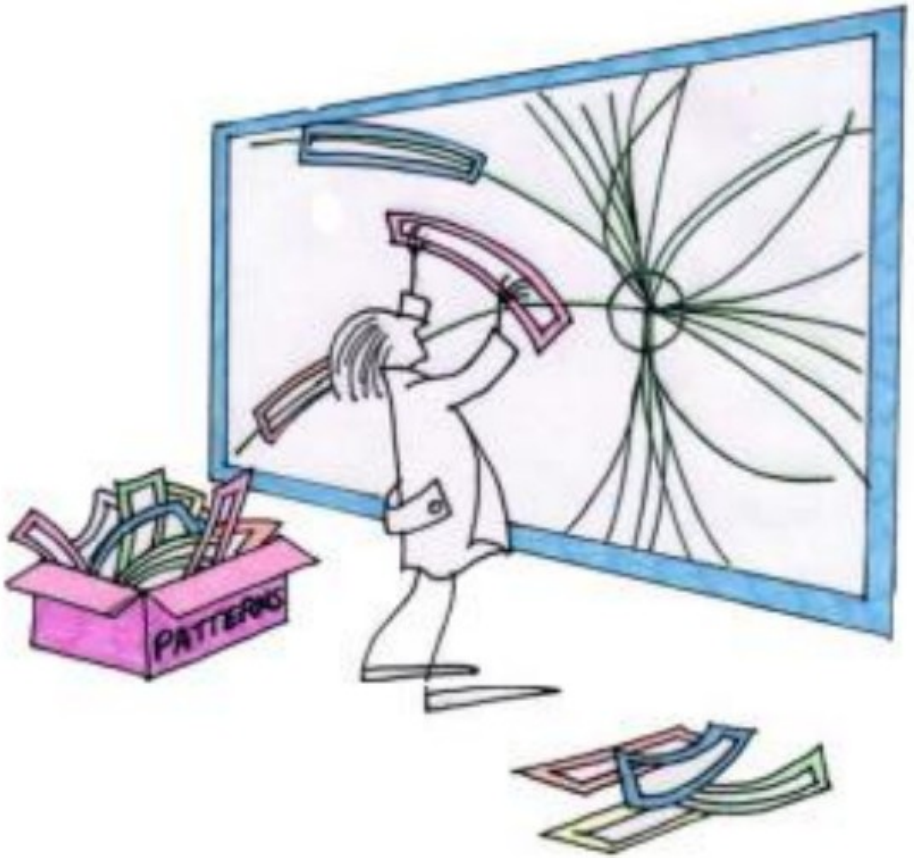
UNIVERSITÀ DI PAVIA

INFN Istituto Nazionale di Fisica Nucleare

Intelligent triggering: pattern recognition with Associative Memories and other tools



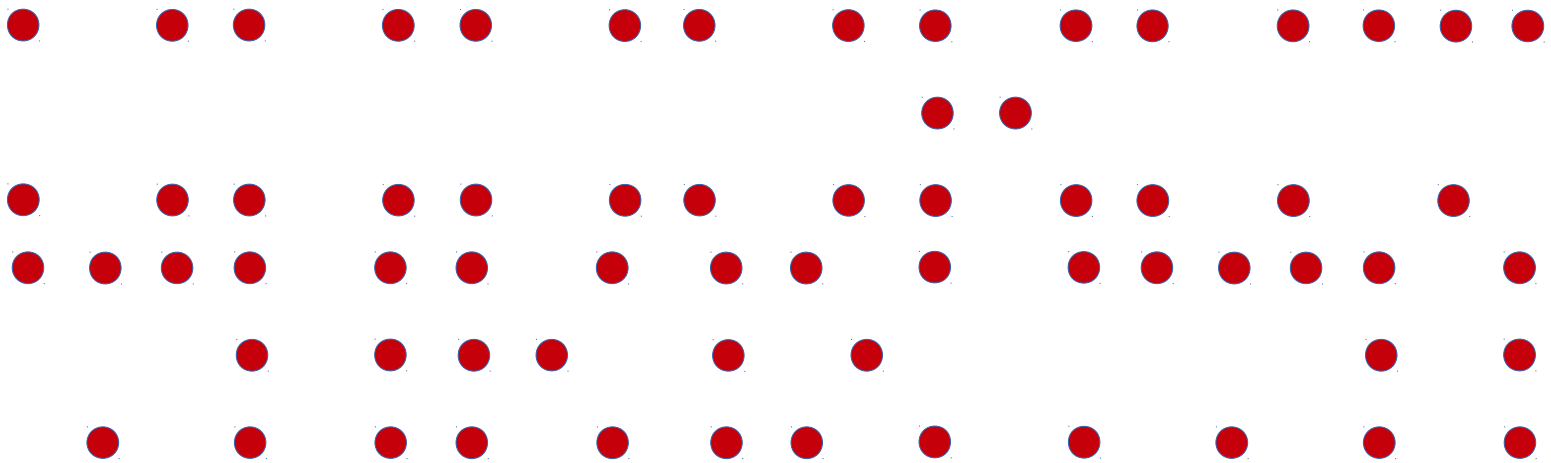
ISOTDAQ 2018: 9th International School of Trigger and Data Acquisition

Vienna, 20 Feb 2017

Andrea.Negri@pv.infn.it

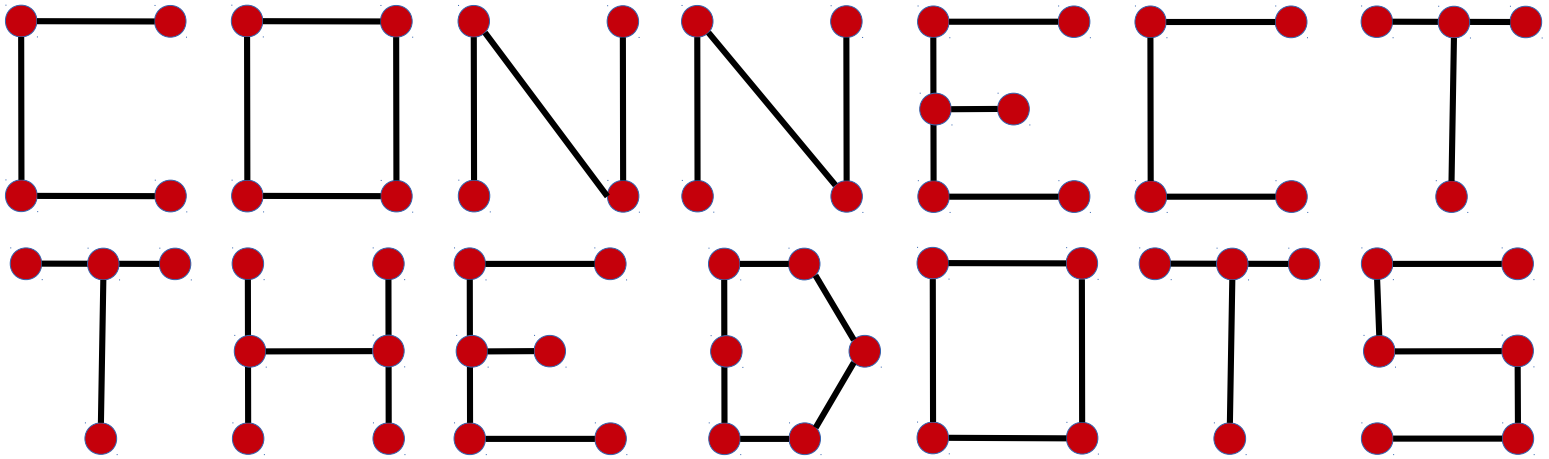
Introduction

- What does it mean “Intelligent triggers”?
  - Find clever ways to bypass new challenges
- Impossible to provide an exhaustive report
  - Focus on pattern recognition in HEP



Introduction

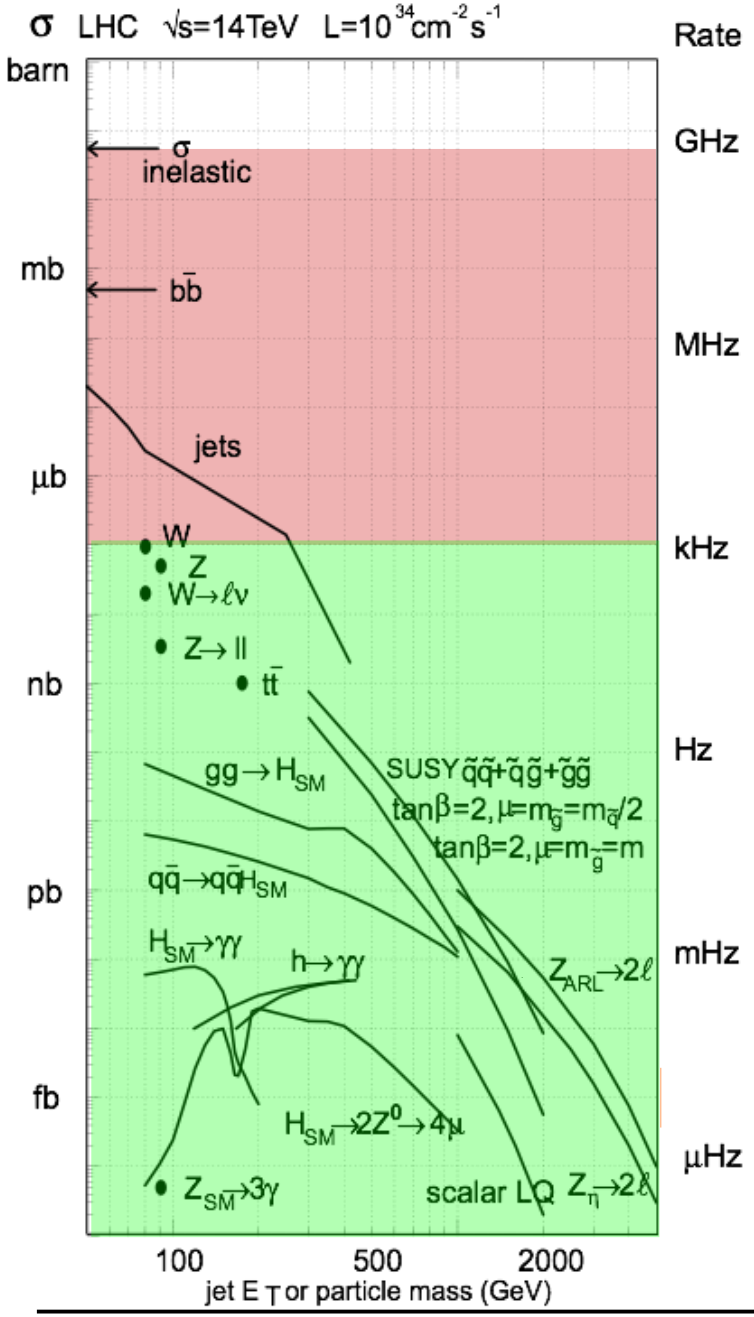
- What does it mean “Intelligent triggers”?
  - Find clever ways to bypass new challenges
- Impossible to provide an exhaustive report
  - Focus on pattern recognition in HEP



Overview

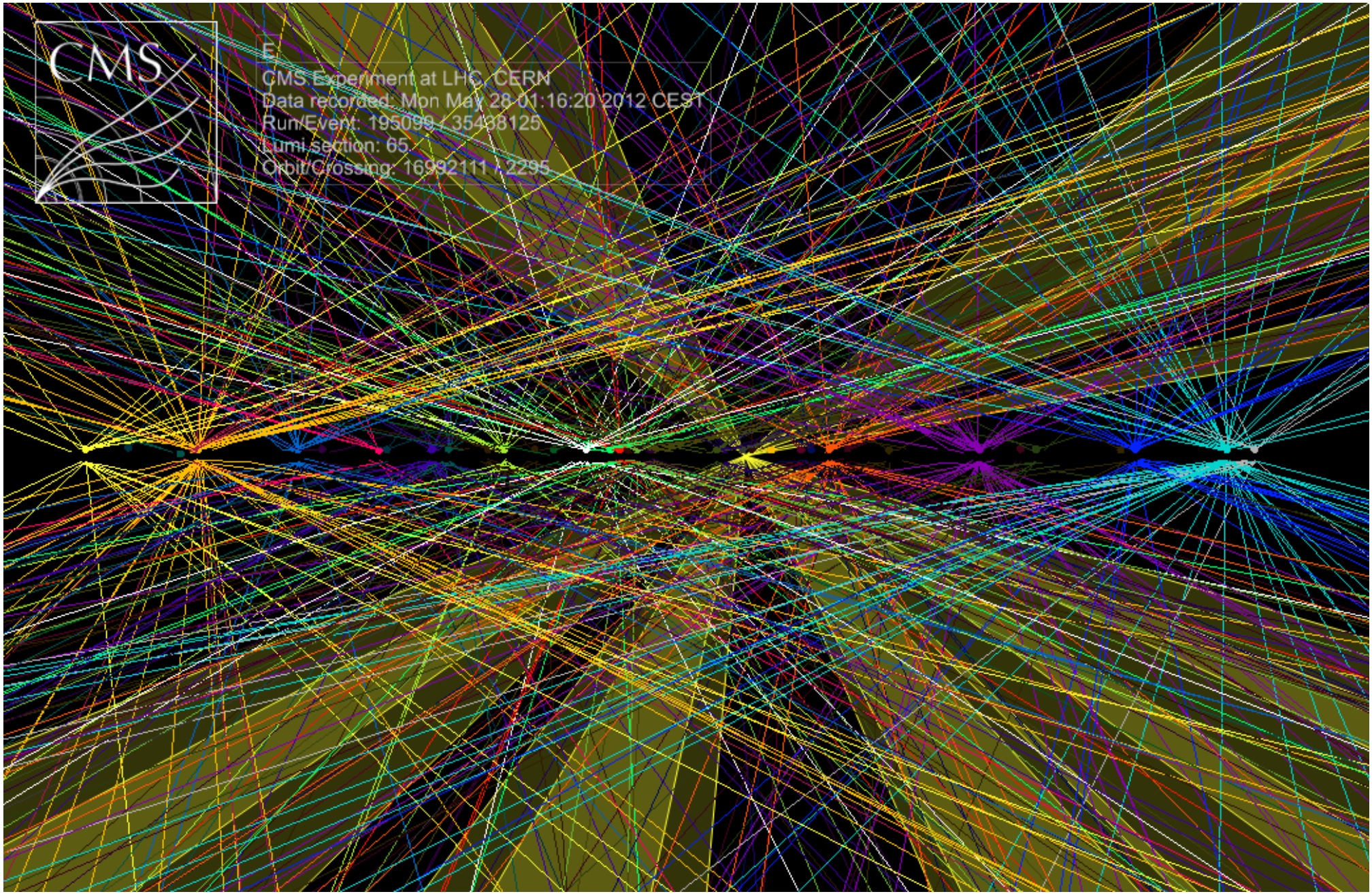
- The LHC luminosity in increasing
  - More and more collisions per bunch crossing
- Tracking is becoming more and more important for triggering purpose
  - Less pile-up dependent
  - Trend: use tracking as early as possible (HLT → L1)
- But tracking is time consuming
  - Not compatible with available latencies
- Look for smart tracking solutions
  - Exploit parallelism: divide et impera
  - Find smart algorithms to be implemented in CPU, GPU, FPGA, ASICs

Luminosity

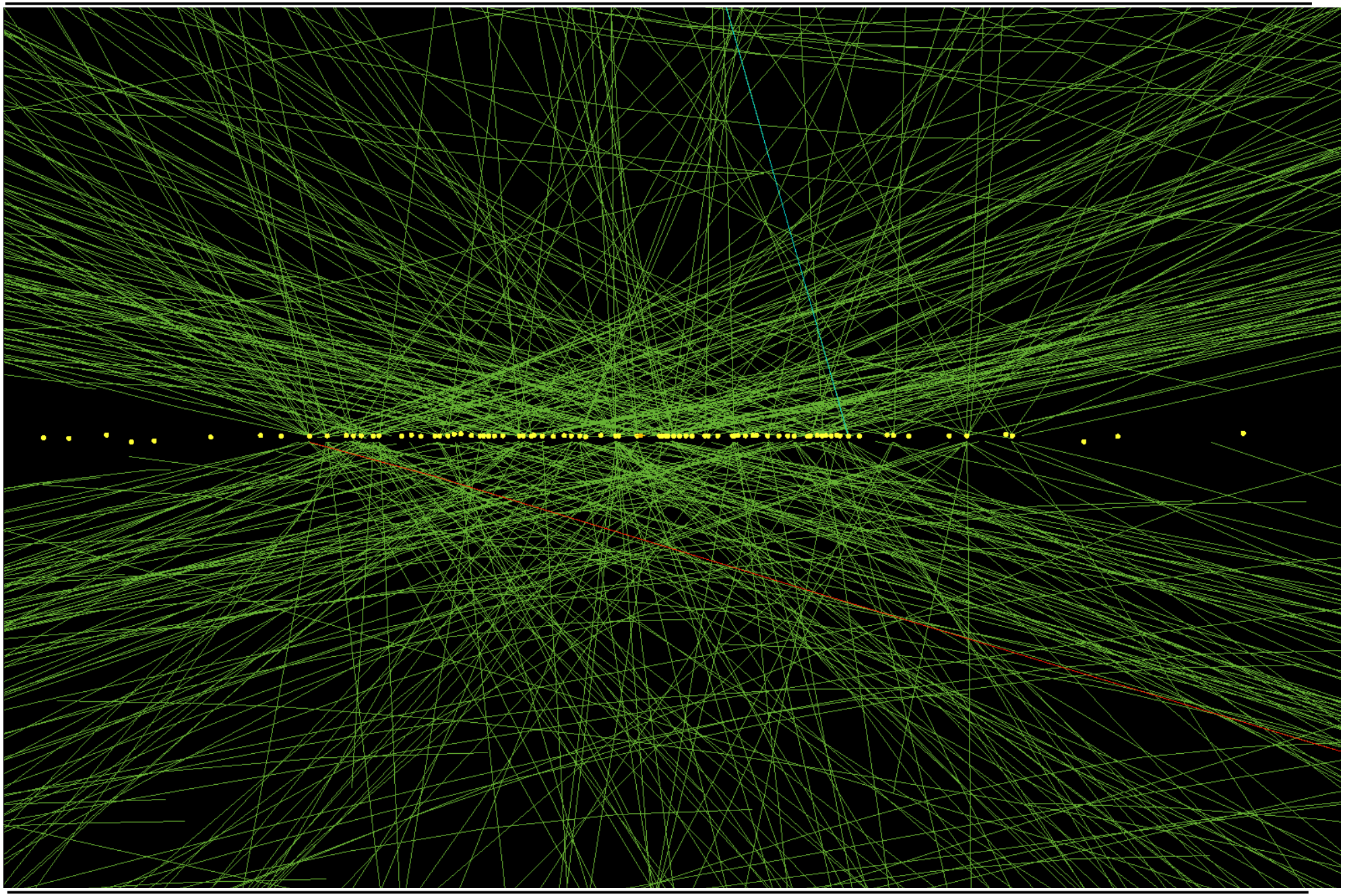


- Luminosity
  - **R**[Hz] = **L**[cm^{-2}s^{-1}] · **σ**[cm^2]
- Goal
  - O(100) events/year for rare channels
  - L ~ 10^{34} cm^{-2}s^{-1}
- σ_{pp} ~ 100 mbarn
  - Rate = 1 GHz
- But collisions @25ns → **pile-up**
  - Rate = 40 MHz
  - At 10^{34} cm^{-2}s^{-1}: 25 collisions per bunch crossing
  - At 10^{35} cm^{-2}s^{-1}: 250

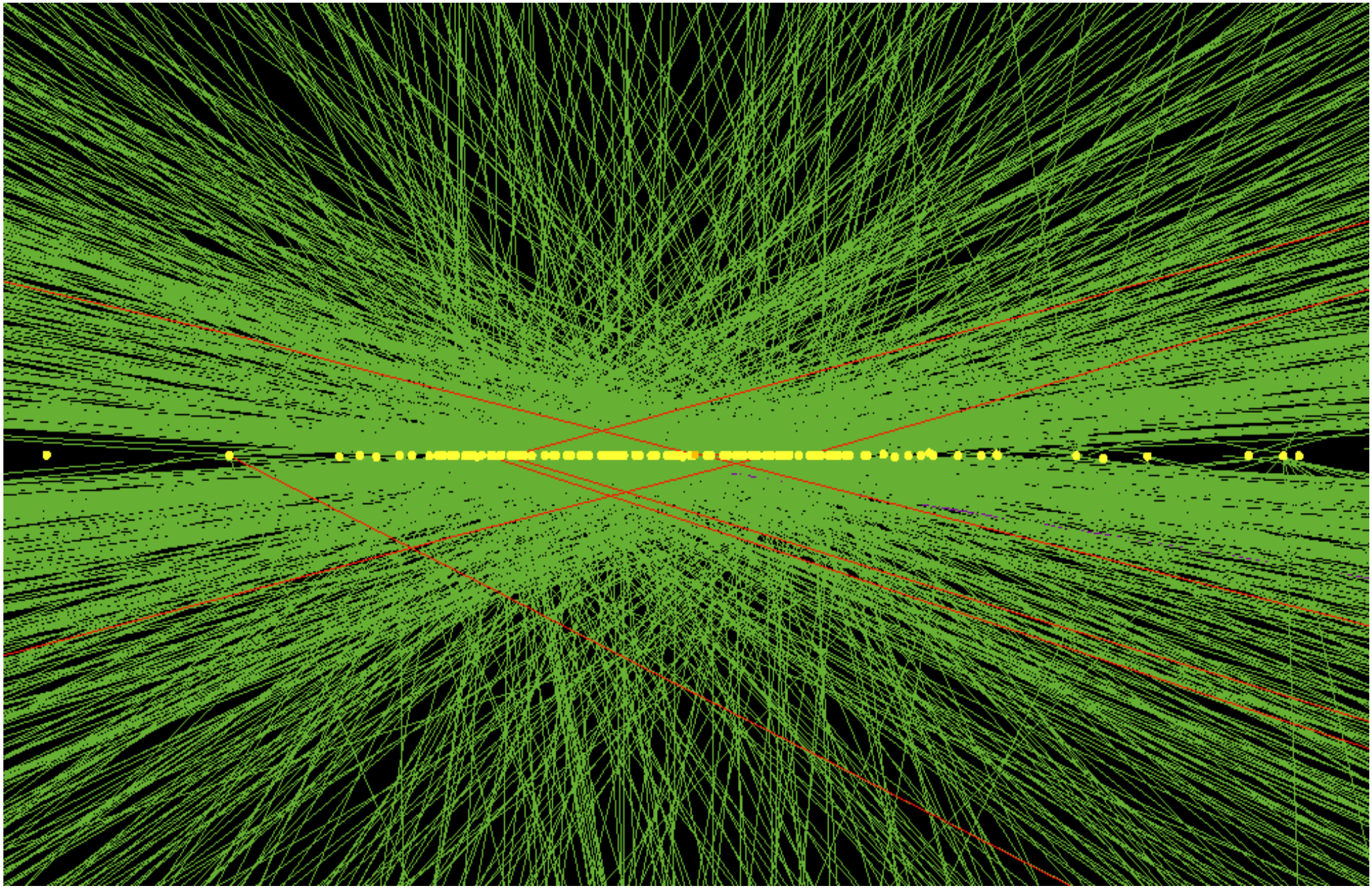
Pile up ~10


CMS
CMS Experiment at LHC, CERN
Data recorded: Mon May 28 01:16:20 2012 CEST
Run/Event: 195099 / 35438125
Lumi section: 65
Orbit/Crossing: 16992111 / 2295

Pile up ~50

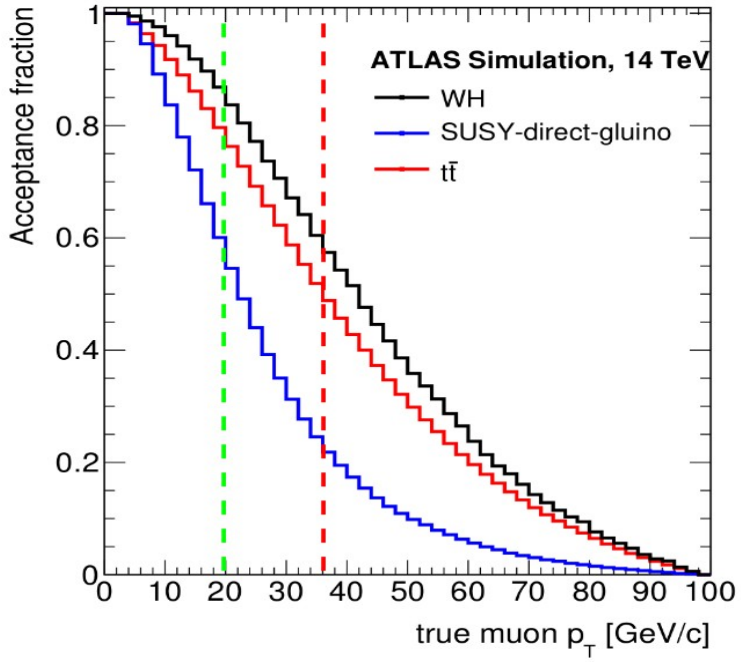


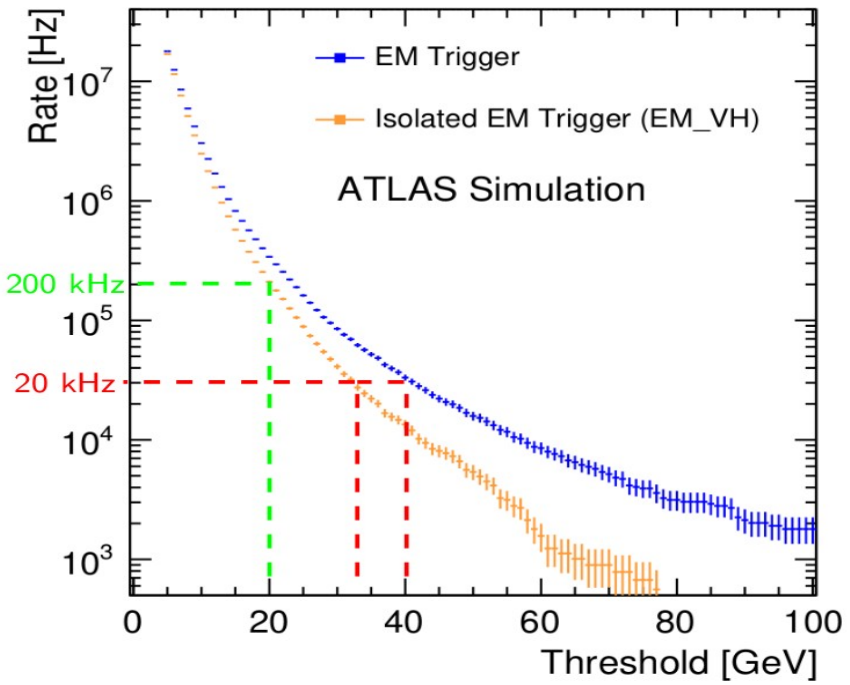
Pile up O(100)



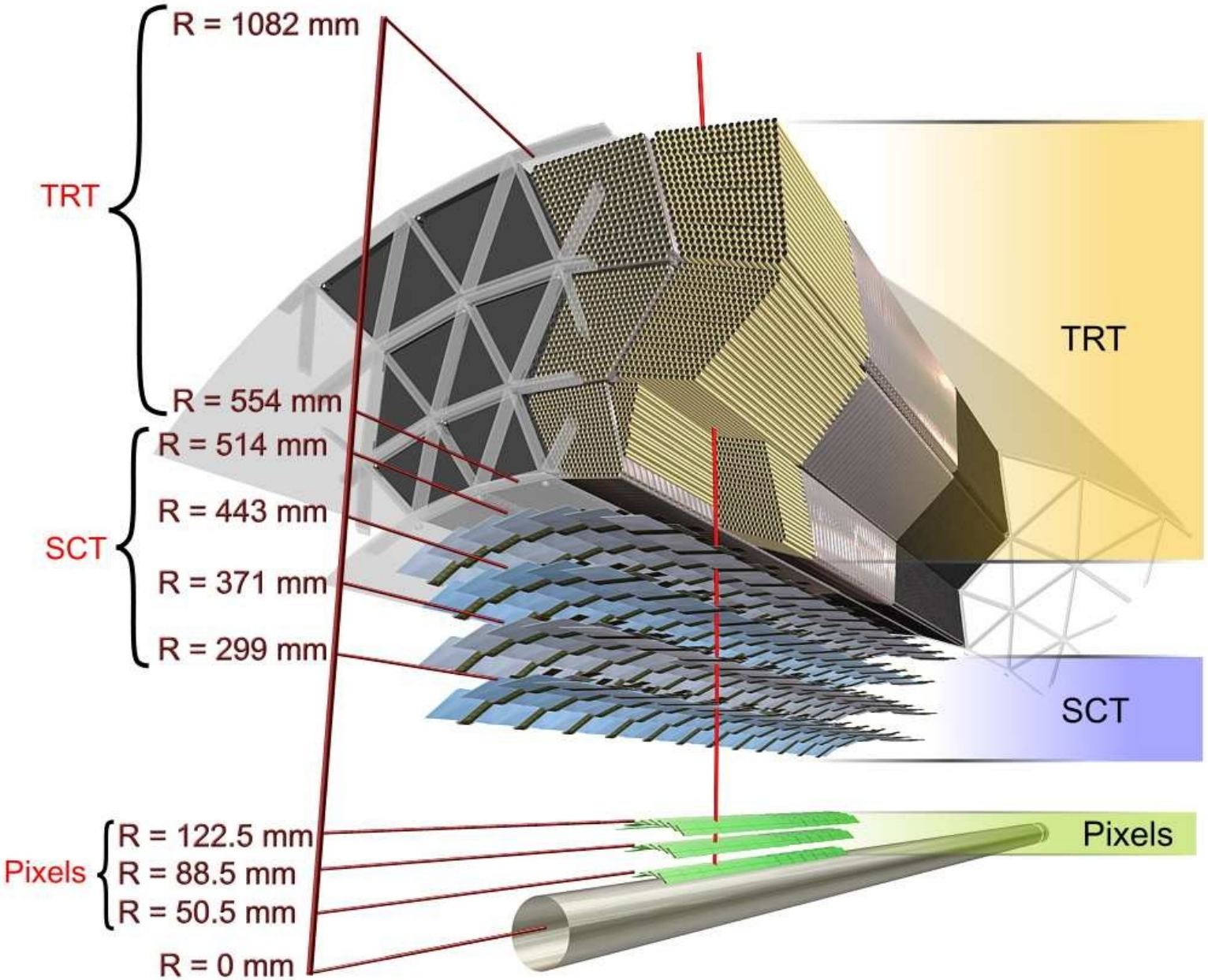
Trigger vs luminosity

- Thresholds already near scale of interesting processes
  - Increasing them will reduce signal efficiency
- T/DAQ base requirements
  - Maintain p_T thresholds at ~ 20 GeV for single electron and muon trigger to preserve acceptance for W, Z, tt, H
  - Maintain system flexibility to be able to adapt to background



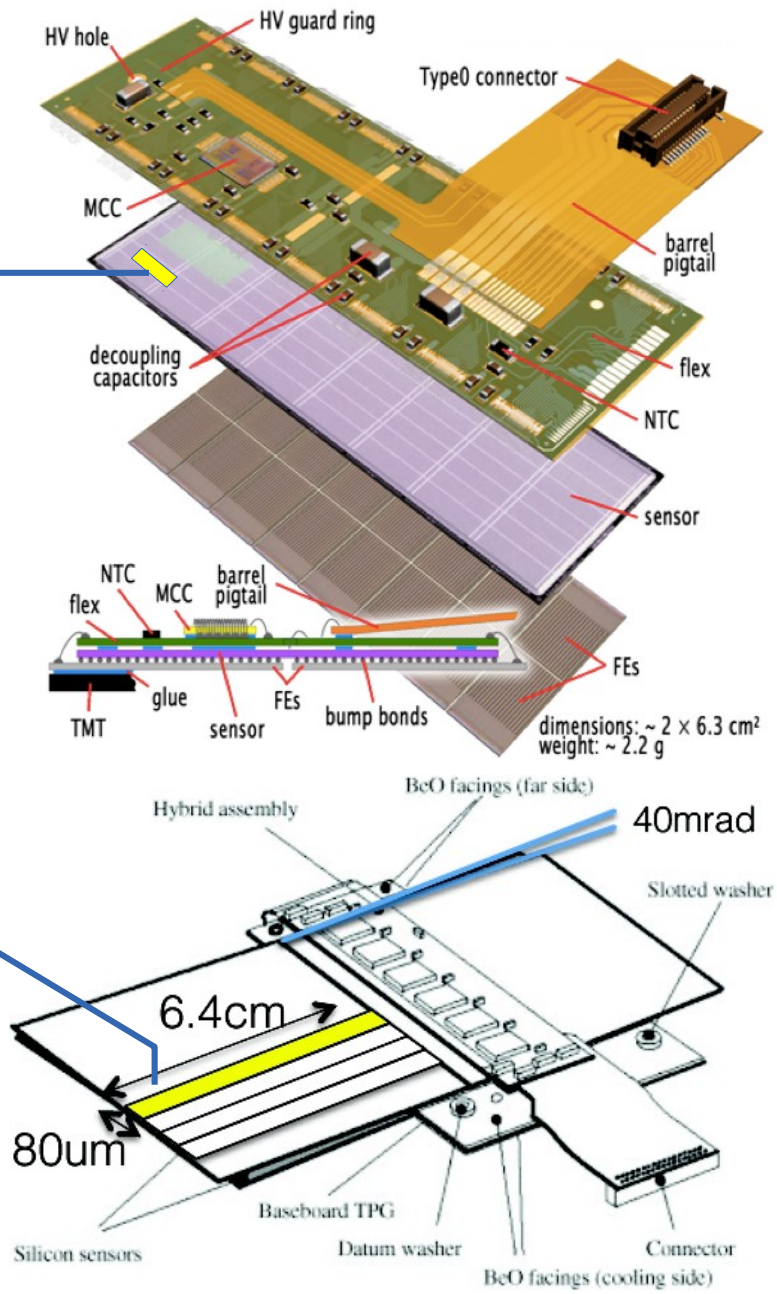


Inner detectors



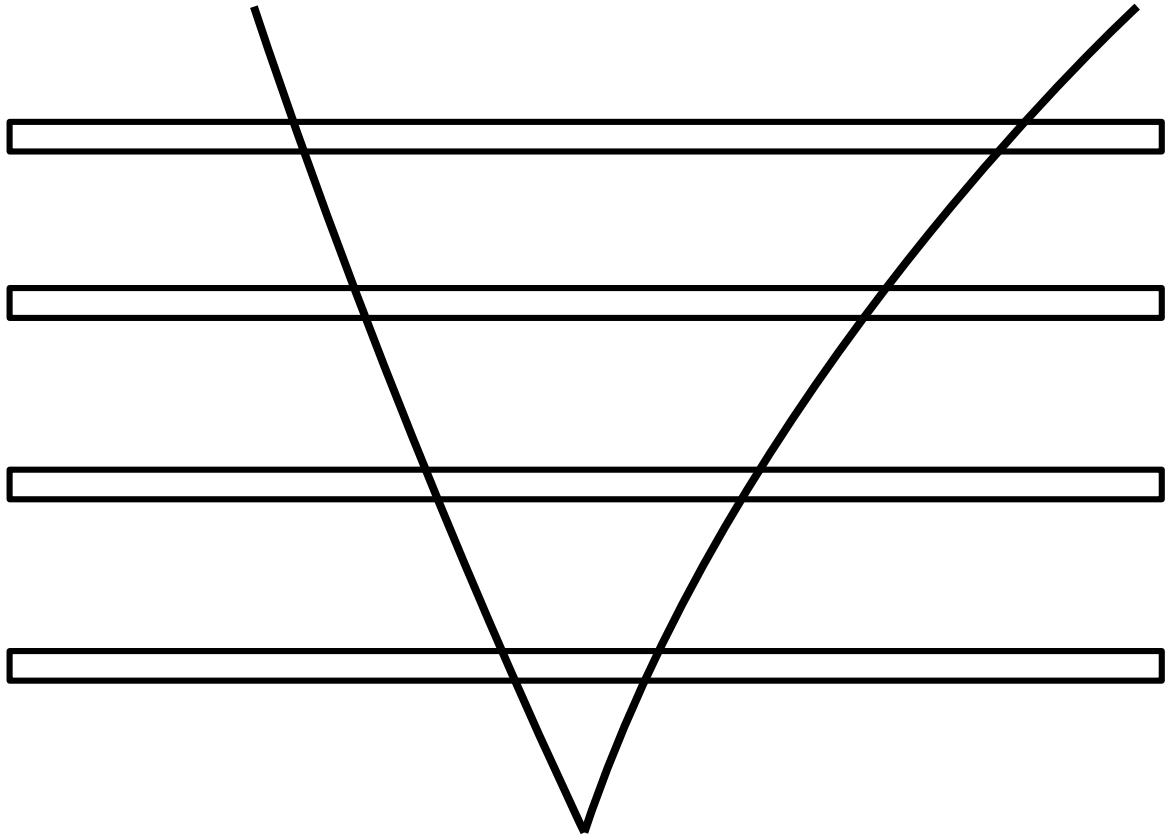
Pixel and strips (ATLAS)

- Pixels
  - **90M** channels
  - 50 μm x 250 μm
  - 4 layers, 3 end cap disks
- Strips
  - **6M** channels
  - 80 μm (x 6.4 cm)
  - Double plane w/ 40 mrad stereo angle for 2nd coordinate
  - 4 layers, 9 end cap disks



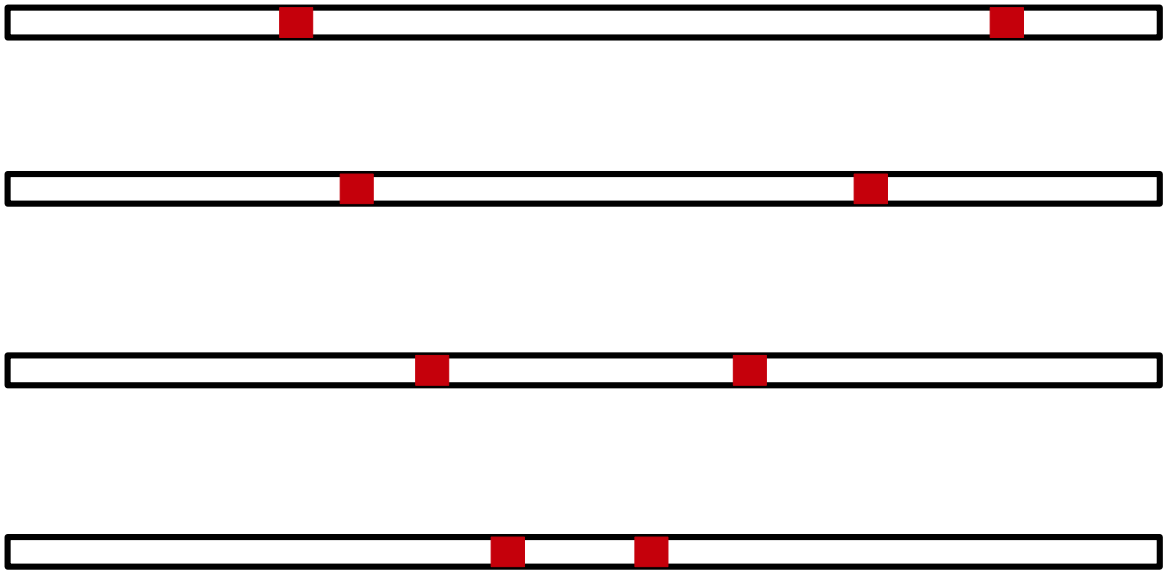
Tracking

- Collisions produce particles



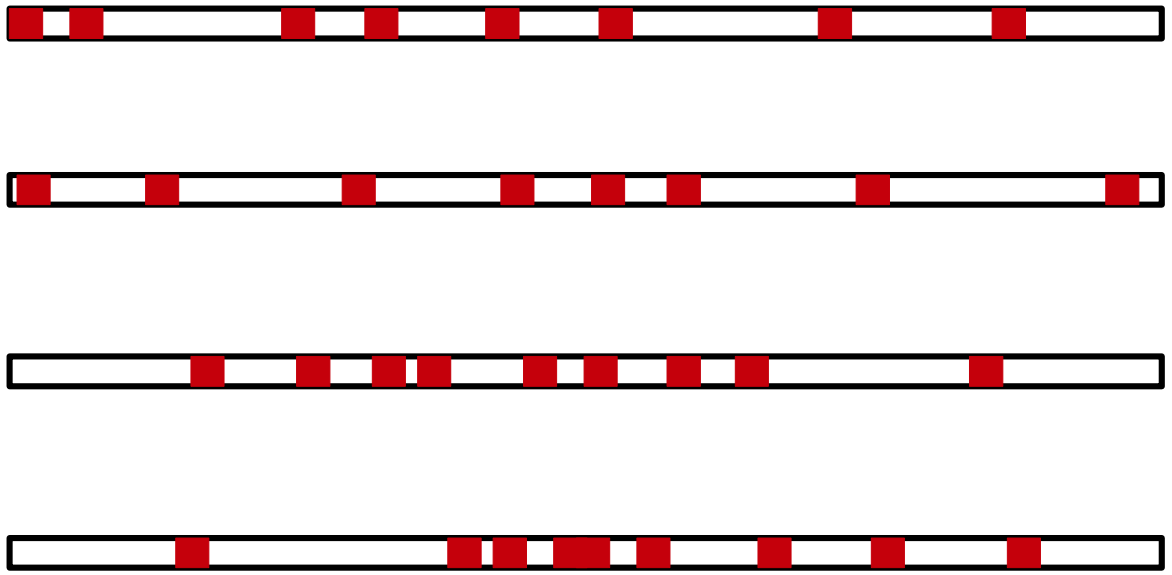
Tracking

- Charged particles produce hits



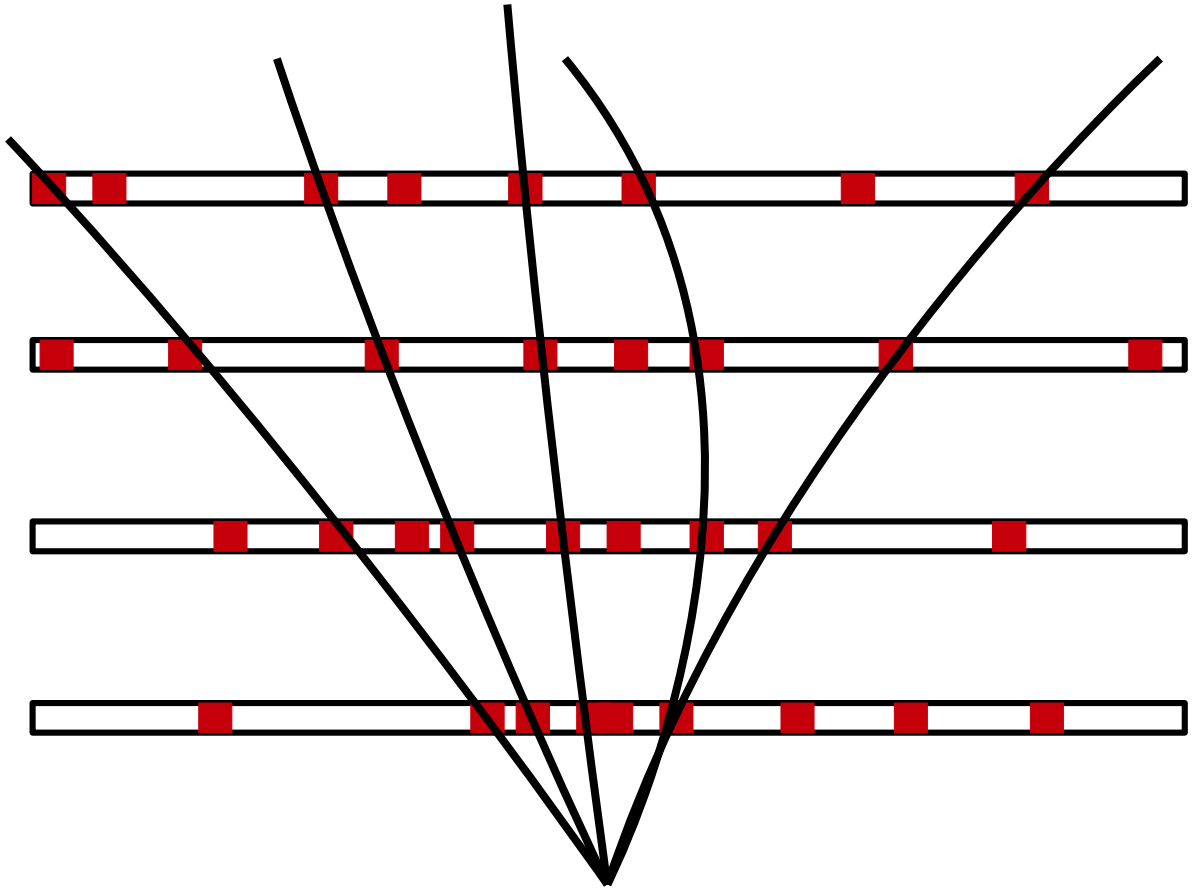
Tracking

- Usually
  - there is noise
  - and pileup

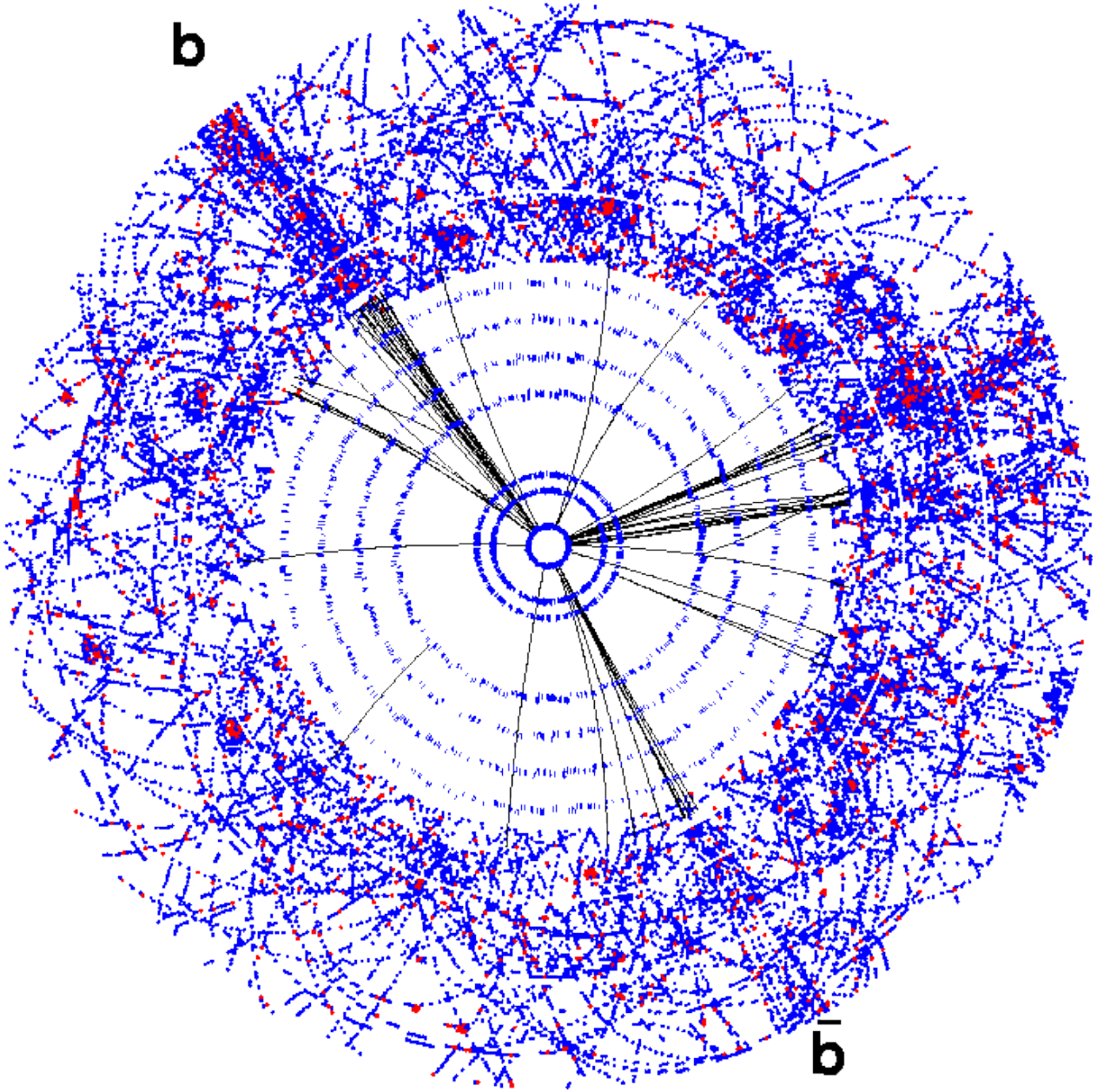


Tracking

- Challenge: reconstruct the tracks from the hits
  - combinatorial problem

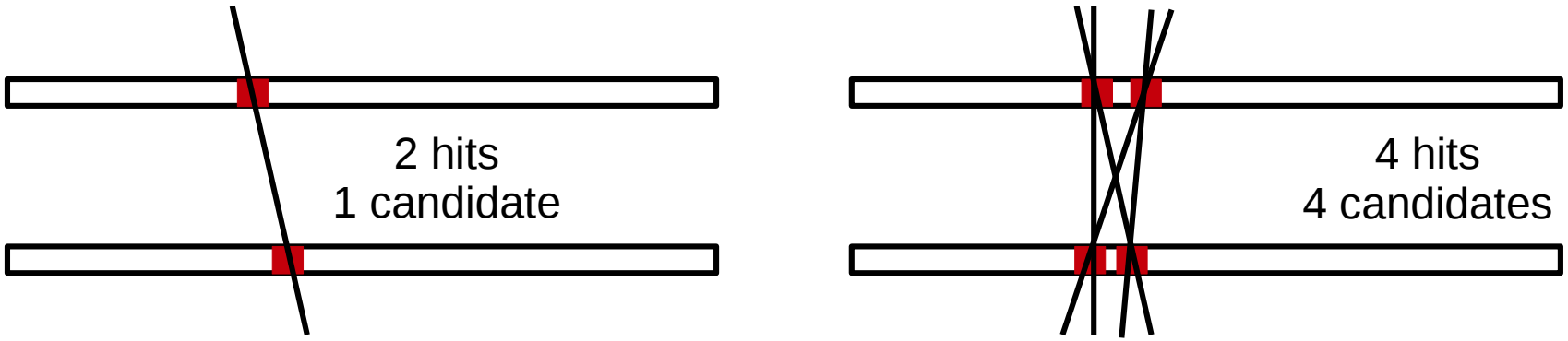


At each bunch crossing



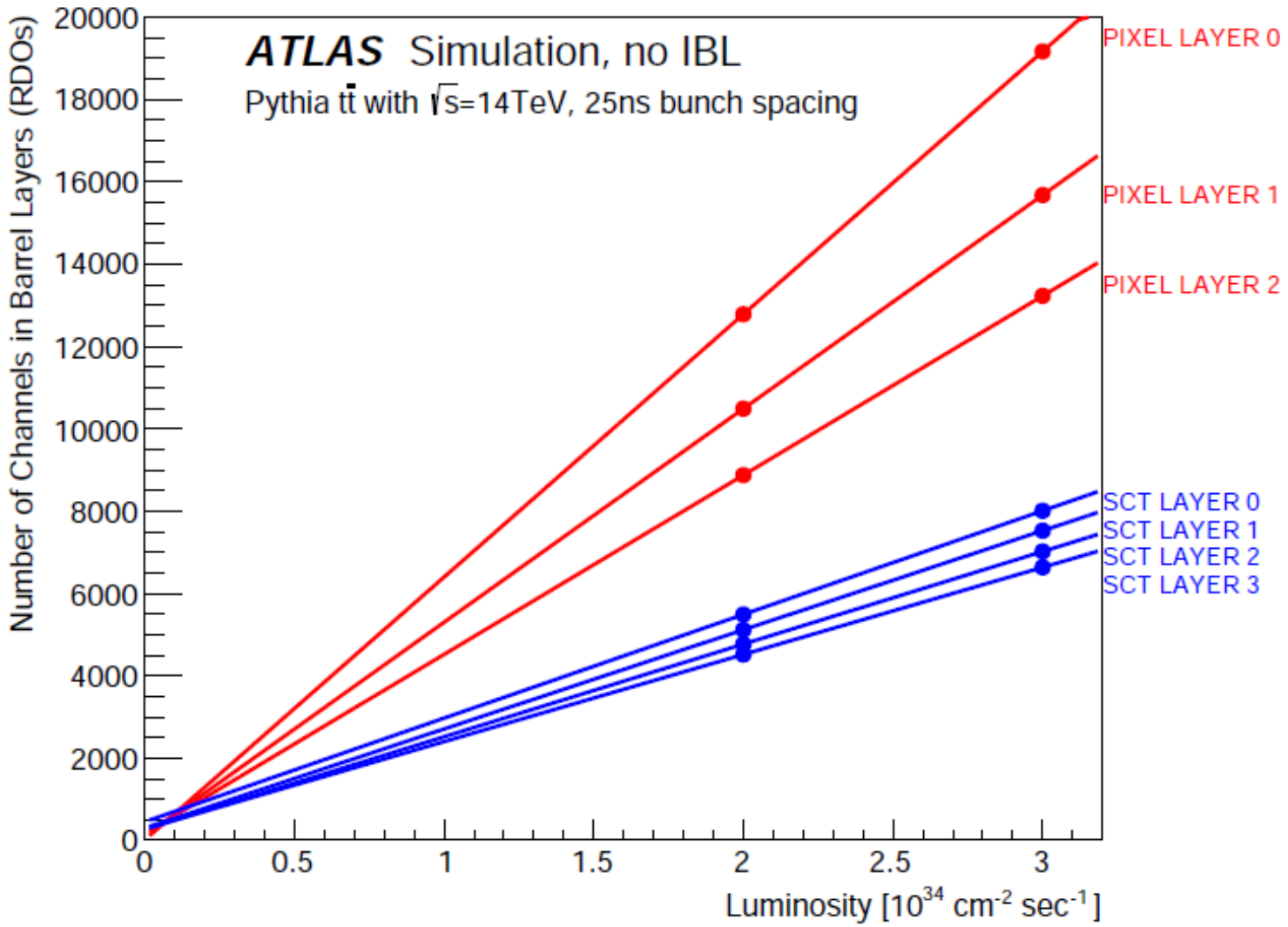
Processing time

- Challenge: track finding in limited latency
  - Number of hit combinations to be tested increases like L^N, where L is the luminosity and N the number of silicon layers
- CPU processing time explodes with luminosity
  - Tracking currently used at HLT but only on small regions of interest, sometimes after calorimetric preselection



Occupancy vs luminosity

- Number of hits per layer vs luminosity

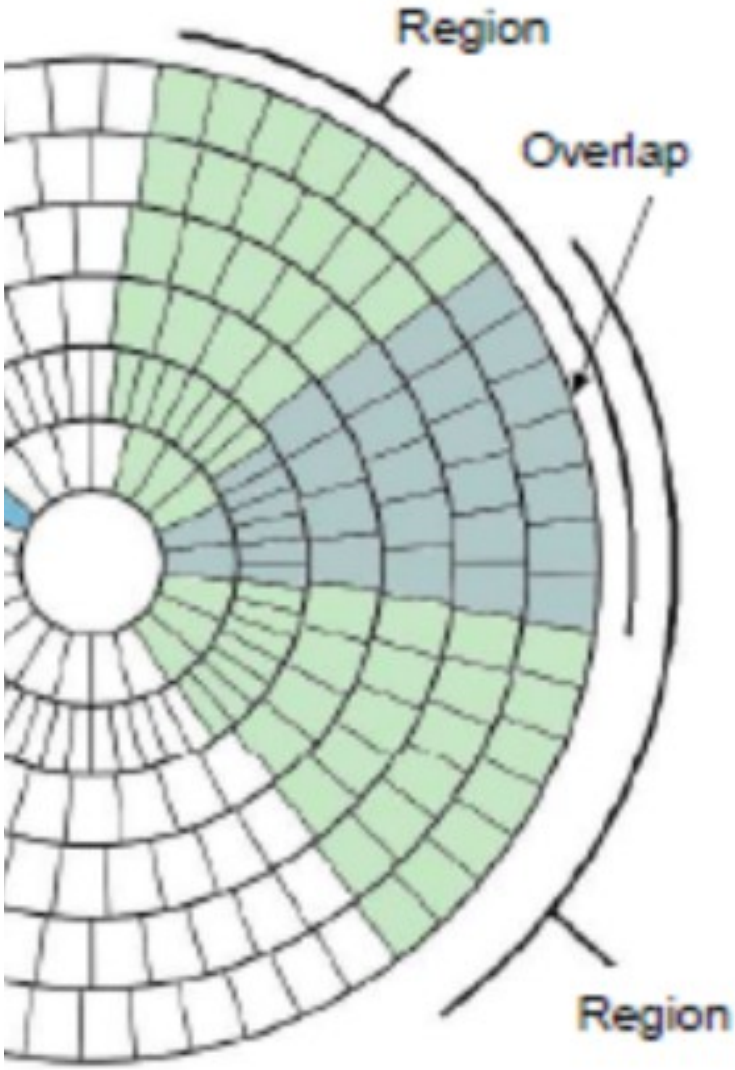


Divide et impera

- Reorganize the work load
  - Parallelize the work in independent lines
  - Decompose each line in serial steps (with selection)

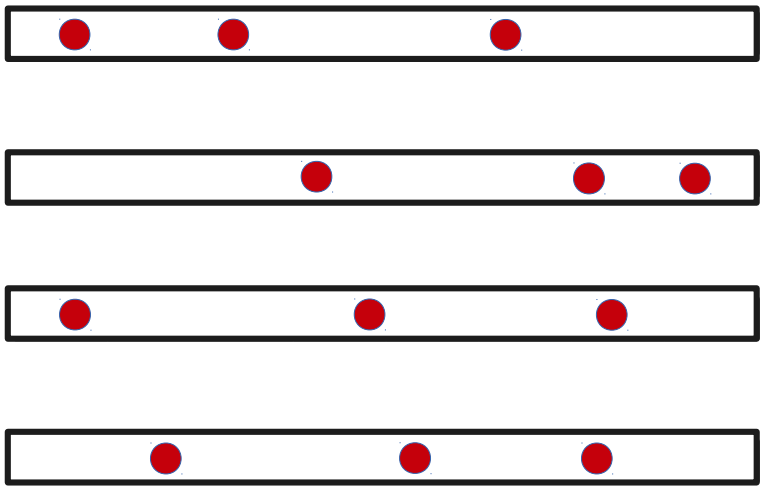
1st Data Formatting

- Divide you detector in regions in η and ϕ
  - with overlaps: each track should belong to only two regions
  - data formatting: properly reorganize channels to properly feed next stages
- Process each region independently
  - merging step needed downstream
  - E.g. to remove duplicated tracks



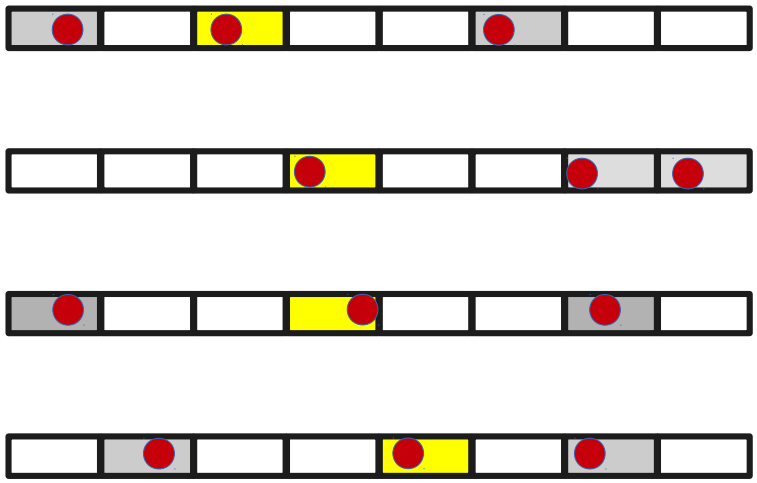
Then find the tracks

- Decompose tracking is separated steps
- **Pattern recognition**
  - Find track candidates (roads)
  - Usually with **lower resolution** to minimize latency
    - Trade off between efficiency and # of false positive
- **Fit**
  - In every roads, fit of the full resolution hits and keep the tracks with best χ^2



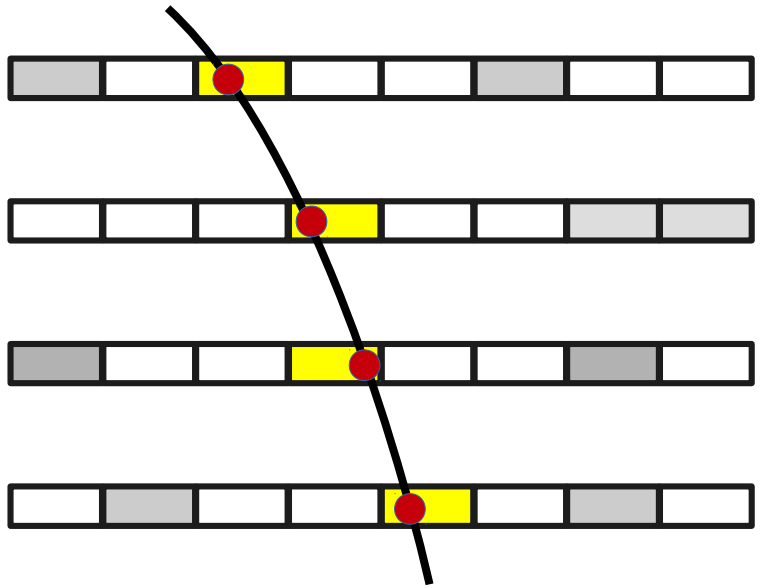
Then find the tracks

- Decompose tracking is separated steps
- **Pattern recognition**
  - Find track candidates (roads)
  - Usually with **lower resolution** to minimize latency
    - Trade off between efficiency and # of false positive
- **Fit**
  - In every roads, fit of the full resolution hits and keep the tracks with best χ^2



Then find the tracks

- Decompose tracking is separated steps
- **Pattern recognition**
  - Find track candidates (roads)
  - Usually with **lower resolution** to minimize latency
    - Trade off between efficiency and # of false positive
- **Fit**
  - In every roads, fit of the full resolution hits and keep the tracks with best χ^2



Intelligent Triggering: steps

- Most approaches are based on those steps
  - data formatting
  - **pattern recognition** at low resolution
  - **track fitting**
  - duplicate removal

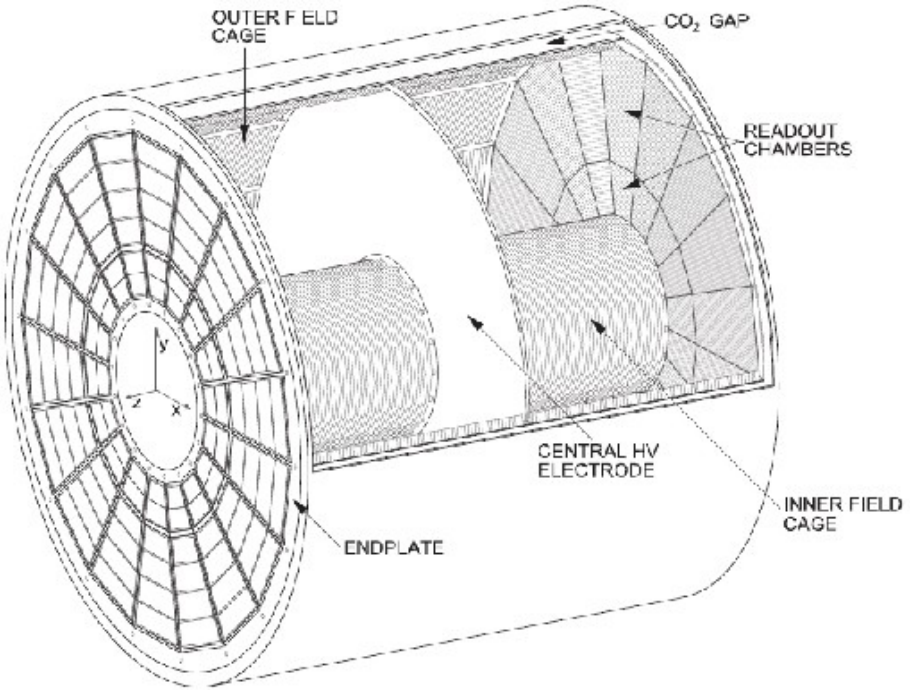
Pattern recognition

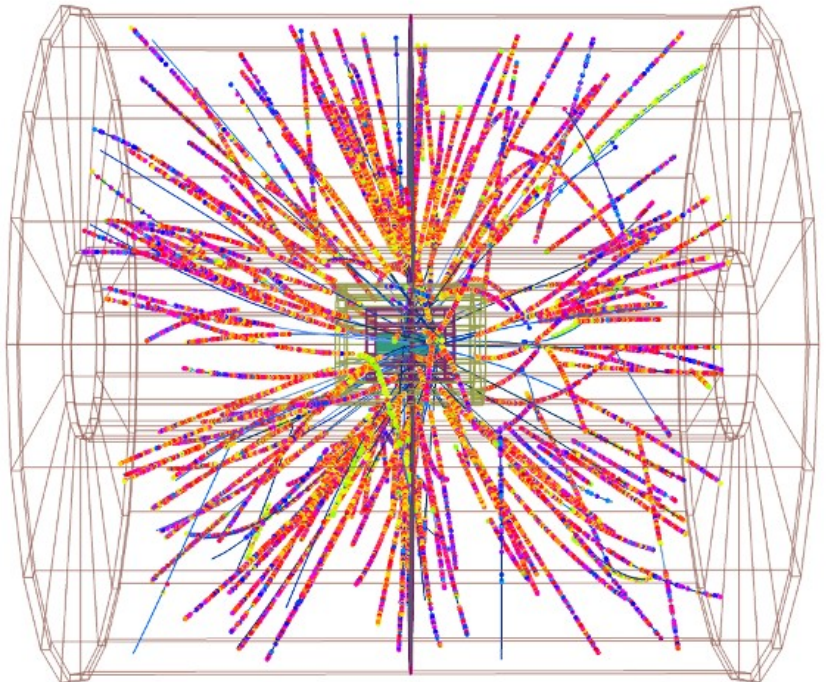
- Goal: find smart algorithms able to select the relevant patterns
  - As fast as possible
  - Producing an output rate sustainable by the next step
  - With the maximum efficiency and lowest fake rate
- Basic principles
  - Not all possible patterns are physically possible
  - Use lower granulary input
- Hardware implementation
  - ~~CPU~~ or GPU
  - FPGA: e.g. Hough transform or Retina algorithm
  - Custom ASICs: e.g. Associative Memories

GPU: Alice

doi :10.1088/1742-6596/898/3/032030

- ALICE HLT uses a GPU-accelerated algorithm for TPC tracking based on Cellular Automaton principle and on Kalman filter
  - Up to 159 hits per trajectory in the drift chamber
  - GPUs accelerated track reconstruction up to a factor of 10 compared to CPU approach

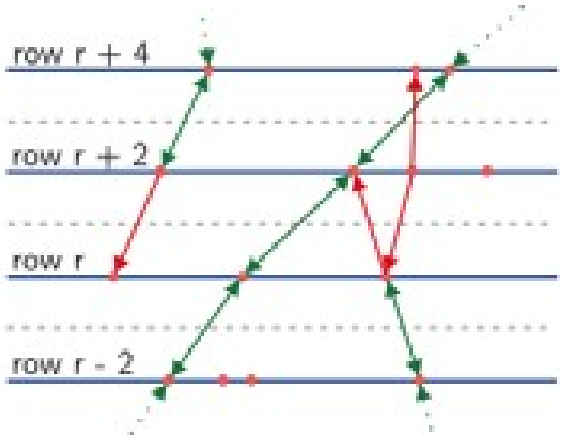


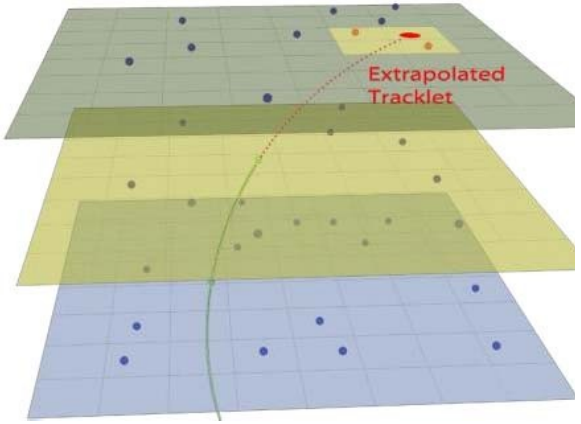


Algoritm

doi :10.1088/1742-6596/898/3/032030

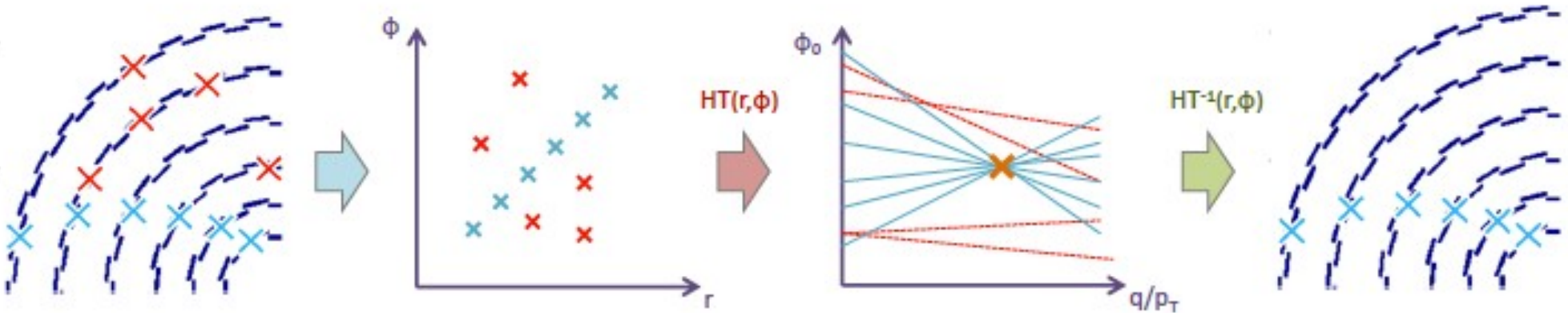
- Separation of TPC into sectors
- Then
  - **Seeding**: finds short track candidates of 3 to 10 clusters using a heuristics in a cellular automaton
  - **Track following**: fits parameters and extrapolates the track through TPC sector volume to find all hits of the track segment
  - **Track merging**: creates the final tracks by merging track segments
  - **Track fit**: refit full track using Kalman filter





Hough transform

- Tracks identified via a “voting procedure” carried out in a parameter space
  - track candidates as local maxima in that space
- Each hit in the x-y plane transformed to a line in the parameter space
  - e.g.: x,y → r, ϕ → ϕ_0, q/p_T
  - Hits from the same track → intersecting lines
  - Suited for FPGA: histogramming approach

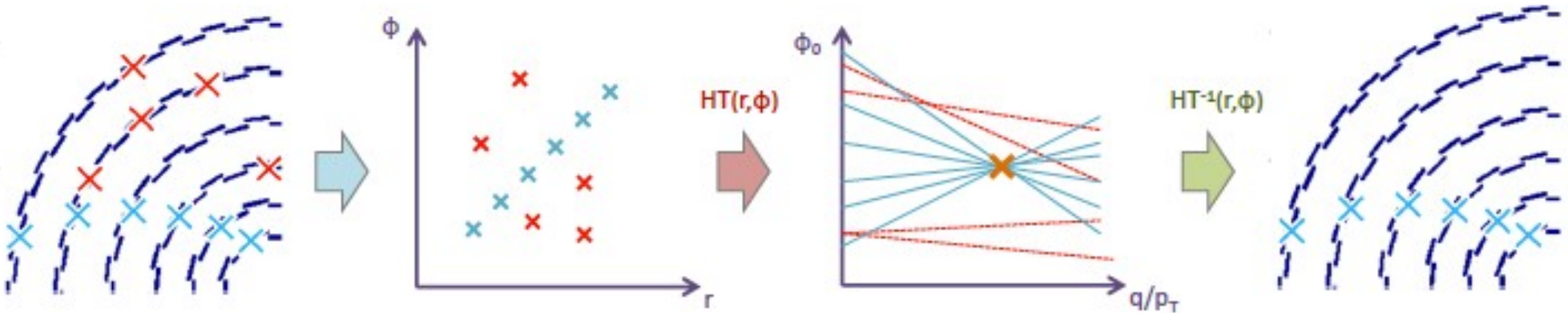


Hough transform

- Suited for FPGA: Histogramming approach
  - Divide the space into bins
  - Fill the bins with data
  - Find maxima

Topic of the secret lab: tonight at 18.30

- Optimal bin size: trade off upstream HW vs downstream HW
  - Usually followed by track fitting stage
- Advantages
  - Fast: once bins are filled all tracks are found
  - Easy to p_T order candidates for further processing



Retina

ELSEVIER

Nuclear Instruments and Methods in Physics Research A 453 (2000) 425–429

NUCLEAR INSTRUMENTS & METHODS IN PHYSICS RESEARCH Section A

www.elsevier.nl/locate/nima

An artificial retina for fast track finding

Luciano Ristori

INFN, Sezione di Pisa, Via Livornese 1291, I-56010 S. Piero a Grado, Pisa, Italy

Accepted 21 June 2000

Abstract

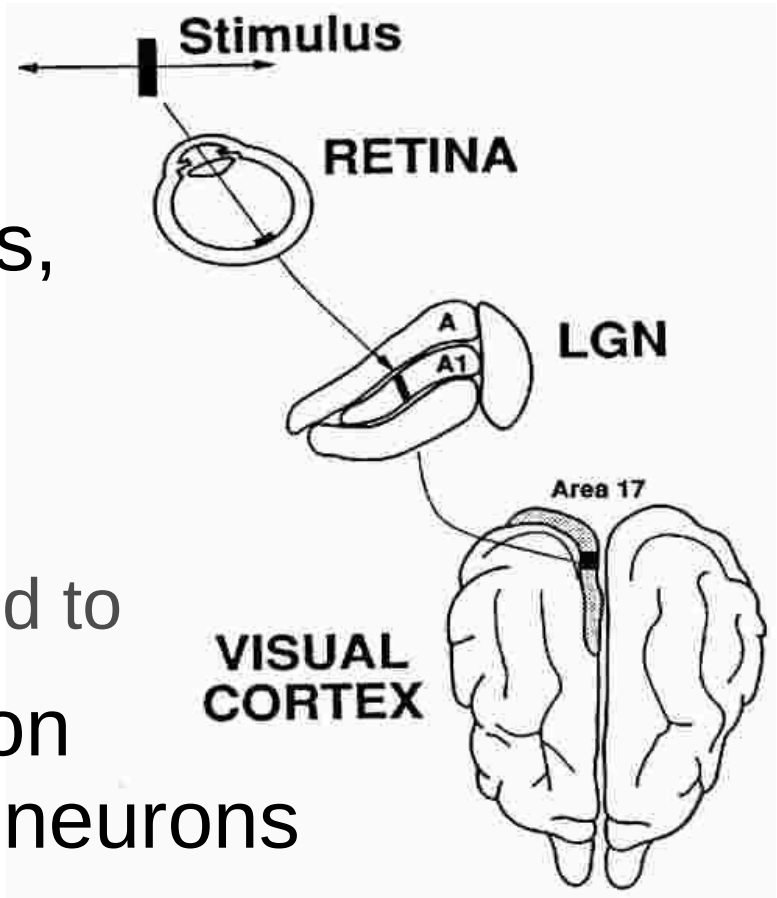
A new approach is proposed for fast track finding in position-sensitive detectors. The basic working principle is modeled on what is widely believed to be the low-level mechanism used by the eye to recognize straight edges. A number of receptors are tuned such that each one responds to a different range of track orientations, each track actually fires several receptors and an estimate of the orientation is obtained through interpolation. The feasibility of a practical device based on this principle and its possible implementation using currently available digital logic is discussed.

Keywords: Trigger; Electronics; Neural networks

Visual cortex

A. Abba et al 2015 JINST **10** C03008

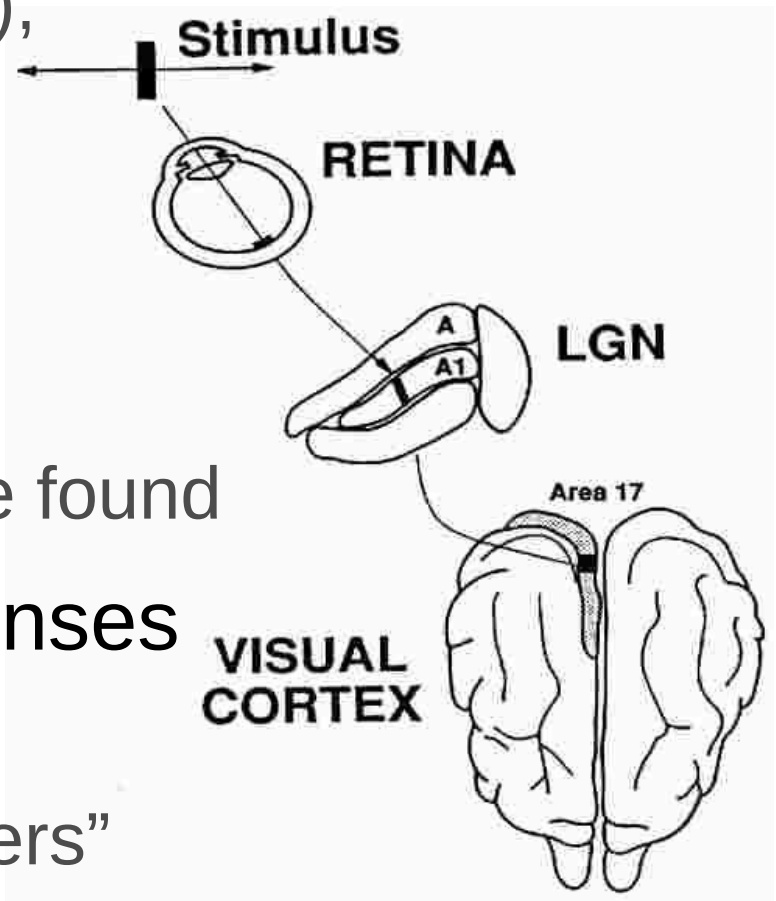
- Inspired by mechanism of visual receptive fields
  - line & edge detection areas of visual cortex
- There are neurons tuned to recognize a specific shape on specific region of the retina: **receptive field**
- All neurons react to a stimulus, each with different strength
  - proportional to how close the shape of the stimulus is to the shape for which neuron is tuned to
- the brain performs interpolation between the responses of all neurons



Retina

A. Abba et al 2015 JINST **10** C03008

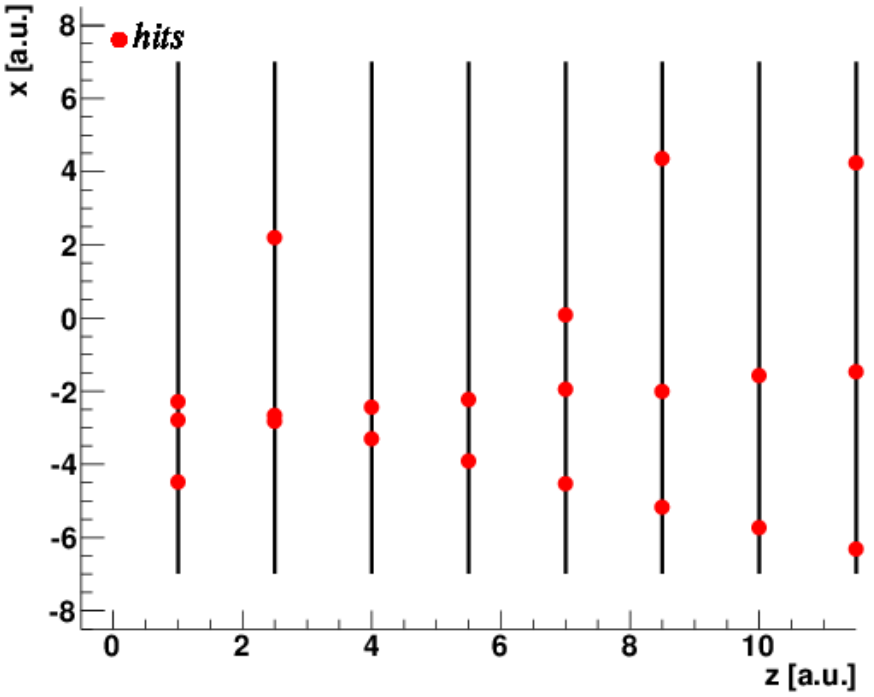
- The algorithm introduces a grid of units: '**cells**'
  - each corresponds to a specific track pattern configuration, as: position and angle
  - At each new observation (hits), each cell measures correspondence between its pattern and the input
- Massively parallel
  - no serialization until tracks are found
- Interpolation of analog responses
  - saves internal storage
  - easy to deal with "missing layers"

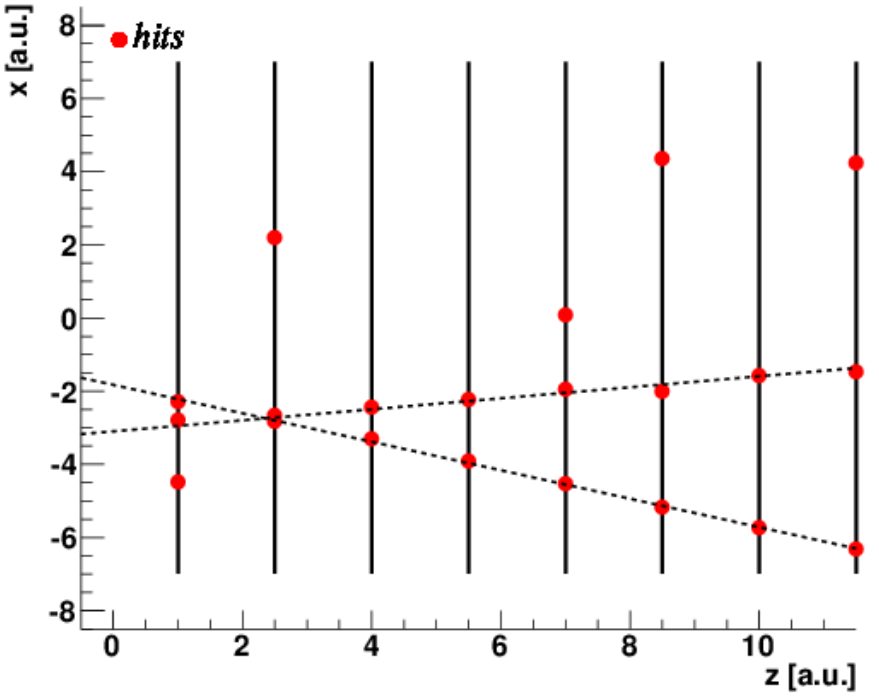


A simple case

A. Abba et al 2015 JINST **10** C03008

- Reconstruction of tracks w/o magnetic field using single-coordinate parallel detector
  - Streight lines
    x = mz + q → 2D space parameter **(m, q)**
  - Tune “receptive fields” to cover all values of **(m, q)**

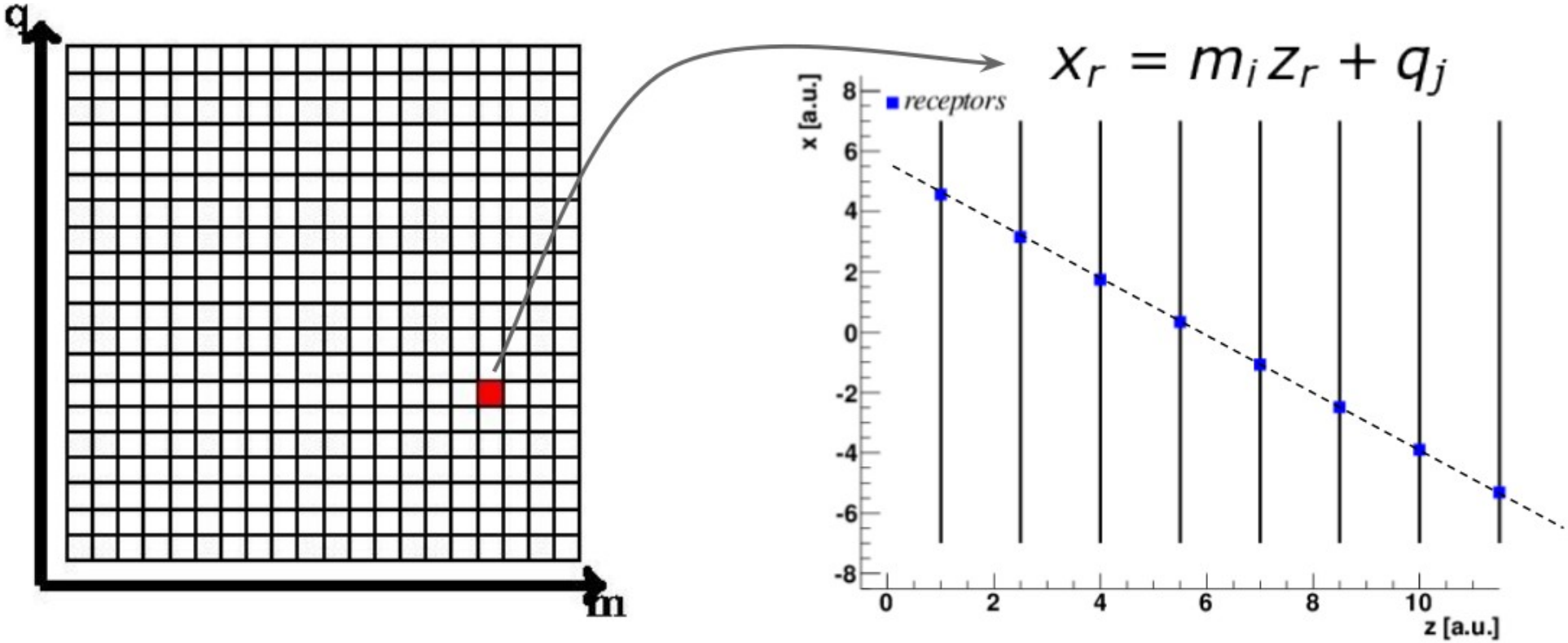




Detector mapping

A. Abba et al 2015 JINST **10** C03008

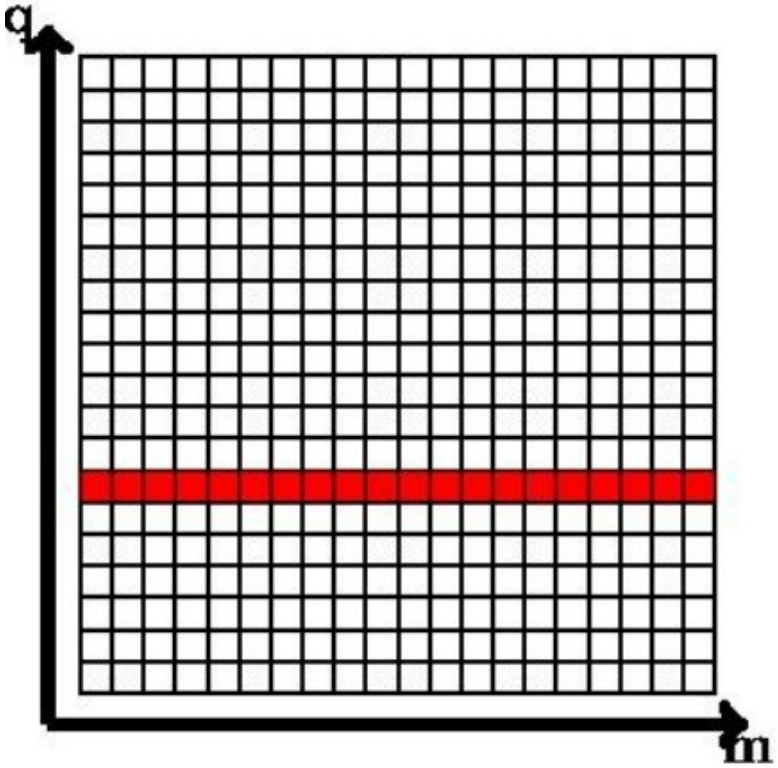
- Discretize track parameter space in "**cells**"
  - The center of each cell identifies a track in the real space that intersects detector layers in "**receptors**"
  - Each cellular unit corresponds to n (=number of layers) cellular receptors **(z_r, x_r)** (r runs over the layers)

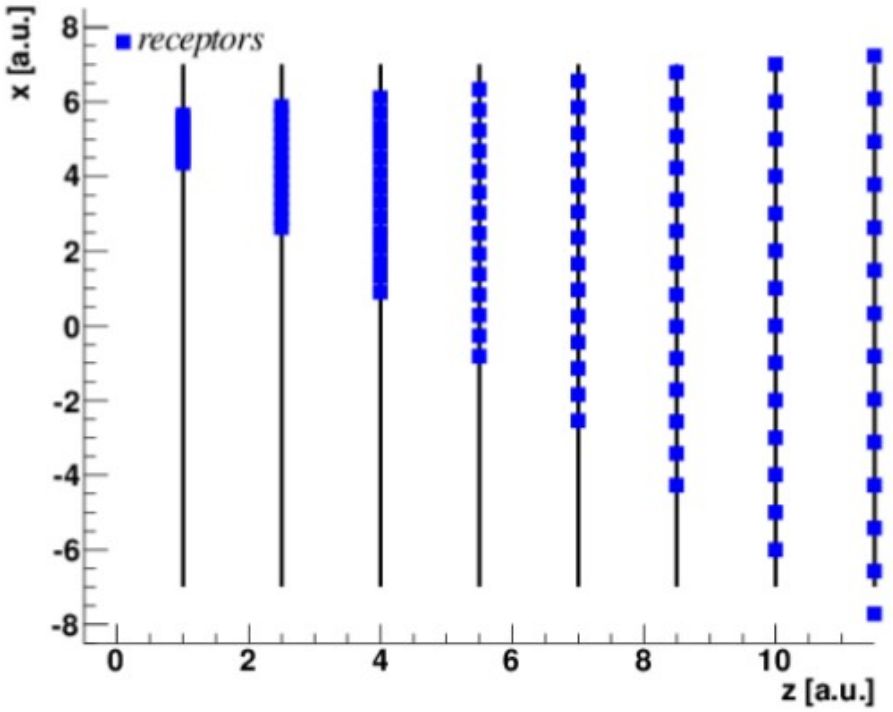


Detector mapping

A. Abba et al 2015 JINST **10** C03008

- Discretize track parameter space in "**cells**"
  - The center of each cell identifies a track in the real space that intersects detector layers in "**receptors**"
  - Each cellular unit corresponds to n (=number of layers) cellular receptors **(z_r, x_r)** (r runs over the layers)

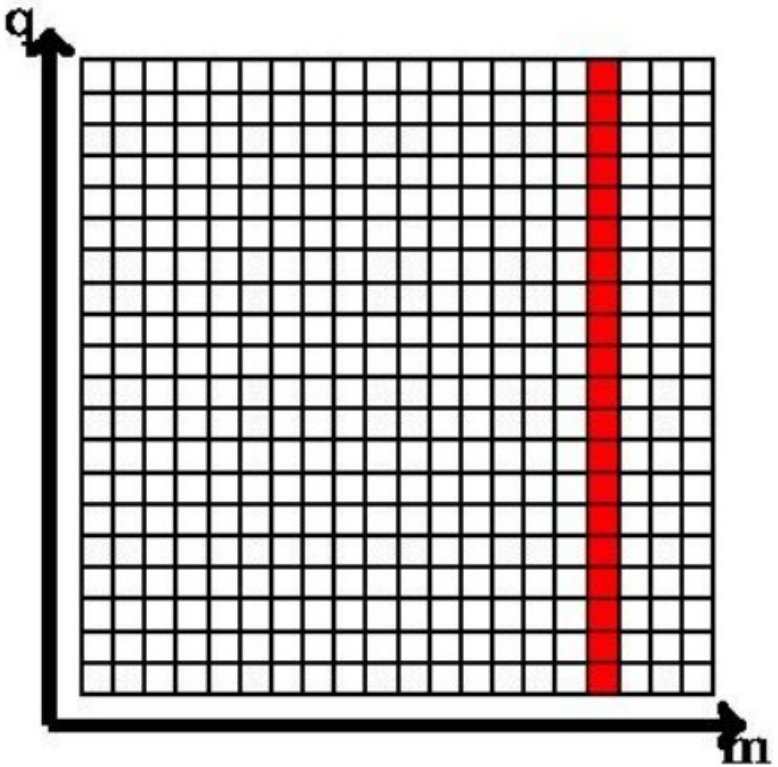


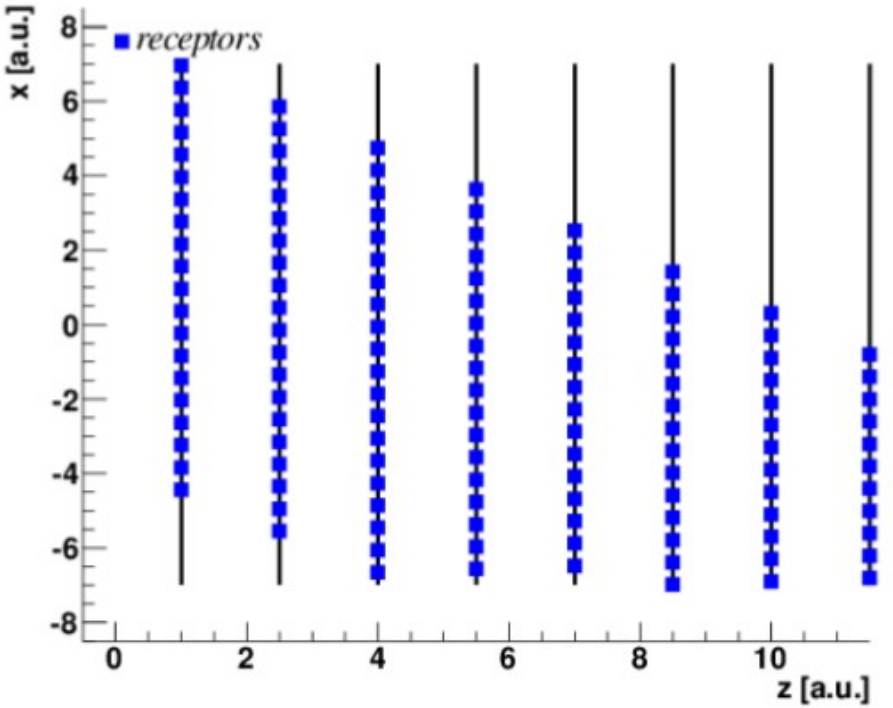


Detector mapping

A. Abba et al 2015 JINST **10** C03008

- Discretize track parameter space in "**cells**"
  - The center of each cell identifies a track in the real space that intersects detector layers in "**receptors**"
  - Each cellular unit corresponds to n (=number of layers) cellular receptors **(z_r, x_r)** (r runs over the layers)

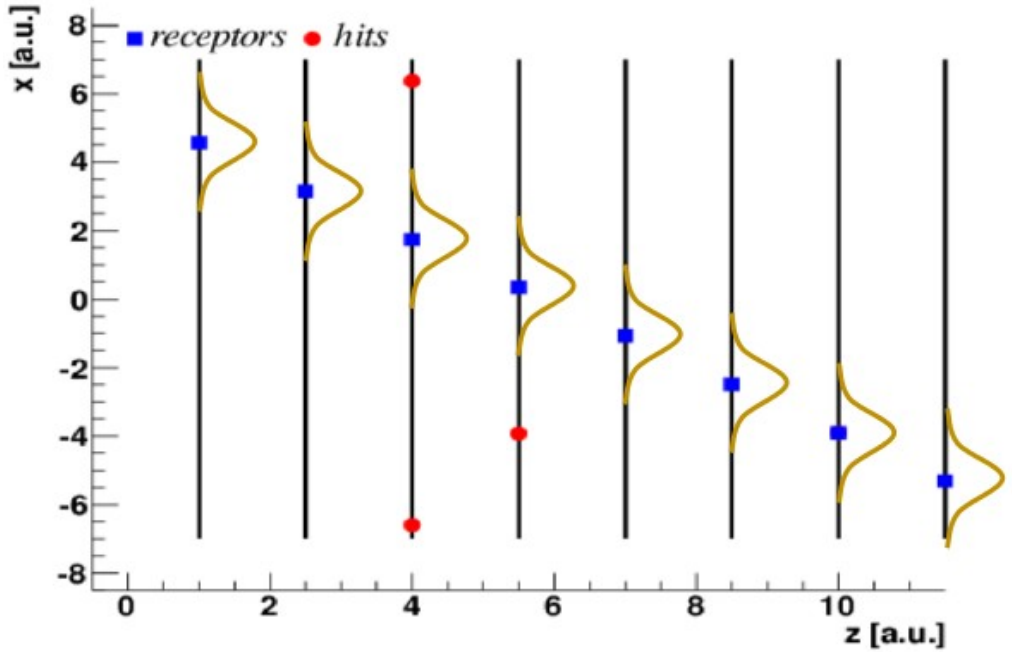




Basic principle

A. Abba et al 2015 JINST **10** C03008

- Once the space of relevant template patterns/tracks is encoded into the device, for all the hits in the detector layers $(z_r, x_r)_k$ (due to real particles or noise)
  - the response R_{ij} of the cellular unit (m_i, q_j) is calculated summing over all hits and layers
  - R_{ij} represents the "excitation" of the receptive field



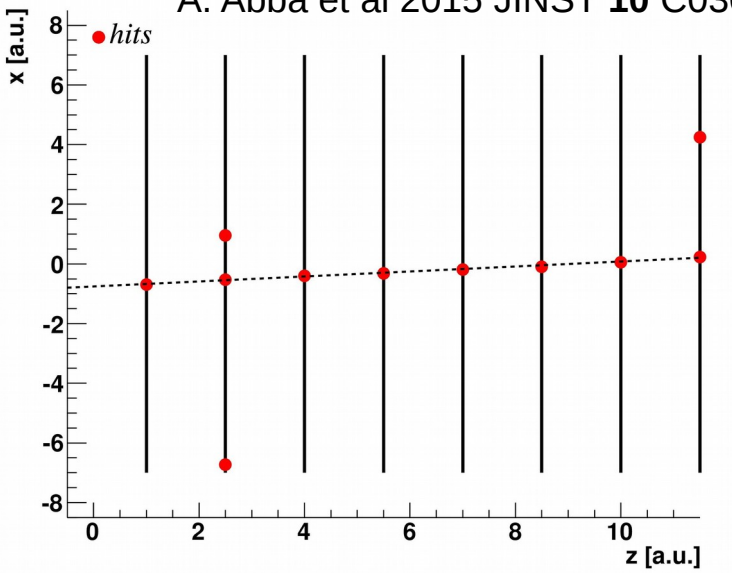
$$s^2_{ijkr} = (x^k_n - x^{ij}_r)^2$$

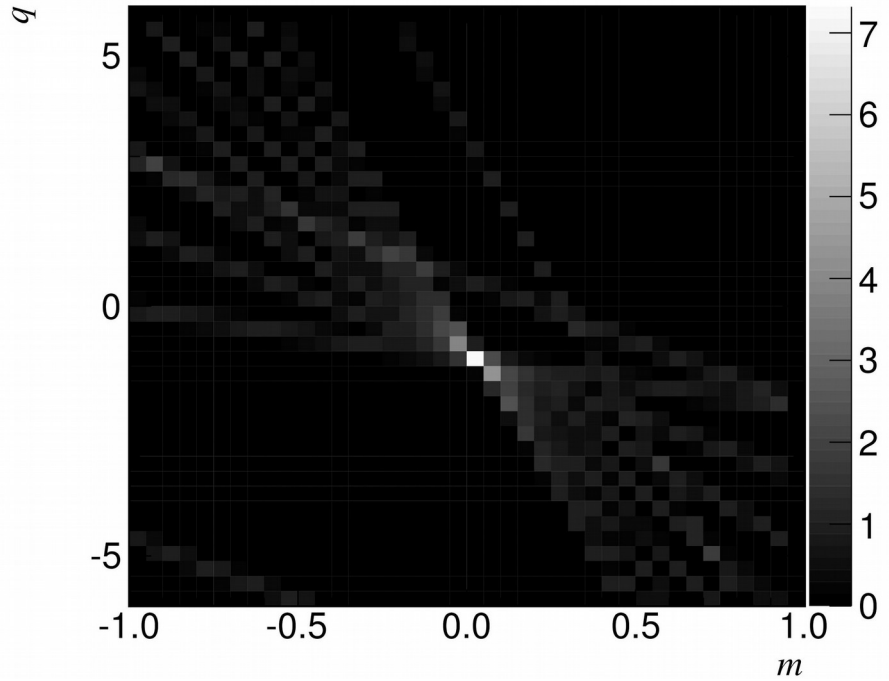
$$R_{ij} = \sum_{kr} \exp\left(-\frac{s^2_{ijkr}}{2\sigma^2}\right)$$

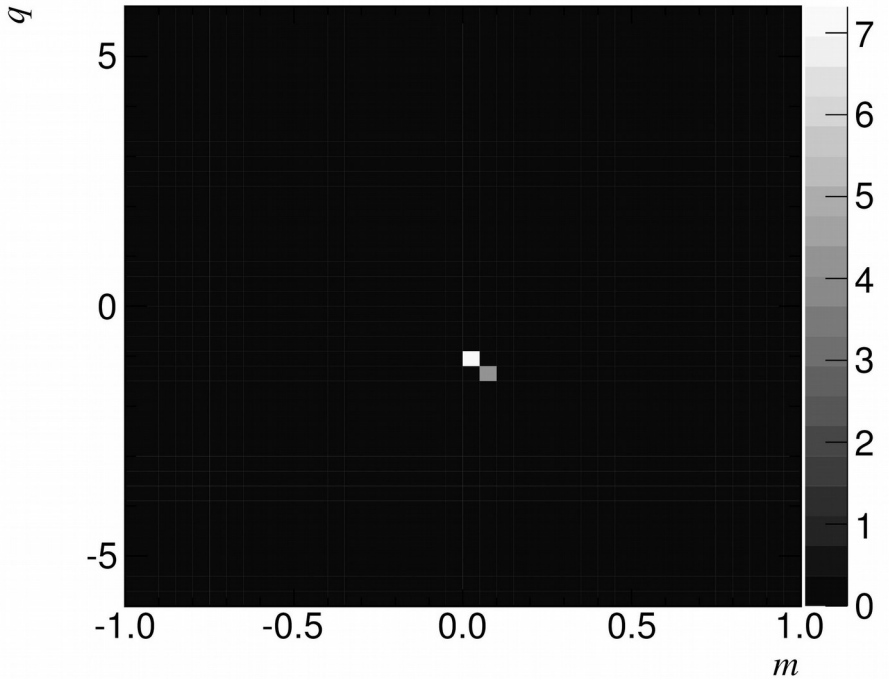
The retina response

- Once all cells are excited and Rij calculated
  - a track is identified by a local maximum in parameter space
  - And a threshold can be applied

A. Abba et al 2015 JINST **10** C03008





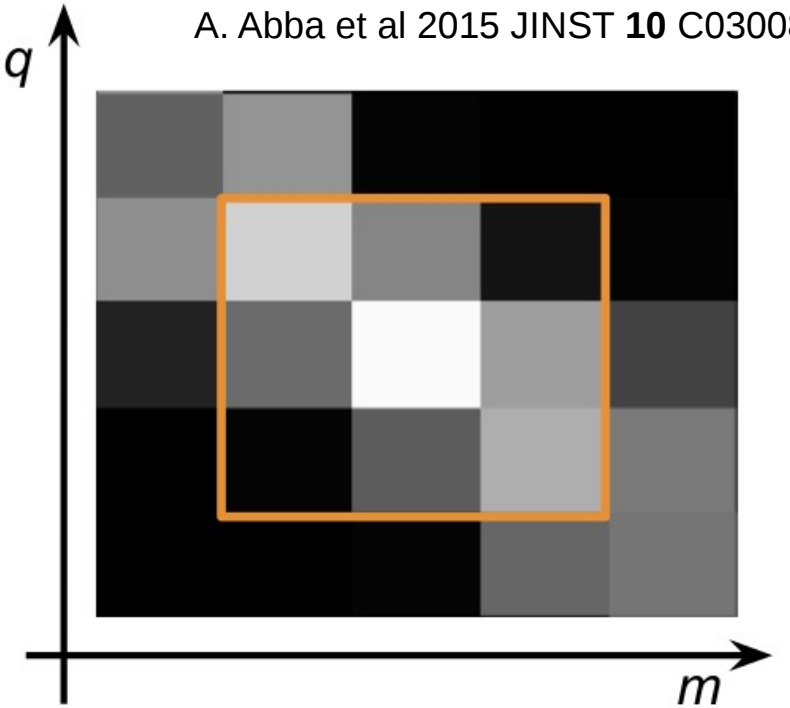


Parameter extraction

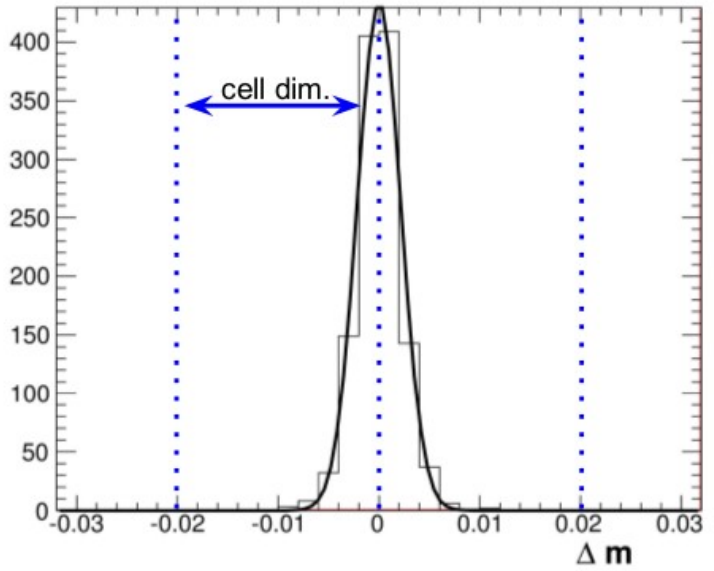
- Once local maxima, i.e. tracks, are found
  - parameter values are extracted by performing the centroid of the nearest cells

$$m = \frac{\sum_{ij} m_i w_{ij}}{\sum_{ij} w_{ij}} \qquad q = \frac{\sum_{ij} q_j w_{ij}}{\sum_{ij} w_{ij}}$$

A. Abba et al 2015 JINST **10** C03008



- A subcell resolution is achieved by interpolation
  - Important since it allows a coarse space granularity
  - → limits the number of cells



Associative memories

436 Nuclear Instruments and Methods in Physics Research A278 (1989) 436–440
North-Holland, Amsterdam

VLSI STRUCTURES FOR TRACK FINDING

Mauro DELL'ORSO

Dipartimento di Fisica, Università di Pisa, Piazza Torricelli 2, 56100 Pisa, Italy

Luciano RISTORI

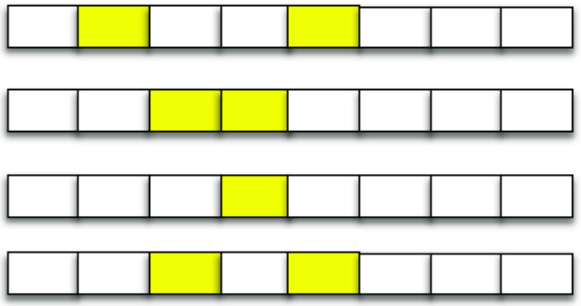
INFN Sezione di Pisa, Via Vecchia Livornese 582a, 56010 S. Piero a Grado (PI), Italy

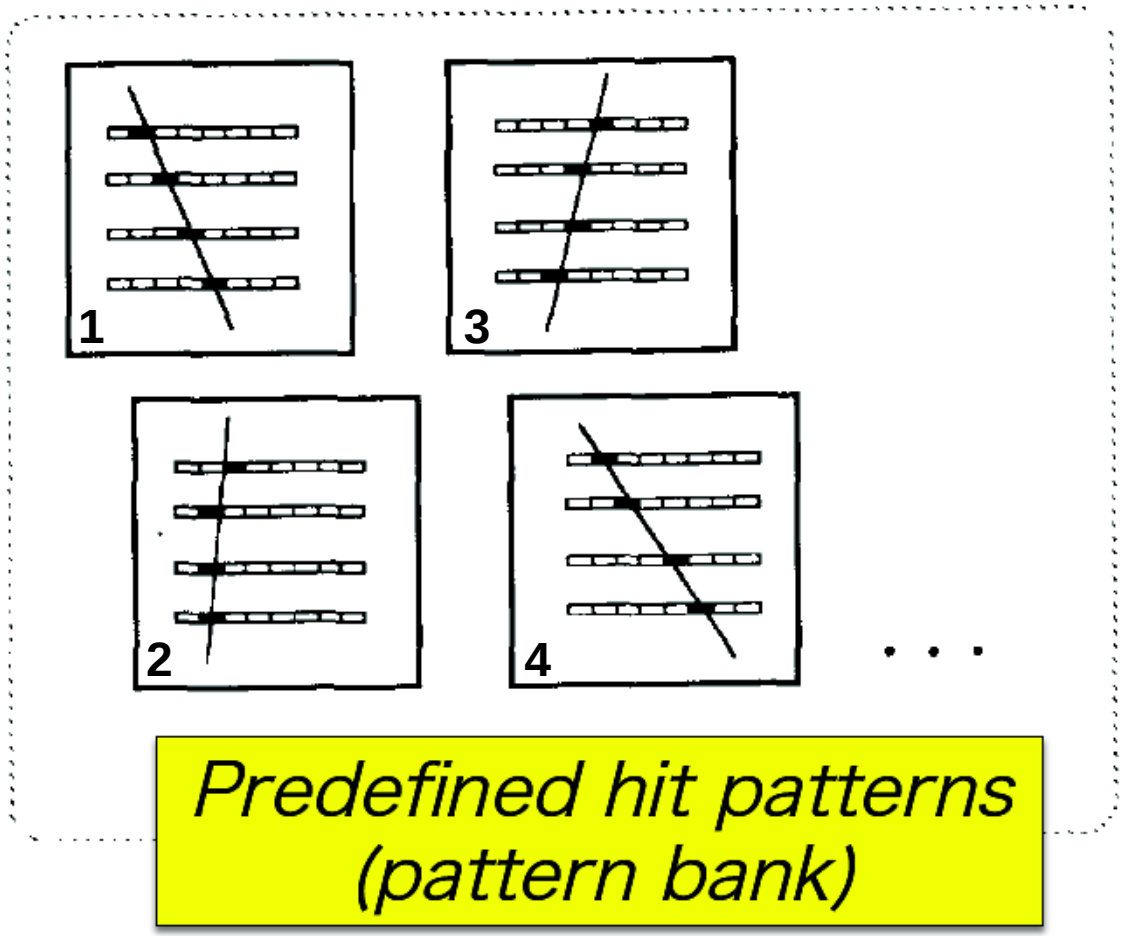
Received 24 October 1988

We discuss the architecture of a device based on the concept of *associative memory* designed to solve the track finding problem, typical of high energy physics experiments, in a time span of a few microseconds even for very high multiplicity events. This "machine" is implemented as a large array of custom VLSI chips. All the chips are equal and each of them stores a number of "patterns". All the patterns in all the chips are compared in parallel to the data coming from the detector while the detector is being read out.

Comparison with pattern bank

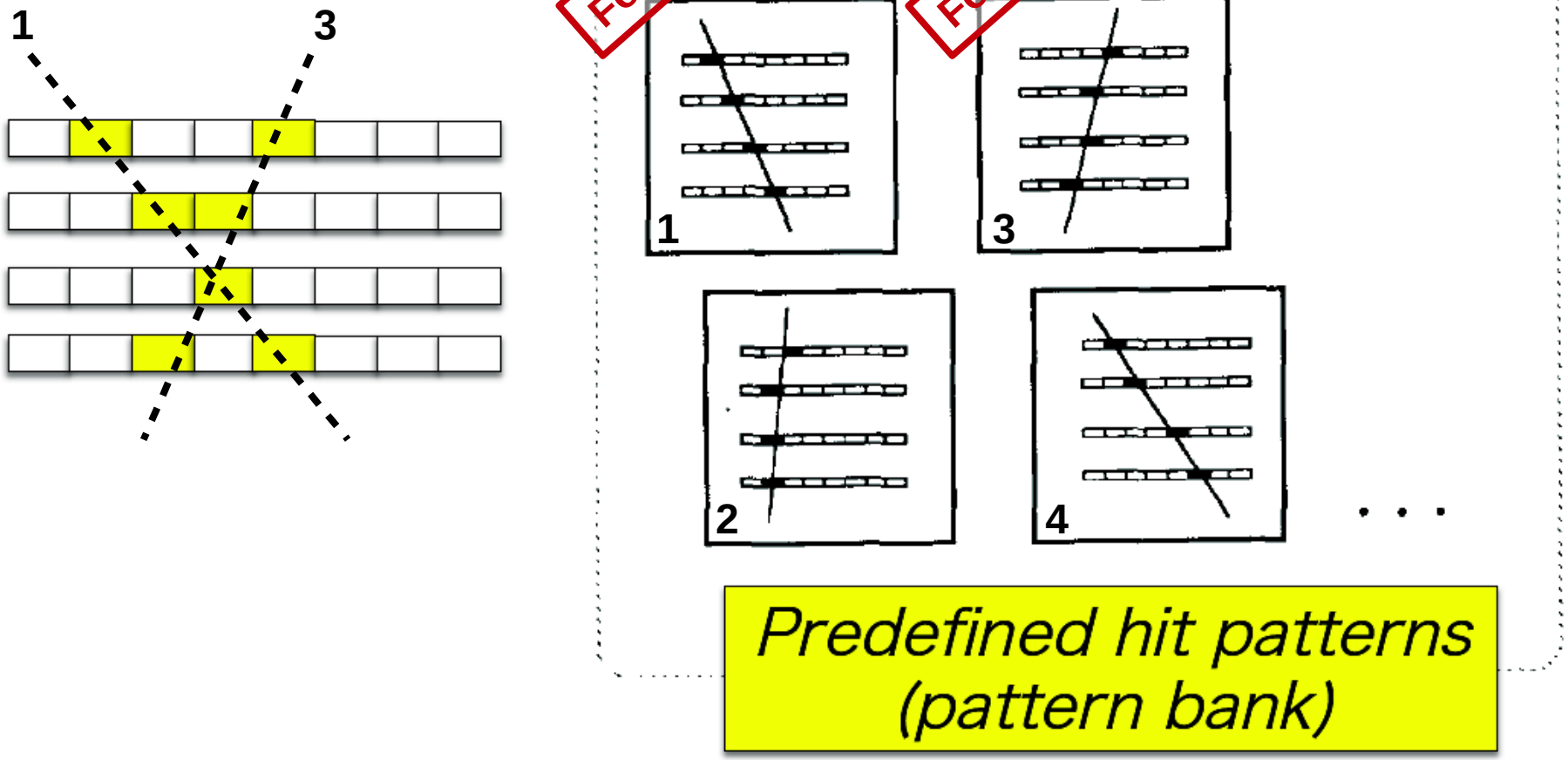
- Comparison between predefined hit patterns from track simulation with hit patterns from data





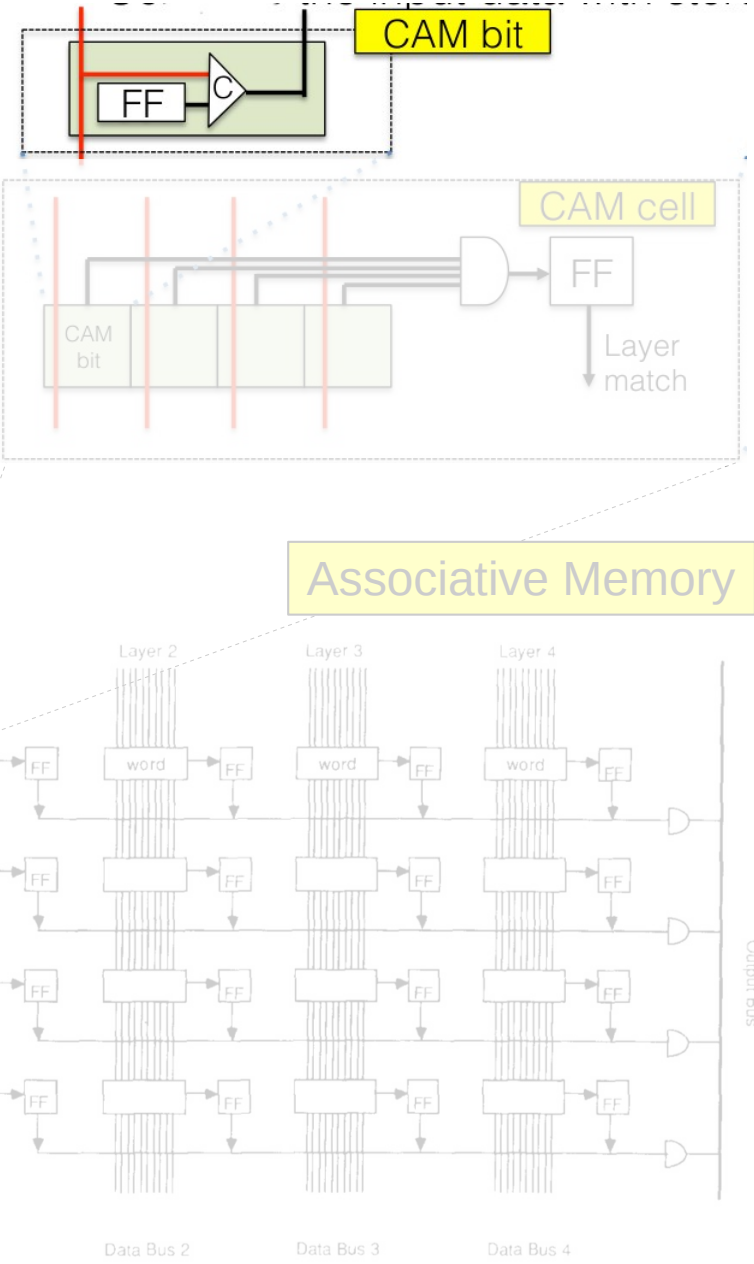
Comparison with pattern bank

- Comparison between predefined hit patterns from track simulation with hit patterns from data



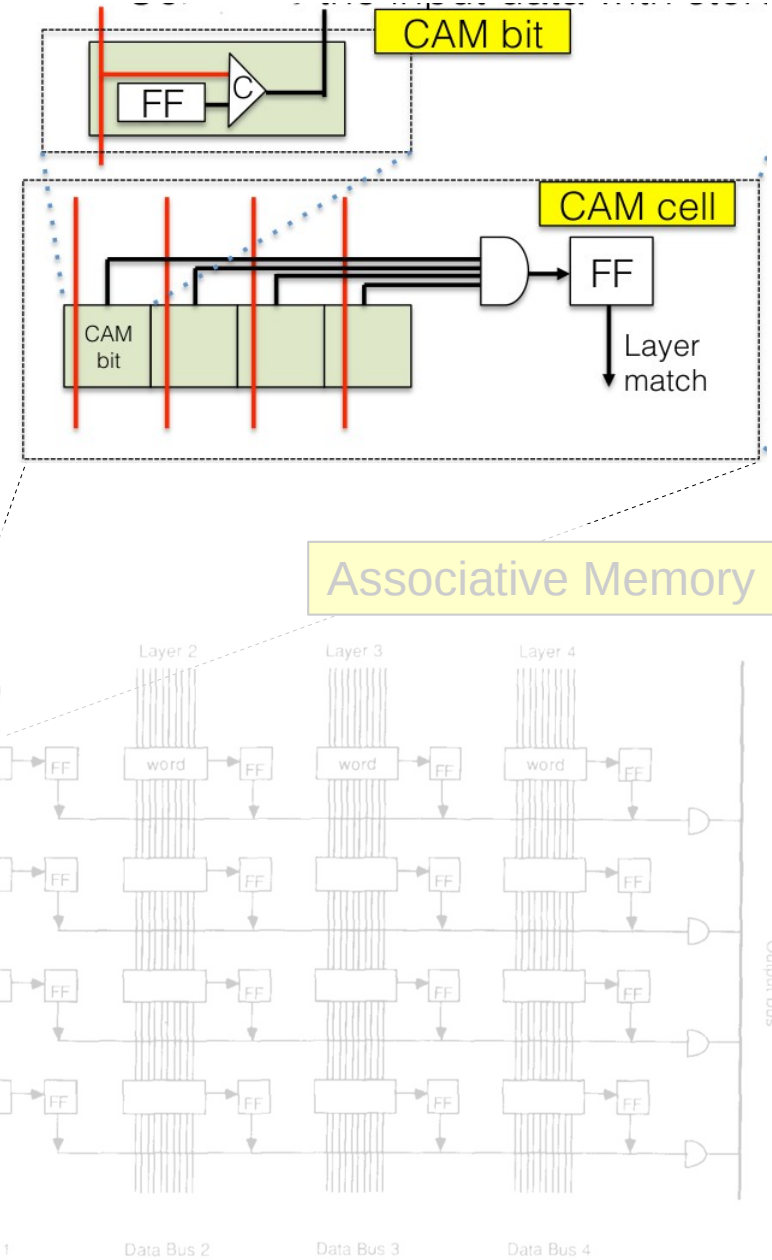
Associative memories

- **C**ontent **A**ccessible **M**emories
- CAM bit
  - Store the data
  - Compare input with stored bit
- CAM cell
  - Array of 18 CAM bits
  - Compare input w/ stored word
  - Word: address/offset on a detector layer
- Associative memory
  - Union of 8 CAM cells
  - One for detector layers



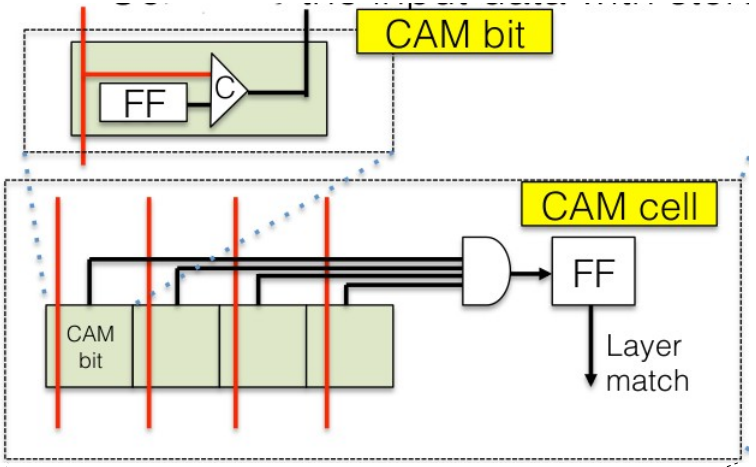
Associative memories

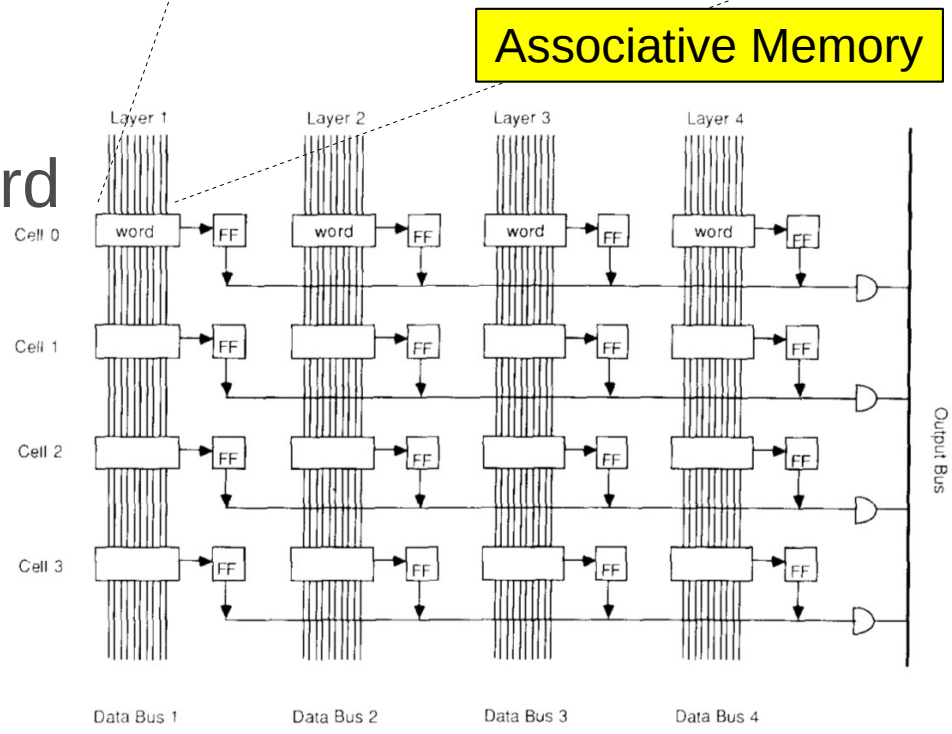
- **C**ontent **A**ccessible **M**emories
- CAM bit
  - Store the data
  - Compare input with stored bit
- CAM cell
  - Array of 18 CAM bits
  - Compare input w/ stored word
  - Word: address/offset on a detector layer
- Associative memory
  - Union of 8 CAM cells
  - One for detector layers



Associative memories

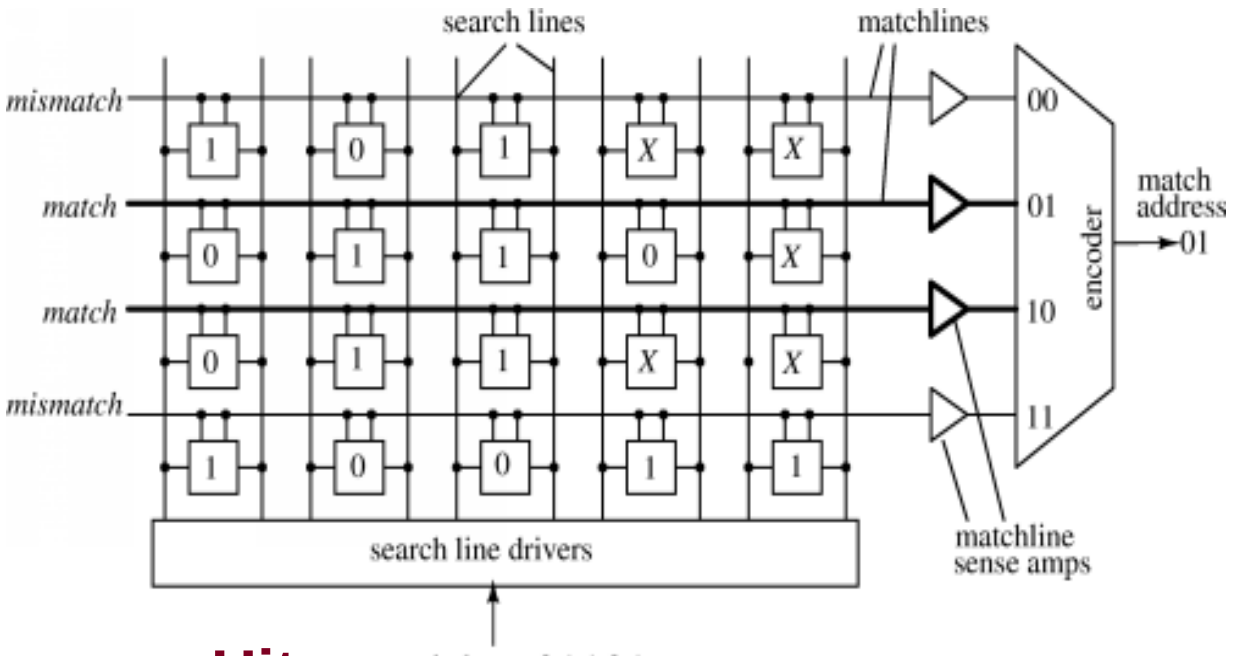
- **C**ontent **A**ccessible **M**emories
- CAM bit
  - Store the data
  - Compare input with stored bit
- CAM cell
  - Array of 18 CAM bits
  - Compare input w/ stored word
  - Word: address/offset on a detector layer
- Associative memory
  - Union of 8 CAM cells
  - One for detector layers





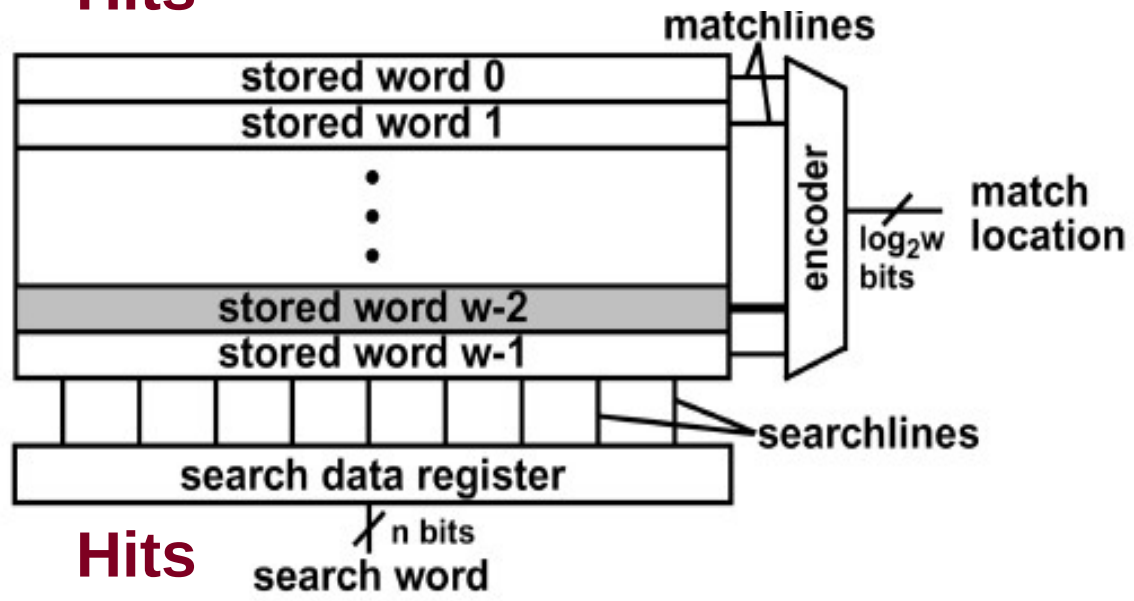
Associative memories

Patterns



Hits

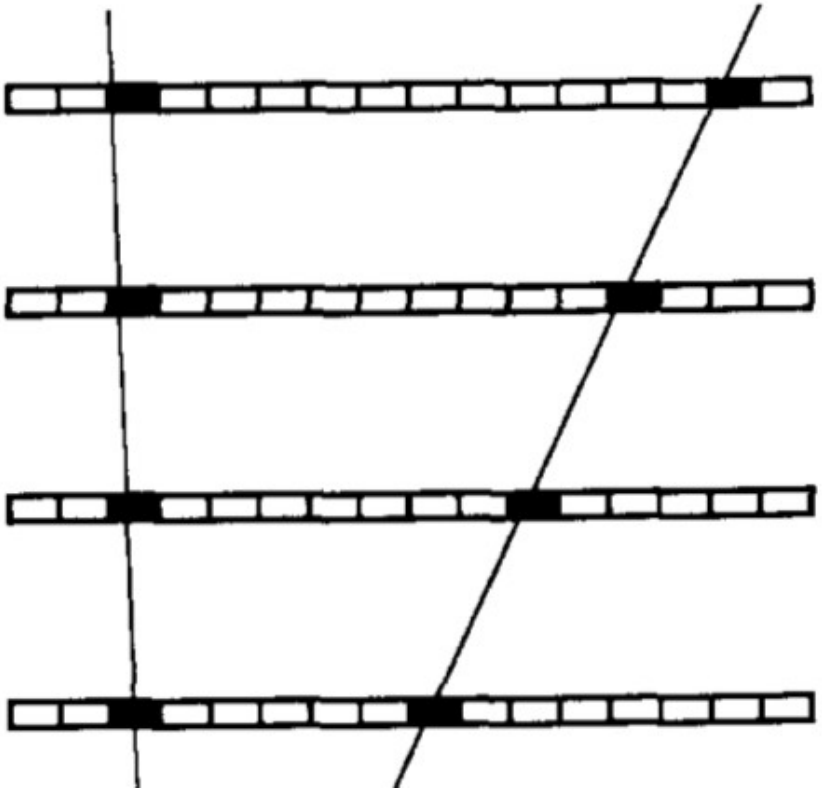
Patterns



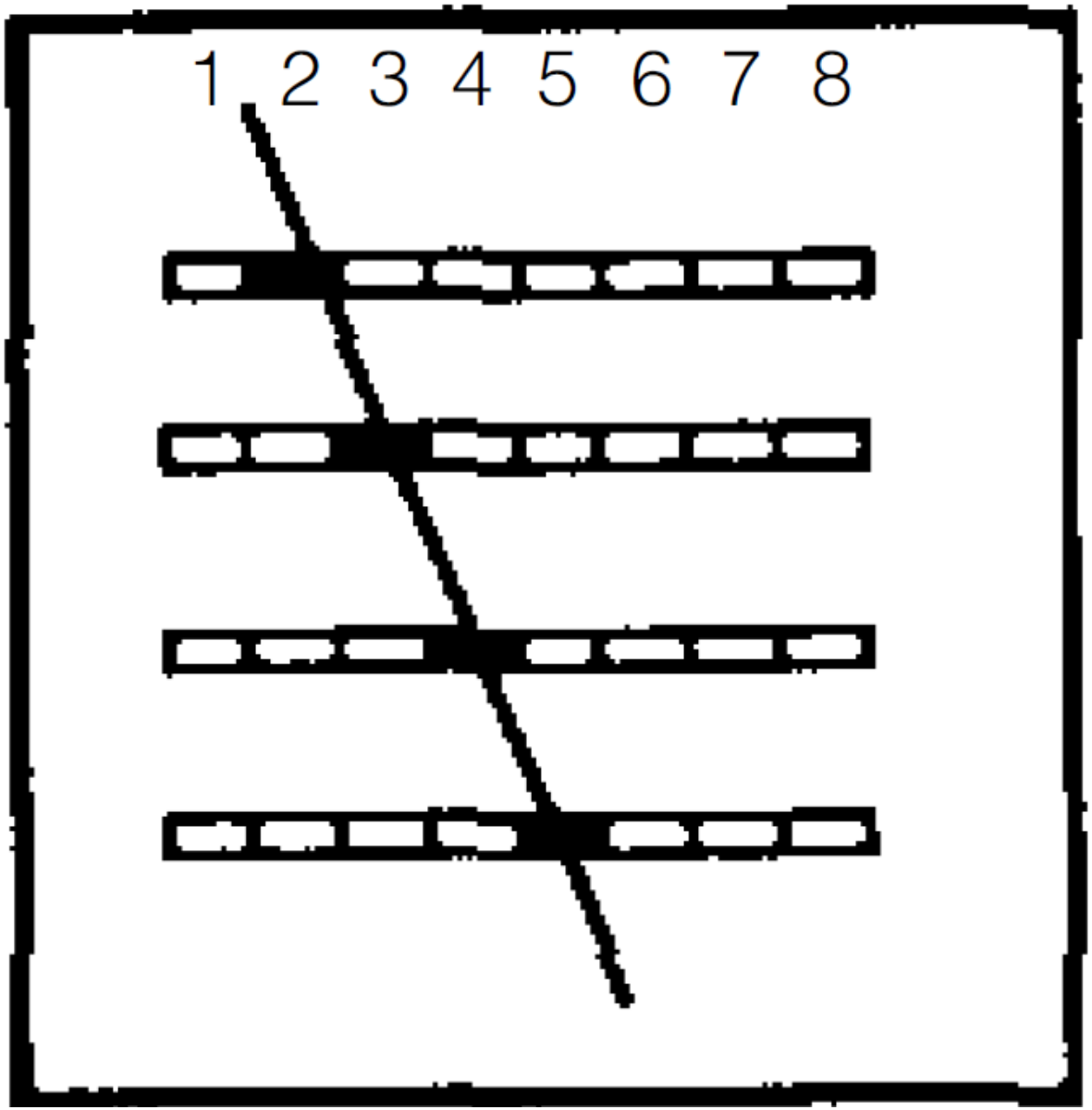
Hits

AM example

- Working principle example
  - four layers of 8 bins
  - NB: bins have reduced granularity compared to detector resolution



Pattern bank: pattern 1



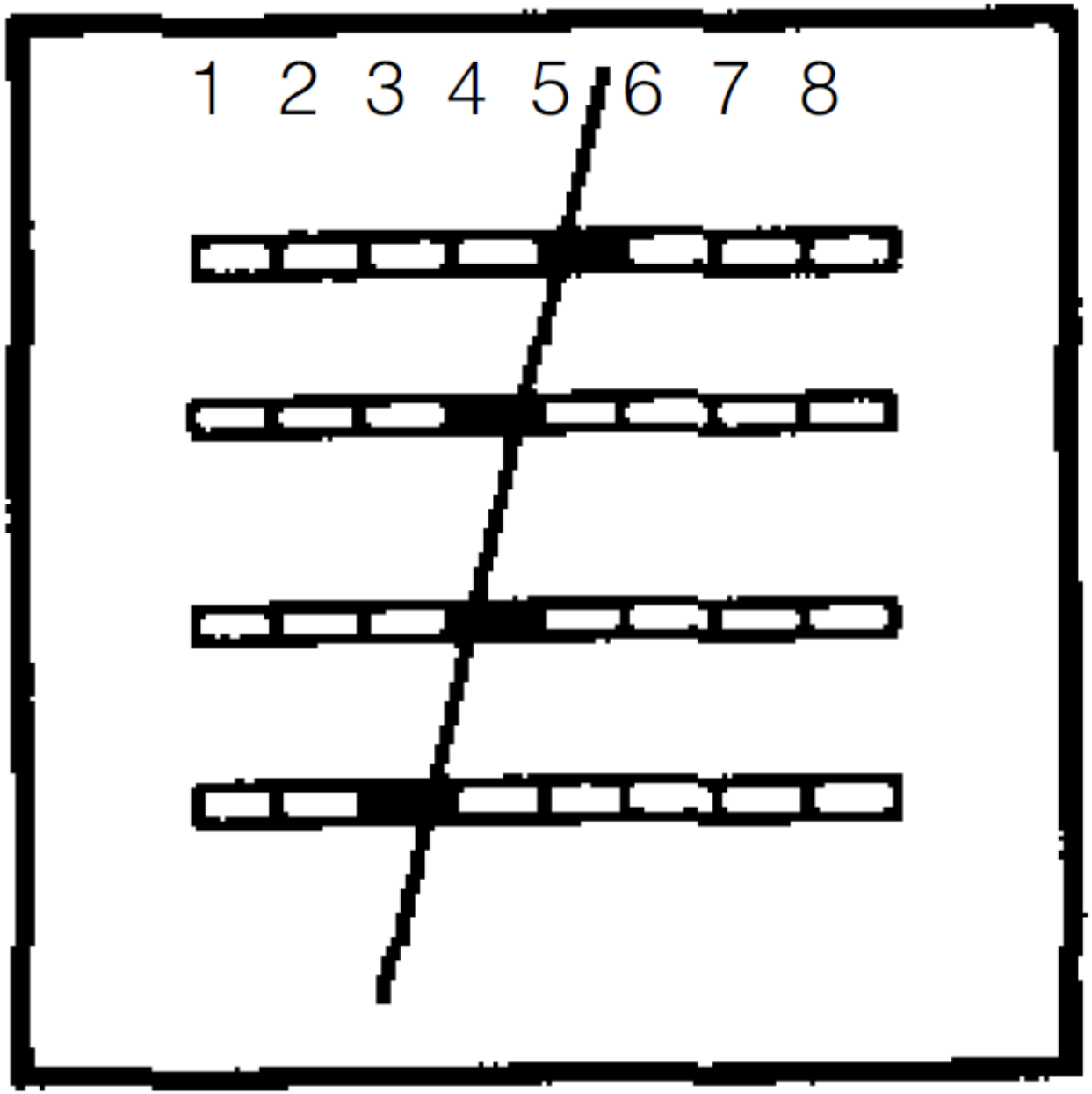
2 @ Layer4

3 @ Layer3

4 @ Layer2

5 @ Layer1

Pattern bank: pattern 4



5 @ Layer4

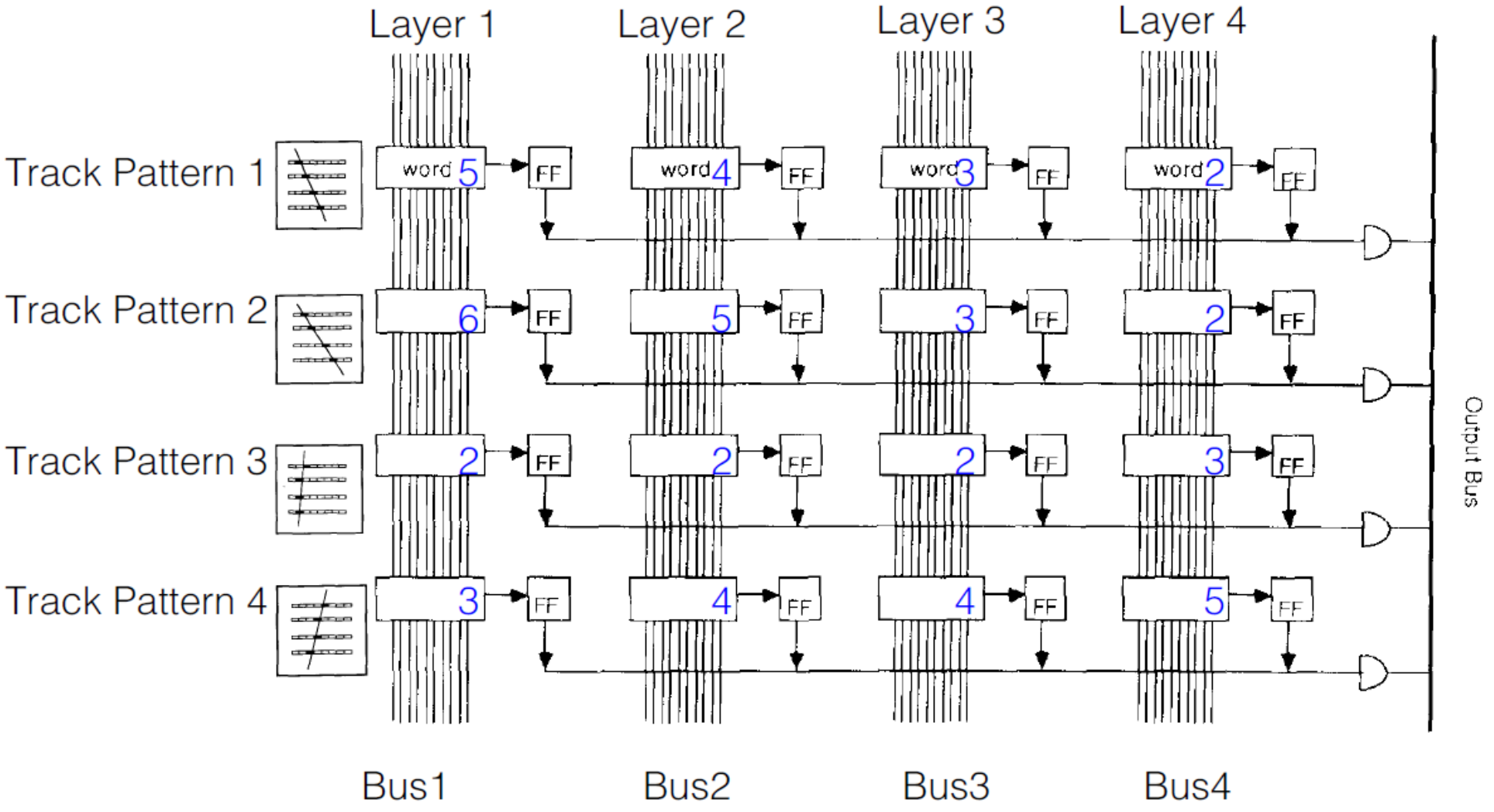
4 @ Layer3

4 @ Layer2

3 @ Layer1

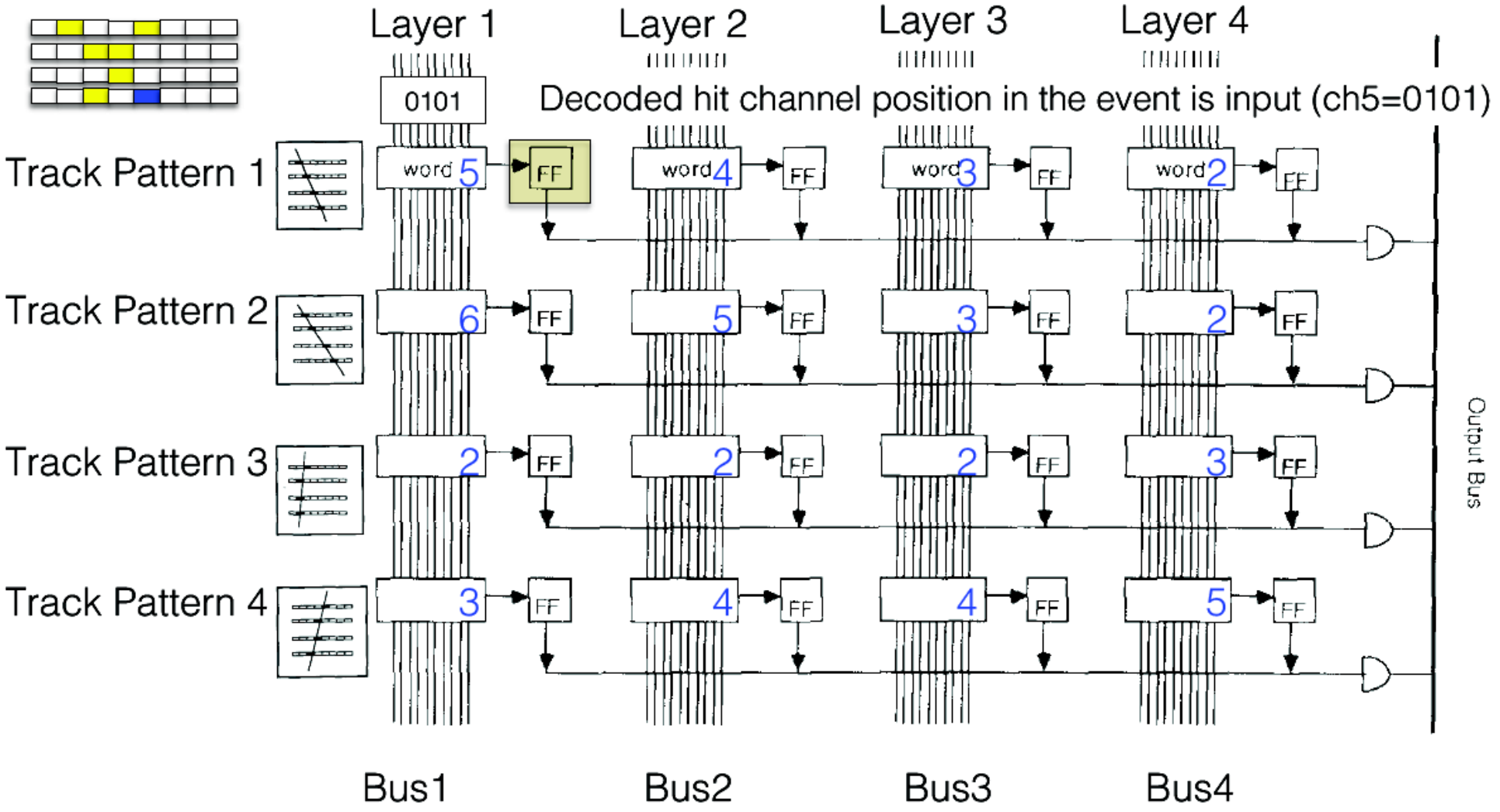
Associative memory example

- Bank filled with 4 patterns
  - 4 layers



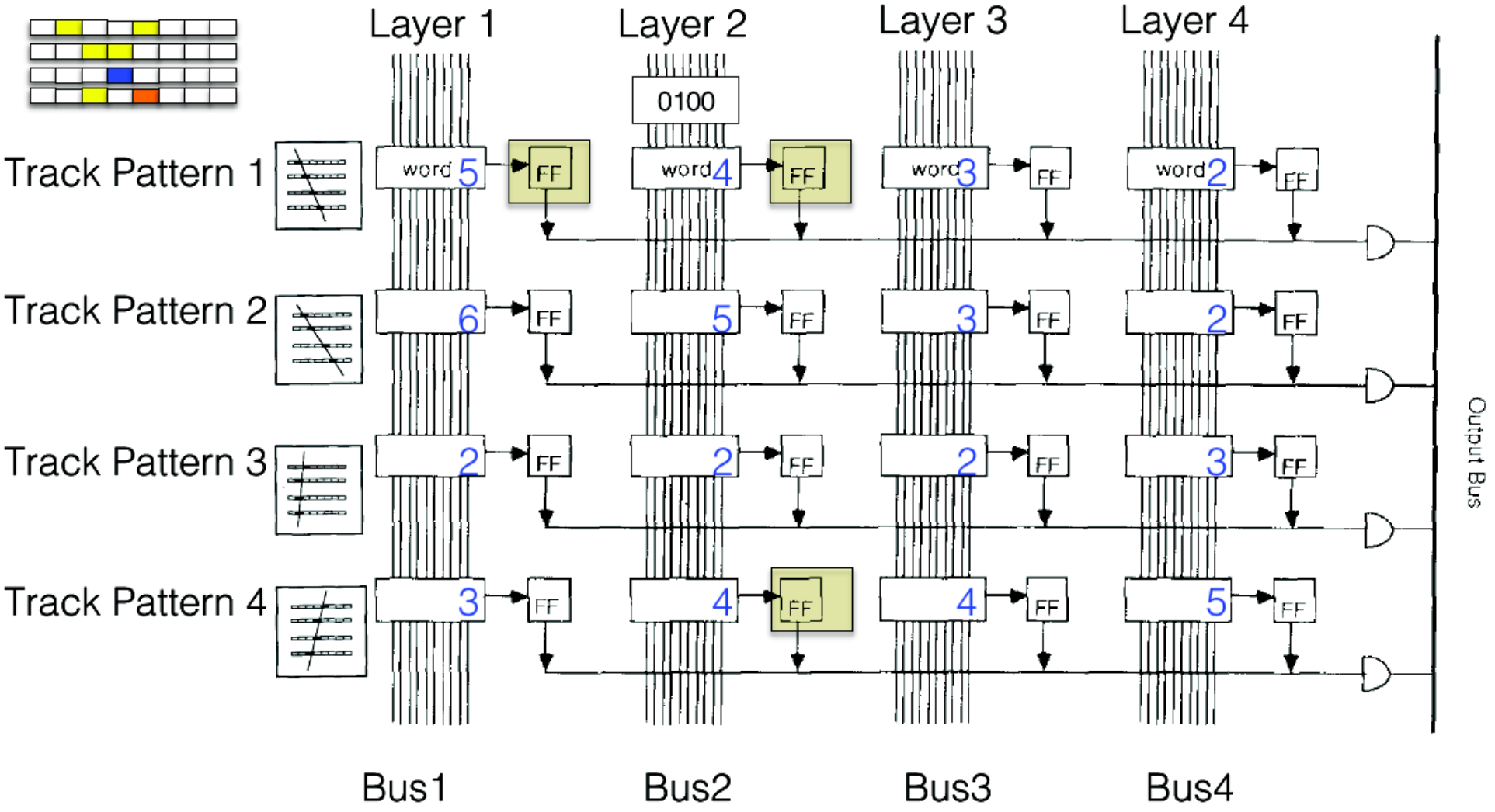
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



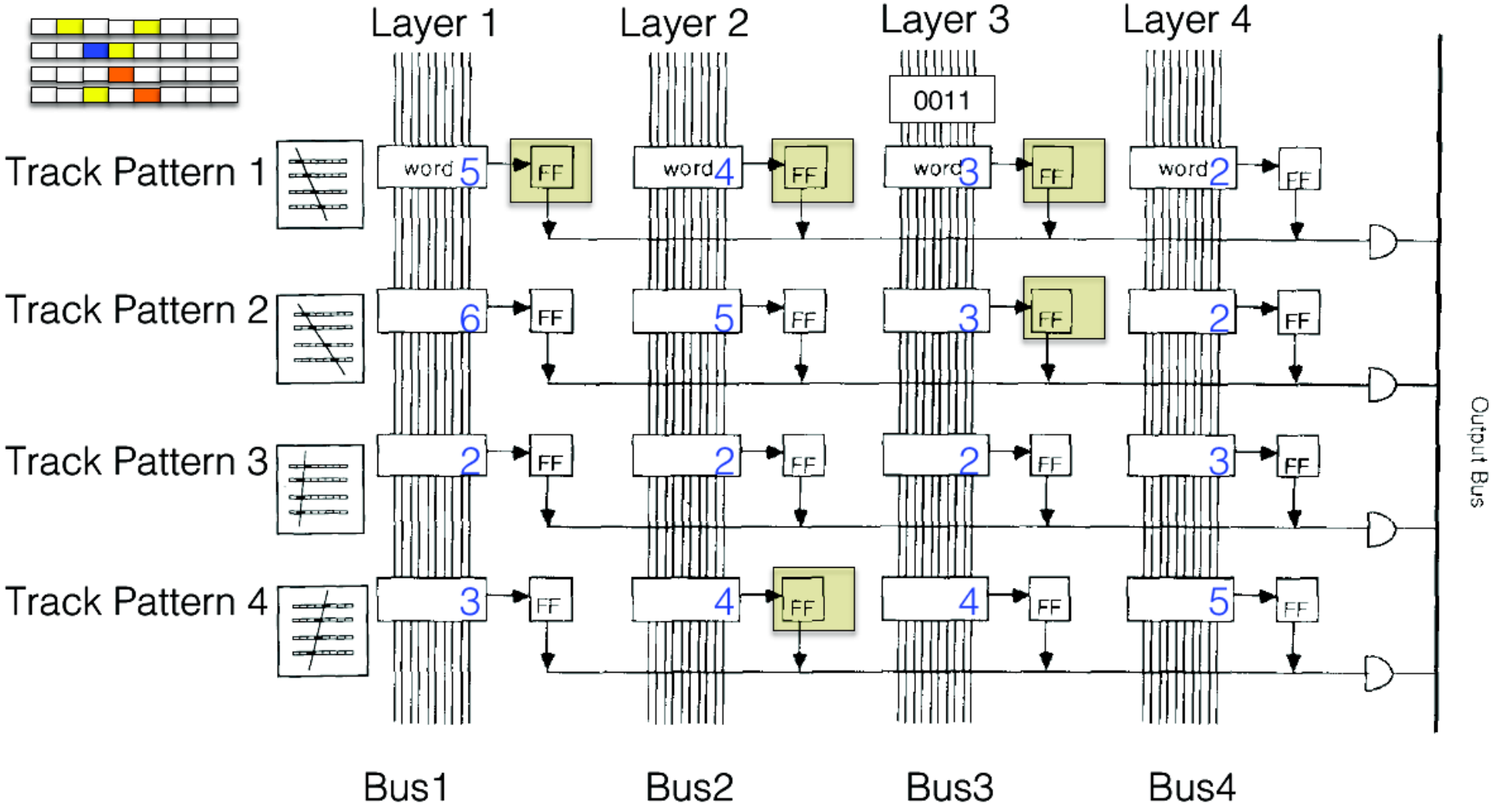
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



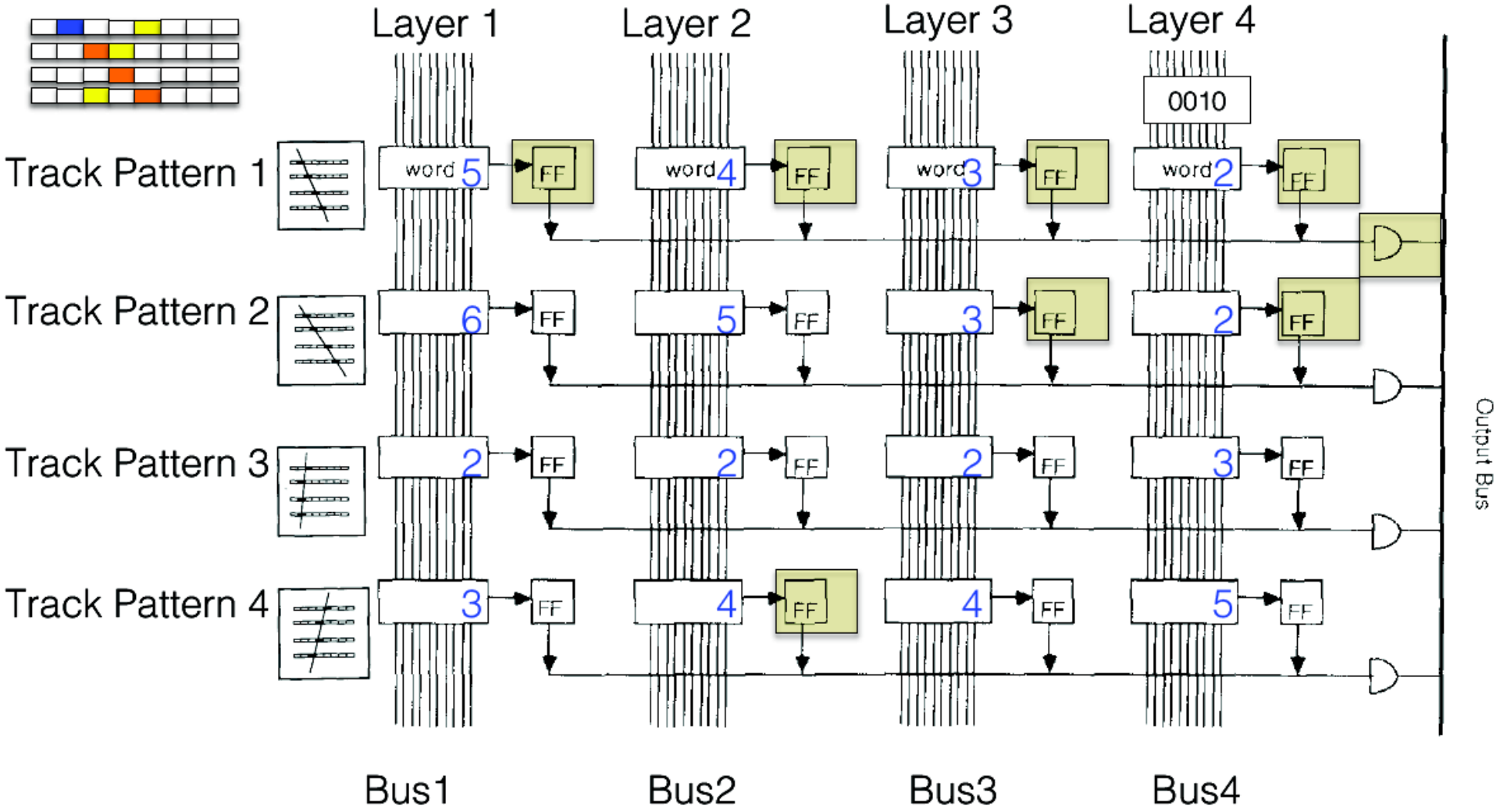
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



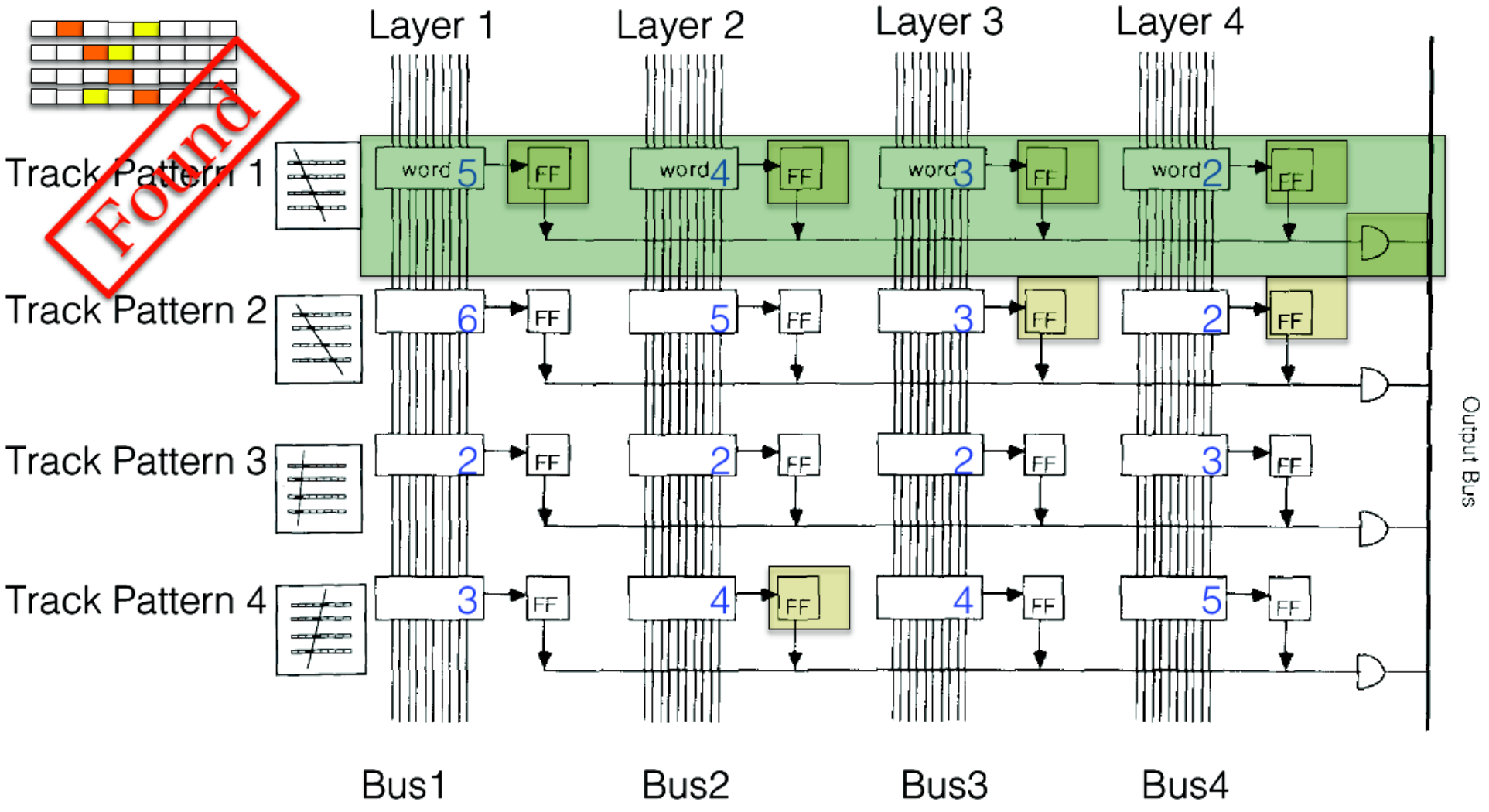
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



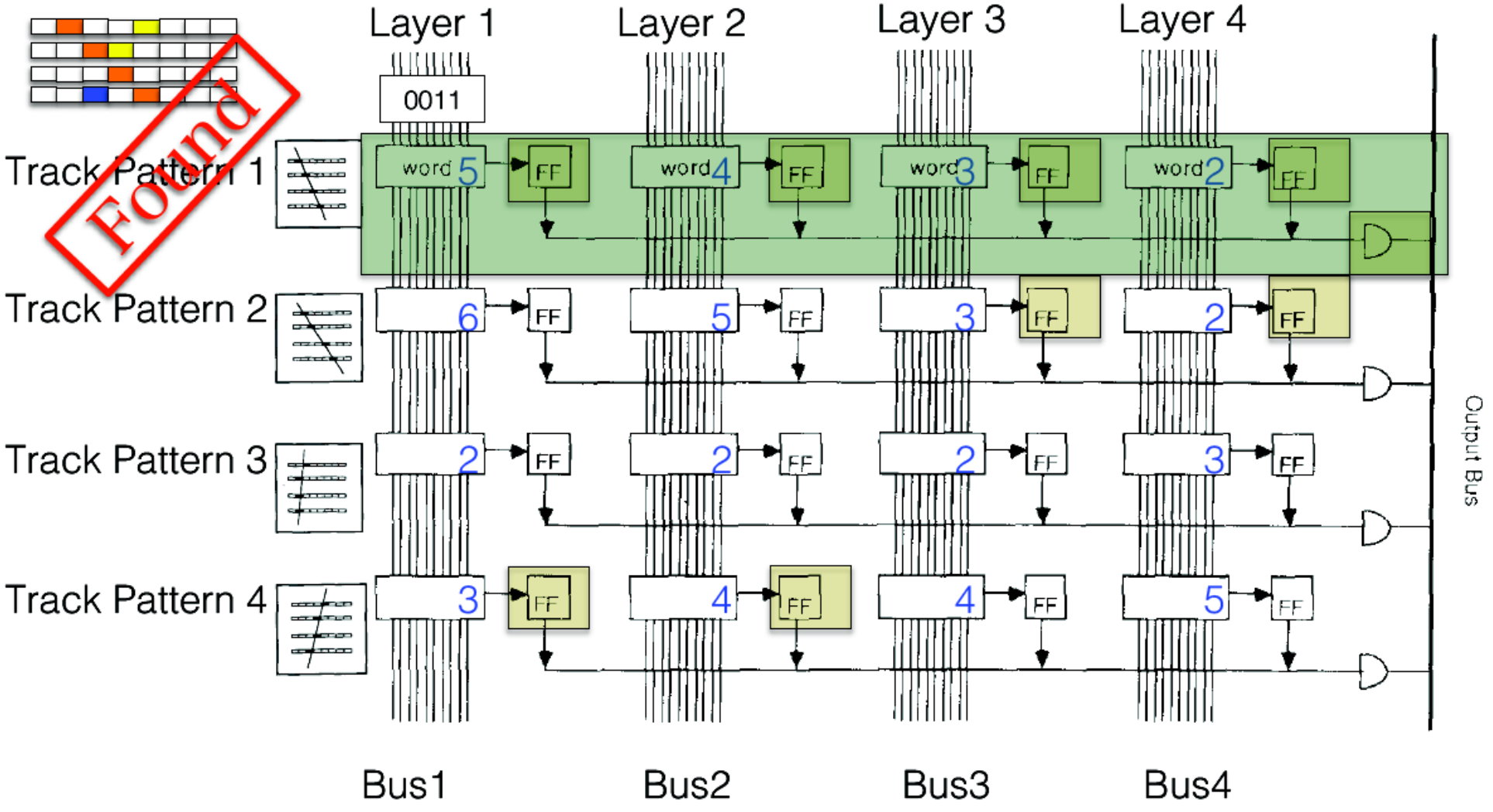
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



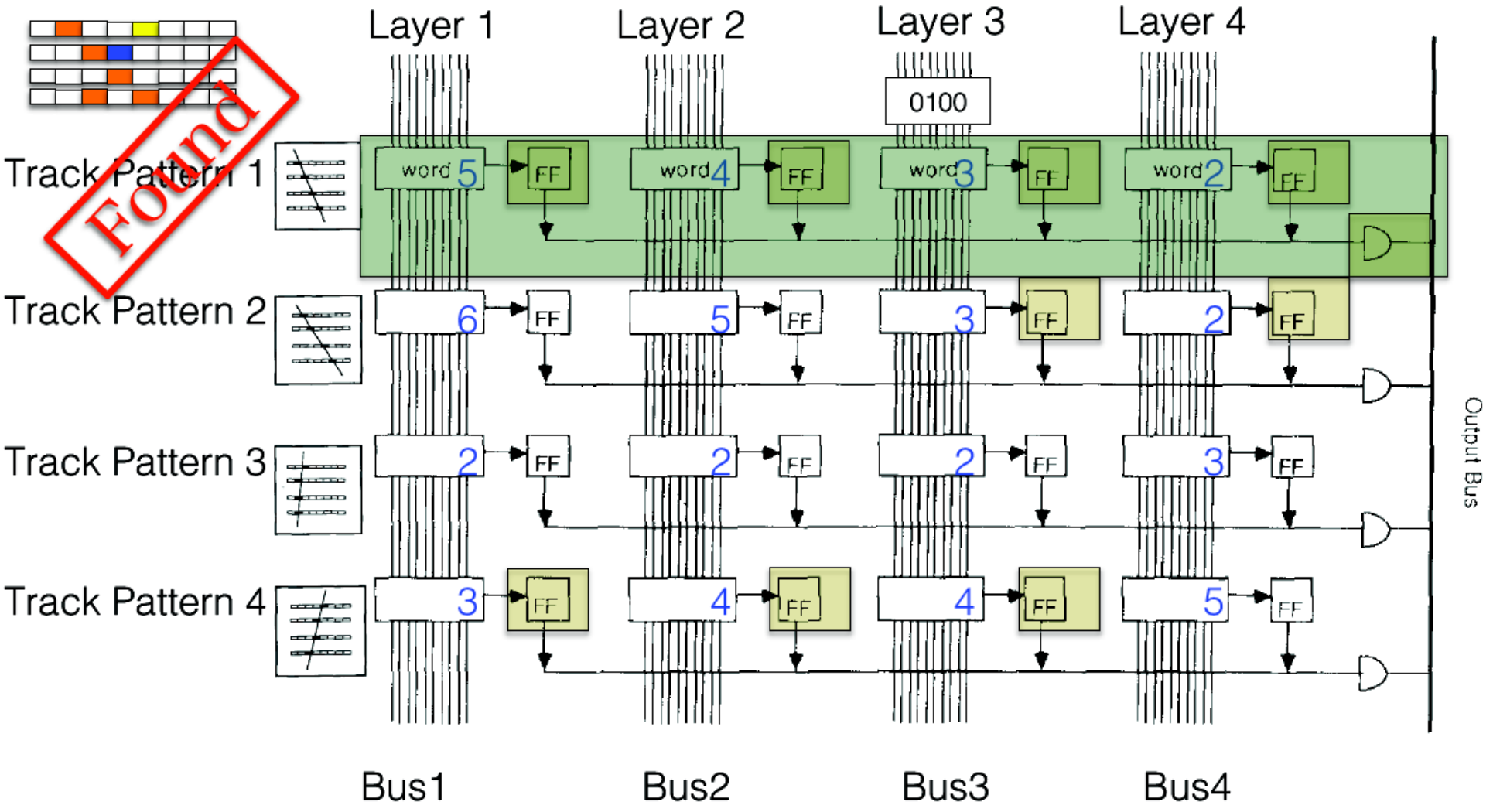
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



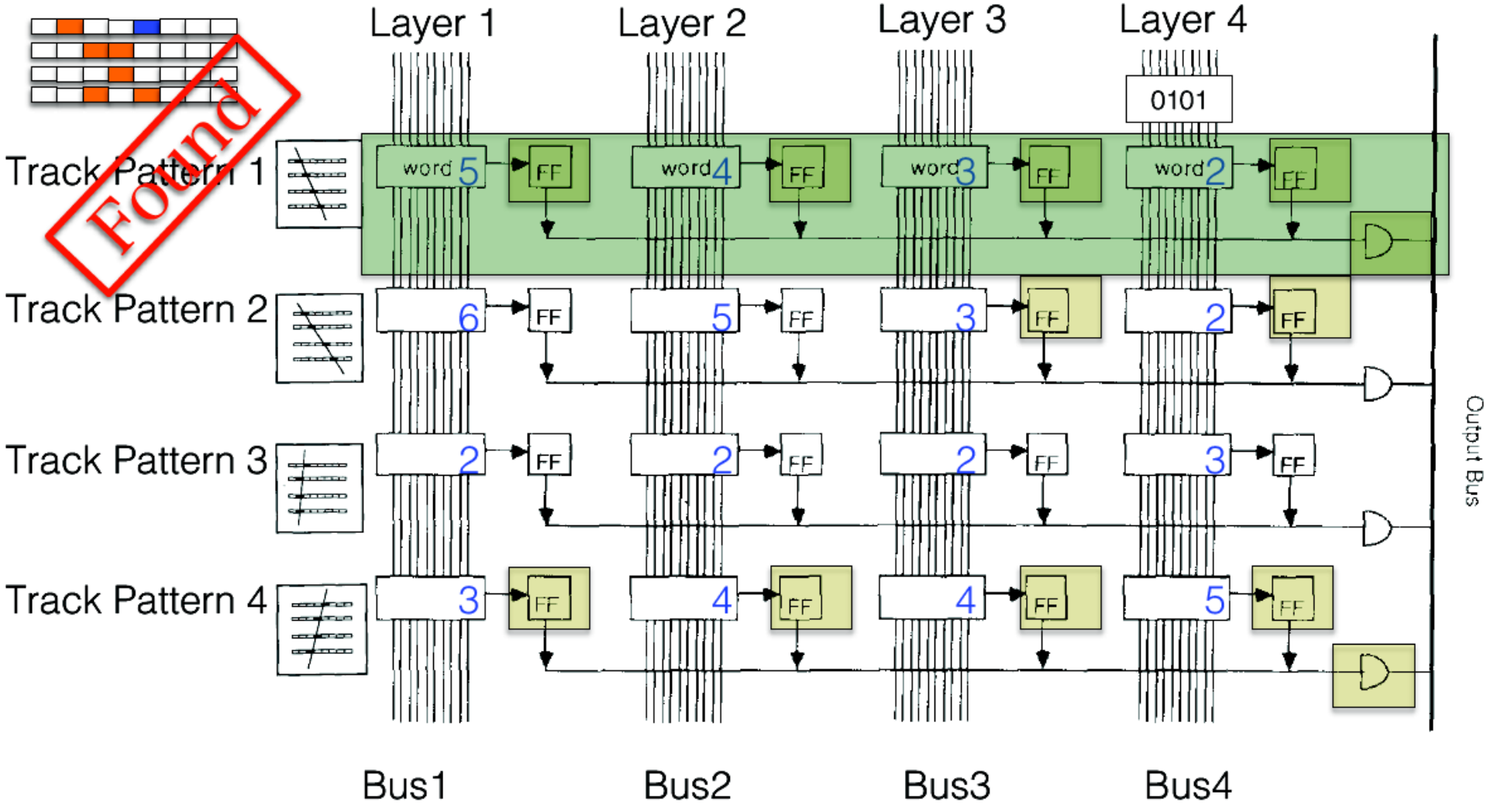
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



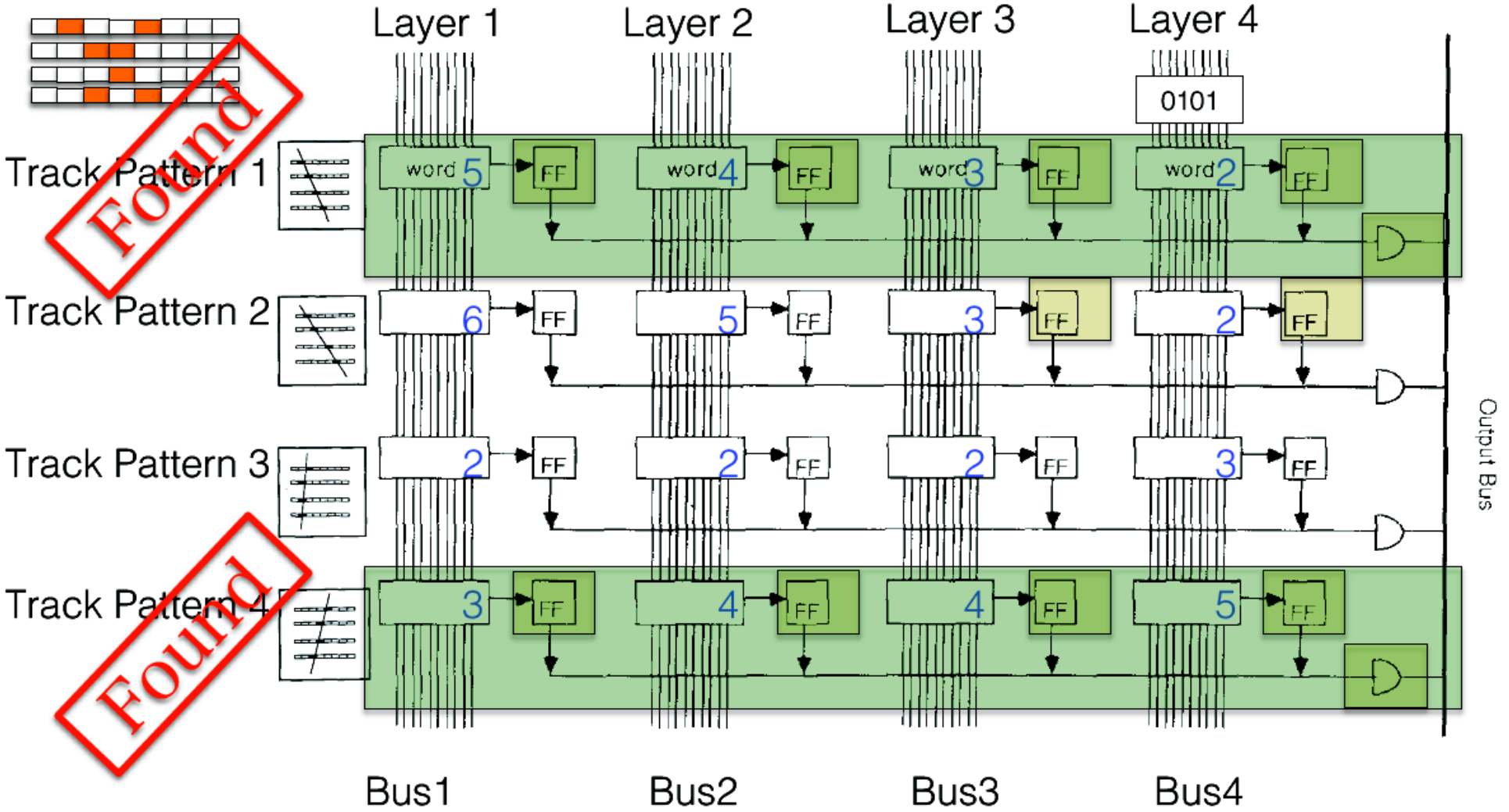
Associative memory example

- Data (hits) injected in parallel on 4 buses
  - Parallel comparison between hits and words in bank



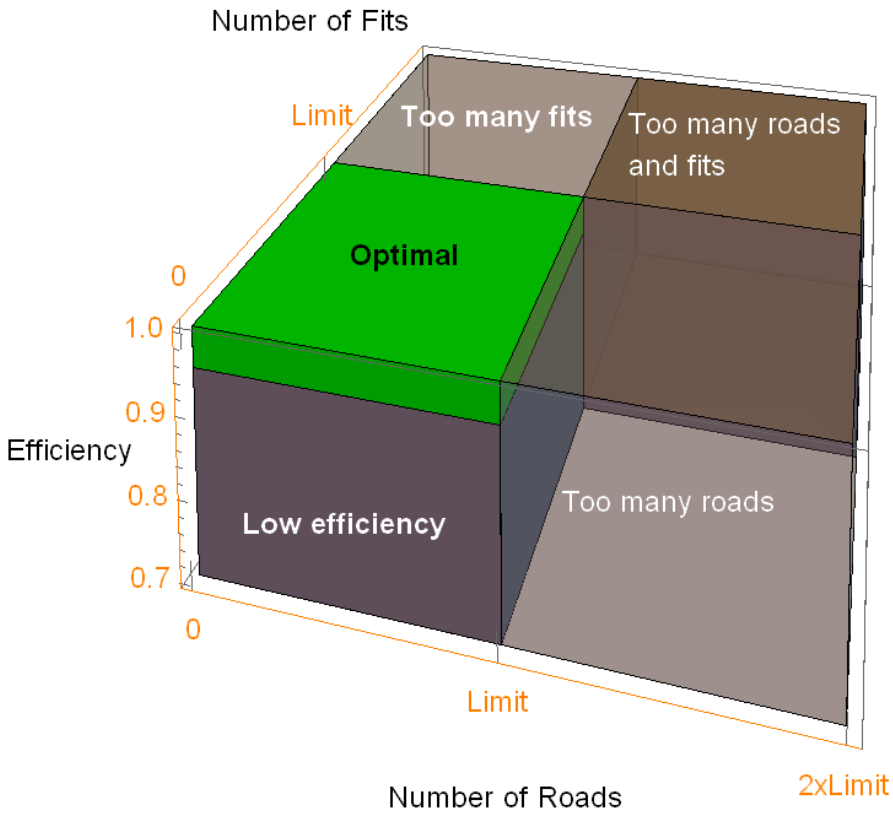
Associative memory example

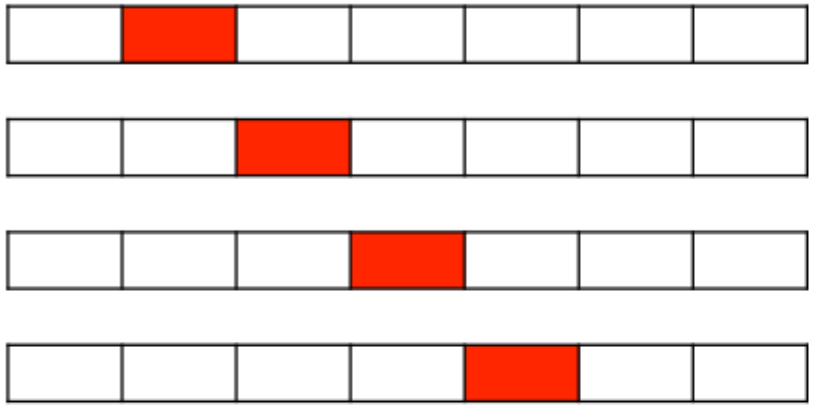
- Pattern matching is completes as soon as data arrives in the buses

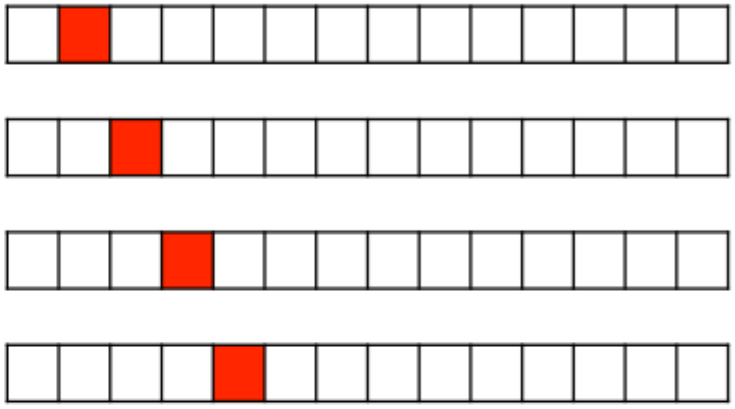


Bank size

- Total number of patterns depends on the “bin size”
  - Wide bins: less patterns, higher efficiency, more fakes and workload for the fitting stage
  - Small bins: more patterns, less efficiency, less fakes and workload downstream

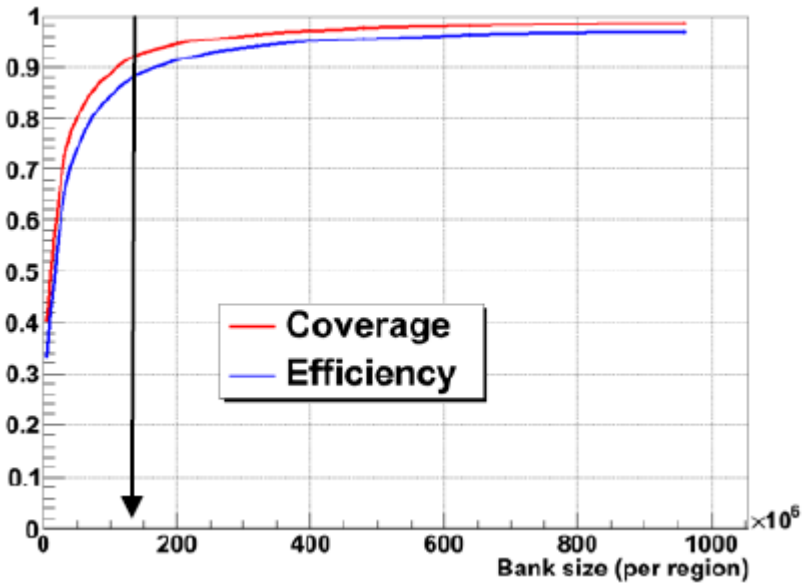




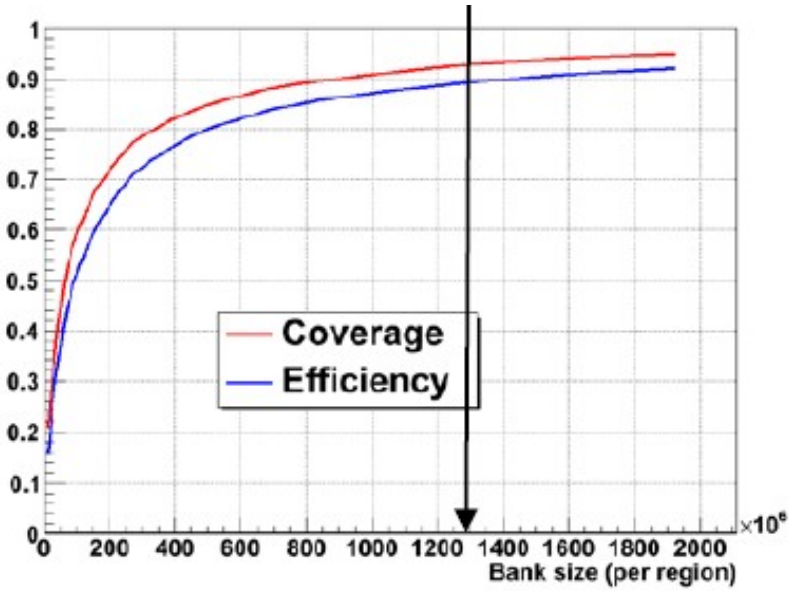


Bank size

- Standard bin size
  - r-ϕ: 16 pixels, 22 strips
- At 90% track efficiency
  - bank size: 100 M
  - matched patterns per event: 342k

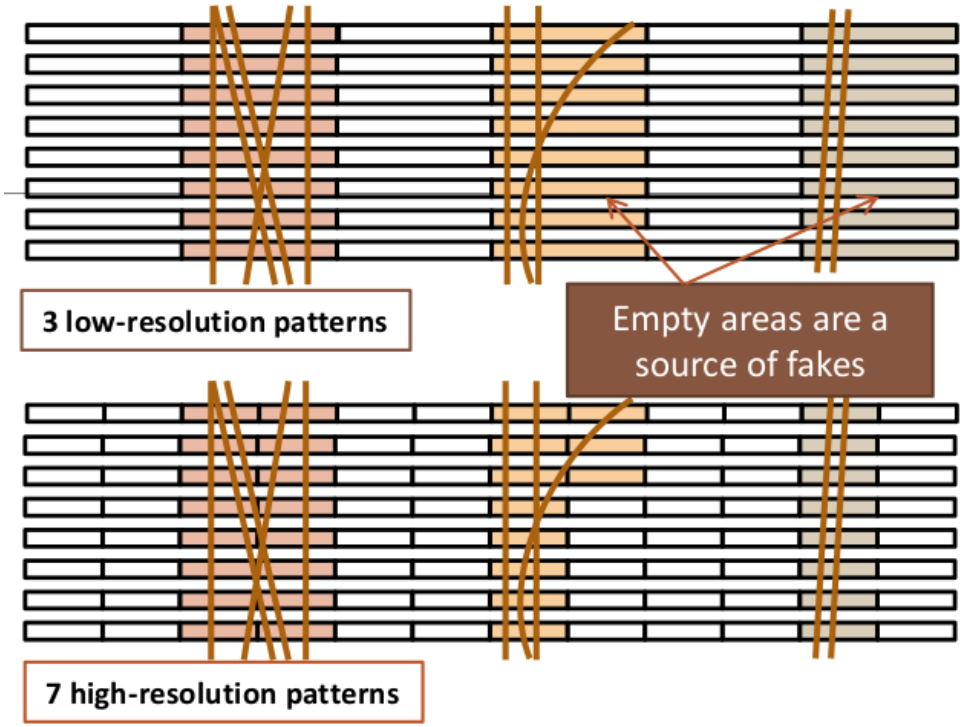


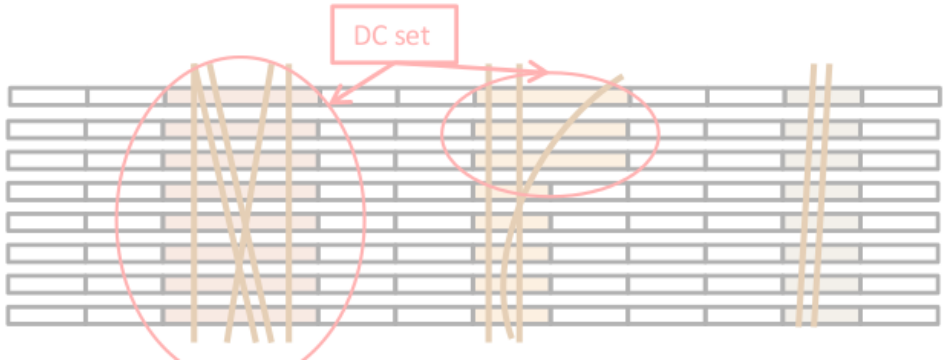
- Half bin size
  - r-ϕ: 8 pixels, 11 strips
- At 90% track efficiency
  - bank size: 1200 M
  - matched patterns per event: 40 k



Don't care bits

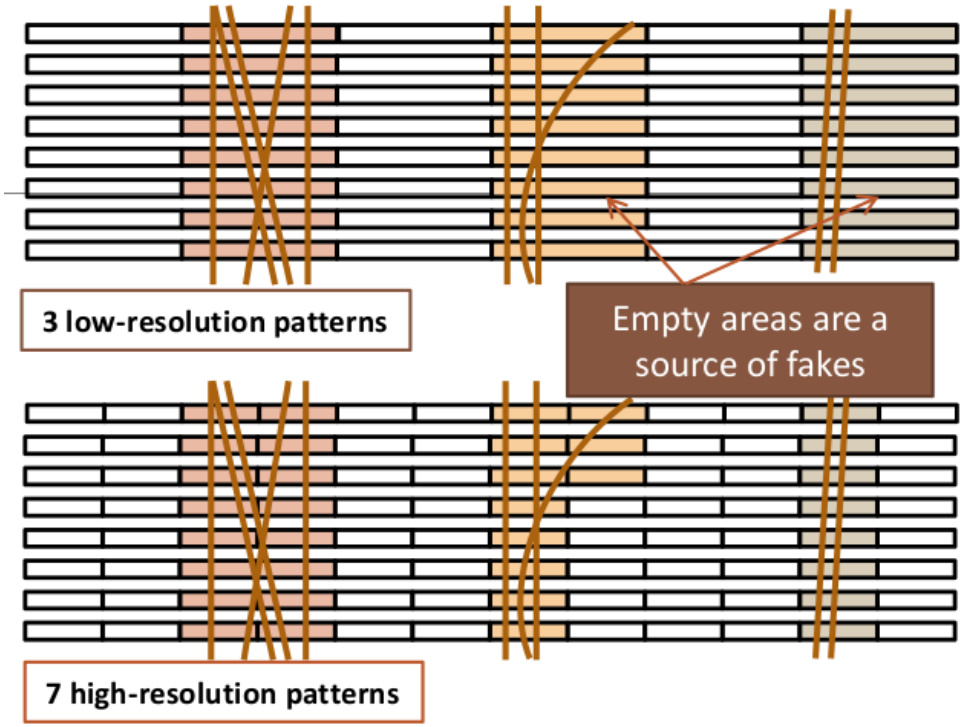
- Pattern matching resolution is crucial
  - Size vs efficiency vs fakes (workload downstream)
- Don't care bits
  - Allows to merge similar patterns in favored configurations maintaining high-resolution & rejection power where needed

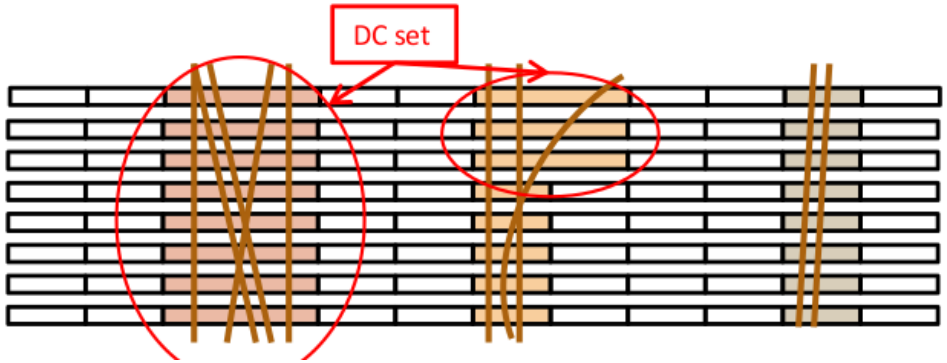




Don't care bits

- Pattern matching resolution is crucial
  - Size vs efficiency vs fakes (workload downstream)
- Don't care bits
  - Allows to merge similar patterns in favored configurations maintaining high-resolution & rejection power where needed





Ternary CAM

- Ternary CAM: add flexibility to the search
- Allows a third matching state of "X" or "**Don't Care**" for one or more bits in the stored pattern word:
  - one pattern matches various data words
- For each layer a “bin” is identified by a number stored in AM
  - The DC bits can be user to OR neighborhood bins, which differ only by few bits, without increasing the number of patterns
- E.g.:
  - the ternary CAM in this example will match all the four search words

10XX0
10000
10010
10100
10110

Ternary CAM

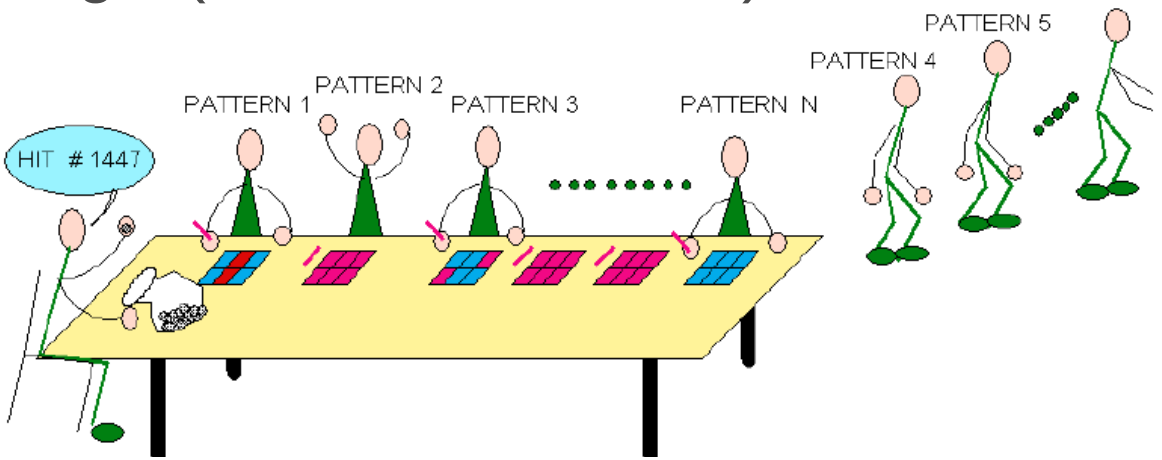
- Ternary CAM: add flexibility to the search
- Allows a third matching state of "X" or "**Don't Care**" for one or more bits in the stored pattern word:
  - one pattern matches various data words
- For each layer a “bin” is identified by a number stored in AM
  - The DC bits can be user to OR neighborhood bins, which differ only by few bits, without increasing the number of patterns

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31

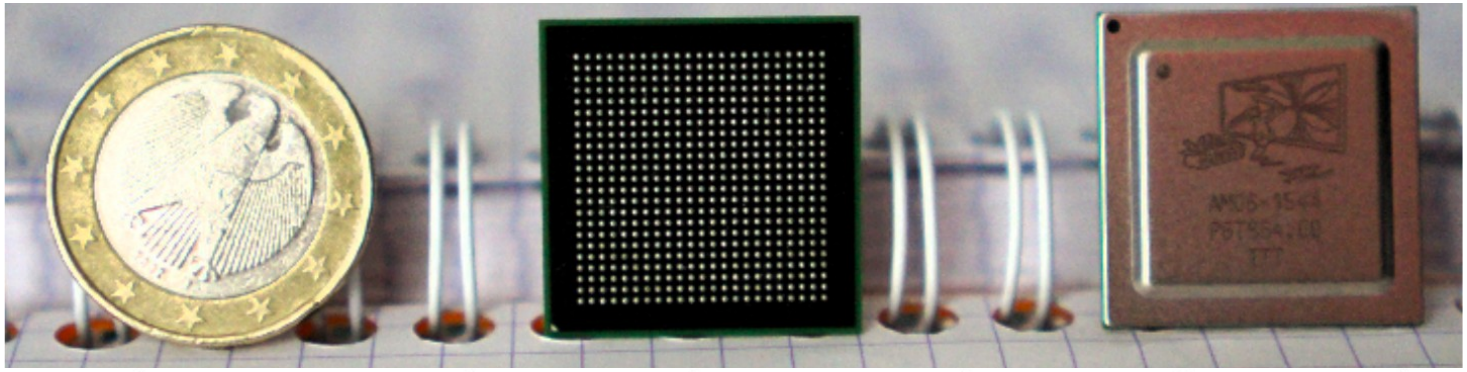
Using binary format
“01010” selects bin 10
“0001x” selects bins 2 or 3
“1x000” selects bins 16 or 24
“0x11x” selects bins 6,7,14, or 15
“111xx” selects bins 28 to 31

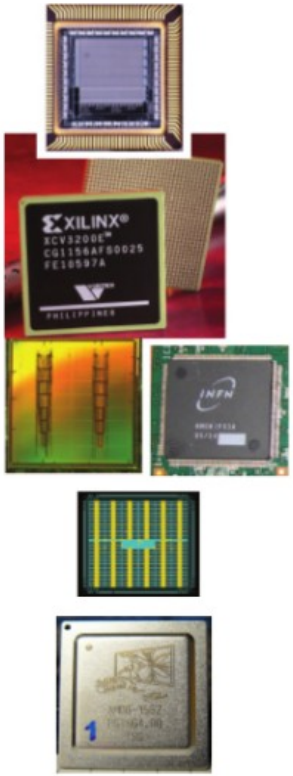
Advantages of AM

- hits compared w/ all stored patterns **simultaneously**
  - Massive “parallelism” of pattern recognition
- Processing time is linear in the number of hits
  - As soon as all the detected hits are loaded, the pattern recognition will be completed
  - For ATLAS run 2-3: overall average latency ~100 μs
- Availability for optimization
  - Majority logic (such as 7 out of 8) for hit inefficiency



AM evolution

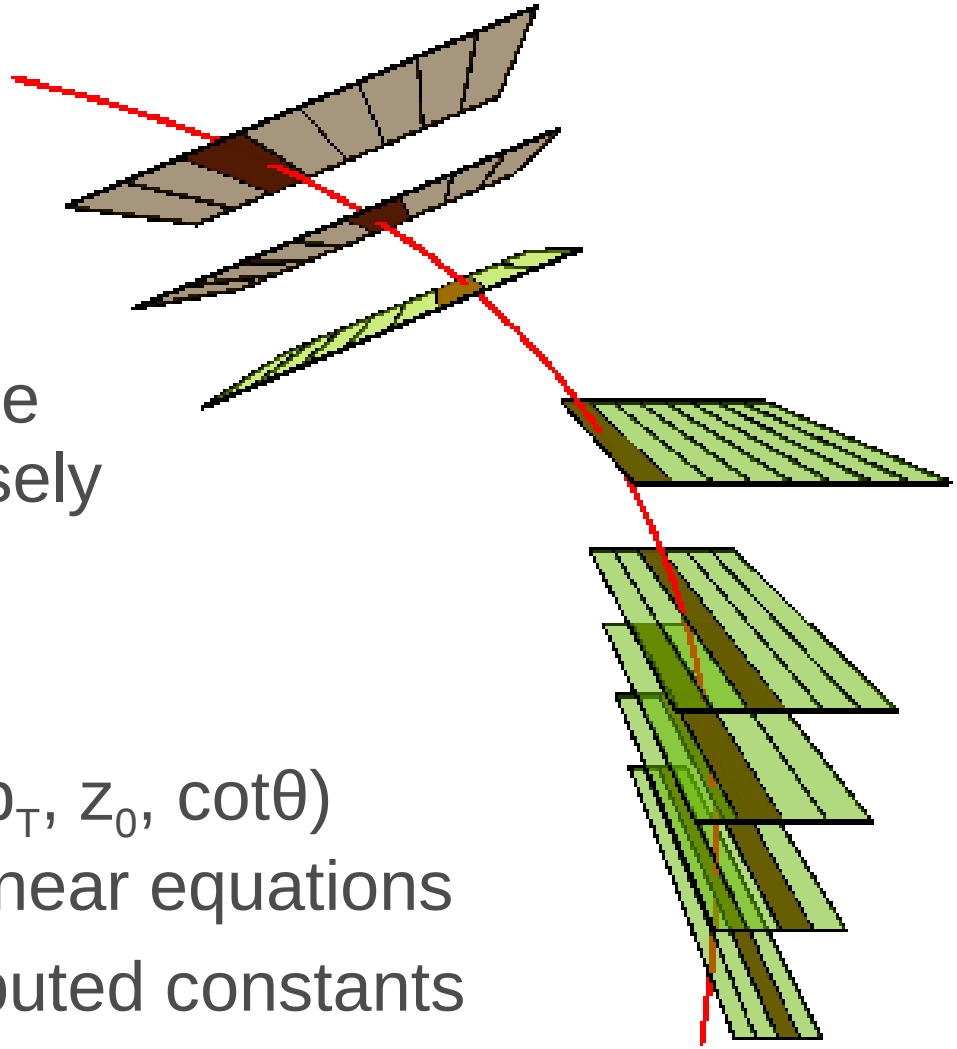




Vers.	Design	Tech.	Area	Patterns	Package	
1	Full custom	700 nm		128	QFP	SVT @CFD
2	FPGA	350 nm		128	QFP	
3	Std cells	180 nm	100 mm^2	5 k	QFP	SVT upgrade
4	Std cells + Full custom	65 nm	14 mm^2	8 k	QFP	
mini-5	Std cells + Full custom	65 nm	4 mm^2	0,5 k	QFP	
5	+ IP blocks		12 mm^2	3 k	BGA	
6	**Std cells + Full custom + IP blocks**	**65 nm**	**168 mm^2**	**128 k**	**BGA**	FTK@Atlas
7	Std cells + Full custom	28 nm	10 mm^2	16 k	BGA, SiP	RD run 4

Fitting stage

- Pattern matching defines **roads** to be refined
  - fetch all the (few now) hits in a road and fit them to a helical track to measure the track parameters precisely



- Fit can be done via a **linear approssimation**
  - track parameters (d_0, φ, $1/p_T$, z_0, $\cot\theta$) related to hit positions by linear equations
  - Multiplications w/ pre-computed constants
  - Track fitting in FPGAs w/ many Digital Signal Processors (DSPs): ~1 Gfits/s per FPGA

Track fitting

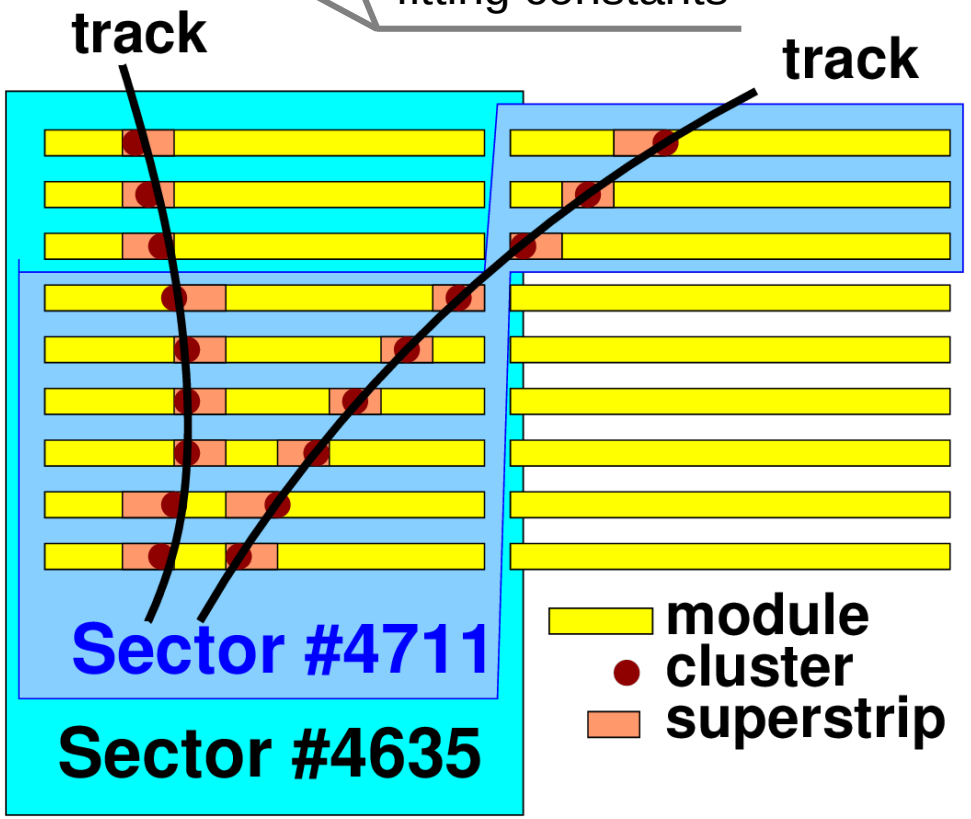
Track parameters
d_0 ,φ,1/p_T, z_0, cotθ, χ^2

Hit coordinates
at full resolution

$$\tilde{p_i} = \sum_{l=1}^{N} C_{il} x_l + q_i$$

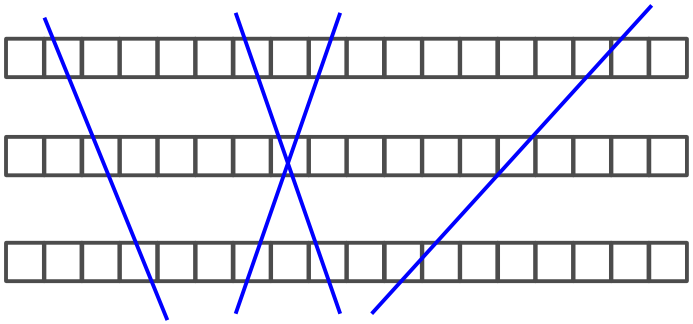
Precalculated
fitting constants

- Model valid for small geometrical regions (**sectors**)
  - Each sector has its own fit constants: coefficients of the linear equations
  - There are more than 10^5 sectors used in the FTK system

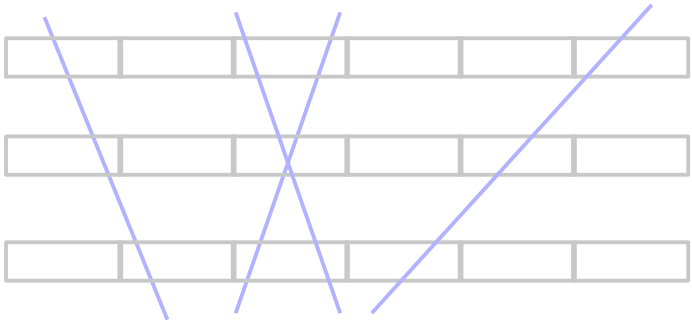


Pattern bank creation

- Generate muon tracks via detector simulation



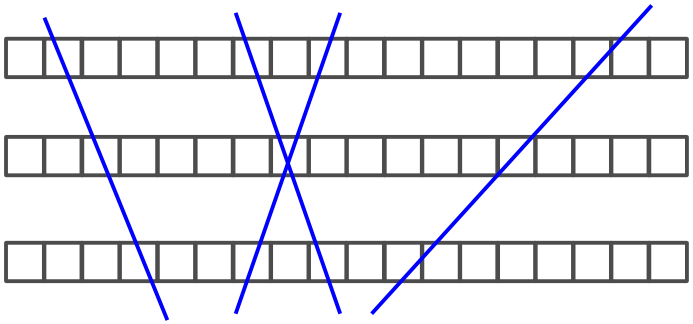
- Create bins merging continguous channels
  - To reduce back size and increase efficiency



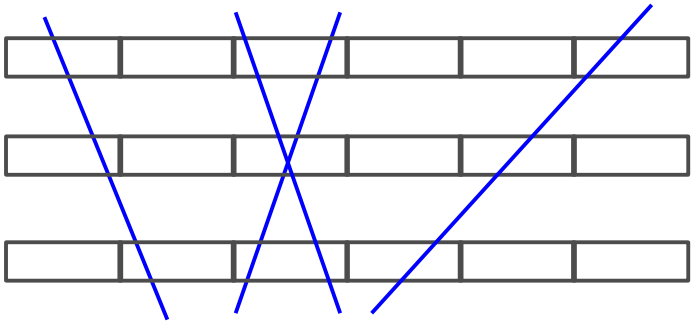
- identify relevant patterns for the bank
  - Composed of bins

Pattern bank creation

- Generate muon tracks via detector simulation



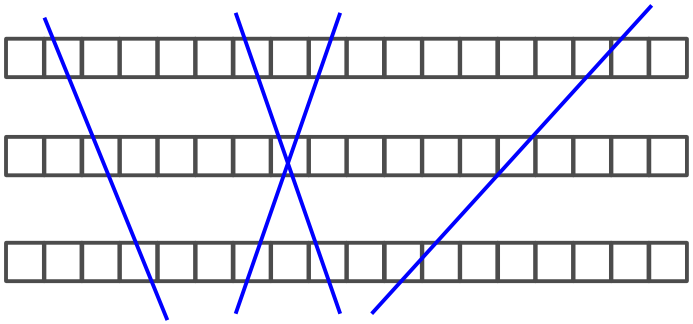
- Create bins merging continguous channels
  - To reduce back size and increase efficiency



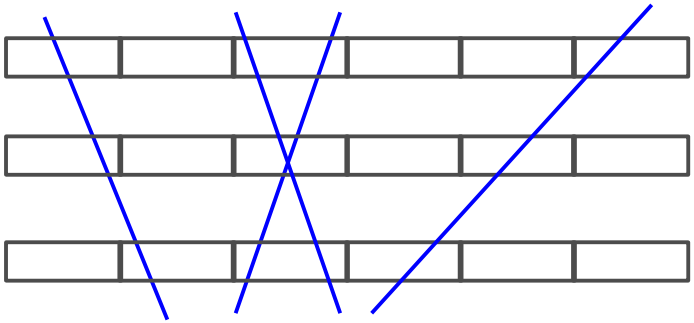
- identify relevant patterns for the bank
  - Composed of bins

Pattern bank creation

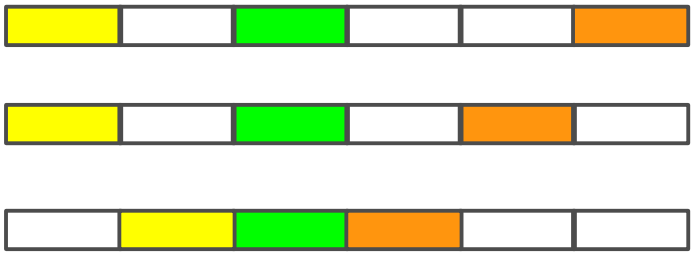
- Generate muon tracks via detector simulation



- Create bins merging continguous channels
  - To reduce back size and increase efficiency

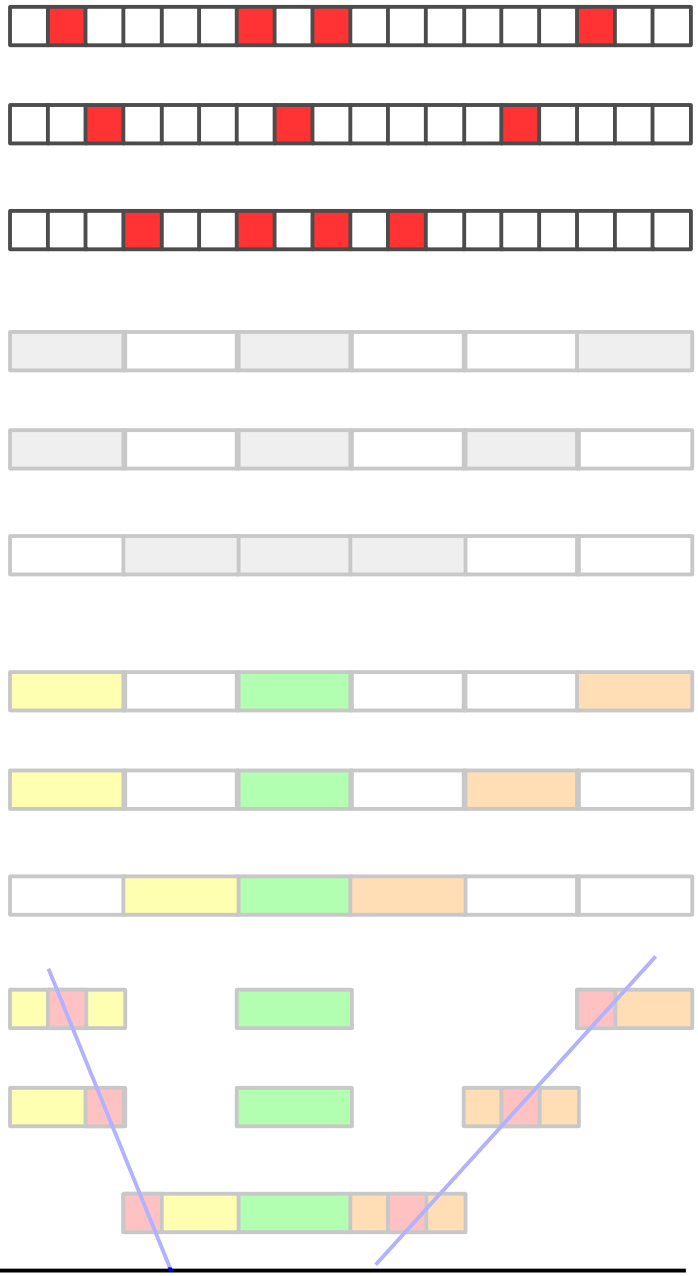


- identify relevant patterns for the bank
  - Composed of bins



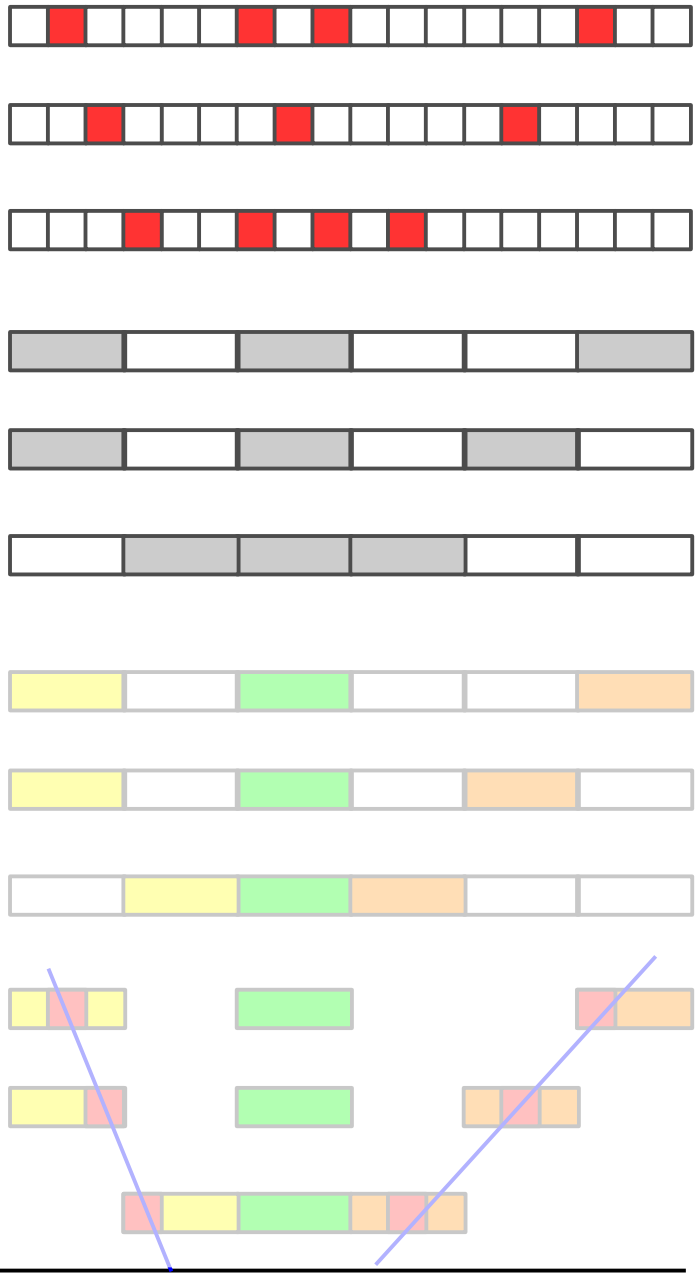
At run time

- Receive hits from detector
  - At full granularity
- Merge contiguous channels
  - Same bin size as AM
  - And properly rearrange data
- Pattern matching
  - Find roads
- Linear fits inside the roads
  - With FPGA
  - Data at full granularity



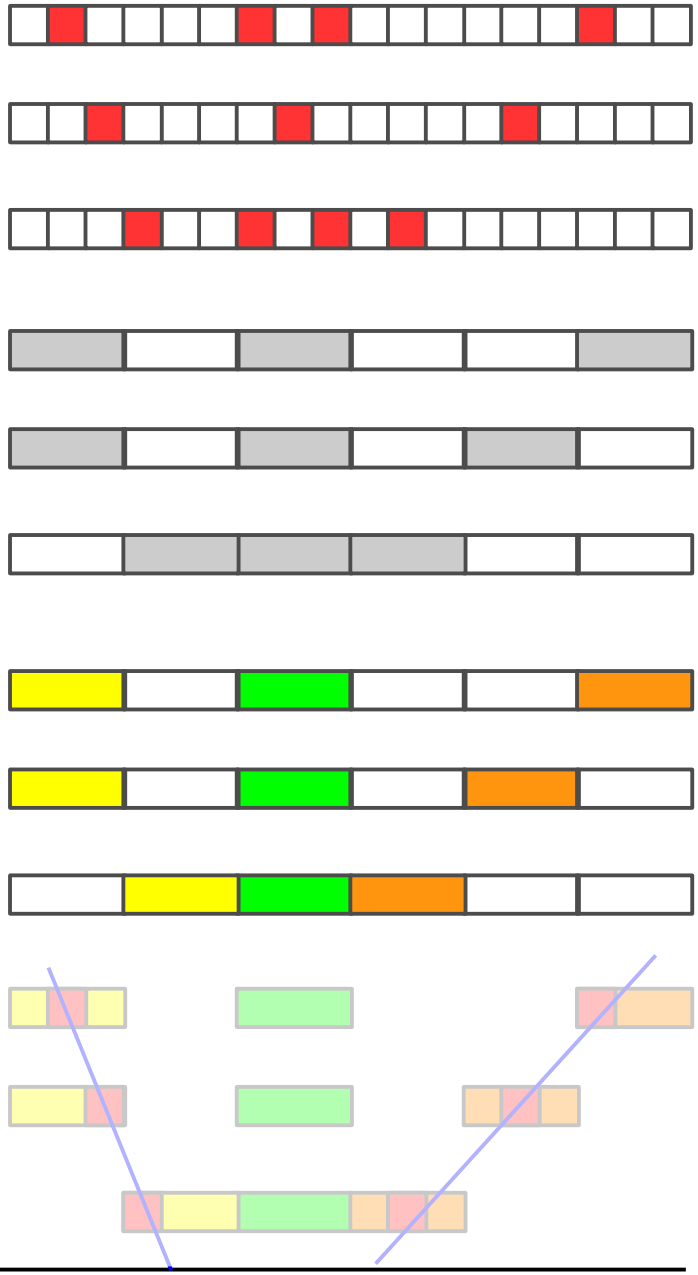
At run time

- Receive hits from detector
  - At full granularity
- Merge contiguous channels
  - Same bin size as AM
  - And properly rearrange data
- Pattern matching
  - Find roads
- Linear fits inside the roads
  - With FPGA
  - Data at full granularity



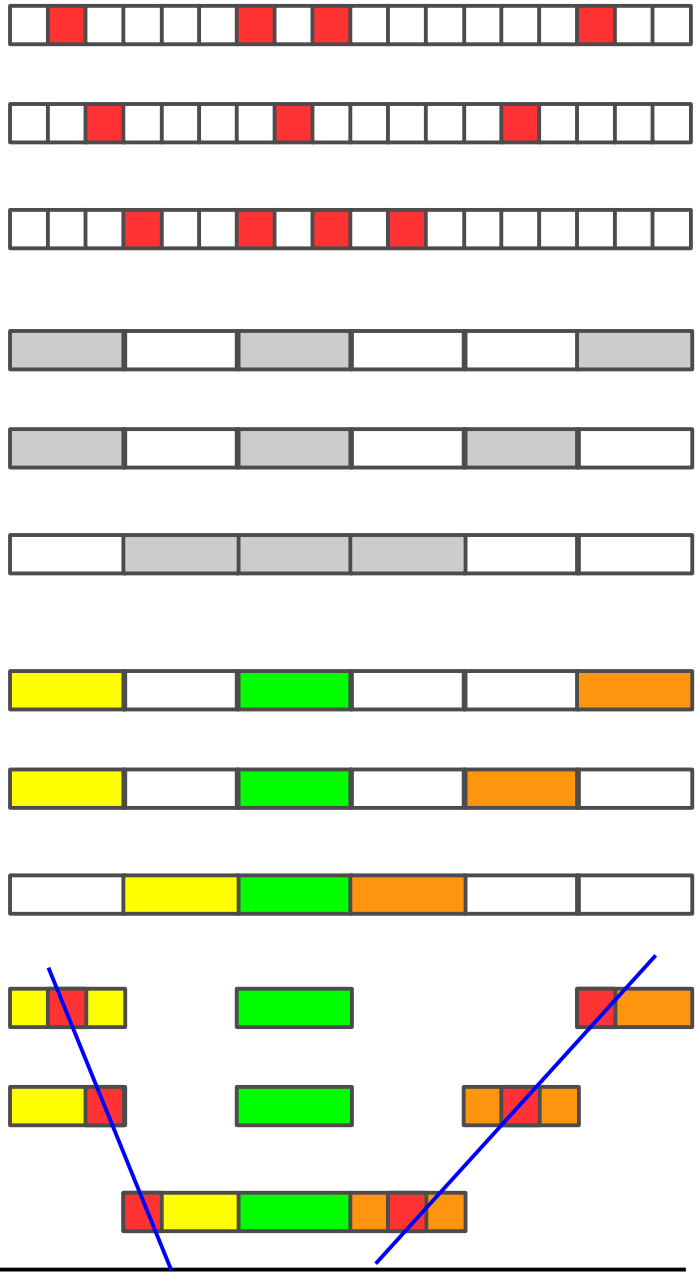
At run time

- Receive hits from detector
  - At full granularity
- Merge contiguous channels
  - Same bin size as AM
  - And properly rearrange data
- Pattern matching
  - Find roads
- Linear fits inside the roads
  - With FPGA
  - Data at full granularity



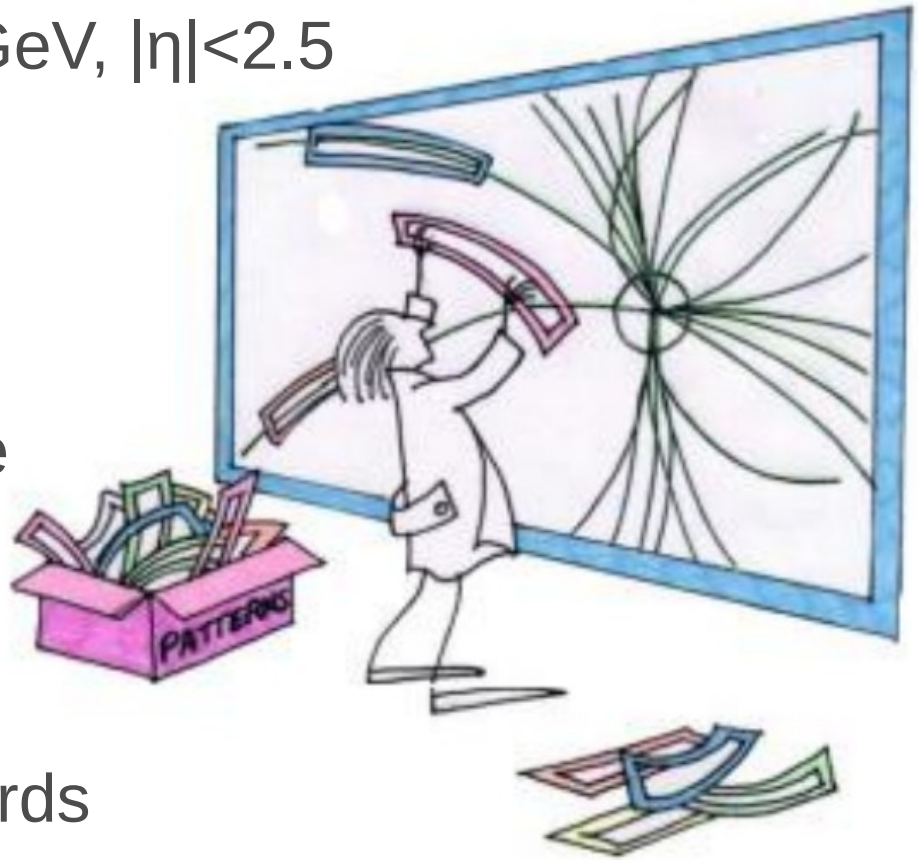
At run time

- Receive hits from detector
  - At full granularity
- Merge contiguous channels
  - Same bin size as AM
  - And properly rearrange data
- Pattern matching
  - Find roads
- Linear fits inside the roads
  - With FPGA
  - Data at full granularity



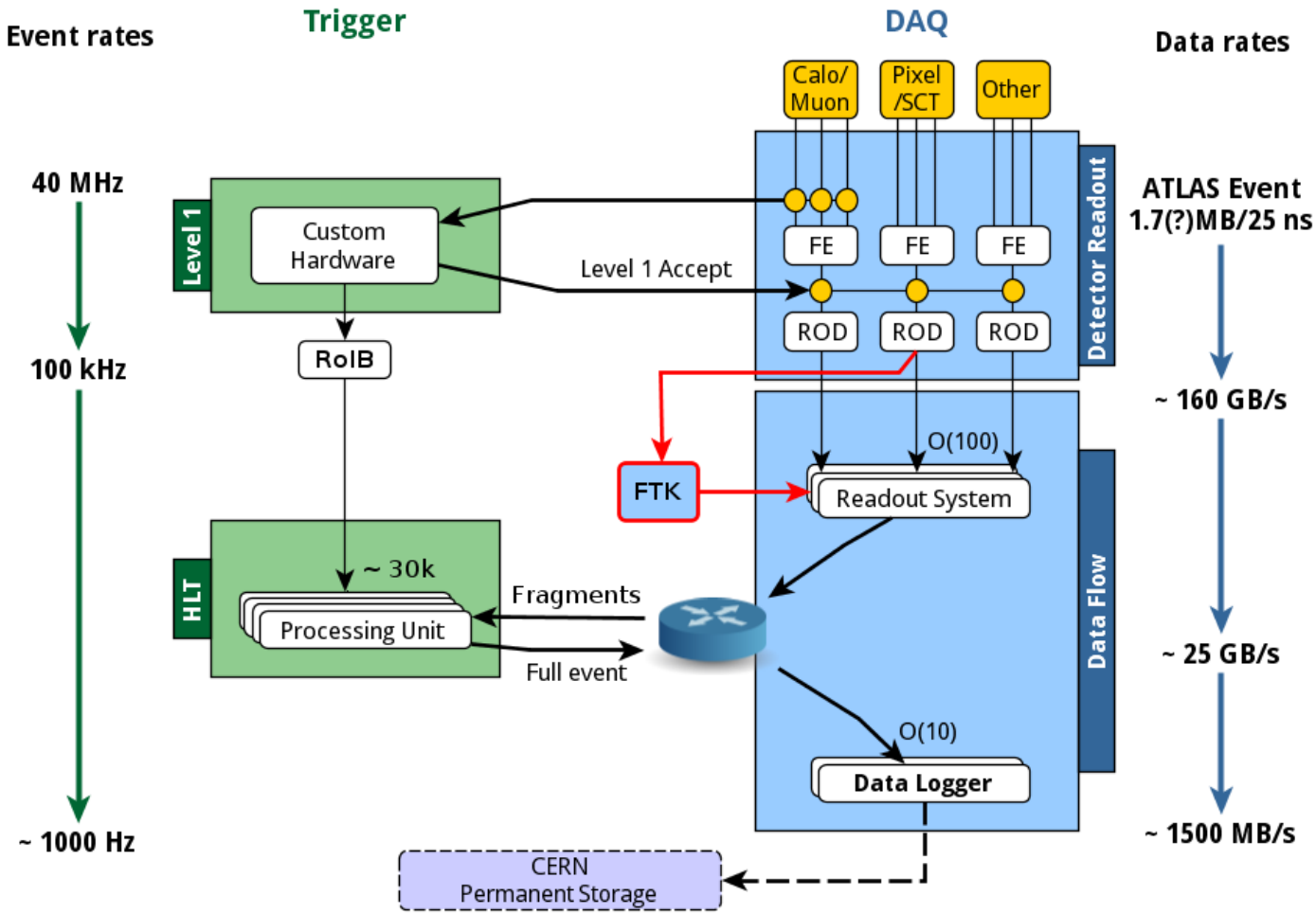
Fast TracKer

- ATLAS upgrade under commissioning
  - coprocessor dedicated to track reconstruction at high rate (100 kHz) and short latency (~100 μs)
  - Efficiency ~ 93% for p_T > 1 GeV, |η|<2.5
- Computational load subdivided in towers able to work in parallel
  - designed around Associative Memories (AM) and FPGAs
  - 8192 AM chips, storing 1 billion patterns
  - Custom VME and ATCA boards
  - Linearized track fitting performed in 2 consecutive stages



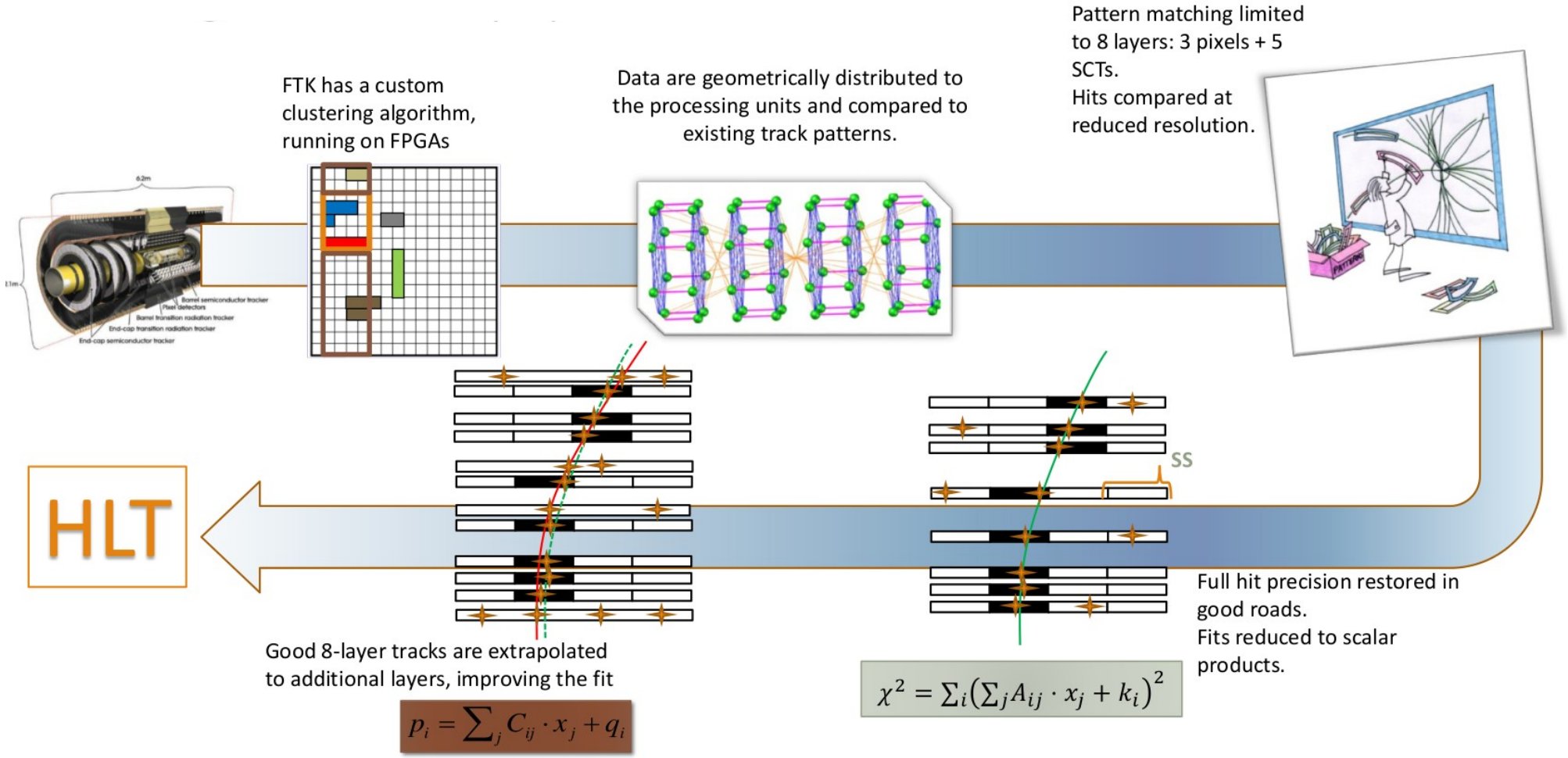
Fast TracKer

- Operating after L1 and before HLT
  - Provides HLT, at 100 kHz, with all tracks with $p_T > 1$ GeV

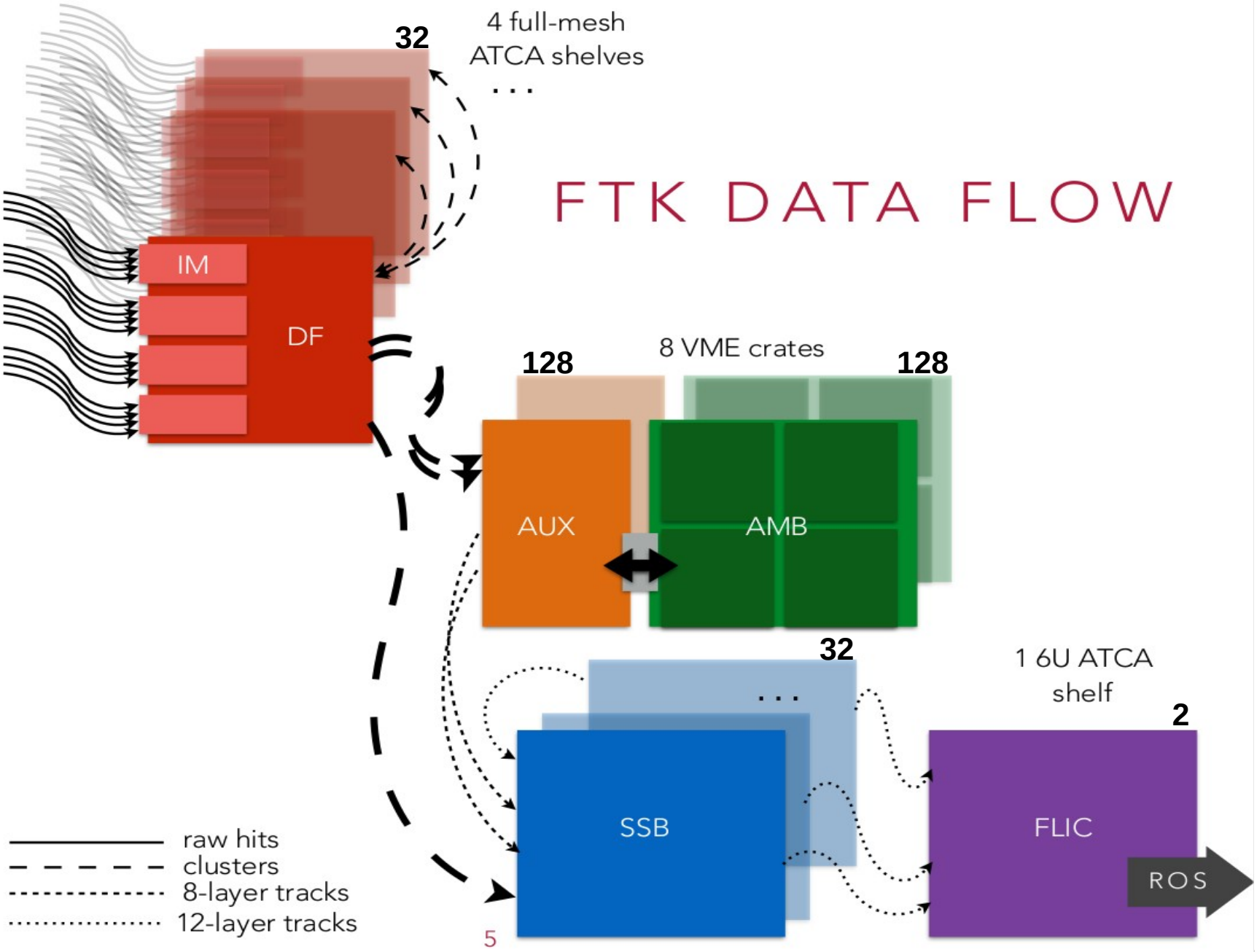


Fast TracKer: pipeline

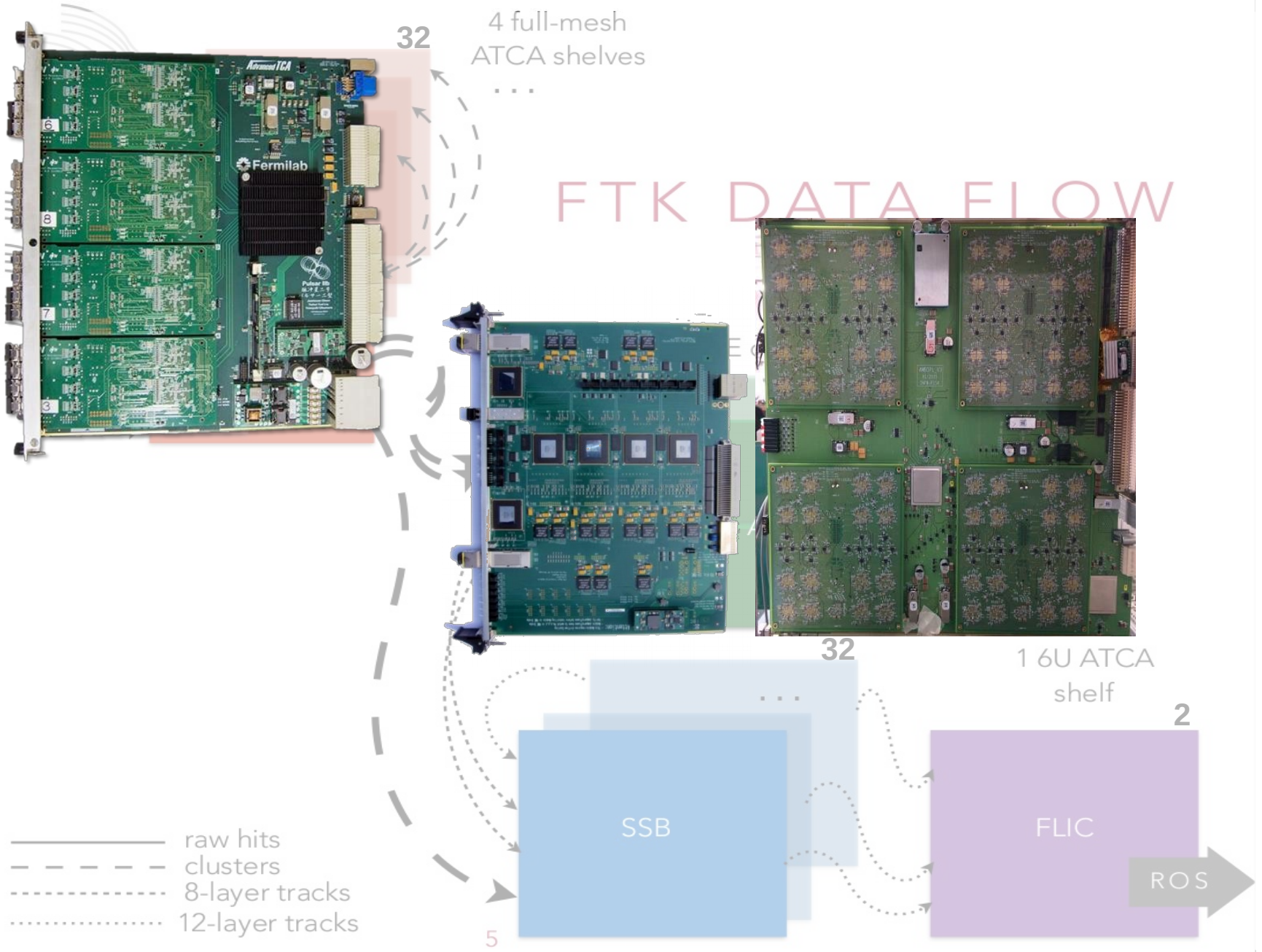
- FTK Pipeline:
  - data formatting → pattern matching → track fitting 8 layers → track fitting 12 layers



FTK: boards


4 full-mesh
ATCA shelves
32
FTK DATA FLOW
IM
DF
128
8 VME crates
128
AUX
AMB
32
1 6U ATCA
shelf
2
SSB
FLIC
ROS
raw hits
clusters
8-layer tracks
12-layer tracks
5

FTK: boards

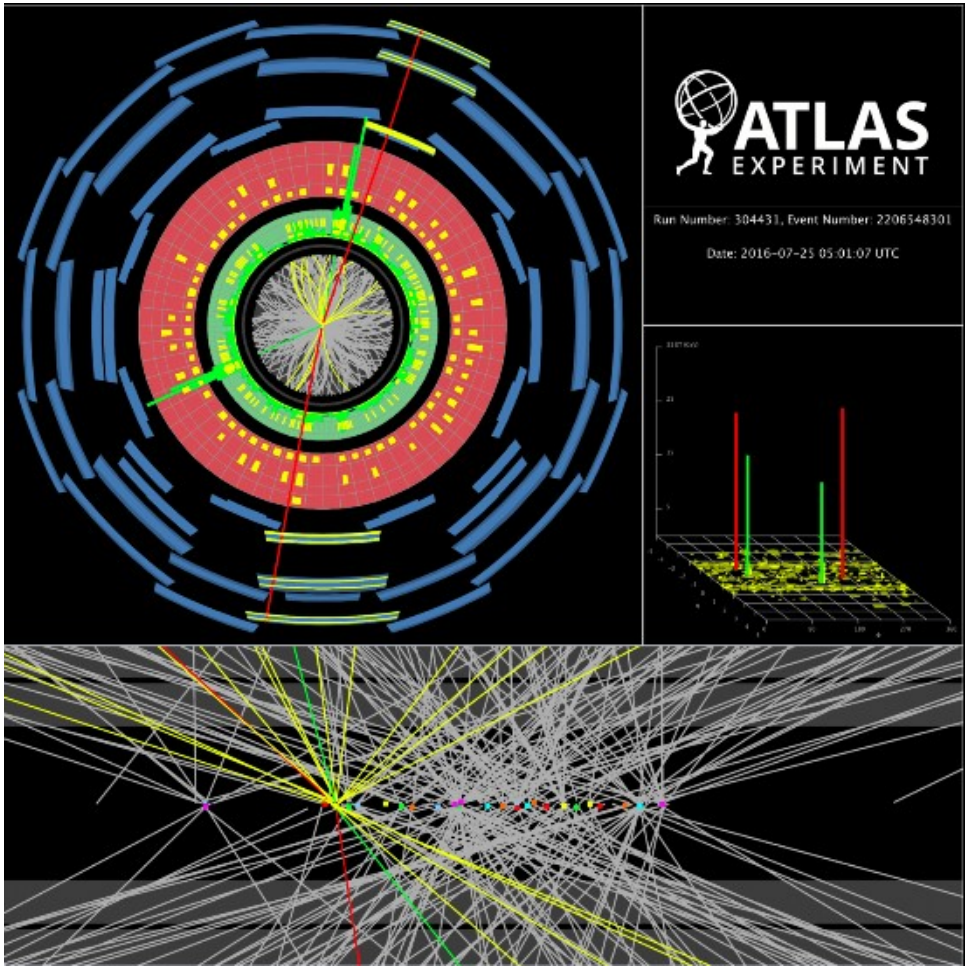

32
4 full-mesh
ATCA shelves
FTK DATA FLOW
32
1 6U ATCA
shelf
2
SSB
FLIC
ROS
5
raw hits
clusters
8-layer tracks
12-layer tracks

Commissioning

- Fast TracKer
  - Really heterogeneous system
  - Many different expertises & skills: HW, FW, SW, network, etc
- Commissioning more demanding than R&D
  - Your system must be fault tolerant
    - You cannot power cycle the crate as in lab
  - Tons of problems not seen in lab
  - As soon as you fix a problem another one shows up at a deeper level
- Joining a working system is even more challenging
  - Pressure from the other subdetectors
  - Physics analysis studies ongoing in parallel:
    - difficult to be involved at 100%

HEP trigger vs Neuroscience

- M. M. Del Viva, G. Punzi, and D. Benedetti. *Information and perception of meaningful patterns*. PloS one, 8(7):e69154, 2013

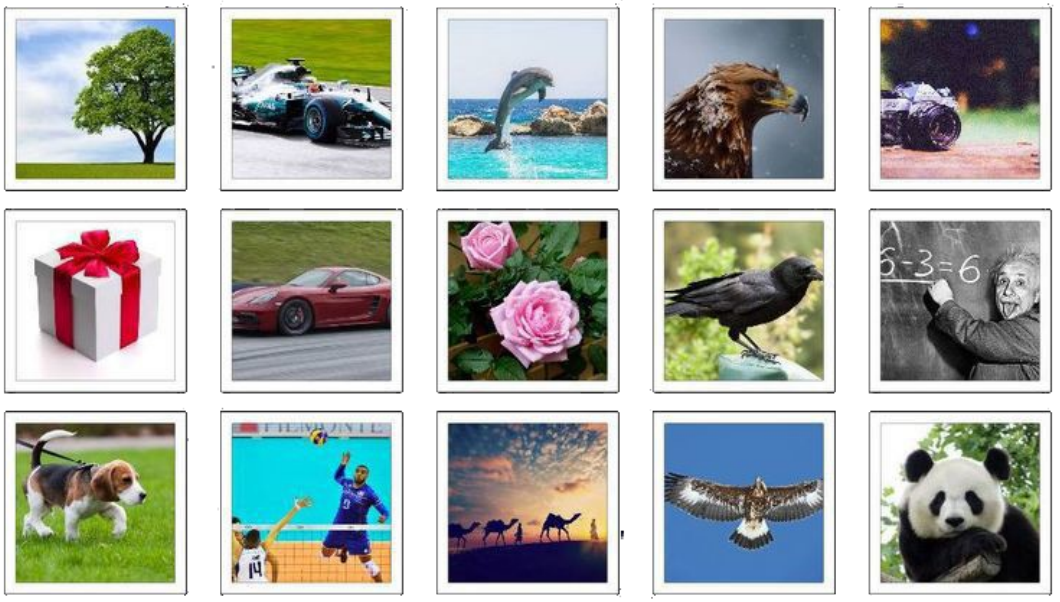


A model for vision

- Goal
  - Extract relevant information from a bulk of data
    → identify elements inside a picture
- Assumptions
  - High data rate
  - Limited output bandwidth
  - Input made by patterns
  - The system can identify a limited number of patterns
- Main principle
  - Maximize the output entropy

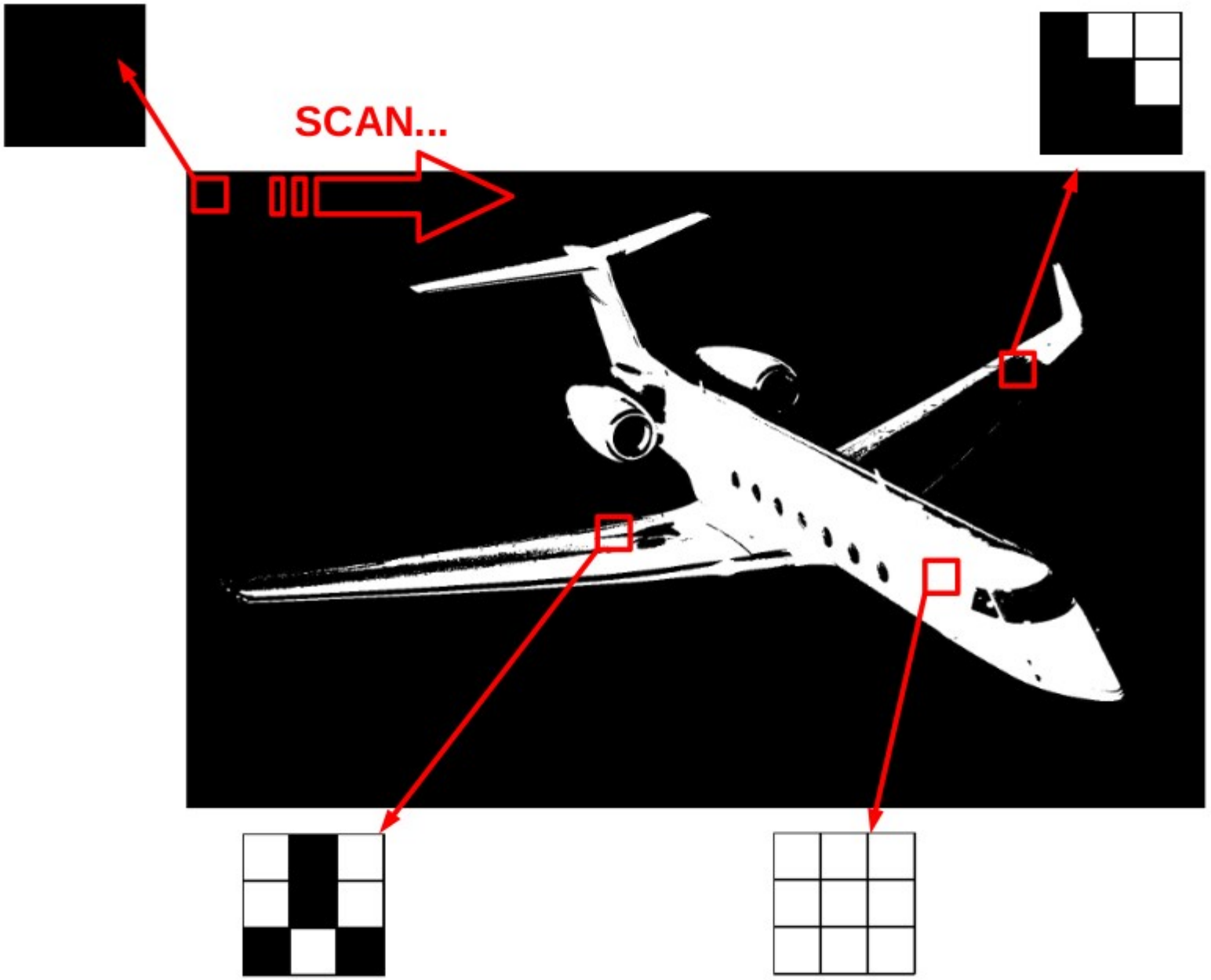
Algorithm

- Input:
  - pictures
- Assumption:
  - patterns of 3x3 pixels
- Initial reduction of information
  - From color to grayscale or B/W



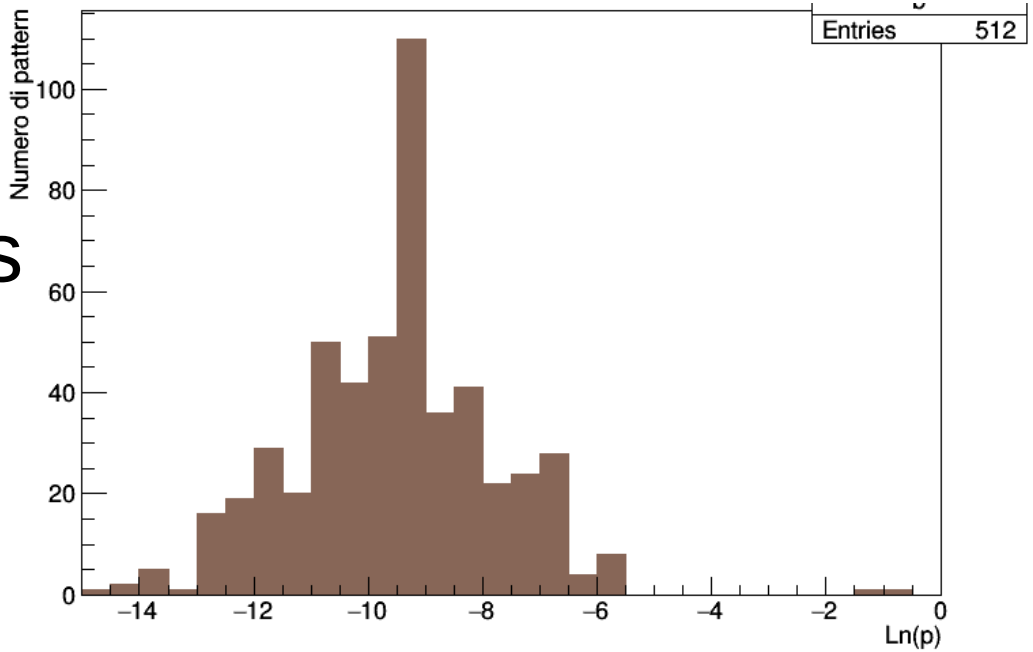
- Two phases
  - **Training**: identify relevant patterns in some training pictures
  - **Pattern matching**: reconstruct a picture using the relevant patterns only and use it as input for selection

Sliding windows scan



Training: analisys

- Training: fill histogram of log of probability of patterns in n pictures
  - Some patterns more popular than others

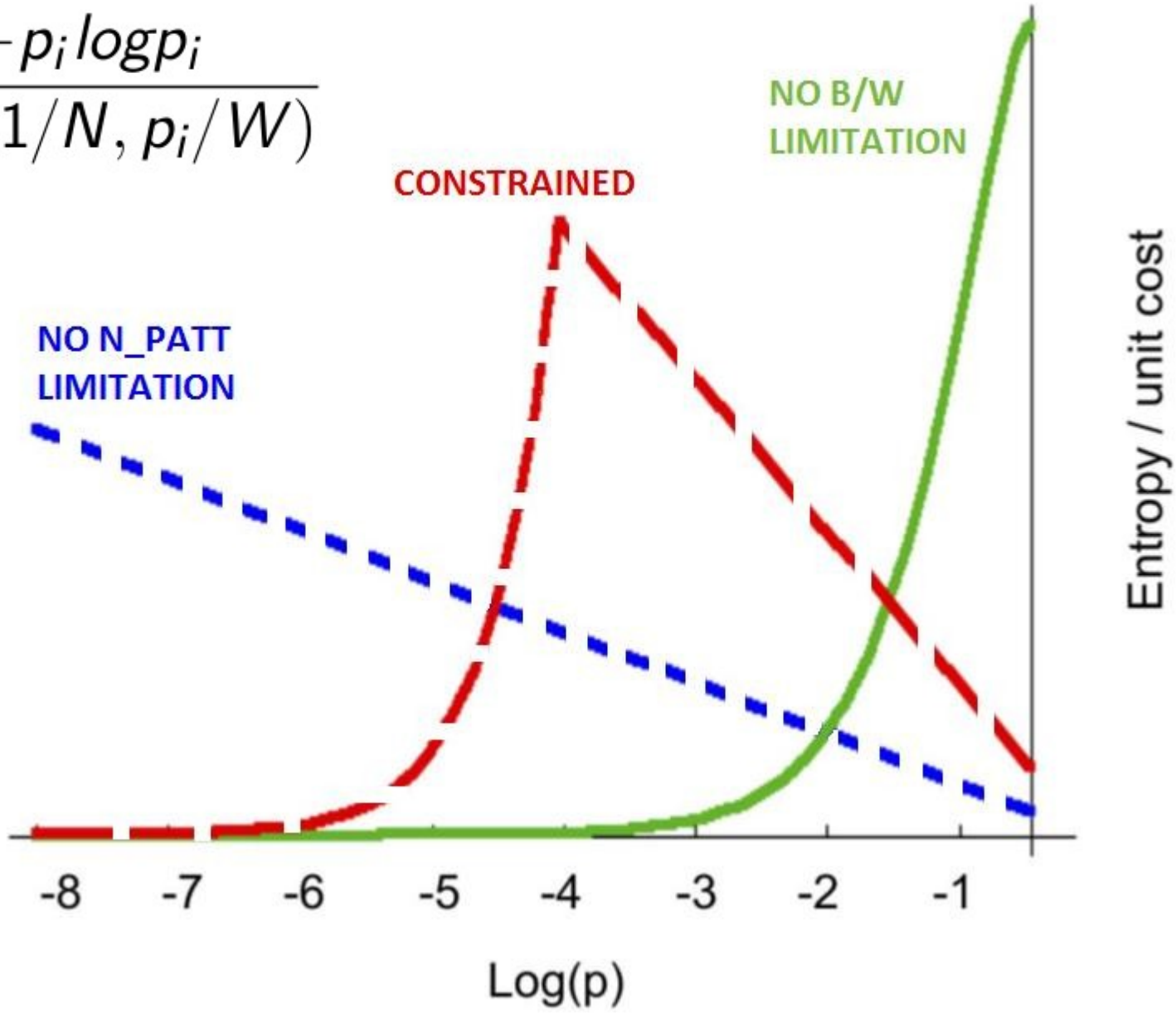


- To select the most relevant patterns authors propose a function based on entropy for unit of cost
  - **p_i**: propability of the given pattern
  - **N**: number of patterns the system can identify
  - **W**: available output bandwidth

$$f(p_i) = \frac{-p_i log p_i}{max(1/N, p_i/W)}$$

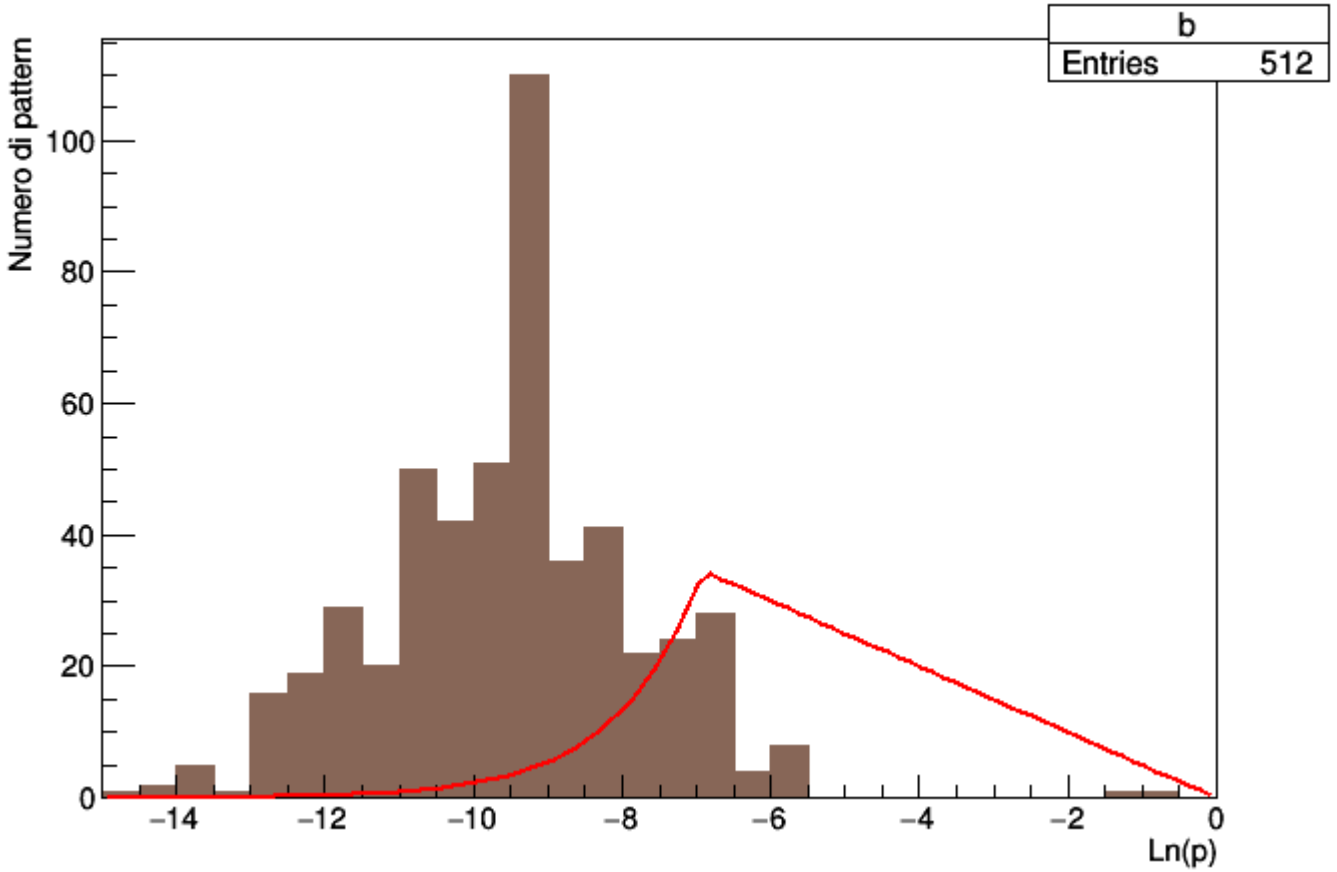
Entropy per unit cost

$$f(p_i) = \frac{-p_i log p_i}{max(1/N, p_i/W)}$$



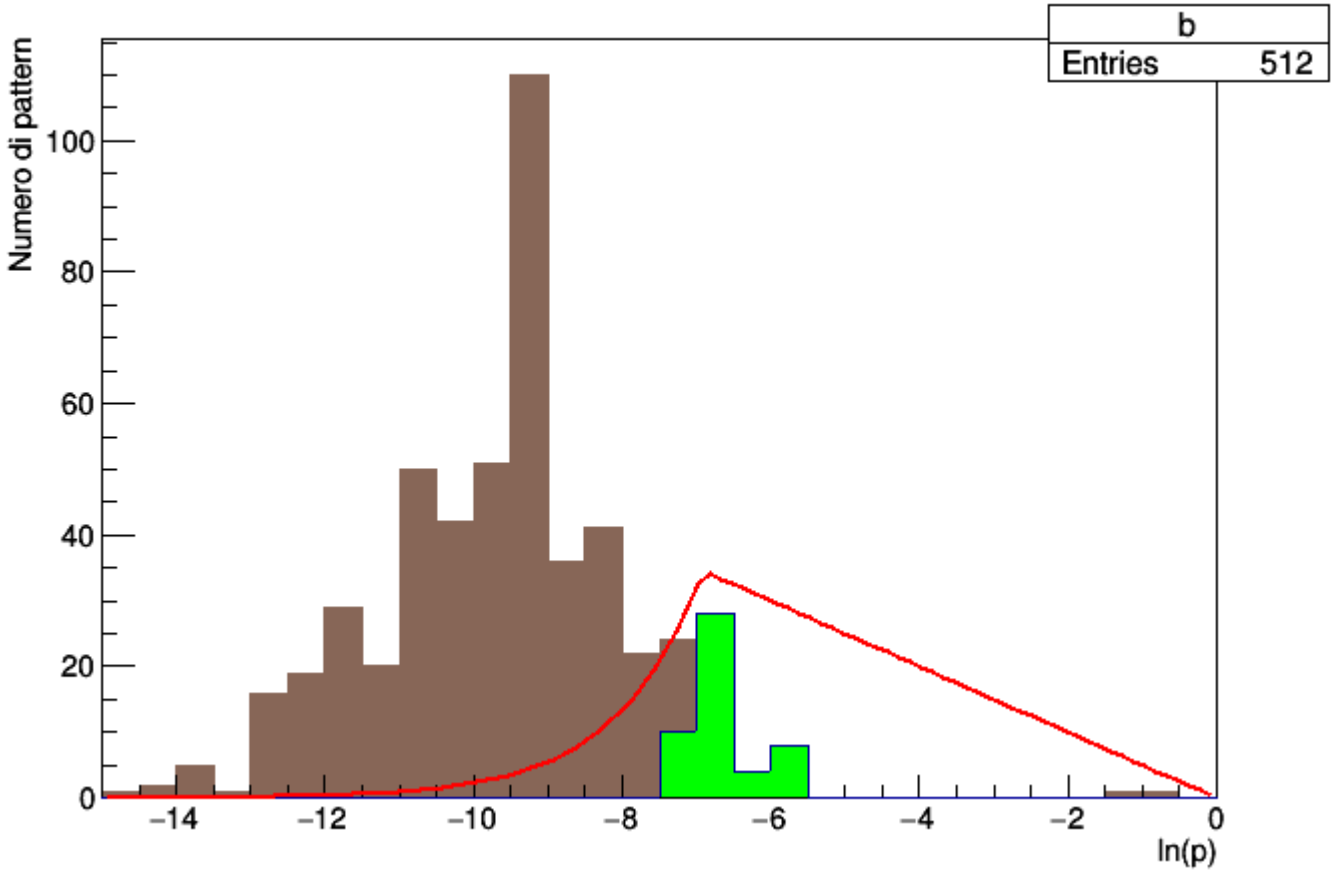
Training: pattern selection

- Histogram from training and selection function
  - N = 50, W = 0.05



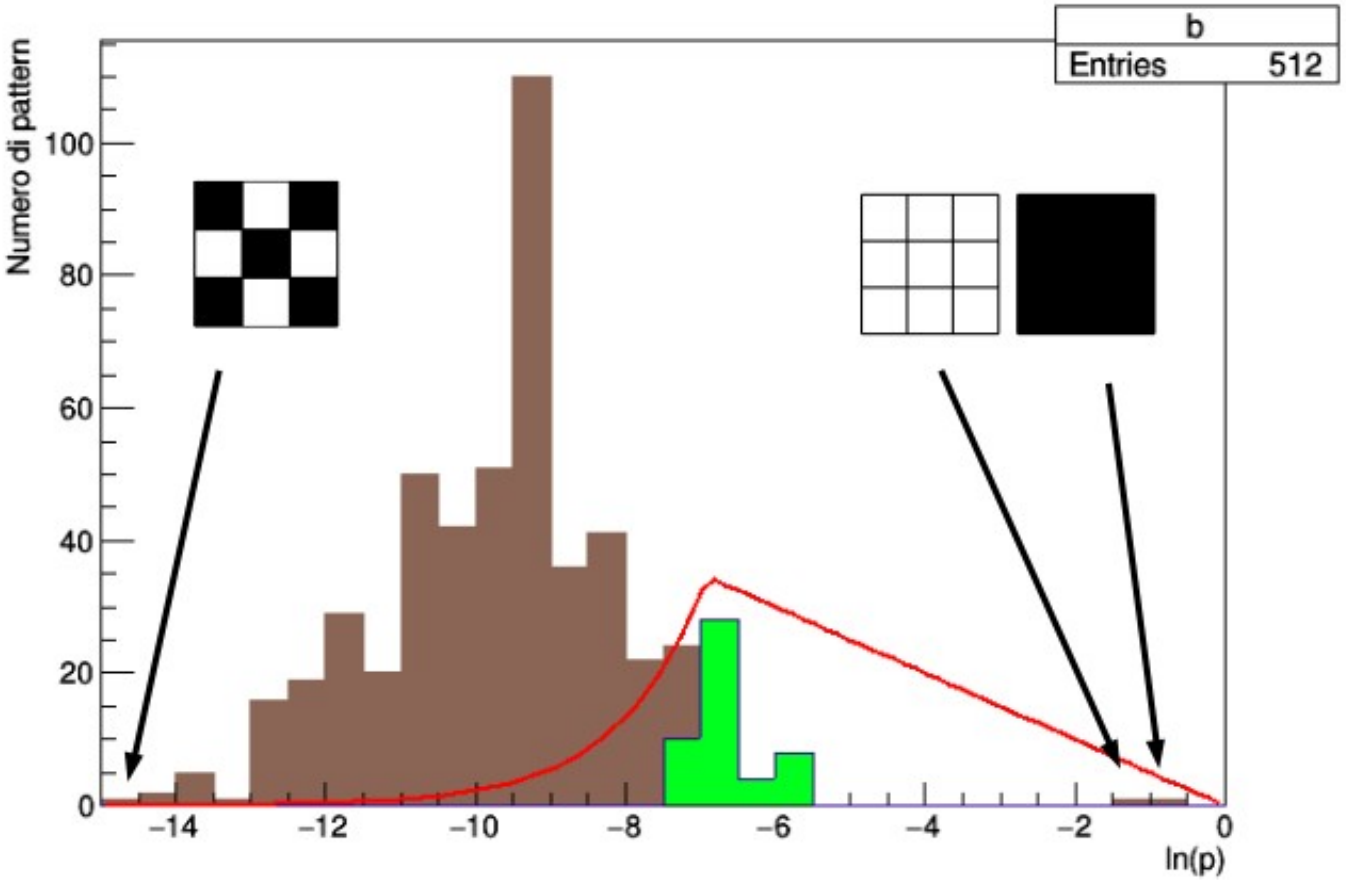
Training: pattern selection

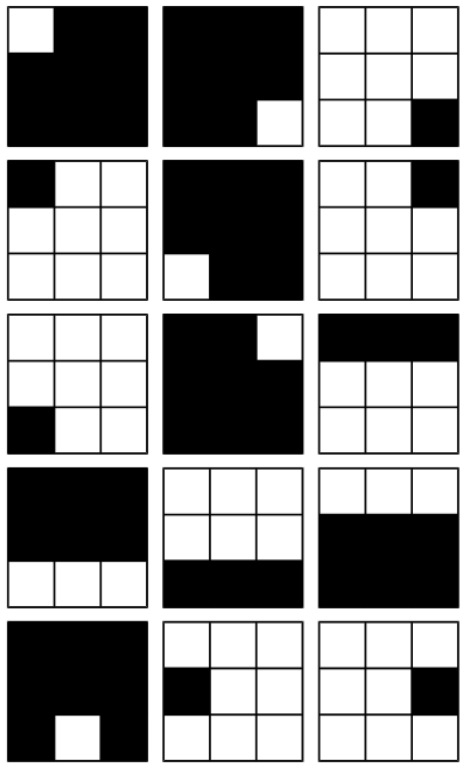
- Keep only the “best” N patterns
  - according to the ranking (entropy for unit of cost)



Training: pattern selection

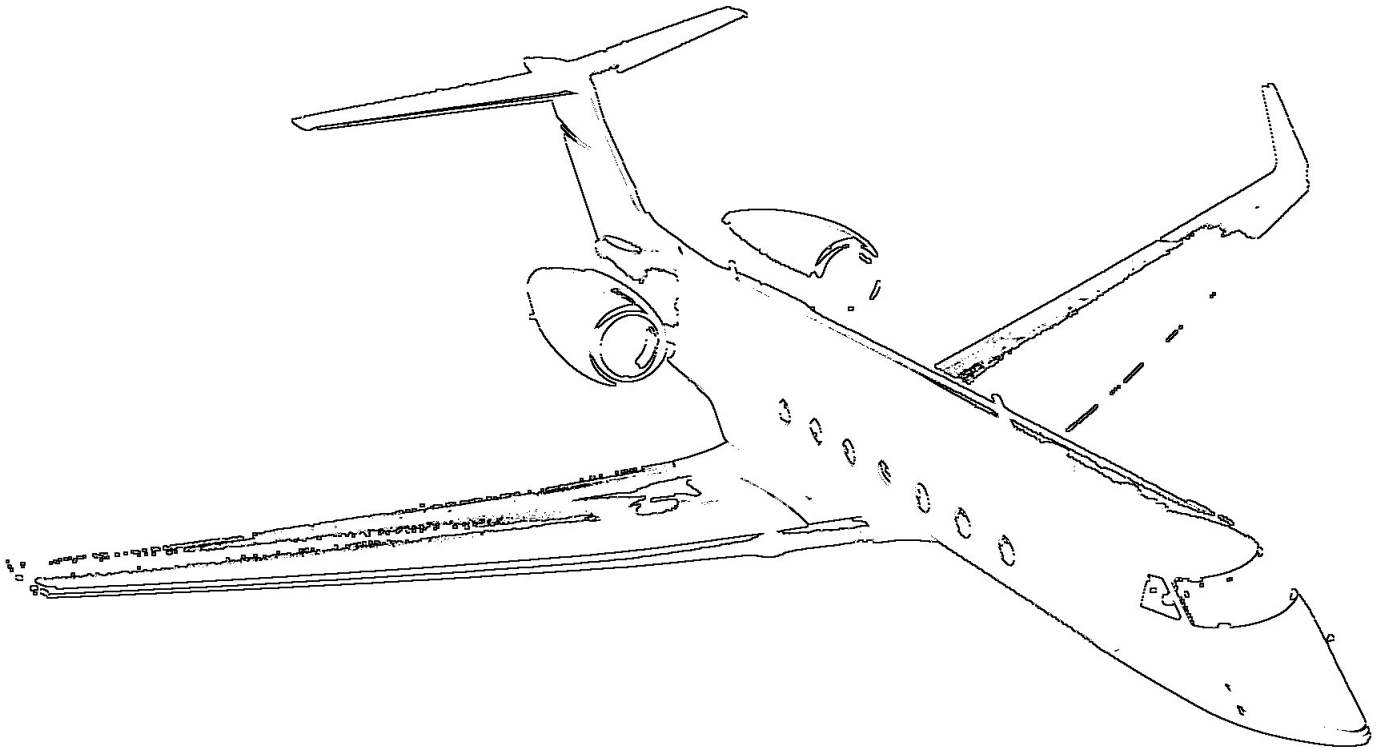
- i.e. the function
  - rejects patterns too rare or too popular
  - selects patterns relevant for edge detection

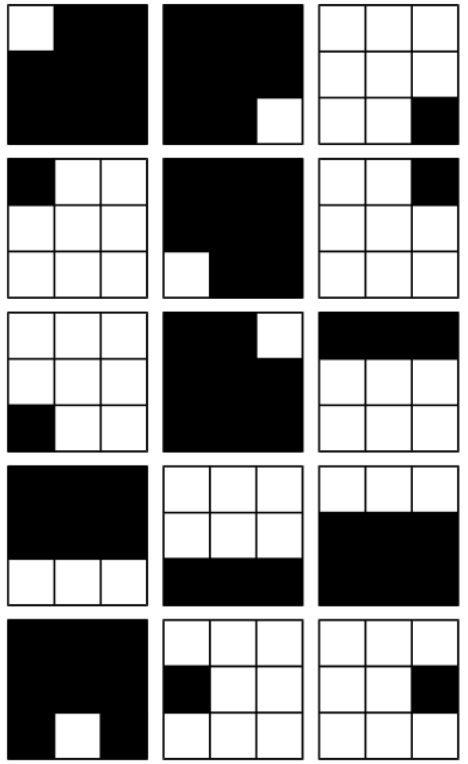




Pattern matching

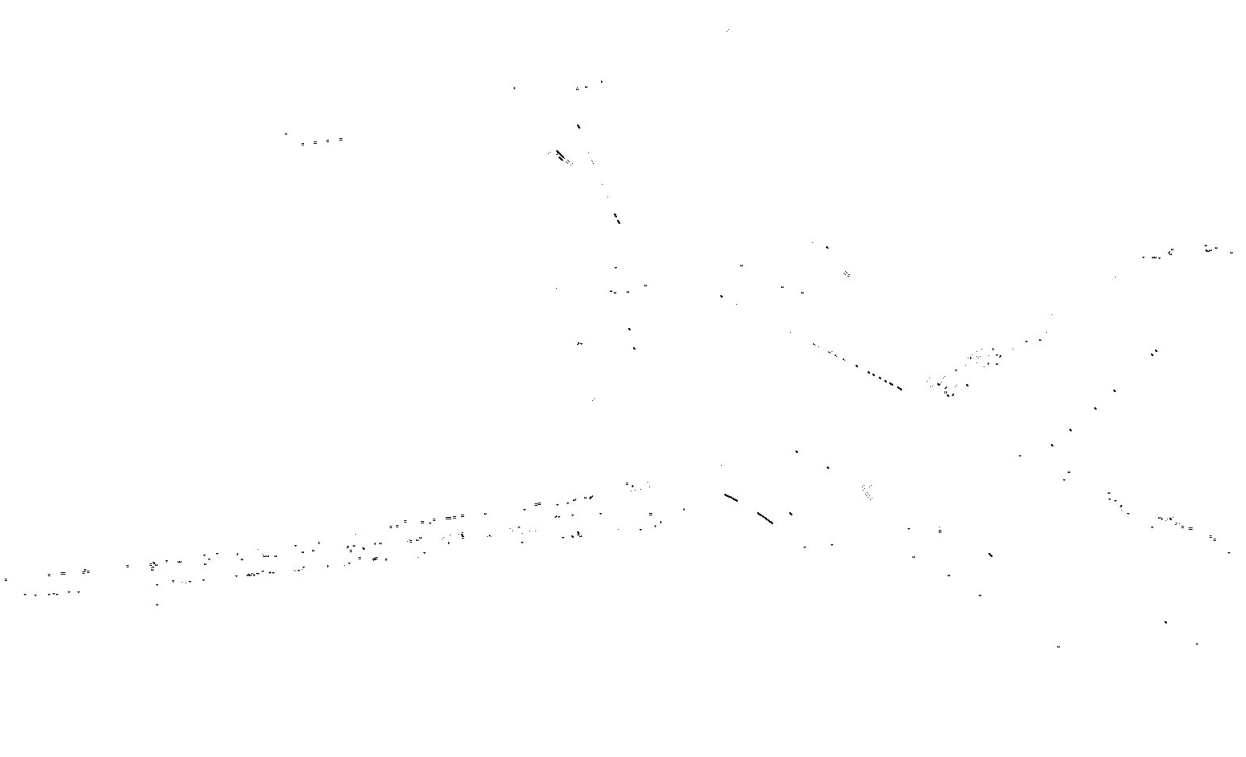
- Use the pattern bank to filter the image
  - The second stage (the fitting) is done by our brains
- If we use the selected pattern bank
  - Object can be identified w/ small amount of data

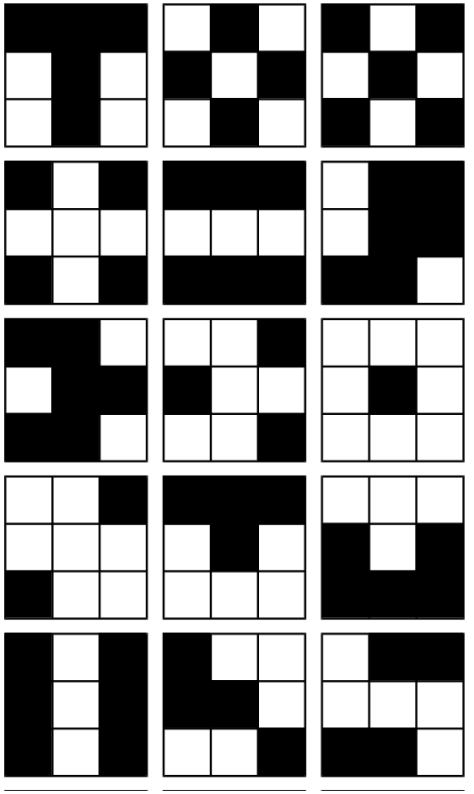




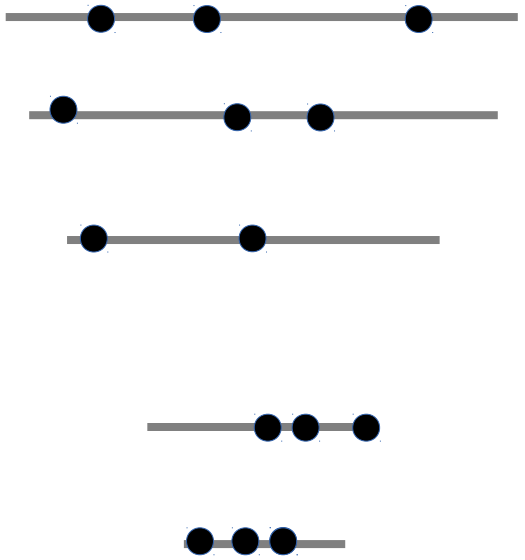
Pattern matching

- Use the pattern bank to filter the image
  - The second stage (the fitting) is done by our brains
- If we use a random pattern bank
  - The brain don't recognize it (same amount of data)

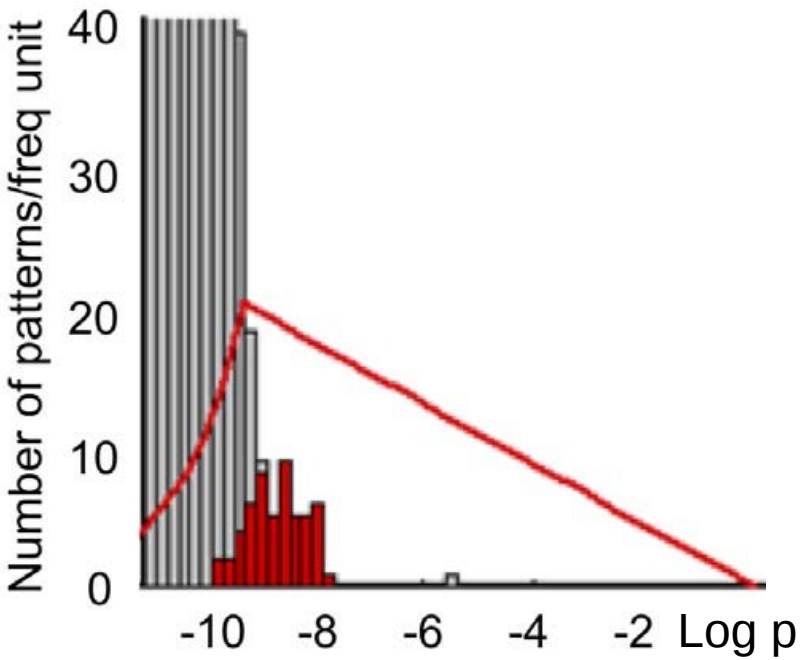




Associative memories

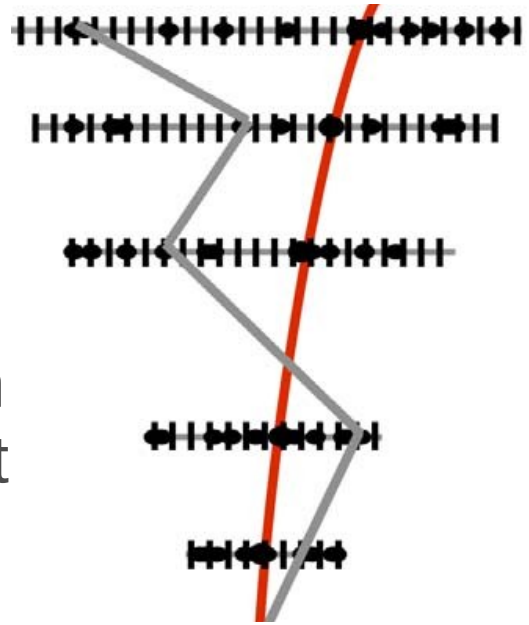


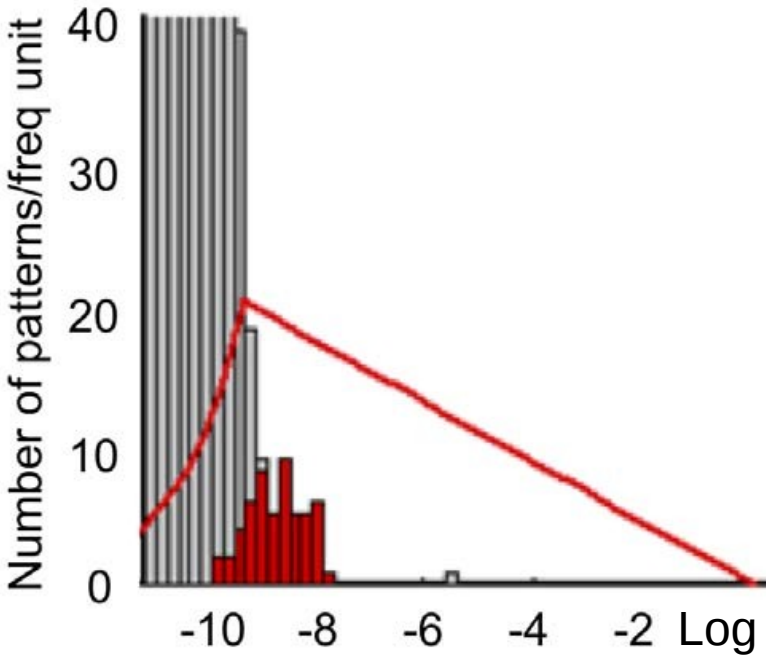
- Apply the model to HEP
  - Five layers of pixels
  - Use hits from collisions for training
  - Select patterns according to the selection function based on entropy per unit of cost
- Selected pattern bank is similar to the ones produced by simulation
  - i.e. the pattern bank is reproduced without any a priori knowledge of physics and detector geometry
  - only via training with data an the maximization of entropy in a system with limited resources



Associative memories

- Apply the model to HEP
  - Five layers of pixels
  - Use hits from collisions for training
  - Select patterns according to the selection function based on entropy per unit of cost
- Selected pattern bank is similar to the ones produced by simulation
  - i.e. the pattern bank is reproduced without any a priori knowledge of physics and detector geometry
  - only via training with data an the maximization of entropy in a system with limited resources

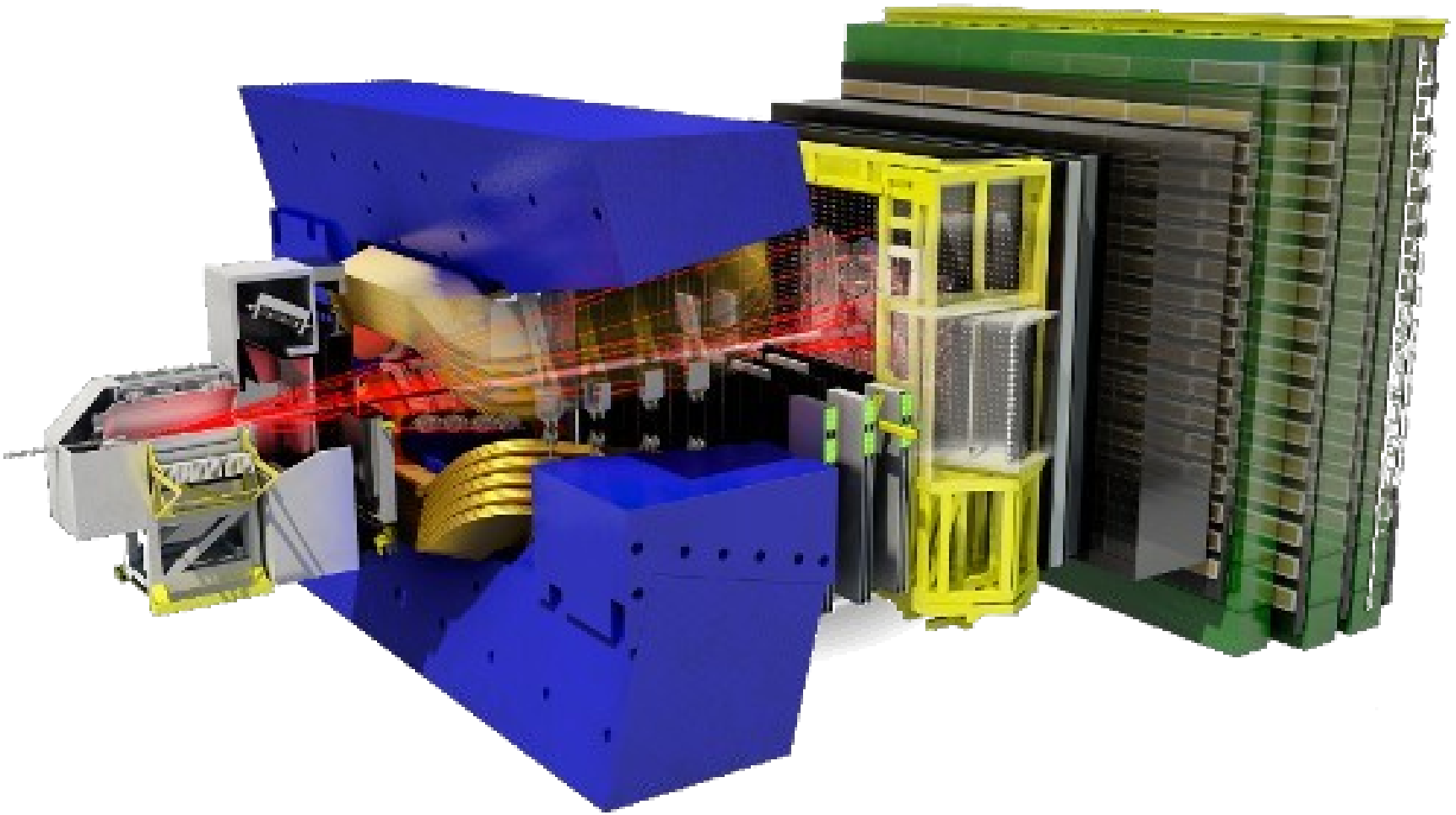




LHC upgrade

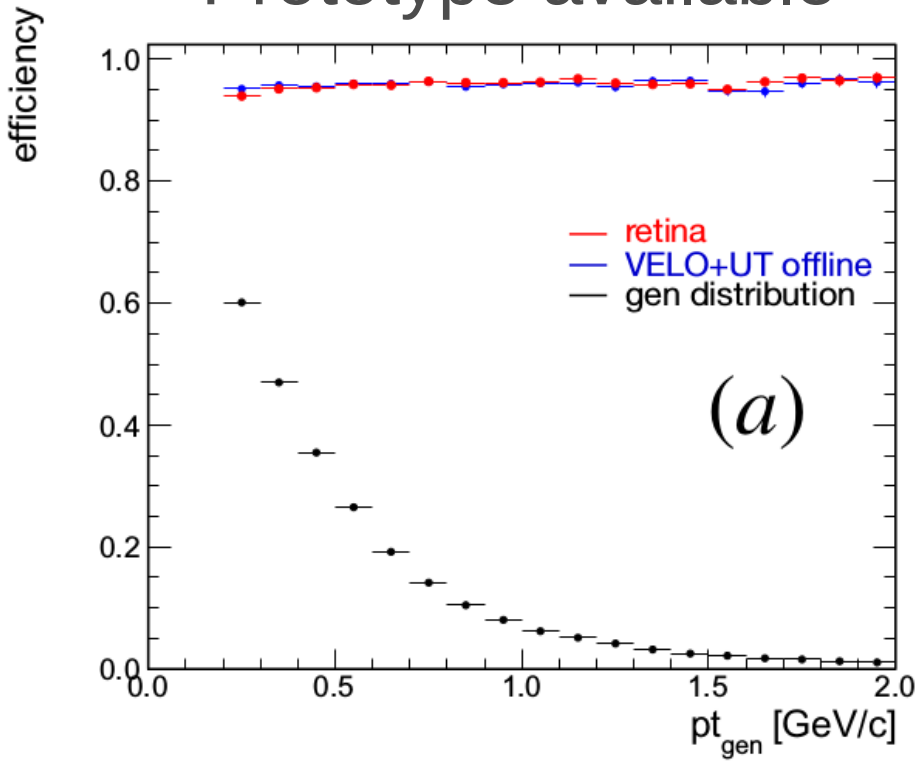
- Full tracking mandatory to cope with pileup
  - at HLT or L1?
- ATLAS
  - Baseline HLT only
  - AM based
- CMS
  - Baseline at L1
  - Multiple implementation options thanks to the preselection provided by the new ID double layers
- LHC-b
  - Moving to a triggerless architecture
  - Option to use retina to not saturate the farm

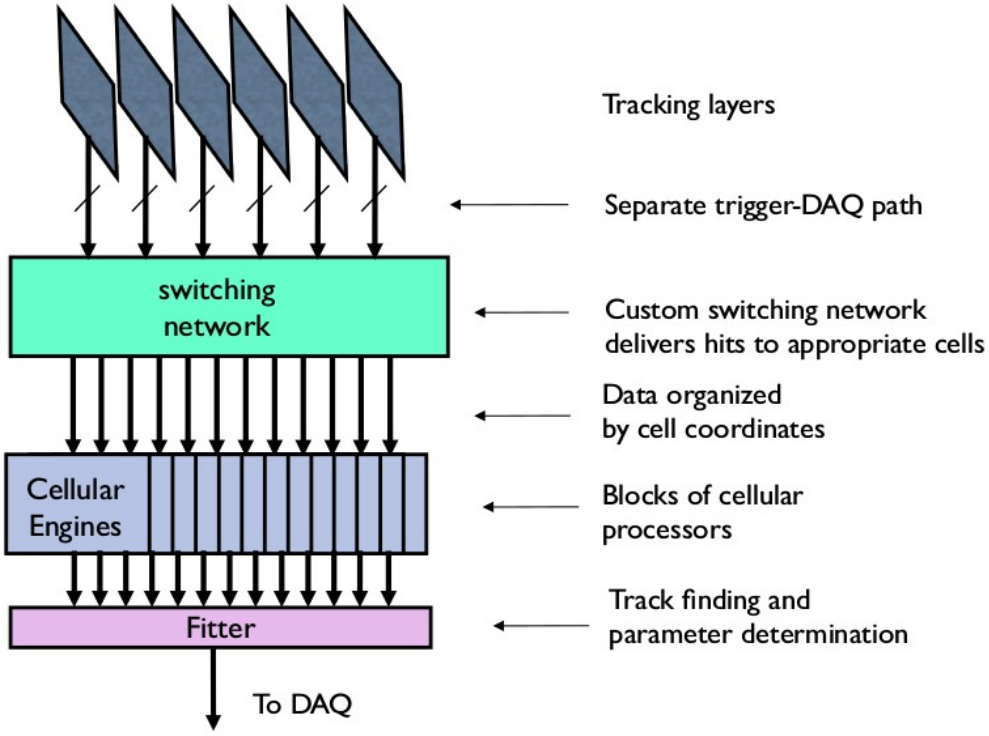
LHC-b



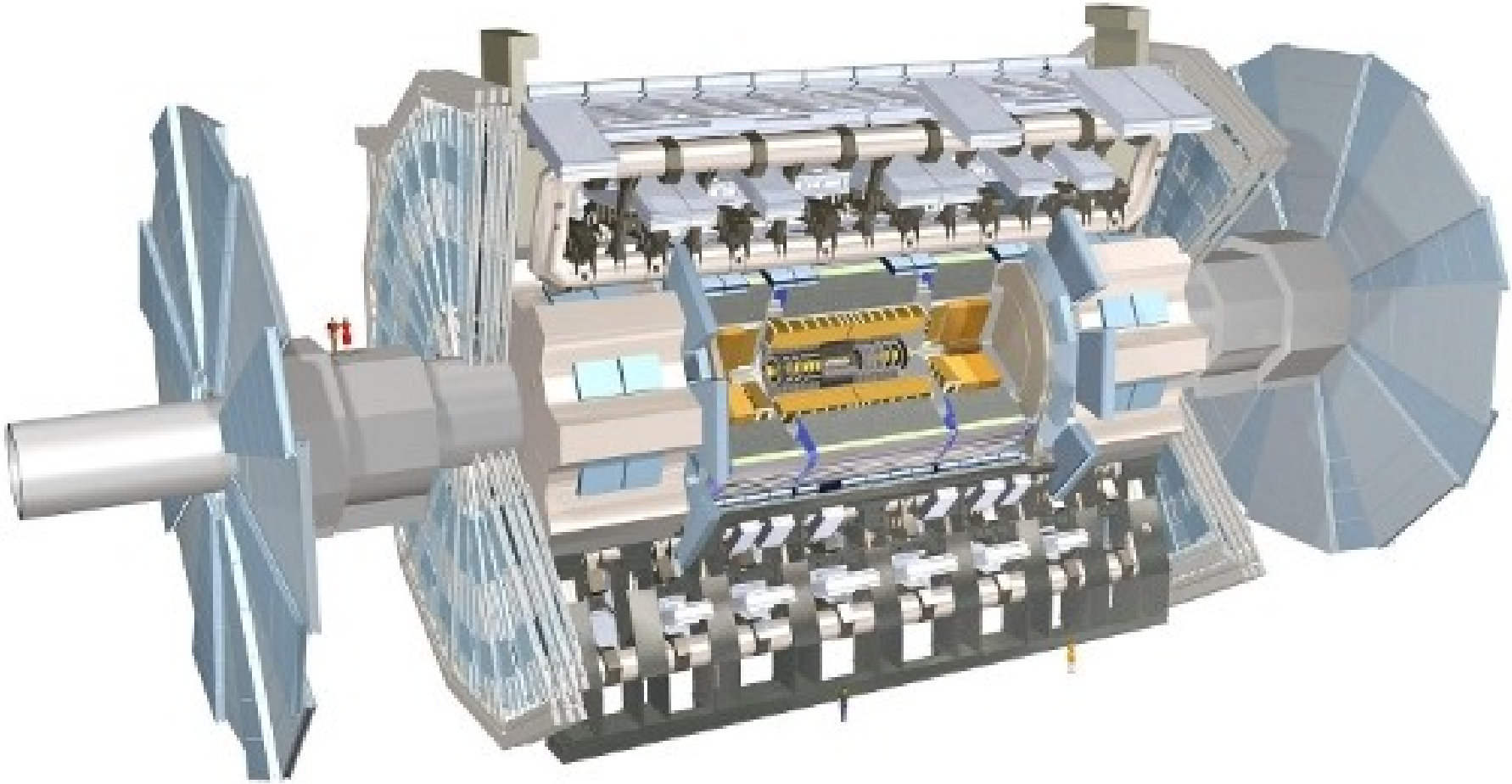
Retina prototype

- LHC-b moving to a trigger less design
  - Event processing at 40 MHz
  - FPGA based tracker before Event Builder can help to make online tracking affortable
  - Prototype available





ATLAS

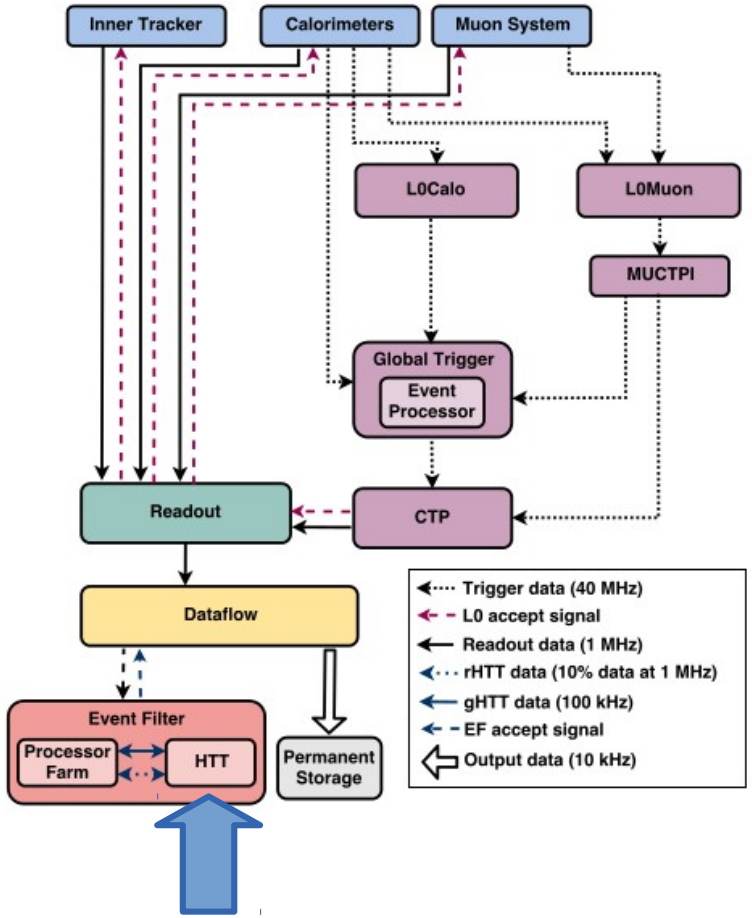


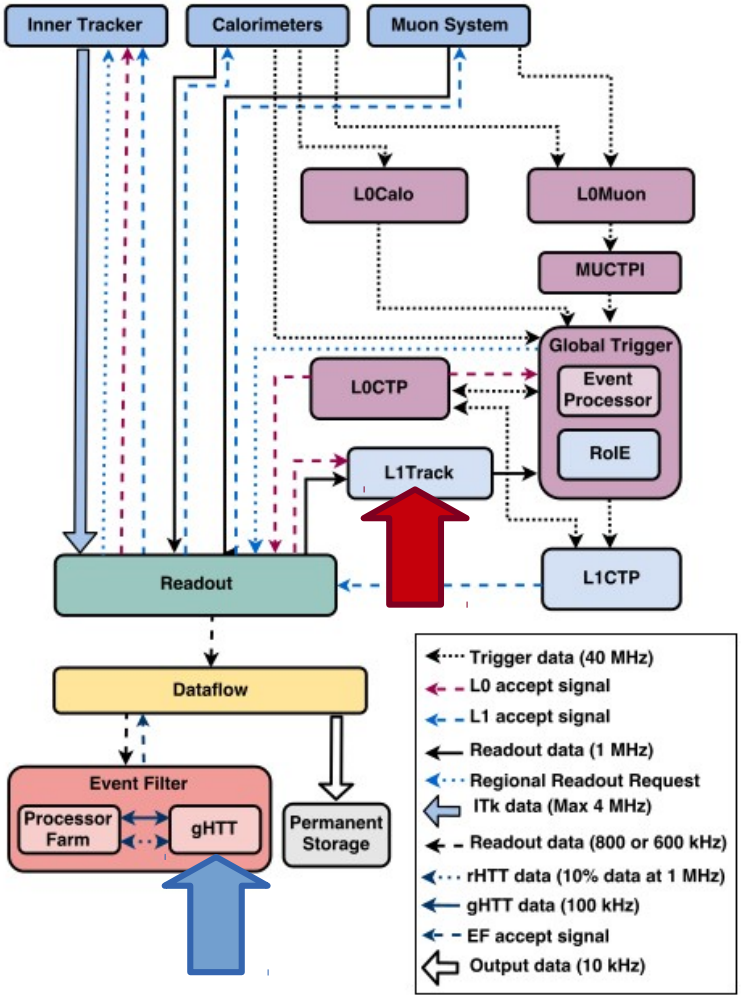
ATLAS

- Baseline: single level HW trigger
  - Level-0 (L0): muon and EM calorimeter trigger
  - w/ option to evolve to a two-stage trigger
- L0-only
  - Run a L0 trigger only, with full detector readout
  - Run tracking as part of the Event Filter: **EFTrack**
- L0/L1 option
  - Move to using regional readout initiated by L0
  - Add regional track triggering as an extra level: **L1Track**
- **L1Track** and **EFTrack** based on the same HW
  - Based on associative memory chips and FPGA for fitting
  - Track p_T down to 4 GeV for L1Track, 2 GeV for EFTrack

ATLAS: options

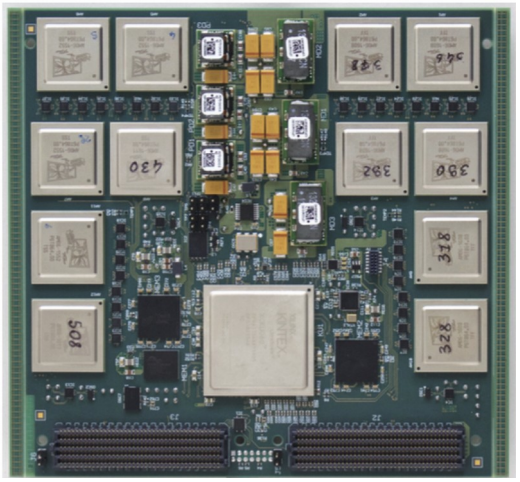
- Baseline: L0-only (left), L0+L1 on the right
  - **EFTrack** (HTT) and **L1Track** using same HW

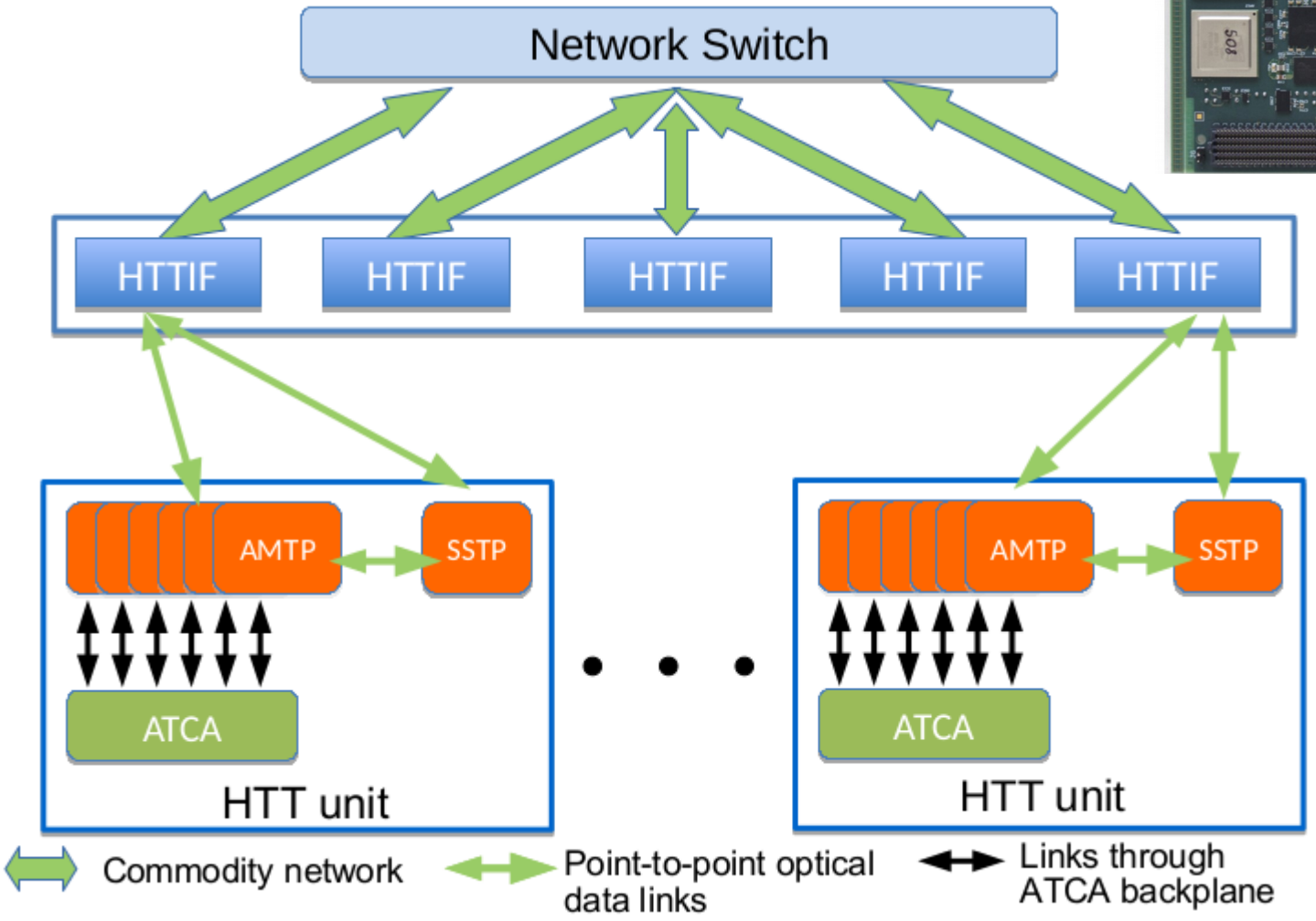




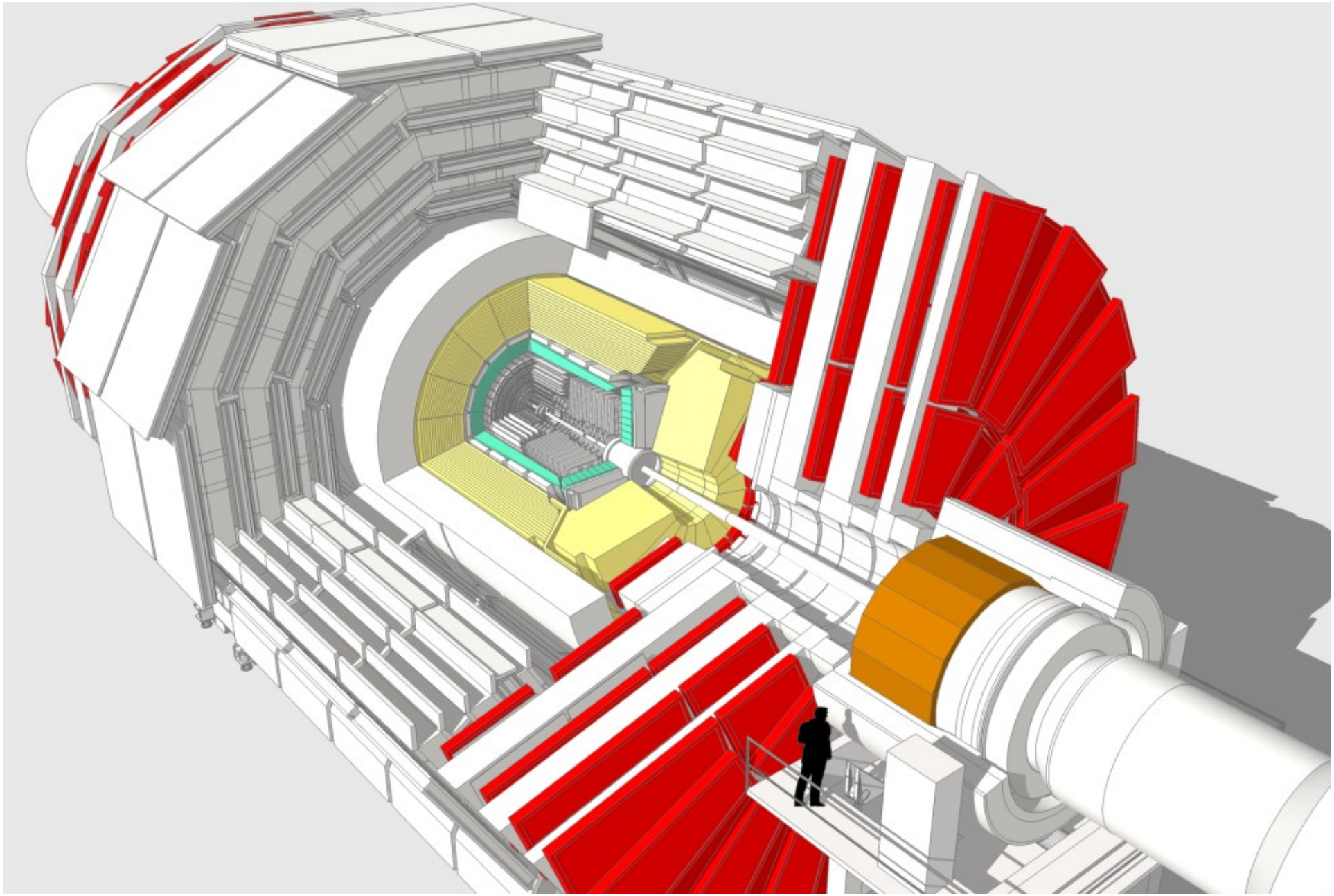
ATLAS: HTT

- Hardware Track Trigger
  - Coprocessor for HLT or … for L1





CMS

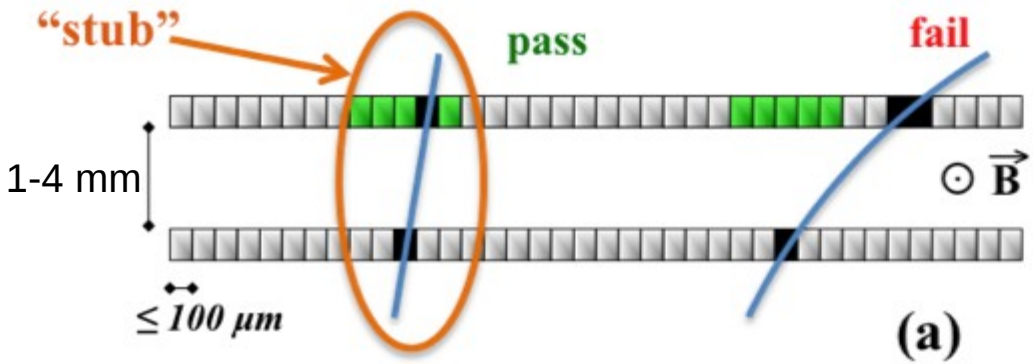


CMS L1 track: the challenge

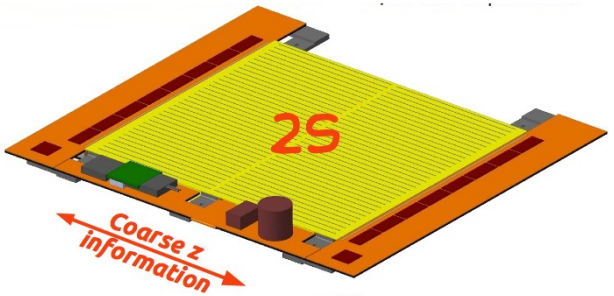
- Single level HW trigger
  - With tracking at L1
- Reconstruct trajectory of charged particles
  - In an extremely dense environment: pileup ~ 140-200
  - At an input rate of 40 MHz
  - With ~ 4 μs of latency
  - O(10K) Tracks/Bunch crossing
  - Tracker data ~ PB/s
- Challenge
  - Reduce the rate to something bearable by readout
  - At least 1 order of magnitude
  - Maybe … cutting on p_T before the readout

L1 track: stubs

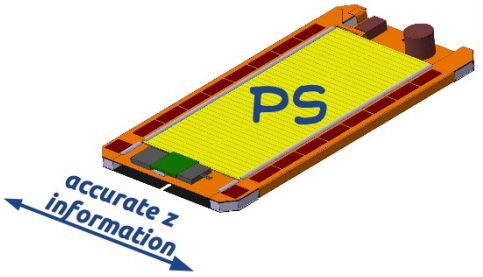
- Outer Tracker: layers/disks made of two modules of closely spaced silicon sensors
  - Charged particles produce pairs of hits: **stubs**
  - Relative position of the two hits determines track p_T (assuming beam-line origin)
- On-detector electronics only transmit off stubs consistent with p_T>2-3 GeV
  - Reduces rate by factor ~ 10



2 Strip sensors
Strips: 5 cm × 90 μm
Strips: 5 cm × 90 μm
P = 2.7 W
~ 92 cm² active area
For r > 40 cm

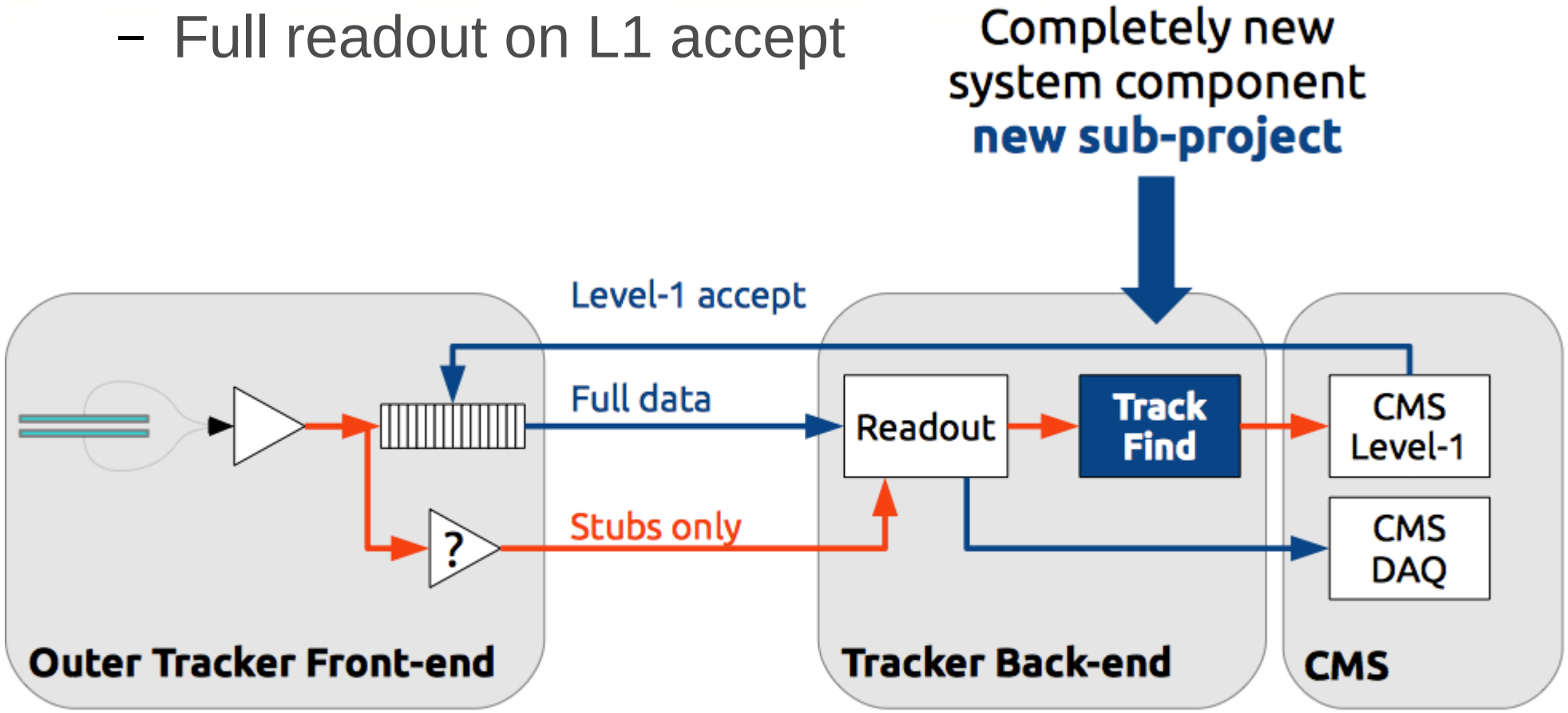


Pixel + **S**trip sensors
Strips: 2.5 cm × 100 μm
Pixels: 1.5 mm × 100 μm
P = 5.0 W
~ 44 cm² active area
For r > 20 cm



Readout

- Stubs read out at 40 MHz
  - Full readout on L1 accept



@ 40 MHz – Bunch crossing
@ ~ 500 kHz – CMS Level-1 trigger

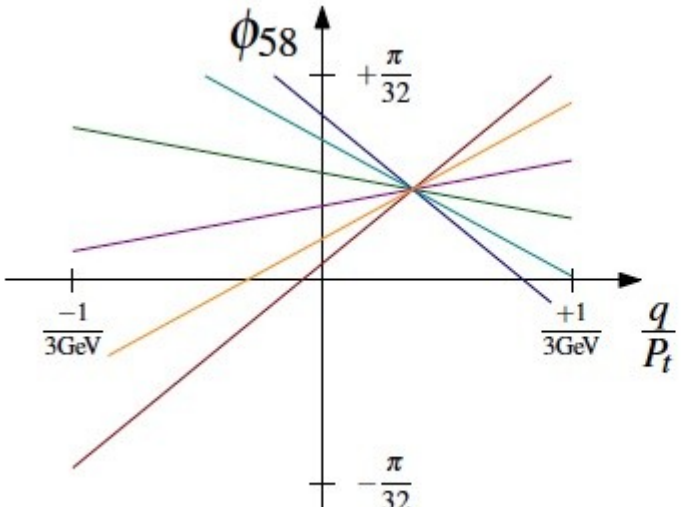
CMS: L1 tracking

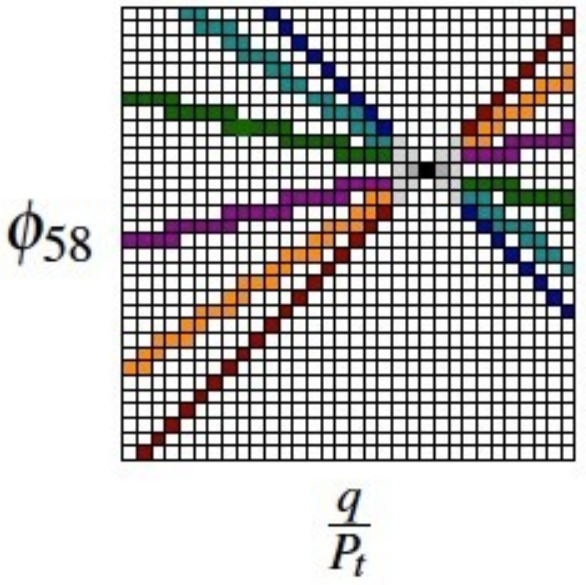
- The stub approach allow CMS to evalute three different tracking options

1. Associative memory + FPGA
   - tackle combinatoric with AM
   - FPGA for parameter estimation
2. Projective binning
   - Full FPGA
   - Hough Transform
3. Combined tracklet builder & linearized track fit
   - Full FPGA

Option 2: projective binning

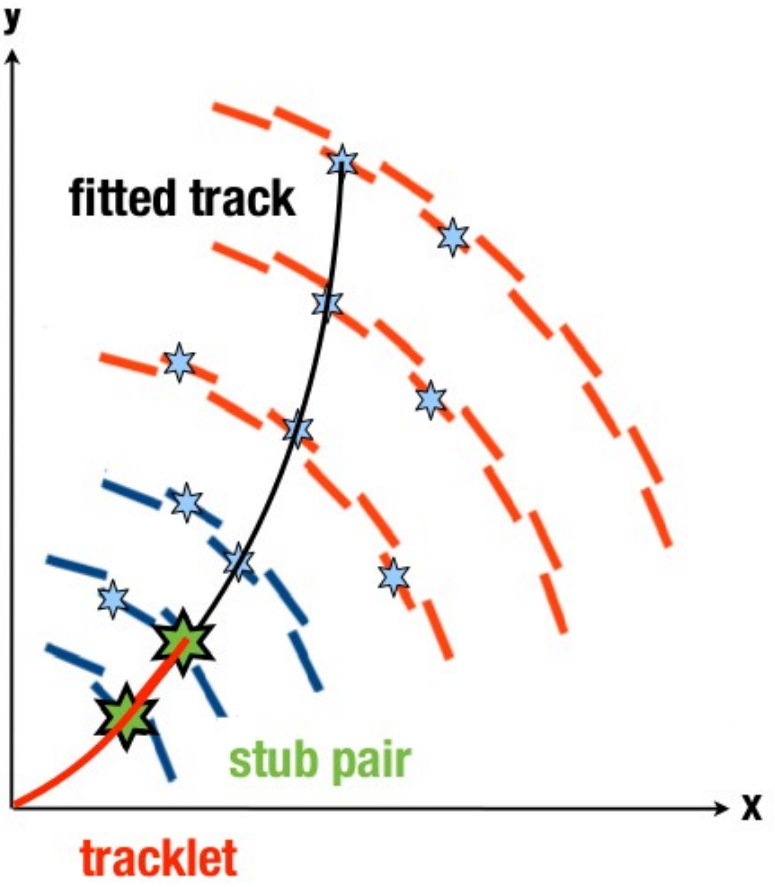
- Geometric processor sorts stubs in 36 subdivisions of the octant
- Patter recognition
  - coarse Hough Transform ran on the stubs
    - Each hit transforms to a line in φ, q/P_T space
    - Hits from the same physical track form intersecting lines
  - duplicates are removed
- Track fitting
  - To determine track parameters





Option 3: tracklets

- Seed by forming tracklets
  - Pairs of stubs in adjacent layers/disks
  - Initial tracklet parameters from stubs + beamspot constraint
- Project to other layers and match with stubs
  - Inside out & inside in
  - Calculate residuals
- Fit stubs
  - Linearized χ^2 fit
- Remove duplicate tracks



Summary

- Tracking in trigger is becoming more and more important in HEP
  - But limited by CPU resources
- Various approaches based on parallelism
  - Some ideas borrowed from other fields
- Common stages
  - Division of the workload based on detector geometry
  - Pattern Recognition at low granularity
  - Track fitting
- HW implementations
  - GPU, FPGA, ASICs (Associative Memories)
  - Or combinations or the above