Add a segment to the path profile with extension data.

Parameters:

path	The path to add segment.
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point
ext	The extension data

Returns:

true if all OK

Definition at line 281 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetAttF (PathID *path*)

Get path attenuations under favorable conditions.

Parameters:

path	The path to get attenuations

Returns:

the favorable attenuation for each frequency range

Definition at line 519 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetAttH (PathID*path*)

Get path attenuations under homogeneous conditions.

Parameters:

path	The path to get attenuations

Returns:

the homogeneous attenuation for each frequency range

Definition at line 546 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_GetFavorableConditionProbability (Position3D const **source*, Position3D const **receiver*, int*nbAngles*, const double **fcpAngles*, double*angleNorth*)

Calculates the favorable conditions probability for the (SR) direction.

Parameters:

source	The source position 3D
receiver	The receiver position 3D
nbAngles	The number of angle probabilities (should be 18)
fcpAngles	The favorable conditions probabilities for the angles 20, 40, 60, ..., 360
angleNorth	The north direction (Ox,ON)

Returns:

the favorable conditions probability for the (SR) direction

Definition at line 637 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetFrequencies (PathID*path*)

Gets the frequencies array.

Parameters:

path	The path containing frequencies

Returns:

the frequencies array

Definition at line 159 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_GetNbFrequencies (PathID*path*)

Gets the frequencies number.

Parameters:

path	The path containing frequencies

Returns:

the frequencies number

Definition at line 132 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_Leq_LT (double*soundLevel_h*, double*soundLevel_f*, double*p*)

Calculates long-term sound level for the given the sound levels (homogeneous and favorable)

Parameters:

soundLevel_h	: sound level in homogeneous conditions
soundLevel_f	: sound level in downward-refraction conditions
p	The average occurrence of downward-refraction conditions in the direction of the path

Returns:

the calculated sound level

Definition at line 577 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_SetOption (PathID*path*, Option*option*, bool*on_off*)

Set an option for the path.

Parameters:

path	The path to set the option
option	The option to set
on_off	True if the option must be set to true

Returns:

true if all OK

Definition at line 104 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_SetReceiverHeight (PathID*path*, double*h*)

Set the receiver height.

Parameters:

path	The path to set receiver height
h	The receiver height

Returns:

the exception number (0 if no exception)

Definition at line 341 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_SetSourceHeight (PathID*path*, double*h*)

Set the source height.

Parameters:

path	The path to set source height
h	The source height

Returns:

the exception number (0 if no exception)

Definition at line 311 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_SumLevels (int *n*, double const **levels*)

Calculates sound levels sum.

Used for § 5.2.4 Formula (8) and § 5.2.5 Formula (9)

Parameters:

n	the sound levels number
levels	The sound levels to sum

Returns:

the sum

Definition at line 605 of file PropagationNMPB08.cpp.

MingW/PropagationNMPB08.h File Reference

`#include "pathdefNMPB.h"`

Defines

- #define _COMPILE_NMPB extern "C"

Typedefs

- typedef void * PathID
  the path reference

Functions

- _COMPILE_NMPB PathID NMPB08_CreatePath ()
  Creates path calculator with default frequency range.
- _COMPILE_NMPB PathID NMPB08_CreatePathEx (int nbFreq, double const *freq)
  Creates path calculator with user defined frequency range.
- _COMPILE_NMPB bool NMPB08_SetOption (PathID, Option option, bool on_off)
  Set an option for the path.
- _COMPILE_NMPB int NMPB08_GetNbFrequencies (PathID)
  Gets the frequencies number.
- _COMPILE_NMPB double const * NMPB08_GetFrequencies (PathID)
  Gets the frequencies array.
- _COMPILE_NMPB bool NMPB08_DeletePath (PathID)
  Delete the path calculator.
- _COMPILE_NMPB bool NMPB08_ClearPath (PathID)

clear the path profile

- _COMPILE_NMPB bool NMPB08_ExtendPath (PathID, Position3D const *point3D, double g)
  Add a segment to the path profile.
- _COMPILE_NMPB bool NMPB08_ExtendPathExt (PathID, Position3D const *point3D, double g, ExtensionNMPB const *ext)
  Add a segment to the path profile with extension data.
- _COMPILE_NMPB int NMPB08_SetSourceHeight (PathID, double h)
  Set the source height.
- _COMPILE_NMPB int NMPB08_SetReceiverHeight (PathID, double h)
  Set the receiver height.
- _COMPILE_NMPB int NMPB08_DoCalculation (PathID)
  Do the propagation calculation.
- _COMPILE_NMPB double const * NMPB08_GetAttF (PathID)
  Get path attenuations under favorable conditions.
- _COMPILE_NMPB double const * NMPB08_GetAttH (PathID)
  Get path attenuations under homogeneous conditions.
- _COMPILE_NMPB double NMPB08_Leq_LT (double soundLevel_h, double soundLevel_f, double p)
  Calculates long-term sound level for the given the sound levels (homogeneous and favorable)
- _COMPILE_NMPB double NMPB08_SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- _COMPILE_NMPB double NMPB08_GetFavorableConditionProbability (Position3D const *source, Position3D const *receiver, int nbAngles, double const *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- _COMPILE_NMPB int NMPB08_CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)
  Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Define Documentation

#define _COMPILE_NMPB extern "C"

Definition at line 42 of file PropagationNMPB08.h.

Typedef Documentation

typedef void* PathID

the path reference

Definition at line 50 of file PropagationNMPB08.h.

Function Documentation

_COMPILE_NMPB int NMPB08_CalculateLeqLT (int *nbFreq*, double const **Lw*, double const **attH*, double const **attF*, double *fcp*, double **LeqH*, double **LeqF*, double **LeqLT*)

Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Parameters:

nbFreq	The frequencies number (user data)
Lw	The sound power levels of the source, in each frequency band (user data)
attH	Attenuations due to the propagation between source and receiver in homogeneous conditions, in each frequency band (user data)
attF	Attenuations due to the propagation between source and receiver in downward-refraction conditions, in each frequency band (user data)
fcp	Probability of occurrence of downward-refraction conditions over a long-term period in a given direction, p in [0, 1] (user data)
LeqH	Sound levels due to source Si at point R in homogeneous conditions, in each frequency band (calculated in this function)
LeqF	Sound levels due to source Si at point R in downward-refraction conditions, in each frequency band (calculated in this function)
LeqLT	Long-term sound levels due to source Si at point R, in each given frequency band (calculated in this function)

Returns:

the exception number (0 if no exception)

Definition at line 682 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ClearPath (PathID *path*)

clear the path profile

Parameters:

path	The path to clear

Returns:

true if all OK

Definition at line 214 of file PropagationNMPB08.cpp.

_COMPILE_NMPB PathID NMPB08_CreatePath ()

Creates path calculator with default frequency range.

Returns:

the new Propagation Path

Definition at line 41 of file PropagationNMPB08.cpp.

_COMPILE_NMPB PathID NMPB08_CreatePathEx (int *nbFreq*, double const **freq*)

Creates path calculator with user defined frequency range.

Parameters:

nbFreq	: frequencies number
freq	: frequencies values

Returns:

the new Propagation Path

Definition at line 74 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_DeletePath (PathID*path*)

Delete the path calculator.

Parameters:

path	The path to delete

Returns:

true if all OK

Definition at line 186 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_DoCalculation (PathID*path*)

Do the propagation calculation.

Parameters:

path	The path to do propagation calculation

Returns:

the exception number (0 if no exception)

Definition at line 398 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ExtendPath (PathID*path*, Position3D const **point3D*, double*g*)

Add a segment to the path profile.

Parameters:

path	The path to add segment
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point

Returns:

true if all OK

Definition at line 247 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ExtendPathExt (PathID*path*, Position3D const **point3D*, double*g*, ExtensionNMPB const **ext*)

Add a segment to the path profile with extension data.

Parameters:

path	The path to add segment.
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point
ext	The extension data

Returns:

true if all OK

Definition at line 281 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetAttF (PathID*path*)

Get path attenuations under favorable conditions.

Parameters:

path	The path to get attenuations

Returns:

the favorable attenuation for each frequency range

Definition at line 519 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetAttH (PathID*path*)

Get path attenuations under homogeneous conditions.

Parameters:

path	The path to get attenuations

Returns:

the homogeneous attenuation for each frequency range

Definition at line 546 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_GetFavorableConditionProbability (Position3D const **source*, Position3D const **receiver*, int*nbAngles*, const double **fcpAngles*, double*angleNorth*)

Calculates the favorable conditions probability for the (SR) direction.

Parameters:

source	The source position 3D
receiver	The receiver position 3D
nbAngles	The number of angle probabilities (should be 18)
fcpAngles	The favorable conditions probabilities for the angles 20, 40, 60, ..., 360
angleNorth	The north direction (Ox,ON)

Returns:

the favorable conditions probability for the (SR) direction

Definition at line 637 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetFrequencies (PathID *path*)

Gets the frequencies array.

Parameters:

path	The path containing frequencies

Returns:

the frequencies array

Definition at line 159 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_GetNbFrequencies (PathID *path*)

Gets the frequencies number.

Parameters:

path	The path containing frequencies

Returns:

the frequencies number

Definition at line 132 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_Leq_LT (double *soundLevel_h*, double *soundLevel_f*, double *p*)

Calculates long-term sound level for the given the sound levels (homogeneous and favorable)

Parameters:

soundLevel_h	: sound level in homogeneous conditions
soundLevel_f	: sound level in downward-refraction conditions
p	The average occurrence of downward-refraction conditions in the direction of the path

Returns:

the calculated sound level

Definition at line 577 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_SetOption (PathID *path*, Option *option*, bool *on_off*)

Set an option for the path.

Parameters:

path	The path to set the option
option	The option to set
on_off	True if the option must be set to true

Returns:

true if all OK

Definition at line 104 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_SetReceiverHeight (PathID*path*, double*h*)

Set the receiver height.

Parameters:

path	The path to set receiver height
h	The receiver height

Returns:

the exception number (0 if no exception)

Definition at line 341 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_SetSourceHeight (PathID*path*, double*h*)

Set the source height.

Parameters:

path	The path to set source height
h	The source height

Returns:

the exception number (0 if no exception)

Definition at line 311 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_SumLevels (int*n*, double const **levels*)

Calculates sound levels sum.

Used for § 5.2.4 Formula (8) and § 5.2.5 Formula (9)

Parameters:

n	the sound levels number
levels	The sound levels to sum

Returns:

the sum

Definition at line 605 of file PropagationNMPB08.cpp.

PropagationNMPB08.h File Reference

```
#include "pathdefNMPB.h"
```

Defines

- #define _COMPILE_NMPB extern "C"

Typedefs

- typedef void * PathID
  the path reference

Functions

- _COMPILE_NMPB PathID NMPB08_CreatePath ()
  Creates path calculator with default frequency range.
- _COMPILE_NMPB PathID NMPB08_CreatePathEx (int nbFreq, double const *freq)
  Creates path calculator with user defined frequency range.
- _COMPILE_NMPB bool NMPB08_SetOption (PathID, Option option, bool on_off)
  Set an option for the path.
- _COMPILE_NMPB int NMPB08_GetNbFrequencies (PathID)
  Gets the frequencies number.
- _COMPILE_NMPB double const * NMPB08_GetFrequencies (PathID)
  Gets the frequencies array.
- _COMPILE_NMPB bool NMPB08_DeletePath (PathID)
  Delete the path calculator.
- _COMPILE_NMPB bool NMPB08_ClearPath (PathID)
  clear the path profile
- _COMPILE_NMPB bool NMPB08_ExtendPath (PathID, Position3D const *point3D, double g)
  Add a segment to the path profile.
- _COMPILE_NMPB bool NMPB08_ExtendPathExt (PathID, Position3D const *point3D, double g, ExtensionNMPB const *ext)
  Add a segment to the path profile with extension data.
- _COMPILE_NMPB int NMPB08_SetSourceHeight (PathID, double h)
  Set the source height.
- _COMPILE_NMPB int NMPB08_SetReceiverHeight (PathID, double h)
  Set the receiver height.
- _COMPILE_NMPB int NMPB08_DoCalculation (PathID)
  Do the propagation calculation.
- _COMPILE_NMPB double const * NMPB08_GetAttF (PathID)
  Get path attenuations under favorable conditions.
- _COMPILE_NMPB double const * NMPB08_GetAttH (PathID)
  Get path attenuations under homogeneous conditions.
- _COMPILE_NMPB double NMPB08_Leq_LT (double soundLevel_h, double soundLevel_f, double p)
  Calculates long-term sound level for the given the sound levels (homogeneous and favorable)
- _COMPILE_NMPB double NMPB08_SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- _COMPILE_NMPB double NMPB08_GetFavorableConditionProbability (Position3D const *source, Position3D const *receiver, int nbAngles, double const *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- _COMPILE_NMPB int NMPB08_CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)

Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Define Documentation

#define _COMPILE_NMPB extern "C"

Definition at line 42 of file PropagationNMPB08.h.

Typedef Documentation

typedef void* PathID

the path reference

Definition at line 50 of file PropagationNMPB08.h.

Function Documentation

_COMPILE_NMPB int NMPB08_CalculateLeqLT (int*nbFreq*, double const **Lw*, double const **attH*, double const **attF*, double*fcp*, double **LeqH*, double **LeqF*, double **LeqLT*)

Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Parameters:

nbFreq	The frequencies number (user data)
Lw	The sound power levels of the source, in each frequency band (user data)
attH	Attenuations due to the propagation between source and receiver in homogeneous conditions, in each frequency band (user data)
attF	Attenuations due to the propagation between source and receiver in downward-refraction conditions, in each frequency band (user data)
fcp	Probability of occurrence of downward-refraction conditions over a long-term period in a given direction, p in [0, 1] (user data)
LeqH	Sound levels due to source Si at point R in homogeneous conditions, in each frequency band (calculated in this function)
LeqF	Sound levels due to source Si at point R in downward-refraction conditions, in each frequency band (calculated in this function)
LeqLT	Long-term sound levels due to source Si at point R, in each given frequency band (calculated in this function)

Returns:

the exception number (0 if no exception)

Definition at line 682 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ClearPath (PathID*path*)

clear the path profile

Parameters:

path	The path to clear

Returns:

true if all OK

Definition at line 214 of file PropagationNMPB08.cpp.

_COMPILE_NMPB PathID NMPB08_CreatePath ()

Creates path calculator with default frequency range.

Returns:

the new Propagation Path

Definition at line 41 of file PropagationNMPB08.cpp.

_COMPILE_NMPB PathID NMPB08_CreatePathEx (int *nbFreq*, double const **freq*)

Creates path calculator with user defined frequency range.

Parameters:

nbFreq	: frequencies number
freq	: frequencies values

Returns:

the new Propagation Path

Definition at line 74 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_DeletePath (PathID *path*)

Delete the path calculator.

Parameters:

path	The path to delete

Returns:

true if all OK

Definition at line 186 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_DoCalculation (PathID *path*)

Do the propagation calculation.

Parameters:

path	The path to do propagation calculation

Returns:

the exception number (0 if no exception)

Definition at line 398 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ExtendPath (PathID *path*, Position3D const **point3D*, double *g*)

Add a segment to the path profile.

Parameters:

path	The path to add segment
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point

Returns:

true if all OK

Definition at line 247 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_ExtendPathExt (PathID *path*, Position3D const **point3D*, double *g*, ExtensionNMPB const **ext*)

Add a segment to the path profile with extension data.

Parameters:

path	The path to add segment.
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point
ext	The extension data

Returns:

true if all OK

Definition at line 281 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetAttF (PathID *path*)

Get path attenuations under favorable conditions.

Parameters:

path	The path to get attenuations

Returns:

the favorable attenuation for each frequency range

Definition at line 519 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetAttH (PathID *path*)

Get path attenuations under homogeneous conditions.

Parameters:

path	The path to get attenuations

Returns:

the homogeneous attenuation for each frequency range

Definition at line 546 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_GetFavorableConditionProbability (Position3D const *source, Position3D const *receiver, int nbAngles, const double *fcpAngles, double angleNorth)

Calculates the favorable conditions probability for the (SR) direction.

Parameters:

source	The source position 3D
receiver	The receiver position 3D
nbAngles	The number of angle probabilities (should be 18)
fcpAngles	The favorable conditions probabilities for the angles 20, 40, 60, ..., 360
angleNorth	The north direction (Ox,ON)

Returns:

the favorable conditions probability for the (SR) direction

Definition at line 637 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double const* NMPB08_GetFrequencies (PathID *path*)

Gets the frequencies array.

Parameters:

path	The path containing frequencies

Returns:

the frequencies array

Definition at line 159 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_GetNbFrequencies (PathID *path*)

Gets the frequencies number.

Parameters:

path	The path containing frequencies

Returns:

the frequencies number

Definition at line 132 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_Leq_LT (double *soundLevel_h*, double *soundLevel_f*, double *p*)

Calculates long-term sound level for the given the sound levels (homogeneous and favorable)

Parameters:

soundLevel_h	: sound level in homogeneous conditions
soundLevel_f	: sound level in downward-refraction conditions
p	The average occurrence of downward-refraction conditions in the direction of the path

Returns:

the calculated sound level

Definition at line 577 of file PropagationNMPB08.cpp.

_COMPILE_NMPB bool NMPB08_SetOption (PathID*path*, Option*option*, bool*on_off*)

Set an option for the path.

Parameters:

path	The path to set the option
option	The option to set
on_off	True if the option must be set to true

Returns:

true if all OK

Definition at line 104 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_SetReceiverHeight (PathID*path*, double*h*)

Set the receiver height.

Parameters:

path	The path to set receiver height
h	The receiver height

Returns:

the exception number (0 if no exception)

Definition at line 341 of file PropagationNMPB08.cpp.

_COMPILE_NMPB int NMPB08_SetSourceHeight (PathID*path*, double*h*)

Set the source height.

Parameters:

path	The path to set source height
h	The source height

Returns:

the exception number (0 if no exception)

Definition at line 311 of file PropagationNMPB08.cpp.

_COMPILE_NMPB double NMPB08_SumLevels (int *n*, double const **levels*)

Calculates sound levels sum.

Used for § 5.2.4 Formula (8) and § 5.2.5 Formula (9)

Parameters:

n	the sound levels number
levels	The sound levels to sum

Returns:

the sum

Definition at line 605 of file PropagationNMPB08.cpp.

---

pathdefNMPB.h File Reference

Definition of main enumerations and structures used in the call of the library functions.

Classes

- struct ExtensionNMPB
- *Extension for path elements*. struct Position3D
- *3D point coordinates* struct Position2D

2D point coordinates in the vertical plane containing Source and Receptor Enumerations

- enum ErrorType { ERRNone = 0, ERRNullPath = 10, ERRNoPoint = 11, ERROnePoint = 12, ERRSideDiff = 13, ERREmbankment = 14, ERRFrequency = 20, ERRAttCoeffFrequency = 21, ERRAngle = 30, ERRProbability = 40, ERRDivZero = 50, ERRSqrtNegative = 51, ERRScreenAbsorption = 60, ERRUnknown = 100 }
- *Error types send by functions.* enum Option { EXCLUDE_ADIV = 1 << 1, EXCLUDE_AATM = 1 << 2, TRACE_DETAILS = 1 << 3, CHECK_EMBANKMENT = 1 << 4, FORCE_CH_EQUAL_ONE = 1 << 5 }
- *Options that can be chosen by user.* enum ExtensionTypeNMPB { ETNone_NMPB = 0, ETScreen_NMPB = 1, ETReflection_NMPB = 2, ETSideDiffraction_NMPB = 3, ETPlatform_NMPB = 4, ETEmbankment_NMPB = 5, ETRoadSource_NMPB = 6 }

extension type

Detailed Description

Definition of main enumerations and structures used in the call of the library functions.

Author:

CSTB

Version:

1.0

Definition in file pathdefNMPB.h.

Enumeration Type Documentation

enum ErrorType

Error types send by functions.

Enumerator:

ERRNone No error.

ERRNullPath The PathID path is null.

ERRNoPoint There is no point in the path.

ERROnePoint There is only one point in the path.

ERRSideDiff There are more than 2 side diffractions (ProfilePointNMPB with ext.type = ETSideDiffraction_NMPB) in the path.

ERREmbankment There are more than 1 embankment (ProfilePointNMPB with ext.type = ETEmbankment_NMPB) in the path.

ERRFrequency The frequency equals 0.

ERRAttCoeffFrequency Frequency not found to get attenuation coefficient.

ERRAngle Angle not found in the array containing favorable conditions probabilities for the angles.

ERRProbability The probability is greater than 1.

ERRDivZero Division by zero.

ERRSqrtNegative Square root of negative number.

ERRScreenAbsorption The given screen absorption is greater than 1.

ERRUnknown None expected error.

Definition at line 15 of file pathdefNMPB.h.

enum ExtensionTypeNMPB

extension type

Enumerator:

ETNone_NMPB No particular extension.

ETScreen_NMPB Screen extension : height must be filled.

ETReflection_NMPB Reflection extension : height and alphaArray must be filled.

ETSideDiffraction_NMPB Side diffraction extension : height must be filled.

ETPlatform_NMPB Platform extension.

ETEmbankment_NMPB Embankment extension : cosTheta must be filled.

ETRoadSource_NMPB Road source extension.

Definition at line 105 of file pathdefNMPB.h.

enum Option

Options that can be chosen by user.

Enumerator:

EXCLUDE_ADIV Don't calculate Adiv.

EXCLUDE_AATM Don't calculate Aatm.

TRACE_DETAILS Displays trace details on console.

CHECK_EMBANKMENT Check embankments on the path.

FORCE_CH_EQUAL_ONE Disable correction term for diffraction by low height obstacles, always use Ch = 1.

Definition at line 78 of file pathdefNMPB.h.

PathStructures.cpp File Reference

Used functions in the library.

```
#include "PathStructures.h"
```

Functions

- double distance2D (Position2D const *position1, Position2D const *position2)
  Computes the distance between 2 2D points.
- double distance3D (Position3D const *position1, Position3D const *position2)
  Computes the distance between 2 3D points.
- double GroundDistance (Position3D const *position1, Position3D const *position2)
  Computes the ground distance between 2 positions (ie distance with x and y, but not z)
- void FillPlanePosition (ProfilePointNMPB *terrain, ProfilePointNMPB const *source, double cumDistance)
  Fills the TerrainItem Position2D.

Detailed Description

Used functions in the library.

Author:

CSTB

Version:

1.0

Definition in file PathStructures.cpp.

Function Documentation

double distance2D (Position2D const *position1, Position2D const *position2)

Computes the distance between 2 2D points.

Parameters:

position1	The first point coordinates
position2	The second point coordinates

Returns:

the distance between the 2 positions

Definition at line 20 of file PathStructures.cpp.

double distance3D (Position3D const *position1, Position3D const *position2)

Computes the distance between 2 3D points.

Parameters:

position1	The first point coordinates
position2	The second point coordinates

Returns:

the distance between the 2 positions

Definition at line 34 of file PathStructures.cpp.

void FillPlanePosition (ProfilePointNMPB *terrain, ProfilePointNMPB const *source, double*cumDistance*)

Fills the TerrainItem Position2D.

Parameters:

terrain	The terrain item
source	The source
cumDistance	The cumulated ground distance

Definition at line 62 of file PathStructures.cpp.

double GroundDistance (Position3D const *position1, Position3D const *position2)

Computes the ground distance between 2 positions (ie distance with x and y, but not z)

Parameters:

position1	The first point coordinates
position2	The second point coordinates

Returns:

the ground distance

Definition at line 48 of file PathStructures.cpp.

PathStructures.h File Reference

Definition of main constants and structures used in the library.

```
#include "pathdefNMPB.h"
#include <vector>
#include "math.h"
#include "stdio.h"
```

Classes

- struct ProfilePointNMPB
- *Profile point Structure.* struct PropagationPath

Structure for the propagation path. Defines

- #define ProfilePointNMPB _Local_PROPAN8_ProfilePointNMPB_
- #define PropagationPath _Local_PROPAN8_PropagationPath_
- #define NULE 0
- #define cSound ((double) 340)
  sound speed in the air
- #define PI ((double) 3.14159265358979323)
  used value for Pi

Functions

- double distance2D (Position2D const *position1, Position2D const *position2)
  Computes the distance between 2 2D points.
- double distance3D (Position3D const *position1, Position3D const *position2)
  Computes the distance between 2 3D points.
- double GroundDistance (Position3D const *position1, Position3D const *position2)
  Computes the ground distance between 2 positions (ie distance with x and y, but not z)
- void FillPlanePosition (ProfilePointNMPB *terrain, ProfilePointNMPB const *source, double cumDistance)
  Fills the TerrainItem Position2D.

Detailed Description

Definition of main constants and structures used in the library.

Author:
CSTB

Version:
1.0

Definition in file PathStructures.h.

Define Documentation

#define cSound ((double) 340)

sound speed in the air

Definition at line 31 of file PathStructures.h.

#define NULE 0

Definition at line 26 of file PathStructures.h.

#define PI ((double) 3.14159265358979323)

used value for Pi

Definition at line 35 of file PathStructures.h.

#define ProfilePointNMPB _Local_PROPAN8_ProfilePointNMPB_

Definition at line 22 of file PathStructures.h.

#define PropagationPath _Local_PROPAN8_PropagationPath_

Definition at line 23 of file PathStructures.h.

Function Documentation

double distance2D (Position2D const *position1, Position2D const *position2)

Computes the distance between 2 2D points.

Parameters:

position1	The first point coordinates
position2	The second point coordinates

Returns:

the distance between the 2 positions

Definition at line 20 of file PathStructures.cpp.

double distance3D (Position3D const *position1, Position3D const *position2)

Computes the distance between 2 3D points.

Parameters:

position1	The first point coordinates
position2	The second point coordinates

Returns:

the distance between the 2 positions

Definition at line 34 of file PathStructures.cpp.

void FillPlanePosition (ProfilePointNMPB **terrain*, ProfilePointNMPB const **source*, double*cumDistance*)

Fills the TerrainItem Position2D.

Parameters:

terrain	The terrain item
source	The source
cumDistance	The cumulated ground distance

Definition at line 62 of file PathStructures.cpp.

double GroundDistance (Position3D const **position1*, Position3D const **position2*)

Computes the ground distance between 2 positions (ie distance with x and y, but not z)

Parameters:

position1	The first point coordinates
position2	The second point coordinates

Returns:

the ground distance

Definition at line 48 of file PathStructures.cpp.

PropagationNMPB08.cpp File Reference

Definition of the library functions that can be called by external software.

```
#include "PropagationNMPB08.h"
#include "CalculPropagation.h"
#include "SousCalculs/ElementaryPath.h"
#include <vector>
#include <math.h>
#include <stdio.h>
```

Functions

- bool CheckPath (PathID path)
  Check if the path is OK.
- PathID NMPB08_CreatePath ()

Creates path calculator with default frequency range.

- PathID NMPB08_CreatePathEx (int nbFreq, double const *freq)
  Creates path calculator with user defined frequency range.
- bool NMPB08_SetOption (PathID path, Option option, bool on_off)
  Set an option for the path.
- int NMPB08_GetNbFrequencies (PathID path)
  Gets the frequencies number.
- double const * NMPB08_GetFrequencies (PathID path)
  Gets the frequencies array.
- bool NMPB08_DeletePath (PathID path)
  Delete the path calculator.
- bool NMPB08_ClearPath (PathID path)
  clear the path profile
- bool NMPB08_ExtendPath (PathID path, Position3D const *point3D, double g)
  Add a segment to the path profile.
- bool NMPB08_ExtendPathExt (PathID path, Position3D const *point3D, double g, ExtensionNMPB const *ext)
  Add a segment to the path profile with extension data.
- int NMPB08_SetSourceHeight (PathID path, double h)
  Set the source height.
- int NMPB08_SetReceiverHeight (PathID path, double h)
  Set the receiver height.
- int NMPB08_DoCalculation (PathID path)
  Do the propagation calculation.
- double const * NMPB08_GetAttF (PathID path)
  Get path attenuations under favorable conditions.
- double const * NMPB08_GetAttH (PathID path)
  Get path attenuations under homogeneous conditions.
- double NMPB08_Leq_LT (double soundLevel_h, double soundLevel_f, double p)
  Calculates long-term sound level for the given the sound levels (homogeneous and favorable)
- double NMPB08_SumLevels (int n, double const *levels)
  Calculates sound levels sum.
- double NMPB08_GetFavorableConditionProbability (Position3D const *source, Position3D const *receiver, int nbAngles, const double *fcpAngles, double angleNorth)
  Calculates the favorable conditions probability for the (SR) direction.
- int NMPB08_CalculateLeqLT (int nbFreq, double const *Lw, double const *attH, double const *attF, double fcp, double *LeqH, double *LeqF, double *LeqLT)
  Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Detailed Description

Definition of the library functions that can be called by external software.

Author:

CSTB

Version:
1.0
Definition in file PropagationNMPB08.cpp.

Function Documentation

bool CheckPath (PathID *path*)

Check if the path is OK.

Parameters:

path	the path

Returns:
true if all is OK
Definition at line 25 of file PropagationNMPB08.cpp.

int NMPB08_CalculateLeqLT (int *nbFreq*, double const **Lw*, double const **attH*, double const **attF*, double *fcp*, double **LeqH*, double **LeqF*, double **LeqLT*)

Calculates the Long-term sound level Leq due to one source at point R, in each given frequency band.

Parameters:

nbFreq	The frequencies number (user data)
Lw	The sound power levels of the source, in each frequency band (user data)
attH	Attenuations due to the propagation between source and receiver in homogeneous conditions, in each frequency band (user data)
attF	Attenuations due to the propagation between source and receiver in downward-refraction conditions, in each frequency band (user data)
fcp	Probability of occurrence of downward-refraction conditions over a long-term period in a given direction, p in [0, 1] (user data)
LeqH	Sound levels due to source Si at point R in homogeneous conditions, in each frequency band (calculated in this function)
LeqF	Sound levels due to source Si at point R in downward-refraction conditions, in each frequency band (calculated in this function)
LeqLT	Long-term sound levels due to source Si at point R, in each given frequency band (calculated in this function)

Returns:
the exception number (0 if no exception)
Definition at line 682 of file PropagationNMPB08.cpp.

bool NMPB08_ClearPath (PathID *path*)

clear the path profile

Parameters:

path	The path to clear

Returns:

true if all OK

Definition at line 214 of file PropagationNMPB08.cpp.

PathID NMPB08_CreatePath ()

Creates path calculator with default frequency range.

Returns:

the new Propagation Path

Definition at line 41 of file PropagationNMPB08.cpp.

PathID NMPB08_CreatePathEx (int *nbFreq*, double const **freq*)

Creates path calculator with user defined frequency range.

Parameters:

nbFreq	: frequencies number
freq	: frequencies values

Returns:

the new Propagation Path

Definition at line 74 of file PropagationNMPB08.cpp.

bool NMPB08_DeletePath (PathID *path*)

Delete the path calculator.

Parameters:

path	The path to delete

Returns:

true if all OK

Definition at line 186 of file PropagationNMPB08.cpp.

int NMPB08_DoCalculation (PathID *path*)

Do the propagation calculation.

Parameters:

path	The path to do propagation calculation

Returns:

the exception number (0 if no exception)

Definition at line 398 of file PropagationNMPB08.cpp.

bool NMPB08_ExtendPath (PathID *path*, Position3D const **point3D*, double *g*)

Add a segment to the path profile.

Parameters:

path	The path to add segment
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point

Returns:

true if all OK

Definition at line 247 of file PropagationNMPB08.cpp.

bool NMPB08_ExtendPathExt (PathID *path*, Position3D const **point3D*, double *g*, ExtensionNMPB const **ext*)

Add a segment to the path profile with extension data.

Parameters:

path	The path to add segment.
point3D	The 3D coordinates to add
g	The impedance value for the segment before the added point
ext	The extension data

Returns:

true if all OK

Definition at line 281 of file PropagationNMPB08.cpp.

double const* NMPB08_GetAttF (PathID *path*)

Get path attenuations under favorable conditions.

Parameters:

path	The path to get attenuations

Returns:

the favorable attenuation for each frequency range

Definition at line 519 of file PropagationNMPB08.cpp.

double const* NMPB08_GetAttH (PathID *path*)

Get path attenuations under homogeneous conditions.

Parameters:

path	The path to get attenuations

Returns:

the homogeneous attenuation for each frequency range

Definition at line 546 of file PropagationNMPB08.cpp.

double NMPB08_GetFavorableConditionProbability (Position3D const *source, Position3D const *receiver, int nbAngles, const double *fcpAngles, double angleNorth)

Calculates the favorable conditions probability for the (SR) direction.

Parameters:

source	The source position 3D
receiver	The receiver position 3D
nbAngles	The number of angle probabilities (should be 18)
fcpAngles	The favorable conditions probabilities for the angles 20, 40, 60, ..., 360
angleNorth	The north direction (Ox,ON)

Returns:

the favorable conditions probability for the (SR) direction

Definition at line 637 of file PropagationNMPB08.cpp.

double const* NMPB08_GetFrequencies (PathID *path*)

Gets the frequencies array.

Parameters:

path	The path containing frequencies

Returns:

the frequencies array

Definition at line 159 of file PropagationNMPB08.cpp.

int NMPB08_GetNbFrequencies (PathID *path*)

Gets the frequencies number.

Parameters:

path	The path containing frequencies

Returns:

the frequencies number

Definition at line 132 of file PropagationNMPB08.cpp.

double NMPB08_Leq_LT (double *soundLevel_h*, double *soundLevel_f*, double *p*)

Calculates long-term sound level for the given the sound levels (homogeneous and favorable)

Parameters:

soundLevel_h	: sound level in homogeneous conditions
soundLevel_f	: sound level in downward-refraction conditions
p	The average occurrence of downward-refraction conditions in the direction of the path

Returns:

the calculated sound level

Definition at line 577 of file PropagationNMPB08.cpp.

bool NMPB08_SetOption (PathID*path*, Option*option*, bool*on_off*)

Set an option for the path.

Parameters:

path	The path to set the option
option	The option to set
on_off	True if the option must be set to true

Returns:

true if all OK

Definition at line 104 of file PropagationNMPB08.cpp.

int NMPB08_SetReceiverHeight (PathID*path*, double*h*)

Set the receiver height.

Parameters:

path	The path to set receiver height
h	The receiver height

Returns:

the exception number (0 if no exception)

Definition at line 341 of file PropagationNMPB08.cpp.

int NMPB08_SetSourceHeight (PathID*path*, double*h*)

Set the source height.

Parameters:

path	The path to set source height
h	The source height

Returns:

the exception number (0 if no exception)

Definition at line 311 of file PropagationNMPB08.cpp.

double NMPB08_SumLevels (int*n*, double const **levels*)

Calculates sound levels sum.

Used for § 5.2.4 Formula (8) and § 5.2.5 Formula (9)

Parameters:

n	the sound levels number
levels	The sound levels to sum

Returns:

the sum

Definition at line 605 of file PropagationNMPB08.cpp.

SousCalculs/Diffraction.cpp File Reference

Calculation of diffraction attenuations.

```
#include "Diffraction.h"
#include <math.h>
#include <stdio.h>
#include <assert.h>
```

Namespaces

- namespace DiffractionNMPB

Functions

- double DiffractionNMPB::CurveRayLength (double distMN, double curvatureRadius)
  Calculates the circular ray length for MN.
- double DiffractionNMPB::PathDifference (Position2D const *source2D, Position2D const *receiver2D, vector< ProfilePointNMPB * > screenItems, bool favourableConditions)
  Calculation of the path difference for the screen elements.
- double DiffractionNMPB::PathDifference (Position2D const *source2D, Position2D const *receiver2D, ProfilePointNMPB *reflectionItem)
  Calculation of the path difference for a reflection element.
- double DiffractionNMPB::SidePathDifference (Position3D const *source3D, Position3D const *receiver3D, vector< ProfilePointNMPB * > screenItems, double &totalDiffDist)
  Calculation of the path difference for the side diffractions.

Detailed Description

Calculation of diffraction attenuations.

Author:

CSTB

Version:

1.0

Definition in file Diffraction.cpp.

SousCalculs/Diffraction.h File Reference

Calculation of diffraction attenuations.

```
#include "GroundEffect.h"
#include "Embankment.h"
```

Classes

- class DiffractionNMPB::Diffraction

Class used to calculate diffraction attenuation. Namespaces

- namespace DiffractionNMPB

Defines

- #define Diffraction _Local_PROPAN8_Diffraction_

Detailed Description

Calculation of diffraction attenuations.

Author:
CSTB

Version:
1.0

Definition in file Diffraction.h.

Define Documentation

#define Diffraction _Local_PROPAN8_Diffraction_

Definition at line 15 of file Diffraction.h.

SousCalculs/ElementaryPath.cpp File Reference

Elementary path determination (3D -> 2D, convex hull)

```
#include "ElementaryPath.h"
#include <assert.h>
```

Namespaces

- namespace ElementaryPathNMPB

Functions

- void ElementaryPathNMPB::ConvexHull (vector< ProfilePointNMPB * > &pathItems, int n1, int n2, int level)
  Finds the convex hull of the screenItems vector.
- void ElementaryPathNMPB::SetElementaryPath (PropagationPath *path)
  Sets the elementary path for the given path (calculates convex hull and plane positions)

Detailed Description

Elementary path determination (3D -> 2D, convex hull)

Author:
CSTB

Version:
1.0

Definition in file ElementaryPath.cpp.

SousCalculs/ElementaryPath.h File Reference

Elementary path determination (3D -> 2D, convex hull)

```
#include <math.h>
#include <stdio.h>
#include "../PathStructures.h"
```

Namespaces

- namespace ElementaryPathNMPB

Functions

- void ElementaryPathNMPB::ConvexHull (vector< ProfilePointNMPB * > &pathItems, int n1, int n2, int level)
  Finds the convex hull of the screenItems vector.
- void ElementaryPathNMPB::SetElementaryPath (PropagationPath *path)
  Sets the elementary path for the given path (calculates convex hull and plane positions)

Detailed Description

Elementary path determination (3D -> 2D, convex hull)

Author:
CSTB

Version:
1.0
Definition in file ElementaryPath.h.

SousCalculs/Embankment.cpp File Reference

Calculation embankment attenuation.
```
#include "Embankment.h"
#include <math.h>
#include <stdio.h>
```

Namespaces

- namespace EmbankmentNMPB

Detailed Description

Calculation embankment attenuation.

Author:
CSTB
Version:
1.0
Definition in file Embankment.cpp.

SousCalculs/Embankment.h File Reference

Calculation embankment attenuation.
```
#include "../PathStructures.h"
```

Classes

- class EmbankmentNMPB::Embankment

Class used to calculate the embankment attenuation. Namespaces

- namespace EmbankmentNMPB

Defines

- #define Embankment _Local_PROPAN8_Embankment_

Detailed Description

Calculation embankment attenuation.

Author:
CSTB

Version:
1.0

Definition in file Embankment.h.

Define Documentation

#define Embankment _Local_PROPAN8_Embankment_

Definition at line 14 of file Embankment.h.

SousCalculs/GroundEffect.cpp File Reference

Calculation of ground effect attenuations.

```
#include "GroundEffect.h"
#include <math.h>
#include <stdio.h>
```

Namespaces

- namespace GroundEffectNMPB

Detailed Description

Calculation of ground effect attenuations.

Author:
CSTB

Version:
1.0

Definition in file GroundEffect.cpp.

SousCalculs/GroundEffect.h File Reference

Calculation of ground effect attenuations.

```
#include "../PathStructures.h"
```

Classes

- class GroundEffectNMPB::MeanPlane
- *Class used to calculate the mean plane.* class GroundEffectNMPB::GroundEffect

Class used to calculate ground effects. Namespaces

- namespace GroundEffectNMPB

Defines

- #define MeanPlane _Local_PROPAN8_MeanPlane_
- #define GroundEffect _Local_PROPAN8_GroundEffect_

Enumerations

- enum GroundEffectNMPB::GroundCalculationType { GroundEffectNMPB::Asol = 1, GroundEffectNMPB::DeltaSol_SO = 2, GroundEffectNMPB::DeltaSol_OR = 3 }

Ground Calculation Type used to know when using _Gpath or _correctedGpath in the ground attenuation calculation.

Detailed Description

Calculation of ground effect attenuations.

Author:
CSTB

Version:
1.0

Definition in file GroundEffect.h.

Define Documentation

#define GroundEffect _Local_PROPAN8_GroundEffect_

Definition at line 15 of file GroundEffect.h.

#define MeanPlane _Local_PROPAN8_MeanPlane_

Definition at line 14 of file GroundEffect.h.

Index

INDEX