SLAC-PUB-426
July 1968
(EXPI)

AUTOMATIC FILM DIGITIZER PERFORMANCE AND PLANS AT SLAC*

J. Brown
Stanford Linear Accelerator Center
Stanford University, Stanford, California

ABSTRACT

This paper contains a description of the hardware and software used in the automatic data analysis of spark chamber pictures, and a report on the results to date. Plans for the analysis of bubble chamber pictures are outlined briefly. The work described herein has been done by members of the Automatic Data Analysis Group at the Stanford Linear Accelerator Center.

(Presented at the International PEPR Colloquium, University of Nijmegen, June 5-7, 1968.)

*Work supported by the U.S. Atomic Energy Commission.

I. HARDWARE

The hardware used to digitize the spark chamber film is the so-called Hummingbird II (or Gooneybird). It is basically a CRT-generated flying spot device, similar but not identical to devices like an HPD, and definitely unlike a PEPR film scanner. With the qualifications noted below, the spot is driven in a raster-type scan across the face of a 7" CRT (having a P-24 phosphor) and is imaged onto the film, at approximately 1:1 magnification, to cover an area 60 mm by 104 mm. A photomultiplier located behind the film delivers a pulse wherever the spot is obscured by a black area on the film. The coordinates of the spot at such a point are effectively measured by the number of the scan line and the time elapsed since the start of that line. There are no gratings used, nor is there any splitting of the beam to generate reference signals.

The scanner is capable of executing a limited number of commands sent to it from the computer. The raster can be started and stopped at arbitrary lines, and digitizings can be gated "on" between two points on these lines. This means we can access arbitrary rectangular areas on the film. One can also select the scan-line density, i.e., one can scan every line, every other line, or every fourth line. The digitizing threshold can be set to any one of 16 levels. One can also set the scan mode, i.e., scan parallel or perpendicular to the direction of film motion. When the mode is switched, amplifier gains are automatically adjusted so that the raster still covers the same area on the film.

Since the current experiment we are working on uses 70 mm single-strip sprocketed film, the film drive is very simple; a stepping motor is used to move the film in either direction. The scanner will accept a "move film" order which allows the program to move the film up to 20 frames in either direction, in increments of 1% of a full frame length.

The data lines to and from the scanner are connected to a so-called "device selector." This is effectively a switch which can be set under computer control, that allows one to select any of our various scanners and display units. Included with this device selector is a simulator, which allows one to set up, by means of toggle switches, the bit patterns corresponding to various computer orders to the scanner (or to any device). One can then run the scanner in an off-line mode. This has proven very useful for testing and debugging purposes.

The device selector (constructed by us) is connected in turn to a parallel data adapter, an IBM supplied interface unit attached to a selector channel. The overall computer configuration is shown in Figure 1. The Model 75 processor, with a half-million bytes of core, will be replaced this September with a Model 91 having two million bytes of fast core.

II. SOFTWARE

A. General System Considerations

Our general approach to the software is based on several considerations, over some of which we had no control.

(1) An early decision was made to provide SLAC with a single very large central computing facility. At the same time, no funds were provided for any small interfacing computer. Thus we are connected directly without buffering into a channel which we tie up almost continuously during our scanning operation.

(2) We felt that data received from the scanner should be processed immediately so that we could be assured of its validity. This implied that we had to have a reasonably sophisticated on-line program resident in the computer while the scanner was in operation.

(3) On the Model 75, the present system, known as MFT (multiprogramming with a fixed number of tasks), permits one (with some difficulty) to run two user

programs concurrently, one occupying a fixed region of 278 K bytes (the so-called "batch" partition), the other, a region of 102 K bytes (the so-called "scope" partition). Present plans for the Model 91 call for it to be run with a fixed number (roughly 4 to 8) of "partitions" or core modules, under MVT (multiprogramming with a variable number of tasks). In this scheme it is expected to be operationally easier to run concurrent programs. In either case, our programs must be designed to run in a multi-programming environment.

Two other considerations relating directly to the particular experiment also affected our program design. (1) The experiment, which studies μ-p elastic and inelastic scattering at a variety of energies and momentum transfers, will, when completed, comprise a fairly large number of events ($>10^5$) and a relatively low fraction of good events per picture (typically one frame in 3 or 5 contains a useful event). Since our processing time is almost independent of whether a frame contains a good event or not, and since we are talking about a large number of pictures, it was decided to prescan the film for good events in order to save computer time. (2) A typical picture (Fig. 2) is fairly complex, from a pattern recognition point of view. This is partially because of some instrumental problems, but basically it is a consequence of SLAC's very poor duty cycle. This leads to a high background rate which complicates automatic recognition of events. It was immediately apparent that the programming which dealt with actually recognizing events should be split off and made into a separate package which could run in the so-called "scope partition" with a display scope for manual interaction.

B. Phase 1 Program

This program drives the film scanner, digitizing the frames of interest as determined from an input deck of scan cards, and writes an output tape for the Phase 2 event recognition program. The Phase 1 program runs in the "batch"

(278 K byte) partition and uses one tape drive, one disk pack, one display scope, and one parallel data adapter in addition to peripherals used by the Operating System.

The program starts by reading in a data deck, which contains updates to program parameters, and scan cards. It then pauses for any last-minute parameter entries from the display scope keyboard. We may note here that all communication with the program/film-scanner operator is handled via messages on the display scope and appropriate responses via the scope light-pen or keyboard. The operator only enters the film scanner room to put either the calibration pattern or film in position; all other operations are conducted from the display scope in the computer room.

Once the program has been initialized, the operator is requested to place the calibration pattern in the scanner. The calibration pattern is a 9×6 array of x-shaped marks whose centers have been accurately measured on a conventional measuring machine. When this request has been met, the operator then sets the threshold to obtain a suitable number of digitizings. The scanner is then calibrated, i.e., appropriate transformation coefficients are determined to relate distorted scanner coordinates to a true rectilinear film plane coordinate system. The operator is then requested to place the film in the scanner, following which he resets the digitizing threshold. This threshold setting is found to be fairly critical, and to vary from roll to roll.

Next the film must be centered in the platen. This is done by merely light-penning a certain fiducial mark on the displayed picture. The program then computes how much to move the film to bring this mark to a standard location such that the entire frame is just covered by the raster scan. Following this the operator can set x and y limits on the raster scan, by typing in various values and observing the resultant picture. In practice the values pre-stored in the program are satisfactory, so this step is usually bypassed.

The next step is to give the program the positions of ten fiducial marks on the film, as well as four registration marks which define the position of a binary data box. There are various ways of doing this, but in practice all that is necessary is to point the light pen at one fiducial mark; from this the program can compute the expected positions of all fiducials and registration marks, using pre-stored information on fiducial separations, etc. If nothing is known a priori then all fiducials must be light-penned.

The final step in the initialization procedure is to check that the film is at the correct starting frame member, corresponding to the first scan card. If not, the operator can advance or back up the film the appropriate number of frames.

Once the initialization procedure has been finished (which takes typically 5 to 10 minutes) the program enters the main processing cycle. In this cycle the scanning is buffered, that is, while the current frame is being processed the film is moved to the next frame and the next frame is scanned. The main steps in the standard processing cycle are outlined below:

(1) The data from the scanner is unpacked into a single array in the form 0yxxxx. . . 0yxx. . . Y denotes the scan-line number, and x the position along that line of the digitizings in order. The zero indicates that the next word is a y coordinate, not an x coordinate.

(2) The raw data are then fitted with straight-line segments where possible, using a subroutine which makes a single pass through the data. Digitizings are connected initially just on the basis of nearness. Once three or four hits have been connected, however, a straight line prediction is made. The connection continues until for some number of scan lines (typically one) no associations can

be made. The string (which in our jargon we call a "blob") is then terminated, and a straight line is least-squares-fitted to the digitizings. Blobs may also be terminated if two blobs approach each other; a mask is then created to prohibit stringing in the potential area of confusion. The stringing subroutine uses the high order bits in the x coordinates to store a backward threaded list. After the raw data has been exhausted, a clean-up pass is made on the output blobs. Segments which have a χ^2-fit worse than some threshold are "pruned" by having the farthest-deviating digitizing removed; the remainder are refit and the process repeated until either the fit is acceptable or the segment has vanished. The "blobbing" subroutine, although written in FORTRAN, can process about 4000 digitizings per second of 75 computer time. Typically on one frame 4000 digitizing are reduced to 400 segments.

(3) Using the blob arrays as input, we next find the fiducial marks. Each fiducial is a V-shaped mark, the position of the apex of which is presumably accurately known in real space. The image of a fiducial is only 200μ, or about 8 scan lines, high. The search procedure is fairly straight-forward; it uses as an expected position the place where the fiducial was last found.

(4) Next the data box is decoded. The data box consists of two rows of vertical bars, each about 200μ high on the film, which form a BCD representation (with parity bit) of 8 decimal digits. Thus nominally we have error detection but not correction. If no parity errors occurred, the decoded frame number should agree with the number on the scan card.

(5) The blobs and fiducial vertices are now transformed from scanner coordinates. The details of this are somewhat complicated. Basically, however, allowance is made on each frame for translation, rotation, and magnification, using information on the fiducial positions on the film. The non-linear distortion

effects are assumed to be constant; this is checked at the end of each run.

(6) The last step is to write the transformed blob and fiducial vertex data on the output tape.

In normal operation, no intervention is required on the part of the operator. An entry may be forced, however, if the program detects some abnormal condition. The list of possible error conditions is very long; it includes non-matching data boxes, fiducials missed more than a preset number of times, failure of scanner to respond to commands, etc. In all cases, the general procedure is the same; the program rings a bell, pauses, and displays the appropriate message on the scope. The operator responds by hitting any key on the scope keyboard. The program then enters the appropriate fix-up routine. Should the operator not know what to do, he can press the question-mark key; this will cause a page of instructions for this routine to be displayed. He then does whatever is necessary to get things running properly again.

If the program is cycling normally, but the operator wants to intervene anyway (to terminate the run, for example), he can force an entry by hitting any key on the keyboard. Processing will then be suspended at the end of the current event. He then has the ability, by typing in the appropriate word, to exercise any of some twenty options, many of which will branch him to various subroutines in the program. This equivalent of a large bank of sense switches is seldom exploited much in normal production, but it adds greatly to the flexibility of the program, and is very helpful in debugging.

The Phase 1 program consists of some 65 subroutines, all but 11 of which are written in FORTRAN IV. The full length of the program is 322 K bytes, of which 171 K bytes is in blank and labelled common. In execution, however, the program is overlaid in a fairly complex way, so that the total amount of core used is actually only 235 K bytes.

In steady normal processing, the Phase 1 program is completely scan-bound; it uses only 30% of the CPU cycles.

C. Phase 2

This program accepts the output tape from the Phase 1 program, and with considerable manual intervention produces events on an output tape in precisely the same format used by the conventional measuring system. The output, which consists mostly of film plane spark coordinate measurements, is then given to the "customer" for processing through his own spatial reconstruction and kinematics programs.

The Phase 2 program, which actually does very little computation, is designed to run in the "scope" (100 K byte) partition. Programs in this partition have priority on the CPU if they need it; they relinquish it to the "batch" partition by doing I/O, or more accurately, by issuing a WAIT macro-instruction. Since the Phase 2 program is waiting most of the time for the program operator to respond at the display scope, this program running in the "scope" partition degrades the "batch" partition very little ($< 10\%$).

The Phase 2 program uses one disk pack (for program storage), one display scope, and three tape drives (one for input, one for output, and one to store the printed output). We have found, in the case of the Phase 2 program, that it is more reliable for us to handle the listing of the printed output ourselves, rather than leave it up to the system spooling program.

The Phase 2 program basically tries to form an event of the type indicated on the scan card from the list of "blobs" passed to it by the Phase 1 program. If it is unable to do this in an unambiguous way, it will ask for manual intervention via the display scope. The operator then provides assistance and the program tries again. If the operator feels that the event is too difficult for him to untangle in a reasonable length of time, he can indicate this fact and the frame will be set aside for measurement on conventional measuring machines later.

The procedures the Phase 2 program uses to recognize events are not particularly apt; they were designed with the hope of making them experiment independent, which in this case is not a particularly wise decision. In each view of each chamber the program first tries to find track segments by gluing one or more blobs together; the basic criteria are that the resultant segment should form a reasonably straight line, and that it cover a reasonable fraction of the gaps in that chamber. It next tries to link one chamber to the next where appropriate. This is not completely trivial, since in the stereo view, at least, straight tracks in real space do not appear straight on the film, due to the optical folding used to image the chambers onto the film. Next it tries to pair tracks in the direct and stereo views. Since pairing by examining the detailed structure of each track segment (e.g. matching missing sparks) is unreliable, we use an alternative approach. If one imagines the connections of tracks in the various chambers in either view as being represented by a graphical tree, then we attempt to pair the direct and stereo views by looking for matching trees. Finally, we use the same sort of technique to connect tracks through the bending magnet (which, to a reasonable approximation bends images only in the direct view, not in the stereo view).

In practice, all this apparatus frequently fails to find the event unambiguously. When this happens, appropriate displays are given to the operator; he can then add or delete individual track segments, or indicate connections between segments, by appropriate use of the light pen. He is also given other controls for convenience; for example, he can magnify a certain area of the picture, turn displays on and off, etc.

The Phase 2 program consists of some 33 subroutines, almost all written in FORTRAN IV. Fully expanded, the program occupies 118 K bytes, of which 46 K bytes are used for blank and labelled common. In execution, the program is overlaid so that it occupies only 85 K bytes of memory.

III. EXPERIENCE TO DATE

For approximately two months we have been in a production mode. Not surprisingly we find that a fair effort (mostly in software) is required to circumvent difficulties which crop up. As an example, one set of problems which has caused a certain amount of annoyance consists of splices in the film, defective binary data boxes, multiple frames with the same frame number, and so on. Such problems have afflicted automatic data processing systems from the beginning, and presumably will forevermore.

A complete discussion of how our measurements compare with hand measurements would be very complicated. We have data from repeated hand measurements, repeated automatic measurements, and combinations of the two; interpretation of the various comparisons is not yet complete. We can make a couple of observations, however. Good hand measurements give the separation between pairs of fiducials constant, within a given roll, to between 6 and 13μ; comparable limits, for automatic measurements of the same fiducials, are 8 and 16μ. Comparisons of essentially corresponding sparks from two manual measurements have spreads of roughly 30μ (on the film) perpendicular to the track direction. Comparisons between hand and automatic measurements yield simular discrepancies. In addition to these spreads, there also appears to be systematic differences between manual and automatic measurements, of comparable magnitude.

By now we have accumulated some reasonable estimates of rates. For film which doesn't have anything seriously wrong with it that we don't already know about, the Phase 1 program can process about 250 frames per hour. Rates for the Phase 2 program depend somewhat on the film, but typically range from 60 to 100 per hour. Not all these events represent measurements delivered to the customer; events are omitted along the way for a variety of reasons. For film which we have recently been processing, the breakdown is roughly as follows

1%	rejected for technical reasons (dirty film, hardware errors, etc.)
2%	mis-handled due to known program errors
10%	skipped by the scope operator as being mistakes by the original scanner
12%	set aside as being too hard to do; to be measured later on conventional measuring machines
75%	have measurements output; about 12% of these have gross errors due to the scope operator.

At present only about 10% of the events we output are recognized automatically. Clearly one of our next efforts will be to increase this fraction.

IV. BUBBLE CHAMBER PLANS

With the effort on spark chamber work starting to taper off, we are now beginning to increase significantly our efforts on bubble chamber data analysis. We will build another film digitizer similar to the HB2. Among the differences will be

(1) a high speed (0 to 7 m/sec in 0.3 sec) film transport suitable for unsprocketed film

(2) a slightly larger (9") CRT

(3) better optics.

We will also continue efforts to make better displays. The IBM-supplied displays are only capable of displaying 1400 points, which is not adequate for our purposes. We have already built a display scope ourselves which can display 4000 points, and which has numerous accessory features not available on the IBM displays. We are also building a TV-type display driven from a disk; in this type of display the flicker rate is independent of the amount of data displayed.

Our general guidelines in designing the system have been (1) to do away with the "road-making" process used on many conventional FSD systems and (2) to try to reduce the reject rate which on other systems can get annoyingly high.

In satisfying the first requirement we plan on using the CERN Minimum Guidance (MG) program. This program finds the digitizings belonging to an event out of the entire frame using only the vertex of the event located with a precision of 100μ. CERN has very kindly supplied us with a copy of their program, and we have converted it to our computer and run it on both their data and on film digitized at SLAC.

We will try to attain the second objective (of reducing the reject rate) by making extensive use of manual intervention via display scopes. Based on our experiences we feel that often very little help is required to put a hard event "over the hump" so that it can be successfully processed.

Our system overall assumes that the film is prescanned as it is for a Spiral Reader i.e., for each event of interest there is a scan card indicating the frame number, the event type, a rough vertex location (to ~ 1% of the picture) and possibly some indicative remarks. This information will be fed to the program which runs the film digitizer. The digitizings in the approximate area of the vertex will then be given to a vertex-finding routine; if this routine is unable to find a vertex, it can be located manually on a display scope. The precise vertex, and all the digitizings, will then be given to the CERN MG program. The output of this will be edited and single-view checks made. If the checks fail, the MG output will be displayed; a scope operator will then edit the input digitizings so that on a second attempt the MG program will hopefully process the event correctly. For chambers which have single strip format we would also make a three-view geometry check. The whole system will eventually be designed to run asynchronously, with each piece of the program accumulating output data queues, in order to maximize the throughput rate. Our design goal is to have a system which will process 100 events/hour while using about 10% of the computer cycles.

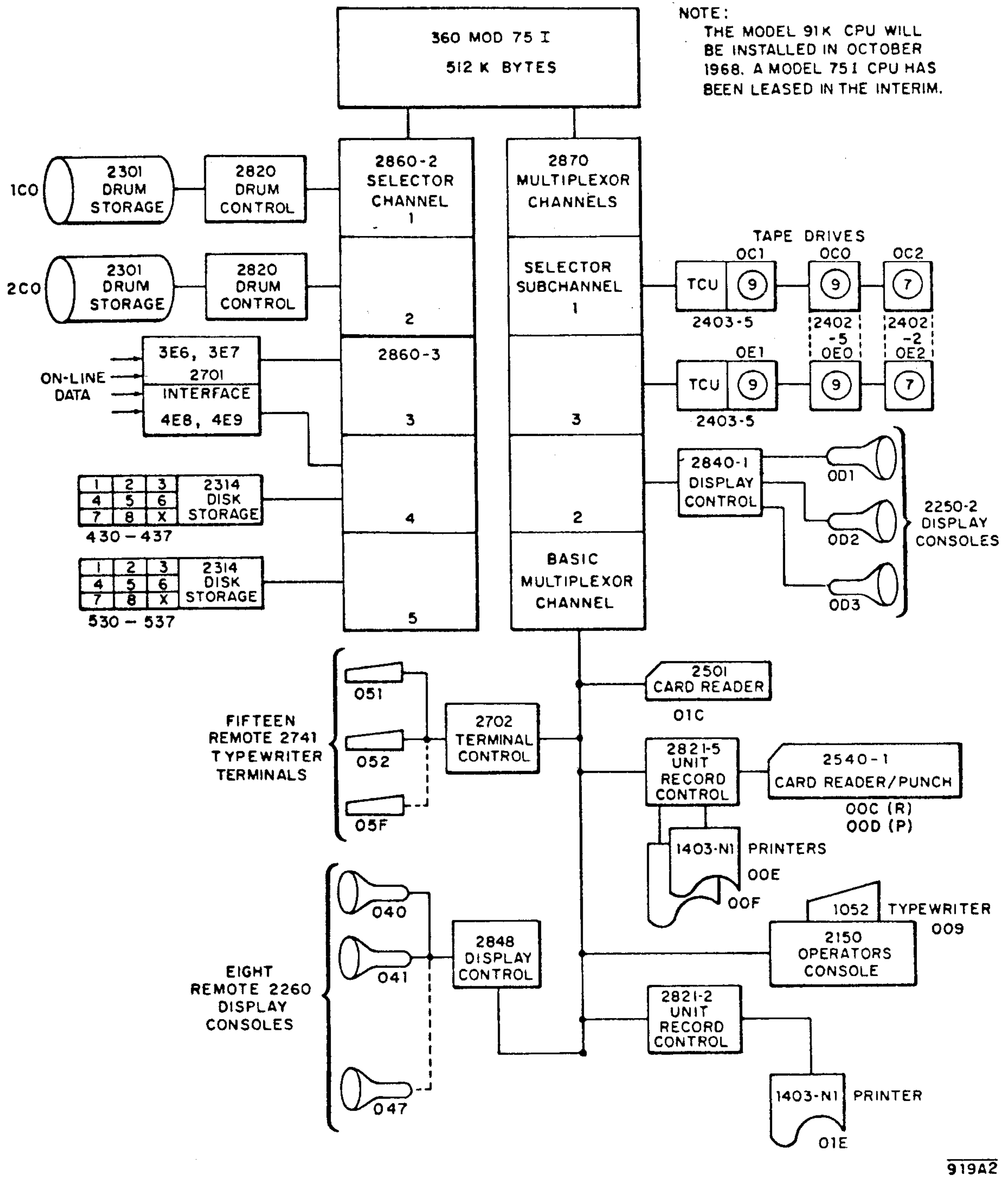

Fig. 1

Configuration of SLAC central computing facility.

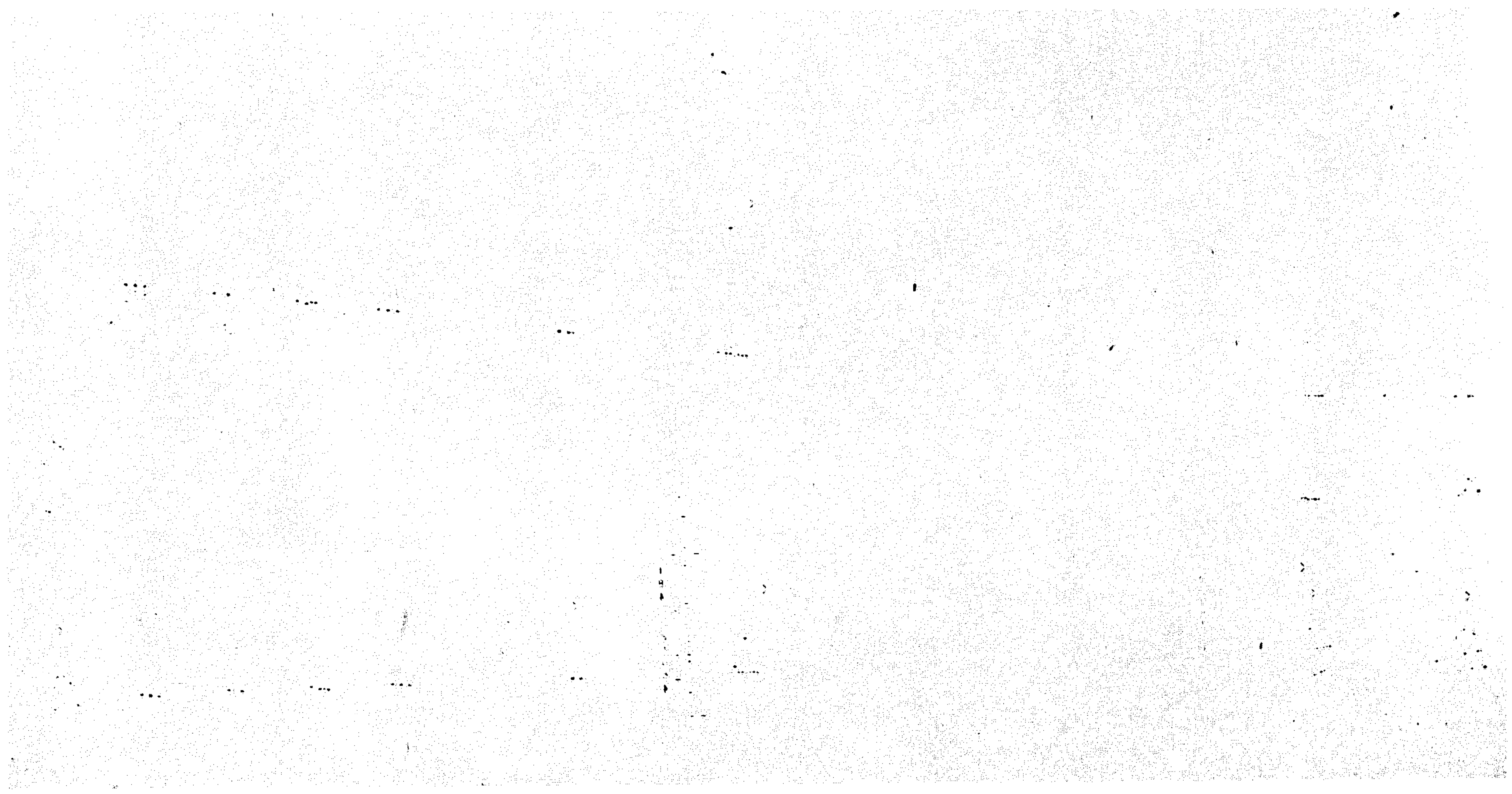

Fig. 2

2. Representative picture of μ-p spark chamber event.