TGCAlign - the TGC Alignment Program

S. Tarem, Z. Terem[1]

Technion, Israel Institute of Technology

Abstract

TGCAlign is a program that reconstructs TGC unit movements based on the TGC Alignment Sensors (TAS) measurements, plus the MDT-based absolute positions. It can also be used in simulation mode to assess the quality of TGC position estimation given a description of the locations of alignment sensors. This helps in finding a good sensor arrangement.

Introduction

The Thin Gap Chamber (TGC) detector will be mounted at both endcaps of the Atlas Muon spectrometer. The middle station of the detector is constructed from three wheels. Each wheel consists of 216 (in wheel 1) or 264 (in wheels 2 and 3) TGC units arranged in concentric layers. The exact geometry of the TGC detector is described elsewhere[2].

Acceptable track reconstruction requires correct information on the position of the TGC units. The positions should be known to within 1-2 cm. The TGC unit positions are expected to change for several reasons: sagging during installation, temperature changes and aging. To measure these changes, the TGC detector includes linear displacement potentiometers which measure relative motion between adjacent TGC units. The voltages across those potentiometers are read out by the TGC Detector Control System (DCS). The Muon Spectrometer alignment system also includes two absolute position measurements per octant. These are part of the end-cap MDT alignment system, and are read by it. This information will be obtained from the online database by the TGC alignment program.

TGCAlign is a program that reconstructs TGC unit movements based on the TGC Alignment Sensors (TAS) measurements, plus the MDT-based absolute positions. It can also be used in simulation mode to assess the quality of TGC position estimation given a description of the locations of alignment sensors. This helps in finding a good sensor arrangement.

The program only considers unit movements in the wheel plane. It does not account for shape distortions and movements perpendicular to the wheel plane. Thus the position of a unit can be described by three parameters: the X-Y location of the center and the rotation of the unit around the center.

Modeling of Sensor Function

[1] On leave from Intel Corp.

[2] ATLAS Muon Spectrometer Technical Design Report CERN/LHCC/97-22 -- ATLAS TDR 10 (ISBN 92-9083-108-1)

The alignment sensors are modeled very simply as two virtual points in space and a direction of measurement. For a sensor between TGC A and TGC B, the two points represent its position on TGC A on which it is mounted, and its position on TGC B. The two points are at identical global coordinates when the TGCs are in their nominal position. The sensor's positions relative to their respective TGC unit remains constant. As TGC A and B move relative to each other, the two points move apart. The expected sensor measurement is the distance between new positions, projected on the measurement direction. Simulated sensor measurements are convoluted with an error of 0.1 mm as found when testing the linear potentiometers we will use.

Setting up

The TGC geometry setup is constructed from AMDB. The user controls the choice of wheel and octants to construct. Alternatively, the program can read all individual TGC unit positions from an external file. This second mode will be used after the TGC is constructed and survey positions are available. Additional input includes constraints on TGC movements coming from the sector and wheel construction. Sensor arrangement is read from one of two file types. The first type, TAS position file (*filename.tas),* contains directives for constructing sensors between adjacent units by specifying a template of the location and the direction of measurement of the sensors. The second file type, TAS list file (*filename.tsl)*, specifies explicitly all individual sensor positions, and is expected to be used after assembly and reflect the actual placement of the sensors.

For visualization purposes, the program can produce a 3D view of the TGC system using the ROOT Geometry facilities. The view shows TGC units in space, with small arrow-like boxes representing sensor position and direction. Figure 1 shows such a drawing.

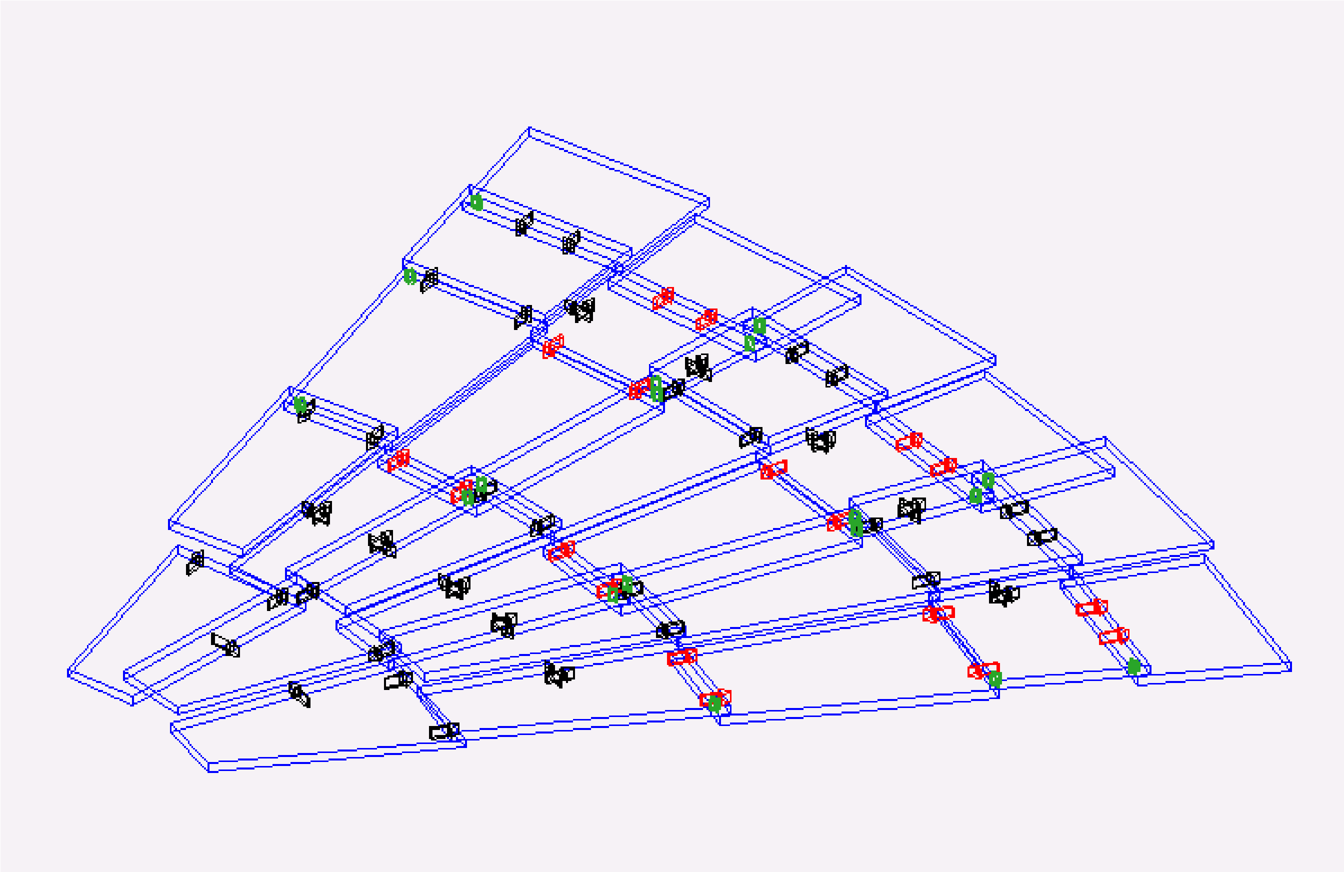


Figure 1: 3D view of Wheel 1, Octant 1. Black arrows represent sensors that are mounted on the inner side of the wheel, red arrows represent sensors that are mounted on the outer side of the wheel and green dots represent rigid mounting points to the frame.

Reconstruction Mode

In reconstruction mode, the program reads sensor measurements from an external file, uses the reconstruction engine to estimate TGC movements and stores the results in a ROOT file for future analysis. The program includes a utility for calculating expected sensor measurements for any given set of TGC unit movements. This calculation is used to find the most likely set of unit movements by minimizing the difference between the calculated and actual sensor readings. The computation is done using Minuit (as implemented in ROOT), and is speeded up by using analytic derivatives.

Simulation Mode

In simulation mode, the TGCAlign program is used to assess the reconstructive capabilities of a sensor arrangement.
In each simulation event:

1. The program generates random TGC movements.
2. The program calculates the resulting sensor measurements.

3. The reconstruction engine estimates the TGC movements from the sensor measurements as in reconstruction mode.
4. The differences between generated and reconstructed movements are stored in a ROOT file for analysis.

The simulator accepts user-controls for the generation of TGC movements and the expected sensor measurement errors, as well as the number of simulation iterations to generate and summarize. Figure 2 and Figure 3 show the distribution of differences between the generated and estimated TGC unit positions for Wheel 1.

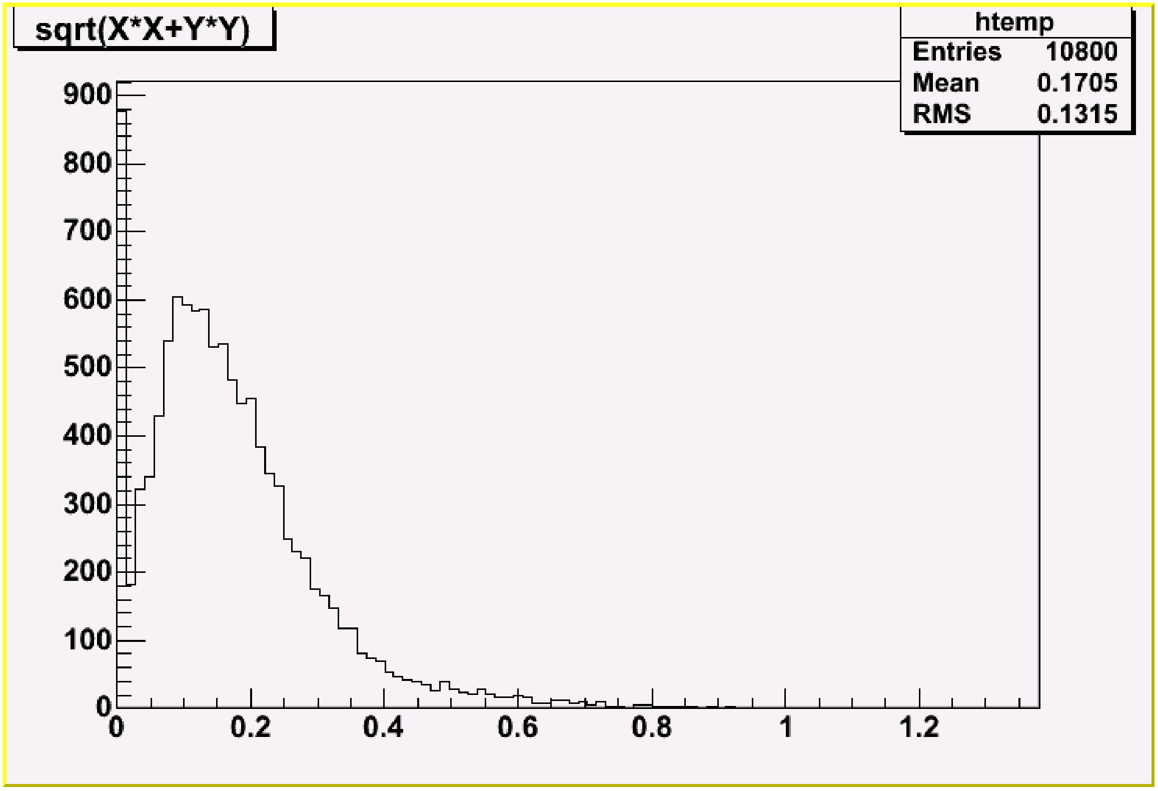


Figure 2: Distribution of the differences between the generated and estimated TGC unit positions for Wheel 1, all octants.

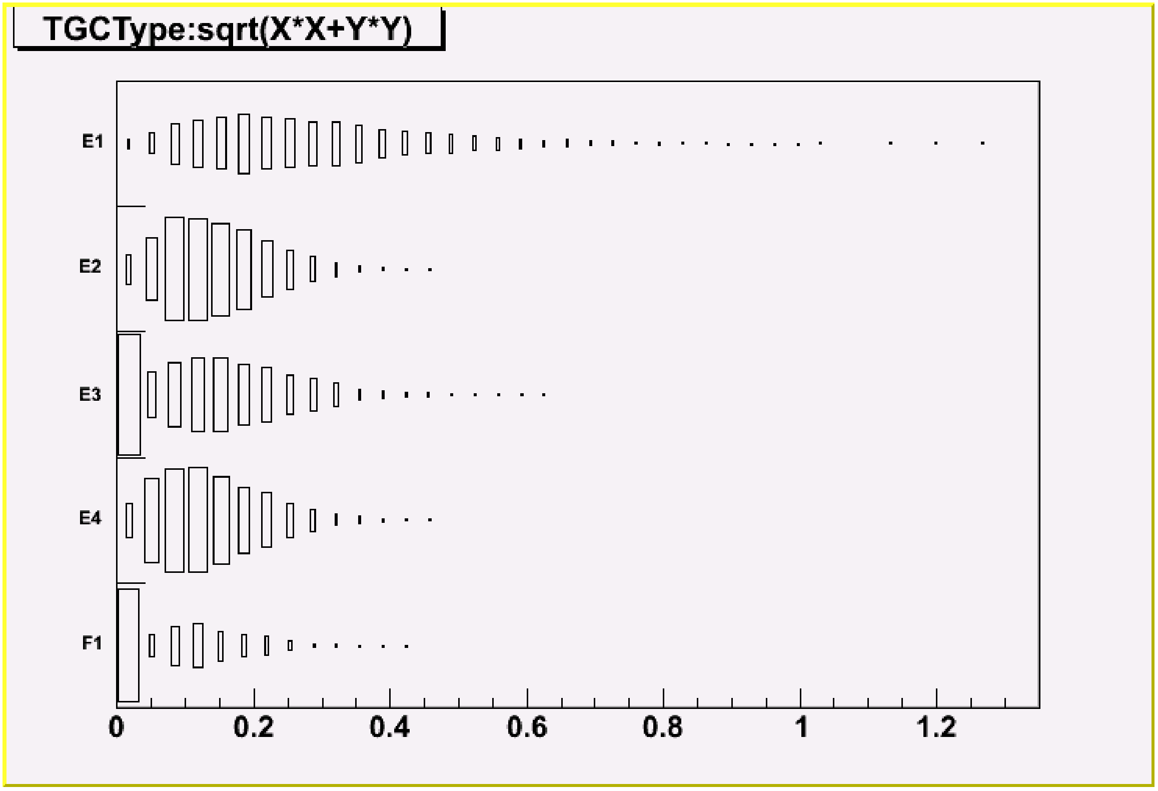


Figure 3: The data of Figure 2 displayed by TGC unit type. The large concentration of zero values for unit types F1 and E3 represents the positions provided by the MDT.

How to run the program

The program is invoked with the following command:

```
Usage: TGCAlignRun [OPTIONS]...

  -h, --help                Print help and exit
  -V, --version             Print version and exit
  -v, --verbose             Log debugging information to run.log  (default=off)
  -d, --draw=INT            Draw option: 0=none, 1=system, 2=system + 5 events
                              (default=`0')
  -s, --setup               Setup only  (default=off)
  -o, --output=file         Name of output ROOT file  (default=`run.root')

Geometry:
  -w, --wheel=INT           Wheel number (1..3)  (default=`1')
  -1, --octant1=INT         Low octant number (1..8)  (default=`1')
  -2, --octant2=INT         High octant number (octant1..8)  (default=`1')
  -p, --taspos=file         Name of TAS Positions file
  -l, --taslist=file        Name of TAS List file
  -t, --outtas              Write TAS List to --taslist file  (default=off)
      --tgclist=file        Name of TGC List file
      --outtgc              Write TGC List to --tgclist file  (default=off)

Simulation options:
  -x, --xydelta=DOUBLE      Perturbation for X/Y (mm)  (default=`5.0')
  -r, --rotdelta=DOUBLE   Perturbation for rotation (mrad)  (default=`2.0')
  -e, --error=DOUBLE      TAS measurement error (mm)  (default=`0.1')
  -u, --use-errors        Use TAS errors in simulation  (default=off)
  -n, --events=INT        Number of simulation events  (default=`1')

Reconstruction options:
  -c, --reconstruct=file  Name of file with TAS measurements

Special Modes:
  -m, --simulate          Simulate and profile TAS measurements  (default=off)

Notes:
  * Input files are searched in $TGCALIGNDIR and in the current directory.
  * If a TAS list file is given, then the file will be written if
    --outtas is given, and will be read otherwise.
  * If a TAS list file is read, the TAS position file is ignored.
  * If a reconstruction file is given, all simulation parameters are ignored.
```

Required inputs:

amdbTGC.txt Contains the geometry and setup of the TGC units

corner.tas A description of the locations of supports that constrain unit movements at certain points in certain directions. The format of the file is the same as the file describing a sensor arrangement.

Other inputs:

TAS positions A description of an arrangement of sensors. Each row is a template for positioning the sensor between two TGC unit types, at a known distance along the overlapping edge, measuring at given direction.

TAS list A file with the absolute position and direction of measurement of all sensors.

TGC list A file with the absolute center position and unit rotation of all TGC units.

TAS measurementsA file with the displacement measurements of all sensors. In the production version of the program these numbers will be provided by the DCS.

Outputs:

ROOT file A root file with a description of the run and data for further analysis. In simulation mode, the file contains the difference between simulated and estimated TGC positions. In reconstruction mode the file contains estimated TGC positions.

Log file If requested, the file contains details of the generation and estimation of all variables. The file can be very large for a significant number of simulation events.

File formats

TAS and corner positions

This format is used to describe a set of measurement points by providing templates for generating the actual sensors or constraints. Each template specifies a pair of TGC unit types, a mask for specifying which pairs within the octant are affected, and a displacement of the sensor or constraint along the common edge. Each row is formatted as follows:

Direction	‘R’ for radial or ‘T’ for tangential direction of measurement
Wheel	The wheel number (1..3)

Type1	The type of the first TGC unit (F1, E1..E4)
Type2	The type of the second TGC unit (F1, E1..E4)
Mask	A string of 3 (for F1) or 6 (for 'E' units) characters that specify which of the units in the octant are affected. The program uses the template to generate only for entries with a '1'
Move	The lateral distance of the point from the center of the common edge, in mm. Can be positive (towards positive φ or R) or negative (towards negative φ or R)

Here are the first few lines of the configuration used for the figures:

```
T 1 F1 F1 111       131.
R 1 F1 E4 101010    111.
R 1 F1 E4 010101   -111.
T 1 E4 E4 111111      0.
R 1 E4 E4 111111      0.
```

TAS list

This format provides the absolute location and direction of measurement of all sensors. Each row is formatted as follows:

Name	The name of the TAS, formatted as "TAS<Wheel><TGC1><TGC2><R|T><#>". For example: TAS1F100F101T1
TGCA	The name of the TGC unit the sensor is mounted on, formatted as "TGC<wheel><Unit Type><#>". For example: TGC1F100
TGCB	The name of the TGC unit adjacent to the one the sensor is mounted on
X	The absolute X position of the sensor, in mm
Y	The absolute Y position of the sensor, in mm
Rot	The absolute direction of measurement of the sensor, in radians in the φ direction

Here are a few lines from a file that describes the same set of sensors:

```
TAS1F120F119T1 TGC1F120 TGC1F119     839.0003   -2984.4502  -2.879796
TAS1F122F123T1 TGC1F122 TGC1F123    2665.6083   -1582.8446   1.047200
TAS1F122F121T1 TGC1F122 TGC1F121    2218.8207   -2165.1095  -2.356197
TAS1F100E400R1 TGC1F100 TGC1E400    3783.7919    -865.8176  -0.196350
TAS1F102E404R1 TGC1F102 TGC1E404    3709.7687    1142.0759   0.327249
```

TGC list

This format provides the absolute location and orientation of the TGC units. Each row is formatted as follows:

Name	The name of the TGC unit formatted as "TGC<wheel><Unit Type><#>". For example: TGC1F100.
R	The radial distance of the unit center, in mm from the center of the wheel
Phi	The angular distance of the unit center, in radians in the φ direction
Z	The forward distance of the unit center from the detector center, in mm
Rot	The rotation of the unit around its center, in radians counter-clockwise

Here a few lines from the file that contains the chosen arrangement:

```
TGC1F121    2938.5500  -0.916298   13485.4950  -2.487094
TGC1F122    2938.5500  -0.654498   13306.4950  -2.225295
TGC1F123    2938.5500  -0.392699   13485.4950  -1.963495
TGC1E400    5010.2500  -0.196350   13385.4950  -1.767146
TGC1E401    5010.2500  -0.065450   13564.4950  -1.636246
TGC1E402    5010.2500   0.065450   13385.4950  -1.505346
```

TAS measurements

This format provides the displacement measurements of the sensors. Each row is formatted as follows:

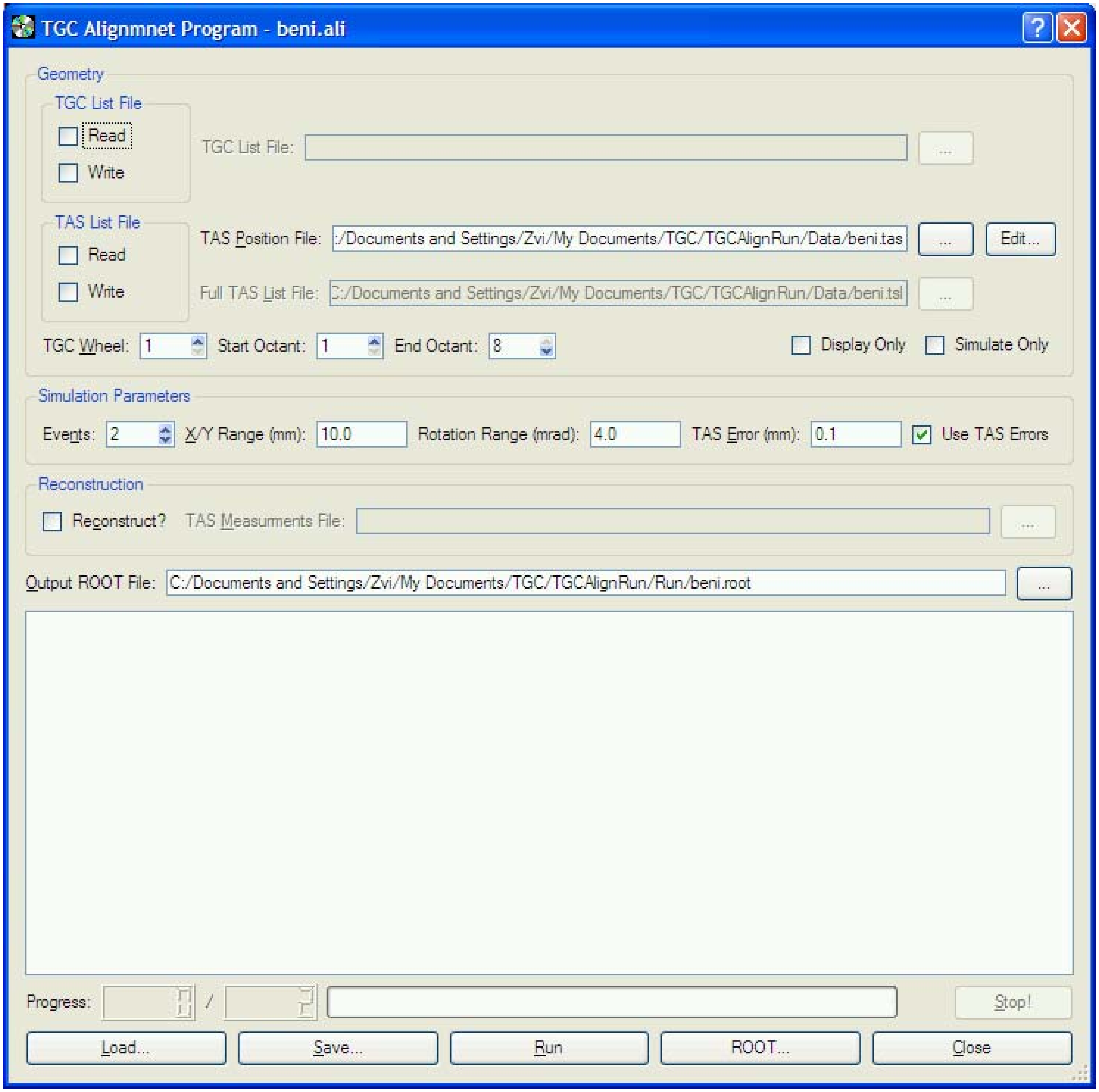
TAS name	The name of the TAS, formatted as "TAS<Wheel><TGC1><TGC2><R|T><#>”. For example: TAS1F100F101T1
Measure	The TAS displacement, in mm

Here are a few lines from a randomly generated measurement list:

```
 TAS1F100F101T1      -3.04226
 TAS1F102F101T1       6.60921
 TAS1F100E400R1        4.0546
 TAS1F102E404R1       1.71815
 TAS1E402F101R1       4.90119
 TAS1F100E401R1       7.24852
 TAS1F101E403R1       5.76343
 TAS1F102E405R1      -1.38096
 TAS1E400E401T1      -5.92003
 TAS1E402E403T1      -5.42764
 TAS1E402E401T1       6.58917
```

Graphical User Interface

A graphical user interface is provided to assist in using TGCAlign. The main panel is shown in **Error! Reference source not found.**. The GUI also enables plotting two kinds of simulation results (shown in this note), and a description of the TAS set, from the output ROOT file. In order to use the GUI, QT™ version 3.2 must be installed on your machine.



TGC Alignmnet Program - beni.ali

Geometry

TGC List File

□ Read

□ Write

TGC List File:

TAS List File

□ Read

□ Write

TAS Position File: :/Documents and Settings/Zvi/My Documents/TGC/TGCAlignRun/Data/beni.tas

Full TAS List File: C:/Documents and Settings/Zvi/My Documents/TGC/TGCAlignRun/Data/beni.tsl

TGC Wheel: 1 Start Octant: 1 End Octant: 8 □ Display Only □ Simulate Only

Simulation Parameters

Events: 2 X/Y Range (mm): 10.0 Rotation Range (mrad): 4.0 TAS Error (mm): 0.1 ☑ Use TAS Errors

Reconstruction

□ Reconstruct? TAS Measuments File:

Output ROOT File: C:/Documents and Settings/Zvi/My Documents/TGC/TGCAlignRun/Run/beni.root

Progress: /

Figure 4: The GUI for running TGCAlign

Acknowledgments

We are grateful to Florian Bauer for his help in confronting the alignment problem.