Trak 6.0

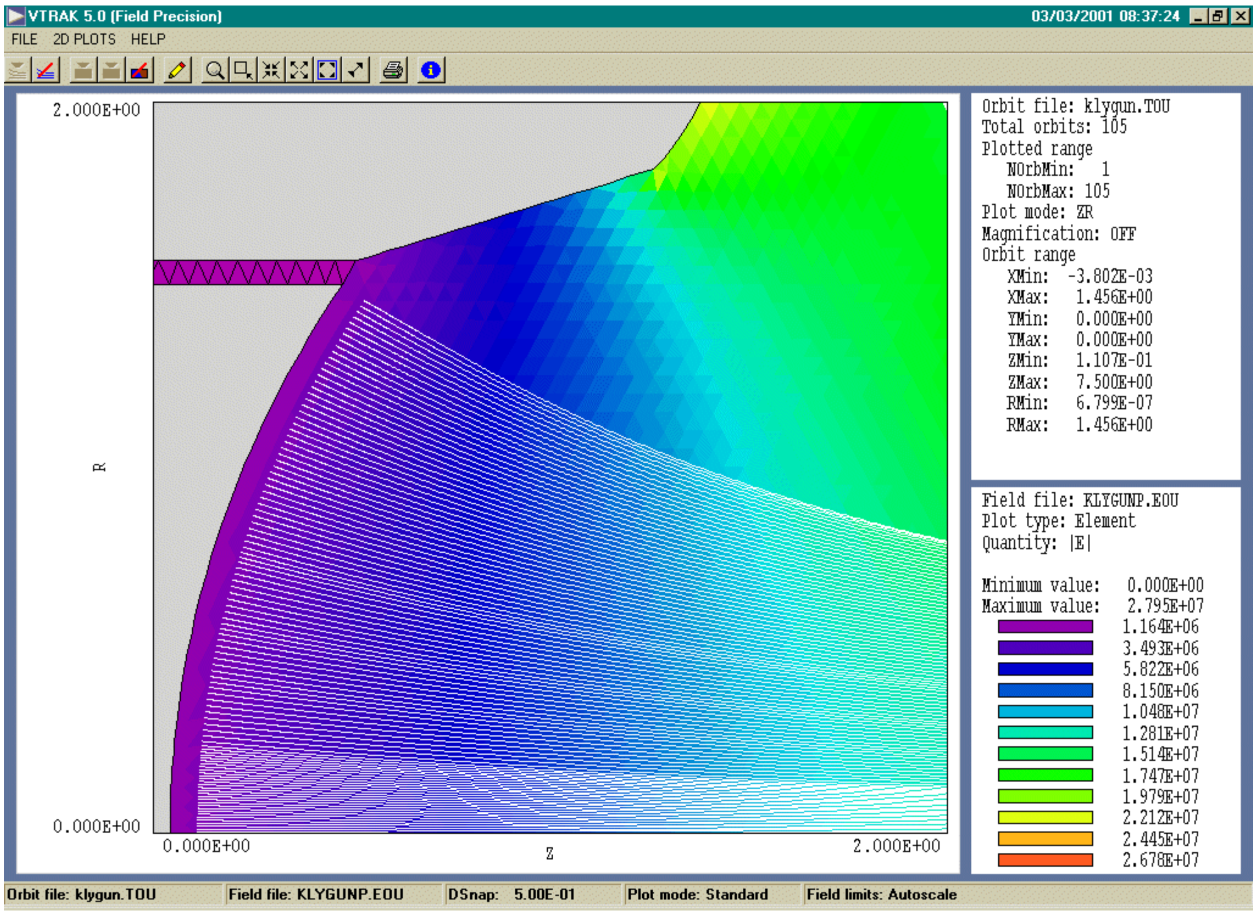
Finite-element Charged-particle Optics, Electron and Ion Gun Design

Field Precision

Copyright 2002

PO Box 13595
Albuquerque, New Mexico 87192 U.S.A.
Telephone: 505-220-3975
FAX: 505-294-0222
E Mail: techinfo@fieldp.com
Internet: http://www.fieldp.com

Table of contents



```
Orbit file: klygun.TOU
Total orbits: 105
Plotted range
   NOrbMin:   1
   NOrbMax: 105
Plot mode: ZR
Magnification: OFF
Orbit range
   XMin:  -3.802E-03
   XMax:   1.456E+00
   YMin:   0.000E+00
   YMax:   0.000E+00
   ZMin:   1.107E-01
   ZMax:   7.500E+00
   RMin:   6.799E-07
   RMax:   1.456E+00
```

```
Field file: KLYGUNP.EOU
Plot type: Element
Quantity: |E|

Minimum value:   0.000E+00
Maximum value:   2.795E+07
  🟪   1.164E+06
  🟪   3.493E+06
  🟦   5.822E+06
  🟦   8.150E+06
  🟦   1.048E+07
  🟩   1.281E+07
  🟩   1.514E+07
  🟩   1.747E+07
  🟩   1.979E+07
  🟨   2.212E+07
  🟧   2.445E+07
  🟥   2.678E+07
```

```
Orbit file: klygun.TOU   Field file: KLYGUNP.EOU   DSnap:  5.00E-01   Plot mode: Standard   Field limits: Autoscale
```

Figure 1.1. **VTrak** graphical-user-interface

Chapter 1. Introduction

1.1. Overview

Trak is a comprehensive software suite for charged-particle optics. Application areas include electron and ion guns, particle accelerators, ion sources, microwave sources, acceleration columns, electrostatic and magnetostatic lenses, vacuum tubes, and electro-optical devices. Version 6.0 of the program conforms to the standard of the Field Precision **TriComp 5.0** series of finite-element programs. **Trak** requires **Mesh 5.0** (the **TriComp** universal mesh generator) and one or more of the following solution programs to create input files: **EStat**, **BStat**, **PerMag** or **Pulse**. Install the other **TriComp** programs before installing **Trak**. You should be familiar with mesh generation and field solution procedures before reading this manual.

This manual concentrates on procedures to run **Trak** and does not give detailed explanations of the physics of charged-particle beams. A comprehensive treatment of charged-particle optics is given in two texts by the program author that are supplied with the code:

S. Humphries, Jr., Principles of **Charged Particle Acceleration** (Wiley, New York, 1986), also available for download on the Field Precision Internet site at http://www.fieldp.com/cpa/cpa.html/.

S. Humphries, Jr., **Charged Particle Beams** (Wiley, New York, 1990), also available for download on the Field Precision Internet site at http://www.fieldp.com/cpb/cpb.html/.

The latter work discusses beam distributions, emittance, beam-generated fields, propagation of high-current beams, the design of electron and ion guns and other topics. Techniques for electric and magnetic field solutions on a conformal triangular mesh are discussed in

S. Humphries, Jr., **Field Solutions on Computers** (CRC Press, Boca Raton, 1997).

1.2. Components of a Trak run

A **Trak** simulation consists of several steps. The first is the generation of conformal triangular meshes for the field solutions. **Trak** can handle independent meshes for electric and magnetic fields that overlap a common region of space. The next step is the calculation and analysis of field solutions. The final two steps are orbit tracking and solution analysis. A complete run could use several programs (such as **Mesh, EStat, VEStat, BStat, VBStat, Trak** and **VTrak**) and may require the following input files.

`EName.MIN`	(Definition of mesh for electrostatic solution)
`BName.MIN`	(Definition of mesh for magnetostatic solution)
`EName.EIN`	(Material and electrode properties for electrostatic solution)
`BName.BIN`	(Material and coil properties mesh for magnetostatic solution)
`EName.TIN`	(Initial particle parameters and run control for **Trak**)
`Name.PRT`	(Specification of an initial particle distribution)

1.3. Manual organization

Trak is a powerful tool for charged particle optics. An extensive set of control commands is required to achieve the program's versatility. As a result, this manual is more detailed than those for the other **TriComp** programs. We have tried to make the learning process relatively painless with the following features.

- Control script commands are summarized at the beginning of the chapter in which they are introduced.
- A comprehensive cross-linked help file is supplied (`trak60.html`).
- Chapter 2 gives a detailed description of a sample calculation.
- There is an extensive library of ready-to-run examples.

Chapter 3 is essential reading. It describes the organization of the **Trak** control script. The script is divided into three sections that mirror the three main activities in a simulation: 1) loading and modifying field solutions, 2) defining particle starting parameters and tracking orbits, and 3) performing diagnostics.

Chapter 3 also introduces the five tracking orbit options:

FLINE
Tracing field lines in an electrostatic solution from **EStat**. A common application is computation of charged-particle flow lines in a resistive medium.

TRACK
Single-particle tracking in applied electric and magnetic fields with no beam-generated components. Particle generation from a user-specified list. Application to low-current or neutralized beams.

SCHARGE
Non-relativistic high-current electron and/or ion beams with self-consistent space-charge effects. Particle generation from a user-specified list and/or Child-law emission surfaces.

RELBEAM
Relativistic high-current electron and/or ion beams with self-consistent effects of beam-generated electric and magnetic fields. Particle generation from a user-specified list and/or Child-law emission surfaces.

FEMIT
Electron field-emission with self-consistent beam-generated fields. Particle generation from a user-specified list and/or Fowler-Nordheim emission surfaces. Applications to vacuum microelectronics.

PLASMA
High-current ion beam generation from free surfaces of plasmas. Particle generation from a user-specified list and/or Child-law emission surfaces with shapes adjusted to provide uniform flux.

Chapters 4 through 13 discuss the preparation of the **Trak** control script, a specification of run parameters in text format. The self-documenting script allows you to reconstruct any solution quickly. It is also easy to exchange setups with colleagues using either the Windows or Linux operating systems. Chapters 4 and 5 discuss the entry and modification of applied field solutions from **TriComp** solution programs. **Trak** can also calculate applied magnetic fields from tables of on-axis values. Chapters 6 through 11 cover script commands to control the different tracking modes. Chapter 6 covers commands that control single-particle tracking. In this mode you enter a list of initial particle parameters and the program calculates orbits in the *single-particle approximation* where the effects of electric and magnetic fields generated by other particles are negligible. The approximation applies to low-current beams in strong applied fields. The **GenDist** utility (supplied with the **Trak** package) is useful for preparing input lists. The program can generate extensive particle distributions from a few simple commands. **GenDist** (described in a separate manual) can be used with both **Trak 6.0** and the three-dimensional **OmniTrak** code. Chapter 7 describes how to make precise calculations of electric field lines in solutions created by **EStat**. Chapter 8 covers commands to track orbits of non-relativistic particles in the present of beam-generated electric fields. Here you can supply a list of particle starting parameters and/or specify one or more Child-law emission surfaces for automatic particle generation. Applications include high-current beam transport, ion injectors with a known emission surface (such as a thermionic source) and low-energy electron guns (< 20 kV). Chapter 9 describes script commands for the relativistic tracking mode where the effects of both beam-generated electric and magnetic fields are included. Again particles can be specified by a list or automatically generated from emission surfaces. Chapter 10 discusses commands to model electron field emission. Here the program creates electrons over emission surfaces with current assigned according to the Fowler-Nordheim equation. You can also include electrons and/or ions through the list method. Chapter 11 describes the plasma tracking mode where ions are created from a free-plasma surface with current assigned to satisfy the Child law. The code automatically changes the shape of the surface to achieve conditions for uniform ion flux. The main application is to high-current ion beam generation from plasma sources with no applied magnetic field. Chapter 12 covers an advanced **Trak** feature, representation of secondary-electron emission. This mode has application to electron multipliers, high-power beam collectors and simulations of electron multipactor. Here secondary electrons are created when a primary electron strikes a material surface. The emission law used in the program includes effects of the primary energy distribution and angle of incidence relative to the surface. **Trak** can follow multi-generation electron histories. Chapter 13 reviews script commands that control diagnostic listings from **Trak**. The code can calculate field values that include beam contributions and particle distributions.

Chapter 14 covers operation of the **Trak** program. The code can run interactively in a window with built-in script editing capabilities and on-line help. You can also run **Trak** autonomously in the background with control by a batch file or the Field Precision **GCon** utility. Chapter 15 discusses the **VTrak** program, a post-processor for files created by **Trak**. The interactive program has extensive capabilities to create plots and to perform analyses of beam distributions. Finally, Chapter 16 describes special features of **VTrak** to analyze input and output beam distributions for **Trak** simulations. Appendix 1 (of interest to experienced users) reviews control script syntax differences between Version 5.0 and Version 6.0. Changes were necessary to support new features and to simplify the script language. The *FILE* menu of **Trak** contains an automatic syntax checker to help you update existing scripts.

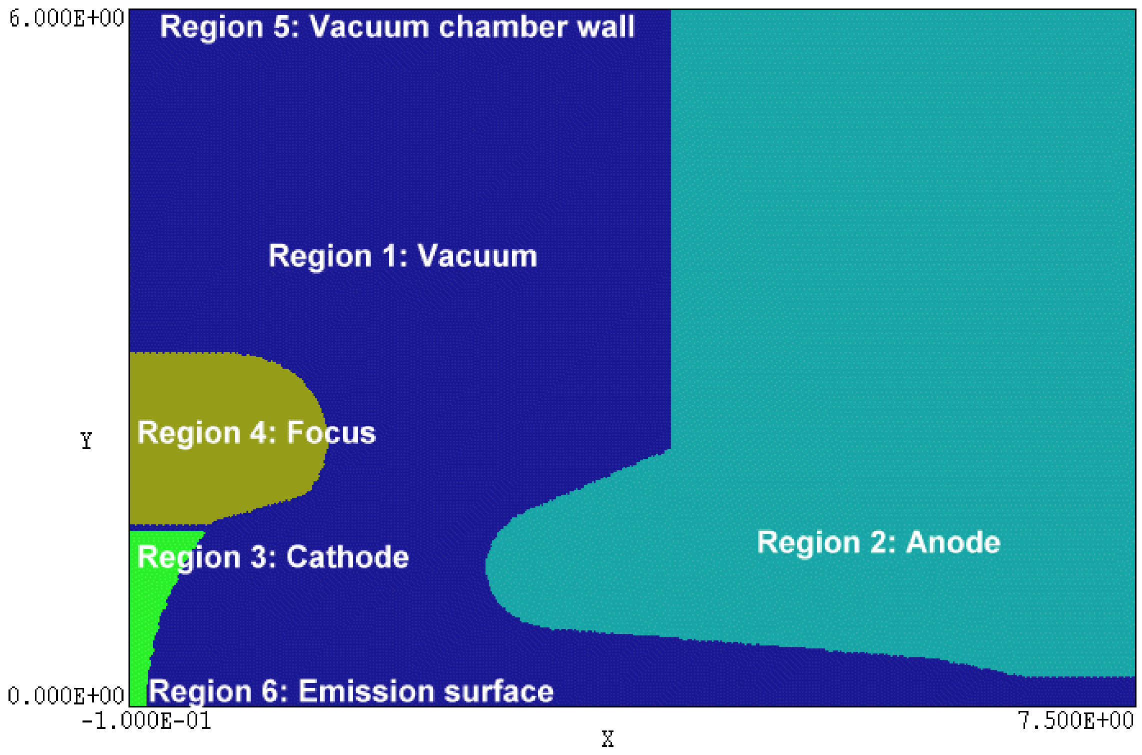


Figure 2.1. Region definitions for the `KLYGUN` example.

Chapter 2. Walkthrough example

2.1. Setup

In this chapter, we will follow an advanced **Trak** application step-by-step. The simulation determines the performance of a high-intensity, relativistic klystron gun. The gun is designed for a strong beam convergence and therefore represents a challenge to the accuracy of a numerical code. Figure 2.1 shows the geometry. An electron beam of current 460 A is extracted from a spherical-section cathode across a 660 kV acceleration gap. It is assumed that you have installed the programs `tc.exe`, `mesh.exe`, `estat.exe`, `vestat.exe`, `trak.exe` and `vtrak.exe` and have created a data directory such as `\tricomp\buffer`. Ensure that **TC** has the correct settings for the program and data directory. Move the following files to the data directory: `klygun.min`, `klygun.ein` and `klygun.tin`.

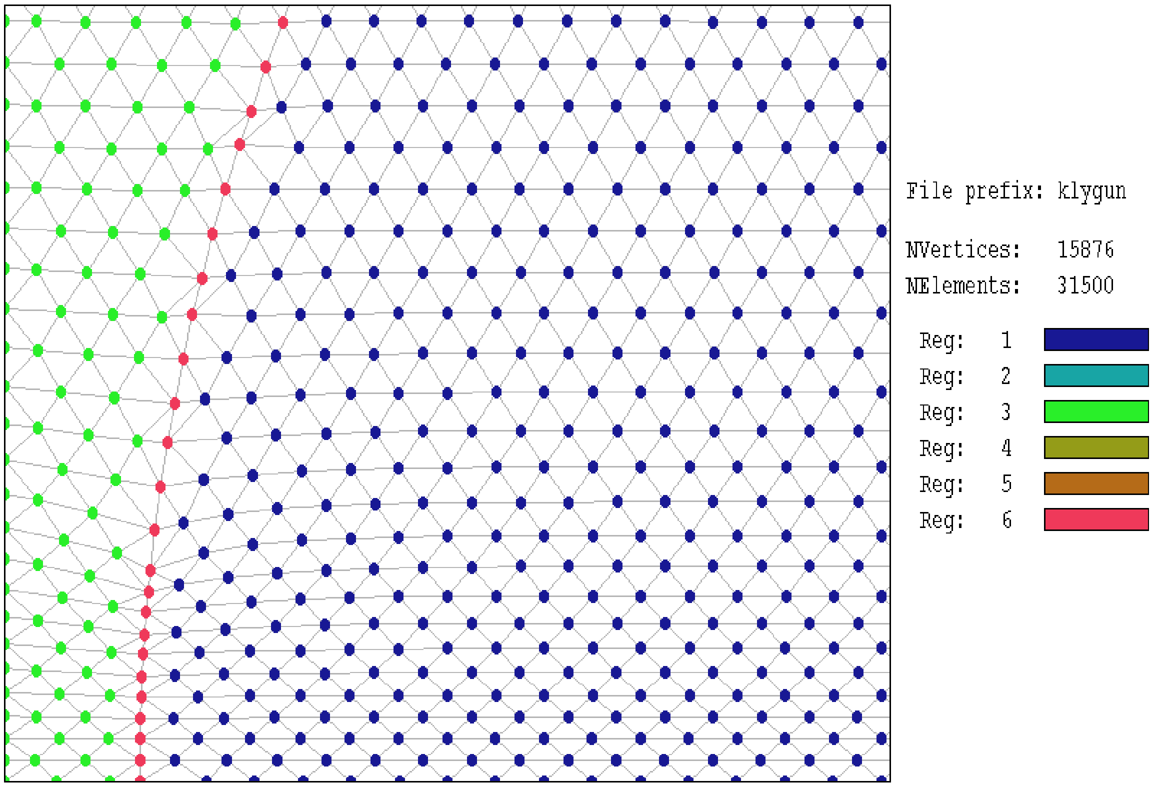


Figure 2.2. Detail of the mesh on the axis near the cathode for the `KLYGUN` example showing node identities

2.2. Defining the geometry

Table 2.1 lists the input file for **Mesh**. The file was prepared from the drawing file `KLYGUN.DXF` using the drawing editor of **Mesh**. Resolutions in the *XMesh* and *YMesh* commands were set manually. Note the fine resolution near the axis for a good representation of electric fields within the converged beam. Figure 2.2 shows a detail of the mesh at the axis near the cathode. Also note that the focus electrode (Region 4) is displaced axially a distance 0.030" to ensure that the beam was completely focused into the transport tube. The displacement was determined from previous runs. The main difference from a standard input mesh for **EStat** is the presence of Region 6, an open region that covers the surface of the cathode (nodes marked in red in Fig. 2.2). The associated nodes have the same potential as those of the cathode (Region 3), so they do not affect the electrostatic solution. The marked vertices are used in **Trak** to identify surface segments that will act as emission sites for electrons.

Table 2.1. File klygun.min

```
* TRICOMP for Windows MESH input file
*  Created with XLate
*  Date: 01/10/01
*  Time: 08:39:30
*
*  Field Precision
*  PO Box 13595, Albuquerque, New Mexico 87192
*  Telephone: 505-220-3975   FAX: 505-294-0222
*  E Mail: techinfo@fieldp.com
*  -----------------------------------------------------
GLOBAL
  XMESH
      -1.00000E-01  7.50000E+00  0.05
    END
  YMESH
       0.00000E+00  2.50000E-01  0.025
       2.50000E-01  6.00000E+00  0.05
    End
END
*  -----------------------------------------------------
REGION  FILL   SolVolume
     L   -1.00000E-01  0.00000E+00  7.50000E+00  0.00000E+00
     L    7.50000E+00  0.00000E+00  7.50000E+00  6.00000E+00
     L    7.50000E+00  6.00000E+00 -1.00000E-01  6.00000E+00
     L   -1.00000E-01  6.00000E+00 -1.00000E-01  0.00000E+00
END
*  -----------------------------------------------------
REGION  FILL   Anode
     L    7.50000E+00  2.50000E-01  6.66700E+00  2.50000E-01
     L    6.66700E+00  2.50000E-01  6.00000E+00  4.08000E-01
     L    6.00000E+00  4.08000E-01  3.08544E+00  6.69126E-01
     A    3.08544E+00  6.69127E-01  2.60001E+00  1.20235E+00  3.13300E+00  1.20000E+00
     A    2.60001E+00  1.20235E+00  2.89721E+00  1.67801E+00  3.13300E+00  1.20000E+00
     L    2.89721E+00  1.67801E+00  4.00000E+00  2.22200E+00
     L    4.00000E+00  2.22200E+00  4.00000E+00  6.00000E+00
     L    4.00000E+00  6.00000E+00  7.50000E+00  6.00000E+00
     L    7.50000E+00  6.00000E+00  7.50000E+00  2.50000E-01
END
*  -----------------------------------------------------
REGION  FILL   CathSupport
     L   -1.00000E-01  0.00000E+00  4.06420E-02  0.00000E+00
     A    4.06420E-02  0.00000E+00  4.76324E-01  1.50000E+00  2.84064E+00  0.00000E+00
     L    4.76324E-01  1.50000E+00 -1.00000E-01  1.50000E+00
     L   -1.00000E-01  1.50000E+00 -1.00000E-01  0.00000E+00
END
*  -----------------------------------------------------
REGION  FILL   FocusElect
     XShift 0.030
     L   -1.30000E-01  1.56500E+00  4.80000E-01  1.56500E+00
     L    4.80000E-01  1.56500E+00  1.23063E+00  1.81616E+00
     A    1.23063E+00  1.81616E+00  6.69000E-01  3.03800E+00  6.69000E-01  2.29800E+00
     L    6.69000E-01  3.03800E+00 -1.30000E-01  3.03800E+00
     L   -1.30000E-01  3.03800E+00 -1.30000E-01  1.56500E+00
END
*  -----------------------------------------------------
REGION   GroundedWall
     L    4.00000E+00  6.00000E+00 -1.00000E-01  6.00000E+00
END
*  -----------------------------------------------------
REGION   EmissSurface
     A    4.76324E-01  1.50000E+00  4.06420E-02  0.00000E+00  2.84064E+00  0.00000E+00
END
*  -----------------------------------------------------
ENDFILE
```

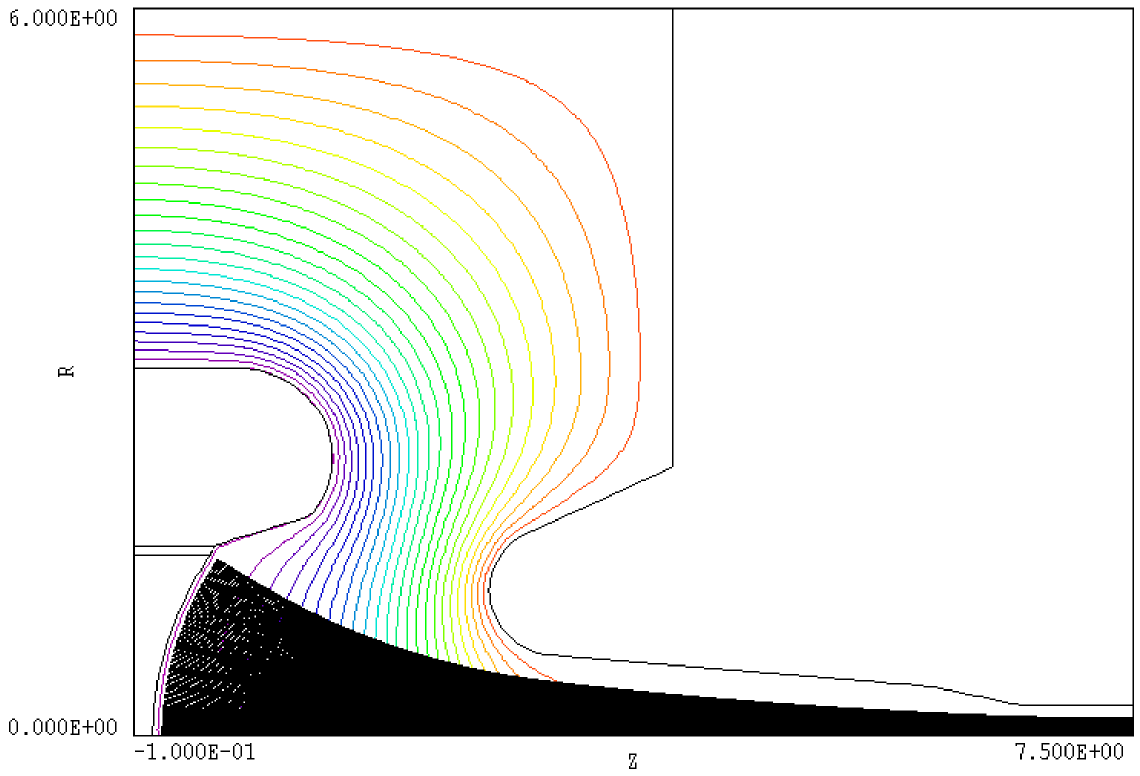


Figure 2.3. `KLYGUN` example: electrodes, equipotential lines and orbit traces

2.3. Creating the applied field solution

From **TC**, run **Mesh** and click on the command *Load script (MIN)*. If the programs and data files are in the correct locations, the file `klygun.min` should be available in the dialog. Pick the file, process the mesh, and save the output file. At this point, you can experiment with different plot functions in **Mesh** to check the geometry. Next, run **EStat** from **TC.** Click on the command *Start run* and pick the file `klygun.ein`. This file assigns the dielectric property of vacuum to Region 1 and sets the potentials of the cathode, emission surface and focusing electrode to -660.0 kV. All other electrodes are set to 0.0 kV. The solution is complete in a few seconds. The program creates an electrostatic solution file `klygun.eou` that can be inspected with the **VEStat** program. This file is one of the required inputs for the **Trak** run. The other input is the **Trak** command script listed in Table 2.2. Preparation of this file is the main subject of this manual.

Table 2.2. File klygun.tin

```
* File KLYGUN.TIN
*
FIELDS
    EFile = KLYGUN.EOU
    DUnit = 39.37
END
*
PARTICLES RELBEAM
   NCycle = 20
   Emit(6) 0.0 -1.0 0.070
   EmitParam(6) 1
   NSearch(E) = 3
   Avg = 0.20
   Dt = 1.0E-12
END
*
DIAGNOSTICS
   EDump = KLYGUNP
   BBScan  0.00  0.00  2.00 1.50  0.00  2.00
   PartList
END

ENDFILE
```

2.4. Contents of the Trak control script

The commands in the file `KLYGUN.TIN` are divided into three sections that must appear in the order shown: *FIELDS, PARTICLES* and *DIAGNOSTICS*. In the *FIELDS* section the *EFIELD* command loads the solution file that we prepared with **EStat**. Note that the *DUnit* command does not affect mesh coordinates loaded from `KLYGUN.EOU`. These quantities have already been converted to meters in **EStat**. The *DUnit* command signifies that the dimensions in script commands that involve spatial quantities (like *BBScan*) are given in inches and **Trak** will convert them to meters. The commands of the *PARTICLES* section control the orbit integration and the Child-law emission algorithm. The command *Emit(6) 0.0 -1.0 0.070* specifies that nodes associated with Region 6 define emission segments. The additional parameters signify that the emitted particles are electrons and that the computational emission surface is 0.07" from the physical surface of the cathode. The *BBScan* command of the *DIAGNOSTICS* section specifies that calculations of the beam magnetic field and current density should be included in the run listing file (`KLYGUN.TLS`). The command *PartList* calls for a record of initial and final particle parameters in the listing file. In response to the command *EDump = KlygunP*, the program creates a file `KLYGUN.EOU` that contains information on the modified electrostatic solution including effects of beam space-charge. The file is in standard **EStat** format and can be analyzed with **VEStat**.

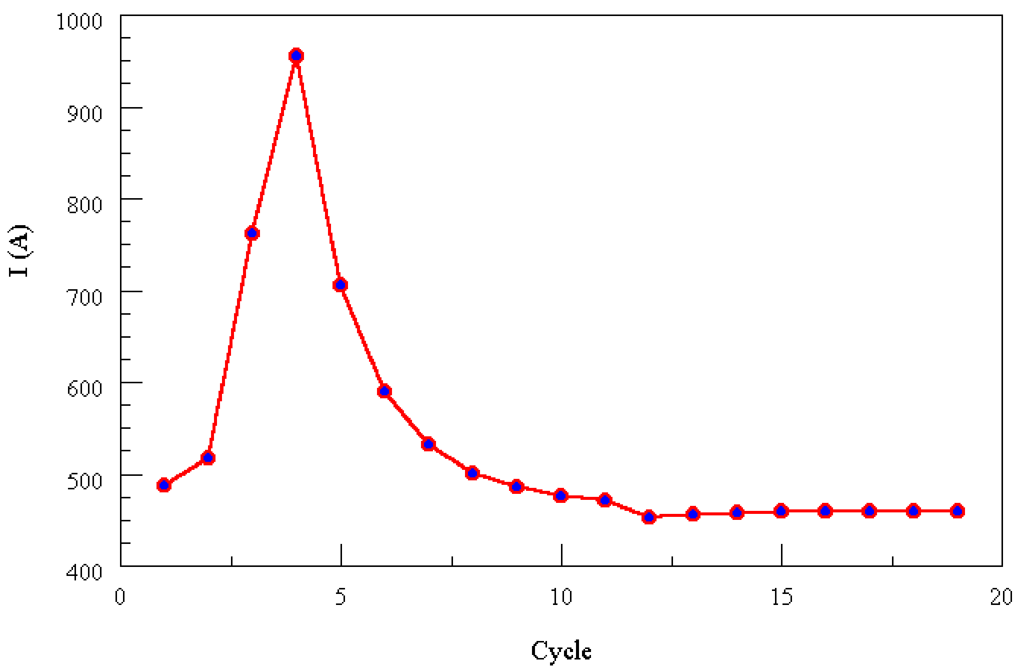


Figure 2.4. `KLYGUN` example - convergence of emitted current.

2.5. Running Trak and analyzing the results

Run **Trak** from **TC** and click on the *Start run* command. Pick the input file `KLYGUN.TIN`. The program identifies emission segments and starts one electron from each segment at an emission surface a short distance from the cathode. **Trak** performs twenty iteration cycles consisting of orbit tracking with space-charge assignment and electrical field recalculation. The run time is less than one minute on a fast computer. When the run is complete, start **VTrak** and click on the *Load orbit file* command. Pick the file `KLYGUN.TOU`, the output plot file created during the **Trak** run. Then click on the *Load electric field file* command and pick the file `KLYGUNP.EOU`. You should see a plot similar to Figure 2.3 showing self-consistent orbit traces superimposed on equipotential lines of the modified field solution.

Trak can record a wide variety of useful data in the listing file. Figures 2.4 and 2.5 illustrate the type of information that can be extracted. Figure 2.4 shows the total current emission from the cathode as a function of the iteration cycle. With proper averaging, the code converges to an equilibrium where the particle orbits and consequent space-charge density are consistent with the total electric fields. Figure 2.5 shows radial current density profiles at different distances (in inches) from the cathode. Because of over-focusing on the outer edge, the compressed beam has a hollow profile.

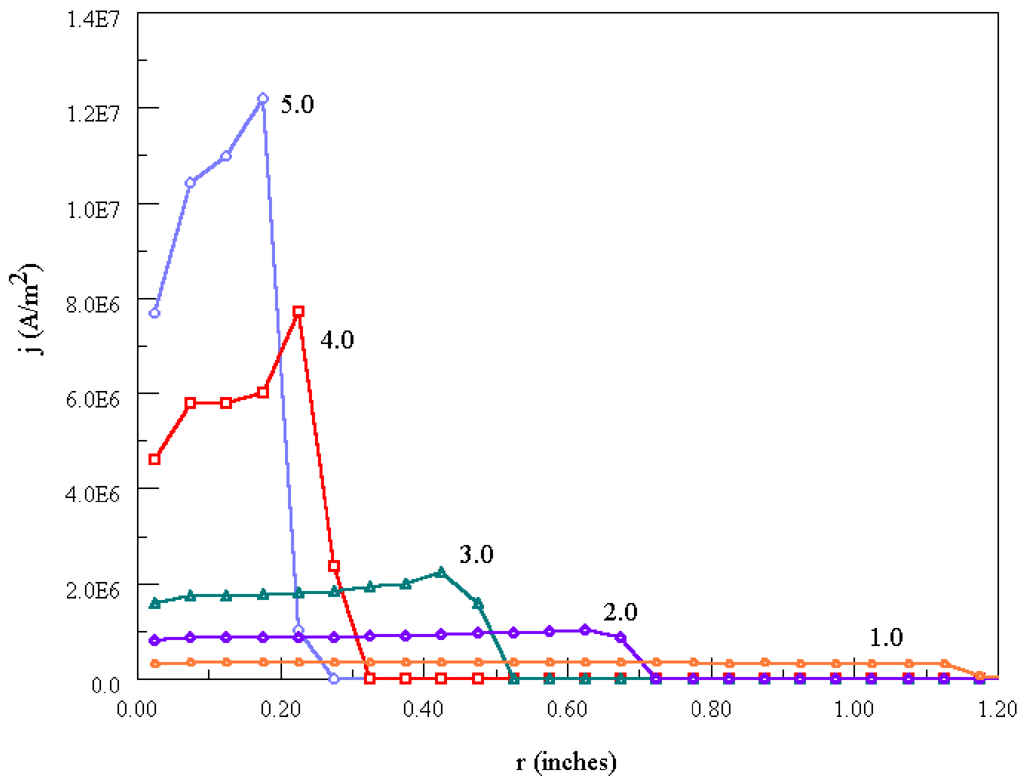


Figure 2.5. `KLYGUN` example - current density profiles at different distances from the cathode (dimensions in inches)

3. Program organization

3.1. Simulation strategy

Learning **Trak** requires some time and effort. Solutions that involve single particle tracking are relatively easy. On the other hand, the preparation and interpretation of space-charge and field emission problems requires experience and judgment. As with any advanced technical program, the best learning method is to study template examples. We have supplied a library of examples on disk spanning the full range of program capabilities. The chapters of this manual are also arranged in order of increasing complexity.

Trak calculates the orbits of point charged particles moving through electric and magnetic fields. These fields may result from external charges and currents (such as the charge induced on biased metal electrodes), surface charges in nearby dielectrics, or current in magnet coils. Such fields are called *applied fields*. For low-intensity beams, it is sufficient to determine the applied fields and then to track any number of particle orbits in space. We use the term *single particle orbits* when the fields created by the charge and effective current of the particles are negligible. In this case, particle orbits are independent of each another and it is unnecessary to modify the applied field to reflect the presence of the beam.

Field evaluations and orbit calculations in **Trak** are entirely numerical. Field calculations use the *finite element method* [see, for instance, S. Humphries, **Field Solutions on Computers** (CRC Press, Boca Raton, 1997)]. Here, a bounded spatial region is divided into small segments – the **TriComp** programs use triangular elements. The sizes and shapes of the triangles are adjusted to fit the boundaries of physical objects like electrodes (Figure 2.2). The geometry of the set of triangle nodes (vertices) is called the *computational mesh*. In electrostatic calculations with the **EStat** code the potential is determined at the nodes and the material characteristics (such dielectric constant, space charge, or constant potential) are assigned to the triangle volumes. A **Trak** run may also include an independent applied magnetic field solution with a different computational mesh fitted to the boundaries of magnetic objects (such as coils or ferrites). **Trak** accepts files created by the **BStat**, **PerMag** and **Pulse** programs. The magnetic solution consists of values of vector potential at the nodes. A single-particle orbit calculation is conceptually simple. The orbit advances in small time steps. At any time, the program must identify the element that contains the particle. With this knowledge it is possible to collect potential values at neighboring nodes and take a spatial derivative to estimate electric fields. Similarly, a knowledge of the occupied triangle in the magnetic mesh leads to the magnetic fields at the particle position. Given the fields and the initial particle momentum, we can advance the orbit. We can also check whether the particle is outside the mesh or in a non-vacuum triangle in order to stop the calculation. A **Trak** calculation always follows an **EStat** and/or **BStat/PerMag/Pulse** solution. Even if there is no applied field, we need to create a computational mesh for the calculation of beam-generated fields.

TriComp solution programs handle fields that vary in two dimensions in either cylindrical structures (variation in *r* and *z*, uniform in θ) or planar structures (variation in *x* and *y*, infinite length along *z*). **Trak** computes particle orbits in three-dimensional Cartesian coordinates (*x*,*y*,*z*) using the two-dimensional field components available from the **TriComp** solution programs. For rectangular problems, the non-zero field components are E_x and E_y or B_x and B_y. Cylindrical solutions generate the components E_r and E_z or B_r and B_z. The radial components are used to derive the Cartesian field components according to

$$E_x(x,y,z) \ = \ B_r(r,z) \ \frac{x}{r}$$

$$E_y(x,y,z) \ = \ B_r(r,z) \ \frac{y}{r}$$

$$B_x(x,y,z) \ = \ B_r(r,z) \ \frac{x}{r}$$

$$B_y(x,y,z) \ = \ B_r(r,z) \ \frac{y}{r} \ .$$

3.2. Beam-generated fields

Orbit calculations are more difficult for high-current beams. In this case, the space-charge of the beam can contribute to electric fields in the propagation region and affect the distribution of image charge on surrounding electrodes. For this reason, **Trak** has the capability to update the electrostatic field using information on the space-charge density associated with the particle flow. This procedure is called a *self-consistent orbit calculation*. A beam is represented by a collection of model particles, each carrying a fraction of the beam current, Δ*I*. At each time step Δ*t* the space-charge of the triangle occupied by a ray is augmented by an amount Δ*I*Δ*t*. **Trak** then recalculates the electric field with the extra space-charge. The problem with this approach is that the final orbits of the particles need not be the same as those used to calculate the space-charge. The resolution is to iterate for several cycles using suitable space-charge averaging. This procedure usually converges to the correct solution, even for intense beams.

High-current charged-particle beams have little effect on solenoid-type magnetic fields (B_z and B_r in cylindrical problems, B_x and B_y in planar geometry). Therefore, **Trak** does not modify the applied magnetic field solution. On the other hand, relativistic beams generate magnetic field components of B_θ (cylindrical) or B_z (rectangular) that may strongly influence particle dynamics. To address this problem, **Trak** has the capability to find spatial variations of beam current and beam-generated magnetic fields on an independent mesh (*RELBEAM* tracking mode).

3.3. Tracking modes

Trak has two options for user-controlled entry of initial particle orbit parameters. The simplest is the list option. Here, you supply a list of parameters (species, position, kinetic energy, direction) for up to 2500 particles. If the particles have zero current, **Trak** performs single particle orbit calculations. When the particles carry current, the program calculates and averages space charge over several cycles to find the self-consistent electric fields (**SCharge** option). For relativistic beams, **Trak** also calculates the beam current distribution and magnetic focusing forces. The list option is useful for transport calculations with a known input beam distribution. It is also applicable to electron/ion guns where the current is source-limited (such as a thermionic cathode at low temperature).

It is not possible to specify particle parameters *a priori* if the current density of a high-current beam depends on the value of the electric field at the source. For this reason, **Trak** has two options for automatic assignment of particle starting positions and field-dependent current over specified emission surfaces. In the *SCHARGE* and *RELBEAM* tracking modes the program starts particles and assigns current following the Child law to model space-charge-limited emission. The program automatically sets starting particle positions and momenta and applies a cyclic process to determine the appropriate current. The criterion for space-charge-limited emission is that the electric field approaches zero on the source surface. In the process, **Trak** finds the full self-consistent electric fields over the propagation volume. In the *FEMIT* tracking mode **Trak** handles electron field emission from metals or semi-conductors. In this case the program assigns initial electron positions over a given emission surface and then calculates the effective current from the Fowler-Nordheim equation.

Iterative space-charge corrections can also be added. Table 3.1 summarizes properties of the six tracking options of the program.

Table 3.1. Tracking modes	
FLINE	
Function	Precision tracking of electric field lines
Input	User-prepared list of starting positions, from 1 to 2500 field lines
Characteristics	Tracks field lines in an **EStat** solution, high-accuracy calculation of stopping points on electrodes
Applications	Ion or electron trajectories in conductive solutions or collisional plasmas
TRACK	
Function	Single-particle orbit calculations
Input	User-specified tables of starting parameters, particle output files from previous runs, output from the **GenDist** program
Characteristics	No beam-generated fields, option for mixed particle species
Applications	Electron or ion transport in low-current devices (photomultipliers, spectrometers, CRT guns,...)
SCHARGE	
Function	High-current guns for and transport of intense, non-relativistic beams. Significant beam-generated electric fields
Input	Specified surfaces for space-charge-limited emission, user-specified tables of starting parameters, particle output files from previous runs, output from the **GenDist** program.
Characteristics	Beam-generated electric fields, option for mixed particle species
Applications	Ions guns and low-voltage electron guns. High-current transport systems.

Table 3.1. Tracking modes

RELBEAM	
Function	High-current guns for and transport of intense relativistic electron beams. Significant beam-generated electric and magnetic fields
Input	Specified surfaces for space-charge-limited emission, user-specified tables of starting parameters, particle output files from previous runs, output from the **GenDist** program.
Characteristics	Beam-generated electric and magnetic fields, option for mixed particle species
Applications	Guns or transport systems for intense electron beams, option for ion backflow
FEMIT	
Function	Automatic generation of electron distributions to model field emission from metal and semiconductor surfaces
Input	Specified surfaces for electron field emission, user-specified tables of starting parameters for ions or electrons.
Characteristics	Electron emission only, although ions can be added with the list method. Beam-generated electric fields
Applications	Electron microscopes, flat surface displays, vacuum microelectronics, ...
PLASMA (Not yet implemented, 04/03)	
Function	Generation of high-current ion beams from non-magnetized plasma sources
Input	Ion generation from specified plasma surfaces adjusted to satisfy the conditions of space-charge-limited emission and uniform flux. User-specified tables of starting parameters, particle output files from previous runs, output from the **GenDist** program.
Characteristics	Ion emission only from flexible plasma surfaces. Option to add electrons or other ions via the list method.
Applications	High-current ion guns with plasma sources

3.4. Input and output files

Trak requires three types of input data

- Electric and/or magnetic field information
- Program control parameters
- Initial particle orbit characteristics

The program gets field information from the output solution files of **EStat**, **BStat**, **Pulse** or **Permag**. These files have the names `FPREFIX.EOU, FPREFIX.BOU`, or `FPREFIX.POU`. A **Trak** run may include electric fields – in this case, the input file is called the *EFile*. Similarly, magnetic field information can be obtained from a **BStat**, **Pulse** or **PerMag** solution (called the *BFile*). A run can combine information from a single *EFile* and single *BFile*. Version 6.0 of **Trak** can also calculate magnetic fields from tables of on-axis field values using paraxial expansions.

An input command script controls operation of **Trak**.. This text file is similar to the command files for **EStat** (`FPREFIX.EIN`) and other solution programs. It consists of comment lines and a series of commands with parameters. The **Trak** input command file has a name of the form `FPREFIX.TIN`. Under the **Track** or **SCharge** options, you may include information on particles (species, position, kinetic energy, direction or current) in the input command file or you can prepare a separate file. The particle file has a name of the form `FPREFIX.PRT`. This file may be an output file from a previous **Trak** run or a text file prepared by a spreadsheet program.. There are two other types of input files allowed in Version 6.0:

Magnetic field table. The program can determine magnetic field values from tables of on-axis field values: $B_z(0,z)$ or $B_x(0,x)$.

Modulation function table. The table defines an arbitrary time variation that can be superimposed on electric or magnetic fields under the *TRACK* option.

Trak uses the prefix of the input command file to name the output files. All output files are in ASCII format and can be inspected with a text editor. **Trak** always makes a text listing file with the name `FPREFIX.TLS`. The listing file contains extensive information about the run. Some items are always included in the file – others are optional, depending on the commands in `FPREFIX.TIN`. **Trak** normally writes a text file of orbit vectors with the name `FPREFIX.TOU`. This file transfers data to **VTrak**, the graphics analysis program. It is relatively easily to port the formatted file to your own graphics programs. The remaining output file is a text listing of final particle parameters. This file has the name `FPREFIX.PRT`. Commands in `FPREFIX.TIN` control creation of this file. Table 3.2 summarizes Trak files and name conventions. The prefixes of all files should be standard strings from 1 to 20 characters in length.

Table 3.2. Trak Input and Output Files

File Name	Function	Input/Output	Status
RUNNAME.TIN	Control script	Input	Required
ENAME.EOU, BNAME.BOU, BNAME.POU	Field and mesh information	Input	One or more required
FPREFIX.PRT	Particle parameters	Input	Optional
Any name	On-axis magnetic field table	Input	Optional
Any name	Temporal modulation functoin	Input	Optional
FPREFIX.TLS	Data listing	Output	Always created
FPREFIX.TOU	Orbit traces for plotting	Output	Optional
FPREFIX.PRT	Particle parameters	Output	Optional

3.5. Structure of the Trak control script

A command file `FPREFIX.TIN` is required for each **Trak** run. There are three main sections: *FIELDS*, *PARTICLES* and *DIAGNOSTICS*. The file has the following structure.

```
*
Fields
   (Commands)
End
*
Particles Track
   (Commands)
End
*
Diagnostics
   (Commands)
End
*
EndFile
```

The three sections must appear in the order shown – a section must be terminated with an *END* command. Each section has a set of allowed commands. Within a section, allowed commands can usually be entered in any order. **Trak** begins processing a section when all commands have been read. The *ENDFILE* command closes all files and stops the program.

The commands in the *FIELDS* section control input of field solution files from other **TriComp** programs. There are also commands to adjust field values, introduce time variations, add constant magnetic field components, adjust the potential of individual electrodes, set distance units, or set up a mesh to calculate beam-generated magnetic fields. **Trak** automatically sets computational boundaries from the field file information.

The *PARTICLES* section controls orbit computations. A string parameter gives the type of calculation: *Field, Track, SCharge, SCEmit* or *FEmit*. The allowed commands in the *PARTICLES* section depend on the type of calculation – **Trak** stops with an error message if it finds an inappropriate command. The commands in the *PARTICLES* section serve three functions.

- Controlling orbit calculations (global boundaries, mesh search options, time step, listing options, ...)

- Setting parameters to initiate particle orbits (particle parameters, marked emission surfaces, ...)

- Setting stopping criteria

Trak reads all commands in the *PARTICLES* section, processes the information, and then finds the orbits. Depending on the type of calculation, the program may also update the potential and recalculate orbits over a number of cycles.

The commands in the *DIAGNOSTICS* section control special output listings of information on fields and calculated orbits. Version 6.0 of **Trak** can make scans of applied magnetic fields, self-consistent electric fields and beam-generated magnetic fields. The program makes formatted listings and output files of initial and final particle parameters. The program can perform automatic analyses of beam distributions. Distribution calculations can also be carried out in the **VTrak** program if the **Trak** run creates an output particle file.

4. Loading and modifying applied electric field solutions

Commands introduced in this chapter

DUnit
Boundary
EFile
Shift
ModFunc
ChangePot
MaxCycle
ResTarget
Omega

4.1. General commands

The first step in a **Trak** run is to define electric and/or magnetic fields by reading one or more **TriComp** solution files. The command of the *FIELDS* section control this function. This chapter introduces commands to load and to modify electric field solutions from **EStat**. Commands are displayed with symbolic parameters and also in the form they would appear in the script file.

DUNIT DUnit
DUNIT = 39.37

This command controls the interpretation of spatial coordinate input for all other commands in the input script. The quantity *DUnit* (real) is the number of distance units per meter. For example, to enter positions in cm, set *DUnit* = 100.0. The *DUNIT* command can appear anywhere in the script and affects all following commands. Note that the values of *DUnit* defined in the **EStat** or **BStat** input scripts set coordinates for the input meshes but have no effect on position quantities entered in the **Trak** script. The quantity *DUnit* is recorded in the `TOU` plot file and used for coordinate display in **VTrak**.

BOUNDARY X1 Y1 X2 Y2
BOUNDARY Z1 R1 Z2 R2
BOUNDARY = (0.0, 0.0, 20.0, 5.0)

Orbit calculations are performed inside a two-dimensional solution box with opposite corners defined by (X_1, Y_1,)-(X_2, Y_2) for planar solutions or (Z_1, R_1,)-(Z_2, R_2) for cylindrical geometry. Enter coordinates in units set by *DUnit*. Particle orbits terminate if they move outside the solution box. **Trak** interpolates the final orbit step so that the stopping point lies exactly on the box surface. If the *BOUNDARY* command does not appear the program sets the solution volume as the largest box that overlaps the electrical and/or magnetic solutions. When the *BOUNDARY* command appears **Trak** sets the *ballistic flag*. In this case orbits continue even if they leave the boundaries of the electric and/or magnetic solution volume. The program takes **E** = 0 and **B** = 0 in regions that are inside the orbit calculation box but outside the field solution volumes. The ballistic mode is useful, for example, if you want to trace orbits to a focal point removed from a lens with localized field.

4.2. Loading and modifying EStat solutions

EFILE FileName [EMult]
EFILE = KlyGun.EOU

This command loads an electric field solution from **EStat**. The quantity *FileName* is the full name of the file (*i.e.*, `EFTEST.EOU`). The optional real-number parameter *EMult* is a global scaling factor. All values of electrostatic potential at nodes are multiplied by *EMult* when they are loaded into the program. This factor is useful if you want to investigate scaling with applied voltage without regenerating the **EStat** solution. The **EStat** file must be available in the working directory. The *z*-axis in the field solution corresponds to the *z*-axis for particle tracking in **Trak**. A solution with cylindrical symmetry has field components E_r and E_z. The program determines E_x and E_y for tracking from E_r given the particle position. A planar solution has field components E_x and E_y with $E_z = 0.0$.

Note: In runs with both electric and magnetic fields, the field solutions must have the same symmetry – planar or cylindrical. In cylindrical simulations, both the electric and magnetic solutions must share the same *z* axis.

SHIFT E ZEShift
SHIFT(E) = -5.67

The *SHIFT* command with the string parameter *E* moves nodes in the electric field mesh along the *x* direction (planar) or *z* direction (cylindrical) according to

$$z_{\mathrm{new}} = z_{\mathrm{old}} + ZEShift.$$

Enter the real-number parameter *ZEShift* in the current units set by the *DUNIT* command. As an example, you could use this command to make small changes in the position of the electric solution relative to a magnetic solution to optimize beam matching to a magnetic focusing system. The command is also useful in simulations of particle motion through a periodic focusing system. You can split the particle simulation into several stages with shifted field solutions.

MODFUNC E TABLE FileName [TOff TMult FOff FMult]
MODFUNC(E,TABLE) = StepFunc.WAV

You can use the *MODFUNC* command with the keywords *E* and *TABLE* to add an arbitrary temporal modulation of electric fields. This capability can only be applied in the *TRACK* particle tracking mode. The quantity *FileName* is the full name of a file in the current directory with data lines that define the time variation of a function:

```
  t1   fe(t1)
  t2   fe(t2)
  t3   fe(t3)
     ...
  tn   fe(tn)
ENDFILE
```

where *n* is less than or equal to 256. If the optional real-number parameters *TOff, TMult, FOff* and *FMult* appear, values in the table are modified according to

t_{code} (seconds) = $TMult{*}t_{\mathrm{tab}} + TOff,$

f_{code} (seconds) = $FMult{*}f_{\mathrm{tab}} + FOff.$

The modulation file follows the standard rules for **TriComp** tabular functions. You can use any of the standard delimiters to separate quantities on the line including *Space* and *Tab*. Comment lines (beginning with an asterisk) can be included. Note that all particle orbits start from their initial position at $t = 0.0$. Using the elapsed time t of an orbit, the program applies a cubic spline interpolation to find $f_e(t)$. The table values should define a smooth function with continuous first derivatives. All electric field components are then multiplied by $f_e(t)$.

By default, **Trak** interprets the table as a periodic function. If the elapsed time of an orbit exceeds the maximum time on the table, **Trak** goes back to the beginning. To illustrate, if the orbit time is $t = 5.0$ ns and if the maximum time value in the table is $t_n = 4.0$ ns, then f_e is evaluated at 1.0 ns. If you want to apply a single electric-field pulse with zero fields thereafter, be sure that the final entry in the table is of the form

```
tn   0.0
```

where t_n is larger than the longest particle transit time.

MODFUNC E SIN A0 A1 FREQ TO
MODFUNC(E) = (0.0, 0.25, 2.0E5, 1.0E-6)

The *MODFUNC* command with the keywords *E* and *SIN* is used to add an harmonic temporal modulation of electric fields. This capability can only be applied in the *TRACK* particle tracking mode. The modulation function is defined by

$$f_e(t) = A_0 + A_1 \sin[2\pi f (t-t_o)].$$

Enter the parameter *FREQ* (f) in Hz and *T0* (t_0) in seconds. Note that the harmonic variation extends over the full transit time for each orbit.

CHANGEPOT RegNo PotNew
CHANGEPOT(6) = 25000.0

This command can be used to change the potential of a individual electrode. The quantity *RegNo* (integer) corresponds to the region number defined in **Mesh**. The region must have a fixed-potential (set by the *POTENTIAL* command in **EStat**). The quantity *PotNew* (real) is the new value of potential (in volts). You can include multiple *CHANGEPOT* commands in the file. After all electrode values are set, the program performs a relaxation operation to correct values of potential at intervening variable points.

Note: If you modify the electric field solution with the *EMult* parameter, *SHIFT* command or *CHANGEPOT* command, you should be careful when displaying electric fields in **VEStat**. If you load the original applied field, the field display may not correspond to fields used to compute the particle orbits. You should use the *EDUMP* command to make a record of the modified electric field used in the **Trak** calculation and then display this file in **VEStat**.

The remaining commands control the iterative procedure used to update the electric field solution. Updates are required if the *CHANGEPOT* command appears or if effects of beam space charge must be included (*SCHARGE*, *RELBEAM*, *FEMIT* and *PLASMA* tracking modes).

4.3. Controlling electric field recalculation

MAXCYCLE MaxCy
MAXCYCLE = 2500

The interger quantity *MaxCycle* is the maximum number of successive over-relaxation cycles in the iterative electric field solution. Larger values usually give higher accuracy. The default values are *MaxCycle* = 2500 for initial adjustments in response to the *CHANGEPOT* command and *MaxCycle* = 250 for field relaxations on each tracking cycle in the *SCHARGE* , *RELBEAM*, *FEMIT* and *PLASMA* tracking modes.

OMEGA Omega
OMEGA = 1.95

The quantity *Omega* is the successive over-relaxation factor for the electric field solution. It should have a value in the range 0.0 to 2.0. Values close to 2.0 usually give faster convergence. Lower the parameter if the solution fails to converge. If the *OMEGA* command does not appear, **Trak** uses default values determined by the Chebyshev prescription.

RESTARGET = ResTarget
RESTARGET = 1.0E-8

The *residual* is the relative error in the electric field solution and should approach a small value compared to unity. The electric field relaxation terminates if the residual falls below the quantity *ResTarget* (real) or if the number of relaxation cycles exceeds *MaxCy*. The default value is *ResTarget* = 5.0×10^{-7}.

5. Loading and modifying applied magnetic field solutions

Commands introduced in this chapter
BFile
BUni
BTable
BTheta

5.1. Loading and magnetic solutions from TriComp programs

The commands to load and to modify magnetic field solutions created by **BStat**, **PerMag** or **Pulse** are similar to those used for electric field files.

BFILE: FileName [BMult]
BFILE = MLENS.POU (1.4)

This command loads a magnetic field solution from **BStat**, **PerMag** or **Pulse**. The quantity *FileName* is the full name of the file (*i.e.*,`SOLENOID.POU`). The optional real-number parameter *BMult* is a global field scaling factor. Values of vector potential at all nodes are multiplied by *BMult* when they are loaded into the program. This factor is useful if you want to investigate scaling with applied magnetic field without regenerating the **BStat** solution. The solution file must be available in the working directory. The *z*-axis in the field solution corresponds to the *z*-axis for particle tracking in **Trak**. A solution with cylindrical symmetry has field components B_r and B_z. The program determines B_x and B_y for tracking from B_r given the particle position. A planar solution has field components B_x and B_y with $B_z = 0.0$.

SHIFT B ZBShift
SHIFT(B) = 0.05

The *SHIFT* command with the key symbol *B* moves nodes in the magnetic field mesh along the *x* direction (planar) or *z* direction (cylindrical) according to

$$z_{\text{new}} = z_{\text{old}} + ZBShift.$$

Enter the real-number parameter *ZEShift* in the current units set by the *DUNIT* command.

MODFUNC B TABLE FileName
MODFUNC(B,TABLE) = SlowRise.PLS
MODFUNC B SIN A0 A1 FREQ TO
MODFUNC(B) = (0.0, 0.25, 5E6, 0.0)

The *MODFUNC* command with the key symbol *B* adds time variations to magnetic fields. Modulation functions were described in Chapter 4. The capability can be applied only in the *TRACK* particle tracking mode.

Note: The modulation function is applied to the total magnetic field calculated from all sources.

Note: **Trak** simply multiplies static field values by the modulation function and makes no checks that the resulting time-dependent fields approximate a solution to Maxwell's equations. You must ensure that the field values are physically valid.

5.2. Alternate magnetic field sources

In contrast to electric fields, **Trak 6.0** can combine several sources of magnetic fields. The field components are additive when multiple sources are defined.

BTABLE = FileName ZOffset ZMult BOffSet BMult
BTABLE = Bucking.DAT (2.0, 1.0, 0.0, 300.0)

Trak can derive fields from expansions based on a table of values of the on-axis magnetic field in both planar and cylindrical geometries. The following is an example of a magnetic table .

```
* Table from SOL_LENS.BOU
*      z           Bz
* ======================
SYMMETRY: Cylin
     -9.000E+00   4.421E-06
     -8.852E+00   5.926E-05
     -8.703E+00   1.173E-04
     -8.555E+00   1.780E-04
       ...
      9.258E+00   1.014E-04
      9.406E+00   7.838E-05
      9.555E+00   5.670E-05
      9.703E+00   3.685E-05
      9.852E+00   1.767E-05
      1.000E+01   7.111E-11
ENDFILE
```

The table may contain comment lines, data lines, an *ENDFILE* command and a *SYMMETRY* command. The *SYMMETRY* command must be the first non-comment line in the file and has the parameters *CYLIN* or *RECT*. In the *CYLIN* option, the data lines contain values of z and $B_z(0,z)$. In the *RECT* option enter x and $B_x(0,x)$. The adjusted values of z should be in the current spatial units defined by *DUnit* and $B_z(0,z)$ should be in tesla. The file may contain up to 256 data and must terminate with the *ENDFILE* command.

You can optionally supply four real numbers as parameters in the command: *ZOffSet, ZMult, BOffSet* and *BMult*. The parameters modify table values entered in the program according to

$$z_{prog} = ZMult \times z_{tab} + ZOffSet,$$

$$B_{zprog} = BMult \times B_{ztab} + BOffSet$$

Note: **Trak** uses cubic spline interpolation to analyze the on-axis magnetic field. The method supplies only first and second derivatives. Therefore, the quantities $B_r(r,z)$ and $B_y(y,x)$ vary linearly in r or y. You must use a finite-element magnetic field solution to investigate effects of non-linear field variations.

BUNI = Bx0 By0 Bz0)
BUNI = (0.25E-4 0.0 0.0)

This command defines spatially-uniform magnetic field components. The feature could be used, for example, to simulate the effects of the earth's magnetic field on an electro-optical device. Enter magnetic field values in tesla. Note that uniform field components can be used only in the *TRACK* mode.

BTHETA AxisCurrent RWire
BTHETA = 1200.0 (0.001)

This command adds a toroidal magnetic field generated by an on-axis wire. One application of the feature is modeling wire transport of an intense electron beam. The field is uniform in *z*. The quantity *AxisCurrent* is the wire current in amperes. Use a negative value for a current in the -*z* direction. The quantity *RWire* is the wire radius in units defined by *DUnit*. The wire radius does not affect the field calculation. It is used to estimate particle losses on the wire. An orbit stops if it reaches a radius less than *RWire*.

6. Single particle tracking

Commands introduced in this chapter

PList	**Material**
PFile	**Stop**
DUnit	**Diag**
Dt	**Record**
TMax	**Reflect**
NTrackMax	**ListOn**
DMax	**OrbInfo**
NSearch	**PlotOff**
Interp	**Emit**
Vacuum	**Start**

6.1. Command functions in the TRACK mode

In this chapter we proceed to commands that may appear in the *PARTICLES* section of the input script under the *TRACK* mode:

```
PARTICLES TRACK
    ...
    (Commands)
    ...
END
```

In this mode **Trak** calculates orbits in the *single-particle* limit. The term implies that the fields created by the particles are negligible compared to the applied fields (*i.e.*, low-current beams). In this case each particle can be treated independently and the calculation of orbits is a straightforward process. Allowed commands in the *PARTICLES TRACK* section serve five functions:

- Set starting points for particle orbits
- Control orbit integrations
- Define region material properties relevant to orbit tracking
- Set conditions for orbit stopping
- Control diagnostic listing that can be generated during orbit integrations

Many of the commands in the last four categories appear in all particle and field-line tracking modes.

6.2. Starting particles from a list

The most straightforward way to start particles in the *TRACK* mode is through list input. Here you specify the particle species, kinetic energy, start position and direction of from 1 to 2500 particles. Two commands are used for list input.

PLIST

This command signals that a list of starting conditions follows in the control script.

The following example illustrates a standard particle list:

```
PLIST
*    Mass Chrg   Eng       x    y      z    px    py    pz
*   =======================================================
     0.0  -1.0 0.7399E6  0.1  0.0   -8.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.2  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.3  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.4  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.5  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.6  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.7  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.8  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  0.9  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.0  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.1  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.2  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.3  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.4  0.0   -9.0  0.00  0.00  1.00
     0.0  -1.0 0.7399E6  1.5  0.0   -9.0  0.00  0.00  1.00
END
```

The maximum number of lines is 2500. Comment lines (starting with an asterisk) may be included. Each data line contains nine real numbers to represent the following quantities:

Mass. The particle mass in AMU (one atomic mass unit corresponds to 1.65979×10^{-27} kg). An entry of 1.0 designates a proton. If 0.0 appears in the column, the program inserts the value for an electron. A run in the *TRACK* mode may contain particles with different values of *Mass* and *Charge*.

Charge. The particle charge in units of e (1.60210×10^{-19} coulomb). An ion has charge +1.0 and an electron has charge -1.0.

Energy. The initial particle energy in eV (1 electron volt = 1.60210×10^{-19} joules).

x, y, z. The particle position in units set by the current value of *DUnit*.

px, py, pz. Normalized momentum fractions (*i.e.*, p_x/p_{total}) that give the direction of particle motion. **Trak** will normalize the numbers; therefore, the sum of the squares of the components need not equal to unity. To represent a particle with zero initial kinetic energy, set p_x, p_y and p_z equal to 1.0 (not 0.0).

The *END* command signals the end of the list.

PFILE = FPrefix
PFILE = Run01

Trak can read particle starting point information from a file rather than from the control script. The parameter *FPrefix* is the prefix of a file with a name of the form `FPREFIX.PRT` in the current directory. The file contains from 1 to 2500 particle data lines in the format described above terminated by an *END* command. You can include text annotations in any format after the *END* command.

Standard particle files (`PRT`) play an important role in **Trak**:

■ Particle files are in ASCII format and can use any of the standard **TriComp** delimiters including spaces and tabs. Therefore, you can create standard particle files using a text editor, a spreadsheet, or your own programs.

■ The **GenDist** program (described in a separate manual) is a utility for creating PRT files with complex particle distributions using a few simple script commands.

■ **Trak** can generate output PRT files of final particle parameters that can be used as input to subsequent runs.

■ The Field Precision 3D gun design code **OmniTrak** uses the same file format. Therefore, you can transfer particle distributions between the two programs.

■ The **VEStat** post-processor can use input or output PRT files to perform beam analyses and to create a variety of distribution plots. You can also use **VEStat** to filter particle files (*i.e.*, remove unwanted particles).

6.3. Starting particles from a surface

Emission surfaces are critical to the operation of **Trak** in the *SCHARGE*, *RELBEAM*, *FEMIT* and *PLASMA* modes. Emission surfaces can be used in the *TRACK* mode for a specific application: generation of a distribution of particles with zero kinetic energy from electrodes or surfaces of arbitrary shape. The surface emission procedure is also used for field line tracing in the *FLINE* mode (Chapter 7). Clearly, emission surfaces are useful in the *TRACK* mode only for simulations that include an electric field. In a pure magnetic field, particles with zero kinetic energy would simply remain at the same position.

This section addresses the following questions:

■ What is an emission surface?

■ How do you create an emission surface in **Mesh**?

■ How does **Trak** identify an emission surface and generate particles?

■ How do you control emission surfaces through **Trak** commands?

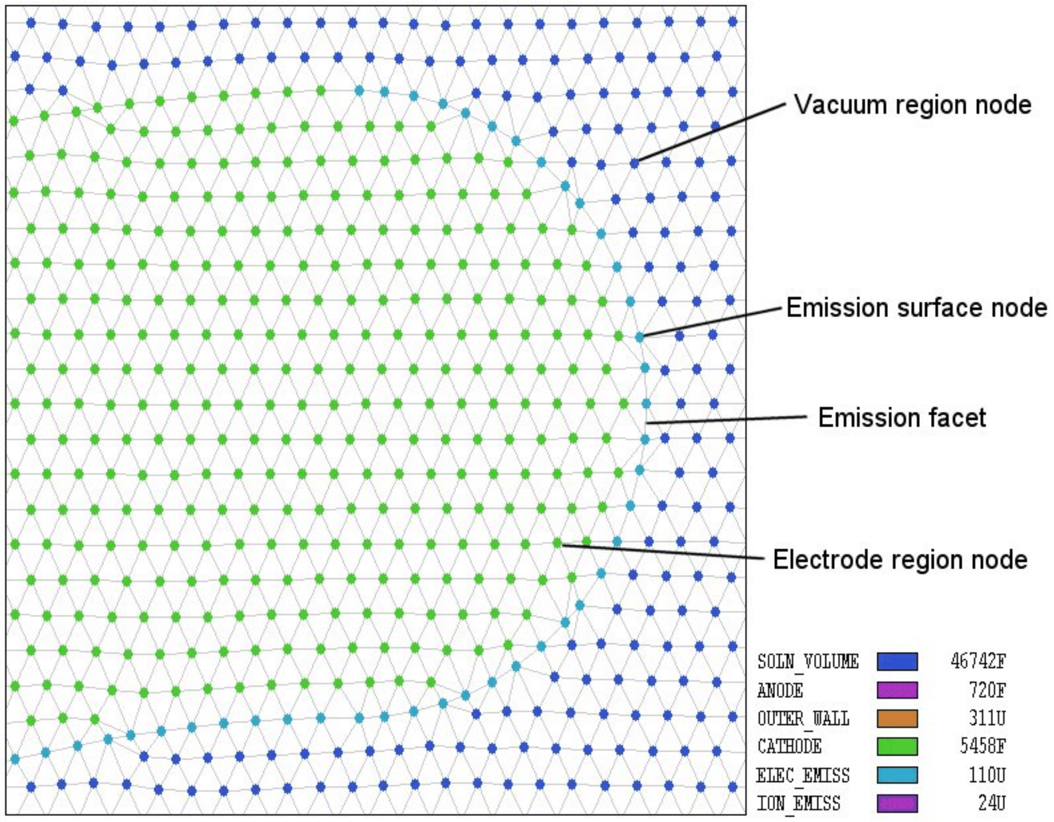


Figure 6.1. Definition of an emission surface on a conformal triangular mesh

An emission surface is simply a contiguous line region in the electric field mesh. In the **Mesh** program a line region is defined as one that does not have the *FILL* keyword in the *REGION* command. In this case **Mesh** assigns the region number to nodes along the lines and arcs that constitute the boundary, but the program does not change the identity of bordering elements (Figure 6.1). A line region need not outline a closed volume. In the *SCHARGE*, *RELBEAM* and *FEMIT* modes the line region must be on the surface of a fixed-potential filled region so that **Trak** can apply the correct emission physics. We can envision that the line region paints a portion of the electrode surface to define the area of emission. Emission surfaces are more flexible in the *TRACK* mode – the line region may be on the surface of an electrode or within the volumes of dielectric and vacuum regions. If the line region is on the surface of an electrode, set it to the fixed-potential condition in **EStat** and assign the same potential as the electrode. With these settings the presence of the emission surface will not perturb the electric field solution. Similarly, a line region in a dielectric volume should be assigned the same value of relative dielectric constant ϵ_r.

If region number *NReg* is defined as an emission region, **Trak** searches the mesh and collects all nodes with that number. The program then creates a list of emission *facets*: element sides that connect two emission nodes (Fig. 6.1). **Trak** attempts to place the facets in a logical connected order starting from the node nearest the axis and then checks that the facets form a contiguous set. If we specify that one particle should be created per segment, the code finds the midpoint of each facet. If the facet has dielectric elements on both sides (*i.e.*, the electric field at the midpoint is non-zero), the location is used as the start point for the particle orbit. If one element has the fixed-potential condition, the code moves the emission point slightly into the dielectric element for a valid field interpolation. **Trak** issues an error message if both elements adjacent to an emission facet are part of a fixed-potential region. To create multiple particles per facet, the code first divides each facet into a number of segments and then starts particles from the midpoints of the segments. Emission surfaces can have arbitrary shape; therefore, emission points may not be uniformly spaced.

The following commands control emission surfaces in the *TRACK* mode.

EMIT NReg Mass Charge NPerSeg
EMIT(5) = (0.0, -1.0, 3)

This command states that nodes with region number *NReg* constitute an emission surface. The real number parameters *Mass* and *Charge* give the mass and charge of the particles that will be created. Enter the mass in AMU (atomic mass units, 1.65979×10^{-27} kg). If 0.0 appears in the column, the program inserts the value for an electron. The quantity *Charge* is the particle charge in units of *e* (1.60210×10^{-19} coulomb). An ion has charge +1.0 and an electron has charge -1.0. Note that only one particle species can be created on an emission surface. You can define several emission surfaces in a run, each with a different particle species. The final integer parameter, *NPerSeg*, is the segments per facet. For example, the value *NPerSeg* = 3 instructs the code to create three particle orbits per facet.

START NReg XStart YStart
START NReg ZStart RStart
START(5) = (0.0, 5.0)

In processing an emission surface **Trak** must start with a node at the end of the line region to arrange the facets properly. Sometimes the end point may not be the point closest to the axis, or there may be an ambiguity (such as a horizontal line). The code issues an error message if facet ordering fails. You can correct the problem by actively setting the endpoint of the line region. The integer parameter *NReg* is the region number of surface. Enter the coordinates of a point close to the end in units set by the current value of *DUnit*. For advanced debugging note that **Trak** records information on the selection of facets and the ordering procedure in the listing file `FPrefix.TLS`.

6.4. Controlling orbit integrations

The following commands control the numerical solution of particle orbits.

DT Dt
DT = 1.0E-12

This command sets the integration time step (in seconds). The quantity *Dt* should be less than the minimum time it takes for a particle to cross an element. In solutions with a magnetic field, the time step must also be smaller than $1/\omega_g$ (where $\omega_g = eB/\gamma m_o$) for accurate and stable representations of gyro-orbits. Lower values of *Dt* give higher accuracy but extend the run time. If the command does not appear, **Trak** makes a guess based on estimates of the minimum element size and maximum particle velocity. The velocity is determined by checking the initial kinetic energies and/or the maximum change in potential energy in the solution space. The default value may be in error if there is a strong applied magnetic field.

DT DTRef MASS
DT = 2.3E-8 (MASS)

This form of the *DT* command with the keyword *MASS* must be used whenever a simulation contains particles with different masses. The quantity *DtRef* is a reference time step for particles with a mass of 1.0 AMU (atomic mass unit). The actual time step used for a particle orbit is given by $Dt = DtRef \times m^{1/2}$, where m is the particle mass in AMU. Therefore in a mixed simulation with protons and electrons the time step for electron orbits will be about 1/43 of the time step used for the protons. As an example, consider the calculation of electron extraction and ion back-flow in a gun with 250 kV applied voltage. Suppose the minimum element size is 1 mm. The maximum velocity of a proton would be 6.92×10^6 m/s. Therefore, we set $DtRef = 10^{-3}/6.92 \times 10^6 = 1.44 \times 10^{-10}$ seconds.

TMAX TMax
TMAX = 3.0E-9

This command sets the maximum duration (elapsed time) of orbits. Enter *TMax* in seconds. A common use of the command is to prevent infinite calculations in a system with closed orbits. In the event an orbit exceeds the maximum duration, **Trak** determines its final position and momentum by an accurate interpolation to *TMax*. This feature is useful for analyzing isochronous systems. The default value is *TMax* = +∞

DMAX DMax
DMAX = 15.0

This command sets a maximum total length for orbits. Enter the distance in units set by *DUnit*. When the distance exceeds *DMax*, the code determines the stopping point by interpolation so that all orbits have almost exactly the same length. This command gives an alternate way to prevent infinite orbits. The default value is *DMax* = +∞.

NTRACKMAX = NTrackMax
NTRACKMAX = 400

This command sets the maximum number of integration steps. Its main use is to prevent infinite orbits. The default value is *NTrackMax* = 10,000.

NSEARCH E NSearchE
NSEARCH B NSearchB
NSEARCH(E) = 3

Trak must identify elements in the electric and/or magnetic field meshes occupied by particles during orbit integrations. The procedure is challenging on the conformal meshes used in **TriComp** because there is no unique relationship between a node's index and its position. Therefore it is necessary to check individual elements. Furthermore, separate searches must be performed for the calculations of electric and magnetic fields because they are defined on independent meshes. To speed the process **Trak** uses a fast search procedure. The elements occupied by the starting point are located by full searches on each mesh. In subsequent steps, local searches are made in the vicinity of the last occupied element. The parameter *NSearchE* governs the width of the local search region in the electric field mesh. For a reasonable choice of *Dt* , the code defaults of *NSearchE* = *NSearchB* = 6 are sufficient. You can speed up integrations by choosing a smaller value. Larger values may be necessary if a mesh contains regions with very small elements.

INTERP E [LIN,LSQ]
INTERP B [LIN,LSQ]
INTERP BB [LIN, LSQ]
INTERP(B) = LIN

By default **Trak** uses a second-order least-squares fit procedure to calculate electric and applied magnetic field during an integration (*LSQ* option). The *LIN* option activates a simple linear routine that returns a uniform value of electric field, applied magnetic field or beam-generated magnetic field in each element. Although it yields reduced accuracy, there are two reasons to use the *LIN* option: 1) it is faster, and 2) it is less prone to errors when the solution volume includes small enclosed regions. In a *cul de sac* **Trak** may have difficulty locating enough points to make the *LSQ* fit.

6.5. Stopping conditions

The implementation of precise stopping conditions is a critical component of a charged-particle simulation. For example, you may want to determine whether a particle strikes a detector and then to find the exact particle parameters at the impact point. Similarly, in characterizing a lens we want a precise prediction of beam properties in a plane normal to the axis. This section describes the wide variety of available methods to stop particles in **Trak**.

We shall first address stopping at region surfaces in the finite-element meshes. Regions in the electric and/or magnetic field meshes can be assigned one of three properties that affect particle transport.

Vacuum

Particles move unimpeded through *Vacuum* elements.

Material

Particles stop when they enter a *Material* element.

Secondary

This feature applies only to electron tracking. Electrons are re-emitted when they enter a *Secondary* element. The process of secondary emission is described in detail in Chap. 12.

When a particle enters a *Material* or *Secondary* element, **Trak** employs a sophisticated procedure to project the orbit back to the entry surface to find the final or re-emission position.

Trak makes default assignments of region material properties. In the electric mesh, all elements with $\epsilon_r = 1.0$ are set as *Vacuum*, and all elements with fixed potential or $\epsilon \neq 1.0$ are set as *Material*. In the magnetic mesh elements with $\mu_r = 1.0$ are initialized as *Vacuum*, and elements with fixed vector potential or $\mu_r \neq 1.0$ are *Material* . You can use the following commands to change the assignments.

VACUUM E RegNo
VACUUM(E) = 2
VACUUM B RegNo
VACUUM(B) = 5

These commands set all elements with region number *RegNo* in the electric or magnetic mesh to the *Vacuum* condition.

MATERIAL E RegNo
MATERIAL(E) = 4
MATERIAL B RegNo
MATERIAL(B) = 1

This command sets all elements with region number *RegNo* in the electric or magnetic mesh to the *Material* condition.

SECONDARY E RegNo DeltaMax0 EngMax0
SECONDARY(E) = 8 2.35 300.0
SECONDARY B RegNo DeltaMax0 EngMax0
SECONDARY(B) = 3 1.2 250.0

This command sets all elements with region number *RegNo* in the electric or magnetic mesh to the *Secondary* condition. Chapter 12 discusses the properties of secondary emission materials.

The next set of commands controls the definition of special planes in the solution space to stop, to diagnose or to reflect particles. Normally the location of the plane should be inside the common volume defined by the electric and/or magnetic field meshes. If the script contains the *BOUNDARY* command in the *FIELDS* section, the planes can be located anywhere inside the specified boundary area.

STOP [UP,DOWN] [X,Y,Z,R] StopPosition
STOP(UP, X) = 5.50

Particle orbits terminate when they cross a stopping plane. **Trak** projects the orbit back to the plane to find precise values of final position, momentum, total distance and elapsed time. The *STOP* command has three parameters with the following options:

UP, *DOWN***.** The string parameter sets the direction of particle motion along the axis normal to the stopping plane.

X, *Y*, *Z*, *R***.** The string parameter sets the axis normal to the stopping plane. Note that the *R* option can function in simulations with both cylindrical and planar symmetry. In the latter case, the program records parameters with the particle crosses a circular boundary in the x-y plane.

*StopPosition***.** The real number parameter is the position of the stopping plane along the normal axis. Enter the value in units set by the current value of *DUnit*.

As an example, suppose a source at z = 0.0 creates electrons trapped in an axial magnetic field. We want to stop electrons if they reach an axial displacement of ±5.0 cm from the source. In this case, we include the commands

```
STOP (UP, Z) = 5.0
STOP (DOWN, Z) = -5.0
```

DIAG [UP,DOWN] [X,Y,Z,R] DiagPosition
DIAG (Up, X) = 4.54

Trak records a set of final particle parameters that are used in the *DIAGNOSTIC* commands discussed in Chapter 13. The values can also be recorded in a standard particle file using the *PARTFILE* command. Sometimes it is useful to record these parameters and then to continue the particle orbits. For example, suppose you want to find the distribution at the waist point of a high-current beam. If the particles stopped at the waist, the electrostatic fields would be incorrect because there is no downstream space-charge assignment. In this case you can define a *DIAG* plane. When a particle crosses such a plane, **Trak** records interpolated particle parameters in the plane and continues the orbit until a stop condition occurs. Allowed parameters are the same as those for the *STOP* command. You can define multiple *DIAG* planes – note that parameters are recorded at the last plane encountered.

RECORD [UP,DOWN] [X,Y,Z,R] RP1 RP2 ... RPN
DIAG (Up, Z) = 19.0 20.0 21.0

With this command you can record precise particle parameters at a number of positions normal to an axis. There are three differences from the *DIAG* command: 1) you can define up to 10 record planes, 2) the information transferred to `PRT` files during the orbit integrations and is not available for operations with commands of the *DIAGNOSTICS* section and 3) only a single *RECORD* command may appear in the script. The second and third string parameters specify the direction of particle motion and the axis for normal planes. In cylindrical simulations the most common choice would be `UP Z`. Up to 10 real-number parameters may follow giving the positions along the axis for diagnostics. If there are three entries, **Trak** opens three files with names of the form `RunName01.PRT, RunName02.PRT and RunName03.PRT` and records values of energy, position and momentum as each particle crosses the corresponding planes. In simulations of the type *SCHARGE*, *RELBEAM* and *PLASMA*, data are recorded only on the final cycle. The following rules apply to the *RECORD* command:

■ The file may contain only one *RECORD* command. Therefore, all planes are normal to a single axis.

■ The positions *`RP1, RP2, ... RPN`* must appear in order of increasing value for the *UP* option or decreasing value for the *DOWN* option.

■ The command must contain between 1 and 10 position values.

■ For the *UP* option, particles with starting positions less than *RP1* will not be recorded in any of the files. For the *DOWN* option, a particle must start at a position > *RP1* to be recorded.

■ The recording process may not function correctly if the spacing between planes is less than the particle integration step size.

REFLECT [UP,DOWN] [X,Y,Z,R] ReflectPosition
REFLECT (Down, Y) = 0.0

Reflection planes are useful to simulate periodic systems with symmetry in both the fields and particle motions. When a particle crosses a reflection plane, Trak projects its position back to the plane and reverses its momentum normal to the plane. Allowed parameters are the same as those for the *STOP* and *DIAG* commands. Multiple *REFLECT*, *DIAG* and *STOP* commands may appear in a control script.

6.6. Orbit listings

The final three commands control information that can be recorded during orbit tracking.

LISTON NStep [P,E,B]
LISTON (2, B)

This command controls records of orbit quantities in the listing file (`RUNNAME.TLS`). This information is more detailed than the simple coordinate data recorded in the plot file (`RUNNAME.TOU`). The list option is normally deactivated because complex runs could generate huge listings. The optional parameter *NStep* is the number of integration steps between records. The optional key symbol *P*, *E* or *B* determines the type of information recorded. Under the *P* option (the default) the data lines contain the following quantities: step number, elapsed time, coordinates (in units set by the current value of *DUnit*), total distance, and normalized momentum components ($\mathbf{p}_{\text{norm}} = \mathbf{p}/m_o c$). In the field modes (*E* and *B*) **Trak** records coordinates of the integration points and calculated field components. An extract from a file is shown in Table 6.1. The *LISTON* command is valuable for debugging runs and checking field interpolations along particle trajectories.

Table 6.1. Extract of orbit list under the E option

```
Tracking particle number   2
 NTrack      t           x           y           z
 ======================================================
      0  0.0000E+00  5.0000E-02  0.0000E+00 -2.4990E+00
      5  4.0997E-11  5.0012E-02  0.0000E+00 -2.2599E+00
     10  8.1993E-11  5.0105E-02  0.0000E+00 -2.0223E+00
     15  1.2299E-10  5.0400E-02  0.0000E+00 -1.7873E+00
     20  1.6399E-10  5.1073E-02  0.0000E+00 -1.5574E+00
     25  2.0498E-10  5.2292E-02  0.0000E+00 -1.3364E+00
     30  2.4598E-10  5.3994E-02  0.0000E+00 -1.1286E+00
     ...

          Dist         Ex          Ey          Ez
     ================================================
       0.0000E+00 -9.0412E+00  0.0000E+00  3.8246E+04
       2.3908E-01 -2.7417E+03  0.0000E+00  5.0646E+04
       4.7669E-01 -6.7280E+03  0.0000E+00  9.1283E+04
       7.1166E-01 -1.2867E+04  0.0000E+00  1.7572E+05
       9.4158E-01 -1.9363E+04  0.0000E+00  3.1533E+05
       1.1626E+00 -1.7718E+04  0.0000E+00  4.7408E+05
       1.3704E+00 -2.4400E+03  0.0000E+00  5.5682E+05
       ...
```

ORBINFO

Use the *ORBINFO* command on those occasions when particles mysteriously stop or refuse to move. When the command is issued Trak writes a record in the listing file of the orbit termination status. Table 6.2. shows an example.

Table 6.2. Example of a termination statement

```
Termination status of particle number   2
  Orbit outside the boundary of the electric field solution
Final position: [  5.4806E-03,  0.0000E+00,  3.9995E+00]
Final momentum: [ -2.0874E-03,  0.0000E+00,  1.9884E-01]
Final kinetic energy:   1.0006E+04
```

PLOTOFF

When this command is issued, the program does not make a plot file, `RUNNAME.TOU`. This feature saves disk space if you are making a run with a large number of particles and you are interested only in the starting and ending parameters.

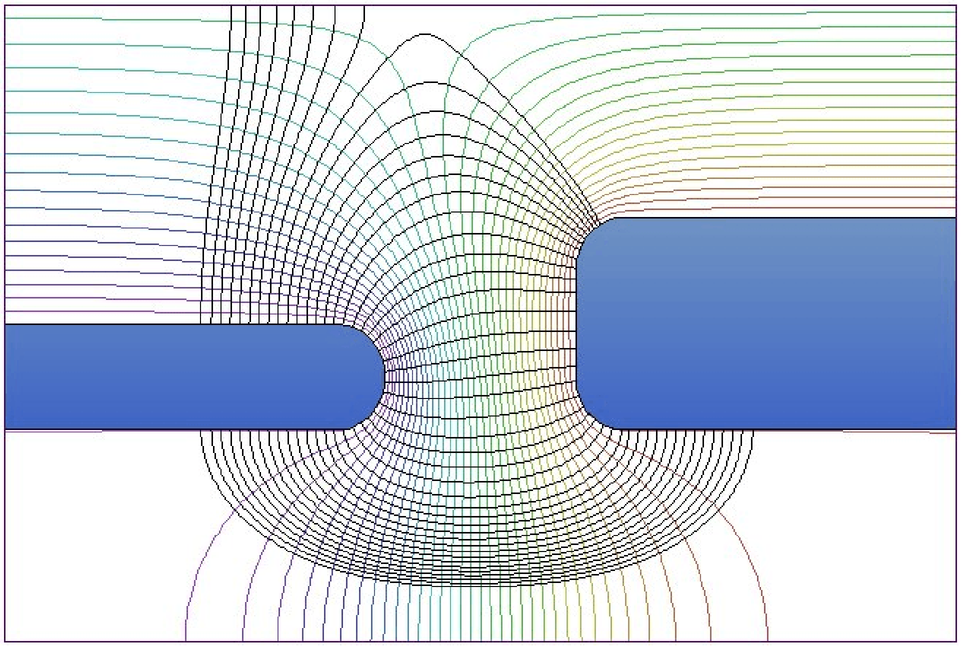


Figure 7.1. Field lines generated from an emission surface on the left electrode.

7. Tracing electric field lines

Commands introduced in this chapter

FList

FFile

Ds

The *PARTICLES FLINE* tracking mode is used to find precision traces of electric field lines in **EStat** solutions. The path integral continues until a field line leaves the volume of the electric field solution or enters an element included in a fixed-potential region (*i.e.*, an electrode). In the first case, **Trak** determines the stopping point by projecting the orbit back to the solution boundary. In the second case, the code performs a search for the intersected segment on the surface between the fixed and non-fixed regions and performs an interpolation. In this way, **Trak** gives accurate predictions of the termination points of field lines on electrodes. Allowed commands in the *PARTICLES FLINE* section have three functions:

- Set starting points for traces
- Control the trace integration
- Control diagnostics in the listing file during integrations.

We shall first discuss commands to start field line traces. You can start traces from a list or an emission surface (Section 6.3).

FLIST

This command signals that a list of starting positions follows in the control script. The following example illustrates a field list:

```
  FLIST
*   XStart  YStart  ZStart Polarity
*   ===============================
    0.9999  0.0000  -0.50   -1.00
    0.9999  0.0000  -0.75   -1.00
    0.9999  0.0000  -1.00   -1.00
    0.9999  0.0000  -1.25   -1.00
    0.9999  0.0000  -1.50   -1.00
    0.9999  0.0000  -1.75   -1.00
    0.9999  0.0000  -2.00   -1.00
    0.9999  0.0000  -2.25   -1.00
  END
```

The maximum number of lines is 2500. Each data line contains four real numbers. The first three, *XStart, YStart* and *ZStart* give the initial position in units set by the current value of *DUnit* . To ensure a valid initial field interpolation, the point must not be inside a fixed-potential element. To avoid errors, make sure that the starting positions are displaced a small distance into the field region from the electrode surface. The quantity *Polarity*, which determines the direction of integration, may assume the values ±1.0. For *Polarity* = +1.0, the integration procedures along the direction of positive electric field (*i.e.*, from positive to negative potential). The *END* command signals the end of the list.

FFILE FPrefix
FFILE = PReactor

Trak can read field line starting points from a file rather than the control script. This feature is useful if there are a large number of starting points or if you want to use the same starting points in several different solutions. The parameter *FPrefix* is the prefix of a file with a name of the form `FPREFIX.FLD` in the current directory. The file has a format similar to the *FLIST* command.

DS Ds
DS = 0.015

This command sets the spatial step size for field line integrations. Enter the distance *Ds* in units set by the current value of *DUnit*. Smaller values of *Ds* give higher accuracy. If the *DS* command does not appear, **Trak** picks a default equal to the diagonal length of the electrostatic solution space divided by 500.0.

You can also start field lines from emission surfaces using the *EMIT* command. This feature is particularly useful for field lines because we often want to find the destinations of lines that start from different positions on an electrode. The line regions that define emission surfaces can be placed on an electrode or they may follow arbitrary paths in a dielectric region. If an emission segment is on an electrode, **Trak** picks a starting point slightly displaced into the adjoining dielectric element.

The following two commands are described in detail in Section 6.3.

EMIT NReg NPerSeg [Polarity]
EMIT(5) 3 -1.0

This command states that nodes with region number *NReg* constitute an emission surface. The integer parameter, *NPerSeg*, is the number of segments per facet. For example, the value *NPerSeg* = 4 instructs the code to start four evenly-spaced field lines from each facet. The quantity *Polarity*, which determines the direction of integration, may assume the values ±1.0. For *Polarity* = +1.0, the integration procedures along the direction of positive electric field (*i.e.*, from positive to negative potential). The default is +1.0.

START NReg XStart YStart
START(5) = (0.0, 5.0)

This command is used to prevent ambiguities in the definition of an emission surface. It is described in detail in Section 6.3.

Several of the commands introduced in Chapter 6 for integration control in the *TRACK* mode can also be applied in the *FLINE* mode:

DMAX = DMax

NTRACKMAX = NTrackMax

NSEARCH E NSearch

INTERP E [LIN,LSQ]

Field lines terminate if they leave the volume of the electric field solution or enter a fixed-potential region. You can define additional stopping conditions with the following commands:

STOP [UP,DOWN] [X,Y,Z,R] StopPosition

DIAG [UP,DOWN] [X,Y,Z,R] DiagPosition

REFLECT [UP,DOWN] [X,Y,Z,R] ReflectPosition

Finally, the following commands can be used to write diagnostic information to the listing file.

LISTON NStep
LISTON (2)

In contrast to the *TRACK* mode, the listing file data lines in the *FLINE* mode always contain the point coordinates and components of the calculated electric field. The key symbols *P*, *E* and *B* are ignored.

ORBINFO

This command gives a record if the termination status of the field line. To gauge the accuracy of the integration, Trak also records the difference in electrostatic potential between the starting and stopping points compared to the line integral ∫ - **E·dl**.

Table 7.1. Termination listing for an electric field line

```
Termination status of field line number   3
  Point inside fixed potential region
Final position: [  1.0000E+00,  0.0000E+00,  1.0001E+00]
Accuracy check
   Phi(Initial):  -5.0000E+03
   Phi(Final):   5.0000E+03
   Phi(Final)-Phi(Initial):   1.0000E+04
   Integral -Edl:   9.9991E+03
```

8. Tracking particles with self-consistent space-charge effects and Child-law emission

Commands introduced in this chapter

PList
PFile
Emit
NCycle
Avg
Suppress
RelMode
Restart

8.1. List input for high-current beams

The *SCHARGE* mode is used to model high-current non-relativistic electron and ion beams. In this mode **Trak** calculates beam space-charge density while tracing particle orbits. Electric fields are then recalculated with the charge contributions of the beam. Several iteration cycles are necessary to find a self-consistent combination of orbits and fields. You can start particles from a list or request that Trak create particles on emission surfaces and assign current to satisfy the Child law (space-charge-limited emission) The data lines of the *PLIST* and *PFILE* commands (discussed in Chapter 6) have an extra entry to define the current (or current per length in planar simulations) of the model particles.

PLIST

The *PLIST* command signals that a list of starting points follows in the control script. Each data line contains ten real numbers representing the following quantities:

Mass
Charge
Energy
x, y, z
px, py, pz
Current

Except for the final parameter, the quantities have the same meanings as those discussed for the *TRACK* option. In cylindrical simulations the *Current* of a model particle represents an annular region (extending from 0 to 2π in θ) centered at the particle position *r*. Enter the particle current in amperes. For an injected beam of particles uniformly-spaced in radius with uniform current-density, the particle current should be proportional to r_i (the initial particle radius). In planar simulations, enter the current per length (along *z*) in units of A/m. Note that the current is always a positive number. The direction of the current is given by the sign of the quantity *Charge* and the axial velocity of the particle (v_z or v_x). Again, the *END* command marks the end of the list.

PFILE FPrefix
PFILE = RelInject

This command has the same form as the *PFILE* command in the *TRACK* mode. The only difference is the addition of the quantity *Current* in each data line of the file.

Note: In planar simulations in the *SCHARGE* mode, the *x*-axis must be the direction of average beam motion. Furthermore, the solution must be symmetric about $y = 0.0$. In a typical simulation, you would model only the space $y \geq 0.0$ and apply symmetry conditions at $y = 0.0$ (*i.e.*, Neumann condition for the electrostatic potential, reflection condition for particles).

8.2. Self-consistent electric field calculations

Self-consistent gun and transport calculations with beam-generated electric fields present a challenge: particle orbits depend on the electric fields while the fields depend upon the orbits through the space charge. Trak employs an iterative approach, the *ray tracing technique*, to solve the boundary-value problem. The program computes particle orbits in the applied electric fields. The space charge associated with the orbits is then used to find a modified solution of the Poisson equation. Next, the orbits are recalculated in the new fields. The process continues over several cycles until the solution converges. The following commands control the cyclic process.

NCYCLE NCycle
NCYCLE = 15

This command sets the number of orbit-tracking/field-recalculation cycles. The number required for convergence depends on the nature of the problem. A simulation where the beam-generated electric field makes a small contribution to the total field may converge in a few cycles. On the other hand, a simulation where beam fields predominate (*i.e.*, a beam expanding in a field-free transport region) may require 10-30 cycles. In a multi-cycle run, **Trak** writes listing information in response to the *LISTON* command and entries in the plot file only on the final cycle. One sign that a solution has converged is that the electric field recalculation requires fewer than the maximum number of cycles (*MAXCYCLE* command).

AVG Avg
AVG = 0.20

When beam-generated fields are strong it is necessary to adjust the beam space-charge gradually to achieve a convergent solution. The *Avg* parameter controls the degree of charge deposition. It has a value between 0.0 and 1.0. On any cycle, the space-charge in an element is given by

$$NewCharge = (1.0 - Avg) \times OldCharge + Avg \times BeamCharge,$$

where *OldCharge* is the previous value and *BeamCharge* is the value calculated from a sum over orbit traces on the present cycle. For a small value of *Avg* you should use larger values of *NCycle*. For example, if Avg = 0.15, then about 30 cycles would be required to ensure that the space-charge density was within 1% of its equilibrium value. The default is *Avg* = 0.50.

Note: The same averaging factor controls current deposition on the electric field mesh in the *RELBEAM* tracking mode.

For review, the following commands of the *FIELDS* section (introduced in Chapter 4) control electric field recalculations during each cycle:

MAXCYCLE = 2500
RESTARGET = 2.0E-8
OMEGA = 1.95

8.3. Child law emission

In the *SCHARGE* mode, emission surfaces represent space-charge limited sources of electrons or ions. An example is the surface of a thermionic cathode where the current density is below the source limit. **Trak** automatically creates model particles on the emission surface and assigns current to satisfy the the Child law. This condition is that the electric field on the surface approaches zero. The numerical method employed is described in S. Humphries, *Numerical modeling of space-charge-limited emission on a conformal triangular mesh*, J. Comp. Phys. **125**, 488 (1996). Here we shall give a brief description of the procedure so you can understand the functions of parameters in the related commands:

■ As in the *TRACK* and *FLINE* modes, **Trak** creates a set of emission facets and determines particle initiation points based on the value of *NPerSeg*.

■ A numerical orbit calculation would not be possible if particles were created on the emission surface. The Child condition of zero electric field implies that zero-energy particles would not move, and the calculation would stall.

■ To resolve the impasse, **Trak** creates a *generation surface* by projecting the particle initiation points a distance *DEmit* from the emission surface. Analytic formulas for space-charge limited flow in a planar gap of width *DEmit* are employed to find the appropriate current and kinetic energy to assign to model particles at the generation surface. The problem of stalled orbits does not occur because the kinetic energy and electric field are non-zero at the generation surface.

■ Trak employs a novel backtracking technique to ensure correct assignment of space charge in the volume between the emission and generation surfaces. The combination of this capability with correction factors for curved electrodes ensures high-accuracy solutions for space-charge-limited flow.

■ An iteration cycle in **Trak** consists of the following operations: 1) create model particles at the generation surface, 2) assign current and kinetic energy based on the present values of local electric field, 3) reverse-track the orbits at fixed energy until they strike the emission surface to set space charge in the gap between the generation and emission surfaces, 4) forward-track the orbits, 5) solve the Poisson equation using the space-charge associated with the orbits. The code makes extensive records in the file `FPrefix.TLS` so you can check the validity of the process.

The following commands control Child-law emission surfaces.

EMIT(NReg) Mass Charge DEmit [NPerSeg JLimit DTheta]
EMIT(4) = (0.0, -1.0, 0.05, 3)

This command identifies the line region *NReg* as an emission region. The region must be on the surface of an electrode (fixed-potential region). The parameters *Mass* and *Charge* have the same meaning as in the *EMIT* command of the *TRACK* mode. Enter the mass in AMU (atomic mass unit). **Trak** inserts the value for an electron if *Mass* = 0.0. Enter the charge of the emitted particles in units of *e* (*i.e.*, -1.0 for electrons and +1.0 for protons). The quantity *DEmit* is the distance from the emission surface to the generation surface. Enter the value in units set by the current value of *DUnit*. The integer parameter *NPerSeg* is the number of model particles created per facet of the emission surface (default, *NPerSeg* = 1). The real-number parameter *JLimit* (in A/m^2) is a source limit for current emission and *DTheta* (in degrees) equals the angular divergence of particles at the generation surface.

Notes on the EMIT command

■ The quantity *DEmit* should be small enough so that the generation surface closely follows the contours of the emission surface. On the other hand, the field interpolations necessary to calculate the Child-law current density will be inaccurate if *DEmit* is less than the local element width. As a rule, pick *DEmit* equal to about 1.5 times the element width. It may be necessary to use small elements near the emission surface to achieve a good fit.

■ If you set a value for the parameter **JLimit**, **Trak** can model mixed space-charge and source-limited emission. The program calculates the Child value at the generation surface and projects it back to the emission surface. The program then chooses the smaller of this value or *JLimit*. The default value for all emission regions is *JLimit* = ∞.

■ If you set a value for the parameter *DTheta*, **Trak** assigns an angular spread to particles leaving the emission surface. An angle in the range 0.0 to *DTheta* is assigned to particles with random- uniform weighting. The angle is the displacement (in three-dimensional space) from the unit vector pointing out of the source surface. The default value for all emission regions is *DTheta* = 0.0 degrees.

SUPPRESS SVal1 SVal2 SVal3 ...
SUPPRESS 0.20 0.30 0.40 0.60 0.80 1.00

The current density assigned at the generation surface must be suppressed below the Child-law value on the first few iteration cycles. Otherwise the initial current based on the applied field values would be much too high and the code may oscillate between high and low current solutions on subsequent cycles. To ensure convergence **Trak** applies suppression factors on the first few cycles. The code uses the following default values for space-charge-limited emission:

```
NCycle  Supression value
========================
   1            0.250
   2            0.300
   3            0.500
   4            0.750
   5            1.000
   6            1.000
   7            1.000
   8            1.000
   9            1.000
  10            1.000
```

For unusual solutions you can set you own values. Enter from 1 to 10 real numbers in the range 0.0 to 1.0. Any undefined values default to 1.0.

8.5. Relativistic mode

In the *SCHARGE* mode, **Trak** does not perform a detailed calculation of beam-generated magnetic fields. The mode is therefore not appropriate for simulations of relativistic electron guns. On the other hand, **Trak** can find accurate results for certain types of relativistic-beam transport problems using the *relativistic mode*. This approach is faster than the complete calculation under the *RELBEAM* tracking option and may give better accuracy for beams with $\gamma \gg 1$.

To begin, we shall review some relativistic beam physics. Consider a circular paraxial electron beam traveling along the *z* axis in free space. The term *paraxial* implies that 1) electrons have about the same axial velocity ($v_z \cong \beta c$) and 2) orbits make small angles with respect to the axis:

$$z' = dr/dz \ll 1. \tag{8.1}$$

Equation (8.1) implies that changes in the beam radius r_o take place over axial distances much greater than r_o. Therefore local beam-generated fields are approximately equal to those of an infinite-length beam.

Suppose the beam carries current I and that the space-charge density is a function of radius, $n(r)$. The electric and magnetic fields created by the beam can be determined from Poisson's equation and Ampere's law:

$$E_z \cong 0, \tag{8.2}$$

$$E_\perp = E_r(r) = -\frac{e}{2\pi\epsilon_o r}\int_0^r 2\pi r' dr' n(r') \ , \tag{8.3}$$

$$B_z \cong 0, \tag{8.4}$$

$$B_\perp = B_\theta(r) = -\frac{e\beta c\mu_o}{2\pi r}\int_0^r 2\pi r' dr' n(r') \ . \tag{8.5}$$

Equations (8.3) and (8.5) imply that the transverse electric and magnetic forces acting on individual electrons are related by

$$F_\perp(magnetic) = -\beta^2\ F_\perp(electric) \ . \tag{8.6}$$

The quantity β is small compared to unity for non-relativistic beams; therefore, the magnetic force can usually be neglected. In contrast, the repulsive electric force and attractive magnetic force are almost balanced for highly relativistic beams ($\beta \cong 1$).

The transverse forces of a planar beam also satisfy Eq. (8.6). In fact, the relationship holds for paraxial beams with any shape and with any nonuniform distribution of space-charge. The equation also holds if we include field contributions of perfectly-conducting boundaries of any shape whose dimensions change gradually in the axial direction. We can prove the result by 1) making a Lorentz transformation by velocity -βc to the average rest frame of the beam, 2) calculating the electrostatic fields resulting from the stationary distribution of charge, and 3) calculating transformed electric and magnetic field values in the laboratory frame.

This derivation also shows that because of Lorentz contraction the criteria underlying paraxial beam motion and the definition of gradual changes in boundary conditions is less stringent than that implied by Eq. (10.1). If D is the transverse scale size of the system and L is the axial distance for a significant change in the beam or boundary dimensions, then Eq. (8.6) holds if

$$\frac{R}{\gamma L} \ll 1, \qquad (8.7)$$

where

$$\gamma = \frac{1}{\sqrt{1 - \beta^2}} \ . \qquad (8.8)$$

The quantity γ is related to the rest energy m_o and kinetic energy T of the particle by:

$$\gamma = 1 + \frac{T}{m_o c^2} \ . \qquad (8.9)$$

Equations (8.1) through (8.9) imply a strategy (that we shall refer to as the *relativistic mode*) to avoid explicit calculations magnetic fields for relativistic beams. The total transverse force is related to the electric force by:

$$F_\perp(total) = F_\perp(electric) - F_\perp(magnetic) =$$

$$(1-\beta^2)\ F_\perp(electric) = \frac{F_\perp(electric)}{\gamma^2} \ . \qquad (8.10)$$

We simply calculate the electrostatic force and then divide by the square of the particle γ to include the effect of the magnetic field. The relativistic mode holds under the following conditions:

■ Transverse electric fields arise almost entirely from the presence of the beam space and surrounding conducting boundaries.

■ Beam particles move predominantly in one direction and have approximately the same axial velocity $v_z \cong \beta c$

■ The axial scale length for changes in the transverse dimensions of the beam and surrounding boundaries satisfies Eq. (8.7).

You can apply the relativistic approximation to **Trak** calculations in the *PARTICLES SCHARGE* mode. To invoke the relativistic mode, simply include the command *RELMODE* in the *PARTICLES* section. The section then has the general form:

```
PARTICLES Scharge
   ...
  PLIST
     ...
  END
   ...
  RELMODE
   ...
END
```

The command has the format

RELMODE [ZRelTrans]
RELMODE [XRelTrans]
RELMODE = 5.65

When this command is issued **Trak** divides the transverse electric force by γ^2 during orbit calculations. The quantity γ is the local relativistic energy factor of the particle. If you provide a value for the parameter *ZRelTrans*, the code only applies the relativistic mode in the region z > *ZRelTrans*. Enter *ZRelTrans* in units set by *DUnit*. The default value is *ZRelTrans* = -∞.

For example, in a cylindrical simulation the effective forces are:

$$F_r = \frac{qE_r}{\gamma^2}, \qquad F_z = qE_z. \qquad (8.11)$$

Equations (8.11) could apply, for example, to a beam moving in the *z* direction from a narrow to a broad transport tube. In this case application of the expressions would give the relativistically-correct transverse forces and also yield the correct reduction in beam kinetic energy associated with the increase in the magnitude of the space-charge potential. In planar simulations with motion along x the effective forces are:

$$F_y = \frac{qE_y}{\gamma^2}, \qquad F_x = qE_x. \tag{8.12}$$

8.6. Restarting a run

Trak has the capability to restore solutions with space-charge and Child-law emission and to proceed over additional iteration cycles. To restart a run, load the electric field file from an initial run. The values of electrostatic potential in this file reflect the presence of space charge. For convergence, the present run should have approximately the same initial conditions as the initial run (i.e., electrode potentials, particle lists, emission surfaces,...). In addition to reloading the field solution, you must include the *RESTART* command in the *PARTICLES* section. This command initiates two actions that affect runs with emission surfaces:

- **Trak** automatically sets all suppression factors (*SVal1*, *SVal2*, ...) equal to 1.00 because the initial field values are already approximately consistent with Child-law conditions.

- Space-charge averaging is not applied on the first cycle because the initial charge density is close to the final converged distribution.

RESTART

This command signals that an electric field file has been loaded with potential values that are approximately consistent with the presence of beam space charge.

To restart calculations in the *RELBEAM* mode, you should also load values of beam-generated magnetic field and enclosed current using the *BBFILE* command.

8.7. Other commands

The remaining commands allowed in the *PARTICLES SCHARGE* section are identical to those in the *PARTICLES TRACK* mode:

```
DT  Dt
DT DtRef MASS
TMAX  TMax
START(NReg) XStart YStart
NTRACKMAX  NTrackMax
DMAX  DMax
NSEARCH [E,B] NSearch
INTERP [LIN,LSQ]
MATERIAL [E,B] RegNo
VACUUM [E,B] RegNo
SECONDARY [E,B] RegNo DeltaMax0 EngMax0
STOP [UP,DOWN] [X,Y,Z,R] = Position
DIAG [UP,DOWN] [X,Y,Z,R] = Position
REFLECT [UP,DOWN] [X,Y,Z,R] = Position
LISTON [NStep] [P,E,B]
PLOTOFF
```

9. Tracking relativistic particles with self-consistent beam-generated electric and magnetic fields

Commands introduced in this chapter
BackTrack
BBFile
CNeut

9.1. Trak methods for beam-generated magnetic fields

The simulation of high-current relativistic electron beams is the grand challenge of gun design codes. The drawbacks of approaches to the calculation of beam-generated magnetic fields in previous versions of **Trak** and other codes are described in the reference S. Humphries and J. Petillo, Laser and Particle Beams **18**, (2000), 601 (supplied with the **Trak** package). The article gives a detailed description of numerical methods applied in the present version of **Trak**. This section summarizes critical concepts of the method to help you set up effective solutions. The following discussions are oriented to cylindrical beams – note that the method also applies to planar beams with some restrictions.

The new approach has several advantages:

■ It is largely automatic and requires little input from the user. Almost all commands related to beam-generated magnetic field have been removed from the **Trak** script syntax.

■ It gives improved accuracy for field calculations, largely eliminating the problem of filamentation in simulations of highly-relativistic beams. Numerical beam filamentation often occurred in previous codes because of systematic differences in the calculation of E_r and B_θ.

■ The method correctly accounts for boundary currents on conductors. As a result, **Trak** 6.0 can handle emission from convex or concave cathodes of arbitrary shape as well as beam collection on internal targets.

■ It is flexible enough to handle non-laminar beams, counter-flowing particle species and reflex orbits.

■ It provides useful diagnostic information on the distribution of beam current striking target surfaces

The foundation of the method is the calculation of beam-generated magnetic fields on the electric field mesh. There are two motivations for this approach:

■ Calculations of electric and magnetic fields are closely coupled, reducing systematic differences.

■ Fixed-potential regions in the electric mesh usually correspond to surfaces that carry current and exclude magnetic fields generated by pulsed beams.

The goal of the magnetic field calculation is to find $B_\theta(r,z)$ in the propagation volume consistent with the particle orbits. **Trak** determines values of B_θ at nodes in the electric field mesh and then calculates the field at arbitrary positions by interpolation. The azimuthal magnetic field is related to the enclosed current by:

$$B_\theta(r,z) = \frac{\mu_o \, I_{enc}(r,z)}{2\pi \; r} \; . \tag{9.1}$$

The quantity $I_{\text{enc}}(z,r)$ is the axial current passing through a plane of radius r at position z. The key to the method is therefore finding the total axial current of model particles that pass through circles defined by the mesh nodes.

Figure 9.1 shows a segment of the orbit of a particle carrying current I moving through the electric field mesh. The segment corresponds to a time step Δt. The space charge is a volumetric quantity that is conveniently assigned to the elements traversed by the particle. The convention in **Trak** is to augment the space charge of the element at the segment midpoint by an amount $I\Delta t$. In contrast, the current carried by the particle is a flux that is most conveniently accumulated on the surfaces between elements (*facets*). Here the convention in **Trak** is to assign the current I to all facets traversed by the particle in the $+z$ direction and a current of $-I$ if the particle moves in $-z$. The reference listed above discusses details of current assignment on conformal triangular meshes to ensure that the condition

$$\nabla \cdot \boldsymbol{j} = 0, \tag{9.2}$$

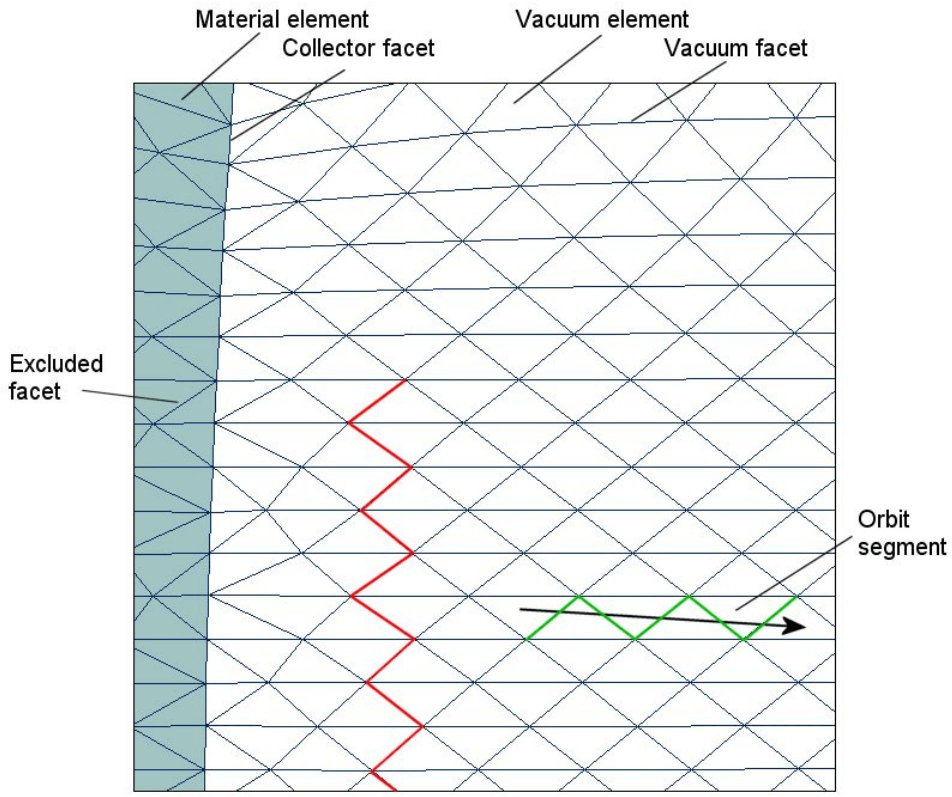


Figure 9.1. Definition of terms for the calculation of beam-generated magnetic field on a conformal mesh. Red lines show the path from a node to the axis along facets. Green lines show facets intersected by an orbit segment.

is always satisfied. The particle tracking process during an iteration cycle gives a set of facet currents that reflect the fluxes of the model particles. To find the current enclosed within a node, we simply trace a path to the axis (red line in Fig. 9.1) and add the associated facet currents. If the currents satisfy Eq. (9.2), the result is independent of the choice of path.

Implementation of the method in **Trak** is complicated by the presence of conductive materials (Fig. 9.1). There are no particle orbits through these regions; instead, they carry a surface current that depends on the distribution of emitted particles. **Trak** uses the following procedure to represent surfaces flows consistent with the condition of continuity of current:

- Facets are divided into three categories: *vacuum* facets have vacuum elements on each side, *collector* facets have one vacuum and one material element on each side, and *excluded* facets are surrounded by material elements.

■ Currents are assigned to vacuum facets by the procedure shown in Fig. 9.1. Currents are assigned to collector facets only when an orbit terminates (*i.e.*, passes from a vacuum to a material element).

■ After calculating all orbits, **Trak** proceeds to the enclosed current calculation. The first step is to set the quantity along the nodes that constitute collector surfaces. The program assumes that a collector surface is a contiguous set of collector facets that connect to the z (or x) axis. **Trak** searches along the axis to find a collector node and then assigns zero enclosed current. The program then searches for the next node connected through a collector facet and assigns the current of the previous node plus the facet current. **Trak** continues in this manner until it reaches the end of the collector surface.

■ After processing all collector surfaces connected to the axis, **Trak** proceeds to the calculation of enclosed current on the remaining nodes connected to vacuum facets. After assigning zero enclosed current to all nodes on the axis, the program works radially outward in layers. Once all the nodes in layer with radial index L are processed, the program moves to layer L+1. For each node, the program seeks the shortest connection to a node attached to a vacuum or collector facet in layer L. Enclosed current for the node in layer L+1 equals the current of the connected node in layer L plus the contribution of the facet. **Trak** stops when upon reaching the boundary of solution volume or if there are no remaining unprocessed nodes.

■ The final step is to find values of B_θ at the nodes using the enclosed current and node radius in Eq. (9.1).

Figure 9.2 illustrates the result of the calculation for the *KLYGUN* example. The example illustrates a potential problem in the method. The calculated fields are valid in the beam propagation region but exhibit dead areas with zero field at large radius. The problem occurs in the region on the right-hand side because the collector surface of the anode does not connect to the axis. Therefore, **Trak** can not assign enclosed current values along the surface. Points in the dead space that lie in the shadow of the anode follow facet paths toward the axis that terminate on anode nodes with zero enclosed current. A similar effect occurs on the cathode side because of the break between the cathode and focusing electrode.

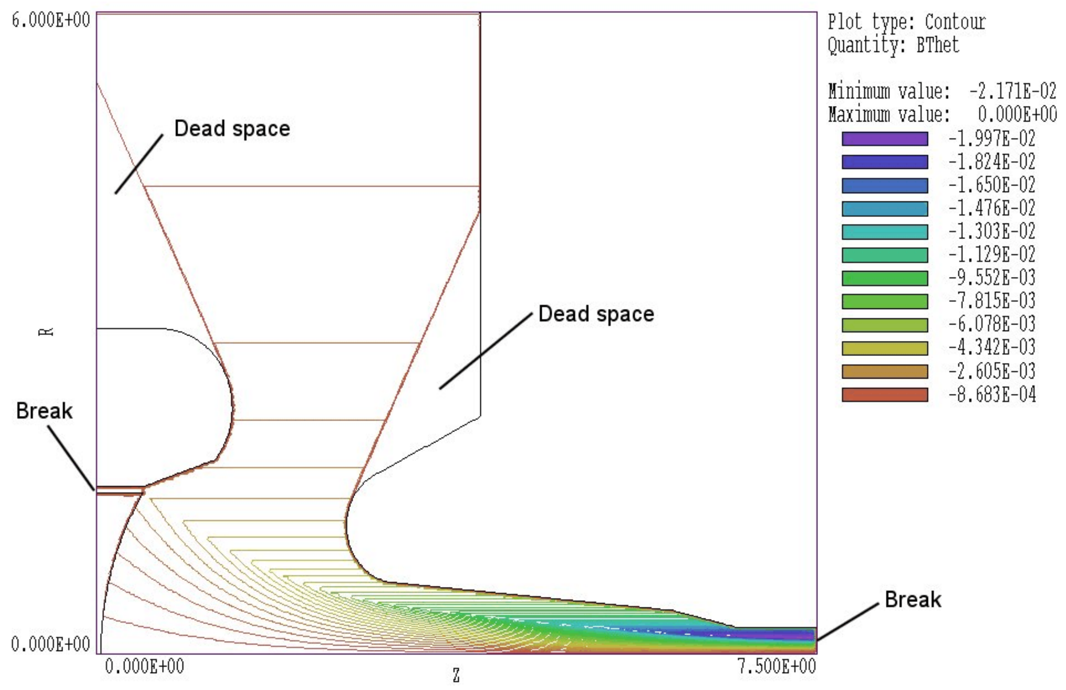
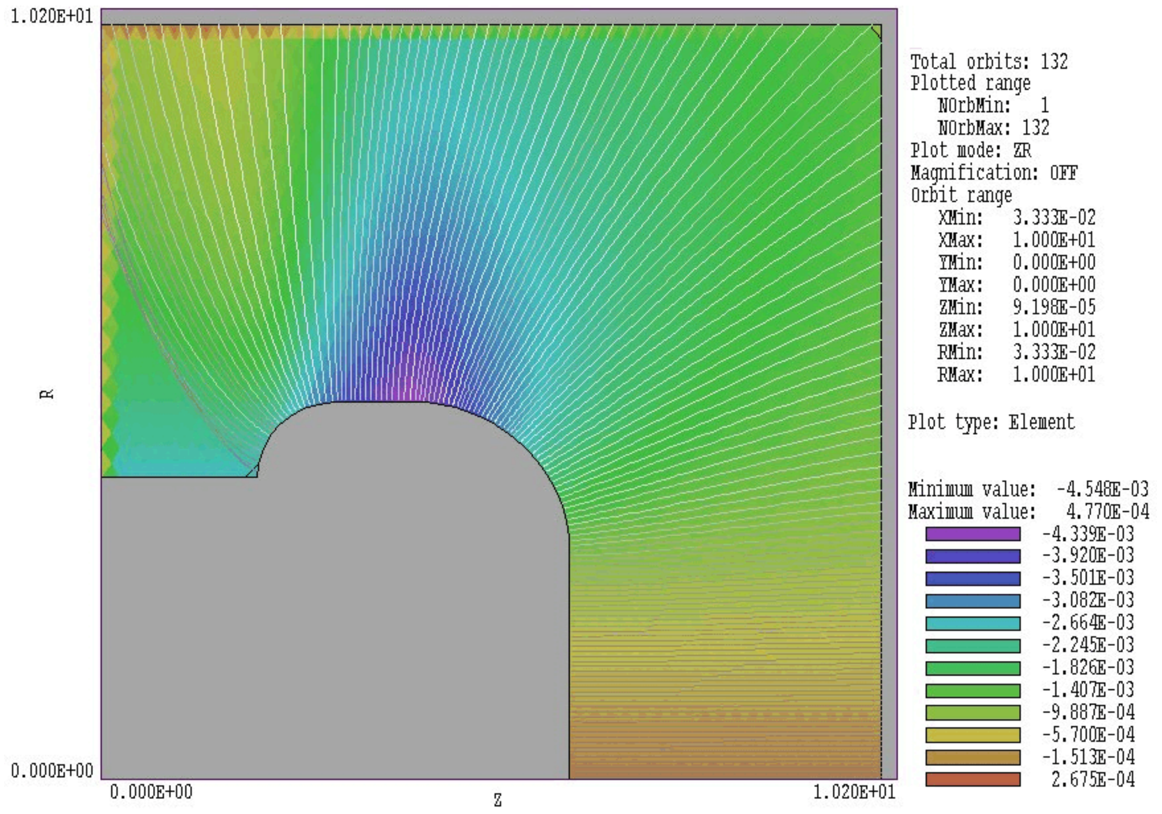


Figure 9.2. Contours lines of B_θ showing missing values in areas shaded by re-entrant collector surfaces that do not connect to the axis.

The dead spaces have no effect on the *KLYGUN* simulation because they occur outside the region of beam propagation. On the other hand, there may be cases where a representation of the beam-generated field is required over the full solution volume. For example, you may want to trace electrons emitted from the outer radius of the focusing electrode. In that case during mesh creation you ensure that all nodes of interest are connected to the axis through a chain of collector facets. The *PINCHBEAM* example in the next section illustrates such a setup.

The simulation of space-charge-limited electron emission from a convex cathode shown in Fig. 9.3 raises an interesting issue. The collector facets on the outside of the cathode are shadowed from the axis. Furthermore, electrons in the plot start from a generation surface and never pass through the cathode surface. In this case, how did **Trak** assign facet currents on the cathode to give the valid field solution shown? The answer is that facet current assignment occurred during back-tracking. Recall from Chap. 8 that in a calculation with space-charge-limited emission **Trak** projects orbits back toward the generation surface at a fixed velocity to ensure correct assignment of space charge in the gap. Facet current assignment is also performed during backtracking. When the particle



```
Total orbits: 132
Plotted range
   NOrbMin:   1
   NOrbMax: 132
Plot mode: ZR
Magnification: OFF
Orbit range
   XMin:   3.333E-02
   XMax:   1.000E+01
   YMin:   0.000E+00
   YMax:   0.000E+00
   ZMin:   9.198E-05
   ZMax:   1.000E+01
   RMin:   3.333E-02
   RMax:   1.000E+01

Plot type: Element

Minimum value:  -4.548E-03
Maximum value:   4.770E-04
                -4.339E-03
                -3.920E-03
                -3.501E-03
                -3.082E-03
                -2.664E-03
                -2.245E-03
                -1.826E-03
                -1.407E-03
                -9.887E-04
                -5.700E-04
                -1.513E-04
                 2.675E-04
```

Figure 9.3. Variations of B_θ for electron emission from a convex cathode.

strikes the cathode, the forward current *I* is assigned to the intersected facet. Note that the backtracked orbits are not recorded in the `TOU` file.

As a final point, the method described can be applied to the calculations of beam-generated fields in planar simulations. In this case **Trak** determines values of B_z at node points with beam motion in the *x* direction. The calculation is performed in the region $y \geq 0.0$ with the assumption that the enclosed current in *x* equals zero along the line $y = 0.0$. Therefore, the line $y = 0.0$ should represent a beam symmetry axis. The line must have the Neumann condition in the **EStat** calculation and must be defined as a reflection boundary in **Trak**.

9.2. Commands and controls

When **Trak** enters the *RELBEAM* mode, the program automatically sets up the mechanisms and allots memory to perform the calculation of B_θ on the electric field mesh. Magnetic field contributions are included in the particle equations of motion. With a good electric field mesh and valid choice of emission surface properties, the calculation proceeds automatically with good reliability. There is only a single command unique to the *RELBEAM* mode.:

BACKTRACK

Particles generated from emission surfaces automatically backtrack and make current contributions to collector facets on the source surface. This may not be the case when particles are created with the *PLIST* and *PFILE* commands. In this case, the particles from an electrode must be generated within a vacuum element near the fixed-potential surface for a valid electric field interpolation. As a result the nodes on the electrode surface may have zero values for enclosed current and B_θ. The invalid field values would give inaccurate field interpolations near the surface. When the *BACKTRACK* command appears, **Trak** performs an initial backtrack operation on list particles. The program reverses the momentum and sign of the current for each particle and follows the orbits until they terminate. Current is assigned to the collector facet intercepted by the particle orbit. Note that backtracking may also improve the distribution of space-charge and the calculation of electric fields near the surface.

9.3. Application example – simulation of a pinched electron beam diode

The following example illustrates a wide variety of **Trak** 6.0 techniques. The application is one that would normally be impossible for a ray-tracing code, a self-pinched high-intensity electron beam with ion flow effects. Figure 9.4 shows the geometry. The anode is an on-axis tungsten rod of diameter 1 mm. The goal is to produce focused electron spot on the anode tip with an axial extent less than 1 mm. A pulsed voltage of +1.2 MV is applied to the anode. The cathode is a thin coaxial cylinder with an inner radius of 5.0 mm. The figure shows a magnified portion of the simulation volume. The assembly is inside a grounded vacuum chamber with a radius of 35.0 mm. The simulation covers an axial region from -25.0 mm to 25.0 mm.

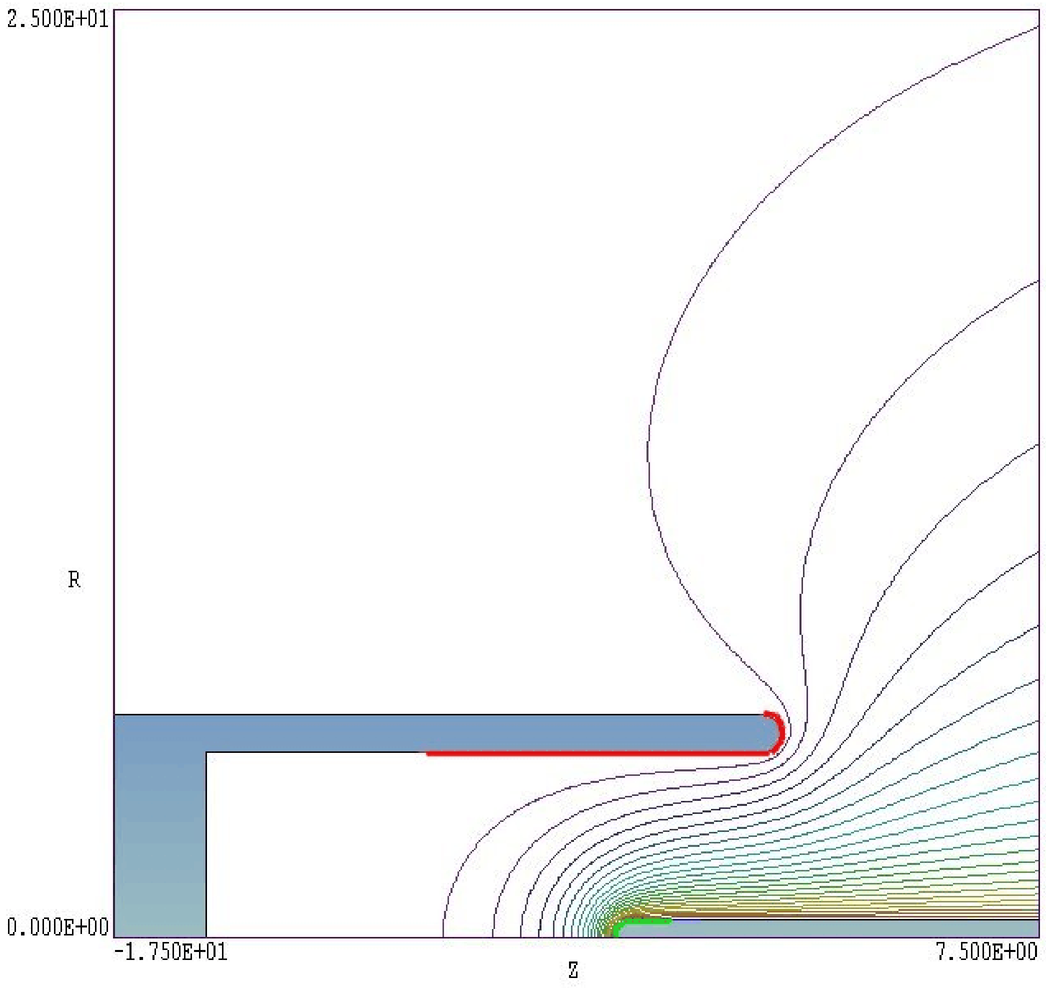


Figure 9.4. Geometry of the PINCHBEAM example – magnified view of the acceleration gap. Red line: electron emission region on the cathode. Green line: ion emission region on the anode.

The file PINCHBEAM.MIN defines the variable resolution mesh. A line region is defined for electron emission on the cathode surface (red line in Fig. 9.4). The electron flow at the anode is so intense that it immediately creates a surface plasma. Therefore, we include the possibility of space-charge-limited ion emission from the anode tip (green line in Fig. 9.4).

Table 1 shows contents of the file PINCHBEAM.TIN with added line numbers. Note the use of the multiplication factor in the **EFILE** command of line 02. The solution PINCHDIODE.EOU was generated with an applied potential of 1.0 V. We can then use the normalized solution with a multiplication factor to investigate the effects of changes in applied

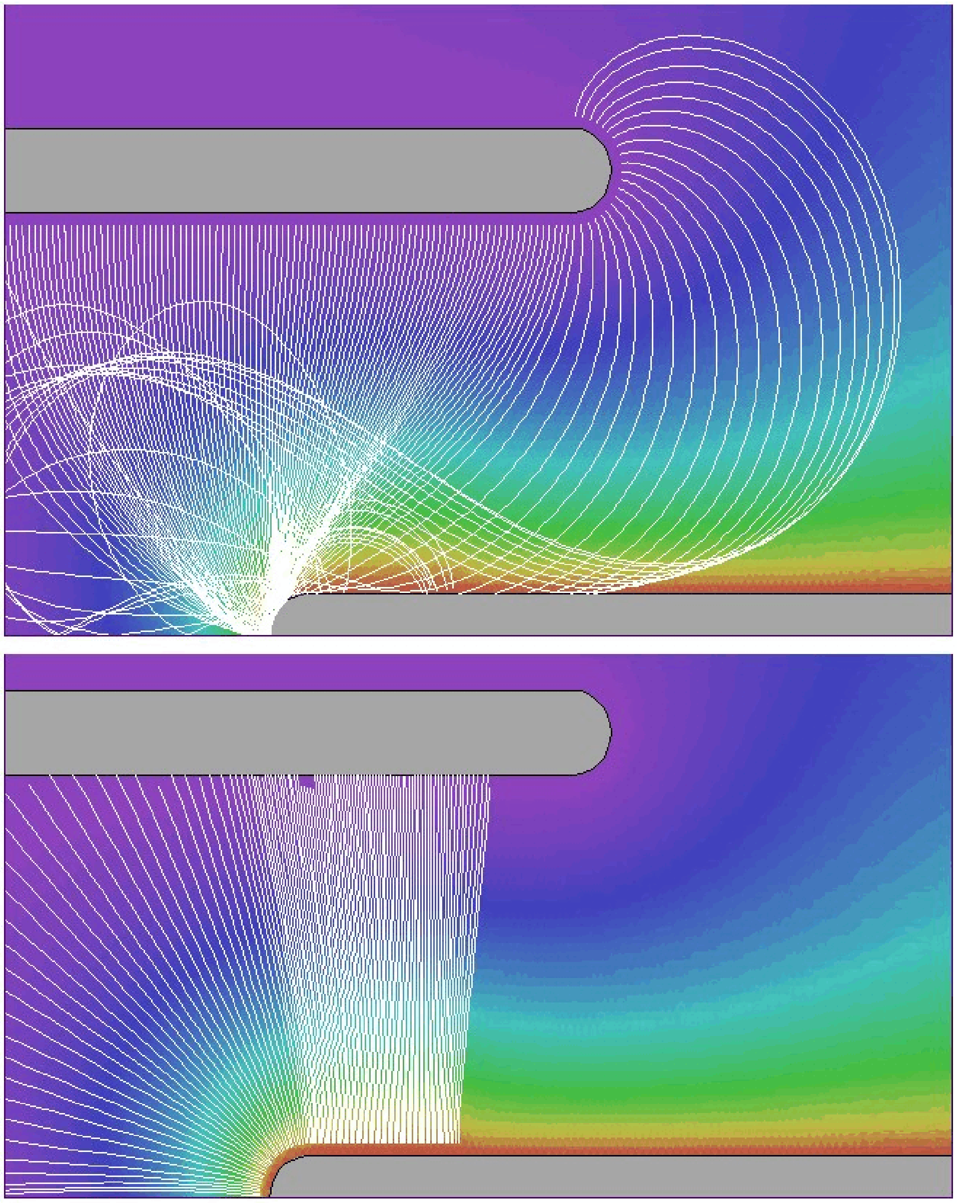


Figure 9.5. Electron (top) and ion orbits (bottom) for the `PINCHBEAM` example with *NSkip* = 5. Color-coding follows the electrostatic potential.

voltage. The solution must be performed gradually because of the complex electron orbits in the strong beam-generated magnetic fields. Note the large value of *NCycle* and small value of *Avg* in lines 08 and 09. Despite the complexity of particle motion, the solution exhibits good convergence. The emitted current of 16.65 kA in Cycle 33 differs by only about 1 percent from the current in Cycle 32. Lines 10 and 11 defined emission surfaces for the electrons and ions.

Table 9.1. Contents of the file PINCHBEAM.TIN

```
FIELDS
  EFile = PINCHDIODE.EOU  1.2E6
  MaxCycle = 200
  ResTarget = 1.0E-7
  DUnit = 1000.0
END

PARTICLES RELBEAM
   NCycle = 33
   Avg = 0.05
   Emit(5) 0.0 -1.0 0.15 5
   Emit(6) 1.0 1.0 0.15 5
   Dt: 1.2E-11 Mass
END

DIAGNOSTICS
 BBDump: PINCHDIODE
 BBBoundary (6)
 EDump: PINCHDIODEP
END

ENDFILE
```

Protons were used as the ion species in the simulation for three reasons:

- Without stringent cleaning processes, exposed surfaces in pulsed power experiments usually have layers of hydrocarbon contaminants that generate protons.

- The equilibrium electric field solution is independent of the ion mass.

- With regard to the magnetic-field solution, ions make a small contribution to current flow in the diode.

The simulation includes a large number of particles (5 per facet) for good statistics. Note that the *MASS* form of the **DT** command has been employed because there are two particle species. With regard to the reference time step, note that the minimum element width in the gap region is 0.1 mm. The velocity of a proton (1.0 AMU) is about 1.52×10^{-7} m/s, giving a minimum element transit time of 6.6×10^{-12} s. The value *DtRef* 1.8×10^{-11} s was sufficient to give a good resolution of the orbits.

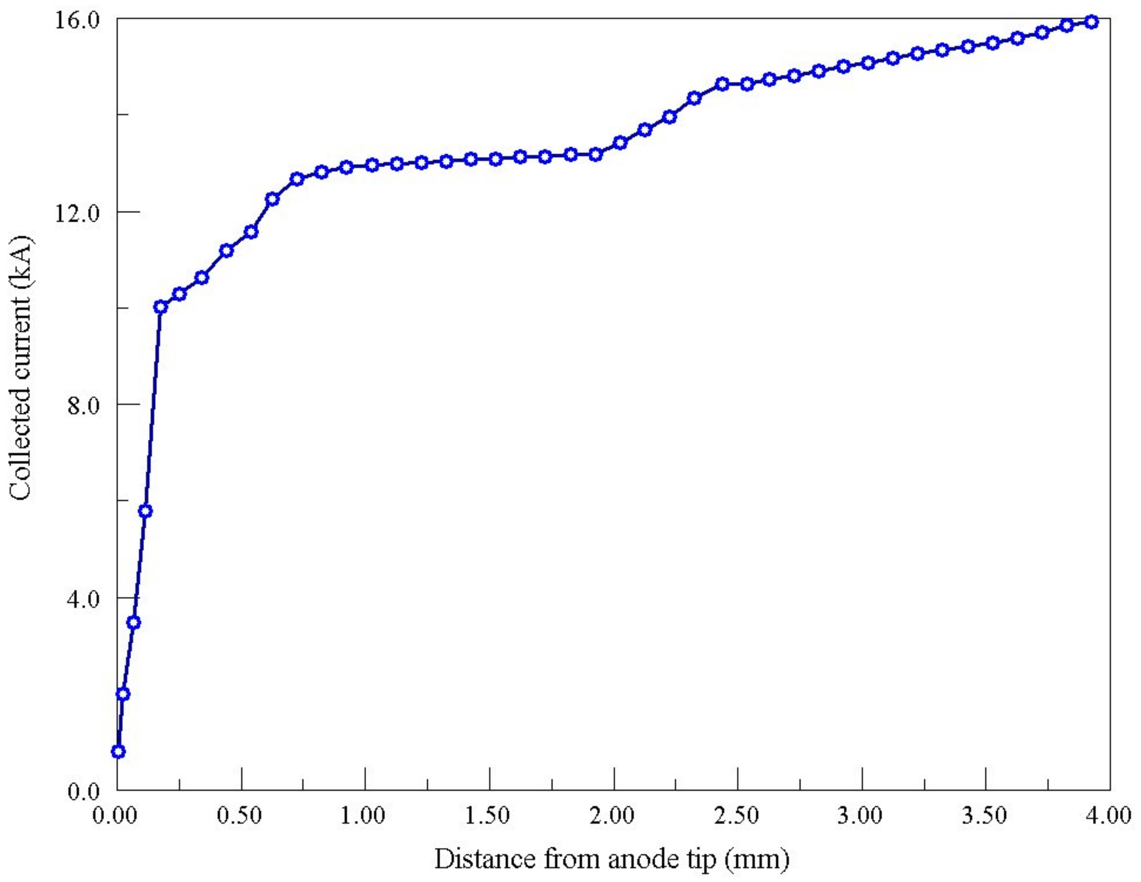


Figure 9.6. `PINCHBEAM` example – collected current on the anode tip as a function fo distance from the top.

The run time was 198 seconds on a 1.6 GHz Pentium IV computer. The equilibrium current values were 15.52 kA for electrons and 1.13 kA for protons. The effect of ions in neutralizing space-charge is critical to achieve high current operation and a well-formed pinch. The current drops by about a factor of 3 if the ions are omitted. The *BBBOUNDARY* command in line 16 of the script initiates a listing of accumulated facet currents on the anode rod. The results are plotted in Fig. 9.6. Note that about 78 per cent of the electron current impinges on the first 1 mm length of the anode. An inspection of the electron orbits in Fig. 9.5 shows that conditions are marginal for a well-contained pinch. Electrons emitted from the inner edge of the cathode tip strike the anode about 2.5 mm from the tip. A small reduction in the cathode inner radius to 4.5 mm gives enhanced current that significantly improves the quality of the focus.

9.4. Restarting a RELBEAM calculation

The following activities are necessary to restart a calculation in the *RELBEAM* mode: 1) load the electric field file created by the *EDUMP* command in a previous run using the *EFILE* command, 2) load values to calculate the beam-generated magnetic field from the same run using the *BBFILE* command, and 3) include the *RESTART* command in the particles section.

BBFILE FileName
BBFILE PINCHBEAM.BBD

Load quantities to calculate beam-generated magnetic fields generated in a previous solution with the *BBDUMP* command. **Trak** issues an error message if an electric field has not been loaded or if the mesh of the `BBD` file does not match the electric field mesh.

You can also use the *BBFILE* command in a run to track single particle orbits in the self-consistent fields of a relativistic beam. In a *TRACK* mode calculation load field information using the *EFILE*, *BFILE* and *BBFILE* commands and start particles of any species using lists or emission surfaces. **Trak** includes contributions to the magnetic field from all sources. Note that field values will not be changed by the presence of particles in the *TRACK* mode.

9.5. Space-charge and current neutralization of intense relativistic electron beams

Although **Trak** does not include models of plasmas, you can investigate effects of global space-charge and current neutralization. Partial space-charge neutralization of a relativistic electron beam may occur if low-energy plasma electrons are expelled, leaving behind an excess ion density. The space-charge neutralization factor f_e is often defined as the ratio of the excess ion density to the beam density. The presence of the ion density reduces the electric field by a factor $1 - f_e$. For example, if $f_e = 0.9$, then the loss of plasma electrons reduces the average fields in the beam volume by a factor of 10.0.

To apply local space-charge neutralization, define a dielectric region in the electric field mesh and assign a relative dielectric constant $\epsilon_r = (1-f_e)^{-1}$ in the **EStat** solution. The dielectric region must be well separated from acceleration gaps so that it does not affect the applied fields. In the subsequent **Trak** solution, contributions to electric fields from the beam space charge will be reduced by a factor $1-f_e$. Be sure to include a command of the form

VACUUM(E) = NReg

in the *PARTICLES* section of the **Trak** file. Here *NReg* is the region number of the dielectric. This command over-rides the default assignment of the *MATERIAL* attribute to all regions in the electric field mesh that have $\epsilon_r \neq 1.0$.

Partial current neutralization may occur if a current of plasma electrons is driven by the inductive fields of a pulsed beam. The current neutralization factor f_m is the ratio of the average plasma current in the propagation volume to the beam current. The effect is to reduce the beam-generated magnetic fields by a factor $(1 - f_m)$.

CNEUT Fm
CNEUT = 0.90

In response to this command, **Trak** multiplies all values of B_θ or B_z calculated from the beam current by the factor$(1-f_m)$. For relativistic beam propagation in a plasma, the factor lies in the range $0 \leq f_m \leq 1.0$.

Although space-charge neutralization may be applied to selected regions through the choice of region dielectric constant, the current neutralization factor must be applied globally to all active nodes in the electric field mesh. Suppose your goal is to model effects of neutralization on a relativistic electron beam injected through a conducting foil into a gas cell. A possible approach would be to divide the solution into two parts. The first part addresses the vacuum injector with particle orbits terminated at the foil. The `PRT` file created by the first solution is used as input to the gas cell solution which could include both space-charge neutralization and a global current neutralization factor.

10. Field emission of electrons with self-consistent space-charge effects

In the *FEMIT* particle tracking mode, **Trak** can model field emission of non-relativistic electrons. As in the *SCHARGE* mode, the program can automatically generate particles over marked source surfaces. The calculations include self-consistent effects of space charge. You can add additional electrons or ions using the *PLIST* or *PFILE* commands. There are three differences from the *SCHARGE* mode:

- Only electrons can be created on emission surfaces.

- Because the electric field always has a non-zero value on the source surface, it is not necessary to create a generation surface. Electrons are generated directly adjacent to the source facets.

- Electron current is assigned according to the Fowler-Nordheim equation rather than the Child-law algorithm.

Trak uses Fowler-Nordheim functions tabulated in A. Modinos, **Emission Spectroscopy** (Plenum Press, New York, 1984), p. 12. If the quantity *E* is the local electric field (including space-charge contributions) at a source facet with work function φ, then current density is given by

$$j_{FE} = 1.537\times10^{-6}\,\frac{e^{\Gamma}\,E^2}{\phi\;t(\chi)^2}\qquad (A/m^2)\ , \tag{10.1}$$

where

$$\Gamma = -6.83\times10^{9}\,\frac{\phi^{3/2}\;v(\chi)}{E}\ , \tag{10.2}$$

and

$$\chi = 3.79\times10^{-5}\,\frac{\sqrt{E}}{\phi}\ . \tag{10.3}$$

In the equations *E* is expressed in V/m and ϕ in eV. Table 10.1 lists values for the functions *v*(χ) and *t*(χ). **Trak** uses a cubic spline interpolation to find intermediate values.

Table 11.1. Fowler-Nordheim functions

χ	v(χ)	t(χ)
0.0000	1.0000	1.0000
0.0500	0.9948	1.0011
0.1000	0.9817	1.0036
0.1500	0.9622	1.0070
0.2000	0.9370	1.0111
0.2500	0.9068	1.0157
0.3000	0.8718	1.0207
0.3500	0.8323	1.0262
0.4000	0.7888	1.0319
0.4500	0.7413	1.0378
0.5000	0.6900	1.0439
0.5500	0.6351	1.0502
0.6000	0.5768	1.0565
0.6500	0.5132	1.0631
0.7000	0.4504	1.0697
0.7500	0.3825	1.0765
0.8000	0.3117	1.0832
0.8500	0.2379	1.0900
0.9000	0.1613	1.0969
0.9500	0.0820	1.1037
1.0000	0.0000	1.1107

The set of allowed commands is similar to that for the *SCHARGE* mode. The one difference is that alternate parameters should appear in the *EMIT* command.

EMIT WorkFunc [NPerSeg Beta]
EMIT(4) (3.56, 2, 1000.0)

This command identifies region number *RegNo* (integer) as a source surface and sets associated emission properties. **Trak** issues an error message if the region does not define a valid source surface (*i.e.*, sets of nodes on the edges of facets between *Material* and *Vacuum* elements). At least one *EMIT* command is required under the *FEMIT* option. Up to twenty *EMIT* commands may appear in the *PARTICLES* section. Because only electrons are allowed, it is not necessary to specify mass and charge of emitted particles. The real-number parameter *WorkFunc* is the work function ϕ in eV. The integer parameter *NPerSeg* governs how many model electrons are created per surface facet. The optional parameter β is a field enhancement factor that may be useful (for example) to simulate emission from carbon nanotube assemblies. If E_{loc} is the local electric field at the facet, then the quantity $E = \beta E_{\mathrm{loc}}$ is used in Eqs. 10.1 through 10.3.

SUPPRESS SVal1 SVal2 SVal3 ...
SUPPRESS 0.20 0.30 0.40 0.60 0.80 1.00

Space-charge effects are usually small in field emission problems so the role of the suppression factors discussed in Chapter 8 is not as critical. The default values in the FEMIT mode are

```
NCycle  Supression value
========================
   1           0.500
   2           0.750
   3           1.000
   4           1.000
   ...
```

Note that the number of iteration cycles should be in the range *NCycle* ≥ 3 for the default values of suppression factors.

The remaining commands allowed in the *PARTICLES FEMIT* section are identical to those in the *PARTICLES TRACK* mode:

```
DT  Dt
DT DtRef MASS
TMAX  TMax
START(NReg) XStart YStart
NTRACKMAX  NTrackMax
DMAX  DMax
NSEARCH [E,B] NSearch
INTERP [LIN,LSQ]
MATERIAL [E,B] RegNo
VACUUM [E,B] RegNo
SECONDARY [E,B] RegNo DeltaMax0 EngMax0
STOP [UP,DOWN] [X,Y,Z,R] = Position
DIAG [UP,DOWN] [X,Y,Z,R] = Position
REFLECT [UP,DOWN] [X,Y,Z,R] = Position
LISTON [NStep] [P,E,B]
PLOTOFF
```

11. Extraction of high-current ion beams from a free plasma surface

When modeling electron extraction from a thermionic or field emission cathode, we can assume that the profile of the source surface is specified *a priori*. Extraction of ions from a plasma is a more difficult challenge because the surface shape that satisfies all physical constraints is not known in advance. In the *PLASMA* mode, **Trak** performs the Child-law emission calculations of the *SCHARGE* mode described in Chap. 8. In addition, the program automatically flexes the plasma surface to ensure a uniform distribution of current-density. The calculation is complex, so it is important that you understand the physical basis of ion extraction from a plasma and the goals of the calculation. Section 11.1 discusses this topic and summarizes numerical methods applied in **Trak**. Section 11.2 reviews new commands that appear in the *PLASMA* mode, while Section 11.3 discusses benchmark examples.

11.1. Ion extraction from a free plasma surface

This section briefly reviews the physical basis of the *PLASMA* mode in **Trak**. For a complete discussion of Child-law emission, the Bohm current density and the formation of a plasma meniscus, see S. Humphries, **Charged-particle Beams** (Wiley, New York, 1990), Chap. 7. This book has been included on the **Trak** distribution disk and is available for download in PDF format at:

http://www.fieldp.com/cpb/cpb.html

To begin, we must define the term *free plasma extraction surface*. Assume that plasma ions generated by a source expand in a field-free region through an aperture into an acceleration region with an applied electric field (Fig. 11.1). The position of the extraction surface (the transition between the plasma environment and the vacuum flow region) is determined by a balance between the plasma ion flux (which we shall denote by an effective ion current density j_p) and the extraction current density governed by the Child law, j_C. In the one-dimensional geometry of Fig. 11.2, the current density limit for ions with charge-state Z_i and mass m_i in a gap of width d and applied voltage V_0 is given by the expression:

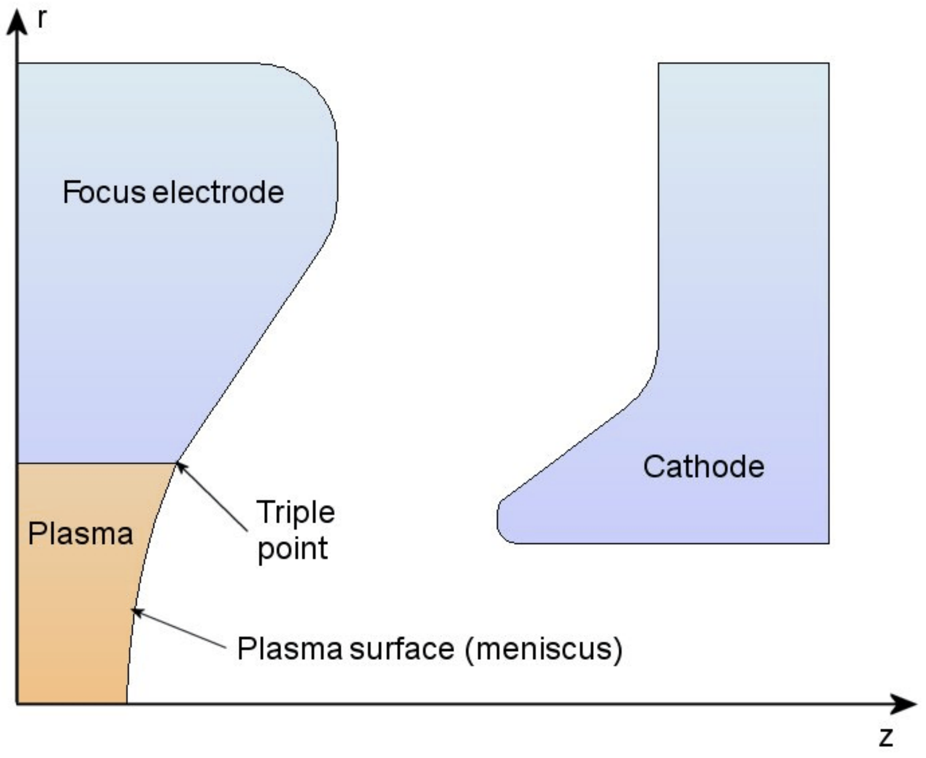


Figure 11.1. Ion extraction from a free plasma surface – terminology.

$$j_C = \frac{4\epsilon_0}{9} \sqrt{\frac{2Z_i e}{m_i}} \frac{V_0^{3/2}}{d^2} \ . \tag{11.1}$$

If the ion flux exceeds the vacuum current limit ($j_p > j_C$), then the plasma expands to decrease the gap width *d* until flux balance is achieved.:

$$j_p = j_C \ . \tag{11.2}$$

In this case, the surface in Fig. 11.2 moves to the right. Conversely, if $j_p < j_C$, the surface recedes to the left. In most plasma sources for ion generation, the ion temperature (T_i) is much smaller than the electron temperature (T_e). In this case, the available effective ion current density is given by the Bohm expression:

$$j_p \cong 0.6 \ eZ_i \ n_i \sqrt{kT_e/m_i} \ . \tag{11.3}$$

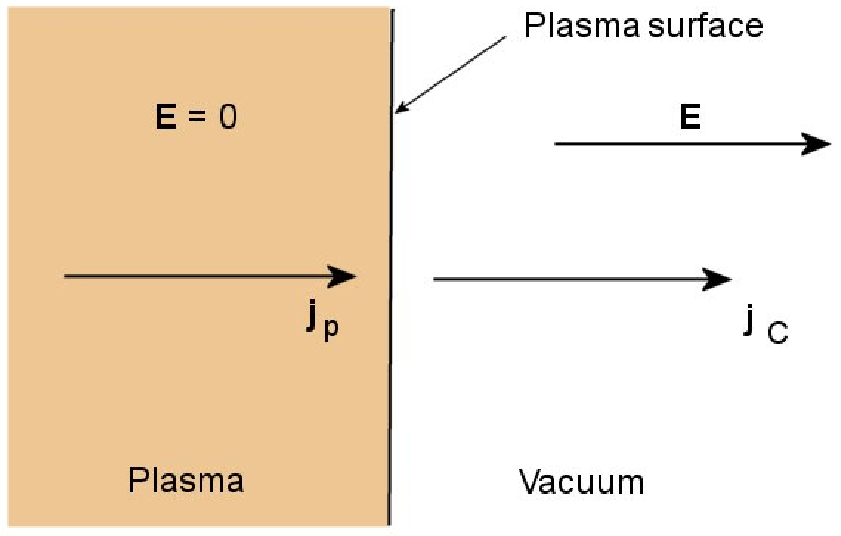


Figure 11.2. One-dimensional extraction from a free plasma surface

The **Trak** model treats a homogenous plasma so that j_p has a uniform value over the extraction surface. Therefore, a *PLASMA* mode simulation has the following primary goal: for given applied voltages and electrode geometry, find a shape of the extraction surface that guarantees a uniform value of j_C. A critical issue is the nature of the extraction surface at the *triple point*: the intersection of the plasma, the aperture electrode and the vacuum acceleration region (Fig. 11.1). Figure 11.3*a* shows the ideal situation where the values of j_p and j_C at the triple point allow a smooth connection between the extraction surface and the focusing electrodes. A reduction in j_p from the optimum value causes the plasma to recede into the aperture. In this case the focusing electrode acts as an electrostatic shield, giving a large reduction in the field amplitude at the plasma edge. The implication is that the plasma edge is effectively tied to the aperture at the triple point. A low value of j_p gives a distorted extraction surface (Fig. 11.3*b*) with attendant poor beam optics. Conversely, a value of j_p above the optimal value causes the plasma to bulge into the extraction gap resulting in a divergent beam (Fig 11.3*c*). In summary, there are two constraints on an acceptable plasma extraction surface:

■ The value of j_C must be uniform.

■ The plasma surface must make a smooth connection at the triple point.

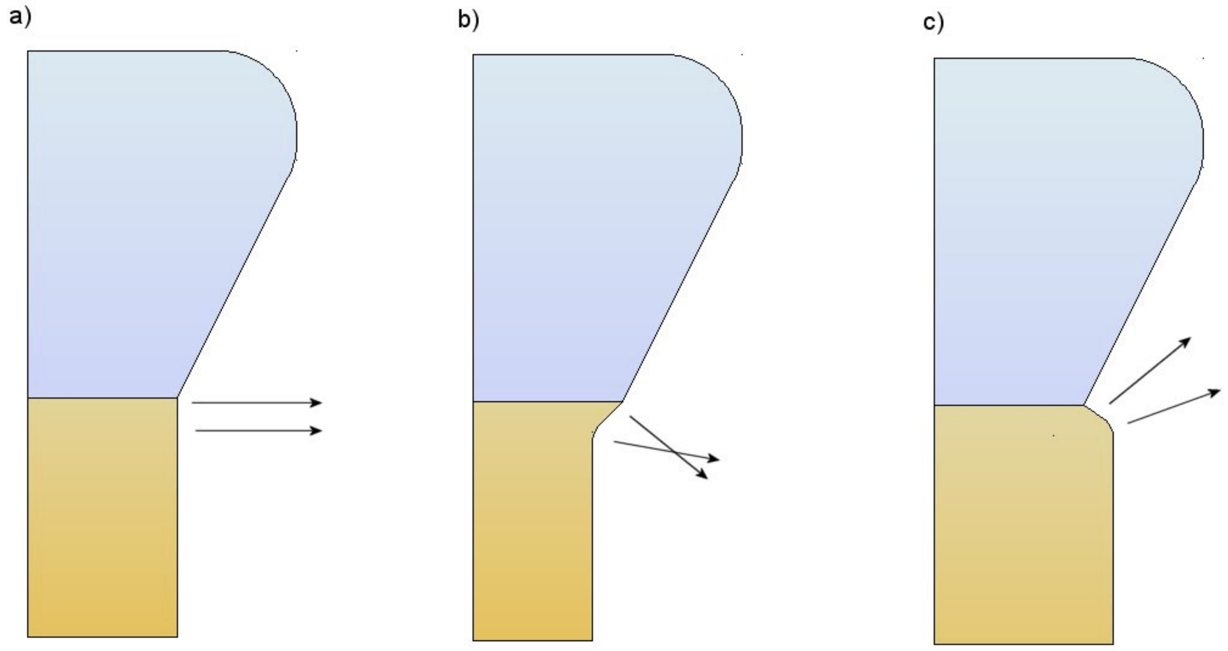
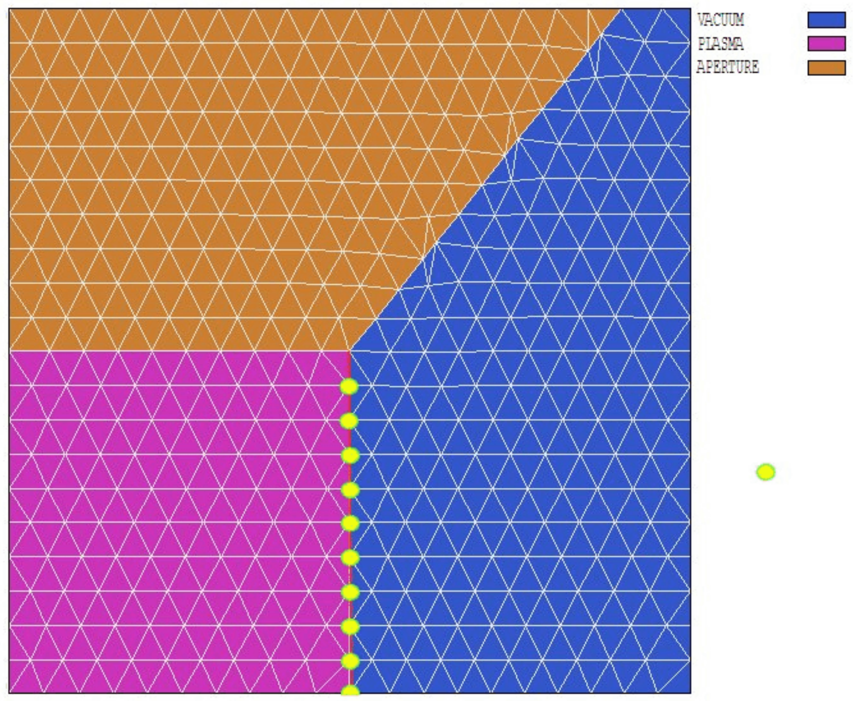


Figure 11.3. Balancing plasma flux j_p with space-charge-limited flux j_C to achieve a smooth connection at the triple point. *a*) Ideal balance. *b*) Insufficient plasma flux. *c*) Excess plasma flux.

The combination of the two conditions with specified electrodes and applied voltages defines a unique plasma surface shape and Bohm current density.

Trak follows the following multi-step procedure to determine self-consistent plasma surfaces. Although the operations are complex, most of them are performed automatically so that the user need not be concerned with details. The benchmark examples of Section 11.3 show that it is relatively easy to set up a *PLASMA* mode calculation.

1. The user creates an electric-field model of the gun geometry in **Mesh**. The mesh must include a filled (volumetric) region to represent the plasma (Fig. 11.1). The plasma region must have enough depth (*i.e.*, number of elements parallel to the displacement direction) to accommodate flexing of the surface. The user makes an initial guess of the extraction surface shape (such as a flat plane). The initial surface should connect smoothly to the aperture at the triple point. The plasma region must appear in the **Mesh** script before the surrounding aperture region. This ordering ensures that nodes on the shared plasma-aperture surface are associated with the aperture.



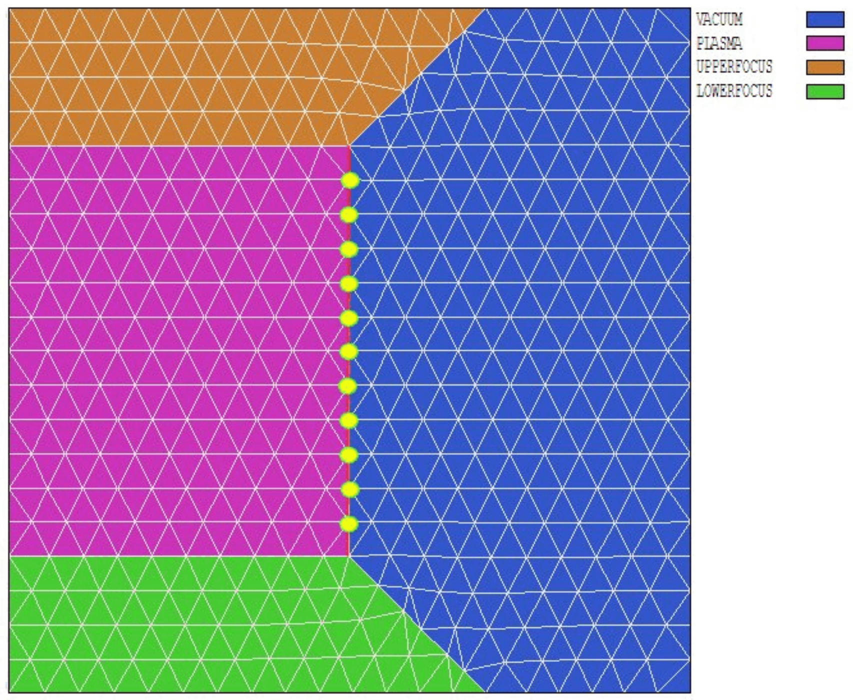


Figure 11.4. Definition of plasma facets (red line) and plasma nodes (yellow dots). Plasma elements are colored violet and vacuum elements are colored blue. Top: Disk plasma with symmetry axis on the lower edge. Bottom: Annular plasma.

2. The user generates an initial electrostatic solution with **EStat**. The plasma and aperture regions should be assigned the fixed-potential condition with the same value of applied voltage.

3. In **Trak**, the plasma region is identified by an *EMIT* statement. The program initially sweeps through the mesh to identify facets and nodes of the source surface. Source facets lie between an element with the region number of the plasma and a *VACUUM* element. Plasma surface nodes lie along the source surface. Figure 11.4 shows the two possible options for assignment of surface nodes: *a*) a cylindrical or planar gun where the surface has an axis of symmetry and *b*) an annular gun.

4. **Trak** orders the facets to form a connected set with respect to distance from the axis or a specified start point. The program determines a set of unit vectors normal to the facets that point out of the plasma region. The vectors are used to construct an emission surface at a distance *DEmit* from the source surface to carry out the Child-law calculation.

5. **Trak** performs a calculation of space-charge-limited flow from the initial surface by creating one or model ions on each facet of the emission surface. The program runs *NInit* cycles to achieve a stable, converged solution using the same procedure as in the *SCHARGE* mode. The physical basis for the Child-law calculation is described in the reference S. Humphries, *Numerical Modeling of Space-charge-limited Charged-particle Emission on a Conformal Triangular Mesh*, J. Computational Phys. **125**, 448 (1996).

6. **Trak** calculates the current density on source facets by averaging and back-projecting the model particle currents at the emission surface. The variation of current density is smoothed to prevent the unstable growth of ripples in the surface during the adjustment procedure.

7. The value of emitted current density at each surface node (j_n) is computed by averaging the values on adjacent facets. In addition, a node unit vector (pointing out of the plasma) is computed by averaging the vectors of adjacent facets. The position of each node is moved along this vector a distance Δ given by:

$$\Delta = \frac{\alpha\ d}{2} \left(1 - \frac{j_n}{j_{out}} \right) . \qquad (11.4)$$

Equation (11.4) follows from the scaling of Eq. (11.1). The quantity d is the width of the acceleration gap, α is a safety factor to ensure stability of the calculation, and j_{out} is the current density on the facet adjacent to the triple point. Normalizing the displacement to j_{out} ensures that the plasma surface intersects the triple point at a small angle. The direction of the shift in Eq. (11.4) is away from the acceleration gap (into the plasma) when $j_n > j_{\text{out}}$. In this case the shift gives a local reduction in the space-charge-limited current density.

8. The positions of plasma and vacuum nodes near the plasma surface are relaxed to preserve element integrity.

9. New normal vectors are computed for the displaced facets and the emission surface is reconstructed.

10. **Trak** performs *NCorrect* cycles of orbit and electric field recalculation to determine current densities for space-charge-limited flow from the new plasma surface. The program then returns to Step 6 to recalculate the plasma surface based on updated values of current density. The cycle of surface regeneration and solution correction is carried out *NSurface* times to achieve a convergent solution.

It is important to recognize cases where the **Trak** model is invalid. The program cannot be applied to systems where there is a strong transverse magnetic field at the extraction surface (*i.e.*, magnetically-insulated ion diodes, transverse extraction from a Penning source,...). In this case, the condition $j_p = j_C$ may not hold. Nonetheless, **Trak** can give useful information on plasma extraction from sources with applied magnetic fields (*i.e.*, duoplasmatron,...) as long as the source has sufficient shielding to isolate the magnetic field from the high-voltage extraction gap. The model also does not apply to sources that produce a mixture of ion species or charge states. In this case the behavior of the extraction sheath is more complex than the simple Child-law model. For example, ions with high-charge state may be preferentially extracted. Finally, we should note that the *PLASMA* mode may also be useful in the design of electron guns with thermionic cathodes. Through an iterative procedure, you can find cathode shapes that will guarantee uniform current density of the extracted beam.

11.2. Plasma mode commands

Trak performs a *PLASMA* mode calculation when the *PARTICLES* section begins with the command:

```
PARTICLES PLASMA
```

The following commands perform the same function as those described in Chap. 8 for the *SCHARGE* mode:

AVG = 0.20
MAXCYCLE = 2500
RESTARGET = 2.0E-8
OMEGA = 1.95
START(5) = 0.25, 0.00
RESTART

Note the *NCYCLE* command is not allowed in the *PLASMA* mode. **Trak** automatically determines *NCycle* from the parameters supplied to control the plasma surface calculation. Note that you can add list particles with the commands:

PLIST
PFILE

The *EMIT* command has a different format in the *PLASMA* mode.

EMIT(NReg) Mass Charge DEmit NPerSeg DGap [DTheta]
EMIT(4) = (4.0, 1.0, 0.05, 3, 2.50)

This command identifies the filled region *NReg* in the electric-field mesh as a plasma. The plasma surface is defined as the boundary of the region adjacent to a *VACUUM* region. The plasma must be a fixed-potential region. Specifications for the following five parameters are required. The parameters *Mass* and *Charge* have the same meaning as in the *EMIT* command of the *TRACK* and *SCHARGE* modes. Enter the mass in AMU (atomic mass unit). **Trak** inserts the value for an electron if *Mass* = 0.0. Enter the charge of the emitted particles in units of *e* (*i.e.*, -1.0 for electrons and +1.0 for protons). The quantity *DEmit* is the distance from the source surface to the emission surface. Enter the value in units set by the current value of *DUnit*. As a rule of thumb, set *DEmit* equal to about 1.5-2.0 times the width of elements adjacent to the plasma surface. Large values of *DEmit* may prevent convergence of the plasma surface search. The integer parameter

NPerSeg is the number of model particles created per facet of the plasma surface. The real-number parameter *DGap* is the approximate width of the acceleration gap. Enter the value in units set by *DUnit*. Larger values of *DGap* may speed the calculation. Reduce *DGap* if the calculation does not converge. The signs of convergence failure are: 1) a wavy of irregular plasma surface, 2) code termination because of distorted elements. The optional parameter *DTheta* (in degrees) equals the angular divergence of particles at the emission surface. **Trak** can handle only a single plasma region in the *PLASMA* mode. Therefore, the *PARTICLES* section must contain one and one *EMIT* command.

The following command (valid only in the *PLASMA* mode) controls the plasma surface calculation.

PLASMAPARAM NSurface NCorrect NInit [NRadius]
PLASMAPARAM 4 5 12

The *PLASMAPARAM* command has four integer parameters, one of them optional. The quantity *NSurface* is the number of cycles of plasma surface adjustment. The quantity *NCorrect* is the number of orbit-field recalculations to determine a stable solution for space-charge-limited flow per surface adjustment. *NInit* is the number of initial orbit-field calculations before the first surface adjustment. The optional parameter *NRadius* controls the width of the region near the surface over which mesh smoothing is applied to plasma and vacuum nodes. (Default values: *NSurface* = 3, *NCorrect* = 5, *NInit* = 12, *NRadius* = 4).

In the remainder of this section, we shall discuss how to choose values for the quantities in the PLASMAPARAM command and how to ensure solution convergence. To begin, the number of required surface adjustments (controlled by *NSurface*) depends on how much the initial surface must be flexed to achieve uniform current density. The solution will converge faster if you make an improved initial guess of the surface shape based on past experience. **Trak** records extensive information on the surface adjustment procedure in the listing file `RUNNAME.TLS`. The table labeled *Adjustment of plasma surface nodes* lists the smoothed values of j_n used in Eq. (11.4) as well as the initial and final values of the plasma surface node coordinates. **Trak** also records the root-mean-squared node displacement. This quantity will be small for a convergent solution. Other indications of convergence are: 1) values of j_n are almost equal and 2) the total emitted current approaches a fixed value.

Successful adjustments using Eq. (11.4) require accurate values for the smoothed, space-charge-limited current density. Therefore, **Trak** must perform several orbit-field recalculations to update the emitted current density for the new source/emission surface positions. The parameter *NCorrect* controls the number of orbit-field recalculations per surface adjustment. The value should be large enough to ensure that total emitted current approaches a stable value between surface adjustments. The parameter *NInit* equals the number of orbit-field recalculations before the first surface adjustment. The number must be sufficient to achieve a converged solution for the initial surface geometry. You can reduce *NInit* by loading a previously-computed solution for space-charge-limited flow from the initial surface and by using the *RESTART* command.

Finally, the parameter *NRadius* controls the half-width of the region near the plasma surface over which mesh smoothing occurs. If a plasma node has indices K_0, L_0, then graded smoothing is applied to nodes in the range K_0 - *NRadius* $\leq K \leq K_0$ + *NRadius*, L_0 - *NRadius* $\leq L \leq L_0$ + *NRadius*. The value of *NRadius* must be larger than the number of elements over which the surface is displaced from the intial guess. Increase *NRadius* if **Trak** reports an element distortion in response to a surface shift. You can reduce the chances for element distortion by making a better guess of the initial surface.

11.3. Plasma mode examples

This section describes some simple examples that illustrate the physics of ion extraction from a plasma as well as **Trak** modeling techniques. Figure 11.5 shows the geometry of a baseline calculation for a cylindrical proton gun defined by the files `PLASMA01.MIN, PLASMA01.EIN` and `PLASMA01.TIN`. The aperture has a radius of 2.0 cm, the extraction gap has width d = 4.0 cm and the applied voltage is V_0 = 50 kV. The focusing electrode is inclined at an angle θ = 22.5° and the cathode is a plane on the right-hand side of the solution volume. With these parameters we expect that the solution will approximate an ideal Pierce gun. (Note that the Pierce derivation holds exactly only for a planar gun). Therefore the ion current density j_C should be almost uniform over a flat plasma surface that intersects the triple point. The current-density magnitude is given approximately by Eq. (11.1): j_C = 380.5 A/m^2. For comparison, the the **Trak** calculation in the *PLASMA* mode gives a surface with a convexity of 0.009 cm on axis and average current density $\langle j_C \rangle$ = 402.6 A/m^2. In the final state, the current density is uniform over the surface to within ±0.06%. The calculation illustrates the stability and convergence of the *PLASMA* mode procedure.

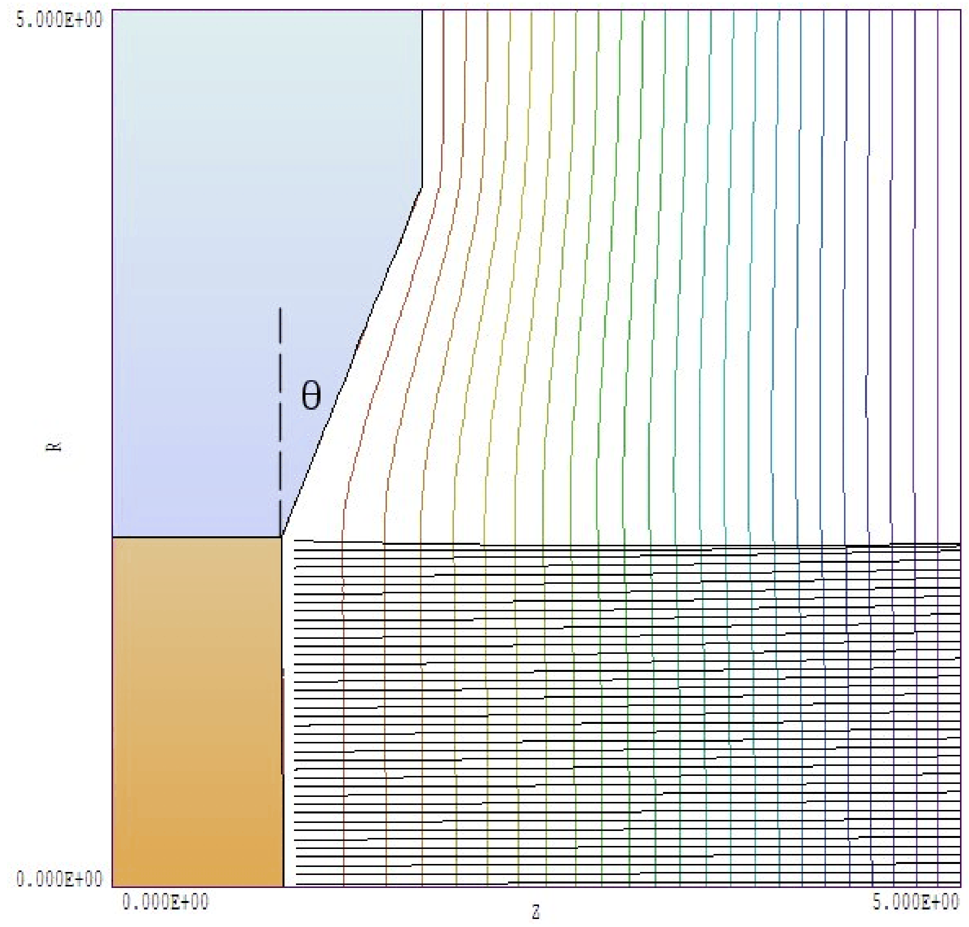


Figure 11.5. Geometry of the example PLASMA01 with focusing electrode angle θ = 22.5°. Figure shows the self-consistent plasma surface, equipotential contours and model particle orbits. Dimensions in cm.

Next, suppose we increase the focus-electrode angle to 30° (example `PLASMA02`). Table 11.1 shows the **Mesh** input file. (Note that the plasma region appears before the focusing electrode.) The extra metal reduces the electric field on the outer edge of the aperture, suppressing the space-charge-limited current density. Therefore, emission is non-uniform over a flat surface that intersects the trip point (Fig. 11.6). In this case, **Trak** must flex the plasma to create a concave surface, thereby reducing the current density near the axis.

Table 11.1. Mesh input file PLASMA02.MIN

```
* File: Plasma02.MIN
*   30.0 degree focus angle
*  ----------------------------------------------------------
GLOBAL
  XMESH
       0.00 0.80  0.100
       0.80 1.20  0.050
       1.20 5.00  0.100
    END
  YMESH
       0.00 7.50  0.100
    END
  SMOOTH 2
END
*  ----------------------------------------------------------
REGION  FILL Vacuum
     L    0.00000E+00  0.00000E+00  5.00000E+00  0.00000E+00
     L    5.00000E+00  0.00000E+00  5.00000E+00  7.50000E+00
     L    5.00000E+00  7.50000E+00  0.00000E+00  7.50000E+00
     L    0.00000E+00  7.50000E+00  0.00000E+00  0.00000E+00
END
*  ----------------------------------------------------------
REGION  FILL Plasma
     L    0.00000E+00  2.00000E+00  1.00000E+00  2.00000E+00
     L    1.00000E+00  2.00000E+00  1.00000E+00  0.00000E+00
     L    1.00000E+00  0.00000E+00  0.00000E+00  0.00000E+00
     L    0.00000E+00  0.00000E+00  0.00000E+00  2.00000E+00
END
*  ----------------------------------------------------------
REGION  FILL Focus
     L    0.00000E+00  2.00000E+00  1.00000E+00  2.00000E+00
     L    1.00000E+00  2.00000E+00  2.15500E+00  4.00000E+00
     L    2.15500E+00  4.00000E+00  2.15500E+00  7.50000E+00
     L    2.15500E+00  7.50000E+00  0.00000E+00  7.50000E+00
     L    0.00000E+00  7.50000E+00  0.00000E+00  2.00000E+00
END
*  ----------------------------------------------------------
REGION Cathode
     L    5.00000E+00  0.00000E+00  5.00000E+00  7.50000E+00
END
*  ----------------------------------------------------------
ENDFILE
```

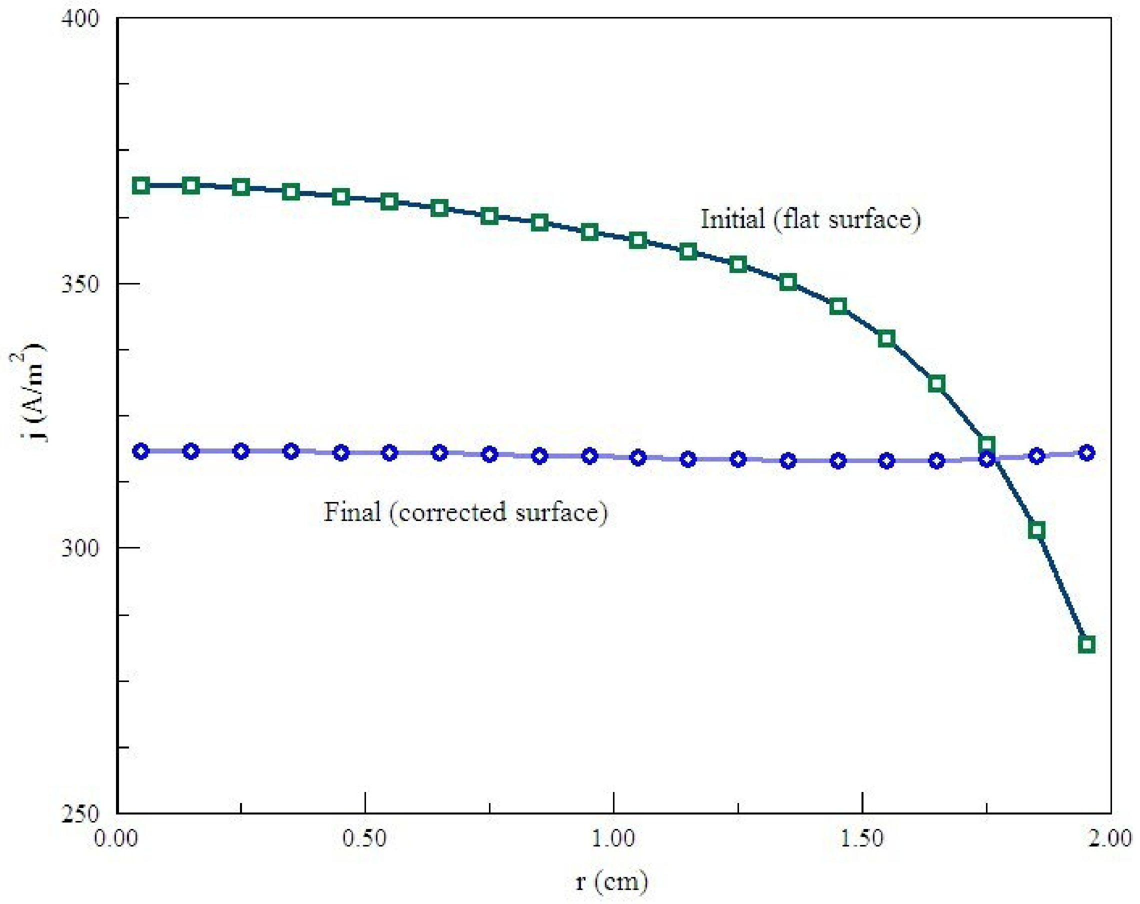


Figure 11.6. Radial variation of current density for example `PLASMA02` with θ = 30°. Green curve shows the initial state while the blue curve shows the current density after the iterative plasma surface adjustment.

Table 11.2 shows that **Trak** input script. The *EMIT* command specifies proton generation using an emission surface a distance *DEmit* = 0.075 cm from the source surface. For comparison, the local element width along *z* is 0.050 cm. Two model particles are created per facet and the gap width is *DGap* = 4.0 cm (Fig. 11.5). The *PLASMAPARAM* command specifies 4 cycles of surface adjustment with 5 orbit-field recalculations for each cycle. There are 12 initial orbit-field recalculations to achieve an accurate starting solution. Figure 11.7 (which shows the root-mean-squared surface displacement) illustrates convergence of the iterative procedure.

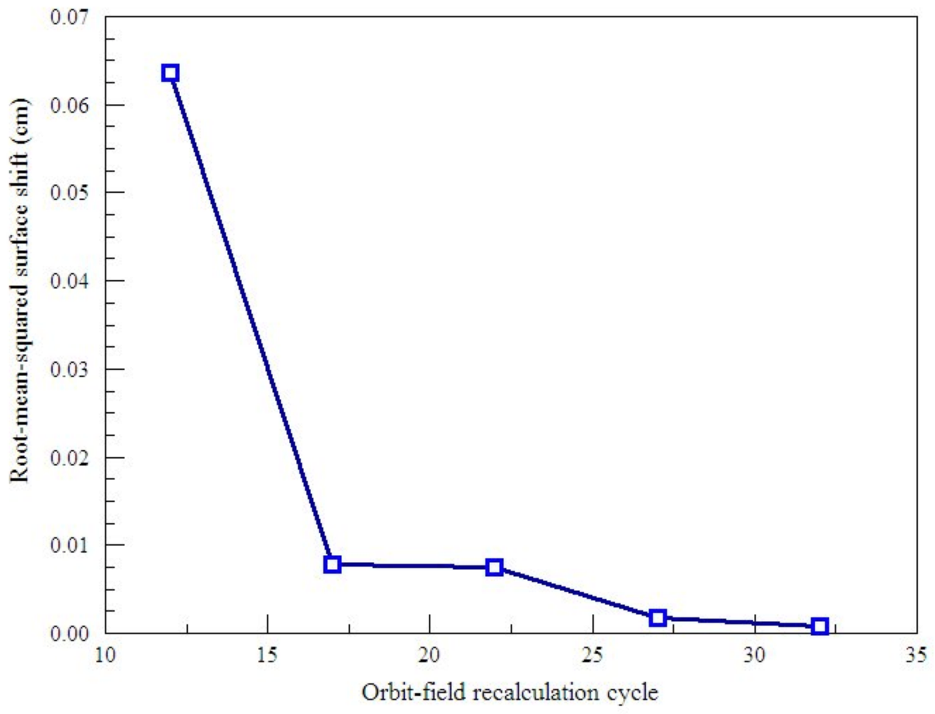


Figure 11.7. Convergence history for example PLASMA02. Root-mean-squared surface displacement versus cycle number.

Table 11.2. Trak input file PLASMA02.TIN

```
* FILE: Plasma02.TIN
FIELDS
    EFILE: Plasma02.EOU
    DUNIT: 100.0
    END
PARTICLES Plasma
   EMIT(2) 1.0 1.0 0.075 2 4.0
   PLASMAPARAM 4 5 12
   AVG 0.30
END
DIAGNOSTICS
   JSOURCE
   PARTLIST
   EDUMP Plasma02P
END
ENDFILE
```

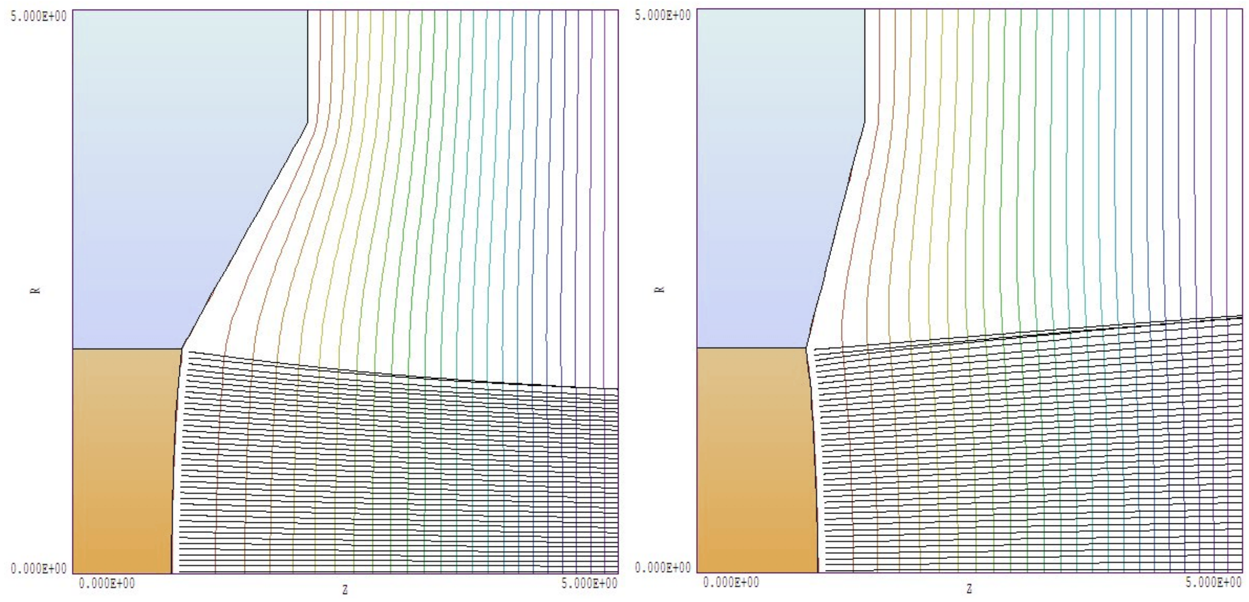


Figure 11.8. Self-consistent plasma surface shape, equipotential lines and model particle orbits. Left: example `PLASMA02` with θ = 30°. Right: example `PLASMA03` with θ = 15°

The left-hand-side of Fig. 11.8 shows the final concave shape of the plasma surface along with computed equipotential lines and model particle orbits in the converging beam. The right-hand side shows the solution for a shallow focusing-electrode angle (θ = 15°). In this case, the electric field magnitude is higher at the outer radius so that the plasma must assume a convex shape for uniform current density.

The examples suggest the following logical sequence for the design of a practical cylindrical ion gun:

1. Starting from the Pierce solution (θ = 22.5°) with flat emission surface, we add an exit aperture in the cathode. The effect of the aperture is to reduce the electric-field magnitude on the plasma surface near the axis.

2. To compensate for the reduction in on-axis current density, we increase the angle of the focusing electrode to suppress emission near the outer radius of the plasma surface.

3. A further increase in the focusing electrode angle would give a converging beam. In this case, it is possible to reduce the radius of the exit aperture (Fig. 11.1).

12. Modeling secondary emission of electrons

Commands introduced in this chapter

Secondary
SecondParam

12.1. Models for secondary electron emission

Trak can represent electron secondary-emission processes to model electro-optical devices and collectors for high-power vacuum tubes. These capabilities apply only to electrons and can be used in the *TRACK, SCHARGE, RELBEAM* and *FEMIT* modes. In the latter modes the space-charge of emitted electrons is added to the electric field recalculation. A single model electron orbit can represent a multi-generational set of electrons. Regions with elements corresponding to secondary emitters are defined with the *SECONDARY* command. When an electron enters a secondary element, the secondary-emission coefficient δ is calculated from the incident energy and particle angle relative to the surface at the entrance point. The current of the electron is multiplied by δ and it is restarted at a point in the vacuum space near the surface entrance point. The kinetic energy of the emitted electron follows a Maxwell distribution with $T_e = 2.0$ eV. The direction of emission relative to the surface follows a cosine weighting function (equal flux per solid angle).

Trak determines the secondary emission coefficient from a parametric model based on work by Jonker[1] and Vaughn[2]. The model involves the angle α between the direction of the incident electron and a vector normal to the surface. The defining function for the angular dependence of δ is

$$F(\alpha) = \frac{1}{\sqrt{\cos(\alpha)}} \ . \qquad (12.1)$$

The maximum value of secondary coefficient and the corresponding incident electron energy are given by

$$\delta_m = \delta_{mo}\, F(\alpha),$$
$$E_m = E_{mo}\, F(\alpha), \qquad (12.2)$$

where δ_{mo} and E_{mo} are the values at normal incidence. These quantities are tabulated for a variety of materials in Table 12.1. The secondary emission coefficient as a function of the angle α and kinetic energy *E* of the incident electron is given approximately as

$$\delta(\alpha,E) \cong \delta_m(\alpha) \left[f\, e^{(1-f)} \right]^a , \qquad (12.3)$$

where $f = E/E_m(\alpha)$. The parameter *a* has the value 0.62 for $f < 1$ and 0.25 for $f \geq 1$. To prevent infinite values of δ, the code takes α = 80° for orbits with incident angles that exceed 80°. Reference 3 (included with the package) gives a detailed description of numerical methods used to treat secondary emission in the **Trak** code.

---

1. J.L.H. Jonker, Phillips Research Reports **6**, 372 (1951), Phillips Research Reports **7**, 1 (1952), Phillips Research Reports **12**, 249 (1957).

2. R.M. Vaughn, IEEE Trans. Electron Devices **ED-36**, 1963 (1989) and IEEE. Trans. Electron Devices **ED-40**, 830 (1993).

3. S. Humphries, N. Dione and J. Petillo, *Secondary-electron emission modeling on a conformal mesh*, **Proc. WorkShop on RF Superconductivity**, Santa Fe, 1999.

Table 12.1. Secondary emission parameters for elements

Element	δ_{mo}	E_{mo} (eV)	Element	δ_{mo}	E_{mo} (eV)
Ag	1.5	800	Li	0.5	85
Al	1.0	300	Mg	0.95	300
Au	1.4	800	Mo	1.25	375
B	1.2	150	Na	0.82	300
Ba	0.8	400	Nb	1.2	375
Bi	1.2	550	Ni	1.3	550
Be	0.5	200	Pb	1.1	500
C (diamond)	2.8	750	Pd	>1.3	>250
C (graphite)	1.0	300	Pt	1.8	700
C (soot)	0.45	500	Rb	0.9	350
Cd	1.1	450	Sb	1.3	600
Co	1.2	600	Si	1.1	250
Cs	0.7	400	Sn	1.35	500
Cu	1.3	600	Ta	1.3	600
Fe	1.3	400	Th	1.1	800
Ga	1.55	500	Ti	0.9	280
Ge	1.15	500	Tl	1.7	650
Hg	13	600	W	1.4	650
K	0.7	200	Zr	1.1	350

Adapted from D.R. Lide, ed., **Handbook of Chemistry and Physics, 74th Edition** (CRC Press, Boca Raton, 1993), 12-107

Table 12.2. Secondary emission parameters for compounds

Element	δ_{mo}	E_{mo} **(eV)**	**Element**	δ_{mo}	E_{mo} **(eV)**
Alkali halides			**Oxides**		
CsCl	6.5		Ag_2O 1.0	2-9	
KBr (crystal)	14	1800	$A1_2O_3$ (layer)	2.3-4.8	400
KCl (crystal)	12	1600	Ba0 (layer)	3.4	2000
KCl (layer)	7.5	1200	Be0	2.2	500
KI (crystal)	10	1600	Ca0	1.2	400
KI (layer)	5.6		Cu_20	20-25	1500
LiF (crystal)	8.5		Mg0 (crystal)	3-15	400-1500
LiF (layer)	5.6	700	Mg0 (layer)	1.2	
NaBr (crystal)	24	1800	MoO_2	2.1-4	400
NaBr (layer)	6.3		SiO_2 (quartz)	3.2	640
NaCl (crystal)	14	1200	SnO_2		
NaCl (layer)	6.8	600	**Sulfides**		
NaF (crystal)	14	1200	MoS_2	1.1	
NaF (layer)	5.7		PbS	1.2	500
NaI (crystal)	19	1300	WS_2	1.0	
NaI (layer)	5.5		ZnS	1.8	350
RbCl (layer)	5.8		**Others**		
			BaF_2 (layer)	4.5	
			CaF_2 (layer)	3.2	
			$BiCs_3$	6	1000
			BiCs	1.9	1000
			GeCs	7	700
			Rb_3Sb	7.1	450
			$SbCs_3$	6	700
			Mica	2.4	350
			Glasses	2-3	300-450

Adapted from D.R. Lide, ed., **Handbook of Chemistry and Physics, 74th Edition** (CRC Press, Boca Raton, 1993), 12-107

12.2. Control commands

The following commands may appear in the *TRACK, SCHARGE, RELBEAM* and *FEMIT* sections.

SECONDARY [E,B] NReg DeltaMax0 EngMax0
SECONDARY (E,5): 2.45 320.0

This command designates that region *NReg* in the electric or magnetic field meshes is a *SECONDARY* type and assigns emission parameters. In contrast to the simple *VACUUM* and *MATERIAL* specifications, the command requires two parameters. The real-number quantity *DeltaMax0* is the maximum value of the secondary emission coefficient for normal incidence. The quantity *EngMax0* (real) is the kinetic energy of the incident electron (in eV) at which the maximum occurs.

SECONDPARAM KECutOff MinFact
SECONDPARAM 1.0 2.0E-3

This command sets global parameters that control the termination of multi-generational electron orbits to prevent infinite calculations. *KECutOff* is a cutoff value (in eV) for kinetic energy. An orbit terminates if the energy of the incident electron falls below this value. The default is *KECutOff* = 2.5 eV. During a multi-generational orbit calculation **Trak** maintains a quantity *MultFact* equal to the effective number of electrons in a generation per incident electron. This quantity equals the product of secondary emission coefficients for all collisions with secondary materials, $MultFact(N) = \delta_1\delta_2\delta_3...\delta_N$. An orbit is terminated if the multiplication factor drops below the value *MinFact*. The default is $MinFact = 1.0 \times 10^{-4}$.

SECONDLIST

In response to this command **Trak** writes a detailed record of secondary emission events during an orbit integral to the listing file. In runs with *NCycle* > 1 a record is made only on the final cycle.

Finally, additional information on particle multiplication is contained in data written in response to the *REGLIST* command in the *DIAGNOSTICS* section.

13. Particle and field diagnostics

Commands introduced in this chapter

NScan	**BBDump**
Interp	**BBPoint**
EDump	**BBScan**
EPoint	**BBBoundary**
EScan	**PartList**
BDump	**PartFile**
BPoint	**RegList**
BScan	**DispList**
	JSource

Trak records information on the progress of the run in the listing file `FPrefix.TLS`. After computing particle orbits, the program can perform several diagnostic operations in response to commands in the input file. These are contained in the *DIAGNOSTICS* section that follows the *PARTICLES* section. Diagnostic commands are contained between the lines *DIAGNOSTIC* and *END*. After the *DIAGNOSTICS* section (or if there is no section), **Trak** continues to the *ENDFILE* command, closes all files, and terminates operation.

13.1. General control commands

Diagnostic commands divide into five classes: 1) general control, 2) electric fields, 3) applied magnetic fields, 4) beam-generated magnetic fields and 5) field-line or particle orbit quantities. There are two general control commands:

NSCAN NScan
NSCAN = 100

This command sets the number of intervals for line scans initiated by the *ESCAN*, *BSCAN* or *BBSCAN* commands. The default value is *NScan* = 50.

INTERP [E,B,BB] [LIN,LSQ]
INTERP(BB) = LIN

The electric and magnetic field calculations in the *EPOINT*, *ESCAN*, *BPOINT*, *BSCAN*, *BBPOINT* or *BBSCAN* commands can use either the linear or least-squares-fit method. The string parameter can have the values *LIN* or *LSQ*. The command affects subsequent field calculations. Multiple *INTERP* commands can appear in the *DIAGNOSTICS* section if you want to compare the accuracy of the two methods.

DUNIT DUnit
DUNIT = 39.37

Change the unit conversion factor for the input of spatial quantities in the diagnostic commands.

13.2. Electric field diagnostics

Three commands can be used to record the final state of the electric field. These calculations function only if an electric field mesh has been loaded. Note that calculations are performed on the final field solution which may include the effects of scaling, spatial shifts and beam space charge.

EDUMP FPrefix
EDUMP = KlyMod

In response to this command, **Trak** records a file of electric field values with the name `FPrefix.EOU` in the current directory. The file is in standard **EStat** format. It can be inspected with **VEStat** or **VTrak**. The file can also be used as input to a subsequent **Trak** run. For instance, you may want to generate detailed edge orbits in the self-consistent fields of an ion gun.

EPOINT = (X, Y, Z)
EPOINT = (0.0, 0.0, 12.0)

This command instructs the program to record a computation of the electric field at the point (*X*,*Y*,*Z*) in the listing file using the current interpolation method. Enter the coordinates in units set by the current value of *DUnit*.

ESCAN (X1, Y1, Z1) (X2, Y2, Z2)
ESCAN (0.0, 0.0, 12.0) (5.0, 0.0, 12.0)

This command instructs the program to list electric field values along the scan line from (*X1*,*Y1*,*Z1*) to (*X2*,*Y2*,*Z2*). Enter the coordinates in units set by *DUnit*.

13.3. Magnetic field diagnostics

The following commands record values of applied magnetic field if a magnetic solution has been loaded with the *BFILE* command. These values reflect the actions of scaling and shift operations.

BDUMP = FPrefix
BDUMP = SolShift

In response to this command, the program records a file of the magnetic field values used in the simulation with the name `FPrefix.BOU` in the current directory. This file is in standard **BStat** format. It can be inspected with **VBStat** and **VTrak** or used as input to subsequent **Trak** runs. The command functions only if a finite-element solution has been loaded with the *BFILE* command.

BPOINT = (X, Y, Z)
BPOINT = (3.0, 3.0, 0.0)

In response to this command, **Trak** lists the value magnetic field calculated from *all sources* at the point (*X*,*Y*,*Z*). Enter coordinates in units set by the current value of *DUnit*. Depending on the commands of the *FIELDS* section, the calculation may include contributions from a finite-element mesh, a table of on-axis values, uniform field components, or an azimuthal field created by a wire.

BSCAN (X1, Y1, Z1) (X2, Y2, Z2)
BSCAN (2.0, 0.0, -3.0) (2.0, 0.0, 15.0)

List magnetic field values calculated from all sources along the scan line from (*X1*,*Y1*,*Z1*) to (*X2*,*Y2*,*Z2*). Enter the coordinates in units set by *DUnit*.

13.4. Diagnostics of beam-generated magnetic fields

The following commands write values of the beam-generated magnetic field. The command functions only for *RELMODE* calculations.

BBDUMP = FPrefix
BBDUMP = HTest

In response to this command, **Trak** records a file of node values used for the calculation of beam-generated magnetic field to the file `FPrefix.BBD` in the current directory. The file has a format similar to the output files of **EStat** and **BStat**. For a cylindrical simulation the recorded quantities are B_θ (in tesla) and the enclosed current at the node (in A). The quantities in a planar simulation are B_z and the enclosed linear current in A/m. You can load the file in **VTrak** and create plots of the beam-generated magnetic field.

BBPOINT = (X, Y, Z)
BBPOINT: 5.5 6.2 12.3

The program lists the beam-generated magnetic field at the point (*X*,*Y*,*Z*). Enter coordinates in units set by *DUnit*.

BBSCAN = (X1, Y1, Z1) (X2, Y2, Z2)
BBSCAN = (2.0, 0.0, -5.0) (2.0, 0.0, 13.0)

The programs lists values of the beam-generated magnetic field along the scan line from (*X1*,*Y1*,*Z1*) to (*X2*,*Y2*,*Z2*). Enter the coordinates in units set by *DUnit*.

BBBOUNDARY NReg
BBBOUNDARY(5)

In response to this command the program writes a list of facet currents along a collector surface composed of nodes with region number *NReg*. **Trak** locates the intersection of the collector surface with the axis and then moves radially outward calculating the total enclosed current and current density from the associated facet currents and areas. The routine can provide useful calculations of current density variations on target surfaces. You may want to define special collector surfaces and/or apply the *RELBEAM* mode in non-relativistic calculations to utilize the capabilities of the *BBBOUNDARY* command..

13.5. Orbit diagnostics

The final set of commands writes information about particle orbits (or field lines) to the listing file. The final positions used for the calculation may depend on the presence of *DIAG* or *STOP* commands in the *PARTICLES* section.

PARTLIST [SymType]
PARTLIST Rect

In response to this command **Trak** creates a formatted listing of initial and final positions and momenta for a particle simulation. The list contains only coordinate values in a field line simulation. The parameter *SymType* can have the values *RECT* or *CYLIN*. In the default mode (*RECT*) components are given in Cartesian coordinates. If *SymType* = *CYLIN*, then values are converted to cylindrical components relative to the *z* axis.

PARTFILE FPrefix
PARTFILE = Part02

In response to this command **Trak** writes a file `FPrefix.PRT` of final orbit parameters to the current directory in the standard particle file format. This file can be used as input to a subsequent **Trak** run or for distribution analyses in **VTrak**. This command does not function in the *FLINE* mode.

REGLIST

This command initiates a list of particle collisions with material volumes organized by region. Information includes the total number of hits, the weighted number of particle collisions in secondary simulations, and the total deposited current. Note that region 0 includes all other particle termination events (i.e., exited the solution volume, exceeded *TMax*, ...).

PARTDIST

In response to this command **Trak** writes a list of beam distribution properties using the final orbit parameters. Calculations are performed relative to the *z*-axis, independent of the symmetries of the field solutions. Quantities include the root-mean-squared widths, angular divergences and emittances in the *x* and *y* directions. If the simulation involves space charge, the distributions are weighted by the model particle currents. You can perform similar calculations in **VTrak** by creating a particle output file with the *PARTFILE* command.

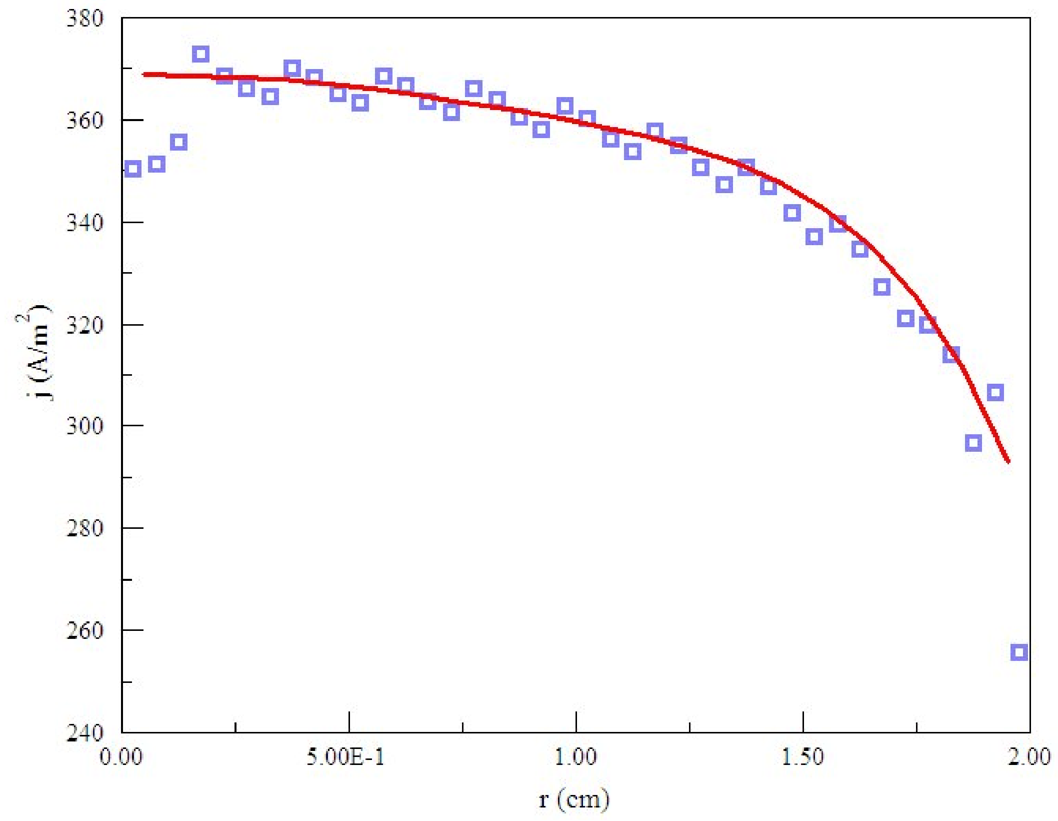


Figure 13.1. Weighted current-density smoothing for a cylindrical beam. Blue squares represent values for individual model particles, the red line is a fourth-order fit. Note the effects of small interpolation errors at element boundaries and near the axis.

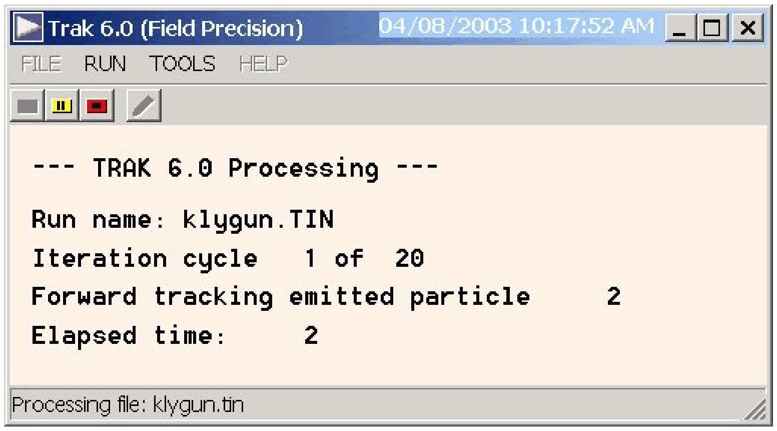
DISPLIST

This command initiates a simple listing of initial and final orbit displacements from the axis. In cylindrical simulations the tabulated quantities are r_i and r_f, while y_i and y_f are recorded for planar simulations. This information can be useful to assess the optical quality of a charged-particle gun. If the gun produces a laminar beam, a plot of r_i-r_f defines a straight line.

JSOURCE

In response to this command, **Trak** writes a list of smoothed current density from emission facets ordered by region. An example is shown below. The smoothing routine uses a least-squares-fit to a fourth-order polynomial. The independent variable is distance from the start point of the region. Only even terms in *r* (or *y*) are used if the emission surfaces contacts an axis of symmetry. In the case of cylindrical solutions, a weighting function proportional to *r* is applied so that errors in the calculated current density for low-current particles near the axis do not skew the results. Figure 13.1 illustrates the fitted function for a cylindrical beam.

```
 ---  Smoothed emission-facet current density ---
   N   NReg    ZAvg       RAvg       JAvg
 ===========================================
    1     2  1.009E+00  5.000E-02  4.020E+02
    2     2  1.009E+00  1.500E-01  4.020E+02
    3     2  1.009E+00  2.500E-01  4.020E+02
    4     2  1.008E+00  3.500E-01  4.020E+02
    5     2  1.008E+00  4.500E-01  4.021E+02
    6     2  1.007E+00  5.500E-01  4.021E+02
    7     2  1.007E+00  6.500E-01  4.021E+02
    8     2  1.006E+00  7.500E-01  4.022E+02
    9     2  1.005E+00  8.500E-01  4.023E+02
   10     2  1.005E+00  9.500E-01  4.023E+02
   11     2  1.004E+00  1.050E+00  4.024E+02
   12     2  1.004E+00  1.150E+00  4.024E+02
   13     2  1.004E+00  1.250E+00  4.024E+02
   14     2  1.003E+00  1.350E+00  4.024E+02
   15     2  1.003E+00  1.450E+00  4.025E+02
   16     2  1.003E+00  1.550E+00  4.024E+02
   17     2  1.003E+00  1.650E+00  4.024E+02
   18     2  1.002E+00  1.750E+00  4.023E+02
   19     2  1.001E+00  1.850E+00  4.023E+02
   20     2  1.000E+00  1.950E+00  4.021E+02
```



```
--- TRAK 6.0 Processing ---

Run name: klygun.TIN
Iteration cycle    1 of   20
Forward tracking emitted particle     2
Elapsed time:     2
```

Processing file: klygun.tin

Figure 14.1. Screen shot of **Trak** in the interactive mode

14. Running Trak in the interactive and autonomous modes

14.1. Interactive mode

The program `Trak.EXE` can run interactively in a window (Fig. 14.1). In this mode you can perform several solutions in a session and temporarily leave the program to work on other tasks. You can launch the program from the **TC** program launcher or create you own shortcuts.

The program has three popup menus: *File, Run* and *Help*. The following commands appear in the *File* menu.

Create input file

Use this command to write a new script file using **Trak**'s internal editor. To help you remember the available commands, the editor starts from a template that contains a command summary as a comment section. Upon exiting, **Trak** creates the file `PREFIX.TIN` from the prefix that you supply.

Update TIN file syntax

Because of the extensive new features in **Trak** 6.0, it was necessary to modify and to streamline the script syntax. To help long-time users make the adjustment, we have included a command for automatic syntax checking of existing scripts. Run this command and pick a `TIN` file in the dialog. **Trak** flags errors and add suggested corrections to the file. The original file is saved as `FPREFIX.BAK`.

Edit input files (TIN)
Edit listing files (TLS)
Edit file

The commands call up the internal editor to inspect or to modify ASCII input and output files. With the *Edit input file* command you can work on files with names of the form `FPREFIX.TIN`. With the *Edit listing file* command you can pick files with names of the form `FPREFIX.TLS`. The *Edit file* command shows all available files. Choosing a file from an alternate directory does not change the working directory of the program.

The *Run* menu has three commands.

Start run

Pick an input file with a name of the form `FPREFIX.TIN` to start a solution. The working directory changes if you pick a file from an alternate directory. The run begins if the all required files are available in the working directory. **Trak** writes information to the screen during extended operations such as orbit tracking and field solution updates.

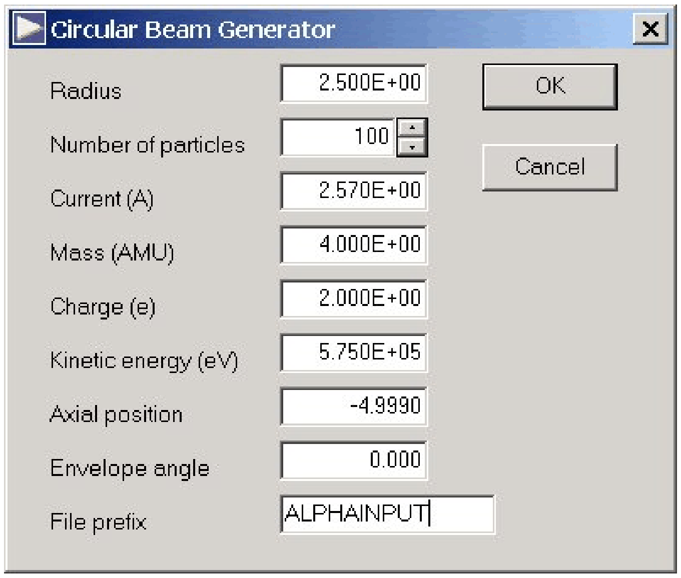
Pause run

The intensive calculations of **Trak** demand the full resources of your computer, causing other tasks to run slowly. If you need to perform critical work, you can pause the solution program during the relaxation process and restart it later without loss of data.

Stop run

This command terminates the program during the relaxation process and saves available information.

We intend to add several useful tools to **Trak** to aid in the design of charged particle devices. The current *TOOL* menu contains the following command:



Radius	2.500E+00
Number of particles	100
Current (A)	2.570E+00
Mass (AMU)	4.000E+00
Charge (e)	2.000E+00
Kinetic energy (eV)	5.750E+05
Axial position	-4.9990
Envelope angle	0.000
File prefix	ALPHAINPUT

Figure 14.2. Dialog for the circular beam tool.

CircBeam generator

Although the full resources of **GenDist** are available to create a variety of input distributions, many applications call for a uniform-current-density, laminar circular beam. The circular-beam tool (Fig. 14.2) offers a quick way to create a standard particle file (*PRT*). It generates a set of *NPart* orbits uniformly spaced in *r*. The assigned current is proportional to *r* to give a uniform current density. The average beam motion is directed along *z*. A non-zero value for the envelope angle gives a diverging or converging beam with momentum components in the *x* direction.

The *Help* menu has a single command.

Program manual

Displays the HTML help file using your default browser. In order for this command to operate correctly, the file `Trak60.HTML` must be in the same directory as `Trak.EXE`.

14.2 Command line operation

Batch file control is a useful option for running powerful technical programs like **Trak**. You can prepare scripts to organize complex operations. The sequenced programs run automatically in the background .

To make a single **Trak** simulation in the background, go to the Command Prompt from Windows and log to the data directory that contains the required input files. For example, suppose the input files `SWITCH.TOU` and `SWITCH.TIN` are stored in `\TRICOMP\BUFFER` and that the program `Trak.EXE` is in the directory `\TRICOMP`. From `\TRICOMP\BUFFER` type

```
..\Trak SWITCH <Enter>
```

The program runs silently, writing detailed information in the listing file `SWITCH.TLS` and the plot file `SWITCH.TOU`. During lengthy runs you can perform other tasks in Windows. Note that the response of the computer will be considerably slower because **TriComp** programs seek to use the full power of the CPU.

14.3. Batch files

The main function of the command mode is autonomous operation under batch file control. As an example, assume you have prepared the input files `SWITCH01.MIN,...,SWITCH08.MIN`, `SWITCH01.EIN,...,SWITCH08.EIN` and `SWITCH01.TIN,...,SWITCH08.TIN` in the directory `\TRICOMP\BUFFER`.. Next you create the following batch file `SW_RUN.BAT` in the data directory using an ASCII text editor.

```
ECHO OFF
ECHO Main switch data run
START ..\MESH.EXE SWITCH01
START ..\ESTAT.EXE SWITCH01
START ..\TRAK.EXE SWITCH01
START ..\MESH.EXE SWITCH02
START ..\ESTAT.EXE SWITCH02
START ..\TRAK.EXE SWITCH02
 ...
START ..\MESH.EXE SWITCH08
START ..\ESTAT.EXE SWITCH08
START ..\TRAK.EXE SWITCH08
```

Type

```
SW_RUN <Enter>
```

to generate all solutions without the need for further keyboard input.

14.4. Introduction to GCon

Microsoft has released over thirty versions of it's 32-bit operating system since Windows 95. There is considerable inconsistency in DOS emulation between versions. For example, Windows 2000 runs a set of programs in the proper sequential order for any valid batch format. Windows 95 starts all programs simultaneously unless you use the command form:

```
START /WAIT  ProgName
```

Windows ME does not recognize the `START` command at all. And so on.... To ensure consistent batch file operation we supply the utility program **GCon** with all our software. To avoid problems we advise running batch scripts from **GCon** rather than from the Command Prompt.

GCon handles the following subset of DOS commands. Additional capabilties are described in the file `GCON.HTML`.

CD
Function: Displays the name of or changes the current directory.
Example: `CD C:\AMaze\Buffer`

COPY ... TO ...
Function: Copies a file to another location.
Example: `COPY Diode.min TO C:\AMaze\Diode.min`

ERASE
Function: Deletes a file.
Example: `ERASE diode.mls`

REM
Function: Displays the line, no action taken

MOVE ... TO ...

Function: Moves a file from one directory to another directory.
Example: `MOVE Diode.mou TO C:\AMaze\Diode.mou`

RENAME

Function: Renames a file or files.
Example: `RENAME  diode.mou  diode_compare.mou`

RUN

Function: Runs a specified program or command.
Example: `RUN  C:\tricomp\mesh  diode`

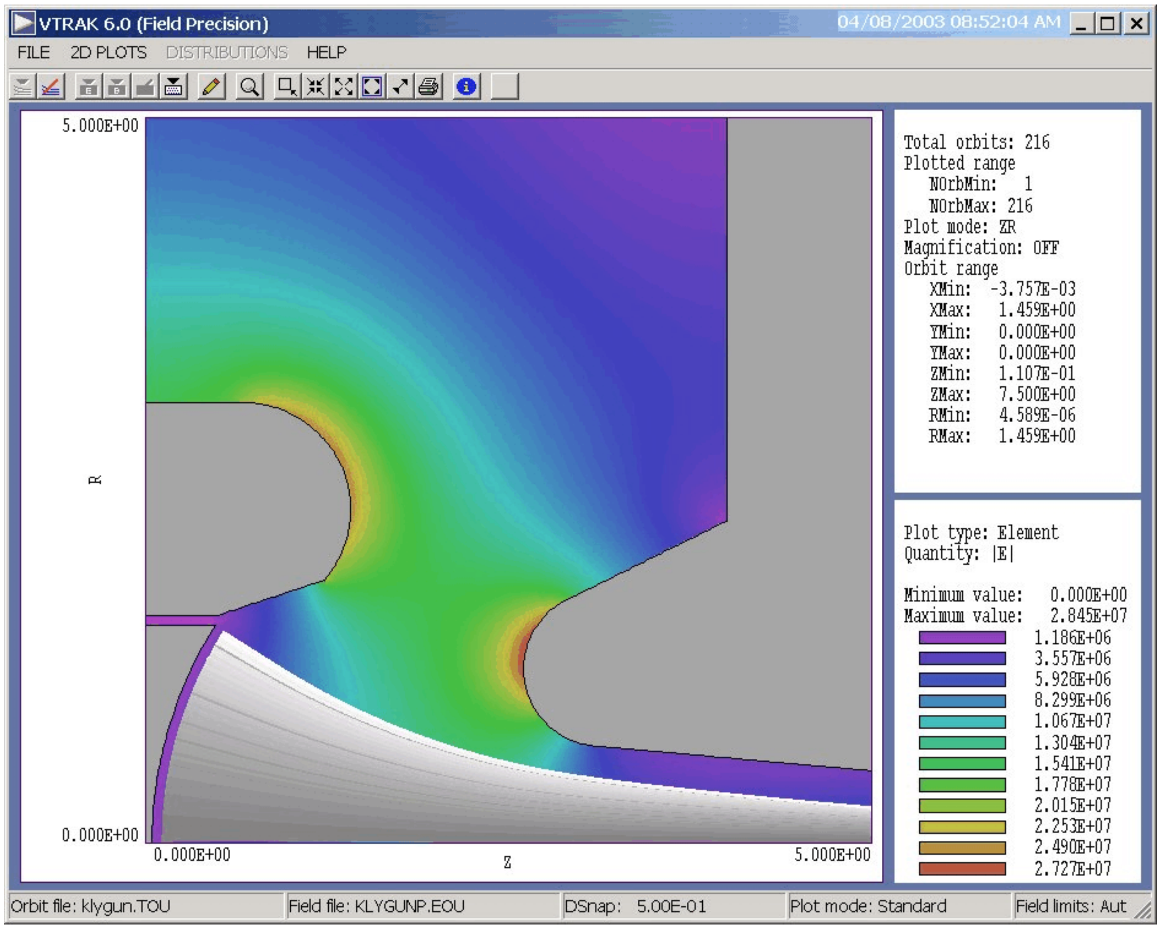
As an example, the following script creates a mesh and an electrostatic solution using the input files `DIODE.MIN` and `DIODE.EIN` in the directory `C:\EStudies\Inputs` and stores the results in the directory `C:\EStudies\Outputs`

```
REM Sample GCon script
CD C:\EStudies\Inputs
RUN C:\TRICOMP\MESH Diode
RUN C:\TRICOMP\ESTAT Diode
MOVE Diode.mou TO C:\EStudies\Outputs\Diode.mou
MOVE Diode.eou TO C:\EStudies\Outputs\Diode.eou
```

14.5. Running GCon

GCon is simple to run. Start the program and click on the command *Run GCon script*. The dialog shows all files with the suffix `GCN`. The working directory is changed if you change directories in the dialog. Pick a batch file and click *OK*. The program executes the commands in the file in sequence and displays the current status.

The *Stop on error* command toggles a switch that determines whether the program will stop the batch process is there is an error in one of the batch commands. The command *Instructions* in the *Help* menu displays a help file in your default browser. The function requires that the file `GCON.HTML` is in the same directory as `GCON.EXE`.



```
Total orbits: 216
Plotted range
   NOrbMin:   1
   NOrbMax: 216
Plot mode: ZR
Magnification: OFF
Orbit range
   XMin:  -3.757E-03
   XMax:   1.459E+00
   YMin:   0.000E+00
   YMax:   0.000E+00
   ZMin:   1.107E-01
   ZMax:   7.500E+00
   RMin:   4.589E-06
   RMax:   1.459E+00
```

```
Plot type: Element
Quantity: |E|

Minimum value:   0.000E+00
Maximum value:   2.845E+07
                1.186E+06
                3.557E+06
                5.928E+06
                8.299E+06
                1.067E+07
                1.304E+07
                1.541E+07
                1.778E+07
                2.015E+07
                2.253E+07
                2.490E+07
                2.727E+07
```

Orbit file: klygun.TOU | Field file: KLYGUNP.EOU | DSnap: 5.00E-01 | Plot mode: Standard | Field limits: Aut

Figure 15.1. **VTrak** screen shot – particle orbits, electrodes and |**E**|.

15. Creating orbit plots with VTRAK

VTrak creates plots of orbit and field quantities using field solutions (`FNAME.EOU`, `FNAME.BOU`, `FNAME.BBD`) and plot files (`RUNNAME.TOU`) created during **Trak** runs. The program also has interactive distribution analysis capabilities using information in standard particle files, `FNAME.PRT` see Chapter 16).

15.1. File menu commands

Load orbit file

The first step in an plotting session is to load orbit and/or field information. The dialog shows a list of files with names `FNAME.TOU`. Changing directories in the dialog changes the working directory. In the absence of a field file, **VTrak** guesses limits for the initial plot. Note that **VTrak** does not check that the currently-loaded orbit and field files match.

Close orbit file

Close the orbit file before loading a different one.

Load E field file

Load a file with a name of the form `FNAME.EOU`. The file could be direct output from **EStat** or one created by **Trak** using the *EDUMP* command.

Note 1: when a field plot is loaded the boundaries of the solution region are used for the default plot boundaries.

Note 2: if you have applied scaling or *SHIFT* operations to the electric field in the **Trak** run, the orbits will not match if you load the original applied field. You must create a file of the modified electric field using the *EDUMP* command for a valid superimposed plot.

Load applied Bfile

Load a file with a name of the form `FNAME.BOU` or `FNAME.POU` generated by **BStat**, **PerMag** or **Pulse**.

Load beam B file

Load a file with a name of the form `FNAME.BBD` generated by **Trak** in the *RELBEAM* mode in response to the *BBDUMP* command.

Close field file

Close the field file before loading a different one.

Orbit file information

Show information on the current orbit file.

Field file information

Show information on the current electric, applied magnetic or beam-generated magnetic field file.

15.2. 2D plot menu commands

Included orbits

Set a range of orbits to be plotted. Specify the starting and stopping orbit numbers as recorded in the `TOU` file. You can find information about the identity of particles created from emission surfaces in the `TLS` listing file. The default plot includes all orbits recorded in the plot file.

Set NSkip

When analyzing runs with large numbers of particles the plotted orbits may overlap so that details are obscured. Use this command to decrease the density of orbit plots. For example, if you set *NSkip* = 5, **VTrak** plots every fifth orbit. The default is *NSkip* = 1.

Toggle orbit shading

In simulations where particles carry current, particle weighting may vary considerably. For example, in a cylindrical geometry orbits near the envelope carry much more current than those near the axis. This command toggles the orbit shading option in **VTrak**. When active the color of particle orbits indicates their weight. In contour and element plots, low-current particles are plotted in light gray while high-current particles are plotted in black. In region and element plots the color changes from gray to white as current increases (Fig. 15.1).

Orbit plot type

Using the (x,y,z) coordinates recorded in the plot file, **VTrak** can make the following two-dimensional plots: y versus x, x versus z, y versus z and r versus z. These plots may or may not contain useful information, depending on the field symmetry.

Field plot style

The available styles are *Mesh, Region, Contour* and *Element*.

Field plot quantity

Use this command to set the plotted field quantity in contour and element plots. The available quantities in contour plots are electrostatic potential ϕ if an electric field file has been loaded, vector potential A_z or stream function rA_θ for an applied magnetic field, and toroidal field B_θ or enclosed current for a beam-generated magnetic field. Available quantities for element plots are ϕ or $|\mathbf{E}|$ (*E* file), vector potential or $|\mathbf{B}|$ (*B* file), and B_θ or enclosed current (*BB* file).

Field plot limits

You can manually set limits for contour and element field plots.

Number of contour lines

Manually set the number of lines for contour plots (the default is 24).

Toggle grid

VTrak can add a reference grid to any plot. The grid intervals are chosen to correspond to easily recognized numbers (*i.e.*, 0.01, 0.02, 0.05, ...). The interval is displayed next to the axis title when the grid is active.

Toggle element outline

You can display boundaries of elements in element plots. This feature may be useful to check whether the grid resolution is adequate for a **Trak** simulation.

Toggle mouse/keyboard

The setting determines whether coordinate input is from the mouse or keyboard for zoom and pan operations.

Toggle snap mode
Set snap distance

The snap mode is described in the **VBStat** and **VEStat** instruction manuals.

XY magnifications

By default, **VTrak** generates the same style of true-scale plots as the standard **TriComp** postprocessors (**VEStat**, **VBStat**, ...). The program has an additional plot mode that is useful for charged-particle beam simulations where the solution size in the transverse direction may be much smaller than the axial size. In the *Magnification* mode, you can manually specify the coordinates of the 2D plot boundaries. **VTrak** adjusts scales so that the plot fills the available plot area. Note that the zoom and pan commands are inactive in the *Magnification* mode.

Zoom window

Zoom in to a window in the *Standard* plot mode. Specify the edges of the plot area using either the mouse or keyboard.

Zoom in

Zoom in to the current center of the plot.

Expand view

Expand the view about the current center of the plot.

Global view

Show the full solution volume.

Pan view

Move the center of a zoomed view.

Hardcopy

Send the current plot to the *Default* Windows printer. Be sure to pick the correct device using *Settings/Printers* before initiating the plot.

Plot file (EPS)

Creates a file in the current directory with a name of the form `FPrefix.EPS` in encapsulated PostScript format.

Plot file (BMP)

Creates a file in the current directory with a name of the form `FPrefix.BMP` in Windows BitMap format.

Plot file (PNG)

Creates a file in the current directory with a name of the form `FPrefix.PNG` in Portable Network Graphics format (GIF equivalent).

Copy to clipboard

Copy the current plot to the clipboard in enhanced Windows Metafile format. You can then paste the illustration into a compatible graphics program like PhotoShop or PaintShop Pro.

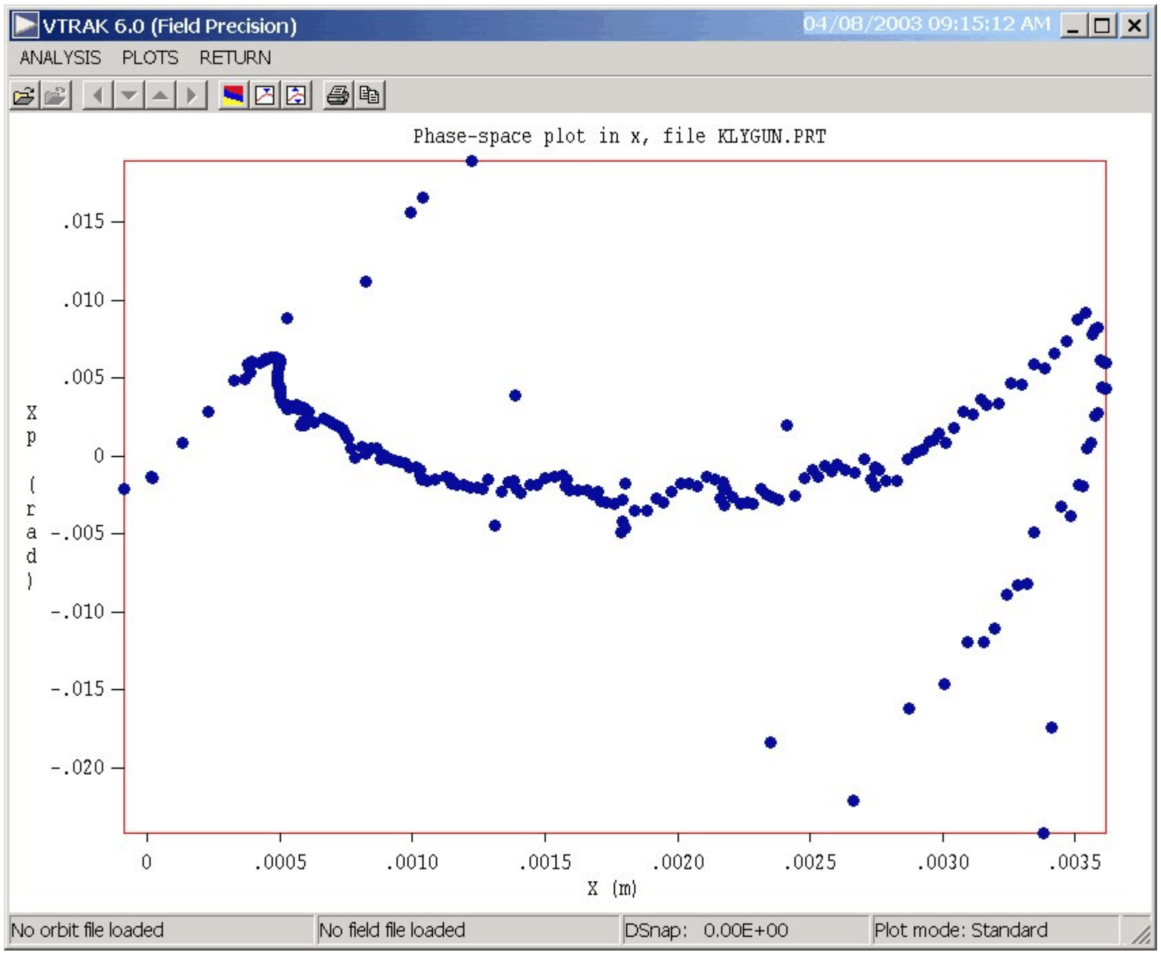


Figure 16.1. **VTrak** screen shot – distribution plot menu.

16. Distribution analyses in VTrak

The *Distribution* menu becomes active when you load a particle file (`FPrefix.PRT`) using the *Load particle file* command in the *File* menu. Two popup menus are displayed when you enter the *Distribution* menu: *Analysis* and *Plots*.

16. 1. Analysis commands

Analysis commands control calculation of numerical data from values in the particle file. It is important to note that particle positions in `PRT` files may be entered in any convenient units (*i.e.*, centimeters, inches, microns, ...). The values are converted to meters when the file is loaded into **Trak** according to the present value of the *DUnit* parameter. In turn, **Trak** uses the same conversion factor to write positions to output `PRT` files. The unit conversion factor must be known for RMS emittance calculations and phase-space plots in **VTrak.** Therefore, **Trak** 6.0, **OmniTrak** and **GenDist** 2.0 insert a comment line of the form

```
* DUnit:   1.00000E+02
```

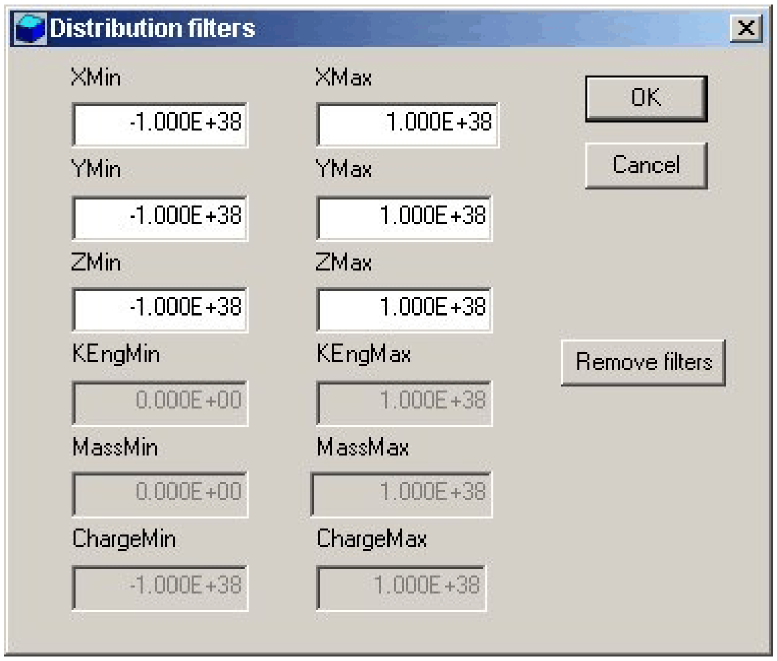
when creating a `PRT` file. **VTrak** checks for the comment when a file is loaded and uses the value of *DUnit* if the information is available. You can add a *DUnit* comment line to `PRT` files that you construct with a spreadsheet. Alternatively, you can set the value using the *Set DUnit* command in the *Analysis* menu. Note that the current value of *DUnit* is displayed in the status bar.

Beam analysis

In response to this command, **Trak** performs a statistical analysis of the particles and records several beam characterization quantities. The program uses current values of the reference axis and unit conversion factor. You can modify settings with the *Set reference axis* and *Set DUnit* commands. The program displays a summary of information on the screen and will record the complete analysis in a data file (see the *Open data file* command). Section 16.3 summarizes equations used in the calculations.

Table 16.1. Reference axis orientation

Reference axis	Normal plane Horizontal	Normal plane Vertical
X	*Y*	*Z*
Y	*Z*	*X*
Z	*X*	*Y*



XMin	XMax
-1.000E+38	1.000E+38
YMin	**YMax**
-1.000E+38	1.000E+38
ZMin	**ZMax**
-1.000E+38	1.000E+38
KEngMin	**KEngMax**
0.000E+00	1.000E+38
MassMin	**MassMax**
0.000E+00	1.000E+38
ChargeMin	**ChargeMax**
-1.000E+38	1.000E+38

Figure 16.2. Particle filter dialog.

Set particle filter

This command calls up the dialog of Fig. 16.2. In addition to position along the three axes, you can filter particles by kinetic energy, mass and charge. The last two options are useful only if the simulation contains multiple particle species.

Set reference axis

Beam quantities are calculated relative to a beam axis. The choices are the *X*, *Y* or *Z* axes of the solution coordinate system. The quantities *HOrig* and *VOrig* give the position of the beam axis in the horizontal and vertical directions of the normal plane. Table 16.1 lists the association of the horizontal and vertical directions with coordinate directions. The convention follows the right-hand and is applied in several of the distribution plots.

Set DUnit

Use this command to set a value for the conversion factor that will be used in the calculation of RMS emittance. The parameter *DUnit* gives the conversion between length units used in the PRT file to meters. For example, if positions in the PRT file are given in inches, set *DUnit* = 39.37.

Write PRT file

This command writes a PRT file that contains data lines only for particles that satisfy the current filter condition. As an application example, consider a model of an electron extractor where some particles strike the anode and others continue through an aperture. To analyze only the extracted electrons, you could set a filter on z_{min}. If you then created a PRT file with this command, it would contain only propagating electrons. The modified file could then be used as input for a simulation of the transport region.

Open data file

Trak will record the results of beam calculations in ASCII format if you open a data file. Supply a complete file name in the dialog. The program will write results of multiple analyses to the file until it is closed.

Close data file

Close the current data file.

16. 2. Plot commands

Trak can create presentation-quality plots based on the parameters in particle files. You can use this feature to display characteristics of beam distributions generated by **Trak** and to check the validity of input distributions created by **GenDist**. Note that only particles that pass the current filter are included in the plots.

The program supports the following plot types:

Spatial projections: particle positions projected to a normal plane. The options are *x*-*y*, *y*-*z* and *z*-*x*. For example, an *x*-*y* plot shows the position of a particle in the plane normal to *z* regardless of the particle position in *z*. You could set a filter to restrict plotted orbits to a range in *z*. The program aborts the plot and displays an error message if the range of values along either the horizontal or vertical axis equals zero.

Phase-space plots: phase-space distributions relative to the reference axis. The options are *x*-*x*’, *y*-*y*’ and *z*-*z*’. To illustrate the meaning of quantities, suppose the reference axis is *Z* and we choose a plot of *x*-

x'. The abscissa of the plot is ($x - x_{orig}$) expressed in meters. The ordinate is the quantity $x' = \tan^{-1}(p_x/p_z)$ expressed in radians. This plot type is useful only if particles have orbits that are paraxial with respect to the *z* axis (*i.e.*, the distribution constitutes a beam moving in ±*z*). A plot is not generated if you pick the option *z*-*z*' when the reference axis is set to *Z*.

1D bins: histograms of number of particles or particle current organized by a quantity. The following plot quantities are available for all PRT files: *f*(*x*), *f*(*y*), *f*(*z*) and *f*(*T*). For example, the option *f*(*T*) shows the distribution of computational particles in terms of kinetic energy. The option *f*(*x*) divides particles into bins according to their position in *x*, regardless their position in *y* and *z*. You can limit the range of particles in either x, y or z using a filter. The resulting histrogram shows the number of model particles in each bin along *x*. The following options are available only when the PRT file contains information on the model particle current: *Curr*(*x*), *Curr*(*y*), *Curr*(*z*) and *Curr*(*T*). In this case, the histogram shows the relative distribution of current in position or kinetic energy. The quality of the plot depends on the number of simulation particles and the number of bins. The height of a bin varies statistically approximately a fraction $1/N_b^{1/2}$, where N_b is the number of particles in the bin. You can change the number of bins using the *Set number of bins* command. You can also manually set limits on the abscissa using *the Set limits* command.

2D bins: histograms of the number of model particles or the current sorted according to position in a normal plane. The following options are available for all PRT files: *f*(*x,y*), *f*(*y,z*) and *f*(*z,x*). For example, in the *f*(*x,y*) option particles are assigned to a two-dimensional array of bins according to their positions in *x* and *y*, regardless of the position in *z*. You can use particle filters to restrict plotted particles to a region of space. The following options are available if the PRT file contains information on the particle current: *Curr*(*x,y*), *Curr*(*y,z*) and *Curr*(*z,x*). You can change the number of bins using the *Set number of bins* command. You can also manually set plot limits using *the Set limits* command.

3D scatter: three-dimensional plot of particle positions, (*x, y, z*). Only particles that pass the present filter are included. When this plot is active, you can shift the viewpoint using the *Rotation* and *Elevation* commands.

In addition to the standard hardcopy options, the following commands appear in the *Plots* popup menu of the *Distributions* menu.

Set plot type

Pick the plot type from the list described above. **VTrak** picks a default plot option valid for the current plot type.

Plot quantity

Pick a plotted quantity from one of the displayed options. **VTrak** displays only options that are valid for the current plot type.

Plot limits

Manually set limits on the plot. The number of quantities to supply depends on the plot type.

Set number of bins

Set the number of bins for 1D and 2D histograms.

Rotation (Up)
Rotation (Down)

These commands function only when the current plot type is *Scatter 3D*. Rotate the plot in azimuth.

Elevation (Up)
Elevation (Down)

These commands function only when the current plot type is *Scatter 3D*. Shift the view (polar angle)

16. 3. Beam analysis equations

The quantities used in beam analyses depend on the choice of reference axis. Table 16.1 lists the association between Cartesian coordinates and the horizontal (h) and vertical (v) directions. In the following equations, the subscript n refers to a model particle and the quantity n_p is the total number of particles. Particle n has position (h_n, v_n) relative to the reference axis and angles h_n' and v_n' given by

$$h_n' = \tan^{-1}\ (p_{hn}\ /p_{rn})\ , \qquad n_n' = \tan^{-1}\ (p_{vn}\ /p_{rn})\ .$$

In the equation, p_{hn} and p_{vn} are the dimensionless momenta in the horizontal and vertical directions while p_{rn} is the momentum along the reference axis. The first group of equations applies when the PRT file does not contain current information:

Average displacement from the reference axis

$$\overline{h} = \frac{\sum_{n=1}^{n_p} (h_n - h_{orig})}{n_p}\ , \qquad \overline{v} = \frac{\sum_{n=1}^{n_p} (v_n - v_{orig})}{n_p}\ .$$

Average angle with respect to the reference axis

$$\overline{h'} = \frac{\sum_{n=1}^{n_p} h_n'}{n_p}\ , \qquad \overline{v'} = \frac{\sum_{n=1}^{n_p} v_n'}{n_p}\ .$$

Average kinetic energy

$$\overline{T} = \frac{\sum_{n=1}^{n_p} T_n}{n_p}\ .$$

RMS displacements from the reference axis

$$h_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (h_n - h_{orig})^2}{n_p}}\ , \qquad v_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (v_n - v_{orig})^2}{n_p}}\ .$$

RMS angles with respect to the reference axis

$$h'_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (h'_n)^2}{n_p}} , \qquad v'_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (v'_n)^2}{n_p}} .$$

The RMS emittance in the horizontal direction is

$$\epsilon_h = 4 \sqrt{\overline{h^2}\ \overline{h'^2} - (\overline{h\ h'})^2} ,$$

where

$$\overline{h^2} = \frac{\sum_{n=1}^{n_p} (h_n - h_{orig})^2}{n_p} ,$$

$$\overline{h'^2} = \frac{\sum_{n=1}^{n_p} (h'_n)^2}{n_p} ,$$

$$\overline{h\ h'} = \frac{\sum_{n=1}^{n_p} (h_n\ h'_n)}{n_p} .$$

Similar equations hold for the vertical direction. The spread in kinetic energy is given by

$$\delta T = \sqrt{\frac{\sum_{n=1}^{n_p} (T_n - \overline{T}\ )^2}{n_p}}$$

VTrak takes current-weighted averages if the particle current is included in the PRT file. The total current equals:

$$I_o = \sum_{n=1}^{n_p} I_n \ .$$

Average displacement from the reference axis

$$\overline{h} = \frac{\sum_{n=1}^{n_p} (h_n - h_{orig})\ I_n}{I_o}\ , \qquad \overline{v} = \frac{\sum_{n=1}^{n_p} (v_n - v_{orig})\ I_n}{I_o}\ .$$

Average angle with respect to the reference axis

$$\overline{h'} = \frac{\sum_{n=1}^{n_p} h'_n\ I_n}{I_o}\ , \qquad \overline{v'} = \frac{\sum_{n=1}^{n_p} v'_n\ I_n}{I_o}\ .$$

Average kinetic energy

$$\overline{T} = \frac{\sum_{n=1}^{n_p} T_n\ I_n}{I_o}\ .$$

RMS displacements from the reference axis

$$h_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (h_n - h_{orig})^2\ I_n}{I_o}}\ , \qquad v_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (v_n - v_{orig})^2\ I_n}{I_o}}\ .$$

RMS angles with respect to the reference axis

$$h'_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (h'_n)^2 \; I_n}{I_o}} \;, \qquad v'_{rms} = \sqrt{\frac{\sum_{n=1}^{n_p} (v'_n)^2 \; I_n}{I_o}} \;.$$

The averages for computing the RMS emittance in the horizontal direction are

$$\overline{h^2} = \frac{\sum_{n=1}^{n_p} (h_n - h_{orig})^2 \; I_n}{I_o} \;,$$

$$\overline{h'^2} = \frac{\sum_{n=1}^{n_p} (h'_n)^2 \; I_n}{I_o} \;,$$

$$\overline{h\; h'} = \frac{\sum_{n=1}^{n_p} (h_n \; h'_n) \; I_n}{I_o} \;.$$

Spread in kinetic energy

$$\delta T = \sqrt{\frac{\sum_{n=1}^{n_p} (T_n - \overline{T}\,)^2 \; I_n}{I_o}}$$

Appendix 1
Control script syntax changes from Version 5.0

It was necessary to make some syntax changes in the control script of **Trak** 6.0 compared to Version 5.0 in order to support new features and to simplify the language. Input files prepared for versions previous to 6.0 may generate syntax errors. In order to facilitate the update process we have added an automatic syntax-checking capability to **Trak** 6.0. To check a `TIN` file, click on *Update TIN file syntax* in the *File* menu. The program adds error marks and suggested corrections to the file. The original file is saved as `FILENAME.BAK`. The following table summarizes changes.

Syntax changes in Trak 6.0

Fields section

Command	**Condition**	**Correction**
EFILE	String MOD in position 3 no longer recognized	Use MODFUNC E for electric field modulations
ESHIFT	Eliminated	Use SHIFT E
EMULT	Eliminated	Include scaling information in EFILE command
EMOD	Eliminated	Use MODFUNC E command
BFILE	String MOD in position 3 no longer recognized	Use MODFUNC B for magnetic field modulations
BSHIFT	Eliminated	Use SHIFT B
BMULT	Eliminated	Include scaling information in BFILE command
BMOD	Eliminated	Use MODFUNC B command
BTABLE	String MOD in position 3 no longer recognized	Use MODFUNC B for magnetic field modulations
BUNI	String MOD in position 3 no longer recognized	Use MODFUNC B for magnetic field modulations

Syntax changes in Trak 6.0		
BTHETA	String MOD in position 3 no longer recognized	Use MODFUNC B for magnetic field modulations
BBCALC	Eliminated	Use RELBEAM tracking mode
BBRELMODE	Eliminated	Use RELMODE command in PARTICLES section
BBMESH	Eliminated	Use RELBEAM tracking mode
Particles section		
Command	**Condition**	**Correction**
SCEMIT	Track option eliminated	Use SCHARGE or RELBEAM modes
INTERP	Expanded function	Include E, B or BB in position 2
LISTON	Modifed syntax	Include P, E or B in position 3 to control the listing information
TRACKMODE	Eliminated	Interpolation type maintained through all cycles
RESTMASS	Eliminated	Information contained in EMIT command
CHARGE	Eliminated	Information contained in EMIT command
NPART	Eliminated	Number of particles set by list and NPerSeg in EPARAM command
MARKREG	Eliminated	Information contained in EMIT command
JLIMIT	Eliminated	Information contained in EMIT command
DEMIT	Eliminated	Information contained in EMIT command
DTHETA	Eliminated	Information contained in EMIT command
WORKFUNC	Eliminated	Information contained in FEMIT command

Syntax changes in Trak 6.0		
EMISSION	Eliminated	Information now entered in the SECONDARY command
Diagnostics section		
Command	**Condition**	**Correction**
BBCDENS	Eliminated	Use distribution capabilities of VTrak or the *BBBOUNDARY* command

The following is an example of an annotated file produce by the Update TIN file syntax command:

```
* File KLYGUN.TIN
*
FIELDS
    EFile = KLYGUN.EOU
    DUnit = 39.37
    BBCalc
* >>> The BBCALC command is no longer valid. Use the RELBEAM mode in the
*     PARTICLES section
* Particle counting to z = 0.5
    BBMesh = (0.500, 7.48, 40, 0.00, 1.60, 32)
* >>> The BBMESH command is no longer valid. Use the RELBEAM mode in the
*     PARTICLES section
* Relativistic mode in the transport tube
    BBRelMode: (4.00 7.50)
* >>> The BBRELMODE command is no longer valid. Use the RELMODE command in the
*     RELBEAM mode of the ARTICLES section
END
*
PARTICLES SCEMIT
* >>> The SCEMIT mode in the PARTICLES section has been eliminated. Use the
*     SCHARGE or RELBEAM modes instead
   NCycle = 20
   NPart = 100
* >>> The NPART command has been eliminated. The number of particles is determined
*     by the particle list, the number of facets in emission surfaces, and the
*     parameter NPERSEG in the EPARAM command
   NSearch = 3
* >>> The EMISSION command has been eliminated. The secondary emission parameters
*     are now specified in the SECONDARY command
   RestMass = 0.0
* >>> The RESTMASS command has been eliminated. The mass of particles from
*     an emission surface is now specified in the EMIT command
   Avg = 0.20
   Charge = -1.0
* >>> The CHARGE command has been eliminated. The charge of particles from
*     an emission surface is now specified in the EMIT command
   MarkReg = 6
* >>> The MARKREG command has been eliminated. Its function is now performed by
*     the EMIT and FEMIT commands
   DEmit(6) = 0.070
* >>> The DEMIT command has been eliminated. The emission distance is now
*     specified in the EMIT command
   TrackMode = 8
* >>> The TRACKMODE command has been eliminated.
   Dt = 1.0E-12
END
*
DIAGNOSTICS
   EDump = KLYGUNP
   BBDump = KLYGUN
   INTERPBB LSQ
   BBCDens: 1.00
* >>> The BBCDENS command has been eliminated. Use the new distribution
*     capabilities in VTRAK
   PartList
END
ENDFILE
```

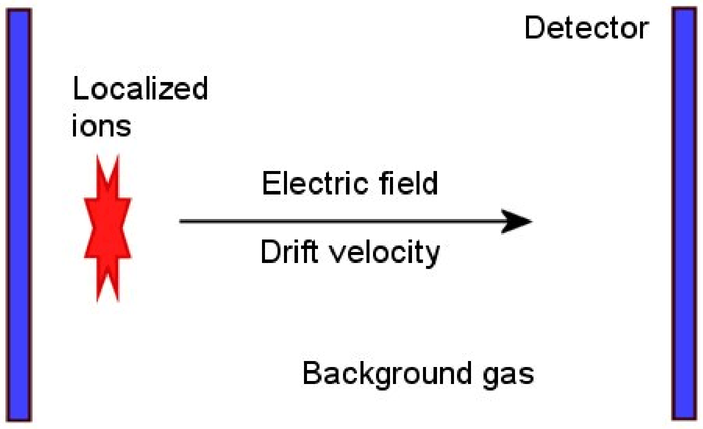


Figure A.2.1. Principle of the ion-mobility time-of-flight spectrometer.

Appendix 2. Features for ion mobility spectrometry

Trak includes useful features for the design of time-of-flight ion-mobility spectrometers. Figure A.2.1 shows the basic principle. The material to be analyzed is ionized in a gas background (typically air at atmospheric pressure). A electric field is applied and the ions drift with velocity vector:

$$\boldsymbol{v} = \mu\ \boldsymbol{E},$$

where **E** is the electric field vector (in V/m) and μ is the mobility (in m^2/V-s). Species with different mobilities separate in transit. With a known electric field distribution, the time-of-flight to a detector can be used to infer μ.

For reference, the absolute value of mobility μ is related to the reduced mobility μ_o (defined at standard temperature and pressure) by:

$$\mu = \mu_0 \left(\frac{760}{P} \right) \left(\frac{T + 273.15}{273.15} \right) .$$

In the equation *P* is the pressure in mm of mercury and *T* is the temperature in °C.

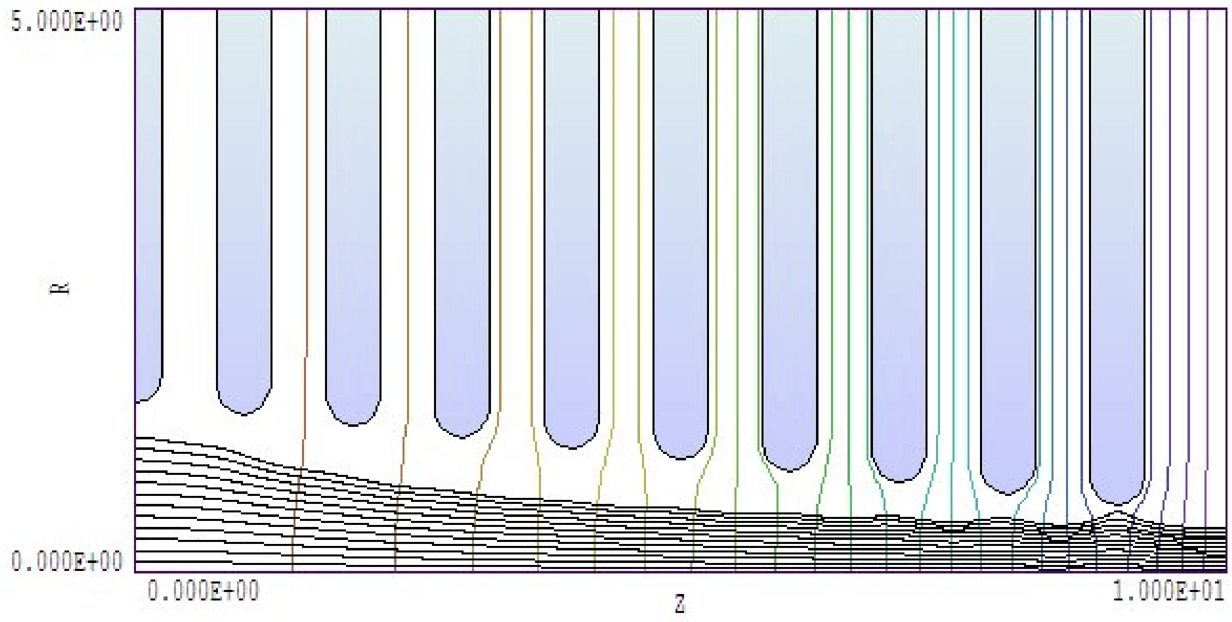


Figure A.2.2. Electrode geometry, equipotential lines and ion drift orbits for example `IMS_DEMO01`.

Because the drifting ions follow lines of **E**, the electric-field tracking capabilities of **Trak** (Chapter 7) are ideal for this application. The program can calculate and plot precise ion trajectories in complex two-dimensional geometries. **Trak** can also compute the ion time-of-flight for a given mobility. The time increment Δt for an ion to move a distance Δs is

$$\Delta t = \frac{\Delta s}{|\boldsymbol{v}|} = \frac{\Delta s}{\mu\ |\boldsymbol{E}|} \ .$$

The time-of-flight from point A to point B is therefore

$$t_{AB} = \int_A^B \frac{ds}{\mu\ |\boldsymbol{E}|} \ .$$

While tracing a field line **Trak** computes the following integral to yield the *normalized transit time*:

$$\tau_{AB} = \mu \; t_{AB} = \int_A^B \frac{ds}{|\boldsymbol{E}|} \; .$$

The quantity τ (in m^2/V) depends only on the nature of the electric field solution. The properties of the background gas and ions are contained in μ. Therefore, a knowledge of the normalized transit time distribution completely characterizes the performance of a detector for any fill gas or ion species.

Table A.2.1. File IMS_DEMO01.TIN

```
FIELDS
  DUnit = 100.0
  EFile = IMS_DEMOP.EOU
END
PARTICLES FLINE
  FList
*     X     Y     Z
*   =================
    0.10  0.00  0.001
    0.20  0.00  0.001
    0.30  0.00  0.001
    0.40  0.00  0.001
    0.50  0.00  0.001
    0.60  0.00  0.001
    0.70  0.00  0.001
    0.80  0.00  0.001
    0.90  0.00  0.001
    1.00  0.00  0.001
    1.10  0.00  0.001
    1.20  0.00  0.001
  End
  Ds = 0.010
END
DIAGNOSTICS
  PARTLIST
END
ENDFILE
```

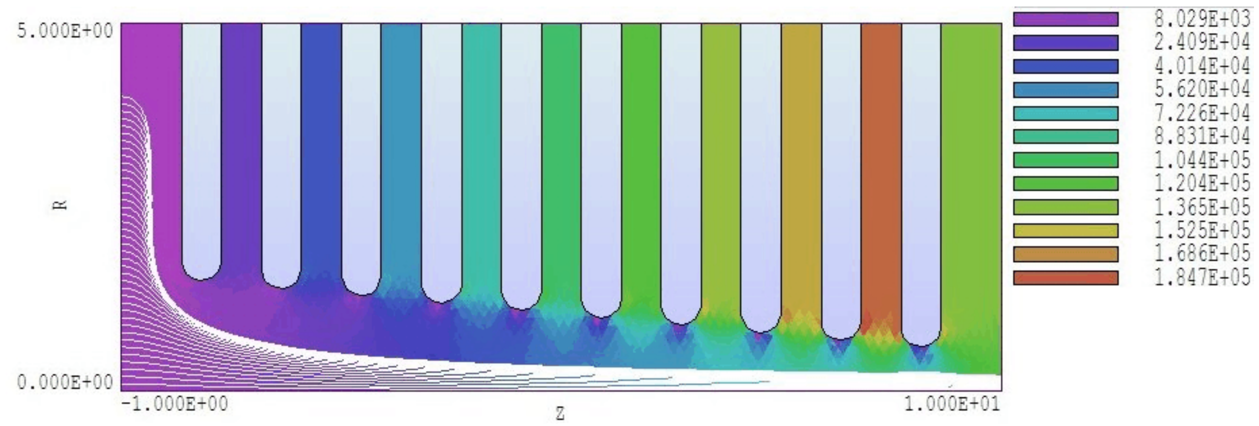


Figure A.2.3. Electrode geometry and ion orbits for example `IMS_DEMO02`. Background color-coding by |**E**|.

To illustrate the procedure consider the geometry of Figure A.2.2. An entrance grid, a set of biased rings and a detector plane create a field that increases in the *z* direction. The goal is to capture ions and compress the distribution to a small-diameter detector. The file `IMS_DEMO01.MIN` describes the geometry and `IMS_DEMO01.EIN` sets the electrode voltages. The **Trak** run is controlled by the file `IMS_DEMO01.TIN` listed in Table A.2.1. Figure A.2.2 also shows equipotential lines and ion drift orbits (corresponding to electric field lines) with entrance radii in the range 0.0 cm $\leq r \leq$ 1.2 cm. Ions with larger radii strike the final ring; therefore, the acceptance could be increased by reducing the thickness of the ring. An orbit with an initial radius of 1.2 cm strike the detector at 0.393 cm, a 9.3:1 area compression. In response to the *PARTLIST* command **Trak** records a statistical analysis of normalized transit time and field-line length in the listing file (Table A.2.2) The average normalized transit time is $\tau_{avg} = 2.68 \times 10^{-6}$ m^2/V. If the ion mobility is $\mu = 1.36 \times 10^{-4}$ m^2/V-s, the predicted average transit time is 19.7 ms. The dispersion in normalized transit time is $\Delta\tau = 6.03 \times 10^{-8}$ m^2/V. Therefore, we expect that the field-magnitude and trajectory variations limit the detector resolution to about $R \geq \Delta\tau/\tau avg$ = 2.25%.

In a second run (`IMS_DEMO02`), we move the entrance grid upstream 1.0 cm from the first ring (Fig. A.2.3). Electric fields have small magnitude and strong curvature in the space between the grid and first ring. As a result, electric field lines from a broad area on the entrance grid are funneled into the acceleration structure. The geometry therefore has very good capture efficiency. On the other hand ion orbits have a huge range of normalized transit time τ (Fig. A.2.4). The reason is is the ions spend

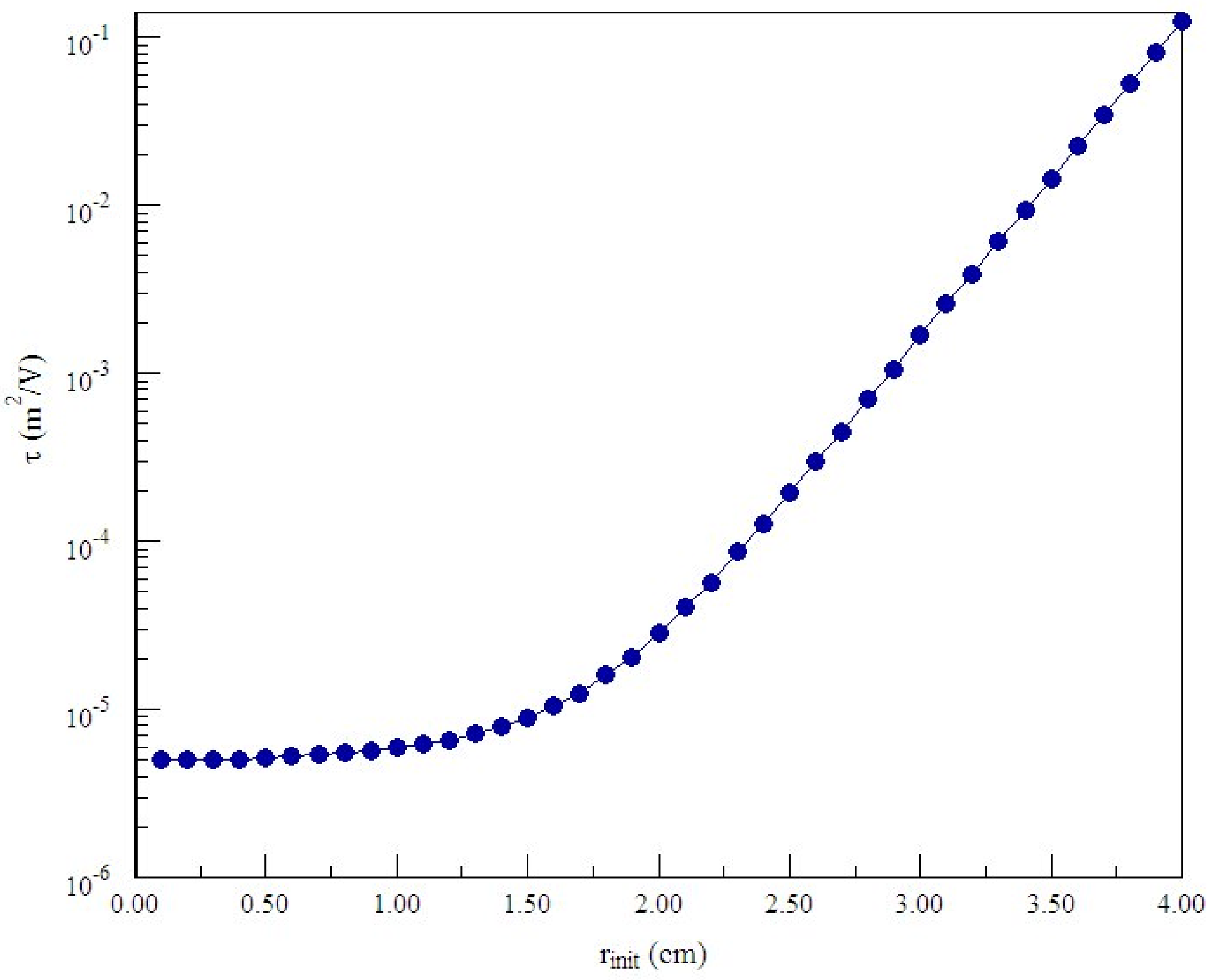


Figure A.2.4. Normalized ion transit time as a function of entrance radius for example `IMS_DEMO02`.

most of the time in the low-field region where there are strong variations in field line length and |**E**|. Therefore, the configuration is not useful for time-of-flight spectrometry.

Table 22.2. Statistical analysis listing, MOBILITY_DEMO01

```
Mobility statistics
Number of values:   12
Normalized transit time (Tau = Int(ds/|E|)
  Tau (average):   2.67762E-06 (m2/V)
  Tau (stddev):   6.02967E-08 (m2/V)
  Tau (minimum):   2.61802E-06 (m2/V)
  Tau (maximum):   2.80662E-06 (m2/V)
Field line length
  D (average):   1.00287E-01 (m)
  D (stddev):   3.18367E-04 (m)
  D (minimum):   9.99953E-02 (m)
  D (maximum):   1.01023E-01 (m)
```