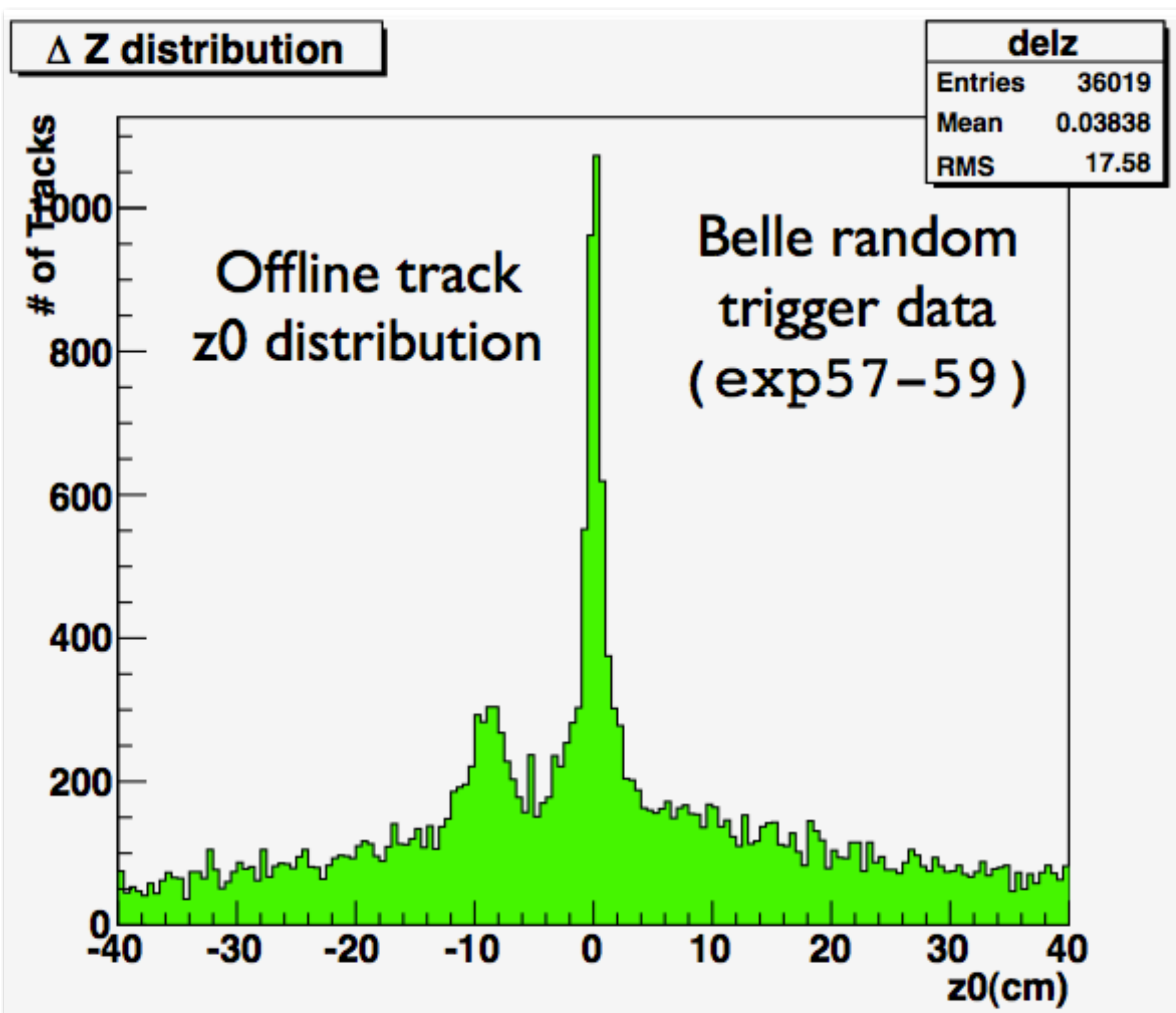
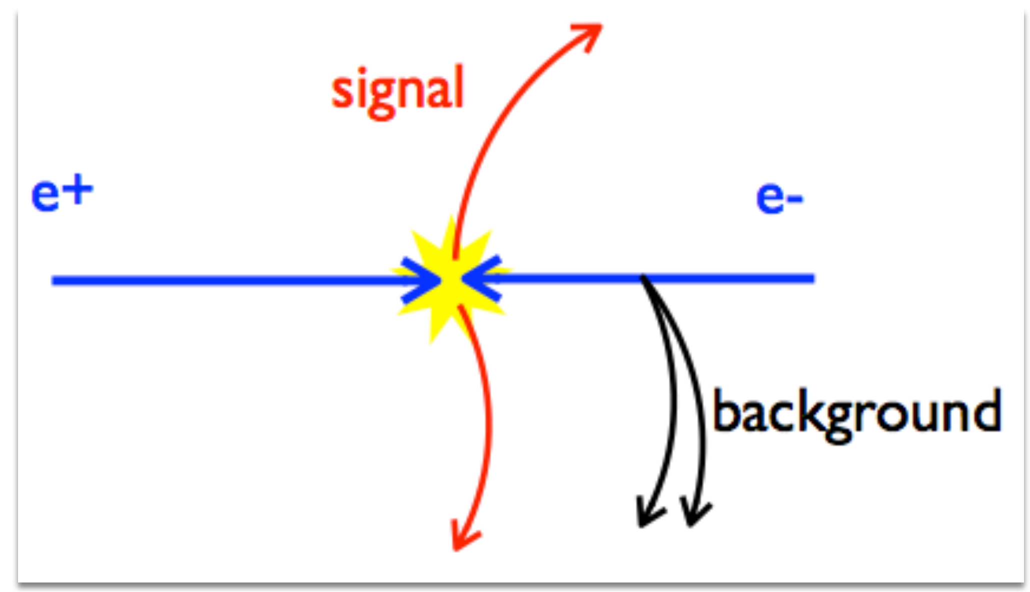
Status of TSIM for 3D-Fitter

Kyungtae Kim(*), Jaebak Kim
Korea University
Jan 25 2011
TRG/DAQ Workshop

Introduction

- Level 1 reconstruction of 3D track parameters may help in reducing beam induced background.

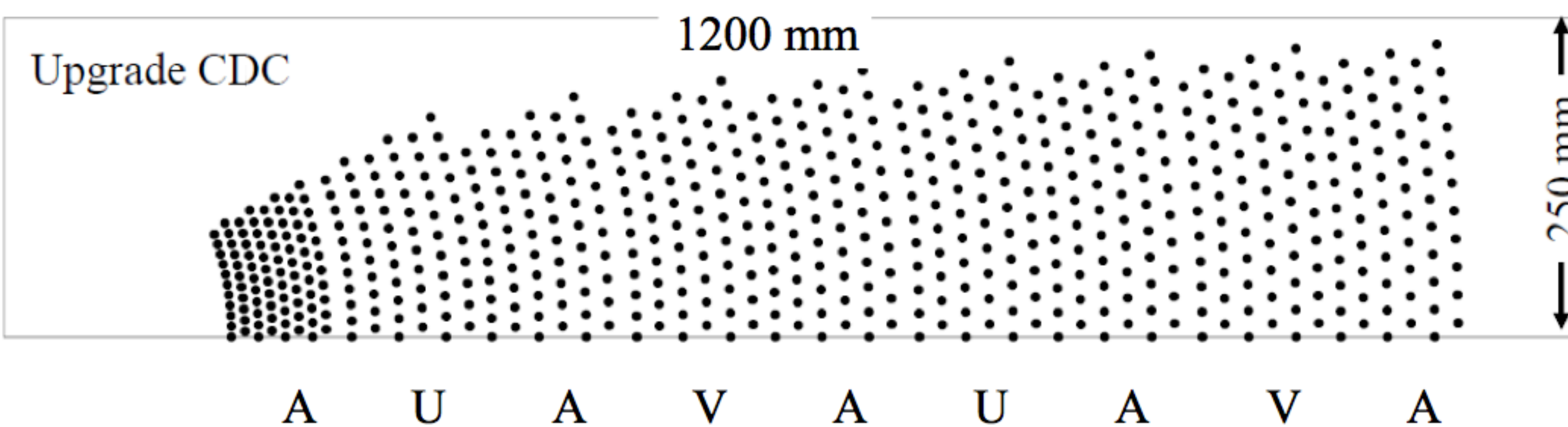




This idea was demonstrated by BaBar
:resolution of z0~4cm
(Nucl. Instr. and Meth.A 518,544(2004).)

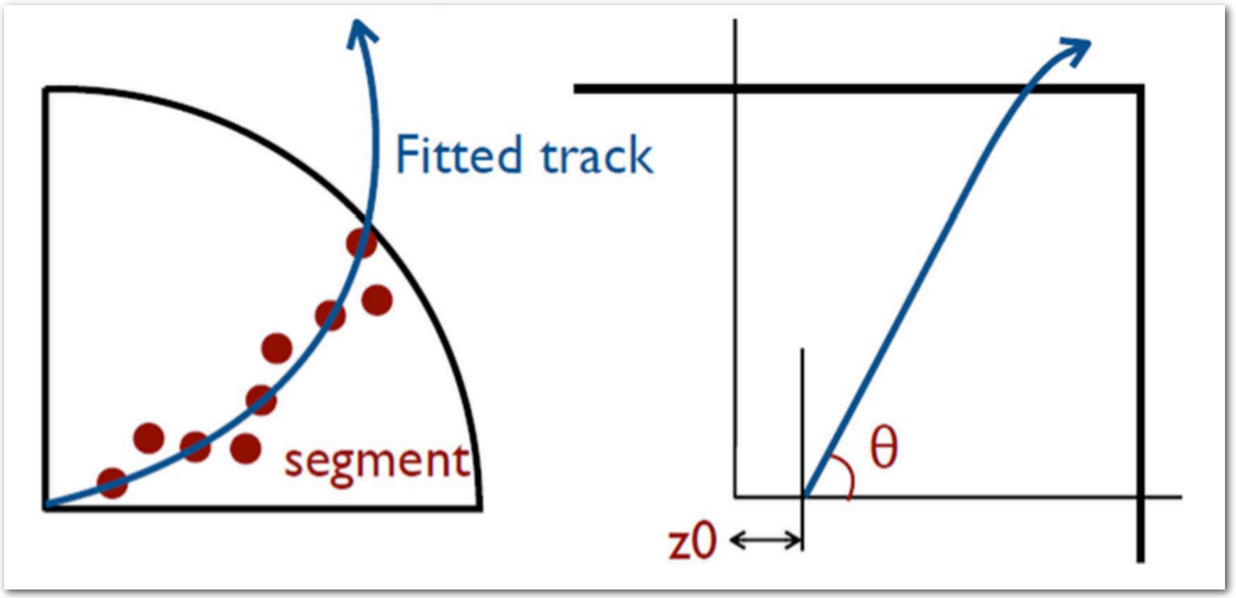
CDC for Belle2

- The new CDC for Belle-II has 9 super layers.(5 axial layers, 4 stereo layers)
- The CDC consists of 14336 cells, each cell is ~16×16㎟ size.



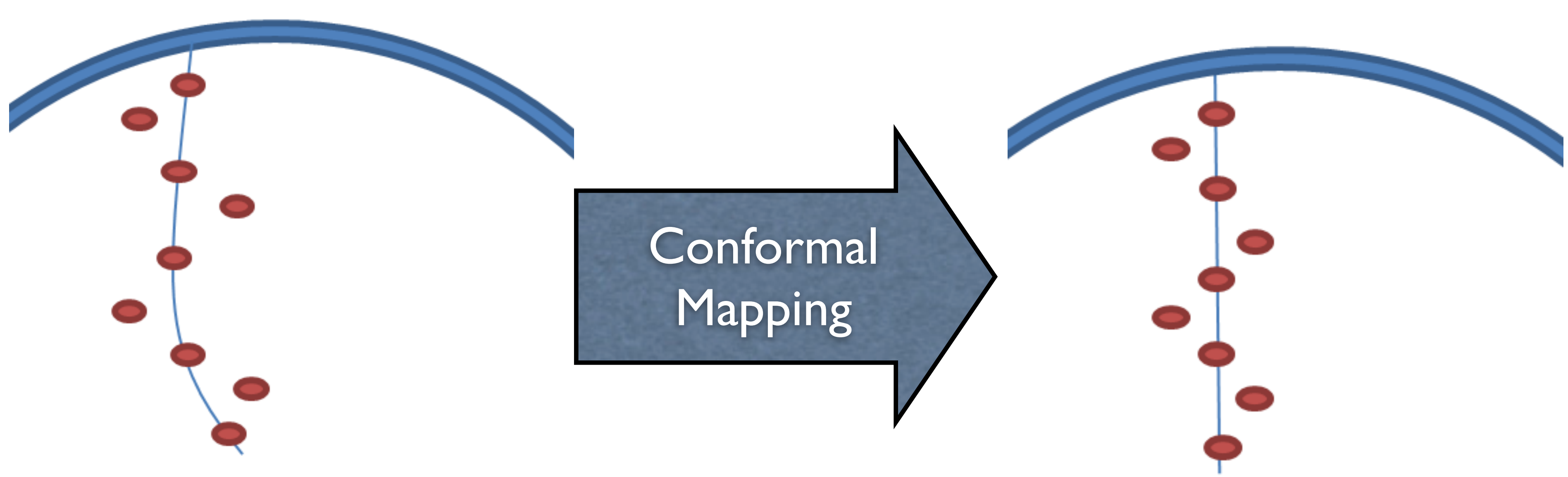
3D-Fitting

- Fit seed tracks from the finder to a helix
  - Seed Track : Set of 9 Track Segments
  - Measure z0, p_T, cotθ
- We have 2 algorithms for 3D fitting with C++, both are working.
  - With Conformal Mapping
  - Without Conformal Mapping

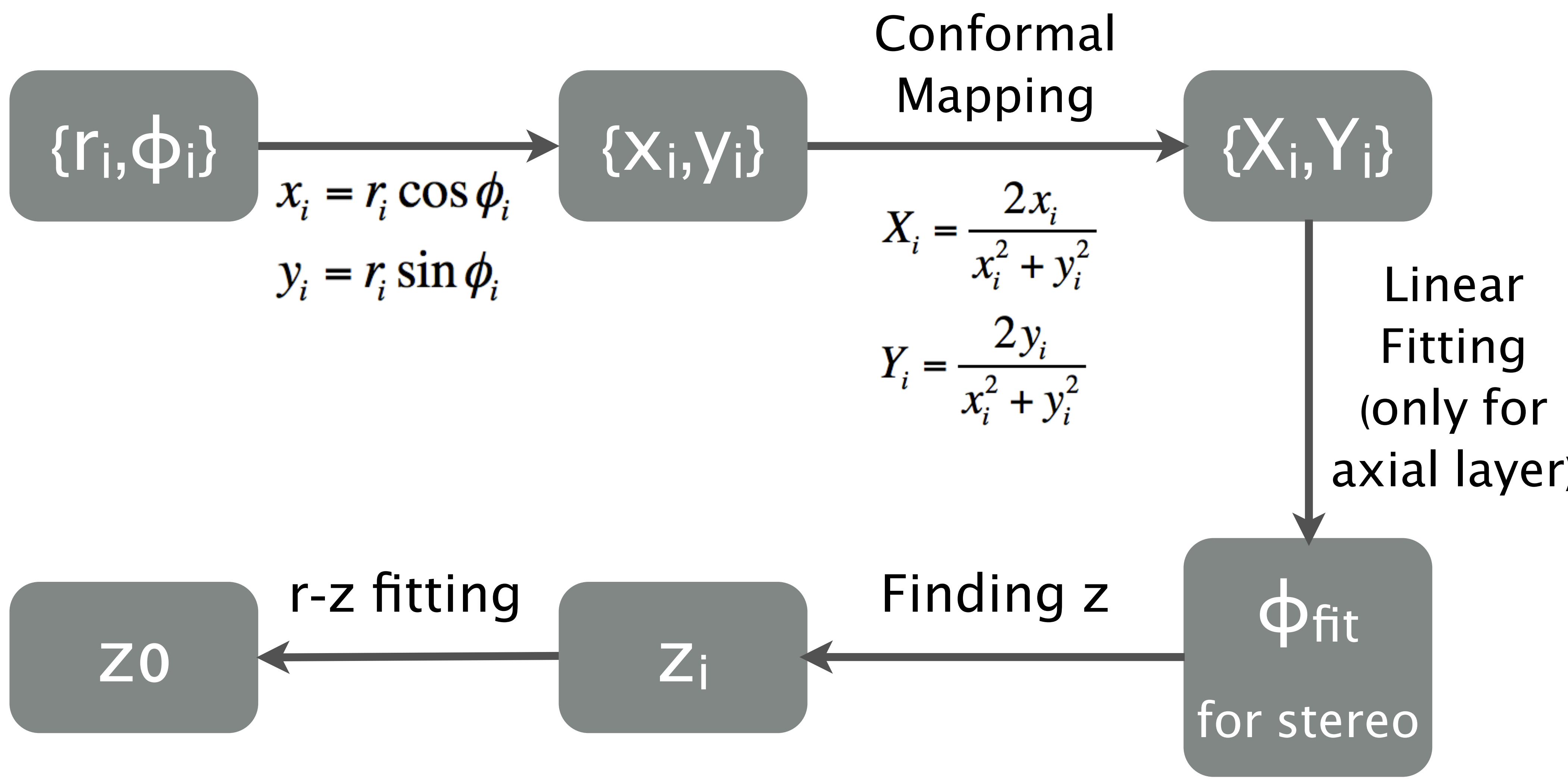


Conformal Mapping

- In mathematics, conformal mapping is a mapping that preserves the angle.
- In our case, we use the conformal mapping to make a helix to a straight line.

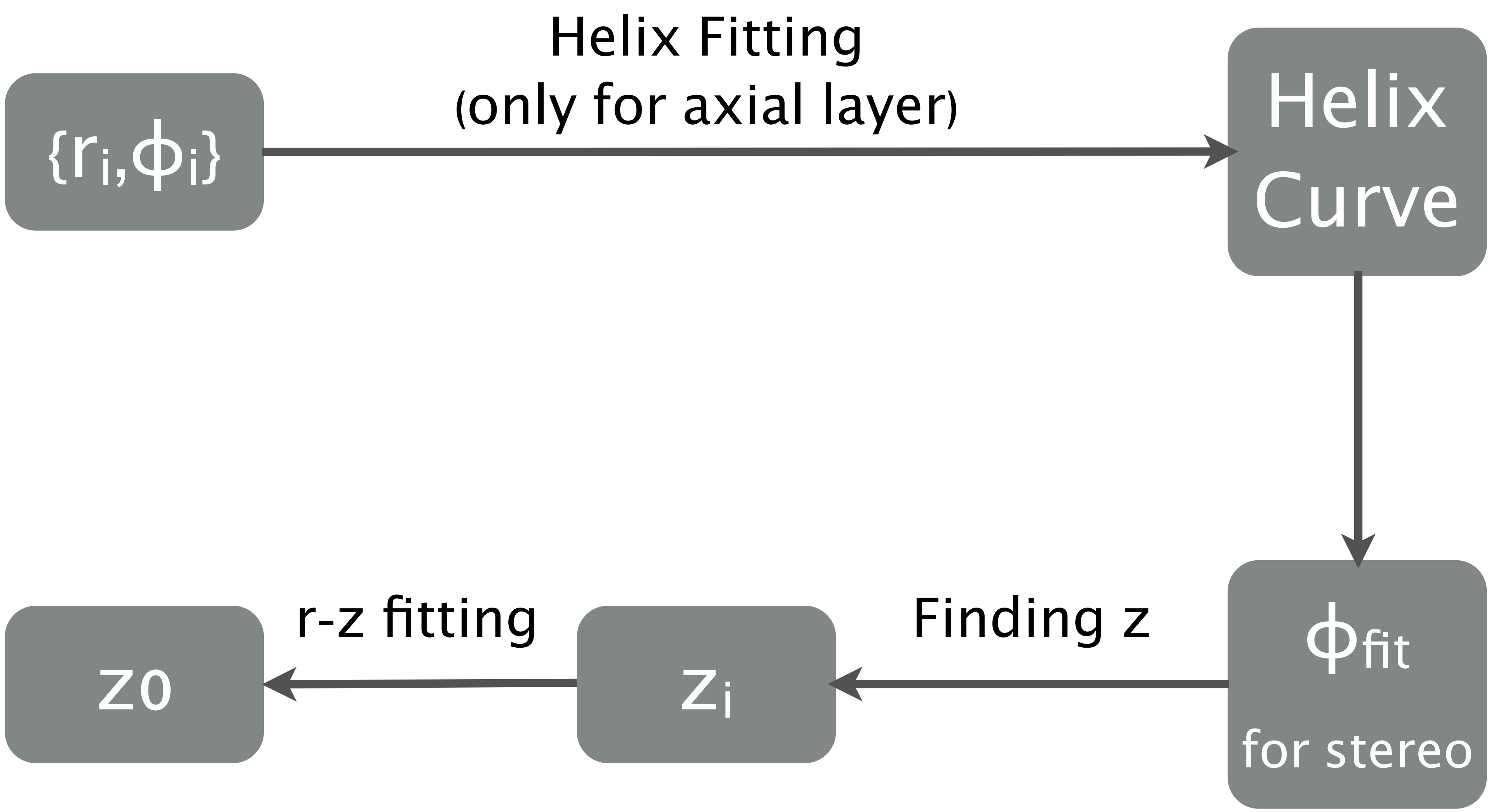


With Conformal Mapping



$$r_i \tan(\phi_{fit} - \phi_i) = z_i \tan\theta_{stereo}$$

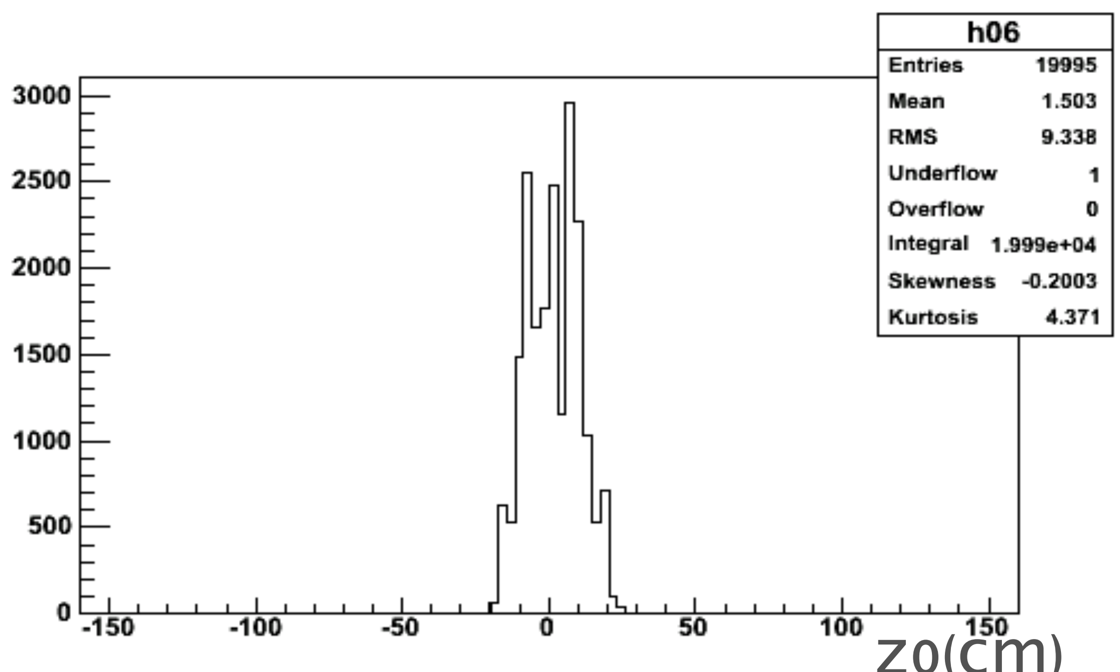
W/O Conformal Mapping



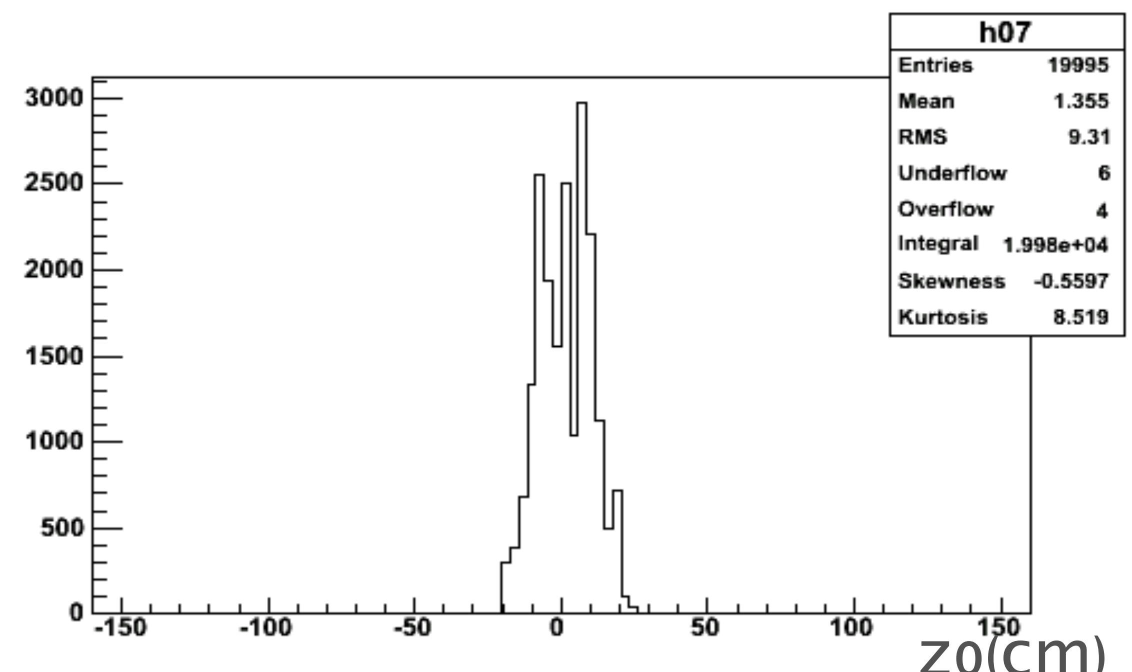
$$r_i \tan(\phi_{fit} - \phi_i) = z_i \tan\theta_{stereo}$$

With or w/o Conformal Mapping

- We tested both algorithms with same data set.
- Both algorithms give similar results.



with C.M



w/o C.M

Last B2GM

- New basf2 CDC geometry has to be in. --> done
- Use timing information to reach z0 resolution ~O(4) cm. --> work in progress
- Write identical C++ into TSIM CDC.
- Move C++ program into integer space.(VHDL)
    - Jaebak Kim is working on this.

TSIM for basf 2

- For 3D-fitting, we need geometry information of the CDC and TS data for input to the fitter.
  - Geometry information --> done
    - Length of layers
    - Stereo angle
  - Track Segment --> work in progress
    - $\{r_i, \phi_i\}$
- And we also need the timing information for better resolution.

Geometric Information

- From the xml code in TSIM, I can calculate the stereo angle.

$$\tan\theta_s = \frac{2R\sin(\pi N_s / N_w)}{L}$$

	Radius (mm)	Length (mm)	θ (mrad)
U2	293.4	733.93	50.84
V4	512.8	1045.23	-61.68
U6	731.2	1087.68	69.43
V8	949.6	1130.13	-74.75

Timing Information

- With some simulation(done by Jaebak), the resolution of drift time($t-t_0$) should be ~38ns to get ~4 cm z0 resolution.

  t : fired time for wire

  t_0 : time that particle passes the cell

- Or ~1.5mm resolution for drift length.

  (Drift velocity is ~40㎛/ns)

Getting t_0

- There are two ideas to get the timing information, especially t_0.
- Using the fastest fired time as t_0 for each events.
- Doing what BaBar did.
  - Actually, they didn't get the t_0

Fastest Time Info.

- Each signal has the fired time and location of wire.
- By modifying TSIM a little, I can get the fastest fired time and its location for each event.

```
void
TRGCDC::fdriftTime(){
	float fdrift=999;
	unsigned a;
	for (unsigned i = 0; i < (unsigned) _hits.size(); i++){
		float dt= _hits[i]->_drift[0]*10*1000/40;
		if (dt<fdrift){
			a=i;
			fdrift=dt;
			}
		}
	cout << endl << "fastest drift Time = " << fdrift << ",";
	cout << "layerId= " << _hits[a]->_wire.layerId()<< endl;
	return;
}
```

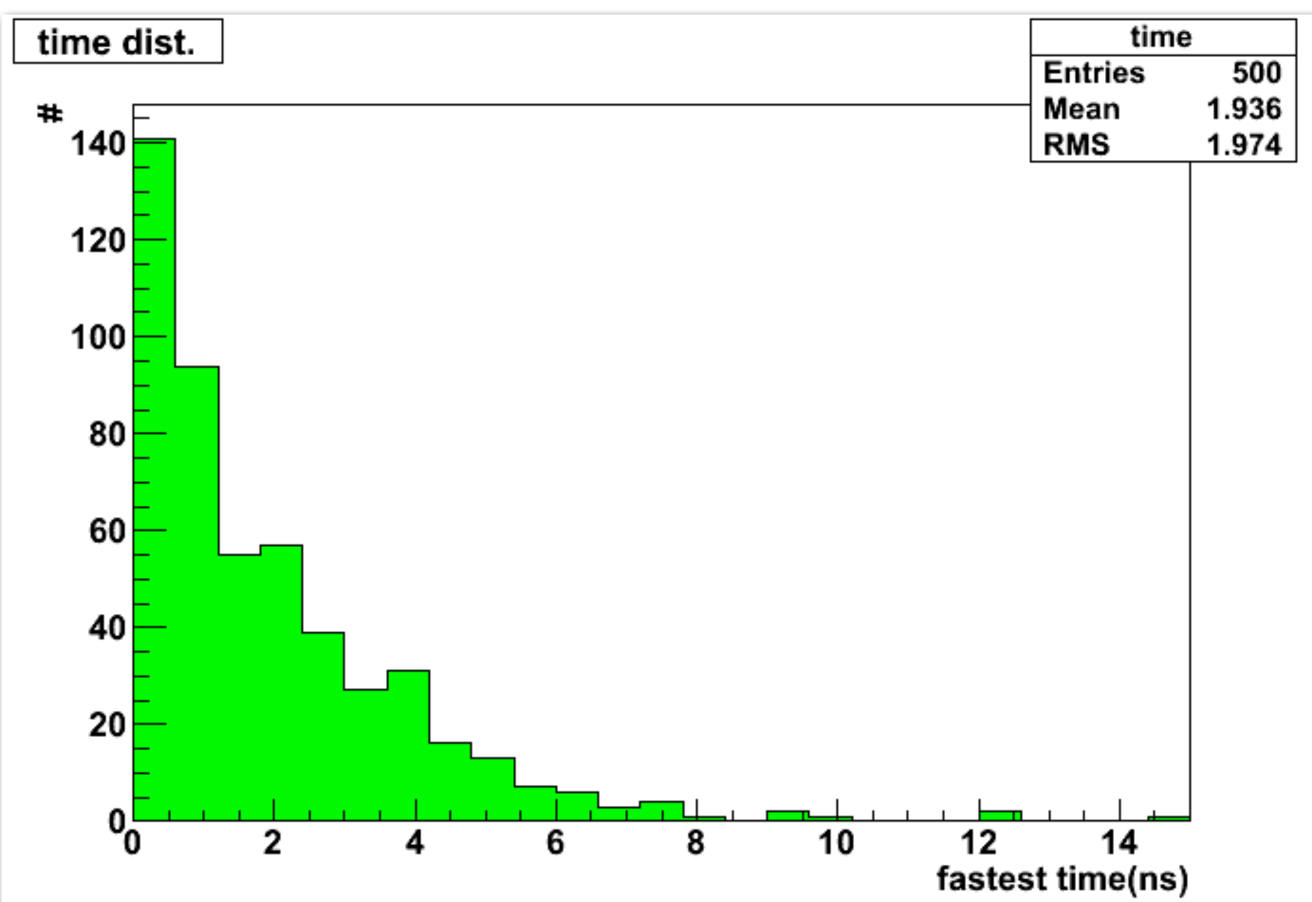
→

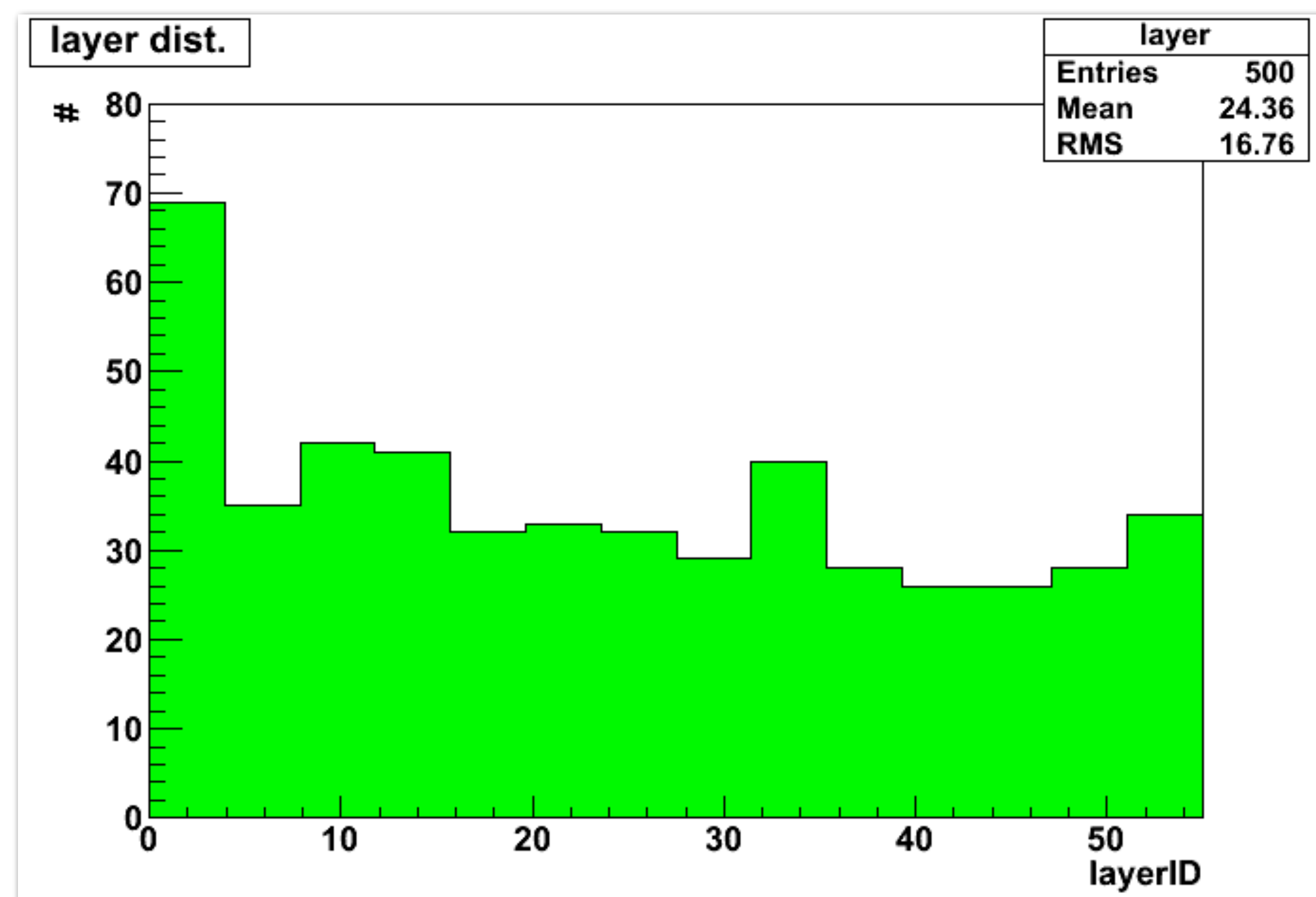
```
[INFO] EXP NUMBER: 1
[INFO] RUN NUMBER: 1
[INFO] EVT NUMBER: 96
 Generator "tester" is called.
G4ParticleGun::pi+
 was defined in terms of KineticEnergy: 0.56
 is now defined in terms Momentum: 1.0055846

fastest drift Time = 6.22683,layerId= 13
```

Fastest Time Distribution

- With 500 event, I can get the distribution of fastest time and layerID for that wire.



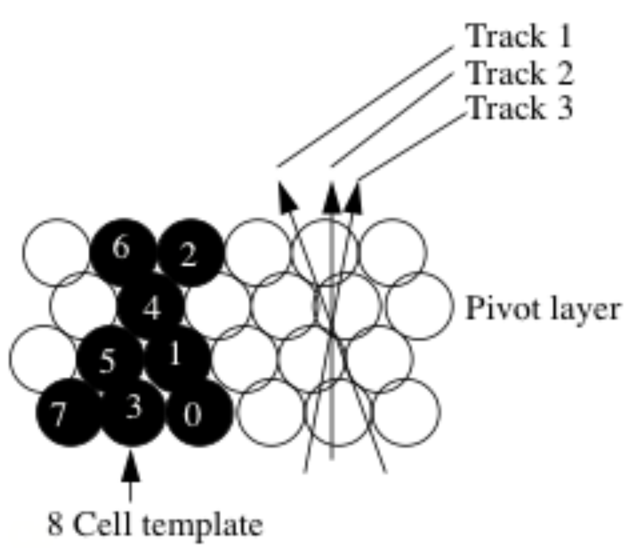


What BaBar did

- From document of BaBar

(http://www.slac.stanford.edu/BFROOT/www/Detector/Trigger/)

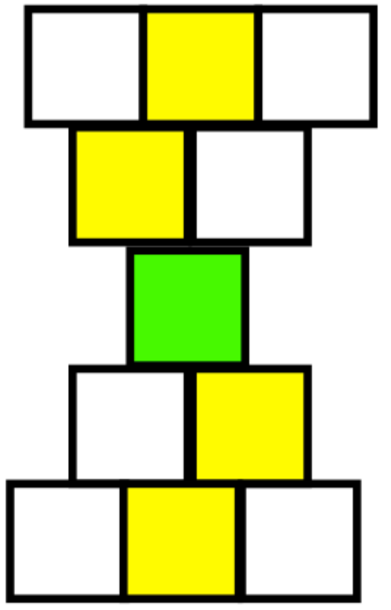
Here we use two bits per cell to give drift time information for the track, instead of just using one to report whether the cell was hit or not. The address reflects how much time has passed since the wire was hit. By incrementing a counter every 269 ns (64 beam crossings) after a hit, we can thus obtain an approximation of the drift time. This is shown on the lower right hand corner of Figure 5.4. From the drift time and the relationship shown in Figure 2.5, we can thus obtain a better axial (ϕ) position resolution, as well as a better time-of-event resolution than in the one-shot case. This approach produces 65536 addresses instead of 256. There are 62721 valid 3/4 segment patterns and 41715 valid 4/4 segment patterns. Of those 4/4 patterns, 367 report a calibrated address as defined in the next section.



Cell/T	T0	1	2	3	4	T0	1	2	3	4	T0	1	2	3	4
6	0	0	0	**1**	2	0	1	2	**3**	0	0	0	0	**1**	2
4	0	0	0	**1**	2	0	0	0	**1**	2	0	0	1	**2**	3
5	0	0	0	**1**	2	0	0	0	**1**	2	0	0	1	**2**	3
3	0	0	0	**1**	2	0	1	2	**3**	0	0	0	0	**1**	2
	Track 1					Track 2					Track 3				

What BaBar did

- In their case, they use 8 cell template and 3/4 (or 4/4) valid segment.
- In our case, we'll use 11 cell template and 4/5(or 5/5) valid segment.
- The number of pattern will be larger than that of BaBar case.



Current Status

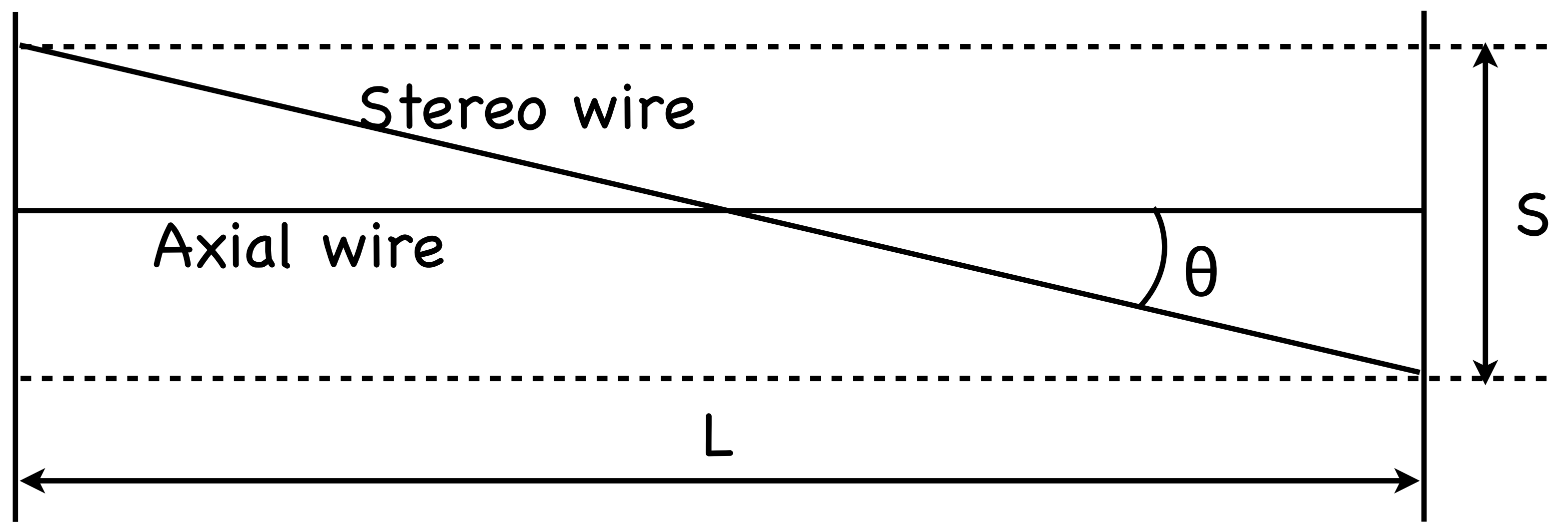
- I'm searching for the track segment from the TSIM for the input to the 3D-fitter.
- I've got the fastest fired time and its location for each event.
- And I'm modifying the code file to save the fastest fired time of each event in ROOT file.

Summary

- We've got the new basf2 CDC geometry info.
- For timing information, we are working with a first method.(fastest fired time)
- If the first method is not enough, we'll do something like what BaBar did.
- We'll write the fitter to the TSIM.

Backup

Calculating θ_s

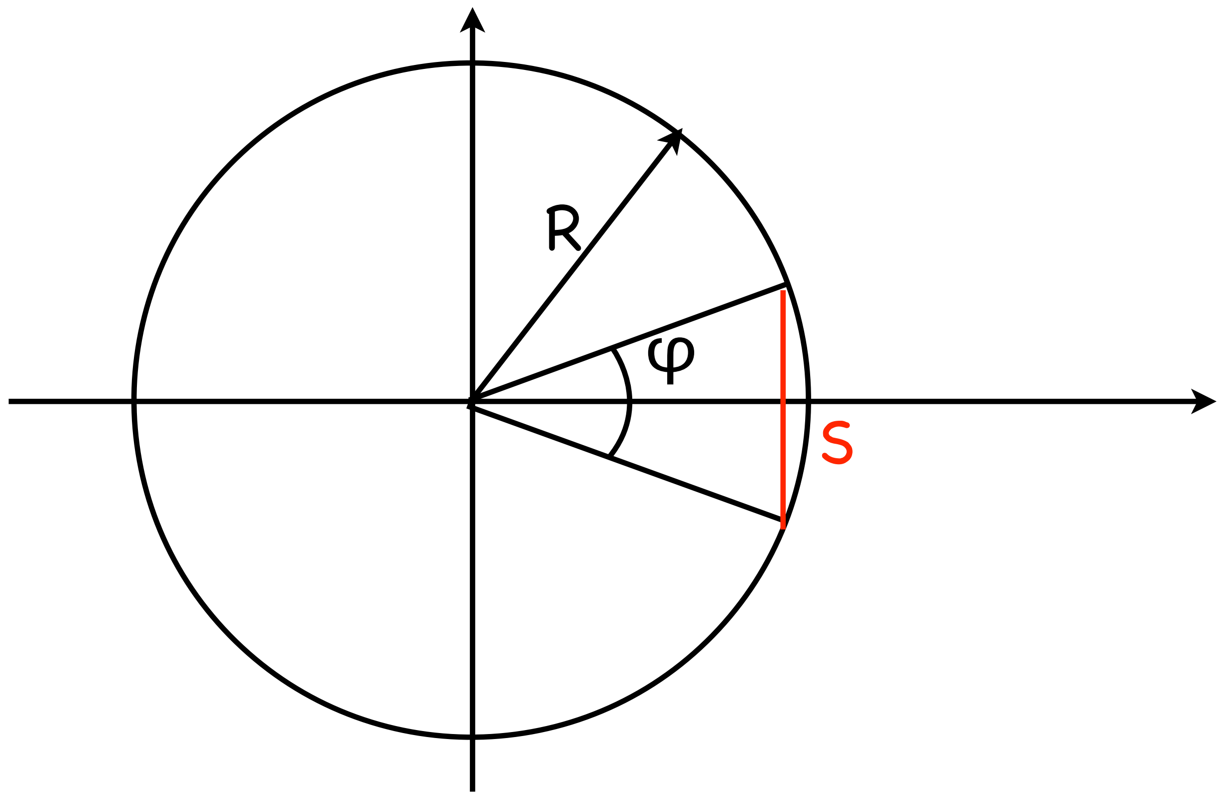


$$\tan\theta = S/L$$

$$L = |Fz - Bz|$$

$$\tan\theta = S/|Fz - Bz|$$

Calculating θ_s



$$s = 2R\sin(\varphi/2)$$

$$\varphi = 2\pi N_s/N_w$$

$$\therefore \tan\theta = \frac{2R\sin(\pi N_s/N_w)}{|Fz - Bz|}$$