SIMION Basics

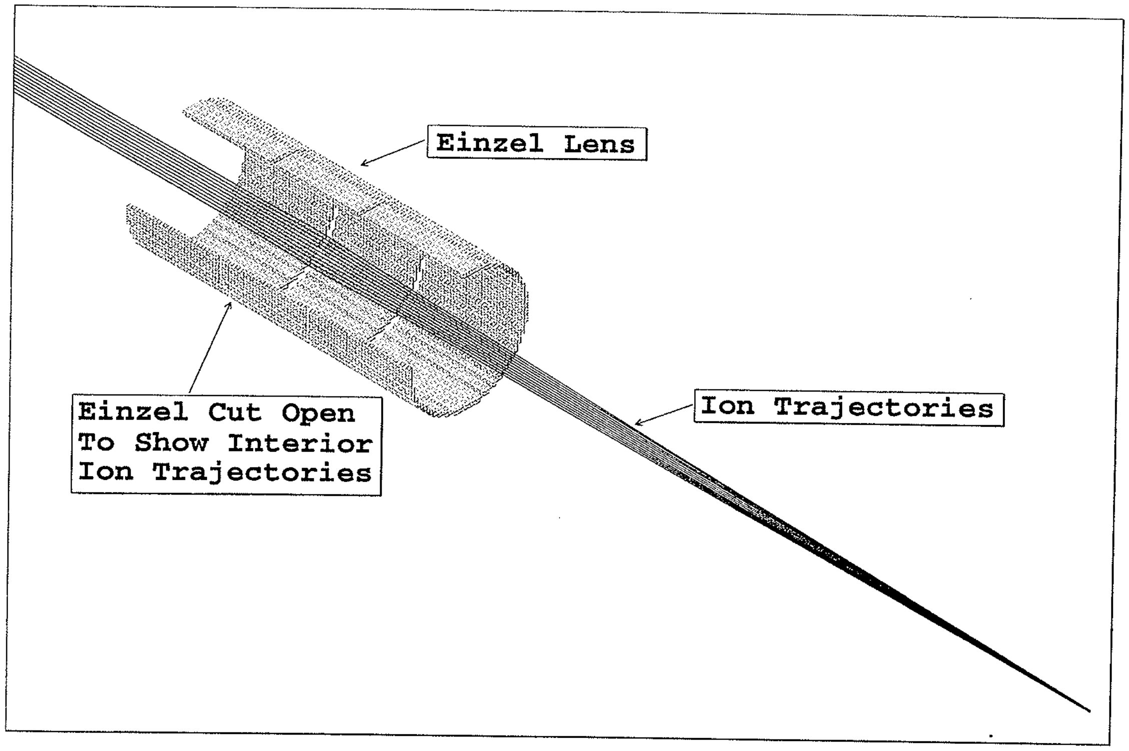


Figure 2-1 Cutaway view of ion trajectories in an einzel lens

Introduction

SIMION makes use of potential arrays that define the geometry and potentials of electrodes and magnetic poles. The potentials of points outside electrodes and poles are determined by solving the Laplace equation by finite difference methods (*see Appendix E - Computational Methods*). In SIMION, this process is called ***refining*** the array. Refined arrays can then be projected as array instances (*3D virtual images*) into an ion optics workbench volume. Ions can be flown within the workbench volume and their trajectories changed by the fields of the potential array instances they fly through.

The illustration above (*figure 2-1*) shows ions flying through a simple einzel lens. The einzel lens has been cut open (*a SIMION display trick*) to show the ion trajectories within the einzel lens. The einzel lens has been created using a simple 2D cylindrical array. Cylindrical 2D arrays form volumes of revolution about their x-axis when refined and displayed in the **View** function.

While this view of the ion trajectories is useful, it doesn't really explain why the ions are focusing. We can use SIMION's potential energy view (*PE View*) of the einzel lens to help us understand the electrostatic focusing process. Figure 2-2 below shows a potential energy view of the einzel lens trajectories above. Note that the potential energy surface is much like the surface of a golf course. Since ions react in much the same way to potential energy surfaces as golf balls react to hills and valleys it is quite easy to see why ions have the trajectories they do in this einzel lens. ***SIMION's capabilities are designed to develop intuition and promote understanding.***

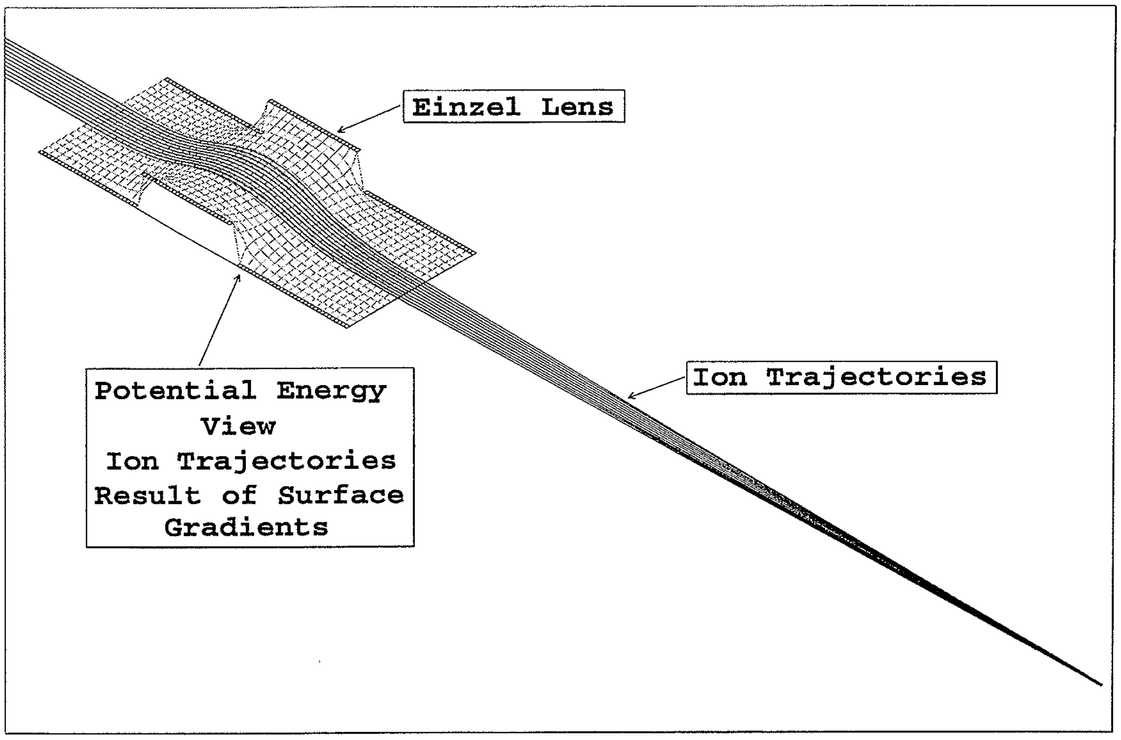


Figure 2-2 Ion trajectories on potential energy surface of einzel lens

Some Basic Ion Optics Concepts

Ion optics utilize the electrostatic and/or the magnetic forces on charged particles to modify ion trajectories. The following is a short (*and simplified*) introduction to ion optics concepts. ***It is intended to introduce rather than educate.***

Equations From Basic Physics:

Force = Mass x Acceleration

$$F = M A$$

Work = Force x Distance

$$\text{Force}_{avg} = \text{Work}/\text{Distance}$$

$$F = dW/dr$$

Coulomb's Law:

$$F_e = k\, Q_i\, Q / r^2 \quad \{\text{Point charge}\}$$

$$F_e = Q_i\, \text{Sum}_n(k\, Q_n / r_n^2) \quad \{\text{Point charges}\}$$

$$Q_i = \text{The Ion's Charge}$$

$$E = F_e / Q_i = d(W/Q_i) / dr \quad \{\text{Electric field intensity}\}$$

$$E = \text{Volts} / \text{Meter}$$

$$\text{Volts} = \text{Joules} / \text{Coulomb}$$

$$F_e = -e\,E \quad \{e = \text{units of positive charge}\}$$

Electrostatic Equation of Motion:

$$A = F / M$$

$$A = dv / dt = -\,(e\,E) / m = -\,E / (m / e)$$

Magnetic Force Equation:

$$F_m = Q_i\,(U \times B) \quad \{\text{Vector Cross Product}\}$$

$$Q_i = \text{The Ion's Charge}$$

$$U = \text{Vector Velocity of Ion}$$

$$B = \text{Vector Magnetic Field Intensity}$$

Magnetic force is always normal to both the B magnetic field vector and the U velocity component normal to the B magnetic field vector (*following the right hand rule*):

$$U \times B = (U_yB_z - U_zB_y)i_x + (U_zB_x - U_xB_z)i_y + (U_xB_y - U_yB_x)i_z$$

A simpler equation results from using ***only*** the vector velocity component normal to the magnetic field:

$$F_m = Q_i\,U_n\,B \quad \{F_m \text{ normal to both } U_n \text{ and } B\}$$

$$U_n = \text{Velocity component normal to B}$$

Static Magnetic Equation of Motion:

$$A_n = F / M$$

$$A_n = (dv/dt)_n = (e\,U_n\,B) / m = U_n\,B / (m / e)$$

Note: Static magnetic fields change an ion's direction of motion but not its speed (*kinetic energy*).

Refraction in Ion Optics:

Refraction (*bending of ion trajectories*) results from electrostatic and/or magnetic forces normal (*at 90 degrees*) to the ion's velocity.

The Electrostatic Radius of Refraction

Normal Electrostatic Force = Centripetal Acceleration

$$- e E_n = m v^2 / r_n$$

$$r_n = m v^2 / (- e E_n) = -(m/e) v^2 / E_n$$

The Magnetic Radius of Refraction

Normal Magnetic Force = Centripetal Acceleration

$$B_n e v = m v^2 / r_n$$

$$r_n = m v / (B_n e) = (m/e) v / B_n$$

Interpretations of Radius of Refraction:

1. The electrostatic radius of refraction is proportional to the ion's kinetic energy per unit charge. Thus all ions with the same starting location, direction, and kinetic energy per unit charge will have identical trajectories in electrostatic (*only*) fields. ***Trajectories <u>are not</u> mass dependent in electrostatic fields.***

2. The magnetic radius of refraction is proportional to the ion's momentum per unit charge. Thus all ions with the same starting location, direction, and momentum per unit charge will have the identical trajectories in static magnetic (*only*) fields. ***Trajectories <u>are</u> mass dependent in static magnetic fields.***

3. Because of the v verses v^2 effects on the radius of refraction, ***magnetic ion lenses have superior refractive power at high ion velocities.***

Light Optics Verses Ion Optics

There are significant differences between light and ion optics:

Radius of Refraction

Light optics make use of sharp transitions of light velocity (*e.g. lens edges*) to refract light. These are very sharp and well defined (*by lens shape*) transitions. The radius of refraction is infinite everywhere (*straight lines*) except at transition boundaries where it approaches zero (*sharp bends*).

Ion optics makes use of electric field intensity and charged particle motion in magnetic fields to refract ion trajectories. This is a distributed effect resulting in gradual changes in the radius of refraction. ***Desired electrostatic/magnetic field shapes are much harder to determine and create since they result from complex interactions of electrode/pole shapes, spacing, potentials and can be modified significantly by space charge.***

Energy (Chromatic) Spreads

Visible light varies in energy by less than a factor of two.

Ions can vary in initial relative energies (*or momentum for magnetic*) by orders of magnitude. ***This is why strong initial accelerations are often applied to ions to reduce the relative energy spread.***

Physical Modeling

Light optics can be modeled using physical optics benches (*interior beam shapes can be seen with smoke, screens, or sensors*).

Ion optics hardware is generally internally ***inaccessible*** and must normally be evaluated via end to end measurements. ***Numerical simulation programs like SIMION allow the user to create a virtual ion optics bench and look inside much like physical light optics benches.***

Determining Field Potentials

The electrostatic or magnetic field potential (*e.g. Volts or Mags - SIMION's magnetic potentials*) at any point within an electrostatic or static magnetic lens can be found by solving the Laplace equation with the electrodes (*or poles*) acting as boundary conditions. The Laplace equation assumes that there are ***no*** space charge effects.

The Laplace Equation

$$\mathbf{DEL^2\ V = DEL \cdot DEL\ V = 0}$$

$$\mathbf{DEL\ V = (dV / dx)_i + (dV / dy)_j + (dV / dz)_k = E}$$

$$\mathbf{DEL^2\ V = DEL \cdot E = dE_x / dx + dE_y / dy + dE_z / dz = 0}$$

The Laplace equation constrains all electrostatic and static magnetic potential fields to conform to a zero charge volume density assumption (*no space charge*). ***This is the equation that SIMION uses for computing electrostatic and static magnetic potential fields.***

Poisson's Equation Allows Space Charge

$$\mathbf{DEL^2\ V = DEL \cdot DEL\ V = -\ p / e}$$

Poisson's equation allows a non-zero charge volume density (*space charge*). When the density of ions becomes great enough (*high beam currents*) they will (*by their presence*) significantly distort the electrostatic potential fields. In these conditions, Poisson's equation should be used (*instead of Laplace's*) to estimate potential fields. ***SIMION does not support Poisson solutions to field equations. It does however employ charge repulsion methods that can estimate certain types of space charge and particle repulsion effects.***

The Nature of Solutions to the Laplace Equation

The Laplace equation really defines the electrostatic or static magnetic potential of any point in space in terms of the potentials of surrounding points. For example, in a 2-dimensional electrostatic field represented by a very fine mesh of points the Laplace equation is satisfied (*to a good approximation*) when the electrostatic potential of any point is ***estimated*** as the average of its four nearest neighbor points:

$$V = (V_1 + V_2 + V_3 + V_4) / 4$$

Physical Models of the Laplace Equation Solutions

Physical models (*as opposed to numerical models*) have historically been used to solve many Laplace equation problems. These models have the advantage of giving physical insight into problems that can be difficult to understand.

Rubber-Sheet Models

Laplace equation solutions for electrostatic potential fields resemble surfaces of rubber-sheets stretched between electrodes where electrode height represents electrostatic potential. In fact, relatively flat rubber-sheet surfaces are good physical representations for electrostatic fields. Rubber-sheet models and marbles (*for ions*) have been used to model electrostatic fields (*e.g. early vacuum tube design*).

Resistance Paper Models

Resistance paper models have also been used. In this method electrodes of the designed shapes are pressed against paper having a uniform surface resistivity. Voltages are applied to the electrodes and electrostatic potential measurements are taken via a probe connected to a voltmeter. These measurements are then used to calculate ion trajectories.

Pros and Cons of Rubber-Sheet Models

Rubber-sheet models give the designer excellent insight into the behavior of ions in electrostatic fields. This is because we humans have each developed a certain physical intuition for the movement of balls on sloping surfaces (*e.g. miniature golf*). ***Note: SIMION's potential energy surfaces (figure 2-2) use the rubber-sheet style of display to assist the user in developing intuition and understanding.***

Unfortunately rubber-sheet models are difficult to build. Moreover, they are limited in their modeling accuracy because forces are proportional to the sine of the slope (*e.g. g dh/dr*) rather than the tangent of the slope (*e.g. dV/dx*) needed in electrostatic ion optics.

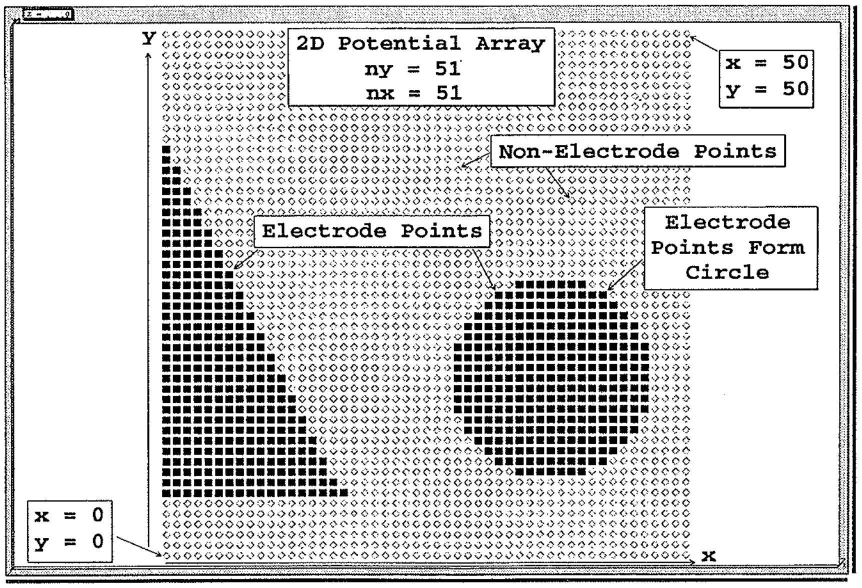


Figure 2-3 Example of a potential array defined by electrode and non-electrode points

The Potential Array

SIMION utilizes potential arrays to define electrostatic and magnetic fields. ***A potential array can be either electrostatic or magnetic <u>but not both</u>***. If you require both electrostatic ***and*** magnetic fields in the same volume, array instances of electrostatic ***and*** magnetic arrays must be ***superimposed*** in the workbench.

A potential array is an array of points organized so the points form equally-spaced ***square*** (*2D*) or ***cubic*** (*3D*) grids. ***Equally-spaced means that all points are equal distances from their nearest neighbor points.*** Figure 2-3 (*above*) shows a **Modify** function view of a 51 by 51 2D potential array.

All points have a potential (*e.g. voltage*) and a type (*e.g. electrode or non-electrode*). Certain points are flagged as electrode or pole points. Points of type electrode/pole create the boundary conditions for the array. ***Groups of electrode points create the electrode and pole shapes (the finer the grid the smoother the shapes).*** Non-electrode and non-pole points within the array represent points outside electrodes and poles. We need to solve the Laplace equation to obtain the potentials of these points (*the non-electrode and non-pole points).*

Potential Array Dimensions

Potential arrays are dimensioned by the number of points in each dimension (*x, y, and z*). Therefore an array of nx = 51 would have 51 points in the positive x direction. All arrays have their lower left corner at the origin (*xmin = ymin = zmin = 0*). Thus if nx equals 51, xmin equals 0 and xmax equals 50 (*a **width** in **x** of **50** grid units*).

All 2D arrays have nz = 1 (*zmin = zmax = 0*). Thus 2D arrays are always located on the z = 0 xy-plane.

Arrays can use a lot of memory. A 100x by 100y by 100z 3D potential array requires one million points. Each point requires 10 bytes of RAM. ***Thus 10 megabytes of RAM are required for each million array points.***

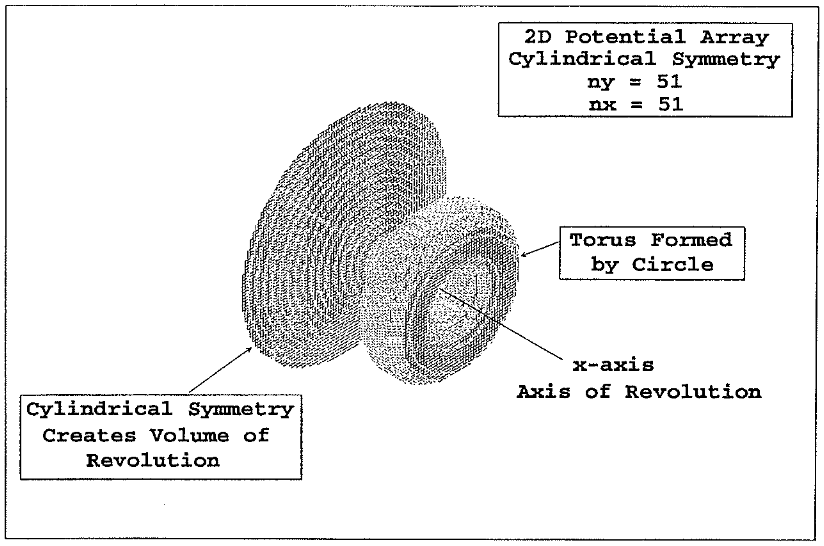


Figure 2-4 2D Potential array with cylindrical symmetry generates a volume of revolution about the x-axis

Potential Array Symmetries

Two potential array symmetries are supported: Planar (*2D and 3D arrays*) and cylindrical (*2D arrays only*). When SIMION refines (*solves the Laplace equation*) or projects an array instance (*3D image*) of an array into its ion optics workbench volume, it uses the array's symmetry. If the 2D array has a cylindrical symmetry it will be projected to create a volume of revolution about its x-axis. Figure 2-4 (*above*) shows an isometric 3D view of the array in figure 2-3 assuming it has cylindrical symmetry. ***Note: The cylindrical 2D array is visualized as if it were 3D planar to take advantage of fast planar visualization methods. However, apertures are really round*** *(despite how they appear) and the* *__cylindrical symmetry is fully retained for ion trajectory calculations__*.

Figure 2-5 (*below*) shows an isometric 3D view of the array in figure 2-3 assuming it has planar symmetry:

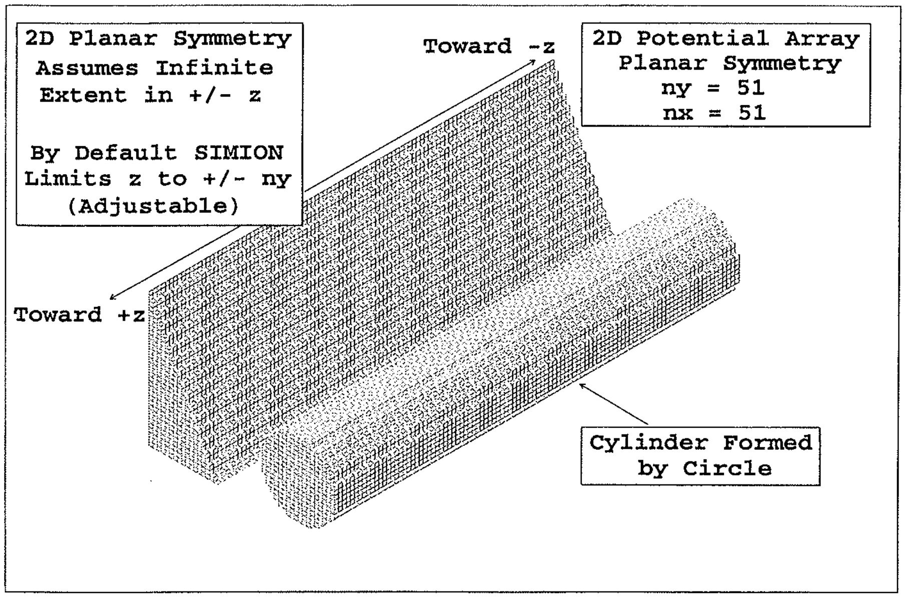


Figure 2-5 2D Potential array with planar symmetry generates a planar symmetrical volume

Note that 2D planar symmetry assumes that the array ***itself*** is only one layer of an infinite number of layers in z (*for **refining** purposes*). When SIMION projects a 2D planar array as an array instance within the workbench volume it ***assumes*** a z depth of +/- ny (*grid units*). ***You may edit the array instance definition to __increase__ this z depth***. This option allows a 2D planar array to be used to represent the interior (*symmetrical*) portions of objects like quadrupole rods (*saves array space*).

The Use of Array Mirroring

Many electrode designs have a natural mirrored symmetry. Designs that mirror in y have a mirror image of the electrode/poles in the negative y. SIMION supports three mirroring symmetries: x, y, and z. Mirroring allows you to use a ***smaller*** array to model a larger area (*or volume*) when conditions permit. For example: A 3D planar array can be mirrored in x, y, and/or z. If the design symmetries permit, this would allow the modeling of a 3D volume with one eighth the number of points required if no mirroring were utilized. ***SIMION takes mirroring into account when refining arrays and projecting their instances into the workbench volume***.

X mirroring is ***allowed*** for all 2D and 3D arrays. Y mirroring is ***required*** of 2D cylindrical and ***allowed*** in 2D and 3D planar arrays. ***Z mirroring is __only__ allowed in 3D planar arrays***.

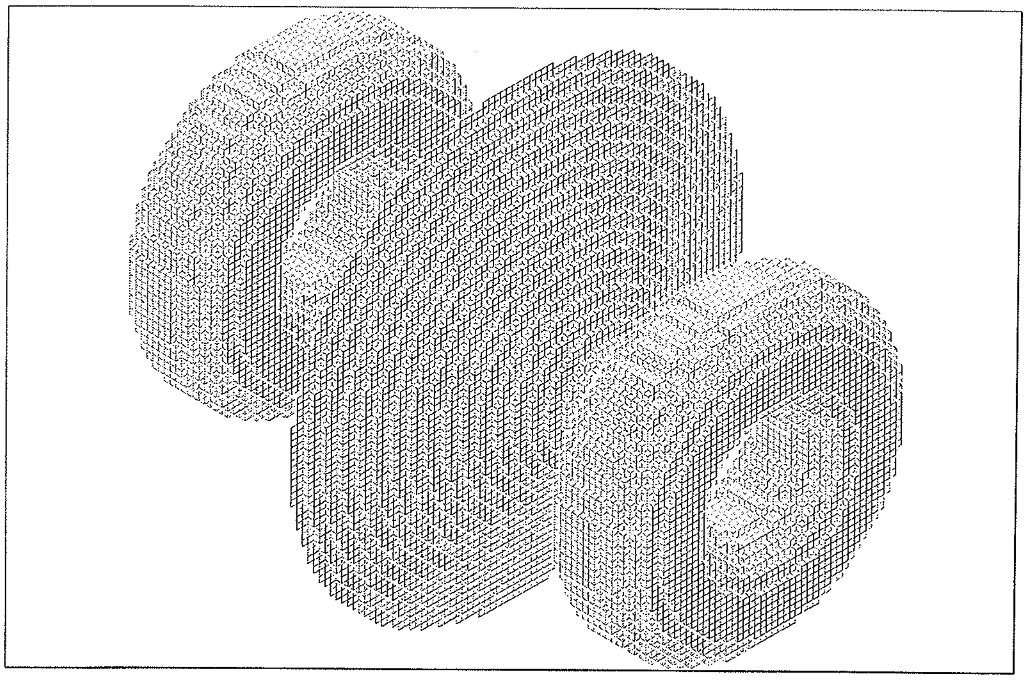


Figure 2-6 Array shown in Figure 2-3 projected as cylindrical symmetry mirrored in both x and y

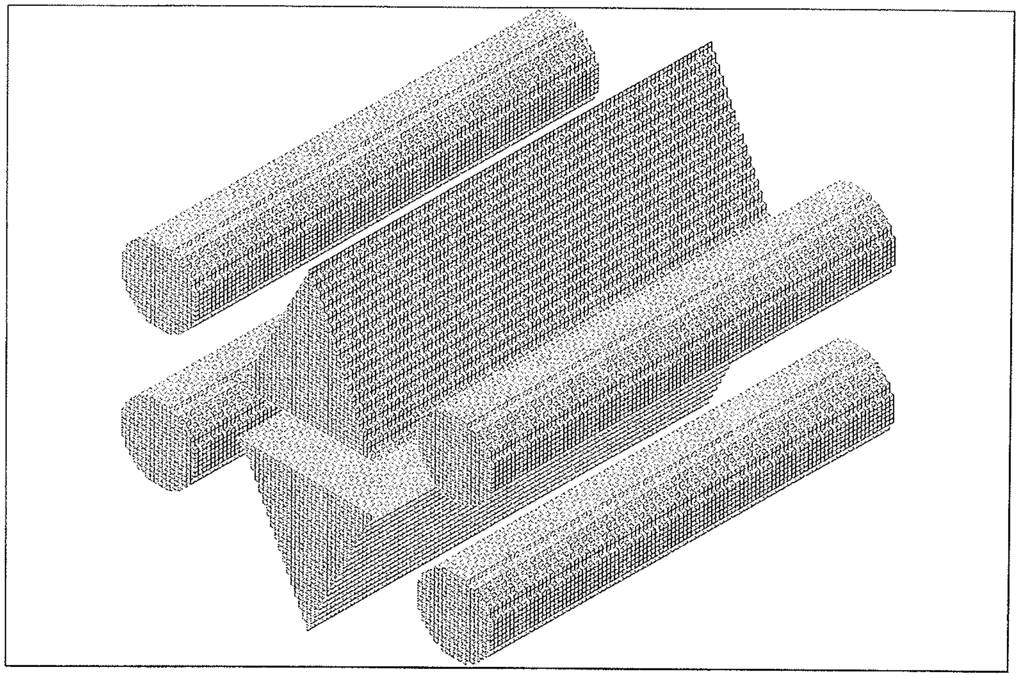


Figure 2-7 Array shown in Figure 2-3 projected as planar symmetry mirrored in both x and y

Potentials and Gradients in Potential Arrays

SIMION uses the same finite difference methods for refining either electrostatic or magnetic potential arrays. However, the definitions for potentials and gradients used are different and it is **important** that you understand these differences.

Potentials and Gradients in Electrostatic Arrays

Potentials in electrostatic arrays are ***always*** in volts. SIMION uses the refined array potentials to determine field gradients (*voltage gradients*). Electrostatic field gradients are ***always*** in volts/mm. When an array instance of a potential array is projected into the workbench volume, it is scaled by a user specified number of millimeters per grid unit (*or defaulted to 1 mm/grid unit*). So although electrostatic gradients start out as volts/grid unit they are divided by the instance scaling factor to obtain volts/mm:

Gradient = PA Gradient / Instance Scaling

Volts / mm = (Volts / grid unit) / (mm / grid unit)

Thus array instance scaling can have a dramatic impact on electrostatic gradients and therefore ion accelerations.

Potentials and Gradients in Magnetic Arrays

SIMION is not a magnetic circuit program. You have to supply the magnetic potentials for it to refine. Unlike electrostatics, in magnetics we normally think of ***and measure*** gradients or more precisely flux (*gauss*) as opposed to potentials. This presents a problem, because SIMION needs magnetic potentials to refine.

In order to deal with this dilemma SIMION defines magnetic potentials in Mags. ***Mags*** are defined to be ***gauss times grid units.*** Thus the gradient of magnetic potential is simply gauss. *Note:* ***Mags are gauss times grid units instead of gauss times millimeters***. If Mags were gauss times millimeters then instance scaling would confound us further. With this approach, magnetic fields remain the same whether the array is scaled to be a certain size or 10 times as large in the workbench volume. ***Thus instance scaling has no impact on the magnetic fields (flux in gauss) produced by magnetic potential arrays***.

SIMION also makes use of a magnetic scaling factor **ng** as a property of magnetic potential arrays. ***The* ng *scaling factor has been provided to further simplify your life***. It would often be very nice for Mags to be directly related to gauss. Let's say we have a simple two pole magnet. We would like to set one pole to 1000 Mags and the other to Zero Mags and have the field in between be approximately 1000 gauss. If the two poles are separated by let's say a 60 grid unit pole gap we would specify the value of 60 for **ng** to automatically scale the Mags potentials ***roughly*** into gauss.

PA Magnetic Flux = PA Magnetic Gradient (PA gauss)
Magnetic Flux = PA Magnetic Flux * ng

gauss = PA gauss * ng (pole gap scaling factor)

Note: The B field vector ***always*** points from greater magnetic potential (*e.g. 1000 Mags*) toward lesser magnetic potential (*e.g. 0 Mags*).

Beware! Magnetic potentials are not as simple as electrostatic potentials. While we can safely assume that all points of an electrode have the same electrostatic potential (*e.g. volts*), it is dangerous to assume that the same is generally true for magnetic poles. ***Magnetic poles do not as a rule have totally uniform magnetic potentials across their surfaces*** (*permeability not* *being infinite*). Thus you must allow for this fact if the effects could be significant enough to impact your results. ***Remember:*** ***User beware***.

The Two Classes of Potential Arrays

User defined potential arrays come in two classes the Basic potential array (*.PA file extension*) and the Fast Adjust Definition array(*.PA# file extension*).

The Basic potential array has its electrode/pole potentials defined as described above. These arrays are then processed by the **Refine** function to solve for the non-electrode and non-pole potentials. The Basic array is always saved with a **.PA** file extension. An array's file extension is ***important*** because SIMION assumes that all arrays with a **.PA** file extension are Basic potential arrays for refining, adjusting, and viewing purposes.

The Fast Adjust Definition array is much like the Basic potential array except the electrode potentials are not explicitly defined. Instead, all the points defining electrode number one are set to a potential of exactly 1 volt/Mag. Thus the Fast Adjust Definition array acts as a template to allow the **Refine** function to automatically create the proper separate solution arrays to support fast voltage adjustment of up to ***30*** electrodes/poles. All Fast Adjust Definition arrays must be explicitly saved with a **.PA#** file extension so that SIMION will recognize and process them properly.

Three Ways to Adjust Array Potentials

We often need to change the electrode/pole potentials of an already refined array to tune a lens or adjust a magnet's field. SIMION supports three different strategies:

Modify the Potential(s) and Re-Refine

The first strategy is the brute force approach:

1. Use the **Modify** function to change the potentials of the points of one or more electrodes or poles in the Basic potential array (*.PA*).
2. The use the **Refine** function to re-refine the Basic potential array (*.PA*) to obtain the resulting potentials of the non-electrode or non-pole points.

This is the **Modify-Refine** cycle of array potential adjustment. It is time consuming and invites errors (*e.g. accidentally not changing the potentials of* ***all*** *the points defining an electrode or pole*).

Proportional Re-scaling of All Basic Array Potentials (.PA)

SIMION also supports proportional re-scaling of ***all*** Basic potential array (*.PA* ***arrays***) potentials. This is useful in those cases when the potentials of all electrodes or poles can be changed proportionally to obtain the desired result. ***This approach works with any Basic potential array (with .PA file extension) that has already been refined:***

1. Use the **Fast Adjust** function to change the potential of the highest **absolute** potential electrode or pole in the potential array (*displayed as electrode* ***0***). The ***only*** illegal proportional scaling potential is **0**, because a re-scaling potential of zero would cause ***all*** the potentials of ***all*** points in the array to be zeroed, and thus the refined field solutions would be lost (*irreversible scaling*).
2. SIMION will automatically scale the potentials of all array points (*electrode/pole and non*) by the same proportion that you changed the potential of the highest potential electrode or pole.

This can be quite useful for a magnetic potential array with two poles and a gap. Proportional scaling provides a quick way to adjust the magnetic field. It even supports proportional scaling of non-uniform potentials on the pole surfaces (*fringe field effects*). ***This is a clever trick only if you can justify that these non-uniform pole surface potentials actually would scale proportionally in your problem.***

Using Fast Adjust Definition Arrays (.PA# arrays)

The third approach supported by SIMION makes use of the additive solution property of the Laplace equation. This involves creating a separate array for each electrode we desire to adjust, setting the points of the desired electrode/pole to a fixed potential, and setting all other electrode/pole points to zero. Each of these separate electrode arrays is then refined, and each refined electrode solution array is then saved to disk.

Composite fields are obtained by scaling each electrode's solution array to its desired voltage and adding the individual field contributions of the various adjustable electrodes together to obtain the desired result. This is all quite fast (*avoids re-refining*) if you can keep track of all the book-work. ***Fortunately, SIMION is designed to do all the hard work for you!***

1. Use the **Modify** function to define the geometry of your electrodes. ***All points of the first adjustable electrode (pole) must be set to exactly 1 volt (Mag).*** Likewise, all points of the ***second*** must be set to 2, and so on. ***Adjustable electrode/pole potentials must not be skipped (e.g. 1,2,4,5).*** Up to **30** adjustable electrodes can be defined in this manner within a potential array. Non-adjustable electrodes or poles may also be defined providing they do not have the ***exact*** potentials from 1 to 30 (*e.g. 10.0001 volts is OK*).

2. Save this potential array to disk with a **.PA#** file extension to signal to SIMION that this is a Fast Adjust Definition File (*e.g. save as **TEST.PA#***).

3. Refine the **.PA#** file. SIMION will ***recognize*** that this is a Fast Adjust Definition File, examine it, create each required electrode solution array, refine it, and save it to your disk automatically.

   SIMION first creates a base solution array **.PA0** (*e.g.* **TEST.PA0**). This array contains the potentials of all non-fast adjustable electrodes/poles. The **.PA0** array is called the Fast Adjust array because this is the array you ***actually*** load and fast adjust. If non-zero potential non-fast adjustable electrode points are found (*e.g. 10.56 volts*), SIMION automatically creates and refines a fast scaling solution file for these points called **.PA_**. This allows you to proportionally re-scale these electrode points as electrode 0 in a manner similar to that described above except a 0 proportional scaling potential is allowed, because the solution is preserved in the **.PA_** solution file.

   The individual fast adjust electrode solution arrays have the extension **.PA1** - **.PA9** and **.PAA** - **PAU** as required by the number of adjustable electrodes defined (*electrode one is stored in a* **.PA1** *file, e.g.* **TEST.PA1**).

4. Use the **Fast Adjust** function on the **.PA0** file to set potentials. Fast adjust electrodes will be numbered from **1-30** and the fast proportional scaling electrode points (*if a **.PA_** file was automatically created*) will be changed via electrode **0**. If you try to fast adjust a **.PA#** file SIMION will automatically load its **.PA0** file for adjustment. ***If you want to save the current potential settings of a .PA0 file between SIMION sessions, simply save the .PA0 file to disk.***

The **Fast Adjust** function is accessible from the Main Menu Screen or from within the **View** function (*even while ions are flying*).

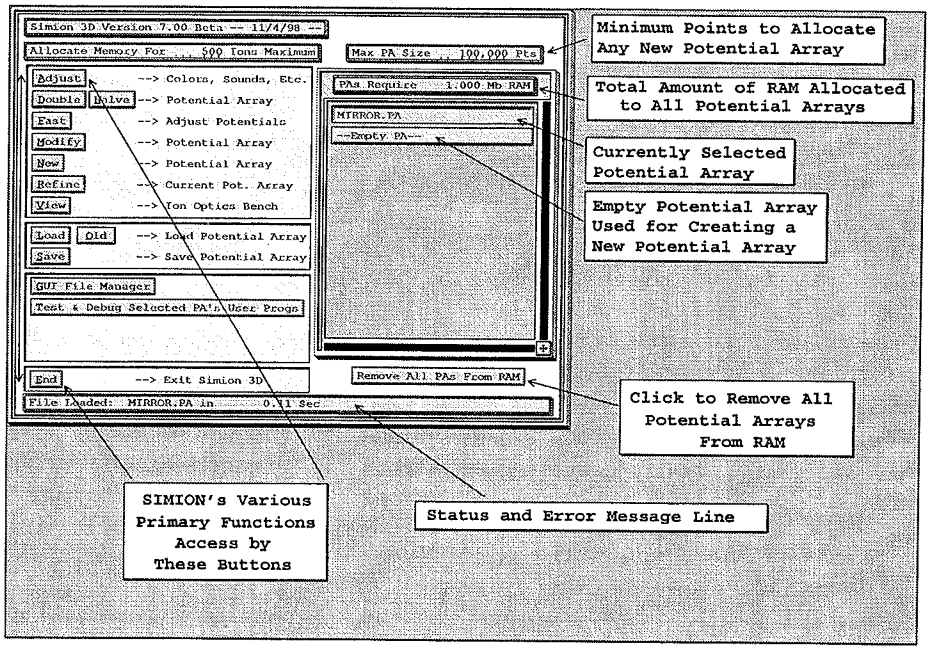

Simion 3D Version 7.00 Beta -- 11/4/98 --
Allocate Memory For ... 500 Ions Maximum
Max PA Size ... 100,000 Pts
Minimum Points to Allocate Any New Potential Array
Adjust --> Colors, Sounds, Etc.
Double Halve --> Potential Array
Fast --> Adjust Potentials
Modify --> Potential Array
New --> Potential Array
Refine --> Current Pot. Array
View --> Ion Optics Bench
Load Old --> Load Potential Array
Save --> Save Potential Array
GUI File Manager
Test & Debug Selected PA's User Progs
End --> Exit Simion 3D
File Loaded: MIRROR.PA in 0.11 Sec
PAs Require 1.000 Mb RAM
Total Amount of RAM Allocated to All Potential Arrays
MIRROR.PA
--Empty PA--
Currently Selected Potential Array
Empty Potential Array Used for Creating a New Potential Array
Remove All PAs From RAM
Click to Remove All Potential Arrays From RAM
SIMION's Various Primary Functions Access by These Buttons
Status and Error Message Line

Figure 2-8 An Illustration of SIMION's Main Menu Screen with various key features annotated

The SIMION Main Menu

Figure 2-8 (*above*) shows the SIMION Main Menu Screen. This is the first screen you will see in SIMION. It serves as the point of departure for all SIMION adventures. The buttons on the left allow you to access the various primary functions (*e.g.* **Modify**). ***Notice that the first letter in each button's label is underlined***. This means that you can access a particular button by ***either*** clicking on it with the mouse ***or*** by entering the underlined key from the keyboard (*e.g.* **m** *for* **Modify**).

The window object on the right contains the list of potential arrays currently in RAM (*in this case, one:* **MIRROR.PA**). Other objects (*buttons and etc.*) are used to display and control potential array: data, parameters, and fate (*more of this later*).

Adjusting User Preferences

The **Adjust** button allows you to adjust various SIMION and GUI characteristics. Clicking the **Adjust** button brings up a screen that allows you to change colors, sounds, blinking, other features (*e.g. mouse speed*), and certain video options. ***If you haven't explored* Adjust*, do it now***. Descriptions of all options can be found in the GUI discussion in Appendix F.

The **Video Adjust** button allows you to select/change the displayed screen font, label positions within buttons, visibility of special characters, and a fix button for a common Windows driver bug associated with xored rectangles (*see Appendix B*).

There may be occasions when you want SIMION just to redraw its entire window. To redraw SIMION's window hold down the **Crtl** or **Alt** key and hit the **V** key (**<Ctrl V>** *or* **<Alt V>**) and SIMION will regenerate its current view (*e.g. color palette and current view*).

The Role of the Subdirectory in SIMION Projects

SIMION requires that all files relating to a project (e.g. ***.PA, *.IOB, *.FLY,** *and etc.) are contained in the same subdirectory of your hard disk.* This is useful because it keeps things together and in so doing forces a touch of organization that many of us need so badly. Note: There can be more than one project in a subdirectory. However, this can often create a lot of clutter.

Each of the demos provided with SIMION is in its own directory below **\SIM7**. These demos help show how various projects might be approached.

The GUI File Manager can be used to quickly create subdirectories. ***Click on the desired drive, click on the desired parent directory, click the*** **Other** ***Button, enter the name of the new subdirectory in the Insert Directory ioline object, and press*** **Enter**. The subdirectory is created. Now click on this subdirectory to make it the currently active directory and click **Return** to exit the file manager. ***Note: The GUI File Manager can also be used to copy or move files between directories and drives (Appendix F).***

Potential Arrays and RAM

SIMION maintains a ***working copy*** of each active potential array (*up to 200*) in memory (*RAM*). ***When you change something in a potential array you are only changing the in-memory copy.*** Likewise when you fly ions through instances of potential arrays in the workbench you are using the ***in-memory*** copies of the potential arrays.

User defined potential arrays are normally saved as **.PA** or **.PA#** (*extension*) files in your project directory. When you create a new potential array ***only the in-memory copy is created.*** It is your responsibility to save any new or changed potential arrays to your project directory as required. ***Saving potential arrays preserves your work between sessions***.

Allocating Memory for Potential Arrays

Potential arrays can use up a lot of memory. Each point of a potential array requires 10 bytes of RAM storage. Thus a 100 x by 100 y by 100 z 3D array has 1,000,000 points and requires 10 megabytes of RAM. ***SIMION allocates memory only once for each PA memory region it creates. Once the PA memory has been allocated it is not returned or changed until the*** **Remove All PAs from RAM** ***button is used to remove all memory allocated PAs (de-fragment the heap).*** This is done to prevent heap fragmentation lockups due to the large size of typical potential arrays.

SIMION normally allocates 100,000 points of memory for each new potential array. This is usually large enough to allow you to increase the array size (*e.g. via* **Modify** *or* **Double**) without exceeding the memory allocated for a PA. ***You have the option of adjusting this default PA memory allocation size up or down to suit your needs (with the*** **Max PA Size** ***panel object) before you create or load a potential array in an empty PA memory region***. The **maximum** size allowed for any single potential array in SIMION 7.0 is ***50 million points***.

Deallocating Potential Array Memory

The Main Menu screen has a **Remove All PAs From RAM** button. This button is used to remove all working copies of PAs from RAM. ***Its primary function is to restore a clean slate when you are about to start a new project or the clutter of PAs has become unmanageable.***

The List of Potential Arrays

The Main Menu Screen (*Figure 2-8*) contains a window with a button for each currently allocated PA memory region and the name of the potential arrays in each (*long file names are supported - however a long file name may be displayed truncated to fit the button's width*). One of these buttons will be ***depressed***. This button selects the ***currently active potential array***. This is the array that will be acted upon by functions like: **Load, Save, Modify, Fast Adjust, Refine, Double,** and etc.

A particular potential array is selected by clicking (depressing) its button.

The last button in the list is *always* marked **empty PA.** ***This* empty PA *is available for allocating memory for a new potential array*.** Up to 200 potential arrays can be loaded in RAM at one time.

Creating, Refining, and Flying Ions in Your First Potential Array

The following material is an illustrated step-by-step example of how to create an electrostatic potential array and use it to fly ions. It is recommended that you use SIMION to follow along with the example below. ***The best way to learn SIMION is to use it.***

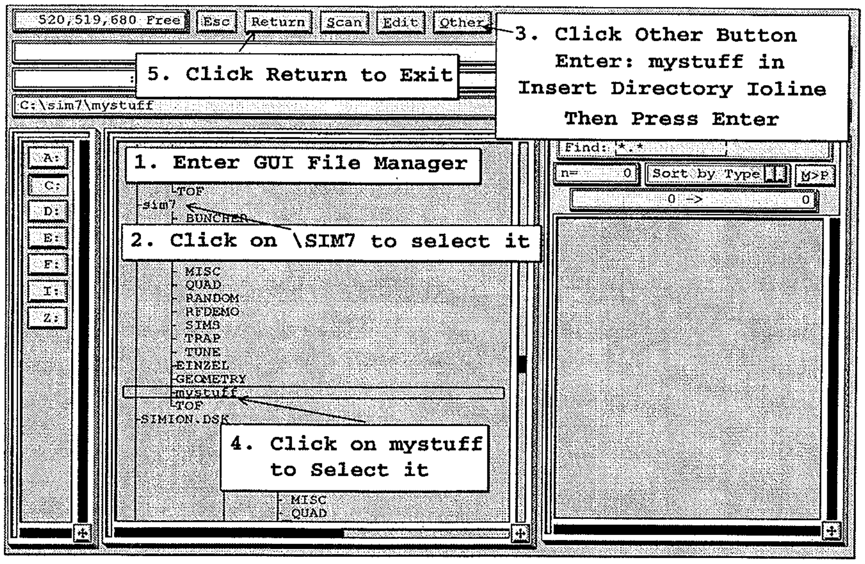

520,519,680 Free
Esc
Return
Scan
Edit
Other
3. Click Other Button Enter: mystuff in Insert Directory Ioline Then Press Enter
5. Click Return to Exit
C:\sim7\mystuff
1. Enter GUI File Manager
2. Click on \SIM7 to select it
4. Click on mystuff to Select it
Find: *.*
Sort by Type
M>P

Figure 2-9 Steps in creating a project directory

Creating a Project Directory

The first step in starting any new SIMION project is to create a project directory to hold all the files you'll create. Remember, SIMION expects that all project files reside in the *same* directory. ***You ignore this rule at your peril!***

Let's create a new directory called **mystuff** that is directly below the **\SIM7** installation directory. Start SIMION. Now, starting at the Main Menu Screen, use the following steps to create the new project directory (*Figure 2-9 above*):

1. Click on the **GUI File Manager** button.

2. Click on the **\SIM7** directory (*middle window*) to make it the current directory if it is not already selected (*marked*).

3. Click the **Other** button and enter **mystuff** in the **Insert Directory** ioline and press **Enter**. SIMION will create the **mystuff** directory beneath the **\SIM7** directory.

4. Click on the **mystuff** directory to make it the ***current*** directory.

5. Click the **Return** button to exit the GUI File Manager.

Controlling Automatic Directory Scanning

SIMION stores images of each drive's directory tree in files so it won't have to rescan the drive for directories each time you access it with the GUI File Manager. To insure that the directory is complete and correct, SIMION ***automatically*** scans each drive for directories the ***first time it is accessed*** in any program session.

This approach assumes that you have added or deleted directories between SIMION sessions. If this is not the case, the automatic scan feature wastes time (particularly *on large drives*), because nothing has changed.

You have the option of turning automatic directory scanning off. However, ***you*** are then ***responsible*** for clicking the **Scan** button in the GUI File Manager to update a directory tree that may be incorrect (*Figure* 2-9). The following procedure turns auto-scanning off (*or back on*):

1. Click the **Adjust** button on the Main Menu Screen.
2. Click the **Other Adjust** button to access the Other Preferences Screen.
3. Click the **Scan On** button. It should now display **Scan OFF**.
4. Click the **OK** button.

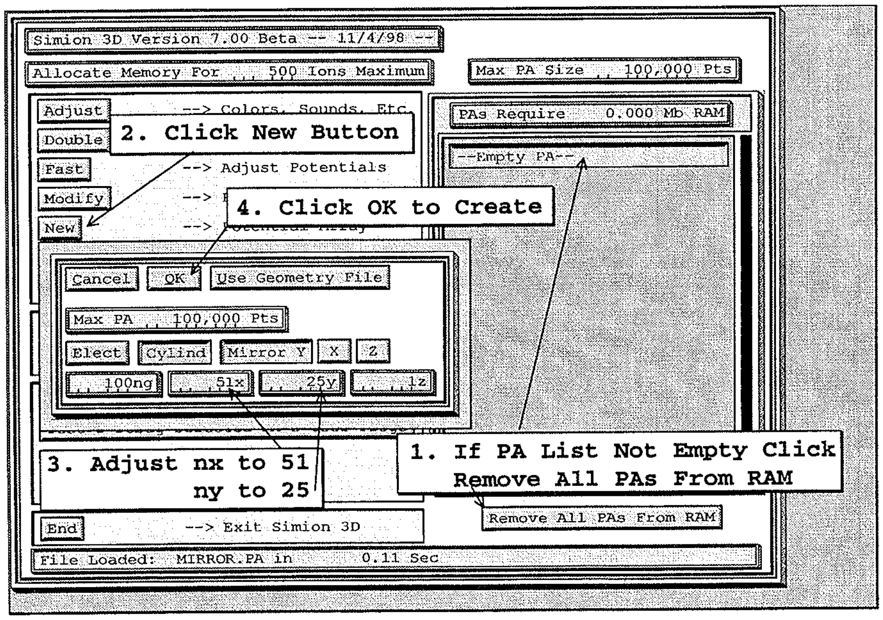

Simion 3D Version 7.00 Beta -- 11/4/98 --
Allocate Memory For 500 Ions Maximum
Max PA Size 100,000 Pts
Adjust --> Colors, Sounds, Etc.
Double
Fast --> Adjust Potentials
Modify -->
New -->
PAs Require 0.000 Mb RAM
--Empty PA--
Cancel
OK
Use Geometry File
Max PA 100,000 Pts
Elect
Cylind
Mirror Y
X
Z
100ng
51x
25y
1z
End --> Exit Simion 3D
Remove All PAs From RAM
File Loaded: MIRROR.PA in 0.11 Sec
2. Click New Button
4. Click OK to Create
3. Adjust nx to 51 ny to 25
1. If PA List Not Empty Click Remove All PAs From RAM

Figure 2-10 Creating a new potential array

Creating a New Potential Array in Memory

The next step is to create a new potential array in memory (*Chapter 4*). Use the following steps for this example (*Figure 2-10 above*):

1. If the potential array list is not empty (*it should be*), click the **Remove All PAs From RAM** button.

2. Click the **New** button to access the Potential Array Creation Screen.

3. Adjust the **X** dimension panel to 51 and the **Y** dimension panel to 25. ***We will use the defaults for all the other array parameters.***

4. Click **OK** to create a new potential array.

Inserting Electrode Geometry

For this exercise, we are going to make a Fast Adjust Definition array (**.PA#**) of a simple three element lens. It is a useful example, because Fast Adjust Definition arrays are the most commonly created array type. The choice of a simple three element lens provides something to play with that will give you insight into how electrostatic ion optics really work.

The Method

The **Modify** function will be used to define the electrode geometry. **Modify** also can be used to change array dimensions, symmetry, mirroring as well as edit the geometry definitions of any existing potential array (*Chapter 5*).

The Plan

We are going to create three adjustable electrodes. Electrode number one will be a circular plate on the left edge of the array. Electrode number two will be a disk with a hole in it in the center of the array (*remember this is a 2D cylindrical array*). Electrode number three will be a circular plate on the right edge of the array.

Electrodes will be ***at least two array points thick*** so that SIMION will treat them as ***solid*** objects rather than as grids when flying ions (*Chapter 5*).

Array points defining electrode number one will be 1.0 volt, those for electrode two will be 2.0 volts, and those for electrode three will be 3.0 volts. ***This is required for SIMION to recognize these points as adjustable electrode points.***

Let's Do It!

Click the **Modify** button on the Main Menu Screen to access the **Modify** function. You should now be looking at the Modify Screen (*Figure 2-11*). Each electrode is defined by setting its point type (*e.g. electrode*) and voltage, marking its area of points, and then clicking the **Replac** button to replace the marked points with the defined values. The followings steps should be used:

1. For electrode number one, make sure the **Electrode** button is ***depressed*** and set the **Voltage** panel to 1.0 volt.

2. Check to see that the **Bx** (*box*) button is ***depressed***. It selects box area marking.

3. Now move your cursor to the ***upper left*** corner point of the array. Hold down the ***left*** mouse button, and drag the cursor (*keeping the **left** mouse button depressed*) to the ***bottom*** of the array ***one*** point to the right (*to mark an area **two** points thick*). Now release the ***left***

mouse button and the area is marked. ***If you make a mistake marking, just re-enter the correct mark.***

4. Click the **Replac** button and click the **YES** button to replace the points in the marked area with 1 volt electrode points. If the wrong points are marked, you can erase them by changing the point definition to Non-Electrode of 0.0 volts, marking the error's area, and replacing the marked points with non-electrode 0.0 volt points.
5. Change the **Voltage** panel to 2.0 volts and insert electrode number two (*as in Figure 2-11*) in the same manner as electrode one.
6. Change the **Voltage** panel to 3.0 volts and insert electrode number three on the ***right edge*** (*as in Figure 2-11*) in the same manner as electrode one.

When you're done creating the electrode definitions, click the **Keep** button to exit **Modify** and keep the array changes (*in-memory copy*).

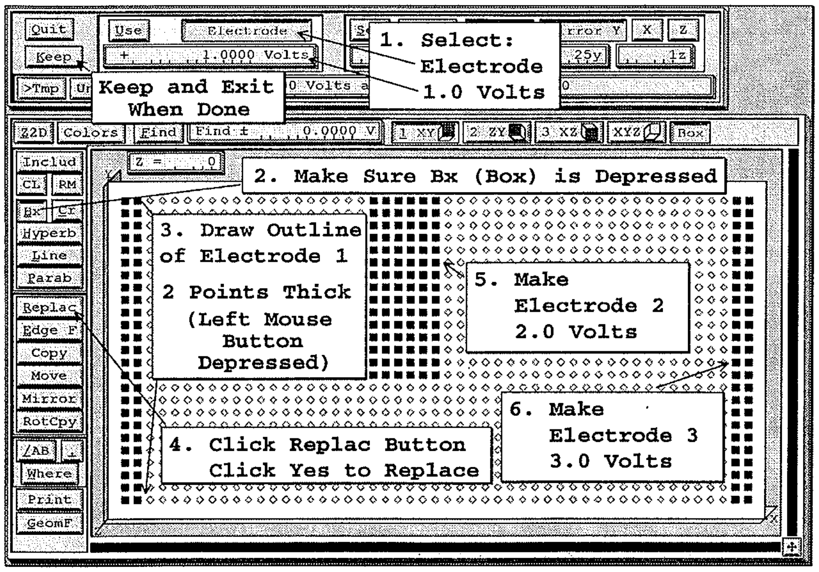

Quit
Keep
>Tmp
Use
Electrode
1.0000 Volts
1. Select: Electrode 1.0 Volts
Keep and Exit When Done
Z2D
Colors
Find
Include
Bx
Hyperb
Line
Parab
Replac
Edge F
Copy
Move
Mirror
RotCpy
Where
Print
GeomF
Z = 0
2. Make Sure Bx (Box) is Depressed
3. Draw Outline of Electrode 1
2 Points Thick (Left Mouse Button Depressed)
5. Make Electrode 2 2.0 Volts
6. Make Electrode 3 3.0 Volts
4. Click Replac Button Click Yes to Replace

Figure 2-11 The steps involved in creating electrodes with the Modify function

Saving The Array as TEST.PA#

At this point the only copy of this array is the in-memory copy. SIMION has assigned it a temporary name of **NONAME01.PA**. We next need to save this file as **test.pa#** in the **mystuff** directory. The **.PA#** file extension tells SIMION that this is a Fast Adjust Definition file (*important*). Use the following steps to save the array (*Figure 2-12 below*):

1. Click the **Save** button on the Main Menu Screen.
2. Make sure the current directory is **mystuff** (*if not highlighted - click on the directory*).
3. Enter **test.pa#** in the **File** ioline and press **Enter**.
4. SIMION will now save the array and display a Memo Screen. You have the choice of entering a short note (*memo*). Just hit the **Enter** key (*to skip the memo*), and the Main Menu Screen returns.

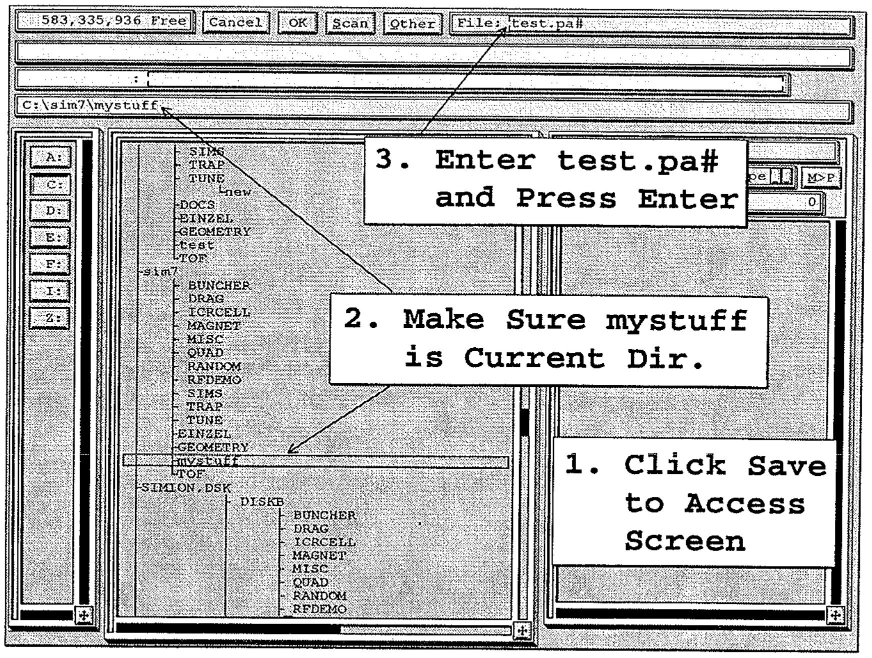
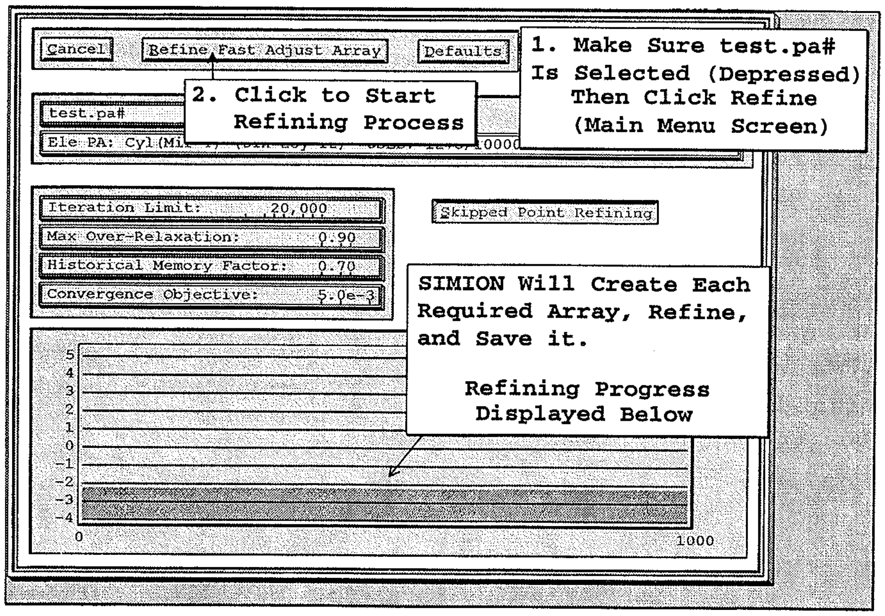

583,335,936 Free
Cancel
OK
Scan
Other
File: test.pa#
C:\sim7\mystuff
A:
C:
D:
E:
F:
I:
Z:
3. Enter test.pa# and Press Enter
2. Make Sure mystuff is Current Dir.
1. Click Save to Access Screen

Figure 2-12 Steps to save potential array as test.pa# in the c:\sim7\mystuff directory

Refining the Fast Adjust Potential Array

The next task is to refine the potential array (*solve for the electrostatic fields*). When you refine a **.PA#** potential array SIMION doesn't actually refine the **.PA#** array itself. It uses the **.PA#** array as a definition for the collection of arrays that **Refine** actually creates, refines, and saves in the current directory (***mystuff***). Use the follow steps to refine the array (*Figure 2-13 below*):


Cancel
Refine Fast Adjust Array
Defaults
1. Make Sure test.pa# Is Selected (Depressed) Then Click Refine (Main Menu Screen)
2. Click to Start Refining Process
test.pa#
Skipped Point Refining
SIMION Will Create Each Required Array, Refine, and Save it.
Refining Progress Displayed Below

```
Iteration Limit:           20,000
Max Over-Relaxation:         0.90
Historical Memory Factor:    0.70
Convergence Objective:     5.0e-3
```

Figure 2-13 How to Refine a .PA# file

1. Check to see that **TEST.PA#** is the currently selected array (*its button is* ***depressed*** *on the Main Menu Screen*). Click the **Refine** button. You should now be looking at the Refine Screen (*Figure 2-13*).

2. Click the **Refine Fast Adjust Array** button to start the refining process.

SIMION will scan the array, determine the number of adjustable electrodes, create each of the four required arrays (**.PA0**, **.PA1**, **.PA2**, **.PA3**), refine each array and save them in the **mystuff** directory.

When all this is completed, you will be returned to the Main Menu Screen. You can verify that these four fast adjust support files have been created by clicking on the **GUI File Manager** button to look at the contents of **mystuff** (*Figure 2-14 below*). Click either the **Esc** or **Return** button to return to the Main Menu Screen.

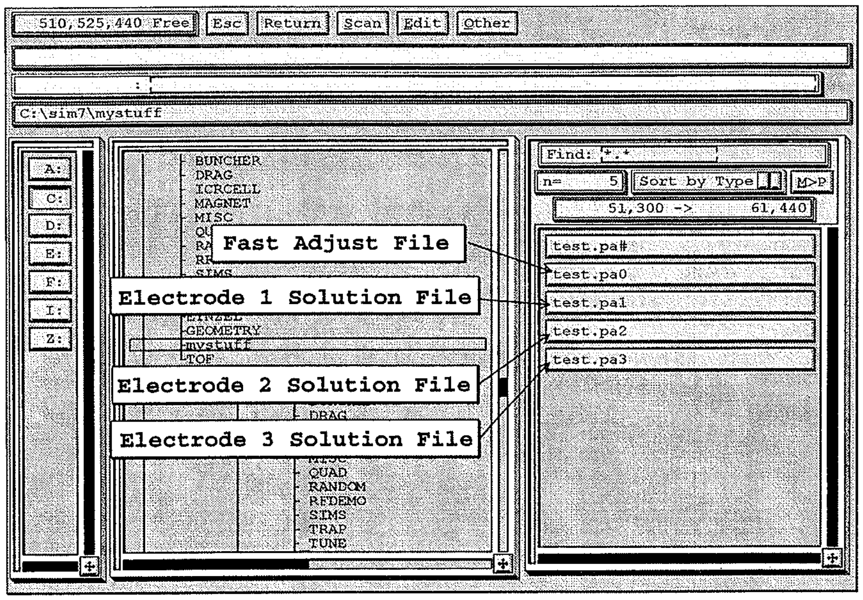


Figure 2-14 File Manager's view of the files created by this Fast Adjust Refine

Fast Adjusting the Voltages of the TEST.PA0 File

The actual Fast Adjust File is the **TEST.PA0** file. This file contains the solutions for any non-fast adjustable electrodes as well as the current potential settings for all adjustable electrodes (*all fast adjustable electrode voltages are initially set to zero when the array was created by* **Refine**). If there had been one or more non-zero non-fast adjustable electrode points (*e.g. one with 85 volts*), SIMION would have automatically created a **.PA_** fast scaling solution file for them and allowed them to be fast proportionally scaled via electrode zero.

The **TEST.PA0** file is the file we must actually fast adjust. We could load it in place of the **TEST.PA#** file by clicking the **Load** button, pointing to the **TEST.PA0** file button, and clicking ***both*** mouse buttons to load the file. However, we're lazy. Let's let SIMION do this automatically for us by clicking the **Fast Adjust** button (*assuming that the* **TEST.PA#** *file is currently selected*). SIMION looks at the file, sees it is a **.PA#** file, and ***automatically*** loads the **.PA0** file in its place and fast adjusts it (*Figure 2-15*).

We want to set electrode number one to 1,000 volts, electrode number two to 1,000 volts, and electrode number three to 0 volts (*its current value*). Use the following steps:

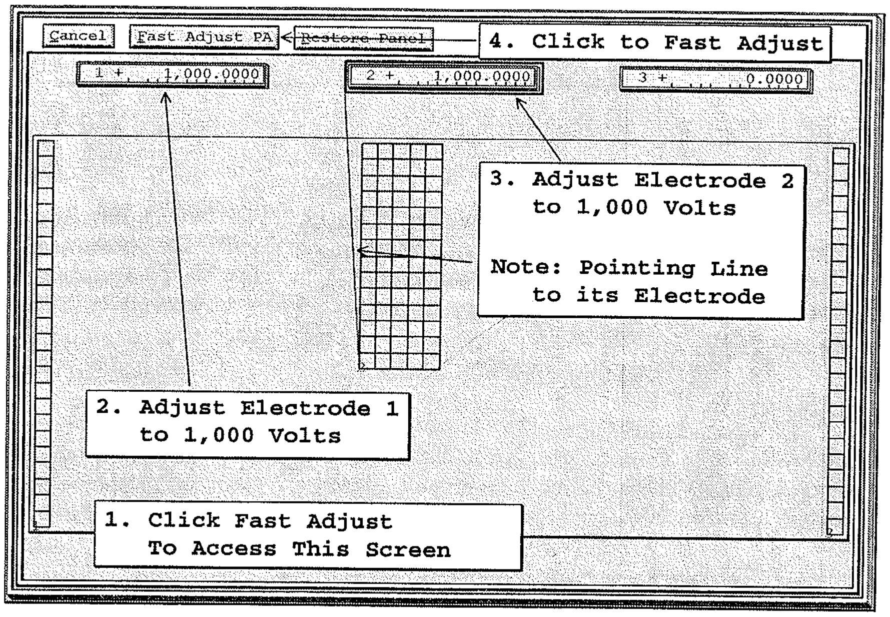

Cancel
Fast Adjust PA
~~Restore Panel~~
4. Click to Fast Adjust
1 + 1,000.0000
2 + 1,000.0000
3 + 0.0000
3. Adjust Electrode 2 to 1,000 Volts
Note: Pointing Line to its Electrode
2. Adjust Electrode 1 to 1,000 Volts
1. Click Fast Adjust To Access This Screen

Figure 2-15 Steps to Fast Adjusting an array with the Fast Adjust Screen

1. Adjust electrode number one's voltage to 1,000 volts using its panel object. Note: SIMION automatically draws a red line connecting an electrode's voltage control panel with one of its electrode points when the cursor is in the panel object.

2. Adjust electrode number two's voltage to 1,000 volts using its panel object.

3. Click the **Fast Adjust** button and the array is fast adjusted.

Note: SIMION fast adjusts the ***in-memory*** copy of the **.PA0** file. If you want to retain these values between sessions you should save the updated in-memory copy back to disk. For Example: Click the **Save** button, hit the **Enter** key (*to save*), click **YES** to replace, and hit the **Enter** key to skip the memo.

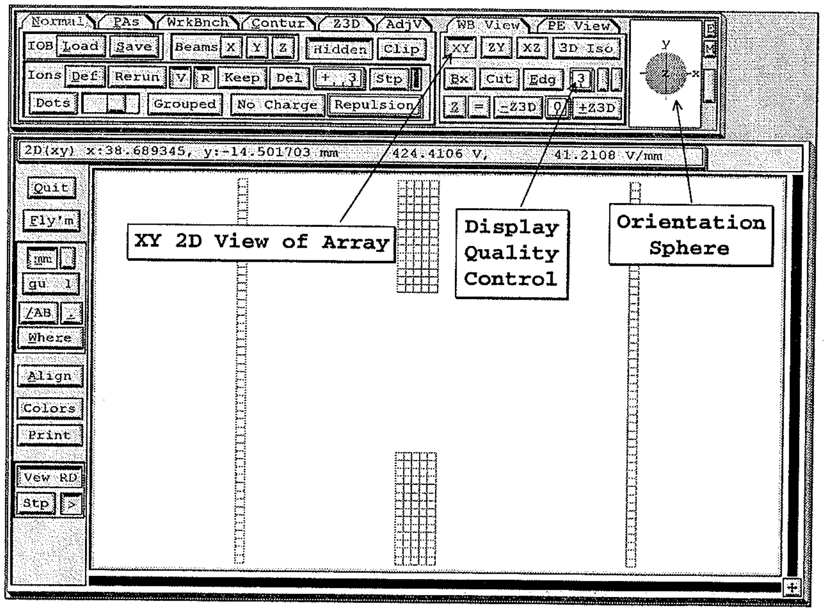


Figure 2-16 Using the View function to visualize the potential array in a 2D XY view

Viewing the Potential Array

To view the **TEST.PA0** potential array, make sure its button is depressed (*on the Main Menu Screen*) and click the **View** button. Your screen should look like Figure 2-16 above.

You can click the **ZY**, **XZ**, and **3D Iso** buttons to see other standard views. Adjust the **Display Quality** panel from 0 (*lowest quality*) to 9 (*highest quality*) in **3D Iso** view to see what it does. Also use the **Orientation Sphere** object to change views (*point the cursor to it and drag the sphere about with* ***either*** *mouse button depressed*). There are 12 2D and 8 3D standard views. See if you can use the **Orientation Sphere** to see all of them. When you're through playing, click the **XY** button to return to your starting view.

Figure 2-17 shows a potential energy view of the potential array. To duplicate this view, click the **XY** button (*or make sure it is depressed*) and then click the **PE View** tab to switch to a potential energy view. The PE view has its own display quality and Orientation Sphere controls. You can toggle back and forth between WB View and PE view and these settings will be remembered.

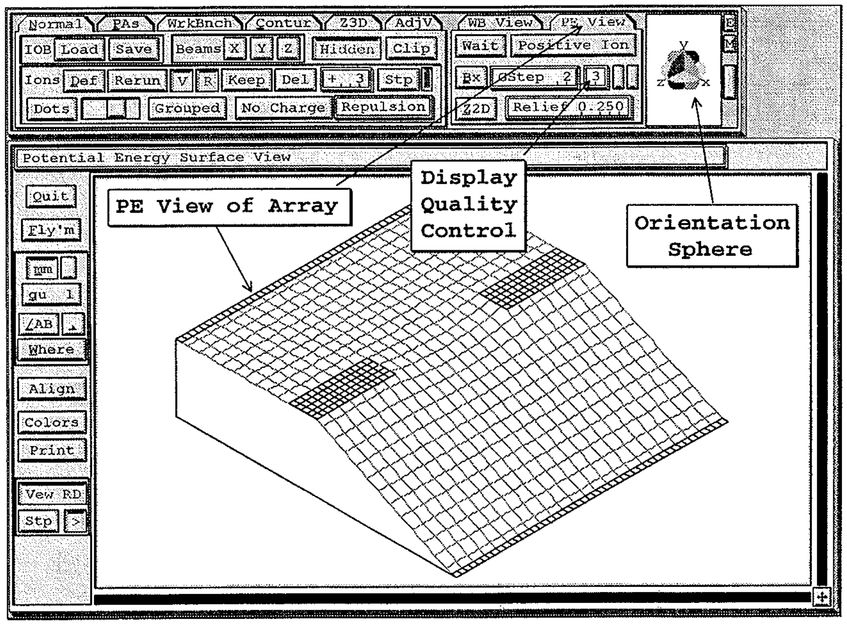


Figure 2-17 Potential energy view of potential array using the View function

Defining Some Ions to Fly

The next task is to define some ions to fly. This is done by clicking the **Def** button on the Normal Controls Screen (*the* **Normal** *tab is selected*). You should now be looking at the Ion Definition Screen (*Figure 2-18 below*).

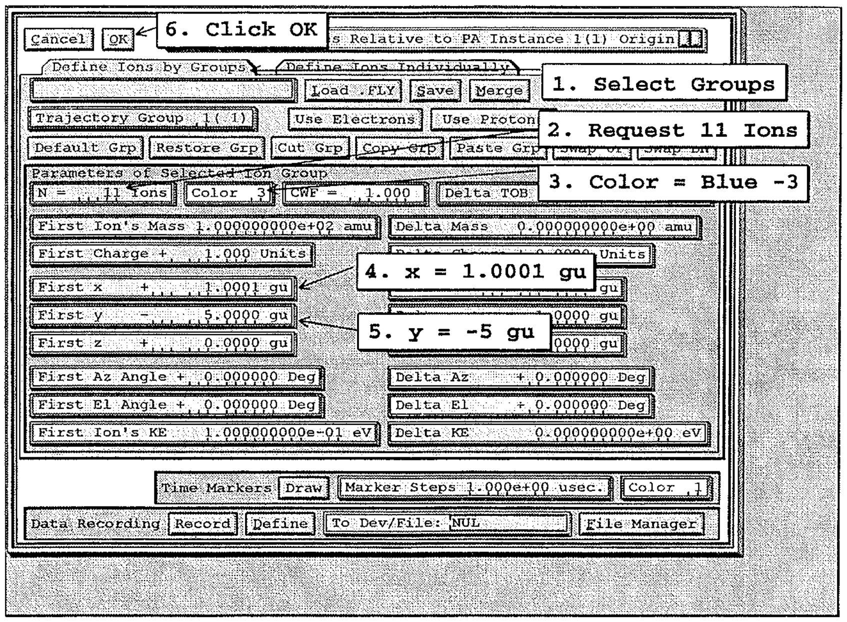
SIMION allows us to define ions by two methods: In groups or individually. For this example we will define ions by groups:

1. Make sure that the **Define Ions by Groups** tab is selected.
2. Set the number of ions in group 1 to 11 using the **N** = panel.
3. Set the color of the ions to blue (*3*) using the **Color** panel.

4. Set the **First x** panel to 1.0001 (*to start just to the right of electrode number one*).

5. Set the **First y** panel to -5.0. and make sure the rest of your settings match those in Figure 2 7-18.

6. Now click **OK** to keep the definitions and return to the View Screen.



```
s Relative to PA Instance 1(1) Origin
Trajectory Group 1( 1)
Parameters of Selected Ion Group
N = 11 Ions    Color 3    CWF = 1.000    Delta TOB
First Ion's Mass 1.000000000e+02 amu    Delta Mass 0.000000000e+00 amu
First Charge + 1.000 Units                          Units
First x + 1.0001 gu                                 gu
First y - 5.0000 gu                                 0000 gu
First z + 0.0000 gu                                 0000 gu
First Az Angle + 0.000000 Deg    Delta Az + 0.000000 Deg
First El Angle + 0.000000 Deg    Delta El + 0.000000 Deg
First Ion's KE 1.000000000e-01 eV    Delta KE 0.000000000e+00 eV
Time Markers  Marker Steps 1.000e+00 usec.  Color 1
Data Recording  To Dev/File: NUL
```

Figure 2-18 Defining a group of ions to fly

Flying Ions

Click the **Fly'm** button to fly the defined ions. Your screen should be something like Figure 2-19 (*below*) if you're still in a PE View.

You can fly the ions together by ***depressing*** the **Grouped** button before you click **Fly'm**. Further, you can keep the ions flying by depressing the **Rerun** button too. It gives a movie effect (*click* **Fly'm** *again or hit the* **Esc** *key to stop*). Trajectory computations will speed up if you change the **Trajectory Computational Quality** panel from 3 (*its default*) to 0. ***There are all sorts of things to learn and try for yourself!***

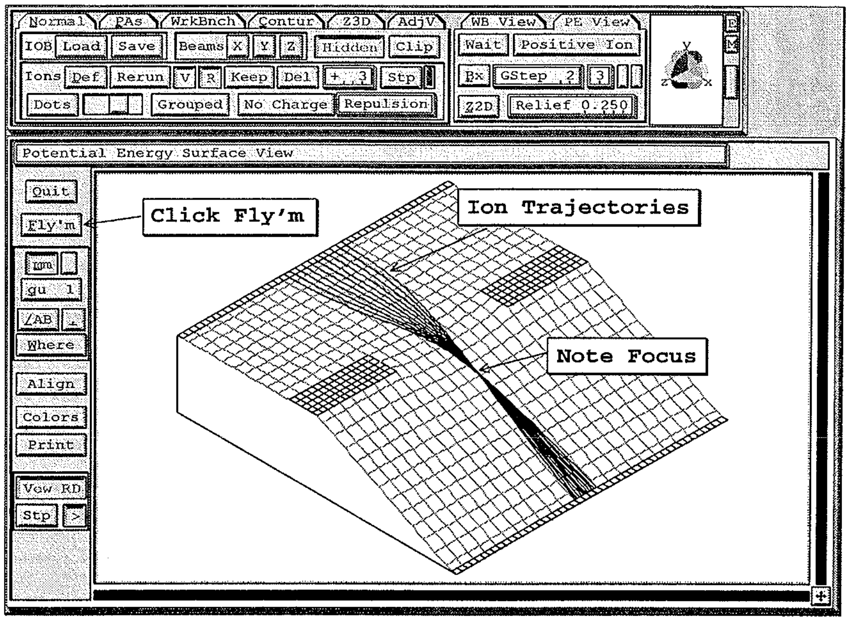


Figure 2-19 Potential energy view of a collection of ion trajectories

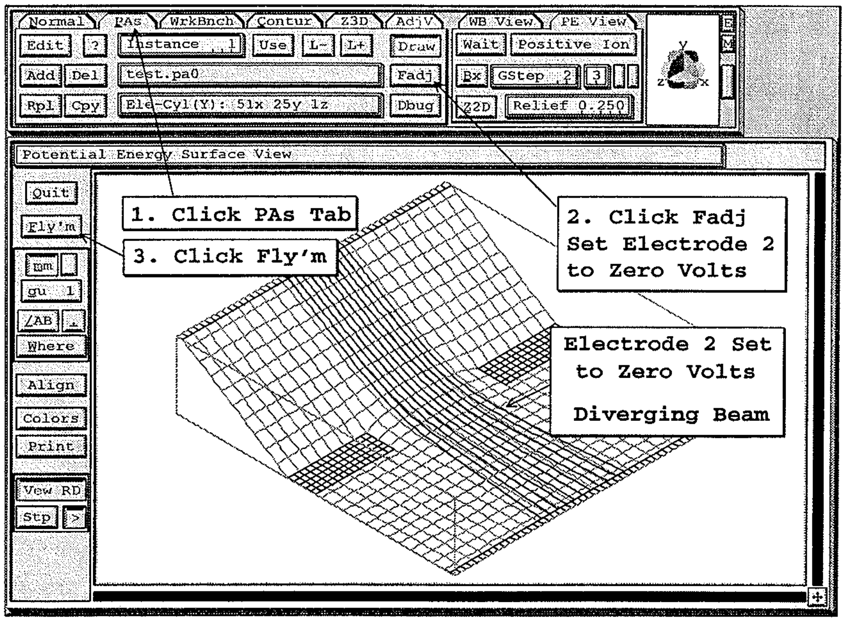


Figure 2-20 Example of fast adjusting electrodes from within View and then re-flying the ions

Fast Adjusting Electrodes From Within View

SIMION allows you to fast adjust electrodes from within the **View** function. This is accomplished by clicking the **PAs** tab to access the PAs Control Screen, use the **Instance Selection** panel to select the desired instance (*there is only one instance in this example*), and then click the **Fadj** button.

For this example adjust the voltage of electrode number two to zero volts and fast adjust it. Now click the **Fly'm** button. Notice the shape of the potential energy surface has changed dramatically. The ions ***diverge*** rather than focus (*Figure 2-20 above*).

Let's assume we want to adjust the voltages so that the ions ***just*** focus when they hit electrode number three. The first step is to ***depress*** the **Rerun** and **Grouped** buttons. Now click the **Fly'm** button. Notice that the ions keep flying like in a movie. Click the **PAs** tab and the **Fadj** button. Adjust electrode number two to 900 volts, and check the focus. If it's not quite right click the **Fadj** button and try again. This is known as interactive tuning.

Saving Your Work

It is important that you save your work. This allows you to quickly resume your efforts in a subsequent SIMION session. The following material shows you how to save your **.IOB** file (*ion optics workbench definition file*) as well as saving an auto-loading **.FLY** (*ion group definition file*).

Saving an Ion Optics Workbench File .IOB

An ion optics workbench file (**.IOB**) retains the current workbench definitions. These include the dimensions of the workspace, the names of the potential arrays used, their adjustable potentials, and the array instance definitions of how these potential arrays are to be projected into the defined workspace.

To save the current workbench definition (*including the voltages used in all instances*), click the **Normal** tab to access the Normal Controls Screen. If ions are currently flying from the above adventure, click the **Fly'm** button (*or hit the* **Esc** *key*) to stop the flight. Now click the **Save** button on the Normal Controls Screen, enter the word **test** for a file name and hit the **Enter** key. SIMION will automatically append the **.IOB** extension to the file, and save your current workbench definitions as **test.iob** in the **mystuff** directory (*assuming it's the current directory*). You can also make use of long filenames if you desire (*e.g.* ***My First Try at SIMION***).

You can save more than one **.IOB** file. Let's say you wanted to save an **.IOB** with different voltage settings. You would adjust the electrodes to the desired voltages and ***then*** save an additional **.IOB** file (*perhaps* **test1.iob**) in the same manner you saved **test.iob** above.

The Value of File Memos and How to Create Them

This example brings up the issue of file memos. If you choose to use names like **test.iob** and **test1.iob** you will probably not know which one to choose a month (*or day*) from now. Although SIMION now supports the use of long filenames (*e.g.* ***My First Try at SIMION.iob***), you may want to consider attaching a file memo to each file when it is saved to describe its features or other details.

If you forget to create a memo, all is not lost. Click the **GUI File Manager** button on the Main Menu Screen. Point to the file's button that you want to create a memo for (*e.g.* **test.iob**). Now move the cursor to the ***left*** off the button. The memo line (*second ioline object from the top of the screen*) should have the file's ***short*** name on it (*if you moved the cursor to the left properly*). Move the cursor into the **Memo** ioline and enter the desired memo text. ***That's all there is to it.***

You can verify that the file has your memo by moving the cursor up and down across the file name buttons. When the cursor touches the file with a memo, the memo will be automatically and instantly displayed. Very handy.

Saving an Ion Group Definition File .FLY

When you save an **.IOB** file SIMION will ***always*** ask you if you want to save an auto-loading ion definition file too (*.FLY or .ION depending on the ions currently defined*). If you click **Yes**, SIMION will save the appropriate ion definition file with the name of the **.IOB** file and proper ion definition file extension (*e.g.* ***test.fly*** *for the above example*). This ion definition file will automatically be loaded whenever its associated **.IOB** file is loaded (*handy*).

You also have the option of saving ion definitions to files of your choice. In this case, click the **Def** button on the Normal Controls Screen. Now click the **Save** button, enter the words **First Ion Definitions** for a file name and hit the **Enter** key. SIMION will automatically append the **.FLY** extension to the file, and save the current ion group definitions as **First Ion Definitions.fly** in the **mystuff** directory (*assuming it's the current directory*).

Reloading Your Work

Let's pretend that this is a new SIMION session and that we want to reload that landmark simulation that we performed above. Click the **Quit** button and **YES** button to exit from **View** back to the Main Menu Screen. Now click the **Remove All PAs From RAM** button and **YES** to return all of PA memory to the available memory heap.

Click the **View** button. SIMION doesn't have a potential array to view (*the* **empty PA** *button is depressed*), so it automatically calls the GUI File Manager with instructions to load an **.IOB** file. Point the cursor at the **test.iob** file button (*saved above*) and click ***both*** mouse buttons. SIMION will load the **.IOB** file and all the potential arrays it references. You will be asked if you want to restore all potentials to what they were when the **.IOB** file was last saved. Click **YES** and all voltages will be automatically restored (*by fast adjust* - **.PA0 files** *or fast scaling methods* - **.PA files**).

SIMION will now check for auto-loading files. In this case it will find a file called **test.fly** (*assuming you said* ***Yes*** *to saving an auto-loading ion definition file when the* ***test.iob*** *was saved*). This file will be automatically loaded and a message will inform you of this action. SIMION also can save and auto-load other files. These include .**ION** (*ion by ion definitions*), .**REC** (*data recording*), and .**KEPT_TRAJ** (*kept ion trajectory files*). These features are discussed in Chapter 7.

Summary

We have covered a lot of ground in this chapter. While this information should serve to get you started with SIMION, there is a whole lot more to learn. You have the option of either using the **F1** (*help*) key to ***crash***, ***burn***, and ***learn***; or you can keep on reading.

It is recommended that you read a bit and use SIMION a bit. This is probably the quickest way to learn how to use SIMION effectively. As you learn, try to use SIMION for some real problems, but be careful not to be too ambitious with what you tackle until you have learned the techniques to attack it properly.

The Adventure Continues . . .