IPC Toolkit

IPC Group

Oct 15, 2021

GENERAL

A set of reusable functions to integrate IPC into an existing simulation.

- **Free Software**: MIT License
- **Github Repository**: https://github.com/ipc-sim/ipc-toolkit
- **Paper**: https://ipc-sim.github.io/file/IPC-paper-fullRes.pdf

CHAPTER

ONE

CHANGELOG

All notable changes to this project will be documented in this file.

1.1 2021-07-26 (1479aae)

1.1.1 Changed

- Updated the CMake system to use modern `FetchContent` to download externals

1.2 2021-07-22 (e24c76d)

1.2.1 Fixed

- Updated CCD strategy when using Tight Inclusion to only perform `no_zero_toi=true` when there is no minimum distance

1.3 2021-07-17 (a20f7a2)

1.3.1 Added

- Added `detect_edge_face_collision_candidates_brute_force` for 3D intersection broad-phase
- Added ability to save an obj of collision candidates
- Added tests for has_intersection (all pass after fixes)

1.3.2 Fixed

- Fixed possible numerical rounding problems in HashGrid `AABB::are_overlapping`
- Fixed HashGrid's function for getting edge-face intersection candidates

1.4 2021-07-15 (7301b42)

1.4.1 Fixed

- Use `ignore_codimensional_vertices` in the brute force broad-phase method
- Fixed AABB inflation in brute force and SpatialHash methods

1.5 2021-07-08 (86ae4e5)

1.5.1 Changed

- Replaced vertex group ids with more powerful can_collide function. By default everything can collide with everything (same as before)
- Reordered parameters in `construct_constraint_set()`, `is_collision_free()`, and `compute_collision_free_stepsize()`
- `update_barrier_stiffness` now requires the `constraint_set` rather than building it
- `update_barrier_stiffness` dropped dhat parameter

1.5.2 Fixed

- SpatialHash for 2D

1.5.3 Removed

- Verison of `initial_barrier_stiffness` that computes the constraint set and barrier gradient because there are a lot of parameters to these functions

1.6 2021-07-05 (4d16954)

1.6.1 Changed

- Renamed directory `src/spatial_hash/` → `src/broad_phase/`
- Renamed files `src/ccd/broad_phase.*` → `src/ccd/aabb.*`

1.7 2021-07-05 (b3808e1)

1.7.1 Added

- Select the broad-phase method for CCD and distance constraints
  - Methods: `HASH_GRID`, `SPATIAL_HASH`, `BRUTE_FORCE`
- CCD parameters for Tight Inclusion's tolerance and maximum iterations

1.7.2 Changed

- `ignore_codimensional_vertices` to `false` by default
- CMake option `TIGHT_INCLUSION_WITH_NO_ZERO_TOI=ON` as default

1.8 2021-06-18 (aa59aeb)

1.8.1 Changed

- `construct_friction_constraint_set` now clears the given `friction_constraint_set`

1.9 2021-05-18 (245b13b)

1.9.1 Changed

- Use TightInclusion degenerate edge-edge for point-point and point-edge CCD

1.10 2021-05-11 (5c34dcd)

1.10.1 Changed

- `char*` exceptions to `std::exceptions`

1.11 2021-05-06 (24056cc)

1.11.1 Changed

- Gave `dhat_epsilon_scale` a default value of `1e-9` in `update_barrier_stiffness`
- **warning** Changed order of parameters to `update_barrier_stiffness`
  - Flipped `bbox_diagonal` and `dhat_epsilon_scale`

1.12 2021-05-06 (81d65f3)

1.12.1 Fixed

- Bug in output min distance of `update_barrier_stiffness`

1.13 2021-05-04 (59ec167)

1.13.1 Changed

- Moved eigen_ext functions into ipc namespace
- Renamed max size matrices with `Max`
  - `Eigen::VectorX([0-9])` → `ipc::VectorMax$1`
  - `Eigen::MatrixXX([0-9])` → `ipc::VectorMax$1`
  - `Eigen::ArrayMax([0-9])` → `ipc::ArrayMax$1`

1.14 2021-05-03 (664d65f)

1.14.1 Added

- Added utility function to check for edge-edge intersection in 2D and edge-triangle intersection in 3D.
- Optionally: use GMP for exact edge-triangle intersection checks

1.15 2021-05-03 (9b4ebfc)

1.15.1 Added

- voxel_size_heuristic.cpp which suggests a good voxel size for the SpatialHash and HashGrid

1.15.2 Changed

- Changed HashGrid voxel size to be the average edge length not considering displacement length. This results in better performance, but can result in large memory usage.

1.16 2021-04-29 (293d0ad)

1.16.1 Added

- Added TBB parallel loops to the main function (`compute_potential`, `compute_friction_potential`, `compute_collision_free_stepsize`, etc.)
- Added function `addVerticesFromEdges` that adds the vertices connected to edges in parallel and avoids duplicates

1.16.2 Changed

- Changed the HashGrid to use `ArrayMax3` over `VectorX3` to simplify the code

1.16.3 Fixed

- Fixed some parameters that were not by reference

1.17 2021-04-21 (c8a6d5)

1.17.1 Added

- Added the SpatialHash from the original IPC code base with some modification to get all candidates in parallel
  - Benchmark results indicate this SpatialHash is faster than the HashGrid with multithreading
  - TODO: Improve HashGrid or fully integrate SpatialHash into ipc.hpp

1.18 2021-02-11 (9c7493)

1.18.1 Changed

- Switched to the correct (conservative) CCD of [Wang and Ferguson et al. 2020]
  - Can select Etienne Vouga's CCD in the CMake (see README.md)

1.19 2021-02-01 (b510253)

1.19.1 Added

- Added minimum seperation distance (thickness) to distance constraints
  - Based on Codimensional Incremental Potential Contact [Li et al. 2020]

1.20 2021-02-01 (a395175)

1.20.1 Added

- Added 2D friction model based on the 3D formulation.
  - TODO: Test this further

1.21 2021-01-12 (deee6d0)

1.21.1 Added

- Added and optional parameter `F2E` to `construct_constraint_set()`. This is similar to `F` (which maps faces to vertices), but maps faces to edges. This is optional, but recommended for better performance. If not provided a simple linear search will be done per face edge!
  - TODO: Add a function to compute this mapping.

1.22 2021-01-09 (deee6d0)

1.22.1 Changed

- Replaced VectorXd and MatrixXd with static size versions for local gradient and hessians

1.23 2020-11-20 (93143ad)

1.23.1 Changed

- Removed TBB parallelization form the hash grid because we get better performance without it.
  - TODO: Improve parallelization in the hash grid or switch to the original IPC spatial hash

1.24 2020-11-06 (4553509)

1.24.1 Fixed

- Fixed multiplicity for point-triangle distance computation to avoid duplicate point-point and point-edge pairs.

1.25 2020-10-22 (51f4903)

1.25.1 Fixed

- Projection of the hessian to PSD. This was completely broken as the projected matrix was never used.

1.26 2020-10-22 (9be6c0f)

1.26.1 Fixed

- Mollification of EE constraints that have a distance type of PP or PE
- If there is no mollification needed then the PP and PE constraints are stored with multiplicity
- Set the parallel EE friction constraint threshold to eps_x like in IPC
  - This avoid needing the mollification for the normal force and these forces are small anyways

1.27 2020-10-10 (cb8b53f)

1.27.1 Fixed

- Assertions in `compute_collision_free_stepsize`

1.28 2020-10-10 (4a5f84f)

1.28.1 Fixed

- Point-triangle distance type by replacing it with the one used in the original IPC code

1.29 2020-10-10 (1d51a61)

1.29.1 Added

- Boolean parameter in `compute_friction_potential_hessian` that controls if the hessian is projected to PSD

1.30 2020-10-09 (b737fb0)

1.30.1 Added

- Parameter for vertex group IDs to exclude some collisions (e.g., self collisions)

1.31 2020-10-08 (6ee60ae)

1.31.1 Added

- Second version of `update_barrier_stiffness()` that takes an already computed minimum distance and world bounding box diagonal

1.32 2020-10-08 (cc3947d)

1.32.1 Added

- Second version of `initial_barrier_stiffness()` that takes an already computed barrier gradient
- Assertions on `initial_barrier_stiffness()` input
  - `average_mass > 0 && min_barrier_stiffness_scale > 0`

1.32.2 Changed

- Fixed typo in `initial_barrier_stiffness()` name (was `intial_barrier_stiffness()`)

1.33 2020-10-07 (5582582)

1.33.1 Added

- `FrictionConstraint` structures to store friction information (i.e., tangent basis, normal force magnitude, closest points, and coefficient of friction)
- Unit test that compares the original IPC code's friction components with the toolkit's

1.33.2 Changed

- `compute_friction_bases()` is now `construct_friction_constraint_set()`
  - It now takes the coefficient of friction (`mu`)
  - It now puts all information inside of the `FrictionConstraints` (`friction_constraint_set`)

1.34 2020-10-06 (b48ba0e)

1.34.1 Changed

- During `construct_constraint_set()` the constraints are added based on distance type
  - Duplicate vertex-vertex and edge-vertex constraints are handled by a multiplicity multiplier
  - Edge-edge constraints are always line-line distances
  - Point-triangle constraints are always point-plane distances

1.35 2020-10-05 (9a4576b)

1.35.1 Fixed

- Fixed a bug in the point-triangle closest points and tangent basis computed in `compute_friction_bases()`
- Fixed a bug in `edge_edge_tangent_basis()` used to compute the tangent basis for friction

1.36 2020-09-19 (31a37e0)

1.36.1 Added

- spdlog for logging information

1.37 2020-09-19 (acb7664)

1.37.1 Changed

- Headers are now include with the prefix `ipc/`
  - E.g., `#include <ipc.hpp>` → `#include <ipc/ipc.hpp>`

1.38 2020-09-04 (7dd2ab7)

1.38.1 Added

- Collision constraint to store distance constraint pairs
  - `EdgeEdgeConstraint` stores the edge-edge mollifier threshold (`eps_x`)

1.38.2 Changed

- Input parameter `dhat_squared` is now `dhat` (i.e., non-squared value)
- Input parameter `epsv_times_h_squared` is now `epsv_times_h` (i.e., non-squared value)
- `Constraints` replaced `Candidates`
- `construct_constraint_set()` now takes the rest vertex position (`V_rest`)
- `compute_barrier_potential*()` no longer take the rest vertex position

CHAPTER

TWO

CONTRIBUTING

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

2.1 Types of Contributions

2.1.1 Report Bugs

Report bugs at https://github.com/ipc-sim/ipc-toolkit/issues.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

2.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" and "help wanted" is open to whoever wants to implement it.

2.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "enhancement" and "help wanted" is open to whoever wants to implement it. Those that are tagged with "first-timers-only" is suitable for those getting started in open-source software.

2.1.4 Write Documentation

IPC Toolkit could always use more documentation, whether as part of the official docs, in code comments, or even on the web in blog posts, articles, and such.

2.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/ipc-sim/ipc-toolkit/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

CHAPTER

THREE

API

3.1 Barrier Function

ipc::barrier()

ipc::barrier_gradient()

ipc::barrier_hessian()

ipc::point_edge_distance_type()

CHAPTER

FOUR

DEVELOPMENT STATUS

- contacts
- friction

Warning: This toolkit is in an early stage of development and consequently the API may change to better improve usability. If you have any problems or find any bugs please post and issue. For a complete list of changes, please see changelog. Meanwhile, for a definitive reference for these functions, please see the IPC source code.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search