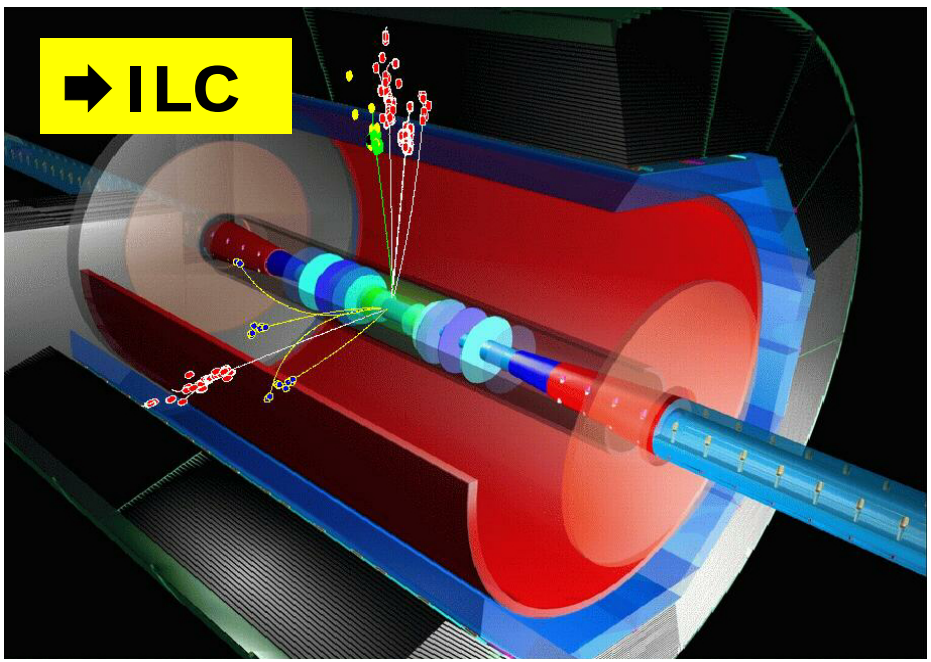
Software Needs for ILC Detector Optimisation

or ……. Why are we here ?

Mark Thomson
University of Cambridge



This talk:

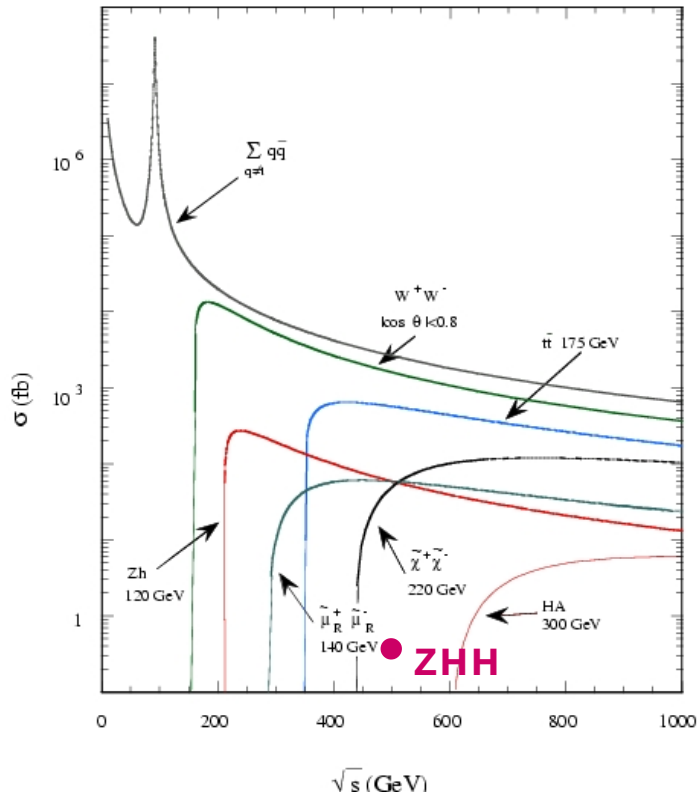
- ★ Motivation
- ★ What to Optimise ?
- ★ How ?
- ★ Hands-on experience
- ★ Software Requirements
- ★ The next step
- ★ Conclusion

❶ Motivation

ILC Physics:

Precision Studies / Measurements

- ★ **Higgs** sector
- ★ **SUSY** particle spectrum
- ★ **SM particles** (e.g. W-boson, top)
- ★ and much more...



Difficult Environment:

- ★ **High Multiplicity final states**
  often **6 / 8 jets**
- ★ **Small cross-sections**
  e.g. **$\sigma(e^+e^- \to ZHH) = 0.3$ fb**

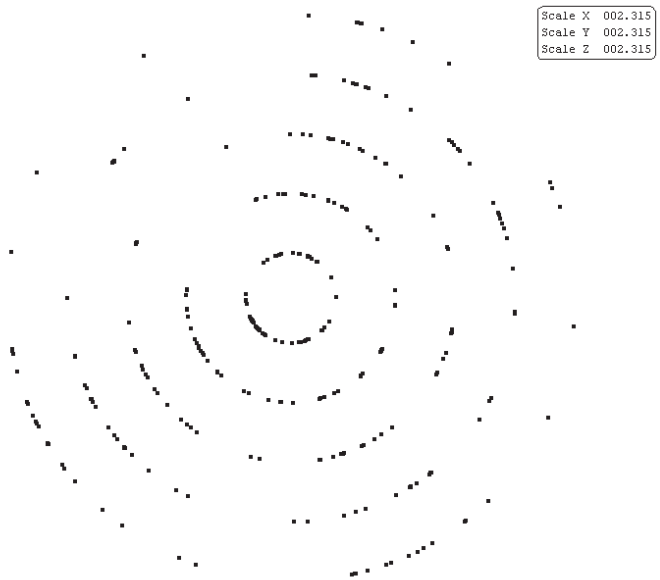
- ✸ **Detector optimized for precision measurements in difficult environment**
- ✸ **Only 1(?) detector – make sure we choose the right options**

❷ What to Optimize

The Big Questions (to first order):

❶ CENTRAL TRACKER

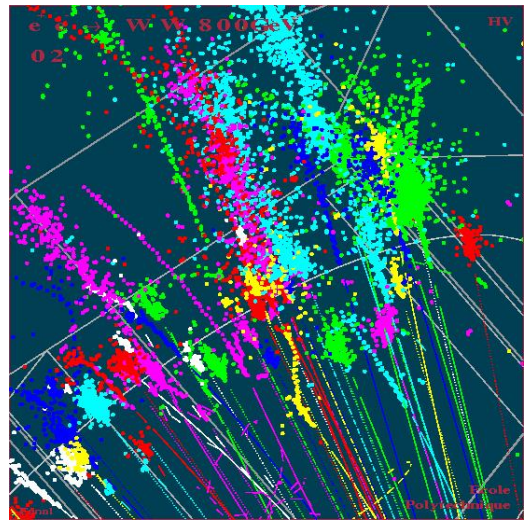
★ **TPC vs Si Detector**



★ **Samples vs. granularity – pattern recognition in a dense track environment with a Si tracker ?**

❷ ECAL

- ★ Widely (but not unanimously) held view that a high granularity SiW ECAL is the right option
- ★ BUT it is expensive
- ★ Need to demonstrate that physics gains outweigh cost
- ★ + optimize pad size / layers



❸ HCAL

- ★ Higher granularity digital vs lower granularity analog option

❹ SIZE

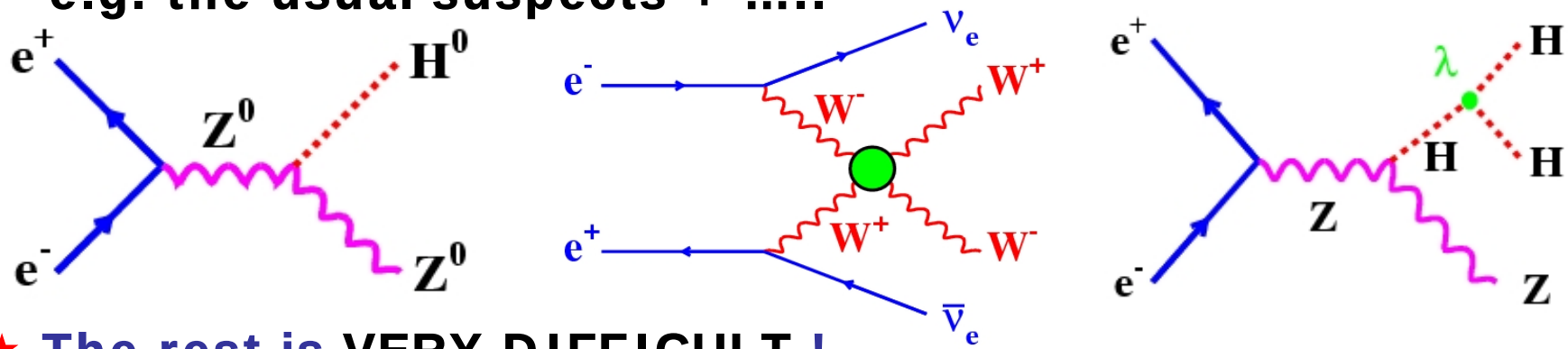
- ★ Physics argues for:
  large + high granularity
- ★ Cost considerations:
  small + lower granularity
- ★ What is the optimal choice ?

❸ How ?

★ **Optimize** detector design using **key** physics processes

★ Choosing the reference processes is relatively **EASY !**

e.g. the usual suspects + …..



★ The rest is **VERY DIFFICULT !**

★ **Need unbiased comparison**

- Same / very similar **reconstruction algorithms**
  - **these need to realistic (i.e. start-of-art)**
- Common reconstruction framework
- Same Monte Carlo events
- Repeatable by others – user friendly software

How to proceed ?

Different approaches for different sub-detectors:

- ★ **VTX : design driven by heavy flavour tagging, machine backgrounds, technology**
- ★ **Tracker : design driven by σ_p, track separation**
- ★ **ECAL / HCAL : single particle σ_E not the main factor ➡ jet energy resolution ! Impact on particle flow drives calormeter design**

For VTX and TRACKER can learn a lot independent of rest of detector design. NOT TRUE for ECAL / HCAL need to consider entire detector

But TRACKER is a big influence on size / cost

Likely Approach to Detector Optimization:

- ★ **Need to consider entire detector**
- ★ **Very wide parameter space !**
- ★ **Choose a few baseline "detector concepts" (2 < few < 8)**
- ★ **Cost on same basis and compare performance**

❹ Some First Hand Experience

c. September 2004

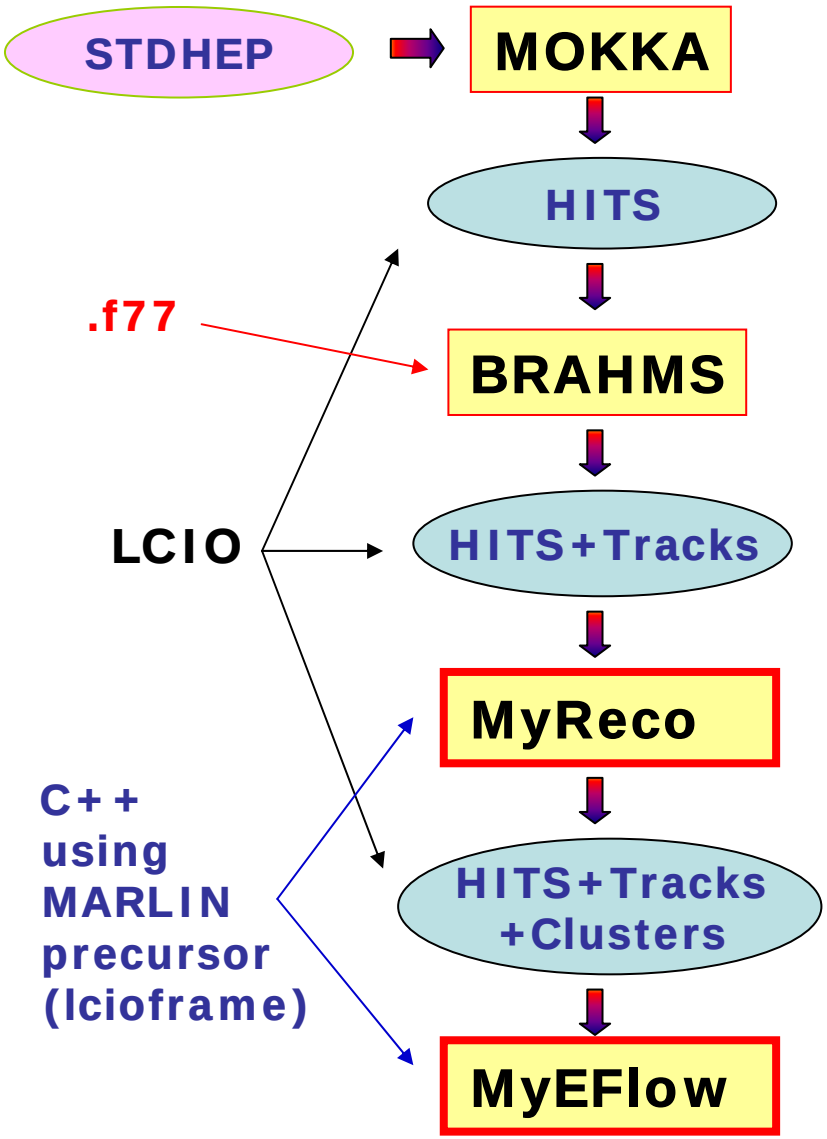
A few relevant questions

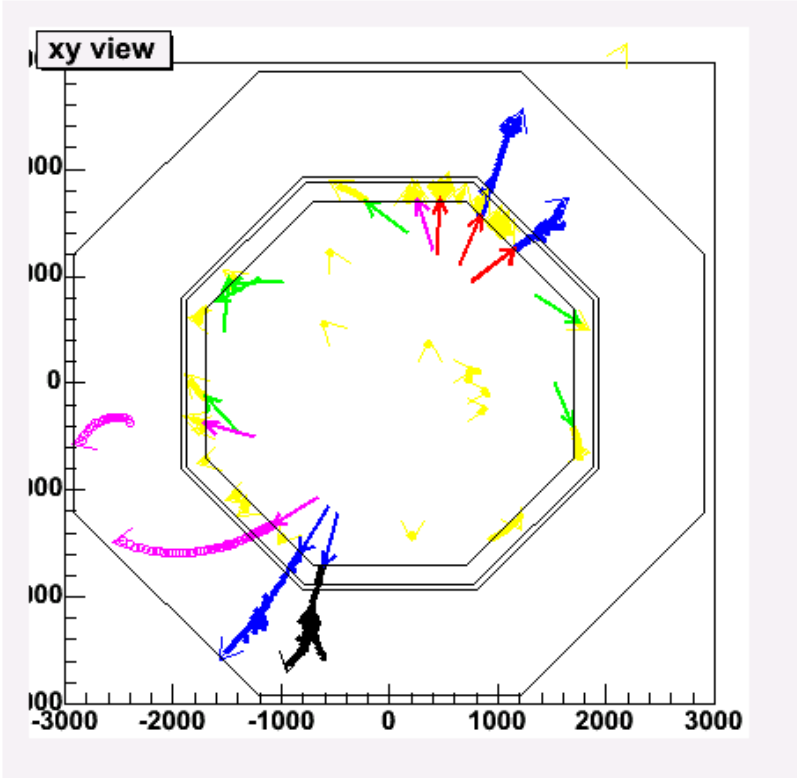
- ★ What software do we need to start to perform these studies ?
- ★ How much already exists ?
- ★ What needs to be worked on ?
- ★ Best way to find out.... give it a try

Basic Plan

- ★ Develop geometry indep. ECAL / HCAL reconstruction using LCIO as data format (starting from code from Chris Ainsley)
- ★ Develop particle flow algorithm in same framework
- ★ Study jet-energy resolution for Z^0s
- ★ Repeat for different detector lengths / radii
- ★ Encountered a number of problems.....

Overview of Code





- ★ Surprisingly easy to get something that worked !
- ★ Not perfect, but OK
- ★ Then came the hard bit.....
- ★ No easy way to modify detector size

The Good, the Bad and the Ugly

The Good:

- ★ Once set up **MOKKA** very user friendly
  - + easy and relatively quick to generate any file wanted
- ★ **LCIO** data format
  - + very easy to use, nice lightweight data format
- ★ **MARLIN-like** reconstruction framework
  - + easy to use, again nice and simple

The Bad:

- ★ No easy way to change detector geometry
  - not surprising, this bit was never going to be easy
- ★ Lots of hard-coded numbers !
  - **ECAL/HCAL** reconstruction was written to be geometry indep.
  - achieved by shoving hard-coded numbers in a custom object
  - need a mechanism within reconstruction framework
- ★ A number of issues with tracking
  - track objects were too lightweight (addressed in **LCIO1.03** ?)
    e.g. difficult to identify/reject bad tracks
  - tracking code would not have worked had geometry changed

The Ugly:

- ★ At time **LCIO** didn't write out tracks
  - wrote out **ASCII** file and added module to create **LCIO** tracks

❺ Software Requirements

To summarise the above:

★ Learnt a lot in a relatively short space of time < 2 weeks
★ Biggest plus: LCIO/Marlin-like framework worked well
- simple and easy to use…
- resist temptation to over-complicate it in the future…

The way forward:

★ So what next……
★ What software tools are needed to perform ILC detector optimisation

Software Requirements : MC

★ Two possible approaches

- Detailed Simulation as in MOKKA / BRAHMS
  - Great – bad harder to modify
- Simplified approach e.g. used in US Studies
  - Not as rigourous but easy to modify

LIKELY APPROACH (2 Stages):

★ A few baseline "detector concepts" decided upon by yet more wise men / women
- these will need to be implemented within MOKKA
- not trivial (i.e. expert job)

★ + some more specific studies, e.g. vary ECAL layers within a detector "concept"
- ideally want easy interface to MOKKA geometry

★ Non-trivial but necessary

Software Reqs : Reconstruction

Some General Comments:

★ **LCIO is the way forward**
- common format for worldwide studies
- will allow packages to be run worldwide

★ **There is already a lot of excellent "Tesla" reconstruction software**
- needs to be put in LCIO/MARLIN framework (either f77, C++, java)
- needs to be written in a geometry independent way i.e. pick up geometry from data

SPECIFIC NEEDS:

❶ **Tracking:** → **TPC**, **SiD**

Very different problems, so probably different algs.

★ **Code must be "geometry independent"**

★ **e.g. TPC code should work for wide range of TPC sizes/pad sizes**

★ **THIS IS A SIGNIFICANT BUT VITAL EFFORT**
- writing good tracking code is far from easy

★ **Ultimately forward tracking needs revisiting !**

❷ ECAL / HCAL Clustering :

- ★ again need “geometry independent” code
- ★ strongly coupled with particle flow

❸ Particle Flow

- ★ lots of excellent work already, e.g. **SNARK**, **REPLIC**
- ★ need to be put in “geometry independent” LCIO framework

❹ VTX : Heavy Flavour Tagging

- ★ it would be really nice to have heavy flavour tagging in the same framework
- ★ has a significant impact on many physics studies

Need to get code into this new framework as soon as possible

All reconstruction code must aim to be flexible enough to handle reasonable range of detector parameters

Software Reqs : Geometry

Need some way of propagating detector geometry to reconstruction code

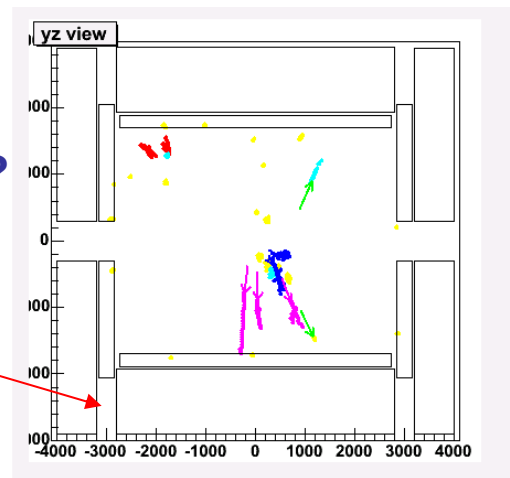
★ **Two possible approaches**

- **Datebase (e.g. MySQL)**
  - Fine – but adds complexity
- **Hard-coded LCIO structures for limited set of geometries**
  - Simple – and would work for studying a few "concepts"

Need to think carefully about what's needed….

e.g. for ECAL reconstruction:

- Layer positions (assume Octagonal geometry ?
- Pad sizes in layers
- Radiation lengths between layers
- + some description of in active volumes
- +…….



❻ Summary

- Timescale is fairly short
  - (being optimistic) we could be talking about writing a detector CDR/TDR within the next 1-2 years.
- The ILC Detector optimisation problem is NOT EASY
  - it will require a lot of work
- BUT a lot of fun projects !
- The framework is easy to use – easy to start real work
- Main Emphasis on developing geometry independent packages in LCIO/MARLIN framework

For this mini-workshop (what I would like to see):

- Try to agree on "geometry object" ?
- Need people/groups to COMMIT to writing new packages (or converting existing packages into new framework) + room for multiple packages