ContactModel

5.3

Generated by Doxygen 1.8.14

Contents

1 Module Index **1**

1.1 Modules 1

2 Namespace Index **3**

2.1 Namespace List 3

3 Hierarchical Index **5**

3.1 Class Hierarchy 5

4 Data Structure Index **7**

4.1 Data Structures 7

5 File Index **9**

5.1 File List 9

6 Module Documentation **11**

6.1 Models 11

6.1.1 Detailed Description 11

6.2 Interactions 12

6.2.1 Detailed Description 12

6.3 Contact 13

6.3.1 Detailed Description 14

7 Namespace Documentation **15**

7.1 jeod Namespace Reference 15

7.1.1 Detailed Description 16

8 Data Structure Documentation **17**

8.1 jeod::Contact Class Reference 17

8.1.1 Detailed Description 18

8.1.2 Constructor & Destructor Documentation 18

8.1.2.1 Contact() [1/2] 18

8.1.2.2 ∼Contact() 19

8.1.2.3 Contact() [2/2] 19

8.1.3 Member Function Documentation 19

8.1.3.1 check_contact() 19

8.1.3.2 find_interaction() 19

8.1.3.3 initialize_contact() 20

8.1.3.4 operator=() 20

8.1.3.5 register_contact() [1/4] 20

8.1.3.6 register_contact() [2/4] 21

8.1.3.7 register_contact() [3/4] 21

8.1.3.8 register_contact() [4/4] 21

8.1.3.9 register_interaction() 22

8.1.3.10 unique_pair() 22

8.1.4 Friends And Related Function Documentation 22

8.1.4.1 init_attrjeod__Contact 23

8.1.4.2 InputProcessor 23

8.1.5 Field Documentation 23

8.1.5.1 active 23

8.1.5.2 contact_limit_factor 23

8.1.5.3 contact_pairs 24

8.1.5.4 dyn_manager 24

8.1.5.5 pair_interactions 24

8.2 jeod::ContactFacet Class Reference 25

8.2.1 Detailed Description 26

8.2.2 Constructor & Destructor Documentation 26

8.2.2.1 ContactFacet() [1/2] . . . 26
8.2.2.2 ∼ContactFacet() . . . 26
8.2.2.3 ContactFacet() [2/2] . . . 26
8.2.3 Member Function Documentation . . . 26
8.2.3.1 calculate_torque() . . . 27
8.2.3.2 create_pair() [1/2] . . . 27
8.2.3.3 create_pair() [2/2] . . . 27
8.2.3.4 create_vehicle_point() . . . 28
8.2.3.5 get_name() . . . 28
8.2.3.6 operator=() . . . 28
8.2.3.7 set_max_dimension() . . . 28
8.2.4 Friends And Related Function Documentation . . . 29
8.2.4.1 init_attrjeod__ContactFacet . . . 29
8.2.4.2 InputProcessor . . . 29
8.2.5 Field Documentation . . . 29
8.2.5.1 active . . . 29
8.2.5.2 max_dimension . . . 29
8.2.5.3 normal . . . 30
8.2.5.4 position . . . 30
8.2.5.5 surface_type . . . 30
8.2.5.6 vehicle_body . . . 30
8.2.5.7 vehicle_point . . . 31
8.3 jeod::ContactMessages Class Reference . . . 31
8.3.1 Detailed Description . . . 32
8.3.2 Constructor & Destructor Documentation . . . 32
8.3.2.1 ContactMessages() [1/2] . . . 32
8.3.2.2 ContactMessages() [2/2] . . . 32
8.3.3 Member Function Documentation . . . 32
8.3.3.1 operator=() . . . 32
8.3.4 Friends And Related Function Documentation . . . 32

8.3.4.1 init_attrjeod__ContactMessages 32
8.3.4.2 InputProcessor 32
8.3.5 Field Documentation 33
8.3.5.1 initialization_error 33
8.3.5.2 initialization_warns 33
8.3.5.3 pre_initialization_error 33
8.3.5.4 runtime_error 34
8.3.5.5 runtime_inform 34
8.3.5.6 runtime_warns 34
8.4 jeod::ContactPair Class Reference 34
8.4.1 Detailed Description 36
8.4.2 Constructor & Destructor Documentation 36
8.4.2.1 ContactPair() [1/2] 36
8.4.2.2 ∼ContactPair() 36
8.4.2.3 ContactPair() [2/2] 36
8.4.3 Member Function Documentation 36
8.4.3.1 check_tree() 36
8.4.3.2 get_subject() 37
8.4.3.3 get_target() 37
8.4.3.4 in_contact() 37
8.4.3.5 in_range() 37
8.4.3.6 initialize_pair() 37
8.4.3.7 initialize_relstate() 38
8.4.3.8 is_active() 38
8.4.3.9 is_complete() 39
8.4.3.10 operator=() 39
8.4.4 Friends And Related Function Documentation 39
8.4.4.1 init_attrjeod__ContactPair 39
8.4.4.2 InputProcessor 39
8.4.5 Field Documentation 39

8.4.5.1 interaction 40
8.4.5.2 interaction_distance 40
8.4.5.3 rel_state 40
8.4.5.4 subject 40
8.4.5.5 target 41
8.5 jeod::ContactParams Class Reference 41
8.5.1 Detailed Description 41
8.5.2 Constructor & Destructor Documentation 42
8.5.2.1 ContactParams() [1/2] 42
8.5.2.2 ∼ContactParams() 42
8.5.2.3 ContactParams() [2/2] 42
8.5.3 Member Function Documentation 42
8.5.3.1 operator=() 42
8.5.4 Friends And Related Function Documentation 42
8.5.4.1 init_attrjeod__ContactParams 42
8.5.4.2 InputProcessor 43
8.6 jeod::ContactSurface Class Reference 43
8.6.1 Detailed Description 44
8.6.2 Constructor & Destructor Documentation 44
8.6.2.1 ContactSurface() [1/2] 44
8.6.2.2 ∼ContactSurface() 44
8.6.2.3 ContactSurface() [2/2] 44
8.6.3 Member Function Documentation 44
8.6.3.1 allocate_array() 44
8.6.3.2 allocate_interaction_facet() 45
8.6.3.3 collect_forces_torques() 45
8.6.3.4 operator=() 45
8.6.4 Friends And Related Function Documentation 46
8.6.4.1 init_attrjeod__ContactSurface 46
8.6.4.2 InputProcessor 46

8.6.5 Field Documentation 46
8.6.5.1 contact_facets 46
8.6.5.2 contact_force 46
8.6.5.3 contact_torque 47
8.6.5.4 facets_size 47
8.7 jeod::ContactSurfaceFactory Class Reference 47
8.7.1 Detailed Description 48
8.7.2 Constructor & Destructor Documentation 48
8.7.2.1 ContactSurfaceFactory() [1/2] 48
8.7.2.2 ∼ContactSurfaceFactory() 49
8.7.2.3 ContactSurfaceFactory() [2/2] 49
8.7.3 Member Function Documentation 49
8.7.3.1 add_facet_params() 49
8.7.3.2 create_surface() 49
8.7.3.3 operator=() 50
8.7.4 Friends And Related Function Documentation 50
8.7.4.1 init_attrjeod__ContactSurfaceFactory 50
8.7.4.2 InputProcessor 50
8.7.5 Field Documentation 50
8.7.5.1 line_contact_facet_factory 50
8.7.5.2 point_contact_facet_factory 51
8.8 jeod::ContactUtils Class Reference 51
8.8.1 Detailed Description 51
8.8.2 Member Function Documentation 51
8.8.2.1 create_relstate_name() 51
8.8.2.2 dist_line_segments() 52
8.9 jeod::LineContactFacet Class Reference 53
8.9.1 Detailed Description 54
8.9.2 Constructor & Destructor Documentation 54
8.9.2.1 LineContactFacet() [1/2] 54

8.9.2.2 ∼LineContactFacet() 54
8.9.2.3 LineContactFacet() [2/2] 54
8.9.3 Member Function Documentation 54
8.9.3.1 calculate_contact_point() 54
8.9.3.2 calculate_torque() 55
8.9.3.3 create_pair() [1/2] 55
8.9.3.4 create_pair() [2/2] 56
8.9.3.5 operator=() 56
8.9.3.6 set_max_dimension() 56
8.9.4 Friends And Related Function Documentation 57
8.9.4.1 init_attrjeod__LineContactFacet 57
8.9.4.2 InputProcessor 57
8.9.5 Field Documentation 57
8.9.5.1 length 57
8.10 jeod::LineContactFacetFactory Class Reference 57
8.10.1 Detailed Description 58
8.10.2 Constructor & Destructor Documentation 58
8.10.2.1 LineContactFacetFactory() [1/2] 58
8.10.2.2 ∼LineContactFacetFactory() 58
8.10.2.3 LineContactFacetFactory() [2/2] 58
8.10.3 Member Function Documentation 59
8.10.3.1 create_facet() 59
8.10.3.2 is_correct_factory() 59
8.10.3.3 operator=() 60
8.10.4 Friends And Related Function Documentation 60
8.10.4.1 init_attrjeod__LineContactFacetFactory 60
8.10.4.2 InputProcessor 60
8.11 jeod::LineContactPair Class Reference 60
8.11.1 Detailed Description 61
8.11.2 Constructor & Destructor Documentation 61

8.11.2.1 LineContactPair() [1/2] 61
8.11.2.2 ∼LineContactPair() 61
8.11.2.3 LineContactPair() [2/2] 62
8.11.3 Member Function Documentation 62
8.11.3.1 in_contact() 62
8.11.3.2 initialize_pair() 62
8.11.3.3 operator=() 63
8.11.4 Friends And Related Function Documentation 63
8.11.4.1 init_attrjeod__LineContactPair 63
8.11.4.2 InputProcessor 63
8.11.5 Field Documentation 63
8.11.5.1 line_subject 63
8.11.5.2 line_target 63
8.12 jeod::LinePointContactPair Class Reference 64
8.12.1 Detailed Description 64
8.12.2 Constructor & Destructor Documentation 64
8.12.2.1 LinePointContactPair() [1/2] 65
8.12.2.2 ∼LinePointContactPair() 65
8.12.2.3 LinePointContactPair() [2/2] 65
8.12.3 Member Function Documentation 65
8.12.3.1 in_contact() 65
8.12.3.2 initialize_pair() 65
8.12.3.3 operator=() 66
8.12.4 Friends And Related Function Documentation 66
8.12.4.1 init_attrjeod__LinePointContactPair 66
8.12.4.2 InputProcessor 66
8.12.5 Field Documentation 66
8.12.5.1 line_subject 67
8.12.5.2 point_target 67
8.13 jeod::PairInteraction Class Reference 67

8.13.1 Detailed Description 68
8.13.2 Constructor & Destructor Documentation 68
8.13.2.1 PairInteraction() [1/2] 68
8.13.2.2 ∼PairInteraction() 68
8.13.2.3 PairInteraction() [2/2] 69
8.13.3 Member Function Documentation 69
8.13.3.1 calculate_forces() 69
8.13.3.2 is_correct_interaction() 69
8.13.3.3 operator=() 70
8.13.4 Friends And Related Function Documentation 70
8.13.4.1 init_attrjeod__PairInteraction 70
8.13.4.2 InputProcessor 70
8.13.5 Field Documentation 70
8.13.5.1 friction_mag 70
8.13.5.2 params_1 71
8.13.5.3 params_2 71
8.14 jeod::PointContactFacet Class Reference 71
8.14.1 Detailed Description 72
8.14.2 Constructor & Destructor Documentation 72
8.14.2.1 PointContactFacet() [1/2] 73
8.14.2.2 ∼PointContactFacet() 73
8.14.2.3 PointContactFacet() [2/2] 73
8.14.3 Member Function Documentation 73
8.14.3.1 calculate_contact_point() 73
8.14.3.2 calculate_torque() 74
8.14.3.3 create_pair() [1/2] 74
8.14.3.4 create_pair() [2/2] 74
8.14.3.5 operator=() 75
8.14.3.6 set_max_dimension() 75
8.14.4 Friends And Related Function Documentation 75

8.14.4.1 init_attrjeod__PointContactFacet 75
8.14.4.2 InputProcessor 76
8.14.5 Field Documentation 76
8.14.5.1 contact_point 76
8.14.5.2 radius 76
8.15 jeod::PointContactFacetFactory Class Reference 77
8.15.1 Detailed Description 77
8.15.2 Constructor & Destructor Documentation 77
8.15.2.1 PointContactFacetFactory() [1/2] 77
8.15.2.2 ∼PointContactFacetFactory() 78
8.15.2.3 PointContactFacetFactory() [2/2] 78
8.15.3 Member Function Documentation 78
8.15.3.1 create_facet() 78
8.15.3.2 is_correct_factory() 78
8.15.3.3 operator=() 79
8.15.4 Friends And Related Function Documentation 79
8.15.4.1 init_attrjeod__PointContactFacetFactory 79
8.15.4.2 InputProcessor 79
8.16 jeod::PointContactPair Class Reference 79
8.16.1 Detailed Description 80
8.16.2 Constructor & Destructor Documentation 80
8.16.2.1 PointContactPair() [1/2] 80
8.16.2.2 ∼PointContactPair() 80
8.16.2.3 PointContactPair() [2/2] 81
8.16.3 Member Function Documentation 81
8.16.3.1 in_contact() 81
8.16.3.2 initialize_pair() 81
8.16.3.3 operator=() 82
8.16.4 Friends And Related Function Documentation 82
8.16.4.1 init_attrjeod__PointContactPair 82

8.16.4.2 InputProcessor 82

8.16.5 Field Documentation 82

8.16.5.1 point_subject 82

8.16.5.2 point_target 82

8.17 jeod::SpringPairInteraction Class Reference 83

8.17.1 Detailed Description 83

8.17.2 Constructor & Destructor Documentation 83

8.17.2.1 SpringPairInteraction() [1/2] 84

8.17.2.2 ∼SpringPairInteraction() 84

8.17.2.3 SpringPairInteraction() [2/2] 84

8.17.3 Member Function Documentation 84

8.17.3.1 calculate_forces() 84

8.17.3.2 operator=() 85

8.17.4 Friends And Related Function Documentation 85

8.17.4.1 init_attrjeod__SpringPairInteraction 85

8.17.4.2 InputProcessor 85

8.17.5 Field Documentation 85

8.17.5.1 damping_b 85

8.17.5.2 mu 85

8.17.5.3 spring_k 86

9 File Documentation 87

9.1 class_declarations.hh File Reference . . . 87

9.1.1 Detailed Description . . . 87

9.2 contact.cc File Reference . . . 87

9.2.1 Detailed Description . . . 88

9.3 contact.hh File Reference . . . 88

9.3.1 Detailed Description . . . 88

9.4 contact_facet.cc File Reference . . . 88

9.4.1 Detailed Description . . . 89

9.5 contact_facet.hh File Reference . . . 89

9.5.1 Detailed Description . . . 89

9.6 contact_messages.cc File Reference . . . 89

9.6.1 Detailed Description . . . 90

9.6.2 Macro Definition Documentation . . . 90

9.6.2.1 MAKE_CONTACT_MESSAGE_CODE . . . 90

9.7 contact_messages.hh File Reference . . . 90

9.7.1 Detailed Description . . . 91

9.8 contact_pair.cc File Reference . . . 91

9.8.1 Detailed Description . . . 91

9.9 contact_pair.hh File Reference . . . 91

9.9.1 Detailed Description . . . 92

9.10 contact_params.cc File Reference . . . 92

9.10.1 Detailed Description . . . 92

9.11 contact_params.hh File Reference . . . 92

9.11.1 Detailed Description . . . 92

9.12 contact_surface.cc File Reference . . . 93

9.12.1 Detailed Description . . . 93

9.13 contact_surface.hh File Reference . . . 93

9.13.1 Detailed Description . . . 94

9.14 contact_surface_factory.cc File Reference . . . 94

9.14.1 Detailed Description 94
9.15 contact_surface_factory.hh File Reference 94
9.15.1 Detailed Description 95
9.16 contact_utils.hh File Reference 95
9.16.1 Detailed Description 95
9.17 contact_utils_inline.hh File Reference 95
9.17.1 Detailed Description 96
9.18 line_contact_facet.cc File Reference 96
9.18.1 Detailed Description 96
9.19 line_contact_facet.hh File Reference 96
9.19.1 Detailed Description 97
9.20 line_contact_facet_factory.cc File Reference 97
9.20.1 Detailed Description 97
9.21 line_contact_facet_factory.hh File Reference 97
9.21.1 Detailed Description 98
9.22 line_contact_pair.cc File Reference 98
9.22.1 Detailed Description 98
9.23 line_contact_pair.hh File Reference 98
9.23.1 Detailed Description 99
9.24 line_point_contact_pair.cc File Reference 99
9.24.1 Detailed Description 99
9.25 line_point_contact_pair.hh File Reference 99
9.25.1 Detailed Description 100
9.26 pair_interaction.cc File Reference 100
9.26.1 Detailed Description 100
9.27 pair_interaction.hh File Reference 100
9.27.1 Detailed Description 101
9.28 point_contact_facet.cc File Reference 101
9.28.1 Detailed Description 101
9.29 point_contact_facet.hh File Reference 102
9.29.1 Detailed Description 102
9.30 point_contact_facet_factory.cc File Reference 102
9.30.1 Detailed Description 103
9.31 point_contact_facet_factory.hh File Reference 103
9.31.1 Detailed Description 103
9.32 point_contact_pair.cc File Reference 103
9.32.1 Detailed Description 104
9.33 point_contact_pair.hh File Reference 104
9.33.1 Detailed Description 104
9.34 spring_pair_interaction.cc File Reference 104
9.34.1 Detailed Description 105
9.35 spring_pair_interaction.hh File Reference 105
9.35.1 Detailed Description 105

Index **107**

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Models 11
Interactions 12
Contact 13

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

jeod
Namespace jeod 15

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

- jeod::Contact . . . 17
- jeod::ContactMessages . . . 31
- jeod::ContactPair . . . 34
    - jeod::LineContactPair . . . 60
    - jeod::LinePointContactPair . . . 64
    - jeod::PointContactPair . . . 79
- jeod::ContactUtils . . . 51
- FacetParams
    - jeod::ContactParams . . . 41
- InteractionFacet
    - jeod::ContactFacet . . . 25
        - jeod::PointContactFacet . . . 71
            - jeod::LineContactFacet . . . 53
- InteractionFacetFactory
    - jeod::LineContactFacetFactory . . . 57
    - jeod::PointContactFacetFactory . . . 77
- InteractionSurface
    - jeod::ContactSurface . . . 43
- InteractionSurfaceFactory
    - jeod::ContactSurfaceFactory . . . 47
- jeod::PairInteraction . . . 67
    - jeod::SpringPairInteraction . . . 83

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

jeod::Contact
An base contact class for use in the surface model 17
jeod::ContactFacet
An contact interaction specific facet for use in the surface model 25
jeod::ContactMessages
Messages associated with use of the contact model 31
jeod::ContactPair
An base contact pair class for use in the contact model 34
jeod::ContactParams
A base class for all contact parameters used in the surface model 41
jeod::ContactSurface
The contact specific interaction surface, for use with the surface model 43
jeod::ContactSurfaceFactory
The surface factory that creates an contact specific surface from a general surface 47
jeod::ContactUtils
Utility string and math functions for the contact model 51
jeod::LineContactFacet
The contact facet based on the distance to a single point, specifically the vehicle point 53
jeod::LineContactFacetFactory
Creates a PointContactFacet from an InteractionFacet 57
jeod::LineContactPair
An point to point contact pair for use in the contact model 60
jeod::LinePointContactPair
An point to point contact pair for use in the contact model 64
jeod::PairInteraction
Simple spring contact parameters 67
jeod::PointContactFacet
The contact facet based on the distance to a single point, specifically the vehicle point 71
jeod::PointContactFacetFactory
Creates a PointContactFacet from an InteractionFacet 77
jeod::PointContactPair
An point to point contact pair for use in the contact model 79
jeod::SpringPairInteraction
Simple spring contact parameters 83

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

class_declarations.hh
Forward declaration of classes defined in the contact model 87
contact.cc
Base Contact for use with contact interaction model 87
contact.hh
(Base class to for the contact manager for use with contact interaction model) 88
contact_facet.cc
Define ContactFacet::create_vehicle_point 88
contact_facet.hh
Individual facets for use with contact interaction models 89
contact_messages.cc
Implement contact_messages 89
contact_messages.hh
Contact message for message handling 90
contact_pair.cc
ContactPair class for use with contact interaction model 91
contact_pair.hh
Base class for pair of contact facets for use with contact interaction model 91
contact_params.cc
Contact parameters for use in the surface model 92
contact_params.hh
A class for contact facet parameters, used to create interaction facets for contact in the InteractionSurfaceFactorys 92
contact_surface.cc
Vehicle surface model for the contact interaction models 93
contact_surface.hh
Vehicle surface model for contact 93
contact_surface_factory.cc
Factory that creates an contact surface, from a surface model 94
contact_surface_factory.hh
Factory that creates an contact interaction surface from a surface model 94
contact_utils.hh
This Model is used for utility rotines 95
contact_utils_inline.hh
Define ContactUtils::create_relstate_name, ContactUtils::copy_const_char_to_char 95

line_contact_facet.cc
Define LineContactFacet functions 96
line_contact_facet.hh
The contact facet based on the distance to a line segment centered on the vehicle point 96
line_contact_facet_factory.cc
Factory that creates a LineContactFacetFactory from a Cylinder facet and a ContactParams object 97
line_contact_facet_factory.hh
Creates a line contact facet from an cylinder facet 97
line_contact_pair.cc
LineContactPair class for use with contact interaction model 98
line_contact_pair.hh
Class for a pair of line contact facets for use with contact interaction model 98
line_point_contact_pair.cc
LinePointContactPair class for use with contact interaction model 99
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 99
pair_interaction.cc
A class to define the interaction type for a pair of contact facets 100
pair_interaction.hh
A class to define the interaction type for a pair of contact facets 100
point_contact_facet.cc
Define PointContactFacet functions 101
point_contact_facet.hh
The contact facet based on the distance to a single point, specifically the vehicle point 102
point_contact_facet_factory.cc
Factory that creates a PointContactFacet from a FlatPlateCircular facet and a ContactParams object 102
point_contact_facet_factory.hh
Creates a point contact facet from an circular flat plate facet 103
point_contact_pair.cc
ContactPair class for use with contact interaction model 103
point_contact_pair.hh
Class for a pair of point contact facets for use with contact interaction model 104
spring_pair_interaction.cc
Spring pair interaction for use in the contact model 104
spring_pair_interaction.hh
A class for pair interactions based on a simple spring 105

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.

6.3.1 Detailed Description

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.
- Contact & operator= (const Contact &)=delete
- Contact (const Contact &)=delete
- 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 (const ContactFacet ∗facet_1, const 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 {true}

  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

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 91 of file contact.hh.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 Contact() [1/2]

```
jeod::Contact::Contact ( )
```

Default Constructor.

Definition at line 50 of file contact.cc.

References contact_pairs, and pair_interactions.

8.1.2.2 ∼Contact()

```
jeod::Contact::∼Contact ( )  [virtual]
```

Destructor.

Definition at line 62 of file contact.cc.

References contact_pairs, and pair_interactions.

8.1.2.3 Contact() [2/2]

```
jeod::Contact::Contact (
            const Contact & )  [delete]
```

8.1.3 Member Function Documentation

8.1.3.1 check_contact()

```
void jeod::Contact::check_contact ( )
```

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

Definition at line 86 of file contact.cc.

References active, and contact_pairs.

8.1.3.2 find_interaction()

```
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 251 of file contact.cc.

References pair_interactions.

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

8.1.3.3 initialize_contact()

```
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 106 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 operator=()

```
Contact& jeod::Contact::operator= (
             const Contact & )  [delete]
```

8.1.3.5 register_contact() [1/4]

```
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 167 of file contact.cc.

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

Referenced by register_contact().

8.1.3.6 register_contact() [2/4]

```
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 184 of file contact.cc.

References register_contact().

8.1.3.7 register_contact() [3/4]

```
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 197 of file contact.cc.

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

8.1.3.8 register_contact() [4/4]

```
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 220 of file contact.cc.

References register_contact().

8.1.3.9 register_interaction()

```
void jeod::Contact::register_interaction (
            PairInteraction * interaction )
```

Register a pair interaction.

Parameters

in	*interaction*	PairInteraction to add to list

Definition at line 240 of file contact.cc.

References pair_interactions.

8.1.3.10 unique_pair()

```
bool jeod::Contact::unique_pair (
            const ContactFacet * facet_1,
            const 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 146 of file contact.cc.

References contact_pairs.

Referenced by initialize_contact().

8.1.4 Friends And Related Function Documentation

8.1.4.1 init_attrjeod__Contact

```
void init_attrjeod__Contact ( )  [friend]
```

8.1.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 93 of file contact.hh.

8.1.5 Field Documentation

8.1.5.1 active

```
bool jeod::Contact::active {true}
```

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

trick_units(–)

Definition at line 97 of file contact.hh.

Referenced by check_contact().

8.1.5.2 contact_limit_factor

```
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 103 of file contact.hh.

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

8.1.5.3 contact_pairs

```
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 158 of file contact.hh.

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

8.1.5.4 dyn_manager

```
DynManager∗ jeod::Contact::dyn_manager {}  [protected]
```

Pointer to the dyn_manager so relstates and be successfully initialized.

trick_units(–)

Definition at line 152 of file contact.hh.

Referenced by initialize_contact(), and register_contact().

8.1.5.5 pair_interactions

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

list of all possible pair interaction types

trick_io(∗∗)

Definition at line 163 of file contact.hh.

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

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

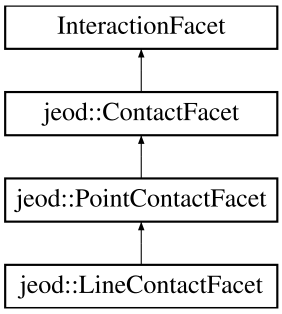
- contact.hh
- contact.cc

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
- ∼ContactFacet () override=default
- ContactFacet & operator= (const ContactFacet &)=delete
- ContactFacet (const ContactFacet &)=delete
- void create_vehicle_point ()

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

  Calculate the max dimension of the facet for range limit determination.
- std::string get_name () 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 ()=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 {true}

  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.

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 85 of file contact_facet.hh.

8.2.2 Constructor & Destructor Documentation

8.2.2.1 ContactFacet() [1/2]

```
jeod::ContactFacet::ContactFacet ( )  [default]
```

8.2.2.2 ∼ContactFacet()

```
jeod::ContactFacet::∼ContactFacet ( )  [override], [default]
```

8.2.2.3 ContactFacet() [2/2]

```
jeod::ContactFacet::ContactFacet (
            const ContactFacet & )  [delete]
```

8.2.3 Member Function Documentation

8.2.3.1 calculate_torque()

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

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

In: N force from one contact interaction.

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

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

8.2.3.2 create_pair() [1/2]

```
virtual ContactPair* jeod::ContactFacet::create_pair ( )  [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. Return: – ContactPair that was created

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

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

8.2.3.3 create_pair() [2/2]

```
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. Return: – ContactPair that was created InOut: – target ContactFacet In: – Contact object used to find the pair interaction

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

8.2.3.4 create_vehicle_point()

```
void jeod::ContactFacet::create_vehicle_point ( )
```

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 50 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 get_name()

```
std::string jeod::ContactFacet::get_name ( ) const  [inline]
```

Accessor for name.

Returns

Point name

Definition at line 173 of file contact_facet.hh.

Referenced by create_vehicle_point().

8.2.3.6 operator=()

```
ContactFacet& jeod::ContactFacet::operator= (
              const ContactFacet &  )  [delete]
```

8.2.3.7 set_max_dimension()

```
virtual void jeod::ContactFacet::set_max_dimension ( )  [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 init_attrjeod__ContactFacet

```
void init_attrjeod__ContactFacet ( )  [friend]
```

8.2.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 87 of file contact_facet.hh.

8.2.5 Field Documentation

8.2.5.1 active

```
bool jeod::ContactFacet::active {true}
```

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

trick_units(–)

Definition at line 91 of file contact_facet.hh.

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

8.2.5.2 max_dimension

```
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 108 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 normal

```
double jeod::ContactFacet::normal[3] {}
```

normal of the facet relative to the vehicle structural frame.

trick_units(–)

Definition at line 118 of file contact_facet.hh.

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

8.2.5.4 position

```
double jeod::ContactFacet::position[3] {}
```

position of the facet in vehicle structural frame.

trick_units(m)

Definition at line 113 of file contact_facet.hh.

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

8.2.5.5 surface_type

```
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 98 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 vehicle_body

```
DynBody* jeod::ContactFacet::vehicle_body {}
```

DynBody associated with this facet for structural frame information.

trick_units(–)

Definition at line 103 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 vehicle_point

```
const BodyRefFrame* jeod::ContactFacet::vehicle_point {}
```

Vehicle point for relstate calculations.

trick_units(--)

Definition at line 123 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>
```

Public Member Functions

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

Static Public Attributes

- static const char ∗ initialization_error = "interactions/contact" "initialization_error"

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

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

  Associated with errors during the setup of the system, before runtime.
- static const char ∗ initialization_warns = "interactions/contact" "initialization_warns"

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

  Associated with warnings given at runtime.
- static const char ∗ runtime_inform = "interactions/contact" "runtime_inform"

  Associated with information given at runtime.

Friends

- class InputProcessor
- void init_attrjeod__ContactMessages ()

8.3.1 Detailed Description

Messages associated with use of the contact model.

Definition at line 85 of file contact_messages.hh.

8.3.2 Constructor & Destructor Documentation

8.3.2.1 ContactMessages() [1/2]

```
jeod::ContactMessages::ContactMessages ( )  [delete]
```

8.3.2.2 ContactMessages() [2/2]

```
jeod::ContactMessages::ContactMessages (
             const ContactMessages & rhs )  [delete]
```

8.3.3 Member Function Documentation

8.3.3.1 operator=()

```
ContactMessages& jeod::ContactMessages::operator= (
             const ContactMessages & rhs )  [delete]
```

8.3.4 Friends And Related Function Documentation

8.3.4.1 init_attrjeod__ContactMessages

```
void init_attrjeod__ContactMessages ( )  [friend]
```

8.3.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 87 of file contact_messages.hh.

8.3.5 Field Documentation

8.3.5.1 initialization_error

```
char const * jeod::ContactMessages::initialization_error = "interactions/contact" "initialization←
_error" [static]
```

Associated with errors during initialization of the contact model.

trick_units(–)

Definition at line 95 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::Contact←
SurfaceFactory::create_surface().

8.3.5.2 initialization_warns

```
char const * jeod::ContactMessages::initialization_warns = "interactions/contact" "initialization←
_warns" [static]
```

Associated with warning during initialization of the contact model.

trick_units(–)

Definition at line 110 of file contact_messages.hh.

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

8.3.5.3 pre_initialization_error

```
char const * jeod::ContactMessages::pre_initialization_error = "interactions/contact" "pre_←
initialization_error" [static]
```

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

trick_units(–)

Definition at line 103 of file contact_messages.hh.

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

8.3.5.4 runtime_error

```
char const * jeod::ContactMessages::runtime_error = "interactions/contact" "runtime_error"
[static]
```

Associated with errors during the runtime of the contact model.

trick_units(–)

Definition at line 99 of file contact_messages.hh.

8.3.5.5 runtime_inform

```
char const * jeod::ContactMessages::runtime_inform = "interactions/contact" "runtime_inform"
[static]
```

Associated with information given at runtime.

trick_units(–)

Definition at line 121 of file contact_messages.hh.

8.3.5.6 runtime_warns

```
char const * jeod::ContactMessages::runtime_warns = "interactions/contact" "runtime_warns"
[static]
```

Associated with warnings given at runtime.

trick_units(–)

Definition at line 114 of file contact_messages.hh.

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

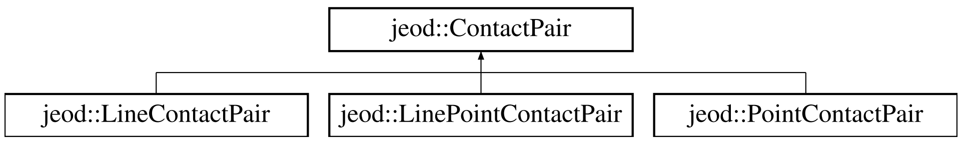
- contact_messages.hh
- contact_messages.cc

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
- virtual ∼ContactPair ()=default
- ContactPair & operator= (const ContactPair &)=delete
- ContactPair (const ContactPair &)=delete
- 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 ()

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

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

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

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 84 of file contact_pair.hh.

8.4.2 Constructor & Destructor Documentation

8.4.2.1 ContactPair() [1/2]

```
jeod::ContactPair::ContactPair ( )  [default]
```

8.4.2.2 ∼ContactPair()

```
virtual jeod::ContactPair::∼ContactPair ( )  [virtual], [default]
```

8.4.2.3 ContactPair() [2/2]

```
jeod::ContactPair::ContactPair (
             const ContactPair &  )  [delete]
```

8.4.3 Member Function Documentation

8.4.3.1 check_tree()

```
bool jeod::ContactPair::check_tree ( )  [virtual]
```

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

Returns

bool

Definition at line 108 of file contact_pair.cc.

References subject, and target.

Referenced by is_active().

8.4.3.2 get_subject()

```
ContactFacet * jeod::ContactPair::get_subject ( )
```

Determine if the pair has a target facet.

Returns

subject ContactFacet

Definition at line 90 of file contact_pair.cc.

References subject.

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

8.4.3.3 get_target()

```
ContactFacet * jeod::ContactPair::get_target ( )
```

Determine if the pair has a target facet.

Returns

target ContactFacet

Definition at line 99 of file contact_pair.cc.

References target.

8.4.3.4 in_contact()

```
virtual void jeod::ContactPair::in_contact ( )  [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 in_range()

```
bool jeod::ContactPair::in_range ( )
```

test whether the pair is in range for interaction

Returns

bool

Definition at line 49 of file contact_pair.cc.

References interaction_distance, and rel_state.

8.4.3.6 initialize_pair()

```
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 initialize_relstate()

```
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 129 of file contact_pair.cc.

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

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

8.4.3.8 is_active()

```
bool jeod::ContactPair::is_active ( )
```

Determine if contact can occur between the two facets.

Returns

bool

Definition at line 64 of file contact_pair.cc.

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

8.4.3.9 is_complete()

```
bool jeod::ContactPair::is_complete ( )
```

Determine if the pair has a target facet.

Returns

bool

Definition at line 77 of file contact_pair.cc.

References target.

8.4.3.10 operator=()

```
ContactPair& jeod::ContactPair::operator= (
             const ContactPair & ) [delete]
```

8.4.4 Friends And Related Function Documentation

8.4.4.1 init_attrjeod__ContactPair

```
void init_attrjeod__ContactPair ( ) [friend]
```

8.4.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 86 of file contact_pair.hh.

8.4.5 Field Documentation

8.4.5.1 interaction

```
PairInteraction* jeod::ContactPair::interaction {}
```

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

trick_units(–)

Definition at line 90 of file contact_pair.hh.

Referenced by jeod::LineContactFacet::create_pair(), jeod::PointContactFacet::create_pair(), jeod::LineContact↩ Pair::in_contact(), jeod::PointContactPair::in_contact(), and jeod::LinePointContactPair::in_contact().

8.4.5.2 interaction_distance

```
double jeod::ContactPair::interaction_distance {}
```

rel_state distance at which in_contact should be called

trick_units(m)

Definition at line 95 of file contact_pair.hh.

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

8.4.5.3 rel_state

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

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

trick_units(–)

Definition at line 145 of file contact_pair.hh.

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

8.4.5.4 subject

```
ContactFacet* jeod::ContactPair::subject {}  [protected]
```

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

trick_units(–)

Definition at line 150 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 target

```
ContactFacet* jeod::ContactPair::target {}  [protected]
```

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

trick_units(–)

Definition at line 155 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:

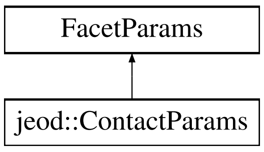
- contact_pair.hh
- contact_pair.cc

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.
- ∼ContactParams () override=default
- ContactParams & operator= (const ContactParams &)=delete
- ContactParams (const ContactParams &)=delete

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 81 of file contact_params.hh.

8.5.2 Constructor & Destructor Documentation

8.5.2.1 ContactParams() [1/2]

```
jeod::ContactParams::ContactParams ( )
```

Default Constructor.

Definition at line 42 of file contact_params.cc.

8.5.2.2 ∼ContactParams()

```
jeod::ContactParams::∼ContactParams ( )  [override], [default]
```

8.5.2.3 ContactParams() [2/2]

```
jeod::ContactParams::ContactParams (
             const ContactParams & )  [delete]
```

8.5.3 Member Function Documentation

8.5.3.1 operator=()

```
ContactParams& jeod::ContactParams::operator= (
             const ContactParams & )  [delete]
```

8.5.4 Friends And Related Function Documentation

8.5.4.1 init_attrjeod__ContactParams

```
void init_attrjeod__ContactParams ( )  [friend]
```

8.5.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 83 of file contact_params.hh.

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

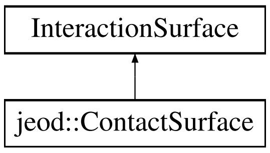
- contact_params.hh
- contact_params.cc

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.
- ∼ContactSurface () override

  Destructor.
- ContactSurface & operator= (const ContactSurface &)=delete
- ContactSurface (const ContactSurface &)=delete
- void allocate_array (unsigned int size) override

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

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

  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.

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 84 of file contact_surface.hh.

8.6.2 Constructor & Destructor Documentation

8.6.2.1 ContactSurface() [1/2]

```
jeod::ContactSurface::ContactSurface ( )
```

Default Constructor.

Definition at line 56 of file contact_surface.cc.

8.6.2.2 ∼ContactSurface()

```
jeod::ContactSurface::∼ContactSurface ( )  [override]
```

Destructor.

Definition at line 64 of file contact_surface.cc.

References contact_facets, and facets_size.

8.6.2.3 ContactSurface() [2/2]

```
jeod::ContactSurface::ContactSurface (
            const ContactSurface & )  [delete]
```

8.6.3 Member Function Documentation

8.6.3.1 allocate_array()

```
void jeod::ContactSurface::allocate_array (
            unsigned int size )  [override]
```

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 75 of file contact_surface.cc.

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

8.6.3.2 allocate_interaction_facet()

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

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 108 of file contact_surface.cc.

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

8.6.3.3 collect_forces_torques()

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

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

Definition at line 177 of file contact_surface.cc.

References contact_facets, contact_force, contact_torque, and facets_size.

8.6.3.4 operator=()

```
ContactSurface& jeod::ContactSurface::operator= (
            const ContactSurface & )  [delete]
```

8.6.4 Friends And Related Function Documentation

8.6.4.1 init_attrjeod__ContactSurface

```
void init_attrjeod__ContactSurface ( )  [friend]
```

8.6.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 86 of file contact_surface.hh.

8.6.5 Field Documentation

8.6.5.1 contact_facets

```
ContactFacet** jeod::ContactSurface::contact_facets {}
```

An array of pointers to contact interaction facets.

trick_units(–)

Definition at line 95 of file contact_surface.hh.

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

8.6.5.2 contact_force

```
double jeod::ContactSurface::contact_force[3] {}
```

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

trick_units(N)

Definition at line 105 of file contact_surface.hh.

Referenced by collect_forces_torques().

8.6.5.3 contact_torque

```
double jeod::ContactSurface::contact_torque[3] {}
```

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

trick_units(N/m)

Definition at line 110 of file contact_surface.hh.

Referenced by collect_forces_torques().

8.6.5.4 facets_size

```
unsigned int jeod::ContactSurface::facets_size {}
```

Size of the contact_facets array.

trick_units(count)

Definition at line 100 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:

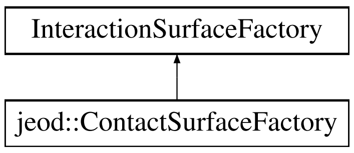
- contact_surface.hh
- contact_surface.cc

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.
- ∼ContactSurfaceFactory () override=default
- ContactSurfaceFactory & operator= (const ContactSurfaceFactory &)=delete
- ContactSurfaceFactory (const ContactSurfaceFactory &)=delete
- void create_surface (SurfaceModel ∗surface, InteractionSurface ∗inter_surface) override

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

  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.

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 86 of file contact_surface_factory.hh.

8.7.2 Constructor & Destructor Documentation

8.7.2.1 ContactSurfaceFactory() [1/2]

```
jeod::ContactSurfaceFactory::ContactSurfaceFactory ( )
```

Default Constructor.

Definition at line 48 of file contact_surface_factory.cc.

References line_contact_facet_factory, and point_contact_facet_factory.

8.7.2.2 ∼ContactSurfaceFactory()

```
jeod::ContactSurfaceFactory::∼ContactSurfaceFactory ( )  [override], [default]
```

8.7.2.3 ContactSurfaceFactory() [2/2]

```
jeod::ContactSurfaceFactory::ContactSurfaceFactory (
            const ContactSurfaceFactory &  )  [delete]
```

8.7.3 Member Function Documentation

8.7.3.1 add_facet_params()

```
void jeod::ContactSurfaceFactory::add_facet_params (
            FacetParams * to_add )  [override]
```

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 176 of file contact_surface_factory.cc.

References jeod::ContactMessages::pre_initialization_error.

8.7.3.2 create_surface()

```
void jeod::ContactSurfaceFactory::create_surface (
            SurfaceModel * surface,
            InteractionSurface * inter_surface )  [override]
```

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 65 of file contact_surface_factory.cc.

References jeod::ContactMessages::initialization_error.

8.7.3.3 operator=()

```
ContactSurfaceFactory& jeod::ContactSurfaceFactory::operator= (
             const ContactSurfaceFactory & ) [delete]
```

8.7.4 Friends And Related Function Documentation

8.7.4.1 init_attrjeod__ContactSurfaceFactory

```
void init_attrjeod__ContactSurfaceFactory ( ) [friend]
```

8.7.4.2 InputProcessor

```
friend class InputProcessor [friend]
```

Definition at line 88 of file contact_surface_factory.hh.

8.7.5 Field Documentation

8.7.5.1 line_contact_facet_factory

```
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 115 of file contact_surface_factory.hh.

Referenced by ContactSurfaceFactory().

8.7.5.2 point_contact_facet_factory

```
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 110 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 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 72 of file contact_utils.hh.

8.8.2 Member Function Documentation

8.8.2.1 create_relstate_name()

```
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 93 of file contact_utils_inline.hh.

8.8.2.2 dist_line_segments()

```
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:

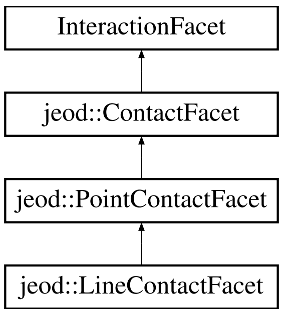
- contact_utils.hh
- contact_utils_inline.hh

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.
- ∼LineContactFacet () override=default
- LineContactFacet & operator= (const LineContactFacet &)=delete
- LineContactFacet (const LineContactFacet &)=delete
- ContactPair ∗ create_pair () override

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

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

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

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

  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.

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 87 of file line_contact_facet.hh.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 LineContactFacet() [1/2]

```
jeod::LineContactFacet::LineContactFacet ( )
```

Default constructor.

Definition at line 52 of file line_contact_facet.cc.

8.9.2.2 ∼LineContactFacet()

```
jeod::LineContactFacet::∼LineContactFacet ( )  [override], [default]
```

8.9.2.3 LineContactFacet() [2/2]

```
jeod::LineContactFacet::LineContactFacet (
            const LineContactFacet &  )  [delete]
```

8.9.3 Member Function Documentation

8.9.3.1 calculate_contact_point()

```
void jeod::LineContactFacet::calculate_contact_point (
            double nvec[3] )  [override], [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 159 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 calculate_torque()

```
void jeod::LineContactFacet::calculate_torque (
            double * tmp_force )  [override], [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.

Definition at line 212 of file line_contact_facet.cc.

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

8.9.3.3 create_pair() [1/2]

```
ContactPair * jeod::LineContactFacet::create_pair ( )  [override], [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.

Definition at line 64 of file line_contact_facet.cc.

References jeod::LineContactPair::initialize_pair().

8.9.3.4 create_pair() [2/2]

```
ContactPair * jeod::LineContactFacet::create_pair (
            ContactFacet * target,
            Contact * contact ) [override], [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

Implements jeod::ContactFacet.

Definition at line 83 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 operator=()

```
LineContactFacet& jeod::LineContactFacet::operator= (
            const LineContactFacet & ) [delete]
```

8.9.3.6 set_max_dimension()

```
void jeod::LineContactFacet::set_max_dimension ( ) [override], [virtual]
```

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

Implements jeod::ContactFacet.

Definition at line 202 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 init_attrjeod__LineContactFacet

```
void init_attrjeod__LineContactFacet ( )  [friend]
```

8.9.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 89 of file line_contact_facet.hh.

8.9.5 Field Documentation

8.9.5.1 length

```
double jeod::LineContactFacet::length {}
```

length of the line along the vehicle point x axis.

trick_units(m)

Definition at line 93 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:

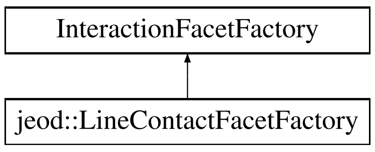
- line_contact_facet.hh
- line_contact_facet.cc

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 () override=default
- LineContactFacetFactory & operator= (const LineContactFacetFactory &)=delete
- LineContactFacetFactory (const LineContactFacetFactory &)=delete
- InteractionFacet ∗ create_facet (Facet ∗facet, FacetParams ∗params) override

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

  PointContactFacetFactory specific implementation of this function.

Friends

- class InputProcessor
- void init_attrjeod__LineContactFacetFactory ()

8.10.1 Detailed Description

Creates a PointContactFacet from an InteractionFacet.

Definition at line 86 of file line_contact_facet_factory.hh.

8.10.2 Constructor & Destructor Documentation

8.10.2.1 LineContactFacetFactory() [1/2]

```
jeod::LineContactFacetFactory::LineContactFacetFactory ( )
```

Default Constructor.

Definition at line 52 of file line_contact_facet_factory.cc.

8.10.2.2 ∼LineContactFacetFactory()

```
jeod::LineContactFacetFactory::∼LineContactFacetFactory ( )  [override], [default]
```

8.10.2.3 LineContactFacetFactory() [2/2]

```
jeod::LineContactFacetFactory::LineContactFacetFactory (
            const LineContactFacetFactory &  )  [delete]
```

8.10.3 Member Function Documentation

8.10.3.1 create_facet()

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

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 65 of file line_contact_facet_factory.cc.

References jeod::ContactFacet::create_vehicle_point(), jeod::ContactMessages::initialization_error, jeod::Line↩ ContactFacet::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 is_correct_factory()

```
bool jeod::LineContactFacetFactory::is_correct_factory (
            Facet * facet ) [override]
```

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 135 of file line_contact_facet_factory.cc.

8.10.3.3 operator=()

```
LineContactFacetFactory& jeod::LineContactFacetFactory::operator= (
             const LineContactFacetFactory &  )  [delete]
```

8.10.4 Friends And Related Function Documentation

8.10.4.1 init_attrjeod__LineContactFacetFactory

```
void init_attrjeod__LineContactFacetFactory ( )  [friend]
```

8.10.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 88 of file line_contact_facet_factory.hh.

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

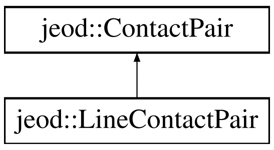
- line_contact_facet_factory.hh
- line_contact_facet_factory.cc

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
- ∼LineContactPair () override=default
- LineContactPair & operator= (const LineContactPair &)=delete
- LineContactPair (const LineContactPair &)=delete
- void in_contact () override

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

  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.

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 84 of file line_contact_pair.hh.

8.11.2 Constructor & Destructor Documentation

8.11.2.1 LineContactPair() [1/2]

```
jeod::LineContactPair::LineContactPair ( )  [default]
```

8.11.2.2 ∼LineContactPair()

```
jeod::LineContactPair::∼LineContactPair ( )  [override], [default]
```

8.11.2.3 LineContactPair() [2/2]

```
jeod::LineContactPair::LineContactPair (
            const LineContactPair &  )  [delete]
```

8.11.3 Member Function Documentation

8.11.3.1 in_contact()

```
void jeod::LineContactPair::in_contact ( )  [override], [virtual]
```

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

Implements jeod::ContactPair.

Definition at line 48 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 initialize_pair()

```
void jeod::LineContactPair::initialize_pair (
            ContactFacet * subject_facet,
            ContactFacet * target_facet )  [override], [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 138 of file line_contact_pair.cc.

References 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 operator=()

```
LineContactPair& jeod::LineContactPair::operator= (
             const LineContactPair &  )  [delete]
```

8.11.4 Friends And Related Function Documentation

8.11.4.1 init_attrjeod__LineContactPair

```
void init_attrjeod__LineContactPair ( )  [friend]
```

8.11.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 86 of file line_contact_pair.hh.

8.11.5 Field Documentation

8.11.5.1 line_subject

```
LineContactFacet* jeod::LineContactPair::line_subject {}
```

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

trick_units(–)

Definition at line 90 of file line_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

8.11.5.2 line_target

```
LineContactFacet* jeod::LineContactPair::line_target {}
```

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

trick_units(–)

Definition at line 95 of file line_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

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

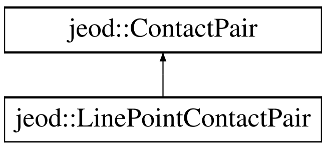
- line_contact_pair.hh
- line_contact_pair.cc

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
- ∼LinePointContactPair () override=default
- LinePointContactPair & operator= (const LineContactPair &)=delete
- LinePointContactPair (const LineContactPair &)=delete
- void in_contact () override

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

  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.

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 85 of file line_point_contact_pair.hh.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 LinePointContactPair() [1/2]

```
jeod::LinePointContactPair::LinePointContactPair ( )  [default]
```

8.12.2.2 ∼LinePointContactPair()

```
jeod::LinePointContactPair::∼LinePointContactPair ( )  [override], [default]
```

8.12.2.3 LinePointContactPair() [2/2]

```
jeod::LinePointContactPair::LinePointContactPair (
            const LineContactPair &  )  [delete]
```

8.12.3 Member Function Documentation

8.12.3.1 in_contact()

```
void jeod::LinePointContactPair::in_contact ( )  [override], [virtual]
```

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

Implements jeod::ContactPair.

Definition at line 49 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 initialize_pair()

```
void jeod::LinePointContactPair::initialize_pair (
            ContactFacet * subject_facet,
            ContactFacet * target_facet )  [override], [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 132 of file line_point_contact_pair.cc.

References 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 operator=()

```
LinePointContactPair& jeod::LinePointContactPair::operator= (
             const LineContactPair &  )  [delete]
```

8.12.4 Friends And Related Function Documentation

8.12.4.1 init_attrjeod__LinePointContactPair

```
void init_attrjeod__LinePointContactPair ( )  [friend]
```

8.12.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 87 of file line_point_contact_pair.hh.

8.12.5 Field Documentation

8.12.5.1 line_subject

`LineContactFacet* jeod::LinePointContactPair::line_subject {}`

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

trick_units(–)

Definition at line 91 of file line_point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

8.12.5.2 point_target

`PointContactFacet* jeod::LinePointContactPair::point_target {}`

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

trick_units(–)

Definition at line 96 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:

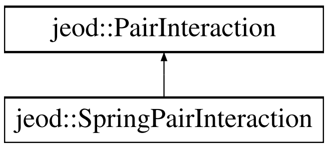
- line_point_contact_pair.hh
- line_point_contact_pair.cc

8.13 jeod::PairInteraction Class Reference

Simple spring contact parameters.

`#include <pair_interaction.hh>`

Inheritance diagram for jeod::PairInteraction:



Public Member Functions

- PairInteraction ()=default
- virtual ∼PairInteraction ()=default
- PairInteraction & operator= (const PairInteraction &)=delete
- PairInteraction (const PairInteraction &)=delete
- 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

- std::string params_1

  contact param type that defines this pair interaction.
- std::string params_2

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

  magnitude of the friction force on the contact surfaces.

Friends

- class InputProcessor
- void init_attrjeod__PairInteraction ()

8.13.1 Detailed Description

Simple spring contact parameters.

Definition at line 89 of file pair_interaction.hh.

8.13.2 Constructor & Destructor Documentation

8.13.2.1 PairInteraction() [1/2]

```
jeod::PairInteraction::PairInteraction ( )  [default]
```

8.13.2.2 ∼PairInteraction()

```
virtual jeod::PairInteraction::∼PairInteraction ( )  [virtual], [default]
```

8.13.2.3 PairInteraction() [2/2]

```
jeod::PairInteraction::PairInteraction (
            const PairInteraction &  )  [delete]
```

8.13.3 Member Function Documentation

8.13.3.1 calculate_forces()

```
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 is_correct_interaction()

```
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 50 of file pair_interaction.cc.

References params_1, and params_2.

8.13.3.3 operator=()

```
PairInteraction& jeod::PairInteraction::operator= (
             const PairInteraction &  )  [delete]
```

8.13.4 Friends And Related Function Documentation

8.13.4.1 init_attrjeod__PairInteraction

```
void init_attrjeod__PairInteraction ( )  [friend]
```

8.13.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 91 of file pair_interaction.hh.

8.13.5 Field Documentation

8.13.5.1 friction_mag

```
double jeod::PairInteraction::friction_mag {}
```

magnitude of the friction force on the contact surfaces.

trick_units(N)

Definition at line 105 of file pair_interaction.hh.

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

8.13.5.2 params_1

```
std::string jeod::PairInteraction::params_1
```

contact param type that defines this pair interaction.

trick_units(–)

Definition at line 95 of file pair_interaction.hh.

Referenced by is_correct_interaction().

8.13.5.3 params_2

```
std::string jeod::PairInteraction::params_2
```

contact param type that defines this pair interaction.

trick_units(–)

Definition at line 100 of file pair_interaction.hh.

Referenced by is_correct_interaction().

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

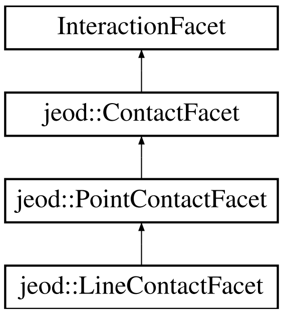
- pair_interaction.hh
- pair_interaction.cc

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.
- ∼PointContactFacet () override=default
- PointContactFacet & operator= (const PointContactFacet &)=delete
- PointContactFacet (const PointContactFacet &)=delete
- ContactPair ∗ create_pair () override

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

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

  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.
- void calculate_torque (double ∗tmp_force) override

  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.

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 87 of file point_contact_facet.hh.

8.14.2 Constructor & Destructor Documentation

8.14.2.1 PointContactFacet() [1/2]

```
jeod::PointContactFacet::PointContactFacet ( )
```

Default constructor.

Definition at line 50 of file point_contact_facet.cc.

8.14.2.2 ∼PointContactFacet()

```
jeod::PointContactFacet::∼PointContactFacet ( )  [override], [default]
```

8.14.2.3 PointContactFacet() [2/2]

```
jeod::PointContactFacet::PointContactFacet (
            const PointContactFacet &  )  [delete]
```

8.14.3 Member Function Documentation

8.14.3.1 calculate_contact_point()

```
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 119 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 calculate_torque()

```
void jeod::PointContactFacet::calculate_torque (
            double * tmp_force )  [override], [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.

Definition at line 141 of file point_contact_facet.cc.

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

8.14.3.3 create_pair() [1/2]

```
ContactPair * jeod::PointContactFacet::create_pair ( )  [override], [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.

Definition at line 61 of file point_contact_facet.cc.

References jeod::PointContactPair::initialize_pair().

8.14.3.4 create_pair() [2/2]

```
ContactPair * jeod::PointContactFacet::create_pair (
            ContactFacet * target,
            Contact * contact )  [override], [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.

Definition at line 80 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 operator=()

```
PointContactFacet& jeod::PointContactFacet::operator= (
             const PointContactFacet &  )  [delete]
```

8.14.3.6 set_max_dimension()

```
void jeod::PointContactFacet::set_max_dimension ( )  [override], [virtual]
```

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

Implements jeod::ContactFacet.

Definition at line 131 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 init_attrjeod__PointContactFacet

```
void init_attrjeod__PointContactFacet ( )  [friend]
```

8.14.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 89 of file point_contact_facet.hh.

8.14.5 Field Documentation

8.14.5.1 contact_point

```
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 100 of file point_contact_facet.hh.

Referenced by jeod::LineContactFacet::calculate_contact_point(), calculate_contact_point(), jeod::LineContact↩ Facet::calculate_torque(), calculate_torque(), jeod::LineContactPair::in_contact(), jeod::PointContactPair::in_↩ contact(), and jeod::LinePointContactPair::in_contact().

8.14.5.2 radius

```
double jeod::PointContactFacet::radius {}
```

radius from the point at which contact takes place.

trick_units(m)

Definition at line 93 of file point_contact_facet.hh.

Referenced by jeod::LineContactFacet::calculate_contact_point(), calculate_contact_point(), jeod::PointContact↩ FacetFactory::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:

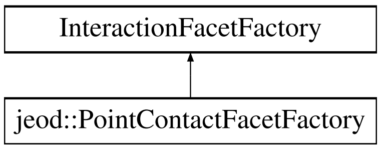
- point_contact_facet.hh
- point_contact_facet.cc

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 () override=default
- PointContactFacetFactory & operator= (const PointContactFacetFactory &)=delete
- PointContactFacetFactory (const PointContactFacetFactory &)=delete
- InteractionFacet ∗ create_facet (Facet ∗facet, FacetParams ∗params) override

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

  PointContactFacetFactory specific implementation of this function.

Friends

- class InputProcessor
- void init_attrjeod__PointContactFacetFactory ()

8.15.1 Detailed Description

Creates a PointContactFacet from an InteractionFacet.

Definition at line 86 of file point_contact_facet_factory.hh.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 PointContactFacetFactory() [1/2]

```
jeod::PointContactFacetFactory::PointContactFacetFactory ( )
```

Default Constructor.

Definition at line 51 of file point_contact_facet_factory.cc.

8.15.2.2 ∼PointContactFacetFactory()

```
jeod::PointContactFacetFactory::∼PointContactFacetFactory ( )  [override], [default]
```

8.15.2.3 PointContactFacetFactory() [2/2]

```
jeod::PointContactFacetFactory::PointContactFacetFactory (
            const PointContactFacetFactory &  )  [delete]
```

8.15.3 Member Function Documentation

8.15.3.1 create_facet()

```
InteractionFacet * jeod::PointContactFacetFactory::create_facet (
            Facet * facet,
            FacetParams * params )  [override]
```

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 64 of file point_contact_facet_factory.cc.

References jeod::ContactFacet::create_vehicle_point(), jeod::ContactMessages::initialization_error, jeod::↩ContactFacet::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 is_correct_factory()

```
bool jeod::PointContactFacetFactory::is_correct_factory (
            Facet * facet )  [override]
```

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 133 of file point_contact_facet_factory.cc.

8.15.3.3 operator=()

```
PointContactFacetFactory& jeod::PointContactFacetFactory::operator= (
             const PointContactFacetFactory & )  [delete]
```

8.15.4 Friends And Related Function Documentation

8.15.4.1 init_attrjeod__PointContactFacetFactory

```
void init_attrjeod__PointContactFacetFactory ( )  [friend]
```

8.15.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 88 of file point_contact_facet_factory.hh.

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

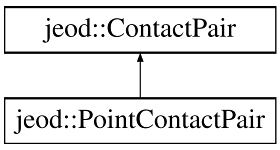
- point_contact_facet_factory.hh
- point_contact_facet_factory.cc

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
- ∼PointContactPair () override=default
- PointContactPair & operator= (const PointContactPair &)=delete
- PointContactPair (const PointContactPair &)=delete
- void in_contact () override

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

  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.

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 84 of file point_contact_pair.hh.

8.16.2 Constructor & Destructor Documentation

8.16.2.1 PointContactPair() [1/2]

```
jeod::PointContactPair::PointContactPair ( )  [default]
```

8.16.2.2 ∼PointContactPair()

```
jeod::PointContactPair::∼PointContactPair ( )  [override], [default]
```

8.16.2.3 PointContactPair() [2/2]

```
jeod::PointContactPair::PointContactPair (
            const PointContactPair & )  [delete]
```

8.16.3 Member Function Documentation

8.16.3.1 in_contact()

```
void jeod::PointContactPair::in_contact ( )  [override], [virtual]
```

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

Implements jeod::ContactPair.

Definition at line 47 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 initialize_pair()

```
void jeod::PointContactPair::initialize_pair (
            ContactFacet * subject_facet,
            ContactFacet * target_facet )  [override], [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 96 of file point_contact_pair.cc.

References 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 operator=()

```
PointContactPair& jeod::PointContactPair::operator= (
              const PointContactPair &  )  [delete]
```

8.16.4 Friends And Related Function Documentation

8.16.4.1 init_attrjeod__PointContactPair

```
void init_attrjeod__PointContactPair ( )  [friend]
```

8.16.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 86 of file point_contact_pair.hh.

8.16.5 Field Documentation

8.16.5.1 point_subject

```
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 90 of file point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

8.16.5.2 point_target

```
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 95 of file point_contact_pair.hh.

Referenced by in_contact(), and initialize_pair().

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

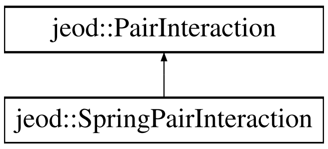
- point_contact_pair.hh
- point_contact_pair.cc

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
- ∼SpringPairInteraction () override=default
- SpringPairInteraction & operator= (const SpringPairInteraction &)=delete
- SpringPairInteraction (const SpringPairInteraction &)=delete
- void calculate_forces (ContactFacet ∗subject, ContactFacet ∗target, RelativeDerivedState ∗rel_state, double ∗penetration_vector, double ∗rel_velocity) override

  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.

Friends

- class InputProcessor
- void init_attrjeod__SpringPairInteraction ()

8.17.1 Detailed Description

Simple spring contact parameters.

Definition at line 83 of file spring_pair_interaction.hh.

8.17.2 Constructor & Destructor Documentation

8.17.2.1 SpringPairInteraction() [1/2]

```
jeod::SpringPairInteraction::SpringPairInteraction ( )  [default]
```

8.17.2.2 ∼SpringPairInteraction()

```
jeod::SpringPairInteraction::∼SpringPairInteraction ( )  [override], [default]
```

8.17.2.3 SpringPairInteraction() [2/2]

```
jeod::SpringPairInteraction::SpringPairInteraction (
           const SpringPairInteraction &  )  [delete]
```

8.17.3 Member Function Documentation

8.17.3.1 calculate_forces()

```
void jeod::SpringPairInteraction::calculate_forces (
           ContactFacet ∗ subject,
           ContactFacet ∗ target,
           RelativeDerivedState ∗ rel_state,
           double ∗ penetration_vector,
           double ∗ rel_velocity )  [override], [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 57 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 operator=()

```
SpringPairInteraction& jeod::SpringPairInteraction::operator= (
             const SpringPairInteraction &  )  [delete]
```

8.17.4 Friends And Related Function Documentation

8.17.4.1 init_attrjeod__SpringPairInteraction

```
void init_attrjeod__SpringPairInteraction ( )  [friend]
```

8.17.4.2 InputProcessor

```
friend class InputProcessor  [friend]
```

Definition at line 85 of file spring_pair_interaction.hh.

8.17.5 Field Documentation

8.17.5.1 damping_b

```
double jeod::SpringPairInteraction::damping_b {}
```

Spring damping constant.

trick_units(N∗s/m)

Definition at line 94 of file spring_pair_interaction.hh.

Referenced by calculate_forces().

8.17.5.2 mu

```
double jeod::SpringPairInteraction::mu {}
```

Coefficent of friction.

trick_units(--)

Definition at line 99 of file spring_pair_interaction.hh.

Referenced by calculate_forces().

8.17.5.3 spring_k

```
double jeod::SpringPairInteraction::spring_k {}
```

Spring stiffness constant.

trick_units(N/m)

Definition at line 89 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.

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.

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/container/include/pointer_list.hh"
#include "utils/sim_interface/include/jeod_class.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)

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/numerical.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.

9.5 contact_facet.hh File Reference

Individual facets for use with contact interaction models.

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

9.6 contact_messages.cc File Reference

Implement contact_messages.

```
#include "utils/message/include/make_message_code.hh"
#include "../include/contact_messages.hh"
```

Namespaces

- jeod

  Namespace jeod.

Macros

- #define MAKE_CONTACT_MESSAGE_CODE(id) JEOD_MAKE_MESSAGE_CODE(ContactMessages, "interactions/contact", id)

9.6.1 Detailed Description

Implement contact_messages.

9.6.2 Macro Definition Documentation

9.6.2.1 MAKE_CONTACT_MESSAGE_CODE

```
#define MAKE_CONTACT_MESSAGE_CODE(
              id ) JEOD_MAKE_MESSAGE_CODE(ContactMessages, "interactions/contact", id)
```

Definition at line 43 of file contact_messages.cc.

9.7 contact_messages.hh File Reference

Contact message for message handling.

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

9.8 contact_pair.cc File Reference

ContactPair class for use with contact interaction model.

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

Namespaces

- jeod

    Namespace jeod.

9.8.1 Detailed Description

ContactPair class for use with contact interaction model.

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.

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

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.

9.12 contact_surface.cc File Reference

Vehicle surface model for the contact interaction models.

```
#include <cstddef>
#include <typeinfo>
#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_facet.hh"
#include "../include/contact_messages.hh"
#include "../include/contact_surface.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.12.1 Detailed Description

Vehicle surface model for the contact interaction models.

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/class_declarations.hh"
#include "utils/surface_model/include/interaction_surface.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.

9.14 contact_surface_factory.cc File Reference

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

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

Namespaces

- jeod

    Namespace jeod.

9.14.1 Detailed Description

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

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/class_declarations.hh"
#include "utils/surface_model/include/interaction_surface_factory.hh"
#include "line_contact_facet_factory.hh"
#include "point_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.

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.

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_messages.hh"
#include "contact_utils.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.17.1 Detailed Description

Define ContactUtils::create_relstate_name, ContactUtils::copy_const_char_to_char.

9.18 line_contact_facet.cc File Reference

Define LineContactFacet functions.

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

Namespaces

- jeod

  Namespace jeod.

9.18.1 Detailed Description

Define LineContactFacet functions.

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

9.20 line_contact_facet_factory.cc File Reference

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

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

Namespaces

- jeod

  Namespace jeod.

9.20.1 Detailed Description

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

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.

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/contact_utils.hh"
#include "../include/line_contact_facet.hh"
#include "../include/line_contact_pair.hh"
#include "../include/pair_interaction.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.22.1 Detailed Description

LineContactPair class for use with contact interaction model.

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 "class_declarations.hh"
#include "contact_pair.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.

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/contact_utils.hh"
#include "../include/line_contact_facet.hh"
#include "../include/line_point_contact_pair.hh"
#include "../include/pair_interaction.hh"
#include "../include/point_contact_facet.hh"
```

Namespaces

- jeod

  Namespace jeod.

9.24.1 Detailed Description

LinePointContactPair class for use with contact interaction model.

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 "class_declarations.hh"
#include "contact_pair.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.

9.26 pair_interaction.cc File Reference

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

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

9.27 pair_interaction.hh File Reference

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

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

9.28 point_contact_facet.cc File Reference

Define PointContactFacet functions.

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

Namespaces

- jeod

  Namespace jeod.

9.28.1 Detailed Description

Define PointContactFacet functions.

9.29 point_contact_facet.hh File Reference

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

```
#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"
#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.

9.30 point_contact_facet_factory.cc File Reference

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

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

Namespaces

- jeod

  Namespace jeod.

9.30.1 Detailed Description

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

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.

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

Namespaces

- jeod

  Namespace jeod.

9.32.1 Detailed Description

ContactPair class for use with contact interaction model.

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 "class_declarations.hh"
#include "contact_pair.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.

9.34 spring_pair_interaction.cc File Reference

spring pair interaction for use in the contact model

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

Namespaces

- jeod

  Namespace jeod.

9.34.1 Detailed Description

spring pair interaction for use in the contact model

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.

Index

∼Contact
  jeod::Contact, 18
∼ContactFacet
  jeod::ContactFacet, 26
∼ContactPair
  jeod::ContactPair, 36
∼ContactParams
  jeod::ContactParams, 42
∼ContactSurface
  jeod::ContactSurface, 44
∼ContactSurfaceFactory
  jeod::ContactSurfaceFactory, 48
∼LineContactFacet
  jeod::LineContactFacet, 54
∼LineContactFacetFactory
  jeod::LineContactFacetFactory, 58
∼LineContactPair
  jeod::LineContactPair, 61
∼LinePointContactPair
  jeod::LinePointContactPair, 65
∼PairInteraction
  jeod::PairInteraction, 68
∼PointContactFacet
  jeod::PointContactFacet, 73
∼PointContactFacetFactory
  jeod::PointContactFacetFactory, 77
∼PointContactPair
  jeod::PointContactPair, 80
∼SpringPairInteraction
  jeod::SpringPairInteraction, 84

active
  jeod::Contact, 23
  jeod::ContactFacet, 29
add_facet_params
  jeod::ContactSurfaceFactory, 49
allocate_array
  jeod::ContactSurface, 44
allocate_interaction_facet
  jeod::ContactSurface, 45

calculate_contact_point
  jeod::LineContactFacet, 54
  jeod::PointContactFacet, 73
calculate_forces
  jeod::PairInteraction, 69
  jeod::SpringPairInteraction, 84
calculate_torque
  jeod::ContactFacet, 26
  jeod::LineContactFacet, 55
  jeod::PointContactFacet, 73
check_contact
  jeod::Contact, 19
check_tree
  jeod::ContactPair, 36
class_declarations.hh, 87
collect_forces_torques
  jeod::ContactSurface, 45
Contact, 13
  jeod::Contact, 18, 19
contact.cc, 87
contact.hh, 88
contact_facet.cc, 88
contact_facet.hh, 89
contact_facets
  jeod::ContactSurface, 46
contact_force
  jeod::ContactSurface, 46
contact_limit_factor
  jeod::Contact, 23
contact_messages.cc, 89
  MAKE_CONTACT_MESSAGE_CODE, 90
contact_messages.hh, 90
contact_pair.cc, 91
contact_pair.hh, 91
contact_pairs
  jeod::Contact, 23
contact_params.cc, 92
contact_params.hh, 92
contact_point
  jeod::PointContactFacet, 76
contact_surface.cc, 93
contact_surface.hh, 93
contact_surface_factory.cc, 94
contact_surface_factory.hh, 94
contact_torque
  jeod::ContactSurface, 46
contact_utils.hh, 95
contact_utils_inline.hh, 95
ContactFacet
  jeod::ContactFacet, 26
ContactMessages
  jeod::ContactMessages, 32
ContactPair
  jeod::ContactPair, 36
ContactParams
  jeod::ContactParams, 42
ContactSurface
  jeod::ContactSurface, 44

ContactSurfaceFactory
    jeod::ContactSurfaceFactory, 48, 49
create_facet
    jeod::LineContactFacetFactory, 59
    jeod::PointContactFacetFactory, 78
create_pair
    jeod::ContactFacet, 27
    jeod::LineContactFacet, 55
    jeod::PointContactFacet, 74
create_relstate_name
    jeod::ContactUtils, 51
create_surface
    jeod::ContactSurfaceFactory, 49
create_vehicle_point
    jeod::ContactFacet, 27

damping_b
    jeod::SpringPairInteraction, 85
dist_line_segments
    jeod::ContactUtils, 52
dyn_manager
    jeod::Contact, 24

facets_size
    jeod::ContactSurface, 47
find_interaction
    jeod::Contact, 19
friction_mag
    jeod::PairInteraction, 70

get_name
    jeod::ContactFacet, 28
get_subject
    jeod::ContactPair, 36
get_target
    jeod::ContactPair, 37

in_contact
    jeod::ContactPair, 37
    jeod::LineContactPair, 62
    jeod::LinePointContactPair, 65
    jeod::PointContactPair, 81
in_range
    jeod::ContactPair, 37
init_attrjeod__Contact
    jeod::Contact, 22
init_attrjeod__ContactFacet
    jeod::ContactFacet, 29
init_attrjeod__ContactMessages
    jeod::ContactMessages, 32
init_attrjeod__ContactPair
    jeod::ContactPair, 39
init_attrjeod__ContactParams
    jeod::ContactParams, 42
init_attrjeod__ContactSurface
    jeod::ContactSurface, 46
init_attrjeod__ContactSurfaceFactory
    jeod::ContactSurfaceFactory, 50
init_attrjeod__LineContactFacet
    jeod::LineContactFacet, 57
init_attrjeod__LineContactFacetFactory
    jeod::LineContactFacetFactory, 60
init_attrjeod__LineContactPair
    jeod::LineContactPair, 63
init_attrjeod__LinePointContactPair
    jeod::LinePointContactPair, 66
init_attrjeod__PairInteraction
    jeod::PairInteraction, 70
init_attrjeod__PointContactFacet
    jeod::PointContactFacet, 75
init_attrjeod__PointContactFacetFactory
    jeod::PointContactFacetFactory, 79
init_attrjeod__PointContactPair
    jeod::PointContactPair, 82
init_attrjeod__SpringPairInteraction
    jeod::SpringPairInteraction, 85
initialization_error
    jeod::ContactMessages, 33
initialization_warns
    jeod::ContactMessages, 33
initialize_contact
    jeod::Contact, 20
initialize_pair
    jeod::ContactPair, 37
    jeod::LineContactPair, 62
    jeod::LinePointContactPair, 65
    jeod::PointContactPair, 81
initialize_relstate
    jeod::ContactPair, 38
InputProcessor
    jeod::Contact, 23
    jeod::ContactFacet, 29
    jeod::ContactMessages, 32
    jeod::ContactPair, 39
    jeod::ContactParams, 42
    jeod::ContactSurface, 46
    jeod::ContactSurfaceFactory, 50
    jeod::LineContactFacet, 57
    jeod::LineContactFacetFactory, 60
    jeod::LineContactPair, 63
    jeod::LinePointContactPair, 66
    jeod::PairInteraction, 70
    jeod::PointContactFacet, 75
    jeod::PointContactFacetFactory, 79
    jeod::PointContactPair, 82
    jeod::SpringPairInteraction, 85
interaction
    jeod::ContactPair, 39
interaction_distance
    jeod::ContactPair, 40
Interactions, 12
is_active
    jeod::ContactPair, 38
is_complete
    jeod::ContactPair, 38
is_correct_factory
    jeod::LineContactFacetFactory, 59

jeod::PointContactFacetFactory, 78
is_correct_interaction
jeod::PairInteraction, 69

jeod, 15
jeod::Contact, 17
∼Contact, 18
active, 23
check_contact, 19
Contact, 18, 19
contact_limit_factor, 23
contact_pairs, 23
dyn_manager, 24
find_interaction, 19
init_attrjeod__Contact, 22
initialize_contact, 20
InputProcessor, 23
operator=, 20
pair_interactions, 24
register_contact, 20, 21
register_interaction, 22
unique_pair, 22
jeod::ContactFacet, 25
∼ContactFacet, 26
active, 29
calculate_torque, 26
ContactFacet, 26
create_pair, 27
create_vehicle_point, 27
get_name, 28
init_attrjeod__ContactFacet, 29
InputProcessor, 29
max_dimension, 29
normal, 29
operator=, 28
position, 30
set_max_dimension, 28
surface_type, 30
vehicle_body, 30
vehicle_point, 30
jeod::ContactMessages, 31
ContactMessages, 32
init_attrjeod__ContactMessages, 32
initialization_error, 33
initialization_warns, 33
InputProcessor, 32
operator=, 32
pre_initialization_error, 33
runtime_error, 33
runtime_inform, 34
runtime_warns, 34
jeod::ContactPair, 34
∼ContactPair, 36
check_tree, 36
ContactPair, 36
get_subject, 36
get_target, 37
in_contact, 37
in_range, 37
init_attrjeod__ContactPair, 39
initialize_pair, 37
initialize_relstate, 38
InputProcessor, 39
interaction, 39
interaction_distance, 40
is_active, 38
is_complete, 38
operator=, 39
rel_state, 40
subject, 40
target, 40
jeod::ContactParams, 41
∼ContactParams, 42
ContactParams, 42
init_attrjeod__ContactParams, 42
InputProcessor, 42
operator=, 42
jeod::ContactSurface, 43
∼ContactSurface, 44
allocate_array, 44
allocate_interaction_facet, 45
collect_forces_torques, 45
contact_facets, 46
contact_force, 46
contact_torque, 46
ContactSurface, 44
facets_size, 47
init_attrjeod__ContactSurface, 46
InputProcessor, 46
operator=, 45
jeod::ContactSurfaceFactory, 47
∼ContactSurfaceFactory, 48
add_facet_params, 49
ContactSurfaceFactory, 48, 49
create_surface, 49
init_attrjeod__ContactSurfaceFactory, 50
InputProcessor, 50
line_contact_facet_factory, 50
operator=, 50
point_contact_facet_factory, 50
jeod::ContactUtils, 51
create_relstate_name, 51
dist_line_segments, 52
jeod::LineContactFacet, 53
∼LineContactFacet, 54
calculate_contact_point, 54
calculate_torque, 55
create_pair, 55
init_attrjeod__LineContactFacet, 57
InputProcessor, 57
length, 57
LineContactFacet, 54
operator=, 56
set_max_dimension, 56
jeod::LineContactFacetFactory, 57
∼LineContactFacetFactory, 58
create_facet, 59

init_attrjeod__LineContactFacetFactory, 60
InputProcessor, 60
is_correct_factory, 59
LineContactFacetFactory, 58
operator=, 60
jeod::LineContactPair, 60
∼LineContactPair, 61
in_contact, 62
init_attrjeod__LineContactPair, 63
initialize_pair, 62
InputProcessor, 63
line_subject, 63
line_target, 63
LineContactPair, 61
operator=, 62
jeod::LinePointContactPair, 64
∼LinePointContactPair, 65
in_contact, 65
init_attrjeod__LinePointContactPair, 66
initialize_pair, 65
InputProcessor, 66
line_subject, 66
LinePointContactPair, 64, 65
operator=, 66
point_target, 67
jeod::PairInteraction, 67
∼PairInteraction, 68
calculate_forces, 69
friction_mag, 70
init_attrjeod__PairInteraction, 70
InputProcessor, 70
is_correct_interaction, 69
operator=, 70
PairInteraction, 68
params_1, 70
params_2, 71
jeod::PointContactFacet, 71
∼PointContactFacet, 73
calculate_contact_point, 73
calculate_torque, 73
contact_point, 76
create_pair, 74
init_attrjeod__PointContactFacet, 75
InputProcessor, 75
operator=, 75
PointContactFacet, 72, 73
radius, 76
set_max_dimension, 75
jeod::PointContactFacetFactory, 77
∼PointContactFacetFactory, 77
create_facet, 78
init_attrjeod__PointContactFacetFactory, 79
InputProcessor, 79
is_correct_factory, 78
operator=, 79
PointContactFacetFactory, 77, 78
jeod::PointContactPair, 79
∼PointContactPair, 80
in_contact, 81
init_attrjeod__PointContactPair, 82
initialize_pair, 81
InputProcessor, 82
operator=, 81
point_subject, 82
point_target, 82
PointContactPair, 80
jeod::SpringPairInteraction, 83
∼SpringPairInteraction, 84
calculate_forces, 84
damping_b, 85
init_attrjeod__SpringPairInteraction, 85
InputProcessor, 85
mu, 85
operator=, 84
spring_k, 85
SpringPairInteraction, 83, 84

length
jeod::LineContactFacet, 57
line_contact_facet.cc, 96
line_contact_facet.hh, 96
line_contact_facet_factory
jeod::ContactSurfaceFactory, 50
line_contact_facet_factory.cc, 97
line_contact_facet_factory.hh, 97
line_contact_pair.cc, 98
line_contact_pair.hh, 98
line_point_contact_pair.cc, 99
line_point_contact_pair.hh, 99
line_subject
jeod::LineContactPair, 63
jeod::LinePointContactPair, 66
line_target
jeod::LineContactPair, 63
LineContactFacet
jeod::LineContactFacet, 54
LineContactFacetFactory
jeod::LineContactFacetFactory, 58
LineContactPair
jeod::LineContactPair, 61
LinePointContactPair
jeod::LinePointContactPair, 64, 65

MAKE_CONTACT_MESSAGE_CODE
contact_messages.cc, 90
max_dimension
jeod::ContactFacet, 29
Models, 11
mu
jeod::SpringPairInteraction, 85

normal
jeod::ContactFacet, 29

operator=
jeod::Contact, 20
jeod::ContactFacet, 28

jeod::ContactMessages, 32
jeod::ContactPair, 39
jeod::ContactParams, 42
jeod::ContactSurface, 45
jeod::ContactSurfaceFactory, 50
jeod::LineContactFacet, 56
jeod::LineContactFacetFactory, 60
jeod::LineContactPair, 62
jeod::LinePointContactPair, 66
jeod::PairInteraction, 70
jeod::PointContactFacet, 75
jeod::PointContactFacetFactory, 79
jeod::PointContactPair, 81
jeod::SpringPairInteraction, 84

pair_interaction.cc, 100
pair_interaction.hh, 100
pair_interactions
jeod::Contact, 24
PairInteraction
jeod::PairInteraction, 68
params_1
jeod::PairInteraction, 70
params_2
jeod::PairInteraction, 71
point_contact_facet.cc, 101
point_contact_facet.hh, 102
point_contact_facet_factory
jeod::ContactSurfaceFactory, 50
point_contact_facet_factory.cc, 102
point_contact_facet_factory.hh, 103
point_contact_pair.cc, 103
point_contact_pair.hh, 104
point_subject
jeod::PointContactPair, 82
point_target
jeod::LinePointContactPair, 67
jeod::PointContactPair, 82
PointContactFacet
jeod::PointContactFacet, 72, 73
PointContactFacetFactory
jeod::PointContactFacetFactory, 77, 78
PointContactPair
jeod::PointContactPair, 80
position
jeod::ContactFacet, 30
pre_initialization_error
jeod::ContactMessages, 33

radius
jeod::PointContactFacet, 76
register_contact
jeod::Contact, 20, 21
register_interaction
jeod::Contact, 22
rel_state
jeod::ContactPair, 40
runtime_error
jeod::ContactMessages, 33
runtime_inform
jeod::ContactMessages, 34
runtime_warns
jeod::ContactMessages, 34

set_max_dimension
jeod::ContactFacet, 28
jeod::LineContactFacet, 56
jeod::PointContactFacet, 75
spring_k
jeod::SpringPairInteraction, 85
spring_pair_interaction.cc, 104
spring_pair_interaction.hh, 105
SpringPairInteraction
jeod::SpringPairInteraction, 83, 84
subject
jeod::ContactPair, 40
surface_type
jeod::ContactFacet, 30

target
jeod::ContactPair, 40

unique_pair
jeod::Contact, 22

vehicle_body
jeod::ContactFacet, 30
vehicle_point
jeod::ContactFacet, 30