TPC Alignment in GEAR

Martin Killenberg

MarlinTPC EVO Meeting

14. April 2010

TPCModule

```
class TPCModule : public PadRowLayout2D
{
  PadRowLayout2D * _padRowLayout; // pad layout with local coordinates

  Vector2D   _offset; // the offset (polar or cartesian, depending on
                     //              global coordinate system)
  double     _angle;  // rotation of pad plane wrt. global coordinate system

  Vector2D   globalToLocal (double c0, double c1);
  Vector2D   localToGlobal (double c0, double c1);

  Vector2D getPadCenter(int padIndex) const // returns the pad centre in
                                            // GLOBAL coordinates
  {
       Vector2D localCenter = _padRowLayout->getPadCenter(padIndex);
       return localToGlobal( localCenter[0],  localCenter[1] );
  }
};
```

- Module is a PadRowLayout2D
- Module has an instance of local pad plane
- Module can be translated of rotated on end plate
- Module always returns global coordinates
- Functions to convert from / to local coordinates of pad plane

Alignment

Current Possibility

- Write alignment corrected offset and angle into the GEAR XML file
  Currently used by the Bonn group for Timepix QuadBoards
  - Cannot be changed at run time

Proposal

Add runtime changeable alignment to TPCModule

- `setAlignment( float xOffset, float yOffset, float angle);`
- `getPadCenter()` returns aligned global coordinates
- Alignment can be performed / applied in a separate processor
- Change is transparent to all existing user code
- New functions `localToUnaligned()` and `globalToUnaligned()` ?
- `AlignmentObject getAlignment();`

The Alignment Object

Proposal:

```
class AlignmentObject
{
  float xOffset, yOffset, zOffset; // the translatory offsets
  float phi, theta, psi; // the rotations around z, x and y axis
  bool alignmentIsLocal; // flag whether alignment is wrt. global coord
                         // or the local subdetector coordinate system
}
```

plus the constructor, `get()` and `set()` functions

- Do we want a function to apply the alignment (plus inverse function)?

  ```
  float3Vec Alignment::applyAlignment(float3Vec unalignedCoordinatesPoint);
  ```

- More evolved scheme: Make it a matrix
  - Alignment can be applied by multiplying with 3D vector
  - Something like this should be available (in gsl?, CLHEP?)