RRT* PATH PLANNING

Generated by Doxygen 1.8.15

Contents

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 3

Class Documentation

3.1 RRTstar::Planner< State, Trajectory, System > Class Template Reference

RRT∗ Planner class.

```
#include <rrts.h>
```

Public Member Functions

- Planner ()

  Planner constructor.
- ∼Planner ()

  Planner destructor.
- int setGamma (double gammaIn)

  Sets the gamma constant of the RRT∗.
- int setSystem (System &system)

  Sets the dynamical system used in the RRT∗ trajectory generation.
- vertex_t & getRootVertex ()

  Returns a reference to the root vertex.
- int initialize ()

  Initializes the RRT∗ algorithm.
- int iteration ()

  Executes one iteration of the RRT∗ algorithm.
- double getBestVertexCost ()

  Returns the cost of the best vertex in the RRT∗.
- vertex_t & getBestVertex ()

  Returns a reference to the best vertex in the RRT∗.
- int getBestTrajectory (std::list< double ∗> &trajectory)

  Returns the best trajectory as a list of double arrays.

Public Attributes

- std::list< vertex_t ∗ > listVertices

  A list of all the vertices.
- int numVertices

  Number of vertices in the list.
- System ∗ system

  A pointer to the system class.

3.1.1 Detailed Description

**template<class State, class Trajectory, class System>
class RRTstar::Planner< State, Trajectory, System >**

RRT∗ Planner class.

3.1.2 Member Function Documentation

3.1.2.1 getBestTrajectory()

```
template<class State , class Trajectory , class System >
int RRTstar::Planner< State, Trajectory, System >::getBestTrajectory (
            std::list< double *> & trajectory )
```

Returns the best trajectory as a list of double arrays.

Parameters

trajectory	The trajectory that contains the best trajectory as a list of double arrays of dimension system->getNumDimensions()

3.1.2.2 setGamma()

```
template<class State , class Trajectory , class System >
int RRTstar::Planner< State, Trajectory, System >::setGamma (
            double gammaIn )
```

Sets the gamma constant of the RRT∗.

Parameters

gamma↩ In	The new value of the gamma parameter

3.1.2.3 setSystem()

```
template<class State , class Trajectory , class System >
int RRTstar::Planner< State, Trajectory, System >::setSystem (
            System & system )
```

Sets the dynamical system used in the RRT∗ trajectory generation.

Parameters

system	A reference to the new dynamical system

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

- rrts.h

3.2 SingleIntegrator::region Class Reference

region class

```
#include <system_single_integrator.h>
```

Public Member Functions

- region ()
  region constructor
- ∼region ()
  region destructor
- int setNumDimensions (int numDimensionsIn)
  Sets the dimensionality of the region.

Public Attributes

- double ∗ center
  Cartesian coordinates of the center of the region.
- double ∗ size
  Size of the region in cartesian coordinates.

3.2.1 Detailed Description

region class

3.2.2 Member Function Documentation

3.2.2.1 setNumDimensions()

```
int SingleIntegrator::region::setNumDimensions (
              int numDimensionsIn )
```

Sets the dimensionality of the region.

Parameters

num↩ DimensionsIn	New number of dimensions.

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

- system_single_integrator.h

3.3 SingleIntegrator::State Class Reference

State Class.

```
#include <system_single_integrator.h>
```

Public Member Functions

- State ()
  State constructor.
- ∼State ()
  State desctructor.
- State (const State &stateIn)
  State copy constructor.
- int setNumDimensions (int numDimensions)
  Sets the dimensionality of the state object.
- State & operator= (const State &stateIn)
  State assignment operator.

Public Attributes

- double ∗ center
  Center position of the object.
- double ∗ size
  Dimensions of the object.

Friends

- class **System**
- class **Trajectory**

3.3.1 Detailed Description

State Class.

The State Class represents the object to move as a box with a center position and dimensions

3.3.2 Constructor & Destructor Documentation

3.3.2.1 State()

```
SingleIntegrator::State::State (
              const State & stateIn )
```

State copy constructor.

It builds up a new State object starting from an existing one

3.3.3 Member Function Documentation

3.3.3.1 operator=()

```
State& SingleIntegrator::State::operator= (
              const State & stateIn )
```

State assignment operator.

Parameters

state←↩ In	The state object to be assigned

3.3.3.2 setNumDimensions()

```
int SingleIntegrator::State::setNumDimensions (
              int numDimensions )
```

Sets the dimensionality of the state object.

Parameters

numDimensions	New number of dimensions.

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

- system_single_integrator.h

3.4 SingleIntegrator::System Class Reference

System Class.

```
#include <system_single_integrator.h>
```

Public Member Functions

- System ()

  System constructor.
- ∼System ()

  System destructor.
- int setNumDimensions (int numDimensionsIn)

  Sets the dimensionality of the Euclidean space.
- int getNumDimensions ()

  Returns the dimensionality of the Euclidean space.
- State & getRootState ()

  Returns a reference to the root state.
- void setRootState (double ∗center, double ∗size)

  Sets the fields of the rrt∗ root state.
- int getStateKey (State &stateIn, double ∗stateKey)

  Returns the statekey for the given state.
- bool isReachingTarget (State &stateIn)

  Returns true of the given state reaches the target.
- int sampleState (State &randomStateOut)

  Returns a sample state.
- int extendTo (State &stateFromIn, State &stateTowardsIn, Trajectory &trajectoryOut, bool &exact←ConnectionOut)

  Returns a the cost of the trajectory that connects stateFromIn and stateTowardsIn. The trajectory is also returned in trajectoryOut.
- double evaluateExtensionCost (State &stateFromIn, State &stateTowardsIn, bool &exactConnectionOut)

  Returns the cost of the trajectory that connects stateFromIn and StateTowardsIn.
- double evaluateCostToGo (State &stateIn)

  Returns a lower bound on the cost to go starting from stateIn.
- int getTrajectory (State &stateFromIn, State &stateToIn, std::list< double ∗> &trajectoryOut)

  Returns the trajectory as a list of double arrays, each with dimension getNumDimensions.

Public Attributes

- region regionOperating

  The operating region.
- region regionGoal

  The goal region.
- std::list< region ∗ > obstacles

  The list of all obstacles.

3.4.1 Detailed Description

System Class.

3.4.2 Member Function Documentation

3.4.2.1 evaluateCostToGo()

```
double SingleIntegrator::System::evaluateCostToGo (
            State & stateIn )
```

Returns a lower bound on the cost to go starting from stateIn.

Parameters

state↩ In	Starting state

3.4.2.2 evaluateExtensionCost()

```
double SingleIntegrator::System::evaluateExtensionCost (
            State & stateFromIn,
            State & stateTowardsIn,
            bool & exactConnectionOut )
```

Returns the cost of the trajectory that connects stateFromIn and StateTowardsIn.

Parameters

stateFromIn	Initial state
stateTowardsIn	Final state
exactConnectionOut	Set to true if the initial and the final states can be connected exactly.

3.4.2.3 extendTo()

```
int SingleIntegrator::System::extendTo (
            State & stateFromIn,
            State & stateTowardsIn,
            Trajectory & trajectoryOut,
            bool & exactConnectionOut )
```

Returns a the cost of the trajectory that connects stateFromIn and stateTowardsIn. The trajectory is also returned in trajectoryOut.

Parameters

stateFromIn	Initial state
stateTowardsIn	Final state
trajectoryOut	Trajectory that starts the from the initial state and reaches near the final state.
exactConnectionOut	Set to true if the initial and the final states can be connected exactly.

3.4.2.4 getStateKey()

```
int SingleIntegrator::System::getStateKey (
            State & stateIn,
            double * stateKey )
```

Returns the statekey for the given state.

Parameters

stateIn	The given state
stateKey	The key to the state. An array of dimension getNumDimensions()

3.4.2.5 getTrajectory()

```
int SingleIntegrator::System::getTrajectory (
            State & stateFromIn,
            State & stateToIn,
            std::list< double *> & trajectoryOut )
```

Returns the trajectory as a list of double arrays, each with dimension getNumDimensions.

Parameters

stateFromIn	Initial state
stateToIn	Final state
trajectoryOut	The list of double arrays that represent the trajectory

3.4.2.6 sampleState()

```
int SingleIntegrator::System::sampleState (
            State & randomStateOut )
```

Returns a sample state.

The sampled state is granted being inside the operating region

Parameters

randomStateOut	contains a new randomly sampled state object

3.4.2.7 setRootState()

```
void SingleIntegrator::System::setRootState (
            double * center,
            double * size )  [inline]
```

Sets the fields of the rrt∗ root state.

Parameters

center	The given center position of the root
size	The given dimensions of the root

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

- system_single_integrator.h

3.5 SingleIntegrator::Trajectory Class Reference

Trajectory Class.

```
#include <system_single_integrator.h>
```

Public Member Functions

- Trajectory ()
  Trajectory constructor.
- ∼Trajectory ()
  Trajectory destructor.
- Trajectory (const Trajectory &trajectoryIn)
  Trajectory copy constructor.
- Trajectory & operator= (const Trajectory &trajectoryIn)
  Trajectory assignment constructor.
- State & getEndState ()
  Returns a reference to the end state of this trajectory.
- State & getEndState () const
  Returns a reference to the end state of this trajectory (constant).
- double evaluateCost ()
  Returns the cost of this trajectory.

Friends

- class **System**

3.5.1 Detailed Description

Trajectory Class.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 Trajectory()

```
SingleIntegrator::Trajectory::Trajectory (
            const Trajectory & trajectoryIn )
```

Trajectory copy constructor.

Parameters

trajectory↩ In	The trajectory to be copied.

3.5.3 Member Function Documentation

3.5.3.1 operator=()

```
Trajectory& SingleIntegrator::Trajectory::operator= (
            const Trajectory & trajectoryIn )
```

Trajectory assignment constructor.

Parameters

trajectory↩ In	the trajectory to be copied.

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

- system_single_integrator.h

3.6 RRTstar::Vertex< State, Trajectory, System > Class Template Reference

RRT∗ Vertex class.

```
#include <rrts.h>
```

Public Member Functions

- Vertex ()

    Vertex constructor.
- ∼Vertex ()

    Vertex destructor.
- Vertex (const Vertex &vertexIn)

    Vertex copy constructor.
- State & getState ()

    Returns a reference to the state.
- State & getState () const

    Returns a reference to the state (constant)
- Vertex & getParent ()

    Returns a reference to the parent vertex.
- double getCost ()

    Returns the accumulated cost at this vertex.

Friends

- class **Planner< State, Trajectory, System >**

3.6.1 Detailed Description

template<class State, class Trajectory, class System>
class RRTstar::Vertex< State, Trajectory, System >

RRT∗ Vertex class.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 Vertex()

```
template<class State , class Trajectory , class System >
RRTstar::Vertex< State, Trajectory, System >::Vertex (
            const Vertex< State, Trajectory, System > & vertexIn )
```

Vertex copy constructor.

Parameters

vertex←In	A reference to the vertex to be copied.

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

- rrts.h

Chapter 4

File Documentation

4.1 rrts.h File Reference

```
#include "kdtree.h"
#include <list>
#include <set>
#include <vector>
```

Classes

- class RRTstar::Planner< State, Trajectory, System >

  RRT∗ Planner class.
- class RRTstar::Vertex< State, Trajectory, System >

  RRT∗ Vertex class.
- class RRTstar::Planner< State, Trajectory, System >

  RRT∗ Planner class.

4.2 system_single_integrator.h File Reference

```
#include <list>
#include <ctime>
```

Classes

- class SingleIntegrator::region

  region class
- class SingleIntegrator::State

  State Class.
- class SingleIntegrator::Trajectory

  Trajectory Class.
- class SingleIntegrator::System

  System Class.

Index