ContactModel

5.0

Generated by Doxygen 1.8.5

Wed Jun 1 2022 12:05:00

Contents

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Module Documentation

6.1 Models

Modules

- Interactions

6.1.1 Detailed Description

6.2 Interactions

Modules

- Contact

6.2.1 Detailed Description

6.3 Contact

Files

- file class_declarations.hh

  Forward declaration of classes defined in the contact model.
- file contact.hh

  (Base class to for the contact manager for use with contact interaction model)
- file contact_facet.hh

  Individual facets for use with contact interaction models.
- file contact_messages.hh

  Contact message for message handling.
- file contact_pair.hh

  Base class for pair of contact facets for use with contact interaction model.
- file contact_params.hh

  A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys.
- file contact_surface.hh

  Vehicle surface model for contact.
- file contact_surface_factory.hh

  Factory that creates an contact interaction surface from a surface model.
- file contact_utils.hh

  This Model is used for utility rotines.
- file contact_utils_inline.hh

  Define ContactUtils::create_relstate_name, ContactUtils::copy_const_char_to_char.
- file line_contact_facet.hh

  The contact facet based on the distance to a line segment centered on the vehicle point.
- file line_contact_facet_factory.hh

  Creates a line contact facet from an cylinder facet.
- file line_contact_pair.hh

  Class for a pair of line contact facets for use with contact interaction model.
- file line_point_contact_pair.hh

  Class for a pair of a line contact facet and a point contact facet for use with contact interaction model.
- file pair_interaction.hh

  A class to define the interaction type for a pair of contact facets.
- file point_contact_facet.hh

  The contact facet based on the distance to a single point, specifically the vehicle point.
- file point_contact_facet_factory.hh

  Creates a point contact facet from an circular flat plate facet.
- file point_contact_pair.hh

  Class for a pair of point contact facets for use with contact interaction model.
- file spring_pair_interaction.hh

  A class for pair interactions based on a simple spring.
- file contact.cc

  Base Contact for use with contact interaction model.
- file contact_facet.cc

  Define ContactFacet::create_vehicle_point.
- file contact_messages.cc

  Implement contact_messages.
- file contact_pair.cc

  ContactPair class for use with contact interaction model.
- file contact_params.cc

contact parameters for use in the surface model
- file contact_surface.cc

  Vehicle surface model for the contact interaction models.
- file contact_surface_factory.cc

  Factory that creates an contact surface, from a surface model.
- file line_contact_facet.cc

  Define LineContactFacet functions.
- file line_contact_facet_factory.cc

  Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object.
- file line_contact_pair.cc

  LineContactPair class for use with contact interaction model.
- file line_point_contact_pair.cc

  LinePointContactPair class for use with contact interaction model.
- file pair_interaction.cc

  A class to define the interaction type for a pair of contact facets.
- file point_contact_facet.cc

  Define PointContactFacet functions.
- file point_contact_facet_factory.cc

  Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object.
- file point_contact_pair.cc

  ContactPair class for use with contact interaction model.
- file spring_pair_interaction.cc

  spring pair interaction for use in the contact model

Namespaces

- jeod

  Namespace jeod.

Macros

- #define PATH "interactions/contact"

6.3.1 Detailed Description

6.3.2 Macro Definition Documentation

6.3.2.1 #define PATH "interactions/contact"

Definition at line 37 of file contact_messages.cc.

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

- class Contact

  An base contact class for use in the surface model.
- class ContactFacet

  An contact interaction specific facet for use in the surface model.
- class ContactMessages

  Messages associated with use of the contact model.
- class ContactPair

  An base contact pair class for use in the contact model.
- class ContactParams

  A base class for all contact parameters used in the surface model.
- class ContactSurface

  The contact specific interaction surface, for use with the surface model.
- class ContactSurfaceFactory

  The surface factory that creates an contact specific surface from a general surface.
- class ContactUtils

  Utility string and math functions for the contact model.
- class LineContactFacet

  The contact facet based on the distance to a single point, specifically the vehicle point.
- class LineContactFacetFactory

  Creates a PointContactFacet from an InteractionFacet.
- class LineContactPair

  An point to point contact pair for use in the contact model.
- class LinePointContactPair

  An point to point contact pair for use in the contact model.
- class PairInteraction

  Simple spring contact parameters.
- class PointContactFacet

  The contact facet based on the distance to a single point, specifically the vehicle point.
- class PointContactFacetFactory

  Creates a PointContactFacet from an InteractionFacet.

- class PointContactPair

  An point to point contact pair for use in the contact model.
- class SpringPairInteraction

  Simple spring contact parameters.

7.1.1 Detailed Description

Namespace jeod.

Chapter 8

Data Structure Documentation

8.1 jeod::Contact Class Reference

An base contact class for use in the surface model.

```
#include <contact.hh>
```

Public Member Functions

- Contact ()

  Default Constructor.
- virtual ∼Contact ()

  Destructor.
- void register_contact (ContactFacet ∗facet)

  Register one ContactFacet with all inclusive interactions with other registered ContactFacets.
- void register_contact (ContactFacet ∗∗facets, unsigned int n_facets)

  Register an array of ContactFacets with all inclusive interactions with other registered ContactFacets.
- void register_contact (ContactFacet ∗facet1, ContactFacet ∗facet2)

  Register two facets as a pair.
- void register_contact (ContactFacet ∗∗facets1, unsigned int n_facets1, ContactFacet ∗∗facets2, unsigned int n_facets2)

  Regiser to arrays of facets and create specific pairs between all of them.
- void register_interaction (PairInteraction ∗interaction)

  Register a pair interaction.
- virtual PairInteraction ∗ find_interaction (ContactParams ∗params_1, ContactParams ∗params_2)

  find a PairInteraction baced on a set of ContactParams.
- void initialize_contact (DynManager ∗manager)

  Initialize ContactFacets and the mananger by cleaning up the pair list.
- bool unique_pair (ContactFacet ∗facet_1, ContactFacet ∗facet_2)

  Check to see if a pair of facets already exists.
- void check_contact ()

  iterate through contact pairs list then call the appropriate contact resolution functions

Data Fields

- bool active

  toggles contact on and off, true=on false=off
- double contact_limit_factor

  factor determines if contact is limited by a muliple of the maximum dimensions of the facets in a pair.

Protected Attributes

- DynManager ∗ dyn_manager

  Pointer to the dyn_manager so relstates and be successfully initialized.
- JeodPointerList< ContactPair >
  ::type contact_pairs

  list of all possible pairings of contact facets registered with this contact class or derived class
- JeodPointerList
  < PairInteraction >::type pair_interactions

  list of all possible pair interaction types

Private Member Functions

- Contact & operator= (const Contact &rhs)
- Contact (const Contact &rhs)

Friends

- class InputProcessor
- void init_attrjeod__Contact ()

8.1.1 Detailed Description

An base contact class for use in the surface model.

Definition at line 56 of file contact.hh.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 jeod::Contact::Contact (void)

Default Constructor.

Definition at line 52 of file contact.cc.

References contact_pairs, and pair_interactions.

8.1.2.2 jeod::Contact::∼Contact (void) [virtual]

Destructor.

Definition at line 69 of file contact.cc.

References contact_pairs, and pair_interactions.

8.1.2.3 jeod::Contact::Contact (const Contact & *rhs*) [private]

8.1.3 Member Function Documentation

8.1.3.1 void jeod::Contact::check_contact (void)

iterate through contact pairs list then call the appropriate contact resolution functions

Definition at line 97 of file contact.cc.

References active, and contact_pairs.

8.1.3.2 PairInteraction ∗ jeod::Contact::find_interaction (ContactParams ∗ *params_1*, ContactParams ∗ *params_2*) [virtual]

find a PairInteraction baced on a set of ContactParams.

Returns

pointer to a PairInteraction

Parameters

in	*params_1*	ContactParams from a ContactFacet
in	*params_2*	ContactParams from a ContactFacet

Definition at line 281 of file contact.cc.

References pair_interactions.

Referenced by jeod::LineContactFacet::create_pair(), and jeod::PointContactFacet::create_pair().

8.1.3.3 void jeod::Contact::initialize_contact (DynManager ∗ *manager*)

Initialize ContactFacets and the mananger by cleaning up the pair list.

Parameters

in,out	*manager*	Dynamics Manager

Definition at line 119 of file contact.cc.

References contact_pairs, jeod::ContactFacet::create_pair(), dyn_manager, jeod::ContactPair::get_subject(), jeod-::ContactPair::initialize_relstate(), and unique_pair().

8.1.3.4 Contact& jeod::Contact::operator= (const Contact & *rhs*) [private]

8.1.3.5 void jeod::Contact::register_contact (ContactFacet ∗ *facet*)

Register one ContactFacet with all inclusive interactions with other registered ContactFacets.

Parameters

in,out	*facet*	ContactFacet

Definition at line 178 of file contact.cc.

References contact_pairs, and jeod::ContactFacet::create_pair().

Referenced by register_contact().

8.1.3.6 void jeod::Contact::register_contact (ContactFacet ∗∗ *facets*, unsigned int *nFacets*)

Register an array of ContactFacets with all inclusive interactions with other registered ContactFacets.

Parameters

in,out	*facets*	array of ContactFacets
in	*nFacets*	number of ContactFacets in array

Definition at line 199 of file contact.cc.

References register_contact().

8.1.3.7 void jeod::Contact::register_contact (ContactFacet ∗ *facet1*, ContactFacet ∗ *facet2*)

Register two facets as a pair.

Parameters

`in,out`	*facet1*	Contact Facet 1
`in,out`	*facet2*	Contact Facet 2

Definition at line 217 of file contact.cc.

References contact_pairs, jeod::ContactFacet::create_pair(), and dyn_manager.

8.1.3.8 void jeod::Contact::register_contact (**ContactFacet** ∗∗ *facets1*, unsigned int *nFacets1*, **ContactFacet** ∗∗ *facets2*, unsigned int *nFacets2*)

Regiser to arrays of facets and create specific pairs between all of them.

Parameters

`in,out`	*facets1*	array of ContactFacets
`in`	*nFacets1*	number of ContactFacets in array
`in,out`	*facets2*	array of ContactFacets
`in`	*nFacets2*	number of ContactFacets in array

Definition at line 244 of file contact.cc.

References register_contact().

8.1.3.9 void jeod::Contact::register_interaction (**PairInteraction** ∗ *interaction*)

Register a pair interaction.

Parameters

`in`	*interaction*	PairInteraction to add to list

Definition at line 266 of file contact.cc.

References pair_interactions.

8.1.3.10 bool jeod::Contact::unique_pair (**ContactFacet** ∗ *facet_1*, **ContactFacet** ∗ *facet_2*)

Check to see if a pair of facets already exists.

Returns

bool

Parameters

`in,out`	*facet_1*	ContactFacet
`in,out`	*facet_2*	ContactFacet

Definition at line 155 of file contact.cc.

References contact_pairs.

Referenced by initialize_contact().

8.1.4 Friends And Related Function Documentation

8.1.4.1 void init_attrjeod__Contact ( ) `[friend]`

8.1.4.2 friend class InputProcessor [friend]

Definition at line 58 of file contact.hh.

8.1.5 Field Documentation

8.1.5.1 bool jeod::Contact::active

toggles contact on and off, true=on false=off

trick_units(–)

Definition at line 64 of file contact.hh.

Referenced by check_contact().

8.1.5.2 double jeod::Contact::contact_limit_factor

factor determines if contact is limited by a muliple of the maximum dimensions of the facets in a pair.

trick_units(–)

Definition at line 70 of file contact.hh.

Referenced by jeod::LineContactFacet::create_pair(), and jeod::PointContactFacet::create_pair().

8.1.5.3 JeodPointerList<ContactPair>::type jeod::Contact::contact_pairs [protected]

list of all possible pairings of contact facets registered with this contact class or derived class

trick_io(∗∗)

Definition at line 129 of file contact.hh.

Referenced by check_contact(), Contact(), initialize_contact(), register_contact(), unique_pair(), and ∼Contact().

8.1.5.4 DynManager∗ jeod::Contact::dyn_manager [protected]

Pointer to the dyn_manager so relstates and be successfully initialized.

trick_units(–)

Definition at line 123 of file contact.hh.

Referenced by initialize_contact(), and register_contact().

8.1.5.5 JeodPointerList<PairInteraction>::type jeod::Contact::pair_interactions [protected]

list of all possible pair interaction types

trick_io(∗∗)

Definition at line 134 of file contact.hh.

Referenced by Contact(), find_interaction(), register_interaction(), and ∼Contact().

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

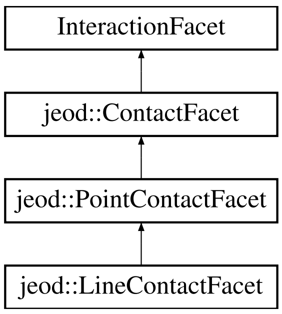
- contact.hh
- contact.cc

8.2 jeod::ContactFacet Class Reference

An contact interaction specific facet for use in the surface model.

`#include <contact_facet.hh>`

Inheritance diagram for jeod::ContactFacet:



Public Member Functions

- ContactFacet ()

  Default constructor.
- virtual ∼ContactFacet ()

  Destructor.
- void create_vehicle_point (void)

  Create a vehicle point to track the state information of the contact facet.
- virtual void set_max_dimension (void)=0

  Calculate the max dimension of the facet for range limit determination.
- const char ∗ get_name (void) const

  Accessor for name.
- virtual void calculate_torque (double ∗tmp_force)=0

  Calculate the torque acting on the facet in the vehicle structural frame.
- virtual ContactPair ∗ create_pair (void)=0

  Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.
- virtual ContactPair ∗ create_pair (ContactFacet ∗target, Contact ∗contact)=0

  Overloaded functions that creates a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

Data Fields

- bool active

  toggles this contact facet on and off, true=on false=off
- ContactParams ∗ surface_type

  Stores the name of surface material that the facet is constructed of.
- DynBody ∗ vehicle_body

  DynBody associated with this facet for structural frame information.
- double max_dimension

  maximum dimension of the contact facet for use in limiting pair in_contact calls
- double position [3]

  position of the facet in vehicle structural frame.
- double normal [3]

normal of the facet relative to the vehicle structural frame.
- const BodyRefFrame ∗ vehicle_point

  Vehicle point for relstate calculations.

Private Member Functions

- ContactFacet & operator= (const ContactFacet &rhs)
- ContactFacet (const ContactFacet &rhs)

Friends

- class InputProcessor
- void init_attrjeod__ContactFacet ()

8.2.1 Detailed Description

An contact interaction specific facet for use in the surface model.

Definition at line 50 of file contact_facet.hh.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 jeod::ContactFacet::ContactFacet (void)

Default constructor.

Definition at line 49 of file contact_facet.cc.

References normal, and position.

8.2.2.2 jeod::ContactFacet::∼ContactFacet (void) [virtual]

Destructor.

Definition at line 65 of file contact_facet.cc.

8.2.2.3 jeod::ContactFacet::ContactFacet (const ContactFacet & *rhs*) [private]

8.2.3 Member Function Documentation

8.2.3.1 virtual void jeod::ContactFacet::calculate_torque (double ∗ *tmp_force*) [pure virtual]

Calculate the torque acting on the facet in the vehicle structural frame.

Implemented in jeod::PointContactFacet, and jeod::LineContactFacet.

Referenced by jeod::SpringPairInteraction::calculate_forces().

8.2.3.2 virtual ContactPair∗ jeod::ContactFacet::create_pair (void) [pure virtual]

Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

This function is called to create a pair that only contains a subject.

Implemented in jeod::PointContactFacet, and jeod::LineContactFacet.

Referenced by jeod::Contact::initialize_contact(), and jeod::Contact::register_contact().

8.2.3.3 virtual ContactPair∗ jeod::ContactFacet::create_pair (ContactFacet ∗ *target*, Contact ∗ *contact*) [pure virtual]

Overloaded functions that creates a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

This function is called when a subject and target are known.

Implemented in jeod::PointContactFacet, and jeod::LineContactFacet.

8.2.3.4 void jeod::ContactFacet::create_vehicle_point (void)

Create a vehicle point to track the state information of the contact facet.

Create a vehicle point from the base facet position and orientation and store the created vehicle point in the contact facet.

Definition at line 78 of file contact_facet.cc.

References get_name(), normal, position, vehicle_body, and vehicle_point.

Referenced by jeod::LineContactFacetFactory::create_facet(), and jeod::PointContactFacetFactory::create_facet().

8.2.3.5 const char ∗ jeod::ContactFacet::get_name (void) const [inline]

Accessor for name.

Returns

Point name

Definition at line 153 of file contact_facet.hh.

Referenced by create_vehicle_point().

8.2.3.6 ContactFacet& jeod::ContactFacet::operator= (const ContactFacet & *rhs*) [private]

8.2.3.7 virtual void jeod::ContactFacet::set_max_dimension (void) [pure virtual]

Calculate the max dimension of the facet for range limit determination.

Implemented in jeod::PointContactFacet, and jeod::LineContactFacet.

8.2.4 Friends And Related Function Documentation

8.2.4.1 void init_attrjeod__ContactFacet ( ) [friend]

8.2.4.2 friend class InputProcessor [friend]

Definition at line 52 of file contact_facet.hh.

8.2.5 Field Documentation

8.2.5.1 bool jeod::ContactFacet::active

toggles this contact facet on and off, true=on false=off

trick_units(–)

Definition at line 58 of file contact_facet.hh.

Referenced by jeod::ContactPair::is_active().

8.2.5.2 double jeod::ContactFacet::max_dimension

maximum dimension of the contact facet for use in limiting pair in_contact calls

trick_units(m)

Definition at line 75 of file contact_facet.hh.

Referenced by jeod::LineContactFacet::create_pair(), jeod::PointContactFacet::create_pair(), jeod::LineContact-Facet::set_max_dimension(), and jeod::PointContactFacet::set_max_dimension().

8.2.5.3 double jeod::ContactFacet::normal[3]

normal of the facet relative to the vehicle structural frame.

trick_units(–)

Definition at line 85 of file contact_facet.hh.

Referenced by ContactFacet(), jeod::PointContactFacetFactory::create_facet(), jeod::LineContactFacetFactory-::create_facet(), and create_vehicle_point().

8.2.5.4 double jeod::ContactFacet::position[3]

position of the facet in vehicle structural frame.

trick_units(m)

Definition at line 80 of file contact_facet.hh.

Referenced by ContactFacet(), jeod::PointContactFacetFactory::create_facet(), jeod::LineContactFacetFactory-::create_facet(), and create_vehicle_point().

8.2.5.5 ContactParams∗ jeod::ContactFacet::surface_type

Stores the name of surface material that the facet is constructed of.

This information is used to deturmine the contact parameters used when pairs are constructed.trick_units(–)

Definition at line 65 of file contact_facet.hh.

Referenced by jeod::LineContactFacetFactory::create_facet(), jeod::PointContactFacetFactory::create_facet(), jeod::LineContactFacet::create_pair(), and jeod::PointContactFacet::create_pair().

8.2.5.6 DynBody∗ jeod::ContactFacet::vehicle_body

DynBody associated with this facet for structural frame information.

trick_units(–)

Definition at line 70 of file contact_facet.hh.

Referenced by jeod::SpringPairInteraction::calculate_forces(), jeod::LineContactFacet::calculate_torque(), jeod-::PointContactFacet::calculate_torque(), jeod::LineContactFacetFactory::create_facet(), jeod::PointContactFacet-Factory::create_facet(), create_vehicle_point(), and jeod::ContactPair::initialize_relstate().

8.2.5.7 const BodyRefFrame∗ jeod::ContactFacet::vehicle_point

Vehicle point for relstate calculations.

trick_units(–)

Definition at line 90 of file contact_facet.hh.

Referenced by jeod::SpringPairInteraction::calculate_forces(), jeod::LineContactFacet::calculate_torque(), jeod::-PointContactFacet::calculate_torque(), create_vehicle_point(), jeod::PointContactPair::initialize_pair(), jeod::Line-ContactPair::initialize_pair(), and jeod::LinePointContactPair::initialize_pair().

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

- contact_facet.hh
- contact_facet.cc

8.3 jeod::ContactMessages Class Reference

Messages associated with use of the contact model.

`#include <contact_messages.hh>`

Static Public Attributes

- static char const ∗ initialization_error

  Associated with errors during initialization of the contact model.
- static char const ∗ runtime_error

  Associated with errors during the runtime of the contact model.
- static char const ∗ pre_initialization_error

  Associated with errors during the setup of the system, before runtime.
- static char const ∗ initialization_warns

  Associated with warning during initialization of the contact model.
- static char const ∗ runtime_warns

  Associated with warnings given at runtime.
- static char const ∗ runtime_inform

  Associated with information given at runtime.

Private Member Functions

- ContactMessages (void)
- ContactMessages (const ContactMessages &rhs)
- ContactMessages & operator= (const ContactMessages &rhs)

Friends

- class InputProcessor
- void init_attrjeod__ContactMessages ()

8.3.1 Detailed Description

Messages associated with use of the contact model.

Definition at line 50 of file contact_messages.hh.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 jeod::ContactMessages::ContactMessages (void) [private]

8.3.2.2 jeod::ContactMessages::ContactMessages (const ContactMessages & *rhs*) [private]

8.3.3 Member Function Documentation

8.3.3.1 ContactMessages& jeod::ContactMessages::operator= (const ContactMessages & *rhs*) [private]

8.3.4 Friends And Related Function Documentation

8.3.4.1 void init_attrjeod__ContactMessages ( ) [friend]

8.3.4.2 friend class InputProcessor [friend]

Definition at line 53 of file contact_messages.hh.

8.3.5 Field Documentation

8.3.5.1 char const * jeod::ContactMessages::initialization_error [static]

Initial value:

```
=
    "interactions/contact"  "initialization_error"
```

Associated with errors during initialization of the contact model.

trick_units(–)

Definition at line 64 of file contact_messages.hh.

Referenced by jeod::ContactSurface::allocate_array(), jeod::ContactSurface::allocate_interaction_facet(), jeod::LineContactFacetFactory::create_facet(), jeod::PointContactFacetFactory::create_facet(), and jeod::ContactSurfaceFactory::create_surface().

8.3.5.2 char const * jeod::ContactMessages::initialization_warns [static]

Initial value:

```
=
    "interactions/contact"  "initialization_warns"
```

Associated with warning during initialization of the contact model.

trick_units(–)

Definition at line 79 of file contact_messages.hh.

Referenced by jeod::LineContactFacet::create_pair(), and jeod::PointContactFacet::create_pair().

8.3.5.3 char const * jeod::ContactMessages::pre_initialization_error [static]

Initial value:

```
=
    "interactions/contact"  "pre_initialization_error"
```

Associated with errors during the setup of the system, before runtime.

trick_units(–)

Definition at line 72 of file contact_messages.hh.

Referenced by jeod::ContactSurfaceFactory::add_facet_params().

8.3.5.4 char const ∗ jeod::ContactMessages::runtime_error [static]

Initial value:

```
=
    "interactions/contact"  "runtime_error"
```

Associated with errors during the runtime of the contact model.

trick_units(–)

Definition at line 68 of file contact_messages.hh.

8.3.5.5 char const ∗ jeod::ContactMessages::runtime_inform [static]

Initial value:

```
=
 "interactions/contact"  "runtime_warns"
```

Associated with information given at runtime.

trick_units(–)

Definition at line 90 of file contact_messages.hh.

8.3.5.6 char const ∗ jeod::ContactMessages::runtime_warns [static]

Initial value:

```
=
    "interactions/contact"  "runtime_warns"
```

Associated with warnings given at runtime.

trick_units(–)

Definition at line 83 of file contact_messages.hh.

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

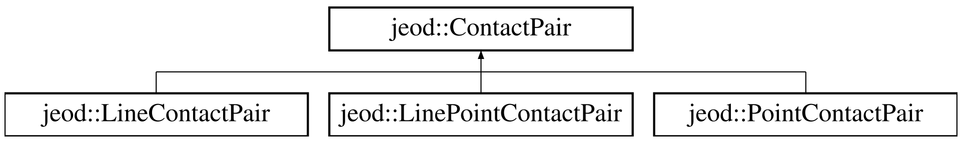
- contact_messages.hh
- contact_messages.cc

8.4 jeod::ContactPair Class Reference

An base contact pair class for use in the contact model.

`#include <contact_pair.hh>`

Inheritance diagram for jeod::ContactPair:



Public Member Functions

- ContactPair ()

    Default Constructor.
- virtual ∼ContactPair ()

    Destructor.
- bool in_range ()

    test whether the pair is in range for interaction
- bool is_active ()

    Determine if contact can occur between the two facets.
- bool is_complete (void)

    Determine if the pair has a target facet.
- ContactFacet ∗ get_subject (void)

    Determine if the pair has a target facet.
- ContactFacet ∗ get_target (void)

    Determine if the pair has a target facet.
- virtual void in_contact (void)=0

    Virtual funtion to determine if the pair is in contact.
- virtual void initialize_pair (ContactFacet ∗subject_facet, ContactFacet ∗target_facet)=0

    Initialize the contact pair by setting the subject, target, and creating the relstate if possible.
- virtual void initialize_relstate (DynManager ∗dyn_manager)

    Initialize the relative state between the facets and register with the dynamics manager.
- virtual bool check_tree ()

    Make sure the two contact facets are not on the same mass tree.

Data Fields

- PairInteraction ∗ interaction

    Parameters that define the force calculation function between the subjec and target.
- double interaction_distance

    rel_state distance at which in_contact should be called

Protected Attributes

- RelativeDerivedState rel_state

    Current relative state between the subject and the target in the subject frame.
- ContactFacet ∗ subject

    pointer to the contact facet that is the subject of the associated relative states.
- ContactFacet ∗ target

    pointer to the contact facet that is the target of the associated relative states.

Private Member Functions

- ContactPair & operator= (const ContactPair &rhs)
- ContactPair (const ContactPair &rhs)

Friends

- class InputProcessor
- void init_attrjeod__ContactPair ()

8.4.1 Detailed Description

An base contact pair class for use in the contact model.

Definition at line 49 of file contact_pair.hh.

8.4.2 Constructor & Destructor Documentation

8.4.2.1 jeod::ContactPair::ContactPair (void)

Default Constructor.

Definition at line 45 of file contact_pair.cc.

8.4.2.2 jeod::ContactPair::∼ContactPair (void) [virtual]

Destructor.

Definition at line 60 of file contact_pair.cc.

8.4.2.3 jeod::ContactPair::ContactPair (const ContactPair & *rhs*) [private]

8.4.3 Member Function Documentation

8.4.3.1 bool jeod::ContactPair::check_tree (void) [virtual]

Make sure the two contact facets are not on the same mass tree.

Returns

bool

Definition at line 136 of file contact_pair.cc.

References subject, and target.

Referenced by is_active().

8.4.3.2 ContactFacet ∗ jeod::ContactPair::get_subject (void)

Determine if the pair has a target facet.

Returns

subject ContactFacet

Definition at line 114 of file contact_pair.cc.

References subject.

Referenced by jeod::Contact::initialize_contact().

8.4.3.3 ContactFacet ∗ jeod::ContactPair::get_target (void)

Determine if the pair has a target facet.

Returns

target ContactFacet

Definition at line 125 of file contact_pair.cc.

References target.

8.4.3.4 virtual void jeod::ContactPair::in_contact (void) [pure virtual]

Virtual funtion to determine if the pair is in contact.

Contact depends on specific geometry so implementation has to wait for a derived class.

Implemented in jeod::LinePointContactPair, jeod::LineContactPair, and jeod::PointContactPair.

8.4.3.5 bool jeod::ContactPair::in_range (void)

test whether the pair is in range for interaction

Returns

bool

Definition at line 71 of file contact_pair.cc.

References interaction_distance, and rel_state.

8.4.3.6 virtual void jeod::ContactPair::initialize_pair (ContactFacet ∗ *subject_facet*, ContactFacet ∗ *target_facet*) [pure virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Parameters

in,out	*subject_facet*	subject ContactFacet
in,out	*target_facet*	target ContactFacet

Implemented in jeod::LinePointContactPair, jeod::LineContactPair, and jeod::PointContactPair.

8.4.3.7 void jeod::ContactPair::initialize_relstate (DynManager ∗ *dyn_manager*) [virtual]

Initialize the relative state between the facets and register with the dynamics manager.

Initialize the relstate using the DynManager provided by Contact class.

Parameters

in	*dyn_manager*	dynamics manager

Definition at line 158 of file contact_pair.cc.

References rel_state, subject, and jeod::ContactFacet::vehicle_body.

Referenced by jeod::Contact::initialize_contact().

8.4.3.8 bool jeod::ContactPair::is_active (void)

Determine if contact can occur between the two facets.

Returns

bool

Definition at line 86 of file contact_pair.cc.

References jeod::ContactFacet::active, check_tree(), subject, and target.

8.4.3.9 bool jeod::ContactPair::is_complete (void)

Determine if the pair has a target facet.

Returns

bool

Definition at line 100 of file contact_pair.cc.

References target.

8.4.3.10 ContactPair& jeod::ContactPair::operator= (const ContactPair & *rhs*) [private]

8.4.4 Friends And Related Function Documentation

8.4.4.1 void init_attrjeod__ContactPair ( ) [friend]

8.4.4.2 friend class InputProcessor [friend]

Definition at line 51 of file contact_pair.hh.

8.4.5 Field Documentation

8.4.5.1 PairInteraction* jeod::ContactPair::interaction

Parameters that define the force calculation function between the subjec and target.

trick_units(–)

Definition at line 57 of file contact_pair.hh.

Referenced by jeod::LineContactFacet::create_pair(), jeod::PointContactFacet::create_pair(), jeod::LineContactPair::in_contact(), jeod::PointContactPair::in_contact(), and jeod::LinePointContactPair::in_contact().

8.4.5.2 double jeod::ContactPair::interaction_distance

rel_state distance at which in_contact should be called

trick_units(m)

Definition at line 62 of file contact_pair.hh.

Referenced by jeod::LineContactFacet::create_pair(), jeod::PointContactFacet::create_pair(), and in_range().

8.4.5.3 RelativeDerivedState jeod::ContactPair::rel_state [protected]

Current relative state between the subject and the target in the subject frame.

trick_units(–)

Definition at line 119 of file contact_pair.hh.

Referenced by jeod::LineContactPair::in_contact(), jeod::PointContactPair::in_contact(), jeod::LinePointContactPair::in_contact(), in_range(), jeod::LineContactPair::initialize_pair(), jeod::PointContactPair::initialize_pair(), jeod::LinePointContactPair::initialize_pair(), and initialize_relstate().

8.4.5.4 ContactFacet∗ jeod::ContactPair::subject [protected]

pointer to the contact facet that is the subject of the associated relative states.

trick_units(–)

Definition at line 124 of file contact_pair.hh.

Referenced by check_tree(), get_subject(), jeod::LineContactPair::initialize_pair(), jeod::PointContactPair::initialize_pair(), jeod::LinePointContactPair::initialize_pair(), initialize_relstate(), and is_active().

8.4.5.5 ContactFacet∗ jeod::ContactPair::target [protected]

pointer to the contact facet that is the target of the associated relative states.

trick_units(–)

Definition at line 129 of file contact_pair.hh.

Referenced by check_tree(), get_target(), jeod::LineContactPair::initialize_pair(), jeod::PointContactPair::initialize_pair(), jeod::LinePointContactPair::initialize_pair(), is_active(), and is_complete().

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

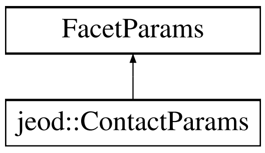
- contact_pair.hh
- contact_pair.cc

8.5 jeod::ContactParams Class Reference

A base class for all contact parameters used in the surface model.

`#include <contact_params.hh>`

Inheritance diagram for jeod::ContactParams:



Public Member Functions

- ContactParams ()

    Default Constructor.
- virtual ∼ContactParams ()

    Destructor.

Private Member Functions

- ContactParams & operator= (const ContactParams &rhs)
- ContactParams (const ContactParams &rhs)

Friends

- class InputProcessor
- void init_attrjeod__ContactParams ()

8.5.1 Detailed Description

A base class for all contact parameters used in the surface model.

Definition at line 46 of file contact_params.hh.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 jeod::ContactParams::ContactParams (void)

Default Constructor.

Definition at line 43 of file contact_params.cc.

8.5.2.2 jeod::ContactParams::∼ContactParams (void) `[virtual]`

Destructor.

Definition at line 54 of file contact_params.cc.

8.5.2.3 jeod::ContactParams::ContactParams (const **ContactParams** & *rhs*) `[private]`

8.5.3 Member Function Documentation

8.5.3.1 **ContactParams**& jeod::ContactParams::operator= (const **ContactParams** & *rhs*) `[private]`

8.5.4 Friends And Related Function Documentation

8.5.4.1 void init_attrjeod__ContactParams ( ) `[friend]`

8.5.4.2 friend class InputProcessor `[friend]`

Definition at line 49 of file contact_params.hh.

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

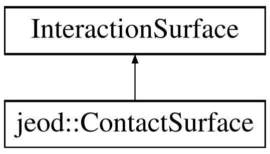
- contact_params.hh
- contact_params.cc

8.6 jeod::ContactSurface Class Reference

The contact specific interaction surface, for use with the surface model.

`#include <contact_surface.hh>`

Inheritance diagram for jeod::ContactSurface:



Public Member Functions

- ContactSurface ()

  Default Constructor.
- virtual ~ContactSurface ()

  Destructor.
- virtual void allocate_array (unsigned int size)

  Allocates an array of ContactFacet pointers, of the size indicated by the input variable.
- virtual void allocate_interaction_facet (Facet ∗facet, InteractionFacetFactory ∗factory, FacetParams ∗params, unsigned int index)

  Creates an interaction facet or more accurately a contact facet from a basic facet and set of parameters.
- virtual void collect_forces_torques (void)

  collect the forces and torques from all the facets in this contact surface

Data Fields

- ContactFacet ∗∗ contact_facets

  An array of pointers to contact interaction facets.
- unsigned int facets_size

  Size of the contact_facets array.
- double contact_force [3]

  Total Force due to contact, resulting from all plates combined.
- double contact_torque [3]

  Total Torque due to contact, resulting from all plates combined.

Private Member Functions

- ContactSurface & operator= (const ContactSurface &rhs)
- ContactSurface (const ContactSurface &rhs)

Friends

- class InputProcessor
- void init_attrjeod__ContactSurface ()

8.6.1 Detailed Description

The contact specific interaction surface, for use with the surface model.

Definition at line 49 of file contact_surface.hh.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 jeod::ContactSurface::ContactSurface (void)

Default Constructor.

Definition at line 59 of file contact_surface.cc.

8.6.2.2 jeod::ContactSurface::∼ContactSurface (void) [virtual]

Destructor.

Definition at line 73 of file contact_surface.cc.

References contact_facets, and facets_size.

8.6.2.3 jeod::ContactSurface::ContactSurface (const ContactSurface & *rhs*) [private]

8.6.3 Member Function Documentation

8.6.3.1 void jeod::ContactSurface::allocate_array (unsigned int *size*) [virtual]

Allocates an array of ContactFacet pointers, of the size indicated by the input variable.

Parameters

in	*size*	The size of the needed array Units: cnt:

Definition at line 99 of file contact_surface.cc.

References contact_facets, facets_size, and jeod::ContactMessages::initialization_error.

8.6.3.2 void jeod::ContactSurface::allocate_interaction_facet (Facet ∗ *facet*, InteractionFacetFactory ∗ *factory*, FacetParams ∗ *params*, unsigned int *index*) [virtual]

Creates an interaction facet or more accurately a contact facet from a basic facet and set of parameters.

Parameters

in	*facet*	The basic facet used to create the interaction facet
in	*factory*	The factory used to create the interaction facet
in	*params*	The contact params used to create the interaction facet
in	*index*	Where the new interaction facet will be placed in the contact_facets array Units: cnt

Definition at line 137 of file contact_surface.cc.

References contact_facets, facets_size, and jeod::ContactMessages::initialization_error.

8.6.3.3 void jeod::ContactSurface::collect_forces_torques (void) [virtual]

collect the forces and torques from all the facets in this contact surface

Definition at line 214 of file contact_surface.cc.

References contact_facets, contact_force, contact_torque, and facets_size.

8.6.3.4 ContactSurface& jeod::ContactSurface::operator= (const ContactSurface & *rhs*) [private]

8.6.4 Friends And Related Function Documentation

8.6.4.1 void init_attrjeod__ContactSurface ( ) `[friend]`

8.6.4.2 friend class InputProcessor `[friend]`

Definition at line 52 of file contact_surface.hh.

8.6.5 Field Documentation

8.6.5.1 ContactFacet** jeod::ContactSurface::contact_facets

An array of pointers to contact interaction facets.

trick_units(–)

Definition at line 65 of file contact_surface.hh.

Referenced by allocate_array(), allocate_interaction_facet(), collect_forces_torques(), and ∼ContactSurface().

8.6.5.2 double jeod::ContactSurface::contact_force[3]

Total Force due to contact, resulting from all plates combined.

trick_units(N)

Definition at line 75 of file contact_surface.hh.

Referenced by collect_forces_torques().

8.6.5.3 double jeod::ContactSurface::contact_torque[3]

Total Torque due to contact, resulting from all plates combined.

trick_units(N/m)

Definition at line 80 of file contact_surface.hh.

Referenced by collect_forces_torques().

8.6.5.4 unsigned int jeod::ContactSurface::facets_size

Size of the contact_facets array.

trick_units(count)

Definition at line 70 of file contact_surface.hh.

Referenced by allocate_array(), allocate_interaction_facet(), collect_forces_torques(), and ∼ContactSurface().

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

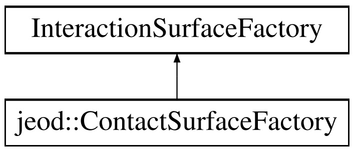
- contact_surface.hh
- contact_surface.cc

8.7 jeod::ContactSurfaceFactory Class Reference

The surface factory that creates an contact specific surface from a general surface.

`#include <contact_surface_factory.hh>`

Inheritance diagram for jeod::ContactSurfaceFactory:



Public Member Functions

- ContactSurfaceFactory ()

  Default Constructor.
- virtual ∼ContactSurfaceFactory ()

  Destructor.
- virtual void create_surface (SurfaceModel ∗surface, InteractionSurface ∗inter_surface)

  Creates an interaction surface, in the inter_surface parameter, from the given SurfaceModel.
- virtual void add_facet_params (FacetParams ∗to_add)

  Add a named set of facet params to the surface factory.

Protected Attributes

- PointContactFacetFactory point_contact_facet_factory

  A factory that can create a point contact facet from a circular flat plate.
- LineContactFacetFactory line_contact_facet_factory

  A factory that can create a line contact facet from a cylinder.

Private Member Functions

- ContactSurfaceFactory & operator= (const ContactSurfaceFactory &rhs)
- ContactSurfaceFactory (const ContactSurfaceFactory &rhs)

Friends

- class InputProcessor
- void init_attrjeod__ContactSurfaceFactory ()

8.7.1 Detailed Description

The surface factory that creates an contact specific surface from a general surface.

Used with the surface model.

Definition at line 51 of file contact_surface_factory.hh.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 jeod::ContactSurfaceFactory::ContactSurfaceFactory (void)

Default Constructor.

Definition at line 51 of file contact_surface_factory.cc.

References line_contact_facet_factory, and point_contact_facet_factory.

8.7.2.2 jeod::ContactSurfaceFactory::∼ContactSurfaceFactory (void) [virtual]

Destructor.

Definition at line 65 of file contact_surface_factory.cc.

8.7.2.3 jeod::ContactSurfaceFactory::ContactSurfaceFactory (const ContactSurfaceFactory & *rhs*) [private]

8.7.3 Member Function Documentation

8.7.3.1 void jeod::ContactSurfaceFactory::add_facet_params (FacetParams ∗ *to_add*) [virtual]

Add a named set of facet params to the surface factory.

Intended to be used when an contact specific surface is created, to convert a basic facet to an contact interaction facet. This MUST be a parameter inheriting from ContactParam, or the function will fail and send a failure message

Parameters

in	*to_add*	The facet parameters to add

Definition at line 203 of file contact_surface_factory.cc.

References jeod::ContactMessages::pre_initialization_error.

8.7.3.2 void jeod::ContactSurfaceFactory::create_surface (SurfaceModel ∗ *surface,* InteractionSurface ∗ *inter_surface*) [virtual]

Creates an interaction surface, in the inter_surface parameter, from the given SurfaceModel.

The InteractionSurfaceFactory should contain all necessary InteractionFacetFactories and FacetParams already

Parameters

in	*surface*	The surface model used to create the interaction surface
out	*inter_surface*	Where the interaction surface will be produced

Definition at line 84 of file contact_surface_factory.cc.

References jeod::ContactMessages::initialization_error.

8.7.3.3 ContactSurfaceFactory& jeod::ContactSurfaceFactory::operator= (const ContactSurfaceFactory & *rhs*) [private]

8.7.4 Friends And Related Function Documentation

8.7.4.1 void init_attrjeod__ContactSurfaceFactory ( ) [friend]

8.7.4.2 friend class InputProcessor [friend]

Definition at line 54 of file contact_surface_factory.hh.

8.7.5 Field Documentation

8.7.5.1 LineContactFacetFactory jeod::ContactSurfaceFactory::line_contact_facet_factory [protected]

A factory that can create a line contact facet from a cylinder.

trick_units(–)

Definition at line 87 of file contact_surface_factory.hh.

Referenced by ContactSurfaceFactory().

8.7.5.2 PointContactFacetFactory jeod::ContactSurfaceFactory::point_contact_facet_factory `[protected]`

A factory that can create a point contact facet from a circular flat plate.

trick_units(–)

Definition at line 82 of file contact_surface_factory.hh.

Referenced by ContactSurfaceFactory().

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

- contact_surface_factory.hh
- contact_surface_factory.cc

8.8 jeod::ContactUtils Class Reference

Utility string and math functions for the contact model.

`#include <contact_utils.hh>`

Static Public Member Functions

- static int create_relstate_name (char ∗name1, char ∗name2, char ∗∗out_str)

  create a name for a relstate out of two facet names
- static int copy_const_char_to_char (const char ∗in_str, char ∗∗out_str)

  create a name for a relstate out of two facet names
- static int dist_line_segments (double p1[3], double p2[3], double p3[3], double p4[3], double ∗pa, double ∗pb)

  calculate the closest points between two line segments

8.8.1 Detailed Description

Utility string and math functions for the contact model.

Definition at line 36 of file contact_utils.hh.

8.8.2 Member Function Documentation

8.8.2.1 int jeod::ContactUtils::copy_const_char_to_char (const char ∗ *in_str,* char ∗∗ *out_str*) `[inline], [static]`

create a name for a relstate out of two facet names

Returns

int

Parameters

`in`	*in_str*	const char input
`in,out`	*out_str*	char output

Definition at line 88 of file contact_utils_inline.hh.

Referenced by jeod::LineContactPair::initialize_pair(), jeod::PointContactPair::initialize_pair(), and jeod::LinePoint-ContactPair::initialize_pair().

8.8.2.2 int jeod::ContactUtils::create_relstate_name (char ∗ *name1,* char ∗ *name2,* char ∗∗ *out_str*) `[inline]`, `[static]`

create a name for a relstate out of two facet names

Returns

char∗∗

Parameters

`in`	*name1*	name of first contact facet
`in`	*name2*	name of second contact facet
`in,out`	*out_str*	output name for the relstate

Definition at line 60 of file contact_utils_inline.hh.

8.8.2.3 int jeod::ContactUtils::dist_line_segments (double *p1[3],* double *p2[3],* double *p3[3],* double *p4[3],* double ∗ *pa,* double ∗ *pb*) `[inline]`, `[static]`

calculate the closest points between two line segments

Returns

success

Parameters

`in`	*p1*	vector to one point of the first line seg Units: M
`in`	*p2*	vector to one point of the first line seg Units: M
`in`	*p3*	vector to one point of the second line seg Units: M
`in`	*p4*	vector to one point of the second line seg Units: M
`out`	*pa*	vector to close_point on line 1 Units: M
`out`	*pb*	vector to close_point on line 2 Units: M

Definition at line 122 of file contact_utils_inline.hh.

Referenced by jeod::LineContactPair::in_contact(), and jeod::LinePointContactPair::in_contact().

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

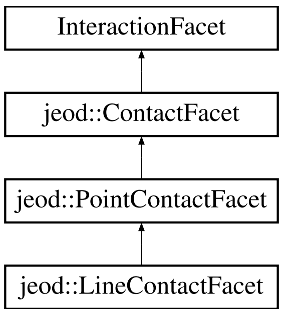
- contact_utils.hh
- contact_utils_inline.hh

8.9 jeod::LineContactFacet Class Reference

The contact facet based on the distance to a single point, specifically the vehicle point.

```
#include <line_contact_facet.hh>
```

Inheritance diagram for jeod::LineContactFacet:



Public Member Functions

- LineContactFacet ()

  Default constructor.
- virtual ∼LineContactFacet ()

  Destructor.
- virtual ContactPair ∗ create_pair ()

  Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.
- virtual ContactPair ∗ create_pair (ContactFacet ∗target, Contact ∗contact)

  Overloaded functions that creates a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.
- virtual void set_max_dimension ()

  calculate the max dimension of the facet for range limit determination.
- virtual void calculate_torque (double ∗tmp_force)

  Calculate the torque generated on the vehicle by the facet.
- virtual void calculate_contact_point (double nvec[3])

  Find the point on the surface that coorisponds to the closest point on line segments using the radius value.

Data Fields

- double length

  length of the line along the vehicle point x axis.

Private Member Functions

- LineContactFacet & operator= (const LineContactFacet &rhs)
- LineContactFacet (const LineContactFacet &rhs)

Friends

- class InputProcessor
- void init_attrjeod__LineContactFacet ()

8.9.1 Detailed Description

The contact facet based on the distance to a single point, specifically the vehicle point.

In effect this represents a sphere.

Definition at line 51 of file line_contact_facet.hh.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 jeod::LineContactFacet::LineContactFacet (void)

Default constructor.

Definition at line 53 of file line_contact_facet.cc.

References jeod::PointContactFacet::contact_point.

8.9.2.2 jeod::LineContactFacet::∼LineContactFacet (void) `[virtual]`

Destructor.

Definition at line 67 of file line_contact_facet.cc.

8.9.2.3 jeod::LineContactFacet::LineContactFacet (const LineContactFacet & *rhs*) `[private]`

8.9.3 Member Function Documentation

8.9.3.1 void jeod::LineContactFacet::calculate_contact_point (double *nvec[3]*) `[virtual]`

Find the point on the surface that coorisponds to the closest point on line segments using the radius value.

Parameters

`in`	*nvec*	vector between line points

Reimplemented from jeod::PointContactFacet.

Definition at line 173 of file line_contact_facet.cc.

References jeod::PointContactFacet::contact_point, length, and jeod::PointContactFacet::radius.

Referenced by jeod::LineContactPair::in_contact(), and jeod::LinePointContactPair::in_contact().

8.9.3.2 void jeod::LineContactFacet::calculate_torque (double ∗ *tmp_force*) `[virtual]`

Calculate the torque generated on the vehicle by the facet.

Assumes that the force is in the vehicle structural frame, but that close_point is not.

Parameters

`in`	*tmp_force*	force from one contact interaction. Units: N

Reimplemented from jeod::PointContactFacet.

Definition at line 231 of file line_contact_facet.cc.

References jeod::PointContactFacet::contact_point, jeod::ContactFacet::vehicle_body, and jeod::ContactFacet::vehicle_point.

8.9.3.3 ContactPair ∗ jeod::LineContactFacet::create_pair (void) [virtual]

Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

This function is called to create a pair that only contains a subject.

Returns

ContactPair that was created

Reimplemented from jeod::PointContactFacet.

Definition at line 83 of file line_contact_facet.cc.

References jeod::LineContactPair::initialize_pair().

8.9.3.4 ContactPair ∗ jeod::LineContactFacet::create_pair (ContactFacet ∗ *target*, Contact ∗ *contact*) [virtual]

Overloaded functions that creates a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

This function is called when a subject and target are known.

Returns

ContactPair that was created

Parameters

in,out	*target*	target ContactFacet
in	*contact*	Contact object used to find the pair interaction

Reimplemented from jeod::PointContactFacet.

Definition at line 104 of file line_contact_facet.cc.

References jeod::Contact::contact_limit_factor, jeod::Contact::find_interaction(), jeod::ContactMessages::initialization-_warns, jeod::LineContactPair::initialize_pair(), jeod::LinePointContactPair::initialize_pair(), jeod::ContactPair-::interaction, jeod::ContactPair::interaction_distance, jeod::ContactFacet::max_dimension, and jeod::ContactFacet-::surface_type.

8.9.3.5 LineContactFacet& jeod::LineContactFacet::operator= (const LineContactFacet & *rhs*) [private]

8.9.3.6 void jeod::LineContactFacet::set_max_dimension (void) [virtual]

calculate the max dimension of the facet for range limit determination.

Reimplemented from jeod::PointContactFacet.

Definition at line 218 of file line_contact_facet.cc.

References length, jeod::ContactFacet::max_dimension, and jeod::PointContactFacet::radius.

Referenced by jeod::LineContactFacetFactory::create_facet().

8.9.4 Friends And Related Function Documentation

8.9.4.1 void init_attrjeod__LineContactFacet ( ) [friend]

8.9.4.2 friend class InputProcessor [friend]

Definition at line 53 of file line_contact_facet.hh.

8.9.5 Field Documentation

8.9.5.1 double jeod::LineContactFacet::length

length of the line along the vehicle point x axis.

trick_units(m)

Definition at line 59 of file line_contact_facet.hh.

Referenced by calculate_contact_point(), jeod::LineContactFacetFactory::create_facet(), jeod::LineContactPair::in-_contact(), jeod::LinePointContactPair::in_contact(), and set_max_dimension().

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

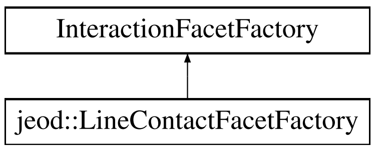
- line_contact_facet.hh
- line_contact_facet.cc

8.10 jeod::LineContactFacetFactory Class Reference

Creates a PointContactFacet from an InteractionFacet.

`#include <line_contact_facet_factory.hh>`

Inheritance diagram for jeod::LineContactFacetFactory:



Public Member Functions

- LineContactFacetFactory ()

    Default Constructor.
- ∼LineContactFacetFactory ()

    Destructor.
- virtual InteractionFacet ∗ create_facet (Facet ∗facet, FacetParams ∗params)

    Create a PointContactFacet from a CircularFlatPlate facet and a ContactParams object.
- virtual bool is_correct_factory (Facet ∗facet)

    PointContactFacetFactory specific implementation of this function.

Private Member Functions

- LineContactFacetFactory & operator= (const LineContactFacetFactory &rhs)
- LineContactFacetFactory (const LineContactFacetFactory &rhs)

Friends

- class InputProcessor
- void init_attrjeod__LineContactFacetFactory ()

8.10.1 Detailed Description

Creates a PointContactFacet from an InteractionFacet.

Definition at line 50 of file line_contact_facet_factory.hh.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 jeod::LineContactFacetFactory::LineContactFacetFactory (void)

Default Constructor.

Definition at line 54 of file line_contact_facet_factory.cc.

8.10.2.2 jeod::LineContactFacetFactory::∼LineContactFacetFactory (void)

Destructor.

Definition at line 65 of file line_contact_facet_factory.cc.

8.10.2.3 jeod::LineContactFacetFactory::LineContactFacetFactory (const LineContactFacetFactory & *rhs*) `[private]`

8.10.3 Member Function Documentation

8.10.3.1 InteractionFacet ∗ jeod::LineContactFacetFactory::create_facet (Facet ∗ *facet*, FacetParams ∗ *params*) `[virtual]`

Create a PointContactFacet from a CircularFlatPlate facet and a ContactParams object.

Returns

The new EllipsoidContactFacet. Note that this is allocated and YOU are responsible for destroying it at the end!

Parameters

`in`	*facet*	The CircularFlatPlate. This MUST be a circular flat plate or the algorithm will send a failure message
`in`	*params*	ContactParams

Definition at line 83 of file line_contact_facet_factory.cc.

References jeod::ContactFacet::create_vehicle_point(), jeod::ContactMessages::initialization_error, jeod::LineContactFacet::length, jeod::ContactFacet::normal, jeod::ContactFacet::position, jeod::PointContactFacet::radius, jeod::LineContactFacet::set_max_dimension(), jeod::ContactFacet::surface_type, and jeod::ContactFacet::vehicle_body.

8.10.3.2 bool jeod::LineContactFacetFactory::is_correct_factory (Facet ∗ *facet*) `[virtual]`

PointContactFacetFactory specific implementation of this function.

If the Facet is of type CircularFlatPlate, returns true. False otherwise

Returns

true if facet is a FlatPlateCircular, false otherwise

Parameters

in	*facet*	The facet to check

Definition at line 162 of file line_contact_facet_factory.cc.

8.10.3.3 LineContactFacetFactory& jeod::LineContactFacetFactory::operator= (const LineContactFacetFactory & *rhs*) [private]

8.10.4 Friends And Related Function Documentation

8.10.4.1 void init_attrjeod__LineContactFacetFactory ( ) [friend]

8.10.4.2 friend class InputProcessor [friend]

Definition at line 53 of file line_contact_facet_factory.hh.

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

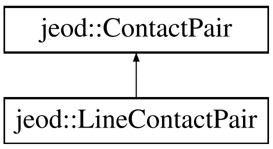
- line_contact_facet_factory.hh
- line_contact_facet_factory.cc

8.11 jeod::LineContactPair Class Reference

An point to point contact pair for use in the contact model.

`#include <line_contact_pair.hh>`

Inheritance diagram for jeod::LineContactPair:



Public Member Functions

- LineContactPair ()

  Default Constructor.
- virtual ~LineContactPair ()

  Destructor.
- virtual void in_contact ()

  Determine if contact has occurred between the facets of the pair.
- virtual void initialize_pair (ContactFacet ∗subject_facet, ContactFacet ∗target_facet)

  Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Data Fields

- LineContactFacet ∗ line_subject

  pointer to the contact facet that is the subject of the associated relative states.
- LineContactFacet ∗ line_target

  pointer to the contact facet that is the target of the associated relative states.

Private Member Functions

- LineContactPair & operator= (const LineContactPair &rhs)
- LineContactPair (const LineContactPair &rhs)

Friends

- class InputProcessor
- void init_attrjeod__LineContactPair ()

Additional Inherited Members

8.11.1 Detailed Description

An point to point contact pair for use in the contact model.

Definition at line 48 of file line_contact_pair.hh.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 jeod::LineContactPair::LineContactPair (void)

Default Constructor.

Definition at line 48 of file line_contact_pair.cc.

8.11.2.2 jeod::LineContactPair::∼LineContactPair (void) [virtual]

Destructor.

Definition at line 61 of file line_contact_pair.cc.

8.11.2.3 jeod::LineContactPair::LineContactPair (const **LineContactPair** & *rhs*) [private]

8.11.3 Member Function Documentation

8.11.3.1 void jeod::LineContactPair::in_contact (void) [virtual]

Determine if contact has occurred between the facets of the pair.

Implements jeod::ContactPair.

Definition at line 71 of file line_contact_pair.cc.

References jeod::LineContactFacet::calculate_contact_point(), jeod::PairInteraction::calculate_forces(), jeod::PointContactFacet::contact_point, jeod::ContactUtils::dist_line_segments(), jeod::ContactPair::interaction, jeod::LineContactFacet::length, line_subject, line_target, and jeod::ContactPair::rel_state.

8.11.3.2 void jeod::LineContactPair::initialize_pair (**ContactFacet** ∗ *subject_facet,* **ContactFacet** ∗ *target_facet*) [virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Parameters

`in,out`	*subject_facet*	subject ContactFacet
`in,out`	*target_facet*	target ContactFacet

Implements jeod::ContactPair.

Definition at line 173 of file line_contact_pair.cc.

References jeod::ContactUtils::copy_const_char_to_char(), line_subject, line_target, jeod::ContactPair::rel_state, jeod::ContactPair::subject, jeod::ContactPair::target, and jeod::ContactFacet::vehicle_point.

Referenced by jeod::LineContactFacet::create_pair().

8.11.3.3 LineContactPair& jeod::LineContactPair::operator= (const LineContactPair & *rhs*) `[private]`

8.11.4 Friends And Related Function Documentation

8.11.4.1 void init_attrjeod__LineContactPair ( ) `[friend]`

8.11.4.2 friend class InputProcessor `[friend]`

Definition at line 50 of file line_contact_pair.hh.

8.11.5 Field Documentation

8.11.5.1 LineContactFacet∗ jeod::LineContactPair::line_subject

pointer to the contact facet that is the subject of the associated relative states.

trick_units(–)

Definition at line 56 of file line_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

8.11.5.2 LineContactFacet∗ jeod::LineContactPair::line_target

pointer to the contact facet that is the target of the associated relative states.

trick_units(–)

Definition at line 61 of file line_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

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

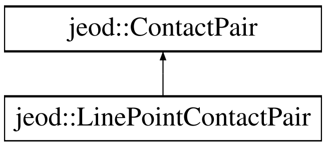
- line_contact_pair.hh
- line_contact_pair.cc

8.12 jeod::LinePointContactPair Class Reference

An point to point contact pair for use in the contact model.

`#include <line_point_contact_pair.hh>`

Inheritance diagram for jeod::LinePointContactPair:



Public Member Functions

- LinePointContactPair ()

  Default Constructor.
- virtual ∼LinePointContactPair ()

  Destructor.
- virtual void in_contact ()

  Determine if contact has occurred between the facets of the pair.
- virtual void initialize_pair (ContactFacet ∗subject_facet, ContactFacet ∗target_facet)

  Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Data Fields

- LineContactFacet ∗ line_subject

  pointer to the contact facet that is the subject of the associated relative states.
- PointContactFacet ∗ point_target

  pointer to the contact facet that is the target of the associated relative states.

Private Member Functions

- LinePointContactPair & operator= (const LineContactPair &rhs)
- LinePointContactPair (const LineContactPair &rhs)

Friends

- class InputProcessor
- void init_attrjeod__LinePointContactPair ()

Additional Inherited Members

8.12.1 Detailed Description

An point to point contact pair for use in the contact model.

Definition at line 49 of file line_point_contact_pair.hh.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 jeod::LinePointContactPair::LinePointContactPair (void)

Default Constructor.

Definition at line 49 of file line_point_contact_pair.cc.

8.12.2.2 jeod::LinePointContactPair::∼LinePointContactPair (void) [virtual]

Destructor.

Definition at line 62 of file line_point_contact_pair.cc.

8.12.2.3 jeod::LinePointContactPair::LinePointContactPair (const **LineContactPair** & *rhs*) [private]

8.12.3 Member Function Documentation

8.12.3.1 void jeod::LinePointContactPair::in_contact (void) [virtual]

Determine if contact has occurred between the facets of the pair.

Implements jeod::ContactPair.

Definition at line 72 of file line_point_contact_pair.cc.

References jeod::LineContactFacet::calculate_contact_point(), jeod::PointContactFacet::calculate_contact_point(), jeod::PairInteraction::calculate_forces(), jeod::PointContactFacet::contact_point, jeod::ContactUtils::dist_line_-segments(), jeod::ContactPair::interaction, jeod::LineContactFacet::length, line_subject, point_target, and jeod-::ContactPair::rel_state.

8.12.3.2 void jeod::LinePointContactPair::initialize_pair (**ContactFacet** ∗ *subject_facet,* **ContactFacet** ∗ *target_facet*) [virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Parameters

in,out	*subject_facet*	subject ContactFacet
in,out	*target_facet*	target ContactFacet

Implements jeod::ContactPair.

Definition at line 158 of file line_point_contact_pair.cc.

References jeod::ContactUtils::copy_const_char_to_char(), line_subject, point_target, jeod::ContactPair::rel_state, jeod::ContactPair::subject, jeod::ContactPair::target, and jeod::ContactFacet::vehicle_point.

Referenced by jeod::LineContactFacet::create_pair().

8.12.3.3 **LinePointContactPair**& jeod::LinePointContactPair::operator= (const **LineContactPair** & *rhs*) [private]

8.12.4 Friends And Related Function Documentation

8.12.4.1 void init_attrjeod__LinePointContactPair ( ) [friend]

8.12.4.2 friend class InputProcessor [friend]

Definition at line 51 of file line_point_contact_pair.hh.

8.12.5 Field Documentation

8.12.5.1 **LineContactFacet**∗ jeod::LinePointContactPair::line_subject

pointer to the contact facet that is the subject of the associated relative states.

trick_units(–)

Definition at line 57 of file line_point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

8.12.5.2 PointContactFacet∗ jeod::LinePointContactPair::point_target

pointer to the contact facet that is the target of the associated relative states.

trick_units(–)

Definition at line 62 of file line_point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

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

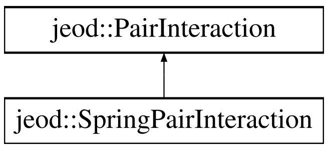
- line_point_contact_pair.hh
- line_point_contact_pair.cc

8.13 jeod::PairInteraction Class Reference

Simple spring contact parameters.

`#include <pair_interaction.hh>`

Inheritance diagram for jeod::PairInteraction:



Public Member Functions

- PairInteraction ()

  Default Constructor.
- virtual ∼PairInteraction ()

  Destructor.
- bool is_correct_interaction (ContactParams ∗subject_params, ContactParams ∗target_params)

  Check a pair of contact params for a match to the ones defined for this pair_interaction.
- virtual void calculate_forces (ContactFacet ∗subject, ContactFacet ∗target, RelativeDerivedState ∗rel_state, double ∗penetration_vector, double ∗rel_velocity)=0

  Pure virtual function that is defined to calculate forces on facets in contact.

Data Fields

- char ∗ params_1

  contact param type that defines this pair interaction.
- char ∗ params_2

  contact param type that defines this pair interaction.
- double friction_mag

  magnitude of the friction force on the contact surfaces.

Private Member Functions

- PairInteraction & operator= (const PairInteraction &rhs)
- PairInteraction (const PairInteraction &rhs)

Friends

- class InputProcessor
- void init_attrjeod__PairInteraction ()

8.13.1 Detailed Description

Simple spring contact parameters.

Definition at line 50 of file pair_interaction.hh.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 jeod::PairInteraction::PairInteraction (void)

Default Constructor.

Definition at line 47 of file pair_interaction.cc.

8.13.2.2 jeod::PairInteraction::∼PairInteraction (void) [virtual]

Destructor.

Definition at line 61 of file pair_interaction.cc.

8.13.2.3 jeod::PairInteraction::PairInteraction (const **PairInteraction** & *rhs*) [private]

8.13.3 Member Function Documentation

8.13.3.1 virtual void jeod::PairInteraction::calculate_forces (**ContactFacet** ∗ *subject,* **ContactFacet** ∗ *target,* RelativeDerivedState ∗ *rel_state,* double ∗ *penetration_vector,* double ∗ *rel_velocity*) [pure virtual]

Pure virtual function that is defined to calculate forces on facets in contact.

Parameters

`in,out`	*subject*	subject of the relative state
`in,out`	*target*	target of the relative state
`in`	*rel_state*	relative state between subject and target in subject frame
`in`	*penetration_-vector*	vector that characterises the interpenetration of the subject and the target
`in`	*rel_velocity*	relative velocity of the subject and the target in the subject frame

Implemented in jeod::SpringPairInteraction.

Referenced by jeod::LineContactPair::in_contact(), jeod::PointContactPair::in_contact(), and jeod::LinePoint-ContactPair::in_contact().

8.13.3.2 bool jeod::PairInteraction::is_correct_interaction (**ContactParams** ∗ *subject_params,* **ContactParams** ∗ *target_params*)

Check a pair of contact params for a match to the ones defined for this pair_interaction.

Returns

bool

Parameters

in	*subject_params*	parameters of the subject
in	*target_params*	parameters of the target

Definition at line 75 of file pair_interaction.cc.

References params_1, and params_2.

8.13.3.3 PairInteraction& jeod::PairInteraction::operator= (const PairInteraction & *rhs*) [private]

8.13.4 Friends And Related Function Documentation

8.13.4.1 void init_attrjeod__PairInteraction ( ) [friend]

8.13.4.2 friend class InputProcessor [friend]

Definition at line 52 of file pair_interaction.hh.

8.13.5 Field Documentation

8.13.5.1 double jeod::PairInteraction::friction_mag

magnitude of the friction force on the contact surfaces.

trick_units(N)

Definition at line 68 of file pair_interaction.hh.

Referenced by jeod::SpringPairInteraction::calculate_forces().

8.13.5.2 char* jeod::PairInteraction::params_1

contact param type that defines this pair interaction.

trick_units(–)

Definition at line 59 of file pair_interaction.hh.

Referenced by is_correct_interaction().

8.13.5.3 char* jeod::PairInteraction::params_2

contact param type that defines this pair interaction.

trick_units(–)

Definition at line 63 of file pair_interaction.hh.

Referenced by is_correct_interaction().

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

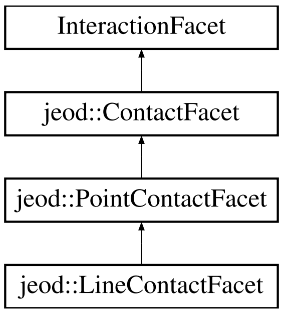
- pair_interaction.hh
- pair_interaction.cc

8.14 jeod::PointContactFacet Class Reference

The contact facet based on the distance to a single point, specifically the vehicle point.

`#include <point_contact_facet.hh>`

Inheritance diagram for jeod::PointContactFacet:



Public Member Functions

- PointContactFacet ()

  Default constructor.
- virtual ∼PointContactFacet ()

  Destructor.
- virtual ContactPair ∗ create_pair ()

  Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.
- virtual ContactPair ∗ create_pair (ContactFacet ∗target, Contact ∗contact)

  Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.
- virtual void set_max_dimension ()

  calculate the max dimension of the facet for range limit determination.
- virtual void calculate_contact_point (double nvec[3])

  Use the relstate and radius of contact to calculate a contact point on this facet.
- virtual void calculate_torque (double ∗tmp_force)

  Calculate the torque generated on the vehicle by the facet.

Data Fields

- double radius

  radius from the point at which contact takes place.
- double contact_point [3]

  Contact point given in facet vehicle point frame, representing point on the point on the surface of a sphere of radius "radius" where contact has occured.

Private Member Functions

- PointContactFacet & operator= (const PointContactFacet &rhs)
- PointContactFacet (const PointContactFacet &rhs)

Friends

- class InputProcessor
- void init_attrjeod__PointContactFacet ()

8.14.1 Detailed Description

The contact facet based on the distance to a single point, specifically the vehicle point.

In effect this represents a sphere.

Definition at line 51 of file point_contact_facet.hh.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 jeod::PointContactFacet::PointContactFacet (void)

Default constructor.

Definition at line 51 of file point_contact_facet.cc.

References contact_point.

8.14.2.2 jeod::PointContactFacet::∼PointContactFacet (void) `[virtual]`

Destructor.

Definition at line 64 of file point_contact_facet.cc.

8.14.2.3 jeod::PointContactFacet::PointContactFacet (const PointContactFacet & *rhs*) `[private]`

8.14.3 Member Function Documentation

8.14.3.1 void jeod::PointContactFacet::calculate_contact_point (double *nvec[3]*) `[virtual]`

Use the relstate and radius of contact to calculate a contact point on this facet.

Parameters

`in`	*nvec*	direction vector between the two facets

Reimplemented in jeod::LineContactFacet.

Definition at line 139 of file point_contact_facet.cc.

References contact_point, and radius.

Referenced by jeod::PointContactPair::in_contact(), and jeod::LinePointContactPair::in_contact().

8.14.3.2 void jeod::PointContactFacet::calculate_torque (double ∗ *tmp_force*) `[virtual]`

Calculate the torque generated on the vehicle by the facet.

Assumes that the force is in the vehicle structural frame, but that close_point is not.

Parameters

in	*tmp_force*	force from one contact interaction. Units: N

Implements jeod::ContactFacet.

Reimplemented in jeod::LineContactFacet.

Definition at line 167 of file point_contact_facet.cc.

References contact_point, jeod::ContactFacet::vehicle_body, and jeod::ContactFacet::vehicle_point.

8.14.3.3 ContactPair ∗ jeod::PointContactFacet::create_pair (void) [virtual]

Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

This function is called to create a pair that only contains a subject.

Returns

ContactPair that was created

Implements jeod::ContactFacet.

Reimplemented in jeod::LineContactFacet.

Definition at line 80 of file point_contact_facet.cc.

References jeod::PointContactPair::initialize_pair().

8.14.3.4 ContactPair ∗ jeod::PointContactFacet::create_pair (ContactFacet ∗ *target,* Contact ∗ *contact*) [virtual]

Overloaded functions that create a ContactPair and pass the address of it to the Contact class for addition to the list of pairs.

This function is called when a subject and target are known.

Returns

ContactPair that was created

Parameters

in,out	*target*	target ContactFacet
in	*contact*	Contact object used to find the pair interaction

Implements jeod::ContactFacet.

Reimplemented in jeod::LineContactFacet.

Definition at line 101 of file point_contact_facet.cc.

References jeod::Contact::contact_limit_factor, jeod::Contact::find_interaction(), jeod::ContactMessages::initialization-_warns, jeod::PointContactPair::initialize_pair(), jeod::ContactPair::interaction, jeod::ContactPair::interaction_-distance, jeod::ContactFacet::max_dimension, and jeod::ContactFacet::surface_type.

8.14.3.5 PointContactFacet& jeod::PointContactFacet::operator= (const PointContactFacet & *rhs*) [private]

8.14.3.6 void jeod::PointContactFacet::set_max_dimension (void) [virtual]

calculate the max dimension of the facet for range limit determination.

Implements jeod::ContactFacet.

Reimplemented in jeod::LineContactFacet.

Definition at line 154 of file point_contact_facet.cc.

References jeod::ContactFacet::max_dimension, and radius.

Referenced by jeod::PointContactFacetFactory::create_facet().

8.14.4 Friends And Related Function Documentation

8.14.4.1 void init_attrjeod__PointContactFacet ( ) `[friend]`

8.14.4.2 friend class InputProcessor `[friend]`

Definition at line 53 of file point_contact_facet.hh.

8.14.5 Field Documentation

8.14.5.1 double jeod::PointContactFacet::contact_point[3]

Contact point given in facet vehicle point frame, representing point on the point on the surface of a sphere of radius "radius" where contact has occured.

trick_units(m)

Definition at line 66 of file point_contact_facet.hh.

Referenced by jeod::LineContactFacet::calculate_contact_point(), calculate_contact_point(), jeod::LineContactFacet::calculate_torque(), calculate_torque(), jeod::LineContactPair::in_contact(), jeod::PointContactPair::in_contact(), jeod::LinePointContactPair::in_contact(), jeod::LineContactFacet::LineContactFacet(), and PointContactFacet().

8.14.5.2 double jeod::PointContactFacet::radius

radius from the point at which contact takes place.

trick_units(m)

Definition at line 59 of file point_contact_facet.hh.

Referenced by jeod::LineContactFacet::calculate_contact_point(), calculate_contact_point(), jeod::PointContactFacetFactory::create_facet(), jeod::LineContactFacetFactory::create_facet(), jeod::PointContactPair::in_contact(), jeod::LineContactFacet::set_max_dimension(), and set_max_dimension().

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

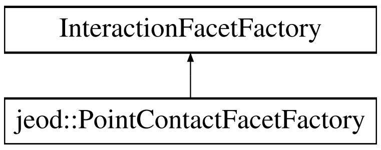
- point_contact_facet.hh
- point_contact_facet.cc

8.15 jeod::PointContactFacetFactory Class Reference

Creates a PointContactFacet from an InteractionFacet.

`#include <point_contact_facet_factory.hh>`

Inheritance diagram for jeod::PointContactFacetFactory:



Public Member Functions

- PointContactFacetFactory ()

  Default Constructor.
- ∼PointContactFacetFactory ()

  Destructor.
- virtual InteractionFacet ∗ create_facet (Facet ∗facet, FacetParams ∗params)

  Create a PointContactFacet from a CircularFlatPlate facet and a ContactParams object.
- virtual bool is_correct_factory (Facet ∗facet)

  PointContactFacetFactory specific implementation of this function.

Private Member Functions

- PointContactFacetFactory & operator= (const PointContactFacetFactory &rhs)
- PointContactFacetFactory (const PointContactFacetFactory &rhs)

Friends

- class InputProcessor
- void init_attrjeod__PointContactFacetFactory ()

8.15.1 Detailed Description

Creates a PointContactFacet from an InteractionFacet.

Definition at line 50 of file point_contact_facet_factory.hh.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 jeod::PointContactFacetFactory::PointContactFacetFactory (void)

Default Constructor.

Definition at line 54 of file point_contact_facet_factory.cc.

8.15.2.2 jeod::PointContactFacetFactory::∼PointContactFacetFactory (void)

Destructor.

Definition at line 65 of file point_contact_facet_factory.cc.

8.15.2.3 jeod::PointContactFacetFactory::PointContactFacetFactory (const **PointContactFacetFactory** & *rhs*) [private]

8.15.3 Member Function Documentation

8.15.3.1 InteractionFacet ∗ jeod::PointContactFacetFactory::create_facet (Facet ∗ *facet,* FacetParams ∗ *params*) [virtual]

Create a PointContactFacet from a CircularFlatPlate facet and a ContactParams object.

Returns

The new EllipsoidContactFacet. Note that this is allocated and YOU are responsible for destroying it at the end!

Parameters

in	*facet*	The CircularFlatPlate. This MUST be a circular flat plate or the algorithm will send a failure message
in	*params*	ContactParams

Definition at line 83 of file point_contact_facet_factory.cc.

References jeod::ContactFacet::create_vehicle_point(), jeod::ContactMessages::initialization_error, jeod::Contact-Facet::normal, jeod::ContactFacet::position, jeod::PointContactFacet::radius, jeod::PointContactFacet::set_max_-dimension(), jeod::ContactFacet::surface_type, and jeod::ContactFacet::vehicle_body.

8.15.3.2 bool jeod::PointContactFacetFactory::is_correct_factory (Facet ∗ *facet*) [virtual]

PointContactFacetFactory specific implementation of this function.

If the Facet is of type CircularFlatPlate, returns true. False otherwise

Returns

true if facet is a FlatPlateCircular, false otherwise

Parameters

in	*facet*	The facet to check

Definition at line 160 of file point_contact_facet_factory.cc.

8.15.3.3 **PointContactFacetFactory**& jeod::PointContactFacetFactory::operator= (const **PointContactFacetFactory** & *rhs*) [private]

8.15.4 Friends And Related Function Documentation

8.15.4.1 void init_attrjeod__PointContactFacetFactory ( ) [friend]

8.15.4.2 friend class InputProcessor [friend]

Definition at line 53 of file point_contact_facet_factory.hh.

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

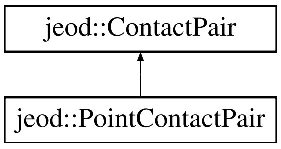
- point_contact_facet_factory.hh
- point_contact_facet_factory.cc

8.16 jeod::PointContactPair Class Reference

An point to point contact pair for use in the contact model.

`#include <point_contact_pair.hh>`

Inheritance diagram for jeod::PointContactPair:



Public Member Functions

- PointContactPair ()

  Default Constructor.
- virtual ∼PointContactPair ()

  Destructor.
- virtual void in_contact ()

  Determine if contact has occurred between the facets of the pair.
- virtual void initialize_pair (ContactFacet ∗subject_facet, ContactFacet ∗target_facet)

  Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Data Fields

- PointContactFacet ∗ point_subject

  pointer to the point contact facet that is the subject of the associated relative states.
- PointContactFacet ∗ point_target

  pointer to the point contact facet that is the target of the associated relative states.

Private Member Functions

- PointContactPair & operator= (const PointContactPair &rhs)
- PointContactPair (const PointContactPair &rhs)

Friends

- class InputProcessor
- void init_attrjeod__PointContactPair ()

Additional Inherited Members

8.16.1 Detailed Description

An point to point contact pair for use in the contact model.

Definition at line 48 of file point_contact_pair.hh.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 jeod::PointContactPair::PointContactPair (void)

Default Constructor.

Definition at line 47 of file point_contact_pair.cc.

8.16.2.2 jeod::PointContactPair::∼PointContactPair (void) [virtual]

Destructor.

Definition at line 60 of file point_contact_pair.cc.

8.16.2.3 jeod::PointContactPair::PointContactPair (const **PointContactPair** & *rhs*) [private]

8.16.3 Member Function Documentation

8.16.3.1 void jeod::PointContactPair::in_contact (void) [virtual]

Determine if contact has occurred between the facets of the pair.

Implements jeod::ContactPair.

Definition at line 70 of file point_contact_pair.cc.

References jeod::PointContactFacet::calculate_contact_point(), jeod::PairInteraction::calculate_forces(), jeod::-PointContactFacet::contact_point, jeod::ContactPair::interaction, point_subject, point_target, jeod::PointContact-Facet::radius, and jeod::ContactPair::rel_state.

8.16.3.2 void jeod::PointContactPair::initialize_pair (**ContactFacet** ∗ *subject_facet,* **ContactFacet** ∗ *target_facet*) [virtual]

Initialize the contact pair by setting the subject, target, and creating the relstate if possible.

Parameters

in,out	*subject_facet*	subject ContactFacet
in,out	*target_facet*	target ContactFacet

Implements jeod::ContactPair.

Definition at line 125 of file point_contact_pair.cc.

References jeod::ContactUtils::copy_const_char_to_char(), point_subject, point_target, jeod::ContactPair::rel_-state, jeod::ContactPair::subject, jeod::ContactPair::target, and jeod::ContactFacet::vehicle_point.

Referenced by jeod::PointContactFacet::create_pair().

8.16.3.3 **PointContactPair**& jeod::PointContactPair::operator= (const **PointContactPair** & *rhs*) [private]

8.16.4 Friends And Related Function Documentation

8.16.4.1 void init_attrjeod__PointContactPair ( ) [friend]

8.16.4.2 friend class InputProcessor [friend]

Definition at line 50 of file point_contact_pair.hh.

8.16.5 Field Documentation

8.16.5.1 PointContactFacet∗ jeod::PointContactPair::point_subject

pointer to the point contact facet that is the subject of the associated relative states.

trick_units(–)

Definition at line 56 of file point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

8.16.5.2 PointContactFacet∗ jeod::PointContactPair::point_target

pointer to the point contact facet that is the target of the associated relative states.

trick_units(–)

Definition at line 61 of file point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

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

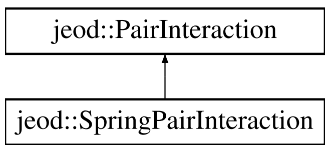
- point_contact_pair.hh
- point_contact_pair.cc

8.17 jeod::SpringPairInteraction Class Reference

Simple spring contact parameters.

```
#include <spring_pair_interaction.hh>
```

Inheritance diagram for jeod::SpringPairInteraction:



Public Member Functions

- SpringPairInteraction ()

  Default Constructor.
- virtual ∼SpringPairInteraction ()

  Destructor.
- virtual void calculate_forces (ContactFacet ∗subject, ContactFacet ∗target, RelativeDerivedState ∗rel_state, double ∗penetration_vector, double ∗rel_velocity)

  force calculation for a simple spring based contact dynamics model, takes in geometry information from the appropriate ContactFacet::calculate_forces but doesn't know about specific type of ContactFacet

Data Fields

- double spring_k

  Spring stiffness constant.
- double damping_b

Spring damping constant.

- double mu

  Coefficent of friction.

Private Member Functions

- SpringPairInteraction & operator= (const SpringPairInteraction &rhs)
- SpringPairInteraction (const SpringPairInteraction &rhs)

Friends

- class InputProcessor
- void init_attrjeod__SpringPairInteraction ()

8.17.1 Detailed Description

Simple spring contact parameters.

Definition at line 47 of file spring_pair_interaction.hh.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 jeod::SpringPairInteraction::SpringPairInteraction (void)

Default Constructor.

Definition at line 51 of file spring_pair_interaction.cc.

8.17.2.2 jeod::SpringPairInteraction::∼SpringPairInteraction (void) `[virtual]`

Destructor.

Definition at line 66 of file spring_pair_interaction.cc.

8.17.2.3 jeod::SpringPairInteraction::SpringPairInteraction (const **SpringPairInteraction** & *rhs*) `[private]`

8.17.3 Member Function Documentation

8.17.3.1 void jeod::SpringPairInteraction::calculate_forces (**ContactFacet** ∗ *subject,* **ContactFacet** ∗ *target,* RelativeDerivedState ∗ *rel_state,* double ∗ *penetration_vector,* double ∗ *rel_velocity*) `[virtual]`

force calculation for a simple spring based contact dynamics model, takes in geometry information from the appropriate ContactFacet::calculate_forces but doesn't know about specific type of ContactFacet

Parameters

`in,out`	*subject*	subject frame of the relative state
`in,out`	*target*	target frame of the relative state
`in`	*rel_state*	relative state between subject and target in subject frame
`in`	*penetration_-vector*	vector that characterises the interpenetration of the subject and the target

in	*rel_velocity*	relative velocity of the subject and the target in the subject frame

Implements jeod::PairInteraction.

Definition at line 87 of file spring_pair_interaction.cc.

References jeod::ContactFacet::calculate_torque(), damping_b, jeod::PairInteraction::friction_mag, mu, spring_k, jeod::ContactFacet::vehicle_body, and jeod::ContactFacet::vehicle_point.

8.17.3.2 SpringPairInteraction& jeod::SpringPairInteraction::operator= (const SpringPairInteraction & *rhs*) [private]

8.17.4 Friends And Related Function Documentation

8.17.4.1 void init_attrjeod__SpringPairInteraction ( ) [friend]

8.17.4.2 friend class InputProcessor [friend]

Definition at line 50 of file spring_pair_interaction.hh.

8.17.5 Field Documentation

8.17.5.1 double jeod::SpringPairInteraction::damping_b

Spring damping constant.

trick_units(N∗s/m)

Definition at line 61 of file spring_pair_interaction.hh.

Referenced by calculate_forces().

8.17.5.2 double jeod::SpringPairInteraction::mu

Coefficent of friction.

trick_units(–)

Definition at line 66 of file spring_pair_interaction.hh.

Referenced by calculate_forces().

8.17.5.3 double jeod::SpringPairInteraction::spring_k

Spring stiffness constant.

trick_units(N/m)

Definition at line 56 of file spring_pair_interaction.hh.

Referenced by calculate_forces().

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

- spring_pair_interaction.hh
- spring_pair_interaction.cc

Chapter 9

File Documentation

9.1 class_declarations.hh File Reference

Forward declaration of classes defined in the contact model.

Namespaces

- jeod

  Namespace jeod.

9.1.1 Detailed Description

Forward declaration of classes defined in the contact model.

Definition in file class_declarations.hh.

9.2 contact.cc File Reference

Base Contact for use with contact interaction model.

```
#include "dynamics/mass/include/mass.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "../include/contact.hh"
#include "../include/contact_facet.hh"
#include "../include/contact_pair.hh"
#include "../include/pair_interaction.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.2.1 Detailed Description

Base Contact for use with contact interaction model.

Definition in file contact.cc.

9.3 contact.hh File Reference

(Base class to for the contact manager for use with contact interaction model)

```
#include <list>
#include "dynamics/dyn_manager/include/class_declarations.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/container/include/pointer_list.hh"
#include "class_declarations.hh"
#include "contact_facet.hh"
#include "contact_pair.hh"
#include "pair_interaction.hh"
```

Data Structures

- class jeod::Contact

    An base contact class for use in the surface model.

Namespaces

- jeod

    Namespace jeod.

9.3.1 Detailed Description

(Base class to for the contact manager for use with contact interaction model)

Definition in file contact.hh.

9.4 contact_facet.cc File Reference

Define ContactFacet::create_vehicle_point.

```
#include <cstddef>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/mass/include/mass_point_init.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "../include/contact_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_utils.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.4.1 Detailed Description

Define ContactFacet::create_vehicle_point.

Definition in file contact_facet.cc.

9.5 contact_facet.hh File Reference

Individual facets for use with contact interaction models.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "class_declarations.hh"
```

Data Structures

- class jeod::ContactFacet

  An contact interaction specific facet for use in the surface model.

Namespaces

- jeod

  Namespace jeod.

9.5.1 Detailed Description

Individual facets for use with contact interaction models.

Definition in file contact_facet.hh.

9.6 contact_messages.cc File Reference

Implement contact_messages.

```
#include "../include/contact_messages.hh"
```

Namespaces

- jeod

  Namespace jeod.

Macros

- #define PATH "interactions/contact"

9.6.1 Detailed Description

Implement contact_messages.

Definition in file contact_messages.cc.

9.7 contact_messages.hh File Reference

Contact message for message handling.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/message/include/message_handler.hh"
```

Data Structures

- class jeod::ContactMessages

  Messages associated with use of the contact model.

Namespaces

- jeod

  Namespace jeod.

9.7.1 Detailed Description

Contact message for message handling.

Definition in file contact_messages.hh.

9.8 contact_pair.cc File Reference

ContactPair class for use with contact interaction model.

```
#include "utils/math/include/vector3.hh"
#include "dynamics/mass/include/mass.hh"
#include "../include/contact_pair.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.8.1 Detailed Description

ContactPair class for use with contact interaction model.

Definition in file contact_pair.cc.

9.9 contact_pair.hh File Reference

Base class for pair of contact facets for use with contact interaction model.

```
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "class_declarations.hh"
#include "contact_facet.hh"
```

Data Structures

- class jeod::ContactPair

  An base contact pair class for use in the contact model.

Namespaces

- jeod

  Namespace jeod.

9.9.1 Detailed Description

Base class for pair of contact facets for use with contact interaction model.

Definition in file contact_pair.hh.

9.10 contact_params.cc File Reference

contact parameters for use in the surface model

```
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/contact_params.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.10.1 Detailed Description

contact parameters for use in the surface model

Definition in file contact_params.cc.

9.11 contact_params.hh File Reference

A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/facet_params.hh"
```

Data Structures

- class jeod::ContactParams

  A base class for all contact parameters used in the surface model.

Namespaces

- jeod

  Namespace jeod.

9.11.1 Detailed Description

A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys.

Definition in file contact_params.hh.

9.12 contact_surface.cc File Reference

Vehicle surface model for the contact interaction models.

```
#include <typeinfo>
#include <cstddef>
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "../include/contact_surface.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_facet.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.12.1 Detailed Description

Vehicle surface model for the contact interaction models.

Definition in file contact_surface.cc.

9.13 contact_surface.hh File Reference

Vehicle surface model for contact.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_surface.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "contact_facet.hh"
```

Data Structures

- class jeod::ContactSurface

  The contact specific interaction surface, for use with the surface model.

Namespaces

- jeod

  Namespace jeod.

9.13.1 Detailed Description

Vehicle surface model for contact.

Definition in file contact_surface.hh.

9.14 contact_surface_factory.cc File Reference

Factory that creates an contact surface, from a surface model.

```
#include <cstring>
#include <cstddef>
#include "dynamics/mass/include/mass.hh"
#include "utils/surface_model/include/facet.hh"
#include "utils/surface_model/include/facet_params.hh"
#include "utils/surface_model/include/surface_model.hh"
#include "utils/surface_model/include/interaction_surface.hh"
#include "../include/contact_surface_factory.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.14.1 Detailed Description

Factory that creates an contact surface, from a surface model.

Definition in file contact_surface_factory.cc.

9.15 contact_surface_factory.hh File Reference

Factory that creates an contact interaction surface from a surface model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/interaction_surface_factory.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "point_contact_facet_factory.hh"
#include "line_contact_facet_factory.hh"
```

Data Structures

- class jeod::ContactSurfaceFactory

  The surface factory that creates an contact specific surface from a general surface.

Namespaces

- jeod

  Namespace jeod.

9.15.1 Detailed Description

Factory that creates an contact interaction surface from a surface model.

Definition in file contact_surface_factory.hh.

9.16 contact_utils.hh File Reference

This Model is used for utility rotines.

```
#include "contact_utils_inline.hh"
```

Data Structures

- class jeod::ContactUtils

  Utility string and math functions for the contact model.

Namespaces

- jeod

  Namespace jeod.

9.16.1 Detailed Description

This Model is used for utility rotines.

Definition in file contact_utils.hh.

9.17 contact_utils_inline.hh File Reference

Define ContactUtils::create_relstate_name, ContactUtils::copy_const_char_to_char.

```
#include <cstring>
#include "utils/math/include/vector3.hh"
#include "contact_utils.hh"
#include "contact_messages.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.17.1 Detailed Description

Define ContactUtils::create_relstate_name, ContactUtils::copy_const_char_to_char.

Definition in file contact_utils_inline.hh.

9.18 line_contact_facet.cc File Reference

Define LineContactFacet functions.

```
#include <cstring>
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/numerical.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
#include "../include/line_contact_facet.hh"
#include "../include/line_contact_pair.hh"
#include "../include/line_point_contact_pair.hh"
#include "../include/contact_utils.hh"
#include "../include/contact.hh"
```

Namespaces

- jeod

    Namespace jeod.

9.18.1 Detailed Description

Define LineContactFacet functions.

Definition in file line_contact_facet.cc.

9.19 line_contact_facet.hh File Reference

The contact facet based on the distance to a line segment centered on the vehicle point.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "class_declarations.hh"
#include "point_contact_facet.hh"
#include "line_contact_pair.hh"
#include "line_point_contact_pair.hh"
```

Data Structures

- class jeod::LineContactFacet

  The contact facet based on the distance to a single point, specifically the vehicle point.

Namespaces

- jeod

  Namespace jeod.

9.19.1 Detailed Description

The contact facet based on the distance to a line segment centered on the vehicle point. In effect this represents a cylinder with spherical ends.

Definition in file line_contact_facet.hh.

9.20 line_contact_facet_factory.cc File Reference

Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object.

```
#include <typeinfo>
#include <cstddef>
#include "utils/surface_model/include/cylinder.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/line_contact_facet_factory.hh"
#include "../include/line_contact_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.20.1 Detailed Description

Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object.

Definition in file line_contact_facet_factory.cc.

9.21 line_contact_facet_factory.hh File Reference

Creates a line contact facet from an cylinder facet.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "class_declarations.hh"
#include "line_contact_facet.hh"
```

Data Structures

- class jeod::LineContactFacetFactory

  Creates a PointContactFacet from an InteractionFacet.

Namespaces

- jeod

  Namespace jeod.

9.21.1 Detailed Description

Creates a line contact facet from an cylinder facet.

Definition in file line_contact_facet_factory.hh.

9.22 line_contact_pair.cc File Reference

LineContactPair class for use with contact interaction model.

```
#include "dynamics/mass/include/mass.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/line_contact_pair.hh"
#include "../include/line_contact_facet.hh"
#include "../include/pair_interaction.hh"
#include "../include/contact_utils.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.22.1 Detailed Description

LineContactPair class for use with contact interaction model.

Definition in file line_contact_pair.cc.

9.23 line_contact_pair.hh File Reference

Class for a pair of line contact facets for use with contact interaction model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "contact_pair.hh"
#include "class_declarations.hh"
```

Data Structures

- class jeod::LineContactPair

  An point to point contact pair for use in the contact model.

Namespaces

- jeod

  Namespace jeod.

9.23.1 Detailed Description

Class for a pair of line contact facets for use with contact interaction model.

Definition in file line_contact_pair.hh.

9.24 line_point_contact_pair.cc File Reference

LinePointContactPair class for use with contact interaction model.

```
#include "dynamics/mass/include/mass.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/line_point_contact_pair.hh"
#include "../include/line_contact_facet.hh"
#include "../include/point_contact_facet.hh"
#include "../include/pair_interaction.hh"
#include "../include/contact_utils.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.24.1 Detailed Description

LinePointContactPair class for use with contact interaction model.

Definition in file line_point_contact_pair.cc.

9.25 line_point_contact_pair.hh File Reference

Class for a pair of a line contact facet and a point contact facet for use with contact interaction model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "contact_pair.hh"
#include "class_declarations.hh"
```

Data Structures

- class jeod::LinePointContactPair

  An point to point contact pair for use in the contact model.

Namespaces

- jeod

  Namespace jeod.

9.25.1 Detailed Description

Class for a pair of a line contact facet and a point contact facet for use with contact interaction model.

Definition in file line_point_contact_pair.hh.

9.26 pair_interaction.cc File Reference

A class to define the interaction type for a pair of contact facets.

```
#include <cstring>
#include "../include/pair_interaction.hh"
#include "../include/contact_facet.hh"
#include "../include/contact_params.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.26.1 Detailed Description

A class to define the interaction type for a pair of contact facets. This is a base class and derived classes define the force generation function when contact between facets occurs.

Definition in file pair_interaction.cc.

9.27 pair_interaction.hh File Reference

A class to define the interaction type for a pair of contact facets.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "dynamics/derived_state/include/class_declarations.hh"
#include "../include/class_declarations.hh"
```

Data Structures

- class jeod::PairInteraction

  Simple spring contact parameters.

Namespaces

- jeod

  Namespace jeod.

9.27.1 Detailed Description

A class to define the interaction type for a pair of contact facets. This is a base class and derived classes define the force generation function when contact between facets occurs.

Definition in file pair_interaction.hh.

9.28 point_contact_facet.cc File Reference

Define PointContactFacet functions.

```
#include <cstring>
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/numerical.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
#include "../include/point_contact_facet.hh"
#include "../include/point_contact_pair.hh"
#include "../include/contact_utils.hh"
#include "../include/contact.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.28.1 Detailed Description

Define PointContactFacet functions.

Definition in file point_contact_facet.cc.

9.29 point_contact_facet.hh File Reference

The contact facet based on the distance to a single point, specifically the vehicle point.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "class_declarations.hh"
#include "contact_facet.hh"
#include "point_contact_pair.hh"
```

Data Structures

- class jeod::PointContactFacet

  The contact facet based on the distance to a single point, specifically the vehicle point.

Namespaces

- jeod

  Namespace jeod.

9.29.1 Detailed Description

The contact facet based on the distance to a single point, specifically the vehicle point. In effect this represents a sphere.

Definition in file point_contact_facet.hh.

9.30 point_contact_facet_factory.cc File Reference

Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object.

```
#include <typeinfo>
#include <cstddef>
#include "utils/surface_model/include/flat_plate_circular.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/point_contact_facet_factory.hh"
#include "../include/point_contact_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_params.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.30.1 Detailed Description

Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object.

Definition in file point_contact_facet_factory.cc.

9.31 point_contact_facet_factory.hh File Reference

Creates a point contact facet from an circular flat plate facet.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "utils/surface_model/include/class_declarations.hh"
#include "utils/surface_model/include/interaction_facet.hh"
#include "utils/surface_model/include/interaction_facet_factory.hh"
#include "class_declarations.hh"
#include "point_contact_facet.hh"
```

Data Structures

- class jeod::PointContactFacetFactory

  Creates a PointContactFacet from an InteractionFacet.

Namespaces

- jeod

Namespace jeod.

9.31.1 Detailed Description

Creates a point contact facet from an circular flat plate facet.

Definition in file point_contact_facet_factory.hh.

9.32 point_contact_pair.cc File Reference

ContactPair class for use with contact interaction model.

```
#include "utils/named_item/include/named_item.hh"
#include "../include/point_contact_pair.hh"
#include "../include/point_contact_facet.hh"
#include "../include/pair_interaction.hh"
#include "../include/contact_utils.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.32.1 Detailed Description

ContactPair class for use with contact interaction model.

Definition in file point_contact_pair.cc.

9.33 point_contact_pair.hh File Reference

Class for a pair of point contact facets for use with contact interaction model.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "contact_pair.hh"
#include "class_declarations.hh"
```

Data Structures

- class jeod::PointContactPair

  An point to point contact pair for use in the contact model.

Namespaces

- jeod

  Namespace jeod.

9.33.1 Detailed Description

Class for a pair of point contact facets for use with contact interaction model.

Definition in file point_contact_pair.hh.

9.34 spring_pair_interaction.cc File Reference

spring pair interaction for use in the contact model

```
#include <cmath>
#include "utils/math/include/vector3.hh"
#include "dynamics/dyn_body/include/body_ref_frame.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "dynamics/derived_state/include/relative_derived_state.hh"
#include "../include/spring_pair_interaction.hh"
#include "../include/contact_facet.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.34.1 Detailed Description

spring pair interaction for use in the contact model

Definition in file spring_pair_interaction.cc.

9.35 spring_pair_interaction.hh File Reference

A class for pair interactions based on a simple spring.

```
#include "utils/sim_interface/include/jeod_class.hh"
#include "pair_interaction.hh"
```

Data Structures

- class jeod::SpringPairInteraction

  Simple spring contact parameters.

Namespaces

- jeod

  Namespace jeod.

9.35.1 Detailed Description

A class for pair interactions based on a simple spring.

Definition in file spring_pair_interaction.hh.

Index