DynamicBodyModel

5.3

Generated by Doxygen 1.8.14

Contents

1	Mod	odule Index 1						
	1.1	Modules	1					
2	Nam	nespace Index	3					
	2.1	Namespace List	3					
3	Hier	rarchical Index	5					
	3.1	Class Hierarchy	5					
4	Data	a Structure Index	7					
	4.1	Data Structures	7					
5	File	Index	9					
	5.1	File List	9					
6	Mod	lule Documentation	11					
	6.1	Models	11					
		6.1.1 Detailed Description	11					
	6.2	Dynamics	12					
		6.2.1 Detailed Description	12					
	6.3	DynBody	13					
		6.3.1 Detailed Description	14					

ii CONTENTS

7	Nam	espace	Docume	ntation	15
	7.1	jeod N	amespace	Reference	15
		7.1.1	Detailed	Description	16
		7.1.2	Function	Documentation	16
			7.1.2.1	accumulate_forces() [1/2]	16
			7.1.2.2	accumulate_forces() [2/2]	17
			7.1.2.3	accumulate_torques() [1/2]	17
			7.1.2.4	accumulate_torques() [2/2]	17
			7.1.2.5	check_frame_ownership()	18
			7.1.2.6	collect_insert()	18
			7.1.2.7	collect_push_back()	19
			7.1.2.8	release_vector()	19
8	Data	Church	una Dagum	mentation	21
•					
	8.1			Collect Class Reference	21
		8.1.1		Description	22
		8.1.2	Construc	ctor & Destructor Documentation	22
			8.1.2.1	BodyForceCollect() [1/2]	22
			8.1.2.2	~BodyForceCollect()	23
			8.1.2.3	BodyForceCollect() [2/2]	23
		8.1.3	Member	Function Documentation	23
			8.1.3.1	operator=()	23
		8.1.4	Field Do	cumentation	23
			8.1.4.1	collect_effector_forc	23
			8.1.4.2	collect_effector_torq	24
			8.1.4.3	collect_environ_forc	24
			8.1.4.4	collect_environ_torq	24
			8.1.4.5	collect_no_xmit_forc	24
			8.1.4.6	collect_no_xmit_torq	25
			8.1.4.7	effector_forc	25
			8.1.4.8	effector_torq	25

CONTENTS

		8.1.4.9	environ_forc	26
		8.1.4.10	environ_torq	26
		8.1.4.11	extern_forc_inrtl	26
		8.1.4.12	extern_forc_struct	27
		8.1.4.13	extern_torq_body	27
		8.1.4.14	extern_torq_struct	27
		8.1.4.15	inertial_torq	28
		8.1.4.16	no_xmit_forc	28
		8.1.4.17	no_xmit_torq	28
8.2	jeod::B	BodyRefFra	ame Class Reference	29
	8.2.1	Detailed	Description	29
	8.2.2	Construc	tor & Destructor Documentation	29
		8.2.2.1	BodyRefFrame() [1/2]	30
		8.2.2.2	~BodyRefFrame()	30
		8.2.2.3	BodyRefFrame() [2/2]	30
	8.2.3	Member	Function Documentation	30
		8.2.3.1	operator=()	30
	8.2.4	Friends A	And Related Function Documentation	30
		8.2.4.1	init_attrjeodBodyRefFrame	30
		8.2.4.2	InputProcessor	30
	8.2.5	Field Doo	cumentation	31
		8.2.5.1	initialized_items	31
		8.2.5.2	mass_point	31
8.3	jeod::B	BodyWrenc	hCollect Class Reference	31
	8.3.1	Detailed	Description	32
	8.3.2	Construc	tor & Destructor Documentation	32
		8.3.2.1	BodyWrenchCollect() [1/2]	32
		8.3.2.2	~BodyWrenchCollect()	33
		8.3.2.3	BodyWrenchCollect() [2/2]	33
	8.3.3	Member	Function Documentation	33

iv CONTENTS

		8.3.3.1	accumulate() [1/2]	 . 33
		8.3.3.2	accumulate() [2/2]	 . 33
		8.3.3.3	operator=()	 . 34
	8.3.4	Field Do	ocumentation	 . 34
		8.3.4.1	collect_wrench	 . 34
8.4	jeod::C	CInterfaceF	Force Class Reference	 . 35
	8.4.1	Detailed	Description	 . 35
	8.4.2	Construc	ctor & Destructor Documentation	 . 35
		8.4.2.1	CInterfaceForce() [1/3]	 . 35
		8.4.2.2	CInterfaceForce() [2/3]	 . 35
		8.4.2.3	~CInterfaceForce()	 . 36
		8.4.2.4	CInterfaceForce() [3/3]	 . 36
	8.4.3	Member	Function Documentation	 . 36
		8.4.3.1	operator=()	 . 36
8.5	jeod::C	CInterfaceT	Torque Class Reference	 . 37
	8.5.1	Detailed	I Description	 . 37
	8.5.2	Construc	ctor & Destructor Documentation	 . 37
		8.5.2.1	CInterfaceTorque() [1/3]	 . 37
		8.5.2.2	CInterfaceTorque() [2/3]	 . 37
		8.5.2.3	~CInterfaceTorque()	 . 38
		8.5.2.4	CInterfaceTorque() [3/3]	 . 38
	8.5.3	Member	Function Documentation	 . 38
		8.5.3.1	operator=()	 . 38
8.6	jeod::C	CollectForc	ce Class Reference	 . 39
	8.6.1	Detailed	I Description	 . 40
	8.6.2	Construc	ctor & Destructor Documentation	 . 40
		8.6.2.1	CollectForce() [1/5]	 . 40
		8.6.2.2	CollectForce() [2/5]	 . 40
		8.6.2.3	CollectForce() [3/5]	 . 41
		8.6.2.4	CollectForce() [4/5]	 . 41

CONTENTS

		8.6.2.5	~CollectForce()	41
		8.6.2.6	CollectForce() [5/5]	41
	8.6.3	Member	Function Documentation	42
		8.6.3.1	create() [1/5]	42
		8.6.3.2	create() [2/5]	42
		8.6.3.3	create() [3/5]	43
		8.6.3.4	create() [4/5]	43
		8.6.3.5	create() [5/5]	43
		8.6.3.6	is_active()	44
		8.6.3.7	operator=()	44
		8.6.3.8	operator==()	44
		8.6.3.9	operator[]() [1/2]	44
		8.6.3.10	operator[]() [2/2]	45
	8.6.4	Field Doo	cumentation	45
		8.6.4.1	active	45
		8.6.4.2	force	46
8.7	jeod::C	CollectTorq	ue Class Reference	46
	8.7.1	Detailed	Description	47
	8.7.2	Construc	etor & Destructor Documentation	47
		8.7.2.1	CollectTorque() [1/5]	47
		8.7.2.2	CollectTorque() [2/5]	47
		8.7.2.3	CollectTorque() [3/5]	48
		8.7.2.4	CollectTorque() [4/5]	48
		8.7.2.5	~CollectTorque()	48
		8.7.2.6	CollectTorque() [5/5]	49
	8.7.3	Member	Function Documentation	49
		8.7.3.1	create() [1/5]	49
		8.7.3.2	create() [2/5]	49
		8.7.3.3	create() [3/5]	50
		8.7.3.4	create() [4/5]	50

vi

		8.7.3.5	create() [5/5]	51
		8.7.3.6	is_active()	51
		8.7.3.7	operator=()	51
		8.7.3.8	operator==()	52
		8.7.3.9	operator[]() [1/2]	52
		8.7.3.10	operator[]() [2/2]	52
	8.7.4	Field Doo	cumentation	53
		8.7.4.1	active	53
		8.7.4.2	torque	53
8.8	jeod::D	ynBody C	lass Reference	53
	8.8.1	Detailed	Description	59
	8.8.2	Construc	etor & Destructor Documentation	59
		8.8.2.1	DynBody() [1/2]	59
		8.8.2.2	~DynBody()	60
		8.8.2.3	DynBody() [2/2]	60
	8.8.3	Member	Function Documentation	60
		8.8.3.1	activate()	60
		8.8.3.2	add_control()	60
		8.8.3.3	add_integrable_object()	61
		8.8.3.4	add_mass_body() [1/2]	61
		8.8.3.5	add_mass_body() [2/2]	61
		8.8.3.6	add_mass_body_frames()	61
		8.8.3.7	add_mass_body_validate()	62
		8.8.3.8	add_mass_point()	62
		8.8.3.9	attach_child() [1/2]	63
		8.8.3.10	attach_child() [2/2]	63
		8.8.3.11	attach_establish_links()	64
		8.8.3.12	attach_to() [1/2]	64
		8.8.3.13	attach_to() [2/2]	65
		8.8.3.14	attach_to_frame() [1/4]	65

CONTENTS vii

8.8.3.15	attach_to_frame() [2/4]	66
8.8.3.16	attach_to_frame() [3/4]	66
8.8.3.17	attach_to_frame() [4/4]	66
8.8.3.18	attach_update_properties()	67
8.8.3.19	attach_validate_child()	67
8.8.3.20	attach_validate_parent()	68
8.8.3.21	clear_integrable_objects()	69
8.8.3.22	collect_forces_and_torques()	69
8.8.3.23	compute_derived_state_forward()	69
8.8.3.24	compute_derived_state_reverse()	70
8.8.3.25	compute_ref_point_transform()	70
8.8.3.26	compute_state_elements_forward()	71
8.8.3.27	compute_state_elements_reverse()	71
8.8.3.28	compute_vehicle_point_derivatives()	72
8.8.3.29	compute_vehicle_point_states()	72
8.8.3.30	create_body_integrators()	73
8.8.3.31	create_integrators()	73
8.8.3.32	deactivate()	74
8.8.3.33	destroy_integrators()	74
8.8.3.34	detach() [1/2]	74
8.8.3.35	detach() [2/2]	75
8.8.3.36	detach_mass_body_frames()	75
8.8.3.37	detach_mass_internal()	76
8.8.3.38	find_body_frame()	76
8.8.3.39	find_vehicle_point()	77
8.8.3.40	get_dynamics_integration_group()	78
8.8.3.41	get_initialized_states()	78
8.8.3.42	get_integ_frame()	78
8.8.3.43	get_integrable_objects()	79
8.8.3.44	get_parent_body()	79

viii CONTENTS

8.8.3.45	get_parent_body_internal()	79
8.8.3.46	get_root_body()	80
8.8.3.47	get_root_body_internal()	80
8.8.3.48	initialize_controls()	80
8.8.3.49	initialize_model()	81
8.8.3.50	initialized_states_contains()	81
8.8.3.51	integrate()	82
8.8.3.52	is_root_body()	82
8.8.3.53	migrate_integrable_objects()	82
8.8.3.54	operator=()	83
8.8.3.55	process_dynamic_attachment()	83
8.8.3.56	propagate_state()	84
8.8.3.57	propagate_state_from_composite()	84
8.8.3.58	propagate_state_from_structure()	84
8.8.3.59	remove_integrable_object()	84
8.8.3.60	remove_mass_body()	85
8.8.3.61	reset_controls()	86
8.8.3.62	reset_integrators()	86
8.8.3.63	rot_integ()	86
8.8.3.64	set_attitude_left_quaternion()	87
8.8.3.65	set_attitude_matrix()	87
8.8.3.66	set_attitude_rate()	88
8.8.3.67	set_attitude_right_quaternion()	88
8.8.3.68	set_integ_frame() [1/2]	89
8.8.3.69	set_integ_frame() [2/2]	89
8.8.3.70	set_name()	90
8.8.3.71	set_position()	90
8.8.3.72	set_state()	91
8.8.3.73	set_state_source()	91
8.8.3.74	set_state_source_internal()	92

CONTENTS

	8.8.3.75	set_velocity()	92
	8.8.3.76	sort_controls()	93
	8.8.3.77	switch_integration_frames() [1/2]	93
	8.8.3.78	<pre>switch_integration_frames() [2/2]</pre>	94
	8.8.3.79	trans_integ()	94
	8.8.3.80	update_integrated_state()	95
8.8.4	Friends A	And Related Function Documentation	95
	8.8.4.1	init_attrjeodDynBody	95
	8.8.4.2	InputProcessor	95
8.8.5	Field Doo	cumentation	95
	8.8.5.1	associated_integrable_objects	96
	8.8.5.2	attitude_source	96
	8.8.5.3	autoupdate_vehicle_points	96
	8.8.5.4	collect	97
	8.8.5.5	composite_body	97
	8.8.5.6	compute_point_derivative	97
	8.8.5.7	core_body	98
	8.8.5.8	derivs	98
	8.8.5.9	dyn_children	98
	8.8.5.10	dyn_manager	99
	8.8.5.11	dyn_parent	99
	8.8.5.12	frame_attach	99
	8.8.5.13	grav_interaction	100
	8.8.5.14	initialized_states	100
	8.8.5.15	integ_frame	100
	8.8.5.16	integ_frame_name	101
	8.8.5.17	integ_results_merger	101
	8.8.5.18	integrated_frame	101
	8.8.5.19	mass	102
	8.8.5.20	mass_children	102

CONTENTS

		8.8.5.21	name	102
		8.8.5.22	position_source	103
		8.8.5.23	rate_source	103
		8.8.5.24	rot_integrator	103
		8.8.5.25	rotation_integration	104
		8.8.5.26	rotational_dynamics	104
		8.8.5.27	structure	104
		8.8.5.28	three_dof	105
		8.8.5.29	time_manager	105
		8.8.5.30	trans_integrator	105
		8.8.5.31	translational_dynamics	106
		8.8.5.32	vehicle_points	106
		8.8.5.33	velocity_source	106
8.9	jeod::D	ynBodyGe	enericFrameAttachment Class Reference	107
	8.9.1	Detailed	Description	107
	8.9.2	Construc	tor & Destructor Documentation	107
		8.9.2.1	DynBodyGenericFrameAttachment()	108
	8.9.3	Member	Function Documentation	108
		8.9.3.1	clear_attachment()	108
		8.9.3.2	get_attach_offset()	108
		8.9.3.3	get_parent_frame()	108
		8.9.3.4	initialize_attachment()	108
		8.9.3.5	isAttached()	109
	8.9.4	Friends A	And Related Function Documentation	109
		8.9.4.1	init_attrjeodDynBodyGenericFrameAttachment	109
		8.9.4.2	InputProcessor	109
	8.9.5	Field Doo	cumentation	109
		8.9.5.1	active	109
		8.9.5.2	rigid_attach_parent	109
		8.9.5.3	rigid_attach_state	110

CONTENTS xi

8.10	jeod::D	nBodyMessages Class Reference
	8.10.1	Detailed Description
	8.10.2	Constructor & Destructor Documentation
		8.10.2.1 DynBodyMessages() [1/2]
		8.10.2.2 DynBodyMessages() [2/2] 11
	8.10.3	Member Function Documentation
		8.10.3.1 operator=()
	8.10.4	Friends And Related Function Documentation
		8.10.4.1 init_attrjeodDynBodyMessages
		8.10.4.2 InputProcessor
	8.10.5	Field Documentation
		8.10.5.1 internal_error
		8.10.5.2 invalid_attachment
		8.10.5.3 invalid_body
		8.10.5.4 invalid_frame
		8.10.5.5 invalid_group
		8.10.5.6 invalid_name
		8.10.5.7 invalid_technique
		8.10.5.8 not_dyn_body
8.11	jeod::Fo	rce Class Reference
	8.11.1	Detailed Description
	8.11.2	Constructor & Destructor Documentation
		8.11.2.1 Force() [1/2]
		8.11.2.2 ~Force()
		8.11.2.3 Force() [2/2]
	8.11.3	Member Function Documentation
		8.11.3.1 operator=()
		8.11.3.2 operator[]() [1/2]
		8.11.3.3 operator[]() [2/2]
	8.11.4	Field Documentation

xii CONTENTS

		8.11.4.1 active	16
		8.11.4.2 force	17
8.12 j	jeod::Fr	rameDerivs Class Reference	17
8	8.12.1	Detailed Description	17
8	8.12.2	Constructor & Destructor Documentation	17
		8.12.2.1 FrameDerivs()	18
8	8.12.3	Field Documentation	18
		8.12.3.1 non_grav_accel	18
		8.12.3.2 Qdot_parent_this	18
		8.12.3.3 rot_accel	18
		8.12.3.4 trans_accel	19
8.13 j	jeod::JF	PVCollectForce Class Reference	19
8	8.13.1	Detailed Description	20
8	8.13.2	Member Function Documentation	20
		8.13.2.1 perform_insert_action()	20
		8.13.2.2 push_back()	20
8	8.13.3	Friends And Related Function Documentation	20
		8.13.3.1 collect_insert	21
		8.13.3.2 collect_push_back	21
8.14 j	jeod::JF	PVCollectTorque Class Reference	21
8	8.14.1	Detailed Description	22
8	8.14.2	Member Function Documentation	22
		8.14.2.1 perform_insert_action()	22
		8.14.2.2 push_back()	22
8	8.14.3	Friends And Related Function Documentation	23
		8.14.3.1 collect_insert	23
		8.14.3.2 collect_push_back	23
8.15 j	jeod::S	tructureIntegratedDynBody Class Reference	23
8	8.15.1	Detailed Description	25
8	8.15.2	Constructor & Destructor Documentation	25

CONTENTS xiii

	8.15.2.1	StructureIntegratedDynBody() [1/2]	. 126
	8.15.2.2	~StructureIntegratedDynBody()	. 126
	8.15.2.3	StructureIntegratedDynBody() [2/2]	. 126
8.15.3	Member F	Function Documentation	. 126
	8.15.3.1	add_constraint()	. 126
	8.15.3.2	attach_update_properties()	. 127
	8.15.3.3	collect_forces_and_torques()	. 127
	8.15.3.4	collect_local_forces_and_torques()	. 128
	8.15.3.5	complete_translational_acceleration()	. 128
	8.15.3.6	compute_inertial_torque()	. 128
	8.15.3.7	compute_rotational_acceleration()	. 129
	8.15.3.8	compute_translational_acceleration()	. 129
	8.15.3.9	compute_vehicle_point_derivatives()	. 129
	8.15.3.10	detach()	. 130
	8.15.3.11	get_vehicle_properties()	. 130
	8.15.3.12	operator=()	. 130
	8.15.3.13	PropagateForcesAndTorques()	. 130
	8.15.3.14	rot_integ()	. 131
	8.15.3.15	set_solver()	. 132
	8.15.3.16	solve_constraints()	. 132
	8.15.3.17	trans_integ()	. 132
8.15.4	Friends A	nd Related Function Documentation	. 133
	8.15.4.1	DynBodyConstraintsSolver	. 133
	8.15.4.2	init_attrjeodStructureIntegratedDynBody	. 133
	8.15.4.3	InputProcessor	. 133
8.15.5	Field Doc	umentation	. 133
	8.15.5.1	constraints_solver	. 134
	8.15.5.2	effector_wrench	. 134
	8.15.5.3	effector_wrench_collection	. 134
	8.15.5.4	inertial_accel_inrtl	. 135

xiv CONTENTS

		8.15.5.5 inertial_accel_struct	35
		8.15.5.6 inertial_accel_struct_omega	35
		8.15.5.7 inertial_accel_struct_omega_dot	35
		8.15.5.8 non_grav_state	36
		8.15.5.9 struct_derivs	36
		8.15.5.10 vehicle_properties	36
8.16	jeod::To	rque Class Reference	37
	8.16.1	Detailed Description	37
	8.16.2	Constructor & Destructor Documentation	37
		8.16.2.1 Torque() [1/2]	37
		8.16.2.2 ~Torque()	38
		8.16.2.3 Torque() [2/2]	38
	8.16.3	Member Function Documentation	38
		8.16.3.1 operator=()	38
		8.16.3.2 operator[]() [1/2]	38
		8.16.3.3 operator[]() [2/2]	38
	8.16.4	Field Documentation	39
		8.16.4.1 active	39
		8.16.4.2 torque	39
8.17	jeod::Ve	hicleNonGravState Class Reference	40
	8.17.1	Detailed Description	40
	8.17.2	Friends And Related Function Documentation	40
		8.17.2.1 init_attrjeodVehicleNonGravState	40
		8.17.2.2 InputProcessor	40
	8.17.3	Field Documentation	41
		8.17.3.1 accel_struct	41
		8.17.3.2 inertial_torque_struct	41
		8.17.3.3 omega_body	41
		8.17.3.4 omega_dot_body	42
		8.17.3.5 omega_dot_struct	42

CONTENTS xv

		8.17.3.6 omega_struct	42
8.18	jeod::Ve	ehicleProperties Class Reference	42
	8.18.1	Detailed Description	43
	8.18.2	Constructor & Destructor Documentation	44
		8.18.2.1 VehicleProperties() [1/2]	44
		8.18.2.2 VehicleProperties() [2/2]	44
	8.18.3	Member Function Documentation	44
		8.18.3.1 get_inertia()	44
		8.18.3.2 get_inverse_inertia()	45
		8.18.3.3 get_inverse_mass()	45
		8.18.3.4 get_mass()	45
		8.18.3.5 get_parent_to_structure_offset()	46
		8.18.3.6 get_parent_to_structure_transform()	46
		8.18.3.7 get_structure_to_body_offset()	46
		8.18.3.8 get_structure_to_body_transform()	46
	8.18.4	Friends And Related Function Documentation	47
		8.18.4.1 init_attrjeodVehicleProperties	47
		8.18.4.2 InputProcessor	47
	8.18.5	Field Documentation	47
		8.18.5.1 inertia	47
		8.18.5.2 inverse_inertia	47
		8.18.5.3 inverse_mass	48
		8.18.5.4 mass	48
		8.18.5.5 parent_to_structure_offset	48
		8.18.5.6 parent_to_structure_transform	48
		8.18.5.7 structure_to_body_offset	49
		8.18.5.8 structure_to_body_transform	49
8.19	jeod::W	/rench Class Reference	49
	8.19.1	Detailed Description	51
	8.19.2	Constructor & Destructor Documentation	51

xvi CONTENTS

	8.19.2.1 Wrench() [1/5]	151
	8.19.2.2 Wrench() [2/5]	151
	8.19.2.3 Wrench() [3/5]	152
	8.19.2.4 ~Wrench()	152
	8.19.2.5 Wrench() [4/5]	152
	8.19.2.6 Wrench() [5/5]	153
8.19.3	Member Function Documentation	153
	8.19.3.1 accumulate() [1/2]	153
	8.19.3.2 accumulate() [2/2]	153
	8.19.3.3 activate()	153
	8.19.3.4 deactivate()	154
	8.19.3.5 get_force()	154
	8.19.3.6 get_point()	154
	8.19.3.7 get_torque()	154
	8.19.3.8 is_active()	154
	8.19.3.9 operator+=()	154
	8.19.3.10 operator=() [1/2]	155
	8.19.3.11 operator=() [2/2]	155
	8.19.3.12 reset_force()	155
	8.19.3.13 reset_force_and_torque()	155
	8.19.3.14 reset_point()	156
	8.19.3.15 reset_torque()	156
	8.19.3.16 scale_force()	156
	8.19.3.17 scale_torque()	156
	8.19.3.18 set()	156
	8.19.3.19 set_force() [1/2]	157
	8.19.3.20 set_force() [2/2]	157
	8.19.3.21 set_point()	157
	8.19.3.22 set_torque()	158
	8.19.3.23 transform_to_parent()	158
	8.19.3.24 transform_to_point()	158
8.19.4	Friends And Related Function Documentation	159
	8.19.4.1 init_attrjeodWrench	159
	8.19.4.2 InputProcessor	159
8.19.5	Field Documentation	159
	8.19.5.1 active	159
	8.19.5.2 force	159
	8.19.5.3 point	160
	8.19.5.4 torque	160

CONTENTS xvii

9	File	Documentation	161
	9.1	body_force_collect.cc File Reference	161
		9.1.1 Detailed Description	161
	9.2	body_force_collect.hh File Reference	161
		9.2.1 Detailed Description	162
	9.3	body_ref_frame.hh File Reference	162
		9.3.1 Detailed Description	162
	9.4	body_wrench_collect.cc File Reference	163
		9.4.1 Detailed Description	163
	9.5	body_wrench_collect.hh File Reference	163
		9.5.1 Detailed Description	163
	9.6	class_declarations.hh File Reference	163
		9.6.1 Detailed Description	164
	9.7	dyn_body.cc File Reference	164
		9.7.1 Detailed Description	164
	9.8	dyn_body.hh File Reference	164
		9.8.1 Detailed Description	165
	9.9	dyn_body_attach.cc File Reference	165
		9.9.1 Detailed Description	165
	9.10	dyn_body_collect.cc File Reference	166
		9.10.1 Detailed Description	166
	9.11	dyn_body_detach.cc File Reference	166
		9.11.1 Detailed Description	167
	9.12	dyn_body_find_body_frame.cc File Reference	167
		9.12.1 Detailed Description	167
	9.13	dyn_body_generic_rigid_attach.hh File Reference	167
		9.13.1 Detailed Description	168
	9.14	dyn_body_initialize_model.cc File Reference	168
		9.14.1 Detailed Description	168
	9.15	dyn_body_integration.cc File Reference	168

xviii CONTENTS

	9.15.1 Detailed Description	169
9.16	dyn_body_messages.cc File Reference	169
	9.16.1 Detailed Description	169
	9.16.2 Macro Definition Documentation	169
	9.16.2.1 MAKE_DYNBODY_MESSAGE_CODE	169
9.17	dyn_body_messages.hh File Reference	170
	9.17.1 Detailed Description	170
9.18	dyn_body_propagate_state.cc File Reference	170
	9.18.1 Detailed Description	170
9.19	dyn_body_set_state.cc File Reference	171
	9.19.1 Detailed Description	171
9.20	dyn_body_vehicle_point.cc File Reference	171
	9.20.1 Detailed Description	172
9.21	force.cc File Reference	172
	9.21.1 Detailed Description	172
9.22	force.hh File Reference	172
	9.22.1 Detailed Description	173
9.23	force_inline.hh File Reference	173
	9.23.1 Detailed Description	173
9.24	frame_derivs.hh File Reference	173
	9.24.1 Detailed Description	173
9.25	structure_integrated_dyn_body.cc File Reference	174
	9.25.1 Detailed Description	174
9.26	structure_integrated_dyn_body.hh File Reference	174
	9.26.1 Detailed Description	174
9.27	structure_integrated_dyn_body_collect.cc File Reference	175
	9.27.1 Detailed Description	175
9.28	structure_integrated_dyn_body_integration.cc File Reference	175
	9.28.1 Detailed Description	176
9.29	structure_integrated_dyn_body_pt_accel.cc File Reference	176

CONTENTS xix

Index		181
	9.36.1 Detailed Description	179
9.36	wrench.hh File Reference	179
	9.35.1 Detailed Description	179
9.35	vehicle_properties.hh File Reference	178
	9.34.1 Detailed Description	178
9.34	vehicle_non_grav_state.hh File Reference	178
	9.33.1 Detailed Description	178
9.33	torque_inline.hh File Reference	178
	9.32.1 Detailed Description	177
9.32	torque.hh File Reference	177
	9.31.1 Detailed Description	177
9.31	torque.cc File Reference	177
	9.30.1 Detailed Description	176
9.30	structure_integrated_dyn_body_solve.cc File Reference	176
	9.29.1 Detailed Description	176

Chapter 1

Module Index

1.1 Modules

Here is a list of all modules:

Models																				•	11
Dynamics .																					12
DynBody						 											 			 -	13

2 Module Index

Chapter 2

Namespace Index

2.1	Namespace	List

riere is a list of all flamespaces with brief t	descriptions.	
jeod		

4 Namespace Index

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

jeod::BodyForceCollect	21
jeod::BodyWrenchCollect	31
jeod::CollectForce	39
jeod::CInterfaceForce	35
jeod::CollectTorque	46
jeod::CInterfaceTorque	37
jeod::DynBodyGenericFrameAttachment	107
jeod::DynBodyMessages	110
jeod::Force	114
jeod::FrameDerivs	117
IntegrableObject	
jeod::DynBody	53
jeod::StructureIntegratedDynBody	123
RefFrame	
jeod::BodyRefFrame	29
RefFrameOwner	
jeod::DynBody	53
jeod::Torque	137
type	
jeod::JPVCollectForce	119
jeod::JPVCollectTorque	121
jeod::VehicleNonGravState	140
jeod::VehicleProperties	142
ieod:·Wrench	149

6 Hierarchical Index

Chapter 4

Data Structure Index

4.1 Data Structures

Here are the data structures with brief descriptions:

jeod::BodyForceCollect	
Serves as the collection point for forces and torques that act on a vehicle	21
jeod::BodyRefFrame	
Extend RefFrame to add coupling between the reference frame tree and the mass tree and to	
keep track of which state items have been set	29
jeod::BodyWrenchCollect	
Serves as the collection point for wrenches that act on a vehicle	31
jeod::CInterfaceForce	
This class is deprecated	35
jeod::CInterfaceTorque	
This class is deprecated	37
jeod::CollectForce	
A CollectForce represents a collected force that acts on a vehicle	39
jeod::CollectTorque	
A CollectTorque represents a collected torque that acts on a vehicle	46
jeod::DynBody	
Class DynBody is the base class for all dynamic bodies	53
jeod::DynBodyGenericFrameAttachment	
A wrench comprises a torque and a force applied at a point on a DynBody	107
jeod::DynBodyMessages	
Specify the message IDs used in the DynBody model	110
jeod::Force	
A Force represents a Newtonian force that acts on a DynBody	114
jeod::FrameDerivs	
Contains translational and rotational second derivatives	117
jeod::JPVCollectForce	
This is a derived version of the template class JeodPointerVector <collectforce>::type with an</collectforce>	
implementation of the method perform_cleanup_action which frees and clears stale data follow-	
ing a restore	119
jeod::JPVCollectTorque	
This is a derived version of the template class JeodPointerVector <collecttorque>::type with an</collecttorque>	
implementation of the method perform_cleanup_action which frees and clears stale data follow-	
ing a restore	121
jeod::StructureIntegratedDynBody	
Extends DynBody to integrate an object's structural reference frame as opposed to its center of	
mass	123

8 Data Structure Index

jeod::Torque	
A Torque represents a Newtonian torque that acts on a DynBody	137
jeod::VehicleNonGravState	
Encapsulates various aspects of a vehicle's state with respect to inertial	140
jeod::VehicleProperties	
Captures pointers to various vehicle properties that are commonly used in the constraint concept	142
jeod::Wrench	
A wrench comprises a torque and a force applied at a point on a DynBody	149

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

body_force_collect.cc	
Define BodyForceCollect member functions	161
body_force_collect.hh	
Define the class BodyForceCollect	161
body_ref_frame.hh	
Define the class BodyRefFrame	162
body_wrench_collect.cc	
Define BodyWrenchCollect member functions	163
body_wrench_collect.hh	
Defines the class BodyWrenchCollect	163
class_declarations.hh	
Forward declarations of classes defined in dyn_body.hh	163
dyn_body.cc	
Define base methods for the DynBody class	164
dyn_body.hh	
Define the class DynBody	164
dyn_body_attach.cc	
Define DynBody attachment methods	165
dyn_body_collect.cc	
Define DynBody methods related to force and torque accumulation and propagation	166
dyn_body_detach.cc	
Define DynBody detachment methods	166
dyn_body_find_body_frame.cc	
Define DynBody::find_body_frame	167
dyn_body_generic_rigid_attach.hh	
Define the class Wrench	167
dyn_body_initialize_model.cc	
Define DynBody::initialize_model	168
dyn_body_integration.cc	
Define methods for frame switching	168
dyn_body_messages.cc	
Implement the class De4xxMessages	169
dyn_body_messages.hh	
Define the class DynBodyMessages	170
dyn_body_propagate_state.cc	
Define DynBody state propagation / update methods	170

10 File Index

dyn_body_set_state.cc	
Define methods related to setting aspects of a vehicle's state	171
dyn_body_vehicle_point.cc	
Define methods that support vehicle points	171
force.cc	
Define force model member functions	172
force.hh	
Define the JEOD force model	172
force_inline.hh	
Inline functions for the JEOD force model	173
frame_derivs.hh	
Define the FrameDerivs class	173
structure_integrated_dyn_body.cc	
Define base member functions for StructureIntegratedDynBody	174
structure_integrated_dyn_body.hh	
Define the class StructureIntegratedDynBody, which integrates a DynBody object's structural	
state	174
structure_integrated_dyn_body_collect.cc	
Define StructureIntegratedDynBody methods related to force and torque accumulation and prop-	
agation	175
structure_integrated_dyn_body_integration.cc	
Define StructureIntegratedDynBody member functions related to state integration	175
structure_integrated_dyn_body_pt_accel.cc	470
Define StructureIntegratedDynBody::compute_vehicle_point_derivatives	1/6
structure_integrated_dyn_body_solve.cc	
Define StructureIntegratedDynBody methods related to force and torque accumulation and prop-	470
agation	176
torque.cc	177
Define torque model member functions	177
torque.hh	177
Define the JEOD torque model	177
torque_inline.hh Define the JEOD torque model	170
vehicle_non_grav_state.hh	178
Define the class VehicleNonGravState	178
vehicle properties.hh	170
Define the class VehicleProperties	178
wrench.hh	170
Define the class Wrench	179

Chapter 6

Module Documentation

6.1 Models

Modules

- Dynamics
- 6.1.1 Detailed Description

12 Module Documentation

6.2 Dynamics

Modules

• DynBody

6.2.1 Detailed Description

6.3 DynBody 13

6.3 DynBody

Files

• file body_force_collect.hh

Define the class BodyForceCollect.

• file body_ref_frame.hh

Define the class BodyRefFrame.

· file body_wrench_collect.hh

Defines the class BodyWrenchCollect.

· file class_declarations.hh

Forward declarations of classes defined in dyn_body.hh.

• file dyn_body.hh

Define the class DynBody.

• file dyn_body_generic_rigid_attach.hh

Define the class Wrench.

file dyn_body_messages.hh

Define the class DynBodyMessages.

· file force.hh

Define the JEOD force model.

· file force inline.hh

Inline functions for the JEOD force model.

· file frame derivs.hh

Define the FrameDerivs class.

file structure_integrated_dyn_body.hh

Define the class StructureIntegratedDynBody, which integrates a DynBody object's structural state.

· file torque.hh

Define the JEOD torque model.

file torque_inline.hh

Define the JEOD torque model.

file vehicle_non_grav_state.hh

Define the class VehicleNonGravState.

file vehicle_properties.hh

Define the class VehicleProperties.

· file wrench.hh

Define the class Wrench.

• file body_force_collect.cc

Define BodyForceCollect member functions.

• file body_wrench_collect.cc

Define BodyWrenchCollect member functions.

• file dyn_body.cc

Define base methods for the DynBody class.

file dyn_body_attach.cc

Define DynBody attachment methods.

• file dyn_body_collect.cc

Define DynBody methods related to force and torque accumulation and propagation.

file dyn_body_detach.cc

Define DynBody detachment methods.

file dyn_body_find_body_frame.cc

Define DynBody::find_body_frame.

14 Module Documentation

• file dyn_body_initialize_model.cc

Define DynBody::initialize_model.

• file dyn_body_integration.cc

Define methods for frame switching.

• file dyn_body_messages.cc

Implement the class De4xxMessages.

• file dyn_body_propagate_state.cc

Define DynBody state propagation / update methods.

• file dyn_body_set_state.cc

Define methods related to setting aspects of a vehicle's state.

• file dyn_body_vehicle_point.cc

Define methods that support vehicle points.

· file force.cc

Define force model member functions.

• file structure_integrated_dyn_body.cc

Define base member functions for StructureIntegratedDynBody.

• file structure_integrated_dyn_body_collect.cc

Define StructureIntegratedDynBody methods related to force and torque accumulation and propagation.

• file structure_integrated_dyn_body_integration.cc

Define StructureIntegratedDynBody member functions related to state integration.

• file structure_integrated_dyn_body_pt_accel.cc

Define StructureIntegratedDynBody::compute_vehicle_point_derivatives.

• file structure_integrated_dyn_body_solve.cc

Define StructureIntegratedDynBody methods related to force and torque accumulation and propagation.

· file torque.cc

Define torque model member functions.

Namespaces

jeod

Namespace jeod.

6.3.1 Detailed Description

Chapter 7

Namespace Documentation

7.1 jeod Namespace Reference

Namespace jeod.

Data Structures

· class BodyForceCollect

Serves as the collection point for forces and torques that act on a vehicle.

· class BodyRefFrame

Extend RefFrame to add coupling between the reference frame tree and the mass tree and to keep track of which state items have been set.

· class BodyWrenchCollect

Serves as the collection point for wrenches that act on a vehicle.

· class CInterfaceForce

This class is deprecated.

class CInterfaceTorque

This class is deprecated.

class CollectForce

A CollectForce represents a collected force that acts on a vehicle.

• class CollectTorque

A CollectTorque represents a collected torque that acts on a vehicle.

class DynBody

Class DynBody is the base class for all dynamic bodies.

· class DynBodyGenericFrameAttachment

A wrench comprises a torque and a force applied at a point on a DynBody.

class DynBodyMessages

Specify the message IDs used in the DynBody model.

· class Force

A Force represents a Newtonian force that acts on a DynBody.

class FrameDerivs

Contains translational and rotational second derivatives.

class JPVCollectForce

This is a derived version of the template class JeodPointerVector< CollectForce>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

class JPVCollectTorque

This is a derived version of the template class JeodPointerVector<CollectTorque>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

· class StructureIntegratedDynBody

Extends DynBody to integrate an object's structural reference frame as opposed to its center of mass.

· class Torque

A Torque represents a Newtonian torque that acts on a DynBody.

· class VehicleNonGravState

Encapsulates various aspects of a vehicle's state with respect to inertial.

· class VehicleProperties

Captures pointers to various vehicle properties that are commonly used in the constraint concept.

· class Wrench

A wrench comprises a torque and a force applied at a point on a DynBody.

Functions

template < class CollectType >
 void release_vector (CollectType &vec)

Release JEOD-allocated memory in the collect vector.

- template<typename CollectType, typename value_type > void collect insert (CollectType &collect in, value type &elem)
- template<typename CollectType, typename value_type > void collect_push_back (CollectType &collect_in, value_type &elem)
- static void accumulate_forces (const JeodPointerVector< CollectForce >::type &vec, double *cumulation)

 Accumulate forces acting on a vehicle.
- static void accumulate_torques (const JeodPointerVector< CollectTorque >::type &vec, double *cumulation)

 Accumulate torques acting on a vehicle.
- static void check_frame_ownership (const BodyRefFrame &frame, const DynBody *dyn_body, const char *file, unsigned int line)

Check that the dyn_body 'owns' the subject frame.

- static void accumulate_forces (const JeodPointerVector< CollectForce >::type &vec, double *cumulation)

 Accumulate forces acting on a vehicle.
- static void accumulate_torques (const JeodPointerVector< CollectTorque >::type &vec, double *cumulation)

 Accumulate torques acting on a vehicle.

7.1.1 Detailed Description

Namespace jeod.

7.1.2 Function Documentation

7.1.2.1 accumulate_forces() [1/2]

Accumulate forces acting on a vehicle.

in	vec	Forces
out	cumulation	Accumulated force

Definition at line 38 of file structure_integrated_dyn_body_collect.cc.

7.1.2.2 accumulate_forces() [2/2]

Accumulate forces acting on a vehicle.

Parameters

in	vec	Forces
out	cumulation	Accumulated force

Definition at line 57 of file dyn_body_collect.cc.

 $Referenced\ by\ jeod::DynBody::collect_forces_and_torques(),\ and\ jeod::StructureIntegratedDynBody::collect_\hookleftarrow local_forces_and_torques().$

7.1.2.3 accumulate_torques() [1/2]

Accumulate torques acting on a vehicle.

Parameters

in	vec	Torques
out	cumulation	Accumulated torque

Definition at line 56 of file structure_integrated_dyn_body_collect.cc.

7.1.2.4 accumulate_torques() [2/2]

```
static void jeod::accumulate_torques (
```

```
const JeodPointerVector< CollectTorque >::type & vec,
double * cumulation ) [inline], [static]
```

Accumulate torques acting on a vehicle.

Parameters

in	vec	Torques
out	cumulation	Accumulated torque

Definition at line 77 of file dyn_body_collect.cc.

Referenced by jeod::DynBody::collect_forces_and_torques(), and jeod::StructureIntegratedDynBody::collect_ \leftarrow local_forces_and_torques().

7.1.2.5 check_frame_ownership()

Check that the dyn_body 'owns' the subject frame.

Parameters

in	frame	Frame to test
in	dyn_body	Typically this
in	file	Typically FILE
in	line	Typically LINE

Definition at line 61 of file dyn_body_set_state.cc.

 $References\ jeod:: DynBodyMessages:: invalid_frame,\ and\ jeod:: DynBody:: name.$

Referenced by jeod::DynBody::set_attitude_left_quaternion(), jeod::DynBody::set_attitude_matrix(), jeod::DynBody::set_attitude_right_quaternion(), jeod::DynBody::set_position(), jeod::

DynBody::set_state(), and jeod::DynBody::set_velocity().

7.1.2.6 collect_insert()

Definition at line 92 of file body_force_collect.hh.

7.1.2.7 collect_push_back()

Definition at line 128 of file body_force_collect.hh.

7.1.2.8 release_vector()

Release JEOD-allocated memory in the collect vector.

Parameters

in,out	vec	Collected vectors
--------	-----	-------------------

Definition at line 81 of file body_force_collect.hh.

Referenced by jeod::BodyForceCollect:: \sim BodyForceCollect().

Chapter 8

Data Structure Documentation

8.1 jeod::BodyForceCollect Class Reference

Serves as the collection point for forces and torques that act on a vehicle.

```
#include <body_force_collect.hh>
```

Public Member Functions

- BodyForceCollect ()
 - Default constructor.
- ∼BodyForceCollect ()

Destructor.

- BodyForceCollect (BodyForceCollect &)=delete
- BodyForceCollect & operator= (const BodyForceCollect &)=delete

Data Fields

```
• double effector_forc [3] {}
```

Sum of effector forces, struct ref.

double environ_forc [3] {}

Sum of env forces, struct ref.

double no_xmit_forc [3] {}

Sum of local forces, struct ref.

• double extern_forc_struct [3] {}

Sum of external forces, struct ref.

• double extern_forc_inrtl [3] {}

Sum of external forces, inertial.

double effector_torq [3] {}

Sum of effector torques about body CoM, struct ref.

double environ_torq [3] {}

Sum of environment torqs about body CoM, struct ref.

double no_xmit_torq [3] {}

Sum of torqs not transmitted to a parent about body CoM, struct ref.

double inertial_torq [3] {}

Induced inertial torques from second order rotational dynamics, w x lw, body ref.

double extern_torq_struct [3] {}

Sum of external torques, struct ref.

double extern_torq_body [3] {}

Sum of external torques, body ref.

• JPVCollectForce collect_effector_forc

Vector of effector forces, (struct)

JPVCollectForce collect environ forc

Vector of env forces, (struct)

• JPVCollectForce collect_no_xmit_forc

Vector of local forces, (struct)

JPVCollectTorque collect_effector_torq

Vector of effector torques, (struct)

JPVCollectTorque collect_environ_torq

Vector of env torques, (struct)

JPVCollectTorque collect_no_xmit_torq

Vector of local torques, (struct)

8.1.1 Detailed Description

Serves as the collection point for forces and torques that act on a vehicle.

This class is a simple class that is tightly coupled with the DynBody class. The DynBody class contains (has-a) a BodyForceCollect member.

The Trick vcollect mechanism (or a similar mechanism in a non-Trick sim) pushes the individual forces and torques onto the various collect_XXX members of a BodyForceCollect. DynBody members cumulate these collected forces and torques to form the total forces and torques acting on the vehicle.

Definition at line 270 of file body force collect.hh.

8.1.2 Constructor & Destructor Documentation

8.1.2.1 BodyForceCollect() [1/2]

```
jeod::BodyForceCollect::BodyForceCollect ( )
```

Default constructor.

Definition at line 39 of file body_force_collect.cc.

References collect_effector_forc, collect_effector_torq, collect_environ_forc, collect_environ_torq, collect_no_ xmit_forc, and collect_no_xmit_torq.

8.1.2.2 ~BodyForceCollect()

```
jeod::BodyForceCollect::~BodyForceCollect ( )
```

Destructor.

Definition at line 62 of file body force collect.cc.

References collect_effector_forc, collect_effector_torq, collect_environ_forc, collect_environ_torq, collect_no_collect_no_collect_no_xmit_forc, collect_no_xmit_torq, and jeod::release_vector().

8.1.2.3 BodyForceCollect() [2/2]

8.1.3 Member Function Documentation

8.1.3.1 operator=()

8.1.4 Field Documentation

8.1.4.1 collect_effector_forc

```
JPVCollectForce jeod::BodyForceCollect::collect_effector_forc
```

Vector of effector forces, (struct)

trick_io(**)

Definition at line 341 of file body_force_collect.hh.

Referenced by BodyForceCollect(), jeod::DynBody::collect_forces_and_torques(), jeod::StructureIntegratedDyn \leftarrow Body::collect_local_forces_and_torques(), and \sim BodyForceCollect().

8.1.4.2 collect_effector_torq

JPVCollectTorque jeod::BodyForceCollect::collect_effector_torq

Vector of effector torques, (struct)

trick_io(**)

Definition at line 356 of file body_force_collect.hh.

Referenced by BodyForceCollect(), jeod::DynBody::collect_forces_and_torques(), jeod::StructureIntegratedDyn \leftarrow Body::collect_local_forces_and_torques(), and \sim BodyForceCollect().

8.1.4.3 collect_environ_forc

JPVCollectForce jeod::BodyForceCollect::collect_environ_forc

Vector of env forces, (struct)

trick_io(**)

Definition at line 346 of file body_force_collect.hh.

Referenced by BodyForceCollect(), jeod::DynBody::collect_forces_and_torques(), jeod::StructureIntegratedDyn \leftarrow Body::collect_local_forces_and_torques(), and \sim BodyForceCollect().

8.1.4.4 collect_environ_torq

JPVCollectTorque jeod::BodyForceCollect::collect_environ_torq

Vector of env torques, (struct)

trick io(**)

Definition at line 361 of file body_force_collect.hh.

Referenced by BodyForceCollect(), jeod::DynBody::collect_forces_and_torques(), jeod::StructureIntegratedDyn \leftarrow Body::collect_local_forces_and_torques(), and \sim BodyForceCollect().

8.1.4.5 collect_no_xmit_forc

JPVCollectForce jeod::BodyForceCollect::collect_no_xmit_forc

Vector of local forces, (struct)

trick_io(**)

Definition at line 351 of file body_force_collect.hh.

Referenced by BodyForceCollect(), jeod::DynBody::collect_forces_and_torques(), jeod::StructureIntegratedDyn \leftarrow Body::collect_local_forces_and_torques(), and \sim BodyForceCollect().

```
8.1.4.6 collect_no_xmit_torq
```

```
JPVCollectTorque jeod::BodyForceCollect::collect_no_xmit_torq
```

Vector of local torques, (struct)

```
trick io(**)
```

Definition at line 366 of file body_force_collect.hh.

Referenced by BodyForceCollect(), jeod::DynBody::collect_forces_and_torques(), jeod::StructureIntegratedDyn \leftarrow Body::collect_local_forces_and_torques(), and \sim BodyForceCollect().

8.1.4.7 effector forc

```
double jeod::BodyForceCollect::effector_forc[3] {}
```

Sum of effector forces, struct ref.

trick_units(N)

Definition at line 285 of file body force collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces — and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), and jeod::Structure — IntegratedDynBody::PropagateForcesAndTorques().

8.1.4.8 effector_torq

```
double jeod::BodyForceCollect::effector_torq[3] {}
```

Sum of effector torques about body CoM, struct ref.

trick_units(N*m)

Definition at line 310 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces — _and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), and jeod::Structure — IntegratedDynBody::PropagateForcesAndTorques().

8.1.4.9 environ_forc

```
double jeod::BodyForceCollect::environ_forc[3] {}
```

Sum of env forces, struct ref.

trick units(N)

Definition at line 290 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces \leftarrow _and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), and jeod::Structure \leftarrow IntegratedDynBody::PropagateForcesAndTorques().

8.1.4.10 environ_torq

```
double jeod::BodyForceCollect::environ_torq[3] {}
```

Sum of environment torqs about body CoM, struct ref.

trick_units(N*m)

Definition at line 315 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces \leftarrow _and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), and jeod::Structure \leftarrow IntegratedDynBody::PropagateForcesAndTorques().

8.1.4.11 extern_forc_inrtl

```
double jeod::BodyForceCollect::extern_forc_inrtl[3] {}
```

Sum of external forces, inertial.

trick_units(N)

Definition at line 305 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces_ and_torques(), and jeod::StructureIntegratedDynBody::compute_translational_acceleration().

8.1.4.12 extern_forc_struct

```
double jeod::BodyForceCollect::extern_forc_struct[3] {}
```

Sum of external forces, struct ref.

trick_units(N)

Definition at line 300 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces \leftarrow _and_torques(), jeod::StructureIntegratedDynBody::compute_translational_acceleration(), and jeod::Structure \leftarrow IntegratedDynBody::solve_constraints().

8.1.4.13 extern_torq_body

```
double jeod::BodyForceCollect::extern_torq_body[3] {}
```

Sum of external torques, body ref.

trick_units(N*m)

Definition at line 336 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces_
and_torques(), and jeod::StructureIntegratedDynBody::compute_rotational_acceleration().

8.1.4.14 extern_torq_struct

```
double jeod::BodyForceCollect::extern_torq_struct[3] {}
```

Sum of external torques, struct ref.

trick_units(N*m)

Definition at line 331 of file body_force_collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces_\Limin and_torques(), and jeod::StructureIntegratedDynBody::compute_rotational_acceleration().

8.1.4.15 inertial_torq

```
double jeod::BodyForceCollect::inertial_torq[3] {}
```

Induced inertial torques from second order rotational dynamics, w x lw, body ref.

trick_units(N*m)

Definition at line 326 of file body force collect.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces_\(\to \) and_torques(), jeod::StructureIntegratedDynBody::compute_inertial_torque(), jeod::StructureIntegratedDynBody\(\to \) ::compute_rotational_acceleration(), and jeod::StructureIntegratedDynBody::solve_constraints().

8.1.4.16 no_xmit_forc

```
double jeod::BodyForceCollect::no_xmit_forc[3] {}
```

Sum of local forces, struct ref.

trick units(N)

Definition at line 295 of file body force collect.hh.

 $Referenced \ by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces_\leftrightarrow and_torques(), and jeod::StructureIntegratedDynBody::collect_local_forces_and_torques().$

8.1.4.17 no_xmit_torq

```
double jeod::BodyForceCollect::no_xmit_torq[3] {}
```

Sum of torqs not transmitted to a parent about body CoM, struct ref.

trick_units(N*m)

Definition at line 320 of file body_force_collect.hh.

 $Referenced \ by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces_{\leftarrow} and_torques(), and jeod::StructureIntegratedDynBody::collect_local_forces_and_torques().$

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

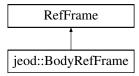
- · body force collect.hh
- body_force_collect.cc

8.2 jeod::BodyRefFrame Class Reference

Extend RefFrame to add coupling between the reference frame tree and the mass tree and to keep track of which state items have been set.

```
#include <body_ref_frame.hh>
```

Inheritance diagram for jeod::BodyRefFrame:



Public Member Functions

- BodyRefFrame ()=default
- ~BodyRefFrame () override=default
- BodyRefFrame (const BodyRefFrame &)=delete
- BodyRefFrame & operator= (const BodyRefFrame &)=delete

Data Fields

• RefFrameItems initialized_items

Specifies which state elements (position, velocity, attitude, and rate) have been initialized.

MassPoint * mass point {}

Pointer to the mass point that defines the origin and orientation of this frame, but with respect to the mass tree rather than with respect to the reference frame tree.

Friends

- · class InputProcessor
- void init_attrjeod__BodyRefFrame ()

8.2.1 Detailed Description

Extend RefFrame to add coupling between the reference frame tree and the mass tree and to keep track of which state items have been set.

Definition at line 77 of file body_ref_frame.hh.

8.2.2 Constructor & Destructor Documentation

```
8.2.2.1 BodyRefFrame() [1/2]
jeod::BodyRefFrame::BodyRefFrame ( ) [default]
8.2.2.2 \simBodyRefFrame()
jeod::BodyRefFrame::~BodyRefFrame ( ) [override], [default]
8.2.2.3 BodyRefFrame() [2/2]
jeod::BodyRefFrame::BodyRefFrame (
            const BodyRefFrame & ) [delete]
8.2.3 Member Function Documentation
```

8.2.3.1 operator=()

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

8.2.4 Friends And Related Function Documentation

8.2.4.1 init_attrjeod__BodyRefFrame

```
void init_attrjeod__BodyRefFrame ( ) [friend]
```

8.2.4.2 InputProcessor

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

Definition at line 79 of file body_ref_frame.hh.

8.2.5 Field Documentation

8.2.5.1 initialized_items

```
RefFrameItems jeod::BodyRefFrame::initialized_items
```

Specifies which state elements (position, velocity, attitude, and rate) have been initialized.

trick_units(-)

Definition at line 86 of file body ref frame.hh.

Referenced by jeod::DynBody::compute_derived_state_forward(), jeod::DynBody::compute_derived_state capture cap

8.2.5.2 mass_point

```
MassPoint* jeod::BodyRefFrame::mass_point {}
```

Pointer to the mass point that defines the origin and orientation of this frame, but with respect to the mass tree rather than with respect to the reference frame tree.

```
trick_units(-)
```

Definition at line 93 of file body_ref_frame.hh.

Referenced by jeod::DynBody::add_mass_body(), jeod::DynBody::add_mass_body_frames(), jeod::DynBody::atda_mass_body_frames(), jeod::DynBody::attach_child(), jeod::DynBody::attach_to_frame(), jeod::DynBody::compute compute compute transform(), jeod::StructureIntegratedDynBody::compute_vehicle_point_derivatives(), jeod::DynBody compute_vehicle_point_derivatives(), and jeod::DynBody::DynBody().

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

• body_ref_frame.hh

8.3 jeod::BodyWrenchCollect Class Reference

Serves as the collection point for wrenches that act on a vehicle.

```
#include <body_wrench_collect.hh>
```

Public Member Functions

• BodyWrenchCollect ()

Default constructor.

∼BodyWrenchCollect ()

Destructor.

- BodyWrenchCollect (const BodyWrenchCollect &)=delete
- BodyWrenchCollect & operator= (const BodyWrenchCollect &)=delete
- · Wrench & accumulate (Wrench &sum) const

Accumulate the collected wrenches.

• Wrench & accumulate (const double point[3], Wrench &sum) const

Accumulate the collected wrenches.

Data Fields

JeodPointerVector < Wrench >::type collect_wrench

Vector of effector wrenches.

8.3.1 Detailed Description

Serves as the collection point for wrenches that act on a vehicle.

This is a simple class that is tightly coupled with the StructureIntegratedDynBody class. This latter class contains (has-a) a BodyWrenchCollect data member.

The Trick vcollect mechanism (or a similar mechanism in a non-Trick sim) pushes pointers to the individual wrenches onto the various collection member of a BodyWrenchCollect. StructureIntegratedDynBody members cumulate these collected wrenches to form the total wrench acting on the vehicle.

Definition at line 78 of file body_wrench_collect.hh.

8.3.2 Constructor & Destructor Documentation

```
8.3.2.1 BodyWrenchCollect() [1/2]
```

```
jeod::BodyWrenchCollect::BodyWrenchCollect ( )
```

Default constructor.

Definition at line 25 of file body wrench collect.cc.

References collect_wrench.

8.3.2.2 ~BodyWrenchCollect()

```
jeod::BodyWrenchCollect::~BodyWrenchCollect ( )
```

Destructor.

Definition at line 32 of file body wrench collect.cc.

References collect_wrench.

8.3.2.3 BodyWrenchCollect() [2/2]

8.3.3 Member Function Documentation

8.3.3.1 accumulate() [1/2]

Accumulate the collected wrenches.

Parameters

sum Wrench into which the accumulated sum is to be placed. The summation is about sum.point.

Returns

Reference to the input wrench.

Definition at line 124 of file body_wrench_collect.hh.

References jeod::Wrench::accumulate(), and collect_wrench.

Referenced by accumulate(), and jeod::StructureIntegratedDynBody::collect_local_forces_and_torques().

8.3.3.2 accumulate() [2/2]

Accumulate the collected wrenches.

point	Point about which summation is to be performed.
sum	Wrench into which the accumulated sum is to be placed.

Returns

Reference to the input wrench.

Definition at line 136 of file body_wrench_collect.hh.

References accumulate(), and jeod::Wrench::set_point().

8.3.3.3 operator=()

8.3.4 Field Documentation

8.3.4.1 collect_wrench

```
{\tt JeodPointerVector}{<}{\tt Wrench}{>}{\tt ::type jeod::BodyWrenchCollect::collect\_wrench}
```

Vector of effector wrenches.

The effector wrenches are collected into the vector at the S_define level via & vcollect containing_body.effector_
wrench_collection.collect_wrench { pointer_to_wrench1, ... pointer_to_wrench_n };

The vector of collected wrenches are processed by the containing body's collect_forces_and_torques member function.trick_io(**)

Definition at line 97 of file body_wrench_collect.hh.

Referenced by accumulate(), BodyWrenchCollect(), and \sim BodyWrenchCollect().

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

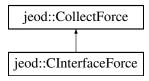
- · body_wrench_collect.hh
- body_wrench_collect.cc

8.4 jeod::CInterfaceForce Class Reference

This class is deprecated.

```
#include <force.hh>
```

Inheritance diagram for jeod::CInterfaceForce:



Public Member Functions

- CInterfaceForce ()=default
- CInterfaceForce (double *vec)

CInterfaceForce constructor for use with C force array.

∼CInterfaceForce () override

CInterfaceForce destructor; frees 'active' but not the force.

- CInterfaceForce (const CInterfaceForce &)=delete
- CInterfaceForce & operator= (const CInterfaceForce &)=delete

Additional Inherited Members

8.4.1 Detailed Description

This class is deprecated.

Definition at line 182 of file force.hh.

8.4.2 Constructor & Destructor Documentation

CInterfaceForce constructor for use with C force array.

Note that the new CInterfaceForce's force *is* the force_3vec.

in,out	force_3vec	Force vector to encapsulate	1
		Units: N	

Definition at line 75 of file force.cc.

References jeod::CollectForce::active, and jeod::CollectForce::force.

8.4.2.3 ~CInterfaceForce()

```
jeod::CInterfaceForce::~CInterfaceForce ( ) [override]
```

CInterfaceForce destructor; frees 'active' but not the force.

Definition at line 85 of file force.cc.

References jeod::CollectForce::active.

8.4.2.4 CInterfaceForce() [3/3]

8.4.3 Member Function Documentation

8.4.3.1 operator=()

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

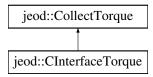
- · force.hh
- force.cc

8.5 jeod::CInterfaceTorque Class Reference

This class is deprecated.

```
#include <torque.hh>
```

Inheritance diagram for jeod::CInterfaceTorque:



Public Member Functions

- CInterfaceTorque ()=default
- CInterfaceTorque (double *vec)

CInterface Torque constructor for use with C torque array.

CInterfaceTorque () override

CInterface Torque destructor; frees 'active' but not the torque.

- CInterfaceTorque (const CInterfaceTorque &)=delete
- CInterfaceTorque & operator= (const CInterfaceTorque &)=delete

Additional Inherited Members

8.5.1 Detailed Description

This class is deprecated.

Definition at line 182 of file torque.hh.

8.5.2 Constructor & Destructor Documentation

CInterfaceTorque constructor for use with C torque array.

Note that the new CInterfaceTorque's torque *is* the torque_3vec.

in,out	torque_3vec	Torque vector to encapsulate
		Units: NM

Definition at line 75 of file torque.cc.

References jeod::CollectTorque::active, and jeod::CollectTorque::torque.

8.5.2.3 ∼CInterfaceTorque()

```
jeod::CInterfaceTorque::~CInterfaceTorque ( ) [override]
```

CInterfaceTorque destructor; frees 'active' but not the torque.

Definition at line 85 of file torque.cc.

References jeod::CollectTorque::active.

8.5.2.4 CInterfaceTorque() [3/3]

8.5.3 Member Function Documentation

8.5.3.1 operator=()

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

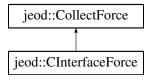
- · torque.hh
- torque.cc

8.6 jeod::CollectForce Class Reference

A CollectForce represents a collected force that acts on a vehicle.

#include <force.hh>

Inheritance diagram for jeod::CollectForce:



Public Member Functions

- CollectForce ()=default
- CollectForce (double vec[3])

CollectForce constructor that encapsulates a C-style 3-vector.

CollectForce (Force &)

CollectForce constructor that encapsulates a Force.

• CollectForce (CollectForce &)

CollectForce constructor that encapsulates another CollectForce.

- virtual ∼CollectForce ()=default
- CollectForce (const CollectForce &)=delete
- CollectForce & operator= (const CollectForce &)=delete
- bool is_active () const

A force is active if it has a non-null force vector and the active pointer is null or the pointed-to boolean is true.

double & operator[] (const unsigned int index)

Access a force element, non-const version.

• double operator[] (const unsigned int index) const

Access a force element, const version.

• bool operator== (const CollectForce &other)

Static Public Member Functions

static CollectForce * create (double *vec)

Create a CollectForce whose force is the specified array.

static CollectForce * create (Force &force)

Create a shallow copy of a Force.

static CollectForce * create (CollectForce &force)

Create a shallow copy of a CollectForce.

static CollectForce * create (Force *force)

Create a shallow copy of a Force.

• static CollectForce * create (CollectForce *force)

Create a shallow copy of a CollectForce.

Data Fields

```
    bool * active {}
        Is this force active?
    double * force {}
        Force vector.
```

8.6.1 Detailed Description

A CollectForce represents a collected force that acts on a vehicle.

The BodyForceCollect class contains STL vectors that in turn contain CollectForce pointers. These vectors are populated via the Trick vcollect mechanism. A Trick simulation issues vcollect statements such as

```
vcollect vehicle.body.collect.collect_XXX_forc CollectForce::create {
   vehicle.force_model1.force,
   vehicle.force_model2.force
}:
```

This invokes the appropriate CollectForce create method on each listed element.

CollectForces should not be used in model code to represent forces. Use the Force class instead.

Definition at line 126 of file force.hh.

8.6.2 Constructor & Destructor Documentation

CollectForce constructor that encapsulates a C-style 3-vector.

Note that the new CollectForce's force *is* the force_3vec.

Parameters

in,out	force_3vec	Force vector to encapsulate
		Units: N

Definition at line 54 of file force.cc.

8.6.2.3 CollectForce() [3/5]

CollectForce constructor that encapsulates a Force.

Note that this performs a shallow copy by intent.

Parameters

Definition at line 43 of file force.cc.

8.6.2.4 CollectForce() [4/5]

CollectForce constructor that encapsulates another CollectForce.

Note that this performs a shallow copy by intent.

Parameters

in,out	source_force	Force to encapsulate

Definition at line 64 of file force.cc.

8.6.2.5 \sim CollectForce()

```
virtual jeod::CollectForce::~CollectForce ( ) [virtual], [default]
```

8.6.2.6 CollectForce() [5/5]

8.6.3 Member Function Documentation

Create a CollectForce whose force is the specified array.

Note that the created instance is actually a CInterfaceForce.

Returns

Constructed CollectForce

Parameters

in,out	force_3vec	Force vector to encapsulate
		Units: N

Definition at line 120 of file force.cc.

Referenced by create().

Create a shallow copy of a Force.

Note that the new CollectForce refers to the Force's active flag and force array.

Returns

Constructed CollectForce

Parameters

in,out	source_force	Force object to encapsulate

Definition at line 97 of file force.cc.

Create a shallow copy of a CollectForce.

Note that both the source and new CollectForces refer to the same active flag and force array.

Returns

Constructed CollectForce

Parameters

```
in, out | source_force | Force to copy
```

Definition at line 132 of file force.cc.

Create a shallow copy of a Force.

Note that the new CollectForce refers to the Force's active flag and force array.

Returns

Constructed CollectForce

Parameters

```
in, out | source_force | Force object to encapsulate
```

Definition at line 109 of file force.cc.

References create().

Create a shallow copy of a CollectForce.

Note that both the source and new CollectForces refer to the same active flag and force array.

Returns

Constructed CollectForce

Parameters

```
in, out source_force Force to copy
```

Definition at line 144 of file force.cc.

References create().

8.6.3.6 is_active()

```
bool jeod::CollectForce::is_active ( ) const [inline]
```

A force is active if it has a non-null force vector and the active pointer is null or the pointed-to boolean is true.

Returns

Is the force active?

Definition at line 93 of file force_inline.hh.

References active, and force.

8.6.3.7 operator=()

8.6.3.8 operator==()

Definition at line 160 of file force.hh.

References force.

```
8.6.3.9 operator[]() [1/2]
```

Access a force element, non-const version.

Returns

Force component at specified index

Units: N

in	index	Index number
	mack	IIIack Hailibei

Definition at line 103 of file force_inline.hh.

References force.

Access a force element, const version.

Returns

Force component at specified index Units: N

Parameters

in	index	Index number
----	-------	--------------

Definition at line 113 of file force_inline.hh.

References force.

8.6.4 Field Documentation

8.6.4.1 active

```
bool* jeod::CollectForce::active {}
```

Is this force active?

trick_units(-)

Definition at line 171 of file force.hh.

Referenced by jeod::CInterfaceForce::CInterfaceForce(), is_active(), and jeod::CInterfaceForce:: \sim CInterfaceForce().

8.6.4.2 force

```
double* jeod::CollectForce::force {}
```

Force vector.

trick units(N)

Definition at line 176 of file force.hh.

Referenced by jeod::CInterfaceForce::CInterfaceForce(), is_active(), operator==(), and operator[]().

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

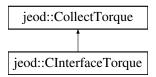
- · force.hh
- force_inline.hh
- · force.cc

8.7 jeod::CollectTorque Class Reference

A CollectTorque represents a collected torque that acts on a vehicle.

```
#include <torque.hh>
```

Inheritance diagram for jeod::CollectTorque:



Public Member Functions

- CollectTorque ()=default
- CollectTorque (double vec[3])

CollectTorque constructor that encapsulates a C-style 3-vector.

CollectTorque (Torque &)

CollectTorque constructor that encapsulates a Torque.

CollectTorque (CollectTorque &)

CollectTorque constructor that encapsulates another CollectTorque.

- virtual ∼CollectTorque ()=default
- CollectTorque (const CollectTorque &)=delete
- CollectTorque & operator= (const CollectTorque &)=delete
- bool is active () const

A torque is active if it has a non-null torque vector and the active pointer is null or the pointed-to boolean is true.

double & operator[] (const unsigned int index)

Access a torque element, non-const version.

double operator[] (const unsigned int index) const

Access a torque element, const version.

bool operator== (const CollectTorque &other)

Static Public Member Functions

```
    static CollectTorque * create (double *vec)
```

Create a CollectTorque whose torque is the specified array.

static CollectTorque * create (Torque &torque)

Create a shallow copy of a Torque.

static CollectTorque * create (CollectTorque &torque)

Create a shallow copy of a CollectTorque.

static CollectTorque * create (Torque *torque)

Create a shallow copy of a Torque.

static CollectTorque * create (CollectTorque *torque)

Create a shallow copy of a CollectTorque.

Data Fields

```
    bool * active {}
        Is this torque active?
    double * torque {}
        Torque vector.
```

8.7.1 Detailed Description

A CollectTorque represents a collected torque that acts on a vehicle.

The BodyTorqueCollect class contains STL vectors that in turn contain CollectTorque pointers. These vectors are populated via the Trick vcollect mechanism. A Trick simulation issues vcollect statements such as

```
vcollect vehicle.body.collect.collect_XXX_forc CollectTorque::create {
   vehicle.torque_model1.torque,
   vehicle.torque_model2.torque
};
```

This invokes the appropriate CollectTorque create method on each listed element.

CollectTorques should not be used in model code to represent torques. Use the Torque class instead.

Definition at line 125 of file torque.hh.

8.7.2 Constructor & Destructor Documentation

CollectTorque constructor that encapsulates a C-style 3-vector.

Note that the new CollectTorque's torque is the torque_3vec.

in,out	torque_3vec	Torque vector to encapsulate
		Units: NM

Definition at line 54 of file torque.cc.

8.7.2.3 CollectTorque() [3/5]

CollectTorque constructor that encapsulates a Torque.

Note that this performs a shallow copy by intent.

Parameters

in,out	source_torque	Torque to encapsulate
--------	---------------	-----------------------

Definition at line 43 of file torque.cc.

8.7.2.4 CollectTorque() [4/5]

CollectTorque constructor that encapsulates another CollectTorque.

Note that this performs a shallow copy by intent.

Parameters

in,out	source_torque	Torque to encapsulate

Definition at line 64 of file torque.cc.

8.7.2.5 \sim CollectTorque()

```
virtual jeod::CollectTorque::~CollectTorque ( ) [virtual], [default]
```

8.7.2.6 CollectTorque() [5/5]

8.7.3 Member Function Documentation

Create a CollectTorque whose torque is the specified array.

Note that the created instance is actually a CInterfaceTorque.

Returns

Constructed CollectTorque

Parameters

in,out	torque_3vec	Torque vector to encapsulate
		Units: NM

Definition at line 120 of file torque.cc.

Referenced by create().

Create a shallow copy of a Torque.

Note that the new CollectTorque refers to the Torque's active flag and torque array.

Returns

Constructed CollectTorque

Definition at line 97 of file torque.cc.

Create a shallow copy of a CollectTorque.

Note that both the source and new CollectTorques refer to the same active flag and torque array.

Returns

Constructed CollectTorque

Parameters

in, out source_torque Torque to copy
--

Definition at line 132 of file torque.cc.

Create a shallow copy of a Torque.

Note that the new CollectTorque refers to the Torque's active flag and torque array.

Returns

Constructed CollectTorque

Parameters

in,out	source_torque	Torque object to encapsulate
--------	---------------	------------------------------

Definition at line 109 of file torque.cc.

References create().

Create a shallow copy of a CollectTorque.

Note that both the source and new CollectTorques refer to the same active flag and torque array.

Returns

Constructed CollectTorque

Parameters

in, out source_torque Torque

Definition at line 144 of file torque.cc.

References create().

8.7.3.6 is_active()

```
bool jeod::CollectTorque::is_active ( ) const [inline]
```

A torque is active if it has a non-null torque vector and the active pointer is null or the pointed-to boolean is true.

Returns

Is the torque active?

Definition at line 93 of file torque_inline.hh.

References active, and torque.

8.7.3.7 operator=()

8.7.3.8 operator==()

Definition at line 160 of file torque.hh.

References torque.

8.7.3.9 operator[]() [1/2]

Access a torque element, non-const version.

Returns

Torque component at specified index

Units: N

Parameters

in	index	Index number
----	-------	--------------

Definition at line 103 of file torque_inline.hh.

References torque.

8.7.3.10 operator[]() [2/2]

Access a torque element, const version.

Returns

Torque component at specified index

Units: N

Parameters

in	index	Index number
----	-------	--------------

Definition at line 113 of file torque_inline.hh.

References torque.

8.7.4 Field Documentation

8.7.4.1 active

```
bool* jeod::CollectTorque::active {}
```

Is this torque active?

trick_units(-)

Definition at line 171 of file torque.hh.

Referenced by jeod::CInterfaceTorque::CInterfaceTorque(), is_active(), and jeod::CInterfaceTorque::~CInterface Torque().

8.7.4.2 torque

```
double* jeod::CollectTorque::torque {}
```

Torque vector.

trick_units(N*m)

Definition at line 176 of file torque.hh.

Referenced by jeod::CInterfaceTorque::CInterfaceTorque(), is_active(), operator==(), and operator[]().

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

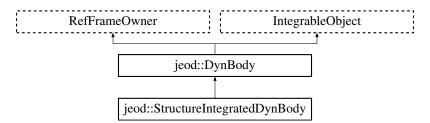
- torque.hh
- · torque_inline.hh
- torque.cc

8.8 jeod::DynBody Class Reference

Class DynBody is the base class for all dynamic bodies.

```
#include <dyn_body.hh>
```

Inheritance diagram for jeod::DynBody:



Public Member Functions

• DynBody ()

DynBody default constructor.

∼DynBody () override

DynBody destructor.

- DynBody (const DynBody &)=delete
- DynBody & operator= (const DynBody &)=delete
- virtual void initialize model (BaseDynManager &dyn manager in)

Initialize internal and external interrelations, including registration / with the dynamics manager.

void activate ()

Activate a DynBody object.

· void deactivate ()

Deactivate a DynBody object.

void set name (const std::string &name in)

Set the name of the vehicle.

virtual void add control (GravityControls *control)

Add a new GravityControls to the list in grav interaction.

virtual void initialize_controls (GravityManager &grav_manager)

Initialize the gravity controls of this DynBody.

• virtual void reset controls ()

Make the frame subscriptions for each control consistent with the requirements for that control.

virtual void sort_controls ()

Sort the gravity controls in ascending acceleration magnitude order.

• virtual void collect_forces_and_torques ()

Collect forces and torques acting on the vehicle.

virtual void create_body_integrators (const er7_utils::IntegratorConstructor &generator, er7_utils::
 —
 IntegrationControls &controls, const JeodIntegrationTime &time_mngr)

 $\textit{Create the integrator (integrators) needed to propagate the translational and rotational state of a \textit{DynBody}.}$

er7_utils::IntegratorResult integrate (double dyn_dt, unsigned int target_stage) override

Integrate state by the specified dynamic time interval.

• virtual EphemerisRefFrame * get_integ_frame () const

Get the integration frame for this body.

virtual void switch_integration_frames (EphemerisRefFrame &new_integ_frame)

Switch the integration frame for this body and all its child bodies to the indicated frame.

virtual void switch_integration_frames (const std::string &new_integ_frame_name)

Switch the integration frame for this body and all its child bodies to the frame indicated by the provided name.

• void create_integrators (const er7_utils::IntegratorConstructor &generator, er7_utils::IntegrationControls &controls, const er7_utils::TimeInterface &time_if) override

This interface is required by er7_utils::IntegrableObject.

void destroy_integrators () override

Destroy the integrators.

• void reset_integrators () override

Reset the translational and rotational integrators.

virtual BodyRefFrame * find body frame (const std::string &frame id) const

Find the BodyRefFrame named by the provided identifier.

DynamicsIntegrationGroup * get_dynamics_integration_group ()

Get the DynamicsIntegrationGroup that integrates this DynBody object.

JeodPointerVector< er7 utils::IntegrableObject >::type get integrable objects ()

Get the IntegrableObjects associated with this DynBody.

• void clear_integrable_objects ()

Remove all IntegrableObjects associated with this DynBody.

void migrate_integrable_objects ()

Call this method before switching this dyn body to a new group if you want the associated integrable objects to follow.

void add_integrable_object (er7_utils::IntegrableObject &associated_integrable_object)

Add an IntegrableObject to be integrated with this DynBody.

void remove_integrable_object (er7_utils::IntegrableObject &associated_integrable_object)

Remove an IntegrableObject from association with this DynBody.

void set_position (const double position[3], BodyRefFrame &subject_frame)

Set the position of the vehicle.

void set velocity (const double velocity[3], BodyRefFrame &subject frame)

Set the velocity of the vehicle.

void set_attitude_left_quaternion (const Quaternion &left_quat, BodyRefFrame &subject_frame)

Set the attitude of the vehicle.

void set attitude right quaternion (const Quaternion & right quat, BodyRefFrame & subject frame)

Set the attitude of the vehicle.

void set attitude matrix (const double matrix[3][3], BodyRefFrame &subject frame)

Set the attitude of the vehicle.

void set attitude rate (const double attitude rate[3], BodyRefFrame &subject frame)

Set the attitude rate of the vehicle.

void set_state (RefFrameItems::Items set_items, const RefFrameState &state, BodyRefFrame &subject_
 frame)

Set the parts of the specified reference frame as indicated by the set_items parameter from the supplied state and propagate these items to all dynamic bodies attached to this body.

void set state source (RefFrameItems::Items items, BodyRefFrame &frame)

Set the source of aspects of the state.

· virtual void propagate_state ()

Propagate state from the integrated state to attached bodies.

virtual void update_integrated_state ()

Propagate state from state owners to the integrated state.

• virtual void compute_vehicle_point_states (RefFrameItems::Items set_items)

Propagate structure frame state to vehicle points.

bool is_root_body ()

Indicates whether this DynBody object is a root body.

virtual const DynBody * get_parent_body () const

Returns this DynBody object's parent body.

virtual const DynBody * get_root_body () const

Finds this DynBody object's root body.

virtual void add_mass_point (const MassPointInit &mass_point_init)

Add a mass point to the dyn body's list of such and make a vehicle point that corresponds to the added mass point.

const BodyRefFrame * find_vehicle_point (const std::string &pt_name) const

Find the vehicle point with the given name.

virtual void compute_vehicle_point_derivatives (const BodyRefFrame &frame, FrameDerivs &derivs)

Compute the state derivatives at a vehicle point.

const RefFrameItems & get_initialized_states () const

Indicate which state elements have been initialized.

• bool initialized_states_contains (RefFrameItems::Items test_items) const

Indicate whether the specified state elements have been initialized.

- virtual bool add_mass_body (const std::string &this_point_name, const std::string &child_point_name, MassBody &child)
- virtual bool add_mass_body (const double offset[3], const double T_pstr_cstr[3][3], MassBody &child)

virtual bool attach_to (const std::string &this_point_name, const std::string &parent_point_name, DynBody &parent)

Attach this dyn body's root body as a child of the specified dyn body such that the specified mass points on the two bodies are coincident and the frames associated with those mass points are related by a 180 degree yaw.

virtual bool attach_to (const double offset_pstr_cstr_pstr[3], const double T_pstr_cstr[3][3], DynBody &parent)
 Attach this dyn body's root body as a child of the specified dyn body such that this body's structural origin is offset from the parent body's structural origin and this body's structural axes are oriented with respect to the parent body's

virtual bool attach_child (const std::string &this_point_name, const std::string &child_point_name, DynBody &child)

Attach a child DynBody by point specification.

structural axes as specified.

 virtual bool attach_child (const double offset_pstr_cstr_pstr[3], const double T_pstr_cstr[3][3], DynBody &child)

Attach a child DynBody by location specification.

- virtual bool attach to frame (const std::string &parent ref frame name)
- virtual bool attach_to_frame (RefFrame &parent)
- virtual bool attach_to_frame (const std::string &this_point_name, const std::string &parent_ref_frame_name, const double offset_pframe_cpt_pframe[3], const double T_pframe_cpt[3][3])
- virtual bool attach_to_frame (const double offset_pframe_cstr_pframe[3], const double T_pframe_cstr[3][3], RefFrame &parent)
- virtual bool detach (DynBody &other_body)

Detach parent and child DynBodies, 'this' and the argument body, such that the detachment happens at the parent body level.

• virtual bool detach ()

Detach this DynBody from its parent RefFrame or DynBody parent.

virtual bool remove_mass_body (MassBody &child)

Remove connectivity between this (parent) DynBody and the argument (child) MassBody mass subbody.

Data Fields

· MassBody mass

Mass properties of the vehicle, defined about the structure reference frame.

NamedItem & name

Body name, reference linked to mass.name.

std::string integ_frame_name

The name of the reference frame with respect to which the body's reference frames (core, composite, structure, plus vehicle point frames) are to be represented and propagated.

· BodyRefFrame core body

Vehicle core body reference frame.

BodyRefFrame composite_body

Vehicle composite body reference frame.

• BodyRefFrame structure

Vehicle structural reference frame.

• bool translational_dynamics {}

Is translational dynamics enabled? The body's translational state is integrated only if this member is true.

bool rotational_dynamics {}

Is rotational dynamics enabled? The body's rotational state is integrated only if this member is true.

bool compute point derivative {}

Should the point derivatives for the body be computed? A child body's translational and rotational derivatives are only computed if this is true.

bool three_dof {}

Is this a three degrees of freedom (translation only) body? This data member has effect only when set prior to the creation of the body's integrators.

• GeneralizedSecondOrderODETechnique::TechniqueType rotation_integration

Specifies the preferred mechanism for integrating rotational state.

bool autoupdate vehicle points (true)

Are vehicle points automatically updated? The vehicle points are automatically calculated at initialization time but are only automatically updated at runtime if this member is true.

· GravityInteraction grav_interaction

Gravitational interactions.

· FrameDerivs derivs

Translational/rotational accelerations.

• BodyForceCollect collect

Force/Torque collection mechanism.

Protected Member Functions

virtual void set_integ_frame (EphemerisRefFrame &new_integ_frame)

Set the integration frame for this body and all its child bodies to the provided frame.

virtual void set_integ_frame (const std::string &new_integ_frame_name)

Set the integration frame for this body and all its child bodies to the frame indicated by the provided name.

virtual er7_utils::IntegratorResult trans_integ (double dyn_dt, unsigned int target_stage)

Integrate the vehicle's translational state.

virtual er7_utils::IntegratorResult rot_integ (double dyn_dt, unsigned int target_stage)

Integrate the vehicle's rotational state.

• void set_state_source_internal (RefFrameItems::Items items, BodyRefFrame &frame)

Set the source of aspects of the state.

virtual DynBody * get_parent_body_internal ()

Returns this DynBody object's parent body.

virtual DynBody * get_root_body_internal ()

Finds this DynBody object's root body.

virtual bool attach_validate_parent (const DynBody &parent, bool generate_message) const

Validate whether the pending attachment is legal from a connectivity point of view.

virtual bool attach_validate_child (const DynBody &child, bool generate_message) const

Validate whether the pending attachment is legal from a physical point of view.

• virtual bool add_mass_body_validate (const MassBody &child, bool generate_message) const

Validate whether the pending sub body is legal from a mass tree point of view.

virtual void add_mass_body_frames (MassBody &subbody)

For a newly attached mass sub-body, create body frames for the root sub-body and all child sub-bodies via recursion.

virtual void detach mass body frames (MassBody &subbody)

For a newly detached mass sub-body, remove body frames for the root sub-body and all child sub-bodies via recursion.

virtual void attach_establish_links (DynBody &parent)

Establish the logical connectivity between parent and child.

virtual void attach_update_properties (const double offset_pstr_cstr_pstr[3], const double T_pstr_cstr[3][3],
 DynBody &child)

Set the relation between parent and child and update the mass properties.

virtual void process_dynamic_attachment (const double offset_pstr_cstr_pstr[3], const double T_pstr_cstr[3][3], DynBody &root_body, DynBody &child_body)

Process the attachment event of one body from another.

virtual void detach mass internal (MassBody &child)

Update parent and child properties to reflect that they are detached.

virtual void propagate_state_from_structure ()

Propagate state to attached bodies starting from this body's structural frame.

virtual void propagate_state_from_composite ()

Propagate state to attached bodies starting from this body's composite frame.

 void compute_ref_point_transform (const BodyRefFrame &source_frame, const MassPoint **const ref_point, MassPointState &rel_state)

Compute the relative state between the integrated frame's mass point and the source frame's mass point.

 void compute_derived_state_forward (const BodyRefFrame &source_frame, const MassPoint &rel_state, BodyRefFrame &derived frame) const

Compute a derived state given the source state and the position/attitude transformation from the source to the derived state

 void compute_state_elements_forward (const BodyRefFrame &source_frame, const MassPoint &rel_state, const RefFrameItems &state items, BodyRefFrame &derived frame) const

Compute selected aspects of the derived state given the source state and the position/ attitude transformation from the source to the derived state.

 void compute_derived_state_reverse (const BodyRefFrame &source_frame, const MassPoint &rel_state, BodyRefFrame &derived_frame) const

Compute a derived state given the source state and the position/attitude transformation from the derived to the source state.

• void compute_state_elements_reverse (const BodyRefFrame &source_frame, const MassPoint &rel_state, const RefFrameItems &state items, BodyRefFrame &derived frame) const

Compute selected aspects of the derived state given the source state and the position/ attitude transformation from the derived to the source state.

Protected Attributes

• EphemerisRefFrame * integ_frame {}

The current integration frame.

BaseDynManager *& dyn manager

The dynamics manager for the simulation.

const JeodIntegrationTime * time_manager {}

The time manager to be used to obtain timestamp information.

DynBody * dyn_parent {}

The DynBody to which this body is attached.

DynBodyGenericFrameAttachment frame_attach

The RefFrame this body is attached to.

std::list< DynBody * > dyn_children

The subset of the dynamic bodies attached to this dynamic body.

std::list< MassBody * > mass_children

The subset of the mass bodies attached to this dynamic body that are themselves not dynamic bodies.

std::list< BodyRefFrame * > vehicle points

An array of vehicle points associated with this dynamic body.

• RefFrameItems initialized_states {RefFrameItems::No_Items}

Enum value indicating which of position, velocity, attitude, and rate have been initialized.

• BodyRefFrame * position_source {}

The reference frame that contains the user-set position.

BodyRefFrame * velocity_source {}

The reference frame that contains the user-set velocity.

BodyRefFrame * attitude_source {}

The reference frame that contains the user-set attitude.

BodyRefFrame * rate_source {}

The reference frame that contains the user-set attitude rate.

BodyRefFrame * integrated_frame {}

The reference frame whose state is updated via the state integrator.

std::vector< er7_utils::IntegrableObject * > associated_integrable_objects

List of integrable objects to be integrated with this DynBody.

er7_utils::IntegratorResultMergerContainer integ_results_merger

The object that merges integration results.

RestartableT3SecondOrderODEIntegrator trans integrator

Translational state checkpointable/restartable integrator generator.

• RestartableSO3SecondOrderODEIntegrator rot_integrator

Rotational state checkpointable/restartable integrator generator.

Friends

- class InputProcessor
- void init_attrjeod__DynBody ()

8.8.1 Detailed Description

Class DynBody is the base class for all dynamic bodies.

A DynBody is a MassBody that is connected to the outside world. These connections are in the form of three reference frames tied to the body – the structural, core body, and composite body frames.

For a non-root body, the states for each of these frames is calculated based on the parent body's state and on the body attachment.

For a root body, one of these three frames must be integrated. The details of how that integration is performed is the subject of classes that derive from DynBody.

Definition at line 111 of file dyn_body.hh.

8.8.2 Constructor & Destructor Documentation

```
8.8.2.1 DynBody() [1/2]
jeod::DynBody::DynBody ( )
```

DynBody default constructor.

Definition at line 61 of file dyn_body.cc.

References composite_body, core_body, integrated_frame, mass, jeod::BodyRefFrame::mass_point, rot_integrator, structure, and trans_integrator.

8.8.2.2 \sim DynBody()

```
jeod::DynBody::~DynBody ( ) [override]
```

DynBody destructor.

Definition at line 85 of file dyn body.cc.

References composite_body, core_body, detach(), dyn_children, dyn_manager, dyn_parent, mass_children, remove_mass_body(), rot_integrator, structure, trans_integrator, and vehicle_points.

8.8.2.3 DynBody() [2/2]

8.8.3 Member Function Documentation

8.8.3.1 activate()

```
void jeod::DynBody::activate ( ) [inline]
```

Activate a DynBody object.

The current implementation does nothing. DynBody objects are always active.

Definition at line 146 of file dyn_body.hh.

8.8.3.2 add_control()

Add a new GravityControls to the list in grav_interaction.

Parameters

in	control	Control to be added

Definition at line 185 of file dyn_body.cc.

References grav_interaction.

8.8.3.3 add_integrable_object()

Add an IntegrableObject to be integrated with this DynBody.

Note that the associated IntegrableObject may or may not follow this DynBody if it is moved to a new integration group/loop.

Parameters

|--|

Definition at line 241 of file dyn_body.cc.

References associated integrable objects.

8.8.3.4 add_mass_body() [1/2]

Definition at line 574 of file dyn_body_attach.cc.

References find_vehicle_point(), jeod::DynBodyMessages::invalid_attachment, mass, and jeod::BodyRefFrame \leftarrow ::mass_point.

8.8.3.5 add_mass_body() [2/2]

Definition at line 681 of file dyn_body_attach.cc.

References add_mass_body_frames(), add_mass_body_validate(), mass, mass_children, and name.

8.8.3.6 add_mass_body_frames()

For a newly attached mass sub-body, create body frames for the root sub-body and all child sub-bodies via recursion.

Returns

Validity indicator

in	subbody	the root of the newly attached sub-bodies	1
----	---------	---	---

Definition at line 752 of file dyn_body_attach.cc.

References dyn_manager, integ_frame, jeod::BodyRefFrame::mass_point, name, and vehicle_points.

Referenced by add_mass_body().

8.8.3.7 add_mass_body_validate()

Validate whether the pending sub body is legal from a mass tree point of view.

Note

Assumptions and Limitations

· The subject mass, child, must not belong to a child body.

Returns

Validity indicator

Parameters

in	child	The child body; the body to be attached to this body.
in	generate_message	Generate message if invalid?

Definition at line 194 of file dyn_body_attach.cc.

References dyn_manager, and name.

Referenced by add_mass_body().

8.8.3.8 add_mass_point()

Add a mass point to the dyn body's list of such and make a vehicle point that corresponds to the added mass point.

in	mass_point_init	Mass point specification
----	-----------------	--------------------------

Definition at line 51 of file dyn_body_vehicle_point.cc.

References dyn_manager, integ_frame, jeod::DynBodyMessages::invalid_body, mass, jeod::BodyRefFrame ← ::mass point, name, and vehicle points.

```
8.8.3.9 attach_child() [1/2]
```

Attach a child DynBody by point specification.

See corresponding DynBody::attach_to() method for more information.

Definition at line 386 of file dyn_body_attach.cc.

References find_vehicle_point(), jeod::DynBodyMessages::invalid_attachment, mass, and jeod::BodyRefFrame ← ::mass_point.

Referenced by attach_to().

```
8.8.3.10 attach_child() [2/2]
```

Attach a child DynBody by location specification.

See corresponding <code>DynBody::attach_to()</code> method for more information. Note that the offset and transformation are specified w.r.t. the parent in both <code>attach_to()</code> and <code>attach_child()</code>

Definition at line 507 of file dyn_body_attach.cc.

References attach_establish_links(), attach_update_properties(), attach_validate_child(), attach_validate_parent(), get root body internal(), mass, and name.

8.8.3.11 attach_establish_links()

Establish the logical connectivity between parent and child.

Extensibility comments -

- This method is invoked before the computing the physical relation between parent and child.
- The generic purpose of this method is to establish the logical connectivity between parent and child in terms of the child class.
- · Any class that overrides this method must either invoke this method or perform the actions performed herein.

Note

Assumptions and Limitations

· The attachment is valid; not checked.

Parameters

	in,out	parent	The new parent body; the body to which this body is to be attached.]
--	--------	--------	---	---

Definition at line 777 of file dyn_body_attach.cc.

References dyn_children, dyn_parent, get_integ_frame(), integ_frame, mass, and set_integ_frame().

Referenced by attach_child().

Attach this dyn body's root body as a child of the specified dyn body such that the specified mass points on the two bodies are coincident and the frames associated with those mass points are related by a 180 degree yaw.

Returns

Success indicator: true=success, false=attachment not performed.

Parameters

in	this_point_name	The name of a mass point contained in this dyn body's list of mass points.
in	parent_point_name	The name of a mass point contained in the parent body's list of mass points.
in,out	parent	The parent body; the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body's root body is to the body to which this body is to the body to which this body is to the body is to the body to which the body to

Definition at line 250 of file dyn_body_attach.cc.

References attach_child().

Attach this dyn body's root body as a child of the specified dyn body such that this body's structural origin is offset from the parent body's structural origin and this body's structural axes are oriented with respect to the parent body's structural axes as specified.

Returns

Success indicator: true=success, false=attachment not performed.

Parameters

in	offset_pstr_cstr_pstr	Location of this body's structural origin with respect to the new parent body's structural origin, specified in structural coordinates of the parent body. Units: M
in	T_pstr_cstr	Transformation matrix from the parent body's structural frame to this body's structural frame.
in,out	parent	The parent body; the body to which this body's root body is to be attached.

Definition at line 267 of file dyn_body_attach.cc.

References attach_child().

Definition at line 272 of file dyn_body_attach.cc.

References dyn_manager, jeod::DynBodyMessages::invalid_attachment, and mass.

```
8.8.3.15 attach_to_frame() [2/4]
```

Definition at line 294 of file dyn_body_attach.cc.

References frame_attach, get_root_body_internal(), jeod::DynBodyGenericFrameAttachment::initialize_← attachment(), and structure.

```
8.8.3.16 attach_to_frame() [3/4]
```

Definition at line 303 of file dyn_body_attach.cc.

References dyn_manager, find_vehicle_point(), frame_attach, get_root_body_internal(), jeod::DynBodyGeneric FrameAttachment::initialize_attachment(), jeod::DynBodyMessages::invalid_attachment, mass, jeod::BodyRef Frame::mass_point, and structure.

```
8.8.3.17 attach_to_frame() [4/4]
```

Definition at line 368 of file dyn_body_attach.cc.

References frame_attach, get_root_body_internal(), and jeod::DynBodyGenericFrameAttachment::initialize_ \leftarrow attachment().

8.8.3.18 attach_update_properties()

Set the relation between parent and child and update the mass properties.

Extensibility comments -

- This method is sent to the parent body of the attachment after the child body has established the logical connectivity between the parent body and child body.
- The generic purpose of this method is to establish the physical relation between parent and child and to update any physical properties that change as a result of the attachment.
- Any class that overrides this method must either invoke this method or perform the actions performed herein.

Note

Assumptions and Limitations

- · The attachment is valid
- · Logical connectivity has been established.

Neither assumption is checked.

Parameters

in	offset_pstr_cstr_pstr	Location of this body's structural origin with respect to the new parent body's structural origin, specified in structural coordinates of the new parent body. Units: m
in	T_pstr_cstr	Transformation matrix from the new parent body's structural frame to this body's structural frame.
in,out	child	The child body; the body newly attached to this body.

Reimplemented in jeod::StructureIntegratedDynBody.

Definition at line 800 of file dyn_body_attach.cc.

References get_dynamics_integration_group(), get_root_body_internal(), initialized_states, mass, process_dynamic_attachment(), propagate_state(), set_state_source_internal(), and structure.

Referenced by attach_child(), and jeod::StructureIntegratedDynBody::attach_update_properties().

8.8.3.19 attach_validate_child()

Validate whether the pending attachment is legal from a physical point of view.

Extensibility comments -

• This method determines whether invoking attach_update_properties makes sense.

Note

Assumptions and Limitations

• The subject body, child, must be a root body. This is not checked.

Returns

Validity indicator

Parameters

in	child	The child body; the body to be attached to this body.
in	generate_message	Generate message if invalid?

Definition at line 112 of file dyn body attach.cc.

References get_root_body(), initialized_states, jeod::DynBodyMessages::invalid_attachment, and name.

Referenced by attach_child().

8.8.3.20 attach_validate_parent()

Validate whether the pending attachment is legal from a connectivity point of view.

Extensibility comments -

- This method determines whether invoking attach_establish_links makes sense.
- · Any class that overrides this method must either invoke this method or perform the actions performed herein.

Note

Assumptions and Limitations:

• The subject body, this, must be a root body. This is not checked.

Returns

Validity indicator

in	parent	The new parent body; the body to which this body is to be attached.
in	generate_message	Generate message if invalid?

Definition at line 57 of file dyn_body_attach.cc.

References dyn_manager, get_root_body(), jeod::DynBodyMessages::invalid_attachment, jeod::DynBody← Messages::invalid_body, name, and jeod::DynBodyMessages::not_dyn_body.

Referenced by attach child().

8.8.3.21 clear integrable objects()

```
void jeod::DynBody::clear_integrable_objects ( )
```

Remove all IntegrableObjects associated with this DynBody.

You might do this if you want to switch the DynBody to a new group without switching the associated Integrable ← Objects.

Definition at line 276 of file dyn_body.cc.

References associated integrable objects.

8.8.3.22 collect_forces_and_torques()

```
void jeod::DynBody::collect_forces_and_torques ( ) [virtual]
```

Collect forces and torques acting on the vehicle.

Reimplemented in jeod::StructureIntegratedDynBody.

Definition at line 92 of file dyn_body_collect.cc.

8.8.3.23 compute_derived_state_forward()

Compute a derived state given the source state and the position/ attitude transformation from the source to the derived state.

in	source_frame	Source state
in	rel_state	Relative state
out	derived_frame	Derived state

Definition at line 159 of file dyn_body_propagate_state.cc.

References jeod::BodyRefFrame::initialized items.

Referenced by compute_vehicle_point_states(), propagate_state_from_composite(), and propagate_state_from __structure().

8.8.3.24 compute_derived_state_reverse()

Compute a derived state given the source state and the position/ attitude transformation from the derived to the source state.

Parameters

in	source_frame	Source state
in	rel_state	Relative state
out	derived_frame	Derived state

Definition at line 257 of file dyn_body_propagate_state.cc.

References jeod::BodyRefFrame::initialized_items.

Referenced by propagate_state_from_composite().

8.8.3.25 compute_ref_point_transform()

Compute the relative state between the integrated frame's mass point and the source frame's mass point.

Note

Assumptions and Limitations

• This method is only called to be called for a root body. This assumption is not enforced.

in	source_frame	The frame that contains the relevant state data.
in,out	ref_point	The mass point corresponding to the previous call to this function. This is an efficiency hack used to avoid duplicative computations.
in,out	rel_state	The relative state between the integration frame mass point and the source frame mass point.

Definition at line 49 of file dyn_body_propagate_state.cc.

References composite_body, integrated_frame, jeod::DynBodyMessages::invalid_frame, mass, jeod::BodyRef← Frame::mass_point, name, and structure.

Referenced by update_integrated_state().

8.8.3.26 compute_state_elements_forward()

Compute selected aspects of the derived state given the source state and the position/ attitude transformation from the source to the derived state.

Parameters

in	source_frame	Source state
in	rel_state	Relative state
in	state_items	States to compute
out	derived_frame	Derived state

Definition at line 201 of file dyn_body_propagate_state.cc.

References jeod::BodyRefFrame::initialized_items.

Referenced by compute_vehicle_point_states(), propagate_state_from_composite(), and propagate_state_from __structure().

8.8.3.27 compute_state_elements_reverse()

Compute selected aspects of the derived state given the source state and the position/ attitude transformation from the derived to the source state.

in	source_frame	Source state
in	rel_state	Relative state
in	state_items	States to compute
out	derived_frame	Derived state

Definition at line 300 of file dyn_body_propagate_state.cc.

References jeod::BodyRefFrame::initialized_items.

Referenced by propagate_state_from_composite().

8.8.3.28 compute_vehicle_point_derivatives()

Compute the state derivatives at a vehicle point.

Parameters

in	vehicle⊷	Vehicle point reference frame
	_pt	
out	pt_derivs	Computed derivatives

 $Reimplemented\ in\ jeod :: Structure Integrated Dyn Body.$

Definition at line 131 of file dyn_body_vehicle_point.cc.

References composite_body, derivs, get_root_body(), grav_interaction, jeod::DynBodyMessages::invalid_frame, mass, jeod::BodyRefFrame::mass_point, name, jeod::FrameDerivs::non_grav_accel, jeod::FrameDerivs::Qdot_ \leftarrow parent_this, jeod::FrameDerivs::rot_accel, and jeod::FrameDerivs::trans_accel.

Referenced by collect_forces_and_torques().

8.8.3.29 compute_vehicle_point_states()

Propagate structure frame state to vehicle points.

Parameters

in	set_items	States truly propagated

Definition at line 728 of file dyn_body_propagate_state.cc.

 $References\ compute_derived_state_forward(),\ compute_state_elements_forward(),\ structure,\ and\ vehicle_points.$

Referenced by propagate_state_from_composite(), and propagate_state_from_structure().

8.8.3.30 create_body_integrators()

Create the integrator (integrators) needed to propagate the translational and rotational state of a DynBody.

Create the translational and rotational integrators for a DynBody.

Parameters

	in	generator	Integrator constructor to be used to create state integrators.
	in	controls	The integration ontrols created the integrator constructor's create_integration_controls method.
Ī	in	time_mngr	The JEOD time manager object.

A DynBody integrates forces and torques in the body frame and forces induced by changes in mass properties.

Parameters

in	generator	Integrator constructor to be used to create state integrators.
in	controls	The integration ontrols created the integrator constructor's create_integration_controls method.
in	time_mngr	The JEOD time manager object.

Definition at line 215 of file dyn_body_integration.cc.

References integ_results_merger, name, rot_integrator, rotation_integration, three_dof, time_manager, and trans
_integrator.

Referenced by create_integrators().

8.8.3.31 create_integrators()

This interface is required by er7_utils::IntegrableObject.

It should not be used. Use DynBody::create_body_integrators instead.

in	generator	Unused.
in	controls	Unused.
in	time_if	Unused.

Definition at line 253 of file dyn_body_integration.cc.

References create_body_integrators(), and jeod::DynBodyMessages::internal_error.

8.8.3.32 deactivate()

```
void jeod::DynBody::deactivate ( ) [inline]
```

Deactivate a DynBody object.

The current implementation does nothing. DynBody objects are always active.

Definition at line 153 of file dyn body.hh.

8.8.3.33 destroy_integrators()

```
void jeod::DynBody::destroy_integrators ( ) [override]
```

Destroy the integrators.

Does nothing, but must be implemented to complete abstract function from the inherited IntegrableObject

Definition at line 277 of file dyn_body_integration.cc.

```
8.8.3.34 detach() [1/2]
```

Detach parent and child DynBodies, 'this' and the argument body, such that the detachment happens at the parent body level.

Returns true if successfully detached the bodies. Returns false if unable to detach. Will fail if, for example, the bodies are not in the same mass tree.

Assumptions and Limitations

The detach point between non-immediate attachments (i.e. not parent/child attachments) takes place at
whichever body is a progenitor. For example, a call to A.detach(D) in an A->B->C->D attachment is
interpreted as a call desiring A // B->C->D. A call to D.detach(B) is interpreted as a call to A->B // C->D.

Returns

Success flag

|--|

Reimplemented in jeod::StructureIntegratedDynBody.

Definition at line 48 of file dyn_body_detach.cc.

References detach_mass_internal(), dyn_children, dyn_parent, jeod::DynBodyMessages::invalid_attachment, mass, and name.

Referenced by \sim DynBody().

```
8.8.3.35 detach() [2/2]
bool jeod::DynBody::detach ( ) [virtual]
```

Detach this DynBody from its parent RefFrame or DynBody parent.

If detaching from a DynBody, evoking this method is the equivalent to the above function via detach(*dyn_parent)

Assumptions and Limitations

· Will inform and return false if the body has no parent.

Returns

Success flag

Definition at line 138 of file dyn body detach.cc.

References jeod::DynBodyGenericFrameAttachment::clear_attachment(), dyn_parent, frame_attach, jeod::Dyn← BodyMessages::invalid_technique, jeod::DynBodyGenericFrameAttachment::isAttached(), and name.

Referenced by jeod::StructureIntegratedDynBody::detach(), remove_mass_body(), and ~DynBody().

8.8.3.36 detach_mass_body_frames()

For a newly detached mass sub-body, remove body frames for the root sub-body and all child sub-bodies via recursion.

Returns

Validity indicator

in	subbody	the root of the newly attached sub-bodies
----	---------	---

Definition at line 237 of file dyn_body_detach.cc.

References dyn_manager, find_body_frame(), and vehicle_points.

Referenced by remove_mass_body().

8.8.3.37 detach_mass_internal()

Update parent and child properties to reflect that they are detached.

Extensibility comments -

- This method is sent to the parent body of the detachment after the child body has severed the logical connectivity between the parent body and child body.
- The generic purpose of this method is to update any physical properties that change as a result of the detachment.
- · Any class that overrides this method must either invoke this method or perform the actions performed herein.

Note

Assumptions and Limitations

• The detachment is valid and logical connectivity has been severed. Neither assumption is checked.

Parameters

```
in, out | child | The child body; the body newly detached from this body.
```

Definition at line 262 of file dyn_body_detach.cc.

References core_body, get_root_body_internal(), mass, propagate_state(), and set_state_source_internal().

Referenced by detach(), and remove_mass_body().

8.8.3.38 find_body_frame()

Find the BodyRefFrame named by the provided identifier.

The name of a BodyRefFrame must be prefixed by the body name. The provided identifier can include or exclude this prefix. The body name is used as the prefix if the the provided name does not start with the body name.

Note

Assumptions and Limitations

- · Limitation: Provided identifier must be non-NULL and non-empty. Failure to comply is a fatal error.
- Limitation: The found frame must be a BodyRefFrame. Finding a non-BodyRefFrame that matches the name is a fatal error.
- Assumption: Failure to find a frame is not an error. The method returns NULL if this is the case.

Returns

Found frame

Parameters

in	frame←	Frame ID suffix
	_id	

Definition at line 47 of file dyn_body_find_body_frame.cc.

References dyn_manager, jeod::DynBodyMessages::invalid_name, and name.

Referenced by detach_mass_body_frames().

8.8.3.39 find_vehicle_point()

Find the vehicle point with the given name.

Returns

Vehicle point

Parameters

in	pt_name	Vehicle point name

Definition at line 98 of file dyn_body_vehicle_point.cc.

References name, and vehicle_points.

Referenced by add_mass_body(), attach_child(), and attach_to_frame().

```
8.8.3.40 get_dynamics_integration_group()
```

```
DynamicsIntegrationGroup * jeod::DynBody::get_dynamics_integration_group ( )
```

Get the DynamicsIntegrationGroup that integrates this DynBody object.

Returns

Pointer to the DynamicsIntegrationGroup of this DynBody.

Definition at line 214 of file dyn_body.cc.

References jeod::DynBodyMessages::internal_error.

Referenced by attach_update_properties(), and set_integ_frame().

8.8.3.41 get_initialized_states()

```
const RefFrameItems& jeod::DynBody::get_initialized_states ( ) const [inline]
```

Indicate which state elements have been initialized.

Returns

Initialized states indicator.

Definition at line 502 of file dyn_body.hh.

8.8.3.42 get_integ_frame()

```
EphemerisRefFrame * jeod::DynBody::get_integ_frame ( ) const [virtual]
```

Get the integration frame for this body.

Returns

Pointer to the integration frame.

Definition at line 58 of file dyn_body_integration.cc.

References integ_frame.

Referenced by attach_establish_links().

```
8.8.3.43 get_integrable_objects()
```

```
JeodPointerVector<er7_utils::IntegrableObject>::type jeod::DynBody::get_integrable_objects ( )
[inline]
```

Get the IntegrableObjects associated with this DynBody.

Returns

A pointer to a JeodPointerVector containing the associated integrable objects.

Definition at line 299 of file dyn_body.hh.

```
8.8.3.44 get_parent_body()
```

```
const DynBody * jeod::DynBody::get_parent_body ( ) const [virtual]
```

Returns this DynBody object's parent body.

Returns

Const pointer to the parent body.

Definition at line 150 of file dyn_body.cc.

Referenced by jeod::StructureIntegratedDynBody::detach().

```
8.8.3.45 get_parent_body_internal()
```

```
DynBody * jeod::DynBody::get_parent_body_internal ( ) [protected], [virtual]
```

Returns this DynBody object's parent body.

Returns

Pointer to parent body.

Definition at line 157 of file dyn_body.cc.

References dyn parent.

8.8.3.46 get_root_body()

```
const DynBody * jeod::DynBody::get_root_body ( ) const [virtual]
```

Finds this DynBody object's root body.

Returns

Const pointer to the root body.

Definition at line 163 of file dyn_body.cc.

Referenced by attach_validate_child(), attach_validate_parent(), jeod::StructureIntegratedDynBody::compute_ wehicle_point_derivatives(), compute_vehicle_point_derivatives(), and set_state_source().

8.8.3.47 get_root_body_internal()

```
DynBody * jeod::DynBody::get_root_body_internal ( ) [protected], [virtual]
```

Finds this DynBody object's root body.

Returns

Pointer to the root body.

Definition at line 170 of file dyn body.cc.

References dyn parent.

Referenced by attach_child(), attach_to_frame(), attach_update_properties(), detach_mass_internal(), set_ \hookleftarrow attitude_left_quaternion(), set_attitude_matrix(), set_attitude_rate(), set_attitude_right_quaternion(), set_position(), set_state(), set_state_source(), set_velocity(), and update_integrated_state().

8.8.3.48 initialize_controls()

Initialize the gravity controls of this DynBody.

Note

Initialization phasing:

The following must have been called prior to calling this method:

- · GravityManager::initialize_model to register the GravityManager object with the dynamics manager.
- GravityManager::add_grav_source to register the pertinent GravitySource objects with the Gravity Manager.
- Planet::register_model to associate the planet with a GravitySource.

in

Definition at line 192 of file dyn_body.cc.

References dyn_manager, and grav_interaction.

8.8.3.49 initialize_model()

Initialize internal and external interrelations, including registration / with the dynamics manager.

Parameters

in,out	dyn_manager <i>←</i>	Dynamics manager
	_in	

Definition at line 43 of file dyn_body_initialize_model.cc.

References composite_body, core_body, dyn_manager, initialized_states, integ_frame, integ_frame_name, jeod::DynBodyMessages::invalid_name, mass, name, set_integ_frame(), and structure.

8.8.3.50 initialized_states_contains()

Indicate whether the specified state elements have been initialized.

Parameters

```
test_items States to test.
```

Returns

True if all test items have been initialized, false otherwise.

Definition at line 512 of file dyn_body.hh.

8.8.3.51 integrate()

Integrate state by the specified dynamic time interval.

Integrate the translational and rotational state and propagate the integrated state to derived states.

Parameters

in	dyn_dt	Dynamic time step, in dynamic time seconds.
in	target_stage	The stage of the integration process that the integrator should try to attain.

Returns

The status (time advance, pass/fail status) of the integration.

Definition at line 305 of file dyn_body_integration.cc.

References frame_attach, jeod::DynBodyGenericFrameAttachment::get_attach_offset(), jeod::DynBodyGeneric FrameAttachment::get_parent_frame(), initialized_states, integ_frame, integ_results_merger, jeod::DynBody GenericFrameAttachment::isAttached(), propagate_state(), rot_integ(), rotational_dynamics, set_state(), structure, trans_integ(), and translational_dynamics.

8.8.3.52 is_root_body()

```
bool jeod::DynBody::is_root_body ( )
```

Indicates whether this DynBody object is a root body.

Returns

Is this a root body?

Definition at line 144 of file dyn body.cc.

References dyn_parent.

8.8.3.53 migrate_integrable_objects()

```
void jeod::DynBody::migrate_integrable_objects ( )
```

Call this method before switching this dyn body to a new group if you want the associated integrable objects to follow.

Definition at line 283 of file dyn_body.cc.

References associated_integrable_objects, jeod::DynBodyMessages::invalid_group, and name.

8.8.3.54 operator=()

8.8.3.55 process_dynamic_attachment()

Process the attachment event of one body from another.

This method is called by the attach method after the links have established or severed and is invoked twice:

- On the parent, in which case the parent argument is null and the child argument is the child that attached from the parent, and
- On the detaching child, in which case the child argument is null and the parent argument is the body from which the child was detached.

Note

Assumptions and Limitations:

- Instances of more derived classes, with presumably more involved dynamics, are situated higher in the
 mass tree than are more basic instances. For example, a simple MassBody can be a child of a DynBody,
 but not the other way around.
- The attachment in the mass tree between the immediate child and the superior body is assumed to reflect a real physical attachment.

Parameters

in	offset_pstr_cstr_pstr	Location of this body's structural origin with respect to the new parent body's structural origin, specified in structural coordinates of the new parent body. Units: m
in	T_pstr_cstr	Transformation matrix from the new parent body's structural frame to this body's structural frame.
in,out	root_body	Body at the root of the mass tree
in, out	child_body	Body that is being attached to this body.

Definition at line 877 of file dyn_body_attach.cc.

References composite_body, core_body, mass, propagate_state(), set_state_source_internal(), and structure.

Referenced by attach_update_properties().

8.8.3.56 propagate_state()

```
void jeod::DynBody::propagate_state ( ) [virtual]
```

Propagate state from the integrated state to attached bodies.

Definition at line 526 of file dyn_body_propagate_state.cc.

References composite_body, dyn_parent, initialized_states, integrated_frame, jeod::DynBodyMessages::invalid_
frame, name, propagate_state(), propagate_state_from_composite(), propagate_state_from_structure(), structure, and update integrated state().

Referenced by attach_update_properties(), detach_mass_internal(), integrate(), process_dynamic_attachment(), propagate state(), and switch integration frames().

8.8.3.57 propagate_state_from_composite()

```
void jeod::DynBody::propagate_state_from_composite ( ) [protected], [virtual]
```

Propagate state to attached bodies starting from this body's composite frame.

Note

Assumptions and Limitations

· At least some states are set.

Definition at line 645 of file dyn_body_propagate_state.cc.

References autoupdate_vehicle_points, composite_body, compute_derived_state_forward(), compute_derived
_state_reverse(), compute_state_elements_forward(), compute_state_elements_reverse(), compute_vehicle_
point_states(), core_body, dyn_children, jeod::BodyRefFrame::initialized_items, mass, and structure.

Referenced by propagate_state().

8.8.3.58 propagate_state_from_structure()

```
void jeod::DynBody::propagate_state_from_structure ( ) [protected], [virtual]
```

Propagate state to attached bodies starting from this body's structural frame.

Note

Assumptions and Limitations

· At least some states are set.

Definition at line 564 of file dyn body propagate state.cc.

References autoupdate_vehicle_points, composite_body, compute_derived_state_forward(), compute_state_ celements_forward(), compute_vehicle_point_states(), core_body, dyn_children, jeod::BodyRefFrame::initialized citems, mass, and structure.

Referenced by propagate_state().

8.8.3.59 remove_integrable_object()

in associated_integrable_object The IntegrableObject to be as

Definition at line 258 of file dyn_body.cc.

References associated_integrable_objects.

8.8.3.60 remove_mass_body()

Remove connectivity between this (parent) DynBody and the argument (child) MassBody mass subbody.

The MassBody and associated body frames are removed, such that the MassBody effectively "jettisons" from dynamics operations.

Extensibility comments -

- This method is invoked before the updating the parent/child states.
- The generic purpose of this method is to sever all connectivity links between parent and child, most importantly mass properties.
- · Any class that overrides this method must either invoke this method or perform the actions performed herein.

Note

Assumptions and Limitations

• The detachment must be valid or it is not performed. The MassBody must not belong to a DynBody-derived dynamic body.

Parameters

in,out	child	The child mass subbody; the body to be detached
--------	-------	---

Definition at line 165 of file dyn_body_detach.cc.

References detach(), detach_mass_body_frames(), detach_mass_internal(), jeod::DynBodyMessages::invalid_content technique, mass, mass_children, and name.

Referenced by \sim DynBody().

8.8.3.61 reset_controls()

```
void jeod::DynBody::reset_controls ( ) [virtual]
```

Make the frame subscriptions for each control consistent with the requirements for that control.

Definition at line 200 of file dyn body.cc.

References dyn_manager, and grav_interaction.

8.8.3.62 reset_integrators()

```
void jeod::DynBody::reset_integrators ( ) [override]
```

Reset the translational and rotational integrators.

Definition at line 285 of file dyn_body_integration.cc.

 $References\ rot_integrator,\ rotational_dynamics,\ trans_integrator,\ and\ translational_dynamics.$

8.8.3.63 rot_integ()

Integrate the vehicle's rotational state.

Integrate the rotational state of a DynBody.

Parameters

in	target stage	The stage of the integration process that the integrator should try to attain.
	151.91.21.31	···· ··· ··· ··· ··· ··· ··· ··· ···

Returns

The status (time advance, pass/fail status) of the integration.

Parameters

in	dyn_dt	Dynamic time step, in dynamic time seconds.	
in	target_stage	The stage of the integration process that the integrator should try to attain.	

Returns

The status (time advance, pass/fail status) of the integration.

Reimplemented in jeod::StructureIntegratedDynBody.

Definition at line 368 of file dyn_body_integration.cc.

References composite_body, derivs, jeod::FrameDerivs::Qdot_parent_this, jeod::FrameDerivs::rot_accel, and rot
_integrator.

Referenced by integrate().

8.8.3.64 set_attitude_left_quaternion()

Set the attitude of the vehicle.

Note

Assumptions and Limitations

· Provided quaternion is a unit quaternion.

Parameters

in	left_quat	Attitude wrt integ frame
out	subject_frame	Frame to update

Definition at line 205 of file dyn_body_set_state.cc.

References jeod::check_frame_ownership(), get_root_body_internal(), and set_state_source_internal().

8.8.3.65 set_attitude_matrix()

Set the attitude of the vehicle.

Note

Assumptions and Limitations

· Provided matrix is orthogonal.

in	matrix	Attitude wrt integ frame
out	subject_frame	Frame to update

Definition at line 233 of file dyn_body_set_state.cc.

References jeod::check_frame_ownership(), get_root_body_internal(), and set_state_source_internal().

8.8.3.66 set attitude rate()

Set the attitude rate of the vehicle.

Note

Assumptions and Limitations

· Provided vector is expressed in body frame coordinates.

Parameters

in	attitude_rate	Attitude wrt integ frame
		Units: r/s
out	subject_frame	Frame to update

Definition at line 247 of file dyn_body_set_state.cc.

References jeod::check_frame_ownership(), get_root_body_internal(), and set_state_source_internal().

8.8.3.67 set_attitude_right_quaternion()

Set the attitude of the vehicle.

Note

Assumptions and Limitations

• Provided quaternion is a unit quaternion.

in	right_quat	Attitude wrt integ frame
out	subject_frame	Frame to update

Definition at line 219 of file dyn_body_set_state.cc.

References jeod::check frame ownership(), get root body internal(), and set state source internal().

Set the integration frame for this body and all its child bodies to the provided frame.

Note

Assumptions and Limitations

• Provided frame is a valid integration frame.

Parameters

in	new_integ_frame	New integration frame
----	-----------------	-----------------------

Definition at line 64 of file dyn_body_integration.cc.

References composite_body, core_body, dyn_children, dyn_manager, get_dynamics_integration_group(), grav_cinteraction, integ_frame, structure, and vehicle_points.

Referenced by attach_establish_links(), initialize_model(), set_integ_frame(), and switch_integration_frames().

Set the integration frame for this body and all its child bodies to the frame indicated by the provided name.

Note

Assumptions and Limitations

- · Assumption: Provided string is a non-NULL, non-empty string.
- · Assumption: State is not to be updated.
- · Limitation: Assocated frame must be a valid integration frame.

in new_integ_frame_name New integration frame

Definition at line 120 of file dyn_body_integration.cc.

References dyn_manager, jeod::DynBodyMessages::invalid_name, name, and set_integ_frame().

8.8.3.70 set_name()

Set the name of the vehicle.

Parameters

in	name⊷	Name of this body
	_in	

Definition at line 138 of file dyn_body.cc.

References mass.

8.8.3.71 set_position()

Set the position of the vehicle.

Parameters

in	position	Position wrt integ frame Units: M
out	subject_frame	Frame to update

Definition at line 179 of file dyn_body_set_state.cc.

References jeod::check_frame_ownership(), get_root_body_internal(), and set_state_source_internal().

8.8.3.72 set_state()

Set the parts of the specified reference frame as indicated by the set_items parameter from the supplied state and propagate these items to all dynamic bodies attached to this body.

This method forms an integral part of the state initialization process and can also be used by a simulation that that receives state overrides from some other simulation.

Note

Assumptions and Limitations

• The subject reference frame is owned by this dynamic body. This limitation is enforced.

Parameters

in	set_items	Items to set
in	state	State to be copied
out	subject_frame	Frame to be set

Definition at line 78 of file dyn_body_set_state.cc.

References jeod::check_frame_ownership(), get_root_body_internal(), and set_state_source_internal().

Referenced by integrate().

8.8.3.73 set_state_source()

Set the source of aspects of the state.

The setting is applied to the root of the DynBody tree.

Note

Assumptions and Limitations

• The supplied frame must either be owned directly by this body or this body must be a root body and the owner of the supplied frame must be a child body of this body.

in	items	Items to propagate
in	frame	Frame containing state

Definition at line 124 of file dyn_body_set_state.cc.

References dyn_parent, get_root_body(), get_root_body_internal(), jeod::DynBodyMessages::invalid_frame, name, and set_state_source_internal().

8.8.3.74 set_state_source_internal()

Set the source of aspects of the state.

Note

Assumptions and Limitations

- · Assumptions, neither of which is checked:
 - This is a root body.
 - The supplied frame is owned by a body that is a child of this body.

Parameters

in	items	Items to propagate
in	frame	Frame containing state

Definition at line 261 of file dyn body set state.cc.

References attitude_source, jeod::BodyRefFrame::initialized_items, initialized_states, position_source, rate $_{\leftarrow}$ source, and velocity_source.

Referenced by attach_update_properties(), detach_mass_internal(), process_dynamic_attachment(), set_ attitude_left_quaternion(), set_attitude_matrix(), set_attitude_rate(), set_attitude_right_quaternion(), set_position(), set_state(), set_state source(), and set_velocity().

8.8.3.75 set_velocity()

Set the velocity of the vehicle.

in	velocity	Velocity wrt integ frame
		Units: M/s
out	subject_frame	Frame to update

Definition at line 192 of file dyn_body_set_state.cc.

References jeod::check_frame_ownership(), get_root_body_internal(), and set_state_source_internal().

8.8.3.76 sort_controls()

```
void jeod::DynBody::sort_controls ( ) [virtual]
```

Sort the gravity controls in ascending acceleration magnitude order.

Definition at line 207 of file dyn_body.cc.

References grav_interaction.

8.8.3.77 switch_integration_frames() [1/2]

Switch the integration frame for this body and all its child bodies to the indicated frame.

Note

Assumptions and Limitations

• Limitation: Assocated frame must be a valid integration frame.

Parameters

in	new integ frame	New integration frame
	_ 5_	

Definition at line 142 of file dyn_body_integration.cc.

References dyn_manager, dyn_parent, integrated_frame, jeod::DynBodyMessages::invalid_frame, name, propagate_state(), set_integ_frame(), switch_integration_frames(), and update_integrated_state().

Referenced by switch_integration_frames().

8.8.3.78 switch_integration_frames() [2/2]

Switch the integration frame for this body and all its child bodies to the frame indicated by the provided name.

Note

Assumptions and Limitations

- · Assumption: Provided string is a non-NULL, non-empty string.
- Limitation: Assocated frame must be a valid integration frame.

Parameters

in <i>new</i>	v_integ_frame_name	New integration frame
---------------	--------------------	-----------------------

Definition at line 184 of file dyn_body_integration.cc.

References dyn_manager, jeod::DynBodyMessages::invalid_name, name, and switch_integration_frames().

8.8.3.79 trans_integ()

Integrate the vehicle's translational state.

Integrate the translational state of a DynBody.

Parameters

in	target_stage	The stage of the integration process that the integrator should try to attain.
----	--------------	--

Returns

The status (time advance, pass/fail status) of the integration.

Parameters

in	dyn_dt	Dynamic time step, in dynamic time seconds.	
in	target_stage	The stage of the integration process that the integrator should try to attain.	

Returns

The status (time advance, pass/fail status) of the integration.

Reimplemented in jeod::StructureIntegratedDynBody.

Definition at line 351 of file dyn_body_integration.cc.

References composite_body, derivs, jeod::FrameDerivs::trans_accel, and trans_integrator.

Referenced by integrate().

8.8.3.80 update_integrated_state()

```
void jeod::DynBody::update_integrated_state ( ) [virtual]
```

Propagate state from state owners to the integrated state.

Definition at line 357 of file dyn_body_propagate_state.cc.

References attitude_source, compute_ref_point_transform(), dyn_parent, get_root_body_internal(), jeod::Body RefFrame::initialized_items, initialized_states, integrated_frame, position_source, rate_source, time_manager, update_integrated_state(), and velocity_source.

Referenced by propagate_state(), switch_integration_frames(), and update_integrated_state().

8.8.4 Friends And Related Function Documentation

8.8.4.1 init_attrjeod__DynBody

```
void init_attrjeod__DynBody ( ) [friend]
```

8.8.4.2 InputProcessor

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

Definition at line 114 of file dyn_body.hh.

8.8.5 Field Documentation

8.8.5.1 associated_integrable_objects

```
std::vector<er7_utils::IntegrableObject *> jeod::DynBody::associated_integrable_objects [protected]
```

List of integrable objects to be integrated with this DynBody.

```
trick_io(**)
```

Definition at line 1158 of file dyn body.hh.

Referenced by add_integrable_object(), clear_integrable_objects(), migrate_integrable_objects(), and remove_\circ integrable_object().

8.8.5.2 attitude_source

```
BodyRefFrame* jeod::DynBody::attitude_source {} [protected]
```

The reference frame that contains the user-set attitude.

trick_units(-)

Definition at line 1141 of file dyn_body.hh.

Referenced by set state source internal(), and update integrated state().

8.8.5.3 autoupdate_vehicle_points

```
bool jeod::DynBody::autoupdate_vehicle_points {true}
```

Are vehicle points automatically updated? The vehicle points are automatically calculated at initialization time but are only automatically updated at runtime if this member is true.

Setting this member to false indicates the responsibility for updating vehicle point states is performed elsewhere, such as in a scheduled call to compute vehicle point states.trick units(-)

Definition at line 714 of file dyn_body.hh.

Referenced by propagate_state_from_composite(), and propagate_state_from_structure().

8.8.5.4 collect

```
BodyForceCollect jeod::DynBody::collect
```

Force/Torque collection mechanism.

trick units(-)

Definition at line 732 of file dyn_body.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect_forces_and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), jeod::StructureIntegratedDynBody::compute_inertial_torque(), jeod::StructureIntegratedDynBody::compute_rotational_acceleration(), jeod::StructureIntegratedDynBody::compute_structureIntegratedDynBody::compute_rotational_acceleration(), jeod::StructureIntegratedDynBody::compute_structu

8.8.5.5 composite_body

```
BodyRefFrame jeod::DynBody::composite_body
```

Vehicle composite body reference frame.

The reference frame origin is at the composite body center of mass, and the reference frame axes are the body frame axes as defined in the composite mass properties.trick_units(-)

Definition at line 649 of file dyn_body.hh.

Referenced by collect_forces_and_torques(), compute_ref_point_transform(), compute_vehicle_point_ \leftarrow derivatives(), DynBody(), initialize_model(), process_dynamic_attachment(), propagate_state(), propagate_state \leftarrow _from_composite(), propagate_state_from_structure(), jeod::StructureIntegratedDynBody::PropagateForcesAnd \leftarrow Torques(), rot_integ(), set_integ_frame(), jeod::StructureIntegratedDynBody::solve_constraints(), trans_integ(), and \sim DynBody().

8.8.5.6 compute_point_derivative

```
bool jeod::DynBody::compute_point_derivative {}
```

Should the point derivatives for the body be computed? A child body's translational and rotational derivatives are only computed if this is true.

If this is false, they will be 0.trick_units(-)

Definition at line 683 of file dyn_body.hh.

Referenced by collect_forces_and_torques().

8.8.5.7 core_body

BodyRefFrame jeod::DynBody::core_body

Vehicle core body reference frame.

The reference frame origin is at the core body center of mass, and the reference frame axes are the body frame axes as defined in the core mass properties.trick units(–)

Definition at line 641 of file dyn body.hh.

Referenced by detach_mass_internal(), DynBody(), initialize_model(), process_dynamic_attachment(), propagate \leftarrow _state_from_composite(), propagate_state_from_structure(), set_integ_frame(), and \sim DynBody().

8.8.5.8 derivs

FrameDerivs jeod::DynBody::derivs

Translational/rotational accelerations.

trick_units(-)

Definition at line 727 of file dyn_body.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect_forces_and_torques(), jeod::StructureIntegratedDynBody::complete_translational_acceleration(), jeod::StructureIntegratedDynBody::compute_rotational_acceleration(), jeod::StructureIntegratedDynBody::compute_translational_acceleration(), compute_vehicle_point_derivatives(), jeod::StructureIntegratedDynBody::rot_integ(), rot_integ(), jeod::StructureIntegratedDynBody::rot_integ(), rot_integ(), jeod::StructureIntegratedDynBody::solve constraints(), and trans integ().

8.8.5.9 dyn_children

```
std::list<DynBody *> jeod::DynBody::dyn_children [protected]
```

The subset of the dynamic bodies attached to this dynamic body.

Definition at line 1109 of file dyn_body.hh.

Referenced by attach_establish_links(), jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect __ forces_and_torques(), detach(), propagate_state_from_composite(), propagate_state_from_structure(), set_ _ integ_frame(), and ~DynBody().

8.8.5.10 dyn_manager

```
BaseDynManager*& jeod::DynBody::dyn_manager [protected]
```

The dynamics manager for the simulation.

trick_units(-)

Definition at line 1083 of file dyn body.hh.

Referenced by add_mass_body_frames(), add_mass_body_validate(), add_mass_point(), attach_to_frame(), attach_validate_parent(), detach_mass_body_frames(), find_body_frame(), initialize_controls(), initialize_model(), reset_controls(), set_integ_frame(), switch_integration_frames(), and \sim DynBody().

8.8.5.11 dyn_parent

```
DynBody* jeod::DynBody::dyn_parent {} [protected]
```

The DynBody to which this body is attached.

This points to exactly the same object as does the links.parent member. While a mass body can be attached to any kind of mass body, a dynamic body can only be attached to another dynamic body.trick_units(–)

Definition at line 1096 of file dyn body.hh.

Referenced by attach_establish_links(), jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect_forces_and_torques(), detach(), get_parent_body_internal(), get_root_body_internal(), is_root_body(), propagate_state(), jeod::StructureIntegratedDynBody::PropagateForcesAndTorques(), set_state_source(), jeod \rightleftharpoons ::StructureIntegratedDynBody::solve_constraints(), switch_integration_frames(), update_integrated_state(), and \sim DynBody().

8.8.5.12 frame_attach

```
DynBodyGenericFrameAttachment jeod::DynBody::frame_attach [protected]
```

The RefFrame this body is attached to.

Once attached, the DynBody will no longer numerically integrate rotational or dynamic states and is considered fixed wrt the RefFrame. The DynBody's integration frame will continue to be used to populate the composite_body, structure, core_body and mass point dynamic states.

Definition at line 1104 of file dyn_body.hh.

Referenced by attach to frame(), detach(), and integrate().

8.8.5.13 grav_interaction

GravityInteraction jeod::DynBody::grav_interaction

Gravitational interactions.

This data member specifies how the vehicle interacts gravitationally with various planetary bodies in the simulation and contains the computed acceleration toward those planetary bodies.trick units(–)

Definition at line 722 of file dyn_body.hh.

Referenced by add_control(), collect_forces_and_torques(), jeod::StructureIntegratedDynBody::complete_ \leftarrow translational_acceleration(), jeod::StructureIntegratedDynBody::compute_vehicle_point_derivatives(), compute_ \leftarrow vehicle_point_derivatives(), initialize_controls(), reset_controls(), set_integ_frame(), and sort_controls().

8.8.5.14 initialized_states

```
RefFrameItems jeod::DynBody::initialized_states {RefFrameItems::No_Items} [protected]
```

Enum value indicating which of position, velocity, attitude, and rate have been initialized.

trick_units(-)

Definition at line 1126 of file dyn_body.hh.

Referenced by attach_update_properties(), attach_validate_child(), initialize_model(), integrate(), propagate_ \leftarrow state(), set_state_source_internal(), and update_integrated_state().

8.8.5.15 integ_frame

```
EphemerisRefFrame* jeod::DynBody::integ_frame {} [protected]
```

The current integration frame.

trick_units(-)

Definition at line 1078 of file dyn_body.hh.

Referenced by add_mass_body_frames(), add_mass_point(), attach_establish_links(), get_integ_frame(), initialize_model(), integrate(), and set_integ_frame().

8.8.5.16 integ_frame_name

```
std::string jeod::DynBody::integ_frame_name
```

The name of the reference frame with respect to which the body's reference frames (core, composite, structure, plus vehicle point frames) are to be represented and propagated.

The value must identify a valid integration frame, i.e., a non-rotating, ephemeris based reference frame.

This member is used at initialization time only. To change the integration frame post-initialization use the function DynBody::switch_integration_frames. This can be invoked directly, or indirectly via a FrameSwitch body action. ← trick units(–)

Definition at line 633 of file dyn body.hh.

Referenced by initialize_model().

8.8.5.17 integ_results_merger

```
er7_utils::IntegratorResultMergerContainer jeod::DynBody::integ_results_merger [protected]
```

The object that merges integration results.

trick_units(-)

Definition at line 1163 of file dyn_body.hh.

Referenced by create_body_integrators(), and integrate().

8.8.5.18 integrated_frame

```
BodyRefFrame* jeod::DynBody::integrated_frame {} [protected]
```

The reference frame whose state is updated via the state integrator.

All other reference frames are calculated from this frame.trick_units(-)

Definition at line 1152 of file dyn_body.hh.

Referenced by compute_ref_point_transform(), DynBody(), propagate_state(), jeod::StructureIntegratedDynBody ::StructureIntegratedDynBody(), switch_integration_frames(), and update_integrated_state().

8.8.5.19 mass

MassBody jeod::DynBody::mass

Mass properties of the vehicle, defined about the structure reference frame.

Definition at line 615 of file dyn_body.hh.

Referenced by add_mass_body(), add_mass_point(), attach_child(), attach_establish_links(), attach_to_ \leftarrow frame(), attach_update_properties(), jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect_ \leftarrow forces_and_torques(), jeod::StructureIntegratedDynBody::complete_translational_acceleration(), jeod::Structure \leftarrow IntegratedDynBody::compute_inertial_torque(), compute_ref_point_transform(), jeod::StructureIntegrated \leftarrow DynBody::compute_rotational_acceleration(), jeod::StructureIntegratedDynBody::compute_translational_ \leftarrow acceleration(), jeod::StructureIntegratedDynBody::compute_vehicle_point_derivatives(), compute_vehicle_point \leftarrow _derivatives(), detach(), detach_mass_internal(), DynBody(), initialize_model(), process_dynamic_attachment(), propagate_state_from_composite(), propagate_state_from_structure(), jeod::StructureIntegratedDynBody:: \leftarrow PropagateForcesAndTorques(), remove_mass_body(), set_name(), and jeod::StructureIntegratedDynBody ::solve_constraints().

8.8.5.20 mass_children

```
std::list<MassBody *> jeod::DynBody::mass_children [protected]
```

The subset of the mass bodies attached to this dynamic body that are themselves not dynamic bodies.

Definition at line 1115 of file dyn body.hh.

Referenced by add_mass_body(), remove_mass_body(), and ~DynBody().

8.8.5.21 name

NamedItem& jeod::DynBody::name

Body name, reference linked to mass.name.

trick_units(-)

Definition at line 620 of file dyn body.hh.

Referenced by add_mass_body(), add_mass_body_frames(), add_mass_body_validate(), add_mass_point(), attach_child(), jeod::StructureIntegratedDynBody::attach_update_properties(), attach_validate_child(), attach validate_properties(), index_validate_properties(), attach_validate_child(), attach validate_properties(), index_validate_properties(), compute_ref_point_transform(), compute_vehicle_properties(), compute_vehicle_properties(), compute_vehicle_properties(), index_vehicle_properties(), index_vehicl

```
8.8.5.22 position_source
```

```
BodyRefFrame* jeod::DynBody::position_source {} [protected]
```

The reference frame that contains the user-set position.

```
trick_units(-)
```

Definition at line 1131 of file dyn_body.hh.

Referenced by set_state_source_internal(), and update_integrated_state().

```
8.8.5.23 rate_source
```

```
BodyRefFrame* jeod::DynBody::rate_source {} [protected]
```

The reference frame that contains the user-set attitude rate.

trick_units(-)

Definition at line 1146 of file dyn_body.hh.

Referenced by set state source internal(), and update integrated state().

8.8.5.24 rot_integrator

```
RestartableSO3SecondOrderODEIntegrator jeod::DynBody::rot_integrator [protected]
```

Rotational state checkpointable/restartable integrator generator.

Rotational state is much harder to integrate. The canonical position is the attitude quaternion, canonical velocity is angular velocity, and the time derivative of the attitude quaternion is a function of the orientiation and the angular velocity.trick_units(-)

Definition at line 1180 of file dyn_body.hh.

Referenced by create_body_integrators(), DynBody(), reset_integrators(), jeod::StructureIntegratedDynBody::rot \leftarrow _integ(), rot_integ(), and \sim DynBody().

8.8.5.25 rotation_integration

GeneralizedSecondOrderODETechnique::TechniqueType jeod::DynBody::rotation_integration

Initial value:

```
{
    GeneralizedSecondOrderODETechnique::LieGroup}
```

Specifies the preferred mechanism for integrating rotational state.

This data member has effect only when set prior to the creation of the body's integrators. The body's rotational integrator will be created based on the value of this data member.trick units(–)

Definition at line 703 of file dyn_body.hh.

Referenced by create body integrators().

8.8.5.26 rotational_dynamics

```
bool jeod::DynBody::rotational_dynamics {}
```

Is rotational dynamics enabled? The body's rotational state is integrated only if this member is true.

Setting this member to false indicates the responsibility for updating the rotational state is performed elsewhere, such as by a user-defined forced rotation model.trick_units(-)

Definition at line 675 of file dyn body.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect_forces_and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), integrate(), jeod::StructureIntegrated \hookleftarrow DynBody::PropagateForcesAndTorques(), reset_integrators(), and jeod::StructureIntegratedDynBody::solve_ \hookleftarrow constraints().

8.8.5.27 structure

```
BodyRefFrame jeod::DynBody::structure
```

Vehicle structural reference frame.

The reference frame origin is at the structural origin, and the reference frame axes are the structure frame axes as defined in the composite mass properties.trick_units(–)

Definition at line 657 of file dyn body.hh.

Referenced by attach_to_frame(), attach_update_properties(), collect_forces_and_torques(), jeod::Structure \leftarrow IntegratedDynBody::complete_translational_acceleration(), jeod::StructureIntegratedDynBody::compute_ \leftarrow inertial_torque(), compute_ref_point_transform(), jeod::StructureIntegratedDynBody::compute_translational_ \leftarrow acceleration(), compute_vehicle_point_states(), DynBody(), initialize_model(), integrate(), process_dynamic_ \leftarrow attachment(), propagate_state(), propagate_state_from_composite(), propagate_state_from_structure(), jeod::StructureIntegratedDynBody::rot_integ(), set_integ_frame(), jeod::StructureIntegratedDynBody::completedDynBody::com

8.8.5.28 three_dof

```
bool jeod::DynBody::three_dof {}
```

Is this a three degrees of freedom (translation only) body? This data member has effect only when set prior to the creation of the body's integrators.

The body's rotational integrator is not created and rotational dynamics is set to false if this member's value is true.

Note that very bad mojo (a core dump) will result if this member is set to true at initialization time and rotational_\circ} dynamics is later enabled during run time.trick_units(-)

Definition at line 695 of file dyn_body.hh.

Referenced by create_body_integrators().

8.8.5.29 time_manager

```
const JeodIntegrationTime* jeod::DynBody::time_manager {} [protected]
```

The time manager to be used to obtain timestamp information.

trick_units(-)

Definition at line 1088 of file dyn_body.hh.

Referenced by create_body_integrators(), and update_integrated_state().

8.8.5.30 trans_integrator

```
RestartableT3SecondOrderODEIntegrator jeod::DynBody::trans_integrator [protected]
```

Translational state checkpointable/restartable integrator generator.

Translational state is comparatively easy to integrate. The canonical position is just position, canonical velocity is just velocity, and the time derivative of position is velocity.trick_units(–)

Definition at line 1171 of file dyn_body.hh.

Referenced by create_body_integrators(), DynBody(), reset_integrators(), jeod::StructureIntegratedDynBody \leftrightarrow ::trans_integ(), trans_integ(), and \sim DynBody().

8.8.5.31 translational_dynamics

```
bool jeod::DynBody::translational_dynamics {}
```

Is translational dynamics enabled? The body's translational state is integrated only if this member is true.

Setting this member to false indicates the responsibility for updating the translational state is performed elsewhere, such as by a user-defined forced translation model.trick units(–)

Definition at line 666 of file dyn_body.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), collect_forces_and_torques(), jeod::StructureIntegratedDynBody::collect_local_forces_and_torques(), integrate(), jeod::StructureIntegrated \hookleftarrow DynBody::PropagateForcesAndTorques(), reset_integrators(), and jeod::StructureIntegratedDynBody::solve_ \hookleftarrow constraints().

8.8.5.32 vehicle_points

```
std::list<BodyRefFrame *> jeod::DynBody::vehicle_points [protected]
```

An array of vehicle points associated with this dynamic body.

Definition at line 1120 of file dyn_body.hh.

Referenced by add_mass_body_frames(), add_mass_point(), compute_vehicle_point_states(), detach_mass_ \leftarrow body_frames(), find_vehicle_point(), set_integ_frame(), and \sim DynBody().

8.8.5.33 velocity_source

```
BodyRefFrame* jeod::DynBody::velocity_source {} [protected]
```

The reference frame that contains the user-set velocity.

trick_units(-)

Definition at line 1136 of file dyn_body.hh.

Referenced by set_state_source_internal(), and update_integrated_state().

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

- · dyn body.hh
- dyn_body.cc
- dyn_body_attach.cc
- dyn_body_collect.cc
- dyn_body_detach.cc
- · dyn_body_find_body_frame.cc
- dyn_body_initialize_model.cc
- dyn_body_integration.cc
- dyn_body_propagate_state.cc
- dyn body set state.cc
- dyn_body_vehicle_point.cc

8.9 jeod::DynBodyGenericFrameAttachment Class Reference

A wrench comprises a torque and a force applied at a point on a DynBody.

```
#include <dyn_body_generic_rigid_attach.hh>
```

Public Member Functions

- DynBodyGenericFrameAttachment ()=default
 - Default constructor.
- void initialize_attachment (RefFrame &parent_frame, const RefFrameState &attach_state)
- void clear attachment ()
- · bool isAttached () const
- RefFrame * get_parent_frame () const
- const RefFrameState & get_attach_offset () const

Private Attributes

```
    bool active {}
        trick_units(-)
    RefFrame * rigid_attach_parent {}
        trick_units(-)
    RefFrameState rigid_attach_state
        trick_units(-)
```

Friends

- · class InputProcessor
- void init_attrjeod__DynBodyGenericFrameAttachment ()

8.9.1 Detailed Description

A wrench comprises a torque and a force applied at a point on a DynBody.

The torque should not include the torque due to the application of the force.

A Trick simulation issues vcollect statements such as

```
vcollect vehicle.dyn_body.collect_wrench.collection
{
    wrench_model1.wrench,
    wrench_model2.wrench
};
```

Definition at line 77 of file dyn_body_generic_rigid_attach.hh.

8.9.2 Constructor & Destructor Documentation

8.9.2.1 DynBodyGenericFrameAttachment()

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

Default constructor.

8.9.3 Member Function Documentation

```
8.9.3.1 clear_attachment()
```

```
void jeod::DynBodyGenericFrameAttachment::clear_attachment ( ) [inline]
```

Definition at line 97 of file dyn_body_generic_rigid_attach.hh.

Referenced by jeod::DynBody::detach().

8.9.3.2 get_attach_offset()

```
const RefFrameState& jeod::DynBodyGenericFrameAttachment::get_attach_offset ( ) const [inline]
```

Definition at line 112 of file dyn_body_generic_rigid_attach.hh.

Referenced by jeod::DynBody::integrate().

8.9.3.3 get_parent_frame()

```
RefFrame* jeod::DynBodyGenericFrameAttachment::get_parent_frame ( ) const [inline]
```

Definition at line 107 of file dyn_body_generic_rigid_attach.hh.

Referenced by jeod::DynBody::integrate().

8.9.3.4 initialize_attachment()

Definition at line 88 of file dyn_body_generic_rigid_attach.hh.

References active, rigid_attach_parent, and rigid_attach_state.

Referenced by jeod::DynBody::attach_to_frame().

8.9.3.5 isAttached()

```
bool jeod::DynBodyGenericFrameAttachment::isAttached ( ) const [inline]
```

Definition at line 102 of file dyn_body_generic_rigid_attach.hh.

Referenced by jeod::DynBody::detach(), and jeod::DynBody::integrate().

8.9.4 Friends And Related Function Documentation

8.9.4.1 init_attrjeod__DynBodyGenericFrameAttachment

```
void init_attrjeod__DynBodyGenericFrameAttachment ( ) [friend]
```

8.9.4.2 InputProcessor

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

Definition at line 79 of file dyn_body_generic_rigid_attach.hh.

8.9.5 Field Documentation

8.9.5.1 active

```
bool jeod::DynBodyGenericFrameAttachment::active {} [private]
```

trick_units(-)

Definition at line 120 of file dyn_body_generic_rigid_attach.hh.

Referenced by initialize_attachment().

8.9.5.2 rigid_attach_parent

```
RefFrame* jeod::DynBodyGenericFrameAttachment::rigid_attach_parent {} [private]
```

trick_units(-)

Definition at line 122 of file dyn_body_generic_rigid_attach.hh.

Referenced by initialize_attachment().

8.9.5.3 rigid_attach_state

```
RefFrameState jeod::DynBodyGenericFrameAttachment::rigid_attach_state [private]
```

trick_units(-)

Definition at line 124 of file dyn_body_generic_rigid_attach.hh.

Referenced by initialize_attachment().

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

· dyn body generic rigid attach.hh

8.10 jeod::DynBodyMessages Class Reference

Specify the message IDs used in the DynBody model.

```
#include <dyn_body_messages.hh>
```

Public Member Functions

- DynBodyMessages ()=delete
- DynBodyMessages (const DynBodyMessages &)=delete
- DynBodyMessages & operator= (const DynBodyMessages &)=delete

Static Public Attributes

- static const char * invalid_body = "dynamics/dyn_body/" "invalid_body"
 Issued when a body is invalid such as not being initialized.
- static const char * invalid_group = "dynamics/dyn_body/" "invalid_group"
 Issued when a group is invalid such as not initialized or NULL.
- static const char * invalid_name = "dynamics/dyn_body/" "invalid_name"
 Issued when a name is invalid NULL, empty, a duplicate, ...
- static const char * invalid_frame = "dynamics/dyn_body/" "invalid_frame"

 Issued when a frame is invalid not an integ frame, ...
- static const char * invalid_attachment = "dynamics/dyn_body/" "invalid_attachment"

 Issued when a attachment is invalid from a state point of view.
- static const char * invalid_technique = "dynamics/dyn_body/" "invalid_technique" Issued when an integration technique is invalid.
- static const char * not_dyn_body = "dynamics/dyn_body/" "not_dyn_body"
 Issued when a MassBody is expected to be a DynBody but that is not the case.
- static const char * internal_error = "dynamics/dyn_body/" "internal_error"
 Error issued when some internal error occurred.

Friends

- class InputProcessor
- void init_attrjeod__DynBodyMessages ()

8.10.1 Detailed Description

Specify the message IDs used in the DynBody model.

Assumptions and Limitations

- This is a complete catalog of all the messages sent by the DynBody model.
- This is not an exhaustive list of all the things that can go awry.

Definition at line 80 of file dyn_body_messages.hh.

8.10.2 Constructor & Destructor Documentation

8.10.3 Member Function Documentation

```
8.10.3.1 operator=()
```

8.10.4 Friends And Related Function Documentation

8.10.4.1 init_attrjeod__DynBodyMessages

```
void init_attrjeod__DynBodyMessages ( ) [friend]
```

8.10.4.2 InputProcessor

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

Definition at line 82 of file dyn_body_messages.hh.

8.10.5 Field Documentation

8.10.5.1 internal_error

```
char const * jeod::DynBodyMessages::internal_error = "dynamics/dyn_body/" "internal_error"
[static]
```

Error issued when some internal error occurred.

These errors should never happen.trick units(-)

Definition at line 125 of file dyn body messages.hh.

Referenced by jeod::DynBody::create_integrators(), and jeod::DynBody::get_dynamics_integration_group().

8.10.5.2 invalid attachment

```
\label{lem:const} char const * jeod::DynBodyMessages::invalid_attachment = "dynamics/dyn_body/" "invalid\_\leftrightarrow attachment" [static]
```

Issued when a attachment is invalid from a state point of view.

trick_units(-)

Definition at line 108 of file dyn_body_messages.hh.

Referenced by jeod::DynBody::add_mass_body(), jeod::DynBody::attach_child(), jeod::DynBody::attach_to_ \leftarrow frame(), jeod::StructureIntegratedDynBody::attach_update_properties(), jeod::DynBody::attach_validate_child(), jeod::DynBody::attach_validate_parent(), jeod::StructureIntegratedDynBody::detach(), and jeod::DynBody \leftarrow ::detach().

8.10.5.3 invalid_body

```
char const * jeod::DynBodyMessages::invalid_body = "dynamics/dyn_body/" "invalid_body" [static]
```

Issued when a body is invalid such as not being initialized.

trick_units(-)

Definition at line 88 of file dyn_body_messages.hh.

Referenced by jeod::StructureIntegratedDynBody::add_constraint(), jeod::DynBody::add_mass_point(), jeod::DynBody::attach_validate_parent(), jeod::StructureIntegratedDynBody::set_solver(), and jeod::Structure \leftarrow IntegratedDynBody::solve_constraints().

8.10.5.4 invalid_frame

```
char const * jeod::DynBodyMessages::invalid_frame = "dynamics/dyn_body/" "invalid_frame" [static]
```

Issued when a frame is invalid - not an integ frame, ...

trick_units(-)

Definition at line 103 of file dyn_body_messages.hh.

Referenced by jeod::check_frame_ownership(), jeod::DynBody::compute_ref_point_transform(), jeod::Structure Untegrated DynBody::compute_vehicle_point_derivatives(), jeod::DynBody::compute_vehicle_point_derivatives(), jeod::DynBody::initialize_model(), jeod::DynBody::propagate_state(), jeod::DynBody::set_state_source(), and jeod::DynBody::switch_integration_frames().

8.10.5.5 invalid_group

```
char const * jeod::DynBodyMessages::invalid_group = "dynamics/dyn_body/" "invalid_group" [static]
```

Issued when a group is invalid such as not initialized or NULL.

trick_units(-)

Definition at line 93 of file dyn_body_messages.hh.

Referenced by jeod::DynBody::migrate_integrable_objects().

8.10.5.6 invalid_name

```
char const * jeod::DynBodyMessages::invalid_name = "dynamics/dyn_body/" "invalid_name" [static]
```

Issued when a name is invalid – NULL, empty, a duplicate, ...

trick_units(-)

Definition at line 98 of file dyn_body_messages.hh.

Referenced by jeod::DynBody::find_body_frame(), jeod::DynBody::initialize_model(), jeod::DynBody::set_integ_ frame(), and jeod::DynBody::switch_integration_frames().

8.10.5.7 invalid_technique

```
char const * jeod::DynBodyMessages::invalid_technique = "dynamics/dyn_body/" "invalid_technique"
[static]
```

Issued when an integration technique is invalid.

trick_units(-)

Definition at line 113 of file dyn_body_messages.hh.

Referenced by jeod::DynBody::detach(), and jeod::DynBody::remove_mass_body().

```
8.10.5.8 not_dyn_body
```

```
char const * jeod::DynBodyMessages::not_dyn_body = "dynamics/dyn_body/" "not_dyn_body" [static]
```

Issued when a MassBody is expected to be a DynBody but that is not the case.

trick_units(-)

Definition at line 119 of file dyn_body_messages.hh.

Referenced by jeod::DynBody::attach validate parent().

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

- dyn_body_messages.hh
- dyn_body_messages.cc

8.11 jeod::Force Class Reference

A Force represents a Newtonian force that acts on a DynBody.

```
#include <force.hh>
```

Public Member Functions

- Force ()=default
- virtual ∼Force ()=default
- Force (const Force &)=delete
- Force & operator= (const Force &)=delete
- double & operator[] (const unsigned int index)

Access a force element, non-const version.

double operator[] (const unsigned int index) const

Access a force element, const version.

Data Fields

· bool active {true}

Is this force active?

double force [3] {}

Force vector.

8.11.1 Detailed Description

A Force represents a Newtonian force that acts on a DynBody.

The class encapsulates an active flag and a 3-vector that contains the force components. Forces are collected in one of a DynBody object's force collection STL vectors. The force vector is expressed in the structural frame of that DynBody object.

The Force class is the recommended mechanism for representing forces in JEOD. While 3-vectors can also be collected into a collect STL vector, theee is is no way to turn off these collected 3-vectors. Even worse, there is no way to tell whether a collected 3-vector does indeed represent a force – or even if it is a 3-vector. In comparison, Force objects can be turned on and off, and more importantly, they are type-safe.

Definition at line 81 of file force.hh.

8.11.2 Constructor & Destructor Documentation

```
8.11.2.1 Force() [1/2]
jeod::Force::Force ( ) [default]
8.11.2.2 \simForce()
virtual jeod::Force::~Force ( ) [virtual], [default]
8.11.2.3 Force() [2/2]
jeod::Force::Force (
            const Force & ) [delete]
8.11.3 Member Function Documentation
8.11.3.1 operator=()
Force& jeod::Force::operator= (
           const Force & ) [delete]
8.11.3.2 operator[]() [1/2]
double & jeod::Force::operator[] (
           const unsigned int index ) [inline]
Access a force element, non-const version.
```

Returns

Force component at specified index Units: N

in	index	Index number
	mack	IIIack Hailibei

Definition at line 73 of file force_inline.hh.

References force.

Access a force element, const version.

Returns

Force component at specified index Units: N

Parameters

- 6			
	in	index	Index number
	T11	IIIUEX	IIIuex IIuiiibei

Definition at line 83 of file force_inline.hh.

References force.

8.11.4 Field Documentation

8.11.4.1 active

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

Is this force active?

trick_units(-)

Definition at line 95 of file force.hh.

8.11.4.2 force

```
double jeod::Force::force[3] {}
```

Force vector.

trick_units(N)

Definition at line 100 of file force.hh.

Referenced by operator[]().

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

- · force.hh
- force_inline.hh

8.12 jeod::FrameDerivs Class Reference

Contains translational and rotational second derivatives.

```
#include <frame_derivs.hh>
```

Public Member Functions

• FrameDerivs ()=default

Data Fields

• double non_grav_accel [3] {}

Non-gravitational acceleration.

• double trans_accel [3] {}

Total acceleration.

Quaternion Qdot_parent_this {0.0}

 $\label{thm:constructor} \textit{Time derivative of Q_parent_this, 0.0 is NOT the same as the default constructor.}$

double rot_accel [3] {}

Total rotational acceleration (expressed in body frame)

8.12.1 Detailed Description

Contains translational and rotational second derivatives.

Definition at line 68 of file frame_derivs.hh.

8.12.2 Constructor & Destructor Documentation

8.12.2.1 FrameDerivs()

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

8.12.3 Field Documentation

8.12.3.1 non_grav_accel

```
double jeod::FrameDerivs::non_grav_accel[3] {}
```

Non-gravitational acceleration.

trick units(m/s2)

Definition at line 76 of file frame_derivs.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces - _and_torques(), jeod::StructureIntegratedDynBody::complete_translational_acceleration(), jeod::Structure - IntegratedDynBody::compute_translational_acceleration(), jeod::StructureIntegratedDynBody::compute_vehicle - _point_derivatives(), jeod::DynBody::compute_vehicle_point_derivatives(), and jeod::StructureIntegratedDyn - Body::solve_constraints().

8.12.3.2 Qdot_parent_this

```
Quaternion jeod::FrameDerivs::Qdot_parent_this {0.0}
```

Time derivative of Q_parent_this, 0.0 is NOT the same as the default constructor.

trick_units(1/s)

Definition at line 86 of file frame derivs.hh.

Referenced by jeod::StructureIntegratedDynBody::compute_vehicle_point_derivatives(), jeod::DynBody::compute_vehicle_point_derivatives(), jeod::StructureIntegratedDynBody::rot_integ(), and jeod::DynBody::rot-integ().

8.12.3.3 rot_accel

```
double jeod::FrameDerivs::rot_accel[3] {}
```

Total rotational acceleration (expressed in body frame)

trick_units(rad/s2)

Definition at line 91 of file frame_derivs.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces - _and_torques(), jeod::StructureIntegratedDynBody::complete_translational_acceleration(), jeod::Structure - _integratedDynBody::compute_rotational_acceleration(), jeod::StructureIntegratedDynBody::compute_vehicle_ - point_derivatives(), jeod::DynBody::rot - _integ(), jeod::DynBody::rot_integ(), and jeod::StructureIntegratedDynBody::solve_constraints().

8.12.3.4 trans_accel

```
double jeod::FrameDerivs::trans_accel[3] {}
```

Total acceleration.

trick units(m/s2)

Definition at line 81 of file frame_derivs.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques(), jeod::DynBody::collect_forces = _and_torques(), jeod::StructureIntegratedDynBody::complete_translational_acceleration(), jeod::Structure = IntegratedDynBody::compute_vehicle_point_derivatives(), jeod::DynBody::compute_vehicle_point_derivatives(), jeod::StructureIntegratedDynBody::trans_integ(), and jeod::DynBody::trans_integ().

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

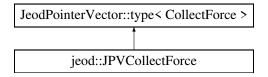
· frame derivs.hh

8.13 jeod::JPVCollectForce Class Reference

This is a derived version of the template class JeodPointerVector<CollectForce>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

```
#include <body_force_collect.hh>
```

Inheritance diagram for jeod::JPVCollectForce:



Public Member Functions

- void perform_insert_action (const std::string &value) override
 Interpret the provided value and add it to the list.
- void push back (CollectForce *const &elem)

Add an element to the end of the contents.

Friends

- template<typename CollectType , typename value_type >
 void collect_insert (CollectType &collect_in, value_type &elem)
- template<typename CollectType , typename value_type > void collect_push_back (CollectType &collect_in, value_type &elem)

8.13.1 Detailed Description

This is a derived version of the template class JeodPointerVector<CollectForce>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

Definition at line 169 of file body_force_collect.hh.

8.13.2 Member Function Documentation

8.13.2.1 perform_insert_action()

Interpret the provided value and add it to the list.

For a JPVCollectForce, the value should specify (in string form) the address of a unique force vector pointer in active memory. If the entry already exists, check and delete the "restored" CollectTorque

Definition at line 184 of file body_force_collect.hh.

References collect insert.

8.13.2.2 push_back()

Add an element to the end of the contents.

Parameters

```
elem Element to be added.
```

Definition at line 206 of file body_force_collect.hh.

References collect_push_back.

8.13.3 Friends And Related Function Documentation

8.13.3.1 collect_insert

Definition at line 92 of file body_force_collect.hh.

Referenced by perform_insert_action().

8.13.3.2 collect_push_back

Definition at line 128 of file body_force_collect.hh.

Referenced by push_back().

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

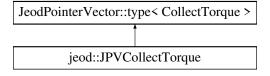
· body_force_collect.hh

8.14 jeod::JPVCollectTorque Class Reference

This is a derived version of the template class JeodPointerVector<CollectTorque>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

```
#include <body_force_collect.hh>
```

Inheritance diagram for jeod::JPVCollectTorque:



Public Member Functions

- void perform_insert_action (const std::string &value) override
 Interpret the provided value and add it to the list.
- void push_back (CollectTorque *const &elem)

Add an element to the end of the contents.

Friends

- template<typename CollectType, typename value_type > void collect_insert (CollectType &collect_in, value_type &elem)
- template<typename CollectType, typename value_type > void collect_push_back (CollectType &collect_in, value_type &elem)

8.14.1 Detailed Description

This is a derived version of the template class JeodPointerVector<CollectTorque>::type with an implementation of the method perform cleanup action which frees and clears stale data following a restore.

Definition at line 217 of file body force collect.hh.

8.14.2 Member Function Documentation

8.14.2.1 perform_insert_action()

Interpret the provided value and add it to the list.

For a JPVCollectTorque, the value should specify (in string form) the address of a unique torque vector pointer in active memory. If the entry already exists, check and delete the "restored" CollectTorque

Definition at line 232 of file body_force_collect.hh.

References collect_insert.

8.14.2.2 push_back()

Add an element to the end of the contents.

Parameters

```
elem Element to be added.
```

Definition at line 254 of file body_force_collect.hh.

References collect_push_back.

8.14.3 Friends And Related Function Documentation

8.14.3.1 collect insert

Definition at line 92 of file body force collect.hh.

Referenced by perform_insert_action().

8.14.3.2 collect_push_back

Definition at line 128 of file body force collect.hh.

Referenced by push_back().

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

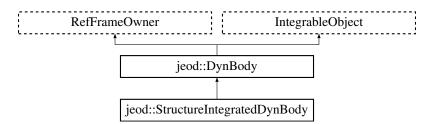
· body_force_collect.hh

8.15 jeod::StructureIntegratedDynBody Class Reference

Extends DynBody to integrate an object's structural reference frame as opposed to its center of mass.

```
#include <structure_integrated_dyn_body.hh>
```

Inheritance diagram for jeod::StructureIntegratedDynBody:



Public Member Functions

- StructureIntegratedDynBody ()
- ~StructureIntegratedDynBody () override=default
- StructureIntegratedDynBody (const StructureIntegratedDynBody &)=delete
- StructureIntegratedDynBody & operator= (const StructureIntegratedDynBody &)=delete
- void collect_forces_and_torques () override

Compute the rotational and translational accelerations that result from the collected forces and torques acting on the vehicle.

void set solver (DynBodyConstraintsSolver &solver in)

Set the solver to be used to solve contraints.

void add_constraint (DynBodyConstraint *constraint)

Add a constraint to the constraints solver.

• virtual void solve_constraints ()

Solve for constraint forces and torques acting on the vehicle and apply them to the vehicle.

- void compute_vehicle_point_derivatives (const BodyRefFrame &frame, FrameDerivs &derivs) override
 Compute the state derivatives at a vehicle point.
- · bool detach (DynBody &other_body) override

Break the logical connectivity between parent and child.

Data Fields

BodyWrenchCollect effector_wrench_collection

Collection of effector wrenches.

Protected Member Functions

void attach_update_properties (const double offset_pstr_cstr_pstr[3], const double T_pstr_cstr[3][3],
 DynBody &child) override

Set the relation between parent and child and update the mass properties.

• const VehicleProperties & get_vehicle_properties () const

Get the vehicle properties as a const reference.

er7_utils::IntegratorResult trans_integ (double dyn_dt, unsigned int target_stage) override

Integrate the translational state of a StructureIntegratedDynBody.

• er7_utils::IntegratorResult rot_integ (double dyn_dt, unsigned int target_stage) override

Integrate the rotational state of a StructureIntegratedDynBody.

• void collect_local_forces_and_torques ()

Collect the local forces and torques that directly act on the vehicle.

• void PropagateForcesAndTorques ()

Propagate forces and torques up the kinematic chain.

void compute_inertial_torque ()

Compute the inertial torque.

• void compute rotational acceleration ()

Compute the body- and structure-referenced rotational acceleration.

void compute_translational_acceleration ()

Compute the inertial-referenced translational acceleration vector.

• void complete_translational_acceleration ()

Finalize computation of the inertial-referenced translational acceleration vector.

Protected Attributes

DynBodyConstraintsSolver * constraints_solver {}

The solver for constraint forces and torques, if there are any.

· Wrench effector_wrench

Wrench into which the effector wrenches are accumulated.

FrameDerivs struct_derivs

Translational/rotational accelerations of the structural frame.

· VehicleProperties vehicle_properties

Various properties of the vehicle, for the constraints solver.

VehicleNonGravState non_grav_state

Rotational and translational behaviors, for the constraints solver.

double inertial accel struct omega [3] {}

Structure-referenced inertial acceleration at the structure frame origin due to vehicle angular velocity.

double inertial_accel_struct_omega_dot [3] {}

Structure-referenced inertial acceleration at the structure frame origin due to vehicle angular acceleration.

• double inertial_accel_struct [3] {}

Structure-referenced inertial acceleration at the structure frame origin.

double inertial_accel_inrtl [3] {}

Inertial-referenced inertial acceleration at the structure frame origin.

Friends

- class InputProcessor
- · class DynBodyConstraintsSolver
- void init_attrjeod__StructureIntegratedDynBody ()

8.15.1 Detailed Description

Extends DynBody to integrate an object's structural reference frame as opposed to its center of mass.

In addition to structure integration, this class introduces two new concepts, wrenches and constrained objects. A wrench encapsulates a force applied at a point and a torque, with the torque induced by the force due to an off-centerline force direction automatically calculated by JEOD. A constrained object is an object that lies outside the DynBody system boundary that exchanges translational and/or rotational momentum with the DynBody and that is somehow constrained by the translation and/or rotational behavior of the DynBody.

These new concepts might be migrated up the DynBody inheritance chain in subsequent releases of JEOD.

Definition at line 88 of file structure_integrated_dyn_body.hh.

8.15.2 Constructor & Destructor Documentation

8.15.2.1 StructureIntegratedDynBody() [1/2]

```
\verb|jeod::StructureIntegratedDynBody::StructureIntegratedDynBody ( )\\
```

Definition at line 36 of file structure_integrated_dyn_body.cc.

References jeod::DynBody::integrated frame, and jeod::DynBody::structure.

8.15.2.2 ~StructureIntegratedDynBody()

```
jeod::StructureIntegratedDynBody::~StructureIntegratedDynBody ( ) [override], [default]
```

8.15.2.3 StructureIntegratedDynBody() [2/2]

8.15.3 Member Function Documentation

8.15.3.1 add_constraint()

Add a constraint to the constraints solver.

Note

Both the constraint and the solver must be non-null.

Parameters

oolyor	aint to be added to the sol	t Th	constraint	
	aint to be added to the	t∣ Th	constraint	

Definition at line 112 of file structure_integrated_dyn_body_solve.cc.

References constraints_solver, and jeod::DynBodyMessages::invalid_body.

8.15.3.2 attach_update_properties()

Set the relation between parent and child and update the mass properties.

Parameters

in	offset_pstr_cstr_pstr	Location of the child body's structural origin with respect to the parent body's structural origin, specified in structural coordinates of the parent body.
in	T_pstr_cstr	Transformation matrix from the parent body's structural frame to the child body's structural frame.
in,out	child	The child body being attached to this body.

Reimplemented from jeod::DynBody.

Definition at line 34 of file structure integrated dyn body solve.cc.

References jeod::DynBody::attach_update_properties(), constraints_solver, jeod::DynBodyMessages::invalid_ \leftarrow attachment, and jeod::DynBody::name.

8.15.3.3 collect_forces_and_torques()

```
void jeod::StructureIntegratedDynBody::collect_forces_and_torques ( ) [override], [virtual]
```

Compute the rotational and translational accelerations that result from the collected forces and torques acting on the vehicle.

This function should be called as a derivative class job, with a moderately high phase number. Functions that calculate the gravitational acceleration and the effector, environmental, and non-transmitted forces and torques should be called as scheduled jobs or as lower phase derivative class jobs.

Reimplemented from jeod::DynBody.

Definition at line 69 of file structure integrated dyn body collect.cc.

References jeod::DynBody::collect, collect_local_forces_and_torques(), compute_inertial_torque(), compute containing collect::effector_sortional_acceleration(), compute_translational_acceleration(), jeod::DynBody::dyn_parent, jeod::BodyForceCollect::effector_forc, jeod::BodyForceCollect::effector_torq, effector_wrench, jeod::BodyForceCollect::environ_forc, jeod::BodyForceCollect::environ_torq, jeod::BodyForceCollect::extern_forc_inrtl, jeod::BodyForceCollect::extern_forc_struct, jeod::BodyForceCollect::extern_torq_body, jeod::BodyForceCollect::extern_torq_struct, jeod::Wrench::get_force(), jeod::Wrench::get_torque(), jeod::BodyForceCollect::no_xmit_forc, jeod::BodyForceCollectci:no_xmit_torq, jeod::FrameDerivs::no_grav_accel, PropagateForcesAndTorques(), jeod::FrameDerivs::rot_accel, jeod::DynBody::rotational_dynamics, struct_derivs, jeod::FrameDerivs::trans_accel, jeod::Wrench::transform_toc_point(), and jeod::DynBody::translational_dynamics.

8.15.3.4 collect_local_forces_and_torques()

```
void jeod::StructureIntegratedDynBody::collect_local_forces_and_torques ( ) [protected]
```

Collect the local forces and torques that directly act on the vehicle.

Definition at line 160 of file structure integrated dyn body collect.cc.

References jeod::BodyWrenchCollect::accumulate(), jeod::accumulate_forces(), jeod::accumulate_torques(), jeod::DynBody::collect, jeod::BodyForceCollect::collect_effector_forc, jeod::BodyForceCollect::collect_effector - __torq, jeod::BodyForceCollect::collect_environ_forc, jeod::BodyForceCollect::collect_environ_torq, jeod::BodyForceCollect::collect_no_xmit_torq, jeod::BodyForceCollect:- jeod:- jeod:-

Referenced by collect_forces_and_torques().

8.15.3.5 complete_translational_acceleration()

```
void jeod::StructureIntegratedDynBody::complete_translational_acceleration ( ) [protected]
```

Finalize computation of the inertial-referenced translational acceleration vector.

Definition at line 353 of file structure integrated dyn body collect.cc.

References jeod::DynBody::derivs, jeod::DynBody::grav_interaction, inertial_accel_inrtl, inertial_accel_struct, inertial_accel_struct_omega, inertial_accel_struct_omega_dot, jeod::DynBody::mass, jeod::FrameDerivs::non_comparav_accel, jeod::FrameDerivs::rot_accel, struct_derivs, jeod::DynBody::structure, and jeod::FrameDerivs::transcomparaccel.

Referenced by compute_translational_acceleration(), and solve_constraints().

8.15.3.6 compute_inertial_torque()

```
void jeod::StructureIntegratedDynBody::compute_inertial_torque ( ) [protected]
```

Compute the inertial torque.

Definition at line 292 of file structure_integrated_dyn_body_collect.cc.

References jeod::DynBody::collect, jeod::BodyForceCollect::inertial_torq, jeod::DynBody::mass, and jeod::Dyn \leftarrow Body::structure.

Referenced by collect_forces_and_torques().

8.15.3.7 compute_rotational_acceleration()

```
\verb|void jeod::StructureIntegratedDynBody::compute\_rotational\_acceleration ( ) | [protected]| \\
```

Compute the body- and structure-referenced rotational acceleration.

Definition at line 309 of file structure integrated dyn body collect.cc.

References jeod::DynBody::collect, jeod::DynBody::derivs, jeod::BodyForceCollect::extern_torq_body, jeod:: \leftarrow BodyForceCollect::extern_torq_struct, jeod::BodyForceCollect::inertial_torq, jeod::DynBody::mass, jeod::Frame \leftarrow Derivs::rot accel, and struct derivs.

Referenced by collect forces and torques().

8.15.3.8 compute_translational_acceleration()

```
void jeod::StructureIntegratedDynBody::compute_translational_acceleration ( ) [protected]
```

Compute the inertial-referenced translational acceleration vector.

Definition at line 331 of file structure_integrated_dyn_body_collect.cc.

References jeod::DynBody::collect, complete_translational_acceleration(), jeod::DynBody::derivs, jeod::Body ForceCollect::extern_forc_inrtl, jeod::BodyForceCollect::extern_forc_struct, inertial_accel_struct_omega, jeod::

DynBody::mass, jeod::FrameDerivs::non_grav_accel, and jeod::DynBody::structure.

Referenced by collect_forces_and_torques().

8.15.3.9 compute_vehicle_point_derivatives()

Compute the state derivatives at a vehicle point.

Parameters

frame	The vehicle point, as a BodyRefFrame, at which derivatives are to be calculated.
derivs	The calculated derivatives.

Reimplemented from jeod::DynBody.

Definition at line 31 of file structure_integrated_dyn_body_pt_accel.cc.

References jeod::DynBody::get_root_body(), jeod::DynBody::grav_interaction, jeod::DynBodyMessages::invalid - _ frame, jeod::DynBody::mass, jeod::BodyRefFrame::mass_point, jeod::FrameDerivs::non_grav_accel, jeod:: FrameDerivs::Qdot parent this, jeod::FrameDerivs::rot accel, and jeod::FrameDerivs::trans accel.

8.15.3.10 detach()

Break the logical connectivity between parent and child.

Parameters

in,out	other_body	The other body to detach from
--------	------------	-------------------------------

Reimplemented from jeod::DynBody.

Definition at line 61 of file structure_integrated_dyn_body_solve.cc.

References constraints_solver, detach(), jeod::DynBody::detach(), jeod::DynBody::get_parent_body(), jeod::DynBody::DynBody::name, and vehicle_properties.

Referenced by detach().

8.15.3.11 get_vehicle_properties()

```
const VehicleProperties& jeod::StructureIntegratedDynBody::get_vehicle_properties ( ) const
[inline], [protected]
```

Get the vehicle properties as a const reference.

Definition at line 243 of file structure_integrated_dyn_body.hh.

8.15.3.12 operator=()

8.15.3.13 PropagateForcesAndTorques()

```
void jeod::StructureIntegratedDynBody::PropagateForcesAndTorques ( ) [protected]
```

Propagate forces and torques up the kinematic chain.

Definition at line 207 of file structure_integrated_dyn_body_collect.cc.

References jeod::DynBody::collect, jeod::DynBody::composite_body, jeod::DynBody::dyn_parent, jeod::Body ForceCollect::effector_torq, effector_wrench, jeod::BodyForceCollect ::environ_forc, jeod::BodyForceCollect::environ_torq, jeod::DynBody::mass, jeod::DynBody::rotational_dynamics, jeod::DynBody::structure, jeod::Wrench::transform_to_parent(), and jeod::DynBody::translational_dynamics.

Referenced by collect_forces_and_torques().

```
8.15.3.14 rot_integ()
```

Integrate the rotational state of a StructureIntegratedDynBody.

Parameters

in	dyn_dt	Dynamic time step, in dynamic time seconds.
in	target_stage	The stage of the integration process that the integrator should try to attain.

Returns

The status (time advance, pass/fail status) of the integration.

Reimplemented from jeod::DynBody.

Definition at line 47 of file structure_integrated_dyn_body_integration.cc.

References jeod::DynBody::derivs, jeod::FrameDerivs::Qdot_parent_this, jeod::FrameDerivs::rot_accel, jeod::

DynBody::rot_integrator, struct_derivs, and jeod::DynBody::structure.

8.15.3.15 set_solver()

Set the solver to be used to solve contraints.

Definition at line 97 of file structure_integrated_dyn_body_solve.cc.

References constraints_solver, jeod::DynBodyMessages::invalid_body, and jeod::DynBody::name.

8.15.3.16 solve_constraints()

```
void jeod::StructureIntegratedDynBody::solve_constraints ( ) [virtual]
```

Solve for constraint forces and torques acting on the vehicle and apply them to the vehicle.

This function should be called as a derivative class job, with a very high phase number. Functions that calculate the constraints should be called as derivative class jobs with a phase intermediate between that of collect_forces — and_torques and of this function.

Definition at line 127 of file structure_integrated_dyn_body_solve.cc.

References jeod::VehicleNonGravState::accel_struct, jeod::DynBody::collect, complete_translational_acceleration(), jeod::DynBody::composite_body, constraints_solver, jeod::DynBody::derivs, jeod::DynBody::dyn_parent, jeod::

BodyForceCollect::extern_forc_struct, jeod::BodyForceCollect::inertial_torq, jeod::VehicleNonGravState::inertial _torque_struct, jeod::DynBodyMessages::invalid_body, jeod::DynBody::mass, jeod::FrameDerivs::non_grav_accel, non_grav_state, jeod::VehicleNonGravState::omega_body, jeod::VehicleNonGravState::omega_dot_body, jeod::VehicleNonGravState::omega_dot_body, jeod::VehicleNonGravState::omega_struct, jeod::FrameDerivs::rot_-collectional-dynamics, struct_derivs, jeod::DynBody::structure, jeod::DynBody::translational-collectional-dynamics, and vehicle_properties.

8.15.3.17 trans_integ()

Integrate the translational state of a StructureIntegratedDynBody.

Parameters

in	dyn_dt	Dynamic time step, in dynamic time seconds.
in	target_stage	The stage of the integration process that the integrator should try to attain.

Returns

The status (time advance, pass/fail status) of the integration.

Reimplemented from jeod::DynBody.

Definition at line 36 of file structure_integrated_dyn_body_integration.cc.

References struct_derivs, jeod::DynBody::structure, jeod::FrameDerivs::trans_accel, and jeod::DynBody::trans_ \leftarrow integrator.

8.15.4 Friends And Related Function Documentation

8.15.4.1 DynBodyConstraintsSolver

friend class DynBodyConstraintsSolver [friend]

Definition at line 90 of file structure_integrated_dyn_body.hh.

8.15.4.2 init_attrjeod__StructureIntegratedDynBody

void init_attrjeod__StructureIntegratedDynBody () [friend]

8.15.4.3 InputProcessor

friend class InputProcessor [friend]

Definition at line 90 of file structure_integrated_dyn_body.hh.

8.15.5 Field Documentation

8.15.5.1 constraints_solver

```
DynBodyConstraintsSolver* jeod::StructureIntegratedDynBody::constraints_solver {} [protected]
```

The solver for constraint forces and torques, if there are any.

This needs to be assigned prior to initialization time in simulations that invoke member function solve_constraints() during runtime. This can be left unassigned (null) in simulations that do not have vehicular constraints.trick_units(–)

Definition at line 179 of file structure integrated dyn body.hh.

Referenced by add constraint(), attach update properties(), detach(), set solver(), and solve constraints().

8.15.5.2 effector_wrench

```
Wrench jeod::StructureIntegratedDynBody::effector_wrench [protected]
```

Wrench into which the effector wrenches are accumulated.

```
trick_units(-)
```

Definition at line 184 of file structure_integrated_dyn_body.hh.

Referenced by collect_forces_and_torques(), collect_local_forces_and_torques(), and PropagateForcesAnd

Torques().

8.15.5.3 effector_wrench_collection

```
BodyWrenchCollect jeod::StructureIntegratedDynBody::effector_wrench_collection
```

Collection of effector wrenches.

The effector wrenches are assembled into the collection at the S_define level via

The collected effector wrenches are processed by the collect_forces_and_torques member function.

Note: For completion, there probably should be collected environmental and non-transmitted wrenches as well as effector wrenches.trick_units(-)

Definition at line 112 of file structure_integrated_dyn_body.hh.

Referenced by collect_local_forces_and_torques().

```
8.15.5.4 inertial_accel_inrtl
double jeod::StructureIntegratedDynBody::inertial_accel_inrtl[3] {} [protected]
Inertial-referenced inertial acceleration at the structure frame origin.
trick_units(m/s2)
Definition at line 221 of file structure_integrated_dyn_body.hh.
Referenced by complete_translational_acceleration().
8.15.5.5 inertial_accel_struct
double jeod::StructureIntegratedDynBody::inertial_accel_struct[3] {} [protected]
Structure-referenced inertial acceleration at the structure frame origin.
trick units(m/s2)
Definition at line 216 of file structure_integrated_dyn_body.hh.
Referenced by complete translational acceleration().
8.15.5.6 inertial_accel_struct_omega
double jeod::StructureIntegratedDynBody::inertial_accel_struct_omega[3] {} [protected]
Structure-referenced inertial acceleration at the structure frame origin due to vehicle angular velocity.
trick_units(m/s2)
Definition at line 205 of file structure_integrated_dyn_body.hh.
Referenced by complete_translational_acceleration(), and compute_translational_acceleration().
8.15.5.7 inertial_accel_struct_omega_dot
double jeod::StructureIntegratedDynBody::inertial_accel_struct_omega_dot[3] {} [protected]
Structure-referenced inertial acceleration at the structure frame origin due to vehicle angular acceleration.
```

trick_units(m/s2)

Definition at line 211 of file structure_integrated_dyn_body.hh.

Referenced by complete_translational_acceleration().

8.15.5.8 non_grav_state

VehicleNonGravState jeod::StructureIntegratedDynBody::non_grav_state [protected]

Rotational and translational behaviors, for the constraints solver.

trick_units(-)

Definition at line 199 of file structure_integrated_dyn_body.hh.

Referenced by solve constraints().

8.15.5.9 struct_derivs

FrameDerivs jeod::StructureIntegratedDynBody::struct_derivs [protected]

Translational/rotational accelerations of the structural frame.

trick units(-)

Definition at line 189 of file structure_integrated_dyn_body.hh.

Referenced by collect_forces_and_torques(), complete_translational_acceleration(), compute_rotational_\circ} acceleration(), rot_integ(), solve_constraints(), and trans_integ().

8.15.5.10 vehicle_properties

VehicleProperties jeod::StructureIntegratedDynBody::vehicle_properties [protected]

Various properties of the vehicle, for the constraints solver.

trick_units(-)

Definition at line 194 of file structure_integrated_dyn_body.hh.

Referenced by detach(), and solve_constraints().

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

- structure_integrated_dyn_body.hh
- structure_integrated_dyn_body.cc
- structure_integrated_dyn_body_collect.cc
- structure_integrated_dyn_body_integration.cc
- structure_integrated_dyn_body_pt_accel.cc
- structure_integrated_dyn_body_solve.cc

8.16 jeod::Torque Class Reference

A Torque represents a Newtonian torque that acts on a DynBody.

```
#include <torque.hh>
```

Public Member Functions

- Torque ()=default
- virtual ~Torque ()=default
- Torque (const Torque &)=delete
- Torque & operator= (const Torque &)=delete
- double & operator[] (const unsigned int index)

Access a torque element, non-const version.

• double operator[] (const unsigned int index) const

Access a torque element, const version.

Data Fields

· bool active {true}

Is this torque active?

double torque [3] {}

Torque vector.

8.16.1 Detailed Description

A Torque represents a Newtonian torque that acts on a DynBody.

The class encapsulates an active flag and a 3-vector that contains the torque components. Torques are collected in one of a DynBody object's torque collection STL vectors. The torque vector is expressed in the structural frame of that DynBody object.

The Torque class is the recommended mechanism for representing torques in JEOD. While 3-vectors can also be collected into a collect STL vector, theee is is no way to turn off these collected 3-vectors. Even worse, there is no way to tell whether a collected 3-vector does indeed represent a torque, or even if it is a 3-vector. In comparison, Torque objects can be turned on and off, and more importantly, they are type-safe.

Definition at line 81 of file torque.hh.

8.16.2 Constructor & Destructor Documentation

```
8.16.2.1 Torque() [1/2] jeod::Torque::Torque ( ) [default]
```

```
8.16.2.2 \simTorque()
```

```
virtual jeod::Torque::~Torque ( ) [virtual], [default]

8.16.2.3 Torque() [2/2]
```

const Torque &) [delete]

8.16.3 Member Function Documentation

8.16.3.1 operator=()

jeod::Torque::Torque (

8.16.3.2 operator[]() [1/2]

Access a torque element, non-const version.

Returns

Torque component at specified index

Units: NM

Parameters

	in	index	Index number
--	----	-------	--------------

Definition at line 73 of file torque_inline.hh.

References torque.

8.16.3.3 operator[]() [2/2]

Access a torque element, const version.

Returns

Torque component at specified index

Units: NM

Parameters

in index Index numb	er
---------------------	----

Definition at line 83 of file torque_inline.hh.

References torque.

8.16.4 Field Documentation

8.16.4.1 active

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

Is this torque active?

trick_units(-)

Definition at line 95 of file torque.hh.

8.16.4.2 torque

```
double jeod::Torque::torque[3] {}
```

Torque vector.

trick_units(N*m)

Definition at line 99 of file torque.hh.

Referenced by operator[]().

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

- torque.hh
- torque_inline.hh

8.17 jeod::VehicleNonGravState Class Reference

Encapsulates various aspects of a vehicle's state with respect to inertial.

```
#include <vehicle_non_grav_state.hh>
```

Data Fields

• double omega_body [3]

Vehicle angular velocity with respect to inertial, in root body body frame coordinates.

• double omega_struct [3]

Vehicle angular velocity with respect to inertial, in root body structural frame coordinates.

· double omega_dot_body [3]

Vehicle angular acceleration with respect to inertial, in root body body frame coordinates.

• double omega_dot_struct [3]

Vehicle angular acceleration with respect to inertial, in root body structural frame coordinates.

• double inertial_torque_struct [3]

Vehicle inertial torque (w x lw) in root body structural coordinates.

• double accel_struct [3]

Vehicle non-gravitational translational acceleration at the center of mass, in root body structural frame coordinates.

Friends

- · class InputProcessor
- void init_attrjeod__VehicleNonGravState ()

8.17.1 Detailed Description

Encapsulates various aspects of a vehicle's state with respect to inertial.

Definition at line 65 of file vehicle_non_grav_state.hh.

8.17.2 Friends And Related Function Documentation

8.17.2.1 init_attrjeod__VehicleNonGravState

```
void init_attrjeod__VehicleNonGravState ( ) [friend]
```

8.17.2.2 InputProcessor

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

Definition at line 67 of file vehicle_non_grav_state.hh.

8.17.3 Field Documentation

```
8.17.3.1 accel_struct
double jeod::VehicleNonGravState::accel_struct[3]
Vehicle non-gravitational translational acceleration at the center of mass, in root body structural frame coordinates.
trick_units(m/s^2)
Definition at line 101 of file vehicle_non_grav_state.hh.
Referenced by jeod::StructureIntegratedDynBody::solve_constraints().
8.17.3.2 inertial_torque_struct
double jeod::VehicleNonGravState::inertial_torque_struct[3]
Vehicle inertial torque (w x lw) in root body structural coordinates.
trick units(N*m)
Definition at line 95 of file vehicle_non_grav_state.hh.
Referenced by jeod::StructureIntegratedDynBody::solve_constraints().
8.17.3.3 omega_body
double jeod::VehicleNonGravState::omega_body[3]
Vehicle angular velocity with respect to inertial, in root body body frame coordinates.
trick units(1/s)
```

Definition at line 72 of file vehicle_non_grav_state.hh.

Referenced by jeod::StructureIntegratedDynBody::solve constraints().

8.17.3.4 omega_dot_body

```
double jeod::VehicleNonGravState::omega_dot_body[3]
```

Vehicle angular acceleration with respect to inertial, in root body body frame coordinates.

```
trick units(1/s<sup>2</sup>)
```

Definition at line 84 of file vehicle_non_grav_state.hh.

Referenced by jeod::StructureIntegratedDynBody::solve_constraints().

8.17.3.5 omega_dot_struct

```
double jeod::VehicleNonGravState::omega_dot_struct[3]
```

Vehicle angular acceleration with respect to inertial, in root body structural frame coordinates.

```
trick_units(1/s^2)
```

Definition at line 90 of file vehicle_non_grav_state.hh.

Referenced by jeod::StructureIntegratedDynBody::solve_constraints().

8.17.3.6 omega_struct

```
double jeod::VehicleNonGravState::omega_struct[3]
```

Vehicle angular velocity with respect to inertial, in root body structural frame coordinates.

```
trick_units(1/s)
```

Definition at line 78 of file vehicle non grav state.hh.

Referenced by jeod::StructureIntegratedDynBody::solve_constraints().

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

• vehicle_non_grav_state.hh

8.18 jeod::VehicleProperties Class Reference

Captures pointers to various vehicle properties that are commonly used in the constraint concept.

```
#include <vehicle_properties.hh>
```

Public Member Functions

VehicleProperties ()=default

Default constructor, for use by Trick only.

VehicleProperties (SolverTypes::Vector3RefT parent_to_structure_offset_in, SolverTypes::Matrix3x3RefT parent_to_structure_transform_in, double &mass_in, SolverTypes::Vector3RefT structure_to_body_offset
 _in, SolverTypes::Matrix3x3RefT inertia_in, SolverTypes::Matrix3x3RefT structure_to_body_transform_in, double &inverse_mass_in, SolverTypes::Matrix3x3RefT inverse_inertia_in)

Non-default constructor that sets all elements.

- SolverTypes::ConstDecayedVector3T get_parent_to_structure_offset () const
- SolverTypes::ConstMatrix3x3RefT get parent to structure transform () const
- double get_mass () const
- SolverTypes::ConstDecayedVector3T get_structure_to_body_offset () const
- SolverTypes::ConstMatrix3x3RefT get_inertia () const
- SolverTypes::Matrix3x3RefT get_structure_to_body_transform () const
- · double get_inverse_mass () const
- SolverTypes::Matrix3x3RefT get_inverse_inertia () const

Private Attributes

SolverTypes::Vector3PointerT parent_to_structure_offset {}

Pointer to the vehicle's structure_point.position vector.

SolverTypes::Matrix3x3PointerT parent_to_structure_transform {}

Pointer to the vehicle's structure_point.T_parent_this matrix.

double * mass {}

Pointer to the vehicle's composite_properties.mass member.

SolverTypes::Vector3PointerT structure_to_body_offset {}

Pointer to the vehicle's composite_properties.position vector.

SolverTypes::Matrix3x3PointerT inertia {}

Pointer to the vehicle's composite_properties.inertia tensor.

SolverTypes::Matrix3x3PointerT structure_to_body_transform {}

Pointer to the vehicle's composite_properties.T_parent_this matrix.

double * inverse mass {}

Pointer to the vehicle's inverse_mass member.

SolverTypes::Matrix3x3PointerT inverse_inertia {}

Pointer to the vehicle's inverse_inertia member.

Friends

- · class InputProcessor
- void init_attrjeod__VehicleProperties ()

8.18.1 Detailed Description

Captures pointers to various vehicle properties that are commonly used in the constraint concept.

As this is potentially quite dangerous, access to the captured members is limited to const getters.

This class is not designed for extensibility.

Definition at line 71 of file vehicle_properties.hh.

8.18.2 Constructor & Destructor Documentation

8.18.2.1 VehicleProperties() [1/2]

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

Default constructor, for use by Trick only.

8.18.2.2 VehicleProperties() [2/2]

Non-default constructor that sets all elements.

Parameters

parent_to_structure_offset_in	Reference to the vehicle's structure_point.position vector.
parent_to_structure_transform⊷	Reference to the vehicle's structure_point.T_parent_this matrix.
_in	
mass_in	Reference to the vehicle's composite_properties.mass member.
structure_to_body_offset_in	Reference to the vehicle's composite_properties.position vector.
inertia_in	Reference to the vehicle's composite_properties.inertia tensor.
structure_to_body_transform_in	Reference to the vehicle's composite_properties.T_parent_this matrix.
inverse_mass_in	Reference to the vehicle's inverse_mass member.
inverse_inertia_in	Reference to the vehicle's inverse_inertia member.

Definition at line 103 of file vehicle_properties.hh.

8.18.3 Member Function Documentation

8.18.3.1 get_inertia()

```
SolverTypes::ConstMatrix3x3RefT jeod::VehicleProperties::get_inertia ( ) const [inline]
```

Returns

Const reference to the vehicle's inertia tensor, in vehicle body frame coordinates.

Definition at line 169 of file vehicle_properties.hh.

8.18.3.2 get_inverse_inertia()

```
SolverTypes::Matrix3x3RefT jeod::VehicleProperties::get_inverse_inertia ( ) const [inline]
```

Returns

Const reference to the inverse of the vehicle's inertia tensor, in vehicle body frame coordinates.

Definition at line 195 of file vehicle_properties.hh.

8.18.3.3 get_inverse_mass()

```
double jeod::VehicleProperties::get_inverse_mass ( ) const [inline]
```

Returns

The multiplicative inverse of the vehicle's mass.

Definition at line 186 of file vehicle_properties.hh.

8.18.3.4 get_mass()

```
double jeod::VehicleProperties::get_mass ( ) const [inline]
```

Returns

The vehicle mass.

Definition at line 150 of file vehicle_properties.hh.

8.18.3.5 get_parent_to_structure_offset()

```
SolverTypes::ConstDecayedVector3T jeod::VehicleProperties::get_parent_to_structure_offset ( )
const [inline]
```

Returns

Const reference to the offset from the parent vehicle's structural frame origin to this vehicle's structural origin, in parent structural coordinates.

Definition at line 133 of file vehicle_properties.hh.

8.18.3.6 get_parent_to_structure_transform()

```
SolverTypes::ConstMatrix3x3RefT\ jeod::VehicleProperties::get\_parent\_to\_structure\_transform\ (\ ) \\ const\ [inline]
```

Returns

Const reference to the transformation matrix from the parent vehicle's structural frame to this vehicle's structural frame

Definition at line 142 of file vehicle_properties.hh.

8.18.3.7 get_structure_to_body_offset()

```
SolverTypes::ConstDecayedVector3T jeod::VehicleProperties::get_structure_to_body_offset ( ) const [inline]
```

Returns

Const reference to the offset from the origin of the vehicle's structural frame to the vehicle's center of mass, in vehicle structural coordinates.

Definition at line 160 of file vehicle properties.hh.

8.18.3.8 get_structure_to_body_transform()

```
SolverTypes::Matrix3x3RefT jeod::VehicleProperties::get_structure_to_body_transform ( ) const [inline]
```

Returns

Const reference to the transformation matrix from the vehicle's structural frame to its body frame.

Definition at line 178 of file vehicle_properties.hh.

8.18.4 Friends And Related Function Documentation

8.18.4.1 init_attrjeod__VehicleProperties void init_attrjeod__VehicleProperties () [friend] 8.18.4.2 InputProcessor friend class InputProcessor [friend]

8.18.5 Field Documentation

Definition at line 76 of file vehicle_properties.hh.

8.18.5.1 inertia

```
SolverTypes::Matrix3x3PointerT jeod::VehicleProperties::inertia {} [private]
```

Pointer to the vehicle's composite_properties.inertia tensor.

trick_units(m^2*kg)

Definition at line 224 of file vehicle_properties.hh.

8.18.5.2 inverse_inertia

```
SolverTypes::Matrix3x3PointerT jeod::VehicleProperties::inverse_inertia {} [private]
```

Pointer to the vehicle's inverse_inertia member.

 $trick_units(1/kg/m^{\wedge}2)$

Definition at line 239 of file vehicle_properties.hh.

trick_units(-)

Definition at line 209 of file vehicle_properties.hh.

```
8.18.5.3 inverse_mass
double* jeod::VehicleProperties::inverse_mass {} [private]
Pointer to the vehicle's inverse_mass member.
trick_units(1/kg)
Definition at line 234 of file vehicle_properties.hh.
8.18.5.4 mass
double* jeod::VehicleProperties::mass {} [private]
Pointer to the vehicle's composite_properties.mass member.
trick_units(kg)
Definition at line 214 of file vehicle_properties.hh.
8.18.5.5 parent_to_structure_offset
SolverTypes::Vector3PointerT jeod::VehicleProperties::parent_to_structure_offset {} [private]
Pointer to the vehicle's structure_point.position vector.
trick_units(m)
Definition at line 204 of file vehicle_properties.hh.
8.18.5.6 parent_to_structure_transform
SolverTypes::Matrix3x3PointerT jeod::VehicleProperties::parent_to_structure_transform {} [private]
Pointer to the vehicle's structure_point.T_parent_this matrix.
```

8.18.5.7 structure_to_body_offset

```
SolverTypes::Vector3PointerT jeod::VehicleProperties::structure_to_body_offset {} [private]
```

Pointer to the vehicle's composite_properties.position vector.

trick_units(m)

Definition at line 219 of file vehicle_properties.hh.

8.18.5.8 structure_to_body_transform

```
SolverTypes::Matrix3x3PointerT jeod::VehicleProperties::structure_to_body_transform {} [private]
```

Pointer to the vehicle's composite_properties.T_parent_this matrix.

trick_units(-)

Definition at line 229 of file vehicle_properties.hh.

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

• vehicle_properties.hh

8.19 jeod::Wrench Class Reference

A wrench comprises a torque and a force applied at a point on a DynBody.

```
#include <wrench.hh>
```

Public Member Functions

Wrench (bool active_in=true)

Default constructor.

• Wrench (const double torque_in[3], const double force_in[3], const double point_in[3], bool active_in=true)

Non-default constructor that sets all elements of the wrench.

• Wrench (const double point_in[3], bool active_in=true)

Non-default constructor that sets the point and active flag.

- virtual ∼Wrench ()=default
- Wrench (const Wrench &)=default
- Wrench & operator= (const Wrench &)=default
- Wrench (Wrench &&)=default
- Wrench & operator= (Wrench &&)=default
- Wrench & operator+= (const Wrench & other)

Increment this wrench by the other, but only if both are active.

· void activate ()

Mark this wrench as active.

void deactivate ()

Mark this wrench as inactive.

• bool is_active () const

Is this wrench active?

· void reset force and torque ()

Set the force and torque to zero.

void reset_torque ()

Set the torque to zero.

void reset force ()

Set the force to zero.

· void reset_point ()

Set the point to zero.

• void set (const double torque in[3], const double force in[3], const double point in[3])

Set all vector elements of the wrench.

• void set_torque (const double torque_in[3])

Set the torque to the specified value.

void set force (const double force in[3])

Set the force to the specified value.

• void set_force (const double force_in[3], const double point_in[3])

Set the force and the point of application to the specified values.

void set point (const double point in[3])

Set the point of application to the specified value.

· void scale_torque (double scale)

Scale the torque by the specified value.

void scale force (double scale)

Scale the force by the specified value.

const double * get_torque () const

Const getter of the torque vector.

const double * get_force () const

Const getter of the force vector.

const double * get_point () const

Const getter of the point vector.

Wrench & accumulate (const std::vector< Wrench *> &collection)

Accumulate the wrenches in the collection to form a combined wrench about the current wrench point, which remains unchanged.

Wrench & accumulate (const std::vector< Wrench *> &collection, const double new_point[3])

Accumulate the wrenches in the collection to form a combined wrench about the specified wrench point.

• Wrench transform_to_point (const double new_point[3]) const

Construct an equivalent Wrench about the specified point.

Wrench transform_to_parent (const MassPointState &point_state) const

Construct an equivalent Wrench about the current point, but in a different reference frame.

Private Attributes

• double torque [3] {}

The torque exerted on the <code>DynBody</code> by the force/torque agent, expressed in structural coordinates.

double force [3] {}

The force exerted on the DynBody by the force/torque agent, expressed in structural coordinates.

double point [3] {}

The structural coordinates of the point at which the force is applied.

· bool active

Indicated whether the wrench is active (true) or inactive (false).

Friends

- class InputProcessor
- void init_attrjeod__Wrench ()

8.19.1 Detailed Description

A wrench comprises a torque and a force applied at a point on a DynBody.

The torque should not include the torque due to the application of the force.

A Trick simulation issues vcollect statements such as

```
vcollect vehicle.dyn_body.collect_wrench.collection
{
    wrench_model1.wrench,
    wrench_model2.wrench
};
```

Definition at line 79 of file wrench.hh.

8.19.2 Constructor & Destructor Documentation

Default constructor.

The wrench is marked as active, and the torque, force, and point vectors are all initialized to zero. This constructor can also be used as a non-default constructor that marks the wrench as inactive by calling it with one argument (a boolean) whose value is false.

Parameters

```
active

_in

True (default) indicates the wrench is active.
```

Definition at line 92 of file wrench.hh.

```
const double force_in[3],
const double point_in[3],
bool active_in = true ) [inline], [explicit]
```

Non-default constructor that sets all elements of the wrench.

Parameters

torque← _in	The intrinsic torque for this wrench.
force_in	The force applied at the point.
point_in	The point at which forces are applied.
active_in	True (default) indicates the wrench is active.

Definition at line 104 of file wrench.hh.

Non-default constructor that sets the point and active flag.

The torque and force and initialized to zero.

Parameters

point_in	The point at which forces are applied.
active⊷	True (default) indicates the wrench is active.
in	

Definition at line 121 of file wrench.hh.

```
8.19.2.4 \simWrench()
```

```
\label{lem:virtual} \mbox{virtual} \ \mbox{jeod::} \mbox{$\mathbb{W}$rench::} \sim \mbox{$\mathbb{W}$rench} \ \ \mbox{()} \ \ \mbox{[virtual], [default]}
```

```
8.19.2.5 Wrench() [4/5]
```

8.19.3 Member Function Documentation

Accumulate the wrenches in the collection to form a combined wrench about the current wrench point, which remains unchanged.

Parameters

collection The wrenches to be acc	umulated.
-----------------------------------	-----------

Definition at line 311 of file wrench.hh.

Referenced by jeod::BodyWrenchCollect::accumulate().

Accumulate the wrenches in the collection to form a combined wrench about the specified wrench point.

Parameters

collection	The wrenches to be accumulated.
new_point	The point about which the wrenches to be accumulated.

Definition at line 327 of file wrench.hh.

8.19.3.3 activate()

```
void jeod::Wrench::activate ( ) [inline]
```

Mark this wrench as active.

Definition at line 160 of file wrench.hh.

```
8.19.3.4 deactivate()
```

```
void jeod::Wrench::deactivate ( ) [inline]
```

Mark this wrench as inactive.

Definition at line 168 of file wrench.hh.

```
8.19.3.5 get_force()
```

```
const double* jeod::Wrench::get_force ( ) const [inline]
```

Const getter of the force vector.

Definition at line 293 of file wrench.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques().

```
8.19.3.6 get_point()
```

```
const double* jeod::Wrench::get_point ( ) const [inline]
```

Const getter of the point vector.

Definition at line 301 of file wrench.hh.

8.19.3.7 get_torque()

```
const double* jeod::Wrench::get_torque ( ) const [inline]
```

Const getter of the torque vector.

Definition at line 285 of file wrench.hh.

Referenced by jeod::StructureIntegratedDynBody::collect_forces_and_torques().

8.19.3.8 is_active()

```
bool jeod::Wrench::is_active ( ) const [inline]
```

Is this wrench active?

Definition at line 176 of file wrench.hh.

8.19.3.9 operator+=()

Increment this wrench by the other, but only if both are active.

The other wrench is effectively reseated to this wrench's point prior to incrementing.

Parameters

other Wrench with which this wrench is to be incremented.

Returns

*this.

Definition at line 143 of file wrench.hh.

8.19.3.12 reset_force()

```
void jeod::Wrench::reset_force ( ) [inline]
```

Set the force to zero.

The torque and point remain unaltered.

Definition at line 201 of file wrench.hh.

```
8.19.3.13 reset_force_and_torque()
void jeod::Wrench::reset_force_and_torque ( ) [inline]
```

Set the force and torque to zero.

The point remains unaltered.

Definition at line 184 of file wrench.hh.

 $Referenced \ by \ jeod::StructureIntegratedDynBody::collect_local_forces_and_torques().$

```
8.19.3.14 reset_point()
```

```
void jeod::Wrench::reset_point ( ) [inline]
```

Set the point to zero.

The torque and force remain unaltered.

Definition at line 209 of file wrench.hh.

8.19.3.15 reset_torque()

```
void jeod::Wrench::reset_torque ( ) [inline]
```

Set the torque to zero.

The force and point remain unaltered.

Definition at line 193 of file wrench.hh.

8.19.3.16 scale_force()

Scale the force by the specified value.

The torque and point of application remain unchanged.

Definition at line 277 of file wrench.hh.

8.19.3.17 scale_torque()

Scale the torque by the specified value.

The force and point of application remain unaltered.

Definition at line 268 of file wrench.hh.

8.19.3.18 set()

Set all vector elements of the wrench.

Parameters

torque← _in	The intrinsic torque for this wrench.
force_in	The force applied at the point.
point_in	The point at which forces are applied.

Definition at line 220 of file wrench.hh.

Set the force to the specified value.

The torque and point of application remain unchanged.

Definition at line 240 of file wrench.hh.

Set the force and the point of application to the specified values.

The torque remain unchanged.

Definition at line 249 of file wrench.hh.

Set the point of application to the specified value.

The force and torque remain unchanged.

Definition at line 259 of file wrench.hh.

Referenced by jeod::BodyWrenchCollect::accumulate().

8.19.3.22 set_torque()

Set the torque to the specified value.

The force and point of application remain unaltered.

Definition at line 231 of file wrench.hh.

8.19.3.23 transform_to_parent()

Construct an equivalent Wrench about the current point, but in a different reference frame.

Parameters

point_state | Contains the position and orientation of the current frame in the parent frame.

Returns

Equivalent wrench in the parent frame.

Definition at line 354 of file wrench.hh.

 $Referenced\ by\ jeod::StructureIntegratedDynBody::PropagateForcesAndTorques().$

8.19.3.24 transform_to_point()

Construct an equivalent Wrench about the specified point.

Parameters

new_point The point about which this is to be represented	d.
---	----

Returns

Equivalent wrench about the specified point.

Definition at line 338 of file wrench.hh.

 $Referenced\ by\ jeod::StructureIntegratedDynBody::collect_forces_and_torques().$

8.19.4 Friends And Related Function Documentation

8.19.4.1 init_attrjeod__Wrench

```
void init_attrjeod__Wrench ( ) [friend]
```

8.19.4.2 InputProcessor

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

Definition at line 81 of file wrench.hh.

8.19.5 Field Documentation

8.19.5.1 active

```
bool jeod::Wrench::active [private]
```

Indicated whether the wrench is active (true) or inactive (false).

inactive wrenches are not collected.trick_units(-)

Definition at line 393 of file wrench.hh.

8.19.5.2 force

```
double jeod::Wrench::force[3] {} [private]
```

The force exerted on the DynBody by the force/torque agent, expressed in structural coordinates.

trick_units(N)

Definition at line 382 of file wrench.hh.

8.19.5.3 point

```
double jeod::Wrench::point[3] {} [private]
```

The structural coordinates of the point at which the force is applied.

trick units(m)

Definition at line 387 of file wrench.hh.

8.19.5.4 torque

```
double jeod::Wrench::torque[3] {} [private]
```

The torque exerted on the DynBody by the force/torque agent, expressed in structural coordinates.

This torque should not include the torque that results from the force not passing through the center of mass. A typical thruster, for example, should have the torque set to zero. On the other hand, a Hall effect thruster will have a non-zero torque due to the swirling of the exhaust.trick_units(N*m)

Definition at line 376 of file wrench.hh.

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

• wrench.hh

Chapter 9

File Documentation

9.1 body_force_collect.cc File Reference

Define BodyForceCollect member functions.

```
#include "../include/body_force_collect.hh"
```

Namespaces

jeod

Namespace jeod.

9.1.1 Detailed Description

Define BodyForceCollect member functions.

9.2 body_force_collect.hh File Reference

Define the class BodyForceCollect.

```
#include "utils/container/include/pointer_vector.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "force.hh"
#include "torque.hh"
```

Data Structures

· class jeod::JPVCollectForce

This is a derived version of the template class JeodPointerVector< CollectForce>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

• class jeod::JPVCollectTorque

This is a derived version of the template class JeodPointerVector<CollectTorque>::type with an implementation of the method perform_cleanup_action which frees and clears stale data following a restore.

class jeod::BodyForceCollect

Serves as the collection point for forces and torques that act on a vehicle.

Namespaces

· jeod

Namespace jeod.

Functions

```
    template < class CollectType >
        void jeod::release_vector (CollectType &vec)
```

Release JEOD-allocated memory in the collect vector.

- template<typename CollectType, typename value_type >
 void jeod::collect_insert (CollectType &collect_in, value_type &elem)
- template<typename CollectType, typename value_type > void jeod::collect_push_back (CollectType &collect_in, value_type &elem)

9.2.1 Detailed Description

Define the class BodyForceCollect.

9.3 body_ref_frame.hh File Reference

Define the class BodyRefFrame.

```
#include <cstddef>
#include "dynamics/mass/include/class_declarations.hh"
#include "utils/ref_frames/include/ref_frame.hh"
#include "utils/ref_frames/include/ref_frame_items.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

• class jeod::BodyRefFrame

Extend RefFrame to add coupling between the reference frame tree and the mass tree and to keep track of which state items have been set.

Namespaces

• jeod

Namespace jeod.

9.3.1 Detailed Description

Define the class BodyRefFrame.

9.4 body_wrench_collect.cc File Reference

Define BodyWrenchCollect member functions.

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

Namespaces

• jeod

Namespace jeod.

9.4.1 Detailed Description

Define BodyWrenchCollect member functions.

9.5 body_wrench_collect.hh File Reference

Defines the class BodyWrenchCollect.

```
#include "wrench.hh"
#include "utils/container/include/pointer_vector.hh"
```

Data Structures

• class jeod::BodyWrenchCollect

Serves as the collection point for wrenches that act on a vehicle.

Namespaces

• jeod

Namespace jeod.

9.5.1 Detailed Description

Defines the class BodyWrenchCollect.

9.6 class_declarations.hh File Reference

Forward declarations of classes defined in dyn_body.hh.

Namespaces

jeod

Namespace jeod.

9.6.1 Detailed Description

Forward declarations of classes defined in dyn_body.hh.

9.7 dyn_body.cc File Reference

Define base methods for the DynBody class.

```
#include <algorithm>
#include <cstddef>
#include "dynamics/dyn_manager/include/dyn_manager.hh"
#include "dynamics/dyn_manager/include/dynamics_integration_group.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

· ieod

Namespace jeod.

9.7.1 Detailed Description

Define base methods for the DynBody class.

9.8 dyn_body.hh File Reference

Define the class DynBody.

```
#include <list>
#include <vector>
#include "body_force_collect.hh"
#include "body_ref_frame.hh"
#include "dyn_body_generic_rigid_attach.hh"
#include "frame_derivs.hh"
#include "dynamics/mass/include/mass.hh"
#include "environment/gravity/include/gravity_interaction.hh"
#include "utils/container/include/simple_checkpointable.hh"
#include "utils/integration/include/generalized_second_order_ode_technique.\leftarrow
#include "utils/integration/include/restartable state integrator.hh"
#include "utils/ref frames/include/ref frame interface.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include "er7_utils/integration/core/include/integrable_object.hh"
#include "er7_utils/integration/core/include/integrator_result.hh"
\verb|#include "er7_utils/integration/core/include/integrator_result_merger\_{\leftarrow}
container.hh"
```

Data Structures

· class jeod::DynBody

Class DynBody is the base class for all dynamic bodies.

Namespaces

jeod

Namespace jeod.

9.8.1 Detailed Description

Define the class DynBody.

9.9 dyn_body_attach.cc File Reference

Define DynBody attachment methods.

```
#include <cstddef>
#include <list>
#include <string>
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "dynamics/mass/include/mass.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message handler.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../../derived_state/include/relative_derived_state.hh"
#include "../../dyn_manager/include/dynamics_integration_group.hh"
#include "../include/body_ref_frame.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
#include "environment/ephemerides/ephem_interface/include/ephem_ref_frame. ←
hh"
```

Namespaces

jeod

Namespace jeod.

9.9.1 Detailed Description

Define DynBody attachment methods.

9.10 dyn_body_collect.cc File Reference

Define DynBody methods related to force and torque accumulation and propagation.

```
#include <cstddef>
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "../include/dyn_body.hh"
```

Namespaces

jeod

Namespace jeod.

Functions

static void jeod::accumulate_forces (const JeodPointerVector< CollectForce >::type &vec, double *cumulation)

Accumulate forces acting on a vehicle.

static void jeod::accumulate_torques (const JeodPointerVector< CollectTorque >::type &vec, double *cumulation)

Accumulate torques acting on a vehicle.

9.10.1 Detailed Description

Define DynBody methods related to force and torque accumulation and propagation.

9.11 dyn_body_detach.cc File Reference

Define DynBody detachment methods.

```
#include <algorithm>
#include <cstddef>
#include "dynamics/dyn_manager/include/dyn_manager.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/ref_frames/include/tree_links_iterator.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

jeod

Namespace jeod.

9.11.1 Detailed Description

Define DynBody detachment methods.

9.12 dyn_body_find_body_frame.cc File Reference

Define DynBody::find_body_frame.

```
#include <cstddef>
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

jeod

Namespace jeod.

9.12.1 Detailed Description

Define DynBody::find_body_frame.

9.13 dyn_body_generic_rigid_attach.hh File Reference

Define the class Wrench.

```
#include "../../mass/include/mass_point_state.hh"
#include "body_ref_frame.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

· class jeod::DynBodyGenericFrameAttachment

A wrench comprises a torque and a force applied at a point on a DynBody.

Namespaces

• jeod

Namespace jeod.

9.13.1 Detailed Description

Define the class Wrench.

9.14 dyn_body_initialize_model.cc File Reference

Define DynBody::initialize model.

```
#include <cstddef>
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

jeod

Namespace jeod.

9.14.1 Detailed Description

Define DynBody::initialize_model.

9.15 dyn_body_integration.cc File Reference

Define methods for frame switching.

```
#include <cstddef>
\#include "er7_utils/integration/core/include/second_order_ode_integrator.\leftarrow
hh"
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "dynamics/dyn_manager/include/dynamics_integration_group.hh"
#include "environment/ephemerides/ephem_interface/include/ephem_ref_frame. ←
hh"
#include "utils/integration/include/generalized_second_order_ode_technique. ←
#include "utils/integration/include/jeod_integration_time.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/named_item/include/named_item.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

· jeod

Namespace jeod.

9.15.1 Detailed Description

Define methods for frame switching.

9.16 dyn_body_messages.cc File Reference

Implement the class De4xxMessages.

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

Namespaces

jeod

Namespace jeod.

Macros

• #define MAKE_DYNBODY_MESSAGE_CODE(id) JEOD_MAKE_MESSAGE_CODE(DynBodyMessages, "dynamics/dyn_body/", id)

9.16.1 Detailed Description

Implement the class De4xxMessages.

9.16.2 Macro Definition Documentation

9.16.2.1 MAKE_DYNBODY_MESSAGE_CODE

```
\label{lem:make_dynbody_message_code} $$id$ ) $$ JEOD_MAKE_MESSAGE_CODE (DynBodyMessages, "dynamics/dyn_body/", id) $$
```

Definition at line 43 of file dyn_body_messages.cc.

9.17 dyn_body_messages.hh File Reference

Define the class DynBodyMessages.

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

Data Structures

• class jeod::DynBodyMessages

Specify the message IDs used in the DynBody model.

Namespaces

· jeod

Namespace jeod.

9.17.1 Detailed Description

Define the class DynBodyMessages.

9.18 dyn_body_propagate_state.cc File Reference

Define DynBody state propagation / update methods.

```
#include <cstddef>
#include "utils/integration/include/jeod_integration_time.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

· jeod

Namespace jeod.

9.18.1 Detailed Description

Define DynBody state propagation / update methods.

9.19 dyn_body_set_state.cc File Reference

Define methods related to setting aspects of a vehicle's state.

```
#include <cstddef>
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/ref_frames/include/ref_frame_items.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

ieod

Namespace jeod.

Functions

• static void jeod::check_frame_ownership (const BodyRefFrame &frame, const DynBody *dyn_body, const char *file, unsigned int line)

Check that the dyn_body 'owns' the subject frame.

9.19.1 Detailed Description

Define methods related to setting aspects of a vehicle's state.

9.20 dyn_body_vehicle_point.cc File Reference

Define methods that support vehicle points.

```
#include <cstddef>
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "environment/ephemerides/ephem_interface/include/ephem_ref_frame.
hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/named_item/include/named_item.hh"
#include "utils/quaternion/include/quat.hh"
#include "../include/dyn_body.hh"
#include "../include/dyn_body_messages.hh"
```

Namespaces

jeod

Namespace jeod.

9.20.1 Detailed Description

Define methods that support vehicle points.

9.21 force.cc File Reference

Define force model member functions.

```
#include <cstddef>
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/force.hh"
```

Namespaces

• jeod

Namespace jeod.

9.21.1 Detailed Description

Define force model member functions.

9.22 force.hh File Reference

Define the JEOD force model.

```
#include "force_inline.hh"
```

Data Structures

· class jeod::Force

A Force represents a Newtonian force that acts on a DynBody.

• class jeod::CollectForce

A CollectForce represents a collected force that acts on a vehicle.

· class jeod::CInterfaceForce

This class is deprecated.

Namespaces

• jeod

Namespace jeod.

9.22.1 Detailed Description

Define the JEOD force model.

9.23 force_inline.hh File Reference

Inline functions for the JEOD force model.

```
#include "force.hh"
#include <cstddef>
```

Namespaces

• jeod

Namespace jeod.

9.23.1 Detailed Description

Inline functions for the JEOD force model.

9.24 frame_derivs.hh File Reference

Define the FrameDerivs class.

```
#include "utils/quaternion/include/quat.hh"
```

Data Structures

class jeod::FrameDerivs

Contains translational and rotational second derivatives.

Namespaces

• jeod

Namespace jeod.

9.24.1 Detailed Description

Define the FrameDerivs class.

9.25 structure_integrated_dyn_body.cc File Reference

Define base member functions for StructureIntegratedDynBody.

```
#include "../include/structure_integrated_dyn_body.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include <cstddef>
```

Namespaces

jeod

Namespace jeod.

9.25.1 Detailed Description

Define base member functions for StructureIntegratedDynBody.

9.26 structure_integrated_dyn_body.hh File Reference

Define the class StructureIntegratedDynBody, which integrates a DynBody object's structural state.

```
#include "body_wrench_collect.hh"
#include "vehicle_non_grav_state.hh"
#include "vehicle_properties.hh"
#include "dynamics/dyn_body/include/dyn_body.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

· class jeod::StructureIntegratedDynBody

Extends DynBody to integrate an object's structural reference frame as opposed to its center of mass.

Namespaces

jeod

Namespace jeod.

9.26.1 Detailed Description

Define the class StructureIntegratedDynBody, which integrates a DynBody object's structural state.

9.27 structure_integrated_dyn_body_collect.cc File Reference

Define StructureIntegratedDynBody methods related to force and torque accumulation and propagation.

```
#include "../include/structure_integrated_dyn_body.hh"
#include "dynamics/dyn_manager/include/base_dyn_manager.hh"
#include "utils/math/include/matrix3x3.hh"
#include "utils/math/include/vector3.hh"
#include <cstddef>
```

Namespaces

jeod

Namespace jeod.

Functions

static void jeod::accumulate_forces (const JeodPointerVector< CollectForce >::type &vec, double *cumulation)

Accumulate forces acting on a vehicle.

static void jeod::accumulate_torques (const JeodPointerVector< CollectTorque >::type &vec, double *cumulation)

Accumulate torques acting on a vehicle.

9.27.1 Detailed Description

Define StructureIntegratedDynBody methods related to force and torque accumulation and propagation.

9.28 structure_integrated_dyn_body_integration.cc File Reference

Define StructureIntegratedDynBody member functions related to state integration.

```
#include "../include/structure_integrated_dyn_body.hh"
#include "dynamics/dyn_body/include/dyn_body_messages.hh"
#include "utils/math/include/vector3.hh"
#include "utils/memory/include/jeod_alloc.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/ref_frames/include/ref_frame_items.hh"
#include "er7_utils/integration/core/include/second_order_ode_integrator.
hh"
#include <cmath>
#include <cstddef>
```

Namespaces

jeod

Namespace jeod.

9.28.1 Detailed Description

Define StructureIntegratedDynBody member functions related to state integration.

9.29 structure_integrated_dyn_body_pt_accel.cc File Reference

Define StructureIntegratedDynBody::compute_vehicle_point_derivatives.

```
#include "../include/structure_integrated_dyn_body.hh"
#include "dynamics/dyn_body/include/dyn_body_messages.hh"
#include "utils/math/include/vector3.hh"
#include "utils/message/include/message_handler.hh"
#include <cstdio>
#include <cstring>
```

Namespaces

· jeod

Namespace jeod.

9.29.1 Detailed Description

Define StructureIntegratedDynBody::compute vehicle point derivatives.

9.30 structure_integrated_dyn_body_solve.cc File Reference

Define StructureIntegratedDynBody methods related to force and torque accumulation and propagation.

```
#include "../include/structure_integrated_dyn_body.hh"
#include "../include/dyn_body_messages.hh"
#include "utils/message/include/message_handler.hh"
#include "utils/math/include/vector3.hh"
#include "experimental/constraints/include/dyn_body_constraints_solver.hh"
```

Namespaces

jeod

Namespace jeod.

9.30.1 Detailed Description

Define StructureIntegratedDynBody methods related to force and torque accumulation and propagation.

9.31 torque.cc File Reference

Define torque model member functions.

```
#include <cstddef>
#include "utils/memory/include/jeod_alloc.hh"
#include "../include/torque.hh"
```

Namespaces

• jeod

Namespace jeod.

9.31.1 Detailed Description

Define torque model member functions.

9.32 torque.hh File Reference

Define the JEOD torque model.

```
#include "torque_inline.hh"
```

Data Structures

class jeod::Torque

A Torque represents a Newtonian torque that acts on a DynBody.

• class jeod::CollectTorque

A CollectTorque represents a collected torque that acts on a vehicle.

• class jeod::CInterfaceTorque

This class is deprecated.

Namespaces

• jeod

Namespace jeod.

9.32.1 Detailed Description

Define the JEOD torque model.

9.33 torque_inline.hh File Reference

Define the JEOD torque model.

```
#include "torque.hh"
#include <cstddef>
```

Namespaces

jeod

Namespace jeod.

9.33.1 Detailed Description

Define the JEOD torque model.

9.34 vehicle_non_grav_state.hh File Reference

Define the class VehicleNonGravState.

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

Data Structures

• class jeod::VehicleNonGravState

Encapsulates various aspects of a vehicle's state with respect to inertial.

Namespaces

• jeod

Namespace jeod.

9.34.1 Detailed Description

Define the class VehicleNonGravState.

9.35 vehicle_properties.hh File Reference

Define the class VehicleProperties.

```
#include "experimental/math/include/solver_types.hh"
#include "utils/sim_interface/include/jeod_class.hh"
```

Data Structures

· class jeod::VehicleProperties

Captures pointers to various vehicle properties that are commonly used in the constraint concept.

Namespaces

• jeod

Namespace jeod.

9.35.1 Detailed Description

Define the class VehicleProperties.

9.36 wrench.hh File Reference

Define the class Wrench.

```
#include "dynamics/mass/include/mass_point_state.hh"
#include "utils/math/include/vector3.hh"
#include "utils/sim_interface/include/jeod_class.hh"
#include <vector>
```

Data Structures

class jeod::Wrench

A wrench comprises a torque and a force applied at a point on a DynBody.

Namespaces

• jeod

Namespace jeod.

9.36.1 Detailed Description

Define the class Wrench.

Index

\sim BodyForceCollect	add_mass_body
jeod::BodyForceCollect, 22	jeod::DynBody, 61
\sim BodyRefFrame	add_mass_body_frames
jeod::BodyRefFrame, 30	jeod::DynBody, 61
\sim BodyWrenchCollect	add_mass_body_validate
jeod::BodyWrenchCollect, 32	jeod::DynBody, 62
\sim CInterfaceForce	add_mass_point
jeod::CInterfaceForce, 36	jeod::DynBody, 62
\sim CInterfaceTorque	associated_integrable_objects
jeod::CInterfaceTorque, 38	jeod::DynBody, 95
~CollectForce	attach_child
jeod::CollectForce, 41	jeod::DynBody, <mark>63</mark>
~CollectTorque	attach_establish_links
jeod::CollectTorque, 48	jeod::DynBody, 63
~DynBody	attach_to
jeod::DynBody, 59	jeod::DynBody, 64, 65
~Force	attach_to_frame
jeod::Force, 115	jeod::DynBody, 65, 66
~StructureIntegratedDynBody	attach_update_properties
jeod::StructureIntegratedDynBody, 126	jeod::DynBody, 66
~Torque	jeod::StructureIntegratedDynBody, 126
jeod::Torque, 137	attach_validate_child
~Wrench	jeod::DynBody, 67
jeod::Wrench, 152	attach_validate_parent
jeodwrendi, 132	.
accel_struct	jeod::DynBody, 68
jeod::VehicleNonGravState, 141	attitude_source
accumulate	jeod::DynBody, 96
jeod::BodyWrenchCollect, 33	autoupdate_vehicle_points
jeod::Wrench, 153	jeod::DynBody, 96
accumulate_forces	hady fares callest as 161
jeod, 16, 17	body_force_collect.cc, 161
•	body_force_collect.hh, 161
accumulate_torques	body_ref_frame.hh, 162
jeod, 17	body_wrench_collect.cc, 163
activate	body_wrench_collect.hh, 163
jeod::DynBody, 60	BodyForceCollect
jeod::Wrench, 153	jeod::BodyForceCollect, 22, 23
active	BodyRefFrame
jeod::CollectForce, 45	jeod::BodyRefFrame, 29, 30
jeod::CollectTorque, 53	BodyWrenchCollect
jeod::DynBodyGenericFrameAttachment, 109	jeod::BodyWrenchCollect, 32, 33
jeod::Force, 116	
jeod::Torque, 139	CInterfaceForce
jeod::Wrench, 159	jeod::CInterfaceForce, 35, 36
add_constraint	CInterfaceTorque
jeod::StructureIntegratedDynBody, 126	jeod::CInterfaceTorque, 37, 38
add_control	check_frame_ownership
jeod::DynBody, 60	jeod, 18
add_integrable_object	class_declarations.hh, 163
jeod::DynBody, 60	clear_attachment

jeod::DynBodyGenericFrameAttachment, 108	compute_vehicle_point_derivatives
clear_integrable_objects	jeod::DynBody, 72
jeod::DynBody, 69	jeod::StructureIntegratedDynBody, 129
collect	compute_vehicle_point_states
jeod::DynBody, 96	jeod::DynBody, <mark>72</mark>
collect_effector_forc	constraints_solver
jeod::BodyForceCollect, 23	jeod::StructureIntegratedDynBody, 133
collect_effector_torq	core_body
jeod::BodyForceCollect, 23	jeod::DynBody, 97
collect_environ_forc	create
jeod::BodyForceCollect, 24	jeod::CollectForce, 42, 43
collect_environ_torq	jeod::CollectTorque, 49–51
jeod::BodyForceCollect, 24	create_body_integrators
collect_forces_and_torques	jeod::DynBody, 73
jeod::DynBody, 69	create_integrators
jeod::StructureIntegratedDynBody, 127	jeod::DynBody, 73
collect_insert	deactivate
jeod, 18	jeod::DynBody, 74
jeod::JPVCollectForce, 120	jeod::Wrench, 154
jeod::JPVCollectTorque, 123	derivs
collect_local_forces_and_torques	jeod::DynBody, 98
jeod::StructureIntegratedDynBody, 127	destroy_integrators
collect_no_xmit_forc	jeod::DynBody, 74
jeod::BodyForceCollect, 24	detach
collect_no_xmit_torq	jeod::DynBody, 74, 75
jeod::BodyForceCollect, 24	jeod::StructureIntegratedDynBody, 129
collect_push_back	detach_mass_body_frames
jeod, 18	jeod::DynBody, 75
jeod::JPVCollectForce, 121	detach_mass_internal
jeod::JPVCollectTorque, 123	jeod::DynBody, 76
collect_wrench	dyn_body.cc, 164
jeod::BodyWrenchCollect, 34	dyn_body.hh, 164
CollectForce	dyn body attach.cc, 165
jeod::CollectForce, 40, 41	dyn_body_collect.cc, 166
CollectTorque	dyn_body_detach.cc, 166
jeod::CollectTorque, 47, 48	dyn_body_find_body_frame.cc, 167
complete_translational_acceleration	dyn body generic rigid attach.hh, 167
jeod::StructureIntegratedDynBody, 128	dyn_body_initialize_model.cc, 168
composite_body	dyn_body_integration.cc, 168
jeod::DynBody, 97	dyn_body_messages.cc, 169
compute_derived_state_forward	MAKE DYNBODY MESSAGE CODE, 169
jeod::DynBody, 69	dyn body messages.hh, 170
compute_derived_state_reverse	dyn body propagate state.cc, 170
jeod::DynBody, 70	dyn_body_set_state.cc, 171
compute_inertial_torque	dyn_body_vehicle_point.cc, 171
jeod::StructureIntegratedDynBody, 128	dyn_children
compute_point_derivative	jeod::DynBody, 98
jeod::DynBody, 97	dyn_manager
compute_ref_point_transform	jeod::DynBody, 98
jeod::DynBody, 70	dyn_parent
compute_rotational_acceleration	jeod::DynBody, 99
jeod::StructureIntegratedDynBody, 128	DynBody, 13
compute_state_elements_forward	jeod::DynBody, 59, 60
jeod::DynBody, 71	DynBodyConstraintsSolver
compute_state_elements_reverse	jeod::StructureIntegratedDynBody, 133
jeod::DynBody, 71	DynBodyGenericFrameAttachment
compute_translational_acceleration	jeod::DynBodyGenericFrameAttachment, 107
jeod::StructureIntegratedDynBody, 129	DynBodyMessages

jeod::DynBodyMessages, 111	get inverse mass
Dynamics, 12	jeod::VehicleProperties, 145
- , · · · · · · · · · · · · · · · · · · ·	get_mass
effector_forc	jeod::VehicleProperties, 145
jeod::BodyForceCollect, 25	get_parent_body
effector_torq	jeod::DynBody, 79
jeod::BodyForceCollect, 25	get_parent_body_internal
effector_wrench	jeod::DynBody, 79
jeod::StructureIntegratedDynBody, 134	get_parent_frame
effector_wrench_collection	jeod::DynBodyGenericFrameAttachment, 108
jeod::StructureIntegratedDynBody, 134	get_parent_to_structure_offset
environ_forc	jeod::VehicleProperties, 145
jeod::BodyForceCollect, 25	get_parent_to_structure_transform
environ_torq jeod::BodyForceCollect, 26	jeod::VehicleProperties, 146
extern_forc_inrtl	get_point
jeod::BodyForceCollect, 26	jeod::Wrench, 154
extern_forc_struct	get_root_body jeod::DynBody, 79
jeod::BodyForceCollect, 26	get_root_body_internal
extern_torq_body	jeod::DynBody, 80
jeod::BodyForceCollect, 27	get_structure_to_body_offset
extern_torq_struct	jeod::VehicleProperties, 146
jeod::BodyForceCollect, 27	get_structure_to_body_transform
,	jeod::VehicleProperties, 146
find_body_frame	get_torque
jeod::DynBody, 76	jeod::Wrench, 154
find_vehicle_point	get_vehicle_properties
jeod::DynBody, 77	jeod::StructureIntegratedDynBody, 130
Force	grav_interaction
jeod::Force, 115	jeod::DynBody, 99
force	
jeod::CollectForce, 45	inertia
jeod::Force, 116	jeod::VehicleProperties, 147
jeod::Wrench, 159	inertial_accel_inrtl
force.cc, 172	jeod::StructureIntegratedDynBody, 134
force.hh, 172	inertial_accel_struct
force_inline.hh, 173	jeod::StructureIntegratedDynBody, 135
frame_attach	inertial_accel_struct_omega
jeod::DynBody, 99	jeod::StructureIntegratedDynBody, 135 inertial_accel_struct_omega_dot
frame_derivs.hh, 173 FrameDerivs	jeod::StructureIntegratedDynBody, 135
	inertial_torq
jeod::FrameDerivs, 117	jeod::BodyForceCollect, 27
get_attach_offset	inertial_torque_struct
jeod::DynBodyGenericFrameAttachment, 108	jeod::VehicleNonGravState, 141
get_dynamics_integration_group	init_attrjeodBodyRefFrame
jeod::DynBody, 78	jeod::BodyRefFrame, 30
get_force	init attrjeod DynBody
jeod::Wrench, 154	jeod::DynBody, 95
get_inertia	init_attrjeodDynBodyGenericFrameAttachment
jeod::VehicleProperties, 144	jeod::DynBodyGenericFrameAttachment, 109
get_initialized_states	init_attrjeodDynBodyMessages
jeod::DynBody, 78	jeod::DynBodyMessages, 111
get_integ_frame	init_attrjeodStructureIntegratedDynBody
jeod::DynBody, 78	jeod::StructureIntegratedDynBody, 133
get_integrable_objects	init_attrjeodVehicleNonGravState
jeod::DynBody, 78	jeod::VehicleNonGravState, 140
get_inverse_inertia	init_attrjeodVehicleProperties
jeod::VehicleProperties, 145	jeod::VehicleProperties, 147

init_attrjeodWrench jeod::Wrench, 159	jeod::DynBodyGenericFrameAttachment, 108
initialize_attachment	jeod, 15
	accumulate_forces, 16, 17
jeod::DynBodyGenericFrameAttachment, 108	accumulate_torques, 17
initialize_controls	check frame ownership, 18
jeod::DynBody, 80	collect_insert, 18
initialize_model	collect_push_back, 18
jeod::DynBody, 81	release_vector, 19
initialized_items	jeod::BodyForceCollect, 21
jeod::BodyRefFrame, 31	~BodyForceCollect, 22
initialized_states	BodyForceCollect, 22, 23
jeod::DynBody, 100	collect_effector_forc, 23
initialized_states_contains	collect_effector_torq, 23
jeod::DynBody, 81	collect_environ_forc, 24
InputProcessor	collect_environ_torq, 24
jeod::BodyRefFrame, 30	collect_no_xmit_forc, 24
jeod::DynBody, 95	collect_no_xmit_torq, 24
jeod::DynBodyGenericFrameAttachment, 109	effector_forc, 25
jeod::DynBodyMessages, 111	effector_torq, 25
jeod::StructureIntegratedDynBody, 133	environ_forc, 25
jeod::VehicleNonGravState, 140	environ_torq, 26
jeod::VehicleProperties, 147	extern_forc_inrtl, 26
jeod::Wrench, 159	extern_forc_struct, 26
integ_frame	extern_torq_body, 27
jeod::DynBody, 100	extern_torq_body, 27 extern_torq_struct, 27
integ_frame_name	inertial_torq, 27
jeod::DynBody, 100	
integ_results_merger	no_xmit_forc, 28
jeod::DynBody, 101	no_xmit_torq, 28
integrate	operator=, 23
jeod::DynBody, 81	jeod::BodyRefFrame, 29
integrated_frame	~BodyRefFrame, 30
jeod::DynBody, 101	BodyRefFrame, 29, 30
internal_error	init_attrjeodBodyRefFrame, 30
jeod::DynBodyMessages, 112	initialized_items, 31
invalid_attachment	InputProcessor, 30
jeod::DynBodyMessages, 112	mass_point, 31
invalid body	operator=, 30
jeod::DynBodyMessages, 112	jeod::BodyWrenchCollect, 31
invalid frame	~BodyWrenchCollect, 32
jeod::DynBodyMessages, 112	accumulate, 33
invalid_group	BodyWrenchCollect, 32, 33
jeod::DynBodyMessages, 113	collect_wrench, 34
invalid name	operator=, 34
jeod::DynBodyMessages, 113	jeod::CInterfaceForce, 35
	~CInterfaceForce, 36
invalid_technique	CInterfaceForce, 35, 36
jeod::DynBodyMessages, 113	operator=, 36
inverse_inertia	jeod::CInterfaceTorque, 37
jeod::VehicleProperties, 147	~CInterfaceTorque, 38
inverse_mass	CInterfaceTorque, 37, 38
jeod::VehicleProperties, 147	operator=, 38
is_active	jeod::CollectForce, 39
jeod::CollectForce, 44	∼CollectForce, 41
jeod::CollectTorque, 51	active, 45
jeod::Wrench, 154	CollectForce, 40, 41
is_root_body	create, 42, 43
jeod::DynBody, 82	force, 45
isAttached	is_active, 44

	C 1 1:1 : 1 - 77
operator=, 44	find_vehicle_point, 77
operator==, 44	frame_attach, 99
operator[], 44, 45	get_dynamics_integration_group, 78
jeod::CollectTorque, 46	get_initialized_states, 78
∼CollectTorque, 48	get_integ_frame, 78
active, 53	get_integrable_objects, 78
CollectTorque, 47, 48	get_parent_body, 79
create, 49–51	get_parent_body_internal, 79
is_active, 51	get_root_body, 79
operator=, 51	get_root_body_internal, 80
operator==, 51	grav_interaction, 99
operator[], 52	init_attrjeodDynBody, 95
torque, 53	initialize_controls, 80
jeod::DynBody, 53	initialize_model, 81
\sim DynBody, 59	initialized_states, 100
activate, 60	initialized_states_contains, 81
add_control, 60	InputProcessor, 95
add_integrable_object, 60	integ_frame, 100
add_mass_body, 61	integ_frame_name, 100
add_mass_body_frames, 61	integ_results_merger, 101
add_mass_body_validate, 62	integrate, 81
add_mass_point, 62	integrated_frame, 101
associated_integrable_objects, 95	is_root_body, 82
attach_child, 63	mass, 101
attach_establish_links, 63	mass_children, 102
attach_to, 64, 65	migrate_integrable_objects, 82
attach_to_frame, 65, 66	name, 102
attach_update_properties, 66	operator=, 82
attach_validate_child, 67	position_source, 102
attach_validate_parent, 68	process_dynamic_attachment, 83
attitude_source, 96	propagate_state, 83
autoupdate_vehicle_points, 96	propagate_state_from_composite, 84
clear_integrable_objects, 69	propagate_state_from_structure, 84
collect, 96	rate_source, 103
collect_forces_and_torques, 69	remove_integrable_object, 84
composite_body, 97	remove_mass_body, 85
compute_derived_state_forward, 69	reset_controls, 85
compute_derived_state_reverse, 70	reset_integrators, 86
compute_point_derivative, 97	rot_integ, 86
compute_ref_point_transform, 70	rot_integrator, 103
compute_state_elements_forward, 71	rotation_integration, 103
compute_state_elements_reverse, 71	rotational_dynamics, 104
compute_vehicle_point_derivatives, 72	set_attitude_left_quaternion, 87
compute_vehicle_point_states, 72	set_attitude_matrix, 87
core_body, 97	set_attitude_rate, 88
create_body_integrators, 73	set_attitude_right_quaternion, 88
create_integrators, 73	set_integ_frame, 89
deactivate, 74	set_name, 90
derivs, 98	set_position, 90
destroy_integrators, 74	set_state, 90
detach, 74, 75	set_state_source, 91
detach_mass_body_frames, 75	set_state_source_internal, 92
detach_mass_internal, 76 dyn_children, 98	set_velocity, 92
dyn_cniidren, 98 dyn_manager, 98	sort_controls, 93 structure, 104
dyn_manager, 98 dyn_parent, 99	switch_integration_frames, 93
DynBody, 59, 60	three dof, 104
find_body_frame, 76	time manager, 105

	trans integ 04	attack undata proportion 100
	trans_integrater_10F	attach_update_properties, 126
	trans_integrator, 105	collect_forces_and_torques, 127
	translational_dynamics, 105	collect_local_forces_and_torques, 127
	update_integrated_state, 95	complete_translational_acceleration, 128
	vehicle_points, 106	compute_inertial_torque, 128
	velocity_source, 106	compute_rotational_acceleration, 128
jeoa	::DynBodyGenericFrameAttachment, 107	compute_translational_acceleration, 129
	active, 109	compute_vehicle_point_derivatives, 129
	clear_attachment, 108	constraints_solver, 133
	DynBodyGenericFrameAttachment, 107	detach, 129
	get_attach_offset, 108	DynBodyConstraintsSolver, 133
	get_parent_frame, 108	effector_wrench, 134
	init_attrjeodDynBodyGenericFrameAttachment,	effector_wrench_collection, 134
	109	get_vehicle_properties, 130
	initialize_attachment, 108	inertial_accel_inrtl, 134
	InputProcessor, 109	inertial_accel_struct, 135
	isAttached, 108	inertial_accel_struct_omega, 135
	rigid_attach_parent, 109	inertial_accel_struct_omega_dot, 135
	rigid_attach_state, 109	init_attrjeodStructureIntegratedDynBody, 133
jeod	::DynBodyMessages, 110	InputProcessor, 133
	DynBodyMessages, 111	non_grav_state, 135
	init_attrjeodDynBodyMessages, 111	operator=, 130
	InputProcessor, 111	PropagateForcesAndTorques, 130
	internal_error, 112	rot_integ, 130
	invalid_attachment, 112	set_solver, 132
	invalid_body, 112	solve_constraints, 132
	invalid_frame, 112	struct_derivs, 136
	invalid_group, 113	StructureIntegratedDynBody, 125, 126
	invalid_name, 113	trans_integ, 132
	invalid_technique, 113	vehicle_properties, 136
	not_dyn_body, 113	jeod::Torque, 137
	operator=, 111	~Torque, 137
ieod	::Force, 114	active, 139
,000	∼Force, 115	operator=, 138
	active, 116	operator[], 138
	Force, 115	Torque, 137, 138
	force, 116	torque, 139
	operator=, 115	jeod::VehicleNonGravState, 140
	operator[], 115, 116	accel_struct, 141
hoei	::FrameDerivs, 117	inertial_torque_struct, 141
jeou	FrameDerivs, 117	init attrjeod VehicleNonGravState, 140
	non_grav_accel, 118	InputProcessor, 140
	Qdot_parent_this, 118	omega body, 141
	rot_accel, 118	omega_body, 141 omega_dot_body, 141
iood	trans_accel, 118	omega_dot_struct, 142
jeod	::JPVCollectForce, 119	omega_struct, 142
	collect_insert, 120	jeod::VehicleProperties, 142
р	collect_push_back, 121	get_inertia, 144
	perform_insert_action, 120	get_inverse_inertia, 145
	push_back, 120	get_inverse_mass, 145
jeoa	::JPVCollectTorque, 121	get_mass, 145
	collect_insert, 123	get_parent_to_structure_offset, 145
	collect_push_back, 123	get_parent_to_structure_transform, 146
	perform_insert_action, 122	get_structure_to_body_offset, 146
	push_back, 122	get_structure_to_body_transform, 146
jeod	::StructureIntegratedDynBody, 123	inertia, 147
	~StructureIntegratedDynBody, 126	init_attrjeodVehicleProperties, 147
	add_constraint, 126	InputProcessor, 147

inverse_inertia, 147	jeod::FrameDerivs, 118
inverse_mass, 147	-
	non_grav_state
mass, 148	jeod::StructureIntegratedDynBody, 135
parent_to_structure_offset, 148	not_dyn_body
parent_to_structure_transform, 148	jeod::DynBodyMessages, 113
structure_to_body_offset, 148	bada
structure_to_body_transform, 149	omega_body
VehicleProperties, 144	jeod::VehicleNonGravState, 141
jeod::Wrench, 149	omega_dot_body
\sim Wrench, 152	jeod::VehicleNonGravState, 141
accumulate, 153	omega_dot_struct
activate, 153	jeod::VehicleNonGravState, 142
active, 159	omega_struct
deactivate, 154	jeod::VehicleNonGravState, 142
force, 159	operator+=
	jeod::Wrench, 154
get_force, 154	operator=
get_point, 154	jeod::BodyForceCollect, 23
get_torque, 154	
init_attrjeodWrench, 159	jeod::BodyRefFrame, 30
InputProcessor, 159	jeod::BodyWrenchCollect, 34
is_active, 154	jeod::CInterfaceForce, 36
operator+=, 154	jeod::CInterfaceTorque, 38
operator=, 155	jeod::CollectForce, 44
point, 159	jeod::CollectTorque, 51
reset_force, 155	jeod::DynBody, 82
reset_force_and_torque, 155	jeod::DynBodyMessages, 111
reset_point, 155	jeod::Force, 115
reset_torque, 156	jeod::StructureIntegratedDynBody, 130
	jeod::Torque, 138
scale_force, 156	jeod::Vrench, 155
scale_torque, 156	-
set, 156	operator==
set_force, 157	jeod::CollectForce, 44
set_point, 157	jeod::CollectTorque, 51
set_torque, 157	operator[]
torque, 160	jeod::CollectForce, 44, 45
transform_to_parent, 158	jeod::CollectTorque, 52
transform_to_point, 158	jeod::Force, 115, 116
Wrench, 151, 152	jeod::Torque, 138
, ,	
MAKE_DYNBODY_MESSAGE_CODE	parent_to_structure_offset
dyn_body_messages.cc, 169	jeod::VehicleProperties, 148
mass	parent to structure transform
jeod::DynBody, 101	jeod::VehicleProperties, 148
	perform_insert_action
jeod::VehicleProperties, 148	jeod::JPVCollectForce, 120
mass_children	jeod::JPVCollectTorque, 122
jeod::DynBody, 102	-
mass_point	point
jeod::BodyRefFrame, 31	jeod::Wrench, 159
migrate_integrable_objects	position_source
jeod::DynBody, 82	jeod::DynBody, 102
Models, 11	process_dynamic_attachment
	jeod::DynBody, 83
name	propagate_state
jeod::DynBody, 102	jeod::DynBody, 83
no_xmit_forc	propagate_state_from_composite
jeod::BodyForceCollect, 28	jeod::DynBody, 84
no_xmit_torq	propagate_state_from_structure
·	jeod::DynBody, 84
jeod::BodyForceCollect, 28	
non_grav_accel	PropagateForcesAndTorques

jeod::StructureIntegratedDynBody, 130	jeod::Wrench, 157
push_back	set_integ_frame
jeod::JPVCollectForce, 120	jeod::DynBody, 89
jeod::JPVCollectTorque, 122	set_name
•	jeod::DynBody, 90
Qdot_parent_this	set_point
jeod::FrameDerivs, 118	jeod::Wrench, 157
	set_position
rate_source	jeod::DynBody, 90
jeod::DynBody, 103	set_solver
release_vector	jeod::StructureIntegratedDynBody, 132
jeod, 19	set_state
remove_integrable_object	jeod::DynBody, 90
jeod::DynBody, 84	set_state_source
remove_mass_body jeod::DynBody, 85	jeod::DynBody, 91
reset controls	set_state_source_internal
jeod::DynBody, 85	jeod::DynBody, 92
reset force	set_torque
jeod::Wrench, 155	jeod::Wrench, 157
reset_force_and_torque	set_velocity
jeod::Wrench, 155	jeod::DynBody, 92
reset_integrators	solve_constraints
jeod::DynBody, 86	jeod::StructureIntegratedDynBody, 132
reset_point	sort_controls
jeod::Wrench, 155	jeod::DynBody, 93
reset_torque	struct_derivs
jeod::Wrench, 156	jeod::StructureIntegratedDynBody, 136
rigid_attach_parent	structure
jeod::DynBodyGenericFrameAttachment, 109	jeod::DynBody, 104
rigid_attach_state	structure_integrated_dyn_body.cc, 174
jeod::DynBodyGenericFrameAttachment, 109	structure_integrated_dyn_body.hh, 174 structure_integrated_dyn_body_collect.cc, 175
rot accel	structure_integrated_dyn_body_integration.cc, 175
jeod::FrameDerivs, 118	structure_integrated_dyn_body_pt_accel.cc, 176
rot integ	structure_integrated_dyn_body_solve.cc, 176
jeod::DynBody, 86	structure_to_body_offset
jeod::StructureIntegratedDynBody, 130	jeod::VehicleProperties, 148
rot_integrator	structure_to_body_transform
jeod::DynBody, 103	jeod::VehicleProperties, 149
rotation_integration	StructureIntegratedDynBody
jeod::DynBody, 103	jeod::StructureIntegratedDynBody, 125, 126
rotational_dynamics	switch_integration_frames
jeod::DynBody, 104	jeod::DynBody, 93
	• •
scale_force	three_dof
jeod::Wrench, 156	jeod::DynBody, 104
scale_torque	time_manager
jeod::Wrench, 156	jeod::DynBody, 105
set	Torque
jeod::Wrench, 156	jeod::Torque, 137, 138
set_attitude_left_quaternion	torque
jeod::DynBody, 87	jeod::CollectTorque, 53
set_attitude_matrix	jeod::Torque, 139
jeod::DynBody, 87	jeod::Wrench, 160
set_attitude_rate	torque.cc, 177
jeod::DynBody, 88	torque.hh, 177
set_attitude_right_quaternion	torque_inline.hh, 178
jeod::DynBody, 88	trans_accel
set_force	jeod::FrameDerivs, 118

```
trans_integ
    jeod::DynBody, 94
    jeod::StructureIntegratedDynBody, 132
trans_integrator
    jeod::DynBody, 105
transform to parent
    jeod::Wrench, 158
transform_to_point
    jeod::Wrench, 158
translational_dynamics
    jeod::DynBody, 105
update_integrated_state
    jeod::DynBody, 95
vehicle_non_grav_state.hh, 178
vehicle_points
    jeod::DynBody, 106
vehicle_properties
    jeod::StructureIntegratedDynBody, 136
vehicle_properties.hh, 178
VehicleProperties
    jeod::VehicleProperties, 144
velocity_source
    jeod::DynBody, 106
Wrench
    jeod::Wrench, 151, 152
wrench.hh, 179
```