BESIII event navigation using EventNavigator

EventNavigator is a tool for fast and convenient navigation between MC truth objects and reconstruction objects. Current implementation supports MDC information only. Provided the response of reconstruction experts of other subsystems, and user requests, its functionality can be easily expanded. EventNavigator provides following methods to access relations between objects in collections McParticleCol (MC truth particle objects), MdcMcHitCol (MonteCarlo MDC hits) , MdcRecHitCol (reconstructed MDC hits), MdcTrackCol (reconstructed MDC tracks):

vector<MdcTrack*>& getMdcTracks (McParticle* mcparticle)
returns STL vector of pointers to reconstructed tracks which contain hits , produced by given simulated particle. Size of this vector typically is 1 (one MC particle produced only one reconstructed track). If size of this vector is 0, it means, that track produced by the MC particle, was not reconstructed. If size of the vector greater than 1, it means that either reconstructed track is split into pieces, or reconstruction program mixed several tracks.

vector<McParticle*>& getMcParticles (MdcTrack* reconstructed_track)
returns vector of pointers to MC particles, which contribute hits to given reconstructed track.

vector<MdcRecHit*>& getMdcRecHits (McParticle* mcparticle)
returns vector of pointers to reconstructed hits, produced by given MC particle.

vector<MdcMcHit*>& getMdcMcHits (MdcTrack* reconstructed_track)
returns vector of pointers to MDC Monte-Carlo hits, which are associated with given reconstructed track.

vector<MdcRecHit*>& getMdcRecHits (MdcMcHit* montecarlo_hit)
returns vector of pointers to reconstructed MDC hits, which correspond to digit, produced by given MDC Monte-Carlo hit. In normal life, this vector must have size 0 or 1.

vector<MdcMcHit*>& getMdcMcHits (MdcRecHit* reconstructed_hit)
returns vector of pointers to MDC Monte-Carlo hits, which contributed to digit, corresponding to the given reconstructed MDC hit.

Additionally, EventNavigator offers fast access methods to get pointer to an object using its unique ID:
const McParticle* getMcParticle (int id)
const MdcTrack* getMdcTrack (int id)

const MdcMcHit* getMdcMcHit (int id)
const MdcRecHit* getMdcRecHit (int id)
const MdcDigi* getMdcDigi (int id)

For MdcDigi and MdcMcHit result of Identifier.get_value() method should be used as ID

Important note: ***getMcParticles*** will give you links to all particles, which contribute given reconstructed track. Some of them produced only 1 hit included into given reconstructed track. This situation, for example, happens, when x-y projection of 2 simulated tracks crosses. If you want to know how many hits produce each of simulated particle for given reconstructed track, use ***getMdcMcHits (MdcTrack*)*** and then check mother McParticle for each of retrieved hits. The same is valid for ***getMdcTrack*** method.

How does it work?

For each event BesNavigatorInit algorithm rolls over hits , calculates all relations and stores them into lookup maps. Finally EventNavigator object is registered in the TDS. So, when user needs some relations, EventNavigator just takes it from the maps.
It is fast operation, and does not affect overall performance of the user's program.

How to use EventNavigator

1. Insert the line

use EventNavigator EventNavigator-* Event

into your *requirements* file.

2. Type ***make,*** to copy algorithm-specific jobOptions.txt file to working directory
3. Add lines

ApplicationMgr.DLLs += { "EventNavigator" };
ApplicationMgr.TopAlg += { "BesNavigatorInit" };

OR

#include "EventNavigator.txt"

into your j*obOptions.txt* file. BesNavigatorInit algorithm should run after MC truth and reconstruction information is placed in the TDS. After BesNavigatorInit algorithm have been executed, all relations are stored in the TDS, and can be analyzed by your algorithms.

4. To get access to EventNavigator object from your algorithm::execute(), insert the following line into your code:

SmartDataPtr<EventNavigator> navigator (eventSvc(),"/Event/Navigator");

After that, you can use it to retrieve relations. For example, to roll over all MdcTracks and get for each track information about its parent McParticle(s), just use:

```
for(MdcTrackCol::iterator it = mdcTrackCol->begin(); it != mdcTrackCol->end(); it++)
{
        vector<McParticle>& mcParticles = navigator->getMcParticles(*it);
        //. . .
        // user's code to analyze MC truth inform
        //. . .
}
```

Example of event navigation

Complete fully-functional example of use EventNavigator facility can be found in *BesExamples/NavigationEx* package. In this example a simple analysis of reconstructed MDC information is done using EventNavigator, and several histograms are filled and stored in ROOT file. These histograms include distribution of difference of reconstructed and true momentum for the same tracks, distribution of number of reconstructed tracks corresponding to one MC particle etc. To use this example, first include EventNavigator facility into your application, as described above, and then

1. Insert the line

use NavigationEx NavigationEx-* BesExamples

into your *requirements* file.
2. Type ***make,*** to copy algorithm-specific jobOptions.txt file to working directory
3. Add line

#include "NavigationExOptions.txt"

into your j*obOptions.txt* file. The NavigationEx algorithm should produce file with default name histo.root (this file name can be changed in NavigationExOptions.txt), which contains resulting histograms.

If you have any questions/comments/corrections,
please contact Alexey Zhemchugov (zhemchugov@jinr.ru)