

Developments in the Electron Gun Simulation Program, EGUN *

W. B. Herrmannsfeldt

*Stanford Linear Accelerator Center, Stanford University
Stanford, California 94309*

Abstract. This paper discusses the developments in the electron gun simulation programs that are based on EGUN with its derivatives and supporting programs. Much of the code development has been inspired by technology changes in computer hardware; the implications of this evolution on EGN2 are discussed. Some examples and a review of the capabilities of the EGUN family are described.

1. INTRODUCTION

Technological progress in the performance of small computers has rapidly changed the demands on the software that is properly matched to the hardware. Table 1 shows running times on some PC-clones and workstations for the electron optics program EGN2 (1) running a simulation of a gun for an Electron Beam Ion Source (EBIS) (2).

The speed increases shown in Table 1 have been accompanied by a reduction in memory cost that now permits from 8 to 32 Mbyte, or more of random access memory (RAM) in PCs. The challenge for the supplier of a simulation code such as EGN2, is to respond to the increasing computer capability, and increasingly challenging demands by users, in the way that best meets the needs of the community while preserving the investment in experience with the program. In this paper, we describe how EGN2 developed in response to these factors.

The original PL-1 and Fortran versions of EGUN for mainframe computers have evolved into today's family of programs including preprocessors, graphics programs, and post processors all working together in support of the C program EGN2 and the Fortran program IGUNe. This family tree is illustrated in Table 2.

*Presented at the Workshop on Pulsed Radio Frequency Sources for Linear Colliders,
Montauk, Long Island, New York, October 2-7, 1994*

*Work supported by Department of Energy contract DE-AC03-76SF00515.

Table 1. EGN2 Running Times for the EBIS Gun and EGUN84

Computer system ^(a)	Compiler ^(a)	EGUN84 (sec)
40286, 8MHz, DOS	MS C-7.0	4990
40386, 25MHz, DOS	MS C-7.0	685
SUN, IPC	Sun OS 4.3.1	128
40486, 66MHz, DOS	MS C-7.0	108
40486, 66MHz, OS/2	Borland OS/2	59.3
RS/6000, 320H, AIX	XL C-1.2	51.5
Pentium, 66MHz, DOS	MS C-7.0	49.7
Pentium, 66MHz, OS/2	MS C-7.0	31.8
RS/6000, 580, AIX	XL C-1.2	13.5
DEC ALPHA, 200MHz	OSF/1	12.5
RS/6000, 590, AIX	XL C-1.2	7.0

^(a) Identifiers in the this table are, of course, all registered trademarks.

Table 2. The EGUN Family

Program Function	Name	Name
Graphic Editor for POLYGON	GPED	
Boundary Preprocessor	POLYGON	
Magnetic Field Preprocessors	COILFIT	INTMAG
Main Programs	EGN2	IGUNe
General Postprocessors	PPPROF	ANALYSE
Special Postprocessor	PPGYRO	
Graphic Systems	YMWPLLOT	XHPLOT

The most obvious result of the increase in computer speed for a mesh-based program is that larger numbers of mesh points can be used while maintaining reasonable computation time. Since the primary reason for needing more mesh points is to get better resolution accuracy, various steps must be taken to assure that improved accuracy is in fact achieved. We will consider the implications of larger size problems in the following sections.

2. PROBLEM SIZE

The program EGN2 is written in the C programming language. One advantage of using C is that it is relatively straight forward to permit the program to do internal array allocations. This is in contrast to the program IGUN (3), the ion source program written by Becker, which uses FORTRAN-77, and has fixed arrays determined by the programmer. The extent to which the choice of array allocation method matters to the user depends only on whether problems can be run with as many mesh points and

trajectories as are desired. Thus for example, by using suitable compilers, both IGUN and EGN2 can be run in protected mode on a PC running either DOS or OS/2. EGN2 can also be run in ordinary DOS, as was done for the PC examples in Table 1, using expanded memory for cases in which the size of the problem exceeds that allowed by the conventional 640 kBYTE RAM of DOS.

There are a number of different ways in which large numbers of mesh points affect the way a program operates. Some of these also affect the user who must be aware of possible pitfalls, of which perhaps the most obvious is that an adequate solution of the potentials is found by the Laplace and Poisson equation solver. Having more mesh points implies a solution time that can increase roughly with the square of the number of points, but there are things that the user can do that will either greatly speed up the convergence or, conversely, slow it down and even result in it being unlikely that a good solution can be achieved. The two most obvious steps that the user can take are

- assist the program in having a good preload of the potential arrays, which means that the initial filling of the potential array should be a reasonable first guess for the final solution, and
- obtain an adequate initial solution for Laplace's Equation (before space charge is added).

For a problem dominated by space charge, such as a space-charge-limited Pierce diode, it is not necessary to have a very good Laplace solution before adding space charge. Conversely, for a device without much space charge, such as a photomultiplier tube or even a field emission diode, it may be adequate to run only one program cycle to get a self-consistent solution, so that a very good solution to Laplace's Equation is important. In EGN2 we have provided the parameter PASS, as well as a modifier of the internal error criterion, ERROR, in the initial control section of the program, to allow users to determine the number of "passes" to be made of Laplace's Equation.

The preload can be improved by the user who takes time to understand the algorithm that we use to determine the initial potential distribution. We interpolate potentials on each successive row of interior points from left to right. Any boundary segment with a fixed potential forms an end point for the interpolation. This means that a pair of parallel plates would have an initial preload that is identical to the final correct solution. However, if a plate with a hole leading to a long beam tunnel is used, this interpolation may result in the critical central part of the problem area having a very bad solution. A concerned user can assist the program by providing a surface of dummy boundary points on which a suitable potential has been defined. Dummy boundary points are points which do not define either a Dirichlet (metal) or Neumann boundary. Such a dummy boundary segment will only affect the preload; used in this way it can easily speed convergence by very large factors.

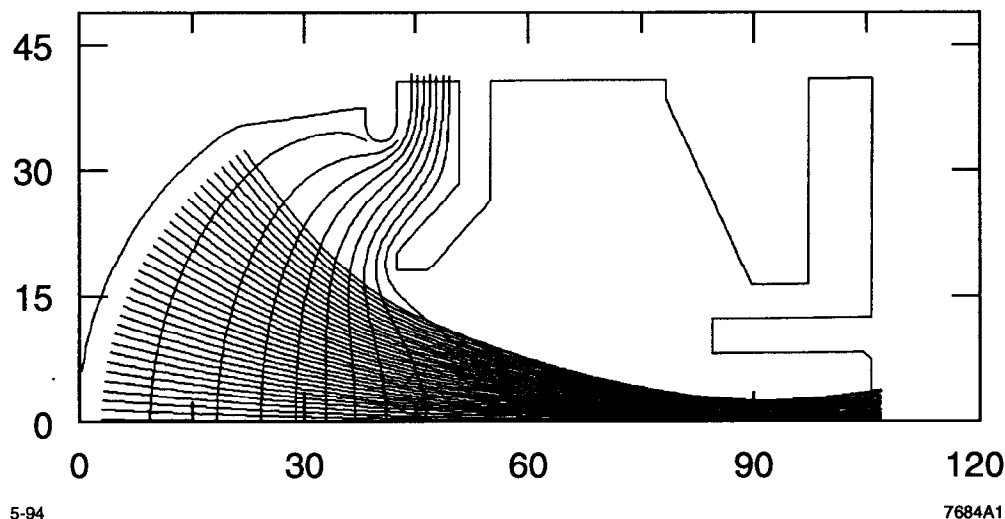


FIGURE 1. The EBIS gun, EGUN84, without the matching magnetic field, was used in this form for the set of benchmark runs for Table 1.

Large problems imply large, complex boundary configurations. Fortunately, the boundary-definition program POLYGON (4) by Reinard Becker is available to ease the creation of boundary files. Recent versions of POLYGON can pass on all the other control and initialization data needed by EGN2, so that users do not have to repeatedly edit these files. POLYGON is provided with every copy of EGN2 and is integrated within IGUN. The Graphical Polygon Editor (GPED) (5) is available to create the POLYGON data and simultaneously trace the problem boundary on a computer screen as it is being defined.

The accuracy of the complete solution of Poisson's equation depends also on the way space charge has been deposited on the mesh nodes. A recent improvement in EGN2 is to do space charge allocation the same way that it has been done in IGUN and in various other programs, such as Particle-in-Cell (PIC) codes; that is, by allocating space charge to the four nearest node points on every integration step. The allocated charge is a product of the step time and the current in the trajectory. This improvement in the space charge allocation has resulted in problems such as the EGUN84 (shown in Fig. 1) having a much more uniform distribution of trajectories. The high area convergence of guns like EGUN84 causes particles to pass through the mesh at very steep angles. This new allocation algorithm deposits space charge correctly for particles with any angle of inclination through the mesh.

3. MAGNETIC FIELDS

There are several significant areas of development in EGN2 relating to magnetic fields, including:

- implementing equations of motion in xyz coordinates to eliminate the singularity on the axis for equations in $r\theta z$ coordinates
- making provision to store the scalar arrays of magnetic field components, B_r and B_z on the same mesh as that used for the electric field solution
- continuous development and support of the magnetic field simulation program INTMAG (6) by Reinard Becker. INTMAG has output formats which are matched to the needs of EGN2 and IGUN

Precise implementation of externally imposed magnetic fields is very important in the correct simulation of the matching conditions of guns with large ratios of cathode area to final beam area. This condition applies to several classes of problems including especially linear beam microwave tubes and EBIS guns. In Figs. 2 and 3 we show the EBIS gun of Fig. 1, now with the imposed magnetic field. The mismatched solution of Fig. 2 contrasts obviously with the well-matched solution of Fig. 3. However, the differences in the external

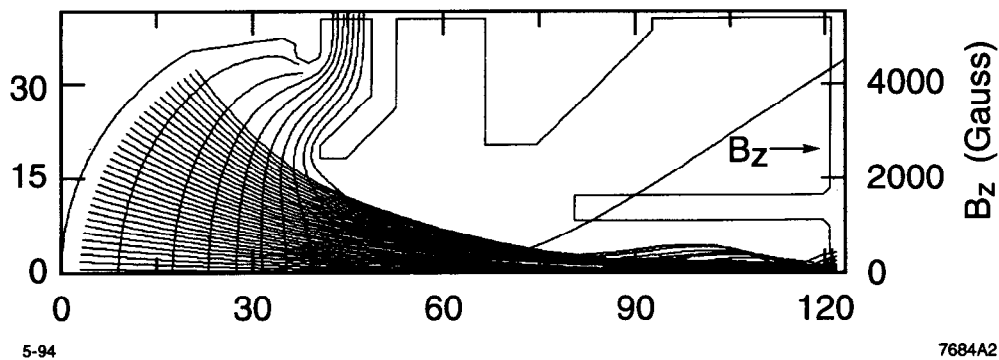


FIGURE 2. The EGUN84 gun, with a slightly different set of shields and pole pieces, is shown with the matching magnetic field. Scalloping is beginning to be apparent.

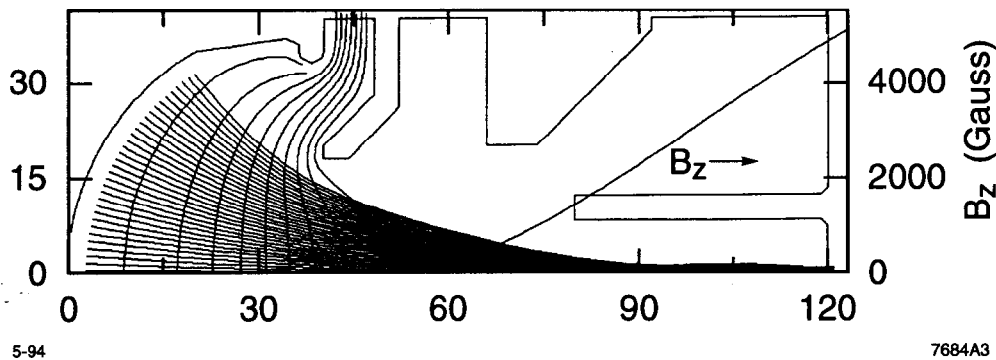


FIGURE 3. The field shown in Fig. 2 was shifted by six mesh units (1.5 mm) to improve the matching condition, as shown here.

magnetic fields are not so great. In fact, the same field solution, provided by INTMAG, is used in both cases, but with a small longitudinal shift.

The beam's own self magnetic field is treated separately from the external magnetic field. The particles are calculated through the entire length of the trajectory path, in a sequence starting from the axis and proceeding outward. In a laminar beam, this mode of calculation allows the self-magnetic field to be calculated for all the current that affects each successive ray. As the particles begin to undergo nonlaminar crossings, this method suffers in accuracy but is designed to do so gradually. Another accuracy problem results for intense relativistic beams, where the self-magnetic field nearly exactly cancels the space charge. The normal mode of operation, in which space charge is calculated from the previous program cycle and self-magnetic field is calculated from the present particle distribution, can result in the taking of differences between two large forces of opposite sign from successive cycles. The resulting errors can be quite large. A solution is suggested by the fact that the residual defocusing force is very small, so that if the space charge and self-magnetic field terms are subtracted directly, rather than by depositing the full charge on the mesh, the program should converge. Because this method does not correctly treat longitudinal space charge, which is what causes the emission to be space charge limited, it is necessary to restrict its application to regions of relativistic velocity. Thus in high-voltage devices, the parameter ZDOTEQ allows the user to specify the relativistic velocity above which the calculation will be made in the way described above, which is known as the EBQ mode. Typically, a ZDOTEQ value of around 0.9 or so would be used to cause the calculation to switch to the EBQ mode. In an electron gun, the value of ZDOTEQ could reasonably be chosen to correspond to about 90% of the tunnel velocity.

The equations of motion derived in Ref. (1) include a conservation of angular momentum term for $r\theta z$ coordinates. No particle with finite angular momentum, or which is in a longitudinal magnetic field, can be allowed to pass through $r = 0$ in such a coordinate system. However, because the integration is made in finite steps, as the trajectory approaches the axis of symmetry, very small steps must be taken to avoid the severe mathematical disaster that occurs by division by a very small number. In fact, even for particles that do not approach the axis, this method was found to give small errors in problems such as, for instance, an electron beam lithography system, in which a test problem consisted of mapping four emission points to four target points in a uniform magnetic field. Most modern PIC codes use xyz coordinates, in which the axis is not a singularity, by resolving the radial terms E_r and B_r into corresponding x and y components of the fields. Since the integration must keep track of the azimuthal position of the trajectory anyway, this does not add significantly to the amount of computation. The NAMELIST input parameter CSYS = 2 switches the program into the xyz coordinate system. However, all the input and output tables still show the cylindrical symmetry formats, so that there is no overt sign that the program is operating with the xyz equations.

There are several methods that can be used to input magnetic fields into EGN2:

1. A polynomial expression for the axial field, $B_z = f(z)$, at $r = 0$
2. An array of values for B_z at $r = 0$
3. A set of ideal point coils, or current loops, where the user specifies the r and z locations of the coils, and the current on each
4. A complete map of all the scalar field terms, B_r and B_z , on each mesh point which has been previously defined as an interior point to the problem

Method 4 is the most general, but obviously depends on having output from a suitable magnet design program. It replaces a capability that was in earlier (Fortran) versions of EGUN, which could accept data from the program POISSON for the vector potential $A(r, z)$. In this new implementation, data from any magnet design program can be taken as input, provided that the program can save magnetic field data in a table so that the r and z coordinates appear on a row with values of B_r and B_z . The EGN2 input routine can be directed to select which columns represent which numbers even if other irrelevant data are in the row. The significant factor which allowed this implementation is the availability of extra memory in most small computers. Thus there is now allocated space in the large arrays for the magnetic field data. When the program reads in the file of magnetic fields, it selects only those points which lie inside the boundaries of the electrostatic problem. It is the responsibility of the user to ensure that at least the part of the problem area that is likely to have any particles is included in the magnetic field data.

Magnetic fields for method 4 are calculated by a simple interpolation of the data for the four nearest mesh points. For the other three methods, the normal way to calculate fields is by a sixth-order radial expansion of the axial fields. The sixth-order expansion has proven remarkably accurate in the past. In the case of the EGUN84 problem, for example, the same results are found by either the full map or the off-axis expansion. However, there are some constraints on the off-axis expansion that users must consider:

- The data must have full double-precision accuracy for the sixth-order differences to be calculated. This is internally controlled if method 1 or 3 is used, but if the user provides an array of field data for method 2, he must be aware that crude data, such as from a magnetic measurement probe, cannot be used directly. (Such data can however be used in one of the ways described below.)
- Expansions cannot go past magnetic elements, such as the point coils.
- As mesh density is increased, the off-axis expansion may be asked to extend farther in the radial direction (by a ratio to the base of 13 mesh points along the axis), and thus ultimately to introduce significant errors. Although such errors have not been observed, this is the new factor that results from larger mesh areas, and is the one factor that inspired the provision of the full map of scalar arrays.

From methods 1 to 4 above, the generally preferred choice is method 3, a set of ideal point coils (or current loops). There are three distinct advantages to this approach:

- The data sets are the most compact.
- Fields can be found anywhere in space, not just in the region near the axis, by using the elliptic integral formulas that are internal to the program. The elliptic functions provide exact solutions for the magnetic fields from a set of coils. Although these routines are rather slow for most ray-tracing applications, they can be used as an option for cases in which the off-axis expansions are not appropriate. An example of such a case is that in which the solenoid is used to defocus a beam that passes outside of the solenoid coils.
- By using both the elliptic integrals and the off-axis expansion method, EGN2 provides a table of field values which can be used to compare the accuracy of the off-axis expansions. This table is provided for any use of the point coils. The user provides a parameter RMAG that determines the radius at which the comparison is to be made. Typically RMAG is set to approximate the outer radius of the beam in the region of strongest magnetic field focusing. RMAG has no other implications besides providing the diagnostic table of magnetic fields.

We do not usually recommend using method 1, the polynomial expression for the axial field, although a simple one-term expression is the easiest way to specify a magnetic field that is uniform over the whole problem area. Other polynomial expressions will generally not be consistent with any feasible configuration of magnetic elements, and thus will result in very nonphysical off-axis fields.

There are two ways in which magnetic field data that lacks the desirable full double-precision accuracy can still be used:

- The less satisfactory way is simply to reduce the extent of the power series that is used to calculate off-axis fields. This series can be reduced to either second or fourth order in r . The field terms for B_z are even and the terms for B_r are odd, so that the highest order term is always a B_z term. The variable MAGORD determines the extent of the power series.
- The better way to use data that does not have the proper precision is with a program that can find a set of ideal point coils that closely approximates the desired field. The program COILFIT does this by making a least squares fit to the input data. The user specifies the number of point coils (less than the number of data points) and the desired radial and axial positions of the coils. COILFIT finds the currents on the coils that fit the desired data, and then can fill in the entire problem length with fields calculated from the ideal coils, to full double precision. Alternatively, the user can choose the coil data itself to input into EGN2, thereby preserving the other noted advantages of using the ideal point coils. COILFIT is available as a preprocessor program for EGN2.

4. OTHER CAPABILITIES

As noted earlier, the EGUN family of programs includes several pre- and post-processor programs. We have briefly touched on the preprocessors above.

The main program EGN2 has as was noted, a companion program IGUN for specific use in ion extraction from a gaseous plasma. IGUN can do the particle ray tracing, including space charge and other effects, just as EGN2 does.

There are two general-purpose postprocessor programs, PPPROF and ANALYSE. They differ mostly in that PPPROF was written to take data from a binary file of records of the particle trajectories, while ANALYSE extracts its information from the printed output file including especially the space charge map. ANALYSE was developed by the Frankfurt group, and PPPROF is a product of the SLAC group. Both have the objective of providing beam profile information as a function of z . Each EGN2 simulation concludes with the usual ray tracing plot, and with phase space and beam profile plots. PPPROF extends the diagnostic information that appears with the final results to an arbitrary number of locations along the axis.

Special purpose postprocessor routines can be developed to read and interpret the binary file. One that has filled a special need is PPGYRO for examining data from a gyrotron beam. Here the objective of the designer is to impart the maximum angular velocity to the beam while maintaining beam quality, meaning especially that particles should all be close together in azimuthal phase. The diagnostic expressions in PPGYRO evaluate the results for each location in z .

Many electron guns, including especially those which, like the EBIS gun, have large area convergence ratios, are limited in the final beam spot size by the transverse temperature of the particles. The limiting small value of this temperature is the cathode temperature, typically around 0.1 eV. EGN2 accepts the beam temperature in degrees-kelvin and calculates a radial energy increment to be added and subtracted from particles that start from the initial coordinates of each "ideal" particle; i.e., before adding the thermal energy. There are three models, using two, three, and five particles, respectively, for each initial particle. The two-particle model is the most satisfactory because it includes a random-number generator to give a statistical sense to the process.

For someone not experienced with thermal effects, the initial transverse energy of 0.1 eV may sound negligible. However, radial compression of the beam results in a proportional increase in the temperature. This temperature increase is a direct consequence of Liouville's theorem and is not influenced by beam intensity. Space charge forces, which are typically nonlinear for a real distribution, are an additional heating mechanism. The measure of all these effects is the beam phase space, which EGN2 calculates in several ways. The calculation is made on the bases of four times the rms emittance to get the effect of the beam edge emittance, sometimes known as the 95% emittance.

It is frequently desirable to be able to segment a problem into parts which then can be computed sequentially. EGN2 creates a data set for the final conditions of the particles in the format that the program needs to read input data. This data can be saved directly to a designated file or, if it is only after the fact that the need is discovered, it can be extracted from the output listing. Sometimes it is useful to have the initial condition data in the same format, so this is included in the above file. The problem scale and initial z location of the particles can be shifted by the parameters SKAL and ZO.

There are also cases in which it is desirable to make changes in the first segment of a simulation and see how the beam is affected in later stages. In such cases, all the segments can be placed together in the input stream, with the parameter SAVE = 2 signalling that input ray data should be taken from the previous problem. The same SKAL and ZO parameters apply in this case.

An alternative method of saving data is used when the boundaries and potential distributions of one solution are desired for subsequent runs. The problems are again placed together in the input stream, with the parameter SAVE = 1 in the first data set indicating that the next one should not read boundaries, and should not clear the potential and space charge arrays. One possible application of this capability might be to a long drifting beam, such as the EBIS simulation, in which both the particles and the boundaries might be saved. Another example would be for an examination of the trajectories of secondary and scattered particles. The particles can be initialized from chosen locations and traced in a configuration that includes the space-charge fields from the first part. This capability can be useful for simulating particles coming, for example, from a beam collector or from electrodes in an accelerating column.

The PC versions of the EGUN family all use the YMWPlot program which makes screen images on all standard combinations of monitors and graphics boards. YMWPlot also contains the capability to make accurate, high-resolution, hard-copy renditions of the results by creating and sending a bitmap image of the figure to either laserjet or dot-matrix printers. This approach is far superior to a screen dump, which is limited to screen resolution and usually does not work anyway for graphic images. EGN2, and the other members of the EGUN family with plotting needs, all make a file of the type *.cpl. It is this file that is read by YMWPlot. The *.cpl files can also be read by the program XHPPlot, which was developed to make plots on pen plotters using the Hewlett-Packard HPGL graphics language. A recent addition to the set of plotting programs is XPSPlot, which makes PostScript files that can be used by word processors and for making plots on UNIX systems.

5. MISCELLANEOUS APPLICATIONS

Although EGN2 was, as the name implies, developed for the purpose of designing electron guns, it has been used to simulate numerous other electronic devices, some quite interesting and worthy of mention.

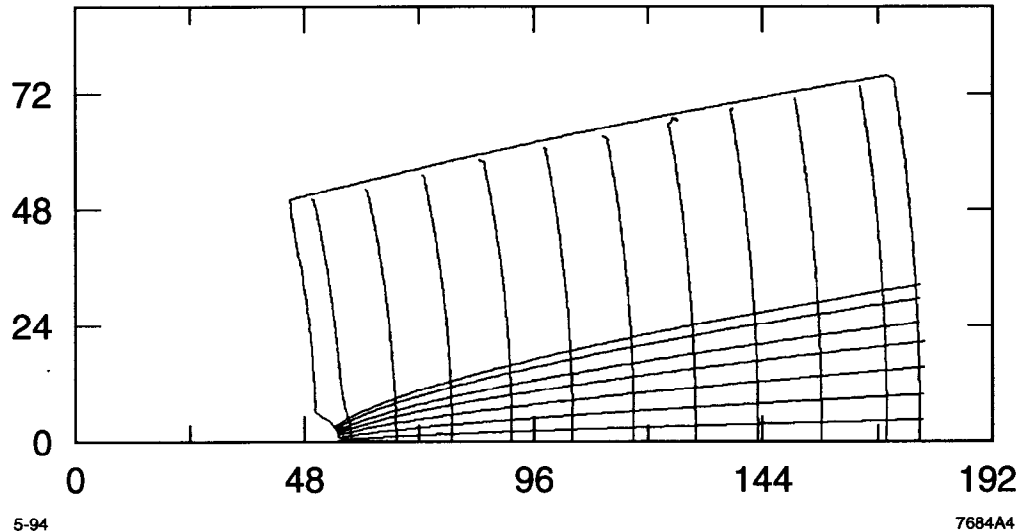


FIGURE 4. An enlarged segment of the tip of a gated field emitter, simulated with a small bump on the tip. The mesh resolution is 1 Å.

It has always been possible to include dielectric materials in EGUN simulations, but recently this capability has been enhanced by the inclusion of special potential numbers that EGN2 interprets as dielectric coefficients. This substantially reduces the work required to create a data file for a problem which includes dielectrics.

We reported on simulations of field emitters at the Toulouse conference on Charged Particle Optics (7). This capability has found increasing application to the new field of microelectronics. The resolution needed to simulate field emitters may be as small as 1 Å, as it was for the simulation of a small bump on the tip of an emitter of 200 Å radius shown in Fig. 4. Since the density of mesh points increases by the square of the magnification, it is easy to see that a faster computer with larger memory makes only a small difference in the allowed resolution. The configuration that ended with the simulation shown in Fig. 4 began with a 500-Å-resolution simulation of a gated field emitter. The most important thing is to reduce the area to be simulated by finding suitable boundary conditions. The boundary input processor POLYGON accepts equipotential-line and field-line coordinates from EGN2, and uses these lines as boundaries for a magnified problem. Typically, magnifications by a factor of 510 can be used; but two or even three magnifications may be required to get the required resolution. Three magnifications were used to achieve the final 500 × magnification shown in this example.

ACKNOWLEDGMENTS

The C programming for EGN2, and all of the graphics support, has been the work of Glen A. Herrmannsfeldt. The boundary preprocessor POLYGON, the magnetic field preprocessor INTMAG, and much of the inspiration for making improvements such as some of the ones discussed here, have been provided by Reinard Becker. We thank them for their assistance in assembling the material for this paper, and most especially, for their many contributions to the EGUN family of programs. We are also indebted to support from colleagues at SLAC and for suggestions and encouragement from the significant user community of the EGUN programs.

REFERENCES

1. Herrmannsfeldt, W. B., *EGUN, An Electron Optics and Gun Design Program*, SLAC Report 331, Stanford Linear Accelerator Center, 1988.
2. Becker, R., Kleinond, M., Thomae, H. , and Donets, E. G., *Proceedings of HCI-92*, eds., Richard, P., Stöckli, M., Cocke, C. L., and Lin, C. D., *AIP Conf. Proc.* **686** (1993).
3. Becker, R. and Herrmannsfeldt, W. B., "IGUN, A Program for the Simulation of Positive ion Extraction Including Magnetic Fields," *RSI* **63**, 2756 (April 1992).
4. Becker, R., "Easy Boundary Definition for EGUN," *Nuclear Instrum. Methods* **B42** 162-164 (1989).
5. Zipfer, B. and Kester, O., "Graphic Editor for Polygon," private communication (1993).
6. Becker, R., "INTMAG, A Program for the Calculation of Magnetic Fields by Integration," *Nuclear Instrum. Methods* **B42** 303-306 (1989).
7. Herrmannsfeldt, W. B., Becker, R., Brodie, I., Rosengreen, A., and Spindt, C., "High-Resolution Simulation of Field Emission," in *Proceedings of the Third International Conference on Charged Particle Optics, Toulouse, France*, *Nuclear Instrum. Methods* **A298**, 39-44 (1990).