
Subject: Re: Crash in reco-macro, PndSttMvdTracking.cxx produces segfault
Posted by [André Zambanini](#) on Tue, 24 Jul 2012 13:24:46 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hello Gianluigi,

Gianluigi Boca wrote on Tue, 24 July 2012 15:16 To be certain, first of all please show me the exact Macro you are using, thanks

I'm using the macro delivered with the current SVN revision of the july12 release (rev 16248).

But for the sake of completeness the contents of the run_reco_sttcombi.C:

Show file content{

//

=====

=

// Verbosity level (0=quiet, 1=event level, 2=track level, 3=debug)

Int_t iVerbose = 0;

// Input file

TString inDigiFile = "digi_sttcombi.root";

TString inSimFile = "points_sttcombi.root";

// Parameter file

TString parFile = "params_sttcombi.root";

// Output file

TString outFile = "reco_sttcombi.root";

// Number of events to process

Int_t nEvents = 0;

// ---- Load libraries -----

gROOT->LoadMacro("\$VMCWORKDIR/gconfig/rootlogon.C");

rootlogon();

TString sysFile = gSystem->Getenv("VMCWORKDIR");

// -----

// In general, the following parts need not be touched

//

=====

=

// ----- Timer -----

TStopwatch timer;

timer.Start();

// -----

// ----- Digitization run -----

FairRunAna *fRun= new FairRunAna();

fRun->SetInputFile(inDigiFile);

fRun->AddFriend(inSimFile);

fRun->SetOutputFile(outFile);

```

FairGeane *Geane = new FairGeane();
fRun->AddTask(Geane);
// -----

// ----- Parameter database -----
TString allDigiFile = sysFile+"/macro/params/all.par";

FairRuntimeDb* rtdb = fRun->GetRuntimeDb();
FairParRootFileIo* parInput1 = new FairParRootFileIo();
parInput1->open(parFile.Data());

FairParAsciiFileIo* parIo1 = new FairParAsciiFileIo();
parIo1->open(allDigiFile.Data(),"in");

rtdb->setFirstInput(parInput1);
rtdb->setSecondInput(parIo1);
// -----

PndMvdRiemannTrackFinderTask* mvdTrackFinder = new
PndMvdRiemannTrackFinderTask();
mvdTrackFinder->SetVerbose(iVerbose);
mvdTrackFinder->SetMaxDist(0.05);
mvdTrackFinder->SetPersistence(kFALSE);
fRun->AddTask(mvdTrackFinder);

// PndSttTrackFinderIdeal* sttTrackFinder = new PndSttTrackFinderIdeal(iVerbose);
PndSttTrackFinderReal* sttTrackFinder = new PndSttTrackFinderReal(0);
PndSttFindTracks* sttFindTracks = new PndSttFindTracks("Track Finder", "FairTask",
sttTrackFinder, iVerbose);
sttFindTracks->AddHitCollectionName("STTHit", "STTPoint");
//sttFindTracks->SetPersistence(kFALSE);
fRun->AddTask(sttFindTracks);

PndSttMvdTracking * SttMvdTracking = new PndSttMvdTracking(0, false, false);
//SttMvdTracking->Cleanup();
SttMvdTracking->SetPersistence(kFALSE);
fRun->AddTask(SttMvdTracking);

//PndMCTrackAssociator* trackMC0 = new PndMCTrackAssociator();
//trackMC0->SetTrackInBranchName("SttMvdTrack");
//trackMC0->SetTrackOutBranchName("SttMvdTrackID");
//trackMC0->SetPersistence(kFALSE);
//fRun->AddTask(trackMC0);

PndSttMvdGemTracking * SttMvdGemTracking = new PndSttMvdGemTracking(0);
//SttMvdGemTracking->SetPdgFromMC();
fRun->AddTask(SttMvdGemTracking);

PndMCTrackAssociator* trackMC = new PndMCTrackAssociator();
trackMC->SetTrackInBranchName("SttMvdGemTrack");
trackMC->SetTrackOutBranchName("SttMvdGemTrackID");
fRun->AddTask(trackMC);

```

```

PndRecoKalmanTask* recoKalman = new PndRecoKalmanTask();
recoKalman->SetTrackInBranchName("SttMvdGemTrack");
recoKalman->SetTrackInIDBranchName("SttMvdGemTrackID");
recoKalman->SetTrackOutBranchName("SttMvdGemGenTrack");
recoKalman->SetBusyCut(50); // CHECK to be tuned
//recoKalman->SetIdealHyp(kTRUE);
//recoKalman->SetNumIterations(3);
fRun->AddTask(recoKalman);

PndMCTrackAssociator* trackMC2 = new PndMCTrackAssociator();
trackMC2->SetTrackInBranchName("SttMvdGemGenTrack");
trackMC2->SetTrackOutBranchName("SttMvdGemGenTrackID");
fRun->AddTask(trackMC2);

PndFtsTrackerIdeal* trackFts = new PndFtsTrackerIdeal();
trackFts->SetRelativeMomentumSmearing(0.02);
trackFts->SetVertexSmearing(0.02, 0.02, 0.02);
trackFts->SetTrackingEfficiency(1.);
trackFts->SetTrackOutput("FtsIdealTrack");
fRun->AddTask(trackFts);

PndRecoKalmanTask* recoKalmanFwd = new PndRecoKalmanTask();
recoKalmanFwd->SetTrackInBranchName("FtsIdealTrack");
//recoKalmanFwd->SetTrackInIDBranchName("FtsIdealTrackID");
recoKalmanFwd->SetTrackOutBranchName("FtsIdealGenTrack");
recoKalmanFwd->SetBusyCut(50); // CHECK to be tuned
//recoKalmanFwd->SetIdealHyp(kTRUE);
//recoKalmanFwd->SetNumIterations(3);
fRun->AddTask(recoKalmanFwd);

PndMCTrackAssociator* trackMC3 = new PndMCTrackAssociator();
trackMC3->SetTrackInBranchName("FtsIdealGenTrack");
trackMC3->SetTrackOutBranchName("FtsIdealGenTrackID");
fRun->AddTask(trackMC3);

// ----- Initiaise and run -----
PndEmcMapper::Init(1);
fRun->Init();
fRun->Run(0, nEvents);

rtdb->saveOutput();
rtdb->print();

// -----

// ----- Finish -----

timer.Stop();
Double_t rtime = timer.RealTime();
Double_t ctime = timer.CpuTime();
cout << endl << endl;

```

```
cout << "Macro finished succesfully." << endl;
cout << "Output file is " << outFile << endl;
cout << "Parameter file is " << parFile << endl;
cout << "Real time " << rtime << " s, CPU time " << ctime << " s" << endl;
cout << endl;
// -----

}
```
