Tracker

Generated by Doxygen 1.8.11

Contents

1 Main Page

Tracker

Tracker is a particle tracking trajectory connector tool that is based on a sparse-matrix `linear assignment problem (LAP)` solver.

- Tracker uses a sparse matrix formulations of the `Jonker-Volgenant Algorithm`
- Tracker provides C++ and Matlab object-oriented interfaces. `tracker::LAPTrack`
- Tracker is designed for cross-platform complilation to Linux and Windows 64-bit targets.

Trajectory connection problem

In single particle tracking applications, a set of likely particles are localized for each frame of a video capture. The goal of trajectory connection is to partition the localizations from all the frames into a set of trajectories. Each trajectory is a sequence of localizations which are likely to be from the same object (point emitter).

The Tracker library implements a two-phase strategy to trajectory connection. First a frame-to-frame algorithm sequentially builds a set of trajectories connecting localizations in adjacent frames, next a gap-closing phase connects shorter trajectories across several frames as particles are often not localized in every frame for various reasons including experimental and photo-chemical effects.

Figure 1: The frame-to-frame trajectory connection problem

Documentation

The Tracker Doxygen documentation can be build with the `OPT_DOC` CMake option and is also available on online:

- `Tracker HTML Manual`
- `Tracker PDF Manual`
- `Tracker github repository`

Dependencies

- *`Armadillo`* - A high-performance array library for C++.

External Projects

These packages are specialized CMake projects. If they are not currently installed, at the start of the build process, the `AddExternalDependency.cmake` will automatically download, configure, build and install CMake-based projects to the `CMAKE_INSTALL_PREFIX`. This process is completed before CMake configure-time so calls to the normal `find_package()` command are used to find the auto-added Dependencies.

- `BacktraceException` - A library to provide debugging output on exception calls. Important for Matlab debugging.
- `MexIFace` - A C++/Matlab object oriented interface library for high-performance numerical computations. Provides cross-compilation to Matlab R2016b+ target environments on Linux and Windows 64-bit targets.

2 Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

3 Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

4 Class Index

4.1 Class List

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

5 File Index

5.1 File List

Here is a list of all files with brief descriptions:

6 Namespace Documentation

6.1 tracker Namespace Reference

Classes

- class LAP_JVSparse
- class LAPTrack
- struct LogicalError

  Parameter value is not valid.
- struct ParameterValueError

  Parameter value is not valid.
- class Tracker

Typedefs

- using TrackerError = backtrace_exception::BacktraceException

6.1.1 Typedef Documentation

6.1.1.1 using tracker::TrackerError = typedef backtrace_exception::BacktraceException

Definition at line 28 of file Tracker.h.

7 Class Documentation

7.1 tracker::LAP_JVSparse< FloatT > Class Template Reference

```
#include </home/travis/build/markjolah/Tracker/include/Tracker/LAP_JVSparse.h>
```

Static Public Member Functions

- static IVecT solve (const SpMatT &C)
- static void solveLAP_orig (const SpMatT &C, IVecT &x, IVecT &y, VecT &u, VecT &v)
- static VecT computeCost (const SpMatT &C, const IVecT &row_sol)
- static bool checkCosts (const SpMatT &C)
- static bool checkSolution (const SpMatT &C, const IVecT &x, const IVecT &y, const VecT &u, const VecT &v)

7.1.1 Detailed Description

template<class FloatT>
class tracker::LAP_JVSparse< FloatT >

Definition at line 21 of file LAP_JVSparse.h.

7.1.2 Member Function Documentation

7.1.2.1 template<class FloatT > bool tracker::LAP_JVSparse< FloatT >::checkCosts (const SpMatT & *C*) `[static]`

Definition at line 95 of file LAP_JVSparse.cpp.

7.1.2.2 template<class FloatT > bool tracker::LAP_JVSparse< FloatT >::checkSolution (const SpMatT & *C*, const IVecT & *x*, const IVecT & *y*, const VecT & *u*, const VecT & *v*) `[static]`

Definition at line 118 of file LAP_JVSparse.cpp.

7.1.2.3 template<class FloatT > LAP_JVSparse< FloatT >::VecT tracker::LAP_JVSparse< FloatT >::computeCost (const SpMatT & *C*, const IVecT & *row_sol*) `[static]`

Compute the total cost of a solution

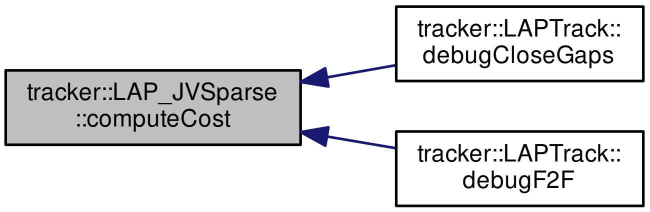
Parameters

`in`	*row_sol*	This is the 'x' output from the solver giving the col assignment for each row in order

Definition at line 85 of file LAP_JVSparse.cpp.

Referenced by tracker::LAPTrack::debugCloseGaps(), and tracker::LAPTrack::debugF2F().

Here is the caller graph for this function:

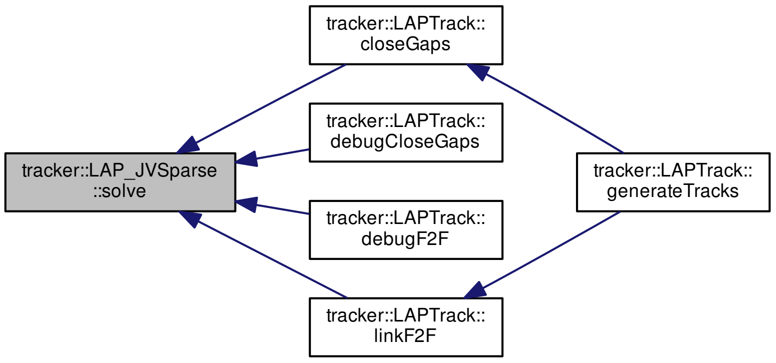

tracker::LAP_JVSparse ::computeCost
tracker::LAPTrack:: debugCloseGaps
tracker::LAPTrack:: debugF2F

7.1.2.4 template< class FloatT > LAP_JVSparse< FloatT >::IVecT tracker::LAP_JVSparse< FloatT >::solve (const SpMatT & *C*) [static]

Definition at line 21 of file LAP_JVSparse.cpp.

Referenced by tracker::LAPTrack::closeGaps(), tracker::LAPTrack::debugCloseGaps(), tracker::LAPTrack::debugF2F(), and tracker::LAPTrack::linkF2F().

Here is the caller graph for this function:


tracker::LAP_JVSparse ::solve
tracker::LAPTrack:: closeGaps
tracker::LAPTrack:: debugCloseGaps
tracker::LAPTrack:: debugF2F
tracker::LAPTrack:: linkF2F
tracker::LAPTrack:: generateTracks

7.1.2.5 template< class FloatT > void tracker::LAP_JVSparse< FloatT >::solveLAP_orig (const SpMatT & *C*, IVecT & *x*, IVecT & *y*, VecT & *u*, VecT & *v*) [static]

This wraps the original sparse lap implementation that for some reason uses 1-based indexing, which we correct with some pointer arrithmetic and adjusting of appropriate indicies in the sparse matrix implementation.

Furthermore because the lap_orig code assumes a compressed-row format, but we pass it the internal datastore of a compressed-col format sparse metrix. We invert x/y and u/v on the call to lap_orig to effectively let the transformation work easily with the legacy code.

This means x is the row sol and y is the col sol, as it normally would be.

Parameters

`in`	*C*	costs sparse matrix
`out`	*x*	- row assignments
`out`	*y*	- col assignments
`out`	*u*	- reduced row costs
`out`	*v*	- reduced column costs

Definition at line 50 of file LAP_JVSparse.cpp.

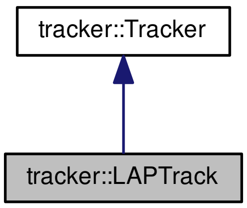
The documentation for this class was generated from the following files:

- LAP_JVSparse.h
- LAP_JVSparse.cpp

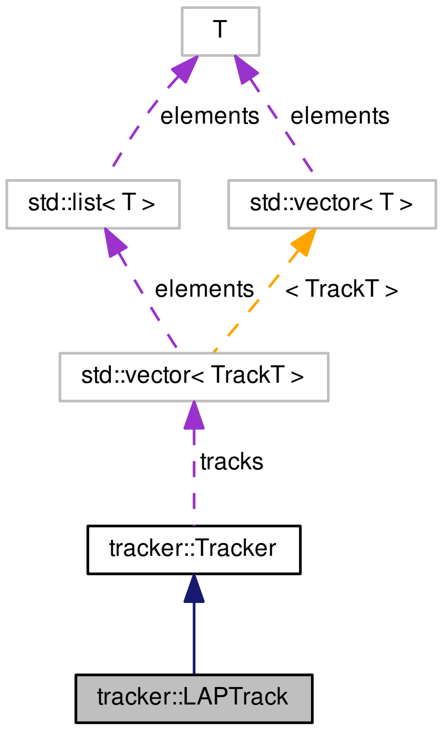
7.2 tracker::LAPTrack Class Reference

```
#include </home/travis/build/markjolah/Tracker/include/Tracker/LAPTrack.h>
```

Inheritance diagram for tracker::LAPTrack:



Collaboration diagram for tracker::LAPTrack:



Public Types

- using SpMatT = arma::SpMat< FloatT >
- using UVecT = arma::Col< arma::uword >
- using UMatT = arma::umat
- using FloatT = double
- using IdxT = int32_t
- using VecT = arma::Col< FloatT >
- using MatT = arma::Mat< FloatT >
- using IVecT = arma::Col< IdxT >
- using IMatT = arma::Mat< IdxT >
- using IVecFieldT = arma::field< IVecT >
- using IndexVectorT = std::vector< IdxT >
- using TrackT = std::list< IdxT >
- using TrackVecT = std::vector< TrackT >
- using ParamT = std::map< std::string, FloatT >
- using VecParamT = std::map< std::string, VecT >

Public Member Functions

- LAPTrack (const VecParamT &param)
- VecParamT getStats () const
- void initializeTracks (const IVecT &frameIdx_, const MatT &position_, const MatT &SE_position_)
- void initializeTracks (const IVecT &frameIdx_, const MatT &position_, const MatT &SE_position_, const MatT &feature_, const MatT &SE_feature_)
- void linkF2F ()
- void closeGaps ()
- SpMatT computeF2FCostMat (int curFrame, int nextFrame) const
- void debugF2F (int frameIdx, IVecT &cur_locs, IVecT &next_locs, SpMatT &cost, IMatT &connections, VecT &conn_costs) const
- void debugCloseGaps (SpMatT &cost, IMatT &connections, VecT &conn_costs) const
- SpMatT computeGapCloseMatrix () const
- void generateTracks ()
- void checkFrameIdxs ()
- void printTracks () const

Public Attributes

- FloatT D
- FloatT kon
- FloatT koff
- FloatT rho
- VecT featureVar
- FloatT maxSpeed = 0
- FloatT maxPositionDisplacementSigma = 5.0
- VecT maxFeatureDisplacementSigma
- IdxT maxGapCloseFrames = 20
- IdxT minGapCloseTrackLength = 1
- IdxT minFinalTrackLength = 1
- const FloatT cost_epsilon = std::numeric_limits<FloatT>::epsilon()
- IdxT N = 0
- IdxT nDims = 0
- IdxT nFeatures = 0
- IVecT frameIdx
- MatT position
- MatT SE_position
- MatT feature
- MatT SE_feature
- IdxT firstFrame = 0
- IdxT lastFrame = 0
- IdxT nFrames = 0
- IVecT nFrameLocs
- IVecFieldT frameLocIdx
- TrackVecT tracks

Protected Types

- enum StateT { UNTRACKED, F2F_LINKED, GAPS_CLOSED }

Protected Attributes

- FloatT minCost = 1e-6
- FloatT log1mkoff
- FloatT log1mkon
- FloatT logrho
- FloatT logkon
- FloatT logkoff
- StateT state
- IndexVectorT birthFrameIdx
- IVecT frameBirthStartIdx
- IVecT trackAssignment

Static Protected Attributes

- static const FloatT log2pi = log(2∗arma::Datum<Tracker::FloatT>::pi)

7.2.1 Detailed Description

Definition at line 16 of file LAPTrack.h.

7.2.2 Member Typedef Documentation

7.2.2.1 using tracker::Tracker::FloatT = double `[inherited]`

Definition at line 47 of file Tracker.h.

7.2.2.2 using tracker::Tracker::IdxT = int32_t `[inherited]`

Definition at line 48 of file Tracker.h.

7.2.2.3 using tracker::Tracker::IMatT = arma::Mat<IdxT> `[inherited]`

Definition at line 52 of file Tracker.h.

7.2.2.4 using tracker::Tracker::IndexVectorT = std::vector<IdxT> `[inherited]`

Definition at line 54 of file Tracker.h.

7.2.2.5 using tracker::Tracker::IVecFieldT = arma::field<IVecT> `[inherited]`

Definition at line 53 of file Tracker.h.

7.2.2.6 using **tracker::Tracker::IVecT** = arma::Col< **IdxT** > `[inherited]`

Definition at line 51 of file Tracker.h.

7.2.2.7 using **tracker::Tracker::MatT** = arma::Mat< **FloatT** > `[inherited]`

Definition at line 50 of file Tracker.h.

7.2.2.8 using **tracker::Tracker::ParamT** = std::map< std::string, **FloatT** > `[inherited]`

A convenient form for reporting dictionaries of named FP data to matlab

Definition at line 57 of file Tracker.h.

7.2.2.9 using **tracker::LAPTrack::SpMatT** = arma::SpMat< **FloatT** >

Definition at line 18 of file LAPTrack.h.

7.2.2.10 using **tracker::Tracker::TrackT** = std::list< **IdxT** > `[inherited]`

A type for an individual track

Definition at line 55 of file Tracker.h.

7.2.2.11 using **tracker::Tracker::TrackVecT** = std::vector< **TrackT** > `[inherited]`

A type for a vector of tracks

Definition at line 56 of file Tracker.h.

7.2.2.12 using **tracker::LAPTrack::UMatT** = arma::umat

Definition at line 20 of file LAPTrack.h.

7.2.2.13 using **tracker::LAPTrack::UVecT** = arma::Col< arma::uword >

Definition at line 19 of file LAPTrack.h.

7.2.2.14 using **tracker::Tracker::VecParamT** = std::map< std::string, **VecT** > `[inherited]`

A convenient form for reporting dictionaries of named FP data to matlab

Definition at line 58 of file Tracker.h.

7.2.2.15 using **tracker::Tracker::VecT** = arma::Col< **FloatT** > `[inherited]`

Definition at line 49 of file Tracker.h.

7.2.3 Member Enumeration Documentation

7.2.3.1 enum tracker::LAPTrack::StateT [protected]

Enumerator

UNTRACKED

F2F_LINKED

GAPS_CLOSED

Definition at line 58 of file LAPTrack.h.

7.2.4 Constructor & Destructor Documentation

7.2.4.1 tracker::LAPTrack::LAPTrack (const VecParamT & *param*)

Definition at line 11 of file LAPTrack.cpp.

References D, featureVar, koff, kon, log1mkoff, log1mkon, logkoff, logkon, logrho, maxFeatureDisplacementSigma, maxGapCloseFrames, maxPositionDisplacementSigma, maxSpeed, minFinalTrackLength, minGapCloseTrackLength, and rho.

7.2.5 Member Function Documentation

7.2.5.1 void tracker::LAPTrack::checkFrameIdxs ( )

Definition at line 335 of file LAPTrack.cpp.

References F2F_LINKED, tracker::Tracker::firstFrame, frameBirthStartIdx, tracker::Tracker::frameIdx, tracker::Tracker↩
::lastFrame, state, and tracker::Tracker::tracks.

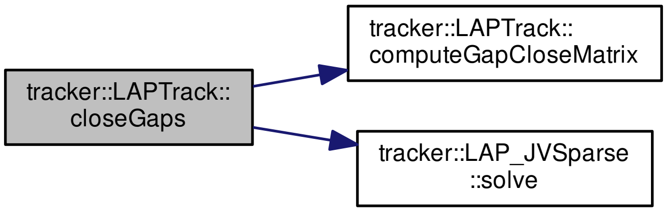
7.2.5.2 void tracker::LAPTrack::closeGaps ( )

Definition at line 376 of file LAPTrack.cpp.

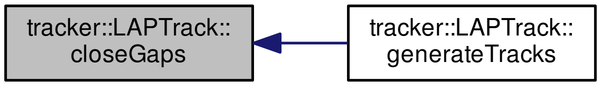
References birthFrameIdx, computeGapCloseMatrix(), F2F_LINKED, frameBirthStartIdx, GAPS_CLOSED, minFinal↩
TrackLength, tracker::LAP_JVSparse$<$ FloatT $>$::solve(), state, tracker::Tracker::trackAssignment, and tracker::↩
Tracker::tracks.

Referenced by generateTracks().

Here is the call graph for this function:



Here is the caller graph for this function:



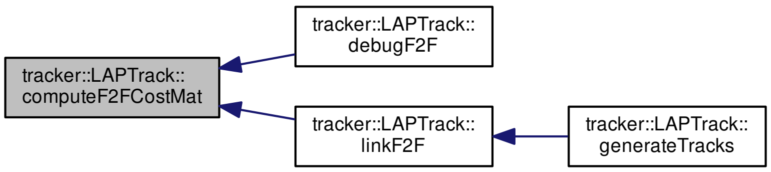
7.2.5.3 LAPTrack::SpMatT tracker::LAPTrack::computeF2FCostMat (int *curFrame*, int *nextFrame*) const

Definition at line 231 of file LAPTrack.cpp.

References cost_epsilon, D, tracker::Tracker::feature, featureVar, tracker::Tracker::firstFrame, tracker::Tracker::frame↩ LocIdx, log1mkoff, tracker::Tracker::log2pi, logkoff, logkon, logrho, maxFeatureDisplacementSigma, maxPosition↩ DisplacementSigma, maxSpeed, tracker::Tracker::nDims, tracker::Tracker::nFeatures, tracker::Tracker::nFrameLocs, tracker::Tracker::position, tracker::Tracker::SE_feature, and tracker::Tracker::SE_position.

Referenced by debugF2F(), and linkF2F().

Here is the caller graph for this function:



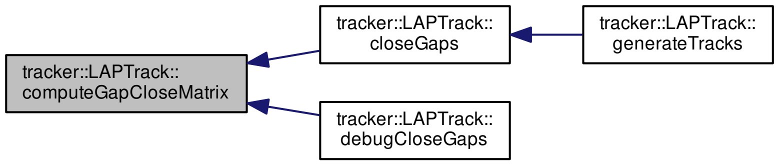
7.2.5.4 LAPTrack::SpMatT tracker::LAPTrack::computeGapCloseMatrix ( ) const

Definition at line 415 of file LAPTrack.cpp.

References birthFrameIdx, cost_epsilon, D, tracker::Tracker::feature, featureVar, tracker::Tracker::firstFrame, frame↩ BirthStartIdx, tracker::Tracker::frameIdx, tracker::Tracker::lastFrame, tracker::Tracker::log2pi, logkoff, logkon, logrho, maxFeatureDisplacementSigma, maxGapCloseFrames, maxPositionDisplacementSigma, maxSpeed, minGapClose↩ TrackLength, tracker::Tracker::nDims, tracker::Tracker::nFeatures, tracker::Tracker::position, tracker::Tracker::SE_↩ feature, tracker::Tracker::SE_position, and tracker::Tracker::tracks.

Referenced by closeGaps(), and debugCloseGaps().

Here is the caller graph for this function:

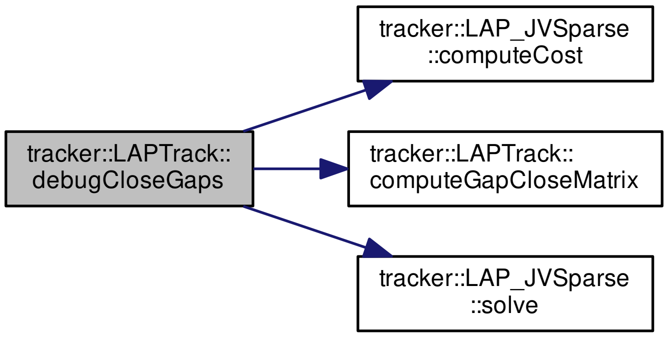


7.2.5.5 void tracker::LAPTrack::debugCloseGaps (SpMatT & *cost,* IMatT & *connections,* VecT & *conn_costs*) const

Definition at line 351 of file LAPTrack.cpp.

References tracker::LAP_JVSparse< FloatT >::computeCost(), computeGapCloseMatrix(), cost_epsilon, F2F_LINK↩ ED, tracker::LAP_JVSparse< FloatT >::solve(), state, and tracker::Tracker::tracks.

Here is the call graph for this function:

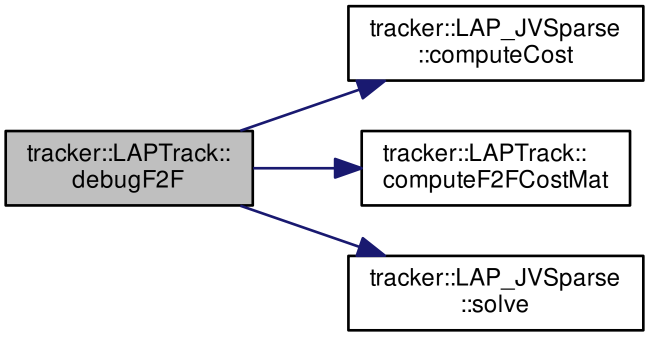


7.2.5.6 void tracker::LAPTrack::debugF2F (int *frameIdx*, IVecT & *cur_locs*, IVecT & *next_locs*, SpMatT & *cost*, IMatT & *connections*, VecT & *conn_costs*) const

Definition at line 92 of file LAPTrack.cpp.

References tracker::LAP_JVSparse< FloatT >::computeCost(), computeF2FCostMat(), cost_epsilon, tracker↩ ::Tracker::firstFrame, tracker::Tracker::frameLocIdx, tracker::Tracker::lastFrame, tracker::Tracker::nFrameLocs, and tracker::LAP_JVSparse< FloatT >::solve().

Here is the call graph for this function:



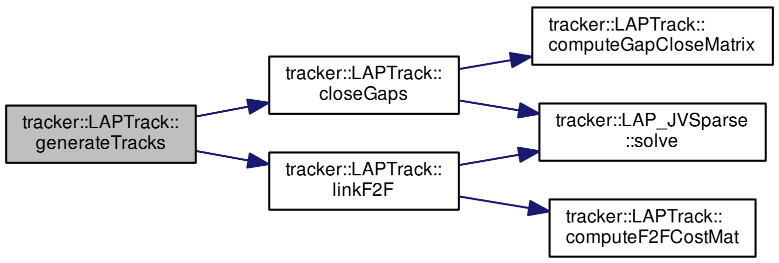
7.2.5.7 void tracker::LAPTrack::generateTracks ( ) [virtual]

Implements tracker::Tracker.

Definition at line 77 of file LAPTrack.cpp.

References closeGaps(), F2F_LINKED, GAPS_CLOSED, linkF2F(), state, and UNTRACKED.

Here is the call graph for this function:



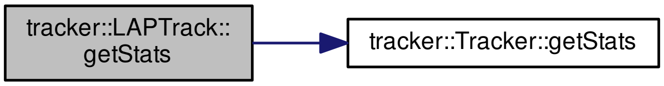
7.2.5.8 LAPTrack::VecParamT tracker::LAPTrack::getStats ( ) const [virtual]

Reimplemented from tracker::Tracker.

Definition at line 46 of file LAPTrack.cpp.

References D, featureVar, tracker::Tracker::getStats(), koff, kon, maxFeatureDisplacementSigma, maxGapCloseFrames, maxPositionDisplacementSigma, maxSpeed, minFinalTrackLength, minGapCloseTrackLength, and rho.

Here is the call graph for this function:



7.2.5.9 void tracker::LAPTrack::initializeTracks (const IVecT & *frameIdx_*, const MatT & *position_*, const MatT & *SE_position_*) [virtual]

Reimplemented from tracker::Tracker.

Definition at line 63 of file LAPTrack.cpp.

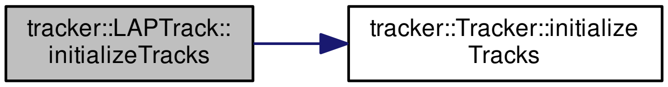
7.2.5.10 void tracker::LAPTrack::initializeTracks (const IVecT & *frameIdx_*, const MatT & *position_*, const MatT & *SE_position_*, const MatT & *feature_*, const MatT & *SE_feature_*) [virtual]

Reimplemented from tracker::Tracker.

Definition at line 69 of file LAPTrack.cpp.

References birthFrameIdx, frameBirthStartIdx, tracker::Tracker::initializeTracks(), state, and UNTRACKED.

Here is the call graph for this function:



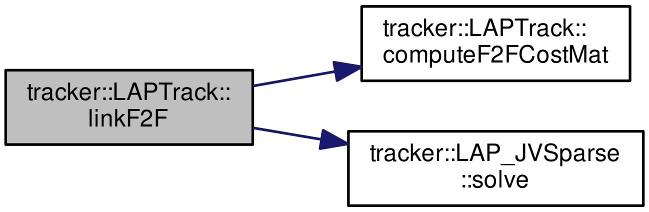
7.2.5.11 void tracker::LAPTrack::linkF2F ( )

Definition at line 130 of file LAPTrack.cpp.

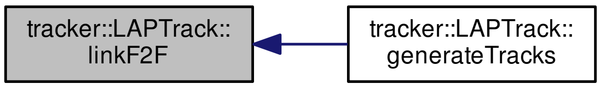
References birthFrameIdx, computeF2FCostMat(), F2F_LINKED, tracker::Tracker::firstFrame, frameBirthStartIdx, tracker::Tracker::frameLocIdx, tracker::Tracker::lastFrame, tracker::Tracker::nFrameLocs, tracker::Tracker::nFrames, tracker::LAP_JVSparse< FloatT >::solve(), state, tracker::Tracker::trackAssignment, tracker::Tracker::tracks, and UN↩TRACKED.

Referenced by generateTracks().

Here is the call graph for this function:



Here is the caller graph for this function:



7.2.5.12 void tracker::Tracker::printTracks ( ) const [inherited]

Definition at line 126 of file Tracker.cpp.

References tracker::Tracker::frameIdx, and tracker::Tracker::tracks.

7.2.6 Member Data Documentation

7.2.6.1 IndexVectorT tracker::LAPTrack::birthFrameIdx [protected]

Definition at line 62 of file LAPTrack.h.

Referenced by closeGaps(), computeGapCloseMatrix(), initializeTracks(), and linkF2F().

7.2.6.2 const FloatT tracker::LAPTrack::cost_epsilon = std::numeric_limits<FloatT>::epsilon()

Definition at line 35 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), debugCloseGaps(), and debugF2F().

7.2.6.3 FloatT tracker::LAPTrack::D

Definition at line 22 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.4 MatT tracker::Tracker::feature [inherited]

Definition at line 66 of file Tracker.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and tracker::Tracker::initializeTracks().

7.2.6.5 VecT tracker::LAPTrack::featureVar

Definition at line 26 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.6 IdxT tracker::Tracker::firstFrame = 0 [inherited]

Definition at line 68 of file Tracker.h.

Referenced by checkFrameIdxs(), computeF2FCostMat(), computeGapCloseMatrix(), debugF2F(), tracker::Tracker↩
::getStats(), tracker::Tracker::initializeTracks(), and linkF2F().

7.2.6.7 IVecT tracker::LAPTrack::frameBirthStartIdx [protected]

Definition at line 63 of file LAPTrack.h.

Referenced by checkFrameIdxs(), closeGaps(), computeGapCloseMatrix(), initializeTracks(), and linkF2F().

7.2.6.8 IVecT tracker::Tracker::frameIdx [inherited]

Definition at line 63 of file Tracker.h.

Referenced by checkFrameIdxs(), computeGapCloseMatrix(), tracker::Tracker::initializeTracks(), and tracker::Tracker↩
::printTracks().

7.2.6.9 IVecFieldT tracker::Tracker::frameLocIdx [inherited]

Definition at line 74 of file Tracker.h.

Referenced by computeF2FCostMat(), debugF2F(), tracker::Tracker::initializeTracks(), and linkF2F().

7.2.6.10 FloatT tracker::LAPTrack::koff

Definition at line 24 of file LAPTrack.h.

Referenced by getStats(), and LAPTrack().

7.2.6.11 FloatT tracker::LAPTrack::kon

Definition at line 23 of file LAPTrack.h.

Referenced by getStats(), and LAPTrack().

7.2.6.12 IdxT tracker::Tracker::lastFrame = 0 `[inherited]`

Definition at line 69 of file Tracker.h.

Referenced by checkFrameIdxs(), computeGapCloseMatrix(), debugF2F(), tracker::Tracker::getStats(), tracker::↩ Tracker::initializeTracks(), and linkF2F().

7.2.6.13 FloatT tracker::LAPTrack::log1mkoff `[protected]`

Definition at line 52 of file LAPTrack.h.

Referenced by computeF2FCostMat(), and LAPTrack().

7.2.6.14 FloatT tracker::LAPTrack::log1mkon `[protected]`

Definition at line 53 of file LAPTrack.h.

Referenced by LAPTrack().

7.2.6.15 const Tracker::FloatT tracker::Tracker::log2pi = log(2∗arma::Datum<Tracker::FloatT>::pi) `[static]`, `[protected]`, `[inherited]`

Definition at line 92 of file Tracker.h.

Referenced by computeF2FCostMat(), and computeGapCloseMatrix().

7.2.6.16 FloatT tracker::LAPTrack::logkoff `[protected]`

Definition at line 56 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and LAPTrack().

7.2.6.17 FloatT tracker::LAPTrack::logkon `[protected]`

Definition at line 55 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and LAPTrack().

7.2.6.18 FloatT tracker::LAPTrack::logrho [protected]

Definition at line 54 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and LAPTrack().

7.2.6.19 VecT tracker::LAPTrack::maxFeatureDisplacementSigma

Definition at line 29 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.20 IdxT tracker::LAPTrack::maxGapCloseFrames = 20

Definition at line 30 of file LAPTrack.h.

Referenced by computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.21 FloatT tracker::LAPTrack::maxPositionDisplacementSigma = 5.0

Definition at line 28 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.22 FloatT tracker::LAPTrack::maxSpeed = 0

Definition at line 27 of file LAPTrack.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.23 FloatT tracker::LAPTrack::minCost = 1e-6 [protected]

Definition at line 51 of file LAPTrack.h.

7.2.6.24 IdxT tracker::LAPTrack::minFinalTrackLength = 1

Definition at line 32 of file LAPTrack.h.

Referenced by closeGaps(), getStats(), and LAPTrack().

7.2.6.25 IdxT tracker::LAPTrack::minGapCloseTrackLength = 1

Definition at line 31 of file LAPTrack.h.

Referenced by computeGapCloseMatrix(), getStats(), and LAPTrack().

7.2.6.26 IdxT tracker::Tracker::N = 0 [inherited]

Definition at line 60 of file Tracker.h.

Referenced by tracker::Tracker::getStats(), and tracker::Tracker::initializeTracks().

7.2.6.27 IdxT tracker::Tracker::nDims = 0 [inherited]

Definition at line 61 of file Tracker.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), tracker::Tracker::getStats(), and tracker::Tracker↩ ::initializeTracks().

7.2.6.28 IdxT tracker::Tracker::nFeatures = 0 [inherited]

Definition at line 62 of file Tracker.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), tracker::Tracker::getStats(), and tracker::Tracker↩ ::initializeTracks().

7.2.6.29 IVecT tracker::Tracker::nFrameLocs [inherited]

Definition at line 73 of file Tracker.h.

Referenced by computeF2FCostMat(), debugF2F(), tracker::Tracker::initializeTracks(), and linkF2F().

7.2.6.30 IdxT tracker::Tracker::nFrames = 0 [inherited]

Definition at line 70 of file Tracker.h.

Referenced by tracker::Tracker::getStats(), tracker::Tracker::initializeTracks(), and linkF2F().

7.2.6.31 MatT tracker::Tracker::position [inherited]

Definition at line 64 of file Tracker.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and tracker::Tracker::initializeTracks().

7.2.6.32 FloatT tracker::LAPTrack::rho

Definition at line 25 of file LAPTrack.h.

Referenced by getStats(), and LAPTrack().

7.2.6.33 MatT tracker::Tracker::SE_feature [inherited]

Definition at line 67 of file Tracker.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and tracker::Tracker::initializeTracks().

7.2.6.34 MatT tracker::Tracker::SE_position [inherited]

Definition at line 65 of file Tracker.h.

Referenced by computeF2FCostMat(), computeGapCloseMatrix(), and tracker::Tracker::initializeTracks().

7.2.6.35 StateT tracker::LAPTrack::state [protected]

Definition at line 59 of file LAPTrack.h.

Referenced by checkFrameIdxs(), closeGaps(), debugCloseGaps(), generateTracks(), initializeTracks(), and linkF2F().

7.2.6.36 IVecT tracker::Tracker::trackAssignment [protected], [inherited]

Definition at line 93 of file Tracker.h.

Referenced by closeGaps(), tracker::Tracker::getStats(), tracker::Tracker::initializeTracks(), and linkF2F().

7.2.6.37 TrackVecT tracker::Tracker::tracks [inherited]

Definition at line 77 of file Tracker.h.

Referenced by checkFrameIdxs(), closeGaps(), computeGapCloseMatrix(), debugCloseGaps(), tracker::Tracker::get←
Stats(), tracker::Tracker::initializeTracks(), linkF2F(), and tracker::Tracker::printTracks().

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

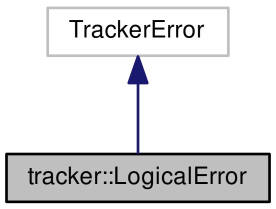
- LAPTrack.h
- LAPTrack.cpp

7.3 tracker::LogicalError Struct Reference

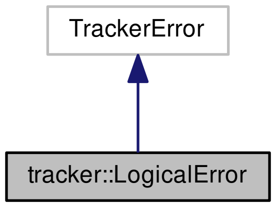
Parameter value is not valid.

`#include </home/travis/build/markjolah/Tracker/include/Tracker/Tracker.h>`

Inheritance diagram for tracker::LogicalError:



Collaboration diagram for tracker::LogicalError:



Public Member Functions

- LogicalError (std::string message)

7.3.1 Detailed Description

Parameter value is not valid.

Definition at line 40 of file Tracker.h.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 tracker::LogicalError::LogicalError (std::string *message*) `[inline]`

Definition at line 42 of file Tracker.h.

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

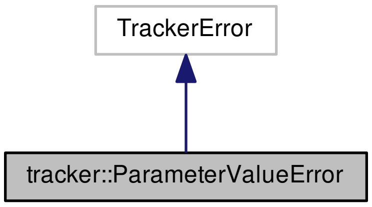
- Tracker.h

7.4 tracker::ParameterValueError Struct Reference

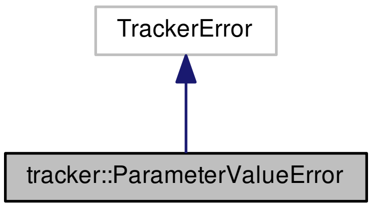
Parameter value is not valid.

`#include </home/travis/build/markjolah/Tracker/include/Tracker/Tracker.h>`

Inheritance diagram for tracker::ParameterValueError:



Collaboration diagram for tracker::ParameterValueError:



Public Member Functions

- ParameterValueError (std::string message)

7.4.1 Detailed Description

Parameter value is not valid.

Definition at line 33 of file Tracker.h.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 tracker::ParameterValueError::ParameterValueError (std::string *message*) `[inline]`

Definition at line 35 of file Tracker.h.

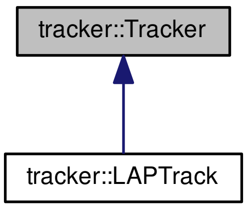
The documentation for this struct was generated from the following file:

- Tracker.h

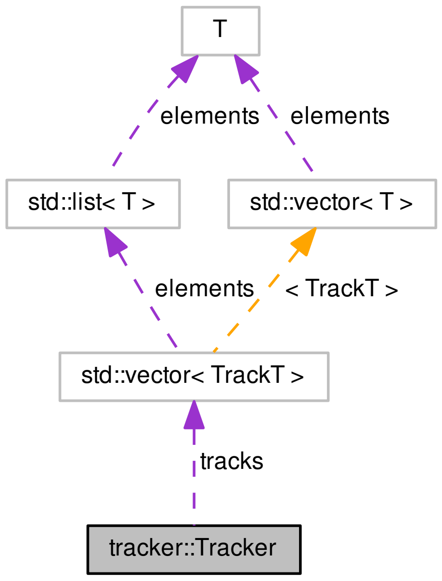
7.5 tracker::Tracker Class Reference

```
#include </home/travis/build/markjolah/Tracker/include/Tracker/Tracker.h>
```

Inheritance diagram for tracker::Tracker:



Collaboration diagram for tracker::Tracker:



Public Types

- using FloatT = double
- using IdxT = int32_t
- using VecT = arma::Col< FloatT >
- using MatT = arma::Mat< FloatT >
- using IVecT = arma::Col< IdxT >
- using IMatT = arma::Mat< IdxT >
- using IVecFieldT = arma::field< IVecT >
- using IndexVectorT = std::vector< IdxT >
- using TrackT = std::list< IdxT >
- using TrackVecT = std::vector< TrackT >
- using ParamT = std::map< std::string, FloatT >
- using VecParamT = std::map< std::string, VecT >

Public Member Functions

- Tracker (const VecParamT &param)
- virtual ∼Tracker ()
- virtual VecParamT getStats () const
- virtual void initializeTracks (const IVecT &frameIdx_, const MatT &position_, const MatT &SE_position_)
- virtual void initializeTracks (const IVecT &frameIdx_, const MatT &position_, const MatT &SE_position_, const MatT &feature_, const MatT &SE_feature_)
- virtual void generateTracks ()=0
- void printTracks () const

Public Attributes

- IdxT N = 0
- IdxT nDims = 0
- IdxT nFeatures = 0
- IVecT frameIdx
- MatT position
- MatT SE_position
- MatT feature
- MatT SE_feature
- IdxT firstFrame = 0
- IdxT lastFrame = 0
- IdxT nFrames = 0
- IVecT nFrameLocs
- IVecFieldT frameLocIdx
- TrackVecT tracks

Protected Attributes

- IVecT trackAssignment

Static Protected Attributes

- static const FloatT log2pi = log(2∗arma::Datum<Tracker::FloatT>::pi)

7.5.1 Detailed Description

Definition at line 45 of file Tracker.h.

7.5.2 Member Typedef Documentation

7.5.2.1 using **tracker::Tracker::FloatT** = double

Definition at line 47 of file Tracker.h.

7.5.2.2 using **tracker::Tracker::IdxT** = int32_t

Definition at line 48 of file Tracker.h.

7.5.2.3 using **tracker::Tracker::IMatT** = arma::Mat<**IdxT**>

Definition at line 52 of file Tracker.h.

7.5.2.4 using tracker::Tracker::IndexVectorT = std::vector<IdxT>

Definition at line 54 of file Tracker.h.

7.5.2.5 using tracker::Tracker::IVecFieldT = arma::field<IVecT>

Definition at line 53 of file Tracker.h.

7.5.2.6 using tracker::Tracker::IVecT = arma::Col<IdxT>

Definition at line 51 of file Tracker.h.

7.5.2.7 using tracker::Tracker::MatT = arma::Mat<FloatT>

Definition at line 50 of file Tracker.h.

7.5.2.8 using tracker::Tracker::ParamT = std::map<std::string,FloatT>

A convenient form for reporting dictionaries of named FP data to matlab

Definition at line 57 of file Tracker.h.

7.5.2.9 using tracker::Tracker::TrackT = std::list<IdxT>

A type for an individual track

Definition at line 55 of file Tracker.h.

7.5.2.10 using tracker::Tracker::TrackVecT = std::vector<TrackT>

A type for a vector of tracks

Definition at line 56 of file Tracker.h.

7.5.2.11 using tracker::Tracker::VecParamT = std::map<std::string,VecT>

A convenient form for reporting dictionaries of named FP data to matlab

Definition at line 58 of file Tracker.h.

7.5.2.12 using tracker::Tracker::VecT = arma::Col<FloatT>

Definition at line 49 of file Tracker.h.

7.5.3 Constructor & Destructor Documentation

7.5.3.1 tracker::Tracker::Tracker (const VecParamT & *param*)

param - A dictionary of floating point values to pass in. This is a flexible interface to the higher-level matlab code allowing each subclass to take in arbitrary floating point arguments.

Definition at line 15 of file Tracker.cpp.

7.5.3.2 virtual tracker::Tracker::∼Tracker ( ) `[inline]`, `[virtual]`

Definition at line 84 of file Tracker.h.

7.5.4 Member Function Documentation

7.5.4.1 virtual void tracker::Tracker::generateTracks ( ) `[pure virtual]`

Implemented in tracker::LAPTrack.

7.5.4.2 Tracker::VecParamT tracker::Tracker::getStats ( ) const `[virtual]`

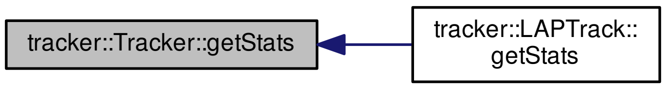
Reimplemented in tracker::LAPTrack.

Definition at line 19 of file Tracker.cpp.

References firstFrame, lastFrame, N, nDims, nFeatures, nFrames, trackAssignment, and tracks.

Referenced by tracker::LAPTrack::getStats().

Here is the caller graph for this function:



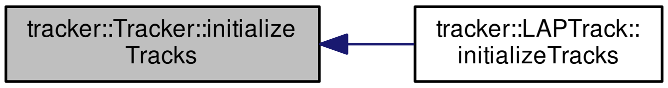
7.5.4.3 void tracker::Tracker::initializeTracks (const IVecT & *frameIdx_*, const MatT & *position_*, const MatT & *SE_position_*) [virtual]

Reimplemented in tracker::LAPTrack.

Definition at line 33 of file Tracker.cpp.

Referenced by tracker::LAPTrack::initializeTracks().

Here is the caller graph for this function:


tracker::Tracker::initialize Tracks
tracker::LAPTrack:: initializeTracks

7.5.4.4 void tracker::Tracker::initializeTracks (const IVecT & *frameIdx_*, const MatT & *position_*, const MatT & *SE_position_*, const MatT & *feature_*, const MatT & *SE_feature_*) [virtual]

Reimplemented in tracker::LAPTrack.

Definition at line 39 of file Tracker.cpp.

References feature, firstFrame, frameIdx, frameLocIdx, lastFrame, N, nDims, nFeatures, nFrameLocs, nFrames, position, SE_feature, SE_position, trackAssignment, and tracks.

7.5.4.5 void tracker::Tracker::printTracks ( ) const

Definition at line 126 of file Tracker.cpp.

References frameIdx, and tracks.

7.5.5 Member Data Documentation

7.5.5.1 MatT tracker::Tracker::feature

Definition at line 66 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::computeGapCloseMatrix(), and initialize↩
Tracks().

7.5.5.2 IdxT tracker::Tracker::firstFrame = 0

Definition at line 68 of file Tracker.h.

Referenced by tracker::LAPTrack::checkFrameIdxs(), tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack↩ ::computeGapCloseMatrix(), tracker::LAPTrack::debugF2F(), getStats(), initializeTracks(), and tracker::LAPTrack::link↩ F2F().

7.5.5.3 IVecT tracker::Tracker::frameIdx

Definition at line 63 of file Tracker.h.

Referenced by tracker::LAPTrack::checkFrameIdxs(), tracker::LAPTrack::computeGapCloseMatrix(), initializeTracks(), and printTracks().

7.5.5.4 IVecFieldT tracker::Tracker::frameLocIdx

Definition at line 74 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::debugF2F(), initializeTracks(), and tracker::LAPTrack::linkF2F().

7.5.5.5 IdxT tracker::Tracker::lastFrame = 0

Definition at line 69 of file Tracker.h.

Referenced by tracker::LAPTrack::checkFrameIdxs(), tracker::LAPTrack::computeGapCloseMatrix(), tracker::LAP↩ Track::debugF2F(), getStats(), initializeTracks(), and tracker::LAPTrack::linkF2F().

7.5.5.6 const Tracker::FloatT tracker::Tracker::log2pi = log(2∗arma::Datum<Tracker::FloatT>::pi) [static], [protected]

Definition at line 92 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), and tracker::LAPTrack::computeGapCloseMatrix().

7.5.5.7 IdxT tracker::Tracker::N = 0

Definition at line 60 of file Tracker.h.

Referenced by getStats(), and initializeTracks().

7.5.5.8 IdxT tracker::Tracker::nDims = 0

Definition at line 61 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::computeGapCloseMatrix(), getStats(), and initializeTracks().

7.5.5.9 IdxT tracker::Tracker::nFeatures = 0

Definition at line 62 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::computeGapCloseMatrix(), getStats(), and initializeTracks().

7.5.5.10 IVecT tracker::Tracker::nFrameLocs

Definition at line 73 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::debugF2F(), initializeTracks(), and tracker::LAPTrack::linkF2F().

7.5.5.11 IdxT tracker::Tracker::nFrames = 0

Definition at line 70 of file Tracker.h.

Referenced by getStats(), initializeTracks(), and tracker::LAPTrack::linkF2F().

7.5.5.12 MatT tracker::Tracker::position

Definition at line 64 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::computeGapCloseMatrix(), and initialize↩ Tracks().

7.5.5.13 MatT tracker::Tracker::SE_feature

Definition at line 67 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::computeGapCloseMatrix(), and initialize↩ Tracks().

7.5.5.14 MatT tracker::Tracker::SE_position

Definition at line 65 of file Tracker.h.

Referenced by tracker::LAPTrack::computeF2FCostMat(), tracker::LAPTrack::computeGapCloseMatrix(), and initialize↩ Tracks().

7.5.5.15 IVecT tracker::Tracker::trackAssignment `[protected]`

Definition at line 93 of file Tracker.h.

Referenced by tracker::LAPTrack::closeGaps(), getStats(), initializeTracks(), and tracker::LAPTrack::linkF2F().

7.5.5.16 TrackVecT tracker::Tracker::tracks

Definition at line 77 of file Tracker.h.

Referenced by tracker::LAPTrack::checkFrameIdxs(), tracker::LAPTrack::closeGaps(), tracker::LAPTrack::compute↩ GapCloseMatrix(), tracker::LAPTrack::debugCloseGaps(), getStats(), initializeTracks(), tracker::LAPTrack::linkF2F(), and printTracks().

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

- Tracker.h
- Tracker.cpp

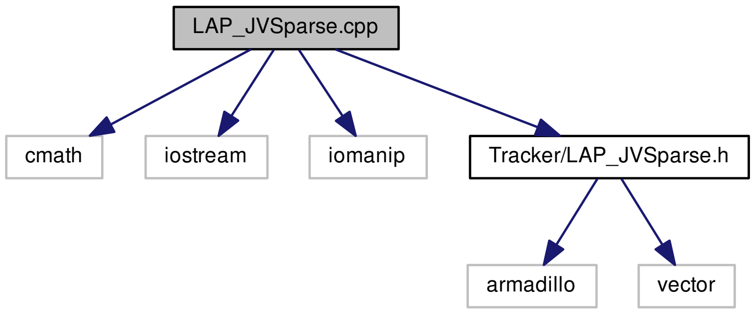
8 File Documentation

8.1 LAP_JVSparse.cpp File Reference

The member definitions for the LAP Jonker Volgenant algorithm.

```
#include <cmath>
#include <iostream>
#include <iomanip>
#include "Tracker/LAP_JVSparse.h"
```

Include dependency graph for LAP_JVSparse.cpp:



Namespaces

- tracker

8.1.1 Detailed Description

The member definitions for the LAP Jonker Volgenant algorithm.

Author

Mark J. Olah (mjo at cs.unm.edu)

Date

05-2015 This is a modern dense/sparse C++ implementation of Jonker Volgenant algoirthm using armadillo and presenting C++ and Matlab interface.

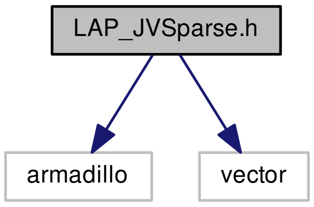
Adapted from text of Jonker and Volgenant. Computing 38, 324-340 (1986)

8.2 LAP_JVSparse.h File Reference

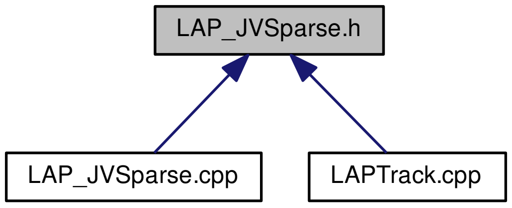
The class declaration for the LAP Jonker Volgenant algorithm.

```
#include <armadillo>
#include <vector>
```

Include dependency graph for LAP_JVSparse.h:



This graph shows which files directly or indirectly include this file:



Classes

- class tracker::LAP_JVSparse< FloatT >

Namespaces

- tracker

8.2.1 Detailed Description

The class declaration for the LAP Jonker Volgenant algorithm.

Author

Mark J. Olah (mjo@cs.unm.edu)

Date

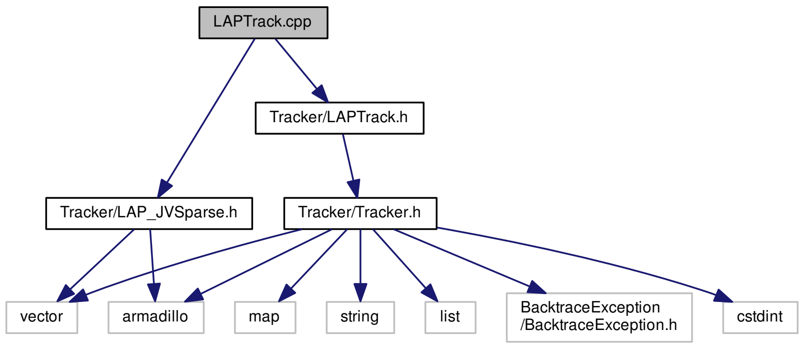
05-2015 This is a modern dense/sparse C++ implementation of Jonker Volgenant algoirthm using armadillo and presenting C++ and Matlab interface.

Adapted from text of Jonker and Volgenant. Computing 38, 324-340 (1986)

8.3 LAPTrack.cpp File Reference

The member definitions for LAPTrack.

```
#include "Tracker/LAPTrack.h"
#include "Tracker/LAP_JVSparse.h"
```
Include dependency graph for LAPTrack.cpp:



Namespaces

- tracker

8.3.1 Detailed Description

The member definitions for LAPTrack.

Author

Mark J. Olah (mjo at cs.unm.edu)

Date

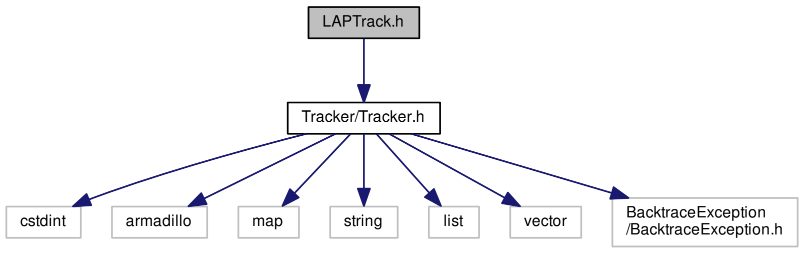
04-2015

8.4 LAPTrack.h File Reference

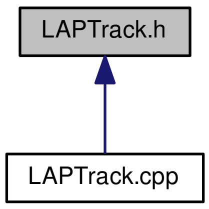
The class declaration and inline and templated functions for LAPTrack.

`#include "Tracker/Tracker.h"`

Include dependency graph for LAPTrack.h:



This graph shows which files directly or indirectly include this file:



Classes

- class tracker::LAPTrack

Namespaces

- tracker

8.4.1 Detailed Description

The class declaration and inline and templated functions for LAPTrack.

Author

Mark J. Olah (mjo@cs.unm.edu)

Date

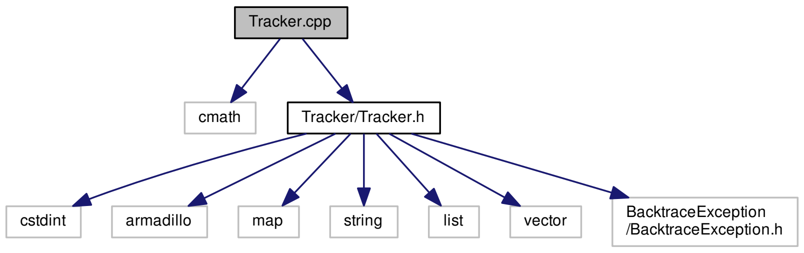
02-2015 A simple LAP/Jaquman based tracker

8.5 README.md File Reference

8.6 Tracker.cpp File Reference

The member definitions for Tracker.

```
#include <cmath>
#include "Tracker/Tracker.h"
```
Include dependency graph for Tracker.cpp:



Namespaces

- tracker

8.6.1 Detailed Description

The member definitions for Tracker.

Author

Mark J. Olah (mjo at cs.unm.edu)

Date

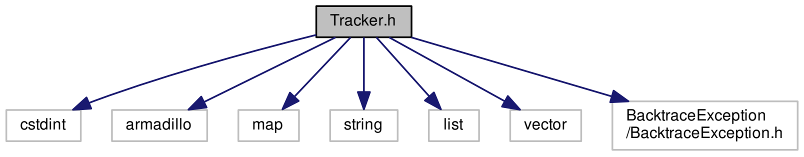
04-2015

8.7 Tracker.h File Reference

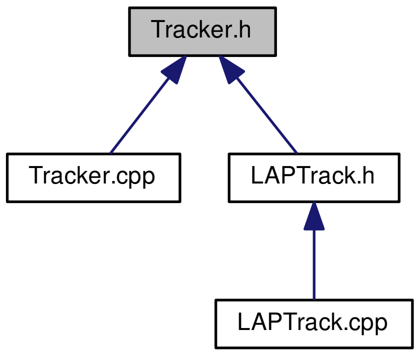
The class declaration and inline and templated functions for Tracker.

```
#include <cstdint>
#include <armadillo>
#include <map>
#include <string>
#include <list>
#include <vector>
#include "BacktraceException/BacktraceException.h"
```

Include dependency graph for Tracker.h:



This graph shows which files directly or indirectly include this file:



Classes

- struct tracker::ParameterValueError

  Parameter value is not valid.
- struct tracker::LogicalError

  Parameter value is not valid.
- class tracker::Tracker

Namespaces

- tracker

Typedefs

- using tracker::TrackerError = backtrace_exception::BacktraceException

8.7.1 Detailed Description

The class declaration and inline and templated functions for Tracker.

Author

Mark J. Olah (mjo@cs.unm.edu)

Date

02-2015 The base class for all Tracking models

Insted of templating on the FloatT type, which is problematic for inheritance hierarchies of templated base classes. Instead wuse a typedef to allow configuration of use with either float/double. Default is double.

Index