Status of Pile-up Task

(CMS)

C. Civinini, L. Silvestris and A. Tricomi
INFN

Aida WP2 - INFN contribution

- Our contribution to WP2:
  - Development of a toolkit to handle high multiplicity events

- Pile up in SLHC will increase with respect to "nominal (2011)" LHC → more efficient way needed to manage high particle mutiplicity events

  - Working on improving CMS Mixing Module and Tracking software for post-long shoutdown 1 (2015) and Tracker Phase 1 upgrade (2017)

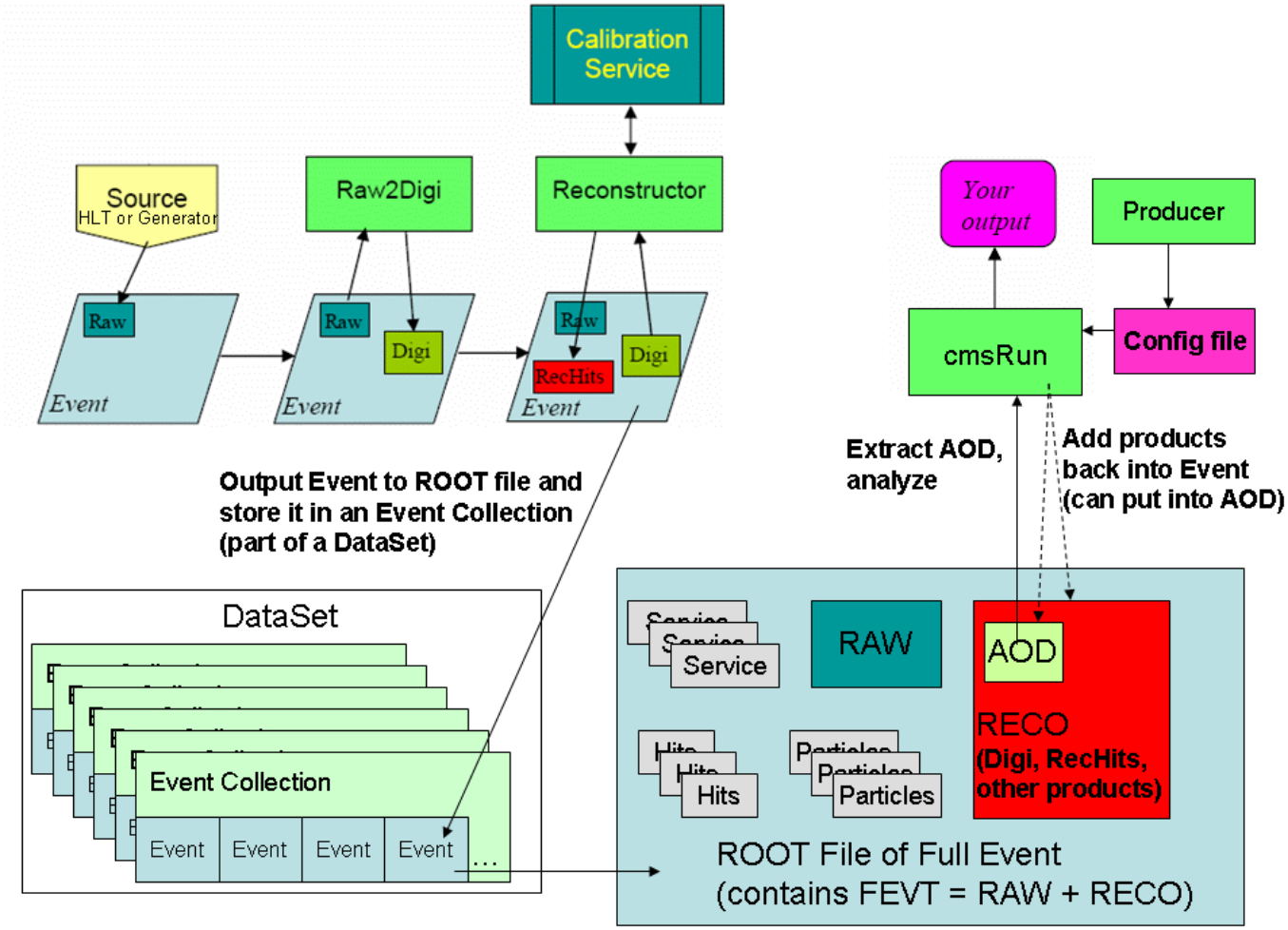
Outline

- A glimpse of the CMS Processing Model
- Mixing Module improvements from 2011 to 2013
- A glimpse on the CMS Tracking implementation
- The tracking evolution from 2011 to 2012
- The challenge of 2015 data taking
- Raw ideas for new tracking algorithms

- For help and material, many thanks to several Tracking group, Offline and PPD team people (CMS Collaboration)

CMS: Processing Model

https://twiki.cern.ch/twiki/bin/view/CMSPublic/WorkBookCMSSWFramework



Modules are configurable and communicate via the Event

Software Components in MC Production

- Workflows for MC production in CMS include the following components;
  - Event Generation
  - Full Detector simulation: SimHits production using Geant4
  - Mixing Module: software for superimposing secondary pile-up (in time and out of time) events to a signal event
  - Digitization: software for modelling electronics response in the different detectors
  - Detector reconstruction (Tracks, ECAL Reco, …)
- Increasing the number of pile-up events: Mixing Module and Reconstruction code (mainly tracking code) need to be optimized

Mixing Module Improvements from 2011 to 2013

- Mixing module superimpose pileup events to a signal event
- 2011 implementation:
  - copies the signal data into the CrossingFrame at bunchcrossing 0
  - Loop over all bunch-crossing (depending from the configuration)
  - For each bunch-crossing it decides which number of events and from which source should be added
  - For each bunch-crossing and for each type of data (PSimHits, PCaloHits,..) it adds the corresponding objects from the read event into a secondary (in memory) stream (Crossing Frame)
  - The Crossing Frame is then used during the digitization step, i.e. during next step in the MC Production
- Major drawback: memory increase linearly with the number of pile-up events up to 2GB/core with 100 pile-up events and 3 bunch-crossing (300 events)

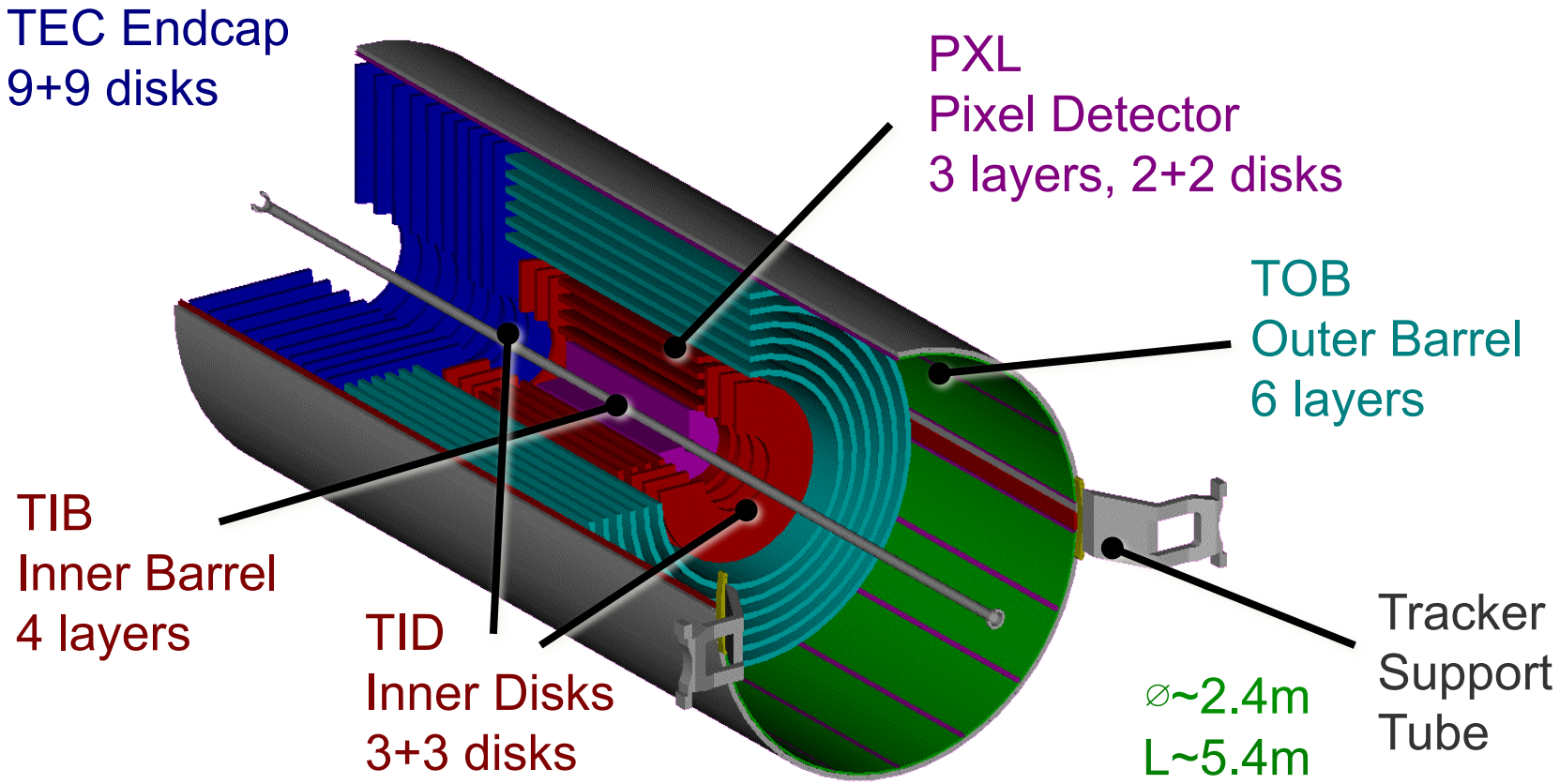
Mixing Module Improvements from 2011 to 2013

- Memory Improvements:
  - The current implementation has changed the MC production workflow
  - The mixing module and the digitization step are done in series
  - In this way we don't need to keep in memory the secondary stream (CrossingFrame)
  - Applying such trick we keep the memory under 2 GB/core with 140 pile-up events and 5 bunch crossing (700 events), i.e. future LHC configurations
- CPU improvements (additional improvements):
  - Now we are moving to study the CPU effects. Up to now the digitization time has been negligible respect to simulation and mixing time, but this will change soon…
    - PU 140 BX 5 (25ns): 700 event to mix → RSS 1.7 GB CPU 59 sec
    - PU 40 BX 15 (25ns): 600 events to mix → RSS 1.5 GB CPU 27 sec (worse by factor 3 for phase1)
    - PU 20 BX 15 (25ns): 300 events to mix → RSS 1.2 GB CPU 12 sec
    - Summer 12 MC Prod → 1 GB CPU 5 sec
  - These results are very very preliminary and most probably will bring additional changes in the MC production workflow; Keep tuned for next time.

The largest Silicon Tracker

Pixel Detector
66M channels
100x150 μm^2 pixel
LHC radiation resistant

Si-Strip detector
~23m^3; ~200m^2 of Si area;
~9x10^6 channels;
LHC radiation resistant

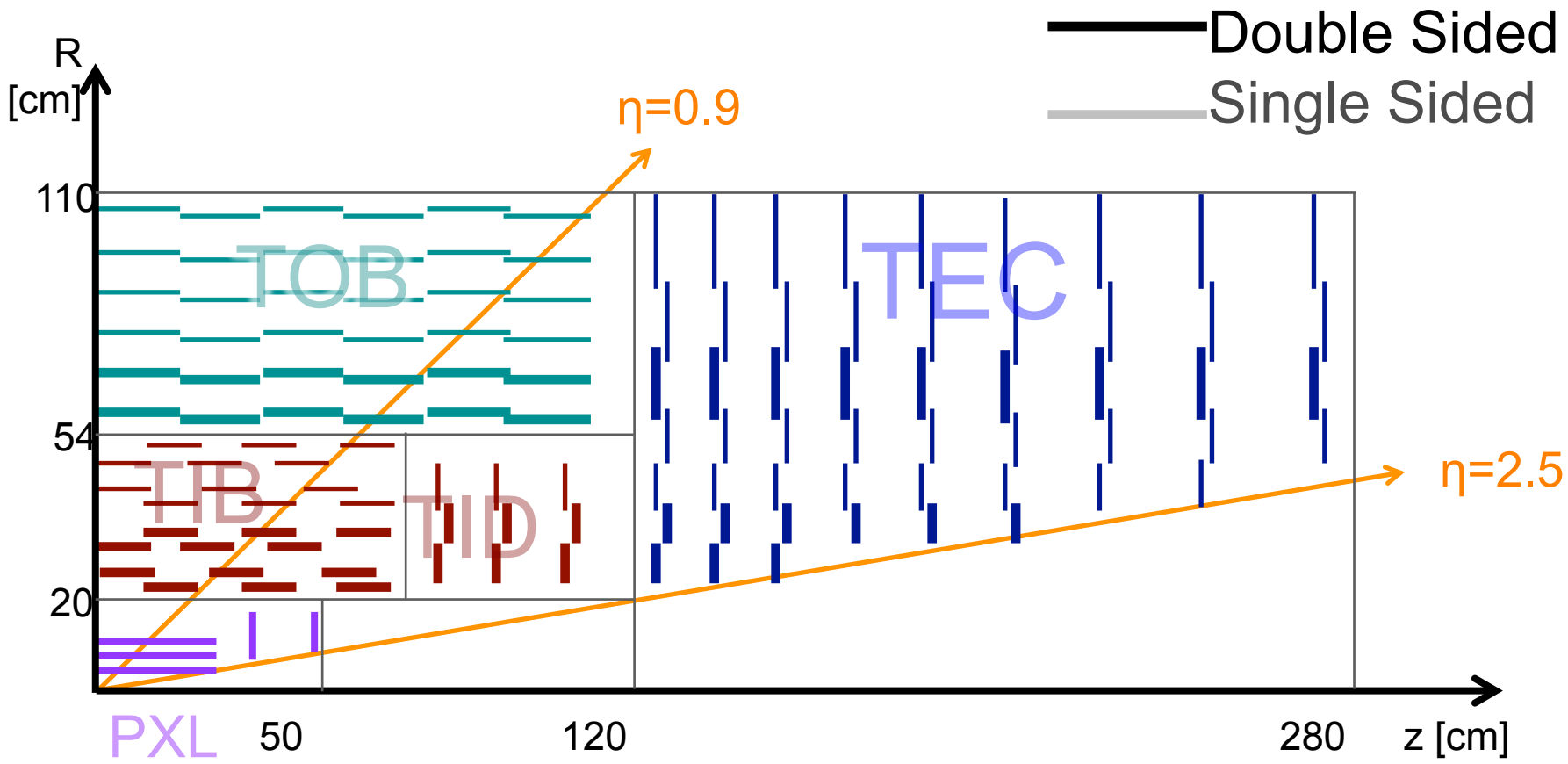


The CMS Silicon Tracker Layout

Basic Performances

$\sigma(P_T)/P_T$ ~1-2% (P_T~100 GeV/c)

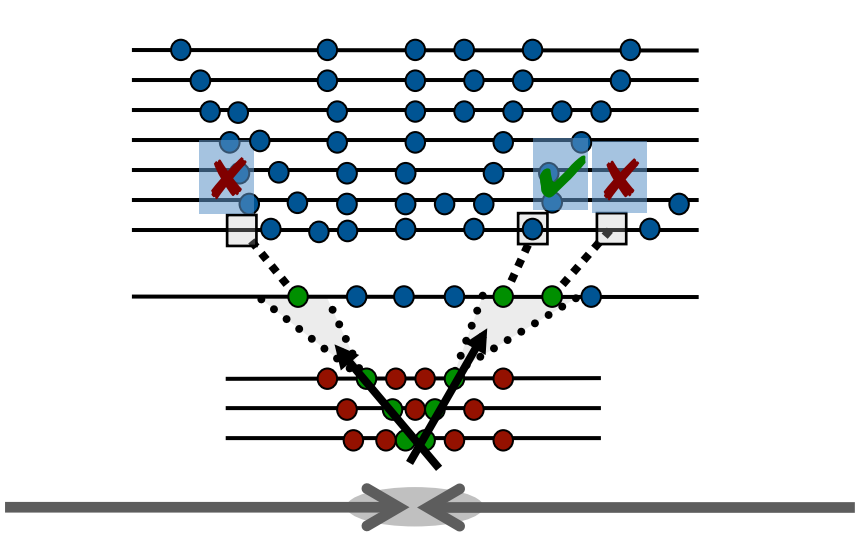
IP resolution ~10-20μm (P_T=100-10 GeV/c)

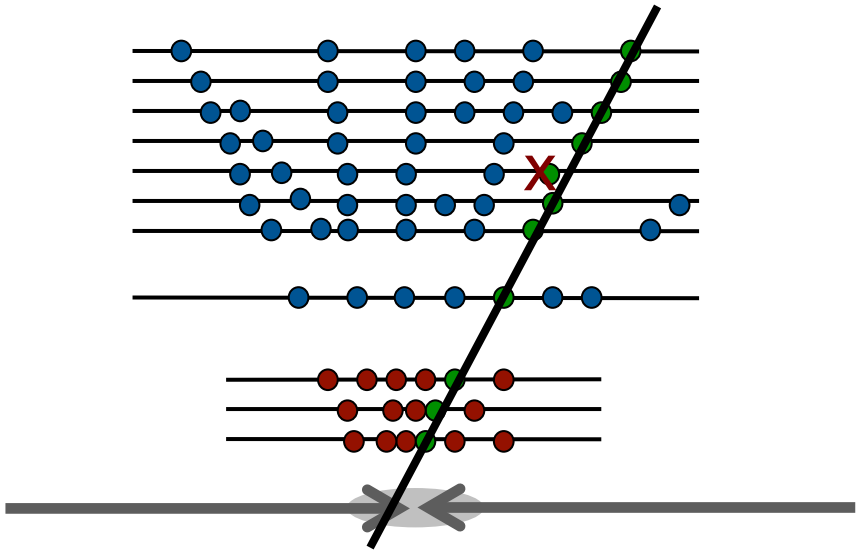


CMS tracking in a nutshell

Seeding starts from innermost pixel layers (pairs + PV, triplets). Inside-out trajectory building through pattern recognition (based on Kalman Filter).

Rejection of outlier hits and final fit, also based on Kalman Filter. Final quality selection of tracks. Primary Vertex used in tracking derived from pixel-based algorithm.

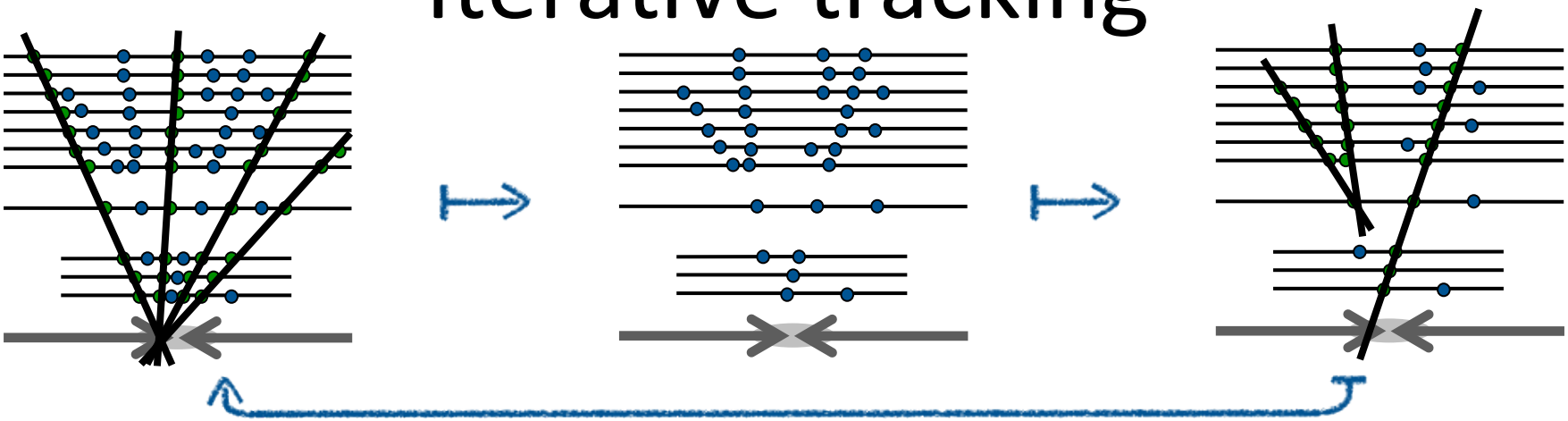




Track Parameters: $^{q}/_{p}$, η, ϕ, d_Z, d_{XY}

Parameters propagated through magnetic field inhomogeneities using **Runge-Kutta** propagator

Iterative tracking



The CMS tracking relies on iterations (steps) of the tracking procedure; each step works on the remaining not-yet-associated hits and is optimized with respect to the seeding topology and to the final quality cuts.

#step	seed type	seed subdetectors	$P_T^{\min}$ [GeV/c]	d_0 cut	z_0 cut
0	triplet	pixel	0.8	0.2 cm	3.0σ
1	pair	pixel/TEC	0.6	0.05 cm	0.6 cm
2	triplet	pixel	0.075	0.2 cm	3.3σ
3	triplet	pixel/TIB/TID/TEC	0.25-0.35	2.0 cm	10.0 cm
4	pair	TIB/TID/TEC	0.5	2.0 cm	12.0 cm
5	pair	TOB/TEC	0.6	6.0 cm	30.0 cm

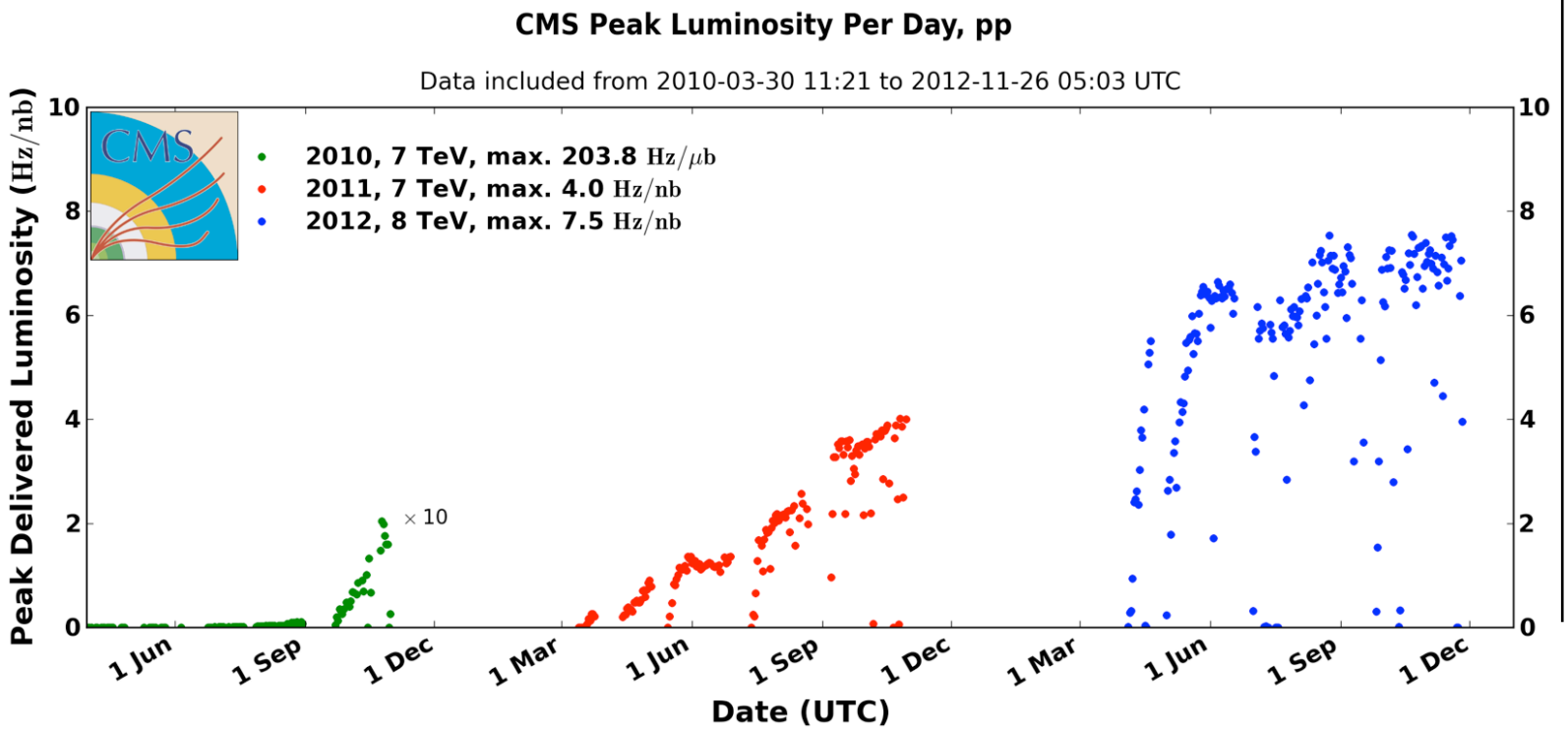
Iterative tracking in 2011 (CMSSW 42x)

Tracking evolution from
from $10^{32}/cm^2/s$ (2011)
to $8 \times 10^{33}/cm^2/s$ (2012)

.

The constraint of prompt reconstruction

Prompt reconstruction is crucial for a discovery experiment: quasi real-time physics results, fast deep feedback on detector conditions. It requires data to be processed at the same pace as they are produced. Resources and algorithm speed must adapt to the instantaneous luminosity. The tracking reconstruction software was too heavy (CPU time and memory) for prompt reconstruction and it was improved in two phases: fall 2011, spring 2012.



20

2015 estimates

Fall 2011 campaign: from CMSSW42x to 44x (1)

Several optimization in object reconstruction like photon conversion, vertices, nuclear interactions with significant CPU time gain

Iterative tracking A factor 2.5 of improvement in the CPU time has been obtained by optimizing the iterative tracking. The net effect is an increase of the effective P_T threshold for track reconstruction together with tighter constraints on impact parameter. This configuration results in a reduced efficiency for P_T <300MeV/c but an efficiency for P_T>0.9 GeV/c larger by ~1% with a ~8% reduction of the fake rate.

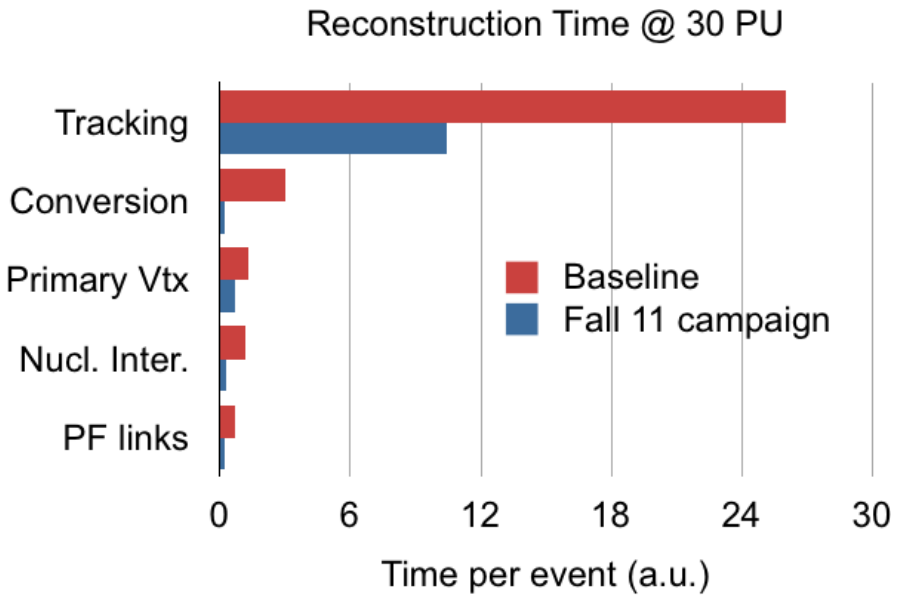
#step	seed type	seed subdetectors	$P_T^{\min}$ [GeV/c]	d_0 cut	z_0 cut
0	triplet	pixel	**0.6**	**0.03** cm	**4.0**σ
1	**triplet**	**pixel**	**0.2**	**0.03** cm	**4.0**σ
2	pair	**pixel**	0.6	**0.01** cm	**0.09** cm
3	triplet	pixel	**0.2**	**1.0** cm	**4.0**σ
4	triplet	pixel/TIB/TID/TEC	**0.35-0.5**	2.0 cm	10.0 cm
5	pair	TIB/TID/TEC	**0.6**	2.0 cm	**10.0** cm
6	pair	TOB/TEC	0.6	2.0 cm	30.0 cm

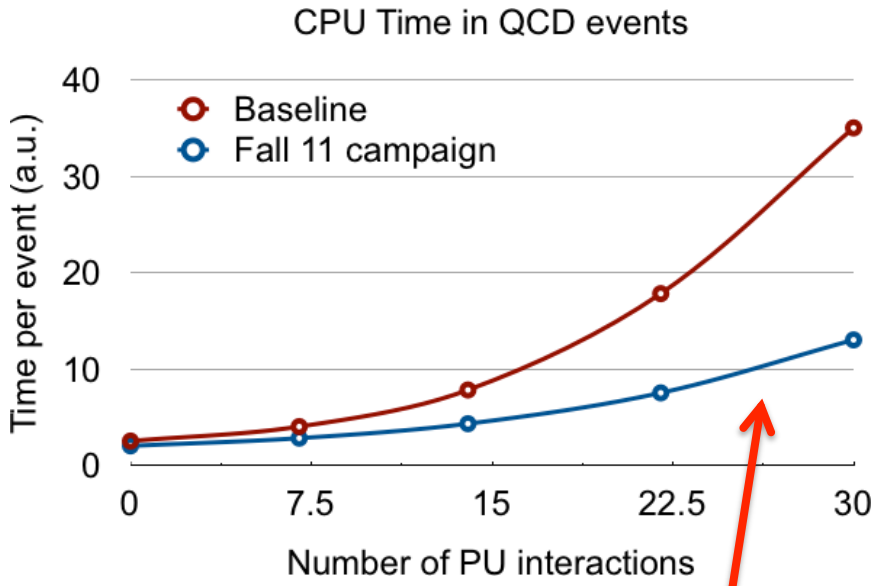
Iterative tracking in late 2011 (CMSSW 44x) / In **bold** the changes with respect to 42x

Results of fall 2011 campaign

reconstruction CPU time @30PU | reconstruction CPU time vs. PU

Simulated QCD events





First big CPU improvements (end 2011)

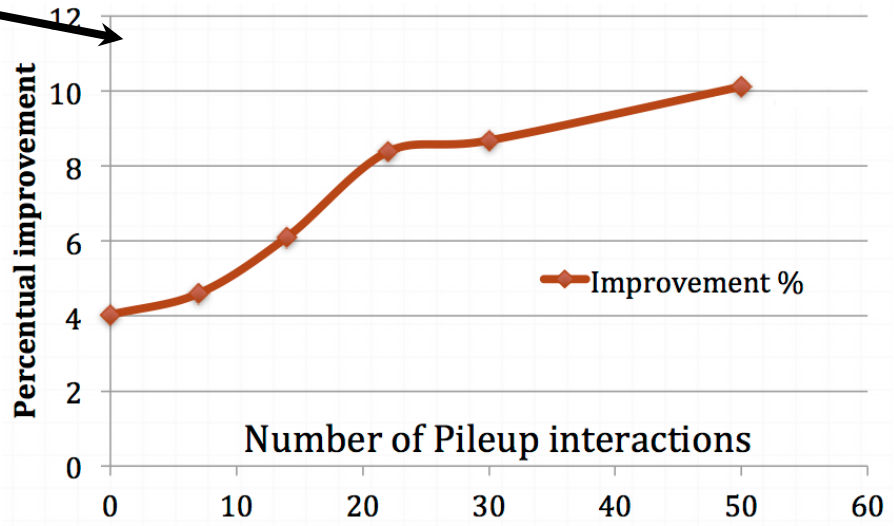
Spring 2012 campaign: from CMSSW44x to 52x (1)

Change of compiler switch from gcc 4.3.4 to gcc 4.6.2: faster code generated (compiler specific optimizations), C++11 support and autovectorization

JEMalloc standard malloc replaced by JEMalloc, highly performant and able to better redeem memory

Improved ROOT version from 5.27 to 5.32 that features several improvements, especially in I/O with less memory required.

Relative change of CPU reconstruction time vs. PU Simulated QCD events



Several design modifications to improve speed and memory consumption; for example, 10% gain in speed and in some 100MB of resident set size (RSS) saved per event from the devirtualization of the BasicTrajectoryState class (an ancillary class for track reconstruction); stereo hit class reduced a factor three in size with RSS memory down to 50MB from 150MB

Spring 2012 campaign: from CMSSW44x to 52x (2)

Offline vertexing based on a deterministic annealing algorithm improved: loops autovectorized (new compiler), exponential functions replaced with fast autovectorizable inlined double precision versions; some configuration parameters optimized. 3x gain in CPU time with no change in performances

Cluster-shape based seed filtering extended to almost all seeding step. 1.5x improvement in CPU time. Fake rate is reduced by ~ 20%. Iterative tracking Tiny optimization plus upgrade of the final track cleaning and selection criteria. No efficiency change for prompts tracks with P_T>0.9 GeV/c, but fake rate ~35% down.

#step	seed type	seed subdetectors	$P_T^{\min}$ [GeV/c]	d_0 cut	z_0 cut
0	triplet	pixel	0.6	**0.02** cm	4.0σ
1	triplet	pixel	0.2	**0.02** cm	4.0σ
2	pair	pixel	0.6	**0.015** cm	0.09 cm
3	triplet	pixel	**0.3**	**1.5** cm	**2.5**σ
4	triplet	pixel/TIB/TID/TEC	**0.5-0.6**	**1.5** cm	10.0 cm
5	pair	TIB/TID/TEC	0.6	2.0 cm	10.0 cm
6	pair	TOB/TEC	0.6	2.0 cm	30.0 cm

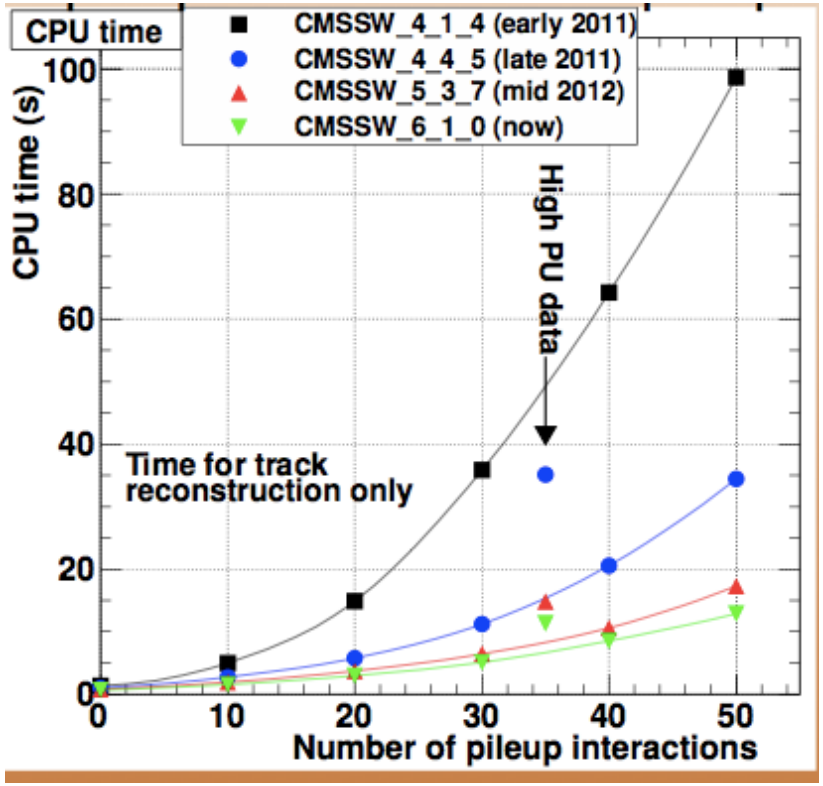
Iterative tracking in 2012 (CMSSW 52x) / In **bold** the changes with respect to 44x

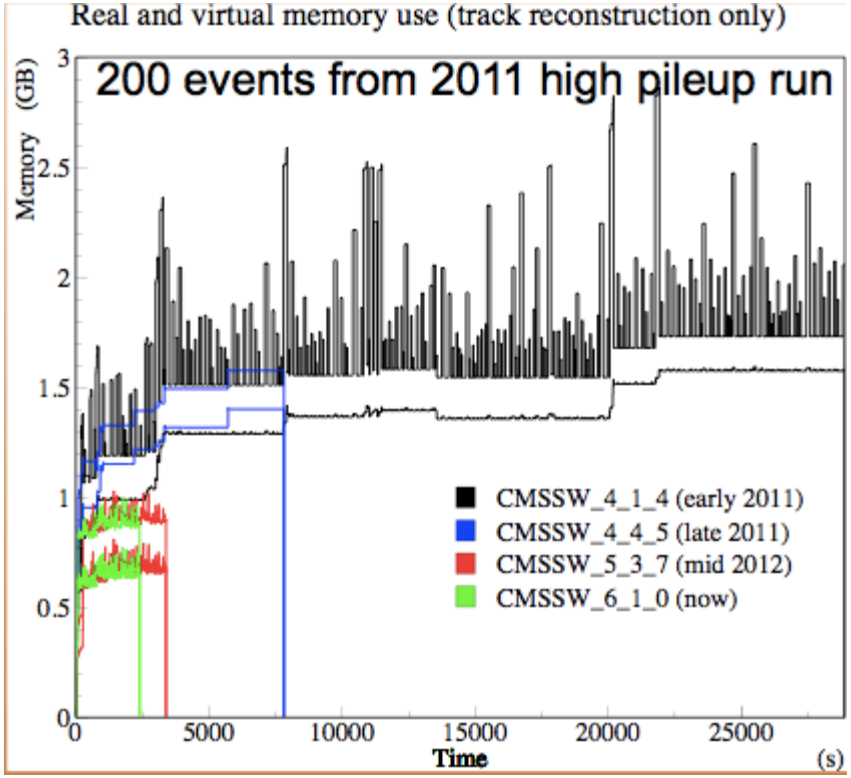
Effects of CPU and memory improvements

Many improvements over past 2 years have yielded substantial saving in CPU and memory use.

But CPU time still shows signification non-linearities with pile-up

Results for track reconstruction only; data results from 2011 high pile-up run with ~ 35 pile-up





Additional big improvements in CPU and Memory

The challenge of 2015 data taking

Potential Performance after LS1

- Determined by the performance of the injector chain
- Different collimator scenarios, not detailed here
- LHC Injector Upgrade (LIU) fruits after LS2

J. Uythoven
CMS week Lisbon

	Number of bunches	β* [m]	Half X-angle [μrad]	Ib SPS	Emit SPS [um]	Peak Lumi [cm^{-2}s^{-1}]	~Pile-up	Int. Lumi [fb^{-1}]
25 ns	2800	0.50	190	1.2e11	2.8	1.1e34	23	~30
50 ns	1380	0.40	140	1.7e11	2.1	1.8e34 β* level	81 β* level	?
25 ns low emit	2600	0.40	150	1.15e11	1.4	2.0e34	48	52
50 ns low emit	1200	0.40	120	1.71e11	1.5	2.2e34	113	?

Presently at 4 TeV, β = 0.6 m, half X-angle 145 μrad*

Strategies for 2015 and Phase 1

Generic improvements as in 2011/2012 (smarter coding, compilers, seed cleaning) and iterative tracking tuning | tracking developers

Tracking code reengineering; major redesign of the tracking code to implement parallelization and vectorization between offline people for the framework (modifications almost transparent for the user) and tracking developers (for modifications to be implemented straight into the tracking code)

New tracking algorithms (Hough transform)

Developments to be done during LS1 but mainly in 2013, with 2014 devoted to full validation and MC productions

Proposal for Hough transform applications in CMS

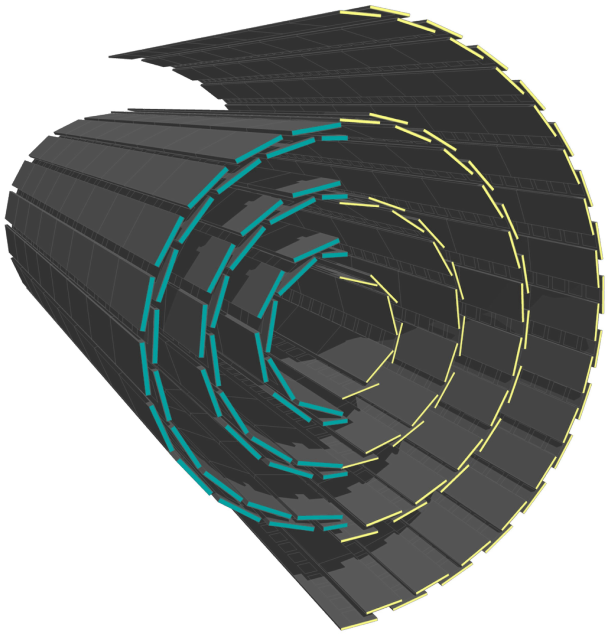
Hough transform methods cannot handle energy loss and multiple scattering; they are probably not suitable for full track reconstruction in CMS (where material effects are substantial).

Nevertheless, Hough transform could represent a natural way to combine more information than just two/three hits at the seeding level in a fast way and without entering in the time consuming propagation. Given the reduced lever arm and the reduced resolution needed, material effects can be probably neglected at the seeding level.

Proposal for Hough transform method implementation:

= seeding in the outer tracker layers combining information from more than three layers;

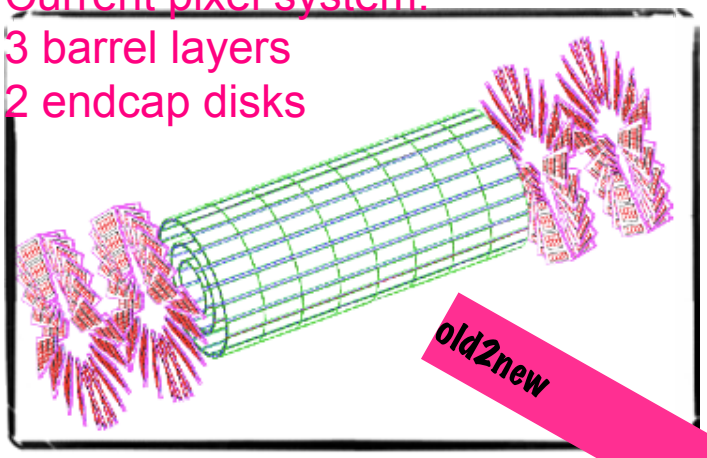
= 4-layer seeding for the phase-I upgraded pixel detector.



Pixel Phase 1 Upgrade

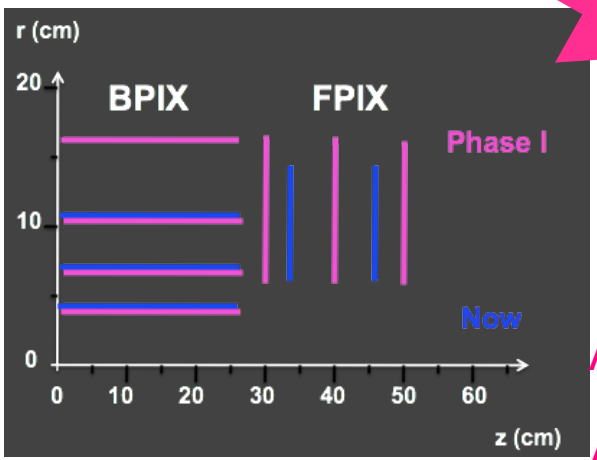
CMS Pixel @ Phase I

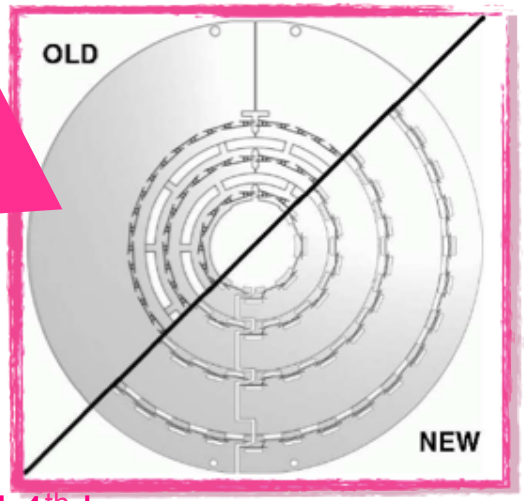
Current pixel system:
3 barrel layers
2 endcap disks



Current system designed to withstand $\mathcal{L}$ ~200-300 fb^{-1}
Significant radiation damage @ $\mathcal{L}$>10^{34}cm^{-2}s^{-1}
1st layer - 16% inefficient @ 2x10^{34}cm^{-2}s^{-1}

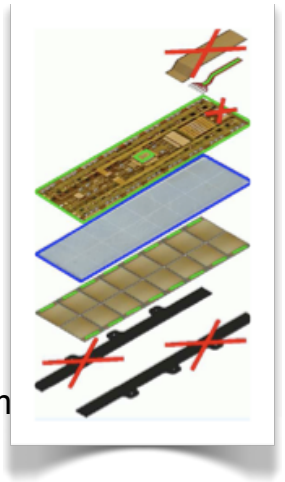
old2new





Add 4th layer

Add 3rd disc
Aggressive material reduction



Module
Thinner sensor
285µm to 225µm
Thinner ROC
175µm to 75µm
No HV capacitor
Minimise SMD components
Micro-twisted pair
No base strips
ONE TYPE ONLY

Upgrade Iterative Tracking (Stdgeom)

- 5_2_0 tracking for current pixel geometry (from "2012 tune")
  - Close to 5_2_0 tracking, use steps 0-2, and 4A (for high eta)
  - Reduce step 4A d_0 cut to reduce CPU and memory usage

Release CMSSW_4_2_8_SLHCstd2_patch1 Tracking steps

Iteration	Seeds	p_T cut (GeV)	d_0 cut (cm)	d_z cut (cm)	Min hits
0	pixel triplets	0.6	0.02	$4.0\sigma_{bs}$	3
1	low p_T pixel triplets	0.2	0.02	$4.0\sigma_{bs}$	3
2	pixel pairs with vtx	0.6	0.015	$4.0\sigma_{bs}$	3
3	detached triplets	0.3	1.5	15.0	3
4A	pixel +(TEC(1 ring)) triplets	0.4	**0.02**	10.0	3
4B	BPIX+TIB triplets	0.6	1.5	10.0	3
5	TIB, TID, TEC pairs (fewer)	0.7	2.0	10.0	4
6	TOB, TEC pairs	0.6	6.0	30.0	6

Upgrade Iterative Tracking (Phase 1)

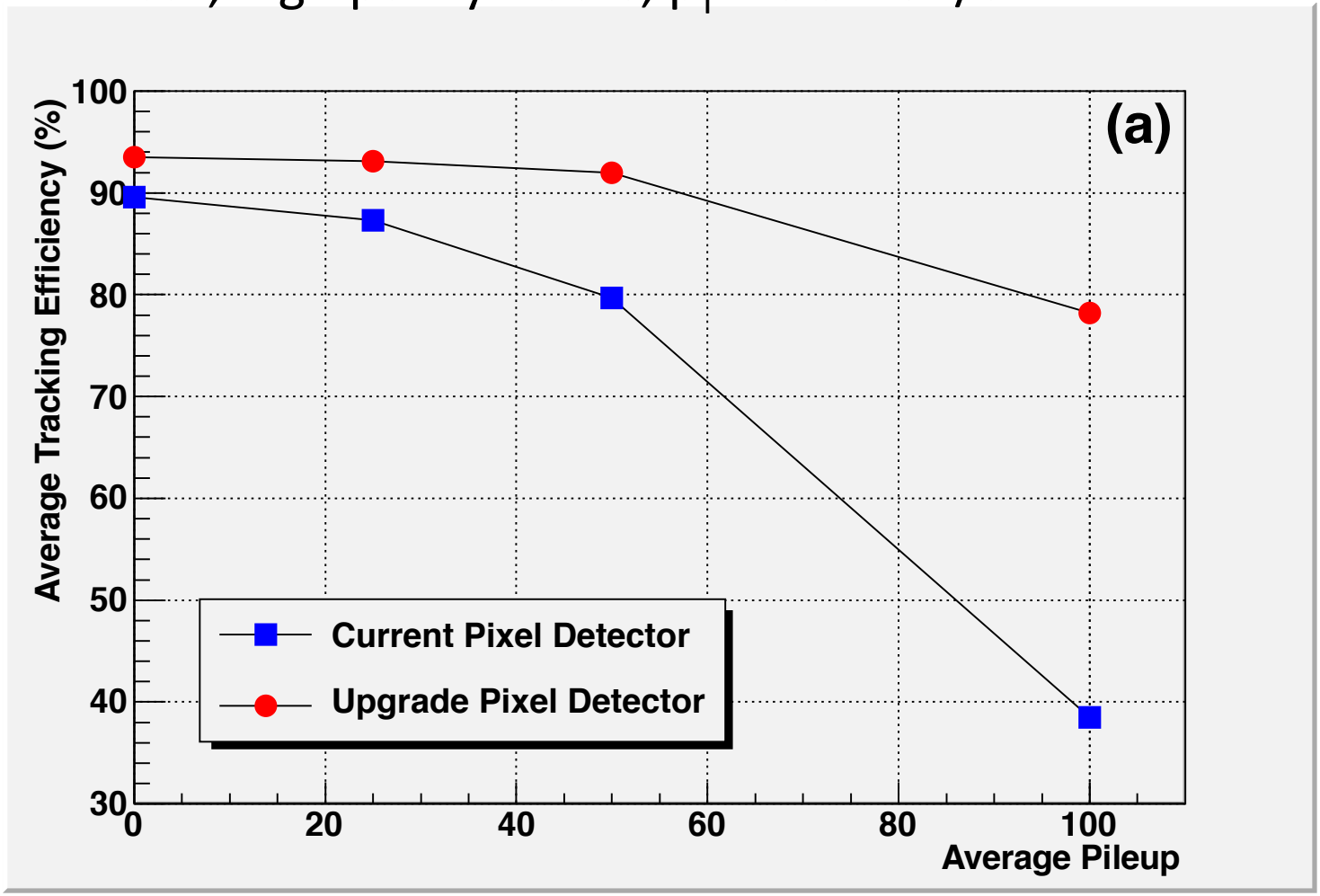
- 5_2_0 tracking for Phase 1 geometry (not optimized)
  - Make close to 5_2_0 tracking, use steps 0-2, and 4A, add step "-1"
  - Step 3 (pixel pairs) to recover efficiency in eta ~1.2–1.4 region

Release CMSSW_4_2_8_SLHCtk3_patch1 Tracking steps

Iteration	Seeds	p_T cut (GeV)	d_0 cut (cm)	d_z cut (cm)	Min hits
0	pixel quadruplets	0.6	0.02	$4.0\sigma_{bs}$	3
1	pixel triplets	0.6	0.02	$4.0\sigma_{bs}$	3
2	low p_T pixel triplets	0.2	0.02	$4.0\sigma_{bs}$	3
3	pixel pairs with vtx	0.6	0.015	$4.0\sigma_{bs}$	3
3old	detached triplets	0.3	1.5	15.0	3
4A	pixel +(TEC(1 ring)) triplets	0.4	**0.02**	10.0	3
4B	BPIX+TIB triplets	0.6	1.5	10.0	3
5	TIB, TID, TEC pairs (fewer)	0.7	2.0	10.0	4
6	TOB, TEC pairs	0.6	6.0	30.0	6

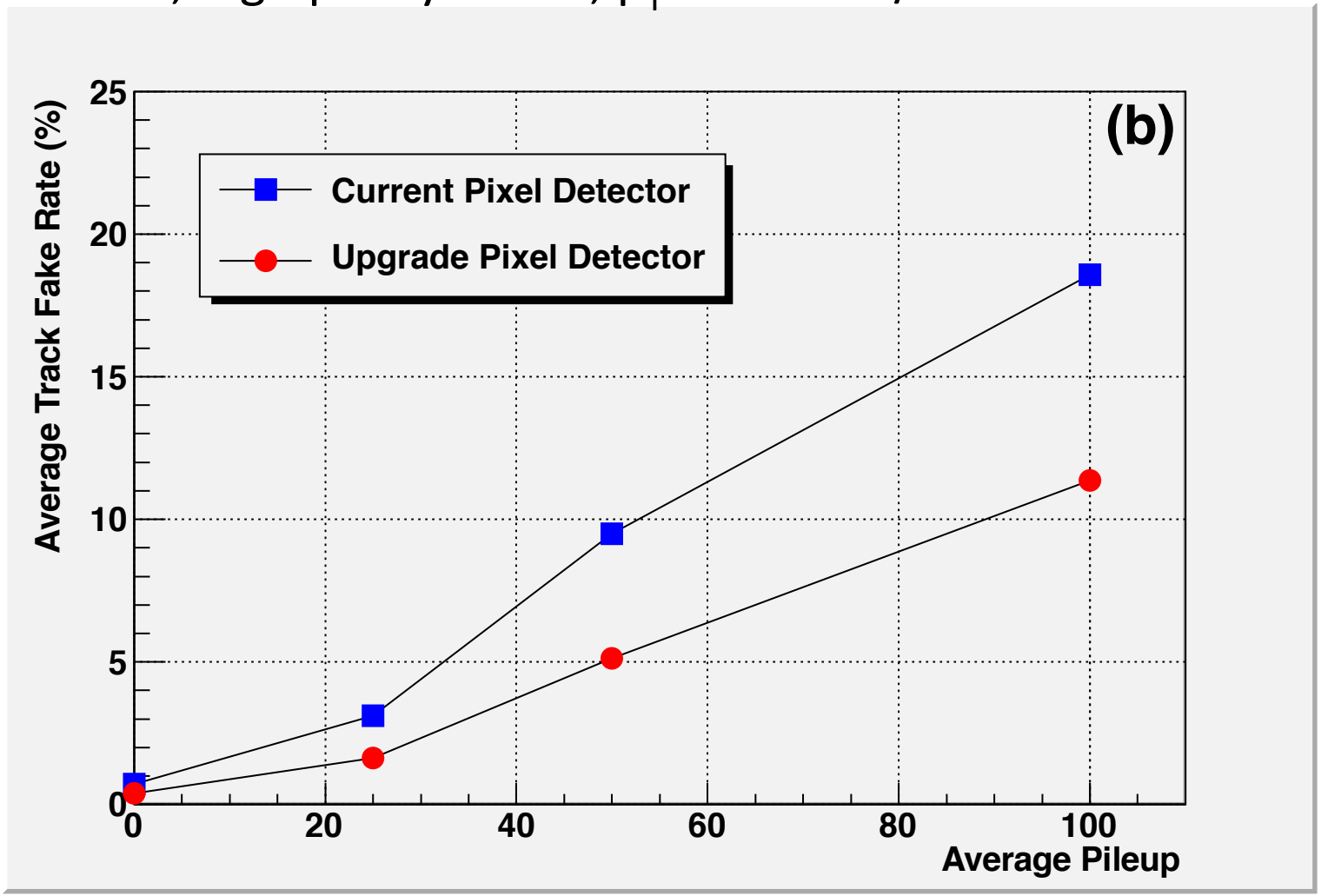
Performance: Tracking vs PU

- Average tracking efficiencies vs PU
  - ttbar, high purity tracks, $p_T > 0.9$ GeV/c



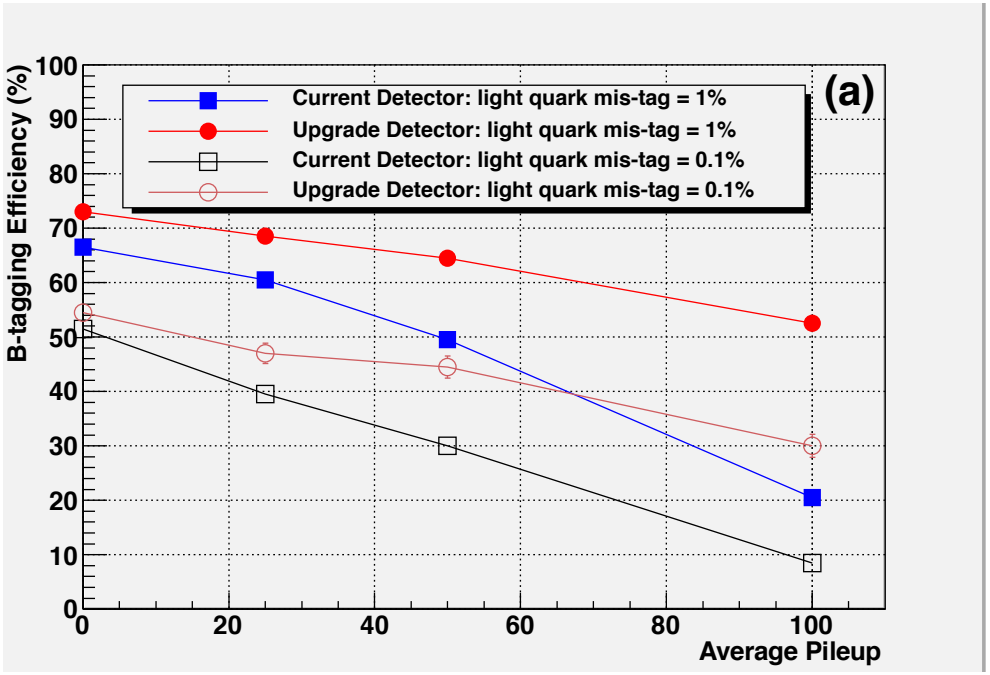
Performance: Tracking vs PU

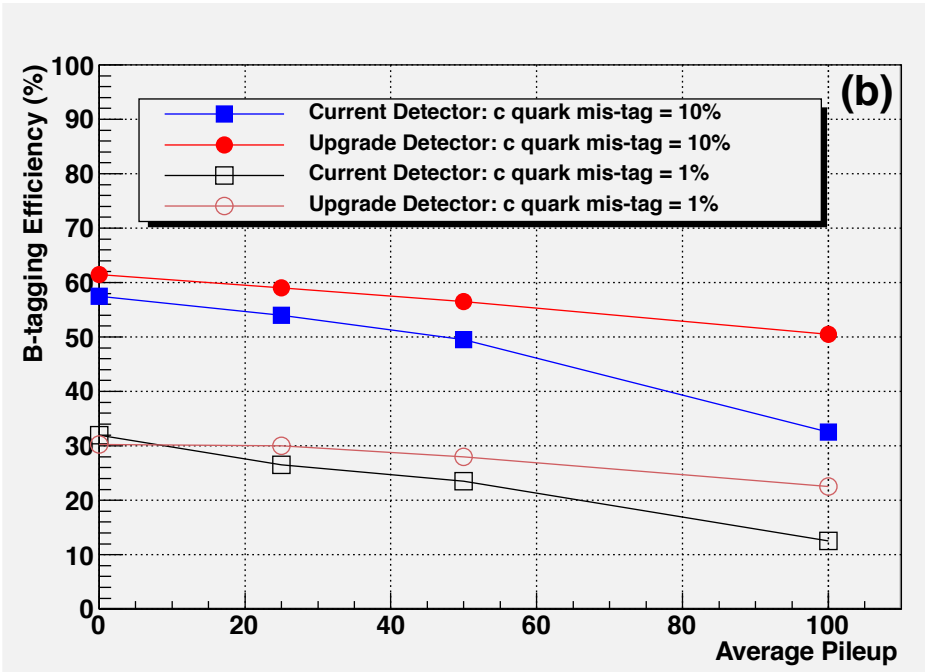
- Average track fake rates vs PU
  - ttbar, high purity tracks, $p_T > 0.9$ GeV/c



B-tagging Performance vs PU

- ttbar, CSV tagger, compare current and upgrade, <PU>=50
  - ak5PFjets PFnoPU, jet p_T > 30 GeV, DUS,b jets





- Much better handling high Pile-Up

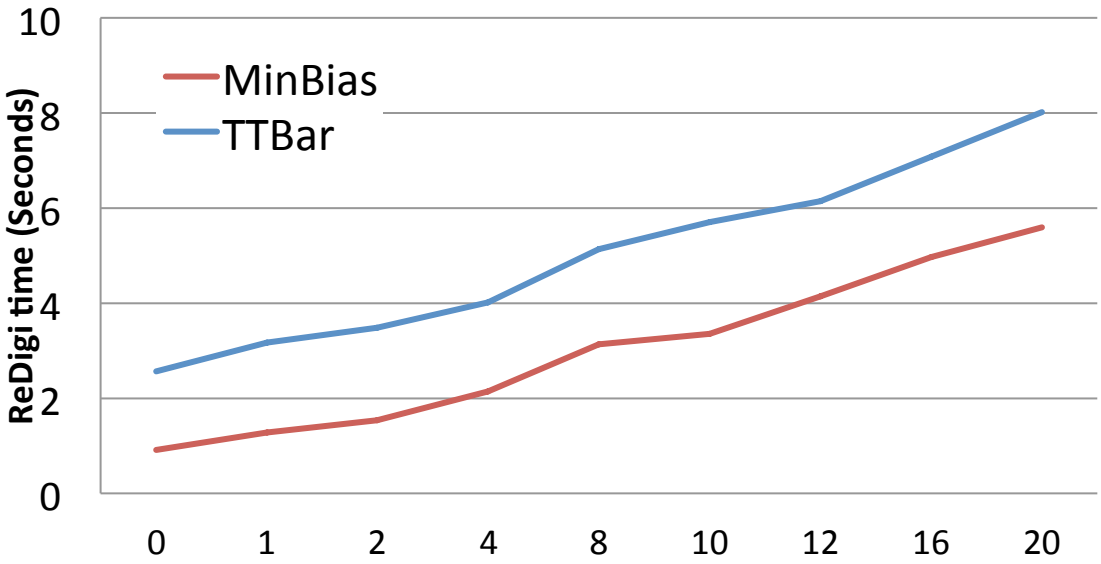
Future activities

- April – May 2013
  - New person fully dedicated to AIDA project will be hired
  - New person will be fully involved in the tracking code developing for 2015 data taking and phase 1 and phase 2 preparation
- May 2013 – Jul 2013
  - Identify components that need to be optimized and/or redesigned
  - Prioritize list of components
  - Start the iterative procedure for each component according to the priority list
- From Aug 2013 - *for each component the following step will be done*
  - Prototype
  - Integration into the CMSSW
  - Validation of the Physics performance

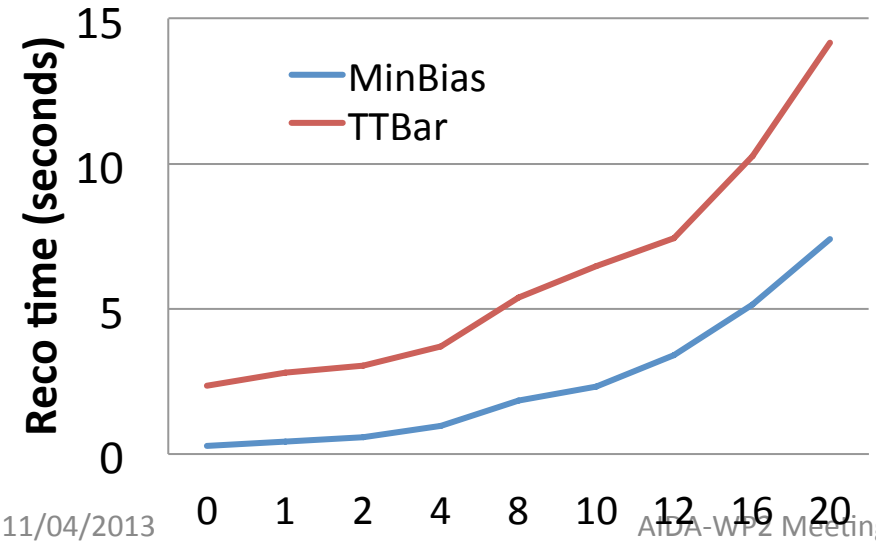
Back up

CMSSW Performance: Digitization and Reconstruction

2011

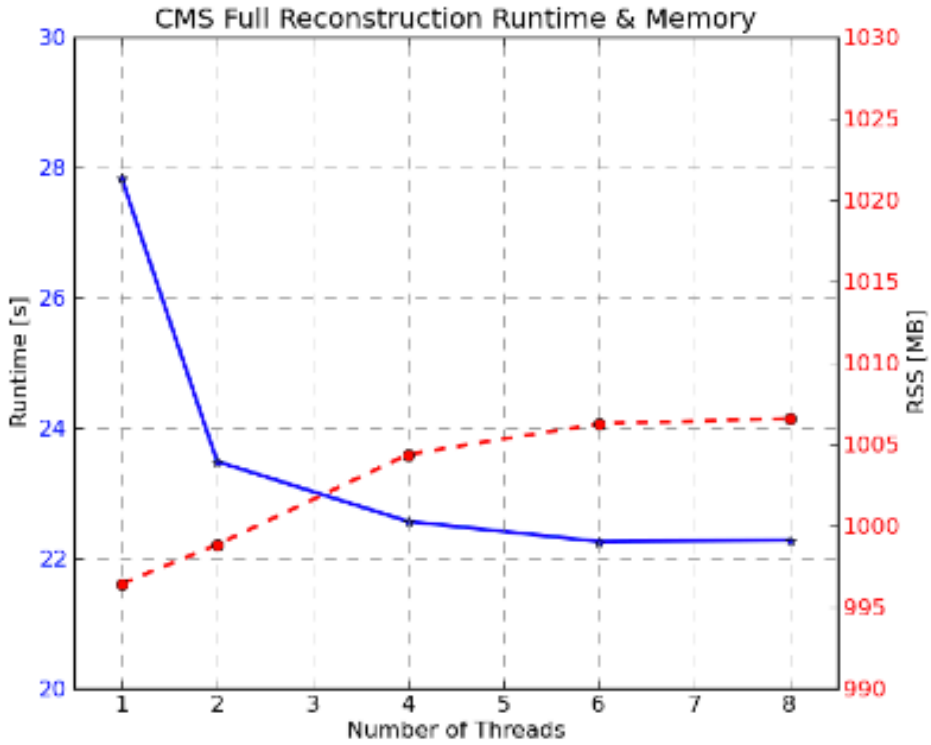


Timing includes: Digitization from SimHit, Packing to RAW format
Performance review ongoing up to 20 PU events



Timing based on MC
Track reconstruction accounts for ~ 40-50% total CPU time
Next to leading contributors: Particle Flow, conversions, Muon ID

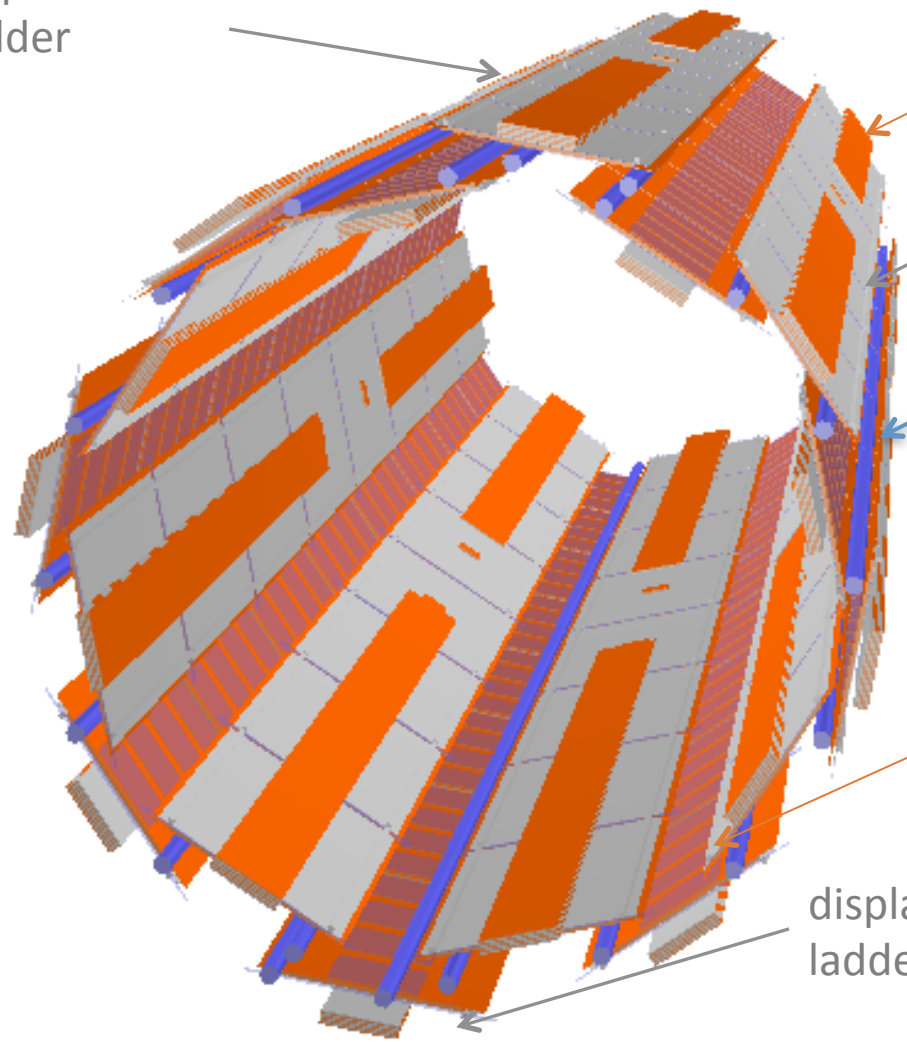
CMS Reconstruction Runtime and Memory



- Each thread adds about 1 MB to the overall memory consumption. Negligible compared to the memory footprint of the application (~ 1 GB) > lightweight scaling
- Higher-than-expected scaling from 1 to 2 cores, probably due to the positive effects of using the L1/L2 caches of two cores simultaneously

THauth

CMS-Phase1 new Pixels



First Barrel Layer
16 faces version

Data loss for Upgrade Studies

– Peak luminosity values

Current Detector	Radius (cm)	% Data loss at 1×10^{34} @25ns	% Data loss at 2×10^{34} @25ns	% Data loss at 2×10^{34} @50ns
BPIX1	4.4	4.0	16	50
BPIX2	7.3	1.5	5.8	18.2
BPIX3	10.2	0.7	3.0	9.3
FPIX1&2		0.7	3.0	9.3

Phase 1 Detector	Radius (cm)	% Data loss at 1×10^{34} @25ns	% Data loss at 2×10^{34} @25ns	% Data loss at 2×10^{34} @50ns
BPIX1	3.0	1.19	2.38	4.76
BPIX2	6.8	0.23	0.46	0.93
BPIX3	10.9	0.09	0.18	0.36
BPIX4	16.0	0.04	0.08	0.17
FPIX1-3		0.09	0.18	0.36

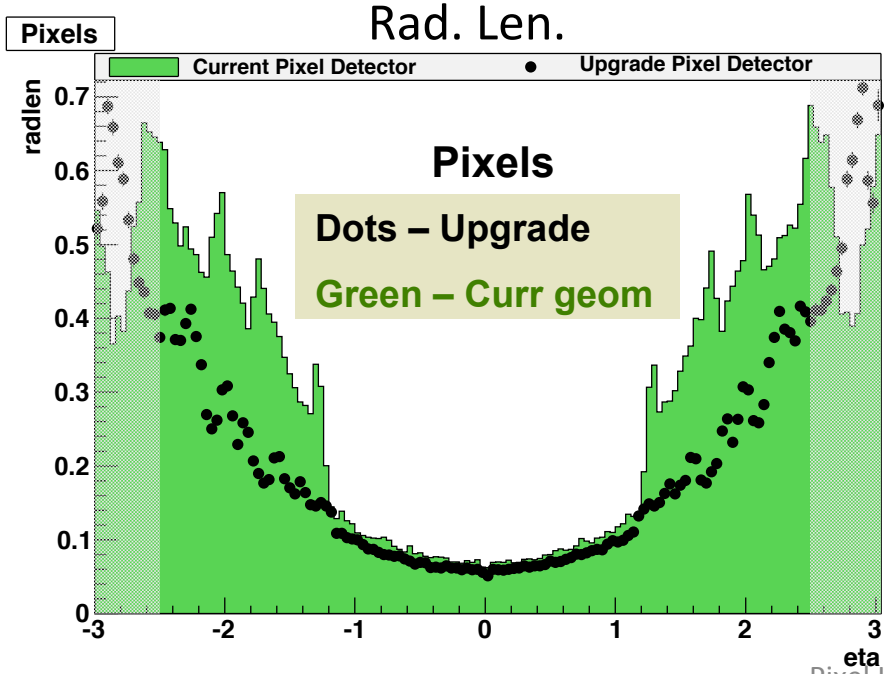
Pixel Upgrade Material Budget

Reduced material even with more layers

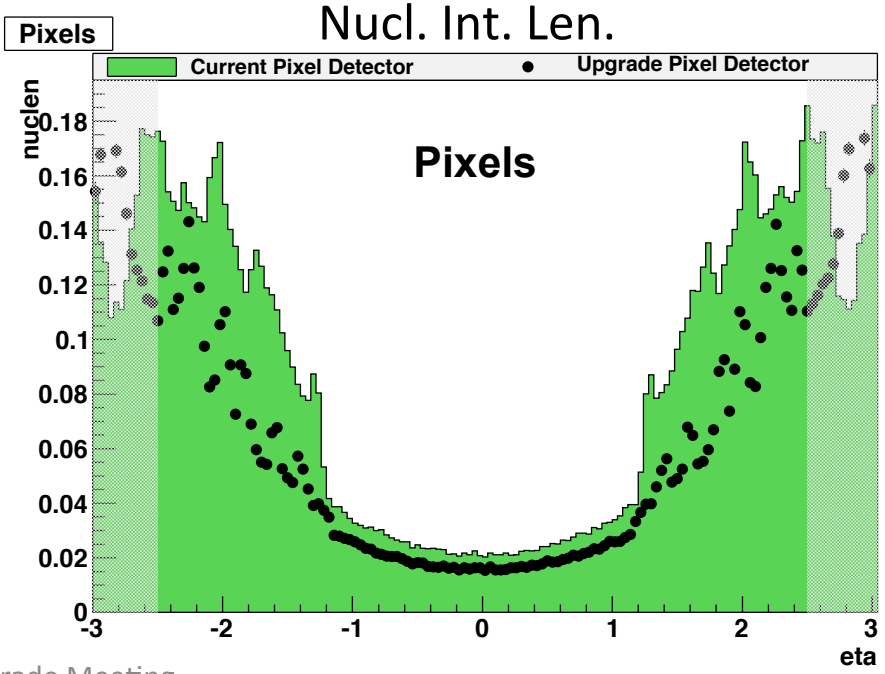
"Volumes"	Mass (g)	
	Current	Design Upgrade
BPIX η<2.16	16801	6618
FPIX η<2.50	8582	7024

50% less photon conversion in/ before pixel at eta 1.5

Rad. Len.

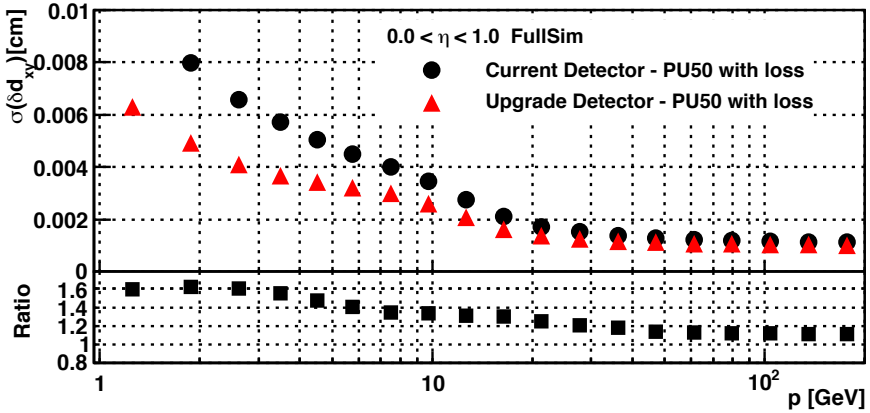


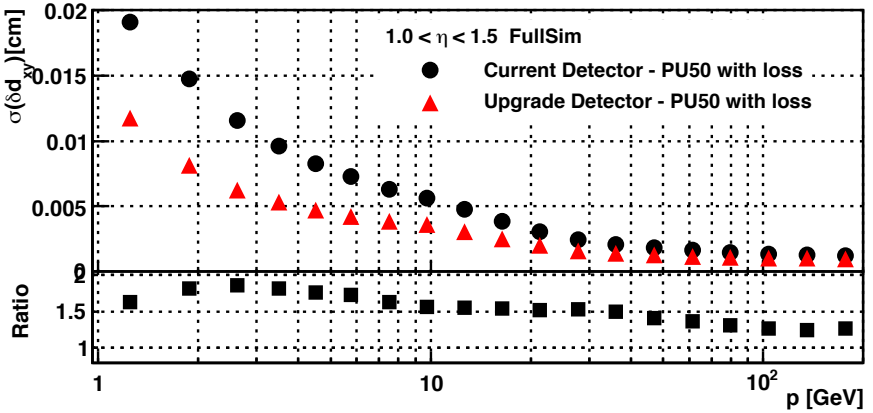
Nucl. Int. Len.

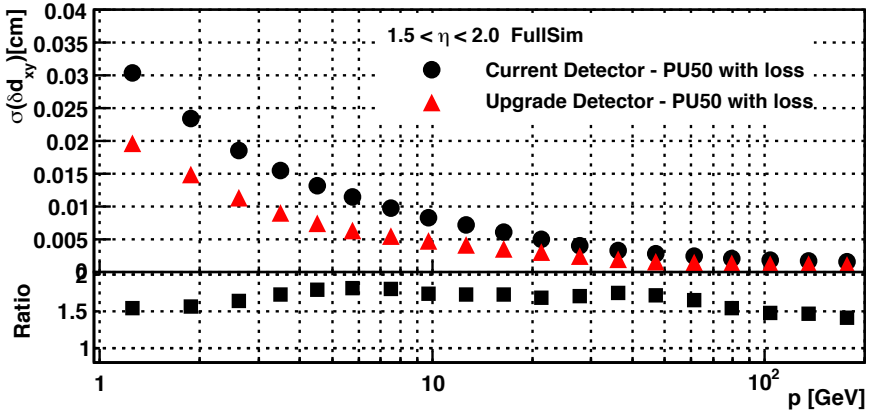


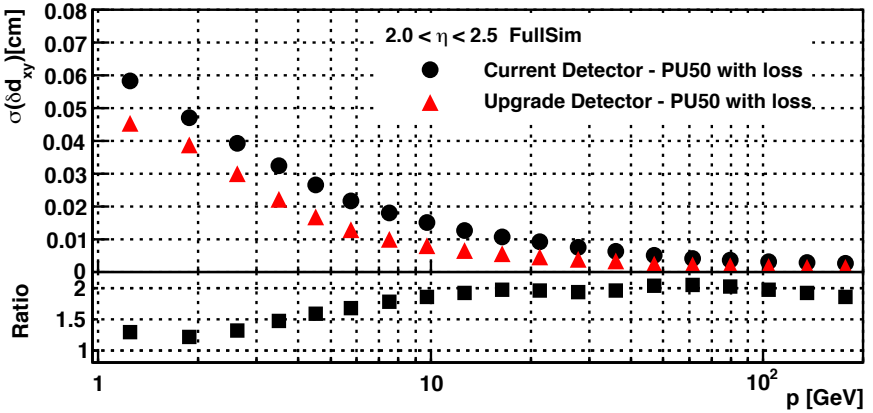
Impact Parameter Resolutions

- Transverse: muon sample (10 muons/event), <PU>=50
  - Generated flat in E and eta (plot vs absolute p and in 4 eta regions)
  - Compare current and upgrade detectors









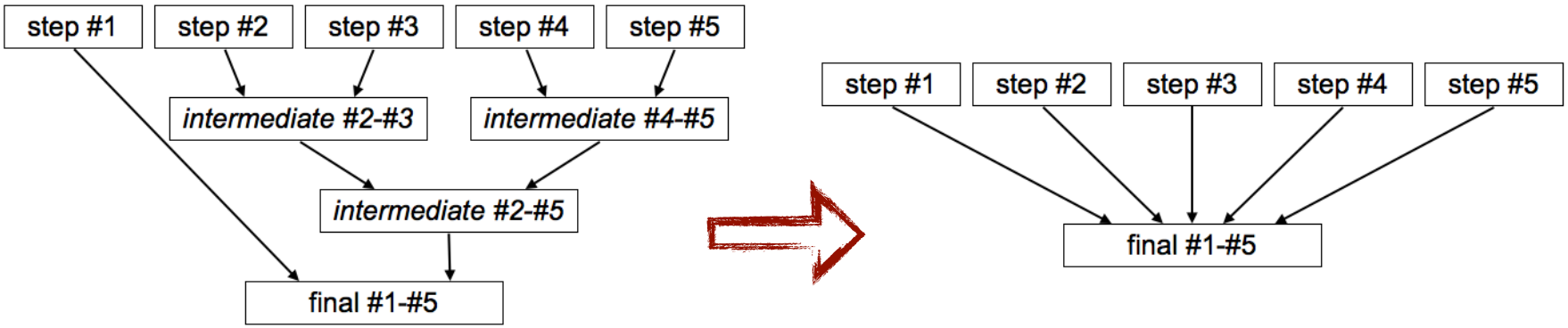
Fall 2011 campaign: from CMSSW42x to 44x (2)

Copy-less hit masking Each step of the iterative tracking works on the hits not yet associated to any track. This was done by creating a new collection of surviving hits at each step. Implemented a data member to store masking bits

Batch cleaning of seeds successfully propagated (track candidates) The track candidates are filtered in 1k batches to avoid storing too many of them

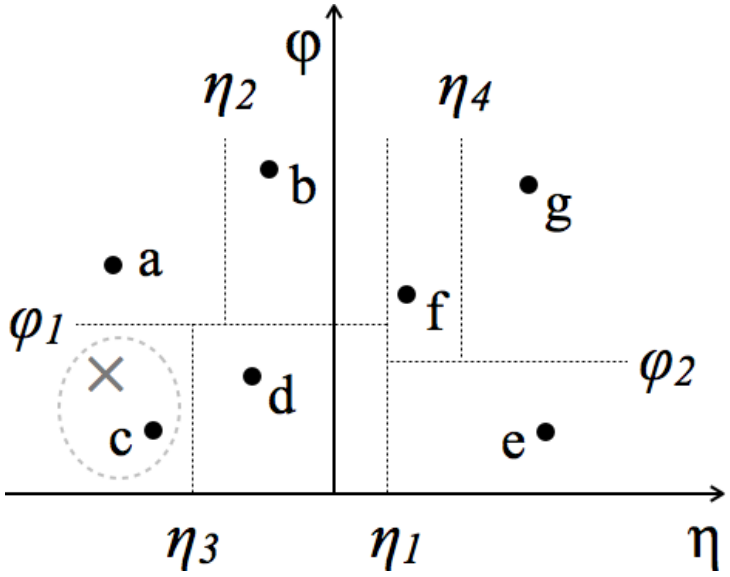
Efficient quality assignment Each iterative step assigns tracks to a quality tier. Old implementation just created a track copy per each tier; implemented a data member to store the quality tier bits

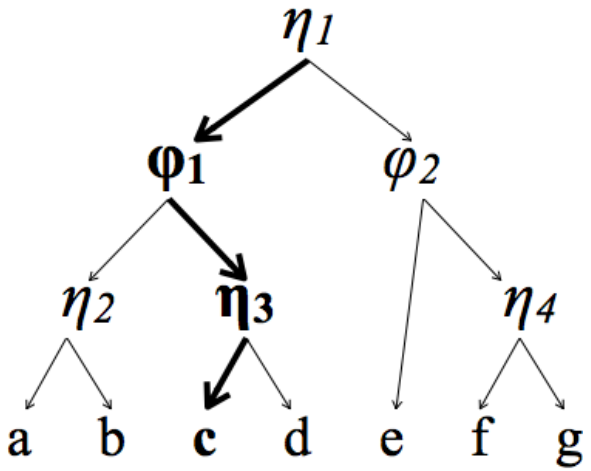
Efficient track merging The track collections resulting from steps have to be merged and cleaned. The merging algorithm has been improved by getting rid of intermediate collections.



Fall 2011 campaign: from CMSSW42x to 44x (3)

Particle flow links The PF algorithm links tracks to calorimetric clusters in the (η,ϕ) space. Done in 42x by CPU intensive nested loops, with a complexity that scales quadratically with the multiplicity N. In 44x implemented a **kd-tree** based algorithm: the (η,ϕ) space is split into appropriate domains, each containing one single object, organized in a tree. The cluster closest to a given track is found with a very fast binary search that ends up in the closest neighbor domain. The complexity that scales as N·logN. Already extended to other CMSSW modules by the implementation of a generic kd-tree class.

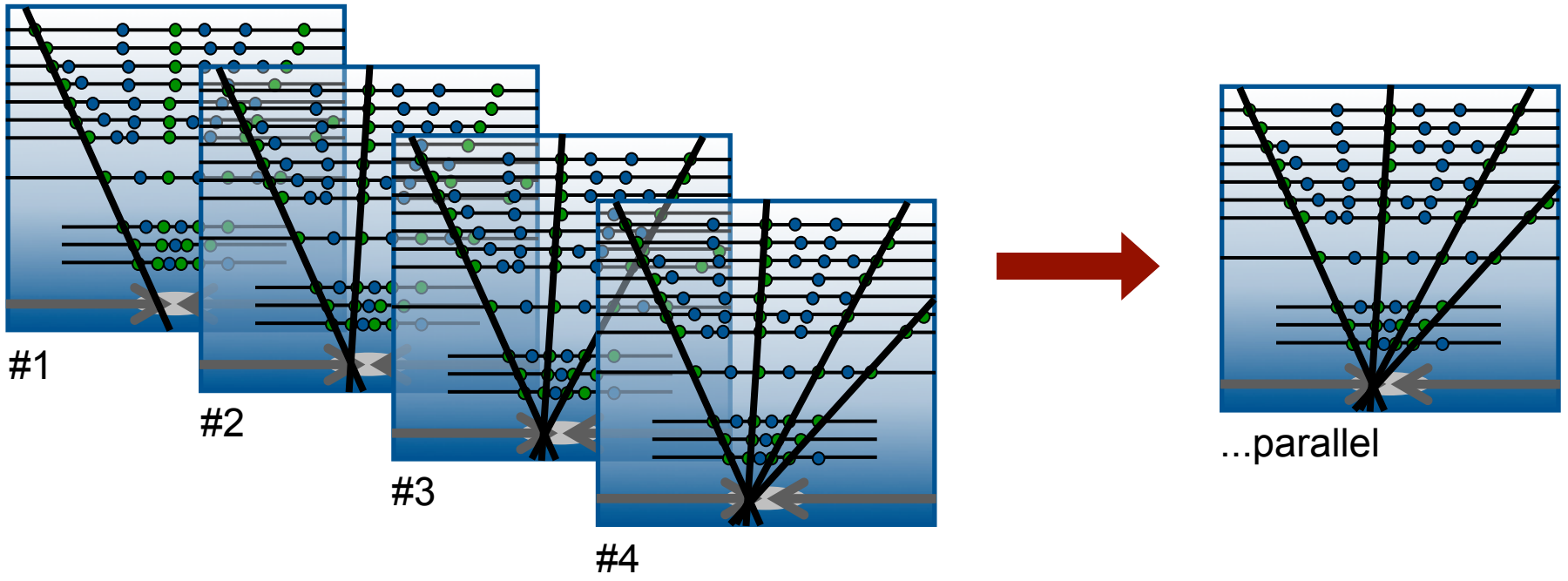




Parallelization within the modules

Identify modules that perform tasks where parallelization can be easily implemented; tracking is a clear candidate, i.e. track building after seeding (pattern recognition) within an iterative step

Made the algorithms thread safe (could not be trivial in case of tracking) CAVEAT: going thread safe could result in significant memory overhead...



Hough transform basics

Each hit is compatible to many trajectory hypotheses that can be represented by curves in an appropriate trajectory parameter space (typically straight lines); the intersection between many of these curves is a reconstructed track. So hits are transformed into lines (or curves, more in general) in the track parameter space by an appropriate conformal transformation, and accumulation points are identified.