Python Structures

class `isaacgym.gymapi.Version`

Defines a major and minor version

property `major`

property `minor`

class `isaacgym.gymapi.Vec3`

`cross`(*self: Vec3*, *arg0: Vec3*)→ Vec3

`dot`(*self: Vec3*, *arg0: Vec3*)→ float

`dtype`= *dtype([('x', '<f4'), ('y', '<f4'), ('z', '<f4')])*

static `from_buffer`(*arg0: buffer*)→ object

`length`(*self: Vec3*)→ float

`length_sq`(*self: Vec3*)→ float

`normalize`(*self: Vec3*)→ Vec3

property `x`

property `y`

property `z`

class `isaacgym.gymapi.Quat`

Quaternion representation in Gym

`dtype`= *dtype([('x', '<f4'), ('y', '<f4'), ('z', '<f4'), ('w', '<f4')])*

static `from_axis_angle`(*arg0: Vec3*, *arg1: float*)→ Quat *static* **`from_buffer`**(*arg0: buffer*)→ object

static **`from_euler_zyx`**(*arg0: float*, *arg1: float*, *arg2: float*)→ Quat

`inverse`(*self: Quat*)→ Quat

`normalize`(*self: Quat*)→ Quat

`rotate`(*self: Quat*, *arg0: Vec3*)→ Vec3

`to_euler_zyx`(*self: Quat*)→ Tuple[float, float, float]

property **`w`**

property **`x`**

property **`y`**

property **`z`**

class `isaacgym.gymapi.Transform`

Represents a transform in the system

`dtype`= *dtype([('p', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')]), ('r', [('x', '<f4'), ('y', '<f4'), ('z', '<f4'), ('w', '<f4')])])*

static **`from_buffer`**(*arg0: buffer*)→ object

`inverse`(*self: Transform*)→ Transform

Returns:: the inverse of this transform.

Return type:: `isaacgym.gymapi.Transform`

property **`p`**

Position, in meters

property **`r`**

Rotation Quaternion, represented in the format $x\hat{i} + y\hat{j} + z\hat{k} + w$

`transform_point`(*self: Transform*, *arg0: Vec3*)→ Vec3

Rotates point by transform quatertnion and adds transform offset

Parameters:: param1 (`isaacgym.gymapi.Vec3`) – Point to transform.

Returns:: The transformed point.

Return type:: `isaacgym.gymapi.Vec3`

`transform_points`(*self: Transform*, *arg0: numpy.ndarray[Vec3]*)→ numpy.ndarray[Vec3]

Rotates points by transform quatertnion and adds transform offset

Parameters:: param1 (`numpy.ndarray` of `isaacgym.gymapi.Vec3`) – Points to transform.

Returns:: The transformed points.

Return type:: numpy.ndarray[`isaacgym.gymapi.Vec3`]

`transform_vector`(*self: Transform*, *arg0: Vec3*)→ Vec3

Rotates vector by transform quatertnion

Parameters:: param1 (`isaacgym.gymapi.Vec3`) – Vector to transform.

Returns:: The transformed vector.

Return type:: `isaacgym.gymapi.Vec3`

`transform_vectors`(*self: Transform*, *arg0: numpy.ndarray[Vec3]*)→ numpy.ndarray[Vec3]

Rotates vectors by transform quatertnion

Parameters:: param1 (`numpy.ndarray` of `isaacgym.gymapi.Vec3`) – Vectors to transform.

Returns:: The transformed vectors.

Return type:: numpy.ndarray[`isaacgym.gymapi.Vec3`]

class `isaacgym.gymapi.RigidBodyState`

Containing states to get/set for a rigid body in the simulation

`dtype`
= dtype([('pose', [('p', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')]), ('r', [('x', '<f4'), ('y', '<f4'), ('z', '<f4'), ('w', '<f4')])]), ('vel', [('linear', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')]), ('angular', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')])])])

property **`pose`**

Transform with position and orientation of rigid body

property **`vel`**

Set of angular and linear velocities of rigid body

class `isaacgym.gymapi.RigidBodyProperties`

Set of properties used for rigid bodies.

property **`com`**

center of mass in body space

property **`flags`**

Flags to enable certain behaivors on Rigid Bodies simulation. See `isaacgym.gymapi.BodyFlags`

property **`inertia`**

Inertia tensor relative to the center of mass.

property **`invInertia`**

Inverse of Inertia tensor.

property **`invMass`**

Inverse of mass value.

property **`mass`**

mass value, in kg

class `isaacgym.gymapi.DofState`

States of a Degree of Freedom in the Asset architecture

`dtype`= *dtype([('pos', '<f4'), ('vel', '<f4')])*

property **pos**

DOF position, in radians if it’s a revolute DOF, or meters, if it’s a prismatic DOF

property **vel**

DOF velocity, in radians/s if it’s a revolute DOF, or m/s, if it’s a prismatic DOF

class `isaacgym.gymapi.DofFrame`

Frame of a Degree of Freedom

property **axis**

direction for the DOF action

dtype= *dtype([('origin', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')]), ('axis', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')])])*

static **from_buffer**(*arg0: buffer*)→ object

property **origin**

position in environment for the DOF

class `isaacgym.gymapi.Velocity`

Holds linear and angular velocities, in m/s and $radians/s$

property **angular**

angular velocity component

dtype= *dtype([('linear', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')]), ('angular', [('x', '<f4'), ('y', '<f4'), ('z', '<f4')])])*

static **from_buffer**(*arg0: buffer*)→ object

property **linear**

Linear velocity component

class **isaacgym.gymapi.Mat33**

3x3 Matrix used for inetia tensor

property **x**

property **y**

property **z**

class **isaacgym.gymapi.Mat44**

4x4 Matrix

property **w**

property **x**

property **y**

property **z**

class **isaacgym.gymapi.IndexRange**

Used for passing start and end indexes of a vector when setting or getting data of a slice of the vector.

property **count**

property **start**

class **isaacgym.gymapi.PlaneParams**

Parameters for global ground plane

property **distance**

Ground plane distance from origin

property **dynamic_friction**

Coefficient of dynamic friction

property **normal**

Ground plane normal coefficient

property **restitution**

Coefficient of restitution

property **segmentation_id**

SegmentationID value for segmentation ground truth

property **static_friction**

Coefficient of static friction

class **isaacgym.gymapi.AttractorProperties**

The Attractor is used to pull a rigid body towards a pose. Each pose axis can be individually selected.

property **`axes`**

Axes to set the attractor, using GymTransformAxesFlags. Multiple axes can be selected using bitwise combination of each axis flag. if axis flag is set to zero, the attractor will be disabled and won’t impact in solver computational complexity.

property **`damping`**

Damping to be used on attraction solver.

property **`offset`**

Offset from rigid body origin to set the attractor pose.

property **`rigid_handle`**

Handle to the rigid body to set the attractor to

property **`stiffness`**

Stiffness to be used on attraction for solver. Stiffness value should be larger than the largest agent kinematic chain stifness

property **`target`**

Target pose to attract to.

class `isaacgym.gymapi.RigidShapeProperties`

Set of properties used for all rigid shapes.

property **`compliance`**

Coefficient of compliance. Determines how compliant the shape is. The smaller the value, the stronger the material will hold its shape. Value should be greater or equal to zero.

property **`contact_offset`**

Distance at which contacts are generated (used with PhysX only *property* **`filter`**

Collision filter bitmask - shapes A and B only collide if (filterA & filterB) == 0.

property **`friction`**

Coefficient of static friction. Value should be equal or greater than zero.

property **`rest_offset`**

How far objects should come to rest from the surface of this body (used with PhysX only

property **`restitution`**

Coefficient of restitution. It's the ratio of the final to initial velocity after the rigid body collides. Range [0,1]

property **`rolling_friction`**

Coefficient of rolling friction.

property **`thickness`**

How far objects should come to rest from the surface of this body (used with Flex only)

property **`torsion_friction`**

Coefficient of torsion friction.

class **`isaacgym.gymapi.ForceSensorProperties`**

Set of properties used for force sensors.

property **`enable_constraint_solver_forces`**

Enable to receive forces from constraint solver (default = True).

property **`enable_forward_dynamics_forces`**

Enable to receive forces from forward dynamics (default = True).

property **`use_world_frame`**

Enable to receive forces in the world rotation frame, otherwise they will be reported in the sensor's local frame (default = False).

class `isaacgym.gymapi.SoftMaterial`

Soft Material definition

property **`activation`**

Current fiber activation.

property **`activationMax`**

Maximum activation value.

property **`damping`**

Material damping.

property **`model`**

Model type, See `isaacgym.gymapi.SoftMaterialType`

property **`poissons`**

Poisson Ration.

property **`youngs`**

Young Modulus.

class `isaacgym.gymapi.Tensor`

Internal wrapper class of tensors.

property **`data_address`**

address of data

property **`data_ptr`**

pointer to buffer

property **`device`**

property **`dtype`**

data type

property **`ndim`**

number of dimensions

property **`own_data`**

flag for ownership

property **`shape`**

tensor shape

class `isaacgym.gymapi.RigidContact`

Rigid Bodies contact information. Each contact in simulation generates a set of information.

property **`body0`**

Colliding rigid body indexes in the environment, -1 if it is ground plane

property **`body1`**

Colliding rigid body indexes in the environment, -1 if it is ground plane

property **`env0`**

Environment contact body0 belongs to, -1 if it is shared/unrecognized env

property **`env1`**

Environment contact body1 belongs to, -1 if it is shared/unrecognized env

property **`friction`**

Effective coefficient of Friction between bodies pair

property **`initial_overlap`**

Amount of overlap along normal direction at the start of the time-step

property **`lambda`**

Contact force magnitude

property **`lambda_friction`**

Friction forces magnitudes. The direction of the friction force is the projection on the normal plane of the relative velocity of the bodies.

property **`local_pos0`**

Local space position of the body0 contact feature excluding thickness, normal forces applied here *property* **`local_pos1`**

Local space position of the body1 contact feature excluding thickness, normal forces applied here

property **`min_dist`**

Minimum distance to try and maintain along the contact normal between the two points

property **`normal`**

Contact normal from body0->body1 in world space

property **`offset0`**

The local space offset from the feature localPos0 to the surface. That's the location where friction will be applied

property **`offset1`**

The local space offset from the feature localPos1 to the surface. That's the location where friction will be applied

property **`rolling_friction`**

Effective coeffitienc of Rolling Friction between bodies pair

property **`torsion_friction`**

Effective coefficient of Torsional friction between bodies pair

class **`isaacgym.gymapi.ActionEvent`**

property **`action`**

property **`value`**

class **`isaacgym.gymapi.FlexParams`**

Simulation parameters used for FleX physics engine *property* **`contact_regularization`**

Distance for soft bodies to maintain against ground planes

property **`deterministic_mode`**

Flag to activate deterministic simulation. Flex Newton solver only

property **`dynamic_friction`**

Coefficient of friction used when colliding against shapes

property **`friction_mode`**

Type of friction mode:

> **0 single friction dir, non-linear cone projection, but can't change direction during linear solve**

- - 1 two friction dirs, non-linear cone projection, can change direction during linear solve
  - 2 same as above plus torsional (spinning) friction

property **`geometric_stiffness`**

Improves stability of joints by approximating the system Hessian

property **`max_rigid_contacts`**

Max number of rigid body contacts

property **`max_soft_contacts`**

Max number of soft body contacts

property **`num_inner_iterations`**

Number of inner loop iterations taken by the solver per simulation step. Is used only by Newton solver.

property **`num_outer_iterations`**

Number of iterations taken by the solver per simulation step.

property **`particle_friction`**

Coefficient of friction used when colliding particles

property **`relaxation`**

Control the convergence rate of the parallel solver. Values greater than 1 may lead to instability.

property **`return_contacts`**

Read contact information back to CPU

property **`shape_collision_distance`**

Distance for soft bodies to maintain against rigid bodies and ground plane

property **`shape_collision_margin`**

Distance for rigid bodies at which contacts are generated

property **`solver_type`**

Type of solver used:

- 0 = XPBD (GPU)
- 1 = Newton Jacobi (GPU)
- 2 = Newton LDLT (CPU)
- 3 = Newton PCG (CPU)
- 4 = Newton PCG (GPU)
- 5 = Newton PCR (GPU)
- 6 = Newton Gauss Seidel (CPU)
- 7 = Newton NNCG (GPU)

property **`static_friction`**

Coefficient of static friction used when colliding against shapes

property **`warm_start`**

Fraction of the cached Lagrange Multiplier to be used on the next simulation step.

class **`isaacgym.gymapi.PhysXParams`**

Simulation parameters used for PhysX physics engine

property **`always_use_articulations`**

If set, even single-body actors will be created as articulations

property **`bounce_threshold_velocity`**

A contact with a relative velocity below this will not bounce. A typical value for simulation stability is about 2*gravity*dt/num_substeps.

property **`contact_collection`**

Contact collection mode

property **`contact_offset`**

Shapes whose distance is less than the sum of their contactOffset values will generate contacts.

property **`default_buffer_size_multiplier`**

Default buffer size multiplier

property **`friction_correlation_distance`**

Friction correlation distance

property **`friction_offset_threshold`**

Friction offset threshold

property **`max_depenetration_velocity`**

The maximum velocity permitted to be introduced by the solver to correct for penetrations in contacts.

property **`max_gpu_contact_pairs`**

Maximum number of contact pairs

property **`num_position_iterations`**

PhysX solver position iterations count. Range [1,255]

property **`num_subscenes`**

Number of subscenes for multithreaded simulation

property **`num_threads`**

Number of CPU threads used by PhysX. Should be set before the simulation is created. Setting this to 0 will run the simulation on the thread that calls PxScene::simulate(). A value greater than 0 will spawn numCores-1 worker threads.

property **`num_velocity_iterations`**

PhysX solver velocity iterations count. Range [1,255]

property **`rest_offset`**

Two shapes will come to rest at a distance equal to the sum of their restOffset values.

property **`solver_type`**

Type of solver used.

- 0 : PGS (Iterative sequential impulse solver
- 1 : TGS (Non-linear iterative solver, more robust but slightly more expensive

property **`use_gpu`**

Use PhysX GPU. Disabled at the moment.

class **`isaacgym.gymapi.SimParams`**

Gym Simulation Parameters

property **`enable_actor_creation_warning`**

property **`dt`**

Simulation step size

property **`flex`**

Flex specific simulation parameters (See `isaacgym.gymapi.FlexParams`)

property **`gravity`**

3-Dimension vector representing gravity force in Newtons.

property **`num_client_threads`**

property **`physx`**

PhysX specific simulation parameters (See `isaacgym.gymapi.PhysXParams`)

property **`stress_visualization`**

property **`stress_visualization_max`**

property **`stress_visualization_min`**

property **`substeps`**

Number of subSteps for simulation

property **`use_gpu_pipeline`**

property **`up_axis`**

Up axis

class **`isaacgym.gymapi.AssetOptions`**

Defines a set of properties for assets imported into Gym.

property **`angular_damping`**

Angular velocity damping for rigid bodies

property **`armature`**

The value added to the diagonal elements of inertia tensors for all of the asset's rigid bodies/links. Could improve simulation stability

property **`collapse_fixed_joints`**

Merge links that are connected by fixed joints.

property **`convex_decomposition_from_submeshes`**

Whether to treat submeshes in the mesh as the convex decomposition of the mesh. Default False.

property **`default_dof_drive_mode`**

Default mode used to actuate Asset joints. See `isaacgym.gymapi.DriveModeFlags`.

property **`density`**

Default density parameter used for calculating mass and inertia tensor when no mass and inertia data are provided, in kg/m^3.

property **`disable_gravity`**

Disables gravity for asset.

property **`enable_gyroscopic_forces`**

Enable gyroscopic forces (PhysX only).

property **`fix_base_link`**

Set Asset base to a fixed placement upon import.

property **`flip_visual_attachments`**

Switch Meshes from Z-up left-handed system to Y-up Right-handed coordinate system.

property **`linear_damping`**

Linear velocity damping for rigid bodies.

property **`max_angular_velocity`**

Maximum angular velocity for rigid bodies. In rad/s.

property **`max_linear_velocity`**

Maximum linear velocity for rigid bodies. In m/s.

property **`mesh_normal_mode`**

How to load normals for the meshes in the asset. One of FROM_ASSET, COMPUTE_PER_VERTEX, or COMPUTE_PER_FACE. Defaults to FROM_ASSET, falls back to COMPUTE_PER_VERTEX if normals not fully specified in mesh.

property **`min_particle_mass`**

Minimum mass for particles in soft bodies, in Kg

property **`override_com`**

Whether to compute the center of mass from geometry and override values given in the original asset.

property **`override_inertia`**

Whether to compute the inertia tensor from geometry and override values given in the original asset.

property **`replace_cylinder_with_capsule`**

flag to replace Cylinders with capsules for additional performance.

property **`slices_per_cylinder`**

Number of faces on generated cylinder mesh, excluding top and bottom.

property `tendon_limit_stiffness`

Default tendon limit stiffness. Choose small as the limits are not implicitly solved. Avoid oscillations by setting an apporpriate damping value.

property `thickness`

Thickness of the collision shapes. Sets how far objects should come to rest from the surface of this body

property `use_mesh_materials`

Whether to use materials loaded from mesh files instead of the materials defined in asset file. Default False.

property `use_physx_armature`

Use joint space armature instead of links inertia tensor modififcations.

property `vhacd_enabled`

Whether convex decomposition is enabled. Used only with PhysX. Default False.

property `vhacd_params`

Convex decomposition parameters. Used only with PhysX. If not specified, all triangle meshes will be approximated using a single convex hull.

class `isaacgym.gymapi.CameraProperties`

Properties for a camera in Gym

property `enable_tensors`

CUDA interop buffers will be available only if this is true.

property `far_plane`

distance in world coordinates to far-clipping plane

property `height`

Height of output images in pixels

property **`horizontal_fov`**

Horizontal field of view in degrees. Vertical field of view is calculated from height to width ratio

property **`near_plane`**

distance in world coordinate units to near-clipping plane

property **`supersampling_horizontal`**

oversampling factor in the horiziontal/X direction

property **`supersampling_vertical`**

oversampling factor in the vertical/Y direction

property **`use_collision_geometry`**

If true, camera renders collision meshes instead of visual meshes

property **`width`**

Width of output images in pixels

class **`isaacgym.gymapi.PerformanceTimers`**

Amount of time in seconds spent doing the respective activity since last query

property **`frame_idling`**

idling to keep updates close to graphics framerate

property **`graphics_image_retrieval`**

Copying images from the GPU to CPU

property **`graphics_sensor_rendering`**

Rendering image sensors

property **`graphics_viewer_rendering`**

Rendering the viewer

property **`physics_data_movement`**

Copying physics state to/from the GPU

property **`physics_sim`**

Running physics simulation

property **`total_time`**

sum of all other timers

class **`isaacgym.gymapi.VhacdParams`**

VHACD Convex Decomposition parameters

property **`alpha`**

Controls the bias toward clipping along symmetry planes. 0.0-1.0. Default 0.05.

property **`beta`**

Controls the bias toward clipping along revolution axes. 0.0-1.0. Default 0.05.

property **`concavity`**

Maximum concavity. 0.0-1.0. Default 0.0.

property **`convex_hull_approximation`**

Default True.

property **`convex_hull_downsampling`**

Controls the precision of the convex-hull generation process during the clipping plane selection stage. 1-16. Default 4.

property **`max_convex_hulls`**

Maximum number of convex hulls. Default 64.

property **`max_num_vertices_per_ch`**

Controls the maximum number of vertices per convex-hull. 4-1024. Default 64.

property **`min_volume_per_ch`**

Controls the adaptive sampling of the generated convex-hulls. 0.0-0.01. Default 0.0001.

property **`mode`**

tetrahedron-based approximate convex decomposition. Default 0.

Type:: 0

Type:: voxel-based approximate convex decomposition, 1

property **`ocl_acceleration`**

Default True.

property **`pca`**

Enable/disable normalizing the mesh before applying the convex decomposition. 0-1. Default 0.

property **`plane_downsampling`**

Controls the granularity of the search for the “best” clipping plane. 1-16. Default 4.

property **`project_hull_vertices`**

Default True.

property **`resolution`**

Maximum number of voxels generated during the voxelization stage. 10,000-64,000,000. Default 100,000.

class **`isaacgym.gymapi.HeightFieldParams`**

The heightfield origin is at its center (height = 0), and it is oriented to be perpendicular to the the Gym up-axis.

property **`column_scale`**

Spacing of samples [m] in column dimension

property **`dynamic_friction`**

Coefficient of dynamic friction

property **`nbColumns`**

-Y)

Type:: Number of samples in column dimension (Y-up

Type:: Z, Z-up

property **`nbRows`**

Number of samples in row dimension (X)

property **`restitution`**

Coefficient of restitution

property **`row_scale`**

Spacing of samples [m] in row dimension

property **`segmentation_id`**

SegmentationID value for segmentation ground truth

property **`static_friction`**

Coefficient of static friction

property **`transform`**

Transform to apply to heightfield

property **`vertical_scale`**

Vertical scaling [m] to apply to integer height samples

class **`isaacgym.gymapi.TriangleMeshParams`**

Triangle Mesh properties

property **`dynamic_friction`**

Coefficient of dynamic friction

property **`nb_triangles`**

Number of triangles

property **`nb_vertices`**

Number of vertices

property **`restitution`**

Coefficient of restitution

property **`segmentation_id`**

SegmentationID value for segmentation ground truth

property **`static_friction`**

Coefficient of static friction

property **`transform`**

Transform to apply to heightfield