“Alignment framework” with RCP R/W

D0 workshop, NIU

Dhiman Chakraborty, 06/19/00

What’s there

A system to read alignment constants for each detector element from an ASCII file (RCP), and create the corresponding geometry for the entire detector. It serves two main purposes:

1. Making survey data available to reconstruction software.

2. Simulation of detector misalignments necessary for the development of alignment software.

Software packages (CVS modules)

alignment_system:

(non-D0-specific components of the alignment framework)

GeometryAddress:

A storage class for IP-address-style names (string) for GeometryElements with methods to construct an address through level-by-level concatenation, determine parent-child relationships etc.

GeometryAddressManager:

A singleton registry (one-to-one map) of GeometryElement-GeometryAddress pairs. A GeometryElement has to be registered with this with a “name”. Provides various utilities to traverse the geometry tree using char strings.

BaseAligner:

A virtual base class for individual XAligners to derive from (X=SMT/CFT/CAL/CPS/FPS/MUO). Provides common methods such as moveElement, write, etc. Individual Xaligners are responsible for the implementation of detector-specific methods such as registerElements, align etc.

- d0_alignment:
  - D0Aligner:

    The singleton global aligner responsible for invoking XAligners for individual detectors (only SMT and FPS are currently implemented). Takes one RCP.
  - Test program
    - creates default detector from Geometry RCPs, creates D0Aligner, invokes its align() and write() methods. The (modified) geometry is written out in D0Alignment.rcp

- fps_alignment:
  - FPSAligner:

    Creates GeometryAddress for every FPS GeometryElement, and registers it with the GeometryAddressManager.

    For each element, searches fps_move.rcp for a corresponding GeometryXform. If one is found, it searches fps_reference.rcp for the initial GeometryXform. If one is found, then the result is a sum of the two, else just the first. The results are written out in fps_positions_out.rcp

    Example (a ReferencePoint):

```
double D0_FPSN_layer0 = (      //   element name
        -0.987 -100 399.58     //   translation vector
        0 -1 0                 //   rotation axis
        3.14159)               //   rotation angle
```

- smt_alignment:
  - SMTAligner:

    Creates GeometryAddress for every SMT GeometryElement, and registers it with the GeometryAddressManager.

    For each element, searches smt_move.rcp for a corresponding GeometryXform. If one is found, it searches smt_reference.rcp for the initial GeometryXform. If one is found, then the result is a sum of the two, else just the first. The results are written out in smt_positions_out.rcp

    Example (a leaf-level GeometryElement)

```
double D0_SMT_central_NBH1_barrel_layer3_ladder04 = (
                                          //   element name
 -0.0922813 -4.33372 -7.43925            //   translation vector
 0.222286 -0.788959 0.572828             //   rotation axis
 2.7958)                                  //   rotation angle
```

How to use it?

```
> setup n32
> setup D0RunII test
> newrel -t test test
> cd test
> d0setwa
> addpkg -h d0_alignment
> gmake all
> bin/IRIX6-KCC_3_3/test_aligner
```

See fps_aligner or smt_aligner for an example of X_aligner

More detailed documentation will be available very soon in the doc areas of the respective packages.

Examples:

(fps_reference.rcp)

```
double D0_FPSN_layer0 = (              //   element name
          0 0 -397.35                  //   translation vector
          0 0 1                        //   rotation axis
          0)                           //   rotation angle
double D0_FPSN_layer0_wedge0 = (       //   element name
          0 0 -397.35                  //   translation vector
          0 0 1                        //   rotation axis
          0)                           //   rotation angle
```

(fps_move.rcp)

```
double D0_FPSN_layer0 = (              //   element name
          1.23 0 -3.45                 //   translation vector
          0 0 1                        //   rotation axis
          0)                           //   rotation angle
double D0_FPSN_layer0_wedge0 = (       //   element name
          0 0 0                        //   translation vector
          0 0 1                        //   rotation axis
          0.1)                         //   rotation angle
```

(fps_positions_out.rcp)

```
double D0_FPSN_layer0 = (              //   element name
          1.23 0 -400.8                //   translation vector
          0 0 1                        //   rotation axis
          0)                           //   rotation angle
double D0_FPSN_layer0_wedge0 = (       //   element name
          1.23 0 -400.8                //   translation vector
          0 0 1                        //   rotation axis
          0.1)                         //   rotation angle
```

What's missing

- X_alignment:

  For X=CFT,CAL,CPS,MUO

- Classes for alignment by track-fitting:

  Residual, AlignmentTrack, TrackSelector, …

Needed: Input from users (FPS and SMT ready to use for storing survey data)