Event-Driven, Asynchronous Control and Monitoring

by

Aaron Richard McCabe
B.S. Electrical Engineering, Massachusetts Institute of Technology (1997)

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of
Masters of Engineering in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology
June 1998

Signature of Author . . . . . .
Department of Electrical Engineering and Computer Science
May 15, 1998

Certified by . . . . . .
Chathan M. Cooke
Principal Research Engineer
Thesis Supervisor

Accepted by . . . . . .
Arthur C. Smith
Chairman, Committee on Graduate Students
Department of Electrical Engineering and Computer Science

"Peoples is peoples."
—The Chef from *The Muppets Take Manhattan*

Event Driven, Asynchronous Control and Monitoring

by

Aaron Richard McCabe
B.S. Electrical Engineering, Massachusetts Institute of Technology (1997)

Submitted to the Department of Electrical Engineering and Computer Science
on May 15, 1998 in Partial Fulfillment of the Requirements for the Degree of
Masters of Engineering in Electrical Engineering and Computer Science

ABSTRACT

This thesis presents and explores an atypical method of designing a modern (computer driven) digital monitoring and control system. There are two tenets which distinguish this method from others. The first is the realization that, from the perspective of a computer driving the system, the monitoring processes and control processes are very similar: they both require the computer to send a request to an external instrument, and then lie in wait for a response. The second is that there is a natural division between the user interface(s) and the instrument driver(s) that can be exploited to lighten the load on any one computer.

From these tenets derive the "concentrator" and the "observer". A concentrator is a conceptual entity that is in charge of driving external instruments (through autonomous processes) to observe important measurements or control important settings within the system, convert them to relevant values, and then buffer the information. An observer is the graphicals, the user interface, and in charge of any special calculations. It is from here that the buffered values are interpreted into expert decisions, either by an algorithm or an operator. Concentrators and observers are, by definition, independent and can therefore be run on separate machines (arbitrary numbers of each) communicating over a network. The minimal configuration, however, is one concentrator and one observer; the concentrator cannot run without first having a parameter file downloaded from an observer, and the observer would be useless without a concentrator.

The beauty of this system comes from its flexibility and expandability. A new process can be added to a concentrator within minutes, and likewise its values' interpretation on the observer. These processes need not rely on timed loops to generate actions within the system. Every process is event-driven, generating an event (and triggering a process to run or data to flow) based on any imaginable occurrence; a mouse-click, or an external trigger, or a separate timed loop if required. Furthermore, there are no bottlenecks in the system due to one parameter having a slower loop than others; all processes are independent and therefore asynchronous, reporting data and setting values when needed, not just when available.

Thesis Supervisor: Chathan M. Cooke
Title: Principal Research Engineer

Acknowledgments

- To Mom, Dad, and James: I don't know what I'd do without you.
- To Dr. Cooke: Thank you for giving me the freedom to explore when I needed it, and direction when I didn't.
- To Andy Loening: 8.01, 8.02, 18.04, 5.12, 6.001, 6.002, 6.003, 6.004, 6.041, 6.011, 6.012, 6.013, 6.121, 6.023, 6.302, 6.241, and 6.344. Thanks for putting up with my silly antics and half-brained witticisms. Oh, and for section 2.3.
- To Ether: Here's to all-you-can-eat wings and Chinese food, passing out at 8PM on Fridays, and punchbuggy's for all...or at least you and me. You kept me sane this year.
- To Jawa: Hey ... if I can do this computer stuff, so can you!
- To Jessica: It's been a great year — thanks for your patience and keeping me on my toes.
- To Amy: Imagine the luck, earning such an amazing friend. All from acting like a moron at a party. Keep studying your Minnesotan!
- To (Dr!) Robert Lyons: My LaTeX and thesis-in-general mentor, thanks for putting up with my pacing.
- To Myself: Five years of work pursuing my ideals and searching for "Education", visiting home and learning the difference between permanent and temporary, and now returning there, wondering what to do next. And yet, I wouldn't trade a day of it in for anything.
- and finally, in memory of Phil: I'll never know why, but at least I was privileged enough to know who ...

So we weep for a person who lived at a great cost.
Yet we barely knew his powers till we sensed what we had lost.

—Dar Williams, Mark Rothko Song

Contents

List of Figures

Chapter 1

Structural Approach

1.1 Overview

Computer driven control systems can be both a blessing and a curse. While there are numerous advantages to digital systems, and a wealth of digital signal processing techniques, there are just as many drawbacks. In fact, the very thing that causes it's advantage is also the source of it's trouble: the ability to remember state. In short, digital control, particularly that which involves the use of computers, can easily generate too much information, and burden the system. Hence, if computers are to be used in a control system, it is essential to develop powerful and flexible tools for data management. One such method has been described by Bauman, et al. [2], and is expounded upon here.

The concept is two-fold, but simple: separate the user-interface from the control, and optimize the control side for efficiency while maintaining the flexibility and expandability of a digital system.

1.2 Process-Based Control and Monitoring

To a computer, both control and monitoring tasks have a lot in common. They both involve the computer issuing a command to an external system, and both expect a response. For instance, a control task will send a request to an outside system (i.e. to set a voltage to a specific value), and then receive a validation that the voltage was indeed set. A monitoring task could send a request to an outside system to read a voltage or capture a waveform, and then it receives this data as a response.

These tasks are independent, as well. Provided unlimited resources, it doesn't matter if there is just one task monitoring, or whether there are thousands reading and controlling. These tasks, then, should each be designed to operate independently and asynchronously. A task, control or monitoring, that behaves as such will be referred to as a "process" in this thesis. These processes should be so completely independent that an arbitrary number of than can reside on an arbitrary number of computers,

without conflict. Of course, the processes need to be managed to a certain extent — even if only for error handling purposes — and the data that they generate (responses) should be consolidated into one (easy to search) database on each computer. As will be seen in section 1.3, this management and consolidation will occur in what is called a concentrator.

Computations, charts, other visuals, and the user's requests for data, then, are left to the observer. The observer is described in section 1.4

1.3 The Concentrator

1.3.1 Concepts of the Concentrator

The concentrator serves as a translator between data which is unintelligible to an operator, termed raw data (such as the voltage being read from an RTD, or unscaled values read from a scanner) and data which is intelligible, called values (such as temperature or dose). The distinction is necessary; we, as operators, want to see values which are relevant to our current experiment or operation, whereas the processes doing the controlling want to deal in values that are practical, such as voltages and currents. Values may involve more than just scaling a raw data, they could involve other values, and all involve a time stamp and label. It is just as important to know the name of the value and what time it was created as to know the contents of that value. Furthermore, the concentrator is a (temporary) reservoir for incoming data — it will hold onto values for an assignable predesignated amount of time, so that all the observers (see section 1.4) that need the data can obtain it. Each concentrator, before it can run, must be configured with a parameter file. This parameter file, downloaded from an observer, will define (among other things, see section 1.3.2) which processes that particular concentrator is allowed to run, the "universal" conversion factors that allow raw data to be transformed into values, and how many records the buffer can hold for each value.

Figure 1.1 exhibits a typical concentrator. This concentrator has been configured to access four processes, one for reading temperature, one to read a waveform, another to set digital logic, and the last sets the reference voltage for an external control system. These four processes would have been loaded according to the parameter file, which would also have defined some important constants. For example, in the first process the parameter file would have defined the relationship between the voltage read from the RTD and the actual temperature, in the second it would have set the sample rate and length, in the third it could have defined the baud rate that the particular logic system uses, and in the fourth it set the programmable gain of the control system to keep the feedback stable. The appropriate processes read raw data, timestamp it, transform it into a value, and then place it in the buffer. The buffer is a database that is referenced by the value's name. This information, then, can be passed along according to the requests by the observer.

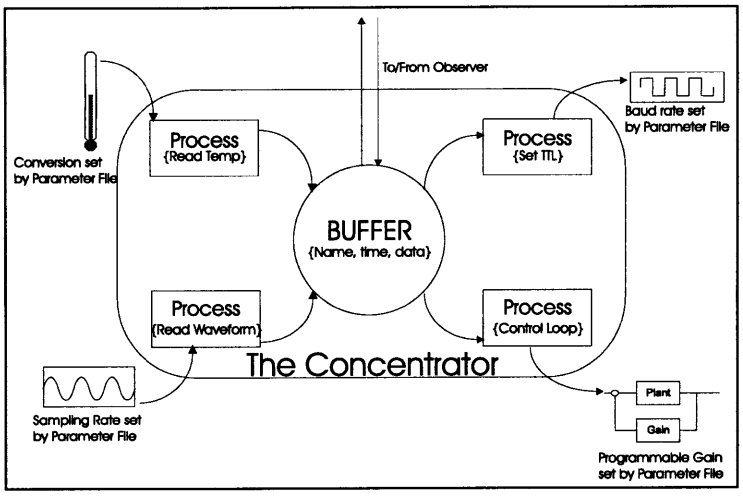

Figure 1.1: A typical concentrator.

1.3.2 Details of the Concentrator

This section provides details about the concentrator that may be confusing without first examining a working version. Please refer to the sourcecode for a concentrator (an example of which is included in appendix A) while reading this section.

When the concentrator begins operation it starts to listen on one specific TCP/IP port (arbitrarily chosen to be port 4321), for a connection from an observer. In order to support many connections as quickly as possible it uses multiple buffers to support them. There is a very fast loop waiting for connections — when a connection attempt is made, it buffers that connection information (remote address, port, etc.) and then resumes waiting for connections. Another loop operates only when there is buffered connection information, and reads the data from the client as a raw string and buffers that string, then goes back to waiting for more buffered connections. The third loop is where the processing of the received data occurs. This double-buffered system allows the concentrator to be listening for new potential connections at the fastest rate possible, to help avoid refused connection errors. Likewise, it can handle long processing times for the received text without causing a bottleneck in the portion of the program that actually receives the text.

Before it is fully functional, the concentrator must receive a parameter file from an observer. This parameter file contains all the instructions that the concentrator needs to operate on a specific computer. There are four commands that the parameter file handler can interpret:

1. The parameter file determines which processes will be running on this particular concentrator, and the concentrator then loads them into the computer's memory.

2. The parameter file sets "global" values, such as the scaling factors that relate the raw data to values, or other metrics that won't change during the operation of

the concentrator. These are numbers that are sent to the processes.

3. The parameter file sets the maximum number of values that the buffer will keep in storage for a particular measurement. This number can be infinite, and can be different for each measurement. This is to prevent the buffer from becoming too unwieldy, and slowing the concentrator down. It is important to note that one value can be a single number (such as the current temperature), or it can be thousands of numbers (such as a captured waveform), each only has one timestamp, and is considered to be one value. It is, therefore, essential that the buffer has different maximum lengths for different measurements.

4. The parameter file can also instruct the concentrator to send a one-time raw text string to any of the instruments of which it is in charge. This is to allow for expansion, in case new processes are added that don't have robust enough control over their respective instrument. This message, for instance, could be sent over the GPIB, or a serial bus, or TCP/IP.

The parameter file is a text file that can be edited in any standard text editor. It uses a very specific scripting language. The text in figure 1.2 is an example parameter file. Note that the beginning and end are delimited with a double colon, the beginning of each line is a backslash, the end of each line is a forward slash, a line is commented out with a percent sign, the command name is separated from it's arguments with a double semi-colon, and each argument is separated with a single semi-colon. This parameter file starts a number of processes, including the localstartup.proc process which is unique to each concentrator, initializing the instruments that are specific to it. It has a commented out GPIB send (the command P0X to the instrument at address 21), sets a few buffer lengths, and defines some scaling factors.

As the processes run, they fill the buffer with data. There are two modes in which they can update the buffer. The first is termed "poke-overwrite" mode, where the process actually overwrites the entire buffer with it's current update. This is one way to force the buffer to only hold one datapoint. The other is "poke-append", which fills the buffer. The buffer is not First In First Out, as may be expected. Since everything is time-stamped, the buffer sends data according to times requested, and most often will be sending the most recent data point. The types of requests for data that are allowed will be discussed in section 1.4.2.

Although it is implied that all processes are independent, there are times when this may not be so. If it is necessary to use the same voltage scanner to observe data which will eventually become two entirely different values (i.e. the four voltage reading consisting of i_{nw}, i_{ne}, i_{sw}, i_{se} and the one voltage reading, the terminal voltage, are both read using the same scanner, see section 2.3) then two competing processes may actually need to draw from the same resource. To avoid any potential conflicts between these two processes, a "token-ring" prioritizing scheme is used. A virtual "token" is passed to a process, and it is allowed to use the shared resource, and as long as that process has the token, no other process may use that resource. When the process

```
::
\STARTPROC;;localstartup.proc;/
\%GPIBSEND;;21;P0X;/
\STARTPROC;;Scan Channels 1-4.proc;/
\STARTPROC;;Terminal Voltage.proc;/
\STARTPROC;;Total Current.proc;/
\STARTPROC;;Partial Current.proc;/
\STARTPROC;;Set Current #1.proc;/
\STARTPROC;;Set Current #2.proc;/
\SETMAXLENGTH;;Channels 1-4;20;/
\SETMAXLENGTH;;Terminal Voltage;20;/
\SETMAXLENGTH;;Total Current;20;/
\SETMAXLENGTH;;Partial Current;20;/
\SETVAL;;Channel 1 Scaling Factor;10;/
\SETVAL;;Channel 2 Scaling Factor;10;/
\SETVAL;;Channel 3 Scaling Factor;10;/
\SETVAL;;Channel 4 Scaling Factor;10;/
\SETVAL;;Channel 5 Scaling Factor;-5.32;/
\SETVAL;;Channel 6 Scaling Factor;100;/
\SETVAL;;Channel 7 Scaling Factor;100;/
\SETVAL;;Channel 8 Scaling Factor;1;/
::
```

Figure 1.2: A typical parameter file.

completes its task, it passes the token along to the next process, and so forth. In this way no two processes sharing resources will override one another.

1.4 The Observer

1.4.1 Concepts of The Observer

As the name implies, one function of the observer is to serve as the visuals that relate values to an educated operator. There are a few important things to realize about observers, namely:

1. Observers, like concentrators and their processes, are independent and autonomous. A network of observers can run without interfering with one another.

2. An observer is both active and passive (from the perspective of the operator). It can both actively send commands to the concentrator (those commands issued by the operator) and passively, or automatically, send commands to the concentrator and receive data back from the concentrator.

3. An observer has no access to raw data, that raw data never leaves a process. It is therefore essential that the values that the concentrator creates are relevant to the observer.

Observers can do a lot more than simply displaying current data, however. This is the power of the time-stamped data in the concentrators. It is a trivial task for an observer to integrate data over time, or calculate rates. It is just as simple to use multiple values (even from multiple concentrators) as it is to use one.

1.4.2 Details of the Observer

Like section 1.3.2, this section provides details about the observer that may be confusing without first examining a working version. Please refer to the sourcecode for an observer (an example of which is included in appendix A) while reading this section.

The way that the observer handles network connections is nearly identical to the concentrator. The only difference is that the observer is listening on a different TCP/IP port (arbitrarily chosen to be 1234). This was necessary to allow an observer and a concentrator to run on the same machine at the same time, which may be useful to check out some local operations. Otherwise, the observer uses the same "double buffered" scheme for incoming connections as the concentrator.

The functionality, however, is very different. There are two possible messages that the observer can receive from a concentrator. It can receive error messages (perhaps pointing out mistakes in an attempted parameter file), and it can receive data. There are a number of messages it can send to the concentrator, and these are detailed below.

- As mentioned before, the observer is responsible for downloading the parameter file to initialize a concentrator.
- The observer can request a value to be sent immediately. This command, when issued to the concentrator, polls the buffer for a particular value name, and sends it back immediately to the IP address from where it was requested.
- The observer can request for a value to be updated in an event-driven fashion. When the concentrator receives this command it sets a flag in the buffer to send this data to the specified IP address(es) whenever that value is updated in the buffer.
- For each of these requests, data may be asked for delivery in three different ways.
    1. "Peek-one" mode delivers the most recent value to enter the buffer for that value-name.
    2. "Peek-all" mode delivers all of the values in the buffer for that value-name.
    3. "Peek-since" mode delivers all of the values in the buffer for that value-name that occured since a specified time, sent with the request for data.
- The observer can also pass along commands along to control oriented processes. For instance you could tell the process "set current #1.proc" (a la figure 1.2) to set a current to 15 mA. The data-processing loop of the concentrator has very

little to do with this, when it receives the command to pass a value to a process, it passes the remaining text statement along unaltered to the process.

- The observer can halt and restart a process that has been loaded into a concentrator by a parameter file.
- Finally, observers are responsible for the error handling. The concentrator is not intended to have a display, so all errors are reported to the observers. In general it would be difficult to determine which observer to send a particular error to, but fortunately most errors are caused by inappropriate commands from an observer (i.e. a parameter file with a syntax error) and are therefore returned to that observer.

As of now, there remains one unsolved problem with the observers. It is unclear how the observers are to recognize what concentrators are online, and which concentrators are responsible for what values. There are many solutions to this problem, but currently the observer must have a local file that tells it what values are where — a solution that must be changed for better functionality. The best solution that the author can think of is to have the observers keep track of what processes have been configured on which concentrator (something they already do, to know what processes can be halted and restarted) and then as a new concentrator is configured, perhaps by another observer, that list be updated on each observer. Whether updating that list implies observer to observer communication, or if that master list should be sent through a concentrator, is yet to be determined. It seems to be the job of a new concentrator — concentrate the processes running on each concentrator and what values they eventually create.

1.5 Combined Relationship

Although the details are highlighted in the previous few sections, the relationship between the concentrator and the observer is the key element in this data management scheme, and will be restated here for robustness. This relationship is probably best described with diagrams. A generalized picture of the relationship between one observer and a concentrator is exhibited in figure 1.3. The observer is the "front end" to the control and monitoring system. It is from here that the operator can view relevant values in the system, and can evaluate equations that use multiple values. It is also from here that the operator can send instructions to the instruments operating the machine. A concentrator collects and sends values based on its set of instructions.

As can be seen from figure 1.4, every concentrator and observer are completely autonomous. There can be an arbitrary number of concentrators and observers in a system and, within limits, they will not conflict with one another.

This type of control system is completely asynchronous; all processes are autonomous and may report data at any time. Thus, slower processes do not affect faster processes. The problem of the slowest time-constant dominating is altogether avoided. Likewise, this control system is completely event-driven. A user clicking a

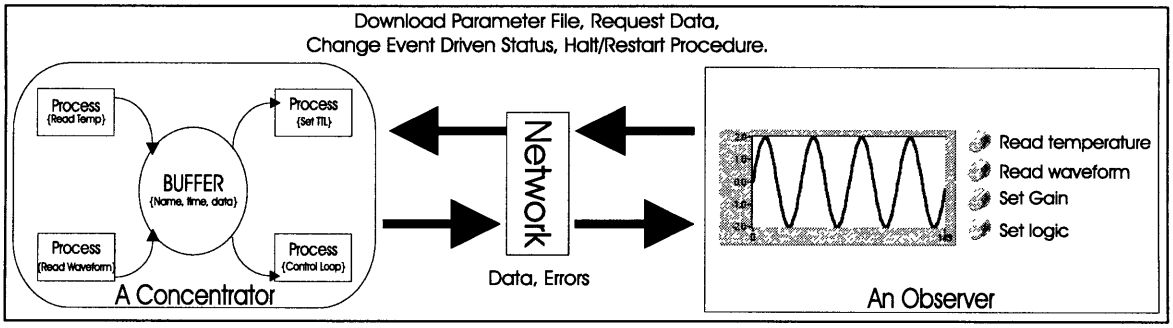

Figure 1.3: The communication between a concentrator and an observer.

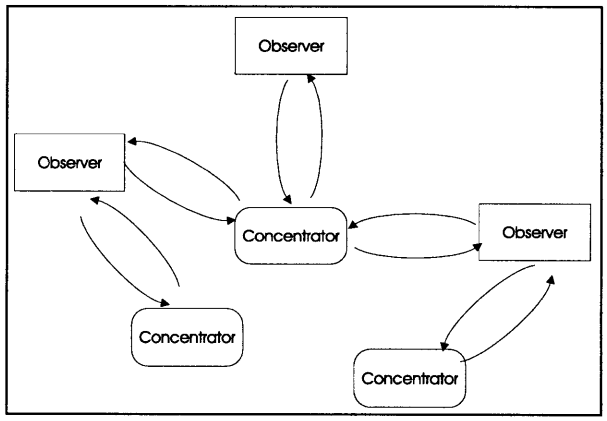

Figure 1.4: A network of concentrators and observers.

button on the observer generates an event to send instantaneous values, in the same way that a value that has been designated as event-driven is passed. Timed loops are certainly not necessary any longer, but if needed the timer for the loop becomes a process, generating a timed event to another process within its concentrator, another concentrator altogether, or even to an observer.

Chapter 2

Application: Hardware and Modeling

2.1 An Introduction to the Van de Graaff Generator

A Van de Graaff generator uses a corona source, powered by a high voltage power supply, as its source of electrons. (Please see Figure 2.1 for a sketch of the important elements of a Van de Graaff generator). These electrons ionize the air in the vicinity of the corona points so that the charge is "sprayed" onto a rapidly moving conveyor belt (the charge belt). This belt carries the charge up to an isolated high voltage terminal that is in a pressurized environment (SF_6 gas) that helps prevent it from breaking down the gas and "sparking". This belt introduces an inherent delay in the system that is the height of the belt divided by its velocity. There are practical limitations on both the speed of the belt and the electrons sprayed on: if the belt goes too fast there are diminishing returns due to electrons being lost to the environment, and if too much voltage is applied to the belt it can break or spark.

The high voltage terminal has a natural capacitance with ground. Furthermore, it has a 10^{11} Ω resistor running to ground. The electrons are coerced off of this terminal by means of a filament. The current flowing through the filament sets its temperature, and this serves as a coarse-adjust for electron flow: the hotter the element the more amperage. The filament current is adjusted via a variac that is controlled from below by a rod and motor setup. The fine-adjust for electron flow, then, comes from the bias voltage of the filament. This can be controlled to precisely define the electron beam. Next, the electrons are accelerated in a vacuum tube, and directed and focused by means of the field created by magnets at the exit side of the acceleration tube. There are three magnets all together, a solenoidal focusing magnet that keeps the beam reasonably columnated, a north-south magnet and an west-east magnet, which have the effect of pulling the beam in their respective directions, to aim it. The acceleration tube is lined with glass rings, to have the total voltage drop be realized as the sum of

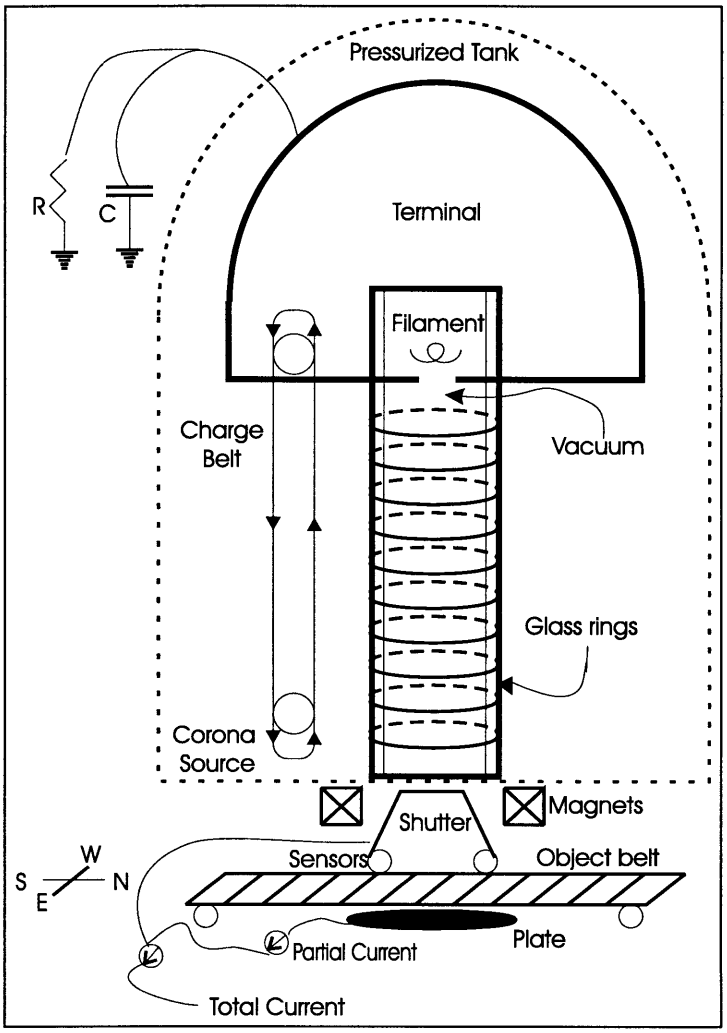

Figure 2.1: The important elements of the Van de Graaff Generator

small voltage drops from ring to ring. The electrons then exit the tube and bombard an object below, which is being slid underneath the tube by means of a motor driven belt (the object belt). The beam can be prevented from leaving the tube by closing a shutter. This allows the beam to be measured and evaluated without irradiating below. The spatial distribution of the electron beam can be determined by monitoring the current at the four corners (not affixed to the moving belt) of the area that is being irradiated. (See figure 2.2 for a birds-eye view of the Van de Graaff generator, looking down onto the object belt.) If the current, the number of electrons leaving through the four designated exit slits per second, is relatively equal at these four locations, the field is approximately uniform. Below the object-belt is a plate that can be used for measuring the "partial current" of the electron beam, the current with which the object actually would be bombarded. The "total current," then, is the partial current added to any leakage current, measured off of the shutter and the final tube. As an

object slides through the charge area, optical sensors are able to relate its position to an observer.

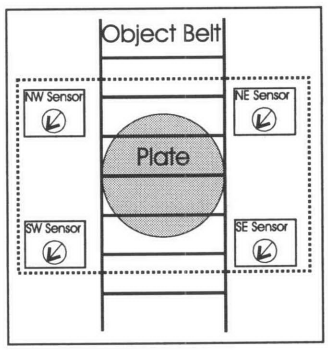

Figure 2.2: View looking down onto the object belt from the shutter.

2.2 Overvoltage Protection

2.2.1 Problem Statement

The required measurements of this system are small currents that are read as a voltages over a known (large) resistor. For example, all four sensor currents, the partial current, the total current, and even the terminal voltage, are all read as voltages by various means. Due to the excessive charges that are present in this system (the terminal voltage is typically measured in megavolts) it is necessary to protect all measurement equipment from possible spikes. It is necessary, therefore, to design a Voltage Protection Circuit that possesses the following traits:

1. The output voltage needs to be read over a large resistor, to account for the typically small input currents.

2. Any incoming spike should be sufficiently attenuated when viewed at the output. (By a factor of ten or greater)

3. The spike should be attenuated at all nodes of the circuit, not just a differential output — to prevent any probe from seeing the spike.

4. There should be a delay between the peak of the spike at the input and the peak of the spike at the output that is 1 ms or greater.

5. The output should settle within 100 ms.

One possible implementation of a protection scheme is presented in the following section. It is then developed into a system that satisfies all of the above constraints.

2.2.2 Basic Voltage Protection Circuit

Interpreting the Goals

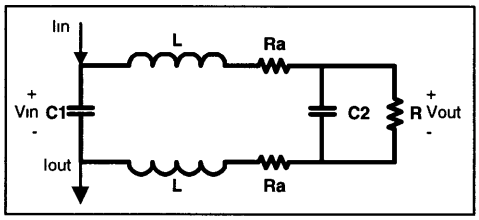

Figure 2.3: A Voltage Protection Circuit

Consider the circuit layout exhibited in figure 2.3. A quick examination shows that on the most theoretical level, this circuit's DC response is just a 10kΩ resistor, and as the frequency of the input approaches infinity, the circuit appears to be a short from the node at I_{in} to the node at I_{out}. In essence, the voltage at the output should see very little of the spike. The circuit is symmetrical. Since we are unsure of the polarity of the incoming spike, it is best to design it this way so that the spike can pull the voltages in either direction without affecting any one node. Furthermore, analysis on the circuit becomes easier, as a half-circuit technique can be employed. In general, the attenuation of the spike is attained by the relationship between C_1, on the left side, and C_2 and R, on the right side. That is, for a given current, the smaller capacitor (presumably C_1) will drop a larger voltage (according to $V = I/sC$) than the larger capacitor. The resistor, R, is present to dissipate some of the energy as it flows through the inductor and into C_2. Furthermore, R is the resistor over which we read our output. The inductors are present to delay the transfer of energy from C_1 to C_2. R_a is used to prevent any loop of the circuit (i.e. the inner loop) from being undamped, and potentially ringing. R_a shouldn't be large enough to be a significant percent of R. From this brief description, we can surmise that $C_1 > C_2$, $R_a \ll R$, and L is large enough to cause the appropriate delay.

Using the terminology presented in this circuit, the design goals can be restated as follows:

1. R $\geq$ 10 kΩ
2. $V_{out,max}/V_{in,max} \leq 0.1$
3. The voltage of any node should be confined to something less than the maximum voltage of the instrument measuring that node.
4. The time-to-peak (t_p) of the impulse response of V_{out} should be larger than 1 ms.
5. The output should be within 2% of its final value within 100 ms.

A mathematical examination follows.

Analysis of the Voltage Protection Circuit

In an attempt to quantify the previous circuit, the following transfer functions were found. During this analysis, it was assumed that the negative side of V_{in} was ground, as this corresponded to the setup used when attaining the experimental results. The response of the circuit to a voltage over C_1 is:

$$\frac{V_{out}}{V_{in}} = \frac{R}{2RLC_2s^2 + (2L + 2R_aRC_2)s + 2R_aR}$$

This leads directly to the transfer function from I_{in} to V_{in}:

$$\frac{V_{in}}{I_{in}} = \frac{2RLC_2s^2 + (2L + 2R_aRC_2)s + 2R_aR}{2RLC_1C_2s^3 + (2LC_1 + 2R_aRC_1C_2)s^2 + (2R_aC_1 + RC_1 + RC_2)s + 1}$$

Which, in turn, leads us to the transfer function from I_{in} to V_{out}:

$$\frac{V_{out}}{I_{in}} = \frac{R}{2RLC_1C_2s^3 + (2LC_1 + 2R_aRC_1C_2)s^2 + (2R_aC_1 + RC_1 + RC_2)s + 1}$$

Again, the above is true if the negative side of V_{in} is tied to ground. Using the above equations, a MATLAB simulation was employed to develop (realistic) circuit parameters that meet the above requirements. By choosing $C_1 = .15$ μF, $L = .001$ mH, $R_a = 1$ kΩ, $C_2 = 1$ μF, and $R = 10$ kΩ, the following results are conjectured by the model. When both V_{in} and V_{out} are measured in response to an identical I_{in}, there will be a 8.4:1 ratio between V_{in} and V_{out}, there will be a delay of 1 ms until the peak of V_{out}, and V_{out} will be within 2% of its final value after .05 seconds.

Results

The circuit was constructed and given an input current via a charged capacitor, and output voltage was recorded on a digital oscilloscope, and transferred to a computer. Figure 2.4 shows the entire response by using an appropriately long time-frame.

Note that the response is at its final value by 100 ms and that the input to output ratio is approximately 10:1. Figure 2.5 shows the same response, only zoomed in to see the V_{out} rise to its maximum value.

Note that there is a 1 ms time-to-peak in the output. Figures 2.4 and 2.5 imply that the model has accurately predicted the response of the circuit. In fact, after feeding the measured input voltage into the model, a direct comparison between the measured and the predicted response can be made. As can be seen, the two plots in figure 2.6 are nearly identical. This explains the accuracy with which the model anticipated the response goals.

There is, however, one problem with this design. It does not satisfy criterion number 3 of the design goals. To see this, please examine figure 2.11. The top chart in this figure clearly exhibits an artifact of the current spike that can be seen when referencing a node to ground instead of differentially. The bottom graph plots the response of the circuit that solves this problem, which is also developed in section 2.2.3.

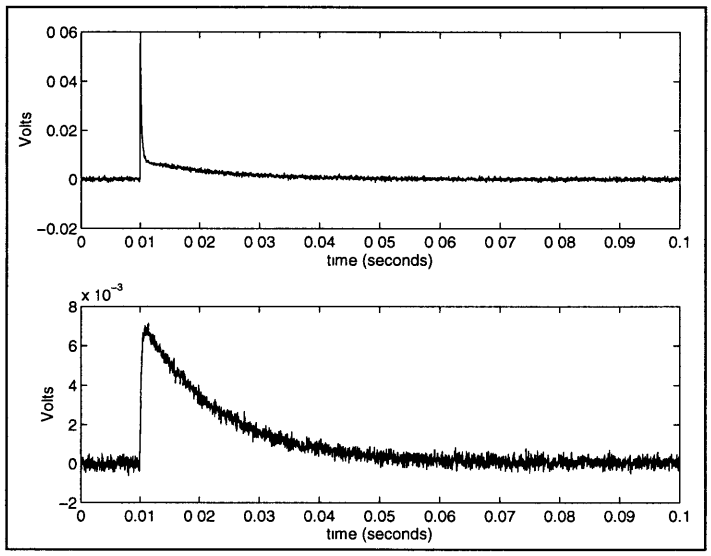

Figure 2.4: The long-term response of the Voltage Protection Circuit to a spark in input current.

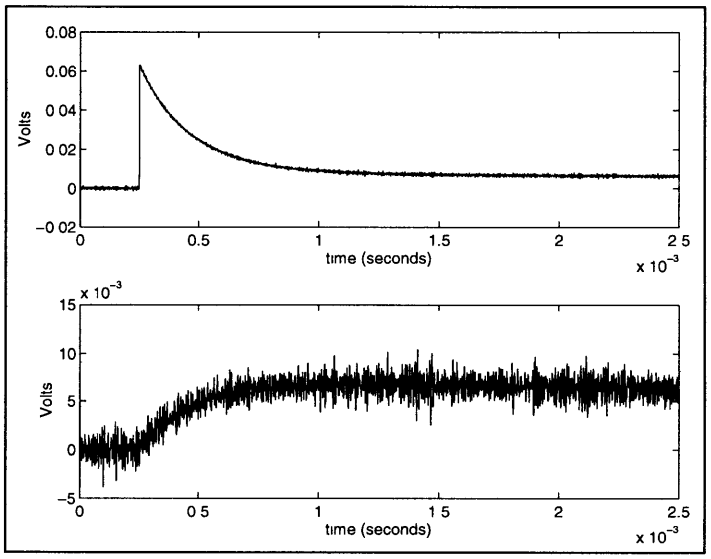

Figure 2.5: The short-term response of the Voltage Protection Circuit to a spark in input current.

2.2.3 An Improved Voltage Protection Circuit

A natural solution to this problem is to place capacitors (C_3) between the positive side of V_{out} and ground, and between the negative side of V_{out} and ground. The capacitor will tend to shield against a spike by "integrating" it out over time. A capacitor value of 1 μF is chosen to keep the RC time constant ($RC_3 = 10$ ms, well below our settling

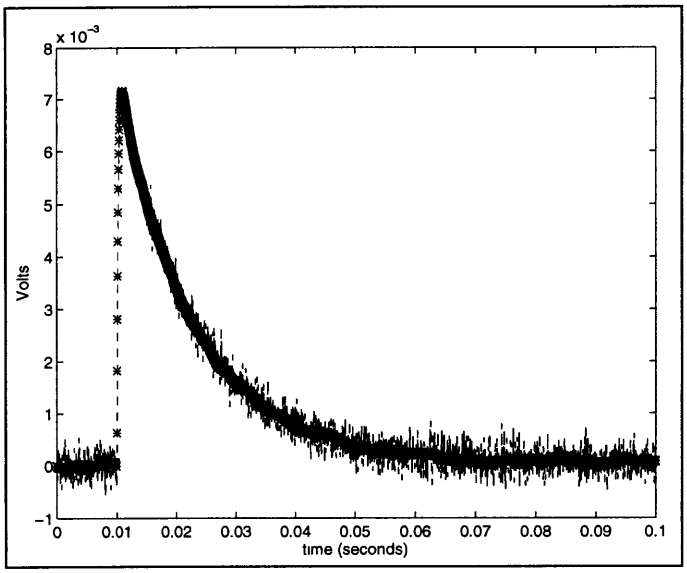

Figure 2.6: Predicted and actual responses of the Voltage Protection Circuit

time requirement of 100 ms) manageable. Figure 2.7 shows this improved circuit, with parameter names replaced with their respective values in this implementation.

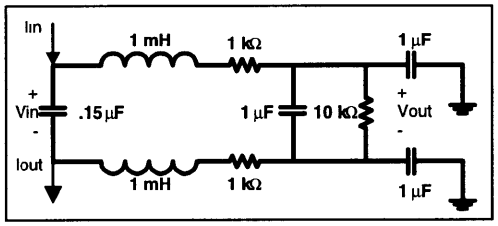

Figure 2.7: An improved voltage protection circuit, with parameter values.

Analysis of the Improved Voltage Protection Circuit

The math for this circuit is considerably more in depth than the previous circuit. As in the last case, we assume that the negative side of both C_3's is grounded to the same ground as the node that provides I_{out}, as this was the experimental setup used for verification. Unfortunately, using this setup removes symmetry from the circuit, and with it the easy half-circuit analytic method. The transfer function from V_{in} to V_{out} is relatively easy to find, but the other two transfer functions (I_{in} to V_{in} and I_{in} to V_{out}) are much more algebra-intensive. To solve them, the symbolic mathematics package Maple was used. The results were used in the model (see section 2.2.3, but are too long to publish in closed form here. (The denominator is fifth order.)

From V_{in} to V_{out} we have:

$$\frac{V_{out}}{V_{in}} = \frac{R}{(2LRC_2)s^2 + (2L + 2R_aRC_2 + R_aRC_3)s + 2R_a + R}$$

Please note that the negative side of V_{in} is grounded in the above equation, due to the experimental setup. When used in its final application this circuit will be placed in series with the rest of the system.

Results

As before, the circuit was built and given a spike in I_{in} via a charged capacitor. The output (V_{out}) was recorded on the digital scope and transferred to a computer. The circuit's response is shown here, in a similar fashion as before. Figure 2.8 displays the entire response, by using a large time-step. Figure 2.9 is the same circuit (different input, as can be noted by comparison), only with a smaller time step to see the rise time.

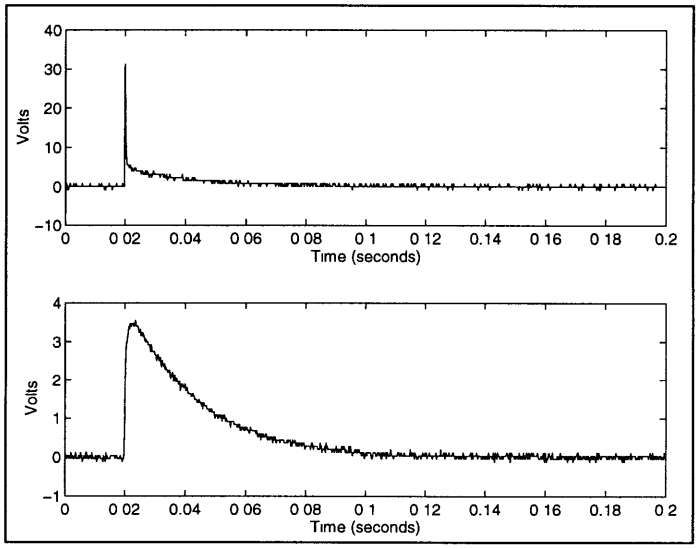

Figure 2.8: The long-term response of the Improved Voltage Protection Circuit to a spark in input current.

The results are similar: this circuit also has an approximate 1 ms time-to-peak in the output (perhaps longer!), the response has settled well before 100ms, and there is a ratio of approximately 10:1 from the input to the output. Furthermore, the model used to predict the response of this circuit is very accurate. It is almost difficult to see the difference between the theoretical and measured results in figure 2.10. The thin line (theoretical results) barely deviates from the noisier measured results.

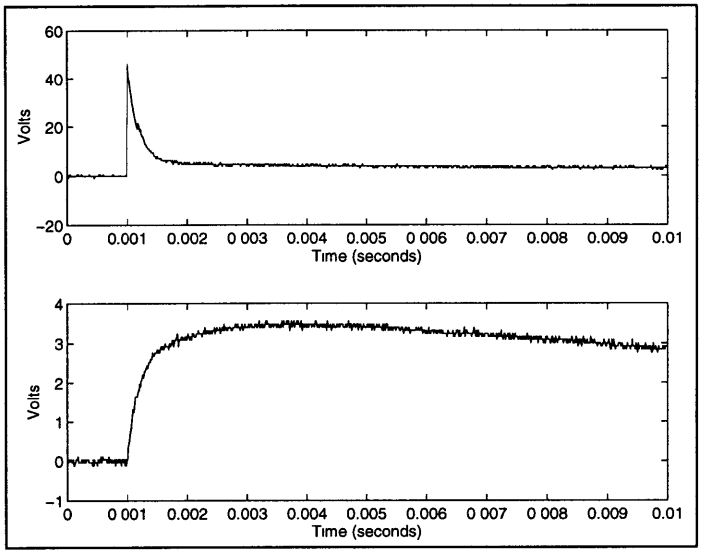

Figure 2.9: The short-term response of the Improved Voltage Protection Circuit to a spark in input current.

2.2.4 Comparison of the original Voltage Protection Circuit and the Improved Voltage Protection Circuit

It is fair to say, then, that this circuit performs at least as well as the original Voltage Protection Circuit, even with the added capacitors. The final test is to see if there is a marked improvement protection against spikes at the other nodes. Namely, the voltage from the positive side of V_{out} to ground shouldn't spike as it did for the previous circuit. Figure 2.11 shows a direct comparison of this voltage in the original circuit (the darker plot) and in the improved circuit (lighter plot). As can clearly be seen, the spike that was present in the original circuit has disappeared. This circuit, then, is suitable for use in protecting the computer equipment from possible spikes from the Van de Graaff generator.

2.2.5 Final Implementation and Improvements

Of course, if the circuit is hit with a very large surge, something will give. Some component will overload, burn out, or otherwise fail. To help prevent this, an extra safeguard can be taken by placing passive "surge suppressor" components across the input and output terminals of the protection circuit. This surge suppressor is essentially a very nonlinear resistor that shorts in the presence of excessive voltage. To remain practical, 18 volt surge suppressors are used – any voltage above 18 volts is deemed excessive. In its final form, the circuit can be made in a layout such as the one in figure 2.12. Note that this layout uses the actual size of components.

There are a lot of engineering decisions involved with the layout in figure 2.12.

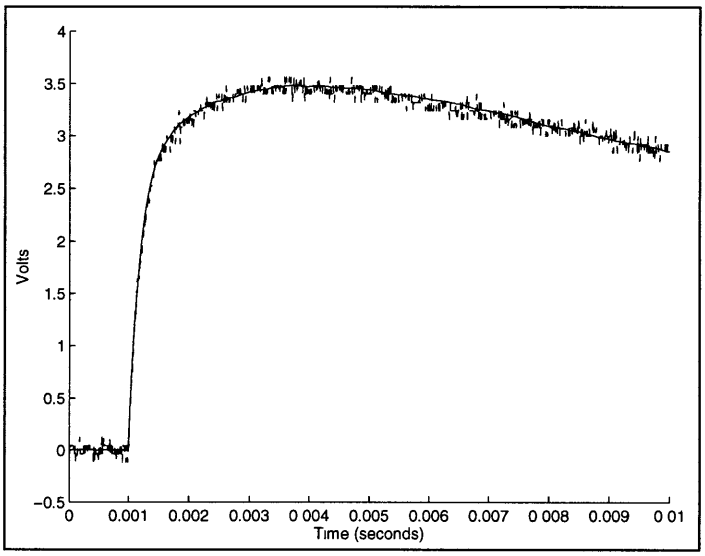

Figure 2.10: Predicted and actual responses of the Improved Voltage Protection Circuit.

For instance, note the spacing between the input terminals. They are placed close so that in the case of a very large spike it will actually break down the air between the two terminals and spike through the circuit – showing no effect at the output. There is a gap between the high voltage and low voltage sections of the circuit to prevent crossover between these two sides.

2.3 Magnet Control

2.3.1 Problem Statement

During the irradiation of an object, it makes sense (for efficiency, predictability, and reproducibility's sake) to maximize the amount of the beam that hits the object. In other words, provided a terminal voltage we must maximize the strength of the beam that hits the object. There are three different ways to measure this beam strength. One (the direct method) is to use the partial current, the number of electrons per second that arrive on the circular disk underneath the object-belt (see figure 2.2). A potential problem may arise, however if the object is thick; not allowing any electrons through the object (much less the object belt). Another is to use the four nano-ampere sensors that are located in the northwest, northeast, southwest, and southeast corners of the radiation area. It has been experimentally shown that when the partial current is maximized so is the sum of the four sensors. These four sensors also need to read (approximately) equal values, in order to assure that the beam is distributed evenly on the radiation area. Finally, the total current (see figure 2.1) can be used as well. It is important to note that there is the potential of a physical coupling between the steering

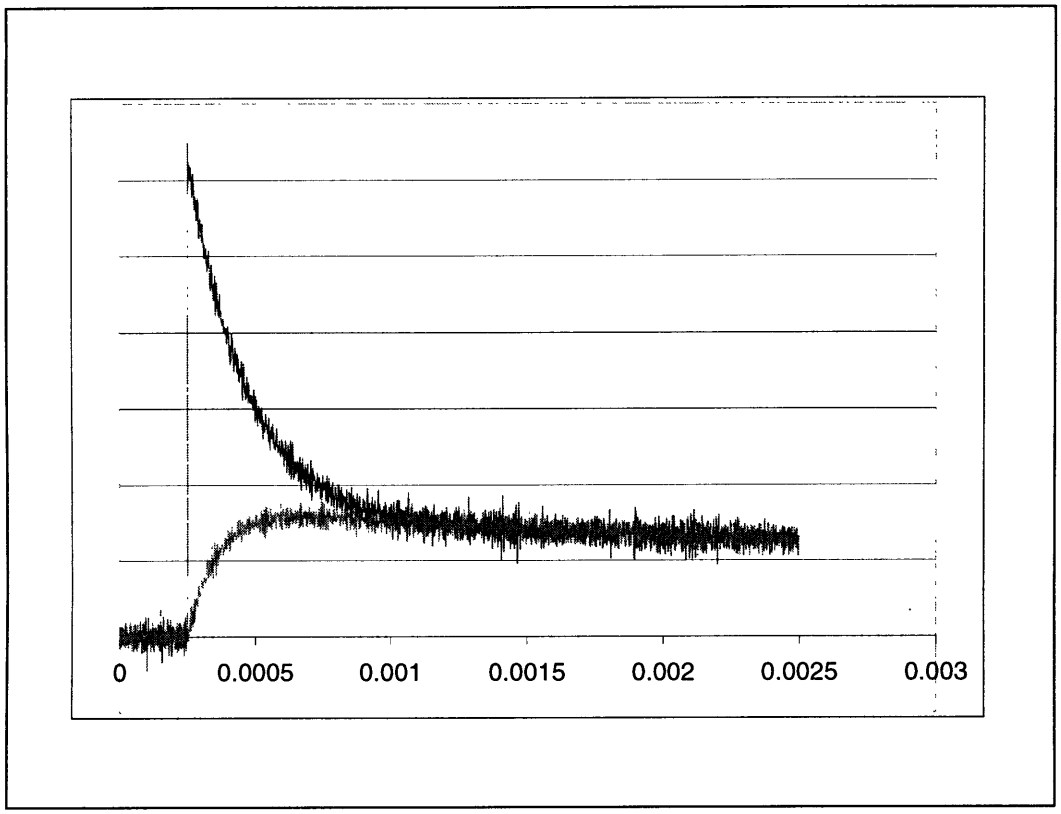

Figure 2.11: Response of the Voltage Protection Circuit from the positive side of V_{out} to ground

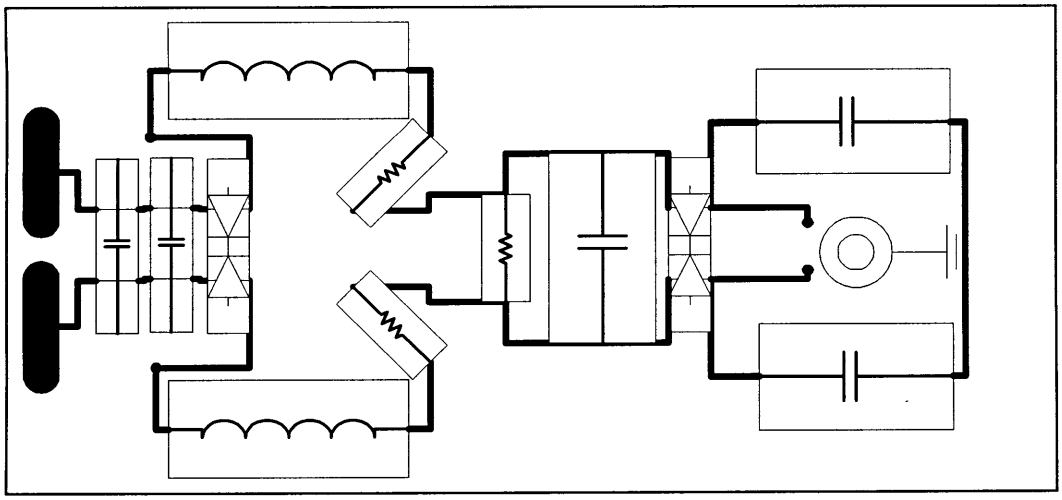

Figure 2.12: Layout of the final implementation of this voltage protection circuit

magnets. The columnating magnet could interfere with the north-south magnet, and likewise with the west-east magnet, or vice-versa. Currently the following manual protocol (as developed by Ken Wright, MIT High Voltage Research Lab) is used to maximize the beam current:

1. Fix the energy (the charge belt current) that the Van de Graaff generator will be using.
2. Adjust the solenoidal magnet until the partial current is maximized.
3. Adjust the N/S magnet until the partial current is maximized.
4. Adjust the W/E magnet until the partial current is maximized.
5. Adjust the solenoidal magnet once more.

It has been observed that the magnet settings vary both with beam energy and with beam current (which is proportional to the filament temperature). Another potential problem is that the acceleration tube is made out of glass, and could take on a charge over time, which may introduce "second-order" dynamics. If a model of the system can be derived, this protocol can be forgone, by setting the magnets at their proper value and then allowing the system to warm up to them.

2.3.2 Observations and Assumptions

This section is an attempt to provide the reader with an intuitive feel for the response of the Van de Graaff generator to various inputs. There are a lot of variables, dependencies, and conclusions drawn in the rest of this section which are based upon this "feel", and it is of benefit to the reader to have some understanding of these non-obvious conclusions before continuing. This section is non-rigorous and non-mathematical.

For the purposes of this paper we have full control of the current in the north-south magnet (i_{ns}), the current in the west-east magnet (i_{we}), the current in the solenoidal focusing magnet (i_f), and the terminal voltage. We do not have complete control of the filament temperature, which determines the total current (i_{total}) and the partial current ($i_{partial}$). Current flow (by means of the filament) depends on a temperature gradient, and the temperature varies during an irradiation. Provided no other changes, however, the total and partial current move proportionally. That is, the ratio between partial and total current remains approximately constant with variations in temperature. Figure 2.13 shows a set of data that exhibits this ratio. Unfortunately this ratio breaks down at lower temperatures, as there is less current flowing through the machine and therefore fewer electrons are lost to the surroundings.

Figure 2.13 also raises another issue. It is assumed throughout this paper that the sum of the four directional nano-ampere sensors is as good (perhaps better) a metric for representing the beam in the irradiation area as the partial current. Logically, there are a number of arguments for this. A sense of beam dispersion can be obtained

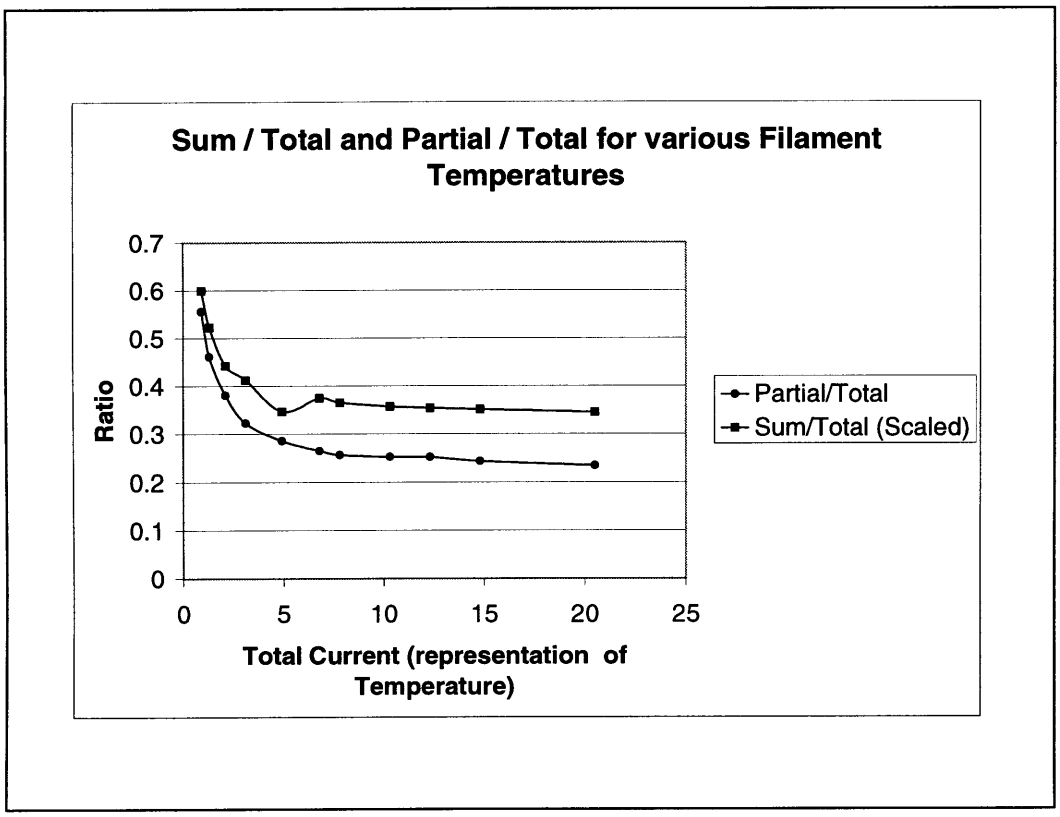

Figure 2.13: The Ratio of Sum of the four direction sensors divided by total current, and partial current divided by total current for an adjustment in filament temperature.

through the four sensors which is not seen in the partial current. Also, the partial current plate is located underneath the object belt, so lower energy electrons may not be able to penetrate the belt (or at least to the extent that they would at higher energies), whereas the four sensors are above the belt, avoiding this problem. To see an initial "proof" that the sum of the four sensors can at least replace the partial current measurement, please examine Figure 2.14. It is evident that the sum current is proportional to the partial current. Although this proportionality changes with terminal voltage (perhaps representing the amount of energy absorbed by the object belt) it is still a linear relationship.

When examining the output of the Van de Graaff generator, there is one optimal setting of i_{ns}, i_{we}, and i_f for each terminal voltage. By optimal it is meant that the ratio of partial current to total current (or for that matter the ratio of the sum of the four directional sensors to total current) is maximized, and the beam is distributed evenly over the irradiation area (as determined by the four directional sensors). There are, however, plenty of local maximums and minimums over the set of possible input currents, which present a notable algorithm development problem.

If we place the system at its optimal point (as determined by the manual method

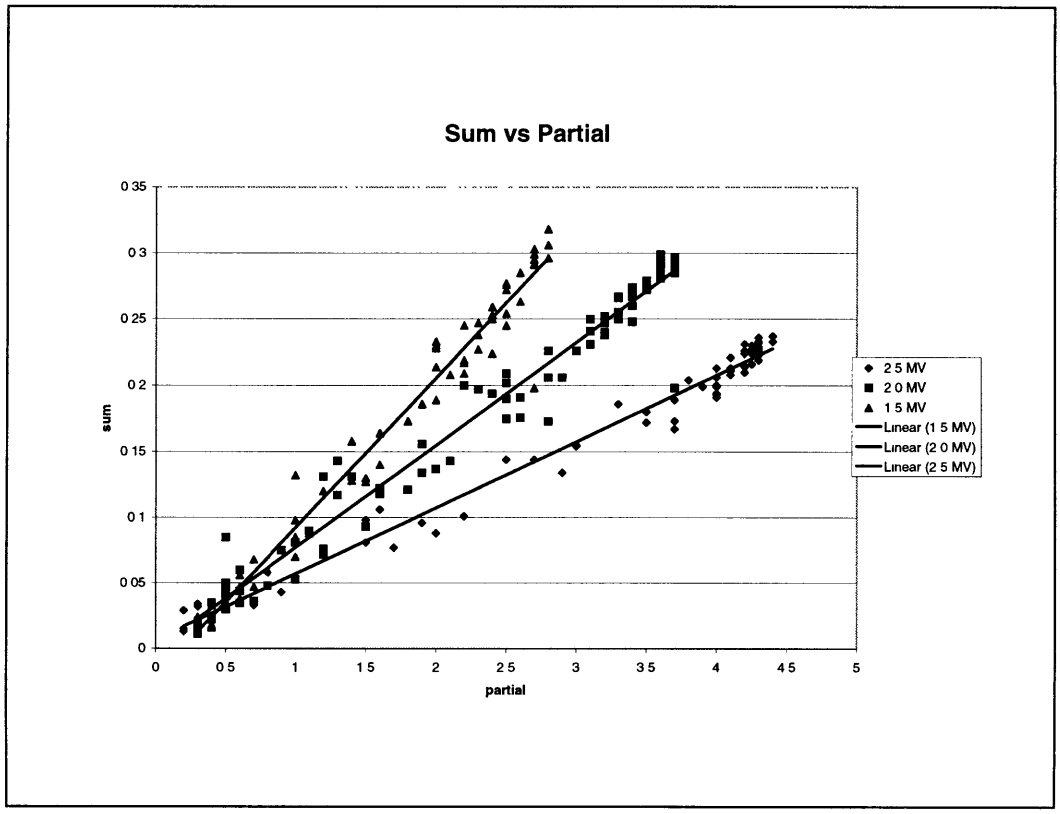

Figure 2.14: The Summed Current versus Partial Current.

described in the introduction) and vary the magnet currents one at a time we see the response (the ratios) taper off as the current leaves the optimal point in either direction. Put most simply it looks like a parabola. This is true for all i_{ns}, i_{we}, and usually true for i_f. The focusing current has the peculiarity that, at low terminal voltages, as i_f goes to 0 (leaving the optimal point going to the "left") the response gets stuck at a constant level. (It bears a strange resemblance to the bode magnitude plot of a second-order low-pass filter.) It has been assumed that this is due to hysteresis in the focusing magnet (the magnet has been driven with only positive current for as long as it has been in operation) such that as the current goes to zero there is still a magnetic field present strong enough to affect beams that have relatively little energy. Regardless, all other settings look roughly parabolic, influencing decision to use a second order fit for this system.

As we change the terminal voltage, a large non-linearity (that will be addressed later) is introduced into the system. Despite this, though, it can be said that the magnitude of i_{ns}, i_{we}, and i_f at the optimal point move monotonically with terminal voltage. How they move may vary from day to day (as is shown in the set of data), but it is the observation of those who run this system daily that on a given day the necessary currents to optimize the system increase with terminal voltage. Please see

section 2.3.6 later in this section for a more detailed discussion of this.

Lastly, it is a basic assumption in this paper that the effect of the glass rings taking on a charge during operation (and thus affecting later results) is minimal and not worth taking into account. To test this assumption two runs of the machine were performed at a terminal voltage of 1.5 MV. In between these runs, the machine was operated for a prolonged period at a significantly higher (greater than 2.5 MV) terminal voltage. The results of the two 1.5 MV results are presented in figure 2.15, and are nearly identical.

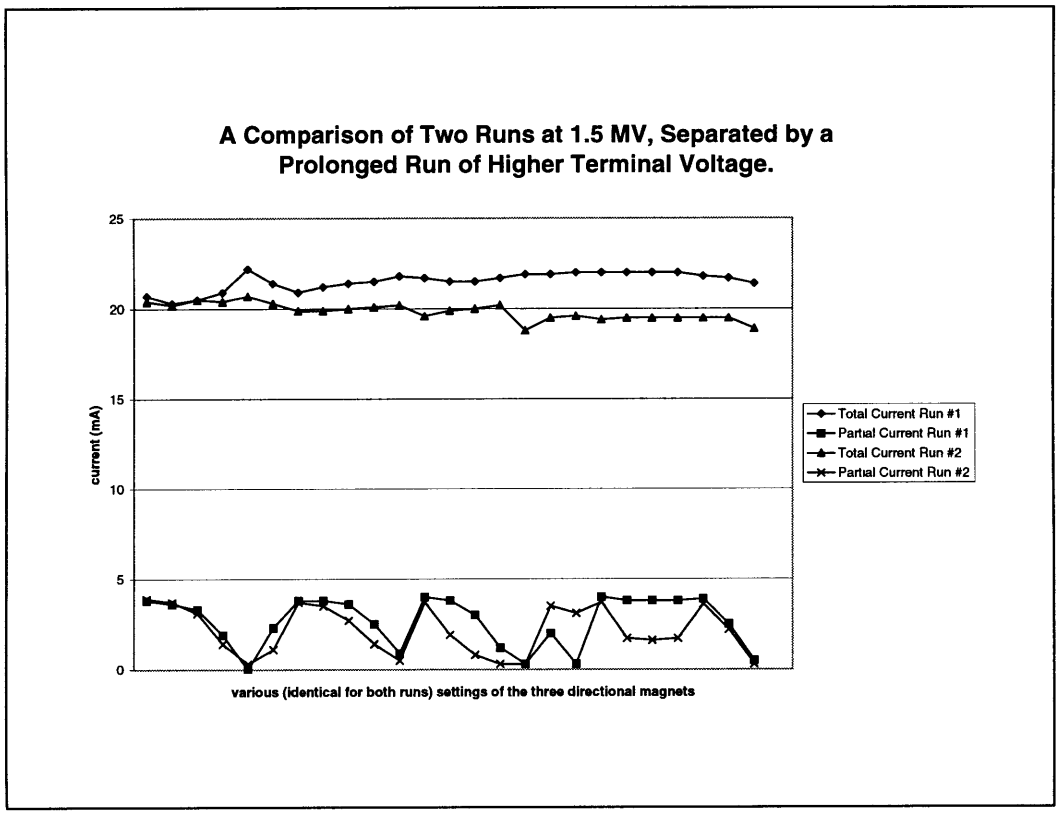

Figure 2.15: The effect of the glass rings taking on a charge is inconsequential.

2.3.3 Modeling

While a state-space model for this system could potentially be used, in terms of implementation such a model is entirely unnecessary. The generator is controlled via data acquisition and output hardware interfaced to a computer with a GPIB bus. Latencies within the computer, the bus, and the acquisition hardware entail that the entire system cannot operate at speeds much greater then 1 Hz. Any intermediate variables can be assumed to arise from phenomena that are orders of magnitude faster, and have reached their steady-state values by the time the next data acquisition cycle has been performed. For this reason, a simple input-output coupling matrix model has been assumed for the purposes of this analysis, which is of the form:

$$y = Ax$$

The output vector will simply consist of the outputs available to us:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} i_{nw} \\ i_{ne} \\ i_{sw} \\ i_{se} \\ i_{total} \\ i_{partial} \end{bmatrix}$$

$$\begin{cases} i_{nw} & \text{is the current measured off of the north-west nano-ampere sensor.} \\ i_{ne} & \text{is the current measured off of the north-east nano-ampere sensor.} \\ i_{sw} & \text{is the current measured off of the south-west nano-ampere sensor.} \\ i_{se} & \text{is the current measured off of the south-east nano-ampere sensor.} \\ i_{partial} & \text{is the partial current as defined in the introduction.} \\ i_{total} & \text{is the total current as defined in the introduction.} \end{cases}$$

The input vector will consist of the variables over which we have direct control (i.e. filament temperature is excluded). We will also be including second order terms for the inputs in accordance with the notes made in the observations section. So:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} i_{ns}^2 \\ i_{ns} \\ i_{we}^2 \\ i_{we} \\ i_{focus}^2 \\ i_{focus} \\ C \end{bmatrix}$$

$$\begin{cases} i_{ns} & \text{is the current through the north-south magnet.} \\ i_{we} & \text{is the current through the west-east magnet.} \\ i_{focus} & \text{is the current through the focusing magnet.} \\ C & \text{is the combined DC term of the three inputs.} \end{cases}$$

The coupling matrix, A, will then consist of coefficients that match the inputs to the outputs. These coefficients are assumed to be functions of primarily the terminal voltage. i.e.

$$A = \begin{bmatrix} a_{11} & \cdots & a_{17} \\ \vdots & \ddots & \vdots \\ a_{61} & \cdots & a_{67} \end{bmatrix}$$

where $a_{ij} = f(\text{terminal voltage})$.

2.3.4 Modeling for fixed Terminal Voltage

Theory

The use of the coupling matrix for a given terminal voltage, $A_{t.v.}$, will be to determine the initial optimal operating point for that terminal voltage. Ideally, the characteristics of the Van de Graaff generator are fixed, and once a coupling matrix is determined, it will hold for all time. In reality, the generator's characteristics vary greatly from day to day depending on innumerable conditions, such as changing sensor locations, tank pressures, vacuum pump efficiencies, room temperatures, and the vacuum tube power amplifiers used for setting the filament temperature. Because of this, a more realistic approach is to start at a guessed optimal point, conduct iterations around this point, use recursive least squares to update the coupling matrix $A_{t.v.}$, and then find the true optimal point *du jour*.

Having a decent guess for the initial optimal point is vital for a quick convergence. Therefore, the hope is that a general coupling matrix $\mathcal{A}$ can be obtained, where each coefficient, a_{ij}, is a function of terminal voltage. The constant coefficients of the coupling matrix of interest, $A_{t.v.}$, can then be obtained for any given terminal voltage by simply evaluating the general coupling matrix coefficients at a set terminal voltage. The initial optimal point can then be obtained from this first guess of $A_{t.v.}$.

In order to obtain $\mathcal{A}$, several different $A_{t.v.}$ matrices will be obtained, and then, hopefully, the coefficients of the general matrix $\mathcal{A}$ can be formed by fitting curves (non-linear if necessary) to the coefficients of the $A_{t.v.}$'s. To accomplish this, sets of output and input measurements were taken around optimal points obtained from the manual method described in the introduction at several different terminal voltages (0.5, 1.0, 1.5, 2.0, and 2.5 MV).

For each of the sets of data, the coupling matrix $A_{t.v.}$ was obtained as follows. The input and output data could each be combined into stacked input and output vectors.

$$\overline{y} = \begin{bmatrix} y_1(1) & \cdots & y_1(k) \\ \vdots & \ddots & \vdots \\ y_6(1) & \cdots & y_6(k) \end{bmatrix} = \begin{bmatrix} y(1) & y(2) & \cdots & y(k) \end{bmatrix}$$

and

$$\overline{y} = \begin{bmatrix} x_1(1) & \cdots & x_1(k) \\ \vdots & \ddots & \vdots \\ x_7(1) & \cdots & x_7(k) \end{bmatrix} = \begin{bmatrix} x(1) & x(2) & \cdots & x(k) \end{bmatrix}$$

such that

$$\overline{y} = A_{t.v.}\overline{x}$$

and k is the number of points taken in the set.

Now we'd like to find the coefficients of $A_{t.v.}$. This is not quite the normal least squares problem, as usually we're trying to find the vector x, not the matrix $A_{t.v.}$. We

can transform this into the familiar form by taking the transpose of the last equation, so that:

$$\overline{y}' = (A\overline{x})' = \overline{x}'A'$$

the solution of A follows as:

$$A = \left((\overline{xx}')^{-1}\overline{xy}'\right)'$$

which can be re-written as

$$A = \overline{y}'\overline{x}'\,(\overline{xx}')^{-1}$$

by noting that

$$(\overline{xx}')' = \overline{xx}'$$

Results and Discussion

The input and output data from the different runs at different terminal voltages were fit to coupling matrices in the least squares sense. The results are shown in the following figures.

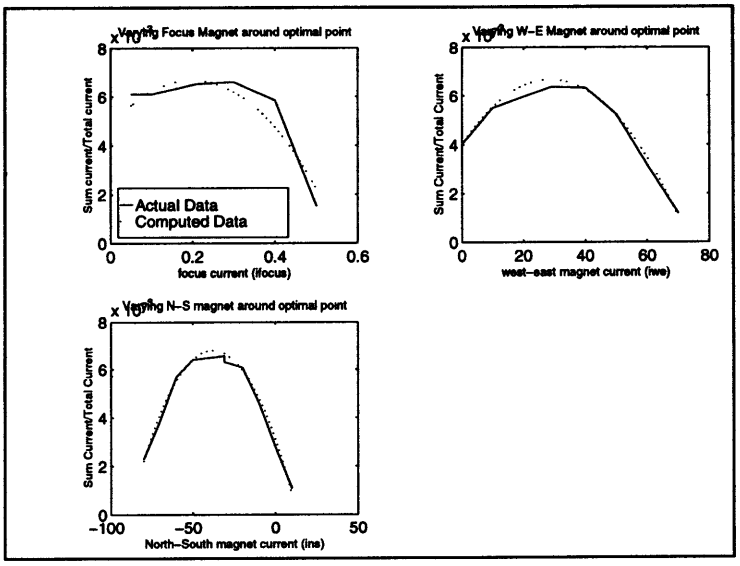

Figure 2.16: Various responses at 0.5 MV Terminal Voltage.

As is easily seen from Figures 2.16–2.20, the model that uses the previously mentioned least squares technique fits the experimental data well. A more specific discussion of the figures follows.

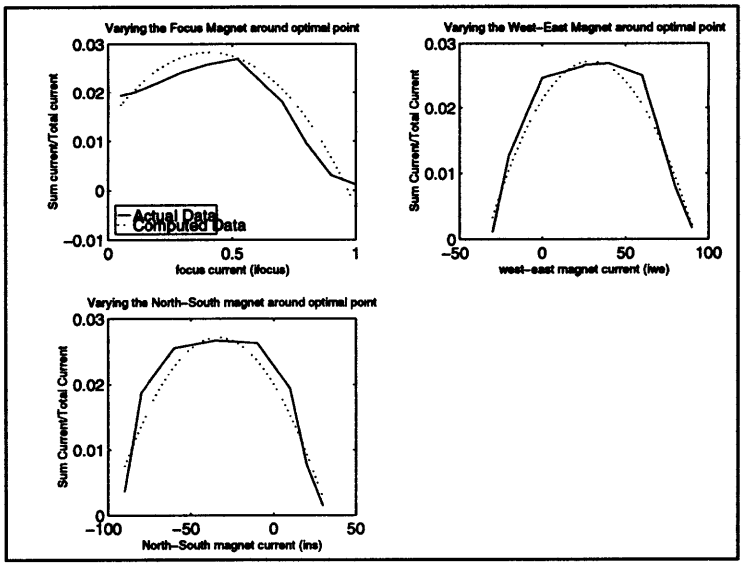

Figure 2.17: Various responses at 1.0 MV Terminal Voltage.

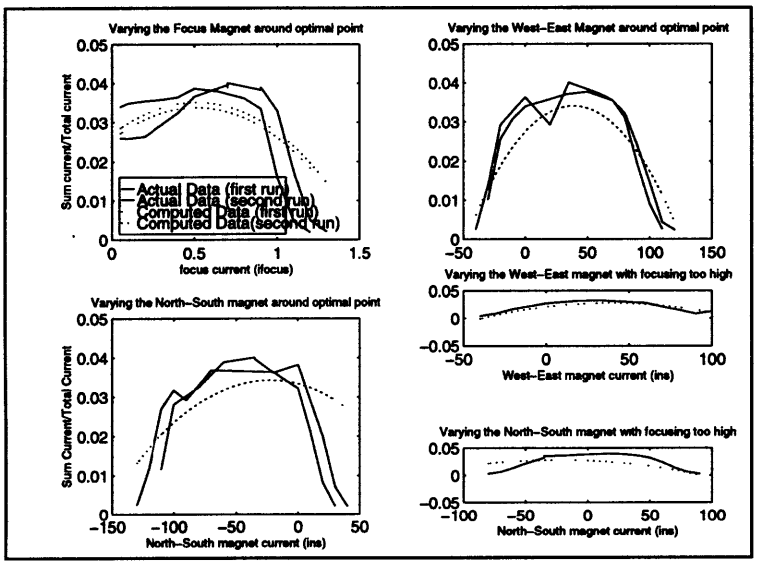

Figure 2.18: Various responses at 1.5 MV Terminal Voltage.

Figure 2.16: While the north-south and west-east predictions are very close, the focusing magnet prediction misses the target. This can be attributed to the lack of a parabolic shape in the real data. As mentioned before, the focusing magnet seems to have hysteresis that affects low energy beams.

Figure 2.17: Again, the north-south and west-east predictions are close, but the focusing magnet prediction is much closer then before.

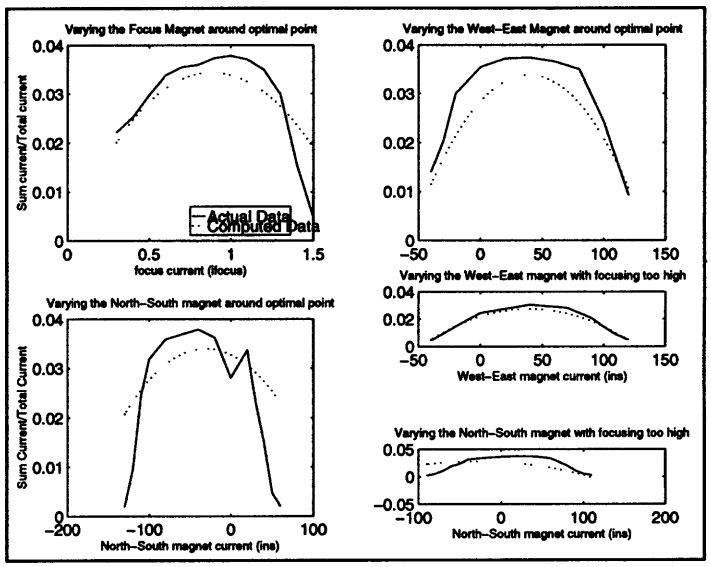

Figure 2.19: Various responses at 2.0 MV Terminal Voltage.

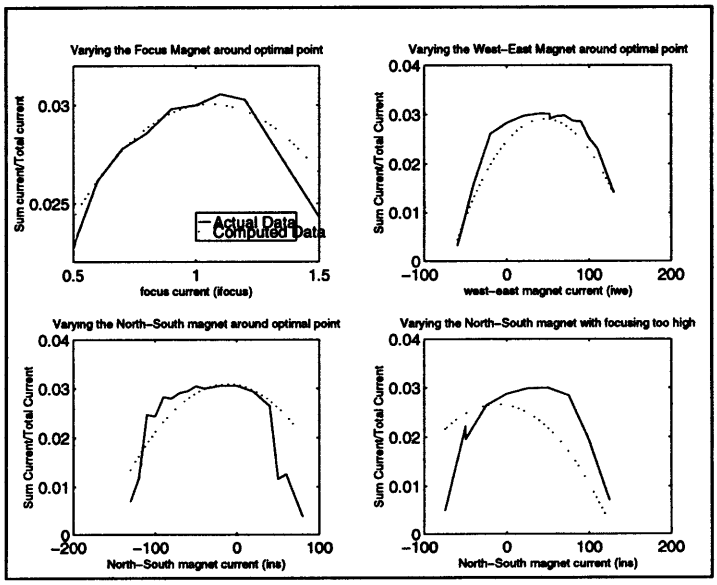

Figure 2.20: Various responses at 2.5 MV Terminal Voltage.

Figure 2.18: This time we see an improvement in the focusing magnet prediction, but the north-south prediction is off. Although it (approximately) guesses the appropriate optimal current, the predicted magnitude of the response isn't accurate. The 1.5 MV set of data was actually taken on two different days, and ended up having vastly different optimal north-south magnet currents, hence the discrepancy. Note that the 1.5 MV data does a fairly good job of predicting data that is not near the optimal point as well. It is the expectation that the model

should be "close" in these ranges, but an exact match isn't necessary as we will be using this model to predict the neighborhood of the optimal point.

Figure 2.19: Again, acceptable fits all across the board, especially when considering using the $A_{t.v.}$ matrix to pick out the location of the optimum point.

Figure 2.20: The fits to the data in the vicinity of the optimal point are good, but there is a serious degradation of the data when looking outside of this optimal point. It does however, match the appropriate range of values when the focusing current is set too high, it just shifts the peak slightly.

Now that the data has been collected and all of the $A_{t.v.}$ matrices have been tabulated, the next step in our process would be to fit the coefficients of each $A_{t.v.}$ matrix to some function to form the $\mathcal{A}$ matrix. Unfortunately, there does not seem to be any pattern to match for the entries of the individual $A_{t.v.}$ matrices. Figure 2.21, albeit confusing at first, is a graphical representation of the combination of the $A_{t.v.}$ matrices for the set of terminal voltages described above. For example, the graph in the top left corner is the function that would become the top left entry of the $\mathcal{A}$ (i.e. the coefficient formula a_{ij}), if a generic function could be found to describe it.

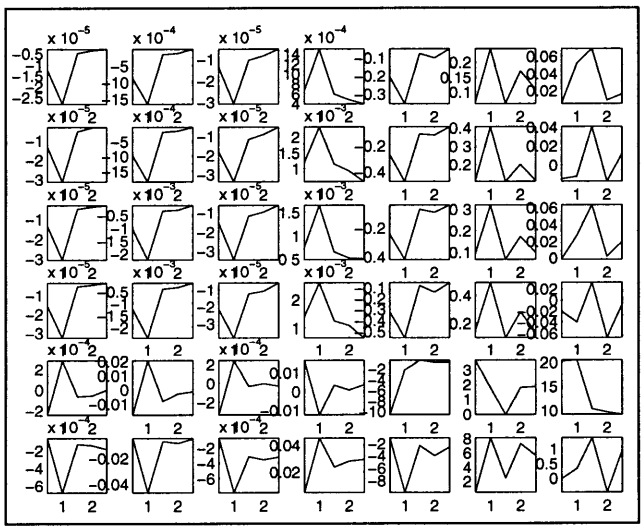

Figure 2.21: A graphical representation of the $\mathcal{A}$ matrix.

Clearly there really is no obvious function, linear or non-, to describe even one entry of this matrix, and even if there were it would be very difficult to determine from the limited data available to date. Thus, the attempt to form the non-linear coupling matrix $\mathcal{A}$ is not viable.

2.3.5 Modeling over all Terminal Voltage

Theory

Another attempt at modeling the system was made, this time using brute force. Since creating coefficients that were functions of terminal voltage didn't work, our next attempt was to include the terminal voltage as an input into the model. The new model for this system would then be:

$$y = \mathbb{A}x$$

$\mathbb{A}$ is a general coupling matrix relating the input x to the output y. This matrix could then be used to pick the initial optimal point for a given terminal voltage from which the recursive least squares of the matrix $A_{t.v.}$ would start. The input vector, x, in this case would be:

$$x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \end{bmatrix} = \begin{bmatrix} i_{ns}^2 \\ i_{ns} \\ i_{we}^2 \\ i_{we} \\ i_{focus}^2 \\ i_{focus} \\ T.V. \\ C \end{bmatrix}$$

where $T.V.$ is the terminal voltage of the Van de Graaff, and all the other entries are the same as before. The output vector, y, is the same output vector as was used previously.

The data from the various runs were then stacked into input and output matrices, and the coupling matrix $\mathbb{A}$ was fitted to the data in a least squares sense in the same general manner as before.

Results and Discussions

The performance of the coupling matrix $\mathbb{A}$ is compared against the actual data in Figures 2.22–2.24. A discussion follows.

Figure 2.22: These six graphs show that the general coupling matrix model actual performs surprisingly well for our purposes. Although the magnitudes of the fits can be off by as much as fifty percent, the location of the peaks of the fit, the important issue to us, corresponds quite well to the actual data.

Figure 2.23: Again, in terms of actual magnitude, the fits are significantly off from the actual data. The peaks, however, are acceptable although this time they are not as closely correlated to the actual data peaks as in the west-east magnet case. The important thing to note is that even in the worst case of the peak being off (T.V. = .05), the calculated peak is still within the broad peak area of the actual data.

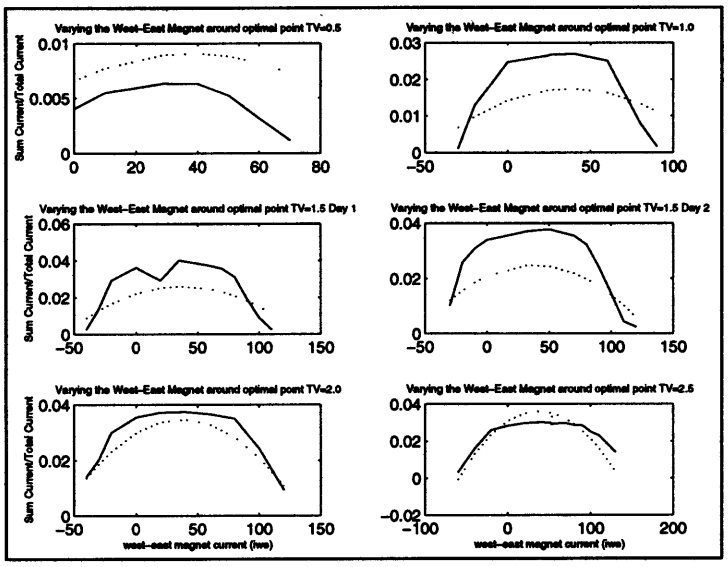

Figure 2.22: Comparison of the actual data and the computed data from the general matrix $\mathbb{A}$ for the west-east magnet and several terminal voltages.

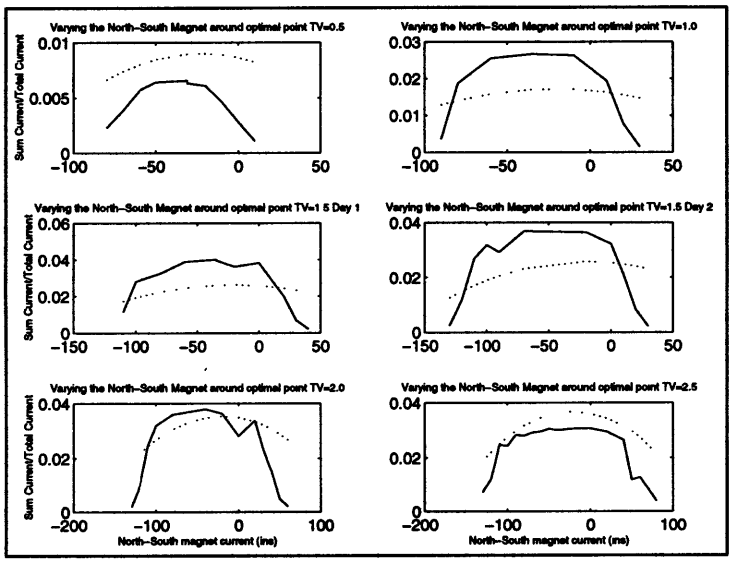

Figure 2.23: Comparison of the actual data and the computed data from the general matrix $\mathbb{A}$ for the north-south magnet and several terminal voltages.

Figure 2.24: These graphs show that the general coupling matrix idea performs quite poorly when compared to the actual focusing magnet data. In terms of fitting the actual magnitudes, the performance is a little worse than in the north-south and west-east magnet cases. However, in terms of getting the peaks right, the model really starts to fall apart. At the higher terminal voltages, the calculated

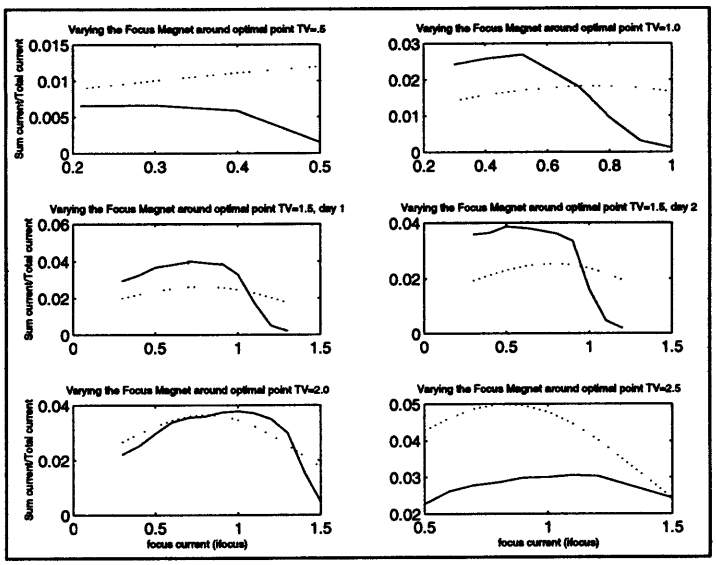

Figure 2.24: Comparison of the actual data and the computed data from the general matrix $\mathbb{A}$ for the focusing magnet and several terminal voltages.

peak is fairly close, but as the model moves into the lower terminal voltages, the calculated peak is no longer within the broad actual data peak.

Overall, using the general coupling matrix, $\mathbb{A}$, does not seem like it will be a viable option for picking an initial optimal point for the system. While the fit performs decently well compared to the actual data for the north-south and west-east magnets; the performance to the actual focusing magnet is simply too far off to justify this method. This method could conceivably be used in practice, but a more efficient method is desirable.

2.3.6 Picking Optimal Points

Since the last method did not work well for predicting the optimal point at any specific terminal voltage, we are still stuck deriving a method whereby this optimal point can be determined. The knowledge that we have gained from the previous sections of this paper will allow us to do this easily. We will make use of two related facts for this method. The first is that when examining two proximal terminal voltages, the $A_{t.v.}$ matrix from one can be used to approximate the other. Figures 2.25 and 2.26 exemplify this.

The other useful observation is that to a good approximation the optimal points vary monotonically with terminal voltage, as presented in the observations section. In fact, a simple linear fit comes reasonably close to this data, as shown in Figures 2.27 through 2.29.

Using this knowledge, the final algorithm will follow this general plan:

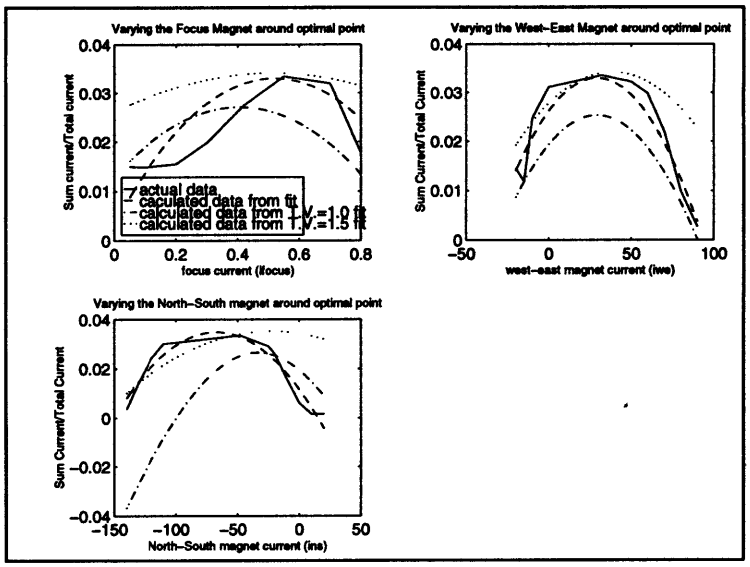

Figure 2.25: Graphs showing T.V. = 1.25 MV data along with a fit to that set of data, and the fits from the T.V. = 1.0 MV and the T.V. = 1.5 MV data sets.

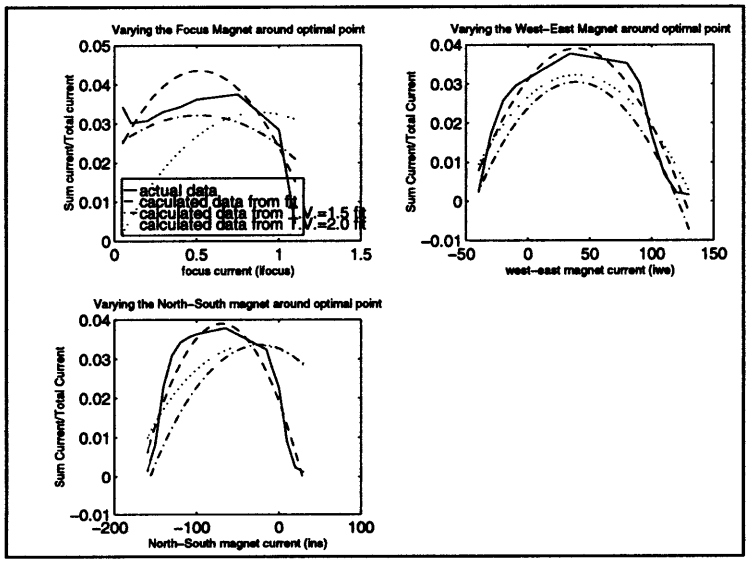

Figure 2.26: Graphs showing T.V. = 1.75 MV data along with a fit to that set of data, and the fits from the T.V. = 1.5 MV and the T.V. = 2.0 MV data sets.

1. Fix the terminal voltage of the Van de Graaff generator.

2. Compute the optimal point for that terminal voltage by plugging the value into the linear fit of the actual optimal points.

3. Pick the closest $A_{t.v.}$ matrix to that terminal voltage. Previous $A_{t.v.}$'s are stored

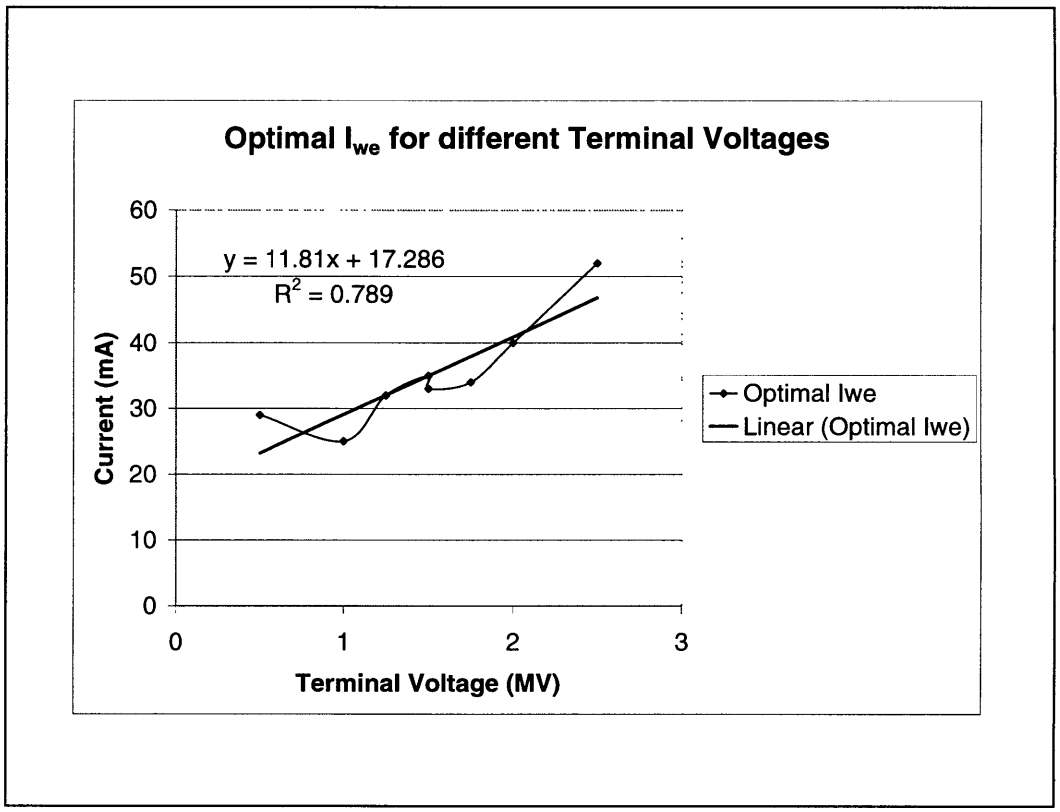

Figure 2.27: Optimal points for I_{we} versus Terminal Voltage.

on the computer.

4. Use this $A_{t.v.}$ matrix to initiate a recursive least squares fit, starting with data points around the initial optimal points derived in step 2.

5. Keep generating data points with the Van de Graaff generator extending around this initial data point until a new optimal point can be identified.

6. Set the Van de Graaff generator at this optimal point for the duration of the experiment.

7. Save the computed $A_{t.v.}$ matrix and the optimal point for this terminal voltage for later use.

2.3.7 Similarity Transform Observations

Since many of the inputs and outputs of the modeled system (i.e. the north-south and west-east magnets, and the northwest, southwest, southeast, and northeast sensors) are directional in nature, it may prove interesting to look for patterns in the A matrix. We would not expect to see any patterns as is, because the coordinate directions of

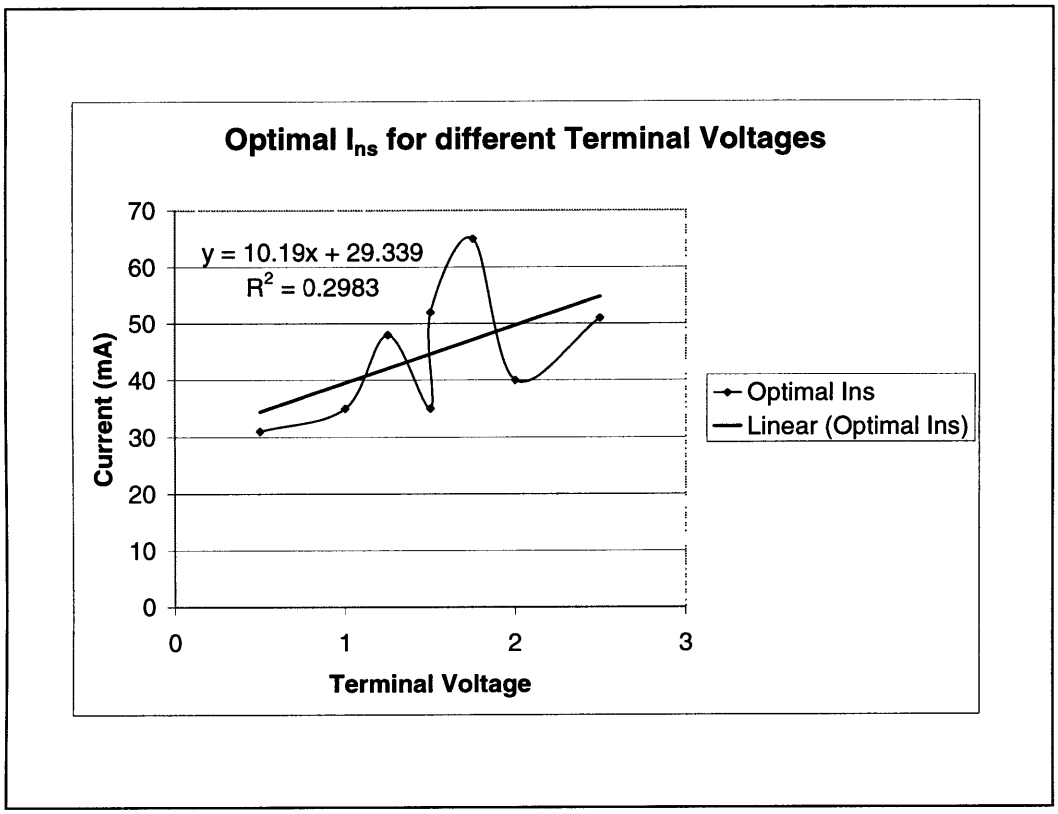

Figure 2.28: Optimal points for I_{ns} versus Terminal Voltage.

the outputs of the system do not match the coordinate directions of the inputs. To get around this problem, we make use of a similarity transform, changing the directional outputs from northwest, southwest, southeast, and northeast, to north, south, west, and east. i.e.:

$$y^* = \begin{bmatrix} \text{north component} \\ \text{south component} \\ \text{west component} \\ \text{east component} \\ \text{total component} \\ \text{partial component} \end{bmatrix} = \begin{bmatrix} .5(i_{nw} + i_{ne}) \\ .5(i_{sw} + i_{se}) \\ .5(i_{nw} + i_{sw}) \\ .5(i_{ne} + i_{se}) \\ i_{total} \\ i_{partial} \end{bmatrix} = \begin{bmatrix} .5 & .5 & 0 & 0 & 0 & 0 \\ 0 & 0 & .5 & .5 & 0 & 0 \\ .5 & 0 & .5 & 0 & 0 & 0 \\ 0 & .5 & 0 & .5 & 0 & 0 \\ .5 & .5 & 0 & 0 & 1 & 0 \\ .5 & .5 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i_{nw} \\ i_{ne} \\ i_{sw} \\ i_{se} \\ i_{total} \\ i_{partial} \end{bmatrix} = Ty$$

where T is the similarity transform of interest.

We can multiply our model by this T to obtain a new coupling matrix, TA, whose structure, if we assume orthogonality between the north-south and west-east magnets, can be expected to follow the form:

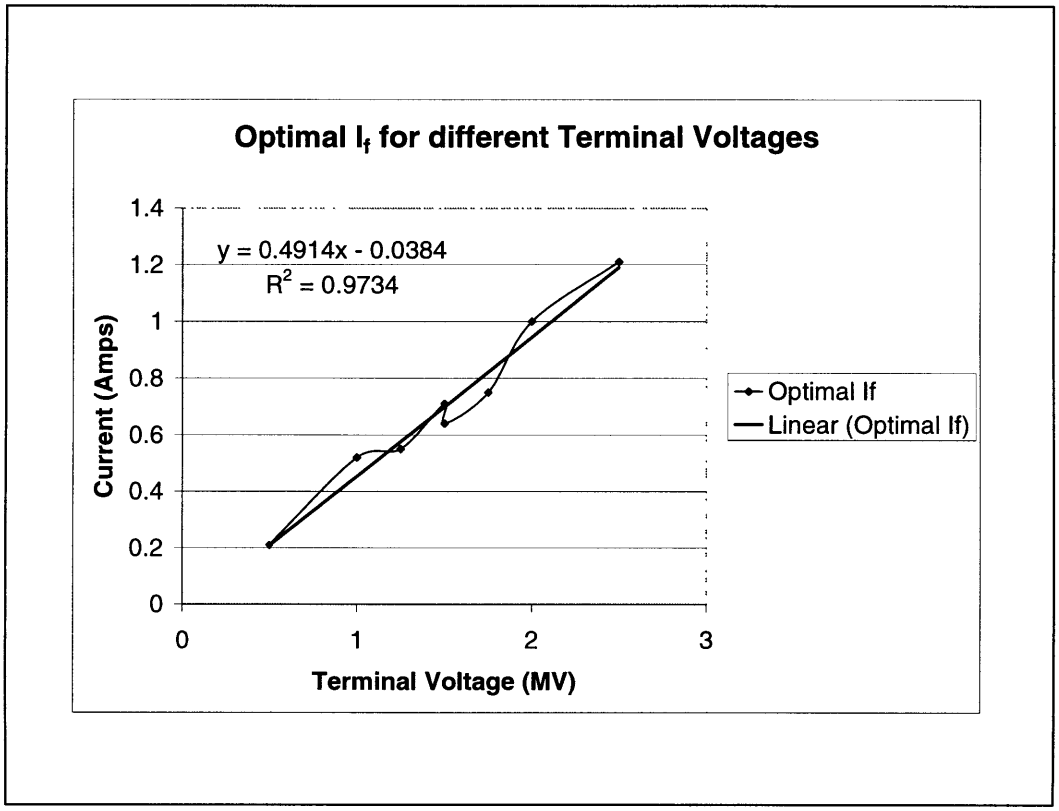

Figure 2.29: Optimal points for I_f versus Terminal Voltage

$$y^* = Ty = YAx = \begin{bmatrix} \text{north component} \\ \text{south component} \\ \text{west component} \\ \text{east component} \\ \text{total component} \\ \text{partial component} \end{bmatrix} = \begin{bmatrix} * & * & 0 & 0 & ? & ? & * \\ & * & 0 & 0 & ? & ? & * \\ 0 & 0 & * & * & ? & ? & * \\ 0 & 0 & * & * & ? & ? & * \\ 0 & 0 & 0 & 0 & 0 & 0 & * \\ & * & * & * & ? & ? & * \end{bmatrix} \begin{bmatrix} i_{ns}^2 \\ i_{ns} \\ i_{we}^2 \\ i_{we} \\ i_f^2 \\ i_f \\ C \end{bmatrix}$$

$$\begin{cases} * & \text{indicates an expected entry in the coupling matrix.} \\ 0 & \text{indicates an expected 0 in the coupling matrix.} \\ ? & \text{indicates no expectations for that coefficient in the coupling matrix.} \end{cases}$$

To test out this similarity matrix, the 2.5 MV data was first normalized. This normalization was done so that the coefficients of coupling matrices could be directly compared. A coupling matrix, $A_{2.5^*}$, was then fit to this data and multiplied by T. The result was:

$$Ty = TA_{2.5^*}x \rightarrow TA_{2.5^*} = \begin{bmatrix} -0.0771 & 0.0162 & -0.2389 & 0.4308 & -1.5127 & 2.4522 & 0.2434 \\ -0.0677 & 0.0822 & -0.2395 & 0.4376 & -1.2396 & 2.2243 & 0.0906 \\ -0.0745 & 0.0475 & -0.2562 & 0.3844 & -1.3687 & 2.2792 & 0.3015 \\ -0.0745 & 0.0509 & -0.2221 & 0.4840 & -1.3836 & 2.3974 & 0.0325 \\ -0.0005 & 0.0054 & -0.0078 & 0.0169 & -0.1170 & 0.2265 & 0.8855 \\ -0.0749 & 0.0568 & -0.2438 & 0.4348 & -1.2440 & 2.0951 & 0.2787 \end{bmatrix}$$

Recall that the output and input vectors are composed of:

$$\begin{cases} y_1 & \text{north current} \\ y_2 & \text{south current} \\ y_3 & \text{west current} \\ y_4 & \text{east current} \\ y_5 & \text{total current} \\ y_6 & \text{partial current} \end{cases} \text{ as inputs and } \begin{cases} x_1 & (\text{north-south magnet})^2 \\ x_2 & \text{north-south magnet} \\ x_3 & (\text{west-east magnet})^2 \\ x_4 & \text{west-east magnet} \\ x_5 & (\text{focus magnet})^2 \\ x_6 & \text{focus magnet} \\ x_7 & \text{constant} \end{cases} \text{ as outputs.}$$

There are a couple of things which can be said about this matrix ($TA_{2.5^*}$).

- Looking at the row which determines total current (row 5), we see that this output is determined primarily by the constant term (column 7). Intuitively this makes sense, as the current should be set primarily by the speed of the Van de Graaff generator's belt and partially by the terminal voltage the generator is operating at.

- We see that the partial current (row 6) has a stronger dependence on the focusing magnet (columns 5 and 6) then the other inputs. Again, this makes sense. The partial current sensor has a large surface area, and will depend primarily whether the beam coming down the tube is focused rather then if the beam is a little to the north, south, west, or east.

- Finally, there is no obvious relationship between the directional outputs (rows 1–4), and the directional inputs (columns 1–4). This is because there is a very small range over which, for example, an increase in the north-south magnet will cause an increase in the north current component, a decrease in the south current component and no change in the west or east current components. Outside of this range, an increase in the north-south magnet will cause decreases in the north, south, west, and east current components as the electron beam begins to lose current to the metal sides of the acceleration tube.

To show that there is a range over which the north-south magnet primarily affects the north and south currents, a linear fit was made to data from the linear range of the TV = 2.5 MV run. Again, this data was normalized to itself so that the coefficients

of the transformed matrix could be directly compared. To highlight this point, the focusing magnet was not taken into account and the north-south and west-east magnets were assumed to be linear. So:

$$\begin{cases} y_1 & \text{north current} \\ y_2 & \text{south current} \\ y_3 & \text{west current} \\ y_4 & \text{east current} \end{cases} \text{ as inputs and } \begin{cases} x_1 & (\text{north-south magnet})^2 \\ x_2 & \text{north-south magnet} \\ x_3 & (\text{west-east magnet})^2 \\ x_4 & \text{constant} \end{cases} \text{ as outputs.}$$

and the transform:

$$T = \begin{bmatrix} .5 & .5 & 0 & 0 \\ 0 & 0 & .5 & .5 \\ .5 & 0 & .5 & 0 \\ 0 & .5 & 0 & .5 \end{bmatrix}$$

was used. The resulting $TA_{2.5^{**}}$ matrix from this fit was then:

$$Ty = TA_{2.5^{**}} = \begin{bmatrix} -0.0050 & 0.0040 & 1.0010 \\ -0.0372 & -0.0012 & 1.0384 \\ -0.0195 & 0.0775 & 0.9420 \\ -0.0227 & -0.0746 & 1.0973 \end{bmatrix}$$

This matrix shows that, over the linear range:

- The east current component (row 4) depended primarily on the negative of the west-east magnet (column 2).
- The west current component (row 3) depended primarily on the west-east magnet (column 2).
- The south current component (row 2) depended primarily on the negative of the north-south magnet (column 1).
- And it's anyone's guess what is going on with the north current component (row 1).

What these results do show us, though, is that the coupling between the different inputs and the different outputs are often non-intuitive over the entire range of the beam. This might explain why previous attempts to model the system from a physical standpoint met with failure. The results also show that, over a small range near the optimal point, the directional inputs can be expected to affect the directional outputs in a somewhat predictable manner.

Chapter 3

Application: System Structure

A few words about implementing this proposed control scheme are included for here.

3.1 External Interface Issues

- All information will be shared between the instruments and a concentrator via a GPIB network using the IEEE 488.2 communication protocol, to insure that all instruments will be safe from the dangerous voltages present in the Van de Graaff. Instruments that use the RS-232 or RS-485 communication protocol will also be used, but will still communicate with the computer over the GPIB bus, by means of a serial-to-GPIB converter.

- The north-south and west-east magnets will be current driven to avoid hysteresis. The approximate range of currents needed to drive these magnets is -100 mA to 100 mA, at a resistance of approximately 40 Ω. That is, the current source needed to drive this must be able to operate in a constant current mode, that can output up to 100 mA at up to 4 Volts. There are 0-20 mA current sources available for RS-485 that when appropriately biased and amplified will work in this range.

- The focusing magnet, on the other hand, requires some special instrumentation. It operates in a range of 0-2 Amps, at up to 200 Volts. For this application, a high current/voltage power supply (capable of 0-7 Amps at 300 Volts) that is analog programmable will be used. The analog programming will be performed by a GPIB compatible D/A converter that can create a very accurate and fast settling 0-1 Volt output. This output will serve as the analog input to the high voltage power supply, which will in turn drive the focusing magnet.

- The four beam-position currents (i_{nw}, i_{ne}, i_{sw}, i_{se}), the terminal voltage, the partial current, and the total current are all read through a Keithley 8 channel voltage scanner, after being scaled into range by appropriate resistor networks (or in the case of the terminal voltage a generating voltmeter).

- At present, the system is also capable of reading data from a slew of other instruments, including the Analog Devices 6B serial products (read voltage, set current, read RTD, read/set 32 bits of digital logic), and Tektronix 200 and 500 series oscilloscopes.

3.2 Data Processing within the Concentrator

Figures 3.1 and 3.2 show pictures of the working concentrator and observer. All code was written in LabVIEW 4.1, and is portable to any system that has LabVIEW compiled for it; no platform dependent operations were used in the design of this project. Figure 3.1 is a hierarchy of the concentrator. Consider each square a functioning program on the computer. A line connecting two programs indicates a data dependency. The box marked "conc" is the part of the concentrator that is running immediately at startup, the part that receives data over the network. This part of the code runs before a parameter file has been downloaded from an observer. The remaining boxes on the top row are all processes that are loaded by the parameter file. They are (from left to right):

- Set HP Voltage. This process is used to control the focusing magnet by means of a GPIB programmable voltage supply which is connected to a large (analog programmable) current supply.
- Set Analog Devices 6B21 Current #1. This process sets a current of 0-20 mA which can then be used (through proper amplification) to control the north-south magnet's current.
- Set Analog Devices 6B21 Current #2. This process is used to control the west-east magnet's current.
- Local Startup process. This process initializes the instruments for which this particular concentrator is responsible.
- Partial Current. This process monitors the Partial Current from the Keithley Scanner.
- Scan Channels 1-4. This monitors i_{nw}, i_{ne}, i_{sw}, and i_{se} by means of the Keithley Scanner.
- Terminal Voltage. This process monitors the Terminal Voltage from the Keithley Scanner.
- Total Current. This process monitors the Total Current from the Keithley Scanner.

The processes in the second row all are subroutines of the startup concentrator program. From left to right they are:

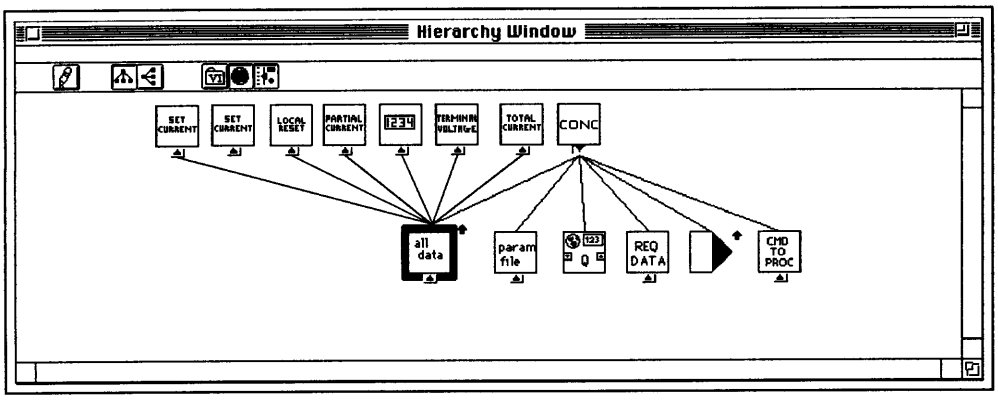

Figure 3.1: A schematic diagram of the concentrator.

- The data buffer. Note that this is shared between the initial concentrator program (which is the part of the code that receives messages from the observer, and therefore must access the buffer to send data) and all of the processes launched by the parameter file (which must access the buffer to read or send data).
- The parameter file interpreter. When a parameter file is received from the observer, the concentrator passes it along to this subroutine which checks it for errors and, if finding none, follows the script.
- The network queue. This is the memory allocation for the double buffer mentioned in reference to handling the incoming network connections (sections 1.3.2 and 1.4.2).
- Requested data handler. When an observer requests data from the buffer, this subroutine handles it.
- Send TCP/IP message. This subroutine is responsible for actually sending the data across the network to the appropriate computer. All data is sent in a platform-independent "big endian" form.
- Send command to process. This subroutine handles all direct messages to processes that are running (lines from the processes to this subroutine were left off for legibility) such as halting and restarting.

3.3 Data Processing within the Observer

Figure 3.2 shows an example user interface for the observer. The big panel in the upper-left monitors the four beam-position currents. The three analog-esque panels show the Terminal Voltage, the Total Current, and the Partial Current. Values can be requested and set using the buttons in the lower left. Values that are requests for

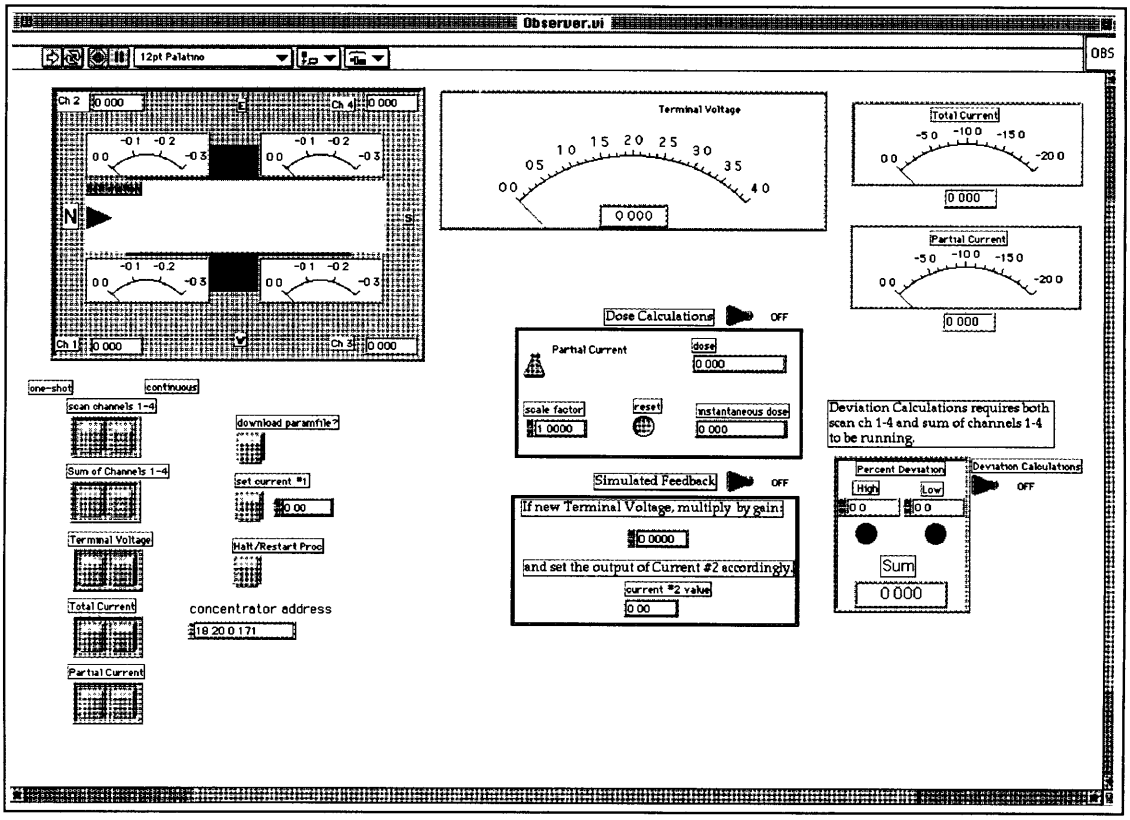

Figure 3.2: The user interface of an observer.

data can be requested in two ways: by clicking the left button the value is requested for immediate delivery, by clicking the right button the value is sent when new data is available (event-driven mode). The parameter file can selected, edited and then downloaded by clicking a button in the lower left as well. The remaining three groups of buttons show "higher-level" calculations. The dose and instantaneous dose can be calculated by turning the appropriate box on. The dose calculation can be based on either Partial Current or Total Current by means of a toggle switch. The simulated feedback box, when turned on, sets a current as a scaled value of the Terminal Voltage whenever the Terminal Voltage value arrives at the observer (regardless of the means by which it was requested). The percent deviation calculation, when selected, actually uses two values, which could (but don't) come from different concentrators. It looks to see if the one of the beam-position currents is a certain percentage higher or lower than the average of the four, and sets a flag if it has. The individual currents is one value, and the sum of those four is another value, both required by this calculation. While the example is trivial, as the sum could certainly be derived from the four individuals it represents the possibility of more meaningful, complex calculations based on multiple values.

Some examples of other possible complex computations for the observer follow.

- A running average, with possible weights.
- Integration or other methods of filtering values.
- Complex chart displays
- FFTs or other spectrum analyses.

Chapter 4

Performance Results

Included in this chapter are some charts showing some timing constraints of a working concentrator/observer system.

4.1 Timing the Buffer

The first figure, figure 4.1, shows a plot of access times in the buffer. This was done outside of the context of the concentrator or observer, it is merely a test of the computers' ability to sort data. This test was setup as follows. N datapoints were placed into the buffer (horizontal axis of the plot) as a certain value, all with different timestamps. The buffer was then asked to return all the datapoints since time zero in "peek-since" mode (defined in section 1.4.2), which will have the effect of returning all the datapoints in the buffer for that value. The time it took to complete this request is shown on the vertical axis, in milliseconds. There are a couple of things to note about this experiment. First, it would be much more efficient to use "peek-all" mode, since we know we are requesting all the data, but this test was designed as a worst case scenario. Second, the code for the buffer is not necessarily optimized in "peek-since" mode. The code assumes that the datapoints can be in the buffer for that value in *any* order, which is the only safe assumption. This is not optimal, however, if we know that the values will enter the buffer sequentially. In this case, the code could be modified to only look for the first value whose timestamp comes after the requested time and then send the remaining values. This would greatly increase the performance of the buffer, but at the cost of true generality in the system. Regardless, as expected the access time increases with the complexity of the request. The code is written to take time of order N to service the request, and in the long run it does seem to be linear. The two curves on the plot represent the results from two different computers, as described in the legend.

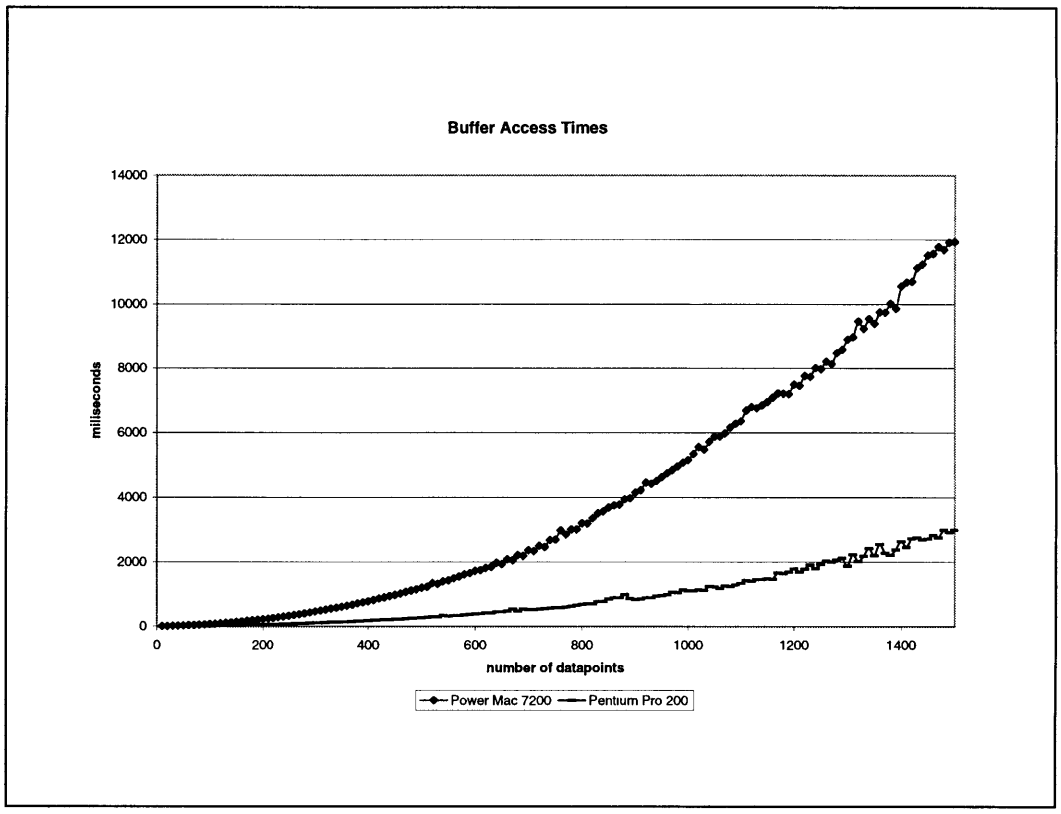

Figure 4.1: Access times for multiple reads from the buffer.

4.2 Calculating the Network Latency

Figure 4.2 exhibits a histogram of the latency in the network. In this experiment, the time was recorded just after sending a request for one value ("peek-one") from the buffer on a remote concentrator, and recorded again after the concentrator replied. The difference in these two times was recorded and was binned into a histogram. Note that while most values are clustered, there are occasional requests that take a much longer time to return. This can be explained: the concentrator in question was running four processes in tandem, to simulate a busy system, during this test. If one of these processes is writing to the buffer when the request to see data is received from the observer, the request cannot be fulfilled until the buffer is no longer in use, thereby stalling the system temporarily and increasing the latency time.

4.3 Exploring the Keithley

Figures 4.3 and 4.4 show the results of an experiment verifying the properties of the Keithley eight channel scanner that is used to measure the terminal voltage, partial current, total current, and the four beam-position currents. The scanner has two res-

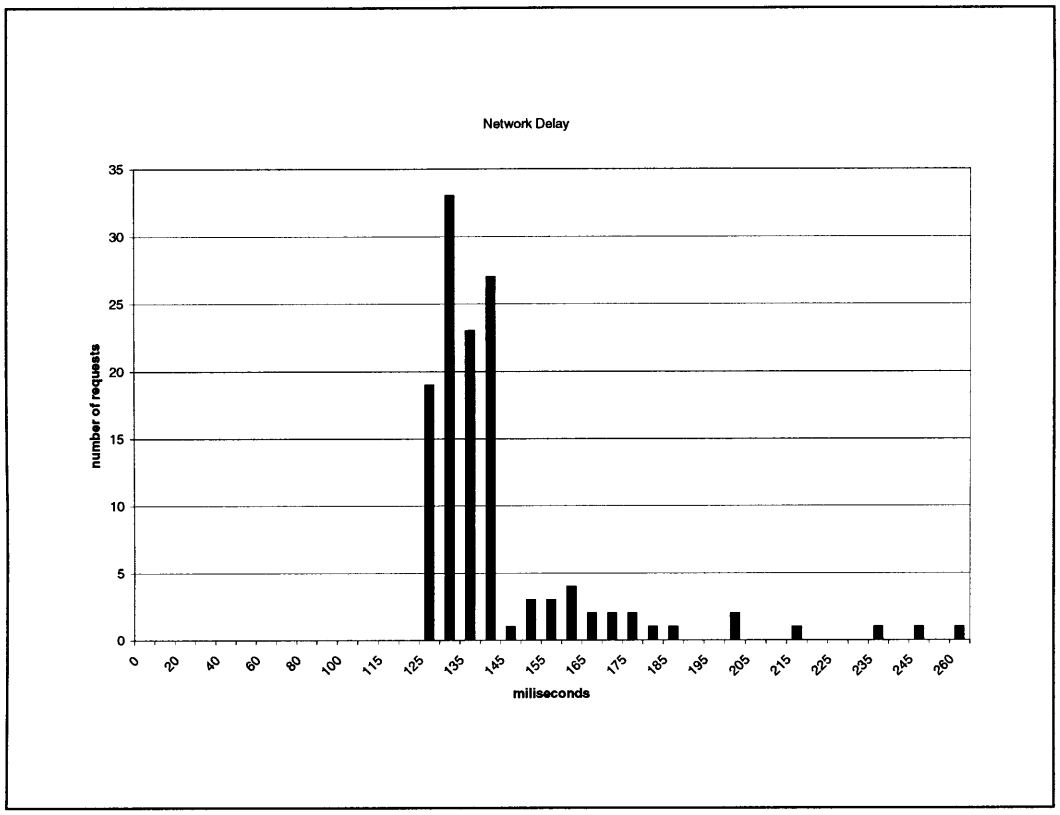

Figure 4.2: The latency in the network.

olutions with which it can operate: 4.5 decimal accuracy and 5.5 decimal accuracy. Furthermore, it has the option to filter the data while scanning, or it can just report values without filtering. The four combinations of the above modes were tested for speed. In both of these charts, the horizontal axis represents the number of points buffered into the Keithley's memory before sent over the GPIB, by means of the process, and into the data buffer. The vertical axis is the time, in milliseconds, that the entire operation took. Figure 4.3 has the scanner reading one channel per instruction, as the process that manages the measuring of the terminal voltage would require; and figure 4.4 has the scanner configured to read four channels in one shot, as the process that monitors the four beam-position currents would need. The most apparent conclusion to draw from this figure is that if speed is a requirement, it is quite a detriment to use both internal filtering and 5.5 digit resolution mode.

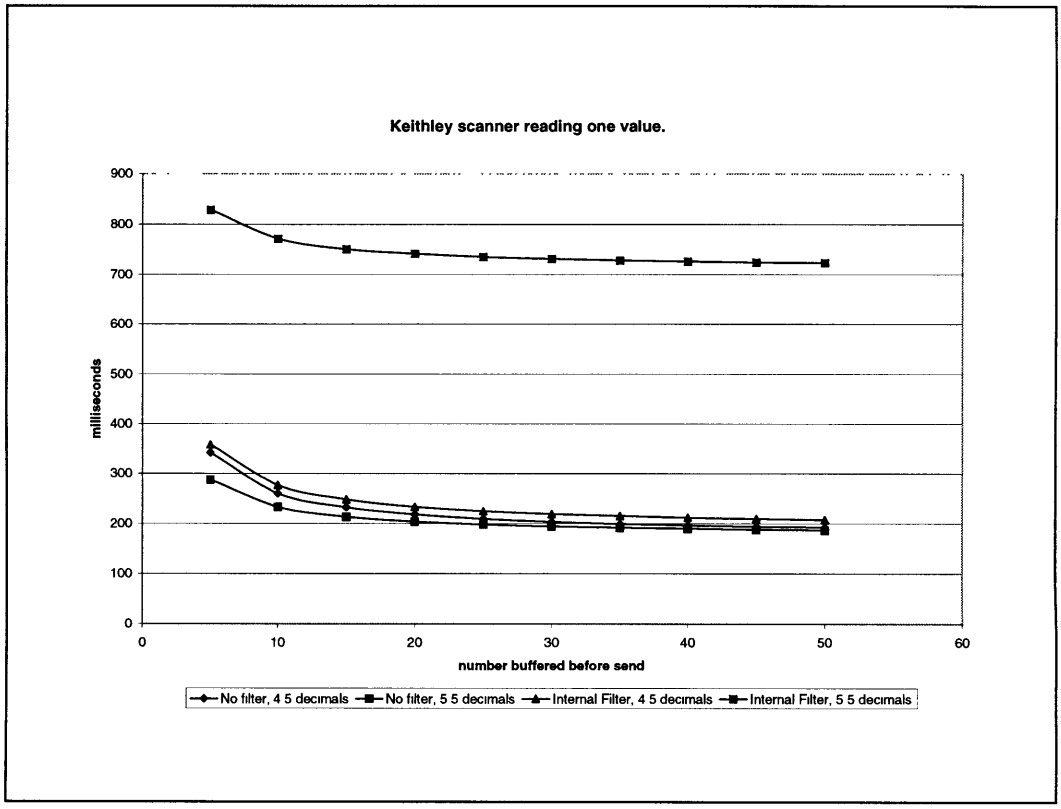

Figure 4.3: Keithley response time for scanning one channel and buffering N reads before sending.

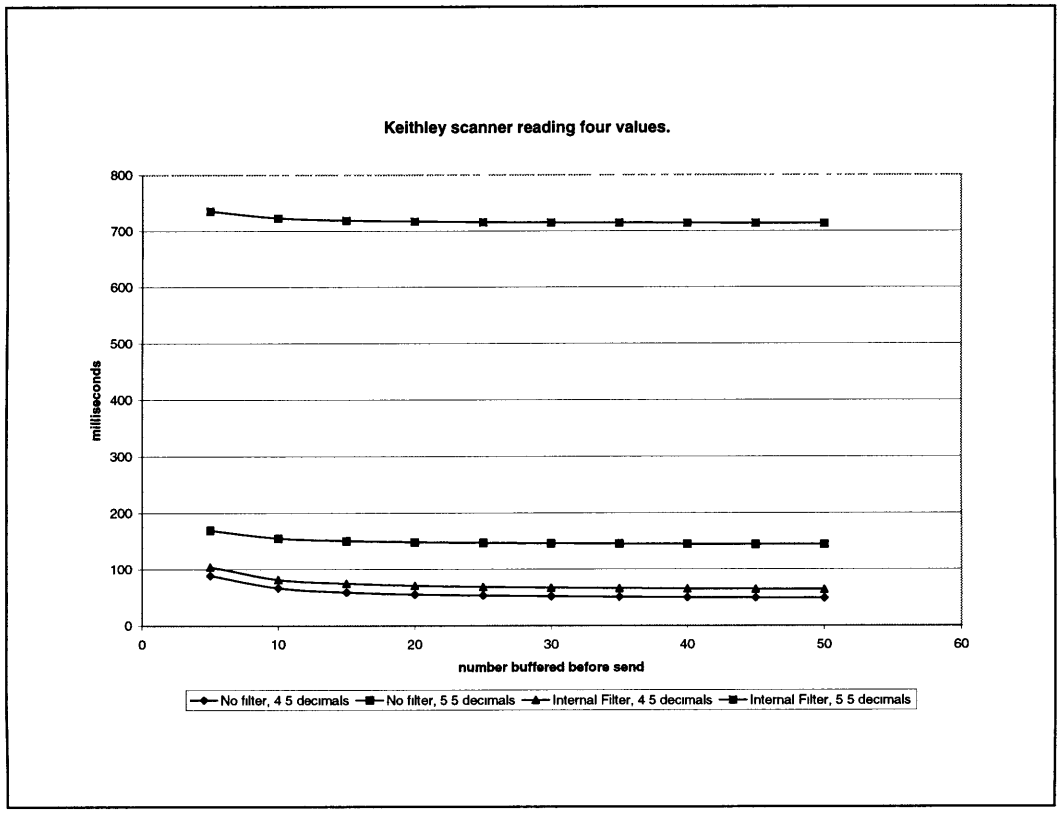

Figure 4.4: Keithley response time for scanning four channels and buffering N reads before sending.

Chapter 5

Conclusion

This thesis has developed and followed into application a distributed method for approaching digital control and monitoring. As the "Digital Age" continues there will be more powerful computers available at trivial prices, and with them the expectation that difficult tasks will be split among multiple computing resources. Gone are the days of the mainframe where all applications must be run on one machine, locally.

The natural division between the concentrator and the observer makes this a very intuitive scheme for devising a distributed monitoring and control system, one that is truly asynchronous and purely event-driven. Certainly, while the Van de Graaff control system displays all the qualities of a concentrator/observer system, the system need not be as simple or as focused. One can envision a large manufacturing plant with all of its controls and displays as concentrators and observers. While still being usable enough for any skilled operator to run, the system could include communication between machines that have traditionally remained isolated, and use an observer-based expert-system to increase throughput. Or, leaving industrial controls altogether, one can easily see how a home or business could profit from being able to externally monitor the hot water and electricity usage all at one display, and use an appropriate control algorithm to prepare accurately prepare for high traffic times (when a lot of hot water or electricity is being used) and to shut down in low traffic times. There are as many applications as there are traditional control systems waiting to be replaced by a modern system. This system's flexibility and expandability make it an ideal choice for controlling large systems, particularly those complex enough to require many computers to perform all of the necessary calculations.

Appendix A

Example source code

Included for completeness are screenshots that exemplify the sourcecode of the concentrator and the observer. Due to LabVIEW's graphical programming structure, it is impractical to include a complete transcript of the code here, the following figures are simply examples. This may enlighten the reader slightly to the double-buffered network connection scheme that both the concentrator and the observer uses. If the reader is unfamiliar with the LabVIEW programming language, he is encouraged to examine manuals at the National Instruments web page, `http://www.natinst.com`.

Figure A.1 shows one state of the block diagram for the observer, and figure A.2 does the same for the concentrator.

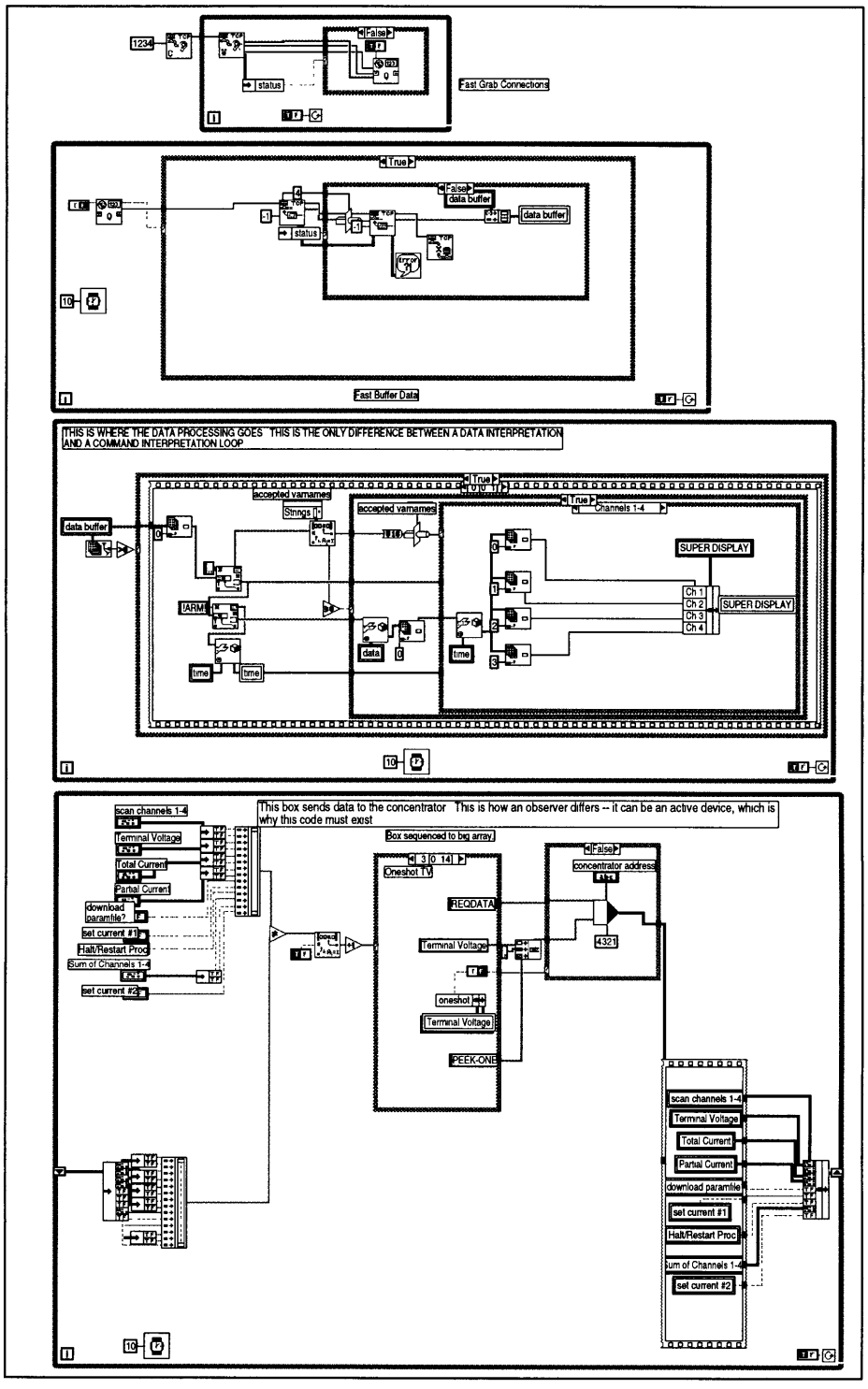

Figure A.1: The block diagram for the observer.

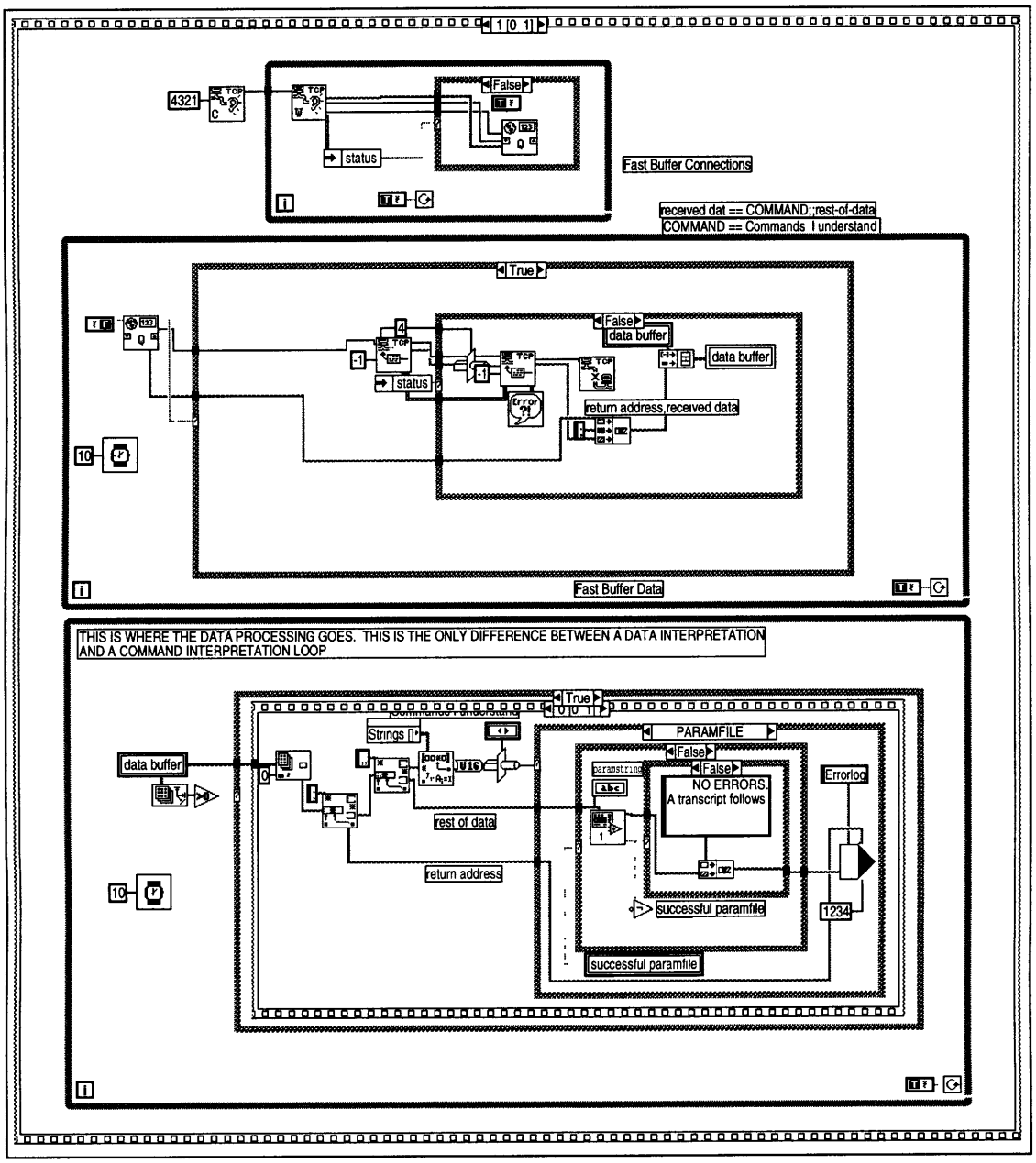

Figure A.2: The block diagram for the concentrator.

Bibliography

[1] Astrom, Wittenmark, *Computer Controlled Systems*

[2] R. Bauman, T. Diaz, et. al, *The Control of the Vivitron*, Presented at the SNEAP Conference (Jülich), 1997

[3] J. G. Trump, K. A. Wright, A. M. Clarke *Distribution of Ionization in Materials Irradiated by Two and Three Million-Volt Cathode Rays*, Journal of Applied Physics, Volume 21, No. 3, pp. 345-348, 1950

[4] Bygrave, Treado, and Lambert. *Accelerator Nuclear Physics — Fundamental Experiments with a Van de Graaff Accelerator*, High Voltage Engineering Corporation, 1970

[5] C. M. Cooke, *Dialectric Diagnosis and Expert Systems*, ISH, 1995

[6] P. Katz, *Digital Control Using Microprocessors*, Prentice-Hall International, 1981

[7] A. S. Tanenbaum, *Modern Operating Systems*, Prentice-Hall Incorporated, 1992