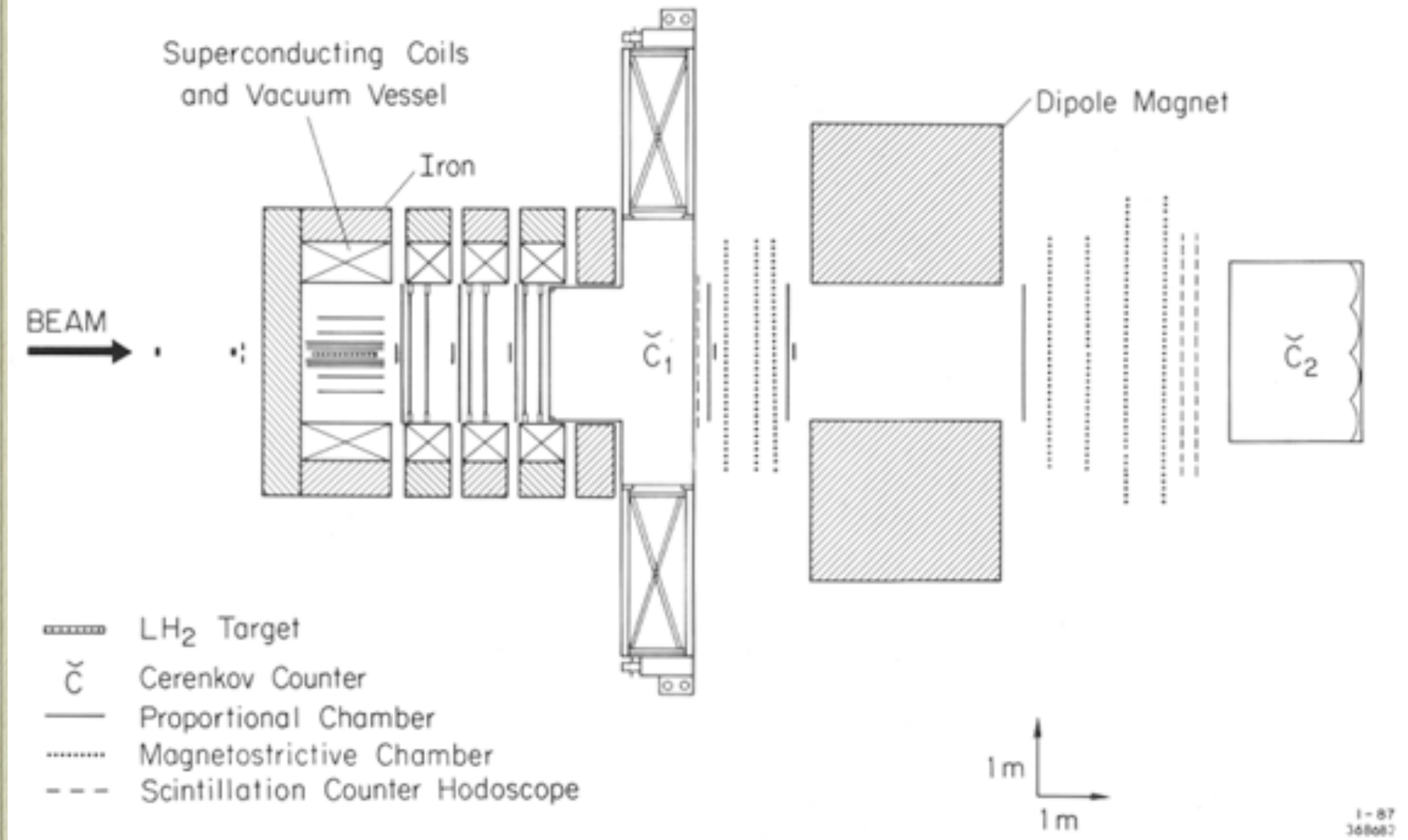
LASS–E135 Experiment Data Preservation

David Aston, SLAC

Presented at the

2nd Workshop on Data Preservation and Long Term Analysis in HEP

May 26, 2009



"Electronic Bubble Chamber"

Open trigger

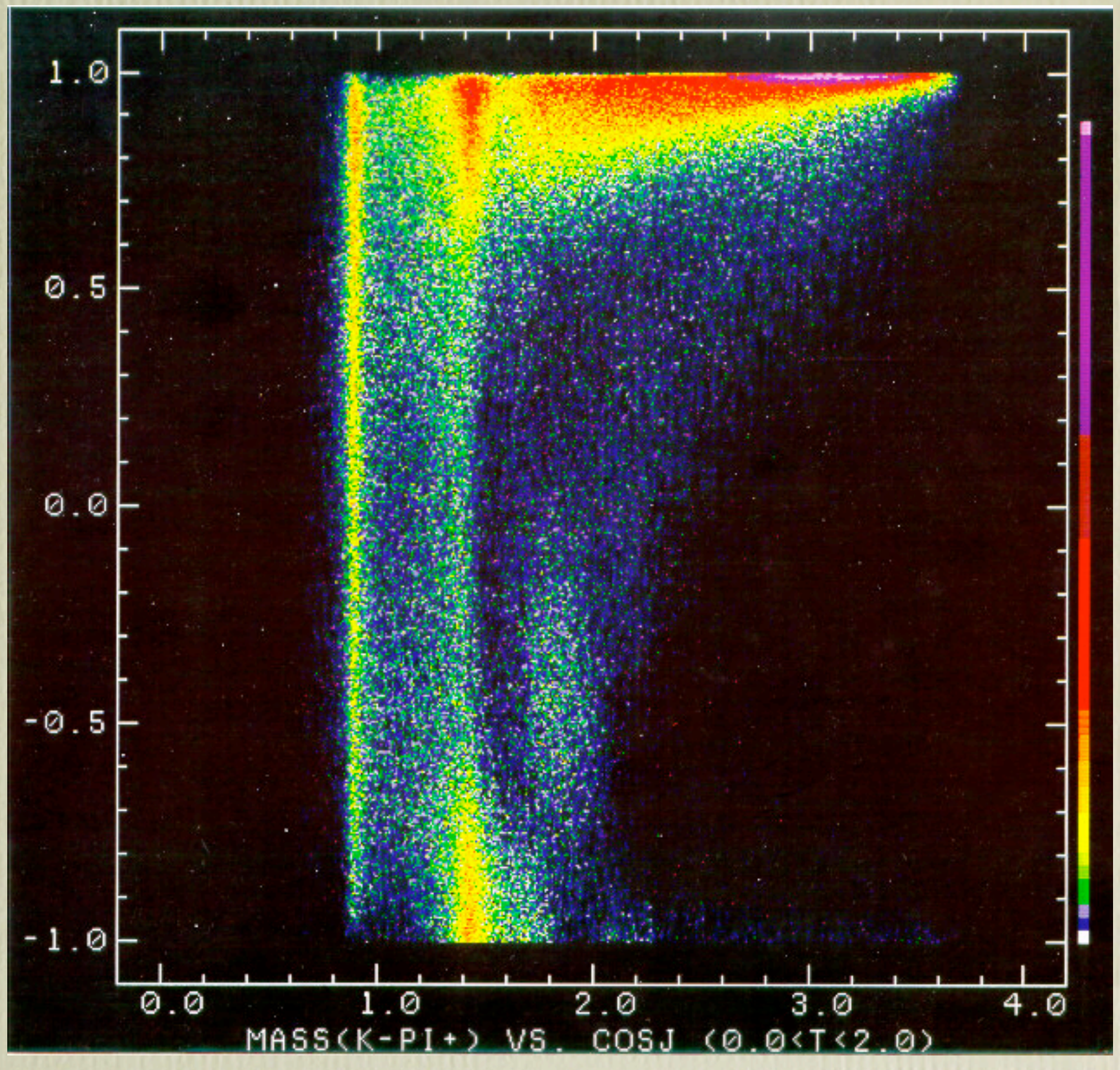
K^+ and K^-p at 11GeV/c

Took data in 1981 and 1982

Kπ scattering

- Underlying states in S– and P–waves, derived from this $K^- p \rightarrow \pi^+ n$ data

- Probably our most well-known result



Data scale of the experiment (1)

- total ~140M triggers
- but ~100M physics events (reconstruct all!)
- on ~3000 raw data tapes (2400′, 6250Bpi; 0.5TB)
- CPU needed to reconstruct all the data once was ~6 IBM370/168 years —2 years of the total SLAC "Triplex" analysis capacity!

Data scale of the experiment (2)

- Solution! —168E's (project led by Paul Kunz)
- together with night/weekend access to a Hitachi mainframe at Nagoya University we completed reconstruction in just over one year. [They actually used to switch it off outside working hours! Our collaborators persuaded them to keep it running.]
- results on ~1000 DST's (150MB)

Software characteristics (1)

- All reconstruction was (pre–77) FORTRAN, targetting in particular, the IBM FortHX compiler. Most analyses later adopted Fortran 77.

- We recognised early the need for standard `COMMON` definitions. We used a "SLAC standard" pre-processor which keyed on "=" in column 1:

  `=SLTRKS` equivalent to

  `include "SLTRKS  COMMON"`

- our "objects" were tracks and vertices which had customised code to pack/unpack to their standard `COMMON`s (`xxDMP, xxREAD`)

Software characteristics (2)

- We used "structures of arrays" defined in our standard COMMONs. Object counts were small, so this was quite acceptable, eg:

```
C-----------------------------------------------------
C   E135 DST Solenoid track bank.
C
      COMMON /SLTRKS/ NSLOUT,RTR(40,20),SERRMT(15,20)
      INTEGER      NSLOUT
      REAL         RTR
      REAL         SERRMT
      INTEGER*4                ITR(40,20)
      EQUIVALENCE  (RTR(1,1),ITR(1,1)
C
C     NSLOUT = # of solenoid tracks written onto DST event record.
C
C     RTR(40,*) is described below.  A * means INTEGER variable.
C...
```

and what each array item means is documented in the COMMON itself

Data Preservation Strategy (1)

- Make exact copies of tape files
- Port the software to run on Unix
  - use cpp and #include for `COMMON`
  - write code to parse IBM VBS record structures (already a solved problem — Vax/VMS)
  - convert from IBM floating point to IEEE (our `xxREAD, xxDMP` conventions made this quite simple)
- OK for big–endian (SunOS, MacOSX(PPC))

Data Preservation Strategy (2)

- Deal with byte–swapping, also runs on x86
- Analysis code works on "*any*" Unix with g77
- Reconstruction and MC programs still to be ported!
- Data copying has been done twice:
  - "round" to "square" tapes (only 200MB capacity) DST's only, raw data was trashed ($$)
  - to 50GB(?) tapes in HPSS. Few tapes, so we have duplicate copies.

Conclusion

- It's time to end this exercise in computing nostalgia/archaeology!
- Lessons learned:
  - simplicity helps
  - keep focussed on the *data* rather than its wrappings
  - the asymptotic cost of preserving your data →0!

Thank you!