

FPGA Curved Track Fitter With Very Low Resource Usage

Jinyuan Wu, M. Wang, E. Gottschalk and Z. Shi

Abstract—Standard least-squares curved track fitting process is tailored for FPGA implementation. The coefficients in the fitting matrices are carefully chosen so that only shift and accumulation operations are used in the process. The divisions and full multiplications are eliminated. Comparison in an application example shows that the fitting errors of the low resource usage implementation are less than 4% bigger than the fitting errors of the exact least-squares algorithm. The implementation is suitable for low-cost, low-power applications such as high energy physics detector trigger systems.

Index Terms—Trigger, Track Fitting, FPGA Firmware, FPGA Computation

I. INTRODUCTION

IN high-energy physics experiment detectors, track fitting is normally considered as a software task in the higher level trigger stage or analysis stage. Although direct porting the fitting algorithm into today's large size FPGA is not impossible, the cost and power consumption quickly become concerns without careful resource usage control. In fact, many silicon area and power consuming operations like multiplications and divisions in many algorithms can be eliminated or replaced by low resource usage operations such as shifts, additions and subtractions. A process deviating from the mathematically accurate one certainly produces less perfect results. However, significant reduction in FPGA logic elements and power consumption overweighs small imperfectness.

In this paper, we describe a curved track fitting functional block suitable for FPGA implementation. The fitting is based on standard least-squares algorithm with modifications on the matrix coefficients to eliminate divisions and full multiplications. The functional block is designed to match data fetching speed so that it can be used to process flowing data stream in trigger and DAQ systems.

The FPGA track fitter was developed for the level 1 pixel trigger of the Fermilab BTeV experiment[1][2]. The pixel detector consists of measurement planes as shown in Fig. 1.

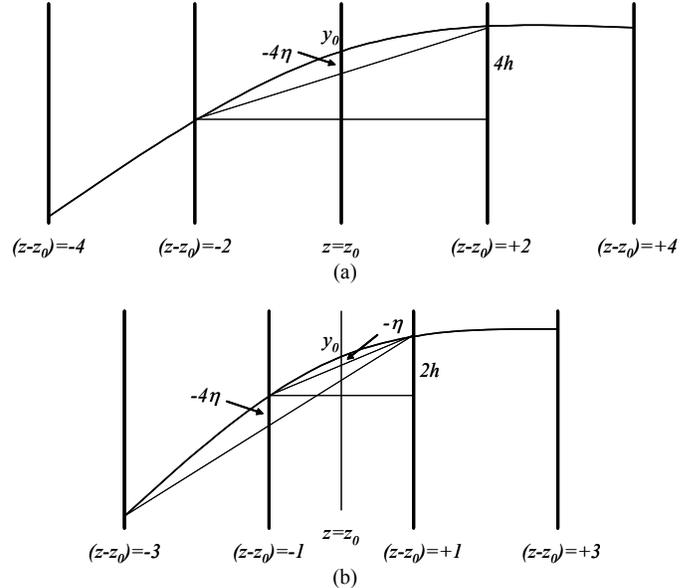


Fig. 1. Tracks in multi-plane detector: (a) The configuration for tracks with odd number of hits. (b) The configuration for tracks with even number of hits.

The hits on pixel planes are first grouped into track segments in the FPGA segment tracker using a triplet finding algorithm like Tiny Triplet Finder (TTF)[3]. Then the hits from a track are grouped together, followed by track fitting, i.e., calculation of the track parameters. In the original baseline of BTeV trigger system, the track fitting is done in the trigger CPU farm partially because the hits from full detector are not switched together until reaching the CPU farm. With our 2004 architecture [4], events are built parasitically in the early stages. Therefore, hits from full detector are available in the FPGA segment tracker, which makes it possible to perform track fitting in FPGA.

II. PRINCIPLE

A. Computations for Track Fitting

The track is projected to both the non-bend and bend views and its equations can be written approximately:

$$x = x_0 + l(z - z_0) \quad (1)$$

$$y = y_0 + h(z - z_0) + \eta(z - z_0)^2$$

Either a track has odd or even number of hits, a center of the track is chosen with $z=z_0$ as shown in Fig. 1. The parameters x_0 and y_0 are offsets of the track and l and h are slopes at track

Manuscript received October 31, 2006. This work was supported in part Operated by Universities Research Association Inc. under Contract No. DE-AC02-76CH03000 with the United States Department of Energy.

J. Wu, M. Wang, E. Gottschalk and Z. Shi are with Fermi National Accelerator Laboratory, Batavia, IL 60510 USA (phone: 630-840-8911; fax: 630-840-2950; e-mail: jyw168@fnal.gov).

center. The parameter η represents the curvature of the track that represents track momentum. With a set of coordinate measurements x_i and y_i , the parameters of the tracks can be found with the following linear combinations.

$$x_0 = \sum_i a_i x_i / \sum_i a_i \quad l = \sum_i b_i x_i / \sum_i b_i (z_i - z_0) \quad (2)$$

$$y_0 = \sum_i c_i y_i / \sum_i c_i \quad h = \sum_i d_i y_i / \sum_i d_i (z_i - z_0)$$

$$\eta = \sum_i e_i y_i / \sum_i e_i (z_i - z_0)^2$$

The coefficients can be chosen nearly freely as long as the following constraints are met:

$$\sum_i a_i (z_i - z_0) = 0 \quad (3)$$

$$\sum_i b_i = 0 \quad (4)$$

$$\sum_i c_i (z_i - z_0) = 0 \quad \sum_i c_i (z_i - z_0)^2 = 0 \quad (5)$$

$$\sum_i d_i = 0 \quad \sum_i d_i (z_i - z_0)^2 = 0 \quad (6)$$

$$\sum_i e_i = 0 \quad \sum_i e_i (z_i - z_0) = 0 \quad (7)$$

Depending on the choice of the coefficients in the linear combination, the errors of the fitting can be different and the coefficients derived from the least-squares fitting provide minimum errors for track reconstruction.

B. Non-Division Integer-Only Operations for FPGA

In general, the computations above need floating point multiplications and divisions. To simplify the computation so that it can be done in FPGA with low resource usage, the advantage of invariance of Equations (2) through (7) under re-scaling can be taken. Each set of coefficients can be multiplied with a common factor so that the sums in the denominators in Equation (2) become 32, 512 or 4096. The constraints for the coefficients can be summarized as following:

$$\sum_i a[i] = 32 \quad \sum_i a[i](z_i - z_0) = 0 \quad (8)$$

$$\sum_i b[i](z_i - z_0) = 512 \quad \sum_i b[i] = 0 \quad (9)$$

$$\sum_i c[i] = 32 \quad \sum_i c[i](z_i - z_0) = 0 \quad \sum_i c[i](z_i - z_0)^2 = 0 \quad (10)$$

$$\sum_i d[i](z_i - z_0) = 512 \quad \sum_i d[i] = 0 \quad \sum_i d[i](z_i - z_0)^2 = 0 \quad (11)$$

$$\sum_i e[i](z_i - z_0)^2 = 4096 \quad \sum_i e[i] = 0 \quad \sum_i e[i](z_i - z_0) = 0 \quad (12)$$

Then the linear combinations for calculating the track parameters can be rewritten:

$$xx32 = \sum_i a[i]x[i] \approx 32x_0 \quad ll512 = \sum_i b[i]x[i] \approx 512l \quad (13)$$

$$yy32 = \sum_i c[i]y[i] \approx 32y_0 \quad hh512 = \sum_i d[i]y[i] \approx 512h$$

$$eta4096 = \sum_i e[i]y[i] \approx 4096\eta$$

The unit of measured coordinates $x[i]$ and $y[i]$ are chosen so that they are integers. For example, the channel number in the silicon pixel detector can be used as the hit coordinates. The

coefficients in the linear combinations are also chosen to be integers and the results of the linear combinations: $xx32$, $ll512$, $yy32$, $hh512$ and $eta4096$ are also integers, representing the corresponding parameters scaled by factors of 32, 512 and 4096, respectively. Note that divisions are not needed anymore. The only computations needed in Equation (13) are integer multiplications and accumulation.

The unit in z direction is chosen so that the separation between two detector planes is 2. In this unit, the possible values of $(z_i - z_0)$ are also integers, which are even when the number of hits is odd with the middle plane being 0 and are odd when the number of hits is even with the two center detector planes being -1 and +1.

C. Eliminating Full Multiplications

To reduce computations further, the coefficients in the linear combinations are limited to the ‘‘weight-two’’ or ‘‘two-bit’’ integers, e.g. 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 14, 15, which are $2^m + 2^n$ or $2^m - 2^n$ with positive integers m and n . An example of choosing $e[i]$ for tracks with odd numbers of hits is shown in Table I.

TABLE I
COEFFICIENTS FOR THE FPGA TRACK FITTER (CURVATURE, ODD HITS)

		Half-length of the Track													
		16		14		12		10		8		6		4	
$z-z_0$		e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$
-16	5.3	6													
-14	3.3	2	7.5	8											
-12	1.6	2	4.3	4	11.3	12									
-10	0.1	0	1.6	2	5.6	5	17.9	18							
-8	-1.1	0	-0.7	-2	1.0	1	7.2	7	31.0	31					
-6	-2.0	-3	-2.4	-2	-2.6	-4	-1.2	-1	7.8	8	61.0	56			
-4	-2.6	-3	-3.6	-5	-5.1	-5	-7.2	-8	-8.9	-9	0.0	12	146.3	144	
-2	-3.0	-3	-4.4	-4	-6.6	-5	-10.7	-9	-18.8	-20	-36.6	-40	-73.1	-64	
0	-3.2	-2	-4.6	-2	-7.2	-8	-11.9	-14	-22.2	-20	-48.8	-56	-146.3	-160	
2	-3.0	-3	-4.4	-4	-6.6	-5	-10.7	-9	-18.8	-20	-36.6	-40	-73.1	-64	
4	-2.6	-3	-3.6	-5	-5.1	-5	-7.2	-8	-8.9	-9	0.0	12	146.3	144	
6	-2.0	-3	-2.4	-2	-2.6	-4	-1.2	-1	7.8	8	61.0	56			
8	-1.1	0	-0.7	-2	1.0	1	7.2	7	31.0	31					
10	0.1	0	1.6	2	5.6	5	17.9	18							
12	1.6	2	4.3	4	11.3	12									
14	3.3	2	7.5	8											
16	5.3	6													
Error	2.91	3.02	3.05	3.15	3.22	3.26	3.41	3.43	3.65	3.65	3.93	3.99	4.28	4.29	
Ratio		1.04		1.03		1.01		1.00		1.00		1.02		1.00	

The columns of e_i in Table I represent coefficients derived from the least-squares fitting. The $e[i]$ coefficients are chosen in a spread sheet, guided by the e_i coefficients. Since the parameterization of the track in Equation (1) is chosen with symmetry around z_0 , the coefficients for the least-squares fitting are also symmetric. Some symmetric properties are sufficient conditions for certain constraints in Equations (8) through (12). For example, the symmetric property $e_i = e_{-i}$ exists and it is a sufficient condition for the constraint $\sum e_i (z_i - z_0) = 0$ making it satisfied automatically. Appropriate symmetries are programmed in the spread sheet cells so that the constraints and scaling requirement are satisfied with minimum hand editing. In our work, the coefficient selection is semi-automatic, partially for purpose of our own better understanding to the problem. Clearly it is not too difficult to write a program that chooses the coefficients automatically.

The relative errors contributed by the parameter η for both

algorithms are calculated. The error here is defined as transverse reconstruction RMS error after projecting the track by half-length ($L/2$) from first or last hit of the track, with unit of the RMS error for the $y[i]$ measurements. Assume the errors of $y[i]$ measurements δy_i are independent and they have a same RMS value δy , then the error of calculating parameter η can be estimated:

$$\delta\eta = \delta y \sqrt{\sum_i (e[i])^2 / 4096} \quad (14)$$

The assumption of independence of the measurement errors may be violated if most tracks are high momentum and parallel but it is a good approximation for typical detector configuration. The transverse reconstruction RMS error δY after projecting the track by half-length from first or last hit of the track, i.e., $(z-z_0) = 2(L/2)$, can be calculated:

$$\delta Y = (z - z_0)^2 \delta\eta = 4(L/2)^2 \delta y \sqrt{\sum_i (e[i])^2 / 4096} \quad (15)$$

Note that the ‘‘projecting’’ here should just be viewed as a rescaling process to bring the fitting error of a parameter to a convenient unit. It is an estimate for a single parameter only. If, for example, the vertex measurement error is needed, the errors for all parameters should be considered simultaneously since they could be correlated. The coefficients $e[i]$ for tracks with even numbers of hits is shown in Table II.

TABLE II
COEFFICIENTS FOR THE FPGA TRACK FITTER (CURVATURE, EVEN HITS)

$z-z_0$	Half-length of the Track															
	15		13		11		9		7		5		3			
e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	e_i	$e[i]$	
-15	6.3	6														
-13	3.8	4	9.1	9												
-11	1.6	2	4.9	5	14.1	14										
-9	-0.2	0	1.4	1	6.4	7	23.3	24								
-7	-1.6	-2	-1.4	-1	0.3	0	7.8	8	42.7	40						
-5	-2.7	-4	-3.5	-2	-4.3	-5	-3.9	-6	6.1	10	91.4	96				
-3	-3.4	-2	-4.9	-6	-7.4	-9	-11.6	-14	-18.3	-14	-18.3	-32	256.0	256		
-1	-3.8	-4	-5.6	-6	-9.0	-7	-15.5	-12	-30.5	-36	-73.1	-64	-256.0	-256		
1	-3.8	-4	-5.6	-6	-9.0	-7	-15.5	-12	-30.5	-36	-73.1	-64	-256.0	-256		
3	-3.4	-2	-4.9	-6	-7.4	-9	-11.6	-14	-18.3	-14	-18.3	-32	256.0	256		
5	-2.7	-4	-3.5	-2	-4.3	-5	-3.9	-6	6.1	10	91.4	96				
7	-1.6	-2	-1.4	-1	0.3	0	7.8	8	42.7	40						
9	-0.2	0	1.4	1	6.4	7	23.3	24								
11	1.6	2	4.9	5	14.1	14										
13	3.8	4	9.1	9												
15	6.3	6														
Error Ratio	2.98	3.04	3.13	3.17	3.31	3.34	3.53	3.57	3.78	3.82	4.09	4.13	4.50	4.50	1.00	

The relative errors contributed by the parameter η for both algorithms are also calculated in Table II.

From Tables I and II, it can be seen that the errors increases for approximate algorithm with the two-bit integer-only coefficients comparing with the mathematically perfect least-squares algorithm. However, the imperfectness of coefficients for the FPGA fitting algorithm increases the track reconstruction errors only slightly (less than 4%).

III. FPGA IMPLEMENTATION

The block diagram of the FPGA performing the track fitting functions is shown in Fig. 2. The hit coordinates $x(i)$ and $y(i)$ are clocked through the fitter, one clock cycles per number.

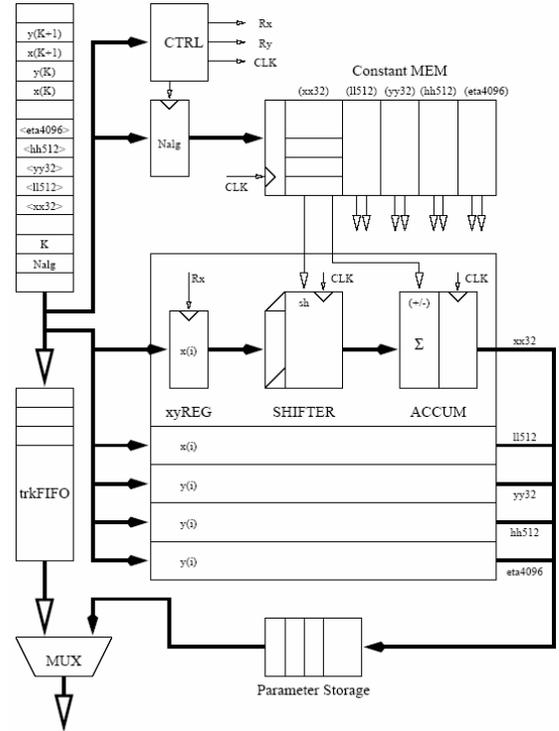


Fig. 2. The FPGA track fitter.

For each parameter, an accumulator is used to calculate the linear combination. The coordinate data is shifted through a logarithmic shifter by pre-defined numbers of bits that are stored in the constant memory. The shifted version of the coordinate is added to or subtracted from the accumulator.

Each coordinate is shifted and added/subtracted twice that is equivalent to multiplying the coordinate by a two-bit integer and accumulating for the linear combination. The operation uses two clock cycles that matches the number of cycles needed to fetch a pair of coordinates $x(i)$ and $y(i)$.

For a fitter processing 16-bit coordinates, 630 logic elements can be accounted for in the register, shifter & accumulator blocks for the 5 parameters. The silicon resource usage of this portion is about 11% of a \$30 Altera EPIC6 [5] device.

IV. FITTING ERRORS AND DISCUSSIONS

The coefficients of fitting matrix for the other two parameters of a curved track $yy32$ and $hh512$ are also calculated using similar procedures as shown in Table III-VI. The coefficients chosen are two-bit integers. The relative fitting errors contributed by these two parameters are also calculated. The relative errors contributed by the three parameters with different track lengths for both least-squares and approximate algorithms are plotted in Fig. 3.

TABLE III
COEFFICIENTS FOR THE FPGA TRACK FITTER (SLOPE, ODD HITS)

		Half-length of the Track															
		16		14		12		10		8		6		4			
$z-z_0$		d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$
-16	-5.0	-5															
-14	-4.4	-4	-6.4	-6													
-12	-3.8	-4	-5.5	-6	-8.4	-8											
-10	-3.1	-3	-4.6	-5	-7.0	-7	-11.6	-12									
-8	-2.5	-3	-3.7	-3	-5.6	-6	-9.3	-9	-17.1	-16							
-6	-1.9	-2	-2.7	-3	-4.2	-5	-7.0	-7	-12.8	-12	-27.4	-28					
-4	-1.3	-1	-1.8	-2	-2.8	-2	-4.7	-5	-8.5	-12	-18.3	-18	-51.2	-56			
-2	-0.6	-1	-0.9	0	-1.4	-2	-2.3	-1	-4.3	-4	-9.1	-8	-25.6	-16			
0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0	0.0	0			
2	0.6	1	0.9	0	1.4	2	2.3	1	4.3	4	9.1	8	25.6	16			
4	1.3	1	1.8	2	2.8	2	4.7	5	8.5	12	18.3	18	51.2	56			
6	1.9	2	2.7	3	4.2	5	7.0	7	12.8	12	27.4	28					
8	2.5	3	3.7	3	5.6	6	9.3	9	17.1	16							
10	3.1	3	4.6	5	7.0	7	11.6	12									
12	3.8	4	5.5	6	8.4	8											
14	4.4	4	6.4	6													
16	5.0	5															
Error	0.79	0.80	0.84	0.84	0.89	0.89	0.95	0.96	1.03	1.05	1.13	1.13	1.26	1.29			
Ratio		1.00		1.01		1.01		1.00		1.01		1.00		1.02			

TABLE IV
COEFFICIENTS FOR THE FPGA TRACK FITTER (SLOPE, EVEN HITS)

		Half-length of the Track												
		15		13		9		7		5		3		
$z-z_0$		d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	d_i	$d[i]$	
-15	-5.6	-6												
-13	-4.9	-5	-7.3	-7										
-11	-4.1	-4	-6.2	-6	-9.8	-10								
-9	-3.4	-3	-5.1	-5	-8.1	-8	-14.0	-14						
-7	-2.6	-3	-3.9	-4	-6.3	-6	-10.9	-10	-21.3	-20				
-5	-1.9	-1	-2.8	-4	-4.5	-5	-7.8	-8	-15.2	-16	-36.6	-36		
-3	-1.1	-1	-1.7	-2	-2.7	-2	-4.7	-6	-9.1	-12	-21.9	-24	-76.8	-80
-1	-0.4	-1	-0.6	0	-0.9	-1	-1.6	-2	-3.0	0	-7.3	-4	-25.6	-16
1	0.4	1	0.6	0	0.9	1	1.6	2	3.0	0	7.3	4	25.6	16
3	1.1	1	1.7	2	2.7	2	4.7	6	9.1	12	21.9	24	76.8	80
5	1.9	1	2.8	4	4.5	5	7.8	8	15.2	16	36.6	36		
7	2.6	3	3.9	4	6.3	6	10.9	10	21.3	20				
9	3.4	3	5.1	5	8.1	8	14.0	14						
11	4.1	4	6.2	6	9.8	10								
13	4.9	5	7.3	7										
15	5.6	6												
Error	0.81	0.82	0.86	0.87	0.92	0.92	0.99	0.99	1.08	1.09	1.20	1.20	1.34	1.35
Ratio		1.01		1.01		1.00		1.00		1.01		1.00		1.01

TABLE V
COEFFICIENTS FOR THE FPGA TRACK FITTER (OFFSET, ODD HITS)

		Half-length of the Track															
		16		14		12		10		8		6		4			
$z-z_0$		c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$
-16	-2.1	-2															
-14	-0.6	-1	-2.3	-2													
-12	0.7	1	-0.4	0	-2.5	-3											
-10	1.8	2	1.2	1	0.0	0	-2.7	-3									
-8	2.7	3	2.5	1	2.0	3	0.7	1	-2.9	-3							
-6	3.4	3	3.5	4	3.6	4	3.3	4	1.9	2	-3.0	-3					
-4	3.9	3	4.3	4	4.7	5	5.1	4	5.4	6	4.6	5	-2.7	-3			
-2	4.2	4	4.7	5	5.4	4	6.3	7	7.5	6	9.1	7	11.0	12			
0	4.3	6	4.8	6	5.6	6	6.6	6	8.2	10	10.7	14	15.5	14			
2	4.2	4	4.7	5	5.4	4	6.3	7	7.5	6	9.1	7	11.0	12			
4	3.9	3	4.3	4	4.7	5	5.1	4	5.4	6	4.6	5	-2.7	-3			
6	3.4	3	3.5	4	3.6	4	3.3	4	1.9	2	-3.0	-3					
8	2.7	3	2.5	1	2.0	3	0.7	1	-2.9	-3							
10	1.8	2	1.2	1	0.0	0	-2.7	-3									
12	0.7	1	-0.4	0	-2.5	-3											
14	-0.6	-1	-2.3	-2													
16	-2.1	-2															
Error	0.36	0.37	0.39	0.40	0.42	0.43	0.46	0.46	0.51	0.51	0.58	0.59	0.70	0.70			
Ratio		1.02		1.02		1.02		1.01		1.02		1.03		1.00			

TABLE VI
COEFFICIENTS FOR THE FPGA TRACK FITTER (OFFSET, EVEN HITS)

		Half-length of the Track													
		15		13		11		9		7		5		3	
$z-z_0$		c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$	c_i	$c[i]$
-15	-2.2	-2													
-13	-0.5	-1	-2.4	-3											
-11	0.9	1	-0.2	0	-2.6	-2									
-9	2.1	2	1.6	3	0.3	0	-2.8	-3							
-7	3.1	4	3.0	3	2.6	2	1.2	1	-3.0	-3					
-5	3.8	4	4.1	4	4.3	3	4.2	5	3.0	3	-3.0	-3			
-3	4.3	4	4.8	4	5.4	7	6.2	7	7.0	7	7.0	7	-2.0	-2	
-1	4.5	4	5.1	5	6.0	6	7.2	6	9.0	9	12.0	12	18.0	18	
1	4.5	4	5.1	5	6.0	6	7.2	6	9.0	9	12.0	12	18.0	18	
3	4.3	4	4.8	4	5.4	7	6.2	7	7.0	7	7.0	7	-2.0	-2	
5	3.8	4	4.1	4	4.3	3	4.2	5	3.0	3	-3.0	-3			
7	3.1	4	3.0	3	2.6	2	1.2	1	-3.0	-3					
9	2.1	2	1.6	3	0.3	0	-2.8	-3							
11	0.9	1	-0.2	0	-2.6	-2									
13	-0.5	-1	-2.4	-3											
15	-2.2	-2													
Error	0.38	0.38	0.40	0.41	0.44	0.45	0.48	0.48	0.54	0.54	0.63	0.63	0.80	0.80	
Ratio		1.01		1.01		1.02		1.01		1.00		1.00		1.00	

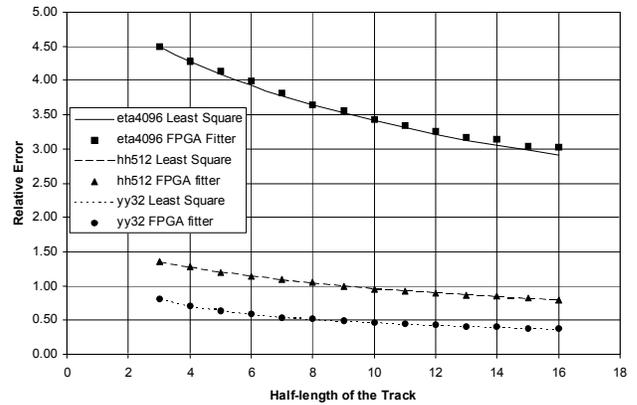


Fig. 3. Relative errors between the least square fit and the FPGA fit.

The approximate algorithm used in FPGA increases the fitting errors for all parameters as expected from mathematic principle. However, the differences are very small. In other words, the fitting errors are relatively insensitive to the variations of the coefficients of the fitting matrix.

It can be seen from Fig. 3 that the relative error contributed by the curvature parameter is significantly higher than the errors contributed by the offset and slope parameters. We will focus on the curvature parameter in our discussion.

In general, fitting of longer tracks yields smaller errors due to two reasons: longer lever arms and more measurement points. In order to compare these two effects, relative errors contributed by the curvature parameter calculated with several fitting schemes are plotted in Fig. 4.

In addition to the least-squares algorithm and the FPGA approximation shown in Table I and II, two other “3-point” algorithms are also studied. One of the 3-point schemes calculates eta4096 using hits of the first three detector planes at the beginning of a track. In this case, the lever arm is a fix length, despite of extra length provided by additional planes. This scheme produces largest errors. When the track is projected over long distances, the errors increase rapidly. The calculation method of this scheme is the simplest. However,

the results from this scheme is only useful as a coarse estimate of track momentum, it causes large errors due to short lever arm when the tracks are to be projected in long distance.

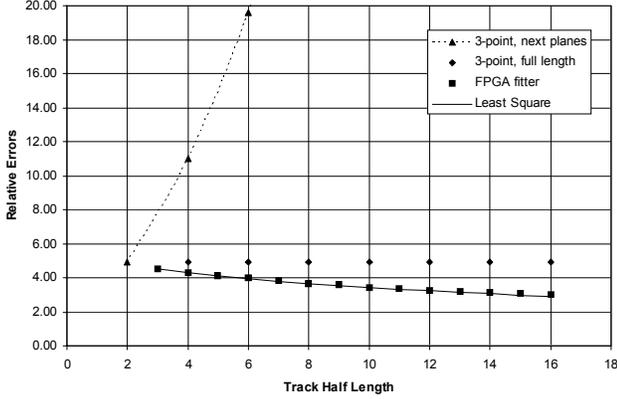


Fig. 4. Relative errors of several track fitting schemes.

The other 3-point scheme calculates the eta4096 using the first, the middle and the last hits of a track. In this situation, advantage of the full length of the track is taken. However, the information provided by the redundant measurements of the other points in the track is not used. The relative errors when the track is projected over half-length are nearly a constant.

The full-length 3-point scheme may appear to be simpler in computation than the FPGA fitting scheme, but actually it is not the case. To calculate eta4096 using the 3-point scheme, a floating point multiplication is needed in order to bring results for different track lengths into a unified scale. In the FPGA fitting scheme, the scale unification is achieved through choosing the two-bit coefficients.

V. CONCLUSION

A fitting algorithm suitable for FPGA implementation has been discussed. With integer shifting and accumulating operations, the fitting errors in the approximate algorithm is only slightly larger than the errors of the least-squares fitting algorithm.

Multippliers are now available in more and more FPGA devices today and it seems that eliminating multiplications is not as critical as several years ago. Intrinsically, however, multiplication is a power and resource consuming operation. It is still a good practice to reserve multipliers to the processes in which multiplications absolutely can not be eliminated, or the substitution of the multiplications causes significant degrade of the quality of the results.

Generally speaking, more computations yield better quality of the results. From the 3-point schemes, to the FPGA fitter, to the least-squares algorithm, the fitting errors reduce as number of total operations increases. However, after certain point, the quality of the results does not improve as rapidly as before. It is common that large amount of computation brings only small improvement in the mathematically perfect algorithms. So it is possible to find algorithms with reasonable amount of computations that produce sufficiently good results, as illustrated in this paper.

APPENDIX

In this section, we first define a matrix called algorithm matrix which represents a generic fitting algorithm. Then we will show that the least-squares algorithm is a special case of the generic algorithms and compare their fitting errors.

Consider fitting n measurement points with a set of m ($m < n$) functions using various algorithms:

$$y = a_0 v_0(x) + a_1 v_1(x) + a_2 v_2(x) + \dots + a_m v_m(x) \quad (A1)$$

In fitting with parabola, e.g., the functions $v_0(x)$, $v_1(x)$, $v_2(x)$ are chosen to be 1 , x , x^2 . The parameters a_0 , a_1 to a_m are to be evaluated through fitting the measurement data (x_1, y_1) , (x_2, y_2) to (x_m, y_m) . The x-coordinates here are known and can have any values. We use an n -row by m -column matrix $V = \{v_{ij}\} = \{v_j(x_i)\}$ to denote the function values at the corresponding measurement points.

Definition 1: The algorithm matrix $g = \{g_{ji}\}$ is defined to be an m -row by n -column left inverse matrix of V :

$$gV = I \quad \sum_{i=1}^n g_{ji} v_{ik} = \delta_{jk} \quad j, k = 0, 1, \dots, m \quad (A2)$$

With an algorithm matrix g , the values calculated from the following linear combinations are called a set of fitting parameters under algorithm g :

$$A = gY \quad a_j = \sum_{i=1}^n g_{ji} y_i \quad j = 0, 1, \dots, m \quad (A3)$$

With constraints (A2), the parameters calculated in (A3) reduce to the “true” values if the fitting model (A1) is correct and there are no measurement errors. In fact, if for a set of parameters A' (the “true” values), the linear equation system $Y = VA'$ holds on all rows, then $A = gY = gVA' = IA' = A'$.

In general the left inverse g of $n \times m$ matrix V is not unique. For each parameter a_j , there are n values of g_{ji} that satisfy $(m+1)$ constraints which permits different values of g_{ji} to be chosen. Each set of g_{ji} values corresponds to a particular fitting algorithm. It can be shown that the least-squares fitting algorithm is a special case of the generic algorithms.

We skip detailed deriving process of least-squares fitting and directly write down the final result:

$$A = (V^T V)^{-1} V^T Y \quad a_j = \sum_{k=0}^m \zeta_{jk} \sum_{i=1}^n v_{ik} y_i \quad \{\zeta_{jk}\} = (V^T V)^{-1} \quad (A4)$$

We use G to represent the algorithm matrix of the least-squares fitting. Comparing with (A3), it can be seen that:

$$G = (V^T V)^{-1} V^T \quad G_{ji} = \sum_{k=0}^m \zeta_{jk} v_{ik} \quad (A5)$$

Clearly matrix G satisfies the left inverse constraints given in (A2):

$$GV = (V^T V)^{-1} V^T V = I \quad (A6)$$

The least-squares algorithm is chosen as a reference. All the other generic fitting algorithms are viewed as a deviation from the least-squares algorithm:

$$g = G + \Delta g \quad g_{ji} = G_{ji} + \Delta g_{ji} \quad (A7)$$

It can be seen from (A7) and (A2) that the row space of Δg is orthogonal to the column space of V :

$$\Delta g V = 0 \quad \sum_{i=1}^n (\Delta g_{ji}) v_{ik} = 0 \quad j, k = 0, 1, \dots, m \quad (A8)$$

The equation (A8) is true for the difference of any two algorithms but we will use Δg to denote only the difference between an arbitrary algorithm and the least-squares algorithm.

Theorem 1: The row space of difference matrix Δg of two algorithms is orthogonal to the row space of the algorithm matrix G of the least-squares fitting:

$$\Delta g G^T = 0 \quad (\text{A9})$$

Combining (A5) and (A8) gives (A9). In fact, (A5) describes that the row space of G is in the column space of V and according to (A8) the row space of Δg is orthogonal to any vectors in the column space of V including row vectors of G .

In order to visualize Theorem 1 in a picture, consider a linear fitting with three data points: (x_1, y_1) , (x_2, y_2) and (x_3, y_3) . The two columns in matrix V can be shown as two vectors in the R^3 space: $v_0 = (x_1^0, x_2^0, x_3^0)^T = (1, 1, 1)^T$ and $v_1 = (x_1^1, x_2^1, x_3^1)^T$ as shown in Fig. 5.

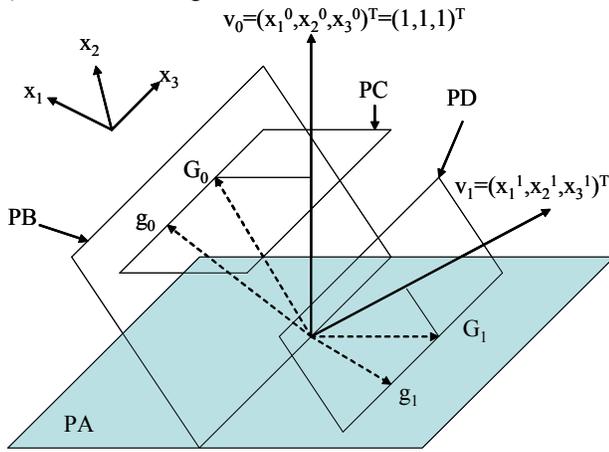


Fig. 5. The algorithm vectors

The row vector g_1 of algorithm matrix g must satisfy the constraints $g_1 v_0 = 0$ which is the equation of plane PA that is perpendicular to v_0 and passes the origin. The constraint $g_1 v_1 = 1$ is the equation of plane PD that is perpendicular to v_1 and passes point $v_1 / (v_1^T v_1)^2$. The vector g_1 can only take values in the intersection of PA and PD. Similarly, g_0 can only take values in the intersection of PB and PC.

The least-squares algorithm corresponds to the choice of G_0 and G_1 which are co-planar with v_0 and v_1 and have shortest lengths. For any other algorithms, the algorithm vector g_0 and g_1 have longer lengths. The differences $(g_0 - G_0)$ and $(g_1 - G_1)$ are orthogonal to G_0 and G_1 .

Now consider fitting error for each of the parameters a_0, a_1 to a_m . If the y -measurements are independents and they have a same standard deviation δy , the variations of the parameters can be written:

$$(\delta \alpha_j)^2 = (\delta y)^2 \sum_{i=1}^n (g_{ji})^2 \quad (\text{A10})$$

The square-sum in (A10) is actually the length of the row vector of the matrix g . It contains contributions from the least-squares algorithm and the additions from the deviation:

$$\sum_{i=1}^n (g_{ji})^2 = \sum_{i=1}^n (G_{ji} + \Delta g_{ji})^2 = \sum_{i=1}^n G_{ji}^2 + \sum_{i=1}^n 2G_{ji} \Delta g_{ji} + \sum_{i=1}^n \Delta g_{ji}^2 \quad (\text{A11})$$

The middle sum term in (A11) vanishes because from (A9),

the row spaces of Δg and G are orthogonal. The variations of the fitting parameters due to measurement errors are simply composed with two parts:

$$(\delta \alpha_j)^2 = (\delta y)^2 \sum_{i=1}^n (G_{ji})^2 + (\delta y)^2 \sum_{i=1}^n (\Delta g_{ji})^2 \quad (\text{A12})$$

The equation (A12) reflects a fact that the variation of the parameters is a minimum when the least-squares fitting algorithm is used. Around this point, the fitting errors are relatively insensitive to the change of g_{ji} values allowing the user to choose different g_{ji} values to reduce total computations without increasing fitting errors significantly. If, for example, all g_{ji} values differ from G_{ji} values by 10%, the error of each parameter increases by only about 0.5% from the minimum value.

The relative fitting errors of the curvature parameter η , (i.e., a_2) with 17 measurement points for various algorithm deviations are shown in Fig. 6. This is the case given in the columns with half length of the track = 16 in Table I.

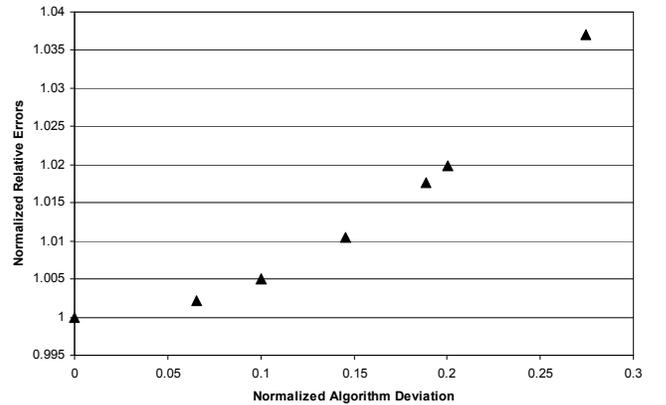


Fig. 6. Relative parameter fitting errors vs. the algorithm deviation

The algorithm deviations are normalized by dividing the algorithm length of the least-squares fitting, i.e., the horizontal axis in Fig. 6 is square root of $\sum \Delta g_{2i}^2 / \sum G_{2i}^2$. The fitting errors are also normalized with the least-squares fitting result given in the first column of Table I. The 17 independent measurement points construct a R^{17} space. The 3 constraints that g_{2i} must satisfy restrict the algorithm vector on a 14-dimensional hyper-plane. The parameters g_{2i} (i.e., the $e[i]$ values in second column of Table I) are chosen randomly deviating from the least-squares fitting algorithm. The relative fitting errors behave as suggested in (A12).

REFERENCES

- [1] Kulyavtsev et al., BTeV proposal, Fermilab, May 2000, available: {<http://www-btev.fnal.gov/DocDB/0000/000066/002/index.html>}
- [2] E.E. Gottschalk, BTeV detached vertex trigger, Nucl. Instrum. Meth. A 473 (2001) 167.
- [3] J. Wu et al., "The application of tiny triplet finder (TTF) in BTeV pixel trigger", IEEE Tans. Nuclear Science, vol. 53, p 671, 2006.
- [4] J. Wu et al., "Integrated Upstream Parasitic Event Building Architecture for BTeV Level 1 Pixel Trigger System", IEEE Tans. Nuclear Science, vol. 53, p 1039, 2006.
- [5] Altera Corporation, "Cyclone FPGA Family Data Sheet", (2003) available via: {<http://www.altera.com/>}