The following is an outline of the current pattern recognition code. It lists the subroutines and a few key functions and gives a brief description of each.

I have addressed the code in two distinct sections. The first includes all routines prior to "Track_Assignment". The second includes "Track_Assignment" and all routines following.

The subroutines in the first section have been subject to critique from Art, Maher, and Carl to varying degrees. Major revisions have been suggested, discussed, and implemented. With the exception of "Clustering", this section has mostly stabilized for the moment. (Meaning that I will be focusing more on the second section for the time being.) The first section has been tested with Monte Carlo data.

There are three types of changes that I am making to the second section:

> First, I am making this section compatible with the changes implemented in the first section. For the most part this is trivial and much is already done.
>
> Second, I am making changes associated with a change in the logical flow. Some of these changes are a direct result of changes in the structures. I am taking advantage of opportunities the changes in structure provide for simpler and quicker processing. Other changes are a result of the feedback that is and will be coming in from those who are helping me to critique the code. These changes are not all trivial, and the amount of change required is not fully determined. However, we are on the way. Much of this section still requires testing with the recent and soon to be changes.
>
> Third, We are incorporating filtering into the process, rather than just appending it to the end. (This is actually a special case of a logic change. However, it is a major step and thus it is listed separately.) All routines prior to "Track_Assignment" are obviously completed without filtering. In addition, the subroutine, "Muon_Track", will be completed prior to filtering. "Find_Track" will determine starting parameters for the filtering and call the filtering routine. The starting parameters can be whatever the filtering routine needs: Helix parameters, hits, intersections, or some combination. The filtering routine will be used to complete the track within a window and to extend the track into overlap windows. The pattern recognition will control the looping of the windows in this context. Many of the details of the interface with the filtering depend on what filtering we choose and how we use it. However, much of the logic and code leading up to the interface are independent of that choice and are being implemented.

Pattern Recognition

June 13, 2000

Pattern_Recognition	Main control subroutine for Pattern Recognition
windows_clear	Clears window structures for new event
log_clear	Clears structures used for debugging purposes
Sort	Quick sort routine that sorts PC hits by time
Partition	Recursive portion of quick sort routine
Interchange_Sort	Interchange sort routine for short lists
Select_PCs	Groups PCs into time windows

insert (function)	Inserts PCs into window structures
analyze_window (function)	Determines characteristics relating to ability to analyze window based on PC data
Create_Intersections	Creates intersections from pairs of U and V PC planes
fill_int (function)	Inserts intersections into appropriate structures
Create_Windows	Calculates boundaries of windows and stores in window structure Windows are of three types: a) Containing hits unambiguously associated with a single group of "simultaneous" PC hits b) Containing hits ambiguously associated with multiple groups of PC hits (in the overlap region of the PC hits) c) Containing hits not associated with any PC hits Any combination of these always yields 2p + 1 windows where p is the number of groups of PC hits (The first and last window might be of 0 width.)
Assign_DCs	Assigns DC hits to windows
insert (function)	Inserts DCs into window structures
Clustering	Finds clusters of DC hits within a plane and creates cluster structures
Create_Intersection	Creates intersections from pairs of U and V DC planes
fill_int (function)	Inserts intersections into appropriate structures
Track_Assignment	Control subroutine that may eventually end up incorporated directly
Muon_Track	Assigns hits to a muon track
Muon_Intersection	Inserts hits from intersections into track structure
Double_Anchor	Recreates the track beginning at both the end and the target
Find_Track	Finds first guess parameters for fitting routine This routine is under major reconstruction
Correlate_Halves	Correlates halves of the detector by joining straight through tracks and relating other data of two-half events