UF_MOTION_animation_run (view source)

Defined in: uf_motion_ugopenint.h

Overview
This function runs animation in interactive mode using the existing user interface.

Environment
Internal and External

See Also
UF_MOTION_articulation_run

History
Released in NX801

```
int UF_MOTION_animation_run
(
    void
)
```

UF_MOTION_articulation_run (view source)

Defined in: uf_motion_ugopenint.h

Overview
This function runs the articulation solve in interactive mode using the existing user interface.

Environment
Internal and External

See Also
UF_MOTION_animation_run

History
Released in NX801

```
int UF_MOTION_articulation_run
(
    void
)
```

UF_MOTION_ask_2D_contact (view source)

Defined in: uf_motion.h

Overview
This function returns the 2D contact structure for an input 2D contact tag.

Environment

Internal and External

See Also
UF_MOTION_set_2D_contact

History
Created in NX2

int UF_MOTION_ask_2D_contact
(
const tag_t contact_tag,
UF_MOTION_2D_contact_t * contact_struct
)

const tag_t	**contact_tag**	Input	The tag of the 2D contact to return. This tag is the one returned from UF_MOTION_create_2D_contact. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_2D_contact_t *	**contact_struct**	Output	The 2D contact structure of the input tag.

UF_MOTION_ask_3D_contact (view source)

Defined in: uf_motion.h

Overview
This function returns the 3D contact structure for an input 3D contact tag.

Environment
Internal and External

See Also
UF_MOTION_set_3D_contact

History
Created in NX2

int UF_MOTION_ask_3D_contact
(
const tag_t contact_tag,
UF_MOTION_3D_contact_t * contact_struct
)

const tag_t	**contact_tag**	Input	The tag of the 3D contact to return. This tag is the one returned from UF_MOTION_create_3D_contact. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_3D_contact_t *	**contact_struct**	Output	The 3D contact structure of the input tag.

UF_MOTION_ask_3D_contact_method (view source)

Defined in: uf_motion.h

Overview
This function asks the 3D contact method to be used.

Return
Zero for success, non-zero is error code

Environment
Internal and External

See Also
UF_MOTION_set_3D_contact_method

History
Released in NX3

```
int UF_MOTION_ask_3D_contact_method
(
    UF_MOTION_3d_contact_method_t * contact_method,
    int * facet_contact_tolerance
)
```

UF_MOTION_3d_contact_method_t *	**contact_method**	Output	the 3d contact method to be used by Adams
int *	**facet_contact_tolerance**	Output	ignored unless contact_method == UF_MOTION_faceted_contact Must be a value 1 to 100. 1 means Fast but may not be accurate 100 means Accurate but may not be Fast

UF_MOTION_ask_acceleration_results (view source)

Defined in: uf_motion.h

Overview
This function is used to get specific acceleration results from the analysis results stored in the database. It can return both translational and rotational accelerations. If this function is called when no results are in the database, an error condition will be returned.

Environment
Internal and External

See Also

UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file
UF_MOTION_ask_trans_displacement_results
UF_MOTION_ask_velocity_results

History

Released in V18

```
int UF_MOTION_ask_acceleration_results
(
    const tag_t motion_object,
    const UF_MOTION_motion_type_t type,
    const UF_MOTION_vector_component_t component,
    const UF_MOTION_reference_frame_t ref_frame,
    int * number_of_results,
    double * * results
)
```

const tag_t	**motion_object**	Input	The tag of the object to get the results for. This is usually a joint, constraint, coupler, marker, or primitive.
const UF_MOTION_motion_type_t	**type**	Input	The type of acceleration: UF_MOTION_rotation or UF_MOTION_translation.
const UF_MOTION_vector_component_t	**component**	Input	The desired component of the acceleration (Example: UF_MOTION_magnitude or UF_MOTION_x_component).
const UF_MOTION_reference_frame_t	**ref_frame**	Input	The reference frame that the results will be resolved in.
int *	**number_of_results**	Output	The number of results returned in the next argument. This will equal the number of steps in the solution.
double * *	**results**	Output to UF_*free*	The results requested. This array must be freed by the calling function using UF_free().

UF_MOTION_ask_active_solution (view source)

Defined in: uf_motion.h

Overview

Ask the active solution for current motion simulation

Environment

Internal or External

Onput : tag_t active_solution

Returns

0 - Successful
else Unsuccessful

History

Released in NX801

int UF_MOTION_ask_active_solution
(
tag_t * active_solution
)

tag_t *	**active_solution**	Output	The active solution

UF_MOTION_ask_angular_units (view source)

Defined in: uf_motion.h

Overview

This function return the angular units stored in this mechanism

Environment

Internal and External

See Also

UF_MOTION_set_angular_units

History

Released in NX3

int UF_MOTION_ask_angular_units
(
UF_MOTION_angular_units_type_t * angle_units
)

UF_MOTION_angular_units_type_t *	**angle_units**	Output	The new units

---

UF_MOTION_ask_artic_step_size (view source)

Defined in: uf_motion.h

Overview

All joints that have a joint motion input defined can be articulated even though the motion input type might not be articulation. This function returns the articulation step size for the input joint tag. The tag must belong to a joint that has a motion input defined or else an error condition is returned.

Environment

Internal and External

See Also

UF_MOTION_edit_artic_step_size
UF_MOTION_step_articulation

History

Released in V18

```
int UF_MOTION_ask_artic_step_size
(
    const tag_t joint_tag,
    double * step_size
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to return the step size for. This tag is returned from the function UF_MOTION_create_joint. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
double *	**step_size**	Output	The size of the articulation joint step. The units of the step are based on the units of the joint.

UF_MOTION_ask_articulation_stop_tolerance (view source)

Defined in: uf_motion.h

Overview

This function asks the mechanism parameters of the current mechanism in the scenario part.

Environment

Internal and External

See Also

UF_MOTION_set_articulation_stop_tolerance

```
int UF_MOTION_ask_articulation_stop_tolerance
(
    double * stop_tolerance
)
```

double *	**stop_tolerance**	Output

UF_MOTION_ask_attachments_of_type (view source)

Defined in: uf_motion.h

Overview

This function returns the motion objects of a specified type that are attached to the input motion object. This function will check the type/subtype requests for validity of attachment. For example, asking for contacts attached to a joint is not valid. Note also, that the subtype input is only needed for certain motion types, and is not required for other motion types. See following table:

type requested subtype needed
==================== ==============
UF_mdm_mechanism_type NO
UF_mdm_link_type NO
UF_mdm_joint_type OPTIONAL - If subtype not defined, all entities of this type will be returned.
UF_mdm_marker_type NO - UF_mdm_user_defined_marker_subtype will be only subtype returned.
UF_mdm_spring_type NO
UF_mdm_damper_type NO
UF_mdm_force_type NO
UF_mdm_torque_type NO
UF_mdm_genforce_type YES - Subtype must be defined.
UF_mdm_curve_curve_contact_type NO
UF_mdm_contact_type NO

Environment

Internal and External

History

Released in NX3

```
int UF_MOTION_ask_attachments_of_type
(
    tag_t entity_tag,
    int type,
    int subtype,
    int * num_attachments,
    tag_p_t * attachments
)
```

tag_t	**entity_tag**	Input	The motion entity to get attachments for.
int	**type**	Input	The type of attachments to retrieve. See uf_object_types.h for definition of motion object types.
int	**subtype**	Input	The subtype of attachments (if applicable) to retrieve. See uf_object_types.h for definition of motion object subtypes.
int *	**num_attachments**	Output	The number of attachment objects.
tag_p_t *	**attachments**	Output to UF_*free*	The array of attached objects.

UF_MOTION_ask_crv_crv_constraint (view source)

Defined in: uf_motion.h

Overview

This function returns the curve curve structure for an input curve curve constraint tag.

Environment

Internal and External

See Also

UF_MOTION_create_crv_crv_constraint

History

Created in NX2

```
int UF_MOTION_ask_crv_crv_constraint
(
    const tag_t crv_crv_tag,
    UF_MOTION_curve_curve_constraint_t * crv_crv_struct
)
```

const tag_t	**crv_crv_tag**	Input	The tag of the constraint to return. This tag is the one returned from the UF_MOTION_create_crv_crv_constraint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_curve_curve_constraint_t *	**crv_crv_struct**	Output	The curve-curve structure for the input tag.

UF_MOTION_ask_cylindrical_bushing (view source)

Defined in: uf_motion.h

Overview

This function returns the cylindrical bushing structure for an input bushing tag.

Environment

Internal and External

See Also

UF_MOTION_set_cylindrical_bushing

History

Created in NX2

```
int UF_MOTION_ask_cylindrical_bushing
(
    const tag_t bushing_tag,
    UF_MOTION_cylindrical_bushing_t * bushing_struct
)
```

const tag_t	**bushing_tag**	Input	The tag of the cylindrical bushing to return information about. This tag is the one returned from the UF_MOTION_create_cylindrical_bushing function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_cylindrical_bushing_t *	**bushing_struct**	Output	The cylindrical bushing structure for the input tag.

UF_MOTION_ask_damper (view source)

Defined in: uf_motion.h

Overview
This function returns the damper structure for an input damper tag.

Environment
Internal and External

See Also
UF_MOTION_set_damper

History
Created in NX2

```
int UF_MOTION_ask_damper
(
    const tag_t damper_tag,
    UF_MOTION_spring_damper_t * damper_struct
)
```

const tag_t	**damper_tag**	Input	The tag of the damper to return information about. This tag is the one returned from the UF_MOTION_create_damper function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_spring_damper_t *	**damper_struct**	Output	The damper structure for the input tag.

UF_MOTION_ask_force_results (view source)

Defined in: uf_motion.h

Overview
This function is used to get specific force results from the analysis results stored in the database. If this function is called when no results are in the database, an error condition will be returned.

Environment
Internal and External

See Also
UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file
UF_MOTION_ask_velocity_results
UF_MOTION_ask_torque_results

History
Released in V18

```
int UF_MOTION_ask_force_results
(
    const tag_t motion_object,
    const UF_MOTION_vector_component_t component,
    const UF_MOTION_reference_frame_t ref_frame,
    int * number_of_results,
    double * * results
)
```

const tag_t	**motion_object**	Input	The tag of the object to get the results for. This is usually a joint, constraint, coupler, marker, or primitive.
const UF_MOTION_vector_component_t	**component**	Input	The desired component of the force (Example: UF_MOTION_magnitude or UF_MOTION_x_component).
const UF_MOTION_reference_frame_t	**ref_frame**	Input	The reference frame that the results will be resolved in.
int *	**number_of_results**	Output	The number of results returned in the next argument. This will equal the number of steps in the solution.
double * *	**results**	Output to UF_*free*	The results requested. This array must be freed by the calling function using UF_free().

UF_MOTION_ask_function (view source)

Defined in: uf_motion.h

Overview
This function returns the function structure for an input function tag.

Environment
Internal and External

See Also

UF_MOTION_set_function
UF_MOTION_find_all_functions
UF_MOTION_ask_function_tag_from_name

History

Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_ask_function
(
    const tag_t function_tag,
    UF_MOTION_function_t * function_struct
)
```

const tag_t	**function_tag**	Input	The tag of the function to query. This tag is the one returned from UF_MOTION_create_function.
UF_MOTION_function_t *	**function_struct**	Output	The function structure for the input function tag.

UF_MOTION_ask_function_tag_from_name (view source)

Defined in: uf_motion.h

Overview

This function finds the motion function tag for a given function name.

Environment

Internal and External

See Also

UF_MOTION_find_all_functions

History

Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_ask_function_tag_from_name
(
    char * function_name,
    tag_t * function_tag
)
```

char *	**function_name**	Input	Name string of function object to find.
tag_t *	**function_tag**	Output	Tag of function object for input name.

UF_MOTION_ask_general_bushing (view source)

Defined in: uf_motion.h

Overview
This function returns the general bushing structure for an input bushing tag.

Environment
Internal and External

See Also
UF_MOTION_set_general_bushing

History
Created in NX2

```
int UF_MOTION_ask_general_bushing
(
    const tag_t bushing_tag,
    UF_MOTION_general_bushing_t * bushing_struct
)
```

const tag_t	**bushing_tag**	Input	The tag of the general bushing to return information about. This tag is the one returned from the UF_MOTION_create_general_bushing function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_general_bushing_t *	**bushing_struct**	Output	The general bushing structure for the input tag.

UF_MOTION_ask_gravitational_constants (view source)

Defined in: uf_motion.h

Overview
This function return the graviational constants stored in this mechanism

Environment
Internal and External

See Also
sUF_MOTION_set_gravitational_constants

History
Released in NX3

```
int UF_MOTION_ask_gravitational_constants
(
    double gravitational_constants [ 3 ]
)
```

double	**gravitational_constants [3]**	Output	Vector defining gravitational force

---

UF_MOTION_ask_gruebler_count (view source)

Defined in: uf_motion.h

Overview
This function gets the total Gruebler count for the current mechanism.

Environment
Internal and External

See Also
UF_MOTION_ask_solver_dof_count

History
Created in NX2

```
int UF_MOTION_ask_gruebler_count
(
    int * gruebler_count
)
```

int *	**gruebler_count**	Output	The total Gruebler count.

UF_MOTION_ask_icon_scale_factor (view source)

Defined in: uf_motion.h

Overview
This function returns the global icon scale.

Environment
Internal and External

See Also
UF_MOTION_set_icon_scale_factor

History
Released in NX3

```
int UF_MOTION_ask_icon_scale_factor
(
    double * scale
)
```

double *	**scale**	Input

UF_MOTION_ask_interference (view source)

Defined in: uf_motion.h

Overview
This function returns the interference structure for an input interference tag.

Environment
Internal and External

See Also
UF_MOTION_edit_interference
UF_MOTION_create_interference
UF_MOTION_interference

History
Released in V18

```
int UF_MOTION_ask_interference
(
    const tag_t interfere_tag,
    UF_MOTION_interference_t * interference_struct
)
```

const tag_t	**interfere_tag**	Input	The tag of the interference to return. This tag is the one returned from the UF_MOTION_create_interference function. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
UF_MOTION_interference_t *	**interference_struct**	Output	The interference structure for the input tag.

UF_MOTION_ask_joint (view source)

Defined in: uf_motion.h

Overview
This function returns the joint structure for an input joint tag.

Environment
Internal and External

See Also
UF_MOTION_set_joint
UF_MOTION_ask_joint_limits
UF_MOTION_ask_joint_motion_input

History
Created in NX2

```
int UF_MOTION_ask_joint
(
    const tag_t joint_tag,
    UF_MOTION_joint_t * joint_struct
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to return information about. This tag is the one returned from the UF_MOTION_create_joint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_joint_t *	**joint_struct**	Output	The joint structure for the input tag.

UF_MOTION_ask_joint_coupler (view source)

Defined in: uf_motion.h

Overview
This function returns the joint coupler structure for an input joint coupler tag.

Environment
Internal and External

See Also
UF_MOTION_set_joint_coupler
UF_MOTION_ask_joint

History
Created in NX2

```
int UF_MOTION_ask_joint_coupler
(
    const tag_t joint_coupler_tag,
    UF_MOTION_joint_coupler_t * coupler_struct
)
```

const tag_t	**joint_coupler_tag**	Input	The tag of the joint coupler to return. This tag is the one returned from the UF_MOTION_create_joint_coupler function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_joint_coupler_t *	**coupler_struct**	Output	The joint coupler structure for the input joint coupler tag.

UF_MOTION_ask_joint_limits (view source)

Defined in: uf_motion.h

Overview

This function returns the joint limits structure for an input joint tag. If joint limits are not defined for the input joint tag, an error condition is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_joint
UF_MOTION_set_joint_limits
UF_MOTION_ask_joint_motion_input

History

Created in NX2

```
int UF_MOTION_ask_joint_limits
(
    const tag_t joint_tag,
    UF_MOTION_joint_limits_t * joint_limits_struct
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to return the joint limits of. This tag is the one returned from the UF_MOTION_create_joint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_joint_limits_t *	**joint_limits_struct**	Output	The joint limits structure for the input tag. If joint limits are not defined for the input joint, zeros will be returned in the structure and an error will be returned from the function.

UF_MOTION_ask_joint_motion_input (view source)

Defined in: uf_motion.h

Overview

This function returns the joint motion input structure for the input joint tag. If motion input has not been defined for the joint, then an error condition is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_joint
UF_MOTION_ask_joint_limits

UF_MOTION_set_joint_motion_input

History
Created in NX2

int UF_MOTION_ask_joint_motion_input
(
    const tag_t joint_tag,
    UF_MOTION_joint_motion_input_t * motion_input_struct
)

const tag_t	**joint_tag**	Input	The tag of the joint to return the motion input info of. This tag is the one returned from the UF_MOTION_create_joint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_joint_motion_input_t *	**motion_input_struct**	Output	The joint motion input structure for the input joint tag.

UF_MOTION_ask_link (view source)

Defined in: uf_motion.h

Overview
This function returns the link structure for an input link tag.

Environment
Internal and External

See Also
UF_MOTION_set_link
UF_MOTION_ask_link_mass_properties
UF_MOTION_ask_link_initial_velocity

History
Created in NX2

int UF_MOTION_ask_link
(
    const tag_t link_tag,
    UF_MOTION_link_t * link_struct
)

const tag_t	**link_tag**	Input	The tag of the link to return. This tag is the one returned from the UF_MOTION_create_link function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)

UF_MOTION_link_t *	**link_struct**	Output	The link structure for the input tag.

UF_MOTION_ask_link_initial_velocity (view source)

Defined in: uf_motion.h

Overview
This function returns the link initial velocity for an input link tag.
All velocities are based on the work coordinate system of the model.

Environment
Internal and External

See Also
UF_MOTION_ask_link
UF_MOTION_ask_link_mass_properties
UF_MOTION_set_link_initial_velocity

History
Created in NX2

```
int UF_MOTION_ask_link_initial_velocity
(
    const tag_t link_tag,
    UF_MOTION_link_initial_velocity_t * init_vel_struct
)
```

const tag_t	**link_tag**	Input	The tag of the link to return the initial velocity of. This tag is the one returned from the UF_MOTION_create_link function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_link_initial_velocity_t *	**init_vel_struct**	Output	Data structure containing the initial velocities.

UF_MOTION_ask_link_mass_properties (view source)

Defined in: uf_motion.h

Overview
This function returns the link mass properties structure for an input link tag. Default mass properties will be calculated for solid bodies if UF_MOTION_set_link_mass_properties has been called previously. If the input link tag has no solid bodies and no mass properties have been set on the link, an error condition will be returned during solve.

Environment

Internal and External

See Also

UF_MOTION_ask_link
UF_MOTION_set_link_mass_properties
UF_MOTION_remove_link_mass_properties

History

Created in NX2

```
int UF_MOTION_ask_link_mass_properties
(
    const tag_t link_tag,
    UF_MOTION_link_mass_properties_t * mass_prop_struct
)
```

const tag_t	**link_tag**	Input	The tag of the link to return mass properties. This tag is the one returned from the UF_MOTION_create_link function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_link_mass_properties_t *	**mass_prop_struct**	Output	The link mass property structure for the input tag. If the mass properties have been set by the user, or default properties were calculated, they will be returned here.

UF_MOTION_ask_link_transfrom_for_given_frame (view source)

Defined in: uf_motion.h

Overview

This function gets the transform matrix of the specified link object in given frame

Environment

Internal and External

History

Released in NX801

```
int UF_MOTION_ask_link_transfrom_for_given_frame
(
    tag_t linkTag,
    int frame,
    double transformMatrix [ 16 ]
)
```

tag_t	**linkTag**	Input	Motion link object tag
int	**frame**	Input	The given frame
double	**transformMatrix [16]**	Output	The transform matrix 1X16

UF_MOTION_ask_marker (view source)

Defined in: uf_motion.h

Overview
This function returns the marker structure for an input marker tag.

Environment
Internal and External

See Also
UF_MOTION_set_marker
UF_MOTION_create_marker

History
Created in NX2

```
int UF_MOTION_ask_marker
(
    const tag_t marker_tag,
    UF_MOTION_marker_t * marker_struct
)
```

const tag_t	**marker_tag**	Input	The tag of the marker to return. This tag is the one returned from UF_MOTION_create_marker. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_marker_t *	**marker_struct**	Output	The marker structure of the input tag.

UF_MOTION_ask_measurement (view source)

Defined in: uf_motion.h

Overview
This function returns the measurement structure for an input measurement tag.

Environment
Internal and External

See Also
UF_MOTION_edit_measurement
UF_MOTION_create_measurement

UF_MOTION_measure

History

Released in V18

```
int UF_MOTION_ask_measurement
(
    const tag_t meas_tag,
    UF_MOTION_measurement_t * meas_struct
)
```

const tag_t	**meas_tag**	Input	The tag of the measurement to return. This tag is the one returned from the creation function (UF_MOTION_create_measurement). It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
UF_MOTION_measurement_t *	**meas_struct**	Output	The measurement structure for the input tag.

UF_MOTION_ask_name_display (view source)

Defined in: uf_motion.h

Overview

This function returns the dislay of object names

Environment

Internal and External

See Also

UF_MOTION_set_name_display

History

Released in NX3

```
int UF_MOTION_ask_name_display
(
    logical * name_display
)
```

logical *	**name_display**	Output	TRUE if names should be displayed. FALSE if names should not be displayed.

UF_MOTION_ask_number_of_animation_frames (view source)

Defined in: uf_motion.h

Overview

This function gets the number of frames for active solution.

Environment
Internal and External

History
Released in NX801

```
int UF_MOTION_ask_number_of_animation_frames
(
    int * numFrames
)
```

int *	**numFrames**	Output	Number of frames for current active solution result

---

UF_MOTION_ask_point_on_surface_constraint (view source)

Defined in: uf_motion.h

Overview
This function asks the structures of point on surface constraint parameters for a given point_on_surface_tag value.

Return
Zero for success, non-zero is error code

Environment
Internal and External

See Also
UF_MOTION_init_point_on_surface_constraint
UF_MOTION_create_point_on_surface_constraint
UF_MOTION_set_point_on_surface_constraint

History
Created in NX3

```
int UF_MOTION_ask_point_on_surface_constraint
(
    tag_t point_on_surface_tag,
    UF_MOTION_point_on_surface_data_p_t pt_on_surf_data
)
```

tag_t	**point_on_surface_tag**	Input	tag of point on surface constraint object to get the data for
UF_MOTION_point_on_surface_data_p_t	**pt_on_surf_data**	Output	The structure corresponding to the input point on surface constraint tag

---

UF_MOTION_ask_pt_crv_constraint (view source)

Defined in: uf_motion.h

Overview
This function returns the point curve structure for an input point curve constraint tag.

Environment
Internal and External

See Also
UF_MOTION_set_pt_crv_constraint

History
Created in NX2

```
int UF_MOTION_ask_pt_crv_constraint
(
    const tag_t pt_crv_tag,
    UF_MOTION_point_curve_constraint_t * pt_crv_struct
)
```

const tag_t	**pt_crv_tag**	Input	The tag of the constraint to return. This tag is the one returned from the UF_MOTION_create_pt_crv_constraint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_point_curve_constraint_t *	**pt_crv_struct**	Output	The point-curve structure for the input tag.

UF_MOTION_ask_rot_displacement_results (view source)

Defined in: uf_motion.h

Overview
This function is used to get specific rotational displacement results from the analysis results stored in the database. If this function is called when no results are in the database, an error condition will be returned.

Environment
Internal and External

See Also
UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file
UF_MOTION_ask_trans_displacement_results
UF_MOTION_ask_torque_results

History
Released in V18

```
int UF_MOTION_ask_rot_displacement_results
(
    const tag_t motion_object,
    const UF_MOTION_disp_angle_t component,
    const UF_MOTION_reference_frame_t ref_frame,
    int * number_of_results,
    double * * results
)
```

const tag_t	**motion_object**	Input	The tag of the object to get the results for. This is usually a joint, constraint, coupler, marker, or primitive.
const UF_MOTION_disp_angle_t	**component**	Input	The desired displacement angle. This can be the Euler angles that follow the body 3-1-3 convention. See the enum definition for more information.
const UF_MOTION_reference_frame_t	**ref_frame**	Input	The reference frame that the results will be resolved in (Example: UF_MOTION_absolute).
int *	**number_of_results**	Output	The number of results returned in the next argument. This will equal the number of steps in the solution.
double * *	**results**	Output to UF_*free*	The results requested. This array must be freed by the calling function using UF_free().

UF_MOTION_ask_scalar_force (view source)

Defined in: uf_motion.h

Overview
This function returns the scalar force structure for an input scalar force tag.

Environment
Internal and External

See Also
UF_MOTION_set_scalar_force

History
Created in NX2

```
int UF_MOTION_ask_scalar_force
(
    const tag_t force_tag,
    UF_MOTION_scalar_force_torque_t * force_struct
)
```

const tag_t	**force_tag**	Input	The tag of the scalar force to return information about. This tag is the one returned from UF_MOTION_create_scalar_force. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_scalar_force_torque_t *	**force_struct**	Output	The scalar force structure for the input tag.

UF_MOTION_ask_scalar_torque (view source)

Defined in: uf_motion.h

Overview
This function returns the scalar torque structure for an input scalar torque tag.

Environment
Internal and External

See Also
UF_MOTION_set_scalar_torque

History
Created in NX2

```
int UF_MOTION_ask_scalar_torque
(
    const tag_t torque_tag,
    UF_MOTION_scalar_force_torque_t * torque_struct
)
```

const tag_t	**torque_tag**	Input	The tag of the scalar torque to return information about. This tag is the one returned from UF_MOTION_create_scalar_torque. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_scalar_force_torque_t *	**torque_struct**	Output	The scalar torque structure for the input tag.

UF_MOTION_ask_solver_dof_count (view source)

Defined in: uf_motion.h

Overview
This function gets the total degree of freedom count as calculated by the motion solver for the current mechanism.

Environment
Internal and External

See Also
UF_MOTION_ask_gruebler_count

History
Created in NX2

```
int UF_MOTION_ask_solver_dof_count
(
    int * dof_count
)
```

int *	**dof_count**	Output	The total degree of freedom count.

UF_MOTION_ask_solver_parameters (view source)

Defined in: uf_motion.h

Overview
This function gets the current solver parameters from Scenario for Motion+.

Environment
Internal and External

See Also
UF_MOTION_edit_solver_parameters

History
Released in V18

```
int UF_MOTION_ask_solver_parameters
(
    UF_MOTION_solver_parameters_t * solver_params_struct
)
```

UF_MOTION_solver_parameters_t *	**solver_params_struct**	Output	The current solver parameters in Scenario for Motion.

UF_MOTION_ask_spring (view source)

Defined in: uf_motion.h

Overview
This function returns the spring structure for an input spring tag.

Environment
Internal and External

See Also
UF_MOTION_set_spring

History
Created in NX2

```
int UF_MOTION_ask_spring
(
    const tag_t spring_tag,
    UF_MOTION_spring_damper_t * spring_struct
)
```

const tag_t	**spring_tag**	Input	The tag of the spring to return information about. This tag is the one returned from the UF_MOTION_create_spring function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_spring_damper_t *	**spring_struct**	Output	The spring structure for the input tag.

UF_MOTION_ask_torque_results (view source)

Defined in: uf_motion.h

Overview
This function is used to get specific torque results from the analysis results in the database. If this function is called when no results are in the database, an error condition will be returned.

Environment
Internal and External

See Also
UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History
Released in V18

```
int UF_MOTION_ask_torque_results
(
    const tag_t motion_object,
    const UF_MOTION_vector_component_t component,
    const UF_MOTION_reference_frame_t ref_frame,
    int * number_of_results,
    double * * results
)
```

const tag_t	**motion_object**	Input	The tag of the object to get the results for. This is usually a joint, constraint, coupler, marker, or primitive.
const UF_MOTION_vector_component_t	**component**	Input	The desired component of the torque (Example: UF_MOTION_magnitude or UF_MOTION_x_component).
const UF_MOTION_reference_frame_t	**ref_frame**	Input	The reference frame that the results will be resolved in.
int *	**number_of_results**	Output	The number of results returned in the next argument. This will equal the number of steps in the solution.
double * *	**results**	Output to UF_*free*	The results requested. This array must be freed by the calling function using UF_free().

UF_MOTION_ask_trace (view source)

Defined in: uf_motion.h

Overview

This function returns the trace structure from an input trace tag.

Environment

Internal and External

See Also

UF_MOTION_edit_trace
UF_MOTION_create_trace
UF_MOTION_trace

History

Released in V18

```
int UF_MOTION_ask_trace
(
    const tag_t trace_tag,
    UF_MOTION_trace_t * trace_struct
)
```

const tag_t	**trace_tag**	Input	The tag of the trace to return. This tag is the one returned from UF_MOTION_create_trace. It

			can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
UF_MOTION_trace_t *	**trace_struct**	Output	The trace structure of the input tag.

UF_MOTION_ask_trace_explosion_to_master (view source)

Defined in: uf_motion.h

Overview
This function returns whether trace/explosion objects will get exported to the master

Environment
Internal and External

See Also
UF_MOTION_set_trace_explosion_to_master

History
Released in NX3

```
int UF_MOTION_ask_trace_explosion_to_master
(
    logical * to_master
)
```

logical *	**to_master**	Input	TRUE or FALSE

UF_MOTION_ask_trans_displacement_results (view source)

Defined in: uf_motion.h

Overview
This function is used to get specific translational displacement results from the analysis results stored in the database. If this function is called when no results are in the database, an error condition will be returned.

Environment
Internal and External

See Also
UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file
UF_MOTION_ask_rot_displacement_results
UF_MOTION_ask_velocity_results

History
Released in V18

```
int UF_MOTION_ask_trans_displacement_results
(
    const tag_t motion_object,
    const UF_MOTION_vector_component_t component,
    const UF_MOTION_reference_frame_t ref_frame,
    int * number_of_results,
    double * * results
)
```

const tag_t	**motion_object**	Input	The tag of the object to get the results for. This is usually a joint, constraint, coupler, marker, or primitive.
const UF_MOTION_vector_component_t	**component**	Input	The desired component of the displacement (Example: UF_MOTION_magnitude or UF_MOTION_x_component).
const UF_MOTION_reference_frame_t	**ref_frame**	Input	The reference frame that the results will be resolved in (Example: UF_MOTION_absolute).
int *	**number_of_results**	Output	The number of results returned in the next argument. This will equal the number of steps in the solution.
double * *	**results**	Output to UF_*free*	The results requested. This array must be freed by the calling function using UF_free().

UF_MOTION_ask_vector_force_torque (view source)

Defined in: uf_motion.h

Overview

This function returns the vector force/torque structure for an input vector force or torque tag.

Environment

Internal and External

See Also

UF_MOTION_set_vector_force_torque

History

Created in NX2

```
int UF_MOTION_ask_vector_force_torque
(
    const tag_t vobject_tag,
    UF_MOTION_vector_force_torque_t * vector_struct
)
```

const tag_t	**vobject_tag**	Input	The tag of the vector force or torque to return information about. This tag is the one returned from UF_MOTION_create_vector_force_torque. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_vector_force_torque_t *	**vector_struct**	Output	The vector force/ torque structure for the input tag.

UF_MOTION_ask_velocity_results (view source)

Defined in: uf_motion.h

Overview

This function is used to get specific velocity results from the analysis results stored in the database. Both translational and rotational velocities can be returned from this function. If this function is called when no results are in the database, an error condition will be returned.

Environment

Internal and External

See Also

UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file
UF_MOTION_ask_trans_displacement_results
UF_MOTION_ask_acceleration_results

History

Released in V18

```
int UF_MOTION_ask_velocity_results
(
    const tag_t motion_object,
    const UF_MOTION_motion_type_t type,
    const UF_MOTION_vector_component_t component,
    const UF_MOTION_reference_frame_t ref_frame,
    int * number_of_results,
    double * * results
)
```

const tag_t	**motion_object**	Input	The tag of the object to get the results for. This is usually a joint, constraint, coupler, marker, or primitive.
const UF_MOTION_motion_type_t	**type**	Input	The velocity type: UF_MOTION_rotation or UF_MOTION_translation.

const UF_MOTION_vector_component_t	**component**	Input	The desired component of the velocity (Example: UF_MOTION_magnitude or UF_MOTION_x_component).
const UF_MOTION_reference_frame_t	**ref_frame**	Input	The reference frame that the results will be resolved in.
int *	**number_of_results**	Output	The number of results returned in the next argument. This will equal the number of steps in the solution.
double * *	**results**	Output to UF_*free*	The results requested. This array must be freed by the calling function using UF_free().

UF_MOTION_calculate_static_equilibrium (view source)

Defined in: uf_motion.h

Overview

This function is used to calculate the static equilibrium of the motion model.
Following a successful completion of this function, there will be one results
step in the database. Similar to UF_MOTION_solve_model, these database results
can be accessed by the following:

1) Packaging options, including measure, trace, and interference.
2) Write the results to disk using UF_MOTION_export_adams_res_file.
3) Get specific results from a joint or marker.

Environment

Internal and External

See Also

UF_MOTION_step_articulation
UF_MOTION_solve_model
UF_MOTION_review_adams_res_file
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History

Released in V18

```
int UF_MOTION_calculate_static_equilibrium
(
    int* static_result_steps
)
```

int*	**static_result_steps**	Output	Total number of successful steps that are available in the database.

UF_MOTION_create_2D_contact (view source)

Defined in: uf_motion.h

Overview

This function creates a 2D contact in the Scenario for Motion+ database
according to the parameters passed in through the 2D contact structure.
If successful, the tag of the new 2D contact is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_2D_contact
UF_MOTION_set_2D_contact

History

Created in NX2

```
int UF_MOTION_create_2D_contact
(
    UF_MOTION_2D_contact_t * contact_struct,
    tag_t * contact_tag
)
```

UF_MOTION_2D_contact_t *	**contact_struct**	Input	The required parameters to create the 2D contact.
tag_t *	**contact_tag**	Output	The new 2D contact tag. This tag identifies the 2D contact in the database.

UF_MOTION_create_3D_contact (view source)

Defined in: uf_motion.h

Overview

This function creates a 3D contact in the Scenario for Motion+ database
according to the parameters passed in through the 3D contact structure.
If successful, the tag of the new 3D contact is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_3D_contact
UF_MOTION_set_3D_contact

History

Created in NX2

```
int UF_MOTION_create_3D_contact
(
    UF_MOTION_3D_contact_t * contact_struct,
    tag_t * contact_tag
)
```

UF_MOTION_3D_contact_t *	**contact_struct**	Input	The required parameters to create the 3D contact.
tag_t *	**contact_tag**	Output	The new 3D contact tag. This tag identifies the 3D contact in the database.

UF_MOTION_create_crv_crv_constraint (view source)

Defined in: uf_motion.h

Overview
This function create curve curve constraint

Environment
Internal and External

See Also
UF_MOTION_ask_crv_crv_constraint
UF_MOTION_create_crv_crv_constraint

History
Created in NX2

```
int UF_MOTION_create_crv_crv_constraint
(
    UF_MOTION_curve_curve_constraint_t * crv_crv_data,
    tag_t * crv_crv_tag
)
```

UF_MOTION_curve_curve_constraint_t *	**crv_crv_data**	Input	input data
tag_t *	**crv_crv_tag**	Output	created constraint

UF_MOTION_create_cylindrical_bushing (view source)

Defined in: uf_motion.h

Overview
This function creates a cylindrical bushing in the Scenario for Motion+ database according to the parameters passed in through the cylindrical bushing structure. If successful, the tag of the new cylindrical bushing is returned.

Environment
Internal and External

See Also

UF_MOTION_ask_cylindrical_bushing
UF_MOTION_set_cylindrical_bushing

History

Created in NX2.

```
int UF_MOTION_create_cylindrical_bushing
(
    UF_MOTION_cylindrical_bushing_t * bushing_struct,
    tag_t * bushing_tag
)
```

UF_MOTION_cylindrical_bushing_t *	**bushing_struct**	Input	The required parameters to create the cylindrical bushing.
tag_t *	**bushing_tag**	Output	The new bushing tag. This tag identifies the cylindrical bushing in the database.

UF_MOTION_create_damper (view source)

Defined in: uf_motion.h

Overview

This function creates a damper in the Scenario for Motion+ database according to the parameters passed in through the damper structure. If successful, the tag of the new damper is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_damper
UF_MOTION_set_damper

History

Created in NX2.

```
int UF_MOTION_create_damper
(
    UF_MOTION_spring_damper_t * damper_struct,
    tag_t * damper_tag
)
```

UF_MOTION_spring_damper_t *	**damper_struct**	Input	The required parameters to create the damper.
tag_t *	**damper_tag**	Output	The new damper tag. This tag identifies the damper in the database.

UF_MOTION_create_function (view source)

Defined in: uf_motion.h

Overview

This function creates a function object in the Scenario for Motion+ database according to the parameters passed in through the function structure.
If successful, the tag of the new function object is returned. It is highly recommended that the function definition syntax is validated before creating the function object.

Environment

Internal and External

See Also

UF_MOTION_ask_function
UF_MOTION_set_function
UF_MOTION_validate_function_syntax
UF_MOTION_get_object_derived_function

History

Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_create_function
(
    const UF_MOTION_function_t * function_struct,
    tag_t * function_tag
)
```

const UF_MOTION_function_t *	**function_struct**	Input	The required parameters to create the function.
tag_t *	**function_tag**	Output	The new function tag. This tag identifies the function object in the database.

UF_MOTION_create_general_bushing (view source)

Defined in: uf_motion.h

Overview

This function creates a general bushing in the Scenario for Motion+ database according to the parameters passed in through the general bushing structure.
If successful, the tag of the new general bushing is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_general_bushing
UF_MOTION_set_general_bushing

History

Created in NX2.

```
int UF_MOTION_create_general_bushing
(
    UF_MOTION_general_bushing_t * bushing_struct,
    tag_t * bushing_tag
)
```

UF_MOTION_general_bushing_t *	**bushing_struct**	Input	The required parameters to create the general bushing.
tag_t *	**bushing_tag**	Output	The new bushing tag. This tag identifies the general bushing in the database.

UF_MOTION_create_interference (view source)

Defined in: uf_motion.h

Overview

This function creates an interference object in the Scenario for Motion+ database according to the parameters passed in through the interference structure. If successful, the tag of the new interference object is returned. The interference object is used in the post processor to determine if two solid bodies interfere with each other at a particular time step. The interference object can then be used to create a new interference body of the intersection volume.

Environment

Internal and External

See Also

UF_MOTION_ask_interference
UF_MOTION_interference
UF_MOTION_create_interference_body

History

Released in V18.

```
int UF_MOTION_create_interference
(
    const UF_MOTION_interference_t * interfere_struct,
    tag_t * interference_tag
)
```

const UF_MOTION_interference_t *	**interfere_struct**	Input	The parameters required to create the interference object.
tag_t *	**interference_tag**	Output	The new interference tag. This tag identifies the interference object in the database.

UF_MOTION_create_interference_body (view source)

Defined in: uf_motion.h

Overview

This function creates a new body from the intersection of two interference bodies at the input step of the analysis results. The interference parameters must be defined previously. Also there must be a valid set of analysis results in the database.

Environment

Internal and External

See Also

UF_MOTION_trace
UF_MOTION_measure
UF_MOTION_solve_model
UF_MOTION_interference

History

Released in V18

```
int UF_MOTION_create_interference_body
(
    const tag_t interference_tag,
    const int analysis_step_num,
    const UF_MOTION_reference_frame_t frame,
    int * num_interference_bodies,
    tag_t * * interference_body_tags
)
```

const tag_t	**interference_tag**	Input	The tag of the interference object that contains the required parameters to perform the interference check. This tag was created using the function UF_MOTION_create_interference.
const int	**analysis_step_num**	Input	The step number to perform the interference check for. It cannot be negative and it cannot be greater than the total number of steps available in the analysis results stored in the database.
const UF_MOTION_reference_frame_t	**frame**	Input	The reference frame to place the newly created body in. If the frame is UF_MOTION_absolute, the new body will remain in the absolute position where it was created. If the frame is UF_MOTION_first_link or UF_MOTION_second_link, the new body will be attached to the links of the first or second interference solid bodies as

			defined in the interference tag. By attaching the new body to a link, it is in a position to be subtracted from the link geometry after it moves back to the design position.
int *	**num_interference_bodies**	Output	The number of interference bodies created.
tag_t * *	**interference_body_tags**	Output to UF_*free*	The newly created bodies made from the intersection of the two interference bodies. The memory for this array of tags must be freed using UF_free().

UF_MOTION_create_joint (view source)

Defined in: uf_motion.h

Overview

This function creates a joint in the Scenario for Motion+ database according to the parameters passed in through the joint structure. If successful, the tag of the new joint is returned. This function does not allow joint limits or joint motion input to be created at this point. To add these things to a joint, use UF_MOTION_set_joint_limits and UF_MOTION_set_joint_motion_input after creating the joint with this function.

Environment

Internal and External

See Also

UF_MOTION_ask_joint
UF_MOTION_set_joint_limits
UF_MOTION_set_joint_motion_input

History

Created in NX2.

```
int UF_MOTION_create_joint
(
    UF_MOTION_joint_t * joint_struct,
    tag_t * joint_tag
)
```

UF_MOTION_joint_t *	**joint_struct**	Input	The required parameters to create the joint.
tag_t *	**joint_tag**	Output	The new joint tag. This tag identifies the joint in the database.

UF_MOTION_create_joint_coupler (view source)

Defined in: uf_motion.h

Overview
This function creates a joint coupler in the Scenario for Motion+ database
according to the parameters passed in through the joint coupler structure.
If successful, the tag of the new joint coupler is returned.

Environment
Internal and External

See Also
UF_MOTION_ask_joint_coupler
UF_MOTION_set_joint_coupler

History
Created in NX2.

```
int UF_MOTION_create_joint_coupler
(
    UF_MOTION_joint_coupler_t * coupler_struct,
    tag_t * joint_coupler_tag
)
```

UF_MOTION_joint_coupler_t *	**coupler_struct**	Input	The required parameters to create the joint coupler.
tag_t *	**joint_coupler_tag**	Output	The new joint coupler tag. This tag identifies the joint coupler in the database.

UF_MOTION_create_link (view source)

Defined in: uf_motion.h

Overview
This function creates a link in the Scenario for Motion+ database according to
the parameters passed in through the link structure. If successful, the tag
of the new link is returned. This function does not create initial velocities
or mass properties. To add initial velocities or mass properties to a link,
use UF_MOTION_set_link_initial_velocity and UF_MOTION_set_link_mass_properties
after creating the link with this function.

Environment
Internal and External

See Also
UF_MOTION_ask_link
UF_MOTION_set_link_mass_properties
UF_MOTION_set_link_initial_velocity

History
Created in NX2.

```
int UF_MOTION_create_link
(
    UF_MOTION_link_t * link_struct,
    tag_t * link_tag
)
```

UF_MOTION_link_t *	**link_struct**	Input	The required parameters to create the link.
tag_t *	**link_tag**	Output	The new link tag. This tag identifies the link in the database.

UF_MOTION_create_marker (view source)

Defined in: uf_motion.h

Overview

This function creates a marker in the Scenario for Motion+ database according to the parameters passed in through the marker structure. This function is used only for the creation of user defined markers. If successful, the tag of the new marker is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_marker
UF_MOTION_set_marker

History

Created in NX2.

```
int UF_MOTION_create_marker
(
    UF_MOTION_marker_t * marker_struct,
    tag_t * marker_tag
)
```

UF_MOTION_marker_t *	**marker_struct**	Input	The required parameters to create the marker.
tag_t *	**marker_tag**	Output	The new marker tag. This tag identifies the marker in the database.

UF_MOTION_create_measurement (view source)

Defined in: uf_motion.h

Overview

This function creates a measurement object in the Scenario for Motion+ database according to the parameters passed in through the measurement structure. If successful, the tag of the new measurement object is returned.

The measurement object is used in the post processor to determine the distance or angle between two objects at a particular time step.

Environment

Internal and External

See Also

UF_MOTION_ask_measurement
UF_MOTION_edit_measurement
UF_MOTION_measure

History

Released in V18.

```
int UF_MOTION_create_measurement
(
    const UF_MOTION_measurement_t * meas_struct,
    tag_t * meas_tag
)
```

const UF_MOTION_measurement_t *	**meas_struct**	Input	The required parameters to create the measurement.
tag_t *	**meas_tag**	Output	The new measurement tag. This tag identifies the measurement in the database.

UF_MOTION_create_point_on_surface_constraint (view source)

Defined in: uf_motion.h

Overview

This function creates the structure of point on surface constraint with provided values and returns the corresponding tag_t pointer.

Return

Zero for success, non-zero is error code

Environment

Internal and External

See Also

UF_MOTION_ask_point_on_surface_constraint
UF_MOTION_init_point_on_surface_constraint
UF_MOTION_set_point_on_surface_constraint

History

Created in NX3

```
int UF_MOTION_create_point_on_surface_constraint
(
    UF_MOTION_point_on_surface_data_p_t pt_on_surf_data,
    tag_p_t point_on_surface_tag
)
```

UF_MOTION_point_on_surface_data_p_t	**pt_on_surf_data**	Input	Input structures of point on surface constraint data
tag_p_t	**point_on_surface_tag**	Output	tag of created point on surface constraint

---

UF_MOTION_create_pt_crv_constraint (view source)

Defined in: uf_motion.h

Overview
This function create point curve constraint

Environment
Internal and External

See Also
UF_MOTION_ask_pt_crv_constraint
UF_MOTION_set_pt_crv_constraint

History
Created in NX2

```
int UF_MOTION_create_pt_crv_constraint
(
    UF_MOTION_point_curve_constraint_t * pt_crv_data,
    tag_t * pt_crv_tag
)
```

UF_MOTION_point_curve_constraint_t *	**pt_crv_data**	Input	
tag_t *	**pt_crv_tag**	Output	created constraint

UF_MOTION_create_scalar_force (view source)

Defined in: uf_motion.h

Overview
This function creates a scalar force in the Scenario for Motion+ database according to the parameters passed in through the scalar force/torque structure. If successful, the tag of the new scalar force is returned.

Environment
Internal and External

See Also
UF_MOTION_ask_scalar_force
UF_MOTION_set_scalar_force

History

Created in NX2.

```
int UF_MOTION_create_scalar_force
(
    UF_MOTION_scalar_force_torque_t * force_struct,
    tag_t * force_tag
)
```

UF_MOTION_scalar_force_torque_t *	**force_struct**	Input	The required parameters to create the scalar force.
tag_t *	**force_tag**	Output	The new scalar force tag. This tag identifies the scalar force in the database.

UF_MOTION_create_scalar_torque (view source)

Defined in: uf_motion.h

Overview

This function creates a scalar torque in the Scenario for Motion+ database according to the parameters passed in through the scalar force/torque structure. If successful, the tag of the new scalar torque is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_scalar_torque
UF_MOTION_set_scalar_torque

History

Created in NX2.

```
int UF_MOTION_create_scalar_torque
(
    UF_MOTION_scalar_force_torque_t * torque_struct,
    tag_t * torque_tag
)
```

UF_MOTION_scalar_force_torque_t *	**torque_struct**	Input	The required parameters to create the scalar torque.
tag_t *	**torque_tag**	Output	The new scalar torque tag. This tag identifies the scalar torque in the database.

UF_MOTION_create_spring (view source)

Defined in: uf_motion.h

Overview

This function creates a spring in the Scenario for Motion+ database according to the parameters passed in through the spring structure. If successful, the tag of the new spring is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_spring
UF_MOTION_set_spring

History

Created in NX2.

```
int UF_MOTION_create_spring
(
    UF_MOTION_spring_damper_t * spring_struct,
    tag_t * spring_tag
)
```

UF_MOTION_spring_damper_t *	**spring_struct**	Input	The required parameters to create the spring.
tag_t *	**spring_tag**	Output	The new spring tag. This tag identifies the spring in the database.

UF_MOTION_create_trace (view source)

Defined in: uf_motion.h

Overview

This function creates a trace object in the Scenario for Motion+ database according to the parameters passed in through the trace structure. If successful, the tag of the new trace object is returned. The trace object is used in the post processor to create a copy of a solid body in its simulated position at a particular time step.

Environment

Internal and External

See Also

UF_MOTION_ask_trace
UF_MOTION_edit_trace
UF_MOTION_trace

History

Released in V18.

```
int UF_MOTION_create_trace
(
    const UF_MOTION_trace_t * trace_struct,
    tag_t * trace_tag
)
```

const UF_MOTION_trace_t *	**trace_struct**	Input	The required parameters to create the trace.
tag_t *	**trace_tag**	Output	The new trace tag. This tag identifies the trace in the database.

UF_MOTION_create_vector_force_torque (view source)

Defined in: uf_motion.h

Overview

This function creates a vector force or torque in the Scenario for Motion+ database according to the parameters passed in through the vector force/torque structure. If successful, the tag of the new vector force or torque is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_vector_force_torque
UF_MOTION_set_vector_force_torque

History

Created in NX2.

```
int UF_MOTION_create_vector_force_torque
(
    UF_MOTION_vector_force_torque_t * vector_struct,
    tag_t * vobject_tag
)
```

UF_MOTION_vector_force_torque_t *	**vector_struct**	Input	The required parameters to create the vector force or torque.
tag_t *	**vobject_tag**	Output	The new vector force or torque tag. This tag identifies the vector force or torque in the database.

UF_MOTION_delete_function (view source)

Defined in: uf_motion.h

Overview

This function deletes the motion function object from the database.
If the function object is still referenced by another motion object,

it will not be deleted and an error code will be returned.

Environment
Internal and External

See Also
UF_MOTION_ask_function_tag_from_name
UF_MOTION_find_all_functions

History
Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_delete_function
(
    const tag_t function_tag
)
```

const tag_t	**function_tag**	Input	Tag of function object to delete.

UF_MOTION_delete_interference (view source)

Defined in: uf_motion.h

Overview
This function deletes the input interference object.

Environment
Internal and External

See Also
UF_MOTION_ask_interference
UF_MOTION_create_interference
UF_MOTION_interference

History
Released in V18

```
int UF_MOTION_delete_interference
(
    const tag_t interfere_tag
)
```

const tag_t	**interfere_tag**	Input	The tag of the interference to delete. This tag is returned from the function UF_MOTION_create_interference. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).

UF_MOTION_delete_measurement (view source)

Defined in: uf_motion.h

Overview
This function deletes the input measurement object.

Environment
Internal and External

See Also
UF_MOTION_ask_measurement
UF_MOTION_create_measurement
UF_MOTION_measure

History
Released in V18

```
int UF_MOTION_delete_measurement
(
    const tag_t meas_tag
)
```

const tag_t	**meas_tag**	Input	The tag of the measurement to delete. This tag is returned from UF_MOTION_create_measurement. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).

UF_MOTION_delete_trace (view source)

Defined in: uf_motion.h

Overview
This function deletes the input trace object.

Environment
Internal and External

See Also
UF_MOTION_ask_trace
UF_MOTION_create_trace
UF_MOTION_trace

History
Released in V18

```
int UF_MOTION_delete_trace
(
    const tag_t trace_tag
)
```

const tag_t	**trace_tag**	Input	The tag of the trace to delete. This tag is returned from UF_MOTION_create_trace. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).

UF_MOTION_edit_artic_step_size (view source)

Defined in: uf_motion.h

Overview

All joints that have a joint motion input defined can be articulated even though the motion input type might not be articulation. This function sets the articulation step size for the input joint tag. The tag must belong to a joint that has a motion input defined or else an error condition is returned.

Environment

Internal and External

See Also

UF_MOTION_ask_artic_step_size
UF_MOTION_step_articulation

History

Released in V18

```
int UF_MOTION_edit_artic_step_size
(
    const tag_t joint_tag,
    const double step_size
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to set the articulation step size on. This tag must belong to a joint that has a motion input defined or else an error condition is returned. This tag is returned from UF_MOTION_create_joint. It can also be found using cycle functions (eg. UF_OBJ_cycle_objs_by_type).
const double	**step_size**	Input	The new size of the articulation joint step. The units of the step are based on the units of the joint.

UF_MOTION_edit_interference (view source)

Defined in: uf_motion.h

Overview

This function sets the input interference parameters to the input interference tag.

Environment

Internal and External

See Also

UF_MOTION_ask_interference
UF_MOTION_create_interference
UF_MOTION_interference

History

Released in V18

```
int UF_MOTION_edit_interference
(
    const tag_t interfere_tag,
    const UF_MOTION_interference_t * interfere_struct
)
```

const tag_t	**interfere_tag**	Input	The tag of the interference to edit. This tag is returned from the function UF_MOTION_create_interference. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
const UF_MOTION_interference_t *	**interfere_struct**	Input	The interference structure for the input tag.

UF_MOTION_edit_measurement (view source)

Defined in: uf_motion.h

Overview

This function sets the input measurement parameters to the input measurement tag.

Environment

Internal and External

See Also

UF_MOTION_ask_measurement
UF_MOTION_create_measurement
UF_MOTION_measure

History

Released in V18

```
int UF_MOTION_edit_measurement
(
    const tag_t meas_tag,
    const UF_MOTION_measurement_t * meas_struct
)
```

const tag_t	**meas_tag**	Input	The tag of the measurement to edit. This tag is returned from UF_MOTION_create_measurement. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
const UF_MOTION_measurement_t *	**meas_struct**	Input	The measurement structure for the input tag.

UF_MOTION_edit_solver_parameters (view source)

Defined in: uf_motion.h

Overview
This function sets new solver parameters to Scenario for Motion+.

Environment
Internal and External

See Also
UF_MOTION_ask_solver_parameters

History
Released in V18

```
int UF_MOTION_edit_solver_parameters
(
    const UF_MOTION_solver_parameters_t * solver_params
)
```

const UF_MOTION_solver_parameters_t *	**solver_params**	Input	The new solver parameters for Scenario for Motion.

UF_MOTION_edit_trace (view source)

Defined in: uf_motion.h

Overview
This function sets the input trace parameters to the input trace tag.

Environment
Internal and External

See Also
UF_MOTION_ask_trace
UF_MOTION_create_trace
UF_MOTION_trace

History
Released in V18

```
int UF_MOTION_edit_trace
(
    const tag_t trace_tag,
    const UF_MOTION_trace_t * trace_struct
)
```

const tag_t	**trace_tag**	Input	The tag of the trace to edit. This tag is the one returned from UF_MOTION_create_trace. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
const UF_MOTION_trace_t *	**trace_struct**	Input	The trace structure for the input tag.

UF_MOTION_export_adams_anl_file (view source)

Defined in: uf_motion.h

Overview

Exports the current Motion model as an ANL file which can then be read into Adams or any other application which reads Adams ANL files.

Environment

Internal or External

See Also

UF_MOTION_init_stl_parameters_struct
UF_MOTION_solve_model
UF_MOTION_calculate_static_equilibrium
UF_MOTION_step_articulation
UF_MOTION_export_adams_res_file

History

Released in NX3

```
int UF_MOTION_export_adams_anl_file
(
    const char * file_name,
    UF_MOTION_anl_geometry_format_t geometry_format,
    UF_MOTION_stl_parameters_p_t stl_params
)
```

const char *	**file_name**	Input	The filename of the .anl file to write. It should include the entire path to the file. The filename should end with .anl followed by a terminating NULL character.
UF_MOTION_anl_geometry_format_t	**geometry_format**	Input	The format to export the geometry.
UF_MOTION_stl_parameters_p_t	**stl_params**	Input	If geometry_format == UF_MOTION_stl, then this arg contains the parameters to export the geometry as STL data. Ignored if geometry_format != UF_MOTION_stl.

UF_MOTION_export_adams_res_file (view source)

Defined in: uf_motion.h

Overview

This function writes the ADAMS .res file to disk. This file contains the results of the latest solve-static equilirium, articulation, or a full simulation. This is the only way to save the solver results; the results are not saved in the part file. However, the results file does not need to be written for any other purpose. This file is only used to store results for later use. If the model is not changed, these results can be read back in to Scenario for Motion for further processing.

Environment

Internal and External

See Also

UF_MOTION_init_stl_parameters_struct
UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_calculate_static_equilibrium
UF_MOTION_export_adams_anl_file
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History

Released in V18

```
int UF_MOTION_export_adams_res_file
(
    char * file_name
)
```

char *	**file_name**	Input	The filename of the .res file to write. It should include the entire path to the file. The filename should end with .res followed by a terminating NULL character.

UF_MOTION_export_to_product_vision (view source)

Defined in: uf_motion.h

Overview

This function is used to export motion animation to product vision.
This function creates a VFM file which is the ProductVision format for motion data.
If this function is called when no results are in the
database, an error condition will be returned.To populate the results into the database the motion model has to be solved.

Environment

Internal and External

See Also

UF_MOTION_solve_model
UF_MOTION_step_articulation

History
Released in V18

int UF_MOTION_export_to_product_vision
(
const char * full_file_name,
UF_MOTION_PV_export_type_t export_option
)

const char *	**full_file_name**	Input	The full name of the VFM file to be exported. The name should include the directory where the newly created file needs to be placed. (e.g., /users/test/file_name note: On windows use '\' instead of '/') If the length of the string is zero then the scenario directory will be the default directory and the scenario name will be the default file name
UF_MOTION_PV_export_type_t	**export_option**	Input	The option to export VFM and JT file or only the VFM file

---

UF_MOTION_file_suppress_warnings (view source)

Defined in: uf_motion.h

Overview
Suppress warning messages for reparenting the master part when open a scenario part.

Environment
Internal or External

Input : logical flag = true will turn off the warnings

Returns
0 - Successful
else Unsuccessful

History
Released in NX801

int UF_MOTION_file_suppress_warnings
(
logical flag
)

logical	**flag**	Input	The suppress flag, true will turn off the warnings

---

UF_MOTION_find_all_functions (view source)

Defined in: uf_motion.h

Overview
This function finds all the motion function objects in the current part.

Environment
Internal and External

See Also
UF_MOTION_ask_function_tag_from_name

History
Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_find_all_functions
(
    tag_t * * function_tags,
    int * num_functions
)
```

tag_t * *	**function_tags**	Output to UF_*free*	Tag array of all function objects in current part.
int *	**num_functions**	Output	Number of function tags.

UF_MOTION_get_object_derived_function (view source)

Defined in: uf_motion.h

Overview
This function returns the string of a derived function. A derived function references a motion object for a particular result type. All motion objects other than links are allowed for reference in a derived function.

Environment
Internal and External

See Also
UF_MOTION_validate_function_syntax
UF_MOTION_create_function

History
Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_get_object_derived_function
(
    const tag_t motion_obj_tag,
    UF_MOTION_func_result_type_t func_type,
    UF_MOTION_func_component_type_t func_comp,
    UF_MOTION_func_ref_frame_type_t ref_frame,
    char * * derived_func_string
)
```

const tag_t	**motion_obj_tag**	Input	The motion object being referenced for the derived function.
UF_MOTION_func_result_type_t	**func_type**	Input	The result type requested for the derived function. See uf_motion_types.h
UF_MOTION_func_component_type_t	**func_comp**	Input	The result component requested for the derived function. See uf_motion_types.h
UF_MOTION_func_ref_frame_type_t	**ref_frame**	Input	The reference frame for the derived function. See uf_motion_types.h
char * *	**derived_func_string**	Output to UF_*free*	Formulated string of a derived function.

UF_MOTION_init_2D_contact_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input 2D contact structure with standard default values. It is highly recommended that all 2D contact structures get initialized with this function before they are used.

Environment

Internal and External

History

Created in NX2

```
int UF_MOTION_init_2D_contact_struct
(
    UF_MOTION_2D_contact_t * contact_struct
)
```

UF_MOTION_2D_contact_t *	**contact_struct**	Output	The 2D_contact structure to be initialized.

UF_MOTION_init_3D_contact_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input 3D contact structure with standard default values. It is highly recommended that all 3D contact structures get initialized with this function before they are used.

Environment
Internal and External

History
Created in NX2

```
int UF_MOTION_init_3D_contact_struct
(
    UF_MOTION_3D_contact_t * contact_struct
)
```

UF_MOTION_3D_contact_t *	**contact_struct**	Output	The 3D_contact structure to be initialized.

UF_MOTION_init_articulation (view source)

Defined in: uf_motion.h

Overview
This function initializes the solver for an articulation run. It must be called prior to any calls to UF_MOTION_step_articulation and it must be paired with a call to UF_MOTION_terminate_articulation.

Environment
Internal and External

See Also
UF_MOTION_step_articulation
UF_MOTION_edit_artic_step_size
UF_MOTION_terminate_articulation
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History
Created in V18

```
int UF_MOTION_init_articulation
(
    void
)
```

UF_MOTION_init_crv_crv_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input curve curve structure with standard default values. It is highly recommended that all curve curve structures get initialized with this function before they are used.

Environment

Internal and External

History

Created in NX2

```
int UF_MOTION_init_crv_crv_struct
(
    UF_MOTION_curve_curve_constraint_t * crv_crv_struct
)
```

UF_MOTION_curve_curve_constraint_t *	**crv_crv_struct**	Input / Output	The curve curve structure to be initialized.

UF_MOTION_init_cylindrical_bushing_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input cylindrical bushing structure with standard default values. It is highly recommended that all cylindrical bushing structures get initialized with this function before they are used.

Environment

Internal and External

History

Created in NX2

```
int UF_MOTION_init_cylindrical_bushing_struct
(
    UF_MOTION_cylindrical_bushing_t * bushing_struct
)
```

UF_MOTION_cylindrical_bushing_t *	**bushing_struct**	Output	The bushing data structure to be initialized.

UF_MOTION_init_function_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input function structure with standard default values. It is highly recommended that all function structures get initialized with this function before they are used.

Environment

Internal and External

History

Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_init_function_struct
(
    UF_MOTION_function_t * function_struct
)
```

UF_MOTION_function_t *	**function_struct**	Output	The function structure to be initialized.

UF_MOTION_init_general_bushing_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input general bushing structure with standard default values. It is highly recommended that all general bushing structures get initialized with this function before they are used.

Environment

Internal and External

History

Created in NX2

```
int UF_MOTION_init_general_bushing_struct
(
    UF_MOTION_general_bushing_t * bushing_struct
)
```

UF_MOTION_general_bushing_t *	**bushing_struct**	Output	The bushing data structure to be initialized.

UF_MOTION_init_interference_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input interference structure with standard default values. It is highly recommended that all interference structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_trace_struct
UF_MOTION_init_measurement_struct

History
Released in V18

```
int UF_MOTION_init_interference_struct
(
    UF_MOTION_interference_t * interference_struct
)
```

UF_MOTION_interference_t *	**interference_struct**	Output	The interference structure to be initialized.

UF_MOTION_init_joint_coupler_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input coupler structure with standard default values. It is highly recommended that all coupler structures get initialized with this function before they are used.

Environment
Internal and External

See Also
UF_MOTION_init_link_struct
UF_MOTION_init_joint_struct

History
Created in NX2

```
int UF_MOTION_init_joint_coupler_struct
(
    UF_MOTION_joint_coupler_t * coupler_struct
)
```

UF_MOTION_joint_coupler_t *	**coupler_struct**	Output	The coupler structure to be initialized.

UF_MOTION_init_joint_limits_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input joint limits structure with standard default values. It is highly recommended that all joint limits structures get initialized with this function before they are used.

Environment
Internal and External

See Also

UF_MOTION_init_joint_struct
UF_MOTION_init_joint_motion_input_struct

History

Created in NX2

int UF_MOTION_init_joint_limits_struct
(
UF_MOTION_joint_limits_t * joint_limits_struct
)

UF_MOTION_joint_limits_t *	**joint_limits_struct**	Output	The joint limits structure to be initialized.

UF_MOTION_init_joint_motion_input_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input joint motion input structure with standard default values. It is highly recommended that all joint motion input structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_joint_struct
UF_MOTION_init_joint_limits_struct

History

Created in NX2

int UF_MOTION_init_joint_motion_input_struct
(
UF_MOTION_joint_motion_input_t * motion_input_struct
)

UF_MOTION_joint_motion_input_t *	**motion_input_struct**	Output	The joint motion input structure to be initialized.

UF_MOTION_init_joint_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input joint structure with standard default values. It is highly recommended that all joint structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_joint_limits_struct
UF_MOTION_init_joint_motion_input_struct

History

Created in NX2

```
int UF_MOTION_init_joint_struct
(
    UF_MOTION_joint_t * joint_struct
)
```

UF_MOTION_joint_t *	**joint_struct**	Output	The joint structure to be initialized.

UF_MOTION_init_link_mass_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input link mass structure with standard default values. It is highly recommended that all link mass structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_link_struct
UF_MOTION_init_link_velocity_struct

History

Created in NX2

```
int UF_MOTION_init_link_mass_struct
(
    UF_MOTION_link_mass_properties_t * link_mass_struct
)
```

UF_MOTION_link_mass_properties_t *	**link_mass_struct**	Output	The link mass structure to be initialized.

UF_MOTION_init_link_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input link structure with standard default values. It is highly recommended that all link structures get initialized with this function before they are used.

Environment
Internal and External

See Also
UF_MOTION_init_link_mass_struct
UF_MOTION_init_link_velocity_struct

History
Created in NX2

```
int UF_MOTION_init_link_struct
(
    UF_MOTION_link_t * link_struct
)
```

UF_MOTION_link_t *	**link_struct**	Input / Output	The link structure to be initialized.

UF_MOTION_init_link_velocity_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input link initial velocity structure with standard default values. It is highly recommended that all link initial velocity structures get initialized with this function before they are used.

Environment
Internal and External

See Also
UF_MOTION_init_link_struct
UF_MOTION_init_link_mass_struct

History
Created in NX2

```
int UF_MOTION_init_link_velocity_struct
(
    UF_MOTION_link_initial_velocity_t * link_velocity_struct
)
```

UF_MOTION_link_initial_velocity_t *	**link_velocity_struct**	Input / Output	The link initial velocity structure

			to be initialized.

UF_MOTION_init_marker_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input marker structure with standard default values. It is highly recommended that all marker structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_link_struct
UF_MOTION_init_joint_struct

History

Created in NX2

```
int UF_MOTION_init_marker_struct
(
    UF_MOTION_marker_t * marker_struct
)
```

UF_MOTION_marker_t *	**marker_struct**	Output	The marker structure to be initialized.

UF_MOTION_init_measurement_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input measurement structure with standard default values. It is highly recommended that all measurement structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_trace_struct
UF_MOTION_init_interference_struct

History

Released in V18

```
int UF_MOTION_init_measurement_struct
(
    UF_MOTION_measurement_t * measurement_struct
)
```

UF_MOTION_measurement_t *	**measurement_struct**	Output	The measurement structure to be initialized.

UF_MOTION_init_point_on_surface_constraint (view source)

Defined in: uf_motion.h

Overview
This function initializes the structures of point on surface constraint parameters with standard default values. It is highly recommended that all point on surface parameters structures get initialized with this function before they are used.

Return
Zero for success, non-zero is error code

Environment
Internal and External

See Also
UF_MOTION_ask_point_on_surface_constraint
UF_MOTION_create_point_on_surface_constraint
UF_MOTION_set_point_on_surface_constraint

History
Created in NX3

```
int UF_MOTION_init_point_on_surface_constraint
(
    UF_MOTION_point_on_surface_data_p_t point_on_surf_data
)
```

UF_MOTION_point_on_surface_data_p_t	**point_on_surf_data**	Output	The empty struct to retrieve the default Point on surface parameters.

UF_MOTION_init_pt_crv_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input point curve structure with standard default values. It is highly recommended that all point curve structures get initialized with this function before they are used.

Environment

Internal and External

History
Created in NX2

int UF_MOTION_init_pt_crv_struct
(
    UF_MOTION_point_curve_constraint_t * pt_crv_struct
)

UF_MOTION_point_curve_constraint_t *	**pt_crv_struct**	Input / Output	The point curve structure to be initialized.

UF_MOTION_init_scalar_force_torque_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input scalar force/torque structure with standard default values. It is highly recommended that all scalar force/ torque structures get initialized with this function before they are used.

Environment
Internal and External

History
Created in NX2

int UF_MOTION_init_scalar_force_torque_struct
(
    UF_MOTION_scalar_force_torque_t * scalar_struct
)

UF_MOTION_scalar_force_torque_t *	**scalar_struct**	Output	The scalar force/torque data structure to be initialized.

UF_MOTION_init_solver_parameters_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input solver parameters structure with standard default values. It is highly recommended that all solver parameters structures get initialized with this function before they are used.

Environment
Internal and External

See Also

UF_MOTION_init_trace_struct
UF_MOTION_init_interference_struct

History
Released in V18

```
int UF_MOTION_init_solver_parameters_struct
(
    UF_MOTION_solver_parameters_t * solver_params_struct
)
```

UF_MOTION_solver_parameters_t *	**solver_params_struct**	Output	The solver parameters structure to be initialized.

UF_MOTION_init_spring_damper_struct (view source)

Defined in: uf_motion.h

Overview
This function initializes the input spring/damper structure with standard default values. It is highly recommended that all spring/damper structures get initialized with this function before they are used.

Environment
Internal and External

History
Created in NX2

```
int UF_MOTION_init_spring_damper_struct
(
    UF_MOTION_spring_damper_t * spring_damper_struct
)
```

UF_MOTION_spring_damper_t *	**spring_damper_struct**	Output	The spring/damper data structure to be initialized.

UF_MOTION_init_stl_parameters_struct (view source)

Defined in: uf_motion.h

Overview
Initialize the UF_MOTION_stl_parameters_t structure with default values in preparation to send to UF_MOTION_export_adams_anl_file.

Environment
Internal or External

See Also

UF_MOTION_export_adams_anl_file

History

Released in NX3

int UF_MOTION_init_stl_parameters_struct
(
UF_MOTION_stl_parameters_t * stl_params
)

UF_MOTION_stl_parameters_t *	**stl_params**	Output	The empty struct to retrieve the default STL parameters.

UF_MOTION_init_trace_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input trace structure with standard default values. It is highly recommended that all trace structures get initialized with this function before they are used.

Environment

Internal and External

See Also

UF_MOTION_init_measurement_struct
UF_MOTION_init_interference_struct

History

Released in V18

int UF_MOTION_init_trace_struct
(
UF_MOTION_trace_t * trace_struct
)

UF_MOTION_trace_t *	**trace_struct**	Output	The trace structure to be initialized.

UF_MOTION_init_vector_force_torque_struct (view source)

Defined in: uf_motion.h

Overview

This function initializes the input vector force/torque structure with standard default values. It is highly recommended that all vector force/ torque structures get initialized with this function before they are used.

Environment

Internal and External

History

Created in NX2

```
int UF_MOTION_init_vector_force_torque_struct
(
    UF_MOTION_vector_force_torque_t * vector_struct
)
```

UF_MOTION_vector_force_torque_t *	**vector_struct**	Output	The vector force/torque data structure to be initialized.

UF_MOTION_initialize (view source)

Defined in: uf_motion.h

Overview

This function initializes the Scenario for Motion+ application by initializing the internal databases and solvers. This function also performs the license checking for Scenario for Motion+. The application cannot be used unless a valid license is found.

Environment

Internal and External

See Also

UF_MOTION_terminate
UF_MOTION_is_initialized

History

Released in V18

```
int UF_MOTION_initialize
(
    void
)
```

UF_MOTION_interference (view source)

Defined in: uf_motion.h

Overview

This function is used to determine if two solid bodies interfere with each other at a particular step of the analysis results. The interference parameters must be defined previously. Also there must be a valid set of analysis results in the database.

Environment

Internal and External

See Also

UF_MOTION_trace
UF_MOTION_measure
UF_MOTION_solve_model
UF_MOTION_create_interference_body

History

Released in V18

```
int UF_MOTION_interference
(
    const tag_t interference_tag,
    const int step_num,
    int * interference_result
)
```

const tag_t	**interference_tag**	Input	The tag of the interference object that contains the required parameters to perform the interference check. This tag was created using the function UF_MOTION_create_interference.
const int	**step_num**	Input	The step number to perform the interference check for. It cannot be negative and it cannot be greater than the total number of steps available in the analysis results stored in the database.
int *	**interference_result**	Output	The results of the interference check are returned as a boolean. A value of TRUE means the objects interfere while a value of false means they do not interfere.

UF_MOTION_is_initialized (view source)

Defined in: uf_motion.h

Overview

This function is used to determine if the Scenario for Motion+ application has been properly initialized.

Environment

Internal and External

See Also

UF_MOTION_initialize
UF_MOTION_terminate

History

Released in V18

```
int UF_MOTION_is_initialized
(
    logical * true_false
```

)

logical *	**true_false**	Output	If Scenario for Motion is properly initialized, this value will be non-zero. If Scenario for Motion is not initialized, this value will be zero.

UF_MOTION_list_connections (view source)

Defined in: uf_motion.h

Overview
Generate an ascii text file containing a listing of the Motion connections, degrees of freedom, etc.

Environment
Internal and External

See Also
UF_MOTION_write_object_info

History
Released in NX3

int UF_MOTION_list_connections
(
const char * output_file_with_path
)

const char *	**output_file_with_path**	Input	The full path to the file that the listing should be written to. If this file already exists it will be overwritten.

UF_MOTION_measure (view source)

Defined in: uf_motion.h

Overview
This function performs a measurement between specific objects at a particular step in the analysis results. The measurement parameters must be set in the database prior to calling this function using UF_MOTION_create_measurement. There must also be a valid set of analysis results in the database before this function is called.

Environment
Internal and External

See Also
UF_MOTION_create_measurement
UF_MOTION_solve_model
UF_MOTION_trace

UF_MOTION_interference

History

Released in V18

```
int UF_MOTION_measure
(
    const tag_t measurement_tag,
    const int animation_step,
    double * measurement
)
```

const tag_t	**measurement_tag**	Input	The tag of the measurement to be performed. It was created using UF_MOTION_create_measurement and stores all of the parameters needed to complete the measurement.
const int	**animation_step**	Input	The step number of the analysis results currently loaded in the database. It cannot be negative and it cannot be greater than the total number of steps in the analysis results.
double *	**measurement**	Output	The results of the measurement. The units of this result depends on the type of measurement and the system of units the model is currently in.

UF_MOTION_remove_joint_limits (view source)

Defined in: uf_motion.h

Overview

This function removes the joint limits from the input joint tag. If the input joint does not have joint limits defined, an error condition is returned.

Environment

Internal and External

See Also

UF_MOTION_set_joint_limits
UF_OBJ_delete_object

History

Created in NX2

```
int UF_MOTION_remove_joint_limits
(
    const tag_t joint_tag
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to remove the joint limits from.

UF_MOTION_remove_joint_motion_input (view source)

Defined in: uf_motion.h

Overview
This function removes the joint motion input from the input joint tag. If the input joint tag does not have a joint input defined, an error condition is returned.

Environment
Internal and External

See Also
UF_MOTION_set_joint_motion_input
UF_OBJ_delete_object

History
Created in NX2

```
int UF_MOTION_remove_joint_motion_input
(
    const tag_t joint_tag
)
```

const tag_t	**joint_tag**	Input	The tag to remove the joint motion inputs from.

UF_MOTION_remove_link_initial_velocity (view source)

Defined in: uf_motion.h

Overview
This function removes all initial velocities from the input link tag.

Environment
Internal and External

See Also
UF_MOTION_set_link_initial_velocity

History
Created in NX2

```
int UF_MOTION_remove_link_initial_velocity
(
    const tag_t link_tag
)
```

const tag_t	**link_tag**	Input	The tag of the link to remove the initial velocities from.

UF_MOTION_remove_link_mass_properties (view source)

Defined in: uf_motion.h

Overview

This function removes mass properties from the input link tag,
To revert to the default calculated mass properties for solid bodies,
another call must be made to UF_MOTION_set_link_mass_properties.
All of the non-solid geometry will be massless. If there are no mass
properties defined on the input link, an error condition will be returned
during solve time. To treat solid bodies as massless, use the mass
property toggle in the solver parameters structure.

Environment

Internal and External

See Also

UF_MOTION_set_link_mass_properties
UF_OBJ_delete_object
UF_MOTION_edit_solver_parameters

History

Created in NX2

```
int UF_MOTION_remove_link_mass_properties
(
    const tag_t link_tag
)
```

const tag_t	**link_tag**	Input	The tag of the link to remove the user defined mass properties from.

UF_MOTION_review_adams_res_file (view source)

Defined in: uf_motion.h

Overview

This function reviews the existing result file of the motion model. Prior to
calling this function the model must be analyzed atleast once.

Inputs: Result file name (without path)

Outputs: None

Return

Zero for success, non-zero is error code

Environment

Internal and External

See Also

UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_calculate_static_equilibrium
UF_MOTION_export_adams_res_file

UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History

Created in NX4

int UF_MOTION_review_adams_res_file
(
char * res_file
)

char *	**res_file**	Input	The filename (only) of the .res file to write. The filename should end with .res followed by a terminating NULL character.

UF_MOTION_review_result_file (view source)

Defined in: uf_motion.h

Overview

This function reviews the existing result file of the motion model. Prior to calling this function the model must be analyzed at least once.

Inputs: Result file name (without path)

Outputs: None

Return

Zero for success, non-zero is error code

Environment

Internal and External

See Also

UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_calculate_static_equilibrium
UF_MOTION_export_adams_res_file
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History

Created in NX801

int UF_MOTION_review_result_file
(
const char * res_file
)

const char *	**res_file**	Input	The filename should be .rad for Recurdyn .res for Adams. The filename should end with .rad or .res followed by a terminating NULL character..

UF_MOTION_set_2D_contact (view source)

Defined in: uf_motion.h

Overview
This function sets 2D contact properties for the input 2D contact tag.

Environment
Internal and External

See Also
UF_MOTION_ask_2D_contact

History
Created in NX2

```
int UF_MOTION_set_2D_contact
(
    const tag_t contact_tag,
    UF_MOTION_2D_contact_t * contact_struct
)
```

const tag_t	**contact_tag**	Input	The tag of the 2D contact to set. This tag is the one returned from UF_MOTION_create_2D_contact. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_2D_contact_t *	**contact_struct**	Input	The 2D contact structure for the input tag.

UF_MOTION_set_3D_contact (view source)

Defined in: uf_motion.h

Overview
This function sets 3D contact properties for the input 3D contact tag.

Environment
Internal and External

See Also
UF_MOTION_ask_3D_contact

History
Created in NX2

```
int UF_MOTION_set_3D_contact
(
    const tag_t contact_tag,
    UF_MOTION_3D_contact_t * contact_struct
)
```

const tag_t	**contact_tag**	Input	The tag of the 3D contact to set. This tag is the one returned from UF_MOTION_create_3D_contact. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_3D_contact_t *	**contact_struct**	Input	The 3D contact structure for the input tag.

UF_MOTION_set_3D_contact_method (view source)

Defined in: uf_motion.h

Overview
This function sets the 3D contact method to be used.

Return
Zero for success, non-zero is error code

Environment
Internal and External

See Also
UF_MOTION_ask_3D_contact_method

History
Released in NX3

```
int UF_MOTION_set_3D_contact_method
(
    UF_MOTION_3d_contact_method_t contact_method,
    int facet_contact_tolerance
)
```

UF_MOTION_3d_contact_method_t	**contact_method**	Input	the 3d contact method to be used by Adams
int	**facet_contact_tolerance**	Input	ignored unless contact_method == UF_MOTION_faceted_contact Must be a value 1 to 100. 1 means Fast but may not be accurate 100 means accurate but may not be fast

---

UF_MOTION_set_active_solution (view source)

Defined in: uf_motion.h

Overview

Set the active solution for current motion simulation

Environment

Internal or External

Input : tag_t active_solution

Returns

0 - Successful
else Unsuccessful

History

Released in NX801

```
int UF_MOTION_set_active_solution
(
    tag_t active_solution
)
```

tag_t	**active_solution**	Input	The active solution

UF_MOTION_set_angular_units (view source)

Defined in: uf_motion.h

Overview

This function sets the angular units for this mechanism

Environment

Internal and External

See Also

UF_MOTION_ask_angular_units

History

Released in NX3

```
int UF_MOTION_set_angular_units
(
    UF_MOTION_angular_units_type_t angle_units
)
```

UF_MOTION_angular_units_type_t	**angle_units**	Input	The new units

---

UF_MOTION_set_articulation_stop_tolerance (view source)

Defined in: uf_motion.h

Overview

This function sets the articulation stop tolerance. This is used by Adams to calculate the stop positions for stop events in Articulation.

Environment
Internal and External

See Also
UF_MOTION_ask_articulation_stop_tolerance

```
int UF_MOTION_set_articulation_stop_tolerance
(
    double stop_tolerance
)
```

double	**stop_tolerance**	Input

UF_MOTION_set_cylindrical_bushing (view source)

Defined in: uf_motion.h

Overview
This function sets the input cylindrical bushing parameters to the input cylindrical bushing tag.

Environment
Internal and External

See Also
UF_MOTION_ask_cylindrical_bushing

History
Created in NX2

```
int UF_MOTION_set_cylindrical_bushing
(
    const tag_t bushing_tag,
    const UF_MOTION_cylindrical_bushing_t * bushing_struct
)
```

const tag_t	**bushing_tag**	Input	The tag of the cylindrical bushing to set. This tag is the one returned from the UF_MOTION_create_cylindrical_bushing function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_cylindrical_bushing_t *	**bushing_struct**	Input	The cylindrical bushing structure for the input tag.

UF_MOTION_set_damper (view source)

Defined in: uf_motion.h

Overview

This function sets the input damper parameters to the input damper tag. The damper parameters that can be set are the damping rate and the link attachment points.

Environment

Internal and External

See Also

UF_MOTION_ask_damper

History

Created in NX2

```
int UF_MOTION_set_damper
(
    const tag_t damper_tag,
    const UF_MOTION_spring_damper_t * damper_struct
)
```

const tag_t	**damper_tag**	Input	The tag of the damper to set. This tag is the one returned from the UF_MOTION_create_damper function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_spring_damper_t *	**damper_struct**	Input	The damper structure for the input tag.

UF_MOTION_set_function (view source)

Defined in: uf_motion.h

Overview

This function sets the input function parameters to the input function tag.

Environment

Internal and External

See Also

UF_MOTION_ask_function
UF_MOTION_validate_function_syntax
UF_MOTION_get_object_derived_function

History

Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_set_function
(
    const tag_t function_tag,
    const UF_MOTION_function_t * function_struct
)
```

const tag_t	**function_tag**	Input	The tag of the function to set. This tag is the one returned from UF_MOTION_create_function.
const UF_MOTION_function_t *	**function_struct**	Input	The function structure for the input tag.

UF_MOTION_set_general_bushing (view source)

Defined in: uf_motion.h

Overview
This function sets the input general bushing parameters to the input general bushing tag.

Environment
Internal and External

See Also
UF_MOTION_ask_general_bushing

History
Created in NX2

```
int UF_MOTION_set_general_bushing
(
    const tag_t bushing_tag,
    const UF_MOTION_general_bushing_t * bushing_struct
)
```

const tag_t	**bushing_tag**	Input	The tag of the general bushing to set. This tag is the one returned from the UF_MOTION_create_general_bushing function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_general_bushing_t *	**bushing_struct**	Input	The general bushing structure for the input tag.

UF_MOTION_set_gravitational_constants (view source)

Defined in: uf_motion.h

Overview

This function sets the graviational constants in this mechanism

Environment
Internal and External

See Also
UF_MOTION_ask_gravitational_constants

History
Released in NX3

```
int UF_MOTION_set_gravitational_constants
(
    double gravitational_constants [ 3 ]
)
```

double	**gravitational_constants [3]**	Input	Vector defining gravitational force

---

UF_MOTION_set_icon_scale_factor (view source)

Defined in: uf_motion.h

Overview
This function sets the global icon scale.

Environment
Internal and External

See Also
UF_MOTION_ask_icon_scale_factor

History
Released in NX3

```
int UF_MOTION_set_icon_scale_factor
(
    double scale
)
```

double	**scale**	Input

UF_MOTION_set_joint (view source)

Defined in: uf_motion.h

Overview
This function sets the input joint parameters to the input joint tag.

Environment

Internal and External

See Also

UF_MOTION_ask_joint
UF_MOTION_set_joint_limits
UF_MOTION_set_joint_motion_input

History

Created in NX2

```
int UF_MOTION_set_joint
(
    const tag_t joint_tag,
    const UF_MOTION_joint_t * joint_struct
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to set. This tag is the one returned from the UF_MOTION_create_joint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_joint_t *	**joint_struct**	Input	The joint structure for the input tag.

UF_MOTION_set_joint_coupler (view source)

Defined in: uf_motion.h

Overview

This function sets the input joint coupler parameters to the input joint coupler tag.

Environment

Internal and External

See Also

UF_MOTION_ask_joint_coupler
UF_MOTION_create_joint_coupler

History

Created in NX2

```
int UF_MOTION_set_joint_coupler
(
    const tag_t joint_coupler_tag,
    UF_MOTION_joint_coupler_t * coupler_struct
)
```

const tag_t	**joint_coupler_tag**	Input	The tag of the joint coupler to set. This tag is the one returned from the UF_MOTION_create_joint_coupler function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)

UF_MOTION_joint_coupler_t *	**coupler_struct**	Input	The joint coupler structure holding the parameters to be set to the input tag.

UF_MOTION_set_joint_limits (view source)

Defined in: uf_motion.h

Overview

This function sets the joint limits for the input joint tag. By default, joint limits are undefined for all joints. Joint limits can only be set for revolute and slider joints. If the input tag is another type of joint, an error condition is returned.

Environment

Internal and External

See Also

UF_MOTION_set_joint
UF_MOTION_ask_joint_limits
UF_MOTION_remove_joint_limits

History

Created in NX2

```
int UF_MOTION_set_joint_limits
(
    const tag_t joint_tag,
    const UF_MOTION_joint_limits_t * joint_limits_struct
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to set the joint limits on. This joint must be a revolute or a slider. This tag is the one returned from the UF_MOTION_create_joint function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_joint_limits_t *	**joint_limits_struct**	Input	The joint limits structure for the input tag. The maximum joint limit must be greater than zero and the minimum must be less than zero.

UF_MOTION_set_joint_motion_input (view source)

Defined in: uf_motion.h

Overview
This function sets the joint motion input to the input joint tag. By default all joints are created without joint motion input defined. Joint motion input can only be defined for revolute and slider joints. If another type of joint tag is used, an error condition will be returned.

Environment
Internal and External

See Also
UF_MOTION_ask_joint
UF_MOTION_ask_joint_motion_input
UF_MOTION_remove_joint_motion_input

History
Created in NX2

```
int UF_MOTION_set_joint_motion_input
(
    const tag_t joint_tag,
    const UF_MOTION_joint_motion_input_t * input_struct
)
```

const tag_t	**joint_tag**	Input	The tag of the joint to set the motion input info to. This tag must belong to either a revolute or a slider joint. This tag is the one returned from the UF_MOTION_create_joint function. It can also be found using object cycle functions (eg. UF_OBJ_cycle_objs_by_type).
const UF_MOTION_joint_motion_input_t *	**input_struct**	Input	The joint motion input structure to be set on the input joint tag.

UF_MOTION_set_link (view source)

Defined in: uf_motion.h

Overview
This function sets the input link parameters to the input link tag.

Environment
Internal and External

See Also
UF_MOTION_ask_link
UF_MOTION_set_link_mass_properties
UF_MOTION_set_link_initial_velocity

History
Created in NX2

```
int UF_MOTION_set_link
(
    const tag_t link_tag,
    const UF_MOTION_link_t * link_struct
)
```

const tag_t	**link_tag**	Input	The tag of the link to set. This tag is the one returned from the UF_MOTION_create_link function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_link_t *	**link_struct**	Input	The link structure for the input tag.

UF_MOTION_set_link_initial_velocity (view source)

Defined in: uf_motion.h

Overview

This function sets the link initial velocities for the input link tag.
All velocities are based on the work coordinate system of the model.
Do not define more velocities than what is necessary to define the system.
There should never be more initial velocities than degrees of freedom.

Environment

Internal and External

See Also

UF_MOTION_set_link
UF_MOTION_ask_link_initial_velocity
UF_MOTION_remove_link_initial_velocity

History

Created in NX2

```
int UF_MOTION_set_link_initial_velocity
(
    const tag_t link_tag,
    const UF_MOTION_link_initial_velocity_t * init_vel_struct
)
```

const tag_t	**link_tag**	Input	The tag of the link to set the initial velocity. This tag is the one returned from the UF_MOTION_create_link function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_link_initial_velocity_t *	**init_vel_struct**	Output	Data structure containing the initial velocities to be defined.

UF_MOTION_set_link_mass_properties (view source)

Defined in: uf_motion.h

Overview

This function sets the link mass properties to the input link tag. If the link contains solid bodies, default mass properties are calculated. If there are no solid bodies in the link, then this function must be used to set valid mass properties before any dynamic analyses can be performed. Any user defined mass properties passed through this function will overwrite the default or existing mass properties.

Environment

Internal and External

See Also

UF_MOTION_set_link
UF_MOTION_ask_link_mass_properties
UF_MOTION_remove_link_mass_properties

History

Created in NX2

```
int UF_MOTION_set_link_mass_properties
(
    const tag_t link_tag,
    const UF_MOTION_link_mass_properties_t * mass_prop_struct
)
```

const tag_t	**link_tag**	Input	The tag of the link to set mass properties. This tag is the one returned from the UF_MOTION_create_link function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_link_mass_properties_t *	**mass_prop_struct**	Input	The link mass propertiy structure for the input tag. These mass properties will overwrite the mass properties that may have been created automatically for the link.

UF_MOTION_set_link_transform (view source)

Defined in: uf_motion.h

Overview
This function will set transformation matrix for given link object

Environment
Internal and External

History
Released in NX801

```
int UF_MOTION_set_link_transform
(
    tag_t linkTag,
    double transformMatrix [ 16 ]
)
```

tag_t	**linkTag**	Input	Motion link object tag
double	**transformMatrix [16]**	Input	The transform matrix 1X16

UF_MOTION_set_marker (view source)

Defined in: uf_motion.h

Overview
This function sets a new marker origin and a new orientation matrix to the input marker tag.

Environment
Internal and External

See Also
UF_MOTION_ask_marker
UF_MOTION_set_marker

History
Created in NX2

```
int UF_MOTION_set_marker
(
    const tag_t marker_tag,
    UF_MOTION_marker_t * marker_struct
)
```

const tag_t	**marker_tag**	Input	The tag of the marker to set. This tag is the one returned from UF_MOTION_create_marker. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_marker_t *	**marker_struct**	Input	The marker structure for the input tag.

UF_MOTION_set_name_display (view source)

Defined in: uf_motion.h

Overview
This function sets the dislay of object names

Environment
Internal and External

See Also
UF_MOTION_ask_name_display

History
Released in NX3

```
int UF_MOTION_set_name_display
(
    logical name_display
)
```

logical	**name_display**	Input	TRUE if names should be displayed. FALSE if names should not be displayed.

UF_MOTION_set_point_on_surface_constraint (view source)

Defined in: uf_motion.h

Overview
This function sets the structures represented by the tag_t with new point on surface constraint values.

Return
Zero for success, non-zero is error code

Environment
Internal and External

See Also
UF_MOTION_init_point_on_surface_constraint
UF_MOTION_create_point_on_surface_constraint
UF_MOTION_ask_point_on_surface_constraint

History
Created in NX3

```
int UF_MOTION_set_point_on_surface_constraint
(
    tag_t point_on_surface_tag,
    UF_MOTION_point_on_surface_data_p_t pt_on_surf_data
)
```

tag_t	**point_on_surface_tag**	Input	tag of point on surface constraint object
UF_MOTION_point_on_surface_data_p_t	**pt_on_surf_data**	Input	Structures of point on surface constraint data

---

UF_MOTION_set_pt_crv_constraint (view source)

Defined in: uf_motion.h

Overview
This function edit point curve constraint

Environment
Internal and External

See Also
UF_MOTION_ask_pt_crv_constraint

History
Created in NX2

```
int UF_MOTION_set_pt_crv_constraint
(
    tag_t pt_crv_tag,
    UF_MOTION_point_curve_constraint_t * pt_crv_data
)
```

tag_t	**pt_crv_tag**	Input	tag of the constraint
UF_MOTION_point_curve_constraint_t *	**pt_crv_data**	Input	point curve constraint data

---

UF_MOTION_set_scalar_force (view source)

Defined in: uf_motion.h

Overview
This function sets the input scalar force parameters to the input force tag. The parameters that can be set are the origin locations and the force function.

Environment
Internal and External

See Also
UF_MOTION_ask_scalar_force

History
Created in NX2

```
int UF_MOTION_set_scalar_force
(
    const tag_t force_tag,
    const UF_MOTION_scalar_force_torque_t * force_struct
)
```

const tag_t	**force_tag**	Input	The tag of the scalar force to set. This tag is the one returned from UF_MOTION_create_scalar_force. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_scalar_force_torque_t *	**force_struct**	Input	The scalar force structure for the input tag.

UF_MOTION_set_scalar_torque (view source)

Defined in: uf_motion.h

Overview

This function sets the input scalar torque parameters to the input torque tag.
The only parameter that can be set is the torque function.

Environment

Internal and External

See Also

UF_MOTION_ask_scalar_torque

History

Created in NX2

```
int UF_MOTION_set_scalar_torque
(
    const tag_t torque_tag,
    const UF_MOTION_scalar_force_torque_t * torque_struct
)
```

const tag_t	**torque_tag**	Input	The tag of the scalar torque to set. This tag is the one returned from UF_MOTION_create_scalar_torque. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_scalar_force_torque_t *	**torque_struct**	Input	The scalar torque structure for the input tag.

UF_MOTION_set_spring (view source)

Defined in: uf_motion.h

Overview
This function sets the input spring parameters to the input spring tag.
The spring parameters that can be set are the rate, the free length (linear)
or free angle (rotational), and the link attachment points.

Environment
Internal and External

See Also
UF_MOTION_ask_spring

History
Created in NX2

int UF_MOTION_set_spring
(
const tag_t spring_tag,
const UF_MOTION_spring_damper_t * spring_struct
)

const tag_t	**spring_tag**	Input	The tag of the spring to set. This tag is the one returned from the UF_MOTION_create_spring function. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
const UF_MOTION_spring_damper_t *	**spring_struct**	Input	The spring structure for the input tag.

UF_MOTION_set_trace_explosion_to_master (view source)

Defined in: uf_motion.h

Overview
This function sets whether trace/explosion objects will get exported to the
master

Environment
Internal and External

See Also
UF_MOTION_ask_trace_explosion_to_master

History
Released in NX3

int UF_MOTION_set_trace_explosion_to_master
(

```
    logical to_master
)
```

logical	**to_master**	Input	TRUE or FALSE

UF_MOTION_set_vector_force_torque (view source)

Defined in: uf_motion.h

Overview
This function sets the input vector force/torque parameters to the input vector force or torque tag.

Environment
Internal and External

See Also
UF_MOTION_ask_vector_force_torque

History
Created in NX2

```
int UF_MOTION_set_vector_force_torque
(
    const tag_t vobject_tag,
    UF_MOTION_vector_force_torque_t * vector_struct
)
```

const tag_t	**vobject_tag**	Input	The tag of the vector force or torque to set. This tag is the one returned from UF_MOTION_create_vector_force_torque. It can also be found using object cycle functions. (eg. UF_OBJ_cycle_objs_by_type)
UF_MOTION_vector_force_torque_t *	**vector_struct**	Input	The vector force/ torque structure for the input tag.

UF_MOTION_solve_model (view source)

Defined in: uf_motion.h

Overview
This function performs the analysis of the motion model. Prior to calling this function the model must be completely ready to analyze. This function is a time-based analysis. If the model has one or more degrees of freedom, this function performs a dynamic analysis. If the model has no degrees of freedom (including redundant constraints) this function performs a kinematic analysis. The results of the analysis are stored in the internal database until another analysis is performed or the model is changed. To access the results, use any of the following:

1) Packaging options, including measure, trace, and interference.
2) Write the results to disk using UF_MOTION_export_adams_res_file.
3) Get specific results from a joint or marker.

Environment
Internal and External

See Also
UF_MOTION_step_articulation
UF_MOTION_measure
UF_MOTION_review_adams_res_file
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History
Released in V18

int UF_MOTION_solve_model
(
const double time,
int * num_steps
)

const double	**time**	Input	The total time for the analysis in the current time units.
int *	**num_steps**	Input / Output	The number of steps for the solver to take during the analysis. This parameter is used with the time parameter to determine the average step size for the analysis. The total number of steps successfully saved in the database is returned.

UF_MOTION_spreadsheet_run_from_file (view source)

Defined in: uf_motion_ugopenint.h

Overview
This function runs animation using the motion data in the specified spreadsheet. Last argument decides whether the animation will be run interactively by bringing up the user interface or not.

Environment
Internal and External

See Also
UF_MOTION_spreadsheet_run_from_file

History
Released in NX3

int UF_MOTION_spreadsheet_run_from_file
(
const char * spreadsheet_file,
int start_step,

```
    int end_step,
    logical invoke_ui
)
```

const char *	**spreadsheet_file**	Input	Full path and name of spreadsheet file containing motion data
int	**start_step**	Input	Step number in spreadsheet to start animation at
int	**end_step**	Input	Step number in spreadsheet till which to animate
logical	**invoke_ui**	Input	TRUE if spreadsheet run should be interactive FALSE if animation should be run without the user interface

---

UF_MOTION_step_articulation (view source)

Defined in: uf_motion.h

Overview

This function solves the motion model for the number of steps of the
articulation model. It is a position based analysis. Multiple calls to this
function will be appended together to form the full analysis. For example, if
this function is called successfully for seven times with a single step each time,
the analysis results in the database will have seven steps.
The exception case is when solver has locked-up, the number of total steps is not
equal to the number of calls.
The difference is it will not append results onto the end of a time-based simulation.
Instead, the first call to this function following a time-based simulation
or a static analysis will clear the database of those results.
Similar to UF_MOTION_solve_model, the database results can be accessed by the following:

1) Packaging options, including measure, trace, and interference.
2) Write the results to disk using UF_MOTION_export_adams_res_file.
3) Get specific results from a joint or marker.

Environment

Internal and External

See Also

UF_MOTION_solve_model
UF_MOTION_edit_artic_step_size
UF_MOTION_ask_velocity_results
UF_MOTION_ask_force_results

History

Released in V18

```
int UF_MOTION_step_articulation
(
    const int num_steps,
    int * total_steps
)
```

const int	**num_steps**	Input	The number of steps to take for the step sizes currently stored on the articulation joints.
int *	**total_steps**	Output	The number of successful steps that are available in the database.

UF_MOTION_terminate (view source)

Defined in: uf_motion.h

Overview
This function closes all internal databases and solvers for Scenario for Motion and returns the memory to the system.

Environment
Internal and External

See Also
UF_MOTION_is_initialized
UF_MOTION_initialize

History
Released in V18

```
int UF_MOTION_terminate
(
    void
)
```

UF_MOTION_terminate_articulation (view source)

Defined in: uf_motion.h

Overview
This function terminates the solver for an articulation run. It must be called after all calls to UF_MOTION_step_articulation and it must be paired with a call to UF_MOTION_init_articulation. Failure to terminate the solver after an articulation run may corrupt the solver.

Environment
Internal and External

See Also
UF_MOTION_init_articulation
UF_MOTION_step_articulation

History
Created in V18

```
int UF_MOTION_terminate_articulation
(
    void
)
```

UF_MOTION_trace (view source)

Defined in: uf_motion.h

Overview

This function performs a trace of a specific object. The object is copied at its position in the analysis step and the tag of the copied object is returned. The trace parameters must first be stored in the model using UF_MOTION_create_trace or UF_MOTION_edit_trace. Before this function can be called, valid analysis results must be present in the database.

Environment

Internal and External

See Also

UF_MOTION_create_trace
UF_MOTION_solve_model
UF_MOTION_measure
UF_MOTION_interference

History

Released in V18

```
int UF_MOTION_trace
(
    const tag_t trace_tag,
    const int step_number,
    tag_t * new_object_tag
)
```

const tag_t	**trace_tag**	Input	The tag of the trace object. This tag was created using UF_MOTION_create_trace and contains all of the parameters needed to perform the trace.
const int	**step_number**	Input	The step number of the analysis results that are currently available in the database. It cannot be negative and it cannot be greater than the number of steps in the results.
tag_t *	**new_object_tag**	Output	The tag of the new object copied from the traced object.

UF_MOTION_trace_model (view source)

Defined in: uf_motion.h

Overview

This function is used to trace all of the links in the model at the input solution step. This function cannot be called until after there is a valid solution in the database. To create a solution, use UF_MOTION_solve_model, UF_MOTION_step_articulation, or UF_MOTION_calculate_static_equilibrium. This function will copy all of the geometry of each link in the model and return the new geometry tags back.

Environment

Internal and External

See Also

UF_MOTION_trace
UF_MOTION_solve_model
UF_MOTION_step_articulation
UF_MOTION_calculate_static_equilibrium

History

Released in V18

```
int UF_MOTION_trace_model
(
    const int step_num,
    const int target_layer,
    int * num_tags,
    tag_t * * geom_tags
)
```

const int	**step_num**	Input	The step number of the current analysis solution in the database. It cannot be negative and it cannot exceed the number of valid steps in the solution.
const int	**target_layer**	Input	The layer to place the new objects on. It must be a valid NX layer.
int *	**num_tags**	Output	The number of geometry tags copied and returned in the next argument.
tag_t * *	**geom_tags**	Output to UF_*free*	The tags of the copied geometry of all the links. The memory for this array of tags must be freed using UF_free().

UF_MOTION_validate_function_syntax (view source)

Defined in: uf_motion.h

Overview

This function checks the syntax of a function definition string to verify its validity.

Environment

Internal and External

See Also

UF_MOTION_get_object_derived_function
UF_MOTION_create_function

History
Released in NX2.0.5 and NX3.0.1

```
int UF_MOTION_validate_function_syntax
(
    char * * function_string,
    const int num_lines
)
```

char * *	**function_string**	Input	Full string for a function definition.
const int	**num_lines**	Input	Number of lines of function string.

UF_MOTION_write_object_info (view source)

Defined in: uf_motion.h

Overview
This function gets the information of motion objects and writes it out to a file.

Environment
Internal and External

See Also
UF_MOTION_list_connections

History
Created in NX3

```
int UF_MOTION_write_object_info
(
    int num_objects,
    tag_p_t object_tags,
    const char * info_file_name
)
```

int	**num_objects**	Input	The number of motion objects in the tag array.
tag_p_t	**object_tags**	Input	Tag array of motion objects to get info for.
const char *	**info_file_name**	Input	Name string (including full path) of output file that object information will be written to. This file should be new and not yet exist.
