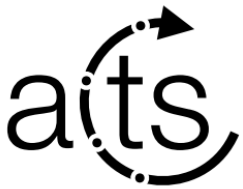
Detray CUDA Navigator

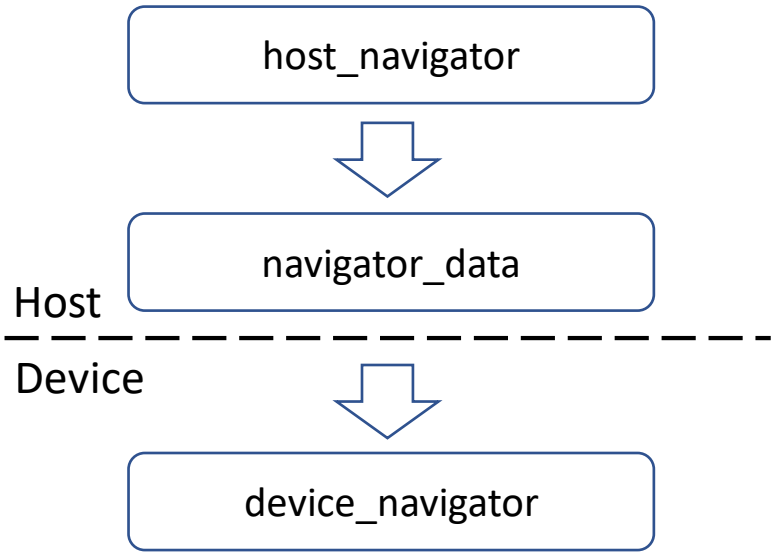
Beomki Yeo



Navigator

```
template <typename detector_t>
class navigator {

    // Host constructor
    navigator (detector_t& det): _det(det){}

    // Device constructor
    template <typename navigator_data_t>
    __device__ navigator(navigator_data_t &n_data)
        : _det(n_data._det_data) {}

    // detector object
    detector_t _det;
};

template <typename navigator_t>
struct navigator_data {

    using detector_type =
       typename navigator_t::detector_type;

    navigator_data(navigator_t &n)
        : _det_data(get_data(n.get_detector())) {}

    detector_data<detector_type> _det_data;
};
```

- Host/device trait of navigator is determined by the type of input detector
- Host navigator can be sent to GPU device via navigator_data
- Device navigator can be constructed in cuda kernel with navigator_data object



Linear propagation in CUDA kernel

```
__global__ void navigator_test_kernel(
    navigator_view<navigator_host_t> n_data,
    vecmem::data::vector_view<track<nav_context>> tracks_data,
    vecmem::data::jagged_vector_view<intersection> candidates_data){

    int gid = threadIdx.x + blockIdx.x * blockDim.x;

    // Get device navigator
    navigator_device_t n(n_data);

    // Get navigator state and surface candidate vector for a thread
    vecmem::device_vector<track<nav_context>> tracks(tracks_data);
    vecmem::jagged_device_vector<intersection> candidates(candidates_data);
    auto& traj = tracks.at(gid);
    navigator_device_t::state state(candidates.at(gid));

    // Set initial volume
    state.set_volume(0u);

    // Start propagation
    bool heartbeat = n.status(state, traj);

    while (heartbeat) {
        heartbeat = n.target(state, traj);

       // Linear stepping
        traj.pos = traj.pos + state() * traj.dir;

        heartbeat = n.status(state, traj);
    }
}
```

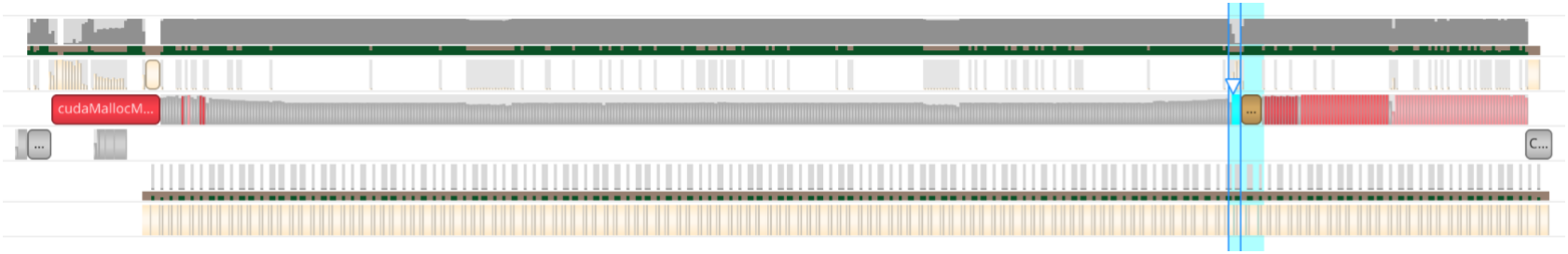
Unit Test Results

- The unit test cross-checks the list of volumes and intersection points passed by CPU and CUDA navigation
  - Unit test with double: OK
  - Unit test with float passed after disabling fast-math mode in CUDA

- Time estimation for 10000 tracks

	float [sec]	double [sec]
cpu	1.74	1.84
cuda	0.26	1.07

- But don't take the numbers seriously
  - No local navigation with surface grid
  - Bunch of memory transfer/allocation for list of volume and intersection points



Next steps

- Enable the local navigation with surface grid
- Implement Runge-Kutta stepper for CPU and CUDA