# SIMULATING ELECTRON IMPACT IONIZATION USING A GENERAL PARTICLE TRACER (GPT) CUSTOM ELEMENT[*]

J. T. Yoskowitz[†], G. A. Krafft, Old Dominion University, Norfolk, Virginia, 23529, USA
S. B. van der Geer, Pulsar Physics, Burghstraat 47, 5614 BC Eindhoven, The Netherlands
J. Grames, Thomas Jefferson National Accelerator Facility, Newport News, Virginia 23606, USA
R. Montoya Soto, Departamento de Física, Universidad de Guanajuato, Leon, 37150 Mexico
C. A. Valerio Lizarraga
Facultad de Ciencias Físico-matemáticas Universidad Autónoma de Sinaloa, 80010 Mexico

### *Abstract*

A new C++ custom element has been developed with the framework of General Particle Tracer (GPT) to simulate electron impact ionization of residual gas molecules. The custom element uses Monte-Carlo routines to determine both the ion production rate and the secondary electron kinetic energy based on user-defined gas densities and theoretical values for the ionization cross section and the secondary electron differential cross section. It then uses relativistic kinematics to track the secondary electron, the scattered electron, and the newly formed ion after ionization. The ion production rate and the secondary electron energy distribution determined by the custom element have been benchmarked against theoretical calculations and against simulations made using the simulation package IBSimu. While the custom element was originally built for particle accelerator simulations, it is readily extensible to other applications. The custom element will be described in detail and examples of applications at the Thomas Jefferson National Accelerator Facility will be presented for ion production in a DC high voltage photo-gun.

## INTRODUCTION

There is a need for detailed simulation modelling of electron impact ionization in electron accelerators. One application in particular is to model ionization in DC high voltage photo-guns in order to better understand the ion back-bombardment mechanism and predict the effectiveness of ion mitigation techniques such as biasing the anode [1, 2]. Particle tracking codes such as IBSimu [3, 4] and PARMELA [5, 6] have been used to model ion back-bombardment. However, the typical output of these codes are plots of particle trajectories, in which it is difficult to discern how individual particles interact with each other in real-time. On the other hand, the simulation package GPT allows the user to create simulations in which particles are individually tracked over time in user-defined step sizes [7, 8]. An important feature of GPT is its extensibility: the functionality of GPT can be extended with the creation and implementation of C++ custom elements. In this work, we present a new custom element written to make simulations of electron impact ionization, a feature not currently supported in GPT. While other GPT custom elements have been made for the simulation of electron impact ionization [9], the defining features of this custom element are: (1) the production of ions by primary electrons in real-time via Monte Carlo routines, (2) the production and tracking of secondary and scattered electrons, (3) the ability to select the ionized gas species from a selection of typical residual gas molecules, such as $H_2$, CO, and $CH_4$, as well as the orbital from which the secondary electron is ejected, (4) the choice of uniform gas density or user-defined density distribution, and (5) the option to have the ionization parameters for each ionization to be included with the simulation output. The functionality and benchmarking of the custom element will be described below. Its application in simulations of ion production and mitigation at the CEBAF photo-gun at Jefferson Lab are presented in a separate presentation at this conference [10].

## CUSTOM ELEMENT DESCRIPTION

### *Ion Production*

In a typical GPT simulation, a number of electron macro-particles, each representing some number of electrons, are set along with their initial spatial and momentum distributions. The electron macro-particles are then tracked over time steps chosen by a fifth-order Runge-Kutta algorithm with adaptive stepsize control to maintain the user-defined accuracy while minimizing computation time. During each time step, the ionization custom element loops over all particles present in the simulation. A particle is eligible for ionization provided: the particle is an electron or electron macro-particle, the particle is within the user-defined ionization region for the duration of the time step, and the kinetic energy of the particle is above the ionization energy of the gas species.

Provided the particle is eligible for ionization, the custom element calculates the number of ions produced in the time step using the formula:

$$N_{ion} = \rho \sigma d N_e \tag{1}$$

where $\rho$ is the gas density (ions/m$^3$), $\sigma$ is the ionization cross section (m$^2$), $d$ is the distance the particle travelled in the time step (m), and $N_e$ is the number of electrons the electron macro-particle represents. The custom element uses the Reiser form of ionization cross section formula, originally derived by Bethe [11, 12]. Figure 1 shows the ionization

[*] Work supported by the U.S. DOE under contract DE-AC05-06OR23177

[†] yoskowij@jlab.org

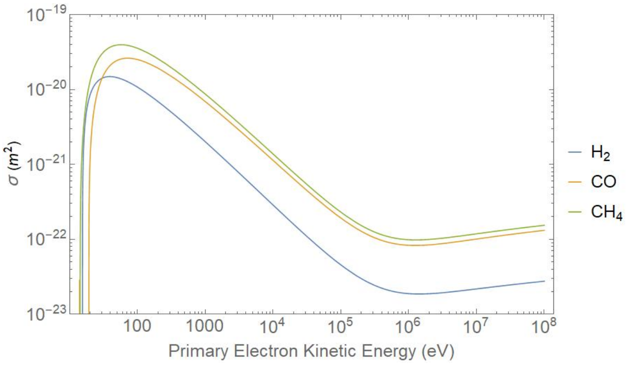

Figure 1: Log-log plot of the ionization cross section for $H_2$, CO, and $CH_4$.

cross section as a function of primary electron energy for $H_2$, CO, and $CH_4$ gases.

The timestep is retried with a smaller stepsize in the unlikely case that $N_{ion}$ is larger than unity for any of the particles. $N_{ion}$ can then be interpreted as the probability that the particle will ionize during the time step. A Monte-Carlo approach is then taken: if $N_{ion}$ is less than a random number between 0 and 1, then the particle ionizes.

### *Secondary and Scattered Electron Production*

For every ion produced in the simulation, a secondary electron is created. The probability distribution of secondary electron energies is given by the differential cross section (DCS) in the binary-encounter-dipole model (see Eq. 44 in [13]). The custom element assigns an energy using a Monte-Carlo algorithm similar to the algorithm used for ionization probabilities above. Let $f(E,T)$ be the integral of the DCS from 0 to $E$ for a given primary electron energy $T$. We can define $g(E,T) = f(E,T)/f(E_{max},T)$ as the ratio between the partial integral and the total integral of the DCS. Note that $g(E,T)$ is strictly between 0 and 1. A random number between 0 and 1 is chosen and is compared to values of $g(E,T)$ for increasing E. The secondary electron energy is assigned the value $E$ corresponding to the value of $g(E,T)$ closest to the random number.

After ionization, the primary electron scatters away. To represent this in simulation, the number of electrons the primary electron macro-particle represents is reduced by one and an electron representing the scattered electron is produced with an energy based on energy conservation. The gas molecule's energy pre-ionization is assumed to be the average kinetic energy at room temperature. The energy of the resulting ion is based on a Maxwellian distribution of velocities, with the most probable velocity corresponding to a kinetic energy of 4 eV, and is determined by the custom element using a Monte-Carlo algorithm similar to the algorithm for secondary electron energies.

## BENCHMARKING AGAINST THEORY

Each section of the ionization custom element was benchmarked against theoretical equations described in the previous section in order to ensure accuracy. To benchmark the ion production rate, consider the following GPT simulation: an electron bunch with a 0.5 mm rms transverse bunch size, 50 ps rms bunch length, travelling along the $z$-axis between $z = 0$ m and $z = 0.5$ m with an initial kinetic energy of 1 keV. The electron bunch has 10 μC of charge distributed uniformly over $10^4$ macro-particles. Space charge forces are neglected. As the bunch travels, it ionizes $H_2$ gas, which has a constant pressure of $10^{-12}$ torr ($\rho \approx 3 \times 10^{10}\,\mathrm{m}^{-3}$) throughout the simulation. Figure 2 shows a snapshot of the GPT simulation when the bunch is at $z \approx 0.25$ m.

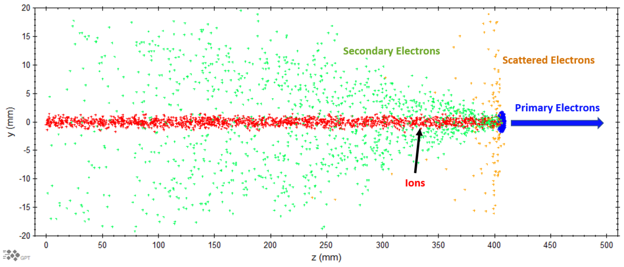

Figure 2: Side view snapshot of the example GPT simulation.

The simulation was repeated for CO and $CH_4$, assuming the same gas density. For each gas species, the simulation was rerun 50 times with different seed values for the random number generators used in the Monte-Carlo algorithms. Table 1 lists the results of simulations and compares the expected total number of ions produced based on Eq. (1) with the average number of ions produced in simulations. The number of ions produced in the simulations agree with the theoretical number of ions to within 1.5%.

Table 1: GPT Simulation Results

| Gas Species | $H_2$ | CO | $CH_4$ |
|---|---|---|---|
| I (eV) | 15.4 | 14.0 | 14.3 |
| $\sigma$ ($10^{-21}$ m$^2$) | 2.0 | 7.0 | 8.7 |
| Pred. # Ions | 2028 | 6971 | 8770 |
| Avg. Sim. # Ions | 2002 | 6864 | 8638 |
| % Diff. | 1.3 | 1.5 | 1.5 |

To benchmark the secondary electron energy distribution, histograms of the secondary electron energies from all simulations for each gas species were created and compared with the DCS. As an example, Fig. 3 shows the histogram of secondary electron energies for the $CH_4$ simulations. To compare the energy distribution with the DCS, the histogram was normalized such that the bin heights represent the probability distribution of secondary electron energies. The distribution of secondary energies is in close agreement with the normalized DCS.

Finally, to benchmark the ion energy distribution, the normalized histograms of ion speeds were compared to the Maxwellian distribution curve. As an example, Fig. 4 com-

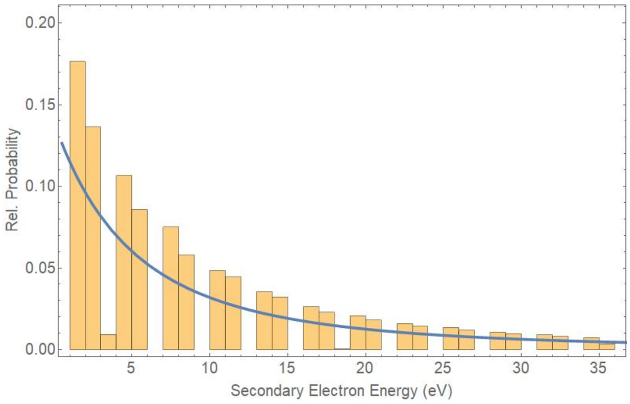

Figure 3: Probability distribution histogram of secondary electron energies from all 50 $CH_4$ simulations with an overlay of the normalized DCS.

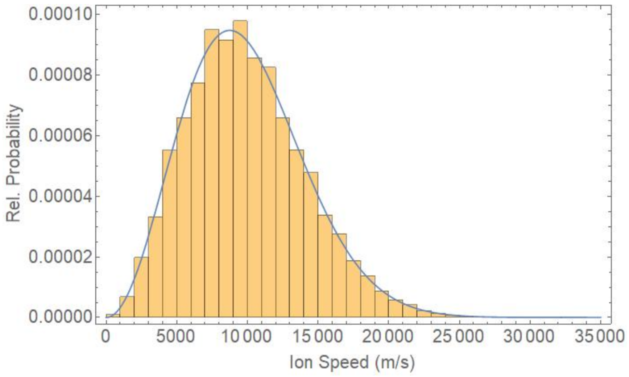

Figure 4: Probability distribution histogram of ion speeds from all 50 $CH_4$ simulations with an overlay of the Maxwellian distribution curve.

pares the normalized histogram of $CH_4{}^+$ ion energies with the Maxwellian distribution curve, with excellent agreement.

## BENCHMARKING AGAINST IBSIMU

To benchmark the custom element against the simulation code IBSimu, the following simulation was performed in both codes: an electron bunch containing $10^4$ electrons with constant, uniform kinetic energy $T$ travels through 1 m of $H_2$ gas. The gas density was tuned such that the electron bunch produces 100 $H_2{}^+$ ions on average. Using Eq. (1), $\rho\,(1\,\text{keV}) = 4.97 \times 10^{18}\,\text{m}^{-3}$ and $\rho\,(130\,\text{keV}) = 2.60 \times 10^{20}\,\text{m}^{-3}$. The simulation was rerun in both codes 20 times with different seed values for $T = 1$ keV and 20 times for $T = 130$ keV. Table 2 compares the simulation results of GPT and IBSimu. The average number of ions produced in both codes for both energies agree with the expected number of ions produced to within ~1%.

## CONCLUSIONS

The C++ custom element has been successfully developed to create GPT simulations of ion production within electron accelerators. The Monte Carlo algorithms ensure the ion and secondary electron production are probabilistic instead of deterministic. The ion production rate, secondary electron energy distribution and the ion speed distribution are all in agreement with theoretical equations. The results of the comparison simulations performed in both GPT and IBSimu showed remarkable agreement with theory and between both codes. The custom element code, along with full documentation, will be available to the GPT community in the near future.

Table 2: Simulation Results for GPT and IBSimu. The percent difference values compare the average number of ions produced to the predicted 100 ions.

| Simulation Code | GPT | IBSimu |
|---|---|---|
| Avg. Sim. Ions (1 keV) | 98.9 | 101 |
| % Diff (1 keV) | 1.1 | 1.0 |
| Avg. Sim. Ions (130 keV) | 99.95 | 100.8 |
| % Diff (130 keV) | 0.05 | 0.8 |

## REFERENCES

[1] E. Pozdeyev, “Ion trapping and cathode bombardment by trapped ions in DC photoguns”, *Phys. Rev. ST Accel. Beams*, vol. 10, no. 8, p. 083501, 2007. doi:10.1103/PhysRevSTAB.10.083501

[2] J. Grames *et al.*, “A biased anode to suppress ion back-bombardment in a DC high voltage photoelectron gun”, *AIP Conf. Proc*, vol. 980, no. 10, p. 110, 2008. doi:10.1063/1.2888075

[3] Ion Beam Simulator (IBSIMU), http://ibsimu.sourceforge.net

[4] C. Hernandez-Garcia *et al.*, “Compact -300 kV dc inverted insulator photogun with biased anode and alkali-antimonide photocathode”, *Phys. Rev. Accel. Beams*, vol. 22, no. 11, p. 113401, 2019. doi:10.1103/PhysRevAccelBeams.22.113401

[5] L. Young and J. Billen, “The particle tracking code PARMELA”, in *Proc. PAC 2003*, Los Alamos, NM, USA doi:10.1109/PAC.2003.1289968

[6] J. Lewellen, “Ion tracking in photocathode rf guns”, *Phys. Rev. ST Accel. Beams*, vol. 5, no. 2, p.020101, 2002. doi:10.1103/PhysRevSTAB.5.020101

[7] General Particle Tracer (GPT), http://www.pulsar.nl/gpt

[8] J. T. Yoskowitz *et al.*, “New simulations for ion-production and back-bombardment in GaAs photo-guns”, in *PoS PSTP2019*, vol. 279, p.040, 2020. doi:10.22323/1.379.0040

[9] J. Biswas and E. Wang, “New dynamic ionizer element to simulate ion back bombardment in DC gun”, Zenodo, 2018. doi:10.5281/zenodo.1413652

[10] J. T. Yoskowitz, “Improving the operational lifetime of the CEBAF photo-gun by anode biasing”, presented at IPAC ’21, May 2021, this conference.

[11] H. Bethe, "Zur Theorie des Durchgangs schneller Korpuskularstrahlen durch Materie" (Theory of the Passage of Fast Corpuscular Rays Through Matter), *Annalen der Physik*, vol. 397, no. 3, pp. 325-400, 1930. doi:10.1002/andp.19303970303

[12] M. Reiser, *Theory and Design of Charged Particle Beams*. Wiley VCH Verlag GmbH, 2008.

[13] Yong-Ki Kim and M. Eugene Rudd, "Binary-encounter-dipole model for electron-impact ionization", *Phys. Rev. A*, vol. 50, no. 5, pp. 3954-3967, 1994. 10.1103/PhysRevA.50.3954