Polylib

4.0.0

Generated by Doxygen 1.8.6

Tue Mar 15 2016 18:39:08

Contents

1 Namespace Index **1**

1.1 Namespace List . . . 1

2 Hierarchical Index **3**

2.1 Class Hierarchy . . . 3

3 Class Index **5**

3.1 Class List . . . 5

4 File Index **7**

4.1 File List . . . 7

5 Namespace Documentation **9**

5.1 PolylibNS Namespace Reference . . . 9

5.1.1 Typedef Documentation . . . 11

5.1.1.1 Vec2f . . . 11

5.1.1.2 Vec2i . . . 11

5.1.1.3 Vec2r . . . 11

5.1.1.4 Vec2uc . . . 11

5.1.2 Enumeration Type Documentation . . . 11

5.1.2.1 ID_FORMAT . . . 11

5.1.3 Function Documentation . . . 11

5.1.3.1 alloc_array_2d_int . . . 11

5.1.3.2 alloc_array_2d_real . . . 11

5.1.3.3 convert_polygons_to_npt . . . 12

5.1.3.4 convert_polygons_to_tri . . . 13

5.1.3.5 copy_polygon . . . 13

5.1.3.6 copy_polygons . . . 13

5.1.3.7 deserialize_polygon . . . 13

5.1.3.8 distance . . . 14

5.1.3.9 distanceSquared . . . 14

5.1.3.10 free_array_2d . . . 14

5.1.3.11 get_ext_fr_path . . . 14

5.1.3.12 get_fname_fr_path . . . 14
5.1.3.13 getrusage_sec . . . 15
5.1.3.14 is_npt_a . . . 15
5.1.3.15 is_stl_a . . . 16
5.1.3.16 lessVec2f . . . 16
5.1.3.17 npt_a_load . . . 16
5.1.3.18 npt_a_load_read . . . 16
5.1.3.19 npt_a_load_read_head . . . 17
5.1.3.20 npt_a_save . . . 17
5.1.3.21 npt_b_load . . . 17
5.1.3.22 npt_b_load_read . . . 18
5.1.3.23 npt_b_load_read_head . . . 18
5.1.3.24 npt_b_save . . . 18
5.1.3.25 operator* . . . 19
5.1.3.26 operator<< . . . 19
5.1.3.27 operator<< . . . 19
5.1.3.28 operator>> . . . 19
5.1.3.29 operator>> . . . 19
5.1.3.30 stl_a_load . . . 19
5.1.3.31 stl_a_load_read . . . 19
5.1.3.32 stl_a_save . . . 20
5.1.3.33 stl_b_load . . . 20
5.1.3.34 stl_b_load_read . . . 20
5.1.3.35 stl_b_load_read_head . . . 21
5.1.3.36 stl_b_save . . . 21
5.1.4 Variable Documentation . . . 21
5.1.4.1 gs_rankno . . . 21
5.2 Vec3class Namespace Reference . . . 22
5.2.1 Typedef Documentation . . . 22
5.2.1.1 Vec3d . . . 22
5.2.1.2 Vec3f . . . 22
5.2.1.3 Vec3i . . . 22
5.2.1.4 Vec3r . . . 22
5.2.1.5 Vec3uc . . . 22
5.2.2 Enumeration Type Documentation . . . 23
5.2.2.1 AxisEnum . . . 23
5.2.3 Function Documentation . . . 23
5.2.3.1 cross . . . 23
5.2.3.2 distance . . . 23
5.2.3.3 distanceSquared . . . 23

5.2.3.4 dot . . . 23
5.2.3.5 lessVec3f . . . 23
5.2.3.6 multi . . . 23
5.2.3.7 operator∗ . . . 23
5.2.3.8 operator<< . . . 23
5.2.3.9 operator<< . . . 23
5.2.3.10 operator>> . . . 23
5.2.3.11 operator>> . . . 23

6 Class Documentation **25**

6.1 PolylibNS::BBox Class Reference . . . 25
6.1.1 Detailed Description . . . 26
6.1.2 Constructor & Destructor Documentation . . . 26
6.1.2.1 BBox . . . 26
6.1.2.2 BBox . . . 26
6.1.2.3 BBox . . . 26
6.1.2.4 BBox . . . 26
6.1.3 Member Function Documentation . . . 26
6.1.3.1 add . . . 26
6.1.3.2 center . . . 26
6.1.3.3 contain . . . 26
6.1.3.4 crossed . . . 26
6.1.3.5 diameter . . . 26
6.1.3.6 getCrossedRegion . . . 26
6.1.3.7 getFace . . . 27
6.1.3.8 getMaxAxis . . . 27
6.1.3.9 getPoint . . . 27
6.1.3.10 getSide . . . 27
6.1.3.11 init . . . 27
6.1.3.12 length . . . 27
6.1.3.13 setMinMax . . . 27
6.1.3.14 size . . . 27
6.1.3.15 vec3to2 . . . 27
6.1.3.16 xsize . . . 27
6.1.3.17 ysize . . . 27
6.1.3.18 zsize . . . 27
6.1.4 Member Data Documentation . . . 27
6.1.4.1 max . . . 27
6.1.4.2 min . . . 28
6.2 PolylibNS::CalcAreaInfo Struct Reference . . . 28

6.2.1 Detailed Description . . . 28
6.2.2 Member Data Documentation . . . 28
6.2.2.1 m_bbsize . . . 28
6.2.2.2 m_bpos . . . 28
6.2.2.3 m_dx . . . 28
6.2.2.4 m_gcell_bbox . . . 28
6.2.2.5 m_gcell_max . . . 29
6.2.2.6 m_gcell_min . . . 29
6.2.2.7 m_gcsize . . . 29
6.3 FParallelBboxStruct Struct Reference . . . 29
6.3.1 Detailed Description . . . 29
6.3.2 Member Data Documentation . . . 29
6.3.2.1 bbsize . . . 29
6.3.2.2 bpos . . . 29
6.3.2.3 dx . . . 29
6.3.2.4 gcsize . . . 29
6.4 PolylibNS::NpatchParam Struct Reference . . . 29
6.4.1 Detailed Description . . . 30
6.4.2 Member Data Documentation . . . 30
6.4.2.1 cp_center . . . 30
6.4.2.2 cp_side1_1 . . . 30
6.4.2.3 cp_side1_2 . . . 30
6.4.2.4 cp_side2_1 . . . 30
6.4.2.5 cp_side2_2 . . . 30
6.4.2.6 cp_side3_1 . . . 30
6.4.2.7 cp_side3_2 . . . 31
6.5 NpatchParamStruct Struct Reference . . . 31
6.5.1 Detailed Description . . . 31
6.5.2 Member Data Documentation . . . 31
6.5.2.1 cp_center . . . 31
6.5.2.2 cp_side1_1 . . . 31
6.5.2.3 cp_side1_2 . . . 31
6.5.2.4 cp_side2_1 . . . 31
6.5.2.5 cp_side2_2 . . . 32
6.5.2.6 cp_side3_1 . . . 32
6.5.2.7 cp_side3_2 . . . 32
6.6 PolylibNS::NptTriangle Class Reference . . . 32
6.6.1 Detailed Description . . . 33
6.6.2 Constructor & Destructor Documentation . . . 33
6.6.2.1 NptTriangle . . . 33

6.6.2.2 NptTriangle . . . 33
6.6.2.3 NptTriangle . . . 33
6.6.2.4 NptTriangle . . . 33
6.6.2.5 ∼NptTriangle . . . 33
6.6.3 Member Function Documentation . . . 34
6.6.3.1 correct . . . 34
6.6.3.2 correct . . . 35
6.6.3.3 get_bbox . . . 35
6.6.3.4 get_npatch_param . . . 35
6.6.3.5 get_pl_type . . . 35
6.6.3.6 rescale . . . 36
6.6.3.7 serialize . . . 36
6.6.3.8 serialized_size . . . 36
6.6.3.9 set_npatch_param . . . 36
6.6.3.10 set_vertexes . . . 37
6.6.3.11 set_vertexes . . . 37
6.6.3.12 used_memory_size . . . 37
6.6.4 Member Data Documentation . . . 37
6.6.4.1 m_npatchParam . . . 37
6.7 NptTriangleStruct Struct Reference . . . 38
6.7.1 Detailed Description . . . 38
6.7.2 Member Data Documentation . . . 38
6.7.2.1 param . . . 38
6.7.2.2 vertex . . . 38
6.8 PolylibNS::ParallelAreaInfo Struct Reference . . . 38
6.8.1 Detailed Description . . . 38
6.8.2 Member Data Documentation . . . 38
6.8.2.1 m_areas . . . 38
6.8.2.2 m_rank . . . 39
6.9 PolylibNS::ParallelBbox Struct Reference . . . 39
6.9.1 Detailed Description . . . 39
6.9.2 Member Data Documentation . . . 39
6.9.2.1 bbsize . . . 39
6.9.2.2 bpos . . . 39
6.9.2.3 dx . . . 39
6.9.2.4 gcsize . . . 39
6.10 ParallelBboxStruct Struct Reference . . . 39
6.10.1 Detailed Description . . . 40
6.10.2 Member Data Documentation . . . 40
6.10.2.1 bbsize . . . 40

6.10.2.2 bpos . . . 40
6.10.2.3 dx . . . 40
6.10.2.4 gcsize . . . 40
6.11 PolylibNS::PolygonGroup Class Reference . . . 40
6.11.1 Detailed Description . . . 42
6.11.2 Constructor & Destructor Documentation . . . 42
6.11.2.1 PolygonGroup . . . 42
6.11.2.2 ∼PolygonGroup . . . 43
6.11.3 Member Function Documentation . . . 43
6.11.3.1 acq_file_name . . . 43
6.11.3.2 acq_fullpath . . . 43
6.11.3.3 add_children . . . 43
6.11.3.4 add_triangles . . . 43
6.11.3.5 build_group_tree . . . 43
6.11.3.6 build_polygon_tree . . . 44
6.11.3.7 check_leaped . . . 44
6.11.3.8 erase_outbounded_polygons . . . 44
6.11.3.9 gather_polygons . . . 44
6.11.3.10 get_atr . . . 45
6.11.3.11 get_children . . . 45
6.11.3.12 get_file_name . . . 45
6.11.3.13 get_group_area . . . 45
6.11.3.14 get_group_global_area . . . 45
6.11.3.15 get_group_num_global_tria . . . 46
6.11.3.16 get_group_num_tria . . . 46
6.11.3.17 get_inbounded_polygons . . . 46
6.11.3.18 get_internal_id . . . 46
6.11.3.19 get_movable . . . 46
6.11.3.20 get_name . . . 47
6.11.3.21 get_num_of_trias_before_move . . . 47
6.11.3.22 get_num_polygon_atrI . . . 47
6.11.3.23 get_num_polygon_atrR . . . 47
6.11.3.24 get_parent . . . 47
6.11.3.25 get_parent_path . . . 48
6.11.3.26 get_polygons_reduce_atrI . . . 48
6.11.3.27 get_polygons_reduce_atrR . . . 49
6.11.3.28 get_triangles . . . 49
6.11.3.29 get_vtree . . . 49
6.11.3.30 init . . . 49
6.11.3.31 init . . . 50

6.11.3.32 init_check_leaped . . . 50
6.11.3.33 is_far . . . 50
6.11.3.34 load_polygons_file . . . 51
6.11.3.35 load_polygons_mem_reduced . . . 51
6.11.3.36 move . . . 51
6.11.3.37 rebuild_polygons . . . 51
6.11.3.38 remove_child . . . 51
6.11.3.39 rescale_polygons . . . 52
6.11.3.40 save_polygons_file . . . 52
6.11.3.41 save_polygons_file . . . 52
6.11.3.42 scatter_polygons . . . 53
6.11.3.43 scatter_polygons . . . 53
6.11.3.44 search . . . 53
6.11.3.45 search . . . 54
6.11.3.46 search . . . 55
6.11.3.47 search_nearest . . . 55
6.11.3.48 search_outbounded . . . 56
6.11.3.49 set_atr . . . 57
6.11.3.50 set_children . . . 57
6.11.3.51 set_file_name . . . 57
6.11.3.52 set_movable . . . 57
6.11.3.53 set_move_func . . . 58
6.11.3.54 set_move_func_c . . . 58
6.11.3.55 set_name . . . 58
6.11.3.56 set_need_rebuild . . . 58
6.11.3.57 set_num_polygon_atr . . . 58
6.11.3.58 set_parent . . . 59
6.11.3.59 set_parent_path . . . 59
6.11.3.60 set_triangles_ptr . . . 59
6.11.3.61 setup_attribute . . . 59
6.11.3.62 setup_user_attribute . . . 60
6.11.3.63 show_group_info . . . 60
6.11.4 Member Data Documentation . . . 60
6.11.4.1 ATT_NAME_CLASS . . . 60
6.11.4.2 m_atr . . . 60
6.11.4.3 m_bbox . . . 60
6.11.4.4 m_children . . . 60
6.11.4.5 m_internal_id . . . 61
6.11.4.6 m_max_elements . . . 61
6.11.4.7 m_movable . . . 61

6.11.4.8 m_move_func . . . 61
6.11.4.9 m_move_func_c . . . 61
6.11.4.10 m_name . . . 61
6.11.4.11 m_need_rebuild . . . 61
6.11.4.12 m_parent . . . 61
6.11.4.13 m_parent_path . . . 61
6.11.4.14 m_polygon_files . . . 61
6.11.4.15 m_tri_list . . . 61
6.11.4.16 m_trias_before_move . . . 61
6.11.4.17 m_vtree . . . 61
6.12 PolylibNS::PolygonIO Class Reference . . . 62
6.12.1 Detailed Description . . . 62
6.12.2 Member Function Documentation . . . 62
6.12.2.1 get_extension_format . . . 62
6.12.2.2 get_polygon_type . . . 63
6.12.2.3 input_file_format . . . 63
6.12.2.4 load . . . 63
6.12.2.5 load . . . 63
6.12.2.6 load_file_close . . . 64
6.12.2.7 load_file_open . . . 64
6.12.2.8 load_file_read . . . 64
6.12.2.9 save . . . 65
6.12.3 Member Data Documentation . . . 65
6.12.3.1 DEFAULT_FMT . . . 65
6.12.3.2 FMT_NPT_A . . . 65
6.12.3.3 FMT_NPT_B . . . 65
6.12.3.4 FMT_STL_A . . . 65
6.12.3.5 FMT_STL_AA . . . 66
6.12.3.6 FMT_STL_B . . . 66
6.12.3.7 FMT_STL_BB . . . 66
6.13 PolylibNS::Polylib Class Reference . . . 66
6.13.1 Detailed Description . . . 68
6.13.2 Constructor & Destructor Documentation . . . 68
6.13.2.1 Polylib . . . 68
6.13.2.2 ∼Polylib . . . 68
6.13.3 Member Function Documentation . . . 68
6.13.3.1 add_pg_list . . . 68
6.13.3.2 allgather_ParallelAreaInfo . . . 68
6.13.3.3 check_group_name . . . 68
6.13.3.4 clearfilepath . . . 69

6.13.3.5 get_group . . . 69
6.13.3.6 get_group . . . 69
6.13.3.7 get_instance . . . 70
6.13.3.8 get_leaf_groups . . . 70
6.13.3.9 get_MPI_Comm . . . 70
6.13.3.10 get_MPI_myrank . . . 70
6.13.3.11 get_MPI_numproc . . . 70
6.13.3.12 get_myproc_area . . . 70
6.13.3.13 get_other_procs_area . . . 71
6.13.3.14 get_proc_area . . . 71
6.13.3.15 get_root_groups . . . 71
6.13.3.16 get_srch_mode . . . 71
6.13.3.17 getVersionInfo . . . 71
6.13.3.18 init_parallel_info . . . 71
6.13.3.19 init_parallel_info . . . 72
6.13.3.20 load . . . 72
6.13.3.21 load_polygons . . . 72
6.13.3.22 make_group_tree . . . 73
6.13.3.23 migrate . . . 73
6.13.3.24 move . . . 73
6.13.3.25 save . . . 73
6.13.3.26 save_at_rank . . . 74
6.13.3.27 save_config_file . . . 74
6.13.3.28 save_parallel . . . 75
6.13.3.29 search_nearest_polygon . . . 75
6.13.3.30 search_nearest_polygon . . . 75
6.13.3.31 search_polygons . . . 76
6.13.3.32 search_polygons . . . 76
6.13.3.33 set_max_memory_size_mb . . . 77
6.13.3.34 set_srch_mode . . . 77
6.13.3.35 setfilepath . . . 77
6.13.3.36 show_all_group_info . . . 78
6.13.3.37 show_group_hierarchy . . . 78
6.13.3.38 show_group_info . . . 78
6.13.3.39 show_group_name . . . 78
6.13.3.40 used_memory_size . . . 79
6.13.3.41 used_memory_size_mb . . . 79
6.13.4 Member Data Documentation . . . 79
6.13.4.1 m_comm . . . 79
6.13.4.2 m_exclusion_map_procs . . . 79

6.13.4.3 m_max_memory_size_mb . . . 79
6.13.4.4 m_myproc_area . . . 79
6.13.4.5 m_myrank . . . 79
6.13.4.6 m_neibour_procs_area . . . 79
6.13.4.7 m_numproc . . . 80
6.13.4.8 m_other_procs_area . . . 80
6.13.4.9 m_pg_list . . . 80
6.13.4.10 tp . . . 80
6.14 PolylibNS::PolylibMoveParams Class Reference . . . 80
6.14.1 Detailed Description . . . 80
6.14.2 Member Data Documentation . . . 80
6.14.2.1 m_current_step . . . 80
6.14.2.2 m_delta_t . . . 80
6.14.2.3 m_next_step . . . 81
6.14.2.4 m_params . . . 81
6.15 PolylibMoveParamsStruct Struct Reference . . . 81
6.15.1 Detailed Description . . . 81
6.15.2 Member Data Documentation . . . 81
6.15.2.1 m_current_step . . . 81
6.15.2.2 m_delta_t . . . 81
6.15.2.3 m_next_step . . . 81
6.15.2.4 m_params . . . 82
6.16 PolylibNS::Triangle Class Reference . . . 82
6.16.1 Detailed Description . . . 83
6.16.2 Constructor & Destructor Documentation . . . 83
6.16.2.1 Triangle . . . 83
6.16.2.2 Triangle . . . 83
6.16.2.3 Triangle . . . 83
6.16.2.4 Triangle . . . 84
6.16.2.5 Triangle . . . 84
6.16.2.6 ∼Triangle . . . 84
6.16.3 Member Function Documentation . . . 84
6.16.3.1 calc_area . . . 84
6.16.3.2 calc_normal . . . 85
6.16.3.3 create_unique_id . . . 85
6.16.3.4 get_area . . . 85
6.16.3.5 get_bbox . . . 85
6.16.3.6 get_exid . . . 85
6.16.3.7 get_id . . . 85
6.16.3.8 get_normal . . . 86

6.16.3.9 get_num_atrI . . . 86
6.16.3.10 get_num_atrR . . . 86
6.16.3.11 get_pAtrI . . . 86
6.16.3.12 get_pAtrR . . . 86
6.16.3.13 get_pl_type . . . 87
6.16.3.14 get_vertexes . . . 87
6.16.3.15 rescale . . . 87
6.16.3.16 serialize . . . 87
6.16.3.17 serialized_size . . . 87
6.16.3.18 set_exid . . . 88
6.16.3.19 set_id . . . 88
6.16.3.20 set_num_atr . . . 88
6.16.3.21 set_vertexes . . . 88
6.16.3.22 update . . . 88
6.16.3.23 used_memory_size . . . 89
6.16.4 Member Data Documentation . . . 89
6.16.4.1 m_area . . . 89
6.16.4.2 m_AtrI . . . 89
6.16.4.3 m_AtrR . . . 89
6.16.4.4 m_exid . . . 89
6.16.4.5 m_id . . . 89
6.16.4.6 m_normal . . . 89
6.16.4.7 m_numAtrI . . . 89
6.16.4.8 m_numAtrR . . . 90
6.16.4.9 m_vertex . . . 90
6.17 TriangleStruct Struct Reference . . . 90
6.17.1 Detailed Description . . . 90
6.17.2 Member Data Documentation . . . 90
6.17.2.1 normal . . . 90
6.17.2.2 vertex . . . 90
6.18 PolylibNS::UsrAtr Struct Reference . . . 90
6.18.1 Member Data Documentation . . . 91
6.18.1.1 key . . . 91
6.18.1.2 value . . . 91
6.19 PolylibNS::Vec2< T > Class Template Reference . . . 91
6.19.1 Detailed Description . . . 92
6.19.2 Constructor & Destructor Documentation . . . 92
6.19.2.1 Vec2 . . . 92
6.19.2.2 Vec2 . . . 92
6.19.2.3 Vec2 . . . 92

6.19.3 Member Function Documentation . . . 92
6.19.3.1 assign . . . 92
6.19.3.2 average . . . 92
6.19.3.3 length . . . 92
6.19.3.4 lengthSquared . . . 92
6.19.3.5 normalize . . . 92
6.19.3.6 normalize . . . 92
6.19.3.7 operator const T * . . . 92
6.19.3.8 operator T * . . . 92
6.19.3.9 operator!= . . . 92
6.19.3.10 operator* . . . 92
6.19.3.11 operator* . . . 92
6.19.3.12 operator*= . . . 92
6.19.3.13 operator*= . . . 92
6.19.3.14 operator+ . . . 93
6.19.3.15 operator+= . . . 93
6.19.3.16 operator- . . . 93
6.19.3.17 operator- . . . 93
6.19.3.18 operator-= . . . 93
6.19.3.19 operator/ . . . 93
6.19.3.20 operator/ . . . 93
6.19.3.21 operator/= . . . 93
6.19.3.22 operator/= . . . 93
6.19.3.23 operator== . . . 93
6.19.3.24 operator[] . . . 93
6.19.3.25 operator[] . . . 93
6.19.3.26 ptr . . . 93
6.19.3.27 ptr . . . 93
6.19.3.28 xaxis . . . 93
6.19.3.29 yaxis . . . 93
6.19.4 Member Data Documentation . . . 93
6.19.4.1 x . . . 93
6.19.4.2 y . . . 93
6.20 Vec3class::Vec3< T > Class Template Reference . . . 93
6.20.1 Constructor & Destructor Documentation . . . 94
6.20.1.1 Vec3 . . . 94
6.20.1.2 Vec3 . . . 94
6.20.1.3 Vec3 . . . 94
6.20.1.4 Vec3 . . . 94
6.20.2 Member Function Documentation . . . 95

6.20.2.1 assign . . . 95
6.20.2.2 average . . . 95
6.20.2.3 length . . . 95
6.20.2.4 lengthSquared . . . 95
6.20.2.5 normalize . . . 95
6.20.2.6 normalize . . . 95
6.20.2.7 operator const T * . . . 95
6.20.2.8 operator T * . . . 95
6.20.2.9 operator!= . . . 95
6.20.2.10 operator* . . . 95
6.20.2.11 operator* . . . 95
6.20.2.12 operator*= . . . 95
6.20.2.13 operator*= . . . 95
6.20.2.14 operator+ . . . 95
6.20.2.15 operator+= . . . 95
6.20.2.16 operator- . . . 95
6.20.2.17 operator- . . . 95
6.20.2.18 operator-= . . . 95
6.20.2.19 operator/ . . . 95
6.20.2.20 operator/ . . . 95
6.20.2.21 operator/= . . . 95
6.20.2.22 operator/= . . . 95
6.20.2.23 operator== . . . 95
6.20.2.24 operator[] . . . 96
6.20.2.25 operator[] . . . 96
6.20.2.26 ptr . . . 96
6.20.2.27 ptr . . . 96
6.20.2.28 xaxis . . . 96
6.20.2.29 yaxis . . . 96
6.20.2.30 zaxis . . . 96
6.20.3 Member Data Documentation . . . 96
6.20.3.1 x . . . 96
6.20.3.2 y . . . 96
6.20.3.3 z . . . 96
6.21 PolylibNS::VElement Class Reference . . . 96
6.21.1 Constructor & Destructor Documentation . . . 96
6.21.1.1 VElement . . . 96
6.21.1.2 ~VElement . . . 97
6.21.2 Member Function Documentation . . . 97
6.21.2.1 get_bbox . . . 97

6.21.2.2 get_pos . . . 97
6.21.2.3 get_triangle . . . 97
6.22 PolylibNS::VNode Class Reference . . . 97
6.22.1 Detailed Description . . . 97
6.22.2 Constructor & Destructor Documentation . . . 98
6.22.2.1 VNode . . . 98
6.22.2.2 ~VNode . . . 98
6.22.3 Member Function Documentation . . . 98
6.22.3.1 get_axis . . . 98
6.22.3.2 get_bbox . . . 98
6.22.3.3 get_bbox_search . . . 98
6.22.3.4 get_elements_num . . . 98
6.22.3.5 get_left . . . 98
6.22.3.6 get_right . . . 99
6.22.3.7 get_vlist . . . 99
6.22.3.8 is_leaf . . . 99
6.22.3.9 set_axis . . . 99
6.22.3.10 set_bbox . . . 99
6.22.3.11 set_bbox_search . . . 99
6.22.3.12 set_element . . . 99
6.22.3.13 split . . . 100
6.23 PolylibNS::VTree Class Reference . . . 100
6.23.1 Detailed Description . . . 100
6.23.2 Constructor & Destructor Documentation . . . 100
6.23.2.1 VTree . . . 100
6.23.2.2 ~VTree . . . 100
6.23.3 Member Function Documentation . . . 100
6.23.3.1 destroy . . . 100
6.23.3.2 memory_size . . . 101
6.23.3.3 search . . . 101
6.23.3.4 search . . . 101
6.23.3.5 search_nearest . . . 101
6.23.3.6 search_nearest_recursive . . . 101

7 File Documentation **103**

7.1 c_lang/CPolylib.h File Reference . . . 103
7.1.1 Function Documentation . . . 104
7.1.1.1 polylib_get_group_tag . . . 104
7.1.1.2 polylib_get_leaf_groups_tags . . . 104
7.1.1.3 polylib_get_root_groups_tags . . . 105

7.1.1.4 polylib_group_get_area . . . 105
7.1.1.5 polylib_group_get_atr . . . 105
7.1.1.6 polylib_group_get_children . . . 106
7.1.1.7 polylib_group_get_global_area . . . 106
7.1.1.8 polylib_group_get_movable . . . 106
7.1.1.9 polylib_group_get_name . . . 107
7.1.1.10 polylib_group_get_num_global_triangles . . . 107
7.1.1.11 polylib_group_get_num_triangles . . . 107
7.1.1.12 polylib_group_get_parent . . . 107
7.1.1.13 polylib_group_get_polygons_reduce_atrI . . . 108
7.1.1.14 polylib_group_get_polygons_reduce_atrR . . . 108
7.1.1.15 polylib_group_get_triangles . . . 108
7.1.1.16 polylib_group_set_atr . . . 109
7.1.1.17 polylib_group_set_movable . . . 109
7.1.1.18 polylib_group_set_move_func_c . . . 109
7.1.1.19 polylib_group_set_name . . . 109
7.1.1.20 polylib_group_set_need_rebuild . . . 110
7.1.1.21 polylib_group_set_num_polygon_atr . . . 110
7.1.1.22 polylib_init_parallel_info . . . 110
7.1.1.23 polylib_init_parallel_info2 . . . 110
7.1.1.24 polylib_instance . . . 111
7.1.1.25 polylib_load . . . 111
7.1.1.26 polylib_migrate . . . 111
7.1.1.27 polylib_move . . . 111
7.1.1.28 polylib_save . . . 112
7.1.1.29 polylib_search_nearest_polygon . . . 112
7.1.1.30 polylib_search_polygons . . . 112
7.1.1.31 polylib_search_polygons_npt . . . 113
7.1.1.32 polylib_search_polygons_triangle . . . 113
7.1.1.33 polylib_show_group_hierarchy . . . 114
7.1.1.34 polylib_show_group_info . . . 114
7.1.1.35 polylib_triangle_get_normal . . . 114
7.1.1.36 polylib_triangle_get_npatchParam . . . 114
7.1.1.37 polylib_triangle_get_num_atrI . . . 115
7.1.1.38 polylib_triangle_get_num_atrR . . . 115
7.1.1.39 polylib_triangle_get_pAtrI . . . 115
7.1.1.40 polylib_triangle_get_pAtrR . . . 115
7.1.1.41 polylib_triangle_get_pl_type . . . 116
7.1.1.42 polylib_triangle_get_vertexes . . . 116
7.1.1.43 polylib_triangle_set_npatchParam . . . 116

7.1.1.44 polylib_triangle_set_vertexes . . . 116
7.1.1.45 polylib_triangle_set_vertexes_npatch . . . 117
7.1.1.46 polylib_used_memory_size . . . 117
7.1.1.47 polylib_used_memory_size_mb . . . 117
7.2 common/BBox.h File Reference . . . 117
7.3 common/PolylibCommon.h File Reference . . . 118
7.3.1 Macro Definition Documentation . . . 118
7.3.1.1 PL_DBGOS . . . 118
7.3.1.2 PL_DBGOSH . . . 119
7.3.1.3 PL_ERROS . . . 119
7.3.1.4 PL_ERROSH . . . 119
7.3.1.5 REAL9_TO_VEC3_3 . . . 119
7.3.1.6 REAL_TO_VEC3 . . . 119
7.3.1.7 REAL_TO_VEC3_3 . . . 119
7.3.1.8 VEC3_3_TO_REAL . . . 119
7.3.1.9 VEC3_3_TO_REAL9 . . . 119
7.3.1.10 VEC3_TO_REAL . . . 120
7.4 common/PolylibDefine.h File Reference . . . 120
7.4.1 Macro Definition Documentation . . . 120
7.4.1.1 FILE_FMT_DEFAULT . . . 120
7.4.1.2 FILE_FMT_NPT_A . . . 120
7.4.1.3 FILE_FMT_NPT_B . . . 120
7.4.1.4 FILE_FMT_STL_A . . . 120
7.4.1.5 FILE_FMT_STL_AA . . . 121
7.4.1.6 FILE_FMT_STL_B . . . 121
7.4.1.7 FILE_FMT_STL_BB . . . 121
7.4.1.8 PL_ELM_TAG . . . 121
7.4.1.9 PL_GRP_TAG . . . 121
7.4.1.10 PL_INT_MAX . . . 121
7.4.1.11 PL_INT_MIN . . . 121
7.4.1.12 PL_MPI_REAL . . . 121
7.4.1.13 PL_NULL_TAG . . . 121
7.4.1.14 PL_REAL . . . 121
7.4.1.15 PL_REAL_MAX . . . 121
7.4.1.16 PL_REAL_MIN . . . 121
7.4.1.17 PL_TYPE_NPT . . . 121
7.4.1.18 PL_TYPE_TRIANGLE . . . 121
7.4.1.19 PL_TYPE_UNKNOWN . . . 121
7.4.2 Enumeration Type Documentation . . . 121
7.4.2.1 PL_OP_TYPE . . . 121

7.5 common/PolylibStat.h File Reference . . . 121
7.5.1 Enumeration Type Documentation . . . 122
7.5.1.1 POLYLIB_STAT . . . 122
7.6 common/tt.h File Reference . . . 122
7.6.1 Macro Definition Documentation . . . 123
7.6.1.1 GL_GLEXT_PROTOTYPES . . . 123
7.6.1.2 M_PI . . . 123
7.6.1.3 sqrtf . . . 123
7.6.2 Typedef Documentation . . . 123
7.6.2.1 uchar . . . 123
7.6.2.2 uint . . . 123
7.6.2.3 ulong . . . 123
7.6.2.4 ushort . . . 123
7.7 common/Vec2.h File Reference . . . 123
7.7.1 Macro Definition Documentation . . . 124
7.7.1.1 REAL_TYPE . . . 124
7.8 common/Vec3.h File Reference . . . 124
7.8.1 Detailed Description . . . 125
7.8.2 Macro Definition Documentation . . . 125
7.8.2.1 REAL_TYPE . . . 125
7.9 f_lang/FPolylib.h File Reference . . . 125
7.9.1 Macro Definition Documentation . . . 127
7.9.1.1 PL_FILE_NAME_LEN . . . 127
7.9.1.2 PL_FILE_PATH_LEN . . . 127
7.9.1.3 PL_FORMAT_LEN . . . 127
7.9.1.4 PL_GRP_ATR_LEN . . . 128
7.9.1.5 PL_GRP_NAME_LEN . . . 128
7.9.1.6 PL_GRP_PATH_LEN . . . 128
7.9.1.7 PL_STR_LEN . . . 128
7.9.1.8 POLYLIB_FALSE . . . 128
7.9.1.9 POLYLIB_TRUE . . . 128
7.9.2 Function Documentation . . . 128
7.9.2.1 fpolylib_get_group_tag_ . . . 128
7.9.2.2 fpolylib_get_leaf_groups_tags_ . . . 128
7.9.2.3 fpolylib_get_leaf_groups_tags_num_ . . . 128
7.9.2.4 fpolylib_get_root_groups_tags_ . . . 129
7.9.2.5 fpolylib_get_root_groups_tags_num_ . . . 129
7.9.2.6 fpolylib_group_get_area_ . . . 129
7.9.2.7 fpolylib_group_get_atr_ . . . 129
7.9.2.8 fpolylib_group_get_children_ . . . 129

7.9.2.9 fpolylib_group_get_children_num_ . . . 129
7.9.2.10 fpolylib_group_get_global_area_ . . . 130
7.9.2.11 fpolylib_group_get_movable_ . . . 130
7.9.2.12 fpolylib_group_get_name_ . . . 130
7.9.2.13 fpolylib_group_get_num_global_triangles_ . . . 130
7.9.2.14 fpolylib_group_get_num_triangles_ . . . 130
7.9.2.15 fpolylib_group_get_parent_ . . . 130
7.9.2.16 fpolylib_group_get_polygons_reduce_atrI_ . . . 131
7.9.2.17 fpolylib_group_get_polygons_reduce_atrR_ . . . 131
7.9.2.18 fpolylib_group_get_triangles_ . . . 131
7.9.2.19 fpolylib_group_set_atr_ . . . 131
7.9.2.20 fpolylib_group_set_movable_ . . . 131
7.9.2.21 fpolylib_group_set_name_ . . . 132
7.9.2.22 fpolylib_group_set_need_rebuild_ . . . 132
7.9.2.23 fpolylib_group_set_num_polygon_atr_ . . . 132
7.9.2.24 fpolylib_init_parallel_info2_ . . . 132
7.9.2.25 fpolylib_init_parallel_info_ . . . 132
7.9.2.26 fpolylib_instance_ . . . 133
7.9.2.27 fpolylib_load_ . . . 133
7.9.2.28 fpolylib_migrate_ . . . 134
7.9.2.29 fpolylib_move_ . . . 134
7.9.2.30 fpolylib_save_ . . . 134
7.9.2.31 fpolylib_search_nearest_polygon_ . . . 134
7.9.2.32 fpolylib_search_polygons_ . . . 135
7.9.2.33 fpolylib_search_polygons_num_ . . . 135
7.9.2.34 fpolylib_show_group_hierarchy_ . . . 136
7.9.2.35 fpolylib_show_group_info_ . . . 136
7.9.2.36 fpolylib_triangle_get_atrI_ . . . 136
7.9.2.37 fpolylib_triangle_get_atrR_ . . . 136
7.9.2.38 fpolylib_triangle_get_normal_ . . . 136
7.9.2.39 fpolylib_triangle_get_npatchParam2_ . . . 137
7.9.2.40 fpolylib_triangle_get_npatchParam_ . . . 137
7.9.2.41 fpolylib_triangle_get_num_atrI_ . . . 137
7.9.2.42 fpolylib_triangle_get_num_atrR_ . . . 137
7.9.2.43 fpolylib_triangle_get_pl_type_ . . . 138
7.9.2.44 fpolylib_triangle_get_vertexes_ . . . 138
7.9.2.45 fpolylib_triangle_set_atrI_ . . . 138
7.9.2.46 fpolylib_triangle_set_atrR_ . . . 138
7.9.2.47 fpolylib_triangle_set_npatchParam2_ . . . 139
7.9.2.48 fpolylib_triangle_set_npatchParam_ . . . 139

7.9.2.49 fpolylib_triangle_set_vertexes_ . . . 139
7.9.2.50 fpolylib_triangle_set_vertexes_npatch2_ . . . 139
7.9.2.51 fpolylib_triangle_set_vertexes_npatch_ . . . 140
7.9.2.52 fpolylib_used_memory_size_ . . . 140
7.9.2.53 fpolylib_used_memory_size_mb_ . . . 140
7.10 f_lang/FPolylib_define.inc File Reference . . . 141
7.11 f_lang/FPolylib_define_f77.inc File Reference . . . 141
7.12 f_lang/FPolylib_precision.inc File Reference . . . 141
7.13 f_lang/FPolylib_precision_def.inc File Reference . . . 141
7.14 f_lang/FPolylib_prototype.inc File Reference . . . 141
7.15 f_lang/FPolylib_prototype_f77_double.inc File Reference . . . 141
7.16 f_lang/FPolylib_prototype_f77_float.inc File Reference . . . 141
7.17 f_lang/FPolylib_prototype_f77_integer.inc File Reference . . . 141
7.18 f_lang/FPolylib_struct.inc File Reference . . . 141
7.19 f_lang/FPolylib_User.inc File Reference . . . 141
7.20 f_lang/FPolylib_User_f77.inc File Reference . . . 141
7.21 file_io/FileIO_func.h File Reference . . . 141
7.22 file_io/PolygonIO.h File Reference . . . 142
7.23 groups/PolygonGroup.h File Reference . . . 142
7.24 groups/VTree.h File Reference . . . 143
7.25 polygons/NptTriangle.h File Reference . . . 143
7.26 polygons/Triangle.h File Reference . . . 143
7.27 Polylib.h File Reference . . . 144
7.28 Polylib_func.h File Reference . . . 144
7.28.1 Macro Definition Documentation . . . 145
7.28.1.1 INLINE . . . 145
7.29 polyVersion.h File Reference . . . 145
7.29.1 Detailed Description . . . 145
7.29.2 Macro Definition Documentation . . . 145
7.29.2.1 PL_REVISION . . . 145
7.29.2.2 PL_VERSION_NO . . . 145
7.30 util/time.h File Reference . . . 146

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

PolylibNS 9
Vec3class 22

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

PolylibNS::BBox . . . 25
PolylibNS::CalcAreaInfo . . . 28
FParallelBboxStruct . . . 29
PolylibNS::NpatchParam . . . 29
NpatchParamStruct . . . 31
NptTriangleStruct . . . 38
PolylibNS::ParallelAreaInfo . . . 38
PolylibNS::ParallelBbox . . . 39
ParallelBboxStruct . . . 39
PolylibNS::PolygonGroup . . . 40
PolylibNS::PolygonIO . . . 62
PolylibNS::Polylib . . . 66
PolylibNS::PolylibMoveParams . . . 80
PolylibMoveParamsStruct . . . 81
PolylibNS::Triangle . . . 82
    PolylibNS::NptTriangle . . . 32
TriangleStruct . . . 90
PolylibNS::UsrAtr . . . 90
PolylibNS::Vec2< T > . . . 91
Vec3class::Vec3< T > . . . 93
Vec3class::Vec3< PL_REAL > . . . 93
PolylibNS::VElement . . . 96
PolylibNS::VNode . . . 97
PolylibNS::VTree . . . 100

Chapter 3

Class Index

3.1 Class List

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

PolylibNS::BBox 25
PolylibNS::CalcAreaInfo
　　計算領域情報。 28
FParallelBboxStruct 29
PolylibNS::NpatchParam 29
NpatchParamStruct 31
PolylibNS::NptTriangle 32
NptTriangleStruct 38
PolylibNS::ParallelAreaInfo
　　並列プロセス領域情報。 38
PolylibNS::ParallelBbox 39
ParallelBboxStruct 39
PolylibNS::PolygonGroup 40
PolylibNS::PolygonIO 62
PolylibNS::Polylib 66
PolylibNS::PolylibMoveParams 80
PolylibMoveParamsStruct 81
PolylibNS::Triangle 82
TriangleStruct 90
PolylibNS::UsrAtr 90
PolylibNS::Vec2< T > 91
Vec3class::Vec3< T > 93
PolylibNS::VElement 96
PolylibNS::VNode 97
PolylibNS::VTree 100

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Polylib.h . . . 144
Polylib_func.h . . . 144
polyVersion.h . . . 145
c_lang/CPolylib.h . . . 103
common/BBox.h . . . 117
common/PolylibCommon.h . . . 118
common/PolylibDefine.h . . . 120
common/PolylibStat.h . . . 121
common/tt.h . . . 122
common/Vec2.h . . . 123
common/Vec3.h
Version 1.1 2014-04-23 . . . 124
f_lang/FPolylib.h . . . 125
f_lang/FPolylib_define.inc . . . 141
f_lang/FPolylib_define_f77.inc . . . 141
f_lang/FPolylib_precision.inc . . . 141
f_lang/FPolylib_precision_def.inc . . . 141
f_lang/FPolylib_prototype.inc . . . 141
f_lang/FPolylib_prototype_f77_double.inc . . . 141
f_lang/FPolylib_prototype_f77_float.inc . . . 141
f_lang/FPolylib_prototype_f77_integer.inc . . . 141
f_lang/FPolylib_struct.inc . . . 141
f_lang/FPolylib_User.inc . . . 141
f_lang/FPolylib_User_f77.inc . . . 141
file_io/FileIO_func.h . . . 141
file_io/PolygonIO.h . . . 142
groups/PolygonGroup.h . . . 142
groups/VTree.h . . . 143
polygons/NptTriangle.h . . . 143
polygons/Triangle.h . . . 143
util/time.h . . . 146

Chapter 5

Namespace Documentation

5.1 PolylibNS Namespace Reference

Classes

- class BBox
- class Vec2
- class PolygonIO
- struct UsrAtr
- class PolygonGroup
- class VElement
- class VNode
- class VTree
- struct NpatchParam
- class NptTriangle
- class Triangle
- struct ParallelBbox
- struct CalcAreaInfo

  計算領域情報。
- struct ParallelAreaInfo

  並列プロセス領域情報。
- class PolylibMoveParams
- class Polylib

Typedefs

- typedef Vec2< unsigned char > Vec2uc
- typedef Vec2< int > Vec2i
- typedef Vec2< float > Vec2f
- typedef Vec2< float > Vec2r

Enumerations

- enum ID_FORMAT { ID_BIN, ID_ASCII }

Functions

- Vec2< float > operator* (float s, const Vec2< float > &v)
- float distanceSquared (const Vec2< float > &a, const Vec2< float > &b)
- float distance (const Vec2< float > &a, const Vec2< float > &b)
- template<typename T >
  std::istream & operator>> (std::istream &is, Vec2< T > &v)
- template<typename T >
  std::ostream & operator<< (std::ostream &os, const Vec2< T > &v)
- std::istream & operator>> (std::istream &is, Vec2uc &v)
- std::ostream & operator<< (std::ostream &os, const Vec2uc &v)
- bool lessVec2f (const Vec2< float > &a, const Vec2< float > &b)
- POLYLIB_STAT stl_a_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT stl_a_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT stl_a_save (std::vector< Triangle * > *tri_list, const std::string &fname)
- POLYLIB_STAT stl_b_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT stl_b_load_read_head (ifstream &ifs)
- POLYLIB_STAT stl_b_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT stl_b_save (std::vector< Triangle * > *tri_list, const std::string &fname)
- bool is_stl_a (const std::string &path)
- POLYLIB_STAT npt_a_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT npt_a_load_read_head (ifstream &ifs)
- POLYLIB_STAT npt_a_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT npt_a_save (std::vector< NptTriangle * > *tri_list, const std::string &fname)
- POLYLIB_STAT npt_b_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT npt_b_load_read_head (ifstream &ifs)
- POLYLIB_STAT npt_b_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT npt_b_save (std::vector< NptTriangle * > *tri_list, const std::string &fname)
- bool is_npt_a (const std::string &path)
- char * get_fname_fr_path (const std::string &path)
- char * get_ext_fr_path (const std::string &path)
- Triangle * deserialize_polygon (int pl_type, const char *pbuff)
- void copy_polygon (Triangle *tri, Triangle *&copy_tri)
- void copy_polygons (const std::vector< Triangle * > &tri_list, std::vector< Triangle * > &copy_tri_list)
- POLYLIB_STAT convert_polygons_to_npt (std::vector< Triangle * > &tri_list, std::vector< NptTriangle * > &npt_list)
- void convert_polygons_to_tri (std::vector< NptTriangle * > &npt_list, std::vector< Triangle * > &tri_list)
- int ** alloc_array_2d_int (int n1, int n2)
- PL_REAL ** alloc_array_2d_real (int n1, int n2)
- void free_array_2d (void **x)
- bool getrusage_sec (double *usr_time, double *sys_time, double *total)

Variables

- std::string gs_rankno

5.1.1 Typedef Documentation

5.1.1.1 typedef Vec2<float> PolylibNS::Vec2f

5.1.1.2 typedef Vec2<int> PolylibNS::Vec2i

5.1.1.3 typedef Vec2< float > PolylibNS::Vec2r

5.1.1.4 typedef Vec2<unsigned char> PolylibNS::Vec2uc

5.1.2 Enumeration Type Documentation

5.1.2.1 enum PolylibNS::ID_FORMAT

三角形ID ファイルフォーマット

Enumerator

ID_BIN バイナリ形式で入出力を行う。

ID_ASCII アスキー形式で入出力を行う。

5.1.3 Function Documentation

5.1.3.1 int** PolylibNS::alloc_array_2d_int (int *n1*, int *n2*) [inline]

２次元配列（整数）の領域確保 領域解放は free_array_2d() を使用すること

Parameters

in	*n1*	１次元目サイズ
in	*n2*	２次元目サイズ

Returns

２次元配列のポインタ

Attention

x[i][j] でアクセスする

5.1.3.2 PL_REAL** PolylibNS::alloc_array_2d_real (int *n1*, int *n2*) [inline]

２次元配列（実数）の領域確保 領域解放は free_array_2d() を使用すること

Parameters

in	*n1*	１次元目サイズ
in	*n2*	２次元目サイズ

Returns

２次元配列のポインタ

Attention

x[i][j] でアクセスする

5.1.3.3 POLYLIB_STAT PolylibNS::convert_polygons_to_npt (std::vector< Triangle * > & *tri_list*, std::vector< NptTriangle * > & *npt_list*) [inline]

ポリゴン vector の型変換 vector<Triangle*> から vector<NptTriangle*> に変換する

Parameters

in	*tri_list*	Triangle のリスト
out	*npt_list*	NptTriangle のリスト

Returns

POLYLIB_STAT で定義される値が返る

5.1.3.4 void PolylibNS::convert_polygons_to_tri (std::vector< NptTriangle * > & *npt_list*, std::vector< Triangle * > & *tri_list*) [inline]

ポリゴン vector の型変換 vector<NptTriangle*> から vector<Triangle*> に変換する

Parameters

in	*npt_list*	NptTriangle のリスト
out	*tri_list*	Triangle のリスト

5.1.3.5 void PolylibNS::copy_polygon (Triangle * *tri*, Triangle *& *copy_tri*) [inline]

ポリゴンの複製 ポリゴンの種別 (Triangle/NptTrinangle) を判別し、適正なポリゴンを生成する

Parameters

in	*tri*	複製元ポリゴン
out	*copy_tri*	複製ポリゴン

Attention

関数内でアロケーションするので、copy_tri 使用後 delete してください

5.1.3.6 void PolylibNS::copy_polygons (const std::vector< Triangle * > & *tri_list*, std::vector< Triangle * > & *copy_tri_list*) [inline]

ポリゴン（複数）の複製・追加 ポリゴンの種別 (Triangle/NptTrinangle) を判別し、適正なポリゴンを生成する

Parameters

in	*tri_list*	複製元ポリゴン（複数）
	in/out]	copy_tri_list 複製ポリゴン （複数） 複製したものが追加される

Attention

関数内でアロケーションするので、 使用後 copy_trias 内のポリゴンは delete してください ディープコピーしています。

5.1.3.7 Triangle* PolylibNS::deserialize_polygon (int *pl_type*, const char * *pbuff*) [inline]

ポリゴンのデシリアリズ シリアライズされたデータよりオブジェクトの生成を行う

Parameters

in	*pl_type*	ポリゴンタイプ (PL_TYPE_TRIANGLE/PL_TYPE_NPT)
in	*pbuff*	バッファ格納位置先頭ポインタ シリアリズされたデータ

Returns

生成したオブジェクト (Triangle/NptTriangle)

5.1.3.8 float PolylibNS::distance (const Vec2< float > & *a*, const Vec2< float > & *b*) [inline]

5.1.3.9 float PolylibNS::distanceSquared (const Vec2< float > & *a*, const Vec2< float > & *b*) [inline]

5.1.3.10 void PolylibNS::free_array_2d (void ** *x*) [inline]

２次元配列の領域解放 alloc_array_2d_*() でアロケーションした領域の解放

Parameters

in	*x*	２次元配列のポインタ

Returns

戻り値なし

5.1.3.11 char* PolylibNS::get_ext_fr_path (const std::string & *path*)

ファイルパスから拡張子のみを取得する

Parameters

in	ファイルパス	

Returns

拡張子

Attention

戻り値の char * は解放不要 内部で static で持っているため多重処理NG

5.1.3.12 char* PolylibNS::get_fname_fr_path (const std::string & *path*)

ファイルパスから名称 (拡張子を除いた部分) を取得する

Parameters

in	ファイルパス	

Returns

拡張子を除いた名称

Attention

戻り値の char * は解放不要 内部で static で持っているため多重処理NG

5.1.3.13 bool PolylibNS::getrusage_sec (double * *usr_time,* double * *sys_time,* double * *total*)

5.1.3.14 bool PolylibNS::is_npt_a (const std::string & *path*)

長田パッチファイルを読み込みバイナリかアスキーかを判定する。

Parameters

in	*NPT* ファイルのフルパス名。	

Returns

true:アスキー形式 / false:バイナリ形式。

5.1.3.15 bool PolylibNS::is_stl_a (const std::string & *path*)

STL ファイルを読み込みバイナリかアスキーかを判定する。

Parameters

in	*STL* ファイルのフルパス名。	

Returns

true:アスキー形式 / false:バイナリ形式。

5.1.3.16 bool PolylibNS::lessVec2f (const Vec2< float > & *a,* const Vec2< float > & *b*) [inline]

5.1.3.17 POLYLIB_STAT PolylibNS::npt_a_load (std::vector< Triangle * > * *tri_list,* const std::string & *fname,* int * *num_tri,* PL_REAL *scale* = 1.0)

ASCII モードの長田パッチファイルを読み込み、tri_list に三角形ポリゴン情報を設定する。

Parameters

in,out	*tri_list*	三角形ポリゴンリストの領域 出力は追加される
in	*fname*	NPT ファイル名 //
in,out	*total*	ポリゴンID の通番
out	*num_tri*	STL ファイル内のポリゴン数

Returns

POLYLIB_STAT で定義される値が返る

Attention

ポリゴンID はシステム内で自動で採番される num_tri は tri_list 全体の個数でないことに注意

5.1.3.18 POLYLIB_STAT PolylibNS::npt_a_load_read (ifstream & *ifs,* std::vector< Triangle * > & *tri_list,* int *num_read,* int & *num_tri,* bool & *eof,* PL_REAL *scale* = 1.0)

ASCII モードの長田パッチファイルからポリゴン指定個数分読み込み

Parameters

in	*ifs*	入力ファイルストリーム

in,out	*tri_list*	三角形ポリゴンのリスト (出力内容)
in	*num_read*	読み込み指定数 EOF に達すれば途中まで読み込まれる -1 の時、読み込み数制限なし（eof まで読む）
out	*num_tri*	実際に読み込んだ数
out	*eof*	ファイル終了フラグ (end of file)
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンID はシステム内で自動で採番される

5.1.3.19 POLYLIB_STAT PolylibNS::npt_a_load_read_head (ifstream & *ifs*)

ASCII モードの長田パッチファイルのヘッダ部を読み飛ばす

Parameters

in	*ifs*	入力ファイルストリーム

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンを分割して読みだす時は npt_a_load_read() にて 順次呼び出して行くが先頭のヘッダ部は npt_a_load_read()では読み出せないため、先に読んでおく

5.1.3.20 POLYLIB_STAT PolylibNS::npt_a_save (std::vector< NptTriangle * > * *tri_list*, const std::string & *fname*)

三角形ポリゴン情報をASCII モードで長田パッチファイルに書き出す。

Parameters

in	*tri_list*	三角形ポリゴン情報
in	*fname*	NPT ファイル名

Returns

POLYLIB_STAT で定義される値が返る。

5.1.3.21 POLYLIB_STAT PolylibNS::npt_b_load (std::vector< Triangle * > * *tri_list*, const std::string & *fname*, int * *num_tri*, PL_REAL *scale* = 1.0)

バイナリモードの長田パッチファイルを読み込み、tri_list に三角形ポリゴン情報を設定 する。

Parameters

in,out	*tri_list*	三角形ポリゴンリストの領域。 出力は追加される
in	*fname*	ファイル名。
out	*num_tri*	STL ファイル内のポリゴン数

Returns

POLYLIB_STAT で定義される値が返る

Attention

ポリゴンID はシステム内で自動で採番される num_tri は tri_list 全体の個数でないことに注意

5.1.3.22 POLYLIB_STAT PolylibNS::npt_b_load_read (ifstream & *ifs*, std::vector< Triangle * > & *tri_list*, int *num_read*, int & *num_tri*, bool & *eof*, PL_REAL *scale* = 1.0)

バイナリモードの長田パッチファイルからポリゴン指定個数分読み込み

Parameters

in	*ifs*	入力ファイルストリーム
in,out	*tri_list*	三角形ポリゴンのリスト (出力内容)
in	*num_read*	読み込み指定数 EOF に達すれば途中まで読み込まれる -1 の時、読み込み数制限なし（eof まで読む）
out	*num_tri*	実際に読み込んだ数
out	*eof*	ファイル終了フラグ (end of file)
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンID はシステム内で自動で採番される

5.1.3.23 POLYLIB_STAT PolylibNS::npt_b_load_read_head (ifstream & *ifs*)

バイナリモードの長田パッチファイルのヘッダ部を読み飛ばす

Parameters

in	*ifs*	入力ファイルストリーム

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンを分割して読みだす時は npt_b_load_read() にて 順次呼び出して行くが先頭のヘッダ部は npt_b_load_read()では読み出せないため、先に読んでおく

5.1.3.24 POLYLIB_STAT PolylibNS::npt_b_save (std::vector< NptTriangle * > * *tri_list*, const std::string & *fname*)

三角形ポリゴン情報をバイナリモードで長田パッチファイルに書き出す。

Parameters

in	*tri_list*	三角形ポリゴン情報。
in	*fname*	NPT ファイル名。

Returns

POLYLIB_STAT で定義される値が返る。

5.1.3.25 Vec2< float > PolylibNS::operator*(float *s*, const Vec2< float > & *v*) [inline]

5.1.3.26 template<typename T > std::ostream& PolylibNS::operator<<(std::ostream & *os*, const Vec2< T > & *v*) [inline]

5.1.3.27 std::ostream& PolylibNS::operator<<(std::ostream & *os*, const Vec2uc & *v*) [inline]

5.1.3.28 template<typename T > std::istream& PolylibNS::operator>>(std::istream & *is*, Vec2< T > & *v*) [inline]

5.1.3.29 std::istream& PolylibNS::operator>>(std::istream & *is*, Vec2uc & *v*) [inline]

5.1.3.30 POLYLIB_STAT PolylibNS::stl_a_load(std::vector< Triangle * > * *tri_list*, const std::string & *fname*, int * *num_tri*, PL_REAL *scale* = 1.0)

ASCII モードのSTL ファイルを読み込み、tri_list に三角形ポリゴン情報を設定する。

Parameters

in,out	*tri_list*	三角形ポリゴンリストの領域。 出力は追加される
in	*fname*	STL ファイル名。 //
in,out	*total*	ポリゴンID の通番
out	*num_tri*	STL ファイル内のポリゴン数

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンID はシステム内で自動で採番される num_tri は tri_list 全体の個数でないことに注意

5.1.3.31 POLYLIB_STAT PolylibNS::stl_a_load_read(ifstream & *ifs*, std::vector< Triangle * > & *tri_list*, int *num_read*, int & *num_tri*, bool & *eof*, PL_REAL *scale* = 1.0)

ASCII モードのSTL ファイルからポリゴン指定個数分読み込み

Parameters

in	*ifs*	入力ファイルストリーム
in,out	*tri_list*	三角形ポリゴンのリスト (出力内容)
in	*num_read*	読み込み指定数 EOF に達すれば途中まで読み込まれる -1 の時、読み込み数制限なし（eof まで読む）

out	*num_tri*	実際に読み込んだ数
out	*eof*	ファイル終了フラグ (end of file)
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンID はシステム内で自動で採番される

5.1.3.32 POLYLIB_STAT PolylibNS::stl_a_save (std::vector< Triangle * > * *tri_list*, const std::string & *fname*)

三角形ポリゴン情報をASCII モードでSTL ファイルに書き出す。

Parameters

in	*tri_list*	三角形ポリゴン情報。
in	*fname*	STL ファイル名。

Returns

POLYLIB_STAT で定義される値が返る。

5.1.3.33 POLYLIB_STAT PolylibNS::stl_b_load (std::vector< Triangle * > * *tri_list*, const std::string & *fname*, int * *num_tri*, PL_REAL *scale* = 1.0)

バイナリモードのSTL ファイルを読み込み、tri_list に三角形ポリゴン情報を設定 する。

Parameters

in,out	*tri_list*	三角形ポリゴンリストの領域。 出力は追加される
in	*fname*	ファイル名。 //
in,out	*total*	ポリゴンID の通番
out	*num_tri*	STL ファイル内のポリゴン数

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンID はシステム内で自動で採番される num_tri は tri_list 全体の個数でないことに注意

5.1.3.34 POLYLIB_STAT PolylibNS::stl_b_load_read (ifstream & *ifs*, std::vector< Triangle * > & *tri_list*, int *num_read*, int & *num_tri*, bool & *eof*, PL_REAL *scale* = 1.0)

バイナリモードのSTL ファイルからポリゴン指定個数分読み込み

Parameters

in	*ifs*	入力ファイルストリーム
in,out	*tri_list*	三角形ポリゴンのリスト (出力内容)
in	*num_read*	読み込み指定数 EOF に達すれば途中まで読み込まれる -1 の時、読み込み数制限なし（eof まで読む）
out	*num_tri*	実際に読み込んだ数
out	*eof*	ファイル終了フラグ (end of file)
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンID はシステム内で自動で採番される ヘッダ部は、先に stl_b_load_read_head() で読みだしておくこと

5.1.3.35 POLYLIB_STAT PolylibNS::stl_b_load_read_head (ifstream & *ifs*)

バイナリモードのSTL ファイルのヘッダ部を読み飛ばす

Parameters

in	*ifs*	入力ファイルストリーム

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンを分割して読みだす時は stl_b_load_read() にて 順次呼び出して行くが先頭のヘッダ部は stl_b_load_read()では読み出せないため、先に読んでおく

5.1.3.36 POLYLIB_STAT PolylibNS::stl_b_save (std::vector< Triangle * > * *tri_list*, const std::string & *fname*)

三角形ポリゴン情報をバイナリモードでSTL ファイルに書き出す。

Parameters

in	*tri_list*	三角形ポリゴン情報。
in	*fname*	STL ファイル名。

Returns

POLYLIB_STAT で定義される値が返る。

5.1.4 Variable Documentation

5.1.4.1 std::string PolylibNS::gs_rankno

デバッグ出力用ランク番号グローバル文字列

5.2 Vec3class Namespace Reference

Classes

- class Vec3

Typedefs

- typedef Vec3< unsigned char > Vec3uc
- typedef Vec3< int > Vec3i
- typedef Vec3< float > Vec3f
- typedef Vec3< double > Vec3d
- typedef Vec3< float > Vec3r

Enumerations

- enum AxisEnum { AXIS_X = 0, AXIS_Y, AXIS_Z, AXIS_ERROR }

Functions

- template<typename T >
  Vec3< T > operator* (T s, const Vec3< T > &v)
- template<typename T >
  Vec3< T > multi (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  T dot (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  Vec3< T > cross (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  T distanceSquared (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  T distance (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  bool lessVec3f (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  std::istream & operator>> (std::istream &is, Vec3< T > &v)
- template<typename T >
  std::ostream & operator<< (std::ostream &os, const Vec3< T > &v)
- std::istream & operator>> (std::istream &is, Vec3uc &v)
- std::ostream & operator<< (std::ostream &os, const Vec3uc &v)

5.2.1 Typedef Documentation

5.2.1.1 typedef Vec3<double> Vec3class::Vec3d

5.2.1.2 typedef Vec3<float> Vec3class::Vec3f

5.2.1.3 typedef Vec3<int> Vec3class::Vec3i

5.2.1.4 typedef Vec3< float > Vec3class::Vec3r

5.2.1.5 typedef Vec3<unsigned char> Vec3class::Vec3uc

5.2.2 Enumeration Type Documentation

5.2.2.1 enum Vec3class::AxisEnum

Enumerator

AXIS_X

AXIS_Y

AXIS_Z

AXIS_ERROR

5.2.3 Function Documentation

5.2.3.1 template<typename T > Vec3<T> Vec3class::cross (const Vec3< T > & *a*, const Vec3< T > & *b*) [inline]

5.2.3.2 template<typename T > T Vec3class::distance (const Vec3< T > & *a*, const Vec3< T > & *b*) [inline]

5.2.3.3 template<typename T > T Vec3class::distanceSquared (const Vec3< T > & *a*, const Vec3< T > & *b*) [inline]

5.2.3.4 template<typename T > T Vec3class::dot (const Vec3< T > & *a*, const Vec3< T > & *b*) [inline]

5.2.3.5 template<typename T > bool Vec3class::lessVec3f (const Vec3< T > & *a*, const Vec3< T > & *b*) [inline]

5.2.3.6 template<typename T > Vec3<T> Vec3class::multi (const Vec3< T > & *a*, const Vec3< T > & *b*) [inline]

5.2.3.7 template<typename T > Vec3<T> Vec3class::operator* (T *s*, const Vec3< T > & *v*) [inline]

5.2.3.8 template<typename T > std::ostream& Vec3class::operator<< (std::ostream & *os*, const Vec3< T > & *v*) [inline]

5.2.3.9 std::ostream& Vec3class::operator<< (std::ostream & *os*, const Vec3uc & *v*) [inline]

5.2.3.10 template<typename T > std::istream& Vec3class::operator>> (std::istream & *is*, Vec3< T > & *v*) [inline]

5.2.3.11 std::istream& Vec3class::operator>> (std::istream & *is*, Vec3uc & *v*) [inline]

Chapter 6

Class Documentation

6.1 PolylibNS::BBox Class Reference

`#include <BBox.h>`

Public Member Functions

- BBox ()
- BBox (PL_REAL _minx, PL_REAL _miny, PL_REAL _minz, PL_REAL _maxx, PL_REAL _maxy, PL_REAL _maxz)
- BBox (PL_REAL _min[3], PL_REAL _max[3])
- BBox (const Vec3< PL_REAL > &_min, const Vec3< PL_REAL > &_max)
- void init ()
- void setMinMax (const Vec3< PL_REAL > &_min, const Vec3< PL_REAL > &_max)
- void add (const Vec3< PL_REAL > &v)
- Vec3< PL_REAL > getPoint (int idx) const
- Vec3< PL_REAL > center () const
- Vec3< PL_REAL > size () const
- PL_REAL xsize () const
- PL_REAL ysize () const
- PL_REAL zsize () const
- PL_REAL length (const AxisEnum &axis) const
- PL_REAL diameter () const
- AxisEnum getMaxAxis (PL_REAL &length) const
- bool contain (const Vec3< PL_REAL > &pos) const
- bool crossed (const BBox &bbox) const
- BBox getCrossedRegion (BBox &other_bbox) const
- Vec2< PL_REAL > vec3to2 (int axis_id, Vec3< PL_REAL > &v3) const
- void getFace (int axis_id, Vec3< PL_REAL > face[2][2]) const
- void getSide (int axis_id, Vec3< PL_REAL > side[4][2]) const

Public Attributes

- Vec3< PL_REAL > min

  メンバー変数
- Vec3< PL_REAL > max

6.1.1 Detailed Description

クラス:BBox Bounding Box を管理するクラス

6.1.2 Constructor & Destructor Documentation

6.1.2.1 PolylibNS::BBox::BBox ( ) [inline]

6.1.2.2 PolylibNS::BBox::BBox (PL_REAL *_minx*, PL_REAL *_miny*, PL_REAL *_minz*, PL_REAL *_maxx*, PL_REAL *_maxy*, PL_REAL *_maxz*) [inline]

6.1.2.3 PolylibNS::BBox::BBox (PL_REAL *_min[3]*, PL_REAL *_max[3]*) [inline]

6.1.2.4 PolylibNS::BBox::BBox (const Vec3< PL_REAL > & *_min*, const Vec3< PL_REAL > & *_max*) [inline]

6.1.3 Member Function Documentation

6.1.3.1 void PolylibNS::BBox::add (const Vec3< PL_REAL > & *v*) [inline]

6.1.3.2 Vec3<PL_REAL> PolylibNS::BBox::center ( ) const [inline]

6.1.3.3 bool PolylibNS::BBox::contain (const Vec3< PL_REAL > & *pos*) const [inline]

引数で与えられた点が、このBBox に含まれるかを判定する。

Parameters

in	*pos*	試行する点

Returns

含まれる場合は true。他は false。

6.1.3.4 bool PolylibNS::BBox::crossed (const BBox & *bbox*) const [inline]

BBox とBBox の交差判定を行う。 KD-Tree の交差判定と同じ。

Parameters

in	*bbox*	試行するBBox

Returns

交差する場合は true。他は false。

6.1.3.5 PL_REAL PolylibNS::BBox::diameter ( ) const [inline]

6.1.3.6 BBox PolylibNS::BBox::getCrossedRegion (BBox & *other_bbox*) const [inline]

BBox とBBox の重複領域の抽出を行う。 自身の面と他方の辺との交差判定を行う。

Parameters

in	*other_bbox*	試行するBBox

Returns

交差する場合は true。他は false。

6.1.3.7 void PolylibNS::BBox::getFace (int *axis_id,* Vec3< PL_REAL > *face[2][2]*) const [inline]

引数 axis_id(0=x,1=y,z=2) に垂直な、このBBox の面の対角点を返す。

Parameters

in	*axis_id*	軸番号。0=x 軸、1=y 軸、2=z 軸。
in	*face*	BBox の面の中で、軸に垂直な面の対角点。

6.1.3.8 AxisEnum PolylibNS::BBox::getMaxAxis (PL_REAL & *length*) const [inline]

6.1.3.9 Vec3<PL_REAL> PolylibNS::BBox::getPoint (int *idx*) const [inline]

6.1.3.10 void PolylibNS::BBox::getSide (int *axis_id,* Vec3< PL_REAL > *side[4][2]*) const [inline]

引数 axis_id(0=x,1=y,z=2) に平行な、このBBox の辺の端点を返す。

Parameters

in	*axis_id*	軸番号。0=x 軸、1=y 軸、2=z 軸。
in	*side*	BBox の辺の中で、軸に平行な辺の端点。

6.1.3.11 void PolylibNS::BBox::init ( ) [inline]

6.1.3.12 PL_REAL PolylibNS::BBox::length (const AxisEnum & *axis*) const [inline]

6.1.3.13 void PolylibNS::BBox::setMinMax (const Vec3< PL_REAL > & *_min,* const Vec3< PL_REAL > & *_max*) [inline]

6.1.3.14 Vec3<PL_REAL> PolylibNS::BBox::size ( ) const [inline]

6.1.3.15 Vec2<PL_REAL> PolylibNS::BBox::vec3to2 (int *axis_id,* Vec3< PL_REAL > & *v3*) const [inline]

引数 axis_id(0=x,1=y,z=2) に垂直な成分を詰めて返す。

6.1.3.16 PL_REAL PolylibNS::BBox::xsize ( ) const [inline]

6.1.3.17 PL_REAL PolylibNS::BBox::ysize ( ) const [inline]

6.1.3.18 PL_REAL PolylibNS::BBox::zsize ( ) const [inline]

6.1.4 Member Data Documentation

6.1.4.1 Vec3<PL_REAL> PolylibNS::BBox::max

6.1.4.2 Vec3<PL_REAL> PolylibNS::BBox::min

メンバー変数

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

- common/BBox.h

6.2 PolylibNS::CalcAreaInfo Struct Reference

計算領域情報。

`#include <Polylib.h>`

Public Attributes

- Vec3< PL_REAL > m_bpos
  基点座標
- Vec3< PL_REAL > m_bbsize
  計算領域のボクセル数
- Vec3< PL_REAL > m_gcsize
  ガイドセルのボクセル数
- Vec3< PL_REAL > m_dx
  ボクセル 1 辺の長さ
- Vec3< PL_REAL > m_gcell_min
  ガイドセルを含めた担当領域の最小位置
- Vec3< PL_REAL > m_gcell_max
  ガイドセルを含めた担当領域の最大位置
- BBox m_gcell_bbox

6.2.1 Detailed Description

計算領域情報。

6.2.2 Member Data Documentation

6.2.2.1 Vec3<PL_REAL> PolylibNS::CalcAreaInfo::m_bbsize

計算領域のボクセル数

6.2.2.2 Vec3<PL_REAL> PolylibNS::CalcAreaInfo::m_bpos

基点座標

6.2.2.3 Vec3<PL_REAL> PolylibNS::CalcAreaInfo::m_dx

ボクセル 1 辺の長さ

6.2.2.4 BBox PolylibNS::CalcAreaInfo::m_gcell_bbox

ガイドセルを含めたBounding Box m_gcell_min,m_gcell_max をBBox 化したのみ

6.2.2.5 Vec3<PL_REAL> PolylibNS::CalcAreaInfo::m_gcell_max

ガイドセルを含めた担当領域の最大位置

6.2.2.6 Vec3<PL_REAL> PolylibNS::CalcAreaInfo::m_gcell_min

ガイドセルを含めた担当領域の最小位置

6.2.2.7 Vec3<PL_REAL> PolylibNS::CalcAreaInfo::m_gcsize

ガイドセルのボクセル数

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

- Polylib.h

6.3 FParallelBboxStruct Struct Reference

`#include <FPolylib.h>`

Public Attributes

- PL_REAL bpos [3]
- int bbsize [3]
- int gcsize [3]
- PL_REAL dx [3]

6.3.1 Detailed Description

領域情報構造体 (Fortran 用)

6.3.2 Member Data Documentation

6.3.2.1 int FParallelBboxStruct::bbsize[3]

6.3.2.2 PL_REAL FParallelBboxStruct::bpos[3]

6.3.2.3 PL_REAL FParallelBboxStruct::dx[3]

6.3.2.4 int FParallelBboxStruct::gcsize[3]

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

- f_lang/FPolylib.h

6.4 PolylibNS::NpatchParam Struct Reference

`#include <NptTriangle.h>`

Public Attributes

- Vec3< PL_REAL > cp_side1_1
  長田パッチ曲面補間用制御点
- Vec3< PL_REAL > cp_side1_2
  p1->p2 辺の３次ベジェ制御点 *2*
- Vec3< PL_REAL > cp_side2_1
  p2->p3 辺の３次ベジェ制御点 *1*
- Vec3< PL_REAL > cp_side2_2
  p2->p3 辺の３次ベジェ制御点 *2*
- Vec3< PL_REAL > cp_side3_1
  p3->p1 辺の３次ベジェ制御点 *1*
- Vec3< PL_REAL > cp_side3_2
  p3->p1 辺の３次ベジェ制御点 *2*
- Vec3< PL_REAL > cp_center
  三角形中央の３次ベジェ制御点

6.4.1 Detailed Description

クラス:NpatchParam 長田パッチパラメータ（制御点情報）

6.4.2 Member Data Documentation

6.4.2.1 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_center

三角形中央の３次ベジェ制御点

6.4.2.2 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_side1_1

長田パッチ曲面補間用制御点
p1->p2 辺の３次ベジェ制御点 1

6.4.2.3 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_side1_2

p1->p2 辺の３次ベジェ制御点 2

6.4.2.4 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_side2_1

p2->p3 辺の３次ベジェ制御点 1

6.4.2.5 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_side2_2

p2->p3 辺の３次ベジェ制御点 2

6.4.2.6 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_side3_1

p3->p1 辺の３次ベジェ制御点 1

6.4.2.7 Vec3<PL_REAL> PolylibNS::NpatchParam::cp_side3_2

p3->p1 辺の 3 次ベジェ制御点 2

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

- polygons/NptTriangle.h

6.5 NpatchParamStruct Struct Reference

```
#include <CPolylib.h>
```

Public Attributes

- float cp_side1_1 [3]
  p1p2 辺の 3 次ベジェ制御点 *1*
- float cp_side1_2 [3]
  p1p2 辺の 3 次ベジェ制御点 *2*
- float cp_side2_1 [3]
  p2p3 辺の 3 次ベジェ制御点 *1*
- float cp_side2_2 [3]
  p2p3 辺の 3 次ベジェ制御点 *2*
- float cp_side3_1 [3]
  p3p1 辺の 3 次ベジェ制御点 *1*
- float cp_side3_2 [3]
  p3p1 辺の 3 次ベジェ制御点 *2*
- float cp_center [3]
  三角形中央の 3 次ベジェ制御点

6.5.1 Detailed Description

長田パッチパラメータ（曲面補間用制御点情報）

6.5.2 Member Data Documentation

6.5.2.1 float NpatchParamStruct::cp_center[3]

三角形中央の 3 次ベジェ制御点

6.5.2.2 float NpatchParamStruct::cp_side1_1[3]

p1p2 辺の 3 次ベジェ制御点 1

6.5.2.3 float NpatchParamStruct::cp_side1_2[3]

p1p2 辺の 3 次ベジェ制御点 2

6.5.2.4 float NpatchParamStruct::cp_side2_1[3]

p2p3 辺の 3 次ベジェ制御点 1

6.5.2.5 float NpatchParamStruct::cp_side2_2[3]

p2p3 辺の 3 次ベジェ制御点 2

6.5.2.6 float NpatchParamStruct::cp_side3_1[3]

p3p1 辺の 3 次ベジェ制御点 1

6.5.2.7 float NpatchParamStruct::cp_side3_2[3]

p3p1 辺の 3 次ベジェ制御点 2

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

- c_lang/CPolylib.h

6.6 PolylibNS::NptTriangle Class Reference

`#include <NptTriangle.h>`

Inherits PolylibNS::Triangle.

Public Member Functions

- NptTriangle ()
- NptTriangle (const NptTriangle &tria)
- NptTriangle (const Vec3< PL_REAL > vertex[3], const NpatchParam &param, long long int id=0, int num_atrI=0, int num_atrR=0, const int *atrI=NULL, const PL_REAL *atrR=NULL)
- NptTriangle (const char *pbuff)
- virtual ~NptTriangle ()
- virtual void rescale (PL_REAL scale)
- virtual size_t used_memory_size (void)
- virtual size_t serialized_size (void)
- virtual char * serialize (const char *pbuff)
- virtual BBox get_bbox (bool detail=false)
- virtual int get_pl_type (void)
- virtual void set_vertexes (const Vec3< PL_REAL > vertex[3], bool update_param, bool calc_area)
- void set_vertexes (const Vec3< PL_REAL > vertex[3], const NpatchParam &param, bool update_param, bool calc_area)
- void set_npatch_param (const NpatchParam &param)
- NpatchParam * get_npatch_param (void) const
- void correct (const Vec3< PL_REAL > &pos, Vec3< PL_REAL > &pos_o)
- void correct (PL_REAL eta, PL_REAL xi, Vec3< PL_REAL > &pos_o)

Protected Attributes

- NpatchParam m_npatchParam

  長田パッチパラメータ

Additional Inherited Members

6.6.1 Detailed Description

クラス:NptTriangle クラス

6.6.2 Constructor & Destructor Documentation

6.6.2.1 PolylibNS::NptTriangle::NptTriangle ( ) [inline]

コンストラクタ

6.6.2.2 PolylibNS::NptTriangle::NptTriangle (const NptTriangle & *tria*) [inline]

コピーコンストラクタ

6.6.2.3 PolylibNS::NptTriangle::NptTriangle (const Vec3< PL_REAL > *vertex[3],* const NpatchParam & *param,* long long int *id* = 0, int *num_atrI* = 0, int *num_atrR* = 0, const int * *atrI* = NULL, const PL_REAL * *atrR* = NULL) [inline]

コンストラクタ

Parameters

in	*vertex*	ポリゴンの頂点
in	*param*	長田パッチパラメータ
in	*id*	三角形ポリゴンID（ユニークな識別子） システム全体でユニークな識別子であること
in	*num_atrI*	ユーザ定義属性数（整数型）
in	*num_atrR*	ユーザ定義属性数（実数型）
in	*atrI*	ユーザ定義属性（整数型）
in	*atrR*	ユーザ定義属性（実数型）

Attention

id=0 の場合、ID は内部で採番される

6.6.2.4 PolylibNS::NptTriangle::NptTriangle (const char * *pbuff*)

コンストラクタ (deserialize) シリアライズされた通信バッファよりオブジェクトの生成を行う

Parameters

in	*pbuff*	バッファ格納先頭位置ポインタ シリアリズされたデータ

Attention

シリアライズは serialize() で行う

6.6.2.5 virtual PolylibNS::NptTriangle::~NptTriangle ( ) [inline], [virtual]

デストラクタ

6.6.3 Member Function Documentation

6.6.3.1 void PolylibNS::NptTriangle::correct (const Vec3< PL_REAL > & *pos*, Vec3< PL_REAL > & *pos_o*)

点の近似曲面補正

Parameters

in	*pos*	3 角形平面内の点
out	*pos_o*	曲面上に補正された座標

Returns

なし

6.6.3.2 void PolylibNS::NptTriangle::correct (PL_REAL *eta*, PL_REAL *xi*, Vec3< PL_REAL > & *pos_o*)

点の近似曲面補正

Parameters

in	*eta*	η パラメータ
in	*xi*	ξ パラメータ
out	*pos_o*	曲面上に補正された座標

Returns

なし

Attention

η と ξ のパラメータで 3 角形上の座標が決まる (参考) 頂点 1 eta = 0.0; xi = 0.0; 頂点 2 eta = 1.0; xi = 0.0; 頂点 3 eta = 1.0; xi = 1.0; 辺 1 の中点 eta = 0.5; xi = 0.0; 辺 2 の中点 eta = 1.0; xi = 0.5; 辺 3 の中点 eta = 0.5; xi = 0.5; 3 角形の重心 eta = 2.0/3.0, xi = 0.5∗2.0/3.0;

6.6.3.3 virtual BBox PolylibNS::NptTriangle::get_bbox (bool *detail* = false) [virtual]

Bounding box of this triangle

Parameters

in	*detail*	曲面補間したBounding box を返すか否か false: 3 角形の頂点にて決定 true: 曲面補間して決定

Returns

Bounding box

Reimplemented from PolylibNS::Triangle.

6.6.3.4 NpatchParam∗ PolylibNS::NptTriangle::get_npatch_param (void) const [inline]

長田パッチパラメータの取得

Parameters

in	*param*	長田パッチパラメータ

6.6.3.5 virtual int PolylibNS::NptTriangle::get_pl_type (void) [inline], [virtual]

ポリゴンタイプ取得

Returns

ポリゴンタイプ PL_TYPE_NPT

Reimplemented from PolylibNS::Triangle.

6.6.3.6 virtual void PolylibNS::NptTriangle::rescale (PL_REAL *scale*) [inline], [virtual]

ポリゴンのリスケール

Parameters

in	*scale*	スケール

Returns

戻り値なし

Reimplemented from PolylibNS::Triangle.

6.6.3.7 virtual char* PolylibNS::NptTriangle::serialize (const char * *pbuff*) [virtual]

ポリゴンのシリアリズ 1 ポリゴンをバッファに格納する

Parameters

out	*pbuff*	バッファ格納位置先頭ポインタ シリアリズされたデータ

Returns

バッファ格納位置Next ポインタ

Attention

デシリアリズはPolylibNS::deserialize_polygon() で行う 最終的にはコンストラクタ NptTriangle(pbuff) が呼び出される

Reimplemented from PolylibNS::Triangle.

6.6.3.8 virtual size_t PolylibNS::NptTriangle::serialized_size (void) [inline], [virtual]

ポリゴンのシリアリズした時のサイズ取得

Returns

シリアリズ後のサイズ

Reimplemented from PolylibNS::Triangle.

6.6.3.9 void PolylibNS::NptTriangle::set_npatch_param (const NpatchParam & *param*) [inline]

長田パッチパラメータの設定

Parameters

in	*param*	長田パッチパラメータ

6.6.3.10 virtual void PolylibNS::NptTriangle::set_vertexes (const Vec3< PL_REAL > *vertex[3],* bool *update_param,* bool *calc_area*) [virtual]

頂点を設定

Parameters

in	*vertex*	三角形の 3 頂点
in	*update_param*	面の法線ベクトル、長田パッチのパラメータ更新するか？
in	*calc_area*	面積を再計算するか？

Attention

長田パッチのパラメータ更新は遅いため 長田パッチとわかっている場合は、長田パッチを同時に 更新するタイプは良い

Reimplemented from PolylibNS::Triangle.

6.6.3.11 void PolylibNS::NptTriangle::set_vertexes (const Vec3< PL_REAL > *vertex[3],* const NpatchParam & *param,* bool *update_param,* bool *calc_area*)

頂点・長田パッチパラメータを設定

Parameters

in	*vertex*	三角形の 3 頂点
in	*param*	長田パッチパラメータ
in	*update_param*	面の法線ベクトルを更新するか？
in	*calc_area*	面積を再計算するか？

Attention

長田パッチのパラメータは絶対座標の制御点であるため 移動の時は、頂点と同様に移動させる必要がある

6.6.3.12 virtual size_t PolylibNS::NptTriangle::used_memory_size (void) [inline], [virtual]

ポリゴンの使用メモリサイズ取得 Polylib::used_memory_size()で使用する

Returns

使用メモリサイズ

Reimplemented from PolylibNS::Triangle.

6.6.4 Member Data Documentation

6.6.4.1 NpatchParam PolylibNS::NptTriangle::m_npatchParam [protected]

長田パッチパラメータ

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

- polygons/NptTriangle.h

6.7 NptTriangleStruct Struct Reference

`#include <CPolylib.h>`

Public Attributes

- float vertex [9]
  ３頂点座標
- NpatchParamStruct param
  長田パッチパラメータ

6.7.1 Detailed Description

長田パッチポリゴン情報構造体

6.7.2 Member Data Documentation

6.7.2.1 NpatchParamStruct NptTriangleStruct::param

長田パッチパラメータ

6.7.2.2 float NptTriangleStruct::vertex[9]

３頂点座標

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

- c_lang/CPolylib.h

6.8 PolylibNS::ParallelAreaInfo Struct Reference

並列プロセス領域情報。

`#include <Polylib.h>`

Public Attributes

- int m_rank
  ランク*No*
- std::vector< CalcAreaInfo > m_areas
  各ランク内の計算領域情報

6.8.1 Detailed Description

並列プロセス領域情報。

6.8.2 Member Data Documentation

6.8.2.1 std::vector<CalcAreaInfo> PolylibNS::ParallelAreaInfo::m_areas

各ランク内の計算領域情報

6.8.2.2 int PolylibNS::ParallelAreaInfo::m_rank

ランクNo

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

- Polylib.h

6.9 PolylibNS::ParallelBbox Struct Reference

`#include <Polylib.h>`

Public Attributes

- PL_REAL bpos [3]
- unsigned int bbsize [3]
- unsigned int gcsize [3]
- PL_REAL dx [3]

6.9.1 Detailed Description

クラス:ParallelAreaInfo 並列プロセス領域情報。

6.9.2 Member Data Documentation

6.9.2.1 unsigned int PolylibNS::ParallelBbox::bbsize[3]

6.9.2.2 PL_REAL PolylibNS::ParallelBbox::bpos[3]

6.9.2.3 PL_REAL PolylibNS::ParallelBbox::dx[3]

6.9.2.4 unsigned int PolylibNS::ParallelBbox::gcsize[3]

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

- Polylib.h

6.10 ParallelBboxStruct Struct Reference

`#include <CPolylib.h>`

Public Attributes

- float bpos [3]
- unsigned int bbsize [3]
- unsigned int gcsize [3]
- float dx [3]

6.10.1 Detailed Description

C 言語用Polylib

注意： C 言語API ではタグを操作（ハンドル）用として使用します。 タグには有効期間があります。セッション中で永続的に有効というわけではありません。 PolygonGroup タグ PolygonGroup の追加、削除、挿入があると該当する箇所以外も含めて全て無効となります。 Triangle タグ Triangle の追加、削除、挿入、ソートがあると該当する箇所以外も含めて全て無効となります。 並列環境でTriangle を移動した場合、migrate 処理を実行しますが この際に削除、挿入、ソートが行われるのでタグが全て無効となります。 領域情報構造体

6.10.2 Member Data Documentation

6.10.2.1 unsigned int ParallelBboxStruct::bbsize[3]

6.10.2.2 float ParallelBboxStruct::bpos[3]

6.10.2.3 float ParallelBboxStruct::dx[3]

6.10.2.4 unsigned int ParallelBboxStruct::gcsize[3]

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

- c_lang/CPolylib.h

6.11 PolylibNS::PolygonGroup Class Reference

`#include <PolygonGroup.h>`

Public Member Functions

- PolygonGroup ()
- virtual ~PolygonGroup ()
- POLYLIB_STAT init (const std::vector< Triangle * > *tri_list, bool clear=true)
- POLYLIB_STAT init (const std::vector< NptTriangle * > *tri_list, bool clear=true)
- void set_triangles_ptr (std::vector< Triangle * > *tri_list, bool build_tree=true)
- POLYLIB_STAT build_group_tree (Polylib *polylib, PolygonGroup *parent, TextParser *tp)
- POLYLIB_STAT build_polygon_tree ()
- POLYLIB_STAT load_polygons_file (PL_REAL scale=1.0)
- POLYLIB_STAT load_polygons_mem_reduced (PL_REAL scale, int size_mb)
- POLYLIB_STAT save_polygons_file (const std::string &rank_no, const std::string &extend, const std::string &format, std::map< std::string, std::string > &polygons_fname_map)
- POLYLIB_STAT save_polygons_file (const std::string &rank_no, const std::string &extend, const std::string &format, std::vector< Triangle * > *tri_list, std::map< std::string, std::string > &polygons_fname_map)
- POLYLIB_STAT scatter_polygons (void)
- POLYLIB_STAT scatter_polygons (std::vector< Triangle * > &tri_list_div_all, std::vector< Triangle * > &tri_list_div_local)
- POLYLIB_STAT gather_polygons (std::vector< Triangle * > &tri_list)
- POLYLIB_STAT move (PolylibMoveParams &params)
- POLYLIB_STAT set_move_func (void(*func)(PolygonGroup *, PolylibMoveParams *))
- POLYLIB_STAT set_move_func_c (void(*func)(PL_GRP_TAG,::PolylibMoveParamsStruct *))
- void set_need_rebuild (void)
- POLYLIB_STAT search (std::vector< Triangle * > &tri_list, const BBox &bbox, bool every) const

- POLYLIB_STAT search (std::vector< Triangle * > &tri_list, const std::vector< BBox > &bboxes, bool every, bool duplicate=false) const
- POLYLIB_STAT search (std::vector< NptTriangle * > &tri_list, const BBox &bbox, bool every) const
- POLYLIB_STAT search_nearest (Triangle *&tri, const Vec3< PL_REAL > &pos) const
- std::string acq_fullpath ()
- std::string acq_file_name ()
- POLYLIB_STAT get_inbounded_polygons (std::vector< Triangle * > &tri_list)
- POLYLIB_STAT erase_outbounded_polygons (void)
- POLYLIB_STAT search_outbounded (std::vector< Triangle * > &tri_list, std::vector< BBox > &neibour_bboxes, std::vector< long long int > &exclude_tria_ids)
- POLYLIB_STAT add_triangles (const std::vector< Triangle * > &tri_list)
- POLYLIB_STAT rebuild_polygons ()
- POLYLIB_STAT show_group_info (int irank=-1, bool detail=false)
- int get_group_num_tria (void)
- int get_group_num_global_tria (void)
- PL_REAL get_group_area (void)
- PL_REAL get_group_global_area (void)
- POLYLIB_STAT get_polygons_reduce_atrI (PL_OP_TYPE op, int atr_no, int &val)
- POLYLIB_STAT get_polygons_reduce_atrR (PL_OP_TYPE op, int atr_no, PL_REAL &val)
- POLYLIB_STAT rescale_polygons (PL_REAL scale)

  ポリゴンの縮尺変換＆*KD* 木再構築
- POLYLIB_STAT init_check_leaped ()
- POLYLIB_STAT check_leaped (std::vector< Vec3< PL_REAL > > &origin, std::vector< Vec3< PL_REAL > > &cell_size)
- int get_internal_id ()
- std::string get_name (void)
- void set_name (const std::string &name)
- bool get_movable ()
- void set_movable (bool movable)
- POLYLIB_STAT get_atr (std::string &key, std::string &val) const
- void set_atr (std::string &key, std::string &val)
- int get_num_polygon_atrI (void)
- int get_num_polygon_atrR (void)
- void set_num_polygon_atr (int num_atrI, int num_atrR)
- void set_parent_path (std::string ppath)
- std::string get_parent_path (void)
- void set_parent (PolygonGroup *p)
- PolygonGroup * get_parent (void)
- void set_children (std::vector< PolygonGroup * > &p)
- std::vector< PolygonGroup * > & get_children (void)
- void add_children (PolygonGroup *p)
- void remove_child (PolygonGroup *p)
- void set_file_name (std::map< std::string, std::string > fname)
- std::map< std::string, std::string > get_file_name () const
- std::vector< Triangle * > * get_triangles ()
- VTree * get_vtree ()
- size_t get_num_of_trias_before_move ()

Static Public Attributes

- static const char * ATT_NAME_CLASS

Protected Member Functions

- POLYLIB_STAT setup_attribute (Polylib ∗polylib, PolygonGroup ∗parent, TextParser ∗tp)
- POLYLIB_STAT setup_user_attribute (const std::string &node_label, TextParser ∗tp)
- bool is_far (Vec3< PL_REAL > &origin, Vec3< PL_REAL > &cell_size, Vec3< PL_REAL > &pos1, Vec3< PL_REAL > &pos2)

Protected Attributes

- int m_internal_id
  グループ*ID*。
- std::string m_name
  自グループ名。
- bool m_movable
  move メソッドにより移動するグループか？
- std::vector< UsrAtr > m_atr
  ユーザ定義属性
- std::string m_parent_path
  親グループのパス名。
- PolygonGroup ∗ m_parent
  親グループへのポインタ。
- std::vector< PolygonGroup ∗ > m_children
  子グループへのポインタリスト。
- std::map< std::string,
  std::string > m_polygon_files
- std::vector< Triangle ∗ > ∗ m_tri_list
  三角形ポリゴンのリスト
- BBox m_bbox
  全三角形ポリゴンを外包する*BoundingBox*
- VTree ∗ m_vtree
  KD 木クラス
- int m_max_elements
  MAX 要素数
- void(∗ m_move_func)(PolygonGroup ∗, PolylibMoveParams ∗)
- void(∗ m_move_func_c)(PL_GRP_TAG,::PolylibMoveParamsStruct ∗)
- bool m_need_rebuild
  KD 木の再構築が必要か？
- std::vector< Triangle ∗ > ∗ m_trias_before_move
  *move()*による移動前三角形一時保存リスト。

6.11.1 Detailed Description

クラス:PolygonGroup ポリゴングループを管理するクラスです。

6.11.2 Constructor & Destructor Documentation

6.11.2.1 PolylibNS::PolygonGroup::PolygonGroup ( )

コンストラクタ

6.11.2.2 virtual PolylibNS::PolygonGroup::~PolygonGroup ( ) [virtual]

デストラクタ

6.11.3 Member Function Documentation

6.11.3.1 std::string PolylibNS::PolygonGroup::acq_file_name ( )

カンマ区切りでSTL ファイル名リストを取得。

Returns

ファイル名リスト。

6.11.3.2 std::string PolylibNS::PolygonGroup::acq_fullpath ( )

PolygonGroup のフルパス名を取得する。

Returns

フルパス名。

6.11.3.3 void PolylibNS::PolygonGroup::add_children (PolygonGroup * *p*) [inline]

子グループを追加

Parameters

in	*p*	子グループ。

6.11.3.4 POLYLIB_STAT PolylibNS::PolygonGroup::add_triangles (const std::vector< Triangle * > & *tri_list*)

三角形リストの追加（ID 重複は追加されない）

Parameters

in	*tri_list*	三角形ポリゴンリストのポインタ。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

内部ID が重複した三角形は追加しない。KD 木の再構築はしない。

6.11.3.5 POLYLIB_STAT PolylibNS::PolygonGroup::build_group_tree (Polylib * *polylib,* PolygonGroup * *parent,* TextParser * *tp*)

PolygonGroup ツリーの作成。 設定ファイルの内容を再帰的に呼び出し、PolygonGroup ツリーを作成する。

Parameters

in	*polylib*	Polygon クラスのインスタンス
in	*parent*	親グループ
in	*tp*	TextParser のインスタンス

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.6 POLYLIB_STAT PolylibNS::PolygonGroup::build_polygon_tree ( )

ポリゴンの法線ベクトルの計算、面積の計算、KD 木の生成を行う。

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.7 POLYLIB_STAT PolylibNS::PolygonGroup::check_leaped (std::vector< Vec3< PL_REAL > > & *origin*, std::vector< Vec3< PL_REAL > > & *cell_size*)

move()メソッド実行により、頂点が隣接セルよりも遠くへ移動した三角形情報 を報告（後処理）。該当する三角形について、以下の情報を cerr へ出力する。 ・ポリゴングループID ・三角形ID ・移動前/後の頂点座標

Parameters

in	*origin*	計算領域起点座標
in	*cell_size*	ボクセルサイズ

Returns

POLYLIB_STAT で定義される値が返る。

Attention

本メソッドはデバッグ用です。 set_move_func()で登録した移動関数内で座標移動処理後に呼ぶこと。

6.11.3.8 POLYLIB_STAT PolylibNS::PolygonGroup::erase_outbounded_polygons (void)

自領域内ポリゴンのみ抽出してポリゴン情報を再構築

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンの load 処理内および migrate 処理後に実行する

6.11.3.9 POLYLIB_STAT PolylibNS::PolygonGroup::gather_polygons (std::vector< Triangle * > & *tri_list*)

ポリゴン情報集約 ポリゴングループ (複数）の各ランクに分散されているポリゴン情報をRank0 に集約する

Parameters

in	*pg*	ポリゴングループ

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.10 POLYLIB_STAT PolylibNS::PolygonGroup::get_atr (std::string & *key*, std::string & *val*) const

ポリゴングループのユーザ定義属性取得。

Parameters

in	*key*	キー
out	*val*	属性値

Returns

OK/NG NG:キーと属性のペアが登録されていない

6.11.3.11 std::vector<PolygonGroup*>& PolylibNS::PolygonGroup::get_children (void) [inline]

子グループを取得

Returns

子グループのリスト（参照型）

6.11.3.12 std::map<std::string, std::string> PolylibNS::PolygonGroup::get_file_name ( ) const [inline]

STL/NPT ファイル名とファイルフォーマットの対応マップ取得。

Returns

STL ファイル名とファイルフォーマットの対応マップ。

6.11.3.13 PL_REAL PolylibNS::PolygonGroup::get_group_area (void)

グループ内のポリゴンの面積を積算して返す 並列化されている場合は該当ランク内のみの面積

Attention

全プロセス通した面積が必要な場合は、 get_group_global_area()を使用する ポリゴンがない場合は 0.0 を返す

6.11.3.14 PL_REAL PolylibNS::PolygonGroup::get_group_global_area (void)

グループ内のポリゴンの面積 (global) を積算して返す 全プロセスを通算した面積 (重複ポリゴン分は無視される） 全プロセスに同じ値が返る ポリゴンがない場合は 0.0 を返す

6.11.3.15 int PolylibNS::PolygonGroup::get_group_num_global_tria (void)

ポリゴングループの要素数 (global) を返す

Returns

ポリゴングループの要素数 (global)

Attention

並列環境用 ポリゴンの重複を削除するための通信あり get_group_num_tria()よりも大幅に処理時間がかかることに注意

6.11.3.16 int PolylibNS::PolygonGroup::get_group_num_tria (void)

ポリゴングループの要素数を返す

Returns

ポリゴングループの要素数

Attention

並列環境の場合、各ランク担当分の要素数 全プロセス通した要素数が必要な場合は、get_group_num_global_tria()を使用する

6.11.3.17 POLYLIB_STAT PolylibNS::PolygonGroup::get_inbounded_polygons (std::vector< Triangle * > & *tri_list*)

自領域内ポリゴンのみ抽出してポリゴンを返す ポリゴン自体の複製は行っていない

Parameters

out	*tri_list*	自身の担当領域内のポリゴン

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.18 int PolylibNS::PolygonGroup::get_internal_id ( ) [inline]

ポリゴングループID を取得 システム内でユニークなID メンバー名修正 (m_id -> m_internal_id) 2010.10.20

Returns

ポリゴングループID。

6.11.3.19 bool PolylibNS::PolygonGroup::get_movable ( ) [inline]

移動対象フラグを取得。

Returns

移動対象フラグ。

6.11.3.20 std::string PolylibNS::PolygonGroup::get_name (void) [inline]

グループ名を取得。

Returns

グループ名。

6.11.3.21 size_t PolylibNS::PolygonGroup::get_num_of_trias_before_move ( ) [inline]

move()による移動前三角形一時保存リストの個数を取得。

Returns

一時保存リストサイズ。

6.11.3.22 int PolylibNS::PolygonGroup::get_num_polygon_atrI (void) [inline]

ポリゴン (Triangle/NptTriangle) のユーザ定義属性数（整数型）の取得

Returns

ユーザ定義属性数（整数型）

Attention

ポリゴングループにポリゴンが存在しない場合は 0 が返る 並列環境でポリゴンが存在しないランクがあるかもしれないので注意

6.11.3.23 int PolylibNS::PolygonGroup::get_num_polygon_atrR (void) [inline]

ポリゴン (Triangle/NptTriangle) のユーザ定義属性数（実数型）の取得

Returns

ユーザ定義属性数（実数型）

Attention

ポリゴングループにポリゴンが存在しない場合は 0 が返る 並列環境でポリゴンが存在しないランクがあるかもしれないので注意

6.11.3.24 PolygonGroup* PolylibNS::PolygonGroup::get_parent (void) [inline]

親グループを取得

Returns

親グループのポインタ。

6.11.3.25 std::string PolylibNS::PolygonGroup::get_parent_path (void) [inline]

親グループのフルパス名を取得。

Returns

親グループのフルパス名。

6.11.3.26 POLYLIB_STAT PolylibNS::PolygonGroup::get_polygons_reduce_atrl (PL_OP_TYPE *op*, int *atr_no*, int & *val*)

グループ内のポリゴン属性（整数）の集合演算値を返す 並列化されている場合は全プロセスを通した値 (PL_OP_SUM：重複ポリゴン分は無視される） 全プロセスに同じ値が返る

Parameters

in	*op*	演算種類　PL_OP_SUM/PL_OP_MAX/PL_OP_MIN
in	*atr_no*	ポリゴン整数属性の何番目か　0〜
out	*val*	属性値

Returns

POLYLIB_STAT で定義される値が返る。 ポリゴンが存在しない ポリゴン属性が存在しないなど

6.11.3.27 POLYLIB_STAT PolylibNS::PolygonGroup::get_polygons_reduce_atrR (PL_OP_TYPE *op*, int *atr_no*, PL_REAL & *val*)

グループ内のポリゴン属性（実数）の集合演算値を返す 並列化されている場合は全プロセスを通した値 (PL_OP_SUM：重複ポリゴン分は無視される） 全プロセスに同じ値が返る

Parameters

in	*op*	演算種類　PL_OP_SUM/PL_OP_MAX/PL_OP_MIN
in	*atr_no*	ポリゴン実数属性の何番目か　0〜
out	*val*	属性値

Returns

POLYLIB_STAT で定義される値が返る。 ポリゴンが存在しない ポリゴン属性が存在しないなど

6.11.3.28 std::vector<Triangle*>* PolylibNS::PolygonGroup::get_triangles ( ) [inline]

ポリゴンリストを取得。

Returns

三角形ポリゴンリスト Triangle/NptTriangle

6.11.3.29 VTree* PolylibNS::PolygonGroup::get_vtree ( ) [inline]

KD 木オブジェクトを取得。

Returns

KD 木ポリゴンリスト

Attention

ユーザは使用不可。Polylib::used_memory_size() のみ使用

6.11.3.30 POLYLIB_STAT PolylibNS::PolygonGroup::init (const std::vector< Triangle * > * *tri_list*, bool *clear* = true)

引数で与えられるポリゴンリストを複製し、KD 木の生成を行う。

Parameters

in	*tri_list*	設定するポリゴンリスト (Triangle)
in	*clear*	true:ポリゴン複製、面積計算、KD 木生成を行う。 false:面積計算、KD 木生成だけを行う。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴンがNptTriangle に限定できるときは 引数を std::vector<NptTriangle*> とした方が高速です。

6.11.3.31 POLYLIB_STAT PolylibNS::PolygonGroup::init (const std::vector< NptTriangle * > * *tri_list*, bool *clear* = true)

引数で与えられるポリゴンリストを複製し、KD 木の生成を行う。

Parameters

in	*tri_list*	設定する長田パッチポリゴンリスト (NptTriangle)
in	*clear*	true:ポリゴン複製、面積計算、KD 木生成を行う。 false:面積計算、KD 木生成だけを行う。

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.32 POLYLIB_STAT PolylibNS::PolygonGroup::init_check_leaped ( )

move()メソッド実行により、頂点が隣接セルよりも遠くへ移動した三角形情報 を報告（前処理）

Returns

POLYLIB_STAT で定義される値が返る。

Attention

本メソッドはデバッグ用です。 set_move_func()で登録した移動関数内で座標移動処理前に呼ぶこと。

6.11.3.33 bool PolylibNS::PolygonGroup::is_far (Vec3< PL_REAL > & *origin*, Vec3< PL_REAL > & *cell_size*, Vec3< PL_REAL > & *pos1*, Vec3< PL_REAL > & *pos2*) [protected]

2 点が隣接ボクセルよりも離れているか？

Parameters

in	*origin*	計算領域起点座標。
in	*cell_size*	ボクセルサイズ。
in	*pos1*	点 (1)。

in	*pos2*	点 (2)。

Returns

true:2 点が隣接ボクセルよりも離れている。

6.11.3.34 POLYLIB_STAT PolylibNS::PolygonGroup::load_polygons_file (PL_REAL *scale* = 1.0)

STL/NPT ファイルからポリゴン情報を読み込む (非メモリ削減版)

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.35 POLYLIB_STAT PolylibNS::PolygonGroup::load_polygons_mem_reduced (PL_REAL *scale,* int *size_mb*)

STL/NPT ファイルの読み込み (MPI メモリ削減版) ポリゴングループに設定されているSTL/NPT ファイルから ポリゴン情報を読み込む。読み込んだ後、KD 木の生成、法線の計算、面積の 計算を行う。

Parameters

in	*scale*	縮尺率
in	*size_mb*	使用可能メモリ (Mbyte)

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.36 POLYLIB_STAT PolylibNS::PolygonGroup::move (PolylibMoveParams & *params*)

三角形ポリゴン移動メソッド カスタマイズのためには set_move_func() で移動関数を登録しておくこと

Parameters

in	*params*	Polylib.h で宣言しているパラメタセットクラス。

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.37 POLYLIB_STAT PolylibNS::PolygonGroup::rebuild_polygons ( )

ポリゴン情報を再構築する。(KD 木の再構築をおこなう)

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.38 void PolylibNS::PolygonGroup::remove_child (PolygonGroup * *p*) [inline]

子グループを削除

Parameters

in	*p*	子グループ。

6.11.3.39 POLYLIB_STAT PolylibNS::PolygonGroup::rescale_polygons (PL_REAL *scale*)

ポリゴンの縮尺変換＆KD 木再構築

6.11.3.40 POLYLIB_STAT PolylibNS::PolygonGroup::save_polygons_file (const std::string & *rank_no,* const std::string & *extend,* const std::string & *format,* std::map< std::string, std::string > & *polygons_fname_map*)

ポリゴン情報をSTL/NPT ファイルに出力する。 TextParser 対応版

Parameters

in	*rank_no*	ファイル名に付加するランク番号。
in	*extend*	ファイル名に付加する自由文字列。
in	*format*	ファイルフォーマット。
in,out	*polygons_fname_map*	STL/NPT ファイル名とポリゴングループのパス

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ポリゴングループが持つポリゴンリストを使用する MPI 環境の場合、各ランク内のポリゴンリストを使う

6.11.3.41 POLYLIB_STAT PolylibNS::PolygonGroup::save_polygons_file (const std::string & *rank_no,* const std::string & *extend,* const std::string & *format,* std::vector< **Triangle** * > * *tri_list,* std::map< std::string, std::string > & *polygons_fname_map*)

ポリゴン情報をSTL/NPT ファイルに出力する TextParser 対応版

Parameters

in	*rank_no*	ファイル名に付加するランク番号。
in	*extend*	ファイル名に付加する自由文字列。
in	*format*	ファイルフォーマット。
in	*tri_list*	三角形ポリゴンリストの領域
in,out	*polygons_fname_map*	STL/NPT ファイル名とポリゴングループのパス

Returns

POLYLIB_STAT で定義される値が返る

Attention

ポリゴングループが持つポリゴンリストではなく、 外部から与えたポリゴンリストを使うことに注意
MPI 環境においてRank0 に集約したポリゴンリストを使う想定

6.11.3.42 POLYLIB_STAT PolylibNS::PolygonGroup::scatter_polygons (void)

ポリゴン情報分散 ポリゴングループ毎にRank0 に集約されているポリゴン情報を他ランクに分散する （非メモリ削減版）

Returns

POLYLIB_STAT で定義される値が返る。

Attention

(IN) ランク 0 : ポリゴングループに全ポリゴン設定 (IN) ランク 0 以外: ポリゴングループにポリゴン設定なし (OUT) 全ランク : ポリゴングループに担当領域内のポリゴン設定

6.11.3.43 POLYLIB_STAT PolylibNS::PolygonGroup::scatter_polygons (std::vector< Triangle * > & *tri_list_div_all*, std::vector< Triangle * > & *tri_list_div_local*)

ポリゴン情報分散 ポリゴングループ毎にRank0 に集約されているポリゴン情報を他ランクに分散する （メモリ削減版） メモリ制限により分割されたポリゴン毎の処理となる

Parameters

in,out	*tri_list_div_all*	ポリゴングループの分割された全ポリゴン rank0 : (in) 分割された全ポリゴン (out) 空リスト rank0 以外 : (in/out) 空リスト
out	*tri_list_div_local*	ポリゴングループの分割されたポリゴン (担当領域内のポリゴン） (in) 空リスト

Returns

POLYLIB_STAT で定義される値が返る。

Attention

(IN) 全ランク : ポリゴングループにポリゴン設定なし (OUT) 全ランク : ポリゴングループにポリゴン設定なし ランク 0 において担当領域内検索を行うため一時的に ポリゴングループにポリゴン tri_list_div_all が設定される

6.11.3.44 POLYLIB_STAT PolylibNS::PolygonGroup::search (std::vector< Triangle * > & *tri_list*, const BBox & *bbox*, bool *every*) const

指定矩形領域に含まれるポリゴンを抽出する。

Parameters

in,out	*tri_list*	検索されたポリゴンリスト (Triangle/NptTriangle)
in	*bbox*	矩形領域。
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

tri_list 内のポリゴン要素は、削除不可 tri_list 内のポリゴン要素はサブクラスのNptTriangle である可能性あり 内部でKD 木探索実施

6.11.3.45 POLYLIB_STAT PolylibNS::PolygonGroup::search (std::vector< Triangle * > & *tri_list*, const std::vector< BBox > & *bboxes*, bool *every*, bool *duplicate* = `false`) const

指定矩形領域（複数）に含まれるポリゴンを抽出する

Parameters

in,out	*tri_list*	検索されたポリゴンリスト (Triangle/NptTriangle)
in	*bbox*	矩形領域。
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。
in	*duplicate*	ポリゴンID 重複指定 false:重複なし true:重複の可能性あり

Returns

POLYLIB_STAT で定義される値が返る。

Attention

tri_list 内のポリゴン要素は、削除不可 tri_list 内のポリゴン要素はサブクラスのNptTriangle である可能性あり tri_list はクリアされず、検索されたポリゴンが追加される。 内部でKD 木探索実施 各ランクの担当領域が複数対応となったため追加 複数領域のため、単純に加算するとID が重複する可能性あり

6.11.3.46 POLYLIB_STAT PolylibNS::PolygonGroup::search (std::vector< NptTriangle * > & *tri_list*, const BBox & *bbox*, bool *every*) const

指定矩形領域に含まれるポリゴンを抽出する。

Parameters

in,out	*tri_list*	検索されたポリゴンリスト (NptTriangle)
in	*bbox*	矩形領域。
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出。 false:1 頂点でも検索領域に含まれるものを抽出。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

tri_list 内のポリゴン要素は、削除不可 tri_list はクリアされず、検索されたポリゴンが追加される 内部でKD 木探索実施

6.11.3.47 POLYLIB_STAT PolylibNS::PolygonGroup::search_nearest (Triangle *& *tri*, const Vec3< PL_REAL > & *pos*) const

指定位置に最も近いポリゴンを検索する

Parameters

out	*tri*	検索されたポリゴン (Triangle/NptTriangle) != 0 検索されたポリゴン
in	*pos*	指定位置

Returns

POLYLIB_STAT で定義される値

Attention

tri は削除不可 内部でKD 木探索実施

6.11.3.48 **POLYLIB_STAT** PolylibNS::PolygonGroup::search_outbounded (std::vector< **Triangle** * > & *tri_list,* std::vector< **BBox** > & *neibour_bboxes,* std::vector< long long int > & *exclude_tria_ids*)

PE 領域間移動する三角形ポリゴンリストの取得。

Parameters

in,out	*tri_list*	検索されたポリゴンリスト (Triangle)
in	*neibour_bboxes*	隣接PE 領域バウンディングボックス
in	*exclude_tria_ids*	領域移動対象外三角形ID リスト

Returns

検索結果三角形リスト

Attention

戻り値は使用後領域を解放すること

6.11.3.49 void PolylibNS::PolygonGroup::set_atr (std::string & *key*, std::string & *val*)

ポリゴングループのユーザ定義属性設定

Parameters

in	*key*	キー
in	*val*	属性値

Returns

なし

Attention

既に登録されていた場合、上書きする

6.11.3.50 void PolylibNS::PolygonGroup::set_children (std::vector< PolygonGroup * > & *p*) [inline]

子グループを設定

Parameters

in	*p*	子グループのリスト。

6.11.3.51 void PolylibNS::PolygonGroup::set_file_name (std::map< std::string, std::string > *fname*) [inline]

STL/NPT ファイル名とファイルフォーマットを設定。

Parameters

in	*fname*	STL ファイル名とファイルフォーマットの対応マップ。

6.11.3.52 void PolylibNS::PolygonGroup::set_movable (bool *movable*) [inline]

移動対象フラグを取得。

Parameters

in	移動対象フラグ。	

6.11.3.53 POLYLIB_STAT PolylibNS::PolygonGroup::set_move_func (void(∗)(PolygonGroup ∗, PolylibMoveParams ∗) *func*) [inline]

三角形ポリゴン移動関数登録 move()と違い、オブジェクトのインスタンス毎に登録が必要

Parameters

in	*func*	移動関数へのポインタ 引数のPolygonGroup∗ は this が設定されて呼ばれる

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.54 POLYLIB_STAT PolylibNS::PolygonGroup::set_move_func_c (void(∗)(PL_GRP_TAG,::PolylibMoveParamsStruct ∗) *func*) [inline]

三角形ポリゴン移動関数登録 move()と違い、オブジェクトのインスタンス毎に登録が必要

Parameters

in	*func_c*	移動関数 (C の関数）へのポインタ 引数のPolygonGroup∗ は this が設定されて呼ばれる

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.55 void PolylibNS::PolygonGroup::set_name (const std::string & *name*) [inline]

グループ名を設定。

Parameters

in	*name*	グループ名。

6.11.3.56 void PolylibNS::PolygonGroup::set_need_rebuild (void) [inline]

KD 木の再構築フラグの設定 ユーザ定義の移動関数内の最後で呼び出す

Returns

戻り値なし

6.11.3.57 void PolylibNS::PolygonGroup::set_num_polygon_atr (int *num_atrI*, int *num_atrR*) [inline]

ポリゴン (Triangle/NptTriangle) のユーザ定義属性数の設定 PolygonGroup 内の全ポリゴンに属性数を設定する

Parameters

in	*num_atrI*	ユーザ定義属性数（整数型）
in	*num_atrR*	ユーザ定義属性数（実数型）

Returns

なし

Attention

ポリゴングループにポリゴンが存在しない場合は何もしない 並列環境でポリゴンが存在しないランクがあるかもしれないので注意 エラーとはしていない

6.11.3.58 void PolylibNS::PolygonGroup::set_parent (PolygonGroup * *p*) [inline]

親グループを設定。

Parameters

in	*p*	親グループのポインタ。

6.11.3.59 void PolylibNS::PolygonGroup::set_parent_path (std::string *ppath*) [inline]

親グループのフルパス名を設定。

Parameters

in	*ppath*	親グループのフルパス名。

6.11.3.60 void PolylibNS::PolygonGroup::set_triangles_ptr (std::vector< Triangle * > * *tri_list*, bool *build_tree* = true)

ポリゴンのポインタの登録

Parameters

in	*tri_list*	設定するポリゴンリスト (Triangle)
in	*build_tree*	KD 木の作成フラグ build_polygon_tree()実行

Returns

なし

Attention

vector のポインタがコピーされる。 以降は、利用者側で vector および vector 内要素 (Triangle) を delete しないこと auto 変数等で自動で解放される変数に注意。 使用法が難しいので一般ユーザは使用不可

6.11.3.61 POLYLIB_STAT PolylibNS::PolygonGroup::setup_attribute (Polylib * *polylib*, PolygonGroup * *parent*, TextParser * *tp*) [protected]

設定ファイルから取得したPolygonGroup の属性情報をインスタンスにセットする。 ユーザ定義属性以外を設定する

"filepath" に関して、先に filepath が複数　(filepath[0]) が存在するかどうか をチェックして、複数ならばその処理を行い、filepath の処理は終了する。 複数でないことが分かったら、filepath が単体で存在するかをチェックして、 存在するならば、処理を行う。

Parameters

in	*polylib*	Polygon クラスのインスタンス。
in	*parent*	親グループ。
in	*tp*	TextParser クラスのインスタンス

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.62 POLYLIB_STAT PolylibNS::PolygonGroup::setup_user_attribute (const std::string & *node_label*, TextParser * *tp*) [protected]

設定ファイルから取得したPolygonGroup のユーザ定義属性情報をインスタンスにセットする。

Parameters

in	*node_label*	ユーザ定義属性ノードの名前
in	*tp*	TextParser クラスのインスタンス

Returns

POLYLIB_STAT で定義される値が返る。

6.11.3.63 POLYLIB_STAT PolylibNS::PolygonGroup::show_group_info (int *irank* = -1, bool *detail* = false)

グループ情報（ランク番号、親グループ名、自分のグループ名、ファイル名、 頂点数、各頂点のXYZ 座標値、法線ベクトルのXYZ 座標値、面積）を出力する。

Parameters

in	*irank*	ランクＮｏ (この値を出力しているのみ)

Returns

POLYLIB_STAT で定義される値が返る。

6.11.4 Member Data Documentation

6.11.4.1 const char* PolylibNS::PolygonGroup::ATT_NAME_CLASS [static]

config ファイルに記述するParam タグのクラス名 (value="...")。

6.11.4.2 std::vector<UsrAtr> PolylibNS::PolygonGroup::m_atr [protected]

ユーザ定義属性

6.11.4.3 BBox PolylibNS::PolygonGroup::m_bbox [protected]

全三角形ポリゴンを外包するBoundingBox

6.11.4.4 std::vector<PolygonGroup*> PolylibNS::PolygonGroup::m_children [protected]

子グループへのポインタリスト。

6.11.4.5 int PolylibNS::PolygonGroup::m_internal_id [protected]

グループID。

6.11.4.6 int PolylibNS::PolygonGroup::m_max_elements [protected]

MAX 要素数

6.11.4.7 bool PolylibNS::PolygonGroup::m_movable [protected]

move メソッドにより移動するグループか？

6.11.4.8 void(* PolylibNS::PolygonGroup::m_move_func)(PolygonGroup *, PolylibMoveParams *) [protected]

6.11.4.9 void(* PolylibNS::PolygonGroup::m_move_func_c)(PL_GRP_TAG,::PolylibMoveParamsStruct *) [protected]

6.11.4.10 std::string PolylibNS::PolygonGroup::m_name [protected]

自グループ名。

6.11.4.11 bool PolylibNS::PolygonGroup::m_need_rebuild [protected]

KD 木の再構築が必要か？

6.11.4.12 PolygonGroup* PolylibNS::PolygonGroup::m_parent [protected]

親グループへのポインタ。

6.11.4.13 std::string PolylibNS::PolygonGroup::m_parent_path [protected]

親グループのパス名。

6.11.4.14 std::map<std::string, std::string> PolylibNS::PolygonGroup::m_polygon_files [protected]

形状ファイル名とファイル形式 フィル名とファイル形式の対

6.11.4.15 std::vector<Triangle*>* PolylibNS::PolygonGroup::m_tri_list [protected]

三角形ポリゴンのリスト

6.11.4.16 std::vector<Triangle*>* PolylibNS::PolygonGroup::m_trias_before_move [protected]

move()による移動前三角形一時保存リスト。

6.11.4.17 VTree* PolylibNS::PolygonGroup::m_vtree [protected]

KD 木クラス

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

- groups/PolygonGroup.h

6.12 PolylibNS::PolygonIO Class Reference

`#include <PolygonIO.h>`

Static Public Member Functions

- static POLYLIB_STAT load (std::vector< Triangle * > *tri_list, const std::map< std::string, std::string > &fmap, PL_REAL scale=1.0)
- static POLYLIB_STAT load (std::vector< Triangle * > *tri_list, const std::string &fname, const std::string &fmt, PL_REAL scale=1.0)
- static POLYLIB_STAT load_file_open (std::ifstream &ifs, const std::string &fname, const std::string &fmt)
- static POLYLIB_STAT load_file_close (ifstream &ifs)
- static POLYLIB_STAT load_file_read (ifstream &ifs, const std::string &fmt, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- static POLYLIB_STAT save (std::vector< Triangle * > *tri_list, const std::string &fname, const std::string &fmt)
- static std::string input_file_format (const std::string &filename)
- static std::string get_extension_format (const std::string &fmt)
- static int get_polygon_type (const std::string &fmt)

Static Public Attributes

- static const std::string FMT_STL_A
  STL アスキーファイル
- static const std::string FMT_STL_AA
  STL アスキーファイル
- static const std::string FMT_STL_B
  STL バイナリファイル
- static const std::string FMT_STL_BB
  STL バイナリファイル
- static const std::string FMT_NPT_A
  長田パッチアスキーファイル
- static const std::string FMT_NPT_B
  長田パッチバイナリファイル
- static const std::string DEFAULT_FMT
  PolygonIO.cxx で定義している値

6.12.1 Detailed Description

クラス:PolygonIO 三角形ポリゴン入出力管理。

6.12.2 Member Function Documentation

6.12.2.1 static std::string PolylibNS::PolygonIO::get_extension_format (const std::string & *fmt*) [static]

ファイルフォーマットより拡張子を求める

Parameters

in	*filename*	ファイルフォーマット PolygonIO::FMT_STL_A 等

Returns

ファイル拡張子

6.12.2.2 static int PolylibNS::PolygonIO::get_polygon_type (const std::string & *fmt*) [static]

ファイルフォーマットよりポリゴンタイプを求める

Parameters

in	*filename*	ファイルフォーマット PolygonIO::FMT_STL_A 等

Returns

ポリゴンタイプ PL_TYPE_TRIANGLE/PL_TYPE_NPT/PL_TYPE_UNKNOWN

6.12.2.3 static std::string PolylibNS::PolygonIO::input_file_format (const std::string & *filename*) [static]

ファイル名を元に入力ファイルのフォーマットを取得する。

Parameters

in	*filename*	入力ファイル名。

Returns

判定したファイルフォーマット。

Attention

ファイル拡張子が"stl"の場合、ファイルを読み込んで判定する。

6.12.2.4 static POLYLIB_STAT PolylibNS::PolygonIO::load (std::vector< Triangle * > * *tri_list*, const std::map< std::string, std::string > & *fmap*, PL_REAL *scale* = 1.0) [static]

STL/NPT ファイルを読み込み、tri_list にセットする。 複数個のポリゴンファイルを読み込む

Parameters

in,out	*tri_list*	三角形ポリゴンリストの領域。
in	*fmap*	ファイル名、ファイルフォーマットのセット。 複数指定可
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

6.12.2.5 static POLYLIB_STAT PolylibNS::PolygonIO::load (std::vector< Triangle * > * *tri_list*, const std::string & *fname*, const std::string & *fmt*, PL_REAL *scale* = 1.0) [static]

STL/NPT ファイルを読み込み、tri_list にセットする。 1 個のポリゴンファイルを読み込む

Parameters

in,out	*tri_list*	三角形ポリゴンリストの領域。
in	*fname*	ファイル名
in	*fmt*	ファイルフォーマット
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

6.12.2.6 static POLYLIB_STAT PolylibNS::PolygonIO::load_file_close (ifstream & *ifs*) [static]

STL/NPT ファイルのClose ポリゴンを分割してロードする時に使用する

Parameters

in	*ifs*	入力ファイルストリーム

Returns

POLYLIB_STAT で定義される値が返る。

6.12.2.7 static POLYLIB_STAT PolylibNS::PolygonIO::load_file_open (std::ifstream & *ifs,* const std::string & *fname,* const std::string & *fmt*) [static]

STL/NPT ファイルのOpen ポリゴンを分割してロードする時に使用する

Parameters

out	*ifs*	入力ファイルストリーム
in	*fname*	ファイル名
in	*fmt*	ファイルフォーマット

Returns

POLYLIB_STAT で定義される値が返る。

Attention

STL とNPT のバイナリファイルに関しては 内部でヘッダ部の読み出しも行う

6.12.2.8 static POLYLIB_STAT PolylibNS::PolygonIO::load_file_read (ifstream & *ifs,* const std::string & *fmt,* std::vector< Triangle * > & *tri_list,* int *num_read,* int & *num_tri,* bool & *eof,* PL_REAL *scale* = 1.0) [static]

STL/NPT ファイルのRead（指定個数読み込む） ポリゴンを分割してロードする時に使用する

Parameters

in	*ifs*	入力ファイルストリーム
in	*fmt*	ファイルフォーマット
in,out	*tri_list*	三角形ポリゴンのリスト (出力内容)

in	*num_read*	読み込み指定数 EOF に達すれば途中まで読み込まれる -1 の時、読み込み数制限なし（eof まで読む）
out	*num_tri*	実際に読み込んだ数
out	*eof*	ファイル終了フラグ (end of file)
in	*scale*	スケール

Returns

POLYLIB_STAT で定義される値が返る。

6.12.2.9 static POLYLIB_STAT PolylibNS::PolygonIO::save (std::vector< Triangle * > * *tri_list,* const std::string & *fname,* const std::string & *fmt*) [static]

tri_list の内容をSTL 形式でファイルへ保存。

Parameters

in	*tri_list*	三角形ポリゴンのリスト (出力内容)
in	*fname*	ファイル名
in	*fmt*	ファイルフォーマット

Returns

POLYLIB_STAT で定義される値が返る

Attention

長田パッチの場合、std::vector<Triangle*>* に キャストすること static_cast ではコンパイルエラーとなる場合、旧キャストを使用 (std::vector<Triangle*>*)ptr

6.12.3 Member Data Documentation

6.12.3.1 const std::string PolylibNS::PolygonIO::DEFAULT_FMT [static]

PolygonIO.cxx で定義している値

6.12.3.2 const std::string PolylibNS::PolygonIO::FMT_NPT_A [static]

長田パッチアスキーファイル

6.12.3.3 const std::string PolylibNS::PolygonIO::FMT_NPT_B [static]

長田パッチバイナリファイル

6.12.3.4 const std::string PolylibNS::PolygonIO::FMT_STL_A [static]

STL アスキーファイル

STL/NPT ファイルのフォーマット種別

Attention

STL/NPT ファイルの拡張子とは異なるので注意すること。

6.12.3.5 const std::string PolylibNS::PolygonIO::FMT_STL_AA [static]

STL アスキーファイル

6.12.3.6 const std::string PolylibNS::PolygonIO::FMT_STL_B [static]

STL バイナリファイル

6.12.3.7 const std::string PolylibNS::PolygonIO::FMT_STL_BB [static]

STL バイナリファイル

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

- file_io/PolygonIO.h

6.13 PolylibNS::Polylib Class Reference

`#include <Polylib.h>`

Public Member Functions

- MPI_Comm get_MPI_Comm (void)
- int get_MPI_myrank (void)
- int get_MPI_numproc (void)
- POLYLIB_STAT init_parallel_info (MPI_Comm comm, PL_REAL bpos[3], unsigned int bbsize[3], unsigned int gcsize[3], PL_REAL dx[3])
- POLYLIB_STAT init_parallel_info (MPI_Comm comm, const std::vector< ParallelBbox > &bboxes)
- ParallelAreaInfo * get_myproc_area ()
- std::vector< ParallelAreaInfo > * get_other_procs_area ()
- POLYLIB_STAT load (const std::string config_name="polylib_config.tp", PL_REAL scale=1.0)
- POLYLIB_STAT save (std::string &config_name_out, const std::string &file_format, std::string extend="")
- POLYLIB_STAT save_parallel (std::string &config_name_out, const std::string &file_format, std::string extend="")
- POLYLIB_STAT move (PolylibMoveParams &params)
- POLYLIB_STAT migrate ()
- std::vector< PolygonGroup * > * get_root_groups () const
- std::vector< PolygonGroup * > * get_leaf_groups () const
- POLYLIB_STAT search_polygons (std::vector< Triangle * > &tri_list, const std::string &group_name, const Vec3< PL_REAL > &min_pos, const Vec3< PL_REAL > &max_pos, const bool every) const
- POLYLIB_STAT search_polygons (std::vector< NptTriangle * > &tri_list, const std::string &group_name, const Vec3< PL_REAL > &min_pos, const Vec3< PL_REAL > &max_pos, const bool every) const
- POLYLIB_STAT search_nearest_polygon (Triangle *&tri, const std::string &group_name, const Vec3< PL_REAL > &pos) const
- POLYLIB_STAT search_nearest_polygon (NptTriangle *&tri, const std::string &group_name, const Vec3< PL_REAL > &pos) const
- POLYLIB_STAT check_group_name (const std::string &pg_name, const std::string &parent_path)
- void add_pg_list (PolygonGroup *pg)
- void show_group_hierarchy (FILE *fp=NULL)
- POLYLIB_STAT show_group_info (const std::string &group_name, bool detail=false)
- void show_all_group_info (bool detail=false)
- size_t used_memory_size ()
- size_t used_memory_size_mb ()

- void set_max_memory_size_mb (int max_size_mb)
- PolygonGroup * get_group (const std::string &name) const
- std::string getVersionInfo ()

  バージョン番号の文字列を返す

Static Public Member Functions

- static Polylib * get_instance ()
- static void set_srch_mode (bool detail)
- static bool get_srch_mode ()

Protected Member Functions

- Polylib ()
- ~Polylib ()
- POLYLIB_STAT allgather_ParallelAreaInfo (ParallelAreaInfo &myproc_area, std::vector< ParallelAreaInfo > &all_procs_area)
- POLYLIB_STAT make_group_tree (TextParser *tp_ptr)
- POLYLIB_STAT load_polygons (PL_REAL scale=1.0)
- char * save_config_file (const std::string &rank_no, const std::string &extend, const std::string &format)
- POLYLIB_STAT clearfilepath (TextParser *tp_ptr)
- POLYLIB_STAT setfilepath (std::map< std::string, std::string > &polygons_fname_map)
- POLYLIB_STAT save_at_rank (std::string &config_name_out, int myrank, int maxrank, const std::string &extend, const std::string &file_format)
- void show_group_name (PolygonGroup *p, std::string tab, FILE *fp)
- PolygonGroup * get_group (int internal_id) const
- ParallelAreaInfo * get_proc_area (int rank)

Protected Attributes

- std::vector< PolygonGroup * > m_pg_list

  全てのポリゴングループリスト
- TextParser * tp

  TextParser へのポインタ
- ParallelAreaInfo m_myproc_area

  自*PE* 担当領域情報
- std::vector< ParallelAreaInfo > m_other_procs_area

  自*PE* を除く全*PE* 担当領域情報リスト
- std::vector< ParallelAreaInfo > m_neibour_procs_area

  隣接*PE* 担当領域情報リスト
- std::vector< std::map< int, std::vector< long long int > > > m_exclusion_map_procs
- int m_myrank

  自プロセスのランク*No*
- int m_numproc

  全プロセス数
- MPI_Comm m_comm

  MPI コミュニケーター
- int m_max_memory_size_mb

  MAX メモリーサイズ *(MB)*

6.13.1 Detailed Description

クラス:Polylib ポリゴンを管理する為のクラスライブラリです。

6.13.2 Constructor & Destructor Documentation

6.13.2.1 PolylibNS::Polylib::Polylib ( ) [protected]

コンストラクタ

Attention

singleton のため、子クラス以外からの呼び出し不可とする

6.13.2.2 PolylibNS::Polylib::~Polylib ( ) [protected]

デストラクタ

6.13.3 Member Function Documentation

6.13.3.1 void PolylibNS::Polylib::add_pg_list (PolygonGroup * *pg*)

PolygonGroup の追加。 本クラスが管理しているPolygonGroup のリストにPolygonGroup を追加する

Parameters

in	*pg*	PolygonGroup

6.13.3.2 POLYLIB_STAT PolylibNS::Polylib::allgather_ParallelAreaInfo (ParallelAreaInfo & *myproc_area*, std::vector< ParallelAreaInfo > & *all_procs_area*) [protected]

全PE の担当領域の収集

Parameters

in	*myproc_area*	自PE の担当領域
out	*all_procs_area*	全PE の担当領域

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.3 POLYLIB_STAT PolylibNS::Polylib::check_group_name (const std::string & *pg_name*, const std::string & *parent_path*)

引数のグループ名が既存グループと重複しないかチェック。

Parameters

in	*pg_name*	グループ名

in	*parent_path*	親グループまでのフルパス

Returns

POLYLIB_STAT で定義される値が返る。

Attention

Polylib 内部で使用する関数であり、通常は利用者が用いるもの ではない。

6.13.3.4 POLYLIB_STAT PolylibNS::Polylib::clearfilepath (TextParser * *tp_ptr*) [protected]

TextParser 内部データから "filepath" "filepath[*]" というリーフを すべて削除する.

recursive の動作の為、引数に tp_ptr が必要

Parameters

in	*tp_ptr*	TextParser へのポインタ.

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.5 PolygonGroup* PolylibNS::Polylib::get_group (const std::string & *name*) const

グループの取得。 name で与えられた名前のPolygonGroup を返す。

Parameters

in	*name*	グループ名

Returns

ポリゴングループクラスのポインタ。エラー時はNULL が返る。

Attention

オーバーロードメソッドあり。

6.13.3.6 PolygonGroup* PolylibNS::Polylib::get_group (int *internal_id*) const [protected]

グループの取得。 internal_id で与えられた m_internal_id を持つPolygonGroup を返す。

Parameters

in	*internal_id*	ポリゴングループID

Returns

ポリゴングループクラスのポインタ。エラー時はNULL が返る。

6.13.3.7 static Polylib* PolylibNS::Polylib::get_instance ( ) [static]

singleton のPolylib インスタンス取得。 デフォルトのFactory クラスであるPolygonGroupFactory を使用してインスタンス を生成する。

Returns

Polylib クラスのインスタンス。

Attention

呼び出し側で delete はできません。

6.13.3.8 std::vector<PolygonGroup *>* PolylibNS::Polylib::get_leaf_groups ( ) const

リーフPolygonGroup リストの取得。 PolygonGroup ツリーの末端ノード（リーフ）をリスト化する。

Returns

リーフPolygonGroup の vector. 返却したPolygonGroup は削除不可。vector は要削除。

6.13.3.9 MPI_Comm PolylibNS::Polylib::get_MPI_Comm (void) [inline]

MPI のコミュニケータを返す

Returns

MPI のコミュニケータ PolygonGroup 内のポリゴンの集合演算を行うときに使用する

6.13.3.10 int PolylibNS::Polylib::get_MPI_myrank (void) [inline]

MPI のランクＮｏを返す

Returns

MPI のランクＮｏ PolygonGroup 内の scatter/gather で使用する

6.13.3.11 int PolylibNS::Polylib::get_MPI_numproc (void) [inline]

MPI の並列プロセス数を返す

Returns

MPI の並列プロセス数 PolygonGroup 内の scatter/gather で使用する

6.13.3.12 ParallelAreaInfo* PolylibNS::Polylib::get_myproc_area ( ) [inline]

自PE 担当領域情報取得

Returns

自PE 領域情報

6.13.3.13 std::vector<ParallelAreaInfo>* PolylibNS::Polylib::get_other_procs_area ( ) [inline]

自PE を除く全PE 担当領域情報リスト取得

Returns

自PE を除く全PE 担当領域情報

6.13.3.14 ParallelAreaInfo* PolylibNS::Polylib::get_proc_area (int *rank*) [protected]

プロセス担当領域クラスのポインタを返す

Parameters

in	*rank*	ランクNo（自ランク以外）

Returns

プロセス担当領域クラスのポインタ

6.13.3.15 std::vector<PolygonGroup *>* PolylibNS::Polylib::get_root_groups ( ) const

PolygonGroup ツリーの最上位ノードの取得。

Returns

最上位ノードの vector。

Attention

返却したPolygonGroup は、削除不可。vector は要削除。

6.13.3.16 static bool PolylibNS::Polylib::get_srch_mode ( ) [static]

検索モードの取得 ポリゴン検索時にポリゴンを曲面補正して検索させるかどうかを指定する

Returns

検索モード true : 曲面補正が可能な場合、曲面補正して検索 false : 3 角形平面で検索

6.13.3.17 std::string PolylibNS::Polylib::getVersionInfo ( ) [inline]

バージョン番号の文字列を返す

6.13.3.18 POLYLIB_STAT PolylibNS::Polylib::init_parallel_info (MPI_Comm *comm*, PL_REAL *bpos[3]*, unsigned int *bbsize[3]*, unsigned int *gcsize[3]*, PL_REAL *dx[3]*)

並列計算関連情報の設定と初期化を行う。 (各ランクが 1 領域を担当している場合） 全 rank で各々設定を行い、その領域情報を全 rank へ配信する。

Parameters

in	*comm*	MPI コミュニケーター
in	*bpos*	自PE 担当領域の基点座標
in	*bbsize*	同、計算領域のボクセル数
in	*gcsize*	同、ガイドセルのボクセル数
in	*dx*	同、ボクセル 1 辺の長さ

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.19 POLYLIB_STAT PolylibNS::Polylib::init_parallel_info (MPI_Comm *comm*, const std::vector< ParallelBbox > & *bboxes*)

並列計算関連情報の設定と初期化を行う。 (各ランクが複数領域を担当している場合） 全 rank で各々設定を行い、その領域情報を全 rank へ配信する。

Parameters

in	*comm*	MPI コミュニケーター
in	*bpos*	自PE 担当領域の基点座標
in	*bbsize*	同、計算領域のボクセル数
in	*gcsize*	同、ガイドセルのボクセル数
in	*dx*	同、ボクセル 1 辺の長さ

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.20 POLYLIB_STAT PolylibNS::Polylib::load (const std::string *config_name* = "polylib_config.tp", PL_REAL *scale* = 1.0)

PolygoGroup、三角形ポリゴン情報の読み込み。 引数で指定された設定ファイル (TextParser 形式) を読み込み、グループツリーを作成する。 続いて設定ファイルで指定されたSTL(or NPT) ファイルを読み込み、KD 木を作成する。

Parameters

in	*config_name*	設定ファイル名
in	*scale*	縮尺率

Returns

POLYLIB_STAT で定義される値が返る。

Attention

set_max_memory_size_mb 関数にてメモリ使用サイズMAX を 指定すると、必要に応じてポリゴンファイルを分割して

6.13.3.21 POLYLIB_STAT PolylibNS::Polylib::load_polygons (PL_REAL *scale* = 1.0) [protected]

STL/NPT ファイルの読み込み。 グループツリーの全リーフについて、設定されているSTL/NPT ファイルから ポリゴン情報を読み込む。読み込んだ後、KD 木の生成、法線の計算、面積の 計算を行う。

Parameters

in	*scale*	縮尺率

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.22 POLYLIB_STAT PolylibNS::Polylib::make_group_tree (TextParser * *tp_ptr*) [protected]

グループツリー作成。 TextParser クラスを使い、 PolygonGroup を作成し、グループツリーに登録する。

Parameters

in	*TextParser*	のインスタンス

Returns

POLYLIB_STAT で定義される値が返る。 //

Attention

オーバーロードメソッドあり。

6.13.3.23 POLYLIB_STAT PolylibNS::Polylib::migrate ( )

ポリゴンデータのPE 間移動 本クラスインスタンス配下の全PolygonGroup のポリゴンデータについて、 move メソッドにより移動した三角形ポリゴン情報を隣接PE 間でやり取りする。

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.24 POLYLIB_STAT PolylibNS::Polylib::move (PolylibMoveParams & *params*)

三角形ポリゴン座標の移動 本クラスインスタンス配下の全PolygonGroup の move メソッドが呼び出される。 move メソッドは、PolygonGroup クラスを拡張したクラスに利用者が記述する。

Parameters

in	*params*	Polylib.h で宣言された移動計算パラメータセット。

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.25 POLYLIB_STAT PolylibNS::Polylib::save (std::string & *config_name_out,* const std::string & *file_format,* std::string *extend* = " ")

PolygoGroup ツリー、三角形ポリゴン情報の保存。 グループツリーの情報を設定ファイルへ出力。三角形ポリゴン情報をSTL/NPT ファイルへ出力。

Parameters

out	*config_name_out*	保存した設定ファイル名の返却用。
in	*file_format*	PolygonIO.h クラスで定義されているSTL/NPT ファイルの フォーマット
in	*extend*	ファイル名に付加する文字列。省略可。省略した 場合は、付加文字列として本メソッド呼び出し時 の年月日時分秒 (YYYYMMDD24hhmmss) を用いる。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ファイル名命名規約は次の通り。 設定ファイル：polylib_config_ランク番号_付加文字.tpp。 STL/NPT ファイル：ポリゴングループ名_ランク番号_付加文字.拡張子。 set_max_memory_size_mb 関数にてメモリ使用サイズMAX を 指定しても、メモリ削減版は動作しません

6.13.3.26 POLYLIB_STAT PolylibNS::Polylib::save_at_rank (std::string & *config_name_out*, int *myrank*, int *maxrank*, const std::string & *extend*, const std::string & *file_format*) [protected]

ランク毎にPolygoGroup ツリー、ポリゴン情報の保存 グループツリー情報を設定ファイルへ出力。ポリゴン情報をSTL/NPT ファイル へ出力。

Parameters

out	*config_name_out*	保存した設定ファイル名の返却用。
in	*myrank*	自ランク番号。
in	*maxrank*	最大ランク番号。
in	*extend*	ファイ名に付加される文字列。
in	*file_format*	ファイルフォーマット指定。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ファイル名命名規約は次の通り。 定義ファイル：polylib_config_ランク番号_付加文字.xml。 STL ファイル：ポリゴングループ名_ランク番号_付加文字.拡張子。
save_parallel()の内部実装用メソッド

6.13.3.27 char* PolylibNS::Polylib::save_config_file (const std::string & *rank_no*, const std::string & *extend*, const std::string & *format*) [protected]

設定ファイルの保存。 メモリに展開しているグループツリー情報から設定ファイルを生成する。

Parameters

in	*rank_no*	ランク番号

in	*extend*	ファイル名に付加する文字列
in	*format*	PolygonIO クラスで定義されているSTL/NPT ファイルのフォー マット。

Returns

作成した設定ファイルの名称。エラー時はNULL が返る。

6.13.3.28 POLYLIB_STAT PolylibNS::Polylib::save_parallel (std::string & *config_name_out,* const std::string & *file_format,* std::string *extend* = " ")

全 rank 並列でのデータ保存。 各 rank の本クラスインスタンスが保持するグループ階層構造を設定ファイルに各 rank 毎に書き出す。 同時にポリゴンデータも指定されたフォーマットのSTL/NPT ファイルに各 rank 毎に書き出す。 設定ファイル命名規則は以下の通り polylib_config_ランク番号_付加文字列.tpp STL/NPT ファイル命名規則は以下の通り ポリゴングループ名称_ランク番号_付加文字列.拡張子

Parameters

out	*config_name_out*	設定ファイル名返却用
in	*file_format*	STL/NPT ファイルフォーマット。 "stl_a":アスキー形式　"stl_b":バイナリ形式 "obj_a":アスキー形式　"obj_b","obj_bb":バイナリ形式,"obj_bb"は、頂点法線付き。
in	*extend*	ファイル名に付加する文字列。省略可。省略 した場合は、付加文字列として本メソッド呼 び出し時の年月日時分秒 (YYYYMMDD24hhmmss) を用いる。

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.29 POLYLIB_STAT PolylibNS::Polylib::search_nearest_polygon (Triangle ∗& *tri,* const std::string & *group_name,* const Vec3< PL_REAL > & *pos*) const

指定した点に最も近い三角形ポリゴンの検索

Parameters

out	*tri*	検索されたポリゴン (Triangle/NptTriangle) != 0 検索されたポリゴン
in	*group_name*	抽出グループ名
in	*pos*	指定した点

Returns

POLYLIB_STAT で定義される値

Attention

tri は削除不可 MPI 並列計算時は,pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため)

6.13.3.30 POLYLIB_STAT PolylibNS::Polylib::search_nearest_polygon (NptTriangle ∗& *tri,* const std::string & *group_name,* const Vec3< PL_REAL > & *pos*) const

指定した点に最も近い長田パッチポリゴンの検索

Parameters

out	*tri*	検索されたポリゴン (NptTriangle) != 0 検索されたポリゴン
in	*group_name*	抽出グループ名
in	*pos*	指定した点

Returns

POLYLIB_STAT で定義される値

Attention

tri は削除不可 MPI 並列計算時は,pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

6.13.3.31 POLYLIB_STAT PolylibNS::Polylib::search_polygons (std::vector< Triangle * > & *tri_list,* const std::string & *group_name,* const Vec3< PL_REAL > & *min_pos,* const Vec3< PL_REAL > & *max_pos,* const bool *every*) const

ポリゴンの検索 位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 ポリゴンを group_name で指定されたグループの下から探索する

Parameters

in,out	*tri_list*	検索されたポリゴンリスト (Triangle/NptTriangle) ポリゴンが追加されて返される
in	*group_name*	抽出グループ名
in	*min_pos*	抽出する矩形領域の最小値
in	*max_pos*	抽出する矩形領域の最大値
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出 false:3 頂点の一部でも検索領域と重なるものを抽出

Returns

POLYLIB_STAT で定義される値

Attention

tri_list 内のポリゴン要素は、削除不可 tri_list 内のポリゴン要素はサブクラスのNptTriangle である可能性あり MPI 並列計算時は,min_pos, max_pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

6.13.3.32 POLYLIB_STAT PolylibNS::Polylib::search_polygons (std::vector< NptTriangle * > & *tri_list,* const std::string & *group_name,* const Vec3< PL_REAL > & *min_pos,* const Vec3< PL_REAL > & *max_pos,* const bool *every*) const

長田パッチポリゴンの検索 位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 ポリゴンを group_name で指定されたグループの下から探索する

Parameters

in,out	*tri_list*	検索された長田パッチポリゴンリスト (NptTriangle) ポリゴンが追加されて返される

in	*group_name*	抽出グループ名
in	*min_pos*	抽出する矩形領域の最小値
in	*max_pos*	抽出する矩形領域の最大値
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出 false:3 頂点の一部でも検索領域と重なるものを抽出

Returns

POLYLIB_STAT で定義される値

Attention

tri_list 内のポリゴン要素は、削除不可 MPI 並列計算時は,min_pos, max_pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

6.13.3.33 void PolylibNS::Polylib::set_max_memory_size_mb (int *max_size_mb*) [inline]

Polylib が利用する最大メモリサイズ (MB) を設定する

Parameters

in	*max_size_mb*	最大メモリサイズ (Mbyte) >= 0 0 を設定するとメモリ制限なしとなる

Returns

戻り値なし

Attention

現状、MPI 環境でPolygonGroup のポリゴンの load 処理に関係するのみ 何個に分割して load するかがこの値で決まる 設定がない場合は、メモリ制限なしとみなす MPI 内部で使用する分は考慮しないので、余裕を持って設定すること

6.13.3.34 static void PolylibNS::Polylib::set_srch_mode (bool *detail*) [static]

検索モードの指定 ポリゴン検索時にポリゴンを曲面補正して検索させるかどうかを指定する

Parameters

in	*detail*	詳細検索指定 true : 曲面補正が可能な場合、曲面補正して検索 false : 3 角形平面で検索

6.13.3.35 POLYLIB_STAT PolylibNS::Polylib::setfilepath (std::map< std::string, std::string > & *polygons_fname_map*) [protected]

TextParser 内部データに　save した stl ファイルの　"filepath"を書き込む。

　save したSTL ファイルとPolygonGroup の階層は、save_stl_file に　　map を渡し保持してもらう。その map の内容に基づき、TextParser 内部のデータを　変更する.

Parameters

in	*polygons_fname_map*	save したSTL/NPT ファイルとその階層の map 型データ

Returns

POLYLIB_STAT で定義される値が返る。

6.13.3.36 void PolylibNS::Polylib::show_all_group_info (bool *detail* = `false`)

全てのグループの情報と配下の三角形ポリゴン情報を標準出力に出力。 親グループ名、自身の名前、STL ファイル名、属性、面積、座標値

Parameters

in	*detail*	ポリゴンの座標値・法線ベクトルを出力するか否か

Returns

戻り値なし

Attention

テスト用の関数であり、通常は利用者が用いるものではない

6.13.3.37 void PolylibNS::Polylib::show_group_hierarchy (FILE * *fp* = `NULL`)

グループ階層構造を標準出力に出力。 2010.10.20 引数FILE * 追加。

Parameters

in	*fp*	出力先ファイル。指定されていれれば、標準出力へ出力する。

Attention

テスト用の関数であり、通常は利用者が用いるものではない。

6.13.3.38 POLYLIB_STAT PolylibNS::Polylib::show_group_info (const std::string & *group_name*, bool *detail* = `false`)

グループの情報と配下の三角形ポリゴン情報を標準出力に出力。 親グループ名、自身の名前、STL ファイル名、属性、面積、座標値

Parameters

in	*group_name*	グループ名
in	*detail*	ポリゴンの座標値・法線ベクトルを出力するか否か

Returns

POLYLIB_STAT で定義される値が返る

Attention

テスト用の関数であり、通常は利用者が用いるものではない

6.13.3.39 void PolylibNS::Polylib::show_group_name (PolygonGroup * *p*, std::string *tab*, FILE * *fp*) [protected]

グループ名の表示。 指定されたグループ以下の階層構造をツリー形式で標準出力に出力する。 2010.10.20 引数FILE * 追加。

Parameters

in	*p*	検索の基点となるPolygonGroup のポインタ
in	*tab*	階層の深さを示すスペース
in	*fp*	出力先ファイル。指定されて行ければ、標準出力へ出力する。

6.13.3.40 size_t PolylibNS::Polylib::used_memory_size ( )

Polylib が利用中の概算メモリ量を返す

Returns

利用中のメモリ量 (byte)

6.13.3.41 size_t PolylibNS::Polylib::used_memory_size_mb ( ) [inline]

Polylib が利用中の概算メモリ量 (MB) を返す

Returns

利用中のメモリ量 (Mbyte)

6.13.4 Member Data Documentation

6.13.4.1 MPI_Comm PolylibNS::Polylib::m_comm [protected]

MPI コミュニケーター

6.13.4.2 std::vector< std::map< int, std::vector<long long int> > > PolylibNS::Polylib::m_exclusion_map_procs [protected]

migrate 除外三角形ID マップ (k:グループID, v:三角形ID リスト) 一時情報 m_neibour_procs_area 数分あり (順番も同一)

6.13.4.3 int PolylibNS::Polylib::m_max_memory_size_mb [protected]

MAX メモリーサイズ (MB)

6.13.4.4 ParallelAreaInfo PolylibNS::Polylib::m_myproc_area [protected]

自PE 担当領域情報

6.13.4.5 int PolylibNS::Polylib::m_myrank [protected]

自プロセスのランクNo

6.13.4.6 std::vector<ParallelAreaInfo> PolylibNS::Polylib::m_neibour_procs_area [protected]

隣接PE 担当領域情報リスト

6.13.4.7 int PolylibNS::Polylib::m_numproc [protected]

全プロセス数

6.13.4.8 std::vector<ParallelAreaInfo> PolylibNS::Polylib::m_other_procs_area [protected]

自PE を除く全PE 担当領域情報リスト

6.13.4.9 std::vector<PolygonGroup*> PolylibNS::Polylib::m_pg_list [protected]

全てのポリゴングループリスト

6.13.4.10 TextParser* PolylibNS::Polylib::tp [protected]

TextParser へのポインタ

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

- Polylib.h

6.14 PolylibNS::PolylibMoveParams Class Reference

`#include <Polylib.h>`

Public Attributes

- int m_current_step
  現在の計算ステップ番号
- int m_next_step
  移動後の計算ステップ番号
- PL_REAL m_delta_t
  1 計算ステップあたりの時間変異
- PL_REAL m_params [10]
  ユーザ定義パラメータ

6.14.1 Detailed Description

クラス:PolylibMoveParams Polylib::move()の引数として利用するパラメタセットクラスです。

6.14.2 Member Data Documentation

6.14.2.1 int PolylibNS::PolylibMoveParams::m_current_step

現在の計算ステップ番号

6.14.2.2 PL_REAL PolylibNS::PolylibMoveParams::m_delta_t

1 計算ステップあたりの時間変異

6.14.2.3 int PolylibNS::PolylibMoveParams::m_next_step

移動後の計算ステップ番号

6.14.2.4 PL_REAL PolylibNS::PolylibMoveParams::m_params[10]

ユーザ定義パラメータ

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

- Polylib.h

6.15 PolylibMoveParamsStruct Struct Reference

```
#include <CPolylib.h>
```

Public Attributes

- int m_current_step
  現在の計算ステップ番号
- int m_next_step
  移動後の計算ステップ番号
- float m_delta_t
  1 計算ステップあたりの時間変異
- float m_params [10]
  ユーザ定義パラメータ

6.15.1 Detailed Description

構造体:PolylibMoveParamsStruct polylib_move()の引数として利用するパラメタセットの構造体です。 本構造体メンバ変数ではパラメタが不足する場合は、C++側のPolylibMoveParams も含めて ユーザ定義する必要がある PolylibMoveParams の継承クラスを作成する、構造体もそれに合わせて新規に作成するなど

6.15.2 Member Data Documentation

6.15.2.1 int PolylibMoveParamsStruct::m_current_step

現在の計算ステップ番号

6.15.2.2 float PolylibMoveParamsStruct::m_delta_t

1 計算ステップあたりの時間変異

6.15.2.3 int PolylibMoveParamsStruct::m_next_step

移動後の計算ステップ番号

6.15.2.4 float PolylibMoveParamsStruct::m_params[10]

ユーザ定義パラメータ

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

- c_lang/CPolylib.h

6.16 PolylibNS::Triangle Class Reference

`#include <Triangle.h>`

Inherited by PolylibNS::NptTriangle.

Public Member Functions

- Triangle ()
- Triangle (const Triangle &tria)
- Triangle (const Vec3< PL_REAL > vertex[3], long long int id=0, int num_atrI=0, int num_atrR=0, const int *atrI=NULL, const PL_REAL *atrR=NULL)
- Triangle (const Vec3< PL_REAL > vertex[3], const Vec3< PL_REAL > &normal, long long int id=0, int num_atrI=0, int num_atrR=0, const int *atrI=NULL, const PL_REAL *atrR=NULL)
- Triangle (const char *pbuff)
- virtual ~Triangle ()
- void set_num_atr (int num_atrI, int num_atrR)
- int get_num_atrI (void)
- int get_num_atrR (void)
- virtual void rescale (PL_REAL scale)
- virtual size_t used_memory_size (void)
- virtual size_t serialized_size (void)
- virtual char * serialize (const char *pbuff)
- virtual BBox get_bbox (bool detail=false)
- virtual int get_pl_type (void)
- virtual void set_vertexes (const Vec3< PL_REAL > vertex[3], bool update_param, bool calc_area)
- virtual void update (bool update_normal, bool update_area)
- Vec3< PL_REAL > * get_vertexes () const
- Vec3< PL_REAL > get_normal () const
- PL_REAL get_area () const
- void set_id (long long int id)
- long long int get_id () const
- void set_exid (int exid)
- int get_exid () const
- int * get_pAtrI () const
- PL_REAL * get_pAtrR () const

Protected Member Functions

- void calc_normal ()
- void calc_area ()
- long long int create_unique_id ()

Protected Attributes

- long long int m_id
  一意となるポリゴン*ID*（内部識別用）
- int ∗ m_Atrl
  ユーザ定義属性（整数型）
- PL_REAL ∗ m_AtrR
  ユーザ定義属性（実数型）
- Vec3< PL_REAL > m_vertex [3]
  三角形の頂点座標（反時計回りで並んでいる）
- Vec3< PL_REAL > m_normal
  三角形の法線ベクトル
- PL_REAL m_area
  三角形の面積
- short int m_exid
  三角形のユーザ定義*ID*（*FFV-C* でのみ使用すること）
- unsigned char m_numAtrl
  ユーザ定義属性数（整数型）
- unsigned char m_numAtrR
  ユーザ定義属性数数（実数型）

6.16.1 Detailed Description

クラス:Triangle

6.16.2 Constructor & Destructor Documentation

6.16.2.1 PolylibNS::Triangle::Triangle ( ) [inline]

コンストラクタ。

6.16.2.2 PolylibNS::Triangle::Triangle (const Triangle & *tria*) [inline]

コンストラクタ（コピーコンストラクタ）

Parameters

in	*tria*	コピー元。

6.16.2.3 PolylibNS::Triangle::Triangle (const Vec3< PL_REAL > *vertex[3]*, long long int *id* = 0, int *num_atrl* = 0, int *num_atrR* = 0, const int ∗ *atrl* = NULL, const PL_REAL ∗ *atrR* = NULL) [inline]

コンストラクタ。

Parameters

in	*vertex*	ポリゴンの頂点
in	*id*	三角形ポリゴンID（ユニークな識別子） システム全体でユニークな識別子であること

in	*num_atrI*	ユーザ定義属性数（整数型）
in	*num_atrR*	ユーザ定義属性数（実数型）
in	*atrI*	ユーザ定義属性（整数型）
in	*atrR*	ユーザ定義属性（実数型）

Attention

面積と法線は vertex を元に自動計算される atrI,atrR=NULL の場合、属性領域は初期値が設定される id=0 の場合、ID は内部で採番される

6.16.2.4 PolylibNS::Triangle::Triangle (const Vec3< PL_REAL > *vertex[3],* const Vec3< PL_REAL > & *normal,* long long int *id* = 0, int *num_atrI* = 0, int *num_atrR* = 0, const int * *atrI* = NULL, const PL_REAL * *atrR* = NULL) [inline]

コンストラクタ。

Parameters

in	*vertex*	ポリゴンの頂点
in	*normal*	法線
in	*id*	三角形ポリゴンID（ユニークな識別子） システム全体でユニークな識別子であること
in	*num_atrI*	ユーザ定義属性数（整数型）
in	*num_atrR*	ユーザ定義属性数（実数型）
in	*atrI*	ユーザ定義属性（整数型）
in	*atrR*	ユーザ定義属性（実数型）

Attention

面積は vertex を元に自動計算される id=0 の場合、ID は内部で採番される

6.16.2.5 PolylibNS::Triangle::Triangle (const char * *pbuff*)

コンストラクタ (deserialize) シリアライズされた通信バッファよりオブジェクトの生成を行う

Parameters

in	*pbuff*	バッファ格納先頭位置ポインタ シリアリズされたデータ

Attention

シリアライズは serialize() で行う

6.16.2.6 virtual PolylibNS::Triangle::~Triangle ( ) [inline], [virtual]

デストラクタ

6.16.3 Member Function Documentation

6.16.3.1 void PolylibNS::Triangle::calc_area ( ) [inline], [protected]

面積算出。

6.16.3.2 void PolylibNS::Triangle::calc_normal ( ) [inline], [protected]

法線ベクトル算出。

6.16.3.3 long long int PolylibNS::Triangle::create_unique_id ( ) [protected]

システムで一意のポリゴンID を作成する MPI 環境の場合、ランク 0 以外で当ルーチンが呼ばれてはならない ポリゴン生成時にユーザ指定でポリゴンID を指定することも可能であるが 全体で整合性のあるものに制御すること

Returns

ポリゴンID

6.16.3.4 PL_REAL PolylibNS::Triangle::get_area ( ) const [inline]

面積を取得。

Returns

面積。

6.16.3.5 virtual BBox PolylibNS::Triangle::get_bbox (bool *detail* = false) [virtual]

Bounding box of this triangle

Parameters

in	*detail*	曲面補間可能要素の時に曲面補間したBounding box を返すか否か false: 3 角形の頂点にて決定

Returns

Bounding box

Reimplemented in PolylibNS::NptTriangle.

6.16.3.6 int PolylibNS::Triangle::get_exid ( ) const [inline]

ユーザ定義ID を取得 FFV-C 専用

Returns

ユーザ定義ID。

6.16.3.7 long long int PolylibNS::Triangle::get_id ( ) const [inline]

ポリゴンID を返す。

Returns

ポリゴンID。

6.16.3.8 Vec3<PL_REAL> PolylibNS::Triangle::get_normal ( ) const [inline]

法線ベクトルを取得。

Returns

法線ベクトル

Attention

return 値はポインタではありません メンバー変数でなく計算により求める方法に変更するかもしれないため

6.16.3.9 int PolylibNS::Triangle::get_num_atrI (void) [inline]

ユーザ定義属性数（整数型）の取得

Returns

ユーザ定義属性数（整数型）

6.16.3.10 int PolylibNS::Triangle::get_num_atrR (void) [inline]

ユーザ定義属性数（実数型）の取得

Returns

ユーザ定義属性数（実数型）

6.16.3.11 int* PolylibNS::Triangle::get_pAtrI ( ) const [inline]

ユーザ定義属性数（整数型）のポインタ取得

Returns

ユーザ定義属性数（整数型）のポインタ =NULL の場合、属性なし

Attention

ユーザ定義属性の設定は以下で行う int* p = get_polygons_pAtrI(); p[2] = 1 属性種類の 3 番目に 1 を設定

6.16.3.12 PL_REAL* PolylibNS::Triangle::get_pAtrR ( ) const [inline]

ユーザ定義属性数（実数型）のポインタ取得

Returns

ユーザ定義属性数（実数型）のポインタ =NULL の場合、属性なし

Attention

ユーザ定義属性の設定は以下で行う PL_REAL* p = get_polygons_pAtrR(); p[2] = 1.0 属性種類の 3 番目に 1.0 を設定

6.16.3.13 virtual int PolylibNS::Triangle::get_pl_type (void) [inline], [virtual]

ポリゴンタイプ取得

Returns

ポリゴンタイプ PL_TYPE_TRIANGLE

Reimplemented in PolylibNS::NptTriangle.

6.16.3.14 Vec3<PL_REAL>* PolylibNS::Triangle::get_vertexes ( ) const [inline]

頂点の取得

Returns

三角形の 3 頂点へのポインタ

6.16.3.15 virtual void PolylibNS::Triangle::rescale (PL_REAL *scale*) [inline], [virtual]

ポリゴンのリスケール

Parameters

in	*scale*	スケール

Returns

戻り値なし

Reimplemented in PolylibNS::NptTriangle.

6.16.3.16 virtual char* PolylibNS::Triangle::serialize (const char * *pbuff*) [virtual]

ポリゴンのシリアリズ 1 ポリゴンを通信バッファに格納する

Parameters

out	*pbuff*	バッファ格納位置先頭ポインタ シリアリズされたデータ

Returns

バッファ格納位置Next ポインタ

Attention

デシリアリズはPolylibNS::deserialize_polygon() で行う 最終的にはコンストラクタ Triangle(pbuff) が呼び出される

Reimplemented in PolylibNS::NptTriangle.

6.16.3.17 virtual size_t PolylibNS::Triangle::serialized_size (void) [inline], [virtual]

ポリゴンのシリアリズした時のサイズ取得 通信バッファにシリアリズした時のサイズ

Returns

シリアリズ後のサイズ

Reimplemented in PolylibNS::NptTriangle.

6.16.3.18 void PolylibNS::Triangle::set_exid (int *exid*) [inline]

ユーザ定義ID を設定 FFV-C 専用

6.16.3.19 void PolylibNS::Triangle::set_id (long long int *id*) [inline]

ポリゴンID を設定。

Parameters

in	*id*	ポリゴンID。

Attention

一般ユーザ使用不可

6.16.3.20 void PolylibNS::Triangle::set_num_atr (int *num_atrI*, int *num_atrR*) [inline]

ユーザ定義属性数の設定 （ユーザ限定関数） PolygonGroup 内で同一属性数である必要があるため 一般的にはPolygonGroup::set_num_atr() にて設定する

Parameters

in	*num_atrI*	ユーザ定義属性数（整数型）
in	*num_atrR*	ユーザ定義属性数（実数型）

Returns

なし

Attention

属性格納領域を allocation する 新規アロケーション時は初期値を設定する 既にアロケーションされていた場合、realloc する

6.16.3.21 virtual void PolylibNS::Triangle::set_vertexes (const Vec3< PL_REAL > *vertex[3]*, bool *update_param*, bool *calc_area*) [virtual]

頂点を設定

Parameters

in	*vertex*	三角形の 3 頂点
in	*update_param*	面の法線ベクトルを再計算するか？
in	*calc_area*	面積を再計算するか？

Attention

長田パッチの場合、update_param=true で 長田パッチのパラメータ更新も更新する 但し、この長田パッチのパラメータ更新は遅いため 長田パッチとわかっている場合は、長田パッチを同時に 更新するタイプは良い

Reimplemented in PolylibNS::NptTriangle.

6.16.3.22 virtual void PolylibNS::Triangle::update (bool *update_normal*, bool *update_area*) [inline], [virtual]

法線ベクトル・面積の更新

Parameters

in	*update_normal*	法線ベクトルを再計算するか？
in	*update_area*	面積を再計算するか？

Attention

get_vertexes()で座標のポインタを取得して 直接座標を書き換えた時などに、法線ベクトル・面積を更新するのに使用する

6.16.3.23 virtual size_t PolylibNS::Triangle::used_memory_size (void) [inline], [virtual]

ポリゴンの使用メモリサイズ取得 Polylib::used_memory_size()で使用する

Returns

使用メモリサイズ

Reimplemented in PolylibNS::NptTriangle.

6.16.4 Member Data Documentation

6.16.4.1 PL_REAL PolylibNS::Triangle::m_area [protected]

三角形の面積

6.16.4.2 int* PolylibNS::Triangle::m_AtrI [protected]

ユーザ定義属性（整数型）

6.16.4.3 PL_REAL* PolylibNS::Triangle::m_AtrR [protected]

ユーザ定義属性（実数型）

6.16.4.4 short int PolylibNS::Triangle::m_exid [protected]

三角形のユーザ定義ID（FFV-C でのみ使用すること）

6.16.4.5 long long int PolylibNS::Triangle::m_id [protected]

一意となるポリゴンID（内部識別用）

6.16.4.6 Vec3<PL_REAL> PolylibNS::Triangle::m_normal [protected]

三角形の法線ベクトル

6.16.4.7 unsigned char PolylibNS::Triangle::m_numAtrI [protected]

ユーザ定義属性数（整数型）

6.16.4.8 unsigned char PolylibNS::Triangle::m_numAtrR [protected]

ユーザ定義属性数数（実数型）

6.16.4.9 Vec3<PL_REAL> PolylibNS::Triangle::m_vertex[3] [protected]

三角形の頂点座標（反時計回りで並んでいる）

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

- polygons/Triangle.h

6.17 TriangleStruct Struct Reference

`#include <CPolylib.h>`

Public Attributes

- float vertex [9]
  ３頂点座標
- float normal [3]
  面法線ベクトル

6.17.1 Detailed Description

三角形ポリゴン情報構造体

6.17.2 Member Data Documentation

6.17.2.1 float TriangleStruct::normal[3]

面法線ベクトル

6.17.2.2 float TriangleStruct::vertex[9]

３頂点座標

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

- c_lang/CPolylib.h

6.18 PolylibNS::UsrAtr Struct Reference

`#include <PolygonGroup.h>`

Public Attributes

- std::string key
  キー
- std::string value
  値

6.18.1 Member Data Documentation

6.18.1.1 std::string PolylibNS::UsrAtr::key

キー

6.18.1.2 std::string PolylibNS::UsrAtr::value

値

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

- groups/PolygonGroup.h

6.19 PolylibNS::Vec2< T > Class Template Reference

`#include <Vec2.h>`

Public Member Functions

- Vec2 (T v=0)
- Vec2 (T _x, T _y)
- Vec2 (const T v[2])
- Vec2< T > & assign (T _x, T _y)
- operator T * ()
- operator const T * () const
- T * ptr ()
- const T * ptr () const
- T & operator[] (int i)
- const T & operator[] (int i) const
- Vec2< T > & operator+= (const Vec2< T > &v)
- Vec2< T > & operator-= (const Vec2< T > &v)
- Vec2< T > & operator*= (const Vec2< T > &v)
- Vec2< T > & operator/= (const Vec2< T > &v)
- Vec2< T > & operator*= (T s)
- Vec2< T > & operator/= (T s)
- Vec2< T > operator+ (const Vec2< T > &v) const
- Vec2< T > operator- (const Vec2< T > &v) const
- Vec2< T > operator* (const Vec2< T > &v) const
- Vec2< T > operator/ (const Vec2< T > &v) const
- Vec2< T > operator* (T s) const
- Vec2< T > operator/ (T s) const
- Vec2< T > operator- () const
- bool operator== (const Vec2< T > &v) const
- bool operator!= (const Vec2< T > &v) const
- float lengthSquared () const
- float length () const
- Vec2< T > & normalize ()
- Vec2< T > & normalize (float *len)
- float average () const

Static Public Member Functions

- static Vec2< T > xaxis ()
- static Vec2< T > yaxis ()

Public Attributes

- T x
- T y

6.19.1 Detailed Description

template<typename T>class PolylibNS::Vec2< T >

クラス:Vec2<T>

6.19.2 Constructor & Destructor Documentation

6.19.2.1 template<typename T> PolylibNS::Vec2< T >::Vec2 (T *v* = 0) [inline]

6.19.2.2 template<typename T> PolylibNS::Vec2< T >::Vec2 (T *_x*, T *_y*) [inline]

6.19.2.3 template<typename T> PolylibNS::Vec2< T >::Vec2 (const T *v[2]*) [inline]

6.19.3 Member Function Documentation

6.19.3.1 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::assign (T *_x*, T *_y*) [inline]

6.19.3.2 template<typename T> float PolylibNS::Vec2< T >::average ( ) const [inline]

6.19.3.3 template<typename T> float PolylibNS::Vec2< T >::length ( ) const [inline]

6.19.3.4 template<typename T> float PolylibNS::Vec2< T >::lengthSquared ( ) const [inline]

6.19.3.5 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::normalize ( ) [inline]

6.19.3.6 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::normalize (float * *len*) [inline]

6.19.3.7 template<typename T> PolylibNS::Vec2< T >::operator const T * ( ) const [inline]

6.19.3.8 template<typename T> PolylibNS::Vec2< T >::operator T * ( ) [inline]

6.19.3.9 template<typename T> bool PolylibNS::Vec2< T >::operator!= (const Vec2< T > & *v*) const [inline]

6.19.3.10 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator* (const Vec2< T > & *v*) const [inline]

6.19.3.11 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator* (T *s*) const [inline]

6.19.3.12 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::operator*= (const Vec2< T > & *v*) [inline]

6.19.3.13 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::operator*= (T *s*) [inline]

6.19.3.14 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator+ (const Vec2< T > & *v*) const [inline]

6.19.3.15 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::operator+= (const Vec2< T > & *v*) [inline]

6.19.3.16 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator- (const Vec2< T > & *v*) const [inline]

6.19.3.17 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator- ( ) const [inline]

6.19.3.18 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::operator-= (const Vec2< T > & *v*) [inline]

6.19.3.19 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator/ (const Vec2< T > & *v*) const [inline]

6.19.3.20 template<typename T> Vec2<T> PolylibNS::Vec2< T >::operator/ (T *s*) const [inline]

6.19.3.21 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::operator/= (const Vec2< T > & *v*) [inline]

6.19.3.22 template<typename T> Vec2<T>& PolylibNS::Vec2< T >::operator/= (T *s*) [inline]

6.19.3.23 template<typename T> bool PolylibNS::Vec2< T >::operator== (const Vec2< T > & *v*) const [inline]

6.19.3.24 template<typename T> T& PolylibNS::Vec2< T >::operator[] (int *i*) [inline]

6.19.3.25 template<typename T> const T& PolylibNS::Vec2< T >::operator[] (int *i*) const [inline]

6.19.3.26 template<typename T> T* PolylibNS::Vec2< T >::ptr ( ) [inline]

6.19.3.27 template<typename T> const T* PolylibNS::Vec2< T >::ptr ( ) const [inline]

6.19.3.28 template<typename T> static Vec2<T> PolylibNS::Vec2< T >::xaxis ( ) [inline], [static]

6.19.3.29 template<typename T> static Vec2<T> PolylibNS::Vec2< T >::yaxis ( ) [inline], [static]

6.19.4 Member Data Documentation

6.19.4.1 template<typename T> T PolylibNS::Vec2< T >::x

6.19.4.2 template<typename T> T PolylibNS::Vec2< T >::y

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

- common/Vec2.h

6.20 Vec3class::Vec3< T > Class Template Reference

`#include <Vec3.h>`

Public Member Functions

- Vec3 (T v=0)

- Vec3 (T _x, T _y, T _z)
- Vec3 (const T v[3])
- Vec3 (const Vec3 &v)
- Vec3< T > & assign (T _x, T _y, T _z)
- operator T * ()
- operator const T * () const
- T * ptr ()
- const T * ptr () const
- T & operator[] (const AxisEnum &axis)
- const T & operator[] (const AxisEnum &axis) const
- Vec3< T > & operator+= (const Vec3< T > &v)
- Vec3< T > & operator-= (const Vec3< T > &v)
- Vec3< T > & operator*= (const Vec3< T > &v)
- Vec3< T > & operator/= (const Vec3< T > &v)
- Vec3< T > & operator*= (T s)
- Vec3< T > & operator/= (T s)
- Vec3< T > operator+ (const Vec3< T > &v) const
- Vec3< T > operator- (const Vec3< T > &v) const
- Vec3< T > operator* (const Vec3< T > &v) const
- Vec3< T > operator/ (const Vec3< T > &v) const
- Vec3< T > operator* (T s) const
- Vec3< T > operator/ (T s) const
- Vec3< T > operator- () const
- bool operator== (const Vec3< T > &v) const
- bool operator!= (const Vec3< T > &v) const
- T lengthSquared () const
- T length () const
- Vec3< T > & normalize ()
- Vec3< T > & normalize (T *len)
- T average () const

Static Public Member Functions

- static Vec3< T > xaxis ()
- static Vec3< T > yaxis ()
- static Vec3< T > zaxis ()

Public Attributes

- T x
- T y
- T z

6.20.1 Constructor & Destructor Documentation

6.20.1.1 template<typename T> Vec3class::Vec3< T >::Vec3 (T *v* = 0) [inline]

6.20.1.2 template<typename T> Vec3class::Vec3< T >::Vec3 (T *_x*, T *_y*, T *_z*) [inline]

6.20.1.3 template<typename T> Vec3class::Vec3< T >::Vec3 (const T *v[3]*) [inline]

6.20.1.4 template<typename T> Vec3class::Vec3< T >::Vec3 (const Vec3< T > & *v*) [inline]

6.20.2 Member Function Documentation

6.20.2.1 template<typename T> Vec3<T>& Vec3class::Vec3< T >::assign (T *_x*, T *_y*, T *_z*) [inline]

6.20.2.2 template<typename T> T Vec3class::Vec3< T >::average ( ) const [inline]

6.20.2.3 template<typename T> T Vec3class::Vec3< T >::length ( ) const [inline]

6.20.2.4 template<typename T> T Vec3class::Vec3< T >::lengthSquared ( ) const [inline]

6.20.2.5 template<typename T> Vec3<T>& Vec3class::Vec3< T >::normalize ( ) [inline]

6.20.2.6 template<typename T> Vec3<T>& Vec3class::Vec3< T >::normalize (T * *len*) [inline]

6.20.2.7 template<typename T> Vec3class::Vec3< T >::operator const T * ( ) const [inline]

6.20.2.8 template<typename T> Vec3class::Vec3< T >::operator T * ( ) [inline]

6.20.2.9 template<typename T> bool Vec3class::Vec3< T >::operator!= (const Vec3< T > & *v*) const [inline]

6.20.2.10 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator* (const Vec3< T > & *v*) const [inline]

6.20.2.11 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator* (T *s*) const [inline]

6.20.2.12 template<typename T> Vec3<T>& Vec3class::Vec3< T >::operator*= (const Vec3< T > & *v*) [inline]

6.20.2.13 template<typename T> Vec3<T>& Vec3class::Vec3< T >::operator*= (T *s*) [inline]

6.20.2.14 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator+ (const Vec3< T > & *v*) const [inline]

6.20.2.15 template<typename T> Vec3<T>& Vec3class::Vec3< T >::operator+= (const Vec3< T > & *v*) [inline]

6.20.2.16 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator- (const Vec3< T > & *v*) const [inline]

6.20.2.17 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator- ( ) const [inline]

6.20.2.18 template<typename T> Vec3<T>& Vec3class::Vec3< T >::operator-= (const Vec3< T > & *v*) [inline]

6.20.2.19 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator/ (const Vec3< T > & *v*) const [inline]

6.20.2.20 template<typename T> Vec3<T> Vec3class::Vec3< T >::operator/ (T *s*) const [inline]

6.20.2.21 template<typename T> Vec3<T>& Vec3class::Vec3< T >::operator/= (const Vec3< T > & *v*) [inline]

6.20.2.22 template<typename T> Vec3<T>& Vec3class::Vec3< T >::operator/= (T *s*) [inline]

6.20.2.23 template<typename T> bool Vec3class::Vec3< T >::operator== (const Vec3< T > & *v*) const [inline]

6.20.2.24 template<typename T> T& Vec3class::Vec3< T >::operator[] (const AxisEnum & *axis*) [inline]

6.20.2.25 template<typename T> const T& Vec3class::Vec3< T >::operator[] (const AxisEnum & *axis*) const [inline]

6.20.2.26 template<typename T> T* Vec3class::Vec3< T >::ptr ( ) [inline]

6.20.2.27 template<typename T> const T* Vec3class::Vec3< T >::ptr ( ) const [inline]

6.20.2.28 template<typename T> static Vec3<T> Vec3class::Vec3< T >::xaxis ( ) [inline], [static]

6.20.2.29 template<typename T> static Vec3<T> Vec3class::Vec3< T >::yaxis ( ) [inline], [static]

6.20.2.30 template<typename T> static Vec3<T> Vec3class::Vec3< T >::zaxis ( ) [inline], [static]

6.20.3 Member Data Documentation

6.20.3.1 template<typename T> T Vec3class::Vec3< T >::x

6.20.3.2 template<typename T> T Vec3class::Vec3< T >::y

6.20.3.3 template<typename T> T Vec3class::Vec3< T >::z

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

- common/Vec3.h

6.21 PolylibNS::VElement Class Reference

`#include <VTree.h>`

Public Member Functions

- VElement (Triangle *tri)
- virtual ~VElement ()
- Triangle * get_triangle () const
- Vec3< PL_REAL > get_pos () const
- BBox get_bbox () const

6.21.1 Constructor & Destructor Documentation

6.21.1.1 PolylibNS::VElement::VElement (Triangle * *tri*)

コンストラクタ。

Parameters

in	*tri*	ポリゴン情報のポインタ。

Attention

ポインタを格納するが、参照のみ。delete は行わない。

6.21.1.2 virtual PolylibNS::VElement::~VElement ( ) [inline], [virtual]

デストラクタ。

6.21.2 Member Function Documentation

6.21.2.1 BBox PolylibNS::VElement::get_bbox ( ) const [inline]

Bounding box of this triangle

6.21.2.2 Vec3<PL_REAL> PolylibNS::VElement::get_pos ( ) const [inline]

Center position of bbox on triangle.

6.21.2.3 Triangle* PolylibNS::VElement::get_triangle ( ) const [inline]

triangle。

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

- groups/VTree.h

6.22 PolylibNS::VNode Class Reference

`#include <VTree.h>`

Public Member Functions

- VNode ()
- ~VNode ()
- void split (const int &max_elem)
- bool is_leaf () const
- BBox get_bbox () const
- void set_bbox (const BBox &bbox)
- BBox get_bbox_search () const
- void set_bbox_search (const VElement *p)
- VNode * get_left ()
- VNode * get_right ()
- AxisEnum get_axis () const
- void set_axis (const AxisEnum axis)
- std::vector< VElement * > & get_vlist ()
- void set_element (VElement *elm)
- int get_elements_num () const

6.22.1 Detailed Description

VNode クラス KD 木構造のノードクラスです。

6.22.2 Constructor & Destructor Documentation

6.22.2.1 PolylibNS::VNode::VNode ( )

コンストラクタ。

6.22.2.2 PolylibNS::VNode::∼VNode ( )

デストラクタ。

6.22.3 Member Function Documentation

6.22.3.1 AxisEnum PolylibNS::VNode::get_axis ( ) const [inline]

Axis を取得

Returns

axis

6.22.3.2 BBox PolylibNS::VNode::get_bbox ( ) const [inline]

BBox の値を取得。

Returns

bbox。

6.22.3.3 BBox PolylibNS::VNode::get_bbox_search ( ) const [inline]

検索用BBox を取得。

Returns

検索用 bbox。

6.22.3.4 int PolylibNS::VNode::get_elements_num ( ) const [inline]

ノードが所持する要素の数を取得。

Returns

要素数。

6.22.3.5 VNode∗ PolylibNS::VNode::get_left ( ) [inline]

左のNode を取得。

Returns

左のNode。

6.22.3.6 VNode* PolylibNS::VNode::get_right ( ) [inline]

右のNode を取得。

Returns

右のNode。

6.22.3.7 std::vector<VElement*>& PolylibNS::VNode::get_vlist ( ) [inline]

要素のリストを取得

Returns

要素のリスト

6.22.3.8 bool PolylibNS::VNode::is_leaf ( ) const [inline]

ノードがリーフかどうかの判定結果。

Returns

true=リーフ/false=リーフでない。

6.22.3.9 void PolylibNS::VNode::set_axis (const AxisEnum *axis*) [inline]

Axis を設定

Parameters

in	*axis*	

6.22.3.10 void PolylibNS::VNode::set_bbox (const BBox & *bbox*) [inline]

BBox の値を設定。

Parameters

in	*bbox*。	

6.22.3.11 void PolylibNS::VNode::set_bbox_search (const VElement * *p*) [inline]

このノードのBounding Box を引数で与えられる要素を含めた大きさに変更する。

Parameters

in	*p*	要素。

6.22.3.12 void PolylibNS::VNode::set_element (VElement * *elm*) [inline]

木の要素を設定

Parameters

in	*elm*	

6.22.3.13 void PolylibNS::VNode::split (const int & *max_elem*)

ノードを２つの子供ノードに分割する。

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

- groups/VTree.h

6.23 PolylibNS::VTree Class Reference

`#include <VTree.h>`

Public Member Functions

- VTree (int max_elem, const BBox bbox, std::vector< Triangle * > *tri_list)
- ~VTree ()
- void destroy ()
- std::vector< Triangle * > * search (const BBox *bbox, bool every) const
- POLYLIB_STAT search (std::vector< Triangle * > &tri_list, const BBox &bbox, bool every) const
- const Triangle * search_nearest (const Vec3< PL_REAL > &pos) const
- const Triangle * search_nearest_recursive (VNode *vn, const Vec3< PL_REAL > &pos) const
- size_t memory_size ()

6.23.1 Detailed Description

クラス:VTree リーフを三角形ポリゴンとするKD 木クラスです。

6.23.2 Constructor & Destructor Documentation

6.23.2.1 PolylibNS::VTree::VTree (int *max_elem,* const BBox *bbox,* std::vector< Triangle * > * *tri_list*)

コンストラクタ。

Parameters

in	*max_elem*	最大要素数。
in	*bbox*	VTree の box 範囲。
in	*tri_list*	木構造の元になるポリゴンのリスト。

6.23.2.2 PolylibNS::VTree::~VTree ( )

デストラクタ。

6.23.3 Member Function Documentation

6.23.3.1 void PolylibNS::VTree::destroy ( )

木構造を消去する

6.23.3.2 size_t PolylibNS::VTree::memory_size ( )

KD 木クラスが利用しているメモリ量を返す。

Returns

利用中のメモリ量 (byte)

6.23.3.3 std::vector<Triangle*>* PolylibNS::VTree::search (const BBox * *bbox*, bool *every*) const

KD 木探索により、指定矩形領域に含まれる三角形ポリゴンを抽出する

Parameters

in	*bbox*	検索範囲を示す矩形領域
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出 false:1 頂点でも検索領域に含まれるものを抽出

Returns

抽出したポリゴンリストのポインタ vector のポインタは内部で allocation されているので領域解放が必要

Attention

MPIPolylib 用のメソッドなので、ユーザは利用しないで下さい

6.23.3.4 POLYLIB_STAT PolylibNS::VTree::search (std::vector< Triangle * > & *tri_list*, const BBox & *bbox*, bool *every*) const

KD 木探索により、指定矩形領域に含まれるポリゴンを抽出する

Parameters

in	*bbox*	検索範囲を示す矩形領域
in	*every*	true:3 頂点が全て検索領域に含まれるものを抽出 false:1 頂点でも検索領域に含まれるものを抽出
in,out	*tri_list*	抽出した三角形ポリゴンリストへのポインタ

Returns

POLYLIB_STAT で定義される値が返る

6.23.3.5 const Triangle* PolylibNS::VTree::search_nearest (const Vec3< PL_REAL > & *pos*) const

KD 木探索により、指定位置に最も近いポリゴンを検索する。

Parameters

in	*pos*	指定位置

Returns

検索されたポリゴン

6.23.3.6 const Triangle* PolylibNS::VTree::search_nearest_recursive (VNode * *vn*, const Vec3< PL_REAL > & *pos*) const

KD 木探索により、指定位置に最も近いポリゴンを検索する。

Parameters

in	*vn*	検索対象のノードへのポインタ。
in	*pos*	指定位置

Returns

検索されたポリゴン

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

- groups/VTree.h

Chapter 7

File Documentation

7.1 c_lang/CPolylib.h File Reference

```
#include "mpi.h"
#include <stdbool.h>
#include "common/PolylibStat.h"
#include "common/PolylibDefine.h"
```

Classes

- struct ParallelBboxStruct
- struct TriangleStruct
- struct NpatchParamStruct
- struct NptTriangleStruct
- struct PolylibMoveParamsStruct

Functions

- POLYLIB_STAT polylib_instance (void)
- POLYLIB_STAT polylib_init_parallel_info (MPI_Comm comm, float bpos[3], unsigned int bbsize[3], unsigned int gcsize[3], float dx[3])
- POLYLIB_STAT polylib_init_parallel_info2 (MPI_Comm comm, int num, ParallelBboxStruct *bboxes)
- POLYLIB_STAT polylib_load (char *config_name, float scale)
- POLYLIB_STAT polylib_save (char **p_fname, char *format, char *extend)
- POLYLIB_STAT polylib_move (PolylibMoveParamsStruct *param)
- POLYLIB_STAT polylib_migrate (void)
- POLYLIB_STAT polylib_get_group_tag (char *group_name, long long int *grp_tag)
- POLYLIB_STAT polylib_get_root_groups_tags (int *n, long long int **tags)
- POLYLIB_STAT polylib_get_leaf_groups_tags (int *n, long long int **tags)
- POLYLIB_STAT polylib_search_polygons (int *num, long long int **tags, char *group_name, float min_pos[3], float max_pos[3], int every)
- POLYLIB_STAT polylib_search_polygons_triangle (int *num, long long int **tags, TriangleStruct **tris, char *group_name, float min_pos[3], float max_pos[3], int every)
- POLYLIB_STAT polylib_search_polygons_npt (int *num, long long int **tags, NptTriangleStruct **tris, char *group_name, float min_pos[3], float max_pos[3], int every)
- POLYLIB_STAT polylib_search_nearest_polygon (long long int *tag, char *group_name, float pos[3])
- void polylib_show_group_hierarchy ()
- POLYLIB_STAT polylib_show_group_info (char *group_name)

- size_t polylib_used_memory_size ()
- size_t polylib_used_memory_size_mb ()
- POLYLIB_STAT polylib_group_get_triangles (long long int tag_pg, int *num_tri, long long int **tags_tri)
- int polylib_group_get_num_triangles (long long int tag_pg)
- int polylib_group_get_num_global_triangles (long long int tag_pg)
- float polylib_group_get_area (long long int tag_pg)
- float polylib_group_get_global_area (long long int tag_pg)
- POLYLIB_STAT polylib_group_get_polygons_reduce_atrI (long long int tag_pg, PL_OP_TYPE op, int atr_no, int *val)
- POLYLIB_STAT polylib_group_get_polygons_reduce_atrR (long long int tag_pg, PL_OP_TYPE op, int atr_no, float *val)
- POLYLIB_STAT polylib_group_set_move_func_c (long long int tag, void(*func)(long long int, PolylibMoveParamsStruct *))
- void polylib_group_set_need_rebuild (long long int tag)
- void polylib_group_set_name (long long int tag, char *name)
- void polylib_group_get_name (long long int tag, char *name)
- void polylib_group_set_movable (long long int tag, int movable)
- void polylib_group_get_movable (long long int tag, int *movable)
- void polylib_group_set_atr (long long int tag, char *key, char *val)
- POLYLIB_STAT polylib_group_get_atr (long long int tag, char *key, char *val)
- void polylib_group_set_num_polygon_atr (long long int tag, int num_atrI, int num_atrR)
- long long int polylib_group_get_parent (long long int tag)
- void polylib_group_get_children (long long int tag, int *num, long long int **child_tags)
- int polylib_triangle_get_pl_type (long long int tag)
- void polylib_triangle_set_vertexes (long long int tag, float vertex[9])
- void polylib_triangle_set_vertexes_npatch (long long int tag, float vertex[9], NpatchParamStruct *param)
- void polylib_triangle_get_vertexes (long long int tag, float vertex[9])
- void polylib_triangle_get_normal (long long int tag, float norm[3])
- POLYLIB_STAT polylib_triangle_set_npatchParam (long long int tag, NpatchParamStruct *param)
- POLYLIB_STAT polylib_triangle_get_npatchParam (long long int tag, NpatchParamStruct *param)
- int polylib_triangle_get_num_atrI (long long int tag)
- int polylib_triangle_get_num_atrR (long long int tag)
- int * polylib_triangle_get_pAtrI (long long int tag)
- float * polylib_triangle_get_pAtrR (long long int tag)

7.1.1 Function Documentation

7.1.1.1 POLYLIB_STAT polylib_get_group_tag (char * *group_name*, long long int * *grp_tag*)

ポリゴングループタグの取得 Polylib::get_group メソッドのラッパー関数

Parameters

in	*group_name*	ポリゴングループ名
out	*tag*	ポリゴングループ

Returns

POLYLIB_STAT で定義される値が返る

7.1.1.2 POLYLIB_STAT polylib_get_leaf_groups_tags (int * *n*, long long int ** *tags*)

リーフPolygonGroup リストの取得（C インターフェース用） Polylib::get_leaf_groups メソッドのラッパー関数

Parameters

out	*n*	PolygonGroup 数
out	*tags*	PolygonGroup タグ

Returns

POLYLIB_STAT で定義される値が返る

Attention

tags は使用後 free して下さい

7.1.1.3 POLYLIB_STAT polylib_get_root_groups_tags (int * *n*, long long int ** *tags*)

PolygonGroup ツリーの最上位ノードの取得（C インターフェース用）Polylib::get_root_groups メソッドのラッパー関数

Parameters

out	*n*	ポリゴングループ数
out	*tags*	ポリゴングループ　タグ

Returns

POLYLIB_STAT で定義される値が返る

Attention

tags は使用後 free して下さい

7.1.1.4 float polylib_group_get_area (long long int *tag_pg*)

PolygonGroup のポリゴンの面積を求める PolygonGroup::get_group_num_tria のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ

Returns

ポリゴンの面積

Attention

全プロセス通したポリゴンの面積が必要な場合は、 polylib_group_get_global_area()を使用する

7.1.1.5 POLYLIB_STAT polylib_group_get_atr (long long int *tag*, char * *key*, char * *val*)

PolygonGroup ユーザ定義属性の取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
in	*key*	キー
out	*val*	属性値

Returns

OK/NG NG:キーと属性のペアが登録されていない

7.1.1.6 void polylib_group_get_children (long long int *tag*, int * *num*, long long int ** *child_tags*)

PolygonGroup 子グループを取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
out	*num*	ポリゴングループ数
out	*child_tags*	ポリゴングループ　タグ

Returns

戻り値なし

Attention

tags は使用後 free して下さい

7.1.1.7 float polylib_group_get_global_area (long long int *tag_pg*)

PolygonGroup のポリゴンの面積 (global) を求める PolygonGroup::get_group_num_global_tria のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ

Returns

ポリゴンの面積 (global)

Attention

並列環境用 ポリゴンの重複を削除するための通信あり polylib_group_get_area()よりも大幅に処理時間がかかることに注意

7.1.1.8 void polylib_group_get_movable (long long int *tag*, int * *movable*)

PolygonGroup 移動対象フラグ取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
out	*moval*	移動対象フラグ 1:true 0:false

7.1.1.9 void polylib_group_get_name (long long int *tag*, char * *name*)

PolygonGroup グループ名取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
out	*name*	グループ名

Returns

戻り値なし

7.1.1.10 int polylib_group_get_num_global_triangles (long long int *tag_pg*)

PolygonGroup のポリゴン要素数 (global) を求める PolygonGroup::get_group_num_global_tria のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ

Returns

三角形ポリゴン数 (global)

Attention

並列環境用 ポリゴンの重複を削除するための通信あり polylib_group_get_num_tria() よりも大幅に処理時間がかかることに注意

7.1.1.11 int polylib_group_get_num_triangles (long long int *tag_pg*)

PolygonGroup のポリゴン要素数を求める PolygonGroup::get_group_num_tria のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ

Returns

三角形ポリゴン数

Attention

全プロセス通した要素数が必要な場合は、 polylib_group_get_num_global_tria() を使用する

7.1.1.12 long long int polylib_group_get_parent (long long int *tag*)

PolygonGroup 親グループを取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ

Returns

親グループのタグ

7.1.1.13 POLYLIB_STAT polylib_group_get_polygons_reduce_atrI (long long int *tag_pg*, PL_OP_TYPE *op*, int *atr_no*, int * *val*)

グループ内のポリゴン属性（整数）の集合演算値を返す 並列化されている場合は全プロセスを通した値 (PL_OP_SUM：重複ポリゴン分は無視される） 全プロセスに同じ値が返る

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ
in	*op*	演算種類　PL_OP_SUM/PL_OP_MAX/PL_OP_MIN
in	*atr_no*	ポリゴン整数属性の何番目か　0〜
out	*val*	属性値

Returns

POLYLIB_STAT で定義される値が返る。 ポリゴンが存在しない ポリゴン属性が存在しないなど

7.1.1.14 POLYLIB_STAT polylib_group_get_polygons_reduce_atrR (long long int *tag_pg*, PL_OP_TYPE *op*, int *atr_no*, float * *val*)

グループ内のポリゴン属性（実数）の集合演算値を返す 並列化されている場合は全プロセスを通した値 (PL_OP_SUM：重複ポリゴン分は無視される） 全プロセスに同じ値が返る

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ
in	*op*	演算種類　PL_OP_SUM/PL_OP_MAX/PL_OP_MIN
in	*atr_no*	ポリゴン実数属性の何番目か　0〜
out	*val*	属性値

Returns

POLYLIB_STAT で定義される値が返る。 ポリゴンが存在しない ポリゴン属性が存在しないなど

7.1.1.15 POLYLIB_STAT polylib_group_get_triangles (long long int *tag_pg*, int * *num_tri*, long long int ** *tags_tri*)

PolygonGroup のポリゴンを求める PolygonGroup::get_triangles のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ
out	*num_tri*	三角形ポリゴン数
out	*tags_tri*	三角形ポリゴンのタグ（ハンドル)

Returns

POLYLIB_STAT で定義される値が返る

Attention

tags_ tris は free してください。

7.1.1.16 void polylib_group_set_atr (long long int *tag*, char * *key*, char * *val*)

PolygonGroup ユーザ定義属性の設定

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
in	*key*	キー
in	*val*	属性値

Returns

なし

Attention

既に登録されていた場合、上書きする

7.1.1.17 void polylib_group_set_movable (long long int *tag*, int *movable*)

PolygonGroup 移動対象フラグ設定

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
in	*moval*	移動対象フラグ 1:true 0:false

Returns

戻り値なし

7.1.1.18 POLYLIB_STAT polylib_group_set_move_func_c (long long int *tag*, void(*)(long long int, PolylibMoveParamsStruct *) *func*)

PolygonGroup に移動関数を登録するためのラッパー関数。 オブジェクトのインスタンス毎に登録が必要

Parameters

in	*tag*	PolygonGroup を操作するためのタグ
in	*func*	移動関数へのポインタ

Returns

POLYLIB_STAT で定義される値が返る。

7.1.1.19 void polylib_group_set_name (long long int *tag*, char * *name*)

PolygonGroup グループ名設定

Parameters

in	*tag*	PolygonGroup を操作するためのタグ

in	*name*	グループ名

Returns

戻り値なし

7.1.1.20 void polylib_group_set_need_rebuild (long long int *tag*)

KD 木の再構築フラグの設定 ユーザ定義の移動関数内の最後で呼び出す

Parameters

in	*tag*	PolygonGroup を操作するためのタグ

Returns

戻り値なし

7.1.1.21 void polylib_group_set_num_polygon_atr (long long int *tag,* int *num_atrI,* int *num_atrR*)

ポリゴングループ内のポリゴンのユーザ定義属性数の設定

Parameters

in	*tag*	PolygonGroup を操作するためのタグ integer∗8 integer∗8
in	*num_atrI*	整数属性数
in	*num_atrR*	実数属性数

Returns

戻り値なし

7.1.1.22 POLYLIB_STAT polylib_init_parallel_info (MPI_Comm *comm,* float *bpos[3],* unsigned int *bbsize[3],* unsigned int *gcsize[3],* float *dx[3]*)

並列計算関連情報の設定と初期化 (各ランクが 1 領域を担当している場合） Polylib::init_parallel_info メソッドのラッパー関数。

Parameters

in	*comm*	MPI コミュニケーター
in	*bpos*	自PE 担当領域の基点座標
in	*bbsize*	同、計算領域のボクセル数
in	*gcsize*	同、ガイドセルのボクセル数
in	*dx*	同、ボクセル 1 辺の長さ

Returns

POLYLIB_STAT で定義される値が返る。

Attention

全 rank で各々設定を行い、その領域情報を全 rank へ配信する

7.1.1.23 POLYLIB_STAT polylib_init_parallel_info2 (MPI_Comm *comm,* int *num,* ParallelBboxStruct ∗ *bboxes*)

並列計算関連情報の設定と初期化 (各ランクが複数領域を担当している場合） 全 rank で各々設定を行い、その領域情報を全 rank へ配信する。

Parameters

in	*comm*	MPI コミュニケーター
in	*num*	自PE が担当する領域
in	*bbox*	担当する boundary box 情報（複数）

Returns

POLYLIB_STAT で定義される値が返る。

7.1.1.24 POLYLIB_STAT polylib_instance (void)

C 言語用Polylib 環境の構築 Polylib インスタンス生成 Polylib::get_instance メソッドの代替関数。

Returns

POLYLIB_STAT で定義される値が返る。

Attention

最初に呼び出すこと

7.1.1.25 POLYLIB_STAT polylib_load (char * *config_name,* float *scale*)

Polylib::load メソッドのラッパー関数。 引数で指定された設定ファイルを読み込み、グループツリーを作成する。 続いて設定ファイルで指定されたSTL ファイルを読み込み、KD 木を作成する。

Parameters

in	*config_name*	設定ファイル名 NULL の場合、デフォルト値 polylib_config.tp とする
in	*scale*	縮尺率

Returns

POLYLIB_STAT で定義される値が返る。

7.1.1.26 POLYLIB_STAT polylib_migrate (void)

ポリゴンデータのPE 間移動 Polylib::mograte メソッドのラッパー関数 オブジェクトのインスタンス毎に登録が必要

Returns

POLYLIB_STAT で定義される値が返る。

7.1.1.27 POLYLIB_STAT polylib_move (PolylibMoveParamsStruct * *param*)

三角形ポリゴン座標の移動 Polylib::move メソッドのラッパー関数 本クラスインスタンス配下の全PolygonGroup の move メソッドが呼び出される。 move 関数は、polylib_group_set_move_func_c 関数で登録する

Parameters

in	*param*	移動計算パラメータセット

Returns

POLYLIB_STAT で定義される値が返る

7.1.1.28 POLYLIB_STAT polylib_save (char ** *p_fname*, char * *format*, char * *extend*)

PolygoGroup ツリー、三角形ポリゴン情報の保存。 Polylib::save メソッドのラッパー関数。 グループツリーの情報を設定ファイルへ出力。三角形ポリゴン情報をSTL/NPT ファイルへ出力

Parameters

out	*p_fname*	設定ファイル名返却用 内部領域のアドレスを返す free 不可
in	*format*	形状ファイルのフォーマット PolylibDefine.h で定義されているFILE_FMT_* 参照
in	*extend*	ファイル名に付加する文字列。NULL を指定した 場合は、付加文字列として本メソッド呼び出し時の 年月日時分秒 (YYYYMMDD24hhmmss) を用いる

Returns

POLYLIB_STAT で定義される値が返る。

Attention

ファイル名命名規約は次の通り。 設定ファイル : polylib_config_付加文字.tpp STL/NPT ファイル : ポリゴングループ名_付加文字.拡張子

7.1.1.29 POLYLIB_STAT polylib_search_nearest_polygon (long long int * *tag*, char * *group_name*, float *pos[3]*)

指定した点に最も近いポリゴンの検索 Polylib::search_nearest_polygon メソッドのラッパー関数。

Parameters

out	*tag*	ポリゴンのタグ（ハンドル) PL_NULL_TAG 検索なし
in	*group_name*	グループ名。
in	*pos*	指定点

Returns

POLYLIB_STAT で定義される値が返る MPI 並列計算時は,pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

7.1.1.30 POLYLIB_STAT polylib_search_polygons (int * *num*, long long int ** *tags*, char * *group_name*, float *min_pos[3]*, float *max_pos[3]*, int *every*)

Polylib::search_polygons メソッドのラッパー関数。 位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 特定のグループとその子孫グループに属する三角形ポリゴンをKD 探索に より抽出する。Polylib 内でメモリ領域が確保される

Parameters

out	*num*	抽出された三角形ポリゴン数
out	*tags*	三角形ポリゴンのタグ（ハンドル)
in	*group_name*	抽出グループ名。
in	*min_pos*	抽出する矩形領域の最小値。(x,y,z 順の配列)
in	*max_pos*	抽出する矩形領域の最大値。(x,y,z 順の配列)
in	*every*	抽出オプション。 1：3 頂点が全て検索領域に含まれるポリゴンを抽出する。 0：三角形のBBox が一部でも検索領域と交差するものを抽出する

Returns

POLYLIB_STAT で定義される値が返る

Attention

tags,tris は free してください。 MPI 並列計算時は,min_pos, max_pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

7.1.1.31 POLYLIB_STAT polylib_search_polygons_npt (int * *num*, long long int ** *tags*, NptTriangleStruct ** *tris*, char * *group_name*, float *min_pos[3]*, float *max_pos[3]*, int *every*)

Polylib::search_polygons メソッドのラッパー関数。 位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 特定のグループとその子孫グループに属する三角形ポリゴンをKD 探索に より抽出する。Polylib 内でメモリ領域が確保される

Parameters

out	*num*	抽出された三角形 (長田パッチ）ポリゴン数
out	*tags*	三角形 (長田パッチ）ポリゴンのタグ（ハンドル)
out	*tris*	三角形 (長田パッチ）ポリゴン
in	*group_name*	抽出グループ名。
in	*min_pos*	抽出する矩形領域の最小値。(x,y,z 順の配列)
in	*max_pos*	抽出する矩形領域の最大値。(x,y,z 順の配列) 1：3 頂点が全て検索領域に含まれるポリゴンを抽出する。 0：三角形のBBox が一部でも検索領域と交差するものを抽出する。
in	*every*	抽出オプション。

Returns

POLYLIB_STAT で定義される値が返る

Attention

tags,tris は free してください。 MPI 並列計算時は,min_pos, max_pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため） ポリゴンに長田パッチ以外が含まれている場合エラーになります。

7.1.1.32 POLYLIB_STAT polylib_search_polygons_triangle (int * *num*, long long int ** *tags*, TriangleStruct ** *tris*, char * *group_name*, float *min_pos[3]*, float *max_pos[3]*, int *every*)

Polylib::search_polygons メソッドのラッパー関数。 位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 特定のグループとその子孫グループに属する三角形ポリゴンをKD 探索に より抽出する。Polylib 内でメモリ領域が確保される。

Parameters

out	*num*	抽出された三角形ポリゴン数
out	*tags*	三角形ポリゴンのタグ（ハンドル)
out	*tris*	三角形ポリゴン
in	*group_name*	抽出グループ名。
in	*min_pos*	抽出する矩形領域の最小値。(x,y,z 順の配列)
in	*max_pos*	抽出する矩形領域の最大値。(x,y,z 順の配列)
in	*every*	抽出オプション。 1：3 頂点が全て検索領域に含まれるポリゴンを抽出する。 0：三角形のBBox が一部でも検索領域と交差するものを抽出する。

Returns

POLYLIB_STAT で定義される値が返る

Attention

tags,tris は free してください。 MPI 並列計算時は,min_pos, max_pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため） ポリゴンに長田パッチが含まれている場合エラーになります。

7.1.1.33 void polylib_show_group_hierarchy ( )

Polylib::show_group_hierarchy メソッドのラッパー関数。 グループ階層構造リストを標準出力に出力する。

7.1.1.34 POLYLIB_STAT polylib_show_group_info (char * *group_name*)

グループの情報を出力する。(親グループ名、自身の名前、ファイル名、 Polylib::show_group_info メソッドのラッパー関数。 登録三角形数、3 頂点ベクトルの座標、法線ベクトルの座標、面積)

Parameters

in	*group_name*	グループ名

Returns

POLYLIB_STAT で定義される値が返る。

7.1.1.35 void polylib_triangle_get_normal (long long int *tag,* float *norm[3]*)

法線ベクトル取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ
out	*norm*	法線ベクトル

Returns

戻り値なし

7.1.1.36 POLYLIB_STAT polylib_triangle_get_npatchParam (long long int *tag,* NpatchParamStruct * *param*)

長田パッチパラメータ取得

Parameters

in	*tag*	NptTriangle を操作するためのタグ
out	*param*	長田パッチパラメータ

Returns

POLYLIB_STAT で定義される値が返る。 オブジェクトが長田パッチでない時はエラーを返す

7.1.1.37 int polylib_triangle_get_num_atrI (long long int *tag*)

ユーザ定義属性数（整数型）の取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ

Returns

整数属性数

7.1.1.38 int polylib_triangle_get_num_atrR (long long int *tag*)

ユーザ定義属性数（実数型）の取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ

Returns

実数属性数

7.1.1.39 int* polylib_triangle_get_pAtrI (long long int *tag*)

ユーザ定義属性（整数型）のポインタ取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ

Returns

ユーザ定義属性（整数型）へのポインタ free してはならない

7.1.1.40 float* polylib_triangle_get_pAtrR (long long int *tag*)

ユーザ定義属性（実数型）のポインタ取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ

Returns

ユーザ定義属性（実数型）へのポインタ free してはならない

7.1.1.41 int polylib_triangle_get_pl_type (long long int *tag*)

ポリゴンタイプ取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ

Returns

PL_TYPE_TRIANGLE / PL_TYPE_NPT（長田パッチ）

7.1.1.42 void polylib_triangle_get_vertexes (long long int *tag*, float *vertex[9]*)

頂点座標取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ
out	*vertex*	３頂点の座標

Returns

戻り値なし

7.1.1.43 POLYLIB_STAT polylib_triangle_set_npatchParam (long long int *tag*, NpatchParamStruct * *param*)

長田パッチパラメータ設定

Parameters

in	*tag*	NptTriangle を操作するためのタグ
in	*param*	長田パッチパラメータ

Returns

POLYLIB_STAT で定義される値が返る。 オブジェクトが長田パッチでない時はエラーを返す

7.1.1.44 void polylib_triangle_set_vertexes (long long int *tag*, float *vertex[9]*)

頂点座標設定 基本はTriangle 用

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ
in	*vertex*	３頂点の座標

Returns

戻り値なし

Attention

頂点設定時、法線ベクトル, 面積も内部で設定する 長田パッチのパラメータは更新されないので注意

7.1.1.45 void polylib_triangle_set_vertexes_npatch (long long int *tag*, float *vertex[9]*, NpatchParamStruct * *param*)

頂点座標・長田パッチパラメータ設定

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ
in	*vertex*	３頂点の座標
in	*param*	長田パッチパラメータ

Returns

戻り値なし

Attention

頂点設定時、法線ベクトル, 面積も内部で設定する

7.1.1.46 size_t polylib_used_memory_size ( )

Polylib が利用中の概算メモリ量を返す

Returns

利用中のメモリ量 (byte)

7.1.1.47 size_t polylib_used_memory_size_mb ( )

Polylib が利用中の概算メモリ量 (MB) を返す

Returns

利用中のメモリ量 (Mbyte)

7.2 common/BBox.h File Reference

```
#include <algorithm>
#include <list>
#include "PolylibCommon.h"
#include "common/Vec2.h"
#include "common/Vec3.h"
```

Classes

- class PolylibNS::BBox

Namespaces

- PolylibNS

7.3 common/PolylibCommon.h File Reference

```
#include <iostream>
#include "common/PolylibDefine.h"
```

Namespaces

- PolylibNS

Macros

- #define VEC3_TO_REAL(v, r)
- #define REAL_TO_VEC3(r, v)
- #define VEC3_3_TO_REAL(v, r1, r2, r3)
- #define REAL_TO_VEC3_3(r1, r2, r3, v)
- #define VEC3_3_TO_REAL9(v, r)
- #define REAL9_TO_VEC3_3(r, v)
- #define PL_DBGOS std::cout
- #define PL_DBGOSH std::cout<<gs_rankno<<"PL:"
- #define PL_ERROS std::cerr
- #define PL_ERROSH std::cerr<<gs_rankno<<"PL:"

Enumerations

- enum PolylibNS::ID_FORMAT { PolylibNS::ID_BIN, PolylibNS::ID_ASCII }

Variables

- std::string PolylibNS::gs_rankno

7.3.1 Macro Definition Documentation

7.3.1.1 #define PL_DBGOS std::cout

デバッグ出力先、エラー時出力先

7.3.1.2 #define PL_DBGOSH std::cout<<gs_rankno<<"PL:"

7.3.1.3 #define PL_ERROS std::cerr

7.3.1.4 #define PL_ERROSH std::cerr<<gs_rankno<<"PL:"

7.3.1.5 #define REAL9_TO_VEC3_3(*r*, *v*)

Value:

```
{   \
                (v)[0].x=(r)[0];   (v)[0].y=(r)[1];   (v)[0].z=(r)[2]; \
                (v)[1].x=(r)[3];   (v)[1].y=(r)[4];   (v)[1].z=(r)[5]; \
                (v)[2].x=(r)[6];   (v)[2].y=(r)[7];   (v)[2].z=(r)[8]; \
            }
```

7.3.1.6 #define REAL_TO_VEC3(*r*, *v*)

Value:

```
{   \
                (v).x=(r)[0];   (v).y=(r)[1];   (v).z=(r)[2]; \
            }
```

7.3.1.7 #define REAL_TO_VEC3_3(r1, r2, r3, v)

Value:

```
{   \
                (v)[0].x=(r1)[0];   (v)[0].y=(r1)[1];   (v)[0].z=(r1)[2]; \
                (v)[1].x=(r2)[0];   (v)[1].y=(r2)[1];   (v)[1].z=(r2)[2]; \
                (v)[2].x=(r3)[0];   (v)[2].y=(r3)[1];   (v)[2].z=(r3)[2]; \
            }
```

7.3.1.8 #define VEC3_3_TO_REAL(v, r1, r2, r3)

Value:

```
{   \
                (r1)[0]=(v)[0].x;   (r1)[1]=(v)[0].y;   (r1)[2]=(v)[0].z; \
                (r2)[0]=(v)[1].x;   (r2)[1]=(v)[1].y;   (r2)[2]=(v)[1].z; \
                (r3)[0]=(v)[2].x;   (r3)[1]=(v)[2].y;   (r3)[2]=(v)[2].z; \
            }
```

7.3.1.9 #define VEC3_3_TO_REAL9(v, r)

Value:

```
{   \
                (r)[0]=(v)[0].x;   (r)[1]=(v)[0].y;   (r)[2]=(v)[0].z; \
                (r)[3]=(v)[1].x;   (r)[4]=(v)[1].y;   (r)[5]=(v)[1].z; \
                (r)[6]=(v)[2].x;   (r)[7]=(v)[2].y;   (r)[8]=(v)[2].z; \
            }
```

7.3.1.10 #define VEC3_TO_REAL(v, r)

Value:

```
{   \
                (r)[0]=(v).x;   (r)[1]=(v).y;   (r)[2]=(v).z; \
            }
```

Vec3 型とプリミティブ型の型変換

7.4 common/PolylibDefine.h File Reference

```
#include <limits.h>
#include <float.h>
```

Macros

- #define PL_REAL float
- #define PL_MPI_REAL MPI_FLOAT
- #define PL_INT_MAX INT_MAX
- #define PL_INT_MIN INT_MIN
- #define PL_REAL_MAX FLT_MAX
- #define PL_REAL_MIN FLT_MIN
- #define PL_GRP_TAG long long int

- #define PL_ELM_TAG long long int
- #define PL_NULL_TAG 0
- #define PL_TYPE_UNKNOWN 0
- #define PL_TYPE_TRIANGLE 1
- #define PL_TYPE_NPT 2
- #define FILE_FMT_STL_A "stl_a"
- #define FILE_FMT_STL_AA "stl_aa"
- #define FILE_FMT_STL_B "stl_b"
- #define FILE_FMT_STL_BB "stl_bb"
- #define FILE_FMT_NPT_A "npt_a"
- #define FILE_FMT_NPT_B "npt_b"
- #define FILE_FMT_DEFAULT FILE_FMT_STL_B

Enumerations

- enum PL_OP_TYPE { PL_OP_SUM = 1, PL_OP_MAX = 2, PL_OP_MIN = 3 }

7.4.1 Macro Definition Documentation

7.4.1.1 #define FILE_FMT_DEFAULT FILE_FMT_STL_B

7.4.1.2 #define FILE_FMT_NPT_A "npt_a"

7.4.1.3 #define FILE_FMT_NPT_B "npt_b"

7.4.1.4 #define FILE_FMT_STL_A "stl_a"

7.4.1.5 #define FILE_FMT_STL_AA "stl_aa"

7.4.1.6 #define FILE_FMT_STL_B "stl_b"

7.4.1.7 #define FILE_FMT_STL_BB "stl_bb"

7.4.1.8 #define PL_ELM_TAG long long int

7.4.1.9 #define PL_GRP_TAG long long int

7.4.1.10 #define PL_INT_MAX INT_MAX

7.4.1.11 #define PL_INT_MIN INT_MIN

7.4.1.12 #define PL_MPI_REAL MPI_FLOAT

7.4.1.13 #define PL_NULL_TAG 0

7.4.1.14 #define PL_REAL float

実数型の指定

- デフォルトでは、PL_REAL=float
- コンパイル時オプション-D_REAL_IS_DOUBLE_を付与することで PL_REAL=double になる

7.4.1.15 #define PL_REAL_MAX FLT_MAX

7.4.1.16 #define PL_REAL_MIN FLT_MIN

7.4.1.17 #define PL_TYPE_NPT 2

7.4.1.18 #define PL_TYPE_TRIANGLE 1

7.4.1.19 #define PL_TYPE_UNKNOWN 0

7.4.2 Enumeration Type Documentation

7.4.2.1 enum PL_OP_TYPE

Enumerator

PL_OP_SUM 総和

PL_OP_MAX MAX.

PL_OP_MIN MIN.

7.5 common/PolylibStat.h File Reference

Enumerations

- enum POLYLIB_STAT {
  PLSTAT_OK = 0, PLSTAT_NG = 1, PLSTAT_INSTANCE_EXISTED = 2, PLSTAT_INSTANCE_NOT_EXIST = 3,
  PLSTAT_MPI_ERROR = 5, PLSTAT_ARGUMENT_NULL = 6, PLSTAT_MEMORY_NOT_ALLOC = 7, PLSTAT_LACK_OF_MEMORY = 8,
  PLSTAT_CONFIG_ERROR = 10, PLSTAT_STL_IO_ERROR = 11, PLSTAT_NPT_IO_ERROR = 12, PLSTAT_UNKNOWN_FILE_FORMAT = 13,
  PLSTAT_LACK_OF_LOAD_MEMORY = 14, PLSTAT_FILE_NOT_SET = 20, PLSTAT_GROUP_NOT_FOUND = 21, PLSTAT_GROUP_NAME_EMPTY = 22,
  PLSTAT_GROUP_NAME_DUP = 23, PLSTAT_POLYGON_NOT_EXIST = 24, PLSTAT_NODE_NOT_FIND = 26, PLSTAT_ROOT_NODE_NOT_EXIST = 27,
  PLSTAT_NOT_NPT = 28, PLSTAT_ATR_NOT_EXIST = 29 }

7.5.1 Enumeration Type Documentation

7.5.1.1 enum POLYLIB_STAT

Polylib で利用するEnum の定義

Enumerator

PLSTAT_OK 処理が成功した。

PLSTAT_NG 一般的なエラー。

PLSTAT_INSTANCE_EXISTED Polylib インスタンスがすでに存在している。

PLSTAT_INSTANCE_NOT_EXIST Polylib インスタンスが存在しない。

PLSTAT_MPI_ERROR MPI 関数がエラーを戻した。

PLSTAT_ARGUMENT_NULL 引数のメモリ確保が行われていない。

PLSTAT_MEMORY_NOT_ALLOC メモリ確保に失敗した。

PLSTAT_LACK_OF_MEMORY メモリ不足

PLSTAT_CONFIG_ERROR 定義ファイルでエラー発生
PLSTAT_STL_IO_ERROR STL ファイルIO エラー
PLSTAT_NPT_IO_ERROR 長田パッチファイルIO エラー
PLSTAT_UNKNOWN_FILE_FORMAT ファイルが.stla、.stlb、.stl、.npta、.nptb、.npt 以外。
PLSTAT_LACK_OF_LOAD_MEMORY ロード処理時のメモリ不足（メモリ削減版使用時）
PLSTAT_FILE_NOT_SET リーフグループにファイル名が未設定。
PLSTAT_GROUP_NOT_FOUND グループ名がPolylib に未登録。
PLSTAT_GROUP_NAME_EMPTY グループ名が空である。
PLSTAT_GROUP_NAME_DUP グループ名が重複している。
PLSTAT_POLYGON_NOT_EXIST ポリゴンが存在しない
PLSTAT_NODE_NOT_FIND KD 木生成時に検索点が見つからなかった。
PLSTAT_ROOT_NODE_NOT_EXIST KD 木のルートノードが存在しない。
PLSTAT_NOT_NPT 長田パッチではない（NptTriangle∗ への dynamic cast 失敗など）
PLSTAT_ATR_NOT_EXIST 属性が未設定

7.6 common/tt.h File Reference

```
#include <math.h>
```

Macros

- #define M_PI 3.14159265358979323846 /∗ pi ∗/
- #define sqrtf(x) (float)sqrt((double)(x))
- #define GL_GLEXT_PROTOTYPES 1

Typedefs

- typedef unsigned char uchar
- typedef unsigned short ushort
- typedef unsigned int uint
- typedef unsigned long ulong

7.6.1 Macro Definition Documentation

7.6.1.1 #define GL_GLEXT_PROTOTYPES 1

7.6.1.2 #define M_PI 3.14159265358979323846 /∗ pi ∗/

7.6.1.3 #define sqrtf(*x*) (float)sqrt((double)(x))

7.6.2 Typedef Documentation

7.6.2.1 typedef unsigned char uchar

7.6.2.2 typedef unsigned int uint

7.6.2.3 typedef unsigned long ulong

7.6.2.4 typedef unsigned short ushort

7.7 common/Vec2.h File Reference

```
#include <iostream>
#include <math.h>
```

Classes

- class PolylibNS::Vec2< T >

Namespaces

- PolylibNS

Macros

- #define REAL_TYPE float

Typedefs

- typedef Vec2< unsigned char > PolylibNS::Vec2uc
- typedef Vec2< int > PolylibNS::Vec2i
- typedef Vec2< float > PolylibNS::Vec2f
- typedef Vec2< float > PolylibNS::Vec2r

Functions

- Vec2< float > PolylibNS::operator* (float s, const Vec2< float > &v)
- float PolylibNS::distanceSquared (const Vec2< float > &a, const Vec2< float > &b)
- float PolylibNS::distance (const Vec2< float > &a, const Vec2< float > &b)
- template<typename T >
  std::istream & PolylibNS::operator>> (std::istream &is, Vec2< T > &v)
- template<typename T >
  std::ostream & PolylibNS::operator<< (std::ostream &os, const Vec2< T > &v)
- std::istream & PolylibNS::operator>> (std::istream &is, Vec2uc &v)
- std::ostream & PolylibNS::operator<< (std::ostream &os, const Vec2uc &v)
- bool PolylibNS::lessVec2f (const Vec2< float > &a, const Vec2< float > &b)

7.7.1 Macro Definition Documentation

7.7.1.1 #define REAL_TYPE float

実数型の指定

- デフォルトでは、REAL_TYPE=float
- コンパイル時オプション-D_REAL_IS_DOUBLE_を付与することで REAL_TYPE=double になる

7.8 common/Vec3.h File Reference

version 1.1 2014-04-23

```
#include <iostream>
#include <math.h>
```

Classes

- class Vec3class::Vec3< T >

Namespaces

- Vec3class

Macros

- #define REAL_TYPE float

Typedefs

- typedef Vec3< unsigned char > Vec3class::Vec3uc
- typedef Vec3< int > Vec3class::Vec3i
- typedef Vec3< float > Vec3class::Vec3f
- typedef Vec3< double > Vec3class::Vec3d
- typedef Vec3< float > Vec3class::Vec3r

Enumerations

- enum Vec3class::AxisEnum { Vec3class::AXIS_X = 0, Vec3class::AXIS_Y, Vec3class::AXIS_Z, Vec3class::AXIS_ERROR }

Functions

- template<typename T >
  Vec3< T > Vec3class::operator* (T s, const Vec3< T > &v)
- template<typename T >
  Vec3< T > Vec3class::multi (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  T Vec3class::dot (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  Vec3< T > Vec3class::cross (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  T Vec3class::distanceSquared (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  T Vec3class::distance (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  bool Vec3class::lessVec3f (const Vec3< T > &a, const Vec3< T > &b)
- template<typename T >
  std::istream & Vec3class::operator>> (std::istream &is, Vec3< T > &v)
- template<typename T >
  std::ostream & Vec3class::operator<< (std::ostream &os, const Vec3< T > &v)

- std::istream & Vec3class::operator>> (std::istream &is, Vec3uc &v)
- std::ostream & Vec3class::operator<< (std::ostream &os, const Vec3uc &v)

7.8.1 Detailed Description

version 1.1 2014-04-23

Author

aics

7.8.2 Macro Definition Documentation

7.8.2.1 #define REAL_TYPE float

実数型の指定

- デフォルトでは、REAL_TYPE=float
- コンパイル時オプション-D_REAL_IS_DOUBLE_を付与することで REAL_TYPE=double になる

7.9 f_lang/FPolylib.h File Reference

```
#include "mpi.h"
#include <stdbool.h>
#include "c_lang/CPolylib.h"
```

Classes

- struct FParallelBboxStruct

Macros

- #define POLYLIB_FALSE 0
- #define POLYLIB_TRUE 1
- #define PL_GRP_PATH_LEN 256

  ポリゴングループ名の *fortran* 文字列長
- #define PL_GRP_NAME_LEN 64
- #define PL_GRP_ATR_LEN 32
- #define PL_FILE_PATH_LEN 256

  ファイルパス、ファイル名の *fortran* 文字列長
- #define PL_FILE_NAME_LEN 64
- #define PL_FORMAT_LEN 8

  ファイルフォーマットの *fortran* 文字列長
- #define PL_STR_LEN 32

Functions

- void fpolylib_instance_ (POLYLIB_STAT *ret)
- void fpolylib_init_parallel_info_ (PL_REAL bpos[3], int bbsize[3], int gcsize[3], PL_REAL dx[3], POLYLIB_STAT *ret)
- void fpolylib_init_parallel_info2_ (int *num, FParallelBboxStruct *bbox, POLYLIB_STAT *ret)
- void fpolylib_load_ (char *config_name, PL_REAL *scale, POLYLIB_STAT *ret)
- void fpolylib_save_ (char *o_fname, char *format, char *extend, POLYLIB_STAT *ret)
- void fpolylib_move_ (PolylibMoveParamsStruct *param, POLYLIB_STAT *ret)
- void fpolylib_migrate_ (POLYLIB_STAT *ret)
- void fpolylib_get_group_tag_ (char *group_name, PL_GRP_TAG *grp_tag, POLYLIB_STAT *ret)
- void fpolylib_get_root_groups_tags_num_ (int *n, POLYLIB_STAT *ret)
- void fpolylib_get_root_groups_tags_ (int *n, PL_GRP_TAG *tags, POLYLIB_STAT *ret)
- void fpolylib_get_leaf_groups_tags_num_ (int *n, POLYLIB_STAT *ret)
- void fpolylib_get_leaf_groups_tags_ (int *n, PL_GRP_TAG *tags, POLYLIB_STAT *ret)
- void fpolylib_search_polygons_num_ (int *num, char *group_name, PL_REAL min_pos[3], PL_REAL max_pos[3], int *every, POLYLIB_STAT *ret)
- void fpolylib_search_polygons_ (int *num, PL_ELM_TAG *tags, char *group_name, PL_REAL min_pos[3], PL_REAL max_pos[3], int *every, POLYLIB_STAT *ret)
- void fpolylib_search_nearest_polygon_ (PL_ELM_TAG *tag, char *group_name, PL_REAL pos[3], POLYLIB_STAT *ret)
- void fpolylib_show_group_hierarchy_ ()
- void fpolylib_show_group_info_ (char *group_name, POLYLIB_STAT *ret)
- int fpolylib_used_memory_size_ ()
- int fpolylib_used_memory_size_mb_ ()
- void fpolylib_group_get_triangles_ (PL_GRP_TAG *tag_pg, int *num_tri, PL_ELM_TAG *tags_tri, POLYLIB_STAT *ret)
- void fpolylib_group_get_num_triangles_ (PL_GRP_TAG *tag_pg, int *num_tri, POLYLIB_STAT *ret)
- void fpolylib_group_get_num_global_triangles_ (PL_GRP_TAG *tag_pg, int *num_tri, POLYLIB_STAT *ret)
- void fpolylib_group_get_area_ (PL_GRP_TAG *tag_pg, PL_REAL *area, POLYLIB_STAT *ret)
- void fpolylib_group_get_global_area_ (PL_GRP_TAG *tag_pg, PL_REAL *area, POLYLIB_STAT *ret)
- void fpolylib_group_get_polygons_reduce_atrI_ (PL_GRP_TAG *tag_pg, PL_OP_TYPE *op, int *atr_no, int *val, POLYLIB_STAT *ret)
- void fpolylib_group_get_polygons_reduce_atrR_ (PL_GRP_TAG *tag_pg, PL_OP_TYPE *op, int *atr_no, PL_REAL *val, POLYLIB_STAT *ret)
- void fpolylib_group_set_need_rebuild_ (PL_GRP_TAG *tag_pg)
- void fpolylib_group_set_name_ (PL_GRP_TAG *tag, char *name)
- void fpolylib_group_get_name_ (PL_GRP_TAG *tag, char *name)
- void fpolylib_group_set_movable_ (PL_GRP_TAG *tag, int *movable)

  移動対象フラグ設定
- void fpolylib_group_get_movable_ (PL_GRP_TAG *tag, int *movable)

  移動対象フラグ取得
- void fpolylib_group_set_atr_ (PL_GRP_TAG *tag, char *key, char *str)

  ユーザ定義属性の設定
- void fpolylib_group_get_atr_ (PL_GRP_TAG *tag, char *key, char *str, POLYLIB_STAT *ret)

  ユーザ定義属性の取得
- void fpolylib_group_set_num_polygon_atr_ (PL_GRP_TAG *tag, int *num_atrI, int *num_atrR)
- void fpolylib_group_get_parent_ (PL_GRP_TAG *tag, PL_GRP_TAG *parent_tag)
- void fpolylib_group_get_children_num_ (PL_GRP_TAG *tag, int *num)
- void fpolylib_group_get_children_ (PL_GRP_TAG *tag, int *num, PL_GRP_TAG *child_tags)
- int fpolylib_triangle_get_pl_type_ (PL_ELM_TAG *tag)
- void fpolylib_triangle_set_vertexes_ (PL_ELM_TAG *tag, PL_REAL vertex[9])
- void fpolylib_triangle_set_vertexes_npatch_ (PL_ELM_TAG *tag, PL_REAL vertex[9], NpatchParamStruct *param)

- void fpolylib_triangle_set_vertexes_npatch2_ (PL_ELM_TAG *tag, PL_REAL vertex[9], PL_REAL cp_side1_1[3], PL_REAL cp_side1_2[3], PL_REAL cp_side2_1[3], PL_REAL cp_side2_2[3], PL_REAL cp_side3_1[3], PL_REAL cp_side3_2[3], PL_REAL cp_center[3])
- void fpolylib_triangle_get_vertexes_ (PL_ELM_TAG *tag, PL_REAL vertex[9])
- void fpolylib_triangle_get_normal_ (PL_ELM_TAG *tag, PL_REAL norm[3])
- void fpolylib_triangle_set_npatchParam_ (PL_ELM_TAG *tag, NpatchParamStruct *param, POLYLIB_STAT *ret)
- void fpolylib_triangle_set_npatchParam2_ (PL_ELM_TAG *tag, PL_REAL cp_side1_1[3], PL_REAL cp_side1_2[3], PL_REAL cp_side2_1[3], PL_REAL cp_side2_2[3], PL_REAL cp_side3_1[3], PL_REAL cp_side3_2[3], PL_REAL cp_center[3], POLYLIB_STAT *ret)
- void fpolylib_triangle_get_npatchParam_ (PL_ELM_TAG *tag, NpatchParamStruct *param, POLYLIB_STAT *ret)
- void fpolylib_triangle_get_npatchParam2_ (PL_ELM_TAG *tag, PL_REAL cp_side1_1[3], PL_REAL cp_side1_2[3], PL_REAL cp_side2_1[3], PL_REAL cp_side2_2[3], PL_REAL cp_side3_1[3], PL_REAL cp_side3_2[3], PL_REAL cp_center[3], POLYLIB_STAT *ret)
- int fpolylib_triangle_get_num_atrI_ (PL_ELM_TAG *tag)
- int fpolylib_triangle_get_num_atrR_ (PL_ELM_TAG *tag)
- int fpolylib_triangle_get_atrI_ (PL_ELM_TAG *tag, int *atr_no)
- void fpolylib_triangle_set_atrI_ (PL_ELM_TAG *tag, int *atr_no, int *val)
- PL_REAL fpolylib_triangle_get_atrR_ (PL_ELM_TAG *tag, int *atr_no)
- void fpolylib_triangle_set_atrR_ (PL_ELM_TAG *tag, int *atr_no, PL_REAL *val)

7.9.1 Macro Definition Documentation

7.9.1.1 #define PL_FILE_NAME_LEN 64

7.9.1.2 #define PL_FILE_PATH_LEN 256

ファイルパス、ファイル名の fortran 文字列長

7.9.1.3 #define PL_FORMAT_LEN 8

ファイルフォーマットの fortran 文字列長

7.9.1.4 #define PL_GRP_ATR_LEN 32

属性用文字列の fortran 文字列長 属性キー, 属性値 etc.

7.9.1.5 #define PL_GRP_NAME_LEN 64

7.9.1.6 #define PL_GRP_PATH_LEN 256

ポリゴングループ名の fortran 文字列長

7.9.1.7 #define PL_STR_LEN 32

文字列の fortran 文字列長 その他

7.9.1.8 #define POLYLIB_FALSE 0

Fortran インターフェース実装用のヘッダ Fortran アプリ側から使用するヘッダではありません

7.9.1.9 #define POLYLIB_TRUE 1

7.9.2 Function Documentation

7.9.2.1 void fpolylib_get_group_tag_ (char * *group_name*, PL_GRP_TAG * *grp_tag*, POLYLIB_STAT * *ret*)

ポリゴングループタグの取得

Parameters

in	*group_name*	ポリゴングループ名 character*256
out	*tag*	ポリゴングループ integer*8
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.2 void fpolylib_get_leaf_groups_tags_ (int * *n*, PL_GRP_TAG * *tags*, POLYLIB_STAT * *ret*)

リーフPolygonGroup リストの取得

Parameters

out	*n*	ポリゴングループ数
out	*tags*	ポリゴングループ　タグ integer*8 get_leaf_groups_tags_num_() で得られる個数以上を allocate 済みであること
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.3 void fpolylib_get_leaf_groups_tags_num_ (int * *n*, POLYLIB_STAT * *ret*)

リーフのポリゴングループの個数取得

Parameters

out	*n*	ポリゴングループ数
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.4 void fpolylib_get_root_groups_tags_ (int * *n*, PL_GRP_TAG * *tags*, POLYLIB_STAT * *ret*)

PolygonGroup ツリーの最上位ノードの取得（C インターフェース用） Polylib::get_root_groups_tags メソッドのラッパー関数

Parameters

out	*n*	ポリゴングループ数
out	*tags*	ポリゴングループ　タグ integer*8 get_root_groups_tags_num_() で得られる個数以上を allocate 済みであること
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.5 void fpolylib_get_root_groups_tags_num_ (int * *n*, POLYLIB_STAT * *ret*)

ルートのポリゴングループの個数取得

Parameters

out	*n*	ポリゴングループ数
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.6 void fpolylib_group_get_area_ (PL_GRP_TAG * *tag_pg*, PL_REAL * *area*, POLYLIB_STAT * *ret*)

PolygonGroup のポリゴンの面積を求める PolygonGroup::get_triangles のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ integer∗8
out	*area*	ポリゴンの面積
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.7 void fpolylib_group_get_atr_ (PL_GRP_TAG ∗ *tag*, char ∗ *key*, char ∗ *str*, POLYLIB_STAT ∗ *ret*)

ユーザ定義属性の取得

7.9.2.8 void fpolylib_group_get_children_ (PL_GRP_TAG ∗ *tag*, int ∗ *num*, PL_GRP_TAG ∗ *child_tags*)

子グループを取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ integer∗8
out	*num*	ポリゴングループ数
out	*child_tags*	ポリゴングループ　タグ integer∗8

Returns

戻り値なし

7.9.2.9 void fpolylib_group_get_children_num_ (PL_GRP_TAG ∗ *tag*, int ∗ *num*)

子グループ数を取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ integer∗8
out	*num*	ポリゴングループ数

Returns

戻り値なし

7.9.2.10 void fpolylib_group_get_global_area_ (PL_GRP_TAG ∗ *tag_pg*, PL_REAL ∗ *area*, POLYLIB_STAT ∗ *ret*)

PolygonGroup のポリゴンの面積 (global) を求める PolygonGroup::get_triangles のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ integer∗8
out	*area*	ポリゴンの面積 (global)
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.11 void fpolylib_group_get_movable_ (PL_GRP_TAG ∗ *tag*, int ∗ *movable*)

移動対象フラグ取得

7.9.2.12 void fpolylib_group_get_name_ (PL_GRP_TAG * *tag*, char * *name*)

7.9.2.13 void fpolylib_group_get_num_global_triangles_ (PL_GRP_TAG * *tag_pg*, int * *num_tri*, POLYLIB_STAT * *ret*)

PolygonGroup のポリゴン数 (global) を求める PolygonGroup::get_triangles のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ integer∗8
out	*num_tri*	三角形ポリゴン数 (global)
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.14 void fpolylib_group_get_num_triangles_ (PL_GRP_TAG ∗ *tag_pg*, int ∗ *num_tri*, POLYLIB_STAT ∗ *ret*)

PolygonGroup のポリゴン数を求める PolygonGroup::get_triangles のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ integer∗8
out	*num_tri*	三角形ポリゴン数
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.15 void fpolylib_group_get_parent_ (PL_GRP_TAG ∗ *tag*, PL_GRP_TAG ∗ *parent_tag*)

PolygonGroup 親グループを取得

Parameters

in	*tag*	PolygonGroup を操作するためのタグ integer∗8
out	*parent_tag*	親グループのタグ integer∗8

7.9.2.16 void fpolylib_group_get_polygons_reduce_atrI_ (PL_GRP_TAG ∗ *tag_pg*, PL_OP_TYPE ∗ *op*, int ∗ *atr_no*, int ∗ *val*, POLYLIB_STAT ∗ *ret*)

グループ内のポリゴン属性（整数）の集合演算値を返す 並列化されている場合は全プロセスを通した値 (PL_OP_SUM：重複ポリゴン分は無視される） 全プロセスに同じ値が返る

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ
in	*op*	演算種類　PL_OP_SUM/PL_OP_MAX/PL_OP_MIN
in	*atr_no*	ポリゴン整数属性の何番目か　0〜
out	*val*	属性値
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.17 void fpolylib_group_get_polygons_reduce_atrR_ (PL_GRP_TAG ∗ *tag_pg*, PL_OP_TYPE ∗ *op*, int ∗ *atr_no*, PL_REAL ∗ *val*, POLYLIB_STAT ∗ *ret*)

グループ内のポリゴン属性（実数）の集合演算値を返す 並列化されている場合は全プロセスを通した値 (PL_OP_SUM：重複ポリゴン分は無視される） 全プロセスに同じ値が返る

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ
in	*op*	演算種類　PL_OP_SUM/PL_OP_MAX/PL_OP_MIN
in	*atr_no*	ポリゴン実数属性の何番目か　0〜
out	*val*	属性値
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.18 void fpolylib_group_get_triangles_ (PL_GRP_TAG * *tag_pg*, int * *num_tri*, PL_ELM_TAG * *tags_tri*, POLYLIB_STAT * *ret*)

PolygonGroup のポリゴンを求める PolygonGroup::get_triangles のラッパー関数

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ integer∗8
out	*num_tri*	三角形ポリゴン数
out	*tags_tri*	三角形ポリゴンのタグ（ハンドル) integer∗8 tags_tri は polylib_group_get-_triangles_num() で 求めた個数分 allocate されていること
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.19 void fpolylib_group_set_atr_ (PL_GRP_TAG ∗ *tag,* char ∗ *key,* char ∗ *str*)

ユーザ定義属性の設定

7.9.2.20 void fpolylib_group_set_movable_ (PL_GRP_TAG ∗ *tag,* int ∗ *movable*)

移動対象フラグ設定

7.9.2.21 void fpolylib_group_set_name_ (PL_GRP_TAG ∗ *tag,* char ∗ *name*)

7.9.2.22 void fpolylib_group_set_need_rebuild_ (PL_GRP_TAG ∗ *tag_pg*)

KD 木の再構築フラグの設定 ユーザ定義の移動関数内の最後で呼び出す

Parameters

in	*tag_pg*	PolygonGroup を操作するためのタグ

Returns

戻り値なし

7.9.2.23 void fpolylib_group_set_num_polygon_atr_ (PL_GRP_TAG ∗ *tag,* int ∗ *num_atrI,* int ∗ *num_atrR*)

ポリゴングループ内のポリゴンのユーザ定義属性数の設定

Parameters

in	*tag*	PolygonGroup を操作するためのタグ integer∗8 integer∗8
in	*num_atrI*	整数属性数
in	*num_atrR*	実数属性数

Returns

戻り値なし

7.9.2.24 void fpolylib_init_parallel_info2_ (int ∗ *num,* FParallelBboxStruct ∗ *bbox,* POLYLIB_STAT ∗ *ret*)

並列計算関連情報の設定と初期化を行う。 (各ランクが複数領域を担当している場合） 全 rank で各々設定を行い、その領域情報を全 rank へ配信する。

Parameters

in	*bpos*	自PE 担当領域の基点座標
in	*num*	自PE が担当する領域
in	*bbox*	担当する boundary box 情報（複数） Fortran の構造体で受けること
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

Attention

C++/C と違いコミュニケーターの指定は出来ません。 固定でMPI_COMM_WORLD となります。 その他のコミュニケータを使う場合は、C++/C のメソッドを使用してください。 構造体に対応していない Fortran では使用出来ません。C++/C のメソッドを使用してください。

7.9.2.25 void fpolylib_init_parallel_info_ (PL_REAL *bpos[3]*, int *bbsize[3]*, int *gcsize[3]*, PL_REAL *dx[3]*, POLYLIB_STAT * *ret*)

並列計算関連情報の設定と初期化を行う。 Polylib::init_parallel_info メソッドのFortran ラッパー関数。 (各ランクが 1 領域を担当している場合） 全 rank で各々設定を行い、その領域情報を全 rank へ配信する。

Parameters

in	*bpos*	自PE 担当領域の基点座標
in	*bbsize*	同、計算領域のボクセル数
in	*gcsize*	同、ガイドセルのボクセル数
in	*dx*	同、ボクセル 1 辺の長さ
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

Attention

C++/C と違いコミュニケーターの指定は出来ません。 固定でMPI_COMM_WORLD となります。 その他のコミュニケータを使う場合は、C++/C のメソッドを使用してください。

7.9.2.26 void fpolylib_instance_ (POLYLIB_STAT * *ret*)

Fortran 用Polylib

注意：Fortran 用インターフェースのためのC 言語インターフェースを提供する Fortran 言語API ではタグを操作（ハンドル）用として使用します。 タグには有効期間があります。セッション中で永続的に有効というわけではありません。 PolygonGroup タグ PolygonGroup の追加、削除、挿入があると該当する箇所以外も含めて全て無効となります。 Triangle タグ Triangle の追加、削除、挿入、ソートがあると該当する箇所以外も含めて全て無効となります。 並列環境でTriangle を移動した場合、migrate 処理を実行しますが この際に削除、挿入、ソートが行われるのでタグが全て無効となります。

Fortran 用インターフェース基本方針 サブルーチン名・関数名：C 言語API の名前に先頭に'f' 末尾に'_' を付加する 引数は全て、アドレス渡しとする 戻り値は基本的に引数に変更。 unsigned int は すべて int に変更 C 言語の char* 型 (NULL termination) はFortran ではNULL 文字以降を すべて space とする 当ラッパー関数内ではC の構造体を使用しているが Fortran 側では、Fortran の構造体のインクルードファイルを使用すること

Fortran 用インターフェース制限事項 (1) Fortran からPolylib 環境を構築する場合は MPI のコミュニケータは MPI_COMM_WORLD となります。 他のコミュニケータが必要な場合は、Polylib 環境をC++/C で構築する 必要があります。 (2) Fortran からは移動関数の登録は出来ません。 C++/C より移動関数を登録してください。 F 言語用Polylib 環境の構築 Polylib インスタンス生成 Polylib::get_instance メソッドの代替関数。

Parameters

out	*ret*	POLYLIB_STAT で定義される値が返る。

Attention

最初に呼び出すこと

7.9.2.27 void fpolylib_load_ (char * *config_name*, PL_REAL * *scale*, POLYLIB_STAT * *ret*)

Polylib::load メソッドのラッパー関数。 引数で指定された設定ファイルを読み込み、グループツリーを作成する。 続いて設定ファイルで指定されたSTL ファイルを読み込み、KD 木を作成する。

Parameters

in	*fname*	設定ファイル名 Fortran 型の文字列 (\0 で終了しない） character(len=PL-_FILE_PATH_LEN), character∗256 Fortran 側から呼ぶときは、文字列の最初の空白の 1 個前までを 有効なファイル名と見なします。 すべて空白の時は"polylib_config.tp"とみなす
in	*scale*	縮尺率 PL_REAL が倍精度の時に 単精度定数 1.0 等を指定するのはNG 1.0d0 等で倍精度定数を指定すること
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.28 void fpolylib_migrate_ (POLYLIB_STAT ∗ *ret*)

Polylib::mograte メソッドのラッパー関数 オブジェクトのインスタンス毎に登録が必要

Parameters

out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.29 void fpolylib_move_ (PolylibMoveParamsStruct ∗ *param,* POLYLIB_STAT ∗ *ret*)

三角形ポリゴン座標の移動。 Polylib::move メソッドのラッパー関数 本クラスインスタンス配下の全PolygonGroup の move メソッドが呼び出される。 move 関数は、Fortran 関数では登録できない。C またｈC++から行う。 C 言語：polygongroup_set_move_func_c 関数 C++ ：PolygonGroup::pg->set_move_func 関数

Parameters

in	*param*	移動計算パラメータセット
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.30 void fpolylib_save_ (char ∗ *o_fname,* char ∗ *format,* char ∗ *extend,* POLYLIB_STAT ∗ *ret*)

Polylib::save メソッドのラッパー関数。 PolygoGroup ツリー、三角形ポリゴン情報の保存。 グループツリーの情報を設定ファイルへ出力。三角形ポリゴン情報をSTL ファイル or NPT ファイルへ出力

Parameters

out	*o_fname*	設定ファイル名 Fortran 型の文字列 (長さPL_FILE_PATH_LEN) (\0 で終了しない） character(len=PL_FILE_PATH_LEN), character∗256
in	*format*	形状ファイルのフォーマット Fortran 型の文字列 (長さPL_FORMAT_LEN) (\0 で終了しない） character(len=PL_FORMAT_LEN), character∗8 FPolylib-_define.inc で定義されているFILE_FMT_∗ 参照 'stl_a ' 'stl_aa ' 'stl_b ' 'stl_bb ' 'npt_a ' 'npt_b '
in	*extend*	ファイル名に付加する文字列 Fortran 型の文字列 (長さPL_STR_LEN) character(len=PL_STR_LEN), character∗32 年月日時分秒 (YYYYMMDD24hhmmss) を用いる。
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

Attention

ファイル名命名規約は次の通り。 設定ファイル : polylib_config_付加文字.tpp STL or NPT ファイル：ポリゴングループ名_付加文字.拡張子

7.9.2.31 void fpolylib_search_nearest_polygon_ (PL_ELM_TAG ∗ *tag,* char ∗ *group_name,* PL_REAL *pos[3],* POLYLIB_STAT ∗ *ret*)

指定した点に最も近いポリゴンの検索

Parameters

out	*tag*	ポリゴンのタグ（ハンドル) integer∗8 ポリゴンがない場合、 PL_NULL_TAGi(=0) が返される
in	*group_name*	グループ名 Fortran 型の文字列 (長さPL_GRP_PATH_LEN) character(len=PL_GRP_PATH_LEN), character∗256
in	*pos*	指定点
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

Attention

MPI 並列計算時は,pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

7.9.2.32 void fpolylib_search_polygons_ (int ∗ *num*, PL_ELM_TAG ∗ *tags*, char ∗ *group_name*, PL_REAL *min_pos[3]*, PL_REAL *max_pos[3]*, int ∗ *every*, POLYLIB_STAT ∗ *ret*)

Polylib::search_polygons メソッドのラッパー関数。 位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 特定のグループとその子孫グループに属する三角形ポリゴンをKD 探索に より抽出する。

Parameters

out	*num*	抽出された三角形ポリゴン数
out	*tags*	ポリゴンのタグ（ハンドル) integer∗8 polylib_search_polygons_num_() で得られる個数以上を allocate 済みであること
in	*group_name*	抽出グループ名 Fortran 型の文字列 (長さPL_GRP_PATH_LEN) character(len=PL_GRP_PATH_LEN)
in	*min_pos*	抽出する矩形領域の最小値。(x,y,z 順の配列)
in	*max_pos*	抽出する矩形領域の最大値。(x,y,z 順の配列)
in	*every*	抽出オプション。 1 : 3 頂点が全て検索領域に含まれるポリゴンを抽出する。 0 : 三角形のBBox が一部でも検索領域と交差するものを抽出する。
in	*max*	三角形ポリゴンMAX 数（領域確保数）
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

Attention

MPI 並列計算時は,min_pos, max_pos は各ランクの矩形領域を 超えないようにして下さい (各ランク内の担当領域内のみ検索するため）

7.9.2.33 void fpolylib_search_polygons_num_ (int ∗ *num*, char ∗ *group_name*, PL_REAL *min_pos[3]*, PL_REAL *max_pos[3]*, int ∗ *every*, POLYLIB_STAT ∗ *ret*)

位置ベクトル min_pos と max_pos により特定される矩形領域に含まれる、 ポリゴン数取得

Parameters

out	*num*	抽出されたポリゴン数
in	*group_name*	抽出グループ名 Fortran 型の文字列 (長さPL_GRP_PATH_LEN) character(len=PL_GRP_PATH_LEN), character∗256
in	*min_pos*	抽出する矩形領域の最小値。(x,y,z 順の配列)
in	*max_pos*	抽出する矩形領域の最大値。(x,y,z 順の配列)
in	*every*	抽出オプション。 1 : 3 頂点が全て検索領域に含まれるポリゴンを抽出する。 0 : 三角形のBBox が一部でも検索領域と交差するものを抽出する。

out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.34 void fpolylib_show_group_hierarchy_ ( )

Polylib::show_group_hierarchy メソッドのラッパー関数。 グループ階層構造リストを標準出力に出力する。

7.9.2.35 void fpolylib_show_group_info_ (char * *group_name*, POLYLIB_STAT * *ret*)

Polylib::show_group_info メソッドのラッパー関数。 グループの情報を出力する。(親グループ名、自信の名前、ファイル名、 登録三角形数、3 頂点ベクトルの座標、法線ベクトルの座標、面積)

Parameters

in	*group_name*	グループ名 Fortran 型の文字列 (長さPL_GRP_PATH_LEN) character(len=PL_GRP_PATH_LEN)
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る

7.9.2.36 int fpolylib_triangle_get_atrI_ (PL_ELM_TAG * *tag*, int * *atr_no*)

ポリゴン（形状）のユーザ定義属性（整数型）の取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer*8
in	*atr_no*	整数属性の何番目か　開始:1

Returns

ユーザ定義属性（整数型）値

7.9.2.37 PL_REAL fpolylib_triangle_get_atrR_ (PL_ELM_TAG * *tag*, int * *atr_no*)

ポリゴン（形状）のユーザ定義属性（実数型）の取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer*8
in	*atr_no*	実数属性の何番目か　開始:1

Returns

ユーザ定義属性（実数型）値

7.9.2.38 void fpolylib_triangle_get_normal_ (PL_ELM_TAG * *tag*, PL_REAL *norm[3]*)

法線ベクトル取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer*8

out	*norm*	法線ベクトル

Returns

戻り値なし

7.9.2.39 void fpolylib_triangle_get_npatchParam2_ (PL_ELM_TAG * *tag*, PL_REAL *cp_side1_1[3]*, PL_REAL *cp_side1_2[3]*, PL_REAL *cp_side2_1[3]*, PL_REAL *cp_side2_2[3]*, PL_REAL *cp_side3_1[3]*, PL_REAL *cp_side3_2[3]*, PL_REAL *cp_center[3]*, POLYLIB_STAT * *ret*)

長田パッチパラメータ取得 引数に構造体を使用しない版

Parameters

in	*tag*	NptTriangle を操作するためのタグ integer*8
out	*cp_side1_1*	長田パッチパラメータ p1p2 辺の 3 次ベジェ制御点 1
out	*cp_side1_2*	長田パッチパラメータ p1p2 辺の 3 次ベジェ制御点 2
out	*cp_side2_1*	長田パッチパラメータ p2p3 辺の 3 次ベジェ制御点 1
out	*cp_side2_2*	長田パッチパラメータ p2p3 辺の 3 次ベジェ制御点 2
out	*cp_side3_1*	長田パッチパラメータ p3p1 辺の 3 次ベジェ制御点 1
out	*cp_side3_2*	長田パッチパラメータ p3p1 辺の 3 次ベジェ制御点 2
out	*cp_center*	長田パッチパラメータ 三角形中央の 3 次ベジェ制御点
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る オブジェクトが長田パッチでない時はエラーを返す

7.9.2.40 void fpolylib_triangle_get_npatchParam_ (PL_ELM_TAG * *tag*, NpatchParamStruct * *param*, POLYLIB_STAT * *ret*)

長田パッチパラメータ取得

Parameters

in	*tag*	NptTriangle を操作するためのタグ integer*8
out	*param*	長田パッチパラメータ
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る オブジェクトが長田パッチでない時はエラーを返す

7.9.2.41 int fpolylib_triangle_get_num_atrI_ (PL_ELM_TAG * *tag*)

ユーザ定義属性数性（整数型）の取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer*8

Returns

整数属性数

7.9.2.42 int fpolylib_triangle_get_num_atrR_ (PL_ELM_TAG * *tag*)

ユーザ定義属性数性（実数型）の取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8

Returns

実数属性数

7.9.2.43 int fpolylib_triangle_get_pl_type_ (PL_ELM_TAG ∗ *tag*)

ポリゴンタイプ取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8

Returns

PL_TYPE_TRIANGLE / PL_TYPE_NPT（長田パッチ）

7.9.2.44 void fpolylib_triangle_get_vertexes_ (PL_ELM_TAG ∗ *tag,* PL_REAL *vertex[9]*)

頂点座標取得

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8
out	*vertex*	３頂点の座標

Returns

戻り値なし

7.9.2.45 void fpolylib_triangle_set_atrI_ (PL_ELM_TAG ∗ *tag,* int ∗ *atr_no,* int ∗ *val*)

ポリゴン（形状）のユーザ定義属性（整数型）の設定

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8
in	*atr_no*	整数属性の何番目か　開始:1
in	*val*	整数属性値

Returns

戻り値なし

7.9.2.46 void fpolylib_triangle_set_atrR_ (PL_ELM_TAG ∗ *tag,* int ∗ *atr_no,* PL_REAL ∗ *val*)

ポリゴン（形状）のユーザ定義属性（実数型）の設定

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8
in	*atr_no*	実数属性の何番目か　開始:1
in	*val*	実数属性値

Returns

戻り値なし

7.9.2.47 void fpolylib_triangle_set_npatchParam2_ (PL_ELM_TAG ∗ *tag*, PL_REAL *cp_side1_1[3]*, PL_REAL *cp_side1_2[3]*, PL_REAL *cp_side2_1[3]*, PL_REAL *cp_side2_2[3]*, PL_REAL *cp_side3_1[3]*, PL_REAL *cp_side3_2[3]*, PL_REAL *cp_center[3]*, POLYLIB_STAT ∗ *ret*)

長田パッチパラメータ設定 引数に構造体を使用しない版

Parameters

in	*tag*	NptTriangle を操作するためのタグ integer∗8
in	*cp_side1_1*	長田パッチパラメータ p1p2 辺の３次ベジェ制御点 1
in	*cp_side1_2*	長田パッチパラメータ p1p2 辺の３次ベジェ制御点 2
in	*cp_side2_1*	長田パッチパラメータ p2p3 辺の３次ベジェ制御点 1
in	*cp_side2_2*	長田パッチパラメータ p2p3 辺の３次ベジェ制御点 2
in	*cp_side3_1*	長田パッチパラメータ p3p1 辺の３次ベジェ制御点 1
in	*cp_side3_2*	長田パッチパラメータ p3p1 辺の３次ベジェ制御点 2
in	*cp_center*	長田パッチパラメータ 三角形中央の３次ベジェ制御点
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る オブジェクトが長田パッチでない時はエラーを返す

7.9.2.48 void fpolylib_triangle_set_npatchParam_ (PL_ELM_TAG ∗ *tag*, NpatchParamStruct ∗ *param*, POLYLIB_STAT ∗ *ret*)

長田パッチパラメータ設定

Parameters

in	*tag*	NptTriangle を操作するためのタグ integer∗8
in	*param*	長田パッチパラメータ
out	*ret*	POLYLIB_STAT(integer) で定義される値が返る オブジェクトが長田パッチでない時はエラーを返す

7.9.2.49 void fpolylib_triangle_set_vertexes_ (PL_ELM_TAG ∗ *tag*, PL_REAL *vertex[9]*)

頂点座標設定 頂点設定時、法線ベクトル, 面積も内部で設定する 長田パッチのパラメータは更新されないので注意

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8
in	*vertex*	３頂点の座標

7.9.2.50 void fpolylib_triangle_set_vertexes_npatch2_ (PL_ELM_TAG ∗ *tag*, PL_REAL *vertex[9]*, PL_REAL *cp_side1_1[3]*, PL_REAL *cp_side1_2[3]*, PL_REAL *cp_side2_1[3]*, PL_REAL *cp_side2_2[3]*, PL_REAL *cp_side3_1[3]*, PL_REAL *cp_side3_2[3]*, PL_REAL *cp_center[3]*)

頂点座標・長田パッチパラメータ設定 引数に構造体を使用しない版

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8
in	*vertex*	３頂点の座標
in	*cp_side1_1*	長田パッチパラメータ p1p2 辺の３次ベジェ制御点 1
in	*cp_side1_2*	長田パッチパラメータ p1p2 辺の３次ベジェ制御点 2
in	*cp_side2_1*	長田パッチパラメータ p2p3 辺の３次ベジェ制御点 1
in	*cp_side2_2*	長田パッチパラメータ p2p3 辺の３次ベジェ制御点 2
in	*cp_side3_1*	長田パッチパラメータ p3p1 辺の３次ベジェ制御点 1
in	*cp_side3_2*	長田パッチパラメータ p3p1 辺の３次ベジェ制御点 2
in	*cp_center*	長田パッチパラメータ 三角形中央の３次ベジェ制御点

Returns

戻り値なし

Attention

法線ベクトル, 面積も内部で設定する

7.9.2.51 void fpolylib_triangle_set_vertexes_npatch_ (PL_ELM_TAG ∗ *tag*, PL_REAL *vertex[9]*, NpatchParamStruct ∗ *param*)

頂点座標・長田パッチパラメータ設定

Parameters

in	*tag*	Triangle/NptTriangle を操作するためのタグ integer∗8
in	*vertex*	３頂点の座標
in	*param*	長田パッチパラメータ

Returns

戻り値なし

Attention

法線ベクトル, 面積も内部で設定する

7.9.2.52 int fpolylib_used_memory_size_ ( )

Polylib が利用中の概算メモリ量を返す

Returns

利用中のメモリ量 (byte) 2GB 以上返せないので注意 2GB 以上が予想される場合は fpolylib_used_memory_size_mb() を使用すること

7.9.2.53 int fpolylib_used_memory_size_mb_ ( )

Polylib が利用中の概算メモリ量 (MB) を返す

Returns

利用中のメモリ量 (Mbyte)

7.10 f_lang/FPolylib_define.inc File Reference

7.11 f_lang/FPolylib_define_f77.inc File Reference

7.12 f_lang/FPolylib_precision.inc File Reference

7.13 f_lang/FPolylib_precision_def.inc File Reference

7.14 f_lang/FPolylib_prototype.inc File Reference

7.15 f_lang/FPolylib_prototype_f77_double.inc File Reference

7.16 f_lang/FPolylib_prototype_f77_float.inc File Reference

7.17 f_lang/FPolylib_prototype_f77_integer.inc File Reference

7.18 f_lang/FPolylib_struct.inc File Reference

7.19 f_lang/FPolylib_User.inc File Reference

7.20 f_lang/FPolylib_User_f77.inc File Reference

7.21 file_io/FileIO_func.h File Reference

```
#include <vector>
#include "common/PolylibCommon.h"
```

Namespaces

- PolylibNS

Functions

- POLYLIB_STAT PolylibNS::stl_a_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::stl_a_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::stl_a_save (std::vector< Triangle * > *tri_list, const std::string &fname)
- POLYLIB_STAT PolylibNS::stl_b_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::stl_b_load_read_head (ifstream &ifs)
- POLYLIB_STAT PolylibNS::stl_b_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::stl_b_save (std::vector< Triangle * > *tri_list, const std::string &fname)
- bool PolylibNS::is_stl_a (const std::string &path)

- POLYLIB_STAT PolylibNS::npt_a_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::npt_a_load_read_head (ifstream &ifs)
- POLYLIB_STAT PolylibNS::npt_a_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::npt_a_save (std::vector< NptTriangle * > *tri_list, const std::string &fname)
- POLYLIB_STAT PolylibNS::npt_b_load (std::vector< Triangle * > *tri_list, const std::string &fname, int *num_tri, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::npt_b_load_read_head (ifstream &ifs)
- POLYLIB_STAT PolylibNS::npt_b_load_read (ifstream &ifs, std::vector< Triangle * > &tri_list, int num_read, int &num_tri, bool &eof, PL_REAL scale=1.0)
- POLYLIB_STAT PolylibNS::npt_b_save (std::vector< NptTriangle * > *tri_list, const std::string &fname)
- bool PolylibNS::is_npt_a (const std::string &path)
- char * PolylibNS::get_fname_fr_path (const std::string &path)
- char * PolylibNS::get_ext_fr_path (const std::string &path)

7.22 file_io/PolygonIO.h File Reference

```
#include <vector>
#include <map>
#include <fstream>
#include "common/PolylibStat.h"
#include "common/PolylibCommon.h"
#include "polygons/Triangle.h"
#include "polygons/NptTriangle.h"
```

Classes

- class PolylibNS::PolygonIO

Namespaces

- PolylibNS

7.23 groups/PolygonGroup.h File Reference

```
#include "common/PolylibCommon.h"
#include "common/PolylibStat.h"
#include "common/Vec3.h"
#include "groups/VTree.h"
#include "TextParser.h"
#include "c_lang/CPolylib.h"
#include <vector>
#include <map>
```

Classes

- struct PolylibNS::UsrAtr
- class PolylibNS::PolygonGroup

Namespaces

- PolylibNS

7.24 groups/VTree.h File Reference

```
#include "common/BBox.h"
#include "common/PolylibStat.h"
#include "common/PolylibCommon.h"
#include "polygons/Triangle.h"
#include "polygons/NptTriangle.h"
#include <vector>
```

Classes

- class PolylibNS::VElement
- class PolylibNS::VNode
- class PolylibNS::VTree

Namespaces

- PolylibNS

7.25 polygons/NptTriangle.h File Reference

```
#include "common/Vec3.h"
#include "polygons/Triangle.h"
```

Classes

- struct PolylibNS::NpatchParam
- class PolylibNS::NptTriangle

Namespaces

- PolylibNS

7.26 polygons/Triangle.h File Reference

```
#include "common/Vec3.h"
#include "common/BBox.h"
```

Classes

- class PolylibNS::Triangle

Namespaces

- PolylibNS

7.27 Polylib.h File Reference

```
#include <string>
#include <vector>
#include <iostream>
#include "polygons/Triangle.h"
#include "polygons/NptTriangle.h"
#include "groups/PolygonGroup.h"
#include "file_io/PolygonIO.h"
#include "file_io/FileIO_func.h"
#include "common/PolylibStat.h"
#include "common/PolylibCommon.h"
#include "common/BBox.h"
#include "common/Vec3.h"
#include "Polylib_func.h"
#include "TextParser.h"
#include "polyVersion.h"
#include "mpi.h"
```

Classes

- struct PolylibNS::ParallelBbox
- struct PolylibNS::CalcAreaInfo

  計算領域情報。
- struct PolylibNS::ParallelAreaInfo

  並列プロセス領域情報。
- class PolylibNS::PolylibMoveParams
- class PolylibNS::Polylib

Namespaces

- PolylibNS

7.28 Polylib_func.h File Reference

```
#include <string>
#include <vector>
#include <iostream>
#include "polygons/Triangle.h"
#include "polygons/NptTriangle.h"
#include "groups/PolygonGroup.h"
#include "common/PolylibStat.h"
#include "common/PolylibCommon.h"
#include "common/BBox.h"
#include "common/Vec3.h"
```

Namespaces

- PolylibNS

Macros

- #define INLINE inline

Functions

- Triangle * PolylibNS::deserialize_polygon (int pl_type, const char *pbuff)
- void PolylibNS::copy_polygon (Triangle *tri, Triangle *&copy_tri)
- void PolylibNS::copy_polygons (const std::vector< Triangle * > &tri_list, std::vector< Triangle * > &copy_tri_list)
- POLYLIB_STAT PolylibNS::convert_polygons_to_npt (std::vector< Triangle * > &tri_list, std::vector< NptTriangle * > &npt_list)
- void PolylibNS::convert_polygons_to_tri (std::vector< NptTriangle * > &npt_list, std::vector< Triangle * > &tri_list)
- int ** PolylibNS::alloc_array_2d_int (int n1, int n2)
- PL_REAL ** PolylibNS::alloc_array_2d_real (int n1, int n2)
- void PolylibNS::free_array_2d (void **x)

7.28.1 Macro Definition Documentation

7.28.1.1 #define INLINE inline

7.29 polyVersion.h File Reference

Macros

- #define PL_VERSION_NO "4.0.0.0"
- #define PL_REVISION "20169999_9999"

7.29.1 Detailed Description

Polylib バージョン情報のヘッダーファイル

7.29.2 Macro Definition Documentation

7.29.2.1 #define PL_REVISION "20169999_9999"

POLYLIB ライブラリのリビジョン

7.29.2.2 #define PL_VERSION_NO "4.0.0.0"

POLYLIB ライブラリのバージョン

7.30 util/time.h File Reference

Namespaces

- PolylibNS

Functions

- bool PolylibNS::getrusage_sec (double *usr_time, double *sys_time, double *total)