3D Simulations and Analysis of Pulsed Discharges

PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus, prof.dr.ir. F.P.T. Baaijens, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op donderdag 12 november 2015 om 16.00 uur

door

Herman Jan Teunissen

geboren te Amsterdam

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	prof.dr. H.J.H. Clercx
1^e promotor:	prof.dr. U.M. Ebert
copromotor:	dr.ir. S. Nijdam
leden:	prof.dr.ir. B. Koren
	prof.dr. M.J. Kushner (University of Michigan)
	Priv.Doz. Dr.-Ing. habil. T. Mussenbrock (Ruhr-Universität Bochum)
	prof.dr. M.M. Turner (Dublin City University)
adviseur:	dr. A. Luque Estepa (Instituto de Astrofísica de Andalucía)

Het onderzoek of ontwerp dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

This research is supported by the Dutch Technology Foundation STW, which is part of the Netherlands Organisation for Scientific Research (NWO) and partly funded by the Ministry of Economic Affairs (project number 10755).

This thesis was typeset using LaTeX. Front cover: Zoom of the Mandelbrot set, centered at $(0.485, 1.18i)$. The mesh was generated with Afivo (chapter 10). Back cover: Plasma globe (credit: Colin, Wikimedia Commons, CC BY-SA 3.0) inside an iris (credit: Nick Fedele, CC BY-NC-SA 2.0), both modified.

Contents

Chapter 1

Thesis goals & contributions

In this short chapter, the main goals and contributions of this thesis are discussed. The thesis focuses on the simulation and analysis of pulsed discharges, specifically *streamers*. Streamers are rapidly growing ionized channels, that enhance the electric field at their tips. Due to this field enhancement, they can penetrate into regions where the electric field is below the breakdown threshold. Streamers occur in nature, where they for example pave the path for lightning leaders, or they appear as sprite discharges high above thunderclouds [1–3]. They are also used in diverse industrial applications [4–8]. A general introduction to discharges, streamers and their numerical modeling can be found in chapter 2.

Two types of simulation models have commonly been used for streamer discharges: particle models [9–11] and fluid models [12–19]. In chapter 3, a comparison study is presented, using four models for the three-dimensional simulation of a short negative streamer: a particle-in-cell model, two plasma fluid models [20] and a *hybrid* model [21, 22], in which the particle-in-cell code was spatially coupled to one of the fluid models. Streamers have a tendency to branch [19, 23–28], and their branching is accelerated by stochastic density fluctuations. An advantage of particle models is that such fluctuations can be included, but a drawback is that they are computationally significantly more expensive. Both the particle and the hybrid model eventually use *super-particles* in chapter 3, but the hybrid model's reduced computational cost allowed to use super-particles with lower weights. The computational cost was a general problem for the simulations described in this chapter. Because the models lacked grid refinement, the background field had to be significantly above breakdown, so that a discharge would quickly form. In order to get a single negative streamer, the simulations were performed in pure nitrogen without photoionization, and with a localized initial seed. In chapters 6–8, more realistic simulations of discharge inception above breakdown are presented.

For the simulation of streamer discharges, the inclusion of adaptive mesh refinement (AMR) is important, see for example [19]. It became clear that the

spatial coupling in the hybrid model significantly complicated the implementation of mesh refinement. We therefore decided to develop a 'pure' 3D particle model with adaptive mesh refinement. To improve the performance, the particle code present in the hybrid model [20–22] was optimized and parallelized – essentially all code was rewritten. But perhaps even more important was the development of an 'Adaptive Particle Management' method, which adaptively adjusted the weights of simulation particles. In chapter 4, we describe how a k-d tree can be used to merge simulation particles, and we discuss various merging schemes. When going from two to one particle, it is generally not possible to conserve both momentum and energy. We therefore introduced schemes that used random numbers, so that on average energy and momentum could both be conserved.

In chapter 5, the implementation of adaptive mesh refinement and a needle electrode into the particle model are discussed. A major challenge was the solution of Poisson's equation. An ideal solver would allow for the inclusion of mesh refinement and electrodes, yet still be highly efficient, and run in parallel. Because no such solver was available to us, mesh refinement was implemented as in [29], using the fast Poisson solvers from [30]. An electrode was included using a custom algorithm, which can be seen as an iterative charge-simulation method [31]. The resulting model is then used to study the inception of pulsed discharges near a positive needle electrode, in various nitrogen/oxygen mixtures. We observe the formation of small 'inception clouds' [32–34], depending on the electrode voltage and the oxygen concentration. The oxygen concentration is important because it changes the typical length scale for photoionization, thereby affecting the density of free electrons around the discharge. Note that the source code of the particle model is available as *free software* at [35].

Chapters 6 – 8 deal with the development of discharges in background fields above the breakdown threshold. Several authors have performed streamer simulations in such background fields in air, often using a plasma fluid model in cylindrical symmetry [14, 14, 15, 36–38]. In these simulations, the formation of so-called 'double-headed' streamers was observed, that grew from an initial seed. In chapters 6 and 8, we show that the presence of background ionization is important for the discharge evolution. Instead of a single (double-headed) streamer, many overlapping electron avalanches start to grow in ambient air. The initial electrons that start these avalanches can come from electron detachment, discussed in some detail in chapter 8.

A more theoretical analysis of discharge growth above breakdown is given in chapter 7, where we introduce the 'ionization screening time' τ_{is}. This time scale is a generalization of the Maxwell time (ε_0/σ), by taking into account electron impact ionization, which increases the conductivity σ over time. We also present estimates for the homogeneity of overvolted discharges, and discuss the effects of photoionization and electron detachment in N_2:O_2 mixtures.

Streamers are generally thought to propagate (approximately) along electric

field lines. In chapter 9, experimental results from S. Nijdam and E. Takahashi show that a positive streamer can be guided by weak pre-ionization from a laser. The guiding is observed in nitrogen/oxygen mixtures with less than about 0.5% oxygen, at 133 mbar. The particle model described in chapter 5 is then used to investigate how such guiding can occur. Photoionization is found to play an important role. When there is little photoionization, almost all free electrons ahead of the streamer come from the laser channel, so that the streamer is likely to follow the pre-ionized path.

In most of the previous chapters, a 3D particle model is used for the discharge simulations. Even with the techniques described in chapter 4 and 5, such a model is computationally more expensive than a plasma fluid model. Furthermore, the Poisson solver of the particle model is sequential, and the mesh refinement has some limitations (see chapter 5). In order to perform larger or faster discharge simulations, we have developed a framework called Afivo: 'Adaptive Finite Volume Octree'. The framework can be used for finite volume simulations in two and three dimensions, on quadtree and octree meshes. In chapter 10, we describe the implementation of Afivo. An important component is the geometric multigrid [39–41] solver, which can operate in parallel on the adaptively refined grids. There already exist several frameworks for doing parallel simulations on adaptively refined structured grids, for example [42–44]. The main motivation for developing Afivo was to provide a smaller, simpler alternative, not designed for large scale parallel computations – instead shared-memory parallelism is included.

Non-locality complicates the parallel implementation of an algorithm, which explains why it is hard to construct efficient parallel Poisson solvers. In many discharges, another non-local process is important: photoionization [45, 46]. The implementation of photoionization has proven to be quite challenging, see for example [17, 47, 48]. In chapter 11, we present the implementation of a Monte Carlo method for photoionization that can be used in plasma fluid models. The method was inspired by the discrete-photon approach of the particle-in-cell code described in [11]. The two main contributions are the adaptive absorption of individual photons on different mesh levels, and the introduction of a systematic sampling method.

Finally, appendix A contains a more personal discussion of the author's experience in trying to become a computational scientist/physicist. Appendix B describes how the Koren flux limiter [49] can be implemented, to demonstrate how the implementation of an algorithm can differ from its description.

Chapter 2

An introduction to discharges, streamers and their numerical modeling

There exists a wide variety of electrical discharges, some of which are well-known, such as lightning. Section 2.1 contains a brief description of what electric discharges are and how they form. Most of the work in my thesis focuses on *streamer discharges*, which are discussed in more detail in section 2.1.5. Even when we only consider streamers, there are still several *numerical discharge models* that can be used. A brief overview of these models is given in section 2.2, in which the *electrostatic approximation* and typical *Poisson solvers* are also discussed.

2.1 Electric discharges

An electric discharge occurs when an electric current flows through an otherwise non-conducting medium. A discharge thus requires a voltage difference and a mechanism to ionize the medium, so that a current can flow.

2.1.1 Conductors and insulators

We can classify most substances based on their electric conductivity. Conductivity can be expressed in S/m (siemens per meter), as the ratio of the electric current J to the electric field E.

Conductors such as copper or iron have a high conductivity. In these materials, some of the electrons can move relatively freely. When a voltage is applied over a conductor, these electrons are accelerated by the resulting electric field. As they move through the medium, collisions with atoms or molecules slow them down again, which generates heat.

Insulators on the other hand contain almost no mobile electrons or other charge carriers. With the same applied voltage, the resulting electrical current will thus be negligible.

Even though we have no perfect conductors or insulators[1], the difference between them is typically large enough to warrant such a distinction. For example, the conductivity of copper is about 10^{22} times larger than that of air. This means that a copper wire with a radius of 0.1 mm is a better conductor than a column of air with the radius of the earth.

2.1.2 Important processes in discharges

As already mentioned above, insulators contain almost no mobile charge carriers. In order to form a discharge, such carriers can be produced by the ionization of neutral atoms or molecules, which generates electron-ion pairs. The ionization of a neutral requires a certain energy, which is called the ionization potential. This energy can be supplied in a number of ways.

Suppose that an insulating gas such as air is exposed to a high electric field. When an electron-ion pair is created, the free electron will be accelerated by this field, gaining energy. The ion will also accelerate, but not as quickly because of its much larger mass. Compared to electrons, ions also lose more energy in elastic collisions with neutrals, because the colliding masses are similar. Therefore we ignore the ions for now. As the electron moves through the medium, it collides with neutral atoms or molecules. There are different types of electron-neutral collisions, some of the most important ones are:

- Elastic collisions,

  $$e + M \rightarrow e + M$$

  in which the internal energy of the colliding particles does not change. An amount of energy proportional to m_e/M can be exchanged, where m_e is the electron mass and M the neutral mass.

- Excitations,

  $$e + \mathrm{M} \rightarrow e + \mathrm{M}^*$$

  in which some of the electron's kinetic energy is transferred to the internal energy of the atom or molecule. The generated excited states can be lost due to collisional or radiative quenching, or due to a de-excitation in which their energy is transferred back to an electron.

- Electron impact ionization,

  $$e + \mathrm{M} \rightarrow e + e + \mathrm{M}^+$$

[1] Superconductors are perfect conductors, but only below a certain current.

which can happen if the electron carries enough energy to liberate another electron from the neutral. This is the main ionization mechanism in most discharges.

- Attachment, in which the electron sticks to the neutral to form a negative ion. In air, typical attachment processes are

$$e + \mathrm{O_2} \rightarrow \mathrm{O^-} + \mathrm{O}$$

$$e + \mathrm{O_2} + \mathrm{M} \rightarrow \mathrm{O_2^-} + \mathrm{M}$$

(The process $e + \mathrm{O_2} \rightarrow \mathrm{O_2^-}$ is less important, because energy and momentum conservation is hard when going from two to one particle.)

Besides electron impact ionization, ionization can also be produced in other ways, which we briefly discuss below. *Photoionization* occurs when excited atoms or molecules emit UV photons with enough energy to ionize other neutrals. A related process is *Penning ionization*: an excited molecule $\mathrm{M_1^*}$ can ionize a molecule $\mathrm{M_2}$ if its excitation energy is higher than the ionization potential of $\mathrm{M_2}$. The heat generated by a discharge can also cause *thermal ionization* to occur, when the gas molecules have kinetic energies comparable to the ionization potential. Ionization can also be produced at surfaces in or around the medium, for example due to the *ion impact* on an electrode, or by *photoemission*. Furthermore, collisions between excited molecules can create higher excited states, which can eventually lead to ionization.

The main mechanism by which ionization can be lost is *recombination*, where one can distinguish between electron-ion recombination and the recombination of positive and negative ions. For laboratory discharges, the diffusion of charged particles to the walls can also be important, especially at low pressures.

2.1.3 Electron avalanches

Suppose that we expose a gas such as air to a high electric field. Every now and then free electrons will appear, for example because a neutral gets ionized by cosmic radiation. Depending on the applied field and the gas, there is a certain probability that electrons ionize further neutrals by impact ionization. There will also be a certain probability of losing electrons due to attachment. These probabilities can be expressed as Townsend coefficients α and η. The number α expresses how many ionizations a single electron produces per unit length. Because an electron can only attach once, we can interpret $\eta\, dx$ as the probability that attachment occurs over a small length dx.

If $\alpha > \eta$, then ionization is more likely than attachment, and we expect *electron avalanches* to form. Each electron that gets liberated can in turn liberate other electrons. The result is an avalanche of electrons, in which the number of electrons grows approximately exponentially in time. The minimal electric field

at which $\alpha > \eta$ is called the *critical* or *breakdown* field E_c. In atmospheric air, this field is about 30 kV/cm, which corresponds to 240 V over a distance of 0.08 mm.

There are several factors that determine the critical field of a gas. The ionization potential and the electronegativity of the gas molecules will play a role, but the excited states can sometimes be even more important. If electrons are likely to lose energy in excitations, then this reduces the probability that they gain enough energy for ionization. This is the reason that discharges form at lower fields in noble gases, which have relatively few excited states below the ionization potential. A related difference between molecular and atomic gases is that molecules have rotationally and vibrationally excited states.

2.1.4 From avalanche to streamer

In electric fields above breakdown we have $\alpha > \eta$, so that electron avalanches can form. Their growth can be described by the *effective ionization coefficient*: $\bar{\alpha} = \alpha - \eta$. Starting from a single electron, an avalanche contains about N_e electrons after propagating a distance d:

$$N_e = e^{\bar{\alpha} d}. \tag{2.1}$$

In atmospheric air exposed to a background field of 10 MV/m, a typical value for $\bar{\alpha}$ is $\bar{\alpha} \approx 9 \times 10^4\,\mathrm{m}^{-1}$. For $d = 100\,\mu\mathrm{m}$ this gives $N_e \sim 10^4$, a reasonable value. But if we take a larger distance, for example $d = 1\,\mathrm{mm}$, equation (2.1) gives $N_e \sim 10^{39}$. Such a large avalanche would never form: as avalanches grow larger, they produce *space charge* effects that modify the electric field, so that equation (2.1) no longer holds.

The *Raether-Meek criterion* is a rule of thumb that states that space charge effects become important when $\alpha d \approx 18$ to 21, or when $N_e \approx 10^8$ to 10^9. At this point, the avalanches become *streamers*[2], which are discussed in the next section.

2.1.5 Streamer discharges

In an electron avalanche, the electrons drift away from the positive ions. This charge separation increases with the size of the avalanche. Eventually, the charge from the avalanche starts to significantly modify the electric field that it propagates in. The avalanche then becomes a *streamer* discharge.

Figure 2.1 shows an illustration of the avalanche-to-streamer transition. The curved space charge layers screen the electric field inside the discharge, which leads to electric field enhancement at its tips. Due to this enhancement, streamers can propagate into regions where the background field is below breakdown.

[2] In chapters 6–8, we will see that streamers do not form if there are sufficiently many overlapping avalanches.

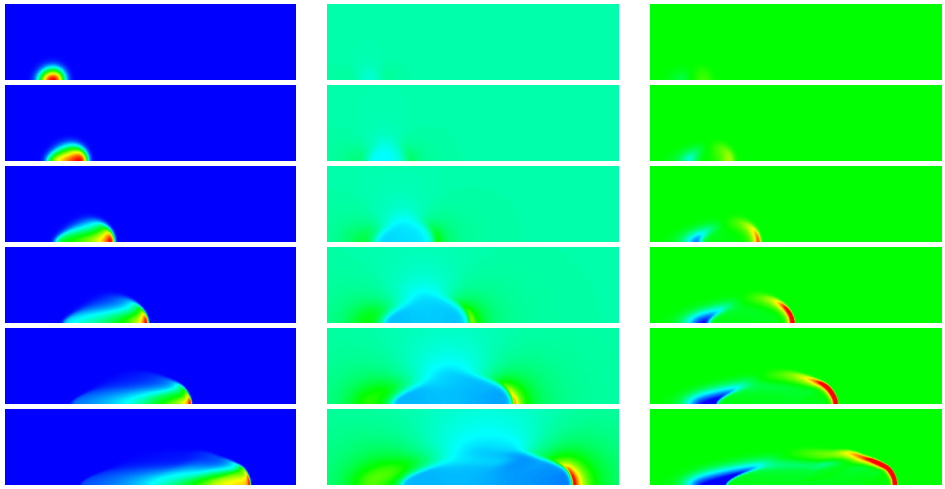

Figure 2.1: Illustration of the avalanche to negative streamer transition, showing the top half of an avalanche propagating to the right. Time proceeds downwards, and n_i indicates the positive ion density. The background field is twice the critical field, and points to the left. Red and blue indicate high and low values, respectively. The color coding for the electron density is adjusted for each row.

As can be seen in figure 2.1, the field enhancement occurs on both ends of the channel. If electrons are present in these high-field regions, they can gain enough energy to ionize neutrals. The degree of ionization increases in time until the region becomes electrically screened. In this way, the channel gradually extends forwards, and the high field translates along with it.

Positive and negative streamers

So-called *positive* streamers propagate in the direction of the applied electric field, whereas *negative* streamers go in the reverse direction. There are also double-headed streamers, which propagate in both directions. A negative streamer propagates in the electron-drift direction, just like an electron avalanche, whereas positive streamers go the opposite way.

The major difference between positive and negative streamers is that positive ones can only grow if there is a source of free electrons ahead of them. These electrons generate ionization in the high-field region around the streamer tip, after which they are absorbed into the channel. In air, such electrons are mostly produced by photoionization. In other gas mixtures photoionization might be weaker or absent, so that the electron density ahead of a streamer will be lower. In such gases, streamer channels are typically thinner and less smooth than in air, see figure 2.2.

A possible explanation for this difference is outlined below. Consider two

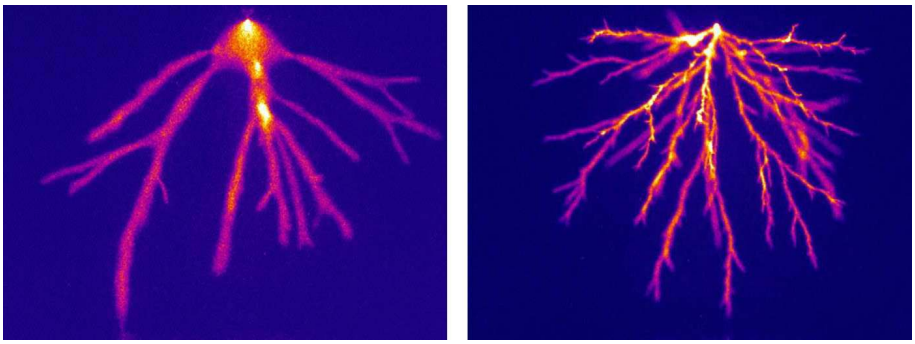

Figure 2.2: Positive streamers in air (left) and 99.9% pure nitrogen (right), under otherwise the same conditions (needle-to-plane geometry, 0.4 bar, 16 kV). Figure taken from [50].

points $\boldsymbol{r}_1$ (on-axis) and $\boldsymbol{r}_2$ (off-axis) in front of a streamer, where the electron density is initially n_0, see figure 2.3. At $\boldsymbol{r}_1$ the electric field is larger, so for the electron growth rate S we have $S_1 > S_2$. Suppose that $S_2 = (1-\eta)S_1$, and that the degree of ionization in the channel is n_{ch}. By the time that $n_1 = n_{\mathrm{ch}}$, we have

$$n_2/n_1 \approx n_2/n_{\mathrm{ch}} = \left(\frac{n_0}{n_{\mathrm{ch}}}\right)^{\eta}, \tag{2.2}$$

where we have for simplicity ignored the effect of electron motion. In other words, the smaller n_0 is, the smaller the ratio n_2/n_1 will be, and with a smaller ratio we expect a thinner streamer.

Perhaps surprisingly, positive streamers often propagate faster than negative ones, and they form at lower voltages in a given electrode configuration [12, 51]. A reason for this is that positive streamers naturally keep their strong field enhancement: when the field ahead of a positive streamer is reduced, its propagation slows down or even stops, because its growth relies on the ionization of the gas in front of it. In negative streamers the electrons drift outwards, which can reduce their field enhancement [12] and slow them down. Because this drift continues in fields below the critical field, it can effectively 'extinguish' a negative streamer.

Typical streamer properties

The properties of lab streamers depend on multiple factors, for example: the type of gas, the gas number density, the presence of background ionization, the experimental geometry and the electric circuit. The properties of a single streamer channel will also depend on its history (e.g., streamer path and conductivity along the path) and on the presence of other streamers. Nevertheless, an approximate characterization is usually possible.

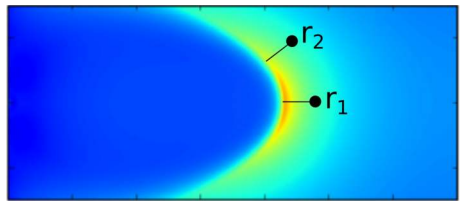

Figure 2.3: Zoom in on the tip of a positive streamer, showing the electric field strength. Two points indicated: $\boldsymbol{r}_1$ (on axis), and $\boldsymbol{r}_2$ (off axis). The streamer picture was extracted from figure 4 of [52].

In [51], the properties of positive and negative streamers were experimentally investigated. In a 4 cm needle-to-plane geometry the voltage was varied from 5 to 96 kV, using three different voltage sources. The properties discussed below have some dependence on the voltage rise time, because (streamer) discharges might already form before the maximum voltage is reached, affecting the further discharge evolution.

In atmospheric air, a minimal radius for positive streamers of about 0.1 mm was observed, at 5 kV. With increasing voltage, the width of these streamers increased, to about 3 mm at 96 kV. Negative streamers started to form at a higher voltage of 40 kV. If positive and negative streamers were created with the same voltage amplitude, they had a comparable radius.

Streamer velocities were also estimated, but in this case there did not appear to be a clear minimum. At 30 kV, positive streamer velocities of about 0.5 mm/ns were observed, which increased approximately linearly to 4 mm/ns at 96 kV. The negative streamers were found to propagate about 25% slower at the same voltage amplitude.

Measuring the electric field at the tip of a streamer is difficult experimentally. In numerical simulations, typical electric fields at the streamer tip are 10 to 20 MV/m in atmospheric air [53]. The electron and ion density in the channel are also hard to measure experimentally. In the appendix of [54], the following estimate was made using a one-dimensional analysis of a negative ionization front:

$$n_e = \frac{\varepsilon_0}{e} \int_0^E \alpha(E')dE', \tag{2.3}$$

where E is the maximum electric field at the streamer tip. In the simulations presented in [52], about 50% higher densities were found. The electron density for a streamer in STP air is typically on the order of 10^{19} to $10^{20}\,\mathrm{m}^{-3}$, which corresponds to a degree of ionization of 10^{-6} to 10^{-5} in atmospheric air.

Scaling laws for streamer heads

Here we briefly introduce some of the scaling laws for streamer discharges [2, 50]. A streamer grows due to the high-field region ahead of it. In this region, electrons typically lose energy in *two-body* collisions with neutrals, such as electron impact excitation or ionization. The electron mean free path l_{mfp} is thus inversely proportional to the gas number density N

$$l_{\mathrm{mfp}} \propto \frac{1}{N}. \tag{2.4}$$

Since the energy gained between collisions is proportional to the applied electric field E, the energy distribution depends on the ratio E/N. This motivates the Townsend unit Td, which is defined as

$$1\,\mathrm{Td} = 10^{-21}\,\mathrm{V}\cdot\mathrm{m}^2. \tag{2.5}$$

If we change N, but keep E/N constant, how do the streamer properties change? Most length scales L are proportional to the electron mean free path, so we have

$$L \propto l_{\mathrm{mfp}} \propto \frac{1}{N}. \tag{2.6}$$

The electron velocity distribution depends on E/N, like the energy distribution. For typical time scales, such as the time between ionizing collisions, we thus have

$$t \propto \frac{1}{N}. \tag{2.7}$$

In electrostatics, the electric field can be computed as

$$\boldsymbol{E} = \frac{1}{4\pi\varepsilon_0}\int \rho(\boldsymbol{r'})\frac{\boldsymbol{r}-\boldsymbol{r'}}{|\boldsymbol{r}-\boldsymbol{r'}|^3}d^3\boldsymbol{r'}, \tag{2.8}$$

where ρ is the charge density. If lengths scale as $1/N$, then for constant E/N we have

$$\rho \propto N^2. \tag{2.9}$$

The electron and ion densities also scale as N^2, and the degree of ionization is proportional to N. The scaling laws given above are not perfect, for example because:

- Three-body processes lead to a different scaling behaviour. An important example is three-body attachment to oxygen molecules.
- Processes in the gas (such as chemical reactions and heating) depend on N, which is especially important inside the streamer channel.
- Although length scales for photoionization are proportional to $1/N$, the collisional quenching of emitting states depends on N.

2.1.6 Other discharges

Depending on the electrode configuration and the voltage source, different types of discharges can be observed in the lab [55]. These include for example:

- **Townsend discharge** A stationary discharge with negligible space-charge effects, in which electron multiplication takes place between two electrodes.
- **Glow discharge** This is a stationary discharge that can be used for lighting. Glow discharges can be seen as Townsend discharges with space charge effects, which cause the formation of different spatial regions.
- **Corona discharge** This is the general name for discharges that form around a high voltage electrode and exhibit space charge effects. There are pulsed and continuous corona discharges, depending on the voltage source. Corona discharges typically include streamers.
- **Arc, spark** The electric current in a discharge can significantly increase the temperature of the gas. Where the gas heats up the pressure increases, which generates an expansion wave. This locally reduces the critical field, making it more likely that current keeps flowing along the heated path. When the kinetic energy (or temperature) of gas molecules, ions and electrons becomes comparable, we speak of an arc or a spark. In such a discharge, thermal ionization plays an important role.

Most people are probably more familiar with discharges in nature, such as lightning or the small electrostatic discharges sometimes occur when one for example takes off a sweater. Other discharges in nature are for example:

- **Lightning leader** In lightning discharges, the conductive channels are generated in different stages. Streamers produce the first ionized channels, mostly without heating the gas. As electric current flows through these channels, they heat up, and some of them become lightning leaders. After a connection to ground, a large amount of charge can suddenly flow, which generates a bright *return stroke*. This causes rapid heating of the ionized channel, which generates a shock wave: thunder.
- **Sprite** A sprite discharge consists of streamers high in the atmosphere, at $50 - 90$ km. At these altitudes, the atmosphere has a much lower density[3]. Therefore sprite channels can be tens of kilometers long, and hundreds of meters wide.
- **Other discharges** High in the atmosphere there are numerous other discharges, e.g., halos, blue jets, and elves [56].

[3] The density decreases exponentially with altitude.

2.1.7 Motivation for studying streamers

Why do we study streamer discharges? There are a number of reasons. First of all, there is scientific curiosity. Even though streamers were already observed and described in the 1920s, we still do not understand all of their properties. For example, even though quite some research has been done on the branching of streamers, see e.g., [14, 25, 26, 28, 57–59], we are not aware of a model that predicts when positive streamers branch. Similarly, there is currently no simple model that predicts how the radius of a single streamer will change in time. For the curious scientist, there are thus enough basic research questions that still need to be answered. Since streamers are often the precursor to other discharges, such as lightning leaders, understanding them can also be important when studying these later discharges.

From the point of view of applications, there are (at least) two reasons why we need a good understanding of streamer properties: To design equipment in such as way that the formation of streamers is prevented, important in the high-voltage industry, and to optimize applications in which streamers occur.

In streamer discharges, the energy from the applied electric field is mostly transferred to electrons, because they accelerate much faster than ions. In the streamer head regions, electrons can reach kinetic energies of a few tens[4] of eV's. One eV already corresponds to an electron temperature of about 10^4 K, so these energetic electrons can be used to catalyze chemical reactions that would otherwise only occur at very high temperatures. Applications in which streamers are used are for example gas and water cleaning [62], ozone generation [6] and plasma medicine [7].

2.2 Modeling streamer discharges

In this section, we look at streamers from the modeling point of view. We first discuss some of the general challenges in simulating streamers, then we briefly introduce particle and fluid models for streamer discharges.

2.2.1 Challenges in streamer modeling

A combination of different factors makes the modeling of streamers quite challenging:

- There are steep gradients in the electron and ion density, which generate thin space charge layers.
- The propagation of streamers is strongly non-linear, because of the coupling with the electric field that is generated by curved space charge layers.

[4] The process of electron-runaway can generate electrons with energies of several keV and higher, see for example [22, 36, 60, 61].

- Streamers are a transient phenomenon.

Because of the non-linear propagation, the thin space charge layers have to be sufficiently resolved, otherwise the behavior can be far from the correct dynamics. In atmospheric air, this means that features of a few μm have to be resolved, whereas the streamer itself is typically much larger (cm or more). This *multiscale* aspect is especially challenging for three-dimensional simulations. Note that at least a two-dimensional description is required to model the curved space charge layers of a streamer.

The transient nature of streamers implies that there is no stationary solution. If streamers would be uniformly translating, a stationary solution could be obtained by going to a co-moving frame. However, streamers are generally not found to be uniformly translating, so that time-dependent simulations are required. In atmospheric air, the smallest time scales that one has to consider are on the order of 10^{-13} to 10^{-11} s:

- The maximal collision rate of electrons is about 10^{13} Hz, which corresponds to a collision time of 10^{-13} s. This is relevant for particle models.

- The Maxwell time $\tau_{\text{Maxwell}} = \varepsilon_0/\sigma$ for electric screening can be very short, see chapter 7. Taking $\sigma = e\mu_e n_e$ as the plasma conductivity due to electrons, we for example get $\tau_{\text{Maxwell}} \approx 10^{-12}$ s for $n_e = 1.5 \times 10^{21}\,\text{m}^{-3}$ and $\mu_e = 4.0 \times 10^{-2}\,\text{m}^2/(\text{Vs})$.

- In general, electrons should move less than one grid spacing at a time. The electron drift velocity divided by the grid spacing can give time scales of 10^{-11} s or less.

Having to resolve such small time scales greatly increases the cost of simulations. An additional complication is that the electric field in such a simulation has to be recomputed at every time step. Using the electrostatic approximation, see section 2.2.5, we need to compute the electric potential, or solve Poisson's equation. Although this is a classic and much studied problem, solving Poisson's equation sufficiently fast is still one of the major challenges in developing streamer simulations, see section 2.2.6.

2.2.2 Particle models

We have already looked at the essential processes in streamer discharges in section 2.1.2: electrons gain energy from an electric field, and lose this energy in various collisions with neutrals. A *particle model* directly describes the behavior of these electrons. In such a model, a large number of electrons is followed over time, by storing their position $\boldsymbol{x}$ and velocity $\boldsymbol{v}$. The electrons are accelerated by the electric field, and depending on their energy, they have a certain probability of

colliding with a neutral. These probabilities are the input data for a particle model, in the form of cross sections.

The neutrals themselves are typically included as a homogeneous background, because it would be computationally unfeasible to simulate them individually[5]. The location of an electron-neutral collision can then be determined by a Monte Carlo procedure, using random numbers. Depending on the application, one can include the ions that are produced as a density or also as particles. Even if we only model the electrons as particles, there will typically be more electrons than we can simulate on a computer. For this reason, so-called *super* or *macro* particles are often used, see chapter 4. Such particles represent multiple physical electrons, which reduces the computational cost at the expense of increased stochastic fluctuations.

Directly evaluating the electric forces between the electrons (and ions) would be too costly. Instead, the particles are mapped to a density on a numerical grid. On this grid, the electric field is computed and interpolated back to the particles. Codes that use this approach are referred to as *particle-in-cell* codes.

In general, the use of a particle model has the following (dis)advantages:

- \- Computationally expensive
- \- Requires detailed input data, in the form of cross sections
- \- The use of super-particles generates artificial noise
- \+ Close to a 'first principles' approach, relatively few assumptions
- \+ The distribution function of particles $f(\boldsymbol{x}, \boldsymbol{v}, t)$ is directly approximated
- \+ Realistic particle density fluctuations can be included

2.2.3 Fluid models

In particle models, the use of super-particles reduces the computational cost. The main reason that this works is that not all electrons have to be described individually to capture their 'average' behavior. In fluid models, this idea is pushed even further: the electrons are no longer described as particles but as a density. A review of fluid models for streamer discharges can be found in [19].

In the simplest case, just the electron and ion density are considered. How these densities change in time can be described by making some (phenomenological) approximations. A classical and popular example is the convection-diffusion-reaction model:

$$\begin{aligned} \partial_t n_e &= \nabla \cdot \boldsymbol{j}_e + S \\ \partial_t n_i &= S \\ \boldsymbol{j}_e &= \boldsymbol{v} n_e + D \nabla n_e, \end{aligned}$$

[5] A mm^3 of atmospheric air already contains 2.5×10^{16} neutral molecules.

where n_e and n_i are the electron and positive ion density, S is the source term, $\boldsymbol{j}_e$ is the electron flux, $\boldsymbol{v}$ the velocity of electrons and D is a scalar diffusion constant. To use such a model, these terms have to be specified. A popular strategy is to use the so-called *local field approximation*, which assumes that the electrons are relaxed to the local electric field. By measuring their properties in various fields, one can then construct tables with coefficients, which can be used in the model.

Fluid models can also be derived in a systematic way, by taking 'moments' of the Boltzmann equation. Several 'moments' can be included, for example the density n, the momentum density $mn\boldsymbol{v}$, the energy density $\frac{1}{2}mnv^2$, and so on. For each moment, an equation can be derived expressing its change in time. Because these equations depend on a higher moment, some approximation has to be made to close the system.

For many applications, fluid models are computationally cheaper than particle models. This speedup comes at a cost: some assumptions have to be made about the particle distribution, which may not always be valid. Fluid models that contain more physics will have a wider range of validity, but their implementation can be challenging. When multiple coupled equations have to be solved, the steep gradients present in a streamer discharge can easily lead to oscillations.

Fluid models do not contain the density fluctuations present in particle models. This can be an advantage, because there is no artificial noise, but also a disadvantage, because physical density fluctuations cannot be described[6].

2.2.4 Other models

Hybrid models

Here we briefly mention two other classes of models that can be used for simulating streamers. The first are the so-called *hybrid* models, which combine a particle and a fluid approach, see for example [20, 21, 63]. There are several ways to combine these models, for example:

- Separate models in energy, by describing high-energy electrons as particles, and the low energy ones as a density.
- Separate models in space, by using a fluid model in part of the domain and a particle model in the rest.

A model in which electrons are treated as particles and ions as a fluid can technically also be called a hybrid model.

[6] In [28], certain density fluctuations have actually been incorporated into a fluid description.

Tree models

Another class of models are *tree* or *macroscopic* models. The microscopic simulation of the propagation of even single streamers has proven to be computationally expensive; in order to simulate larger discharges containing many streamers, we therefore need to greatly simplify their description. In a tree model, this is done by not resolving the micro-physics. Instead, the streamer channels are described as conducting lines or cylinders. Their growth velocity, conductivity, radius and branching behavior then have to come from a macroscopic description. In [64], a phenomenological discharge model was introduced to investigate the fractal nature of discharges in a plane. Several variations of this model have been described in the literature. Recently, a three-dimensional tree model including more realistic physics and consistent charge conservation was presented in [65].

The validity and accuracy of a tree model greatly depends on how well the streamer channels can be characterized. It is not yet known what kind of description would be required. In the ideal scenario, simple equations are found that can predict how a streamer develops. If this does not work, another option would be to perform extensive parametric studies, to generate tables with streamer properties under different conditions. This is a topic that I will work on at the very end of my PhD, so that the results will not be in this thesis.

2.2.5 Electrostatic approximation

In discharge simulations, we typically assume to work under electrostatic conditions, which greatly simplifies the calculation of the electric field. Instead of having to solve Maxwell's equations

$$\nabla \cdot \boldsymbol{E} = \rho/\varepsilon_0, \tag{2.10}$$
$$\nabla \cdot \boldsymbol{B} = 0, \tag{2.11}$$
$$\nabla \times \boldsymbol{E} = -\partial_t \boldsymbol{B}, \tag{2.12}$$
$$\nabla \times \boldsymbol{B} = \mu_0 \left(\boldsymbol{J} + \varepsilon_0 \partial_t \boldsymbol{E}\right), \tag{2.13}$$

only the electrostatic potential ϕ has to be computed

$$\nabla^2 \phi = -\rho/\varepsilon_0, \tag{2.14}$$

after which the electric field is obtained as $\boldsymbol{E} = -\nabla\phi$. A system can be described by electrostatics when

$$\nabla \times \boldsymbol{E} = -\partial_t \boldsymbol{B} = 0, \tag{2.15}$$

or in other words, when $\boldsymbol{E}$ is conservative or rotation-free. With some approximations, we can estimate to what extent this condition holds for a streamer discharge.

For simplicity, we approximate the streamer as a finite wire, see figure 2.4. We assume that the wire carries a constant current I. Assuming that the streamer

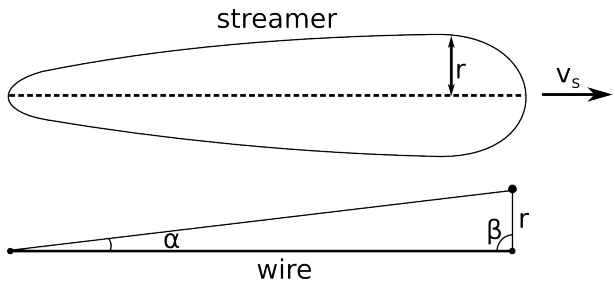

Figure 2.4: Schematic drawing of a streamer (top), which is then represented as a finite wire (bottom). A point that is a distance r above the 'streamer head' is indicated.

velocity v_s is significantly smaller than the speed of light, one can obtain the following expression for the magnetic field at a distance r above the wire [66]

$$B = \frac{\mu_0 I}{4\pi r} \left[\cos\alpha + \cos\beta\right], \tag{2.16}$$

see figure 2.4 for the definition of the angles. The largest magnetic field occurs near the middle of the streamer where α and β are small, so that

$$B \approx \frac{\mu_0 I}{2\pi r}. \tag{2.17}$$

The right-hand side of the above equation is the expression one gets for an infinite current-carrying wire or outside a current-carrying cylinder[7]. The current I can be written as

$$I = A \cdot |\boldsymbol{J}| = \pi r^2 \cdot e n_e \mu_e E_{\text{int}}, \tag{2.18}$$

where A is the cross-sectional area of the streamer, $\boldsymbol{J}$ the current density, n_e the electron density, μ_e the electron mobility and E_{int} the average electric field inside the channel. For atmospheric air, reasonably high values are $r = 1\,\text{mm}$, $n_e = 10^{21}\,\text{m}^{-3}$, $E_{\text{int}} = 5 \times 10^5\,\text{V/m}$ and $\mu_e = 7 \times 10^{-2}\,\text{m}^2/(\text{Vs})$. This gives $B \approx 3.5\,\text{mT}$. Such a magnetic field has little effect on an atmospheric discharge, as it gives an electron cyclotron frequency of about $10^8\,\text{Hz}$, whereas the electron collision frequency is 10^{12} to $10^{13}\,\text{Hz}$.

To determine how good the electrostatic approximation is, we have to estimate the magnitude of $|\nabla \times \boldsymbol{E}| = |\partial_t B|$. We expect $\partial_t B$ to have the largest magnitude right above the 'streamer head', see figure 2.4. If the wire extends at a velocity v_s, then in approximation

$$\partial_t B \approx \frac{\mu_0 I v_s}{4\pi r^2} = \frac{1}{4}\mu_0 v_s \left|\boldsymbol{J}\right|, \tag{2.19}$$

[7] Assuming that the current distribution in the cylinder depends only on the distance from its axis.

where we have used the fact that $\alpha \ll 1$ and $\beta = \pi/2$, so that $\partial_t \cos\alpha \approx 0$ and $\partial_t \cos\beta = v_s/r$. Using the above values and $v_s = 5 \times 10^5$ m/s gives

$$\partial_t B \approx 10^6 \text{ T/s}. \tag{2.20}$$

This is thus also a typical value for the maximum of $|\nabla \times \boldsymbol{E}|$. On the other hand, the typical maximum for $\nabla \cdot \boldsymbol{E} = \rho/\varepsilon_0$ is on the order of 10^{11} T/s. Because we have $|\nabla \times \boldsymbol{E}| \ll |\nabla \cdot \boldsymbol{E}|$, the electrostatic approximation is probably quite accurate.

2.2.6 Solving Poisson's equation

As already mentioned above, solving Poisson's equation sufficiently fast is a major challenge in developing streamer simulations. How difficult this task is depends first of all on the spatial dimension of the simulation. In 1D, $\partial_x E = \rho/\varepsilon_0$ has as a solution

$$E(x) = E(x_0) + \int_{x_0}^{x} \rho(x')/\varepsilon_0 \, dx', \tag{2.21}$$

which can easily be computed numerically. In 2D and 3D, Poisson's equation has to be solved, from which the electric field can be obtained as the numerical gradient. A good (although dated) overview of methods can be found in [67]. There are essentially two classes of Poisson solvers: direct methods and iterative methods. With an iterative method, each iteration improves the solution until there is convergence, whereas a direct method *directly* gives its 'best' solution.

The type of mesh plays an important role in deciding which Poisson or *elliptic* solver to use. Depending on the type of grid, there are probably four competitive options:

- The FACR algorithm [68], a combination of the Fourier transform and cyclic reduction. This direct algorithm is for example implemented in Fishpack [30], which is used for the 3D particle model described in chapter 5. The FACR algorithm has a computational complexity of $N \log(N)$, where N is the number of unknowns.

- Geometric multigrid, an iterative method which uses a hierarchy of grids to improve the rate of convergence. Multigrid solvers are implemented in for example Mudpack [69] or [70], and they have an optimal complexity of order N [39–41]. In Afivo, a simulation framework described in chapter 10, we have implemented geometric multigrid on adaptively refined quadtrees and octrees.

- A direct sparse solver, which can solve arbitrary sparse systems of linear equations. Such solvers have different stages, and typically there is a separate factorization and/or analysis step before the solution can be computed.

The cost of the steps depends on the matrix structure, and there is generally a trade-off: the more work is put into factorization or analysis, the faster solutions for the given matrix can be obtained. Examples of direct sparse solvers are MUMPS [71] and UMFPACK [72].

- An iterative sparse solver combined with a pre-conditioner. In principle, such a method can be used to solve arbitrary sparse systems of linear equations. The efficiency depends strongly on the quality of the pre-conditioning, however. Iterative sparse solvers typically have much lower memory requirements than direct sparse solvers. An iterative method can for example combine GMRES [73] (a *Krylov* method) with BoomerAMG [74] (an algebraic multigrid pre-conditioner), but there are many other options.

These methods vary in their flexibility. Whereas the FACR algorithm can only be used for separable elliptic problems on Cartesian grids[8], geometric multigrid can already be applied much more generally. For complicated grids with e.g., embedded boundaries or irregular cells, one typically has to resort to a more general sparse solver.

In practice, the usage of a direct sparse solver can work well for two-dimensional simulations: one can construct complex mesh geometries as long as one is able to discretize the underlying equations. Each update of the numerical mesh will incur some computing costs, because the sparse solver has to operate on a new matrix. This makes it more attractive to work with a statically refined mesh, or a mesh that is not updated very frequently. In most cases, solution times will not be prohibitive.

If we go to three-dimensional simulations, the number of unknowns can increase by two or more orders of magnitude. This makes the computational cost of direct sparse solvers prohibitive for time-dependent streamer simulations. An attractive alternative is geometric multigrid. With this technique, the numerical mesh of the simulation can directly be used to efficiently compute solutions to elliptic equations. Changing the mesh in time does not incur extra costs, and no matrix needs to be stored. The downside is that geometric multigrid is not as flexible as a direct sparse solver; it is therefore harder to include for example curved electrodes.

[8] Cylindrical coordinates are also possible.

Chapter 3

A comparison of 3D particle, fluid and hybrid simulations for negative streamers

Modeling individual free electrons can be important in the simulation of discharge streamers. Stochastic fluctuations in the electron density accelerate the branching of streamers. And in negative streamers, energetic electrons can even 'run away' and contribute to processes such as terrestrial gamma-ray and electron flashes. To track energies and locations of single electrons in relevant regions, we have developed a 3D hybrid model that couples a particle model for single electrons in the region of high fields and low electron densities with a fluid model in the rest of the domain. Here we validate our 3D hybrid model on a 3D (super-)particle model for negative streamers without photo-ionization in overvolted gaps. We show that the extended fluid model approximates the particle and the hybrid model well until stochastic fluctuations become important, while the classical fluid model underestimates velocities and ionization densities. We compare density fluctuations and the onset of branching between the models, and we compare the front velocities with an analytical approximation.

This chapter has been adapted from [75]:
A comparison of 3D fluid, particle and hybrid model for negative streamers, C. Li, J. Teunissen, M. Nool, W. Hundsdorfer, U. Ebert, *Plasma Sources Sci. Technol.* 21, 055019 (2012)

3.1 Introduction

Streamers are growing ionized fingers that appear when ionizable matter is suddenly exposed to high voltages. Streamers pave the path for lightning leaders and precede sparks, and they occur without the subsequent stages in the form of enormous sprite discharges high above thunderclouds. Streamers are also used in diverse industrial applications. As reviewed, e.g., in [2, 23, 24], the evolution of a single streamer consists of phenomena on several length scales: the ionizing and exciting collisions of fast electrons with molecules, the emergence of an ionization front with an electric screening layer, and the emergence of a streamer finger surrounded by such a screening layer and ionization front. The dynamical instability of a thin screening layer can make a streamer branch [19, 23–28]. In negative streamers with high field enhancement energetic electrons can run away from the front and emit hard electromagnetic radiation; taking the further interaction of these run-away electrons with the atmosphere into account, this is a possible explanation [11, 22, 36, 61, 76, 77] of terrestrial gamma-ray flashes [78], electron beams [79] or even electron positron beams [80] emitted from active thunderstorms.

Streamer propagation is mostly investigated with a density or fluid approximation for the electrons and ions, which continues to be very challenging due to the widely separated scales; for recent articles we refer to [81–86] and for a recent review to [19]. However, there are stages of evolution where the statistics of single electrons matters, either due to their nonthermal energy distribution with long tails at very high energies [11, 21], or due to their stochastic presence in non-ionized regions. Examples include electron run-away from streamers, ionization avalanches created by single electrons that have now been observed experimentally in very clean gases [87–89], or density fluctuations that can accelerate streamer branching, as was shown in recent simulations [28]; this last study investigated positive streamers in air with photo-ionization in a background field below the ionization threshold.

To track the energy and density fluctuations accurately, a Monte Carlo particle model for streamer simulations tracks single free electrons as they move and randomly collide with neutrals; neutral molecules are not simulated but provide a background that electrons stochastically collide with, see for example [9, 11, 21]. Here we look at very short timescales on which ions can be assumed to be immobile. The model contains the energy and location of each electron as well as the spatial distributions of ions and of the different types of excited states. Therefore it also can accurately simulate rare events like electron run-away or avalanche formation from single electrons. But computer memory strongly constrains the number of electrons that can be tracked. Streamers usually form when the total number of free electrons reaches $10^7 - 10^9$ in air at standard temperature and pressure [10, 90, 91], and during streamer growth the electron number continues to increase. This makes computations with real electrons very expensive or

even impossible, and typically super-particles representing many real particles are used to accelerate computations. However, super-particles can introduce unphysical fluctuations and numerical heating, as shown in [10] and below. They also corrupt the statistics of rare events. This statistical problem motivated us to develop our 'spatially hybrid model'.

In a streamer discharge, most electrons reside in high densities in the low field region in the streamer interior. This region is typically in the 'hydrodynamic' regime, that can be well described by a fluid model. Relatively few electrons are in the region of strong field enhancement at the streamer tip or outside the streamer, and only those electrons should be tracked individually with a single particle model (as opposed to super-particles). Therefore we have developed a code that is hybrid in space [20, 21, 76], applying a fluid approximation in the streamer interior and a single particle model at the streamer tip and in the essentially non-ionized region around it. We call it the 'spatially hybrid model' to distinguish it from other hybrid models, see for example [63]. The model follows single electrons and their fluctuations in the dynamically relevant region.

In the present paper, we test the consistency and correct implementation of the particle and the hybrid model on propagating negative streamers in air (while neglecting photo-ionization), and we illustrate the influence of the (super-)particle fluctuations on the destabilization of the streamer ionization front. For comparison, we also present simulations of the same system with a classical and with our extended fluid model, and we compare the computing times. In the conclusion we also discuss the effect of photo-ionization that is excluded in the present work for technical reasons. Therefore the present results are actually more appropriate for discharges in gases without photo-ionization, like pure nitrogen.

3.2 Description of particle, fluid and hybrid model

3.2.1 Particle model

The Monte Carlo particle model is of PIC-MCC type. It describes the motion and collisions of free electrons in a streamer discharge in air without photo-ionization. 'Particle in cell' (PIC) means that the electric charge of electrons and ions is mapped to an electric charge density on a numerical grid; this charge density changes the electric potential (from which the electric field is calculated) according to the Poisson equation. 'Monte Carlo collision' (MCC) means that collisions of the free electrons with the neutral background molecules occur randomly, so neutral molecules do not need to be simulated. Ions are treated as immobile. The Monte-Carlo procedure and the differential cross-sections for the particle model are described in detail in section 2.1 of [21].

The particles represent single electrons during the initial stages of the simulation, as indicated in the text. However, when the particle number becomes

too large for the computer memory, super-particles are introduced that represent several real particles. While particles carry their generic physical distributions and fluctuations of density and energy, the fluctuations of super-particles are unphysically increased and can generate artifacts when fluctuation effects or rare events become important. To avoid this problem, we have developed the spatially hybrid model. The different models are compared in sections 3.3 and 3.4.

3.2.2 Classical fluid model, bulk and flux coefficients

The classical fluid model for streamers has a long tradition in streamer modeling, much longer than the more microscopic particle model. Originally it is a phenomenological model based on the essential physical mechanisms and conservation laws. It can be traced back at least to the 1930s. The classical fluid model approximates the electron dynamics by a reaction-drift-diffusion equation for the electron density, and the reaction and transport coefficients are assumed to depend on the local electric field, in the so-called 'local field approximation'. The model is completed with the reaction equation for the ions and with the Poisson equation for the electric potential.

In order to approximate the particle dynamics well, the coefficients in the fluid model should be derived from the particle model. This can be done either through averaging over the Boltzmann equation for the electron distribution in configuration space, or through evaluating swarm simulations in a Monte Carlo particle model; in a swarm simulation the evolution of an approximately Gaussian electron distribution in a constant electric field is traced by a particle model. For the present paper the transport coefficients and the reaction rates for the fluid model are derived by fitting the fluid coefficients to the swarm dynamics in a Monte Carlo particle model, as evaluated in [21]. (Please note that [21] contains some corrections to the reaction and transport rates described in [20, 76].) The coefficients in [21] were derived up to a field of 250 kV/cm in air at standard temperature and pressure, and for stronger electric fields fit formulas from the same article are used.

Furthermore, in a reactive plasma one needs to distinguish between bulk and flux coefficients [92]. While bulk coefficients characterize the dynamics of a swarm as a whole including its reactions, flux coefficients characterize the dynamics of individual electrons within a swarm. Robson *et al.* [92] express the general opinion that bulk data should not be used in low-temperature plasma simulations, see also figure 2 in [92].

However, essential physics is missing in the classical or minimal model as we found in [20], and as will be discussed much more extensively and systematically in a forth coming article [93]. For example, in strong electric fields, the ionization rate cannot be computed accurately from just the local electric field and the local electron density. Therefore in [20] we extended the generation term in the fluid model with a gradient expansion term while in [93] a higher order

model is derived in a systematic manner by averaging over more moments of the Boltzmann equation. An important conclusion is that accurate results cannot be expected from the classical fluid model, independently of whether flux or bulk coefficients are used, as the functional form of the equations is not sufficiently general.

For comparing the classical fluid model with the other models, we have chosen to use bulk coefficients. From studies of planar (1D) fronts, we know that bulk coefficients in the classical fluid model will lead to approximately correct front velocities. This is the case because the leading edge of an ionization front, that pulls the front along, propagates under swarm-like conditions; for details, we refer to [76] or to a full mathematical analysis of pulled front dynamics to [94]. A particle swarm should be parameterized with bulk coefficients in the classical model (recall that the bulk coefficients are constructed to give agreement for particle swarms). Therefore, the streamer ionization front should also be modeled with bulk coefficients in the classical model. The front velocity is then close to that of a particle model in the same electric field, although the ionization density behind the front is too low [76]. If flux coefficients would be used, the ionization density behind the front would be a bit higher. (Both sets of coefficients produce the same amount of ionization per unit time in a given electric field. The flux mobility is typically lower, therefore a larger amount of electrons per unit length is produced, and the electron density is higher.) But the front velocity would be significantly too low with flux coefficients, and therefore we use bulk coefficients. We stress again that accurate results can not be expected from the classical fluid model with either set of coefficients.

3.2.3 Extended fluid model

The ionization term in the classical fluid model for streamers is calculated in local field and local density approximation, but the comparison with particle models shows that the ionization densities in the streamer interior are too low behind a planar front in a fixed electric field [20, 76, 95]. This is because the mean electron energy varies even within a swarm in a constant electric field: at the front edge of the swarm, the electrons have higher energies and are more likely to ionize the neutrals, while the electrons at the back end of the swarm are slower on average and less likely to ionize. By including the first term of a gradient expansion in the electron density in the impact ionization rate, both particle swarms and planar ionization fronts are approximated well [20]. The extra term was derived in a model with flux coefficients, therefore we need to use flux coefficients for consistency with the model derivation.

3.2.4 Hybrid model

The hybrid model connects the particle model with the extended fluid model through a moving model interface with a buffer zone, as described extensively in [21] and briefly recalled in the introduction. When the flux of electrons across the interface between particle and fluid model is calculated, the same definition of coefficients should be used on both sides in order to be physically consistent; otherwise the physical inconsistency becomes visible in the form of a discontinuity of the electron density at the model interface [20]. For further details and the numerical implementation, we refer to [20, 21, 76]. The important feature of the spatially hybrid model is that it follows the particle dynamics only in the dynamically relevant region, and that it therefore can continue to track the single electron fluctuations much longer than the (super-)particle model.

3.3 Simulation methods and results

3.3.1 The simulated system

We simulate the evolution of a negative streamer in air without photo-ionization, at standard temperature and pressure. The streamer propagates in a background field of -100 kV/cm, or 372 Td; this field is well above the break-down value. The simulation volume is 1.17 mm long in the z-direction parallel to the electric field and extends up to ± 0.29 mm outwards from the axis in the x and y direction where homogeneous Neumann boundary conditions are applied to the electric potential.

The initial distribution of electrons and ions is generated in the following manner: first 500 electrons and ions are placed at a distance of 0.115 mm from the cathode on the z-axis and followed by the particle model for 60 ps. At this time, there are about 2500 electrons and ions with spatial distributions close to a Gaussian; the electrons are at this time all in the interval of 0.120 to 0.136 mm from the cathode. These electron and ion distributions are then used as an initial condition for the simulations in all four models; for the fluid models, the swarm is mapped to densities on the numerical grid.

3.3.2 Numerical implementation

The models were already described in the previous section, and references to more detailed discussions were given there as well. Electric field and electron and ion densities are calculated on a uniform grid of $256 \times 256 \times 512$ points with $\Delta x = \Delta y = \Delta z = 2.3$ μm, using the numerical schemes described in section 2.2 of [21]. The time step is $\Delta t = 0.3$ ps. The Poisson equation for the electric field is solved in all models at each time step with the same fast elliptic solver FISHPACK [30].

In both particle and hybrid simulation, the particle model with single electrons is used in the early stages, until the number of electrons reaches 2×10^7; this occurs at about 0.46 ns. At this time, the particle model switches to super-particles, while the hybrid model switches to the full hybrid scheme: the fluid model is applied inside the streamer channel where the electric field is less than 0.95 E_b or where the electron density is larger than 0.7 $n_{e,max}$, and the particle model in the remaining part of space; here E_b stands for the background field and $n_{e,max}$ is the maximal electron density in the complete simulation volume. When the hybrid model is activated, electrons inside the streamer channel are removed from the particle list, and the particle model is only applied at the streamer head where it continues to trace all single electrons. In the particle model, on the other hand, super-particles are introduced at the time 0.46 ns by removing at random every second electron, and by doubling the weight of the remaining particles.

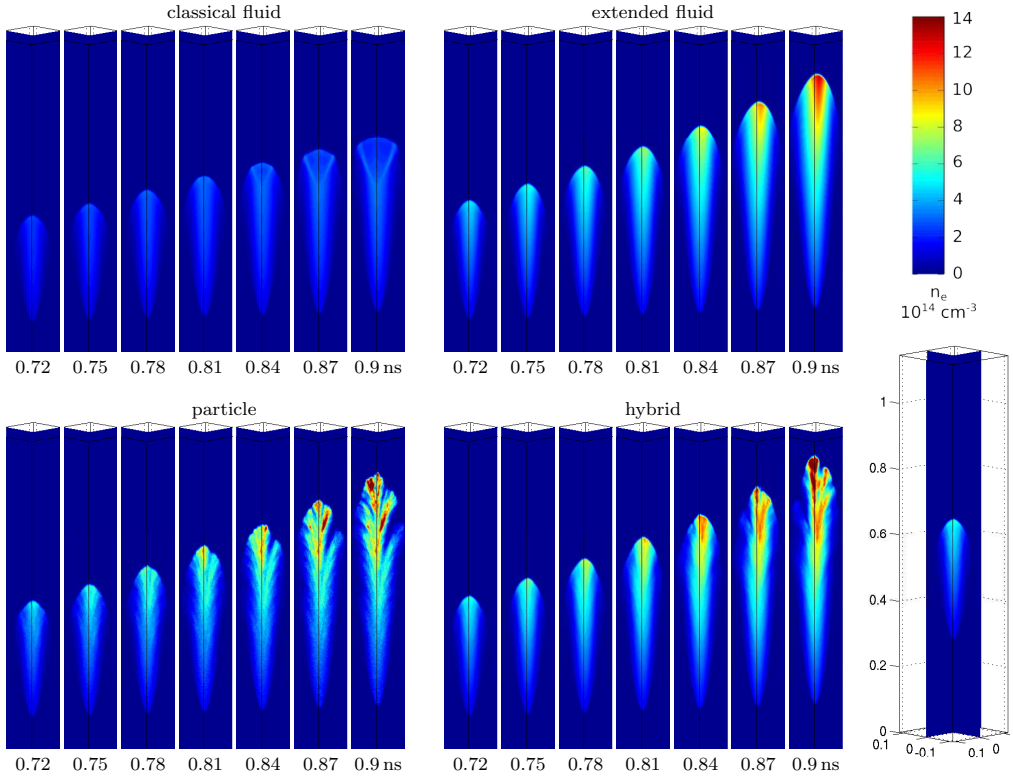

Figure 3.1: The electron density over time in the four simulation models. Densities are plotted on two orthogonal planes intersecting the 3D domain. On the right the full height of the simulation domain (1.17 mm) is shown for the 0.72 ns hybrid model case.

3.3.3 Overview of simulation results for the four models

Figure 3.1 shows the evolution of the electron density from time 0.72 ns up to 0.9 ns, with time increments of 0.03 ns, and figure 3.2 shows the the electric fields at 0.72 ns and at 0.9 ns. The electron density and the electric field on the z-axis are shown in figure 3.3, also at 0.72 ns and 0.9 ns.

Fluctuation and destabilization effects can be seen more clearly in figure 3.4 that zooms into the propagating streamer heads. The electron density, electron minus ion density and electric field are shown at $t = 0.72$, 0.81 and 0.9 ns for the extended fluid model, the particle model and the hybrid model.

3.3.4 Streamer propagation in the four models

At the earlier stage of 0.72 ns, figures 1 and 2 show that a streamer has emerged and grown to about the same length in all models, and that it has approximately the same radius and field enhancement at the tip. The streamer in the classical fluid model (upper row) has stayed a bit behind, and electron and charge density are lower, though the field enhancement is still similar. The propagation differences can be seen more clearly in figure 3.3, which shows the electron density and electric field on the z-axis at 0.72 ns and 0.9 ns. At 0.72 ns, the profiles of extended fluid, particle and hybrid model are about the same, but the classical fluid model has a lower field enhancement, lower electron density and shorter propagation length. That the streamer in the classical fluid model grows more slowly both in space and in electron density, while the other models have comparable results, is also reflected in the total number of electrons: it is $(5.3 \pm 0.2) \times 10^8$ in particle, hybrid and extended fluid model (more precisely 5.1, 5.6 and 5.3×10^8), while it is only 2.9×10^8 in the classical fluid simulation. We

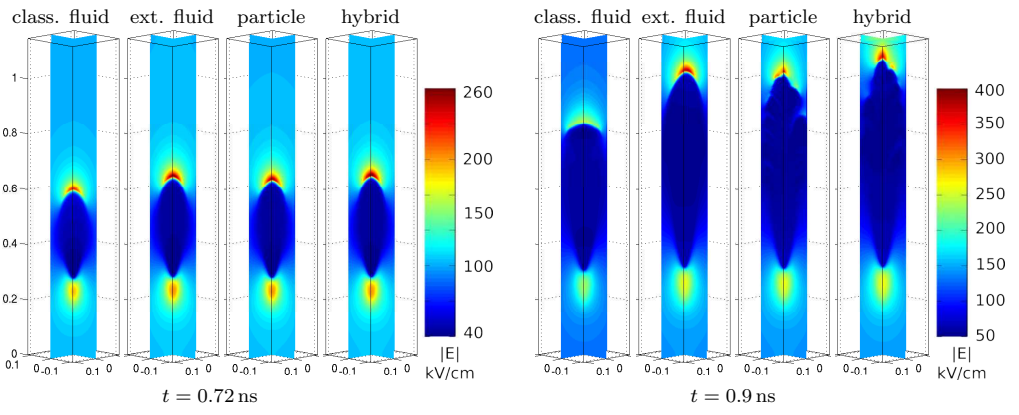

Figure 3.2: The electric field strength in the four models at $t = 0.72$ and 0.9 ns. The full height of the simulation domain is shown (1.17 mm).

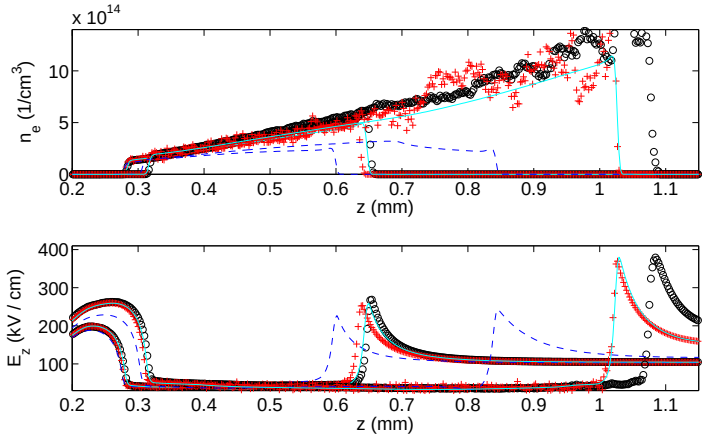

Figure 3.3: Electron density (upper plot) and electric field strength (lower plot) on the vertical z-axis for the same two time steps 0.72 ns and 0.9 ns as in figure 3.2. The different models are classical fluid model (dashed dark blue line), extended fluid model (solid light blue line), particle model (red crosses), and hybrid model (black circles).

will discuss the dynamics of the classical fluid approximation in more detail in section 3.4.1.

At this stage, the particle model uses 1.36×10^7 super-particles with a weight of 32 real electrons, while the hybrid model follows 2×10^7 real electrons and leaves the rest to the fluid region. The super-particles in the particle model already create visible fluctuations of the space charge density, as discussed earlier in [10]. The fluctuations of the local field create numerical heating, and this effect increases as time evolves. Such numerical artifacts can be somewhat suppressed if the super-particles are formed adaptively using particle coalescence techniques [36, 96–98]. (This is discussed in detail in chapter 4.)

But as the number of particles increases, the increased fluctuations will affect the simulations, especially for negative streamers where perturbations in the electron density can grow as they move outwards. The hybrid model does not suffer from such artifacts.

3.3.5 Front destabilization in the four models

During the further evolution up to time 0.9 ns, the streamer ionization front destabilizes in three of the four models, but in characteristically different manners.

The streamer in the classical fluid model is destabilizing into off-axis branches, which are quite symmetric (as we expect in the deterministic fluid model, and as we have seen previously in the simulations of Montijn *et al.* [26]). The actual

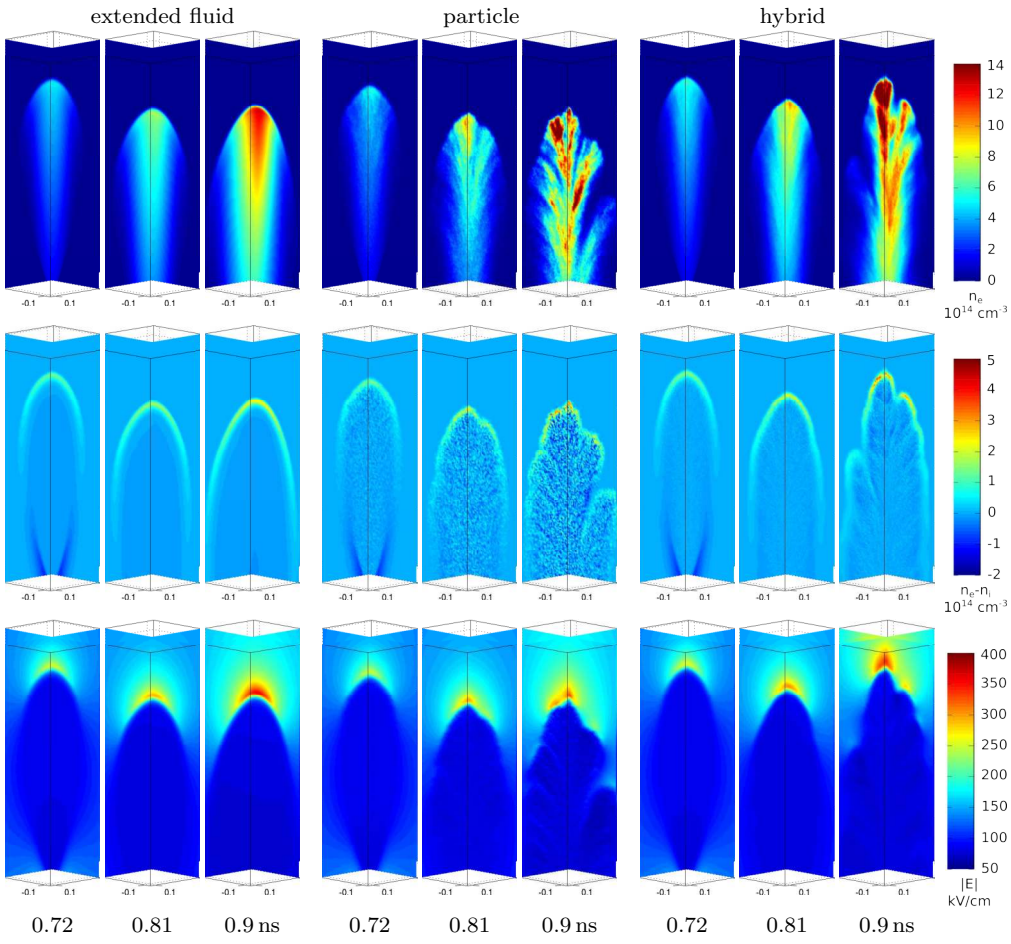

Figure 3.4: Zoom of the electron density (top row), electron minus ion density (middle row) and electric field strength (bottom row) at $t = 0.72$, 0.81 and 0.9 ns. The vertical window shifts upwards with 2.3 mm/ns; it reaches the interval of 0.72 to 1.12 mm at the last time step.

branching can also be seen in the plot for time 0.9 ns in figure 3.3: the electron density on the z-axis in the classical fluid model starts to decrease for $z > 0.69$ mm. As the ionization density is determined by the electric field at the front at the moment when it passed that particular position, the electric field at the front increases until the position 0.69 mm is reached, and decreases thereafter on the axis as the lateral protrusions grow and screen the electric field on the axis.

The streamer in the extended fluid model propagates in a stable manner until the last time step 0.9 ns. The off-axis branching of the classical fluid model is suppressed by the higher ionization rates on the axis in the extended fluid model, that are due to the gradient correction in the ionization term. However, it cannot be excluded a priori that the streamer later destabilizes along a different mode.

The streamers in the particle and in the hybrid model both clearly show density fluctuation effects, both at the front and in the interior charge density. The fluctuations in the particle model are unphysical due to super-particle artifacts while the fluctuations in the hybrid model stay physical. The fluctuations at the front destabilize the streamer into several branches in both models, but the streamer tips close to the axis keep the strongest field enhancement and screen new branches up to the end of the simulation at 0.9 ns. It should be mentioned here that the hybrid model operates with real particles up to 0.69 ns, but has introduced super-particles of weight 8 at 0.9 ns. The weight of the super-particles in the particle model at 0.9 ns is 16 times higher, namely 128.

The extended fluid model and the particle model agree very well in propagation velocity, field enhancement and ionization density, up to the large super-particle fluctuations in the particle model. The hybrid model has similar a ionization density and electric field profile, but is ahead of the other models. At 0.9 ns, there are 3.9, 4.6 and 3.5×10^9 electrons in particle, hybrid and extended fluid model, and only 1.3×10^9 in the classical fluid model.

3.3.6 Computing times

To obtain the results presented here, the computing time for the fluid simulations was about 1 week, for the hybrid simulation about 1.5 weeks, and for the particle simulation about 2 weeks (all on an Intel Q6600 2.4 GHz quadcore processor). The cost of solving the Poisson equation at every time step dominated the total computational cost, therefore computing times are similar for the different models. All the simulations ran sequentially on a single core.

3.4 Discussion of the results

3.4.1 Classical fluid model

Our results show that the streamers in the classical fluid model develop lower velocities, field enhancement and ionization densities than in the other models; the model clearly approximates the microscopic dynamics quite badly, as also found previously in [76]. This is the case even though the transport and reaction coefficients were derived from swarm simulations in the particle model for consistency, and though bulk coefficients were used. As the fields do not exceed 250 kV/cm in the simulations with the classical model, the fluid coefficients were only used in the parameter range in which they were actually derived in [21]. By construction, the classical fluid model with bulk coefficients models electron swarms in a constant electric field well, and earlier numerical studies as well analytical arguments have shown that also the velocity of a planar front in a fixed electric field is well approximated [20, 76]. However, the ionization density behind a planar front in a fixed field is too low in the classical fluid model with bulk coefficients when the maximal field exceeds 50 kV/cm [76]; this is always the case in the present calculations.

In the 3D simulations, the deviation from the other models is larger than in 1D [76]. We argue that this is because in our 1D front simulations, the electric field ahead of the front is fixed, while in 3D the field falls off with distance and varies in time. As the same field in the front creates a lower ionization density in the classical fluid model, also the conductivity in the streamer channel and the consecutive electric screening are lower than in the other models. Therefore the field enhancement is less in 3D and leaves an even lower ionization density behind as the ionization level depends on the field at the front. The lower field enhancement also explains why the streamer is slower than in the other models.

Choosing flux rather than bulk coefficients had not resolved the discrepancy with other models either, as already discussed in section 2.2. The model with flux coefficients does not reproduce the swarm results in a constant electric field. Furthermore, with flux coefficients the electron mobility had been considerably lower (cf. figure 3 in [21]); therefore the front had been even slower than with bulk coefficients.

The streamers within the classical fluid model destabilize and branch at about the same time as in hybrid and particle model, but in a more symmetric manner, as the destabilization is not supported by electron density fluctuations. This deterministic branching in a fluid model is well approximated by moving boundary models as studied in [27] and reviewed in [24].

We remark that the front destabilization in the present fully three-dimensional simulations of the classical fluid model for negative streamers without photoionization occurs in a very similar manner as in previous high accuracy calculations under the constraint of cylindrical symmetry [26]. Earlier simula-

tions [25] demonstrated the mechanism but suffered from lower numerical resolution. In [26], streamers in the same background field were studied, but with a more ionized initial condition and attached to a planar electrode. In those simulations, the branching instability occurred after 1.09 ns, while here it occurs after 0.81 ns. The somewhat longer evolution time until branching in [26] could be due to the different initial and boundary conditions, or to the less accurate transport and reaction coefficients in [26] or to the symmetry constraint; this is subject of future research. In any case, the results support the argument given in [99] that the branching time under the symmetry constraint is an approximation and upper bound of the branching time in the fully 3D calculation.

3.4.2 Extended fluid model, particle model and hybrid model

The extended fluid model was constructed to cure the deficiencies of the classical model. It was shown already in [20, 76] that with the extension in the reaction term and with flux coefficients, it approximates the growth and propagation of particle swarms and of planar streamer fronts well, including the ionization density behind an ionization front. It should be noted though that the reaction and transport coefficients are used here for up to 400 kV/cm while they were derived only for up to 250 kV/cm in [21] and extrapolated to higher field values. The hybrid model uses the fluid coefficients only in the range in which they were derived.

We already discussed above that destabilization into off-axis branches in the extended fluid model is less likely than in the classical fluid model, but we have currently no explanation why branching does not occur at all; possibly this is a mere coincidence and branching does occur at some time after the end of the present simulations at 0.9 ns. The front destabilization in particle and hybrid model occurs at a similar time as in the classical fluid model, but in a different manner: the fastest propagating branch stays close to the axis and screens the other branches that therefore keep staying behind.

The figures show that the extended fluid model approximates particle and hybrid model well up to the moment of destabilization, after this moment there are characteristic differences due to the density fluctuation effects caused by the discreteness of the electrons. These fluctuations have an unphysical distribution when the particle model needs to use super-particles, and therefore the hybrid simulations should be closer to the true dynamics. The stronger destabilization in the particle model seems to be compensated by the increased noise in the space charge distribution (see figure 3.4), so that in the end the front moves with essentially the same velocity as in the extended fluid model.

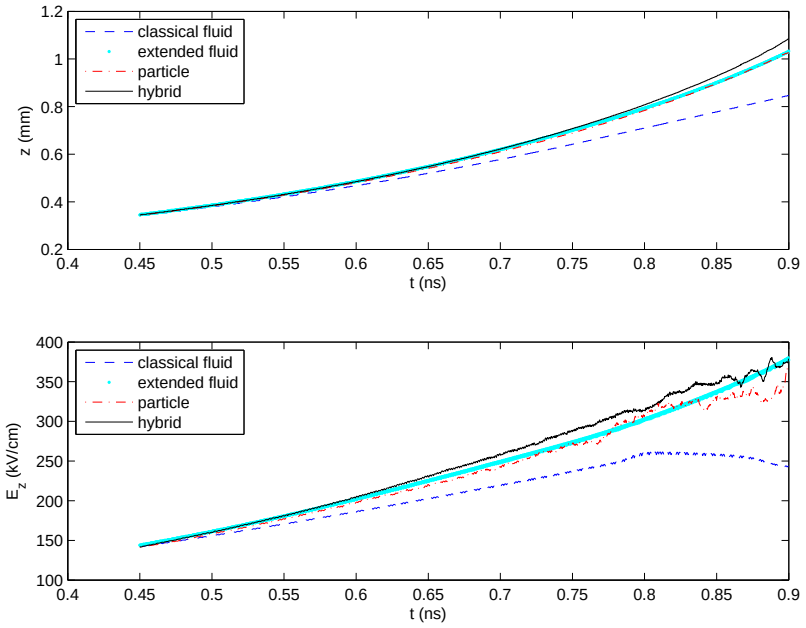

Figure 3.5: Upper panel: Front position as a function of time for the four models. The front position is here defined as the position of the maximum of the electric field on the axis within the ionization front; this maximum of the electric field as a function of time is plotted in the lower panel, also for the four models.

3.4.3 The front velocity in the different models compared to an analytical result

The front positions (see figure 3.5a) and the density and field profiles on the axis (see figure 3.3) agree very well between the extended fluid model and the particle model even at the latest stages, when the particle model shows strong fluctuation effects; the streamer in the hybrid model is a bit faster at the latest stages. This could be due to two different reasons:

- The single particle fluctuations resolved in the hybrid model cover rare electron run-away effects better. This could create more new avalanches ahead of the front, so that eventually the front jumps forward when the electron density within the avalanches has increased sufficiently. In this case the front would be moving faster than expected from reaction, drift and diffusion in the local electric field.

- Or the single particle fluctuations create more branching and thinner streamers with more field enhancement. The front would then propagate faster because the local field is higher, and not because electrons run away.

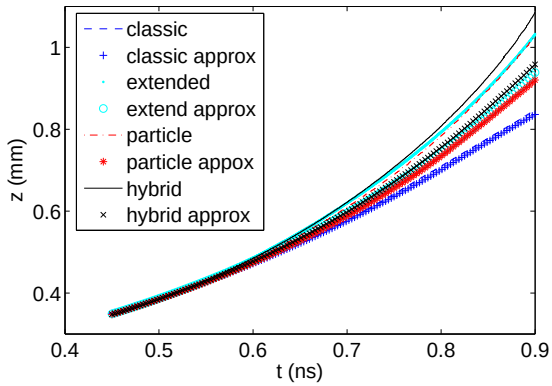

Figure 3.6: Position of the streamer front as a function of time for the four models. Dashed, dotted, dashed-dotted and solid lines indicate the position determined from the simulations, as plotted previously in figure 3.5a. The extended symbols (crosses and circles) indicate the positions for the four models determined through Eq. (3.1) where the maximal electric fields from figure 3.5b were inserted and flux coefficients for $\mu_e(E)$, $D_e(E)$ and $\alpha(E)$ were used.

The maximal electric field as a function of time is plotted in figure 3.5b. Comparison with the front position in figure 3.5a already points to the second statement: where the hybrid model is ahead of the other models, the maximal field enhancement is higher as well. The question is further analyzed in figure 3.6. For planar fronts in a slowly varying electric field E and with a sufficiently rapidly decay of the electron density ahead, the front velocity in the classical fluid model is given by [100]

$$v^* = \mu_e|E| + 2\sqrt{D_e\mu_e|E|\alpha}, \tag{3.1}$$

where μ_e is the electron mobility, D_e the electron diffusion constant and α the effective ionization coefficient. We now use this equation for all models, not only for the classical fluid model. We insert the maximal electric fields $E(t)$ on the z-axis of the respective models into this equation and evaluate it with our flux coefficients for $\mu_e(E)$, $D_e(E)$ and $\alpha(E)$. As the front velocity and the maximal field in the hybrid and particle model fluctuate heavily during the late stages, we do not compare velocities, but the resulting front displacements of the models in figure 3.6. Here the lines indicate the simulation results and the extended symbols the front displacement as predicted by Eq. (3.1).

Up to time 0.6 ns and a maximal field of about 200 kV/cm, the models agree very well with the analytical approximation of (3.1). During the further evolution, the classical fluid model shows almost no deviation from (3.1). (We should remark that this is somewhat accidental, as the classical fluid model uses bulk coefficients and the approximation flux coefficients.) For extended fluid model, particle model and hybrid model, the deviations follow a similar trend

at later times: the simulation models are always a bit ahead of the analytical approximations, and in both the hybrid model is ahead of particle and extended fluid model.

One can conclude (i) that the hybrid model is ahead of the others because the field enhancement is higher, and (ii) that equation (3.1) is a reasonable approximation of the front velocity in all models, but somewhat too low at higher fields. This might be related to the fact that the transport and reaction coefficients were extrapolated from 250 kV/cm to higher fields.

Some high energy electrons can 'run away' from the front in the hybrid and in the particle model. The first electrons with energies above a typical run-away threshold of 200 eV appear at time 0.585 ns in the hybrid model, when the maximal electric field reaches 190 kV/cm, in agreement with the results in [21]. In the particle model the first electrons pass the 200 eV threshold a bit later, at time 0.65 ns. At the end of the simulation the energies of some electrons exceed 1 keV both in hybrid and in particle model. The role of electron run away for the front speed will be investigated further in future work, but it does not seem to have a significant influence on the results presented here, according to our velocity analysis above.

Finally, it should be noted that we used the maximal electric field on the axis in our analysis, while after front destabilization at 0.75 ns, the true maximum fluctuates over some near-axis positions.

3.5 Summary and outlook

3.5.1 Summary

We have tested four 3D models for negative streamers in air without photo-ionization in overvolted gaps. Particle models have advantages when rare events are significant, like electron run-away or like ionization avalanches created by single electrons. With the hybrid model we could lower super-particle weights than with the pure particle model, which limited artificial noise. The extended fluid model is a good approximation up to the moment of front destabilization, but lacks realistic fluctuations. In the classical fluid model propagation speeds and ionization densities and field enhancement are too low.

We have studied short negative streamers in high fields, because this allows us to run physically meaningful 3D simulations with all models, without the need to introduce (adaptive) grid refinement; at this moment, adaptive grid refinement is only available in our fluid model [19], but introducing it across the model boundary in the hybrid model still poses a major challenge. This forced us to exclude photo-ionization from the model for reasons discussed below in the outlook, and in that sense our present results directly apply to discharges in gases like high purity nitrogen [87, 88] or like the atmosphere of Venus [101] where photo-ionization is very weak.

3.5.2 Outlook on physical implications

As the presented investigations do not include photo-ionization, the question rises how they change when the nonlocal photo-ionization mechanism is included for realistic models in atmospheric air. Basically this depends on whether the background field is below or above the breakdown field, see chapter 8.

Now our choice of system parameters was constraint by the fact that we had to resolve both the space charge structure of the ionization front and the complete system with developed streamer in 3D without grid refinement. Therefore we chose a background field well above the breakdown threshold, to initiate the streamer sufficiently rapidly. In such a field, the inclusion of photo-ionization significantly changes the discharge evolution: wherever the nonlocal photo-ionization mechanism creates a new electron ion pair, a new local ionization avalanche will appear. The process was illustrated with the 3D hybrid model in Fig. 1 of [102] and in Fig. 14 of [21], and with Luque's density fluctuation model in Fig. 2 of [28]. The figures show how gradually the whole space fills up with new avalanches, that can suppress the field enhancement at the streamer tip.

Depending on the initial conditions, eventually the whole region above the breakdown field can be filled with plasma. How and when this happens is investigated in chapters 6 – 8. In experiments, a related phenomenon was recently observed: the formation of an ionized cloud in the high-field region around a needle electrode. These 'inception clouds' break up into streamers only beyond some critical radius, as shown experimentally in [33, 103]. In chapter 5, the formation and destabilization of inception clouds is studied with a 3D particle model.

The onset of branching of positive streamers in air with photo-ionization in an undervolted gap was recently investigated in [28] with a model accounting specifically for density fluctuations. There it was found that the density fluctuations that are due to the discrete nature of electrons, accelerate streamer branching. Similarly, the present results show that the fluctuations of electron densities and energies destabilize the ionization front earlier than without fluctuations, but this does not need to create permanent branching in negative streamers. These observations open up many questions: Are negative streamers more self-stabilizing than positive ones as various experiments also seem to indicate? Which differences are caused by over- or undervolted gaps or by the presence or absence of photo-ionization, next to the polarity differences? Which role is played by runaway electrons? Our study lays a methodological basis for future studies of these questions.

Chapter 4

Controlling the weights of simulation particles: adaptive particle management using k-d trees

In particle simulations, the weights of particles determine how many physical particles they represent. Adaptively adjusting these weights can greatly improve the efficiency of the simulation, without creating severe nonphysical artifacts. We present a new method for the pairwise merging of particles, in which two particles are combined into one. To find particles that are 'close' to each other, we use a k-d tree data structure. With a k-d tree, close neighbors can be searched for efficiently, and independently of the mesh used in the simulation. The merging can be done in different ways, conserving for example momentum or energy. We introduce probabilistic schemes, which set properties for the merged particle using random numbers. The effect of various merge schemes on the energy distribution, the momentum distribution and the grid moments is compared. We also compare their performance in the simulation of the two-stream instability.

This chapter has been published as [104]:
Controlling the weights of simulation particles: adaptive particle management using k-d trees, J. Teunissen and U. Ebert, *J. Comput. Phys.* 259, 318 (2014)

4.1 Introduction

Particle-based simulations are widely used, for example to study fluid flows or plasmas. The physical particles of interest are often not simulated individually, but as groups of particles, called *super-particles* or *macro-particles*. Most systems contain so many particles that simulating them individually would be very slow or impossible. And for many macroscopic properties of a system, individual particle behavior is not important. On the other hand, a sufficient number of particles is required to limit stochastic fluctuations.

The weight of a simulation particle indicates how many physical particles it represents. Traditionally, particles had a fixed weight [67, 105]. More recently, Lapenta and Brackbill [96, 106, 107], Assous et al. [108], Welch et al. [98] and others have introduced methods that adapt the weight of particles during a simulation. As discussed in [108], adaptive methods have significant advantages if:

1. Many new particles are created in the simulation. Adaptive re-weighting is required to limit the total number of particles. Examples can be found in [11] and [21].

2. The system has a multiscale nature. In some regions more macro-particles are required, especially if some type of mesh refinement is employed, see for example [109].

3. Control is needed over the number of particles per cell, for example to limit stochastic noise to a realistic value.

Our motivation for investigating the adaptive creation of super-particles originated from the simulation of streamer discharges, as these discharges have both a multiscale nature and strong source terms [21, 110].

When changing the weights of simulation particles, the goal is to reduce the number of simulation particles while not altering the physical evolution of the simulated system. Most methods operate on a single grid cell at a time; arguments for this approach are given in [107]. There are different ways to change the number of particles. One option is to merge two (or sometimes three) particles, to form particles with higher weights. Reversely, splitting can be performed to reduce weights. Another option is to replace all the particles in a cell by a new set of particles, with different weights. We will use the name 'adaptive particle management', introduced in [98], for all such algorithms.

We present a technique for the merging of particles, that extends earlier work of Lapenta [96]. This method can operate independently of the mesh, and in any space dimension. The main idea is to store the particle coordinates (typically position and velocity) in a k-d tree. A k-d tree is a space partitioning data structure that given N points enables searching for neighbors in $O(\log N)$ time [111]. We can then efficiently locate pairs of particles with similar coordinates, and

these pairs can be merged. Because the merged particles are similar, the total distribution of particles is not significantly altered.

In section 4.2, we briefly discuss the general principles of particle management and k-d trees. The implementation of the new particle management algorithm is discussed in section 4.3, where we also introduce different ways to merge particles, which we call 'merge schemes'. In section 4.4, we compare how the merging of particles affects the particle distribution function for two test distributions. We also study the effect on the grid moments, such as the particle density, and compare different ways of constructing a k-d tree. As a more practical example, we show how the different merge schemes affect the evolution of the two-stream instability.

4.2 Adaptive particle management and k-d trees

As stated in the introduction, it is typically impossible to simulate all the physical particles in a system individually. Therefore super-particles are used, representing multiple physical particles. Often, the simulation can run faster or give more accurate results if the weight of these super-particles is controlled adaptively. Different names have been introduced for these algorithms: 'adaptive particle management' [98], 'control of the number of particles' [107], 'particle coalescence' [108], 'particle resampling' [11], 'particle remapping' [61], 'particle rezoning' [96], '(particle) number reduction method' [112] and probably others. There seem to be many independent findings, with independent names. We will use the name 'adaptive particle management' (APM), introduced in [98], to describe this class of algorithms.

4.2.1 Conservation properties

If weights of particles are adjusted, then the 'microscopic details' of a simulation are changed. But the relevant macroscopic quantities should be conserved as much as possible. To specify these macroscopic quantities, we consider a very common type of particle simulation: the particle in cell (PIC) method, also known as the particle mesh (PM) method [67, 105]. In PIC simulations, particles are mapped to moments on a grid. From the grid moments the fields acting on the particles are computed, and the particles move accordingly. For example, in an electrostatic code, the charge density is used to compute the electric field.

An APM algorithm typically changes a set of N_{in} particles to a new set of N_{out} particles. If the two sets give rise to the same grid moments, they give rise to the same fields. Therefore, most algorithms are designed to (approximately) conserve the relevant grid moments.

Only conserving the grid moments is not enough, because the dynamics of a system are not fully determined by the fields. For example, the results of a

simulation can be very sensitive to changes in the momentum or energy distribution. Therefore, some methods try to preserve the shape of these distributions. More generally, we would like to keep the important aspects of the particle distribution function $f(\boldsymbol{x}, \boldsymbol{v}, t)$ the same. The changes to $f(\boldsymbol{x}, \boldsymbol{v}, t)$ should not be significantly larger than the fluctuations that naturally occur. For example, in a collision dominated plasma, particles frequently change direction. Not conserving the momentum distribution in each direction might have little effect on the overall results. But for a collisionless plasma, a change in the momentum distribution might lead to significant differences. Similarly, due to the finite number of particles, fluctuations in the local particle density occur naturally. Therefore, keeping the particle density exactly the same on each grid point might not be necessary, as long as the total number of particles is conserved.

4.2.2 Merging and splitting particles

A set of N_{in} particles can be transformed to a new set of N_{out} particles in many ways. If $N_{\text{in}} > N_{\text{out}}$, we use a pairwise coalescence algorithm, that merges two particles into a single new one. Compared to algorithms that transform multiple particles at the same time, pairwise coalescence has two advantages. First, it is a more local operation, because only closest neighbors in phase space are selected. This ensures that the distribution of particles is not changed very much. Second, it involves fewer degrees of freedom, which makes it simpler to set the properties for new particles. The pairwise coalescence of particles is illustrated in figure 4.1.

In D dimensions, the momentum $\boldsymbol{p}$ of the new particle has D degrees of freedom. Imposing momentum and energy conservation puts $D+1$ constraints on $\boldsymbol{p}$. Therefore, it is in general not possible to conserve both energy and momentum in pairwise coalescence. This means that there is no single best way to merge particles, as different applications require the conservation of different properties. We consider several coalescence schemes, which are discussed in section 4.3.2.

The situation would be very different if N_{in} particles are merged at the same time to form multiple new particles. We still have $D + 1$ constraints, but now $D \cdot N_{\text{out}}$ degrees of freedom in the momenta of the N_{out} new particles. The system is under-determined, and additional information about the particles has to be used. This leads to more complicated algorithms, see for example [98, 108].

If $N_{\text{in}} < N_{\text{out}}$, particles have to be split. Several methods for particle splitting have been compared by Lapenta in [96]. As shown there, choosing the right splitting method can be important, depending on the type of simulation. Here, we will not consider this problem in detail, as our focus is on the merging of particles. A simple strategy is to split single particles into two new ones with the same properties, but half the weight. This can be viable if the simulation includes random collisions, so that the new particles will undergo different collisions and spread out. If there are no such collisions, the split particles should be separated in position or velocity or both.

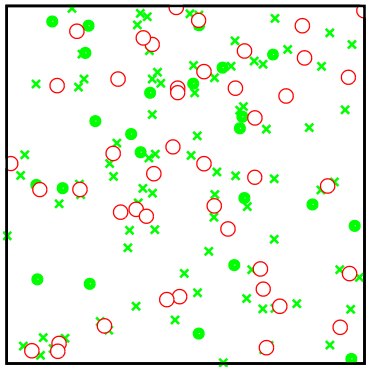

Figure 4.1: Example showing the merging of particles close in space and velocity (velocity is not shown). The particles that were removed after merging are shown as green crosses, particles that were not merged as green filled circles, and the newly formed merged particles as red empty circles. The latter have weight 2, the rest weight 1.

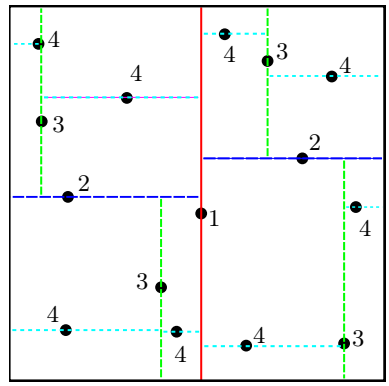

Figure 4.2: Schematic example of how a k-d tree is generated for points in the plane (indicated as black dots). At every step (indicated by the numbers), boxes are split in two parts. The split is located on a point, that is added to the tree. The direction of splitting alternates between vertical and horizontal.

4.2.3 k-d trees

To locate particles with similar coordinates we use a k-d tree [111], which is a space partitioning data structure. A k-d tree can be used to organize a set of points in a k-dimensional space, for any $k \geq 1$. The tree consists of nodes, that contain data (the coordinates of one of the points) and links to at most two 'child'-nodes. The starting point of the tree is the root node, and it contains as many nodes as there are points.

We will briefly explain how such a k-d tree can be generated. To help with the explanation, we let nodes have a **todo** list, that contains points that need to be processed. Suppose we have a collection of points in the (x, y) plane. Initially all points are in the **todo** list of the root node. Then the following algorithm, which is illustrated in figure 4.2, creates the k-d tree:

1. Pick a splitting coordinate, either x or y. A simple choice is to alternate between them.
2. For each node with a non-empty **todo** list:
   (a) Sort the particles in the list along the splitting coordinate. The particle in the middle of the list is the median. If the list contains an even number of particles, pick one of the two middle particles as the median.

(b) The point corresponding to the median is assigned to the node.
(c) The remaining points are moved to the **todo** lists of (at most) two new child nodes. The first one gets the points below the median, the second one those above the median.

3. If there are still points in **todo** lists, go back to step one. Otherwise, the tree is completed.

In k dimensions, the only difference would be that there are now k choices for the splitting coordinate. The computational complexity of creating a k-d tree like this is $O(N \log^2 N)$, with N the number of points in the tree. This can be reduced to $O(N \log N)$ if a linear-time median finding algorithm is used instead of sorting at step 2a. Searching for the nearest neighbor to a location $\boldsymbol{r}$ can be done in $O(\log N)$ time. The basic idea is to first traverse the tree down from the root node, at each step selecting that side of the tree that $\boldsymbol{r}$ lies in. (If $\boldsymbol{r}$ happens to lie exactly on a splitting plane, it is a matter of convention which side to pick.) During the search, the closest neighbor found so far is stored. Then going upward in the tree, at every step determine whether a closer neighbor could lie on the other side of the splitting plane. If so, also traverse that other part of the tree down (but only where it can contain a closer neighbor). Typically, only a small number of these extra traverses is required. When the algorithm ends up at the root node again, the overall closest neighbor is found.

For the numerical tests presented in section 4.4, we have used the Fortran 90 version of the `KDTREE2` [113] library.

4.3 Implementation

We will discuss the implementation of our adaptive particle management algorithm in section 4.3.1. Different schemes that can be used for particle merging are given in section 4.3.2.

4.3.1 Adaptive particle management algorithm

Suppose that we have particles with coordinates $\boldsymbol{x}_i$, $\boldsymbol{v}_i$ and weights w_i. Furthermore, assume there is some function $W_{\text{opt}}(i)$ that gives the user-determined optimal weight for particle i. Then the APM algorithm works as follows:

1. *Create a list **merge** with all the particles for which $w_i < \frac{2}{3} W_{\text{opt}}(i)$, sorted by $w_i/W_{\text{opt}}(i)$ from low to high. Create a list **split** with particles for which $w_i > \frac{3}{2} W_{\text{opt}}(i)$.*
   The function $W_{\text{opt}}(i)$ gives the desired weight for particle i. The factors $\frac{2}{3}$ and $\frac{3}{2}$ ensure that merged particles are not directly split again, and vice versa. A good choice of $W_{\text{opt}}(i)$ will often depend on the application. We typically want to keep the number of particles per cell close to a desired

value N_{ppc}, and use $W_{\text{opt}}(i) = \max(1, N_{\text{phys}}(i)/N_{\text{ppc}})$. Here $N_{\text{phys}}(i)$ denotes the number of physical particles in the cell of particle i.

2. *For the particles in* ***merge****:*
   (a) *Create a k-d tree with the (transformed) coordinates of the particles as input.*
   We construct the k-d trees in two ways: using the coordinates $(\boldsymbol{x}, \lambda_v \boldsymbol{v})$ or using the coordinates $(\boldsymbol{x}, \lambda_v |\boldsymbol{v}|)$, where λ_v is a scaling parameter and $||$ denotes the L^2 norm. We will refer to them as the 'full coordinate k-d tree' and the 'velocity norm k-d tree', and we will denote them with a superscript $^{\boldsymbol{x},\boldsymbol{v}}$ and $^{\boldsymbol{x},|\boldsymbol{v}|}$, respectively. The scaling is necessary because the nearest neighbor search uses the Euclidean distance between points. There is some freedom in the choice of λ_v, which should express the ratio of a typical length divided by a typical velocity. With higher values the differences in velocity become more important than the spatial distances.
   (b) *Search the nearest neighbor of each particle in the k-d tree. If the distance between particles i and j is smaller than $d_{\max}$, merge them. Particles should not be merged multiple times during the execution of the algorithm, so mark them inactive.*
   We let $d_{\max}$ be proportional to the grid spacing Δx, so particles in finer grids need to be closer to be merged. There is no single optimal way to merge two particles. Several schemes for merging are discussed below in section 4.3.2.

3. *Split each of the particles in* ***split*** *into two new particles.*
   The new particles have the same position and velocity as the original particle i, and weights $w_i/2$ and $(w_i + 1)/2$ (both rounded down). As was discussed in section 4.2.2, for some applications a different method should be used.

4.3.2 Merge schemes

When two particles are merged, it is generally not possible to conserve both energy and momentum. Therefore we consider different schemes, that conserve either momentum, energy or other properties. The performance of these schemes is compared in section 4.4. We have not used ternary schemes, that merge three particles into two. As discussed in [96], such schemes do not necessarily perform better, although they can conserve both momentum and energy. Furthermore, they are more complicated to construct in 2D or 3D.

When particles i and j are merged, the weight of the new particle is always the sum of the weights $w_{\text{new}} = w_i + w_j$. For the new position we consider two choices. It can be the weighted average $\boldsymbol{x}_{\text{new}} = (w_i \boldsymbol{x}_i + w_j \boldsymbol{x}_j)/(w_i + w_j)$. It can also be picked randomly as either $\boldsymbol{x}_i$ or $\boldsymbol{x}_j$, with the probabilities proportional

to the weights. If we take the weighted average, then we introduce a (slight) bias in the spatial distribution. On the other hand, picking the position randomly increases stochastic fluctuations. For example, suppose we have a cluster of particles, and particles are being merged until there is only one left. If we use the weighted average position, then we always end up at the center of mass. So the spatial distribution of particles has become very different, a single peak at the center. With the probabilistic method we also end up with a single peak, located at the position of one of the original particles. But now the probability of ending up at particle i is proportional to w_i. Therefore, the 'average' spatial distribution has the same shape as before the merging.

Below we list several schemes for picking a new velocity $\boldsymbol{v}_{\text{new}}$. For convenience of notation, let

$$\begin{aligned}\boldsymbol{v}_{\text{avg}} &= (w_i\boldsymbol{v}_i + w_j\boldsymbol{v}_j)/(w_i + w_j),\\ v^2_{\text{avg}} &= (w_i\,|\boldsymbol{v}_i|^2 + w_j\,|\boldsymbol{v}_j|^2)/(w_i + w_j),\end{aligned}$$

so $\boldsymbol{v}_{\text{avg}}$ is the weighted average velocity and v^2_{avg} is the weighted square norm of the velocity. The schemes are indicated by the following symbols:

p: Conserve momentum strictly by taking $\boldsymbol{v}_{\text{new}} = \boldsymbol{v}_{\text{avg}}$. Because $|\boldsymbol{v}_{\text{avg}}|^2 \leq v^2_{\text{avg}}$, the kinetic energy is reduced by an amount $\frac{1}{2}mw_{\text{new}}\left(v^2_{\text{avg}} - |\boldsymbol{v}_{\text{avg}}|^2\right)$, where m is the mass of a particle with weight one.

ε: Conserve energy strictly by taking $\boldsymbol{v}_{\text{new}} = \sqrt{v^2_{\text{avg}}}\cdot\hat{\boldsymbol{v}}_{\text{avg}}$ (the hat denotes a unit vector). Because the energy is kept the same, the momentum increases by $mw_{\text{new}}\left(\sqrt{v^2_{\text{avg}}} - |\boldsymbol{v}_{\text{avg}}|\right)\cdot\hat{\boldsymbol{v}}_{\text{avg}}$.

$\boldsymbol{v}_r$: Conserve both momentum and energy on average, by randomly taking the velocity of one of the particles. The probability of choosing the velocity of particle i is proportional to its weight w_i.

$\boldsymbol{v}_r\varepsilon$: Randomly take the velocity of one of the particles, but scale it to strictly conserve energy. The expected change in momentum is

$$mw_{\text{new}}\left(\sqrt{v^2_{\text{avg}}}(w_i\hat{\boldsymbol{v}}_i + w_j\hat{\boldsymbol{v}}_j)/w_{\text{new}} - \boldsymbol{v}_{\text{avg}}\right),$$

which is small if $|\boldsymbol{v}_i| \approx |\boldsymbol{v}_j|$.

Although they are quite simple, we are not aware of other authors that have used schemes with randomness. It is possible to use multiple schemes, where the choice of scheme depends on the properties of the particles to be merged.

4.4 Numerical tests and results

It is difficult to come up with a general test of the performance of an APM algorithm. The algorithm should not significantly alter the simulation results, compared to a run without super-particles. At the same time, it should decrease the computational cost as much as possible. But whether these criteria are met depends on the particular simulation that is performed. Therefore we first perform tests on a simplified 2D system, using two Gaussian velocity distributions. In these tests we do not study the time evolution of the system, but focus on the effects of the coalescence algorithm on the particle distribution and on the grid moments. After that, we investigate how these changes in the particle distribution affect the evolution of a 'real' simulation: the two-stream instability in 1D.

4.4.1 Effect of the merge schemes on the energy and momentum distribution

As stated before, our method works in 1D, 2D, 3D or any other dimension. In these tests we use a 2D domain with periodic boundary conditions. The domain consists of 2×2 cells, each of size 1×1. (We let lengths and velocities be of order unity, and give them without a unit.) Initially, particles with weight 1 are distributed uniformly over the domain. Then the coalescence algorithm is performed once, with the desired weight of the particles set to 2. We compare how the different merge schemes change the momentum and energy distribution. We also measure their effect on the density, momentum and energy grid moments.

First test

In the first test, there are 400 particles with a Gaussian velocity distribution. Both components of the velocity have mean 1 and a standard deviation of 1/4. The resulting energy and momentum distribution functions are shown in the top row of figure 4.3. We show the distribution of momentum along the first coordinate, not the total momentum of particles, therefore we label it x-momentum. To convert the velocity of a particle to momentum, we multiply it by the weight of the particle, which represents the mass.

Initially, the particles have weight 1, and a desired weight of 2. Then the particles are coalesced according to a merge scheme, and the changes in the energy, momentum and density distribution are recorded. The whole procedure is repeated 10^5 times for each scheme, using different random numbers, to reduce stochastic fluctuations. We have used both the velocity norm k-d tree (containing $\boldsymbol{x}, \lambda_v |\boldsymbol{v}|$) and the full coordinate k-d tree (containing $\boldsymbol{x}, \lambda_v \boldsymbol{v}$). Somewhat arbitrarily we took $\lambda_v = 4/5$, as the mean velocity plus the standard deviation in velocity was 5/4.

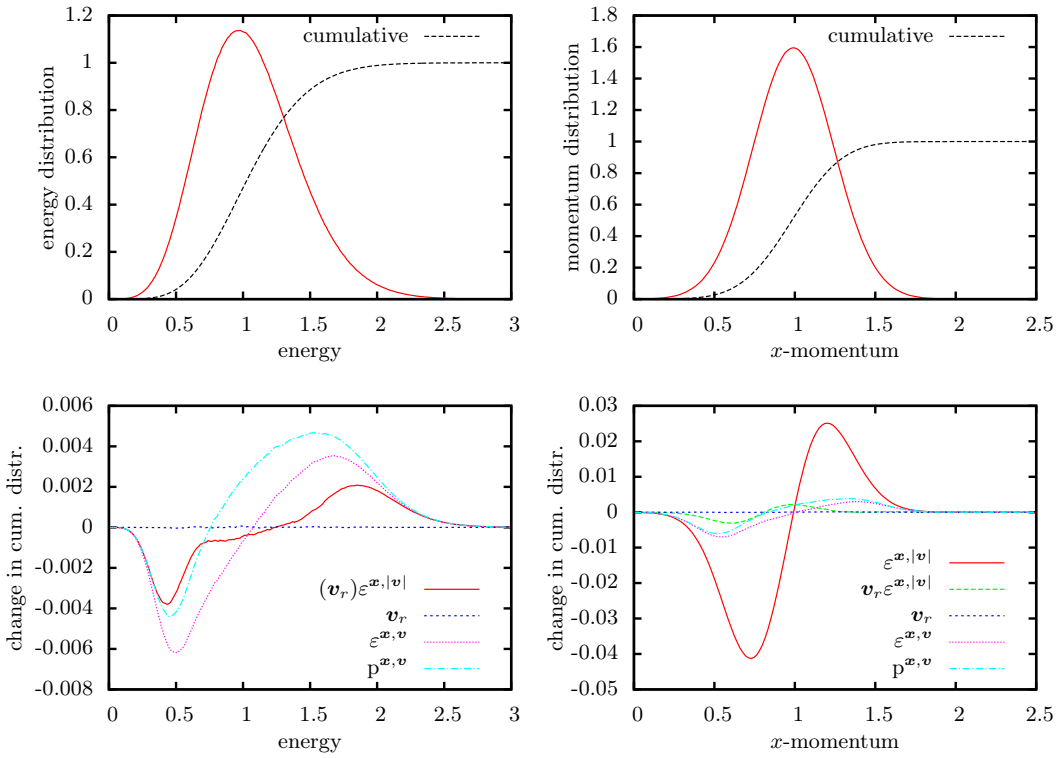

Figure 4.3: Results for the first test. Top row: the initial energy (left) and momentum (right) distribution of the particles. The integrated or cumulative curves are also shown (dashed). Bottom row: the effect of various merge schemes on the cumulative energy (left) and momentum (right) distribution function. The schemes are indicated by the following symbols; ε: conserve energy, p: conserve momentum, $\boldsymbol{v}_r$: conserve energy and momentum on average, $\boldsymbol{v}_r\varepsilon$: take velocity from one of the particles at random, scale to conserve energy, $^{\boldsymbol{x},|\boldsymbol{v}|}$: velocity norm k-d tree, $^{\boldsymbol{x},\boldsymbol{v}}$: full coordinate k-d tree.

The bottom row of figure 4.3 shows the effects of the merge schemes on the cumulative energy and momentum distribution function. The schemes are indicated by the same symbols as in section 4.3.2:

p: conserve momentum
ε: conserve energy
$\boldsymbol{v}_r$: take velocity of one of the particles at random
$\boldsymbol{v}_r\varepsilon$: take velocity from one of the particles at random, scale to conserve energy
$^{\boldsymbol{x},|\boldsymbol{v}|}$: velocity norm k-d tree
$^{\boldsymbol{x},\boldsymbol{v}}$: full coordinate k-d tree

Because they are less noisy and reveal trends more clearly, we present cumulative

differences

$$\Delta F(x) = \int_{x_{\min}}^{x} f_{\text{merged}}(x') - f_{\text{orig}}(x')\, dx', \tag{4.1}$$

where $f_{\text{orig}}(x)$ is the normalized energy or momentum distribution function before merging and $f_{\text{merged}}(x)$ is the distribution after merging.

The schemes $\varepsilon^{\boldsymbol{x},|\boldsymbol{v}|}$ and $\boldsymbol{v}_r\varepsilon^{\boldsymbol{x},|\boldsymbol{v}|}$ have the same effect on the energy distribution, so they are shown together there as $(\boldsymbol{v}_r)\varepsilon^{\boldsymbol{x},|\boldsymbol{v}|}$. The schemes $\boldsymbol{v}_r^{\boldsymbol{x},|\boldsymbol{v}|}$ and $\boldsymbol{v}_r^{\boldsymbol{x},\boldsymbol{v}}$ are also shown together, as $\boldsymbol{v}_r$. They take the new velocity randomly from one of the original particles. Therefore, on average, both do not change the shape of the energy and momentum distribution. The other schemes move particles from the tails of the distribution towards the center. To see this in the cumulative distribution functions, note that particles get removed where the slope is negative, and are moved to where the slope is positive. This happens because these schemes take averages, which are more likely to lie towards the center of the distribution. Results are not shown for the velocity norm k-d tree with the momentum conserving scheme, $\mathrm{p}^{\boldsymbol{x},|\boldsymbol{v}|}$. This combination leads to large changes in the energy distribution.

For all the merge schemes, on average about 40% of the particles is merged. The number is below 50% because the k-d tree is created only once, in a static way. When a particle is merged, it is not removed from the tree, but marked as inactive. So it might later be the nearest neighbor of another particle, that is still to be merged. In that case, the second particle is not merged, and the algorithm moves on to the next particle. Another option would be to search for the second closest neighbor, and so on. But then merging would happen over greater distances towards the end of the algorithm.

Note that even when merging only with the closest neighbor, the order in which the particles are selected for merging can have an effect on the result. For example, suppose the particles are selected based on their energy, so that particles at lower energies are merged first. A particle with high energy now has a lower chance of getting merged, because many of its neighbors of lower energy are already merged. For the same reason, the particle is more likely to be merged with another particle of higher energy. In the adaptive particle management algorithm introduced in section 4.3.1, we therefore sort the particles to be merged by their 'relative weights', from low to high, where 'relative weight' means current weight over desired weight.

Second test

The second test is performed in the same way as the first test, but now the particles have a different velocity distribution. Both components of the velocity have a mean of 1/4 and a standard deviation of 1. The resulting energy and momentum distribution functions are shown in the top row of figure 4.4. Because it is more isotropic, the second velocity distribution poses a bigger challenge for

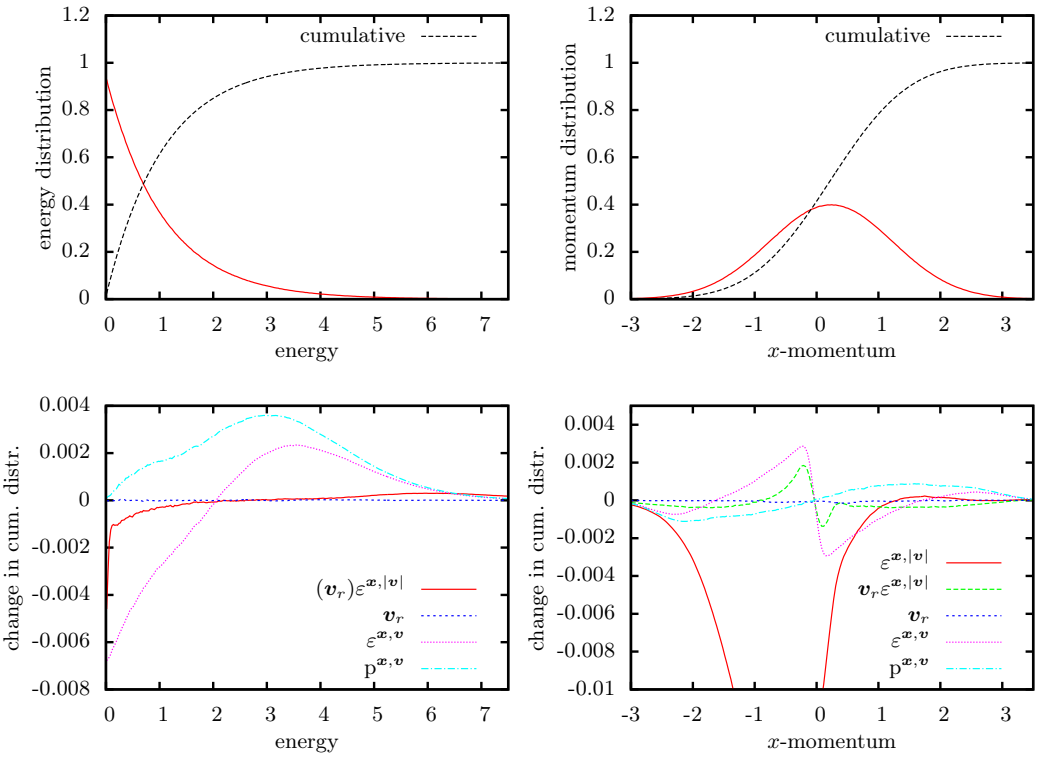

Figure 4.4: Results for the second test. Top row: the initial energy (left) and momentum (right) distribution of the particles. The integrated or cumulative curves are also shown (dashed). Bottom row: the effect of various merge schemes on the cumulative energy (left) and momentum (right) distribution function. The legend is the same as for figure 4.3. In the right figure, the peak for scheme $\varepsilon^{\boldsymbol{x},|\boldsymbol{v}|}$ is cut off, it extends to -0.018.

the merge schemes. The bottom row of figure 4.4 shows the effects of the merge schemes on the cumulative energy and momentum distribution function. Again, the schemes $\boldsymbol{v}_r$ perform best, as the other schemes move particles from the tail of the distribution towards the center. Note that the schemes $\boldsymbol{v}_r\varepsilon^{\boldsymbol{x},|\boldsymbol{v}|}$ and $\varepsilon^{\boldsymbol{x},\boldsymbol{v}}$ also move particles away from zero momentum. As for the first test, on average about 40% of the particles is merged.

4.4.2 Effect on grid moments

In many particle simulations, a grid (or mesh) is used. Grid moments are defined at the grid points, and provide local averages from which the fields acting on the particles can be computed. For example, the first grid moment gives the particle density, the second the current density or momentum density, the third the energy

density and so on. Particles can be mapped to grid moments in different ways, here we use first order interpolation, also know as cloud-in-cell (CIC) [67, 105].

Using the data of the second test, we now look at the effect of the merge schemes on the first three grid moments. An APM algorithm should not induce large differences in these grid moments. The mean difference is often zero, because the corresponding quantity is conserved. Therefore, we also look at the relative standard deviation, or σ/μ, where σ is the standard deviation of a random variable with mean μ. This is a measure of the relative size of fluctuations. We measure these fluctuations at a single grid point, as they would be correlated for multiple grid points. In table 4.1 the changes in the grid moments are given for various schemes. The schemes are labeled by the same symbols as before. In addition, $\boldsymbol{x}_r$ indicates that the new position is picked randomly from one of the merged particles. The bottom part of the table is about cell-by-cell merging, which is discussed in section 4.4.3. The APM fluctuations should be compared to those resulting from advancing the particles in time. Therefore, the table includes entries that list the effect of taking a time step Δt without any merging. Since we have included no collisions, the particles simply move with a constant velocity during this time step.

The average deviation in particle density ρ is zero for all the schemes, because they conserve the total weight of the particles. Therefore this quantity is not included in table 4.1. The induced fluctuations in the grid moments can differ by almost an order of magnitude between the schemes. As expected, conserving momentum reduces the mean energy, and conserving energy increases momentum. This is especially problematic when the velocity norm k-d tree is used. The mean deviations are then larger than 10%. The full coordinate k-d tree in combination with the energy-conserving scheme, $\varepsilon^{\boldsymbol{x},\boldsymbol{v}}$, gives good results regarding energy and momentum conservation. Schemes that select the new velocity at random do not lead to systematic differences in the energy and momentum grid moments. With the $\boldsymbol{v}_r^{\boldsymbol{x},|\boldsymbol{v}|}$ scheme, the fluctuations in momentum can be relatively large. The $\boldsymbol{v}_r^{\boldsymbol{x},\boldsymbol{v}}$ scheme leads to much smaller fluctuations. This scheme performs well: on average it conserves the grid moments and also the shapes of the energy/momentum distribution functions, and it does not create big fluctuations.

Taking the new position at random at one of the original particles ($\boldsymbol{x}_r$) increases the fluctuations in particle density. For all the schemes, the fluctuations in density, momentum or energy are smaller than those resulting from a time step of $\Delta t = 0.4$.

4.4.3 Cell-by-cell merging

Using k-d trees, there is no reason to do cell-by-cell merging. But because this type of merging is commonly used, we briefly evaluate its effects. The bottom part of table 4.1 shows results for cell-by-cell merging, for various schemes, using the second test distribution. The notation is the same as before, and a superscript

Method	N_{merge}	d_{avg}	σ_ρ	Δp_x	σ_{p_x}	$\Delta\varepsilon$	σ_ε
$\Delta t = 0.1$	-	-	1.6%	0.0%	9%	0.0%	3.8%
$\Delta t = 0.2$	-	-	2.9%	0.0%	16%	0.0%	6.7%
$\Delta t = 0.4$	-	-	4.9%	0.0%	25%	0.0%	9.4%
$\varepsilon^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$	39%	0.16	0.3%	12%	16%	0.0%	0.8%
$\mathrm{p}^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$	39%	0.16	0.3%	0.0%	4%	−37%	5.0%
$\boldsymbol{v}_r^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$	39%	0.16	0.3%	0.0%	24%	0.0%	1.2%
$\boldsymbol{v}_r\varepsilon^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$	39%	0.16	0.3%	0.4%	25%	0.0%	0.8%
$\boldsymbol{v}_r\boldsymbol{x}_r^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$	39%	0.16	1.0%	0.0%	24%	0.0%	2.2%
$\varepsilon^{\boldsymbol{x},\boldsymbol{v}}$	40%	0.38	0.7%	0.1%	4%	0.0%	1.5%
$\mathrm{p}^{\boldsymbol{x},\boldsymbol{v}}$	40%	0.38	0.7%	0.0%	4%	−1.2%	1.5%
$\boldsymbol{v}_r^{\boldsymbol{x},\boldsymbol{v}}$	40%	0.38	0.7%	0.0%	6%	0.0%	2.4%
$\mathrm{p}^{\boldsymbol{v}}$, cell	40%	0.19	2.8%	0.0%	12%	−0.9%	3.8%
$\boldsymbol{v}_r^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$, cell	39%	0.17	0.3%	0.0%	23%	0.0%	1.5%
$\boldsymbol{v}_r\boldsymbol{x}_r^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$, cell	39%	0.17	1.0%	0.0%	23%	0.0%	2.2%
$\varepsilon^{\boldsymbol{x},\boldsymbol{v}}$, cell	38%	0.40	0.8%	0.5%	5%	0.0%	1.8%

Table 4.1: The induced differences and fluctuations in the grid moments by the various merge schemes, using the second test distribution. Legend: N_{merge} is the fraction of merged particles, and d_{avg} is the average distance between merged particles. The relative differences in grid moments are indicated by Δp_x (momentum) and $\Delta\varepsilon$ (energy), and relative standard deviations by σ_ρ (density), σ_{p_x} (momentum) and σ_ε (energy). Both are given relative to the mean value. The rows starting with Δt show the fluctuations in the grid moments resulting from a time step (no merging). The merge schemes are indicated by the following symbols; ε: conserve energy, p: conserve momentum, $\boldsymbol{v}_r$: random velocity, $\boldsymbol{v}_r\varepsilon$: random velocity, scale to conserve energy, $\boldsymbol{x}_r$: random position, $^{\boldsymbol{v}}$: k-d tree contains only the velocity, $^{\boldsymbol{x},\lvert\boldsymbol{v}\rvert}$: velocity norm k-d tree, $^{\boldsymbol{x},\boldsymbol{v}}$: full coordinate k-d tree, cell: perform the merging cell-by-cell.

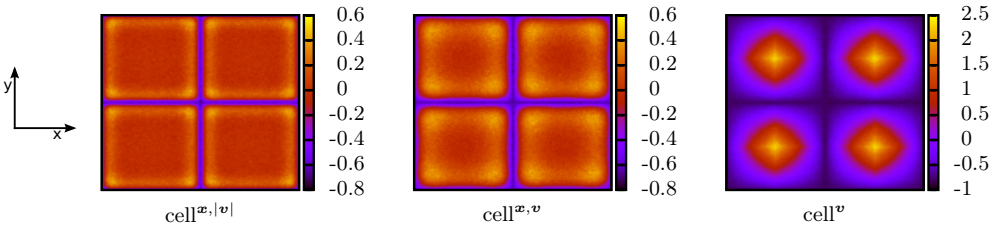

Figure 4.5: The relative change in particle density as a result of cell-by-cell merging, for the test case described in 4.4.1. The 2×2 grid structure is clearly visible. On the left, close neighbors for merging are searched for using the difference in position and in the norm of the velocity. In the middle, all four position and velocity coordinates are used, while on the right only the differences in velocity are considered for merging. Especially in the last case, particles are moved to the center of the cell as they are merged, because it is the expectation value of their mean position.

$^{\boldsymbol{v}}$ indicates that only the velocity was used in the k-d tree, not the position. The fluctuations are mostly similar if the particles are merged locally (cell-by-cell) instead of globally. With fewer particles per cell, the differences would be larger though, as close neighbors are more likely to lie in other cells.

The average spatial distribution of particles directly after merging is shown in figure 4.5. Only the type of k-d tree is important for the effect, because all the shown merge schemes take the average position. From left to right: With the velocity norm k-d tree the spatial distribution of particles is affected close to the cell boundaries. With the full coordinate k-d tree, the effect is similar as with the velocity norm k-d tree. Using the k-d tree that includes only the velocity, particles are moved to the center of the cells. The spatial distribution is severely affected. Furthermore, the fluctuations in particle density are higher, as can be seen in table 4.1. If particles would be merged globally with a k-d tree (not cell-by-cell), the spatial distribution would on average be uniform.

Since the density fluctuations due to cell-by-cell merging occur at length scales smaller than the grid resolution, we do not expect them to have practical consequences for most simulations. But in some cases, for example if spatial features expand in time, they might become important.

4.4.4 Simulation example: the two-stream instability

Above, we have studied the changes in momentum and energy distribution that arise due to the merging of particles. But do these changes in the distribution function affect the physical evolution that one wants to study? The answer will, of course, depend on the type of simulation that is performed. Here, we consider as an example the simulation of the two-stream instability in one dimension [114].

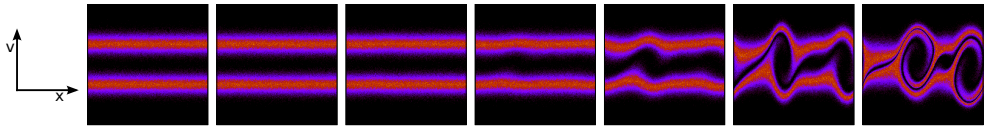

Figure 4.6: The time evolution of the two-stream instability. In each figure, the periodic domain is shown horizontally. Vertically, the density in velocity space is shown. Initially, the two beams of particles are clearly visible. The time t is indicated in inverse plasma frequencies $1/\omega_p$.

To induce this instability we create two beams of particles, that propagate in opposite directions. The particles have the same charge, and are neutralized by a background charge density. A fluctuation in the charge density locally changes the electric field, which affects the beam velocities, which can in turn increase the density fluctuation.

In figure 4.6, an example of the evolution of the two-stream instability is shown. We use periodic boundary conditions, and there are $N_p = 10^6$ particles per beam. The particles have a Gaussian velocity distribution, with a standard deviation or thermal velocity $v_{\text{th}} = 1$ and a drift velocity $v_d = \pm 4$. The spatial grid consists of 10^3 points and has length $L = 1$. To convert the particle line density to a volume density n, scaling by some unit of area is required. We do this in such a way that there are 100 Debye lengths in the domain

$$\lambda_D = \sqrt{\frac{\epsilon_0 k_B T}{n q^2}} = L/100, \tag{4.2}$$

where k_B is the Boltzmann constant, ϵ_0 the permittivity of vacuum and q the charge of the particles. Defining the temperature of the particles in a beam as $T = m v_{\text{th}}^2 / k_B$, with m the mass of the particles, the plasma frequency is then given by

$$\omega_p = \sqrt{\frac{n q^2}{m \varepsilon_0}} = 100 L / v_{\text{th}}, \tag{4.3}$$

or simply $\omega_p = 100$ in dimensionless units. We will give simulation times in inverse plasma frequencies. Note that when defining the simulation parameters like this, the constants used for the mass and charge of the particles do not matter.

We perform two tests to investigate how the merging of particles affects the physical evolution of the system. In the first test, merging starts at $t = 5/\omega_p$, when no instability is yet visible in figure 4.6. In the second test, merging starts later, at $t = 20/\omega_p$, when instabilities have grown to a visible size. The desired weight of the particles is set to 32, and the merging routine is called five times. For the merging, we either use the energy conserving scheme (ε), the momentum

conserving scheme (p), the scheme that picks a velocity at random from one of the original particles ($\boldsymbol{v}_r$) or the scheme that picks a velocity at random but conserves energy ($\boldsymbol{v}_r\varepsilon$). The typical distance between particles in space is $\delta x = L/N_p$, while the typical difference in velocity is $\delta v = v_{\text{th}}$. We construct the coordinates for the k-d tree using $(x, \lambda_v v)$, with $\lambda_v = 10\delta x/\delta v$. The value of λ_v is adjusted as the number of simulation particles changes.

How the physical evolution of the system is affected by the merging is shown in figures 4.7 and 4.8. Figure 4.7 shows simulation results at $t = 60/\omega_p$, when merging was performed at $t = 5/\omega_p$ and figure 4.8 shows results at $t = 90/\omega_p$, for the case of merging at $t = 20/\omega_p$. In both figures, we show seven runs, that differ only in the initial state of the pseudorandom number generator. After merging, there are about $8 \cdot 10^4$ particles per beam, instead of the initial 10^6. The simulation results without any type of merging are also shown, for comparison.

In figure 4.9, we show the velocity distribution just after merging at $t = 5/\omega_p$. The $\boldsymbol{v}_r$ scheme does, on average, not alter the velocity distribution of the particles. The other schemes all take averages in determining the properties of the merged particles, thereby removing the tails from the distribution. However, the effect a merge scheme has *on average* says little about the fluctuations it introduces. Therefore we also include figure 4.10, in which the difference in the velocity distribution as compared to the original simulation is shown over time. More precisely, we show the quantity

$$|\boldsymbol{f}_v(t) - \boldsymbol{f}_{v,0}(t)| \,/\, |\boldsymbol{f}_{v,0}(t)|\,, \tag{4.4}$$

where $\boldsymbol{f}_v(t)$ and $\boldsymbol{f}_{v,0}(t)$ denote the 'merged' and the original velocity distribution function, respectively, that were constructed using 200 bins. (As before, we use $||$ to indicate the L^2 norm.)

From figures 4.7 and 4.10, it can be seen that when merging happens at $t = 5/\omega_p$, the momentum conserving scheme (p) seems to perform best. The schemes that conserve energy (ε and $\boldsymbol{v}_r\varepsilon$) perform almost as good, but the scheme that picks velocities at random ($\boldsymbol{v}_r$) does considerably worse. The reason for this is probably that the $\boldsymbol{v}_r$ schemes induce greater fluctuations in the momentum distribution, see table 4.1. When the two-stream instability has a small magnitude, these induced fluctuations perturb the further evolution of the system.

For the case of merging at $t = 20/\omega_p$ the results look quite different, see figures 4.7 and 4.10. Now, the $\boldsymbol{v}_r$ scheme has the smallest effect on the evolution of the simulation, with the other three schemes performing worse. The larger induced fluctuations of the $\boldsymbol{v}_r$ scheme are probably not as important now, because the two-stream instability has already grown to a larger size before merging.

In the previous sections, we have also presented results for k-d trees that used the norm of the velocity vector. In one dimension, this corresponds to taking the absolute value of the velocity. But because in the two-stream simulation particles flow in two opposite directions, this leads to poor results. Particles from the two

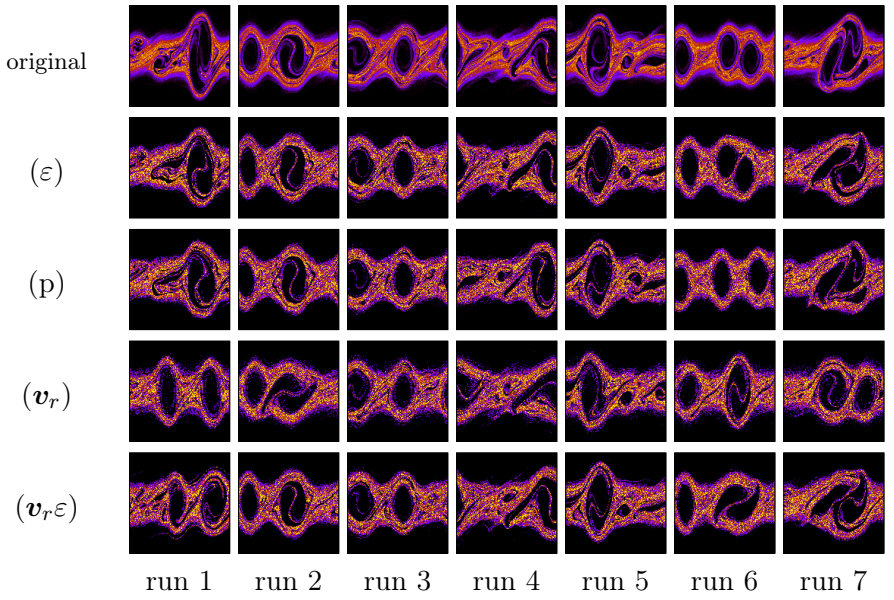

Figure 4.7: Simulation results of the two-stream instability, showing x, v curves at $t = 60/\omega_p$ (see also figure 4.6). The top row shows the original simulation with 10^6 particles per beam. The other rows show results where repeated merging took place at $t = 5/\omega_p$, reducing the particle number to about $8 \cdot 10^4$ particles per beam. The columns show different runs, differing in the initial state of the pseudorandom number generator. The ε scheme seems to give results closest to the original evolution.

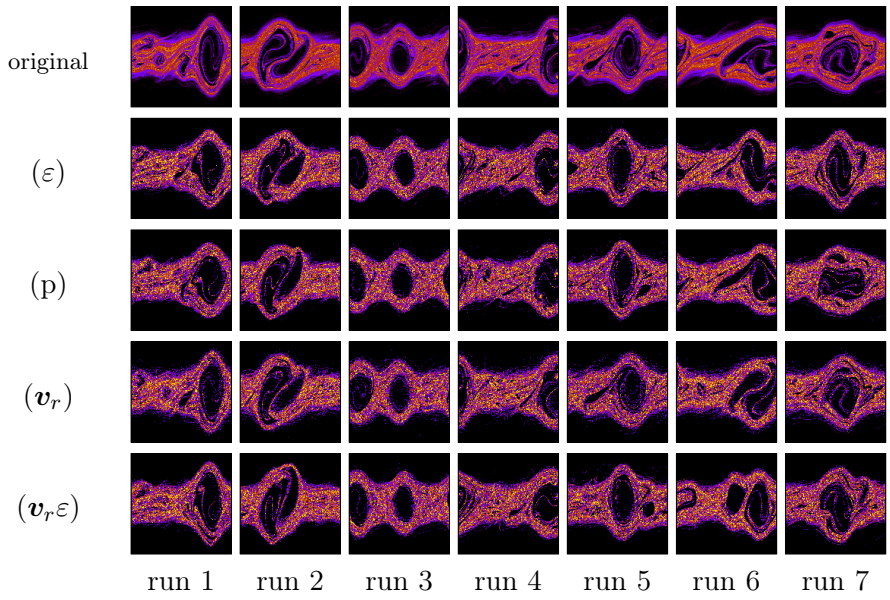

Figure 4.8: Simulation results of the two-stream instability. This figure shows the same as figure 4.7, but at $t = 90/\omega_p$ and with the merging done at $t = 20/\omega_p$. Here, the $\boldsymbol{v}_r$ scheme seems to best preserve the physical evolution.

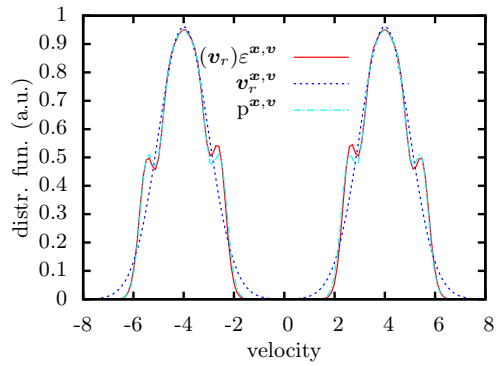

Figure 4.9: The velocity distribution function for various merge schemes, just after merging took place at $t = 5/\omega_p$ in the two-stream simulation. The curves shown represent the average over 100 runs, to smooth out noise. In each run, the number of particles was reduced from 10^6 to about $8 \cdot 10^4$, by merging five times. The scheme $\boldsymbol{v}_r$ does on average not alter the velocity distribution, so its curve coincides with the curve before merging. The ε and $\boldsymbol{v}_r\varepsilon$ scheme lead to almost indistinguishable velocity distributions after merging, and are therefore shown together. The p scheme gives similar results as these two: the tails of the distribution are moved towards the center, creating visible bumps in the velocity distribution function.

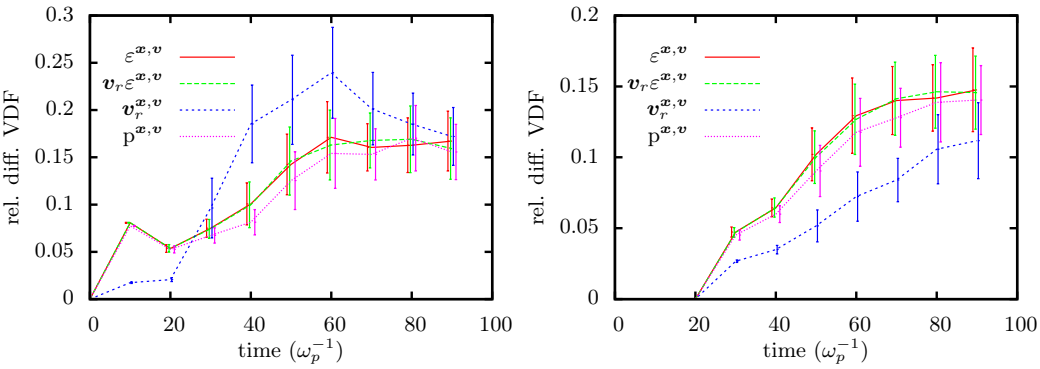

Figure 4.10: The difference in the velocity distribution function over time, caused by the various merge schemes in the two-stream simulation. The left figure shows results for merging at $t = 5/\omega_p$, while the right figure shows results for merging at $t = 20/\omega_p$. The quantity shown is the L^2 norm of the difference in the velocity distribution due to merging divided by the norm of the original velocity distribution, or $|\boldsymbol{f}_v(t) - \boldsymbol{f}_{v,0}(t)| \,/\, |\boldsymbol{f}_{v,0}(t)|$ as in equation (4.4). The error bars indicate the standard deviation in this quantity from run to run, computed from 100 different runs. The $\boldsymbol{v}_r$ scheme performs the worst for merging at $t = 5/\omega_p$, but when merging at $t = 20/\omega_p$ it performs the best.

beams are randomly mixed, significantly altering the velocity distribution. In general, using the norm of the velocity vector should probably only be used when the flow of particles is in a single direction.

When we perform the merging cell-by-cell, the results show no clear differences from those presented in figures 4.7 and 4.8.

4.4.5 Computational costs of k-d trees

The goal of an APM algorithm is to speed up a simulation, so the algorithm itself should not take too much time. Theoretically, the computational complexity of creating a k-d tree is $O(N_\mathrm{p} \log N_\mathrm{p})$, with N_p the number of points in the tree. The average cost of a random search in the tree is $O(\log N_\mathrm{p})$. We have tested the practical performance of the `KDTREE2` library on an Intel i7-2600 CPU. In figure 4.11 the creation time and the average search time are shown for k-d trees of various sizes. Neighbors can be found faster if the k-d tree is constructed in fewer dimensions. Note that the average search time is given for uncorrelated searches, that are done at random locations. This is the worst-case scenario, as the CPU cannot do efficient data caching. If the next search location is picked close to the previous search location, search times in 5D decrease by more than 80%.

The time scales for constructing and searching a k-d tree can be compared,

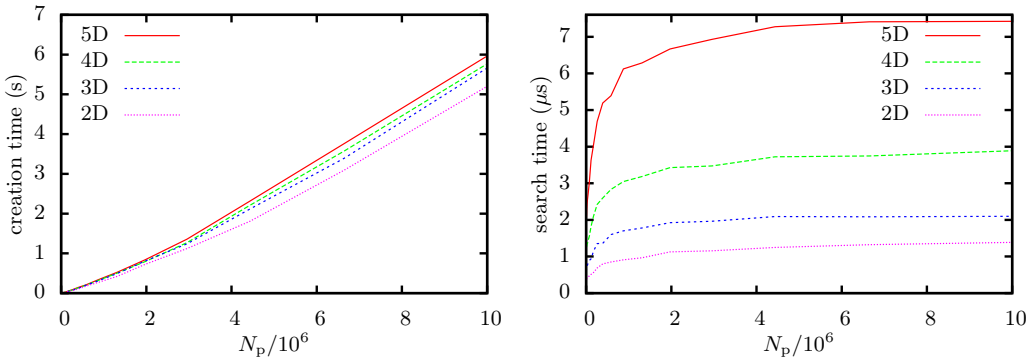

Figure 4.11: Performance figures for k-d trees in 2D-5D with N_p points, using the `KDTREE2` [113] library. Left: the time it takes to create the k-d tree. Right: the time it takes to find a nearest neighbor (for uncorrelated searches). The calculations were performed on an Intel i7-2600 CPU.

for example, with the cost of updating a particle in an electrostatic plasma simulation with collisions. On the same machine, about 0.1–1 μs is spent per particle on interpolating forces from the grid, updating the particle position and velocity, determining whether a collision should occur, mapping the particles to densities again and computing the electric field. In such simulations merging would typically not occur at every time step, and only for a fraction of the particles. Therefore, the computational cost of setting up k-d trees and searching for neighbors would not contribute much to the simulation time.

If in a simulation the cost of advancing particles is very small, but their weights have to adjusted very often, then the use of k-d trees might slow the simulation down. In such cases, it might be better to divide the particles over the grid cells, which can be done much faster than setting up a k-d tree, and then use a fast algorithm that operates on a cell-by-cell basis to adjust the weights.

4.5 Conclusion

Adaptively adjusting the weights of simulated particles can greatly improve the efficiency of simulations. We follow Welch et al. [98] and call algorithms that do this 'adaptive particle management' (APM) algorithms. In this work, we have focused on the pairwise merging of particles. We found that the use of a k-d tree offers several important advantages over present methods. First, only particles that are 'close together' are merged. 'Close together' can be defined as desired (for example close in position and velocity). This ensures that the distribution of particles is not significantly altered. Second, the merging can be performed completely independent of the numerical mesh used in the simulation.

The algorithm works in the same way, whether the simulation is in 1D or in any higher dimension. Third, with a k-d tree, the closest neighbors can be located efficiently. Therefore, the method can be used for simulations with millions of particles. Fourth, from a practical point of view, the use of a k-d tree library greatly simplifies the implementation of pairwise merging.

Two particles can be merged in different ways, and we have compared various merge schemes. An interesting option is to select properties for the merged particle at random from the original particles. With these stochastic schemes fluctuations increase, but on average both momentum and energy can be conserved. In the simulation of the two-stream instability we saw that when system is sensitive to small fluctuations, a scheme that conserves energy or momentum is preferred. But when the physical evolution does not depend strongly on small fluctuations, a stochastic scheme can perform better.

In general, it is more important to preserve the essential characteristics of the particle distribution function than to exactly conserve grid moments. A scheme that conserves energy or momentum should typically be used with a full coordinate k-d tree (containing $\boldsymbol{x}, \boldsymbol{v}$). A velocity norm k-d tree (containing $\boldsymbol{x}, |\boldsymbol{v}|$) can be used with a stochastic scheme. The advantage of a velocity norm k-d tree is that is can be constructed and searched faster than one with the full coordinates. The combination of a stochastic scheme with a full coordinate k-d tree seems a good choice: on average, the shape of the energy and momentum distribution functions is conserved, while the induced fluctuations in the grid moments are relatively small.

Chapter 5

Simulating the inception of nanosecond pulsed discharges in nitrogen/oxygen mixtures

We investigate the inception of nanosecond pulsed discharges with a 3D PIC-MCC (particle-in-cell, Monte Carlo collision) model with adaptive mesh refinement. This model, whose source code is available online, is described in the first part of the paper. Then we present simulation results in a needle-to-plane geometry, using different nitrogen/oxygen mixtures at atmospheric pressure. In these mixtures, non-local photoionization is important for the discharge growth. The typical length scale for this process depends on the oxygen concentration. With 0.02% oxygen, we observe the formation of small branches on the discharges. With 2% or more oxygen, an ionized almost spherical region can form around the electrode tip, which increases in size with the electrode voltage. Eventually this *inception cloud* destabilizes into streamer channels. On the other hand, the discharge velocity is almost independent of the oxygen concentration. We discuss the physical mechanisms behind these phenomena and compare our simulations with experimental observations.

This chapter will be submitted for publication as:
Simulating the inception of nanosecond pulsed discharges in nitrogen/oxygen mixtures, J. Teunissen, U. Ebert

5.1 Introduction

Nanosecond pulsed discharges have many applications, for example pollution control [115], ozone generation [116] and wound sterilization [7]. With current technology, voltage pulses of only a few nanoseconds can be made [117]. On such short time scales, energy is mostly transferred in a few fast electron-neutral processes. This prevents losses due to gas heating, and allows for more control over the electron energy.

In this paper, we investigate the inception of nanosecond pulsed discharges using 3D particle simulations in a needle-to-plane electrode geometry. We consider so-called *positive discharges*, i.e., the needle electrode has a positive voltage. Because electrons drift towards the electrode, a non-local source of electrons is needed for sustained discharge growth. Our simulations are performed in nitrogen/oxygen mixtures, in which photoionization is typically the most important non-local source of electrons [87, 88]. Because the amount of photoionization produced depends on the oxygen concentration [46], we can study the effect of photoionization by varying the amount of oxygen. We perform simulations under qualitatively similar conditions as in the experimental investigations of for example [32, 87, 118].

The paper is organized as follows. In section 5.2, the simulation model is introduced. Then, in section 5.3, simulation results are shown for four gas mixtures (N_2 containing 20%, 2%, 0.2% and 0.02% O_2), at four electrode voltages (3.5, 4.0, 4.5 and 5 kV), all at atmospheric pressure. For comparison, results with different electrode tips and with more particles per cell are also shown. In section 5.4, we analyze the simulation results and we discuss how the discharge morphology depends on the oxygen concentration and the electrode voltage.

5.2 Simulation model

The particle model we use is of the PIC-MCC (particle-in-cell, Monte Carlo collision) type [67, 105, 119]. This model was used before in [110, 120–122], here it is described in more detail. Although we call the model a *particle model*, we only track electrons as particles. Ions, which move much slower than electrons, are included as densities. Neutral gas molecules are not simulated, but they provide a homogeneous background that the electrons randomly collide with. Because we consider time scales up to ~ 10 ns, we can assume the ions to be immobile, and the gas not to heat up.

As indicated by the name particle-in-*cell*, such a model includes a computational grid. Particles are mapped to densities on this grid, and from these densities the electric field can efficiently be computed, by solving Poisson's equation. The electric field is then interpolated back to the particles.

The main advantage of using a 3D particle model is that we can study how stochastic fluctuations affect the three-dimensional discharge evolution. In [28] it

was shown that such density fluctuations can be important, and in chapter 3 such fluctuations were found to be important for negative streamers. The downside of using a particle model is that it is computationally more expensive than a typical plasma fluid model.

The particle model used in this paper was used before in several publications [120–122], and its source code is available at the homepage of our group [35]. Below, we describe the implementation of the model. The electric field solver, the electrode and photoionization are discussed in more detail; for the other parts we give a shorter description and refer to other publications.

5.2.1 Collisions

The discharges that we consider will be weakly ionized, having an ionization degree of about 10^{-5} up to 10^{-4}. Therefore, electron-neutral collisions dominate the behavior and we can neglect electron-electron and electron-ion collisions. We include impact ionization, (in)elastic scattering and electron attachment. The time and type of these collisions is determined using the null-collision method [123].

We use the N_2 and O_2 cross sections from the Siglo database [124]. After a collision, electrons are scattered isotropically. This is not realistic, because elastic scattering already becomes anisotropic at $\sim 10\,\mathrm{eV}$, see for example [125]. However, most electron-neutral cross sections below one keV are normalized under the assumption of isotropic scattering. For consistency, one should then also use this assumption in the simulation model. From a practical point of view, isotropic scattering is simpler to work with, and obtaining differential electron-neutral cross sections is complicated. This is illustrated at [126]: differential cross sections are only available for argon, based on Quantum-mechanical computations [127, 128].

5.2.2 Particle mover & time steps

We use the 'Velocity Verlet' scheme [129] to update the particles' position $\boldsymbol{x}$ and velocity $\boldsymbol{v}$ over time

$$\boldsymbol{x}_{t+\delta_t} = \boldsymbol{x}_t + \boldsymbol{v}_t \delta_t + \tfrac{1}{2}\boldsymbol{a}_t \delta_t^2, \tag{5.1}$$

$$\boldsymbol{v}_{t+\delta_t} = \boldsymbol{v}_t + \tfrac{1}{2}\left(\boldsymbol{a}_t + \boldsymbol{a}_{t+\delta_t}\right)\delta_t, \tag{5.2}$$

where $\boldsymbol{a}_t$ is the acceleration of the particle due to external forces. Because we work under electrostatic conditions without an external magnetic field, $\boldsymbol{a}_t = q/m \cdot \boldsymbol{E}_t$, where q/m is charge over mass and $\boldsymbol{E}_t$ the local electric field.

There are several time step restrictions in the simulations we perform. First of all, we impose a CFL-like condition on the particles, so that they move less than half a grid cell: $\Delta t < \frac{1}{2}\,\Delta x / v_{\max}$. We take $v_{\max}$ as the velocity at $9/10^{\text{th}}$ of the velocity distribution, to reduce fluctuations.

The electric field (see section 5.2.3) is not always recomputed when the particles are moved, instead a separate time step Δt_E is used. Each time the field is recomputed, the maximum relative difference with the previous field is estimated, using 1000 randomly selected samples at particle locations. When the maximum difference is larger than 7.5%, Δt_E is reduced, and when it is smaller, Δt_E is increased.

Although we assume that the plasma in our simulations is weakly ionized, high densities are sometimes produced, for example near the electrode tip. The presence of such regions also imposes a time step restriction, because Δt_E should be smaller than the *dielectric relaxation time* [122, 130]:

$$\Delta t_E < \varepsilon_0/(e n_e \mu_e), \tag{5.3}$$

where ε_0 is the dielectric permittivity, e the elementary charge, n_e the electron density and μ_e the electron mobility. This condition prevents 'over-screening' of an electric field, which can occur when the time step is too large: then the conductivity of the plasma reduces the electric field so much that it actually increases in the opposite direction.

To not have to use a small time step because of a small high-density region, we artificially limit the electron density to $2.5 \cdot 10^{21}$ m^{-3}, which corresponds to an ionization degree of 10^{-4}. Above this density, electrons are gradually converted to negative ions. This effectively limits the conductivity of the plasma to prevent oscillations. The effect on the simulations is usually small, because the maximum density is either never reached, or only in a small region.

5.2.3 Adaptive mesh refinement for the electric field

We assume that the discharge develops under electrostatic conditions, so that the electric field can be calculated from the electric potential, which is obtained by solving Poisson's equation. This approximation can be made because typical electron velocities are much smaller than the speed of light, and because the induced magnetic fields have negligible effect.

Nanosecond pulsed discharges have a 'multiscale' nature: around the space charge layers, a mesh resolution of a few micrometers is required, while a typical discharge measures at least a millimeter. Mesh refinement can therefore greatly help to speed up simulations. We use a 'nested grid' type of adaptive mesh refinement (AMR), similar to the procedure used in [29, 131]. The refinement procedure works in the following way. First, the coordinates at which refinement is needed are collected and grouped into boxes. If two boxes are within two grid points of each other, then they are merged into a single larger box. Each separate box then becomes a new refined grid, with a refinement factor of two. The Dirichlet boundary conditions for the potential on these new grids are interpolated from the parent grid. This procedure is repeated recursively, until there are no more grids to refine.

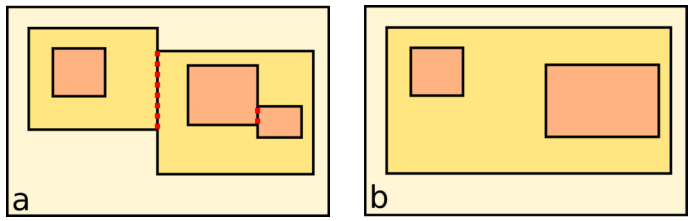

Figure 5.1: Schematic drawing of a mesh, to illustrate how refined patches are merged. a) The regions were refined patches 'touch' are indicated by red dots. b) The touching regions have been merged into larger rectangles.

Because boundary conditions are interpolated from the parent grid, refined boxes are not allowed to 'touch' each other, since this would limit the accuracy at the boundary. This is why boxes within two grid points of each other are merged into one larger box. As illustrated in figure 5.1, the merging of boxes typically increases the total number of fine grid points, which is a downside of the approach described here. We have experimented with a fine-to-coarse correction as described in [132, 133], but decided no to include this, because we observed only modest improvements in our test problems.

Our refinement criterion depends on the local electric field:

$$\Delta x < 2/\alpha(E), \tag{5.4}$$

where $\alpha(E)$ is the effective ionization coefficient in a field of strength E. The reason for using this criterion is that $1/\alpha$ is a typical length scale for ionization, so that the space charge layers of a discharge will have a width of a few times $1/\alpha$. The factor 2 in equation (5.4) is empirical. With a higher number the results visibly change because the mesh is not fine enough; lower numbers increase the computational cost.

We use Fishpack [30], a fast elliptic solver, to compute the electric potential on each block from the charge density. The electric field is then computed by taking central differences of the potential.

5.2.4 Electrode

The solver that we use to compute the electric potential, Fishpack, can only be used for separable, constant-coefficient elliptic problems on Cartesian grids. Including an electrode with such a solver can be done with the *capacitance matrix* or *charge simulation* method [31, 67]. These approached seemed unfeasible for three-dimensional refined grids, so we use the custom method described below.

Our method is based on the linearity and symmetry of Poisson's equation:

$$\nabla^2 \phi = -\rho,$$

where we have omitted ε_0 for simplicity. A point charge $\delta(\boldsymbol{r} - \boldsymbol{r}_j)$ will affect the potential at $\boldsymbol{r}_i$ by an amount $f_{i,j} \equiv f(\boldsymbol{r}_i, \boldsymbol{r}_j)$. In free space we have $f_{i,j} = 1/|\boldsymbol{r}_i - \boldsymbol{r}_j|$, so that there is a symmetry $f_{i,j} = f_{j,i}$. Numerical tests suggest that the discrete Poisson equation in a bounded domain also has this symmetry. This inspired the procedure outlined below.

Suppose that we place N point charges $\delta(\boldsymbol{r} - \boldsymbol{r}_i)$ (for $i = 1, 2, \ldots, N$) at points $\boldsymbol{r}_i$. We can now numerically compute the potential ϕ_i at each point, and store these values as λ_i's:

$$\lambda_i = \phi_i = \sum_j f_{i,j}. \tag{5.5}$$

To get an average potential V_0 at the points $\boldsymbol{r}_i$, we set the point charges to $q_i = V_0/\lambda_i$. The potential at each point is then

$$\phi_i = \sum_j q_j f_{i,j} = \sum_j V_0 f_{i,j}/\lambda_j,$$

so that the average potential is indeed V_0:

$$\frac{1}{N}\sum_i \phi_i = \frac{V_0}{N}\sum_i \sum_j f_{i,j}/\lambda_j \tag{5.6}$$

$$= \frac{V_0}{N}\sum_j \left(\sum_i f_{j,i}\right)/\lambda_j, \tag{5.7}$$

$$= \frac{V_0}{N}\sum_j \lambda_j/\lambda_j = V_0, \tag{5.8}$$

where we have used equation (5.5) and the fact that $f_{i,j} = f_{j,i}$. Although the average potential is now V_0, the potential at individual points might be far off. Suppose that the differences are $\delta_i = V_0 - \phi_i$, then an amount δ_i/λ_i can be added to each charge q_i. From equation (5.6) it follows that the average potential is then still V_0. This adjustment of the charges can be repeated, and in all our test cases we observed convergence towards the solution of having V_0 at each point, up to the discretization error.

To represent an electrode, we generate a large number of points on its surface. For the simulations presented here, these points are spaced by $3\,\mu$m close to the electrode tip. Away from the tip the spacing is increased to reduce the computational cost. When all these points have a potential V_0, they effectively represent an electrode surface.

This convergence towards V_0 is unfortunately rather slow, but that is not too much of a problem, for two reasons. First, in experiments the electrode voltage will also not precisely be V_0 during a nanosecond pulsed discharge. Second, in the simulations the potential is recomputed at every time step. Because each electrode-iteration starts from the previously determined surface charges, and

the potential distribution changes relatively slowly, the slow convergence is less of a problem. After a discharge has formed, the surface charges of the electrode are greatly reduced because of the conductivity of the plasma.

5.2.5 Adaptive particle management

The number of electrons in a typical discharge quickly grows to a value of 10^8 or more. To make particle simulation feasible on modest machines, we have developed an *adaptive particle management* algorithm [104], see chapter 4. This algorithm uses a k-d tree to locate particles that are close in position and velocity. Simulation particles can be merged or split, thereby changing their weights w. For merging we use the $\boldsymbol{v}_r$ scheme, in which the velocity is chosen at random from one of the original particles, see chapter 4. We set the 'desired' number of particles per grid cell (N_{ppc}) to 32, so each particle has a 'desired weight' w_d of

$$w_d = n_e \Delta x^3 / N_{\mathrm{ppc}}, \tag{5.9}$$

where n_e is the local electron density and Δx^3 the volume of the grid cell containing the particle. The particles for which $w < \frac{2}{3} w_d$ are stored in a k-d tree with coordinates $(\boldsymbol{x}, \lambda |\boldsymbol{v}|)$, where $\lambda = 10^{-12}\,\mathrm{s}$, see [104]. Each particle in this tree can then be merged with its nearest neighbor. The resulting particle gets its position and velocity randomly from one of the original particles.

Particles for which $w > \frac{3}{2} w_d$ can be split into $\lceil w_d/w \rceil$ new particles. These new particles have the same velocity as the original particle, but their positions are uniformly distributed over a volume $\Delta x^3 / N_{\mathrm{ppc}}$. This is done to 'smear out' particles moving into a finer grid. In the simulations, merging and splitting is performed each time the total number of particles has grown by a factor of 1.2.

Equation (5.9) gives large weights in the discharge interior, because both n_e and Δx^3 are large there (see section 5.2.3 for a description of the grid refinement used). In high-field regions where the discharge is actively growing, the weights are smaller, because both n_e and Δx^3 are smaller.

The use of 'super-particles' ($w > 1$) leads to increased density fluctuations, so that one has to be careful when studying a system sensitive to such fluctuations. To investigate whether this is a problem for our simulations, we include simulations for $N_{\mathrm{ppc}} = 48$ and $N_{\mathrm{ppc}} = 64$ in section 5.3.4.

Even with the adaptive particle management described above, the number of simulation particles required is still large. Therefore, the particle code was also parallelized using MPI (Message Passing Interface). The simulations shown in section 5.3 were performed on nodes containing two Intel L5640 CPU's, each having 6 cores. They took up to 48 hours, with up to $5 \cdot 10^7$ simulation particles.

5.2.6 Photoionization

Positive discharges can only grow if there are free electrons ahead of them. These electrons can start avalanches that grow towards the discharge, thereby extend-

ing it. In nitrogen/oxygen mixtures, photoionization can create these non-local electrons.

Photoionization can occur when an excited N_2 molecule decays via the emission of a UV-photon. If the photon has a wavelength between 98 and 102.5 nm, it can ionize an O_2 molecule. Although photoionization is relatively well-studied for N_2/O_2 mixtures, it is not clear which excited N_2 state(s) generate the ionizing photons [45, 57]. We therefore use a stochastic version of the photoionization model of [46], as was done before in [11]. In this model, η ionizing photons are (on average) produced per ionization. The factor η can be written as $\eta = \eta_E \cdot \eta_q$, where η_E is an efficiency depending on the local electric field E and η_q is a quenching-factor. For η_E we use the data shown in figure 5.2, taken from [46]. We set the quenching factor η_q to

$$\eta_q = \frac{30\,\text{mbar}}{p + 30\,\text{mbar}}, \tag{5.10}$$

where p is the pressure. At the 'quenching pressure' of 30 mbar, half the excited states decay due to collisions with other molecules instead of emitting a UV-photon. Equation (5.10) was also taken from [46], which makes use of older references [134].

If a simulation particle with weight w (representing w physical particles) undergoes an ionization, the number of photons produced is sampled from the Poisson distribution with mean ηw. Note that these photons are always created individually, i.e., a super-particle produces no 'super-photons'. The absorption depth of a UV-photon depends on its wavelength. As in [11], we assume the wavelength to be uniformly distributed over the interval from 98 to 102.5 nm. Given a uniform random number R, the typical absorption length l of a photon is then given by [46]

$$l = l_{\text{max}}^{R} l_{\text{min}}^{1-R} / p_{O_2}, \tag{5.11}$$

where $l_{\text{min}} = 0.67\,\mu\text{m}$, $l_{\text{max}} = 0.38\,\text{mm}$ and p_{O_2} is the partial pressure of oxygen in bar. The average absorption length $\bar{l}$ resulting from equation (5.11) is

$$\bar{l} \approx 0.093\,\text{mm}/p_{O_2} \tag{5.12}$$

After the absorption length l has been determined according to equation (5.11), the actual distance to absorption is sampled from the exponential distribution with mean l. The photon gets a random isotropic direction, and at the location of absorption an electron-ion pair is created instantaneously.

We would like to point out some uncertainties within this model:

- We only know the quenching factor (5.10) for air, and assume it to be the same for other mixtures.
- There should be some delay between creating the excited state and the emission of a UV-photon. Because we are not aware of good data on this delay, we assume it to be zero in the simulations.

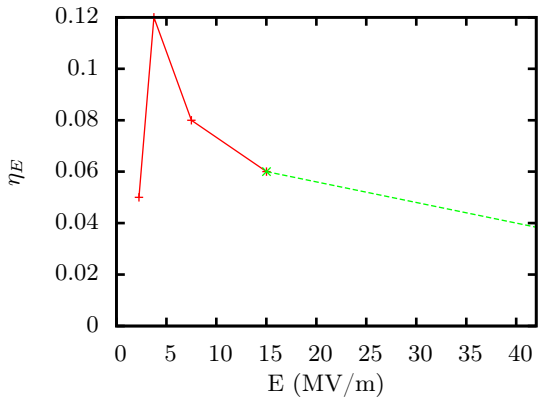

Figure 5.2: The photoionization coefficient η_E versus electric field strength. This number indicates how many ionizing photons are produced per electron-impact ionization, in the absence of quenching. The solid curve shows four tabulated points from [46], the dashed curved is an extrapolation of these values that we use for higher fields.

- There are discrepancies between the photoionization rates measured in different experiments, so it is unclear how accurate the above coefficients are, see the discussion in [45, 57].

However, as we will see, the simulation results are not very sensitive to the photoionization coefficients, as was observed before in [88].

5.2.7 Other sources of free electrons

Besides photoionization, there can be other sources of free electrons. We briefly discuss some of them here, although we do not include them in the simulations of section 5.3.

First, some level of background ionization is always present, mostly in the form of negative ions from which electrons can detach, see for example [121] or chapter 8. However, typical densities are 10^3 - $10^4 \mathrm{cm}^{-3}$ [57], which is negligible compared to typical photoionization levels (even for nitrogen with just 0.02% oxygen). On the other hand, background ionization can be important to start a discharge, since photoionization only occurs when the discharge has already started. For simplicity, we use an ionized seed in front of the electrode, as discussed in section 5.2.8.

Second, many pulsed discharges are repetitive, so that left-over ionization can affect the next discharge [34]. However, we assume here that there is a long enough time between pulses to ignore this.

Third, there can be secondary (electron) emission from the cathode, due to the impact of positive ions. But for nanosecond pulsed discharges, the positive

ions typically cannot reach the cathode within the pulse duration.

5.2.8 Simulation conditions

The simulation conditions were chosen to reveal the influence of the electrode voltage and the oxygen concentration as clearly as possible. We therefore use a smaller domain and a lower voltage than is used in most experiments. Furthermore, the initial conditions were chosen such that the discharges start in the same manner, see below.

Computational domain

A slice through the computational domain is shown in figure 5.3, showing the electrical potential at $t = 0$. The domain measures $(5.12\,\text{mm})^3$, and an electrode with radius 0.25 mm and height 2 mm is centered at the bottom. Unless mentioned otherwise, see section 5.3.3, the radius of curvature of the electrode tip is 0.125 mm. The side and top walls of the domain are grounded. In the plane with the embedded electrode we use an analytic expression for the potential: it decays linearly to ground potential at three times the electrode radius.

The boundary condition at the bottom can locally create strong electric fields. In experiments, a dielectric material is often used to electrically insulate the electrode from the discharge chamber. We cannot implement this in our field solver, but we still want to prevent discharges from starting there. We therefore do not allow electrons or grid refinement below about 1.5 mm, see figure 5.3. We used at most 7 levels of grids, with the coarsest mesh having a resolution of 80 μm and the finest a resolution of 1.25 μm.

Initial conditions

To start the discharge, we place an ionized seed 0.13 mm above the electrode tip, indicated by the pink dot in figure 5.3. The seed contains 10^4 electron-ion pairs, that have a Gaussian distribution of width 0.05 mm. Instead of such a seed we could have used background ionization to start the discharges. Although this is perhaps more realistic, it is not very convenient for the simulations presented in this paper: it can take considerable time before the discharge starts, and because different discharges start at a different locations it is harder to compare them. We do not include a rise-time for the electrode voltage, to ensure that the simulated discharges all start at the same voltage.

5.3 Simulation results

In this section the simulation results are presented. First, we show the discharge evolution for four N_2/O_2 mixtures, containing 0.02 to 20% oxygen, and voltages

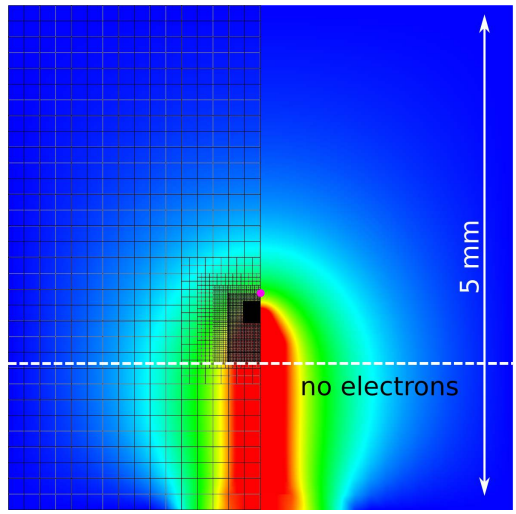

Figure 5.3: A slice through the computational domain at $t = 0$, showing the electrical potential. Red indicates the electrode voltage, blue zero voltage. The grid at $t = 0$ is shown in the left half of the figure. Electrons are not allowed below the dashed line, to prevent discharges from forming near the bottom boundary where the electrode is embedded. The pink dot indicates the location of the initial seed.

between 3.5 and 5 kV. Then we present additional simulations for the 20% O_2 and 5 kV case, but with a blunter and a sharper electrode. We also compare results for a different number of super-particles per cell (see section 5.2.5), to see how much this affects the discharge evolution.

5.3.1 Dependence on voltage and oxygen concentration

In figures 5.5 to 5.8, we show the evolution of the electron density, for electrode voltages of 3.5, 4, 4.5 and 5 kV, respectively. For each voltage, results are shown for four different O_2 concentrations: 0.02%, 0.2%, 2% and 20%.

All images in figures 5.5 – 5.7 are zoomed in on the electrode tip. They use the same scale and viewpoint, see figure 5.4 and its caption. The electron density is visualized with volume rendering, using Visit [135]. Densities above $10^{21}\,\mathrm{m}^{-3}$ are fully opaque, zero density is fully transparent, and in between the opacity is linearly interpolated.

Looking at the results, we can make a few observations. In all cases, an ionized region forms above the electrode tip. This is to be expected: the electric field is strongest there, and the initial seed of electrons and ions was placed right above the electrode. The electrode voltage affects this growth in two ways: With a higher voltage, the discharges grow much faster. But with a higher voltage, the ionization grows also in a more 'spherical' way, compare for example the results

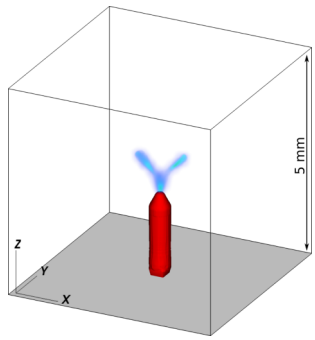

Figure 5.4: A view of the computational domain with the electrode, used for figures 5.5 to 5.8. In those figures we show two views: one is aligned on the +Y axis and the other on the -Z axis, indicated here.

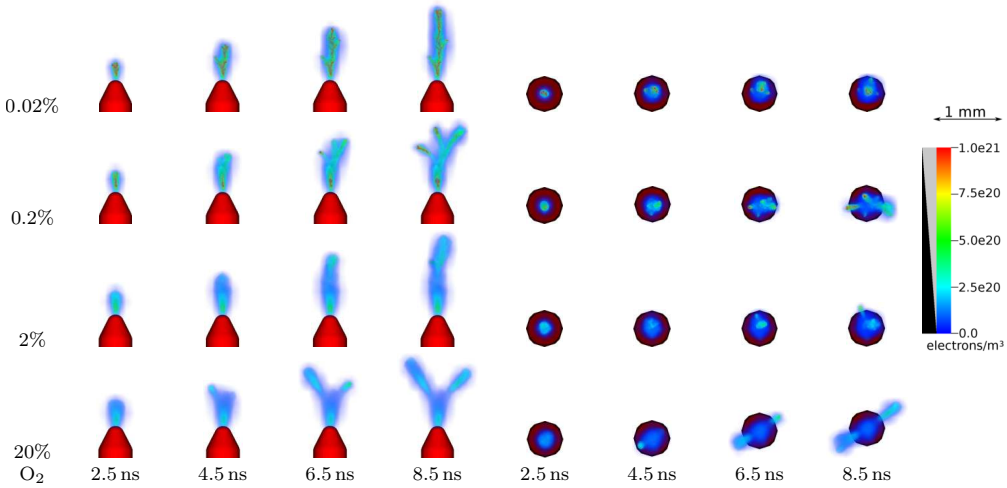

Figure 5.5: Simulation results for a 3.5 kV electrode, for four different oxygen concentrations. The electron density is shown using volume rendering with Visit [135]. The opacity is indicated in the legend, above which a length scale is indicated. The highest electrons densities occur inside the discharges, and are therefore hard to see.

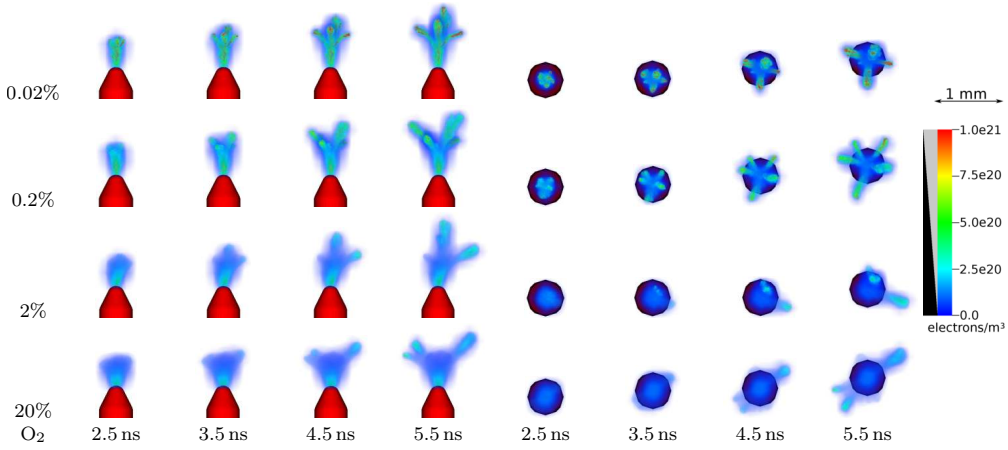

Figure 5.6: Simulation results for a 4 kV electrode.

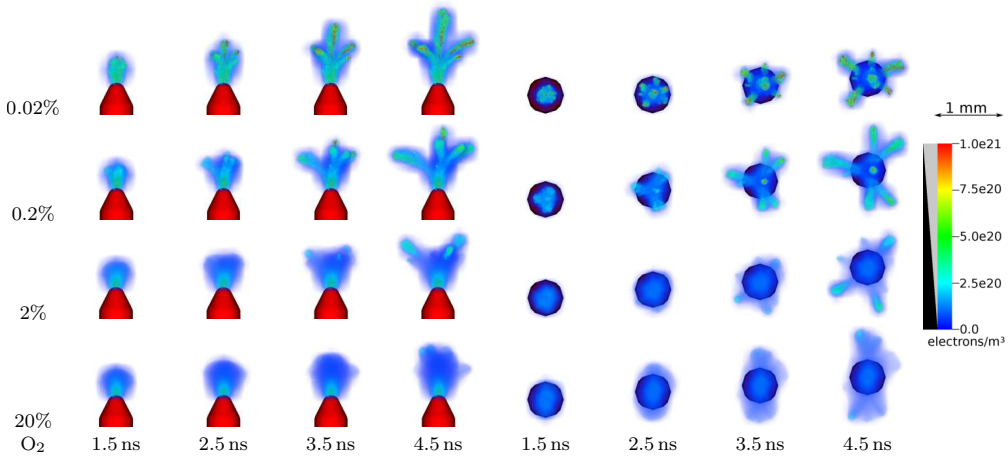

Figure 5.7: Simulation results for a 4.5 kV electrode.

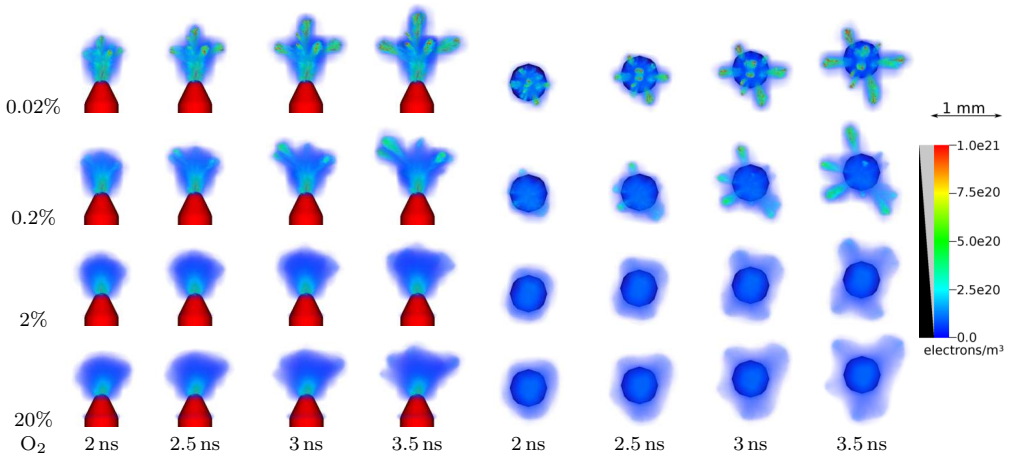

Figure 5.8: Simulation results for a 5 kV electrode.

for 2%–20% oxygen between figures 5.5 and 5.8. Especially at 5 kV the formation of an ionized cloud is clearly visible. These *inception clouds* [32, 33, 50, 50, 103, 136] all destabilize into streamers, with the number of streamers increasing with the voltage. This seems to happen at the moment when the inception clouds have slowed down in their spherical expansion.

The discharges are also affected by the oxygen concentration. With the lowest oxygen concentrations, the discharges primarily develop in the vertical direction, as a single streamer channel. At higher oxygen levels there is more non-local photoionization, which leads to smoother growth and wider discharges that eventually branch out into multiple streamers. This is discussed in more detail in section 5.4.

5.3.2 Cross sections showing the electric field and charge density

Here we have a closer look at the development of the discharges with 20% and 0.02% O_2 at 5 kV. Figure 5.9 shows cross sections (in the x, z-plane) of the electric potential and the electric field for these discharges.

The difference between the case with 20% and 0.02% O_2 is clearly visible in the electric field: with 20% O_2, the discharge develops smoothly, while the discharge boundary is much more irregular with 0.02% O_2. These irregularities strongly enhance the local electric field.

5.3.3 Effect of electrode tip

To investigate how discharge inception depends on the shape of the electrode tip, we have also performed simulations with a sharper and a blunter tip. For the

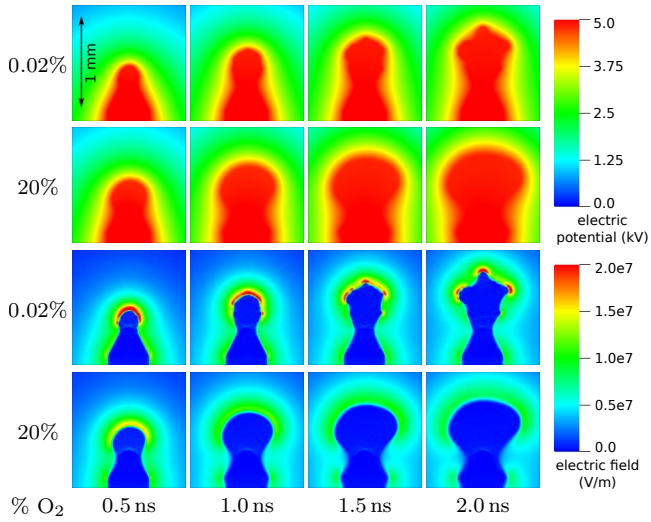

Figure 5.9: Cross sections of the electric potential and the electric field, for nitrogen with 0.02% and 20% oxygen. With 20% O_2, the discharge is in the inception cloud phase for all figures, with 0.02% O_2 it destabilizes around 1.5 ns. The electrode voltage is 5 kV for both cases.

results in section 5.3.1, an electrode with a radius of curvature at its tip $r_c = 2/16$ mm was used. In figure 5.10, additional results are shown for electrodes with $r_c = 1/16$ mm and $r_c = 3/16$ mm, for the case of 5 kV and 20% oxygen.

With a larger radius of curvature, the discharge seems to be less symmetric, and it destabilizes earlier into streamers. Possible reasons for this are:

- The high electric field near a sharp electrode tip is concentrated in a small region. This makes it more likely that a symmetric, 'cloud-like' discharge develops.
- With a sharper electrode the electric field is higher, and therefore also the degree of ionization. Because the amount of photoionization is proportional to the amount of ionization (see section 5.2.6) we expect the development to be more homogeneous with a sharp electrode.

We should emphasize that more simulations have to be performed to make the observations of figure 5.10 statistically significant, ideally combined with experiments.

5.3.4 Varying the number of particles per cell

In the simulations we make use of so-called *super-particles*, see section 5.2.5. For the simulations presented thus far, we used a desired number of particles per cell of $N_{\text{ppc}} = 32$. Due to the way we adjust the weights, the number of

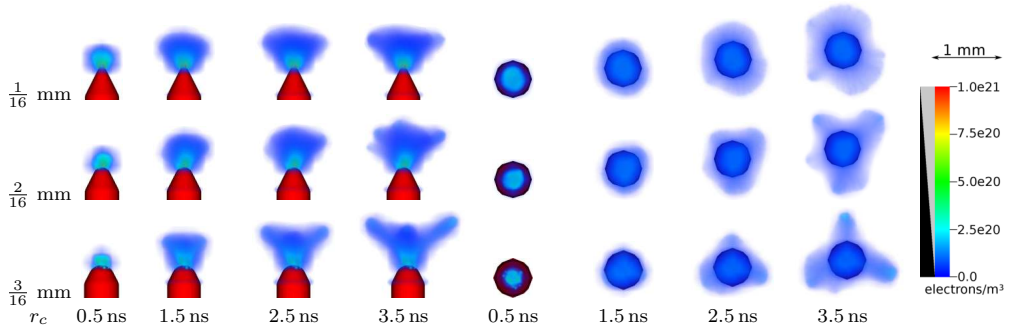

Figure 5.10: Simulation results for a sharper ($r_c = \frac{1}{16}$ mm) and a blunter ($r_c = \frac{3}{16}$ mm) electrode tip, where r_c is the radius of curvature. The electrode with $r_c = \frac{2}{16}$ mm was also used for figures 5.5 – 5.8. The simulations were performed in N_2 with 20% O_2, using a voltage of 5 kV, so that the row with $r_c = \frac{2}{16}$ mm corresponds to the bottom row of figure 5.8.

particles per cell will typically be higher by a factor of about 3/2, so that we would have about 50 particles per cell. In regions where there is strong electron impact ionization, the number of particles per cell will also be higher, because the merging of particles takes time.

To determine how important the parameter N_{ppc} is in our model, we have performed simulations for two additional cases: $N_{\text{ppc}} = 48$ and $N_{\text{ppc}} = 64$, at 5 kV with 20% O_2. The results are shown in figure 5.11. If N_{ppc} is too low, then one would expect the inception cloud to destabilize earlier, due to artificially increased fluctuations in the electron density. For the test cases presented here, no such effect is visible. This means that the value $N_{\text{ppc}} = 32$ is probably high enough for the qualitative description of discharges that we give in this paper.

5.4 Discussion

5.4.1 The growth of positive discharges

In our simulations, the discharge starts to grow in the following way. First, electrons from the seed accelerate towards the electrode, forming many overlapping electron avalanches. These avalanches are absorbed by the positive electrode, but leave positive charge behind around the electrode. Due to photoionization, new free electrons are created, which can form consecutive avalanches. Eventually, so much positive charge is deposited around the electrode that a region around it takes over (almost) the electrode voltage, see figure 5.9. New avalanches continue to grow outside this region, depositing additional positive charge. In this way, the boundary of the electrically screened region (hereafter called the discharge)

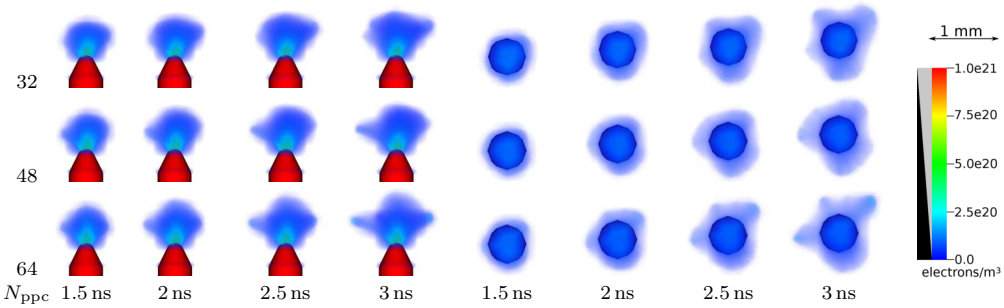

Figure 5.11: Simulation results for three values of N_{ppc}, the number of particles per cell (see section 5.2.5.) A voltage of 5 kV was used, and the gas contained 20% oxygen. The case of $N_{\text{ppc}} = 32$ corresponds to the bottom row of figure 5.8.

continues to move outwards.

5.4.2 Discharge growth velocity

The growth velocity in our simulations was not very sensitive to the oxygen concentration, see figures 5.5 to 5.8. This has been observed before. In [87], experiments on streamers in different nitrogen/oxygen mixtures were performed. When the oxygen concentration was varied over almost six orders of magnitude the streamers propagated with roughly the same velocity, even though their morphology changed significantly. A simulation study in cylindrical symmetry [88] also found that the streamer velocity was insensitive to the amount of photoionization, whereas the streamer diameter did change with the oxygen concentration.

Let us try to understand this. Suppose that we have an already developed discharge, for which we can somehow change the amount of photoionization and the applied voltage. The amount of photoionization affects the electron density n_e ahead of the discharge. This density locally grows approximately exponentially

$$\partial_t n_e \approx \alpha v_d n_e, \tag{5.13}$$

where α is the ionization coefficient and v_d the electron drift velocity. The exponential growth means that the dependence on the electron density ahead is rather weak: if the growth starts at a γ times lower value, then the same final density is reached with a delay of about $\ln(\gamma)/(\alpha v_d)$.

The dependence on the applied voltage is much stronger, see figure 5.12 where αv_d is shown versus the electric field strength for nitrogen and air. This explains why the growth velocity increased with the electrode voltage in our simulations. For example, the discharges at 5 kV and 2.0 ns are larger than those at 4 kV and 3.5 ns, see figures 5.6 and 5.8.

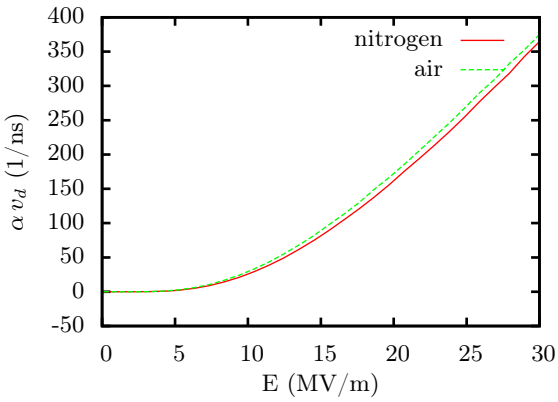

Figure 5.12: The ionization rate αv_d versus the electric field for nitrogen and air at 1 bar.

What we have not taken into account in the above arguments is that the amount of photoionization and the applied voltage will change the shape of a discharge, which will in turn affect the electric field. With less photoionization thinner structures emerge, that give rise to higher electric fields. This could explain why some discharges seemed to grow faster with less oxygen, see figure 5.7.

5.4.3 Relation oxygen / photoionization level

If we vary the oxygen concentration, what is the effect on the photoionization density around a discharge? Equation (5.12) tells us that the mean absorption length of ionizing photons is inversely proportional to the oxygen concentration. This means that the number of photoionization events in a small volume around the discharge is approximately proportional to the oxygen concentration.

We can use equation (5.11) to compute the expected fraction $f(r)$ of UV-photons absorbed within a distance r. By integrating, we get

$$f(r) = 1 + \frac{\mathrm{Ei}(-rp_{\mathrm{O_2}}/l_{\max}) - \mathrm{Ei}(-rp_{\mathrm{O_2}}/l_{\min})}{\ln(l_{\min}/l_{\min})}, \tag{5.14}$$

where $\mathrm{Ei}(x)$ is the exponential integral. In figure 5.13, $f(r)$ is shown for several oxygen concentrations.

5.4.4 Morphology at low oxygen concentrations

The morphology (shape) of a developing discharge is quite sensitive to the oxygen concentration, as can be seen in figures 5.5 to 5.8. As explained in the previous section, a low oxygen concentration means that there will be less photoionization close to the discharge. With fewer electron avalanches coming in, the discharge

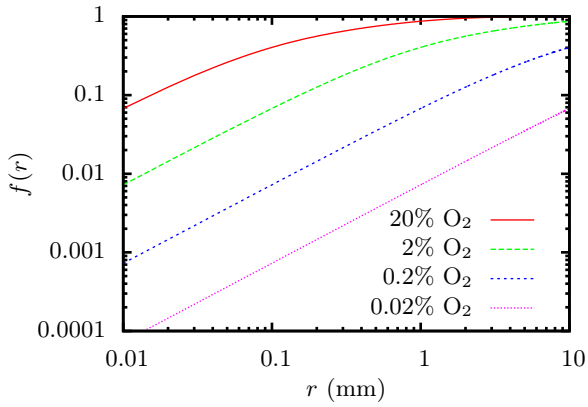

Figure 5.13: The function $f(r)$ shows the fraction of UV-photons that are absorbed within a distance r after being created. For small values of r, we see that $f(r)$ is proportional to the oxygen concentration. See also equation (5.14).

growth is more irregular. The protrusions enhance the electric field, so that they can quickly grow into streamer channels, see figure 5.9. This irregular growth shows some qualitative similarities with 'diffusion limited aggregation' [64].

Especially at the lowest oxygen concentration used in this paper (0.02%), we observe the formation of thin branches on the discharge, see figure 5.14. Such branches have been observed in several experiments in nitrogen and other pure gases, see for example [34, 87, 89]. In these studies, the branches were referred to as *feathers*. It was found that the repetition frequency of a pulsed discharge affects the formation of feathers, because it influences the background ionization density, see the discussion in [34].

One question raised in this earlier work was whether feathers are single avalanches [89] or small branches that carry space charge [34]. For our simulations, the answer is clear: feathers are small branches that carry space charge. These branches locally enhance the electric field, but on average the discharge still grows faster in the forward direction. The main channel reduces the field enhancement at the protrusions, and they stop growing.

5.4.5 The formation of inception clouds

At higher oxygen concentrations and higher voltages, we observed the formation of an ionized, almost spherical region around the electrode tip, see figures 5.7 and 5.8. Such *inception clouds* have been observed experimentally in [33, 50, 50, 103, 136]. In [32], the properties of these inceptions clouds have recently been studied in quite some detail. It was found that the typical radius of the inception cloud $\bar{r}_{\text{cloud}}$ was between $0.6R_0$ and $0.9R_0$, where $R_0 = V/E_c$ is the electrode voltage divided by the critical electric field. For the simulation results with 20% oxygen at 4.5 kV (figure 5.7), we get $\bar{r}_{\text{cloud}} \sim 0.6\,R_0$, and at 5 kV (figure 5.8) we

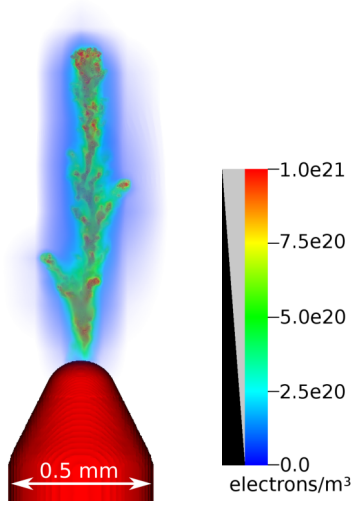

Figure 5.14: Zoom of the 0.02% oxygen case of figure 5.5 at 8.5 ns, showing the small-scale protrusions on the sides of the streamer channel.

get $\bar{r}_{\text{cloud}} \sim 0.7$ to $0.8\,R_0$, in good agreement with the experimental observations.

At a voltage of 4 kV (figure 5.6), the observed inception clouds are much smaller: $\bar{r}_{\text{cloud}} \sim 0.3$ to $0.4\,R_0$, and at 3.5 kV (figure 5.5) they are not visible at all. We are not aware of experimental observations of inception clouds at such 'low' voltages. It could be that inception clouds only form to sizes of $\bar{r}_{\text{cloud}} \sim 0.6$ to $0.9\,R_0$ when $\bar{r}_{\text{cloud}}$ is at least a few times a typical streamer radius. Another possibility is that the electron-ion seed used in our simulations affects the formation of the cloud.

The larger the inception cloud gets, the lower the electric field outside it will be. Assuming that the cloud is equipotential (see figure 5.9), we can relate the electric field at its surface to the local curvature, see for example [137]. The relation between curvature and electric field eventually destabilizes the cloud: regions with stronger curvature grow faster, further increasing their curvature. In [138, 139] this so-called *Laplacian instability* is analyzed in detail. Typical length scales that could be important for the destabilization process are:

- The 'radius' of the area above breakdown.
- The typical separation between incoming electron avalanches. If a typical photoionization density is n_0, then the corresponding length scale is $1/\sqrt[3]{n_0}$.
- The 'ionization length' $1/\alpha$.
- The typical photoionization distance.
- The radius of curvature of the electrode.

We leave a careful analysis of the relevance of these length scales for future research.

5.5 Conclusion

This chapter has made two contributions: First, a 3D PIC-MCC (particle-in-cell, Monte Carlo collision) was introduced, and its source code was made available at [35]. Second, this model was used to investigate the inception of nanosecond pulsed discharges. The main advantage of using a particle model in 3D is that one can observe how stochastic fluctuations affect the discharge. We have performed simulations in a needle-to-plane geometry, for voltages between 3.5 and 5 kV, and for nitrogen with between 0.02% and 20% oxygen.

We found that the discharge velocity was almost independent of the oxygen concentration, in agreement with experimental observations [87]. With 0.02% oxygen, we observed the formation of thin branches on the discharge, which likely correspond to the 'feathers' observed in various experiments [34, 87, 89]. With 2% or more oxygen and a voltage of 4.5 or 5 kV, we observed the formation of an ionized, almost spherical region around the electrode tip. The radius of these observed *inception clouds* was about a factor two smaller than was found in a recent experimental study [32].

Chapter 6

Why isolated streamer discharges hardly exist above the breakdown field in atmospheric air

We investigate streamer formation in the troposphere, in electric fields above the breakdown threshold. With fully three-dimensional particle simulations, we study the combined effect of natural background ionization and of photoionization on the discharge morphology. In previous investigations based on deterministic fluid models without background ionization, so-called double-headed streamers emerged. But in our improved model, many electron avalanches start to grow at different locations. Eventually the avalanches collectively screen the electric field in the interior of the discharge. This happens after what we call the 'ionization screening time', for which we give an analytical estimate. As this time is comparable to the streamer formation time, we conclude that isolated streamers are unlikely to exist in fields well above breakdown in atmospheric air.

This chapter has been published as [110]:
Why isolated streamer discharges hardly exist above the breakdown field in atmospheric air, A.B. Sun, J. Teunissen and U. Ebert, *Geophys. Res. Lett.* 40, 2417 (2013)

6.1 Introduction

Streamers play a key role in the early stages of atmospheric discharges; they appear, e.g., in lightning inception, in the streamer coronas of lightning leaders and of jets, and in sprite discharges. The late D.D. Sentman liked to call streamers the "elementary particles" of discharge physics.

Streamers are rapidly growing plasma filaments that penetrate into non-ionized regions due to the electric field enhancement at their tips. When the local electric field exceeds the breakdown threshold of a gas, the neutral gas molecules start to become ionized by impact of electrons with energies above 12 eV. While the ionization density grows, charged particles move in the electric field and form space charge regions that modify the field. The ionization then grows rapidly at channel edges where the field is enhanced, while the electric field is suppressed in the ionized interior. In this manner long ionized channels, so-called streamers, can grow. Positive or negative streamer channel heads have to be distinguished depending on the net charge in their heads; they propagate along or against the direction of the electric field.

We present a new view on streamer formation in fields above the breakdown threshold. Recently, [140] have shown the importance of detachment from negative ions for delayed sprite formation in the mesosphere. Here, we show that this mechanism also changes our understanding of streamer discharges in the troposphere.

In the past 30 years, simulations that model electrons and ions as densities have developed into a key method for exploring streamer physics. Most simulations are effectively performed in two dimensions (2D), using a longitudinal and a radial coordinate, hence assuming cylindrical symmetry of the streamer. The emergence of a double-headed streamer, with a positive and a negative growing end, was first seen in simulations by [38]. The nonlocal photoionization mechanism that allows positive streamers to propagate in air, was first implemented by [37]; he also extrapolated his numerical results and suggested that such streamers grow exponentially in fields above the breakdown value. Similar observations were later made by [14] who studied how these results depend on atmospheric altitude or on air density. The exponentially growing single streamers in high fields also play a role in a recent theory on terrestrial gamma-ray flashes by [77]. [11] developed a 2D axisymmetric PIC-MCC model to study streamers, and found that a double-headed streamer forms at 10 km altitude, with similar initial conditions as [14]. At sprite altitudes around 70 km, double-headed streamers were simulated by [14], [15] and [11, 36]. Most of these simulations were performed with fluid models in 2D, enforcing cylindrical symmetry.

In the present paper, we reinvestigate streamer formation in electric fields above the breakdown value. Such 'overvolted regions' can for example form around the tip of a lightning leader. We here assume that the field quickly rises to a value above the breakdown threshold and that it is initially homogeneous.

Although not directly corresponding to a particular physical situation, this keeps the analysis more simple and general, and it can serve as a local approximation. Our findings are very different from those of the authors cited above, because our model contains essential additional features: First, we include electron detachment from negative ions, which are present due to natural background ionization. Second, we are able to perform our simulations in full three spatial dimensions. Third, we work with a particle model, following the stochastic motion of individual electrons rather than approximating them as densities with completely deterministic dynamics. In this manner, we include physically realistic stochastic fluctuations, in particular, in the regions with low ionization, similarly as [11, 36], [21, 102], and [28]. The calculations are performed in atmospheric air at 1 bar. Our results show that in a field above breakdown in air, isolated streamers are unlikely to form. This is consistent with lab experiments: [34] and [50] observed 'inception clouds' that form around electrodes when a high voltage is suddenly applied to air. These clouds form essentially in the region where the field is above the breakdown value, and streamers only form beyond this region. We conclude that under normal atmospheric conditions, isolated streamers hardly exist in fields well above the breakdown threshold.

6.2 Model

A 3D particle-in-cell code with a Monte Carlo collision scheme has been developed to simulate the dynamics of streamer formation. In the model, electrons are tracked as particles. Ions are immobile, as they would not move significantly on the time scales we consider. Neutral molecules are not simulated, but they provide a background density that the electrons randomly collide with. We include elastic, inelastic, ionizing and attaching collisions. These collisions were implemented in the same way as in [21], with the same cross sections for collisions. Photoionization is an important process in many discharges, where excited N_2 molecules emit photons that ionize O_2 molecules. We use a stochastic version of the photoionization model of [46], as was done before by [11]. Below we present the most important new features of our model.

6.2.1 Natural background ionization and electron detachment

In atmospheric air near ground pressure, background ionization is mostly present in the form of O_2^- and positive ions. The number of free electrons is much smaller, because they quickly attach to O_2 molecules to form O_2^-. In enclosed areas such as buildings, typical background ion densities are 10^3 - $10^4 \mathrm{cm}^{-3}$, mostly due to the decay of radon [57]. As altitude increases, cosmic radiation becomes the dominant source of background ionization [141]. *Ermakov et al.* measured the concentration of negative ions in the lower atmosphere. The ion concentration increases as altitude increases. A level of approximately $10^3 \mathrm{cm}^{-3}$ was recorded

at 5 km altitude, in agreement with estimates by [142] and [143]. Background ionization can also be present due to previous discharges [34, 140, 144].

Electron detachment can occur when an O_2^- ion collides with a neutral gas particle. The probability of electron detachment from O_2^- depends on the local electric field and on the gas density. We include electron detachment from negative ions in the model, with rate coefficients from [145].

We remark that at mesospheric altitude, most negative background ions are O^- ions as they form by dissociative attachment at low air density. These ions are also a source of electrons by detachment [140, 146, 147].

Electron storage in the form of negative ions, from which they can later be detached, combined with the strong non-local effect of photoionization distinguishes discharges in air from those in other gases, e.g. high purity nitrogen.

6.2.2 Numerical techniques

An *adaptive particle management* algorithm is used to control the number of simulation particles in the code. We use relatively more simulation particles around the streamer head, and relatively few in the streamer interior. And where the electron density is low, electrons are tracked individually. Details of the particle management method are given by [104].

To be able to simulate larger systems, an *adaptive mesh refinement* (AMR) technique is used. The AMR method is similar to the methods of [29] and [12], but now in 3D. The code is electrostatic, as the velocities are much smaller than the speed of light and the induced magnetic fields are negligible compared to the electric fields. At every time step, the electric potential is computed from the charge density by solving the Poisson equation with Fishpack [30]. The electric field is then the numerical gradient of the electrical potential. To increase the performance and the maximum number of simulation particles, the particle code was parallelized using MPI (Message Passing Interface).

6.3 Results and discussion

We perform simulations in a gas mixture of 80% N_2 and 20% O_2, at 1 bar and 293 Kelvin. The simulation domain is cubic, of size $(4\text{ mm})^3$. An external electric field of 7 MV/m is applied in the negative z-direction, which is about 2.3 times of the breakdown field E_k. One electron-ion pair is placed at the center of the domain. We first show 'unrealistic' results with photoionization only, followed by 'realistic' results where natural background ionization is included. Then we indicate how these results depend on the initial presence of free electrons, and we introduce the concept of the 'ionization screening time'. Finally, we discuss discharges at higher altitudes in the atmosphere.

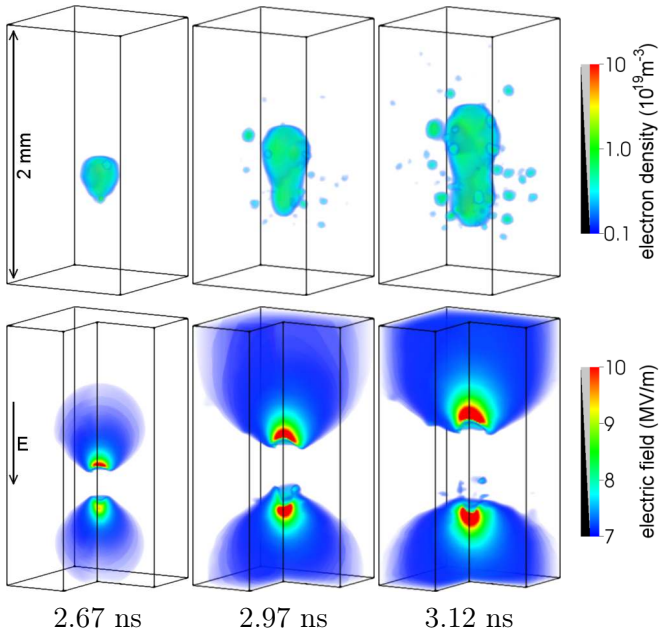

Figure 6.1: The electron density (top row) and the electric field (bottom row) using photoionization only (unrealistic). Times are indicated below each column. The simulation started with a single electron-ion pair in non-ionized air at 1 bar and 293 K in a downward homogeneous background field of 7 MV/m (about 2.3 times E_k). Of the total simulation domain of $(4\text{ mm})^3$, the range from 2 to 4 mm is shown in the vertical direction, and the range from 1.5 to 2.5 mm in the two lateral directions. The figures were generated using volume rendering, and the opacity is shown next to the colorbar; black indicates transparency. For figures in the second row, a quarter of the domain is removed to show the inner structures of electric field.

6.3.1 Photoionization only

We first present results with photoionization only, and no background ionization. This is not very realistic, as some background ionization will always be present in air. But these results help to clearly illustrate the effects of background ionization later on. We remark that other authors have often presented results with photoionization only.

Figure 1 shows the evolution of the electron density and the electric field in three stages, from 2.67 ns to 3.12 ns. The initial electrons are accelerated rapidly in the external electric field. They collide with molecules and ionize them, so the number of electrons and ions increases rapidly. Since the charged particles drift in the electric field, a negative charge layer forms at the upper tip, and a positive charge layer at the lower tip. When space charge effects become significant,

the discharge is in the streamer regime. The positive front requires a source of electrons ahead of it to propagate. Because these electrons have to be created by photoionization, there is a delay in the propagation of the positive side of the streamer.

After ~ 2.7 ns, a double-headed streamer starts to form. The electric field at the streamer tips is approximately three times the breakdown field. Meanwhile, new avalanches start to appear around the main streamer that formed by the initial seed in the middle. The new avalanches are triggered by photoionization. As the avalanches develop, they overlap and interact with the main streamer, see the second and third columns of Figure 1. Eventually, the middle streamer is completely surrounded by new avalanches.

Similar results were presented by [21, 102], who used a hybrid model, a higher background field of 10 MV/m and a larger ionization seed. Therefore, double-headed streamers form earlier in their simulations. We also performed simulations with a background field of 5 MV/m and with all other conditions as for Figure 1. Similar phenomena were observed as in Figure 1, but after a longer time of ~ 8 ns.

We notice a remarkable difference when we compare our results with 2D fluid model simulations [12, 14, 77]. In contrast to our particle model or to the hybrid model by [21], or to the stochastic fluid model by [28], normal fluid models cannot reveal such pronounced multi-avalanche structures in overvolted gaps.

Photo-ionization plays an essential role for positive streamer formation and propagation, if background ionization can be neglected. Without photoionization or background ionization, only negative streamers are able to form, because there are no seed electrons for the positive streamer to grow. This can for example be seen in simulations by [21] and by [36].

Because the gap is overvolted, the photo-electrons can create new avalanches in the whole space. In an undervolted gap, photoionization would only create avalanches in regions where the electric field is enhanced, close to the streamer. Then a pronounced streamer can emerge, with a larger radius and smoother gradients than without photoionization [88].

6.3.2 Background ionization and photoionization

We now turn to the more realistic case where natural background ionization is included. This important mechanism was missing in previous discharge models in air. The initial conditions now include a homogeneous density of O_2^- and positive ions, both $10^3\,\mathrm{cm}^{-3}$. All other conditions are the same as for the case with photoionization only. Figure 2 shows the electron density and the electric field at 2.67 ns and 2.97 ns. We now compare Figure 2 with the first and the second columns of Figure 1. With background ionization, there are more new avalanches, as they can start from detached electrons as well as from photo-electrons. Figure 1 shows that photo-electrons are mostly generated close to the

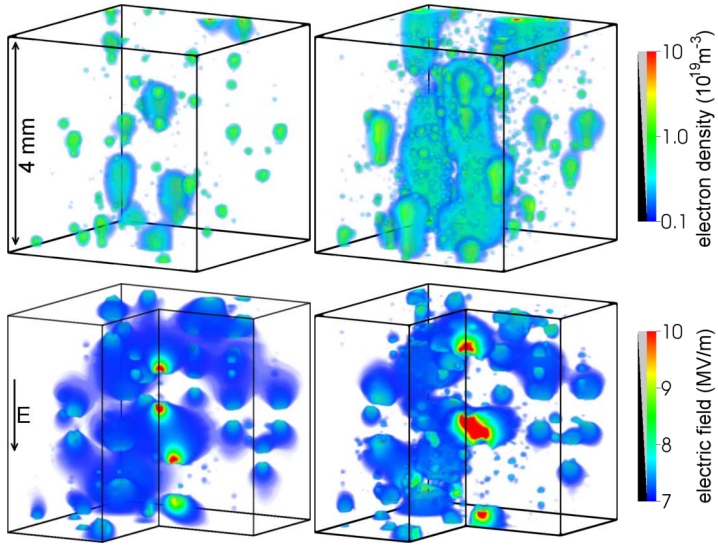

Figure 6.2: The electron density (top row) and the electric field (bottom row) using photoionization and natural background ion density. Times are indicated below each column. The simulation and plots were set up in the same way as for Figure 6.3.1, but now background ionization in the form of O_2^- and positive ions was included, both with a density of 10^3cm^{-3}. Here the full simulation domain is shown from 0 and 4 mm in all directions.

discharge, within 1 mm distance. On the other hand, detachment can happen anywhere, even though it happens faster in higher electric fields. Therefore, the avalanches are much more distributed over the whole domain in Figure 2. As the avalanches grow, they overlap more and more, and it is no longer possible to discern a single streamer. Since the avalanches are close together, the electric field enhancement at their tips is reduced.

Now the difference with the results of 2D fluid model simulations is even greater. Instead of a double-headed streamer, we see a discharge that spreads out over the whole domain. Similar discharges were observed in laboratory experiments by [50] and by [34]. Around a needle shaped high voltage electrode, the field is above breakdown and an ionized 'inception cloud' forms. Farther away from the electrode where the instantaneous field drops below breakdown, the cloud destabilizes into streamer channels.

Therefore the existence of well separated accelerating streamers in the overvolted region near lightning leaders in air, as postulated by [14] and [77], is unlikely.

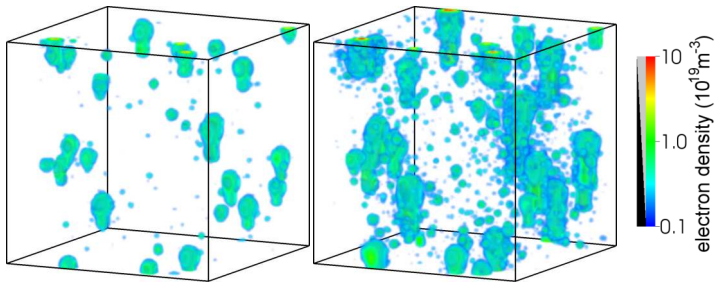

Figure 6.3: The electron density at 2.67 ns and 2.97 ns, using the same simulation parameters as for Figure 6.2, but now without the initial electron-ion pair.

6.3.3 Dependence on the initial seed

Overvolted gaps are sensitive to the initial conditions, because homogeneous breakdown competes with streamer-like breakdown. All fluid model simulations referenced in this paper used big initial electron seeds, without much discussion where these electrons would come from.

For the results presented above, a single electron-ion pair was initially present in the domain. We have also performed the simulation of section 6.3.2 without that initial electron. The electron density at 2.67 ns and 2.97 ns is shown in Figure 6.3. We can see that the discharges start a bit later, due to the delay in the detachment process, and they are also more uniform. Furthermore, we have performed simulations that start with 10 or 100 electron-ion pairs. As expected, with more free seed electrons, the discharge initially grows faster, and is more concentrated around the initial seed.

6.3.4 Ionization screening time

The simulation results we have presented show only the first few nanoseconds of a discharge. Here we will discuss what happens at later times.

If in some region the electric field suddenly rises above the breakdown threshold, then the number of free electrons will grow due to impact ionization. The electrons drift in the field and leave positive ions behind, and this charge separation reduces the electric field in the interior. After some time τ_{is}, the electric field in the interior drops below the breakdown threshold. This we call the 'ionization screening time'. We note that [77] introduced a similar time scale, which was named 'critical time'. For screening to happen, there have to be some free electrons in the overvolted region. These are clearly present above ~ 60 km, but in the troposphere they can appear, for example, due to electron detachment from O_2^- ions.

We first determine τ_{is} using a plasma fluid model, then we give a more general

analytical approximation. We use a simple geometry: there is a uniform electric field E_0, pointing in the negative z-direction, and the initial electron and ion density are n_0 for $z_0 < z < z_1$, elsewhere they are zero. The length $z_1 - z_0$ is taken sufficiently large, then the results do not depend on this length. Figure 6.4 shows the ionization screening time for different fields E_0, starting from an initial density $n_0 = 10^3$ cm^{-3} of electrons or O_2^- ions.

Analytical approximations to these curves are also shown, these are based on a few assumptions: there is no diffusion and the electrons keep their initial drift velocity $v_d(E_0)$ and effective ionization coefficient $\alpha(E_0)$. In the geometry described above, there are then no electrons below $z_0 + v_d t$, as they drift up. The ion density between z_0 and $z_0 + v_d t$ is equal to $n_0 e^{\alpha(z-z_0)}$, so the integrated charge along the z-coordinate is $(e^{\alpha v_d t} - 1)en_0/\alpha$, where e is the elementary charge. Equating this to the charge $\epsilon_0 E_0$ needed to screen an electric field E_0, and solving for t gives the ionization screening time

$$\tau_{\text{is}} \approx \ln\left(1 + \frac{\alpha\epsilon_0 E_0}{en_0}\right)/(\alpha v_d), \tag{6.1}$$

where ϵ_0 is the vacuum permittivity. Using the values α and v_d for the initial field E_0 underestimates the ionization screening time; to compensate for this we compute the time to shield the electric field completely to zero. Note that in the limit $\alpha \to 0$, (6.1) reduces to the dielectric relaxation time $\epsilon_0/(en_0\mu_0)$, with $\mu_0 = v_d/E_0$, also known as the 'Maxwell time' [13]. If we start with negative ions, the delay due to the detachment time τ_D can be included by adding a term $\ln(1 + \alpha v_d \tau_D)/(\alpha v_d)$ to (6.1).

Figure 6.4 also includes the detachment time [145] and the typical streamer formation time based on the Raether-Meek criterion. When the electric field is sufficiently above breakdown, the ionization screening time is close to the streamer formation time. Then, from these time scales alone, we can say that the presence of natural background ionization inhibits the formation of isolated streamers. The reasoning behind this statement is as follows: When there are many seeds, many streamers try to form. Their collective charge separation quickly screens the electric field in the interior of the discharge, which halts the growth of streamers there. Then the discharge grows only at the boundary of the screened, originally overvolted, region.

Under certain conditions, for example when the electric field rises more slowly to a value above breakdown, many streamer-like channels might form that together shield the electric field. We leave this for future research, and note that in such a case one cannot speak of isolated streamers.

In a field of 7 MV/m we find that $\tau_{\text{is}} = 3.2$ ns if an initial density of 10^3 cm^{-3} O_2^- ions is present. These conditions correspond to the simulations shown in Figure 6.2 and 6.3, which end at 2.97 ns. It was not possible to simulate up to the screening time, because the number of free electrons increases rapidly before screening, dramatically slowing down our particle code.

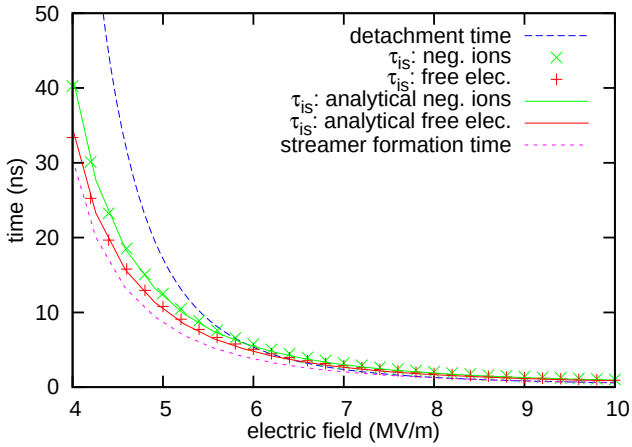

Figure 6.4: The ionization screening time τ_{is} for a preionization density $n_0 = 10^3$ cm^{-3} of electrons or negative O_2^- ions. The corresponding analytical approximations are also shown, see section 6.3.4. Furthermore we include the detachment time and the streamer formation time, based on the Raether-Meek criterion: $18/(\alpha v_d)$ at 1 bar.

6.3.5 Discharges at higher altitudes in the atmosphere

At higher altitudes in the atmosphere, the role of background ionization is qualitatively similar, as was stated in [148]. But there are quantitative differences: First, based on scaling laws, the ionization density, the spatial extension and duration and the electric fields in the streamer tip scale with air density, but natural density fluctuations, photo-ionization and air heating do not simply scale [2]. In the mesosphere where sprite discharges occur, photoionization is about 30 times more efficient than at ground level, because there is no collisional quenching of the photo-emitting states. Furthermore, cosmic radiation supplies a higher level of background ionization, also in the form of free electrons; therefore in the ionosphere electrons start avalanches and screening ionization waves as soon as the electric field increases; they are seen as halos [140, 149]. At lower altitudes like the night time mesosphere, electrons are predominantly attached, but bound as O^- rather than as O_2^- as at ground altitude. Electron detachment from O^- was included into discharge models by [140] and by [147]. If previous cosmic radiation or discharges have supplied sufficient O^-, this ion density can even detach so many electrons that the local breakdown field almost vanishes [140].

6.4 Conclusion

We have studied steamer formation in atmospheric air at ground altitude with a 3D particle code, including the effects of background ionization. Due to detachment of electrons from O_2^- ions, isolated streamers do not emerge in our simulations in fields above breakdown. Instead, many new avalanches appear, that overlap as they grow. This creates a discharge in the whole region above the breakdown field, in agreement with experimental observations [34, 50]. An analysis of the ionization screening time, after which there is global breakdown, leads to the same conclusion. Photo-ionization has a similar effect as background ionization, as was already observed by [21] and [28]. But because photo-electrons are mostly produced close to the discharge, a more localized structure emerges.

Discharges at higher altitudes like halos and sprites evolve in a qualitatively similar manner though ionization rates due to cosmic radiation and reactions of electron attachment and detachment differ quantitatively.

This is the reason why double-headed streamers in the troposphere and double-headed sprites in the mesosphere rarely exist, as was observed by [150]. If the electric field is above breakdown in a larger region, the breakdown is rather uniform due to background ionization and electron detachment, while if the field is below breakdown, positive streamers emerge and propagate much more easily than negative ones [12, 151].

Chapter 7

A time scale for electrical screening in pulsed gas discharges

The Maxwell time is a typical time scale for the screening of an electric field in a medium with a given conductivity. We introduce a generalization of the Maxwell time that is valid for gas discharges: the *ionization screening time*, that takes the growth of the conductivity due to impact ionization into account. We present an analytic estimate for this time scale, assuming a planar geometry, and evaluate its accuracy by comparing with numerical simulations in 1D and 3D. We investigate the minimum plasma density required to prevent the growth of streamers with local field enhancement, and we discuss the effects of photoionization and electron detachment on ionization screening. Our results can help to understand the development of pulsed discharges, for example nanosecond pulsed discharges at atmospheric pressure or halo discharges in the lower ionosphere.

This chapter has been published as [122]:
A time scale for electrical screening in pulsed gas discharges, J. Teunissen, A.B. Sun, U. Ebert, *J. Phys. D: Appl. Phys.* 47, 365203 (2014)

7.1 Introduction

When a weakly ionized plasma is exposed to an external electric field, charges will move to screen the plasma interior from the field. A typical time scale for this process is the Maxwell time, also known as the dielectric relaxation time [130], that depends on the mobility and density of charge carriers in the plasma. In this paper, we present a generalization of the Maxwell time that is also valid for electric fields above breakdown, by taking into account charge multiplication. We call this generalization the *ionization screening time*.

Our motivation for investigating electric screening in discharges came from two other articles [110, 121], in which we simulated the breakdown of ambient air. We included background ionization in the form of negative ions, from which electron avalanches could grow after electron detachment. These avalanches together started screening the electric field, but we could not simulate up to the end of this process. Therefore, we briefly introduced the concept of an *ionization screening time* in [110]. This name was inspired by a similar phenomenon: after a lightning stroke, ionization screening waves can form in the lower ionosphere, also known as halos [149].

In this paper, we investigate the ionization screening time in more detail. The paper is organized in the following way. In section 7.2 the Maxwell time is discussed and the ionization screening time is introduced. Our analytic estimate for the ionization screening time is compared with simulation results in section 7.3. These simulations are performed in 1D and 3D, using a fluid and a particle model. For low levels of initial ionization, discharges become inhomogeneous and local field enhancement becomes important, which is investigated in section 7.4. Finally, we discuss the effect of electron detachment and photoionization on the screening process in section 7.5, which is especially relevant for air.

7.2 The ionization screening time

Below, we first discuss the Maxwell time, also known as the dielectric relaxation time [130]. Then we introduce the ionization screening time, for which we give an analytic estimate.

7.2.1 The Maxwell Time

Although the Maxwell time is valid for any medium with a constant conductivity, we focus here on the case of a plasma. Suppose we have a neutral plasma with an electron density n_e on which an electric field $\boldsymbol{E}$ is applied. The field accelerates the electrons in the plasma, while collisions slow them down again. This gives rise to an electrical current

$$\boldsymbol{J}_e = e n_e \mu_e \boldsymbol{E}, \tag{7.1}$$

where μ_e denotes the electron mobility and e the elementary charge. (We ignore the much smaller contribution of the ions.) This current reduces the electric field inside the plasma. By taking the divergence of Ampère's law, we can relate the current to the time derivative of the electric field

$$\nabla \cdot (\boldsymbol{J}_e + \varepsilon_0 \partial_t \boldsymbol{E}) = 0, \tag{7.2}$$

where ε_0 is the dielectric permittivity. This equation can be interpreted more easily if we assume the system is planar, i.e., effectively one-dimensional, so that we get a scalar equation. If a constant external field E_0 (i.e., $\partial_t E_0 = 0$) is applied from some location outside the plasma, integration of (7.2) gives

$$\partial_t E = -J_e/\varepsilon_0 = -(e n_e \mu_e/\varepsilon_0)\, E. \tag{7.3}$$

A typical time scale for electric screening is given by $-E/\partial_t E$, which is called the Maxwell time:

$$\tau_{\text{Maxwell}} = \varepsilon_0/(e n_e \mu_e). \tag{7.4}$$

For a different derivation see [152]. Note that there is no dependence on the density profile at the plasma boundary.

7.2.2 The ionization screening time

When the field E_0 applied to a plasma is above the breakdown threshold, the Maxwell time is no longer valid, because the electron density n_e grows in time.

We present a generalization of the Maxwell time, which we call the *ionization screening time* or τ_{is}. It estimates how long it takes for the electric field inside a discharge to drop below the breakdown threshold. Below we present a derivation, the result of which is

$$\tau_{\text{is}} = \ln\left(1 + \frac{\alpha_{\text{eff}} \varepsilon_0 E_0}{e n_0}\right) / (\alpha_{\text{eff}} \mu_e E_0), \tag{7.5}$$

where α_{eff} is the effective ionization coefficient. Note that in the limit $\alpha_{\text{eff}} \to 0$, equation (7.5) reduces to the Maxwell time (7.4).

7.2.3 Analytic estimate

To derive an analytic estimate for the ionization screening time, we study a simplified system. The assumptions are listed below:

- The system is planar (effectively one-dimensional); there is spatial variation in the x-direction only.
- Initially, the electron and ion density is n_0 between x_0 and x_1, and zero elsewhere. The width $x_1 - x_0$ is taken larger than the distance the electrons will drift up to the ionization screening time.

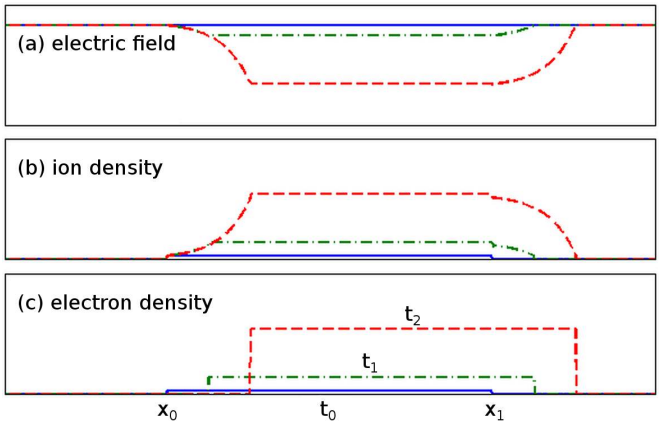

Figure 7.1: Schematic view of (a) electric field, (b) ion density and (c) electron density at three times $t_0 < t_1 < t_2$. The electric field decreases in the ionized region due to the charge separation at the left and right boundary.

- Electrons keep the same drift velocity $v_d = \mu_e E_0$ and effective ionization coefficient α_{eff} as in the initial background field E_0.

- There is no diffusion.

The evolution of this system will resemble the one depicted in figure 7.1. The electrons, which are initially present between x_0 and x_1, drift to the right with velocity v_d. Their number density grows in time as $e^{\alpha_{\text{eff}} v_d t}$. At time t there are no electrons below $x_0 + v_d t$, while they have created an ion density $n_0 e^{\alpha_{\text{eff}}(x-x_0)}$ between x_0 and $x_0 + v_d t$. Therefore, the integrated net charge in this region is $(e^{\alpha_{\text{eff}} v_d t} - 1) e n_0 / \alpha_{\text{eff}}$. Equating this to the charge $\varepsilon_0 E_0$ needed to screen an electric field E_0, and solving for t gives the following expression for the ionization screening time

$$\tau_{\text{is}} = \ln\left(1 + \frac{\alpha_{\text{eff}} \varepsilon_0 E_0}{e n_0}\right) / (\alpha_{\text{eff}} v_d), \tag{7.6}$$

where v_d can be replaced by $\mu_e E_0$.

In deriving equation (7.6) we have assumed that α_{eff} and v_d keep their values for the initial field E_0. This approximation becomes more accurate if the initial electron density n_0 is small compared to the density at the time of screening. Then the electric field stays close to E_0 during most of the screening process, because the charge density is not yet large enough to affect it. Note that by using these initial coefficients we will underestimate the ionization screening time. This is somewhat compensated for by computing the time to shield the electric field to zero, instead of to a value below breakdown.

7.3 Comparison with simulations

We will now compare the predictions of equation (7.6) with numerical simulations. In these simulations, we determine how long it takes for the electric field inside a discharge to drop below the breakdown threshold. We perform these comparisons in nitrogen at 1 bar and 293 Kelvin, for which we have used a breakdown field of 3 MV/m. (Since there are no electron loss mechanisms in pure nitrogen, the breakdown field is not well-defined.) Below, we describe the simulation models.

7.3.1 Simulation Models

We use two types of simulation models here: a plasma fluid model (1D) and a particle model (1D and 3D). It will turn out that in 1D, the fluid model gives almost the same results as the particle model. We nevertheless include both, to provide a link between the 3D particle simulations presented in section 7.3.3 and the plasma fluid description used for equation (7.6).

In all cases, a spatial resolution of 8 μm and a time step of 1 ps was used. In 1D, the computational domain was 16 mm long. In 3D, the computational domain measured 8 mm along the x-direction, with an area of 1×1 mm^2 in the transverse direction. To get the planar structure of the 1D simulations in 3D, we have used periodic boundary conditions in the transverse direction.

1D fluid model

The plasma fluid model that we use is of the drift-diffusion-reaction type [29]. It contains the following equations:

$$\partial_t n_e = \nabla \cdot (\mu_e \boldsymbol{E} n_e + D_e \nabla n_e) + \alpha_{\text{eff}} \mu_e |\boldsymbol{E}| n_e, \tag{7.7}$$

$$\partial_t n^+ = \alpha_{\text{eff}} \mu_e |\boldsymbol{E}| n_e, \tag{7.8}$$

$$\nabla \cdot \boldsymbol{E} = e(n^+ - n_e)/\varepsilon_0, \tag{7.9}$$

where D_e is electron diffusion coefficient and n^+ is the density of positive ions. In the simulations, the coefficients μ_e, D_e and α_{eff} depend on the local electric field, which is recomputed at every time step. These coefficients are computed from the particle cross sections [124] by measuring the properties of simulated particle swarms, see [153]. The same coefficients are used for equation (7.6).

The fluid equations are solved with a third-order upwind scheme, as in [29]. Time stepping was done with the classic fourth order Runge-Kutta scheme.

3D particle model

The 3D model is of the PIC-MCC type, with electrons as particles and ions as densities. The electrons randomly collide with a background of neutral molecules.

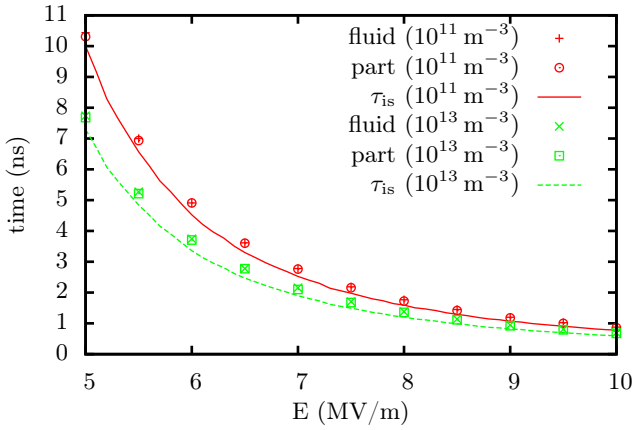

Figure 7.2: The ionization screening time versus the applied electric field, for two initial plasma densities (10^{11} and 10^{13}). Results are shown for a 1D fluid model, a 1D particle model and equation (7.6), for N_2 at 1 bar.

We use cross sections from the Siglo database [124], Fishpack [30] to compute the electric potential and adaptive particle management for the super-particles [104]. This model is described in some detail in [110, 121].

1D particle model

The 1D particle model was constructed from the 3D particle model described above. The 3D model is converted to 1D by projecting the particles onto one spatial dimension for the calculation of the electric field. The particles then have just one coordinate for their position, but their velocities still have three components.

7.3.2 Comparison with 1D simulations

We now compare our analytic approximation to the two numerical simulation models. In figure 7.2, we show the screening time for fields between 5 and 10 MV/m. Two initial conditions are used: an electron and ion density of 10^{13} or 10^{11} m^{-3} was present between 12 and 14 mm. Equation (7.6) predicts shorter screening times than we see in the simulations, but the agreement is nevertheless quite good. Note that the particle and fluid model give almost identical results.

As discussed in section 7.2.3, the partial screening of the electric field was not included in deriving equation (7.6). An example of this partial screening is shown in figure 7.3, where the electric field and the electron density are shown at various times, using the 1D fluid model in a background field of 6 MV/m. Close to the screening time, the exponential growth of the electron density slows down, because the field is partially screened.

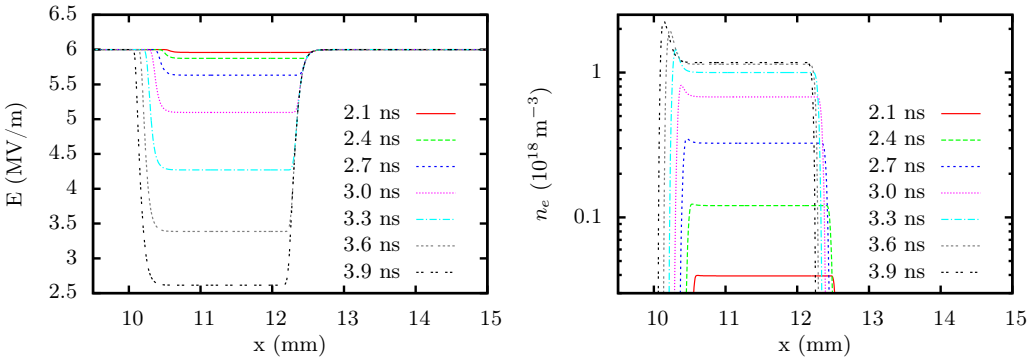

Figure 7.3: Partial screening of the electric field in the 1D fluid simulations, for a background field of 6 MV/m and an initial plasma density of 10^{13} m^{-3}. The electric field (left) and the electron density (right) are shown at various times. The exponential growth of the electron density slows down because the electric field gets screened.

7.3.3 Comparison with 3D simulations

To investigate how inhomogeneities affect the ionization screening time, we have performed 3D particle simulations in a field of 6 MV/m.

We will show results using two initial plasma densities: 10^{13} and 10^{11} m^{-3}. In both cases, the plasma is initially present between 4 and 6 mm. Because the electric field is now a varying 3D vector field, we cannot directly compare it to the 1D results. Therefore, we show the electric field and the electron density averaged over transverse planes. This leaves only the longitudinal component of the field nonzero, due to the periodic boundary conditions.

We first present the results for an initial density of $n_0 = 10^{13}\,\mathrm{m}^{-3}$ between 4 and 6 mm. In figure 7.4 we present averaged electric field and electron density profiles at various times. In figure 7.5, a 3D view of the electron density at 4.05 ns is shown. The screening time is about 3.75 ns, as in the 1D case of figure 7.3. Some noise can be seen in the electric field and density profiles, because the initial density corresponds to 10^4 electrons per mm^3.

With an initial density of $n_0 = 10^{11}\,\mathrm{m}^{-3}$, the results look quite different, see figures 7.6 and 7.7. There is now significant noise in the electric field and especially in the electron density profiles. These larger fluctuations emerge because the initial density corresponds to only 10^2 electrons per mm^3. The screening time is about 5.1 ns, which is still in agreement with the 1D results of figure 7.2.

Compared to the 1D results, we observe almost the same screening times in 3D, but with lower initial densities fluctuations become larger. If we would further reduce the initial electron density, we would eventually get a few separated electron avalanches that develop into streamers.

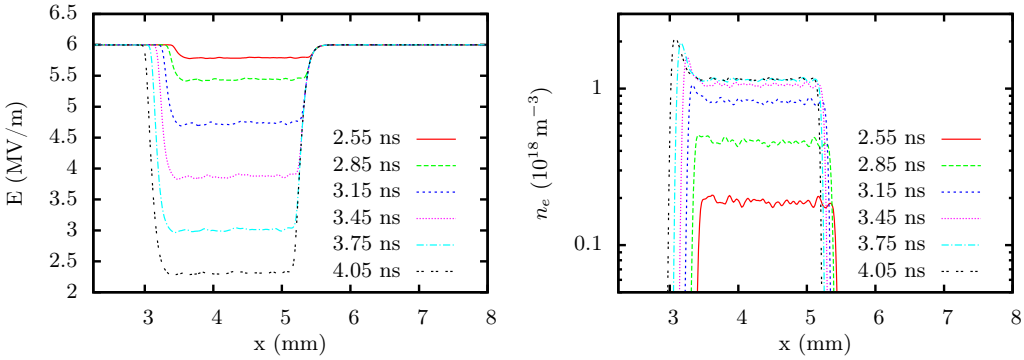

Figure 7.4: Electric field and electron density in the 3D particle simulations, for a background field of 6 MV/m and an initial density of $n_0 = 10^{13}\,\mathrm{m}^{-3}$. The values are averaged over planes perpendicular to the x-direction.

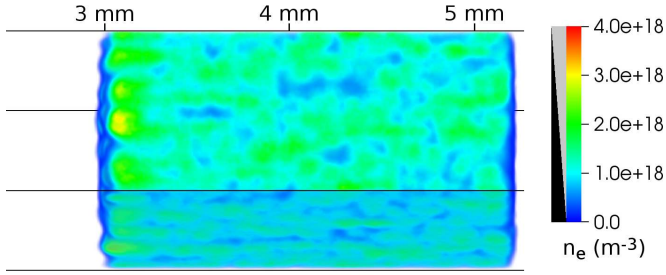

Figure 7.5: The electron density in the 3D particle model at 4.05 ns, for an initial density of $n_0 = 10^{13}\,\mathrm{m}^{-3}$. (This figure is made using volume rendering; transparency is indicated in the legend.)

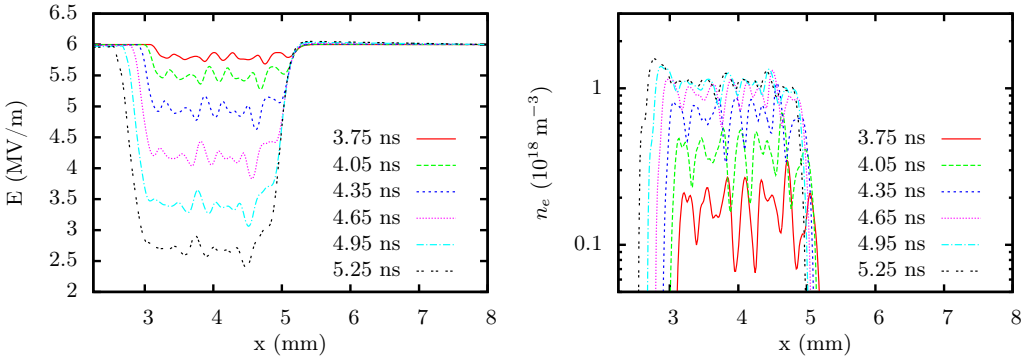

Figure 7.6: Electric field and electron density in the 3D particle simulations, as in figure 7.4, but now for a lower initial density of $n_0 = 10^{11}\,\mathrm{m}^{-3}$.

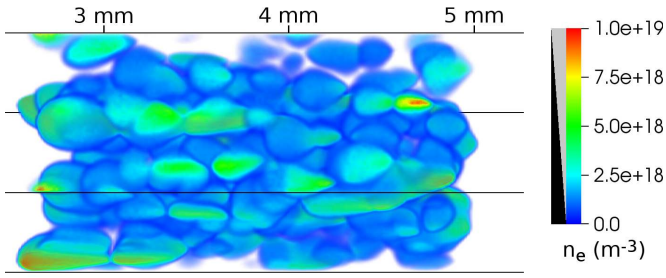

Figure 7.7: The electron density in the 3D particle model at 5.25 ns, for an initial density of $n_0 = 10^{11}\,\mathrm{m}^{-3}$.

7.4 The homogeneity of discharges

In the previous section we have seen that discharges can develop quite irregularly if the initial electron density is low. The irregularities cause field enhancement, that could invalidate our estimate for the ionization screening time. To estimate when this happens, we first discuss how long it takes for space charge effects to develop.

7.4.1 The streamer formation time

If an electron avalanche starts from a single electron, how long does it take for space charge effects to become significant? In other words, how long does it take for a streamer to form? The answer depends on the processes that can affect the space charge fields: ionization, drift and diffusion. The coefficients of these processes can be described in terms of the electric field E and the gas number density N, so that in general the 'streamer formation time' is a function of E and N. According to [2, 91], the number of electrons required for a streamer to form scales as $g(E) \cdot N_0/N$, where $g(E)$ is some function of the electric field and N_0 is the density of air at standard temperature and pressure. Then the time scale for streamer formation can be expressed as

$$\tau_{\mathrm{streamer}} = \ln\left[g(E) \cdot N_0/N\right] / (\alpha_{\mathrm{eff}} v_d).$$

For $N = N_0$, a commonly used empirical approximation is to take $g(E) \approx 10^8$, so that $\ln[g(E)] \approx 18$. This criterion is know as the Raether-Meek criterion, for which the streamer formation time is given by

$$\tau_{\mathrm{RM}} \approx 18/(\alpha_{\mathrm{eff}} v_d). \tag{7.10}$$

7.4.2 Required pre-ionization for homogeneity

From the previous section we have an estimate for the time it takes to develop space charge effects. Given this time, we can estimate how high the initial

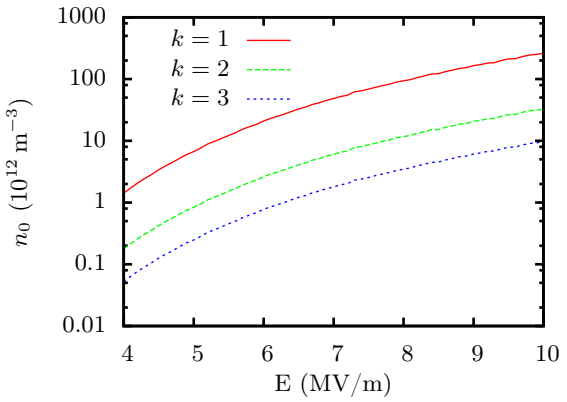

Figure 7.8: The required initial electron density for homogeneous breakdown according to equation (7.11), for three values of k. The curves shown are for N_2 at 1 bar.

electron density n_0 needs to be to prevent streamer formation. Several authors have made such estimates in the past, see for example [154–156]. Much of this research was aimed at generating homogeneous discharges for CO_2 lasers. Below, we derive an alternative criterion for homogeneity that is based on arguments from [154–156], but perhaps simpler.

As long as space charge effects are negligible, the electron avalanche will radially expand due to diffusion. In the radial direction, the electron density at time t will have a Gaussian distribution with a standard deviation of $\sqrt{2D_e t}$. If we let R_s denote the typical radius at the time of streamer formation, see equation (7.10), we get $R_s = 6\sqrt{D_e/(v_d\alpha)}$. If streamer formation is to be prevented, the avalanches need to be sufficiently close to each other. This means that their initial separation should be on the order of R_s. Suppose it is $k \cdot R_s$, where k is about one, then the initial electron density n_0 should be at least

$$n_0 \approx 1/(R_s)^3 = \frac{1}{216k^3}\left(\frac{v_d\alpha}{D_e}\right)^{3/2}. \tag{7.11}$$

In figure 7.8, equation (7.11) is shown against the electric field for N_2 at 1 bar, for three values of k. We can see that the result is quite sensitive to k. Using $1 \leq k \leq 3$, the required initial density lies between 10^{12} and $10^{13}\,\mathrm{m}^{-3}$ for a field of 6 MV/m, in agreement with the results from section 7.3.3.

7.5 The effect of detachment and photoionization

Besides impact ionization, there can be other ways to generate free electrons in a gas, which may affect our estimate for the screening time. This is especially

true for air, in which electron detachment and photoionization can occur. The effect of these processes is discussed below.

7.5.1 Electron detachment

In electronegative gases there might initially be negative ions instead of free electrons. Ionization screening by electrons can still take place in such a gas if electrons are able to detach from the negative ions. If a typical time scale for detachment is τ_d, then the screening process will be delayed by approximately

$$\tau_{\text{delay}} = \ln(1 + \tau_d \alpha_{\text{eff}} v_d)/(\alpha_{\text{eff}} v_d), \tag{7.12}$$

so that the total screening time is given by the sum of equations (7.6) and (7.12)

$$\tau_{\text{is}} = \left[\ln\left(1 + \frac{\alpha_{\text{eff}} \varepsilon_0 E_0}{e n^-}\right) + \ln(1 + \tau_d \alpha_{\text{eff}} v_d)\right] / (\alpha_{\text{eff}} v_d), \tag{7.13}$$

where n^- denotes the negative ion density. Equation (7.12) is the solution to $n_e(\tau_{\text{delay}}) = n^-$ given the following equation

$$\partial_t n_e(t) = n^-/\tau_d + n_e(t)\alpha_{\text{eff}} v_d,$$

with $n_e(0) = 0$. This last equation describes the growth of the electron density in time, but it does not take the depletion of negative ions by detachment into account (n^- should change in time). The underlying assumption is that ionization quickly dominates over detachment. Furthermore, the coefficients τ_d, α_{eff} and v_d are assumed to be constant, since we do not expect the electric field to change during the detachment phase.

Summarizing, if there are negative ions from which electrons first have to detach, then there will be a delay in the ionization screening process. The ionization screening time can then be approximated by equation (7.13).

7.5.2 Photoionization

Photoionization can occur if excited molecules (or atoms) emit photons energetic enough to ionize other molecules (or atoms). With a few assumptions, we can estimate how photoionization will affect the screening time. Suppose that on average η photoionization events take place per electron-impact ionization. Suppose further that these photoionizations take place at a distance that is larger than $n_0^{-1/3}$, where n_0 is the initial density of electrons. If there is no delay in emitting the ionizing photons, and if space charge effects can be neglected, then the electron density will grow as

$$n_e(t) = n_0 e^{(1+\eta)\alpha_{\text{eff}} v_d t}. \tag{7.14}$$

So, photoionization effectively increases α_{eff} with a factor $1+\eta$. For air at atmospheric pressure η is less than 1%, and at low pressures $\eta \lesssim 0.1$ [46]. Therefore photoionization does not change the ionization screening time (7.6) much.

Another effect of photoionization could be to make a discharge more homogeneous. One interpretation of equation (7.14) is that photoionization has effectively increased the initial density n_0 by a factor $e^{\eta\alpha_{\text{eff}}v_d t}$ at time t. From equation (7.10), we get that $\alpha_{\text{eff}}v_d t \approx 18$ when space charge effects set in. For $\eta = 1\%$, the factor $e^{18\eta}$ is about 1.2, so that the effect of photoionization on the homogeneity of a discharge is rather weak. For $\eta \approx 0.1$, the factor is about 6, so that photoionization should be taken into account.

7.6 Conclusion

We have introduced the ionization screening time, a generalization of the Maxwell time that is also valid for electric fields above breakdown. An analytic estimate for this time scale was introduced, which was compared with numerical simulations in 1D and 3D, finding good agreement. We have given an estimate for the required plasma density to prevent the growth of inhomogeneities, and we have discussed the effects of electron detachment and photoionization on ionization screening.

These results can help to understand the development of pulsed discharges, such as nanosecond pulsed discharges at atmospheric pressure or halo discharges in the lower ionosphere. First, our estimate can be used to predict whether such a discharge initially develops homogeneously. If so, then two stages can be distinguished: Before the ionization screening time, growth takes place in the complete discharge volume. After this time, the discharge grows at its boundary, because its interior is electrically screened.

Chapter 8

The inception of pulsed discharges in air: simulations in background fields above and below breakdown

We investigate discharge inception in air, in uniform background electric fields above and below the breakdown threshold. We perform 3D particle simulations that include a natural level of background ionization in the form of positive and O_2^- ions. In background fields below breakdown, we use a strongly ionized seed of electrons and positive ions to enhance the field locally. In the region of enhanced field, we observe the growth of positive streamers, as in previous simulations with 2D plasma fluid models. The inclusion of background ionization has little effect in this case. When the background field is above the breakdown threshold, the situation is very different. Electrons can then detach from O_2^- and start ionization avalanches in the whole volume. These avalanches together create one extended discharge, in contrast to the 'double-headed' streamers found in many fluid simulations.

This chapter has been published as [121]:
The inception of pulsed discharges in air: simulations in background fields above and below breakdown, A.B. Sun, J. Teunissen, U. Ebert, *J. Phys. D: Appl. Phys.* 47, 445205 (2014)

8.1 Introduction

Developments in pulsed power technology have increased the interest in pulsed discharges over the last two decades. These discharges now have a wide range of applications, for example, ozone generation [6, 116, 157, 158], gas and water cleaning [6, 159, 160], flow control and plasma assisted ignition and combustion [161]. Pulsed discharges appear also in thunderstorms and in high voltage technology for electricity networks.

Here, we focus on the initial development of such pulsed discharges in air at standard temperature and pressure, which we study with a 3D particle model. We consider two different cases for the background electric field: it is either only locally above the breakdown threshold, or globally. Homogeneous background fields above breakdown can occur for example between parallel electrodes, or far from charge accumulations, as in thunderclouds [60, 162]. The main objective of the current paper is to show that in air one needs to distinguish between fields above and below breakdown, due to the presence of background ionization.

If the background field is only locally above breakdown, a discharge can only grow in that region, typically forming a *streamer*. Streamers are fast growing plasma filaments that can penetrate into non-ionized regions due the electric field enhancement at their tips. They have been studied in different gases and in different electric field configurations both experimentally [34, 58, 87, 101, 118, 163, 164] and numerically [12, 38, 88, 165–171].

If the background field is globally above breakdown, ionization processes can take place in the whole volume, at least if some background ionization is present to provide the first free electrons. Because most background ionization is present in the form of negative ions, this first requires *electron detachment*, which we discuss in some detail. We will see that the growth of electron avalanches in the whole volume can actually inhibit the formation of separate streamers.

The outline of the paper is as follows. In section 8.2, we first briefly discuss previous work. In Section 8.3, we introduce the simulation model, and discuss background ionization and electron detachment. In Section 8.4, we present simulation results showing streamer formation in fields only locally above breakdown. These results are in qualitative agreement with previous work. Then, in Section 8.5, we investigate discharge formation in background electric fields globally above breakdown. The results here show that the presence of background ionization leads to the formation of a 'global discharge', consisting of many electron avalanches.

8.2 Previous work

Up to now, pulsed discharges in air have mainly been simulated with plasma fluid models [12–18], where the charged particles are approximated by densities. The

most common fluid model assumes that the electrons drift, diffuse and react (ionize), with the coefficients for these processes determined by the local electric field strength. Typically cylindrical symmetry is assumed, and therefore these fluid models need just two spatial coordinates, making them computationally much less expensive than our 3D particle code. Authors typically place some localized initial ionization in the domain to start a discharge [12, 16–18]. In background fields above the breakdown threshold, this ionization seed then develops into a 'double-headed' streamer. The effect of including natural background ionization (and detachment) has not been studied with these models.

In [110], we have recently demonstrated that including background ionization and detachment can be important for discharges in air above breakdown. We there used the same simulation model to compare discharge formation in atmospheric air with and without 'natural' background ionization. The present paper is an extension of [110]. The main differences are that we can discuss the simulation model and the results in more detail here and that we include results in fields only locally above breakdown.

8.3 The set-up of the MC particle model

In recent years, we have developed a 3D particle code of the PIC-MCC (particle-in-cell, Monte Carlo collision) type [172] to study discharge inception. This simulation model is made available on our group's web page [35]. The reason for using a 3D particle model is that the start of discharges is often a stochastic process, that lacks cylindrical (or other) symmetry. In the model, electrons are tracked as particles. Ions are assumed to be immobile, and are included as densities. They only contribute to space charge effects. Neutral gas molecules provide a background that electrons can randomly collide with; they are included in the code as a random background of given density.

The simulations of the present paper are performed in dry air (80% N_2, 20% O_2) at 1 bar and 293 Kelvin. For the electrons, we include elastic, excitation, ionization and attachment collisions with the neutral gas molecules. We use the cross sections from the SIGLO database [124] and the null-collision method to select collisions [173], with isotropic scattering after every collision. We ignore electron-electron and electron-ion collisions, because the degree of ionization in a pulsed discharge in STP air is typically below 10^{-4}, which is also the case in the simulations we perform.

Simulating a discharge with a 3D particle code is computationally expensive, especially as the discharge grows. This limits the simulations we can perform to the first nanoseconds of a discharge, during which the inception takes place. On this time scale, heating, recombination and multi-step excitation or ionization can be neglected.

8.3.1 Adaptive particle management

As the number of electrons in a typical discharge quickly rises to 10^8 or more, so-called super-particles have to be used. Using super-particles with a fixed weight would induce significant stochastic errors, and therefore we employ 'adaptive particle management' as described in [104]. The weight of simulated particles can then be adjusted by merging or splitting them, and care is taken to not alter their properties in a systematic way. A particle i can only be merged with its closest neighbor j that also needs to be merged, with 'closest' defined as minimizing

$$d^2 = (\boldsymbol{x}_i - \boldsymbol{x}_j)^2 + \lambda^2 \left| v_i - v_j \right|^2 , \tag{8.1}$$

where $\boldsymbol{x}$ denotes the Cartesian position vector, v is the norm of the velocity and λ is a scaling factor that we set to one picosecond. A newly formed merged particle gets its velocity at random from one of the original particles, while its position is set to the weighted average position, see [104] for a comparison of different schemes to merge particles. We adjust the weights so that every cell of the grid (see below) contains at least 50 simulation particles. So if no more than 50 electrons are present in a cell, then each simulation particle represents a single electron. But where the electron density is high, with much more than 50 electrons in a cell, most simulation particles represent many electrons.

8.3.2 Adaptive Mesh Refinement for the electric field

In the particle code, the electric field is computed from the electric potential. The potential is computed by solving Poisson's equation with the charge density as the source term, using the HW3CRT solver from the FISHPACK library [30]. When space charge effects become important in a discharge, a grid fine enough to resolve the space charge structures has to be used. In our simulations, we use the following criterion for the grid spacing

$$\Delta x < 1/\alpha(E), \tag{8.2}$$

where $\alpha(E)$ is the ionization coefficient, that describes the average number of ionizations a single electron will generate per unit length in a field of strength E. For air at 1 bar and in an electric field of 15 MV/m, a typical field for streamer tips, this gives $\Delta x \sim 5$ μm. Because a typical simulation domain measures at least a few mm in each direction, using such a fine grid everywhere is computationally infeasible. Therefore, we have implemented block-based adaptive mesh refinement, in the same way as in [29], although now in 3D. First, the electric potential is computed on a uniform, coarse grid. Then the rectangular area that contains the points at which the electric field is larger than some threshold is refined, by a factor of two. The electric potential in the refined rectangle is then computed by imposing Dirichlet boundary conditions interpolated from the

coarse grid. This procedure is repeated with the refinement criterion given by equation (8.2).

For the simulation of streamer discharges, the block-based grid refinement strategy described above works relatively well, because high electric fields are present only in a small region. But for the simulation of discharges that spread out over the whole domain, as we will see in section 8.5, this type of grid refinement does not reduce the computational cost much.

8.3.3 Photoionization

Photoionization provides a non-local ionization mechanism in air. This is especially important for the propagation of positive streamers, that need a source of free electrons ahead of them to propagate. We use the same approach as in [11, 102], where a discrete, stochastic version of Zheleznyak's photoionization model [46] is implemented. In this model, the average density of ionizing photons S_{ph} produced at $\boldsymbol{r}$ is given by

$$S_{\mathrm{ph}}(\boldsymbol{r}) = S_{\mathrm{ion}}(\boldsymbol{r})\ \eta(E), \tag{8.3}$$

where S_{ion} represents the number of ionizations and $\eta(E)$ is an efficiency, estimated from experimental measurements, that depends on the local electric field E and the gas mixture. When an ionizing photon is generated, its place of absorption is determined using random numbers, and at that position an electron-ion pair is created. The average absorption distance is about 0.5 mm in air at 1 bar. For details about the implementation of the photoionization model we refer to [11].

8.3.4 Electron detachment from background ionization

In atmospheric air, there is always some background ionization present, due to radioactivity and cosmic or solar radiation. Previous discharges can also play a role, both in nature [174] and in the lab [34]. At standard temperature and pressure, the free electrons that are created by these sources attach to oxygen molecules mostly by three-body attachment [145]:

$$\mathrm{e} + \mathrm{O_2} + \mathrm{O_2} \quad \rightarrow \mathrm{O_2^-} + \mathrm{O_2}, \tag{8.4}$$
$$\mathrm{e} + \mathrm{N_2} + \mathrm{O_2} \quad \rightarrow \mathrm{O_2^-} + \mathrm{N_2}. \tag{8.5}$$

These negative ions have a longer life time than the electrons. Inside buildings, background ionization levels of 10^3 – $10^4\,\mathrm{cm}^{-3}$ are typical, primarily due to the decay of radon, see [57] for a review. When $\mathrm{O_2^-}$ molecules collide with a neutral gas molecule, the attached electrons can detach again, so that reactions (8.4) and (8.5) are reversed:

$$\mathrm{O_2^-} + \mathrm{O_2} \quad \rightarrow \mathrm{e} + \mathrm{O_2} + \mathrm{O_2}, \tag{8.6}$$
$$\mathrm{O_2^-} + \mathrm{N_2} \quad \rightarrow \mathrm{e} + \mathrm{N_2} + \mathrm{O_2}. \tag{8.7}$$

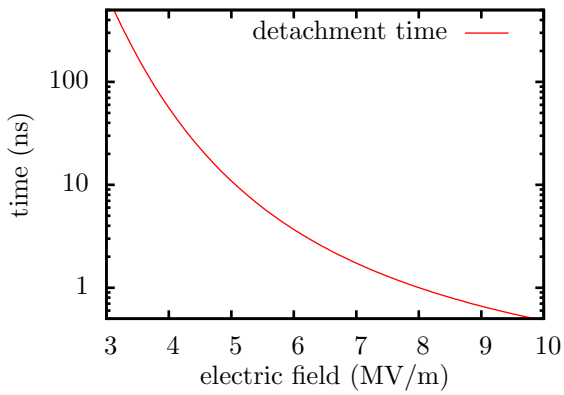

Figure 8.1: The detachment time τ_D as a function of the electric field strength in STP air. In higher fields, negative ions have a higher energy and drift faster, so they are more likely to lose an electron in a collision with a neutral molecule.

We use the following rates for reactions (6) and (7), which were taken from [145] (reactions (56) and (57) in that reference)

$$k_{8.6} = 2.7 \cdot 10^{-10} \sqrt{T/300}\, \exp(-5590/T)\ \mathrm{cm^3\,s^{-1}}, \tag{8.8}$$

$$k_{8.7} = 1.9 \cdot 10^{-12} \sqrt{T/300}\, \exp(-4990/T)\ \mathrm{cm^3\,s^{-1}}, \tag{8.9}$$

where T is the gas temperature in Kelvin, in the absence of an electric field. In our case, there is an applied electric field, which means that the ions will have a higher effective temperature T_{ion} than the background gas. It was suggested to us [175] to take T as the average of the gas and the ion temperature,

$$T = (T_{\mathrm{gas}} + T_{\mathrm{ion}})/2, \tag{8.10}$$

with the latter given by

$$T_{\mathrm{ion}} = T_{\mathrm{gas}} + \frac{2}{3k_B} M_{\mathrm{ion}}\, v_{\mathrm{ion}}^2 = T_{\mathrm{gas}} + \frac{2}{3k_B} M_{\mathrm{ion}}\, (\mu_{\mathrm{ion}} E)^2, \tag{8.11}$$

where k_B is the Boltzmann constant, M_{ion} the ion mass, v_{ion} the ion drift velocity, and μ_{ion} the ion mobility, which we approximate by $\mu_{\mathrm{ion}} = 2 \cdot 10^{-4}\,\mathrm{m^2/(Vs)}$. Note that it is assumed that the total energy of the ion is twice the 'drift energy' $(M_{\mathrm{ion}}\, v_{\mathrm{ion}}^2/2)$ [176].

Using equations (8.6)–(8.11), we can compute the total rate at which electrons detach from O_2^- ions in a given electric field E. We call this rate the *detachment rate*, and its inverse the *detachment time* τ_D. In figure 8.1, the dependence of τ_D on the electric field strength is shown. At the breakdown field (3 MV/m) the detachment time is about 500 ns, but at 5 MV/m it is only 10 ns.

We currently consider only O_2^- ions for detachment, although O^- can also form due to dissociative attachment, mostly at lower pressures or higher electron energies. From these ions electrons can detach in fields much below breakdown [140, 147, 177]. Furthermore, many other types of ions can be generated by chemical reactions [140, 178].

We want to emphasize that both detachment and photoionization are characteristic for nitrogen/oxygen mixtures. In pure gases or other mixtures these processes might be absent or much weaker, see e.g. [87].

8.4 Discharges in background fields below breakdown

In this section, we show an example of discharge formation in background electric fields below breakdown. Of course, the field has to exceed the breakdown threshold in some region, otherwise a discharge cannot start. Such a region can be created by sharp electrodes or by polarizable object such as dust, water droplets or ice crystals. We use a method that has been commonly used in fluid simulations of streamers for the past 30 years, namely to place an ionized seed in the domain [12, 16–18]. The electrons in such a seed move in the background field, polarizing the seed, so that the electric field gets enhanced at the endpoints. Our results are in agreements with previous modeling efforts, and the inclusion of background ionization has little influence.

8.4.1 Conditions for the simulations below breakdown

The computational domain that we use for fields below breakdown is shown in figure 8.2. It contains two parts: an interior grid of $5 \times 5 \times 10$ mm^3, in which we use the particle model, and a four times larger grid around it that is used to set the boundary conditions for the electric potential on the interior grid. Dirichlet boundary conditions are imposed on the sides of the larger grid to get a homogeneous background field $E_0 < E_k$ in the vertical direction. Inside the interior grid we use adaptive mesh refinement, so that the strong electric fields around streamer heads can be resolved. As a background gas we use dry air at 1 bar and 293 Kelvin, with a density of 10^4 O_2^- ions per cm^3, and an equal density of positive ions.

8.4.2 Results

We use a long, neutral ionized column, similar to the initial condition used in [151], but then scaled to ground pressure. The peak ion and electron density is 1.3×10^{13} cm^{-3}. In the two lateral directions, the distribution of electrons and ions is Gaussian, with a width of 0.2 mm. The distribution of plasma in the vertical direction is uniform over a length of 4 mm; at the endpoints there is

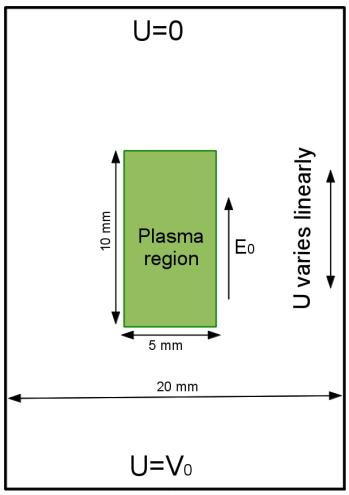

Figure 8.2: Schematic view of the computational domain in the undervolted simulations. The simulated plasma region is embedded in a four times larger domain.

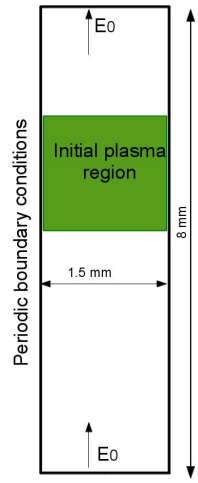

Figure 8.3: Schematic view of the computational domain in the 3D overvolted simulations. Periodic boundary conditions are used in the two lateral directions. Initially background ionization is present in the green region.

again a Gaussian distribution. An external electric field of $\sim 0.5E_k$ is applied in the vertical direction.

Figure 8.4 shows how this seed develops further in the simulations. First, the column gets polarized, and negative and positive charge layers emerge at the top and bottom of the column, respectively. After ~ 10 ns, a positive streamer forms at the upper end of the column, as shown in the first row of figure 8.4. At the lower end, electrons spread out or attach to neutral molecules. On the time scales that can be simulated with our particle model, we have not observed negative streamers emerging. An important difference between positive and negative streamers is that positive streamers grow from electrons drifting inwards towards their head, while negative streamers grow from the electrons drifting outwards. Thus, the space charge layer of a positive streamer head is formed by rather immobile ions, while the space charge layer of a negative streamer head is formed by mobile electrons. Negative streamers are therefore typically wider and more diffusive, with less field enhancement, and they do not form as easily [12].

8.4.3 Effect of background ionization

We have repeated the simulations presented above without initial background ionization, and the results showed no apparent differences. There are two reasons

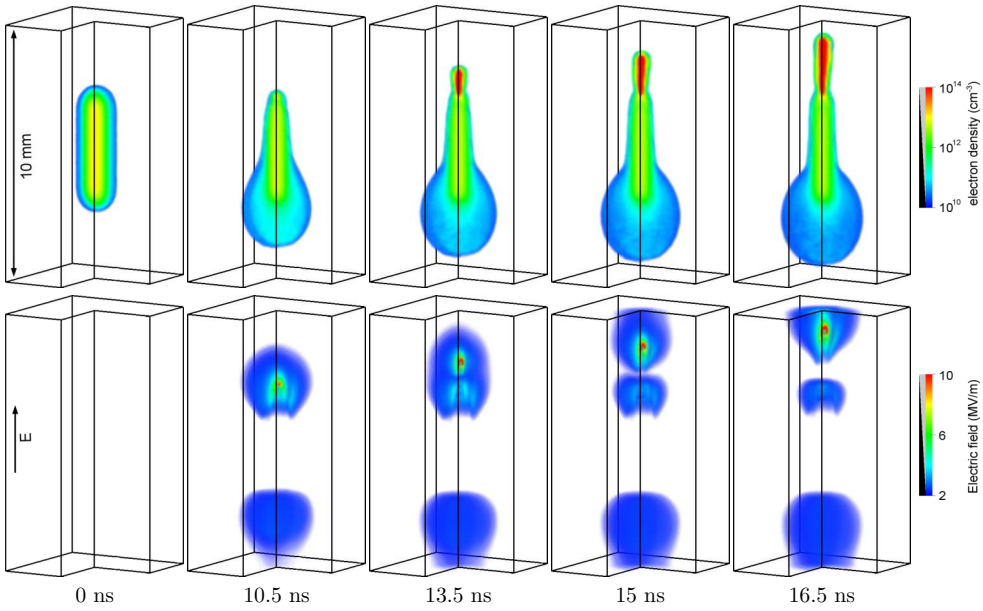

Figure 8.4: Simulation results in a field of 1.7 MV/m, about half the breakdown value. The top row shows the electron density and the bottom row the electric field, at various times. Initially an ionized column is present, that causes local field enhancement. The domain measures 10 mm × $(5\,\mathrm{mm})^2$, and is cut open. The figures were generated using volume rendering; opacity is indicated in the colorbar.

for this. First, photoionization produces most free electrons ahead of the front after the discharge has started [88]. Second, detachment from O_2^- only plays a role in the region above breakdown, but electrons are already present there due to the initial seed. In regions below breakdown, the detachment time is more than 500 ns (see figure 8.1), and if an electron would appear, then it would not produce an ionization avalanche.

The simulation results look qualitatively similar to those obtained with typical 2D fluid simulations. This will be different in the simulations above breakdown presented in the next section.

8.5 Discharges in background fields above breakdown

In this section, we present simulations in a background field globally above breakdown. Below, we first describe the computational domain used for these simulations.

8.5.1 Simulation conditions

In this section, we present new simulation results for a discharge developing in a field of 6 MV/m, twice the breakdown field. The same level of background ionization is present as in section 8.4, namely a density of $10^4\,\mathrm{cm}^{-3}$ O_2^- and positive ions. However, we use a different computational domain here, because we want to simulate the development of a discharge that is not in contact with physical boundaries, like electrodes. Therefore, we use periodic boundary conditions in the x and y direction, while limiting the region where background ionization is present in the z direction. In other words, we simulate the development of a thick discharge layer growing from background ionization. The elongated computational domain is shown in figure 8.3, where the region with background ionization is shaded green. At the top and bottom of the domain we apply Neumann boundary conditions for the electric potential, thereby creating a uniform background field E_0 of 6 MV/m. We remark that in the GRL [110] we were less careful with the boundary conditions, and used something similar to figure 8.2.

We do not use grid refinement to calculate the generated electric field in this simulation, as grid refinement would be required nearly everywhere in the preionized region. The static grid contains $100 \times 100 \times 535$ cells, with a cell length of 15 μm. The domain length is chosen in such a manner that the discharge does not reach its boundaries within the time simulated.

8.5.2 Simulated discharge evolution

Figure 8.5 shows the evolution of the electron density and the electric field in four time steps between 4.5 ns and 5.4 ns. The evolution of the discharge can be

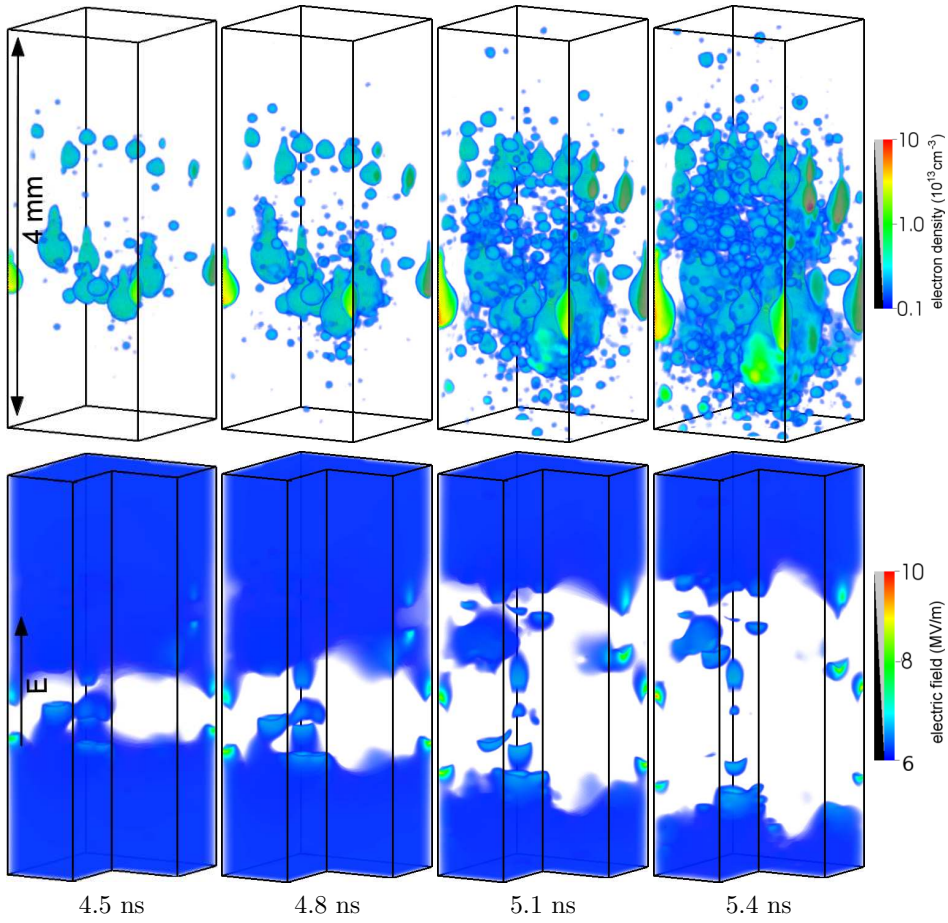

Figure 8.5: The time evolution of the electron density (top row) and of the electric field (bottom row). Background ionization is initially present in the green region of Figure 8.3, in the form of O_2^- and positive ions, both with a density of 10^4 cm^{-3}. The gas is dry air at 1 bar and 293 K in an upward directed homogeneous electric field of 6 MV/m, which is about two times the breakdown field. The domain between 2 mm and 6 mm in the vertical direction of Figure 8.3 is shown.

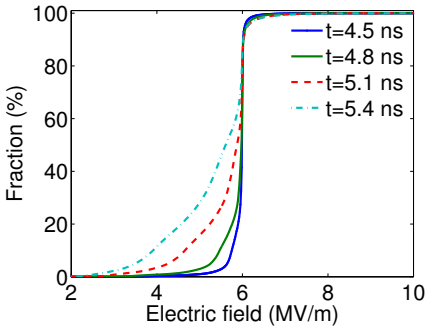

Figure 8.6: The volume fraction with a field smaller than E as a function of E, for the simulation shown in figure 8.5. We evaluate the green domain shown in figure 8.3

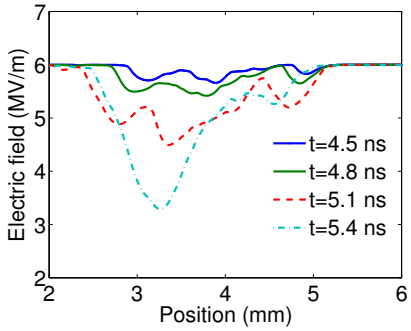

Figure 8.7: The averaged electric field of the simulation shown in figure 8.5. The averaging is performed over planes perpendicular to the background field, see figure 8.3.

characterized as follows. First, free electrons appear due to detachment. As can be seen in figure 8.1, the characteristic detachment time in a field of 6 MV/m is about 3 ns. Then these free electrons start electron avalanches, that quickly grow due to impact ionization. The growing avalanches also produce photoionization, thereby starting additional avalanches. Eventually, many avalanches emerge in the simulation domain.

After about 5 ns, space charge effects start to become important, causing the electric field to increase locally up to $\sim$ 9 MV/m while decreasing elsewhere. These space charge effects increase in magnitude until the simulation stops at 5.4 ns. The distribution of the electric field values is shown in figure 8.6. After 4.5 ns, almost the complete system is still at the background field of 6 MV/m, but after 5.4 ns, about 8% of the volume has a field lower than the breakdown value of 3 MV/m

Figure 8.7 shows the distribution of electric fields in the simulation in another manner; it shows the electric field averaged over the horizontal planes intersecting figure 8.5 and plotted as a function of the longitudinal coordinate. The screening of the electric field occurs in a 'noisy' way, and the electric field varies significantly inside the discharge. This is not so surprising, as initially only about 45 negative ions (O_2^-) are present. With these ions randomly placed in a volume of 4.5 mm^3, we do not expect a discharge homogeneously filled with ionization. The simulation stops when there are too many simulation particles for the computer's memory, which happened here at about $3 \cdot 10^7$ particles.

8.5.3 Effect of background ionization

Above breakdown, we observe a 'global' discharge. Free electrons can appear anywhere in the region containing pre-ionization, due to detachment from O_2^- ions. The electrons then form electron avalanches, building up space charge. The avalanches together reduce the field in the interior of the discharge, which can clearly be seen in figure 8.5. There seems to be competition between local streamer formation and homogeneous breakdown: on the one hand, the collective space charge from all the avalanches inhibits the formation of streamers, which are characterized by strong local field enhancement. On the other hand, there is actually some local field enhancement due to the limited number of avalanches, which makes the discharge rather inhomogeneous.

Our results are very different from previous publications in which 'double-headed' streamers were observed in background fields above breakdown, see for example [11, 14, 15, 38]. Therefore, we think that the inclusion of background ionization (and detachment reactions) is essential for discharge simulations in air above the breakdown threshold. Note that we would observe a similar discharge if we had included a density of free electrons, or other ions from which electrons can detach, instead of O_2^-. The main difference with previous publications is caused by the fact that the pre-ionization is distributed uniformly over the domain, instead of locally into a single seed.

Note the difference between the simulations we have shown below and above breakdown: below breakdown, a localized seed was required to start a streamer discharge, while above breakdown, homogeneous pre-ionization prevents the (immediate) formation of streamers.

8.5.4 Homogeneous breakdown

If we increase the amount of pre-ionization in the simulations, the discharges will become more homogeneous and form a layer, see [179] for a related experimental example. Homogeneous, high-pressure discharges have been generated for use in gas combustion and excimer lasers, see the review in [180]. For the excimer lasers it is important to prevent arc formation, and therefore the discharge should be as homogeneous as possible. Typically, a high level of pre-photoionization is generated for this purpose. There have been several studies estimating the required initial ionization density n_0 to prevent streamer or arc formation [154–156]. These estimates are typically derived by assuming that avalanches should overlap (to some degree) when space charge effects become important. In a recent publication [122], we have also given such an estimate

$$n_0 \approx \frac{1}{216k^3} \left(\frac{v_d \alpha_{\text{eff}}}{D_e} \right)^{3/2}, \tag{8.12}$$

where k is a number around one, v_d the electron drift velocity, α_{eff} the effective ionization coefficient and D_e the electron diffusion coefficient.

In figure 8.5 we could already see that the electric field in the interior in the discharge is reduced in time. In [122], we have presented an estimate for the 'ionization screening time', which is the time it takes for the interior field to drop below breakdown

$$\tau_{\mathrm{is}} = \ln\left(1 + \frac{\alpha_{\mathrm{eff}}\varepsilon_0 E_0}{e n_0}\right) / (\alpha_{\mathrm{eff}} v_d) \tag{8.13}$$

where E_0 is the applied field and n_0 the initial electron density. If there are instead of electrons negative ions, there will be a delay in the screening process because detachment takes some time. This delay can be approximated by [122]

$$\tau_{\mathrm{delay}} = \ln(1 + \tau_d \alpha_{\mathrm{eff}} v_d)/(\alpha_{\mathrm{eff}} v_d), \tag{8.14}$$

where τ_d is the detachment time.

8.6 Conclusion

We have studied pulsed discharge formation in electric fields above and below the breakdown threshold with a 3D particle model for air at standard temperature and pressure. Photoionization, a natural level of $10^4\,\mathrm{cm}^{-3}$ O_2^- ions due to background ionization and electron detachment were included.

In background electric fields below the breakdown value, we observed the formation of a positive streamer if the field is locally sufficiently enhanced. The inclusion of background ionization did not affect the results, and our 3D particle model gives similar results as commonly used 2D plasma fluid models. But in background electric fields above breakdown, we see discharges distributed over the whole domain instead of the 'double-headed' streamers often appearing in other publications. The major cause for this difference is the inclusion of homogeneous background ionization. Free electrons appear at many different places due to detachment from O_2^- ions, and start electron avalanches. These avalanches interact and overlap, and can eventually screen the electric field in the interior of the discharge, which we discuss in a separate paper [122].

The main conclusion of our work is therefore that in background fields above breakdown, the distribution of the pre-ionization determines the evolution of the discharge. Therefore relevant sources of background ionization should be included.

Chapter 9

Streamer discharges can move perpendicularly to the electric field

Streamer discharges are a primary mode of electric breakdown in thunderstorms and high voltage technology; they are generally believed to grow along electric field lines. However, we here give experimental and numerical evidence that streamers can propagate nearly perpendicularly to the background electric field. These streamers are guided by pre-ionization that is orders of magnitude lower than the ionization density in a streamer channel, hardly affecting the background field. Positive streamers could be guided in nitrogen with 0.5 % of oxygen or less, but not in air. This observation also tests the role of photo-ionization in gas mixtures with varying nitrogen-oxygen ratio.

This chapter has been published as [120]:
Streamer discharges can move perpendicularly to the electric field, S. Nijdam, E. Takahashi, J. Teunissen, U. Ebert, *New J. Phys.* 16, 103038 (2014)

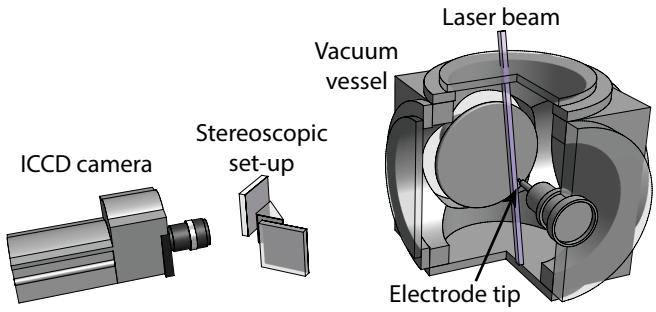

Figure 9.1: Schematic overview of the set-up including the ICCD camera, stereoscopic set-up and vacuum vessel with indication of laser beam path. In the presented images the recorded images are rotated anti-clockwise by 90° so that the electrode tip is shown on top.

9.1 Introduction

Streamer discharges are a primary mode of electric breakdown when a high voltage pulse is applied to ionizable matter. They occur in thunderstorms as streamer coronas ahead of lightning leaders or directly as sprite discharges high above thunderclouds [1–3]. They also occur in plasma and high voltage technology, with applications ranging from air purification to material treatment and wound healing [4–8].

Streamers can have negative or positive polarity, propagating with or against the electron drift. Paradoxically, positive streamers emerge and propagate much more easily than negative ones [12, 51, 151]. Streamers can also guide a gas flow rather than being guided [181]. Here we describe another unexpected observation: positive streamers can propagate nearly perpendicularly to the local electric field.

Streamers are generally believed to follow the direction of the background electric field (i.e., the field in absence of the streamer). When they do not follow the background field lines, they are usually repelled by neighboring streamers [182] and still follow the new local electric field direction. However, we recently have observed cases where streamers propagated along invisible paths in the gas that could not plausibly coincide with the direction of the local electric field. This was, in particular, the case when positive streamers propagated along the edge of a rather spherical cloud of pre-ionization left behind by a previous negative discharge [103]. But in this case it was difficult to establish the actual distribution of pre-ionization and of the electric field at the moment when the positive streamers were growing.

Therefore, we here present experiments with pre-ionization created through very weak laser illumination, allowing full control over timing, location and density of the pre-ionization trail. We demonstrate through experiments and model-

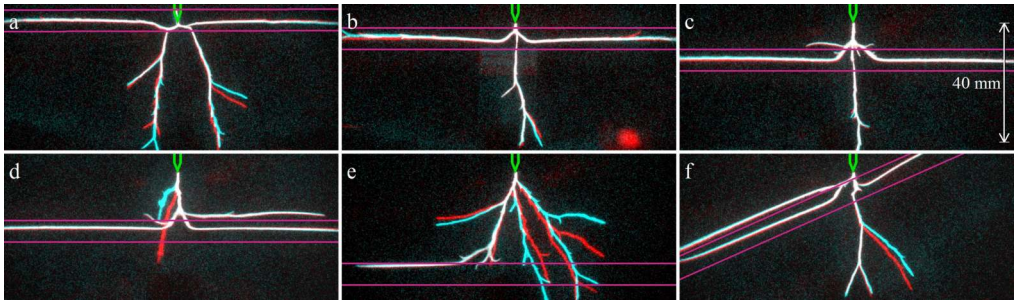

Figure 9.2: Stereoscopic images of laser-guided streamer discharges for various laser positions. The laser path is outlined by the purple lines. The streamers start at the tip, indicated in green. White streamers move in the image plain, the others outside. Measurements in 133 mbar pure nitrogen with a 5.9 kV, 600 ns voltage pulse 1.1 µs after the laser pulse. The images integrate over the duration of the voltage pulse.

ing that positive streamers can follow this very weakly pre-ionized region rather than the electric field lines, even if this pre-ionization is too weak to modify the electric field.

The use of laser-induced weakly ionized channels can help us gain fundamental insight in repetitive discharges [34, 103, 183–185] by providing well-controlled experiments that are easy to model. This can lead to better understanding of the highly complex repetitive discharges themselves. It also allows testing the models of the nonlocal photo-ionization reaction in gas mixtures with varying nitrogen oxygen ratio.

9.2 Set-up and methods of the experiment

We have used an KrF excimer laser to create a trail of increased pre-ionization and studied the effect of this trail on streamer development and morphology, with a focus on streamer guiding. The laser produces pulses at 248 nm of about 1 mJ per 20 ns long pulse at a maximum repetition rate of 10 Hz. The (unfocused) laser beam is shaped by four shutters to a quasi-rectangular beam with a height of 10 mm and a width (perpendicular to the image plane) of 9 mm. This beam enters a vacuum vessel from above as shown in figure 9.1. This vessel contains an electrode tip positioned 103 mm from a cathode plane and is filled with different nitrogen-oxygen mixtures with achieved impurity levels of about 10 ppm. The laser beam was aligned to a symmetry-plane which includes the electrode tip.

The ionization density induced by the laser beam has been determined by measuring the time-integrated current, and thus the total charge, between two flat electrodes induced by a laser pulse. One of these electrodes is grounded and

the other one is kept at a positive potential between 0 and 2000 V. This procedure is performed in the same vessel and under the same conditions as the laser guiding experiments. From the measurements we conclude that the ionization density is about $8 \cdot 10^8\,\mathrm{cm}^{-3}$ in 133 mbar pure nitrogen with the laser at full power, as was the case in all experiments shown here (although guiding was also observed at lower laser powers). In mixtures with low oxygen concentrations the ionization density nearly equals the electron density because attachment and recombination are negligible during our experiments [185].

Some nano- to milliseconds after the laser pulse a single high-voltage pulse was applied to the electrode tip. The voltage pulses have a quite rectangular shape with pulse lengths of 600–1500 ns, rise/fall times of about 15 ns and pulse amplitudes of 4–10 kV. The vacuum vessel and pulse forming network are the same as described in [185].

The created streamers are imaged with an Andor ICCD-camera, combined with a stereo-photography set-up like in past experiments [182, 186]. The full angle between the two image-paths of the stereo set-up is 24 – 28°. The line connecting the two outer mirrors of the stereo-setup is parallel to the line between electrode point and cathode plate, as indicated in figure 9.1. This makes it possible to measure the location of streamers channels following the laser beam. In our previous work the stereo-setup was rotated 90° around its axis to measure channels propagating directly between the point and plane.

During post-processing the images are rotated so that the electrode tip is centered at the top. In our composite (anaglyph) images the images from the two viewing directions are laid over each other in red and cyan respectively. The stereo-images are aligned in such a way that the electrode tip has the same position in both images so that their image plane coincides with the central plane of the experiments. A displacement between the red and cyan colors indicates that channels are located in front or behind the image plane. However, note that channels that propagate vertically down will always be white in the images, regardless of their position. Examples of the stereoscopic images are given in figure 9.2. We can clearly see that many streamer channels are guided by the laser pre-ionization: these streamers propagate parallel to the laser beam and show a very small offset between the red and cyan components. They are therefore mostly represented in white. Each of the figures 9.2a-f also contains some unguided channels that either propagate vertically or show a large, non-constant, offset between the red and cyan trails.

9.3 Experimental observations

9.3.1 General laser guiding observations in nitrogen

All sub-figures from figure 9.2 show guiding of at least one streamer channel. Under the conditions used in this figure, guiding occurred in nearly all imaged

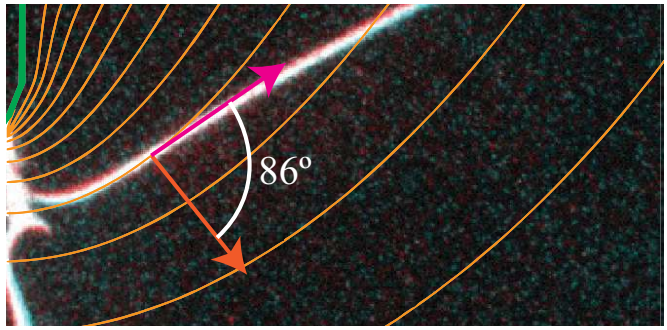

Figure 9.3: Detail of the measurement from figure 9.2(f) with calculated equipotential lines of the background electric field. The arrows indicate the directions of the streamer and the background electric field at a point where streamer and equipotential lines cross.

discharges. In conditions where the laser is quite far from the tip, like in image 9.2e, the streamers have spread out so much that they are clearly in front or behind the image plane (large vertical offset between red and cyan trails), do not cross the laser path and are not guided. There are no indications that the streamers are actually attracted to or repelled from the laser path; this indicates that the laser path does not develop own space charge effects. However, the guided streamer channels can follow the laser beam for a long distance; often until the point where the laser beam enters through the window or where it hits the other wall. The measured velocities of guided streamers are up to 10% higher than those of non-guided streamers. However, this may be caused by the projection, because most of the non-guided streamers propagate out of the image plane. The guided streamers can also become longer than the unguided ones. All streamers in each of the figures 9.2a-f start simultaneously from the electrode tip and therefore also propagate simultaneously, except for the small differences in propagation velocity.

The background electric field has been calculated for a geometry that closely resembles the experimental set-up. This calculation was performed with a cylindrically symmetric Poisson-solver in the COMSOL software package. In figure 9.3 equipotential lines from this calculation are overlaid on a zoom into figure 9.2(f). At one location, the angle between the background electric field and the streamer direction is shown. This angle is about 86°, confirming the observation that the streamer is propagating nearly perpendicularly to the background electric field. Similar angles are found in the other laser guided streamers from figure 9.2 although at most locations the angle is smaller.

9.3.2 Guiding in nitrogen-oxygen mixtures

In our purest nitrogen guiding by the laser is most straightforward. For low pulse voltages and short delays between the laser and the voltage pulses the streamers are almost always guided, when they cross the laser beam. In pure nitrogen the maximal delay between laser and voltage pulse with guiding effect is about 2 ms. However, when the oxygen concentration is increased, guiding becomes more difficult and only occurs for shorter delays between laser and voltage pulses. The highest oxygen concentration for which guided streamers were still observed, was 0.5 %, at a maximal delay between laser and voltage of about 1 µs. Streamer guiding was thus not observed in air with these low pre-ionization densities.

At higher oxygen concentrations free electrons are lost faster but according to our calculations (see [185]) the electron loss at 1 % oxygen in 133 mbar nitrogen is almost the same as in pure nitrogen while in our experiments streamers are not guided in 1 % oxygen. We therefore suggest that the guiding effect is suppressed by photoionization.

9.4 Analysis and modelling

9.4.1 How photoionization can inhibit guiding

Positive streamers need free electrons ahead of them to propagate. These electrons move towards the positive discharge, and where the field is above breakdown, they generate electron avalanches due to impact ionization. In this way, the degree of ionization ahead of the discharge increases, and the discharge gradually extends forwards. The velocity at which this happens depends on the electric field profile ahead and the electron density ahead.

We call a streamer *guided* if it does not grow in the direction of the highest electric field, but in the direction with more free electrons ahead. Streamer guiding requires the electron density ahead of the discharge to be sufficiently anisotropic. Since the laser pre-ionization we use is clearly anisotropic, why we do not observe guiding at higher oxygen concentrations? The reason is photoionization, which produces non-local electrons isotropically around the discharge.

In nitrogen/oxygen mixtures, photoionization can occur when excited nitrogen molecules decay and emit UV photons. The average absorption length l_{abs} of these photons is

$$l_{\mathrm{abs}} \approx 0.093\,\mathrm{mm}\cdot\mathrm{bar}/p_{\mathrm{O_2}},$$

where $p_{\mathrm{O_2}}$ is the oxygen fraction times the pressure [46] (the gas is assumed to be at room temperature). With a higher oxygen concentration there will thus be more photoionization events close to the discharge. To illustrate this, we can estimate the fraction of photons absorbed within one millimeter, using the data from [46]. For 133 mbar with 10 ppm oxygen this fraction is about $5\cdot10^{-5}$ ($l_{\mathrm{abs}} \approx 70\,\mathrm{m}$), for 133 mbar with 1% oxygen it is about 5% ($l_{\mathrm{abs}} \approx 70\,\mathrm{mm}$)

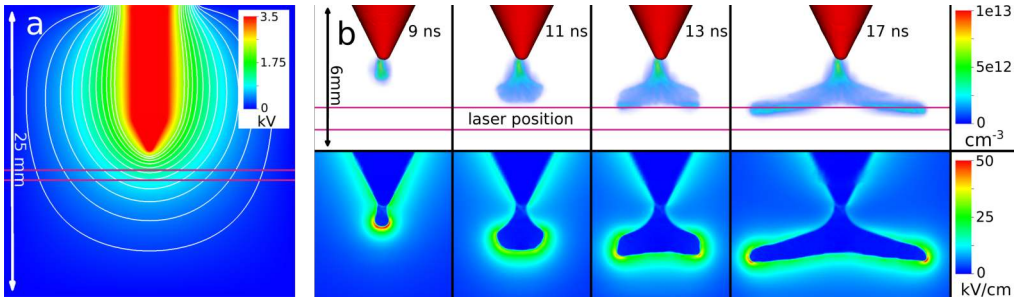

Figure 9.4: (a) The electrical potential at $t = 0$ on a cross section through the center of the 3D computational domain, with equipotential lines. (b) Particle simulation results in N_2 with 10 ppm O_2 at 133 mbar. Top row: the evolution of the electron density, visualized with volume rendering (low densities are transparent); bottom row: slice through the domain showing the evolution of the electric field. Note that (a) and (b) are not to scale and that the laser beam is outlined in purple.

and for atmospheric air it is about 86% ($l_{\text{abs}} \approx 0.5\,\text{mm}$). Because more local photoionization causes a more isotropic distribution of free electrons around the discharge, guiding is inhibited at higher oxygen concentrations.

A related phenomenon was observed in streamer experiments [87]. With a low oxygen concentration, streamers would developed 'feathers' and would propagate in a more erratic way. Because there was little photoionization, the electron density around the streamers was anisotropic, which caused them to not always propagate in the (forward) direction of strongest electric field.

9.4.2 Modelling of laser-induced guiding

We have performed numerical simulations to investigate how a discharge evolves when laser pre-ionization is present.

Simulation model

The simulations were performed with a 3D particle model of the PIC-MCC type. In this model, electrons are tracked as particles. We use *adaptive particle management* [104] to adjust the weights of the simulation particles, which control how many physical electrons they represent. Ions are assumed to be immobile compared to the electrons, and are included as densities. The neutral gas is included as a background that electrons randomly collide with, using the null collision method [123]. Because the simulated discharges have an ionization degree below 10^{-4}, only electron-neutral collisions are included. We use cross sections for elastic, inelastic, ionization and attachment collisions from the SIGLO

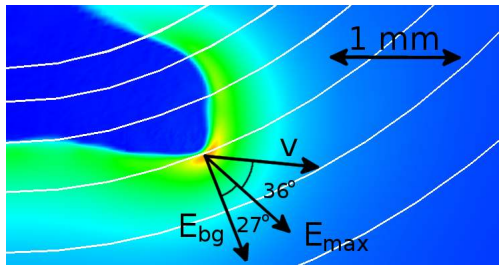

Figure 9.5: Zoom of figure 9.4(b) at 13 ns, showing the electric field strength on a slice through the domain. Arrows indicate the direction of: the background field (E_{bg}), the current maximum field (E_{max}) and the streamer velocity (v). Equipotential lines for the background field are also shown in white.

database [124]. Furthermore, photoionization is included as non-local source of free electrons, implemented similarly as in [11], using data from [46].

The simulations are performed assuming electrostatic conditions. At each time step, the self-consistent electric field is computed from the charged particle densities. To resolve the thin charge layers and high electric fields around streamer heads, we use block-based adaptive grid refinement. An electrode is included by placing 'artificial' charges on its surface which are iteratively updated to keep it close to a desired potential, similar to the charge simulation method. Note that the source code of our simulation model is available online [35].

Simulation conditions

The gas used in the simulations was N_2 with 10 ppm O_2 at 133 mbar and 293 K. The computational domain measured $(25\,\text{mm})^3$, with the finest grid having a resolution of 12.5 µm. In figure 9.4(a) we show the electrical potential at $t = 0$ on a cross section through the 3D computational domain. At the top an electrode is present, at a voltage of 3.5 kV. The other sides of the domain are grounded. At $t = 0$, electrons and positive ions are present along a horizontal channel, representing the laser pre-ionization. The ionization density in this channel is $8 \cdot 10^8\,\text{cm}^{-3}$, and its cross section is $(1\,\text{mm})^2$. Compared to the experiments, the domain is smaller, the laser beam is thinner and the voltage is lower. The reason for using this scaled-down system is that simulations with the experimental conditions are computationally prohibitively expensive. Our comparison between simulations and experiments is therefore qualitative.

Simulation results

In figure 9.4(b), the evolution of a discharge around the 3.5 kV electrode is shown. First, electrons from the pre-ionized channel drift upwards, towards the electrode. During this period the space charge density is too low to have a significant

effect on the electric field amplitude or direction. When the electrons reach the electrode, the discharge starts to grow downward, inside the region where the electrons from the laser beam now are. (Note that the pre-ionization itself is too weak to show up in figure 9.4(b).) When the discharge reaches the boundary of this region, its downward propagation stops, as photoionization has not produced enough free electrons for further growth there. Instead, the discharge now starts to grow horizontally, to both sides of the pre-ionized channel. At the same time it also moves a bit downwards, following the present positions of the electrons from the laser beam. Towards the sides, these electrons have drifted less than in the center. At $t = 13\,\mathrm{ns}$, the streamer is propagating at an angle of more than 60° to the initial background electric field, as indicated in figure 9.5. We can also see that the streamer does not propagate in the direction of the maximum electric field, because it needs the electrons from the laser beam to grow

9.4.3 Comparison with other laser guiding experiments

The phenomenon we have described should not be confused with the laser guiding of (streamer) discharges described by [187, 188] and others, where focused high power lasers create a thin channel with an electron and ion density of 10^{16}–$10^{17}\,\mathrm{cm}^{-3}$ in atmospheric air, while the ionization inside a streamer channel amounts to $10^{15}\,\mathrm{cm}^{-3}$ at most. Such laser channels modify the background electric field at least one order of magnitude more rapidly than a streamer, and effectively act as conductors. In contrast, we have demonstrated that positive streamers can be guided by regions with an electron density of $10^{9}\,\mathrm{cm}^{-3}$ or less in nitrogen/oxygen mixtures at 133 mbar, while a typical streamer would have an ionization density of $10^{12}\,\mathrm{cm}^{-3}$ (when scaling a typical ionization density of $10^{14}\,\mathrm{cm}^{-3}$ at atmospheric pressure to 133 mbar [2]). These channels hardly change the electric field.

Furthermore, in the other studies [188, 189] the laser is usually aligned parallel to the field lines. Here we have shown that very weakly ionized regions that are almost perpendicular to the local field lines, can still guide the streamers. Our numerical simulations generally support the conclusions from the experiments and demonstrate that the laser pre-ionization does not modify the electric field, while it does guide the streamers.

9.5 Conclusions

We have observed that streamers can be guided in a direction nearly perpendicular to the background electric field with a very low level of laser induced pre-ionization ($\lesssim 10^{9}\,\mathrm{cm}^{-3}$). This pre-ionization itself has a negligible effect on the electric field. This occurred in nitrogen/oxygen mixtures at 133 mbar with oxygen concentrations below 0.5 %. At higher oxygen concentrations the streamer locally produces so many free electrons through photoionization that

it is no longer influenced by the laser produced electron trail. Finally, we have presented numerical simulations that showed the same laser guiding effect as in our experiments and confirmed that the space charge effects of the laser trail itself indeed are negligible.

Chapter 10

Afivo: a framework for finite volume simulations on adaptively refined quadtree and octree grids

Afivo is a small framework for doing numerical simulations on adaptively refined quadtrees (2D) and octrees (3D). A geometric multigrid solver suitable for these grids is included. Compared to other simulation frameworks, a 'feature' of Afivo is that it provides less functionality, which can make it easier to adapt. For example, only shared-memory parallelization (OpenMP) is included, so that no code for parallel load balancing or communication is required. The framework is available as free/open source software.

This chapter will be submitted for publication as:
Afivo: a framework for finite volume simulations on adaptively refined quadtree and octree grids, J. Teunissen, U. Ebert

10.1 Introduction

Numerical simulations can often be sped up by having a different mesh in different parts of the domain. There is usually a trade-off: with a more flexible mesh, the cost per computational cell increases. For example, computing a second order approximation of the Laplacian is straightforward on a uniform Cartesian grid. But on an unstructured triangle mesh, such an operation is more complicated: first the neighbors of a cell need to be determined, and then some form of interpolation is required.

Here, we present a framework for simulations on adaptively refined quadtree (2D) and octree (3D) meshes. The main advantage of such meshes is that they provide adaptivity while keeping the grid structure simple. In D dimensions, a quadtree or octree grid consist of boxes that each contain N^D cells. A box with a grid spacing h can be subdivided into 2^D smaller boxes with grid spacing $h/2$. By refining boxes up to different levels, local refinements can be created. In Afivo '2:1 balance' is ensured, which means that between neighboring cells the refinement level differs by at most one. The framework is implemented in Fortran 2003, and is available under the `GNU GPLv3` license.

Below, we first discuss the motivation for developing Afivo, by comparing it to some alternatives. Then the basic data structures and methods are described in sections 10.3 and 10.4. In section 10.5, design choices for parallelization and ghost cells are discussed, after which the multigrid implementation is described in section 10.6.

10.2 Motivation and alternatives

There exist numerous frameworks for doing (parallel) numerical computations. Table 10.1 lists frameworks that operate on structured grids. There are perhaps even more unstructured grid and/or finite element frameworks, but we do not discuss these here.

Two types of structured meshes are commonly used: *block-structured* meshes and *orthtree*[1] meshes. Examples of these meshes are shown in figure 10.1. All the listed frameworks use MPI (which stands for Message Passing Interface) for parallelization. This is a distributed memory technique, so that multiple processors connected by a network can be used in parallel. Some codes also support OpenMP, which is a shared-memory parallelization technique that requires processor cores to have access to the same memory.

Block-structured meshes are more general than orthtree meshes: any orthtree mesh is also a block-structured mesh, whereas the opposite is not true. Some of the advantages and disadvantages of these approaches are:

[1] Note that we here refer to quadtrees and octrees as *orthtrees*, because the general name for quadrants and octants is orthants [190].

Name	Application	Language	Parallel	Mesh
Boxlib [43]	General	C/F90	MPI/OpenMP	Block-str.
Chombo [42]	General	C++/Fortran	MPI	Block-str.
AMRClaw	Flow	F90/Python	MPI/OpenMP	Block-str.
SAMRAI [191]	General	C++	MPI	Block-str.
AMROC	Flow	C++	MPI	Block-str.
Paramesh [44]	General	F90	MPI	Orthtree
Dendro [192]	General	C++	MPI	Orthtree
Peano [193]	General	C++	MPI/OpenMP	Orthtree
Gerris [194]	Flow	C	MPI	Orthtree
Ramses [195]	Self gravitation	F90	MPI	Orthtree

Table 10.1: A list of frameworks for parallel numerical computations on adaptively refined but structured numerical grids. For each framework, the typical application area, programming language, parallelization method and mesh type is listed. This list is largely taken from Donna Calhoun's homepage [196].

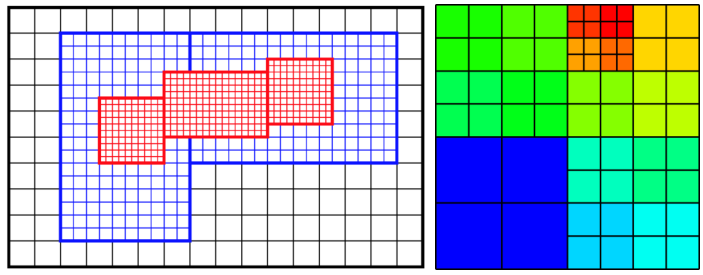

Figure 10.1: Left: example of a block-structured grid, taken from [197]. Right: an quadtree grid consisting of boxes of 2×2 cells.

- In a block-structured mesh, blocks can have a flexible size. Computations on larger blocks are typically more efficient. When ghost cells are required (virtual cells on the boundary of a block), the overhead is smaller when blocks are larger.

- For an orthtree grid, there is a trade-off: larger block sizes allow for more efficient computations, but reduce the adaptivity of the mesh. For a block structured grid, there is a similar trade-off: in principle it can be refined in a more flexible way, but adding many refined blocks increases the overhead.

- The connectivity of the mesh is simpler for an orthtree mesh, because each block has the same number of cells, and blocks are refined in the same way. This also ensures a simple relation between fine and coarse meshes. These properties make operations such as prolongation and restriction easier to implement, especially in parallel.

10.2.1 Motivation: a brief history

Now given the fact that there are already several frameworks available, what was the motivation for developing another one? The main reason was that a *simple* or *basic* framework seemed to be missing – at least to our knowledge. Our motivation came from work on time-dependent simulations of streamer discharges. These discharge have a multiscale nature, and require a fine mesh in the region where they grow. Furthermore, at every time step Poisson's equation has to be solved. A streamer model that uses a uniform Cartesian grid is therefore computationally expensive, especially for 3D simulations.

In [82], Paramesh was used for streamer simulations. The main bottleneck in this implementation was however the Poisson solver. Other streamer models (see e.g., [19, 21, 29]) had the same problem, because the non-local nature of Poisson's equation makes an efficient parallel solution difficult, especially on an adaptively refined grid. An attractive solution method to get around this is geometric multigrid, discussed in section 10.6.

We first considered implementing multigrid in Paramesh [44], which already includes an *alpha* version of a multigrid solver with the following comment [198]:

> This is an ALPHA version of this feature. You should be aware that it may be 'buggy'. Also, construction of multigrid algorithms and AMR is much less straightforward than incorporating AMR into finite-volume hydro codes.

Because Paramesh does not seem to be actively maintained, we decided not to move forward with it after experiencing several problems (creating and visualizing output, performance with a large number of blocks, code organization).

Next, we considered Boxlib [43], an actively maintained framework which is also used in Chombo [42]. Boxlib contains a significant amount of multigrid code, including several examples that demonstrate how a solver can be set up and used. After spending some time getting familiar with the framework, we tried to modify the multigrid solver to our needs. This involves operations like: get the coarse grid values next to refinement boundaries to perform a special type of ghost cell filling (see section 10.6.4). Although such tasks are definitely possible in Boxlib, they are not trivial to implement. The reason is that the framework is quite large and supports many features, uses MPI parallelization and block-structured grids.

In our experience, a large number of scientific simulations fit into the memory of a desktop machine or cluster node, which nowadays typically have 16 or 32 gigabytes of RAM. A practical reason for this is that for larger problems, the visualization of the results becomes quite challenging. For the application we had in mind, efficient large scale parallelism would anyway be hard, due to the non-locality of the Poisson equation. Furthermore, writing parallel code with good scaling takes considerable effort, for which the manpower and resources are often

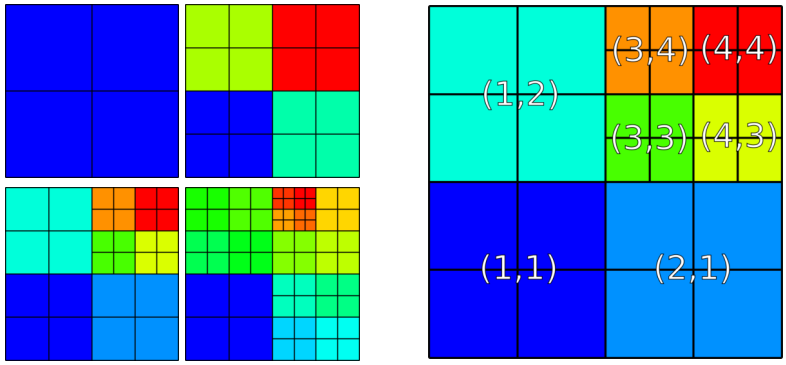

Figure 10.2: Left: Example of a quadtree mesh that gets refined. Here boxes contain 2×2 cells, and different boxes have different colors. Right: the spatial indices of the boxes. When a box with indices (i, j) is refined, its children have indices $(2i - 1, 2j - 1)$ up to $(2i, 2j)$.

lacking. This inspired us to develop a framework that uses shared-memory parallelism, which makes many operations much simpler, because all data can directly be accessed. The goal was to create a relatively simple framework that could easily be modified, to provide an option in between the 'advanced' distributed-memory codes of table 10.1 and simple uniform grid computations.

10.3 Overview of data structures

We now start with the description of Afivo's implementation. First, the properties of *orthtree* meshes are discussed. Three data types are used to store these meshes: boxes, levels and trees. These data types are described below.

10.3.1 Orthtree meshes

In Afivo, quadtree (2D) and octree (3D) meshes are used, which can be described by the following rules:

- The mesh is constructed from boxes that each contain N^D cells, where D is the number of coordinates and N is an even number.
- When a box with a grid spacing h is refined, it is subdivided in 2^D 'children' with grid spacing $h/2$.
- The difference in refinement level for adjacent boxes is at most one. This is called '2:1 balance'.

Figure 10.2 shows an example of a quadtree that gets refined. All the boxes are stored in a single one-dimensional array, so that an integer index can be used to point to a box, see section 10.3.4 below.

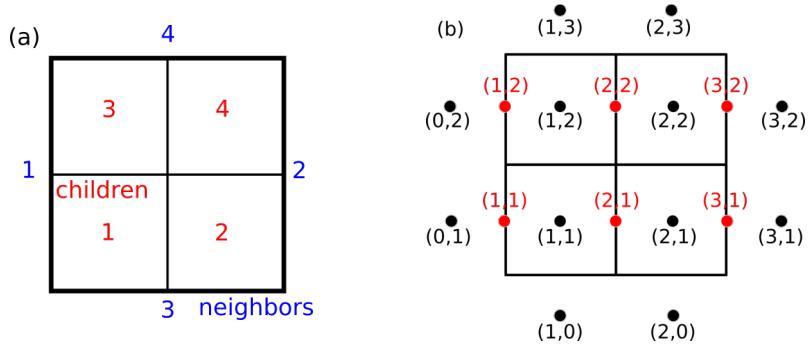

Figure 10.3: a) Each box contains an array of children and neighbors. The ordering of these arrays is shown here for a 2D box. Red: children, blue: neighbors. b) Location and indices of the cell-centered variables (black dots) and the face-centered variables in the x-direction (red dots) for a box of 2×2 cells.

10.3.2 Box data type

A *box* is the basic mesh unit in Afivo. Each box consists of N^D grid cells, where N has to be an even number and D is the spatial dimension. In figure 10.2, there is for example a box with 2×2 cells at coordinate $(1, 1)$. Each box stores its parent, an array of 2^D children and an array of $2D$ neighbors. In figure 10.3a, the indices of the children and the neighbors are shown. A special value `a5_no_box` (which is zero) is used to indicate that a parent, child or neighbor does not exist. In the case of neighbors, boundary conditions are specified by negative numbers.

Two types of cell data are supported by default: cell-centered data and face-centered data, see figure 10.3b. These are stored in $D + 1$-dimensional arrays, so that multiple variables can be stored per location. Furthermore, boxes contain some 'convenience' information, such at their refinement level, minimum coordinate and spatial index.

10.3.3 Level data type

The *level* data type contains three lists:

- A list with all the boxes at refinement level l
- A list with the *parents* (boxes that are refined) at level l
- A list with the *leaves* (boxes that are not refined) at level l

This separation is often convenient, because some algorithms operate only on leaves while others operate on parents or on all boxes. These lists contain the integer indices of the boxes in the tree data structure described below.

10.3.4 Tree data type

The tree data type contains all the data of the mesh. Most importantly, it stores two arrays: one that contains all the boxes and one that contains all the levels[2]. Some other information is also stored: the current maximum refinement level, the number of cells per box-dimension N, the number of face and cell-centered variables and the grid spacing Δx at the coarsest level.

10.4 Methods

In this section we give a brief overview of the most important methods in Afivo. The names of the methods for two-dimensional meshes are used, which have the prefix `a2`. The three-dimensional analogs have, not surprisingly, a prefix `a3`.

10.4.1 Creating the initial mesh

In Afivo the coarsest mesh, which covers the full computational domain, is not supposed to change. To create this mesh there is a routine `a2_set_base`, which takes as input the spatial indices of the coarse boxes and their neighbors. In figure 10.4, a 2D example is shown for creating a single coarse box at index $(1, 1)$. This box is its own neighbor in all four directions, or in other words, there are periodic boundary conditions. Physical (non-periodic) boundaries can be indicated by a negative index for the neighbor. By adjusting the neighbors one can specify different geometries, the possibilities include meshes that contain a hole, or meshes that consist of two isolated parts. The treatment of boundary conditions is discussed in section 10.4.3.

10.4.2 The refinement procedure

Mesh refinement can be performed by calling the `a2_adjust_refinement` routine, which should be passed a user-defined refinement function. This function is then called for each box, and should set the refinement flag of the box to one of three values: refine (add children), derefine (remove this box) or keep refinement. It is also possible to set the refinement flags for other boxes than the current one, which can for example be useful to extend refinements to a neighbor.

In Afivo, a box is either fully refined (with 2^D children) or not refined. Furthermore, 2:1 balance is ensured, so that there is never a jump of more than one refinement level between neighboring boxes. These constraints are automatically handled, so that the user-defined refinement function does not need to impose them.

[2]Since Afivo is implemented in Fortran, these arrays start at index one.

```
! Initialize tree
call a2_init(tree, & ! Tree to initialize
     box_size, &     ! Number of cells per coordinate in a box
     n_var_cell, &   ! Number of face-centered variables
     n_var_face, &   ! Number of cell-centered variables
     dr)             ! Distance between cells on base level

! Set the spatial index and neighbors
ix_list(1:2, 1) = 1  ! One box at (1,1)
nb_list(1:4, 1) = 1  ! Periodic box is its own neighbor

! Create the base mesh
call a2_set_base(tree, ix_list, nb_list)
```

Figure 10.4: Fortran code fragment that shows how a base mesh can be constructed. In this case, there is one box at $(1, 1)$, with periodic boundary conditions.

Each call to `a2_adjust_refinement` changes the mesh by at most one level. To introduce larger changes one should call the routine multiple times. A number of rules is used to make the user-supplied refinement consistent:

- Only leaves can be removed (because the grid changes by at most one level at a time)
- A box flagged for refinement will always be refined, including neighbors that are required for 2:1 balance
- Boxes cannot be removed if that would violate 2:1 balance
- If all the 2^D children of a box are flagged for removal, and the box itself not for refinement, then the children are removed
- Boxes at level one cannot be removed
- Boxes cannot be refined above the maximum allowed refinement level

The `a2_adjust_refinement` routine returns information on the added and removed boxes per level, so that a user can set values on the new boxes or clean up data on the removed ones.

When boxes are added or removed in the refinement procedure, their connectivity is automatically updated. References to a removed box are removed from its parent and neighbors. When a new box is added, its neighbors are found through its parent. Three scenarios can occur: the neighbor can be one of the other children of the parent, the neighbor can be a child from the neighbor of the parent, or the neighbor does not exist. In the latter case, there is a refinement boundary, which is indicated by the special value `a5_no_box`.

10.4.3 Filling of ghost cells

When working with numerical grids that are divided in multiple parts, it is often convenient to use *ghost cells*. The usage of ghost cells has two main advantages: algorithms can operate on the different parts without special care for the boundaries, and algorithms can straightforwardly operate in parallel.

In Afivo each box has one layer of ghost cells for its cell-centered variables, as illustrated in figure 10.3b. The built-in routines only fill the ghost cells on the sides of boxes, not those on the corners. The reasons for implementing ghost cells in this way are discussed in sections 10.5.1 and 10.5.2.

For each side of a box, ghost cells can be filled in three ways

- If there is a neighboring box, then the ghost cells are simply copied from this box
- If there is a physical boundary, then the box is passed to a user-defined routine for boundary conditions
- If there is a refinement boundary, then the box is passed to another user-defined routine

Physical boundaries are indicated by negative values for the neighbor index, and these values are passed on to the user-defined routine. In this way, one can set up different types of boundary conditions.

10.4.4 Interpolation and restriction

Because corner ghost cells are not filled by Afivo, interpolation schemes cannot use diagonal elements. Instead of the standard bilinear and trilinear interpolation schemes, a $2-1-1$ and $1-1-1-1$ scheme is used in 2D and 3D, respectively. These interpolation schemes use information from the closest and second-closest neighbors; the 2D case is illustrated in figure 10.5. Zeroth-order interpolation is also included, in which the coarse values are simply copied without any interpolation. As a restriction method (going from fine to coarse) Afivo just includes averaging, in which the parent gets the average value of its children.

A user can of course implement higher order interpolation and restriction methods, by using information from additional grid locations. It is generally quite complicated to do this consistently near refinement boundaries.

10.4.5 The list of boxes

In Afivo, all the boxes are stored in a single array. New boxes are always added to the end of the array, which means that removed boxes leave a 'hole'. There is a route `a2_tidy_up` to tidy up the array: all the unused boxes are moved to the end of the array, and the boxes that are still in use are sorted by their

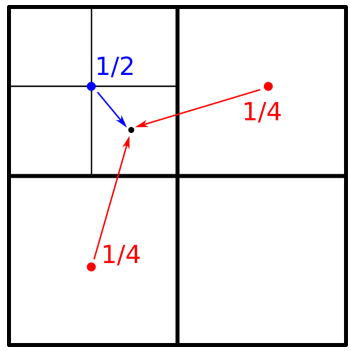

Figure 10.5: Schematic drawing of $2-1-1$ interpolation. The three nearest coarse grid values are used to interpolate to the center of a fine grid cell. Note that the same interpolation scheme can be used for all fine grid cells, because of the symmetry in a cell-centered discretization.

refinement level. Furthermore, for each level, the boxes are ranked according to their Morton index [199].

Sometimes, extra storage is required when `a2_adjust_refinement` has to add new boxes to the mesh. In such a case, the array of boxes is simply resized so that there is enough space.

10.4.6 Producing output

It is important that one is able to quickly and conveniently visualize the results of a simulation. Afivo supports two output formats: VTK unstructured files and Silo files.

For VTK files, Afivo relies on the unstructured format, which support much more general grids than quadtree and octree meshes. This format should probably only be used for grids of moderate sizes (e.g., 10^5 or 10^6 cells), because visualizing larger grids can be computationally expensive. Although there is some support for octrees in VTK, this support does not yet extend to data visualization programs such as Paraview [200] and Visit [135].

Afivo also supports writing Silo files. These files contain a number of Cartesian blocks ('quadmeshes' in Silo's terminology) that can each contain multiple boxes. This is done by starting with a region R that contains a single box. If all the neighbors to the left of R exist, have no children and are not yet included in the output, then these neighbors are added to R. The procedure is repeated in all directions, until R can no longer grow. Then R represents a rectangular collection of boxes which can be added to the output, and the procedure start again from a new box that is not yet included. This merging of boxes is done because writing and reading a large number separate meshes can be quite costly with the Silo library.

10.5 Design discussion

10.5.1 One ghost cell

There are essentially two ways to implement ghost cells in a framework such as Afivo.

1. Ghost cells are not stored for boxes. When a computation has to be performed on a box, there are typically two options: algorithms can be made aware of the mesh structure, or a box can be temporarily copied to an enlarged box on which ghost cells are filled.

2. Each box includes ghost cells, either a fixed number for all variables or a variable-dependent number.

Storing ghost cells can be quite costly. For example, adding two layers of ghost cells to a box of 8^3 cells requires $(12/8)^3 = 3.375$ times as much storage. With one layer, about two times as much storage is required. Not storing ghost cells prevents this extra memory consumption. However, some operations can become more complicated to program, for example when some type of interpolation depends on coarse-grid ghost cells. Furthermore, one has to take care not to unnecessarily recompute ghost values, and parallelization becomes slightly harder.

If ghost cells are stored for each box, then there are still two options: store a fixed number of them for each variable, or let the number of ghost cells vary per variable. In Afivo, we have opted for the simplest approach: there is always one layer of ghost cells for cell-centered variables. For numerical operations that depend on the nearest neighbors, such as computing a second order Laplacian, one ghost cell is enough. When additional ghost cells are required, these can of course still be computed, there is just no default storage for them.

10.5.2 No corner ghost cells

In Afivo, corner ghost cells are not used. The reason for this is that in three dimensions, the situation is quite complicated: there are eight corners, twelve edges and six sides. It is hard to write an elegant routine to fill all these ghost cells, especially because the corners and edges have multiple neighbors. Therefore, only the sides are considered in Afivo. This means that Afivo is not suitable for stencils with diagonal terms.

10.5.3 OpenMP for parallelism

The two conventional methods for parallel computing are OpenMP (shared memory) and MPI (communicating tasks). Afivo was designed for small scale parallelism, for example using at most 16 cores, and therefore only supports OpenMP.

```
do lvl = 1, tree%max_lvl
   !$omp parallel do private(id)
   do i = 1, size(tree%lvls(lvl)%ids)
      id = tree%lvls(lvl)%ids(i)
      call my_method(tree%boxes(id))
   end do
   !$omp end parallel do
end do
```

Figure 10.6: Fortran code fragment that shows how to call `my_method` for all the boxes in a tree, from level 1 to the maximum level. Within each level, the routine is called in parallel using OpenMP.

Compared to an MPI implementation, the main advantage of OpenMP is simplicity: data can always be accessed, sequential (user) code can easily be included, there is no need for load balancing and no communication between processes needs to be set up. For problems that require large scale parallelism, there are already a number of frameworks available, as discussed in section 10.2.

Most operations in Afivo loop over a number of boxes, for example the leaves at a certain refinement level. All such loops have been parallelized by adding OpenMP statements around them, for example as in figure 10.6.

The parallel speedup that one can get depends on the cost of the algorithm that one is using. The communication cost (updating ghost cells) is always about the same, so that an expensive algorithm will show a better speedup. Furthermore, on a shared memory system, it is not unlikely for an algorithm to be memory-bound instead of CPU-bound.

10.6 Multigrid

Multigrid can be seen as a technique to improve the convergence of a relaxation method, by using a hierarchy of grids. Afivo comes with a built-in geometric multigrid solver, to solve problems of the form

$$A(u) = \rho, \tag{10.1}$$

where A is a (nearly) elliptic operator, ρ the right-hand side and u the solution to be computed. In discretized form, we write equation (10.1) as

$$A_h(u_h) = \rho_h, \tag{10.2}$$

where h denotes the mesh spacing at which the equation is discretized.

There already exists numerous sources on the foundations of multigrid, the different cycles and relaxation methods, convergence behaviour and other aspects, see for example [39–41, 201]. Here we will not provide a general introduction to multigrid. Instead we briefly summarize the main ingredients, and focus

on one particular topic: how to implement multigrid on an adaptively refined quadtree or octree mesh. On such a mesh, the solution has to be specified on all levels. Therefore we use FAS multigrid, which stands for Full Approximation Scheme. Below, the implementation of the various multigrid components in Afivo are described.

10.6.1 The V-cycle

Suppose there are levels $l = l_{\min}, l_{\min}+1, \ldots, l_{\max}$, then the FAS V-cycle can be described as

1. For l from $l_{\max}$ to $l_{\min}+1$, perform N_{down} relaxation steps on level l, then update level $l-1$ (see below)
2. Perform N_{base} relaxation steps on level $l_{\min}$, or apply a direct solver
3. For l from $l_{\min}+1$ to $l_{\max}$, perform a correction using the data from level $l-1$
$$u_h \leftarrow u_h + I_H^h(v_H - v'_H), \tag{10.3}$$
then perform N_{up} relaxation steps on level l. (See below for the notation)

The first two steps require some extra explanation. Let us denote the level $l-1$ grid by H and the level l grid by h, and let v denote the current approximation to the solution u. Furthermore, let I_H^h be an interpolation operator to go from coarse to fine and I_h^H a restriction operator to go from fine to coarse. For these operators, the schemes described in section 10.4.4 are used.
In the first step, the coarse grid is updated in the following way

1. Set $v_H \leftarrow I_h^H v_h$, and store a copy v'_H of v_H
2. Compute the the fine grid residual $r_h = \rho_h - A_h(v_h)$
3. Update the coarse grid right-hand side
$$\rho_H \leftarrow I_h^H r_h + A_H(v_H) \tag{10.4}$$

This last equation can also be written as

$$\rho_H \leftarrow I_h^H \rho_h + \tau_h^H,$$

where τ_h^H is given by [39–41, 201, 202]

$$\tau_h^H = A_H(I_h^H v_h) - I_h^H A_h(v_h). \tag{10.5}$$

This term can be seen as a correction to ρ on the coarse grid. When a solution u_h is found such that $A_h(u_h) = \rho_h$, then $u_H = I_h^H u^h$ will be a solution to $A_H(u_H) = \rho_H$.

In the second step, relaxation takes place on the coarsest grid. In order to quickly converge to the solution with a relaxation method, this grid should contain very few points (e.g., 2×2 or 4×4 in 2D). Alternatively, a direct solver can be used on the coarsest grid, in which case it can be larger. Such a direct method has not yet been built into Afivo, although this is planned for the future. As a temporary solution, additional coarse grids can be constructed below the coarsest quadtree/octree level. For example, if a quadtree has boxes of 16×16 cells, then three levels can be added below it (8×8, 4×4 and 2×2), which can then be used in the multigrid routines.

10.6.2 The FMG-cycle

The full multigrid (FMG) cycle that is implemented in Afivo works in the following way.

1. If there is no approximation of the solution yet, then set the initial guess to zero on all levels, and restrict ρ down to the coarsest grid using I_h^H. If there is already an approximation v to the solution, then restrict v down to the coarsest level. Use equation (10.4) to set ρ on coarse grids.

2. For $l = l_{\min}, l_{\min} + 1, \ldots, l_{\max}$

   - Store the current approximation v_h as v_h'
   - If $l > l_{\min}$, perform a coarse grid correction using equation (10.3)
   - Perform a V-cycle starting at level l, as described in the previous section

10.6.3 Gauss Seidel red-black

In Afivo, we have implemented Gauss Seidel red-black or GS-RB as a relaxation method. This method is probably described in almost all textbooks on multigrid, such as [39–41, 201], so we just give a very brief description.

The *red-black* refers to the fact that points are relaxed in an alternating manner, using a checkerboard-like pattern. For example, in two dimensions with indices (i, j) points can be labeled *red* when $i + j$ is even and *black* when $i + j$ is odd. Now consider equation (10.2), which typically relates a value $u_h^{(i,j)}$ to neighboring values and the source term ρ. If we keep the values of the neighbors fixed, then we can determine the value $u_h^{(i,j)}$ that locally solves the linear equation. This is precisely what is done in GS-RB: the linear equations are solved for all the red points while keeping the old black values, and then vice-versa.

10.6.4 Conservative filling of ghost cells

The finer levels will typically not cover the complete grid in Afivo, so that ghost cells have to be used near refinement boundaries. These ghost cells can be

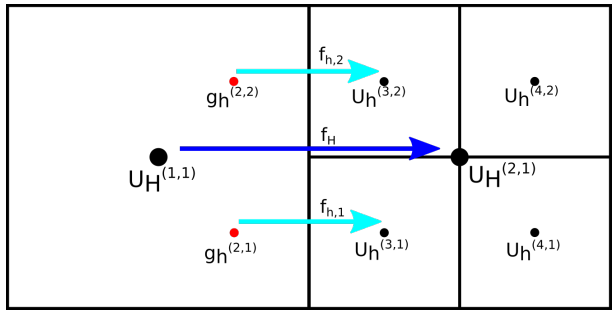

Figure 10.7: Two coarse cells, of which the right one is refined. The cell centers are indicated by dots. There are two ghost values (red dots) on the left of the refinement boundary. Fluxes across the refinement boundary are indicated by arrows.

filled in multiple ways, which will affect the multigrid solution and convergence behavior. Here we consider *conservative* schemes for filling ghost cells [40, 202]. A conservative scheme ensures that the coarse flux across a refinement boundary equals the average of the fine fluxes, see figure 10.7.

Ensuring consistent fluxes near refinement boundaries helps in obtaining a *consistent* solution. For example, if we consider a general equation of the form $\nabla \cdot \boldsymbol{F} = \rho$, then the divergence theorem gives

$$\int_V \rho \, dV = \int_V \nabla \cdot \boldsymbol{F} \, dV = \int \boldsymbol{F} \cdot \boldsymbol{n} \, dS, \tag{10.6}$$

where the last integral runs over the surface of the volume V, and $\boldsymbol{n}$ is the normal vector to this surface. This means that when fine and coarse fluxes are consistent, the integral over ρ will be same on the fine and the coarse grid.

The construction of a conservative scheme for filling ghost cells is perhaps best explained with an example. Consider a 2D Poisson problem

$$\nabla^2 u = \nabla \cdot (\nabla u) = \rho, \tag{10.7}$$

with a standard 5-point stencil for the Laplace operator

$$L_h = h^{-2} \begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix}. \tag{10.8}$$

With this stencil, the coarse flux f_H across the refinement boundary in figure 10.7 is given by

$$f_H = [u_H^{(2,1)} - u_H^{(1,1)}]/H, \tag{10.9}$$

and on the fine grid, the two fluxes are given by

$$f_{h,1} = [u_h^{(3,1)} - g_h^{(2,1)}]/h, \tag{10.10}$$

$$f_{h,2} = [u_h^{(3,2)} - g_h^{(2,2)}]/h. \tag{10.11}$$

The task is now to fill the ghost cells $g_h^{(2,1)}$ and $g_h^{(2,2)}$ in such a way that the coarse flux equals the average of the fine fluxes, i.e., such that

$$f_H = (f_{h,1} + f_{h,2})/2 \tag{10.12}$$

To relate $u_H^{(2,1)}$ to the refined values u_h, the restriction operator I_h^H needs to be specified. In our implementation, this operator does averaging over the children, which can be represented as

$$I_h^H = \frac{1}{4}\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}. \tag{10.13}$$

The constraint from equation (10.12) can then be written as

$$g_h^{(2,1)} + g_h^{(2,2)} = u_H^{(1,1)} + \frac{3}{4}\left(u_h^{(3,1)} + u_h^{(3,2)}\right) - \frac{1}{4}\left(u_h^{(4,1)} + u_h^{(4,2)}\right). \tag{10.14}$$

Any scheme for the ghost cells that satisfies this constraint will be a conservative discretization.

Bilinear *extrapolation* (similar to standard bilinear interpolation) gives the following scheme for $g_h^{(2,1)}$

$$g_h^{(2,1)} = \frac{1}{2}u_H^{(1,1)} + \frac{9}{8}u_h^{(3,1)} - \frac{3}{8}\left(u_h^{(3,2)} + u_h^{(4,1)}\right) + \frac{1}{8}u_h^{(4,2)}. \tag{10.15}$$

(The scheme for $g_h^{(2,2)}$ should then be obvious.) Another option is to use only the closest two neighbors for the extrapolation, which gives the following expression for $g_h^{(2,1)}$

$$g_h^{(2,1)} = \frac{1}{2}u_H^{(1,1)} + u_h^{(3,1)} - \frac{1}{4}\left(u_h^{(3,2)} + u_h^{(4,1)}\right). \tag{10.16}$$

This last scheme is how refinement-boundary ghost cells are filled by default in Afivo.

Three-dimensional case

In three spatial dimensions, the 5-point stencil of equation (10.8) becomes a 7-point stencil with -6 at the center, and the restriction operator has eight entries of $1/8$. The analog of equation (10.16) then becomes

$$g_h^{(2,1,1)} = \frac{1}{2}u_H^{(1,1,1)} + \frac{5}{4}u_h^{(3,1,1)} - \frac{1}{4}\left(u_h^{(4,1,1)} + u_h^{(3,2,1)} + u_h^{(3,1,2)}\right). \tag{10.17}$$

Change in ε at cell face

For the more general equation with a coefficient ε

$$\nabla \cdot (\varepsilon \nabla u) = \rho, \tag{10.18}$$

we consider a special case: ε jumps from ε_1 to ε_2 at a cell face. Local reconstruction of the solution shows that a gradient $(\phi_{i+1} - \phi_i)/h$ has to be replaced by

$$\frac{2\,\varepsilon_1 \varepsilon_2}{\varepsilon_1 \varepsilon_2} \frac{\phi_{i+1} - \phi_i}{h}, \tag{10.19}$$

or in other words, the gradient is multiplied by the harmonic mean of the ε's (see for example chapter 7.7 of [40]). The 5-point stencil for the Laplacian can be modified accordingly.

When a jump in ε occurs on a coarse cell face, it will also be located on a fine cell face, see figure 10.7. In this case, the ghost cell schemes described above for constant ε still ensure flux conservation. The reason is that the coarse and fine flux are both weighted by a factor $2\,\varepsilon_1\varepsilon_2/(\varepsilon_1\varepsilon_2)$.

Cylindrical case

In cylindrical coordinates, the Laplace operator can be written as

$$\nabla^2 u = \frac{1}{r}\partial_r(r\partial_r u) + \partial_z^2 u = \partial_r^2 u + \frac{1}{r}\partial_r u + \partial_z^2 u, \tag{10.20}$$

where we have assumed cylindrical symmetry (no ϕ dependence). At a radius $r \neq 0$, the 5-point stencil is

$$L_h = h^{-2} \begin{bmatrix} & 1 & \\ 1 - \frac{h}{2r} & -4 & 1 + \frac{h}{2r} \\ & 1 & \end{bmatrix}. \tag{10.21}$$

With the cell-centered grids in Afivo, radial grid points are located at $(i - \frac{1}{2})h$ for $i = 1, 2, 3, \ldots$, which means we do not have to consider the special case $r = 0$. For this type of grid indexing, the 5-point stencil can also be written as

$$L_h = h^{-2} \begin{bmatrix} & 1 & \\ \frac{2i-2}{2i-1} & -4 & \frac{2i}{2i-1} \\ & 1 & \end{bmatrix}. \tag{10.22}$$

If we do not modify the restriction operator, then the ghost cells can still be filled with the schemes from equations (10.16) and (10.17). One way to interpret this is that fluxes are computed in the same way in cylindrical coordinates, although their divergence is weighted by the radius:

$$\nabla \cdot \boldsymbol{F} = \frac{1}{r}\partial_r(rF_r) + \ldots \tag{10.23}$$

From figure 10.7, we can see that for refinement in the r-direction, the coarse and fine flux are 'weighted' by the same radius. For the fluxes in the z-direction, the computations are the same as for the Cartesian case. Note that when the restriction operator is changed to include radial weighting, these arguments are no longer valid.

10.6.5 Multigrid test problems

In this section we present two test problems to demonstrate the multigrid behavior on a partially refined mesh. We use the 'method of manufactured solutions': from an analytic solution the right-hand side and boundary conditions are computed. Two test problems are considered, a constant-coefficient Poisson equation

$$\nabla^2 u = \nabla \cdot (\nabla u) = \rho \tag{10.24}$$

and a cylindrical version with a coefficient ε

$$\frac{1}{r}\partial_r(r\varepsilon\partial_r u) + \partial_z(\varepsilon\partial_z u) = \rho, \tag{10.25}$$

both on a two-dimensional rectangular domain $[0,1]\times[0,1]$. For the second case, ε has a value of 100 in the lower left quadrant $[0,0.25]\times[0,0.25]$, and a value 1 in the rest of the domain. In both cases, we pick the following solution for u

$$u(r) = \exp(|\boldsymbol{r} - \boldsymbol{r}_1|/\sigma) + \exp(|\boldsymbol{r} - \boldsymbol{r}_2|/\sigma), \tag{10.26}$$

where $\boldsymbol{r_1} = (0.25, 0.25)$, $\boldsymbol{r_2} = (0.75, 0.75)$ and $\sigma = 0.04$. An analytic expression for the right-hand side ρ is obtained by plugging the solution in equations (10.24) and (10.25)[3], and the solution itself is used to set boundary conditions.

The two different problems can now be solved numerically. For the cylindrical case with the varying ε, a modified Laplacian operator is used, as described in section 10.6.4. The Gauss Seidel red-black relaxation methods are also modified, because they depend on the applied operator, see section 10.6.3. For these examples, we have used $N_{\text{down}} = N_{\text{up}} = N_{\text{base}} = 2$ (number of down/up/base smoothing steps), and a coarsest grid of 2×2 cells.

It is possible to do adaptive mesh refinement in multigrid, for example by using an estimate of the local truncation error based on equation (10.5) (see also chapter 9 of [40]). Such a technique is not used here, instead the refinement criterion is based on the right-hand side: refine if $\Delta x^2|\rho| > 5.0\times 10^{-4}$. The resulting mesh spacing is shown in figure 10.8a.

In both cases, one FMG (full multigrid) cycle is enough to achieve convergence up to the discretization error, which was approximately 10^{-4} for the mesh of figure 10.8a. Consecutive FMG cycles have a negligible effect on the absolute error, although they do reduce the residual $r = \rho - \nabla^2 u$. The maximum value of

[3]Note that jumps in ε also contribute to the source term ρ.

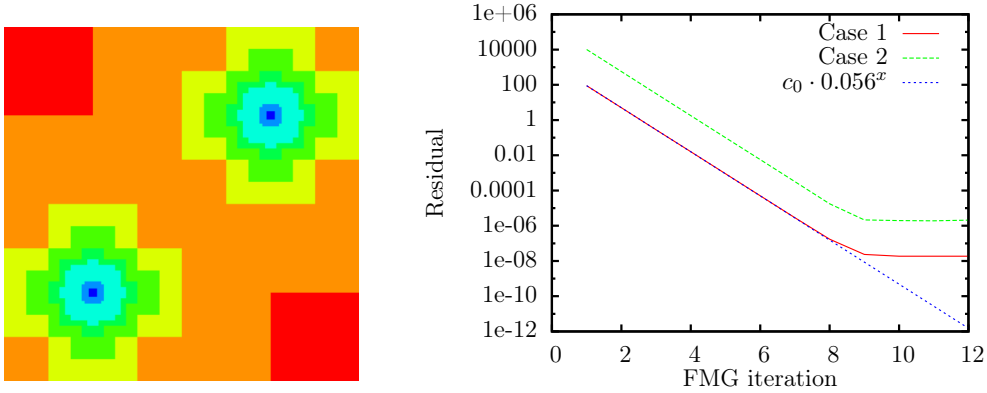

Figure 10.8: Left: mesh spacing used for the multigrid examples, in a $[0,1]\times[0,1]$ domain. Each step in color is a factor two in refinement, with red indicating $\Delta x = 2^{-5}$ and the darkest blue indicating $\Delta x = 2^{-12}$. Right: the maximum residual versus FMG iteration, case 1 corresponds to equation (10.24) and case 2 to equation (10.25).

$|r|$ is shown versus iteration number in figure 10.8b. The convergence behaviour is similar for both cases, with each iteration reducing the residual by a factor of about 0.056. The offset between the lines is caused by the $\varepsilon = 100$ region, which locally amplifies the source term by a factor of 100.

10.7 Implementing a plasma fluid model

To illustrate how Afivo can be used, we describe the implementation of a simple 2D/3D *plasma fluid model* for streamer discharges below. For simplicity, photoionization (see chapter 11) is not included in this example. A review of fluid models for streamer discharges can be found in [19].

10.7.1 Model formulation

We use the so-called drift-diffusion-reaction approximation:

$$\partial_t n_e = -\nabla \cdot \boldsymbol{j}_e + \bar{\alpha}\,|\boldsymbol{j}_e|\,, \tag{10.27}$$

$$\partial_t n_i = \bar{\alpha}\,|\boldsymbol{j}_e|\,, \tag{10.28}$$

$$\boldsymbol{j}_e = -\mu_e n_e \boldsymbol{E} - D_e \nabla n_e, \tag{10.29}$$

where n_e is the electron density, n_i the positive ion density, $\boldsymbol{j}_e$ the electron flux, $\bar{\alpha}$ the effective ionization coefficient, μ_e the electron mobility, D_e the electron diffusion coefficient and $\boldsymbol{E}$ the electric field. The above equations are coupled to the electric field, which we compute in the electrostatic approximation, see

chapter 2.2.5:

$$\boldsymbol{E} = -\nabla\phi, \tag{10.30}$$

$$\nabla^2\phi = -\rho/\varepsilon_0 \tag{10.31}$$

$$\rho = e(n_i - n_e), \tag{10.32}$$

where ϕ is the electric potential, ε_0 the permittivity of vacuum and e the elementary charge. The electric potential is computed with the multigrid routines described in section 10.6.

We make use of the *local field approximation* [76], so that μ_e, D_e and $\bar{\alpha}$ are all functions of the local electric field strength $E = |\boldsymbol{E}|$. These coefficients can be obtained experimentally, or they can be computed with a Boltzmann solver [203, 204] or particle swarms [20].

10.7.2 Flux calculation and time stepping

The electron flux is computed as in [29]. For the diffusive part, we use central differences. The advective part is computed using the Koren limiter [49], whose implementation is discussed in appendix B. The Koren limiter was not designed to include refinement boundaries, and we use linear interpolation to obtain fine-grid ghost values. These ghost cells lie inside a coarse-grid neighbor cell, and we limit them to twice the coarse values to preserve positivity. (We would like to improve this in the future.)

Time stepping is also performed as in [29], using the explicit trapezoidal rule, also known as the modified Euler's method. The time step is determined by a CFL condition for the electron flux and the dielectric relaxation time, as in [29].

10.7.3 Refinement criterion

Our refinement criterion contains two components: a *curvature monitor* c_ϕ for the electric potential and a monitor $\bar{\alpha}\Delta x$ which gives information on how well the ionization length $(1/\bar{\alpha})$ is resolved. For both, we use the maximum value found in a box in order to decide whether to (de)refine it.

Since $\nabla^2\phi = -\rho/\varepsilon_0$, the curvature monitor can be computed as $c_\phi = \Delta x^2|\rho|/\varepsilon_0$. The quantity $\bar{\alpha}\Delta x$ is computed by locating the highest electric field in the box, and looking up the corresponding value of $\bar{\alpha}$. The combined refinement criterion is then as follows, where later rules can override earlier ones:

- If $\bar{\alpha}\Delta x < 0.1$ and $\Delta x < 25\,\mu$m, derefine.
- If $t < 2.5$ ns, ensure that there is enough refinement around the initial seed to resolve it.
- If $\bar{\alpha}\Delta x > 1.0$ and $c_\phi > 0.1$ Volt, refine.

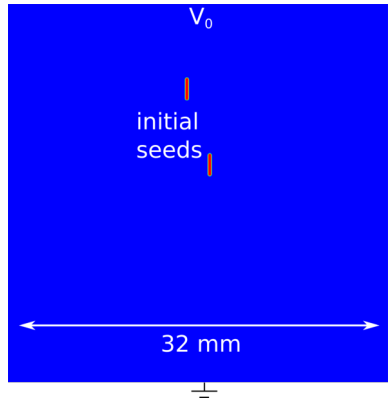

Figure 10.9: Cross section through the center of the three-dimensional simulation domain. The ionized seeds with a density of $10^{20}\,\mathrm{m}^{-3}$ electrons and ions are indicated in red. There is a background density of $5\times10^{15}\,\mathrm{m}^{-3}$ electrons and ions, and the background electric field points down with a magnitude $E_0 = 2.5\,\mathrm{MV/m}$.

10.7.4 Simulation conditions and results

A cross section through the computational domain of $(32\,\mathrm{mm})^3$ is shown in figure 10.9. The background field points down, with a magnitude $E_0 = 2.5\,\mathrm{MV/m}$, which is about $5/6^{\mathrm{th}}$ of the critical field – see chapter 8 for a comparison of discharges above and below the critical field. The background field is applied by grounding the bottom boundary of the domain, and applying a voltage at the top. At the other sides of the domain we use Neumann boundary conditions for the potential. We use transport coefficients (e.g., $\bar{\alpha}$, μ_e) for atmospheric air, but for simplicity photoionization (see chapter 11) has not been included. Instead a background density of $5 \times 10^{15}\,\mathrm{m}^{-3}$ electrons and positive ions is present.

Two seeds of electrons and ions locally enhance the background electric field, see figure 10.9. These seeds have a density of $10^{20}\,\mathrm{m}^{-3}$, a width of about 0.3 mm and a length of 1.6 mm. The electrons from these seeds will drift upwards, enhancing the field at the bottom of the seed where a positive streamer can form.

In figure 10.10, the time evolution of the electron density is shown, and in figure 10.11 the electric field is shown. Two positive streamers grow downwards from the ionized seeds. The upper one is attracted to the negatively charged end of the lower one, and connects to it at around 9.5 ns. The three-dimensional simulation took about 3.5 hours on a 16-core machine, and eventually used about 1.3×10^7 grid cells.

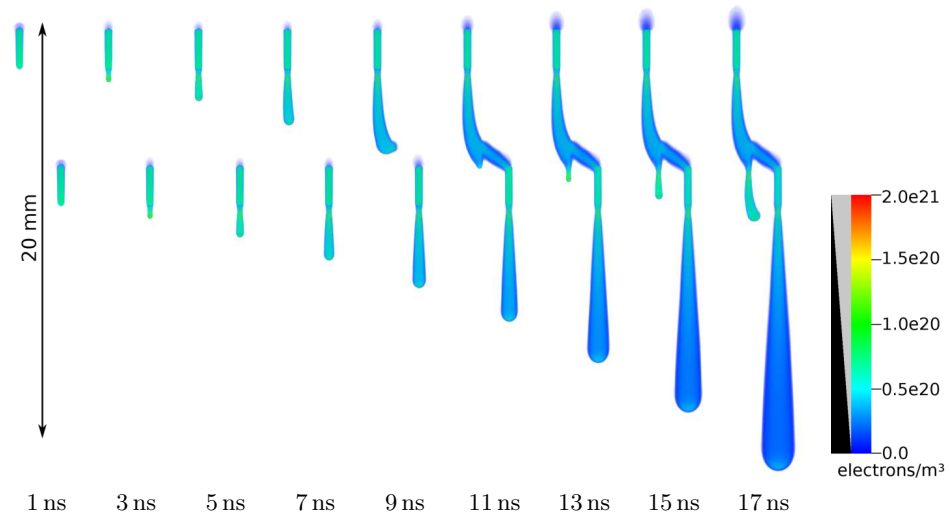

Figure 10.10: A three-dimensional simulation showing two positive streamers propagating downwards. The upper one connects to the back of the lower one. The electron density is shown using volume rendering, for which the opacity is indicated in the legend; low densities are transparent.

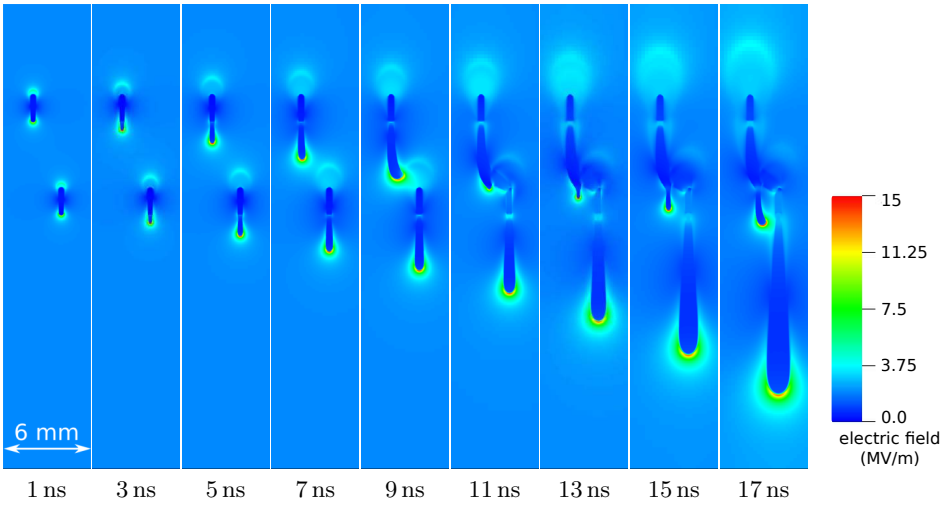

Figure 10.11: Cross section through the three-dimensional domain showing the time evolution of the electric field. The full height of the domain is shown (32 mm), but only 6 mm of the width.

Chapter 11

A Monte Carlo approach for photoionization in discharge simulations

We present a Monte Carlo approach for photoionization in plasma fluid simulations of gas discharges. A number of discrete photons is used to approximate the photoionization profile. Two strategies for photon absorption are considered: the photons can be absorbed on a single uniform grid, or on a grid with multiple levels, with the level chosen for each photon. The method has several advantages. It can be used in 2D and 3D, for a general photon absorption function. The interaction with objects and surfaces can be included relatively easily, and one can trade accuracy for performance by reducing the number of discrete photons. We present numerical examples in a cylindrical geometry.

This chapter will be extended for publication as:
A Monte Carlo approach for photoionization in discharge simulations,
J. Teunissen, U. Ebert

11.1 Introduction

Photoionization can play an important role in electrical discharges in air and other gases. Several numerical methods for including photoionization in discharge models have been described in the literature [11, 17, 47, 48]. Here, we present a new Monte Carlo method that is aimed at plasma fluid simulations, especially those on adaptively refined grids.

If we consider models for streamer discharges, then two typical challenges are solving Poisson's equation to obtain the electrostatic potential, and computing the photoionization profile. Both challenges involve a *non-local* process or interaction – at least when we assume that the speed of light is effectively infinite. However, the required accuracy for these two problems is quite different: discharges are much less sensitive to the amount of photoionization around them than to the electric field [88]. The method presented here was thus designed to be fast and flexible, but not necessarily highly accurate.

The physics of photoionization have recently been reviewed [45] for air, O_2, N_2 and CO_2. In air, photoionization can take place when an excited nitrogen molecule emits a UV photon in the 98 to 102.5 nm range, which is enough energy to ionize an oxygen molecule:

$$N_2^* \rightarrow N_2 + \gamma, \tag{11.1}$$

$$O_2 + \gamma \rightarrow O_2^+ + e. \tag{11.2}$$

How far the photon γ will travel depends on the gas mixture and density. According to the Zheleznyak model [46], the mean absorption distance is about 0.45 mm for air at 1 bar and room temperature, see the next section.

To compute the photoionization profile generated by a distribution of photon sources, we can use the following integral

$$S_{\text{ph}}(\boldsymbol{r}) = \int \frac{I(\boldsymbol{r'}) f(|\boldsymbol{r} - \boldsymbol{r'}|)}{4\pi |\boldsymbol{r} - \boldsymbol{r'}|^2} d^3 r', \tag{11.3}$$

where $S_{\text{ph}}(\boldsymbol{r})$ is the produced photoionization, $I(\boldsymbol{r'})$ the photon source term and the denominator is the standard geometric factor for isotropically distributed photons. The absorption function $f(r)$ gives the probability density of absorption at a distance r, so that

$$\int_0^\infty f(r) dr = 1. \tag{11.4}$$

The factor $f(|\boldsymbol{r} - \boldsymbol{r'}|)$ in the integrand complicates the numerical solution of equation (11.3). In this paper, a method is presented that approximates this integral using Monte Carlo techniques.

The content of the paper is as follows: in section 11.2, past models for photoionization in discharge simulations are briefly reviewed. In section 11.3, the Monte Carlo procedure for approximating the integral is described, and in section 11.4 numerical examples are presented.

11.2 Past work

When discussing photoionization in discharge models, there are two aspects to consider: the physical model for photoionization and the numerical implementation. The physical processes that generate photoionization in O_2, air, N_2 and CO_2 have recently been reviewed in [45].

Below, we will briefly discuss the numerical methods that have thus far been used to implement photoionization in discharge simulations. The authors all used the Zheleznyak model [46] for photoionization in air, in which the absorption function is given by

$$f(r) = \frac{\exp(-\chi_{\min} p_{O_2} r) - \exp(-\chi_{\max} p_{O_2} r)}{r \ln(\chi_{\max}/\chi_{\min})}, \tag{11.5}$$

where $\chi_{\max} \approx 1.5 \times 10^2/(\text{mm bar})$, $\chi_{\min} \approx 2.6/(\text{mm bar})$ and p_{O_2} is the partial pressure of oxygen. Equation (11.5) is based on the assumption that:

- Photon wavelengths λ are uniformly distributed between $\lambda_0 = 98\,\text{nm}$ and $\lambda_1 = 102.5\,\text{nm}$.
- At a wavelength λ, the absorption coefficient is given by $\kappa_u = \chi_{\min}^u \chi_{\max}^{1-u} p_{O_2}$, where $u = (\lambda - \lambda_0)/(\lambda_1 - \lambda_0)$.

Note that u lies between zero (at λ_0) and one (at λ_1). The absorption function for photons at a given wavelength / value of u is

$$f_u(r) = \kappa_u \exp(-\kappa_u r). \tag{11.6}$$

To compute the average absorption function, we integrate over the range $u = 0$ to $u = 1$ (i.e., from λ_0 to λ_1), which gives equation (11.5):

$$f(r) = \int_0^1 f_u(r) du = \frac{\exp(-\chi_{\min} p_{O_2} r) - \exp(-\chi_{\max} p_{O_2} r)}{r \ln(\chi_{\max}/\chi_{\min})}. \tag{11.7}$$

The mean absorption distance can be computed as

$$\bar{r} = \int_0^\infty r f(r) dr \approx 0.093\,\text{mm bar}/p_{O_2}. \tag{11.8}$$

In the Zheleznyak model, the production of ionizing photons is proportional to the electron impact ionization term, with a coupling constant η that depends on the gas and electric field. The model was derived for a steady-state discharge, and does not include the decay time of emitting states. If this decay time is known, it can of course be included when computing the source term $I(\boldsymbol{r}')$ of equation (11.3).

11.2.1 Helmholtz approximation

As stated above, the absorption function in the integrand of equation (11.3) complicates its numerical solution. Luque et al. [47] therefore approximated equation (11.5) by the following expansion

$$\tilde{f}(r) = r \sum_{j=1}^{N} A_j \exp(-\lambda_j r). \tag{11.9}$$

The authors used $N = 2$, and the A_j and λ_j were fitted to get the best agreement with equation (11.5). Equation (11.3) can then be transformed into a set of Helmholtz equations, which can be solved with fast elliptic solvers.

Note that equations (11.5) and (11.9) differ by a factor of r^2. (Compared with equation (11.6), the difference is only a factor r however.) An expansion with few terms can therefore only be accurate in a limited range. Furthermore $\tilde{f}(0) = 0$, whereas the original absorption function has a maximum. As discussed in [47], this difference is not as bad as it seems. Close to the discharge, impact ionization dominates, so that only the non-local photoionization ($r > 0$) is important.

Bourdon et al. [17] used three terms for equation (11.9), and investigated the importance of the boundary conditions for the Helmholtz equations. They also presented results for a 'three group Eddington (SP_3) model', which was introduced in an earlier paper [48].

11.2.2 Photoionization for PIC codes

Chanrion and Neubert [11] proposed a method suitable for PIC-MCC codes (particle-in-cell, Monte Carlo collision). In such a simulation, the particles each have a weight w, which determines how many physical electrons they represent. For each electron impact ionization, there is a probability η that w discrete photons are produced. The photons get a random isotropic direction, and their absorption length is sampled from the distribution underlying equation (11.5). At the locations of absorption, electron-ion pairs are created.

This method can be seen as a Monte Carlo approximation to equation (11.3), and the method presented here shares some of its features.

11.3 Description of the method

In our Monte Carlo method for photoionization, we assume that photon scattering can be neglected, and that the direction of ionizing photons is isotropic. The method can be divided in three parts:

1. Determine the coordinates at which photons are produced, and store these coordinates in a list L_{src}.

2. Determine the coordinates at which photons are absorbed, and store these coordinates in a list L_{dst}.

3. Compute the resulting photoionization profile on a mesh.

The implementation of these steps is described below.

11.3.1 The source of ionizing photons

In a plasma fluid simulation, the computational domain is divided into cells. We assume that for each cell the production of ionizing photons I is known within a given time step Δt. In our Monte Carlo method, this information is converted to a list L_{src} of approximately N discrete photons in the following way.

1. Determine the total photon production I_Σ within the time step Δt, by summing over all the cells.

2. For each cell $n_\gamma = NI/I_\Sigma$ photons should be produced. To convert n_γ to an integer, first draw a uniform random number $U(0,1)$. If $n_\gamma - \lfloor n_\gamma \rfloor > U$ round up, else round down.

3. For each produced photon, add the coordinate of the cell center to the list L_{src} of photons.

Some additional remarks: In principle the number of photons N can be chosen adaptively, for example to create discrete 'super-photons' with a given weight. It is also possible to assign different weights to different photons by modifying the above procedure, see section 11.3.4.

Instead of rounding n_γ to an integer as described above, it can be more realistic to sample from the Poisson distribution with parameter n_γ. For $n_\gamma \ll 1$ the result would be almost identical, but for larger values there are differences. However, most of the random fluctuations in our method are due to the stochastic photon direction and absorption, as discussed in the next section. Therefore our rounding with uniform random numbers should be fine for most applications.

When grid cells are large compared to typical photoionization length scales, one could determine a 'subgrid' production location in the cell, instead of using the cell center. However, this scenario should generally be avoided, because the resulting photoionization profile would itself be insufficiently resolved.

11.3.2 Absorption of ionizing photons

Now that we know where photons are produced, we need to determine where they are absorbed. This is done in the following way. First, we determine the absorption distance r for a photon. Given an absorption function $f(r)$, the cumulative absorption function $F(r) = \int_0^r f(r')dr'$ can be computed, either

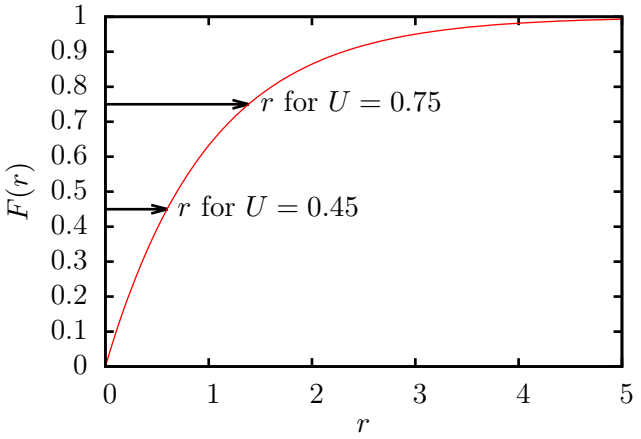

Figure 11.1: Illustration of inversion sampling. If $F(r)$ is a cumulative distribution function, then one can obtain samples r from that distribution by solving $F(r) = U$, where U is a uniform random number between zero and one. In our Monte Carlo procedure, a linear *lookup table* is constructed of r versus $F(r)$, to speed up the sampling.

numerically or analytically. Then a so-called *lookup table* is constructed with columns $F(r)$ and r. Now we can do *inversion sampling*: given a random number $U(0, 1)$, the corresponding distance r is obtained by linear interpolation from the table. The procedure is illustrated in figure 11.1. (In the special case where the inverse of $F(r)$ is known, one could directly compute $r = F^{-1}(U)$, but this will often be slower than using a lookup table.)

Then a random orientation $\hat{\boldsymbol{r}}$ for the photon is determined, using a procedure for picking a point on a sphere proposed by Marsaglia [205]:

1. Get two random numbers $U_1(-1, 1)$ and $U_2(-1, 1)$. If $U_1^2 + U_2^2 \leq 1$ accept them, otherwise draw new numbers.

2. Let $a = U_1^2 + U_2^2$ and $b = 2\sqrt{1 - a}$. The isotropic random direction is $\hat{\boldsymbol{r}} = (bU_1,\, bU_2,\, 1 - 2a)$ in Cartesian coordinates.

In the special case of a 2D Cartesian (x, y) coordinate system, we should not pick points on a sphere but points on a circle. Then the second step is replaced by

$$\hat{\boldsymbol{r}} = \left(\frac{U_1^2 - U_2^2}{U_1^2 + U_2^2},\, \frac{2U_1U_2}{U_1^2 + U_2^2} \right).$$

Given the direction and the distance, the location of absorption $\boldsymbol{r} = r\hat{\boldsymbol{r}}$ is known, which is added to the list L_{dst}.

Sometimes, the typical absorption length of photons is much larger than the domain size. Since most photons will not be absorbed within the domain, the Monte Carlo approximation becomes less accurate. This can be resolved by

changing the lookup table. Suppose that all photons with absorption distances $r > r_{\max}$ can be ignored, then one can construct a lookup table up to $r_{\max}$ and scale the corresponding $F(r)$ values to the range $(0, 1)$. Each photon that is produced now gets a smaller weight $F(r_{\max})$.

11.3.3 Computing the photoionization profile

At this stage, a list L_{dst} with the absorption locations of the photons has been constructed. These coordinates now need to be mapped to a mesh to get the photoionization profile. We consider two options: the photons can be absorbed on a single mesh with a constant grid spacing, or they can be absorbed at different levels in a refined mesh.

In both cases we use *nearest grid point* (NGP) to map the photons to densities, which means that photons are absorbed within a single cell. With linear (and higher order) interpolation the resulting density profiles are smoother, but it becomes harder to handle refinement boundaries.

Absorption using a constant grid spacing

If the photons are absorbed on a mesh with constant grid spacing Δx, two length scales have to be considered. First, Δx should be some fraction of the typical absorption distance of photons, to resolve the absorption profile. Using the lookup table for absorption distances (see section 11.3.2) one can for example use $\Delta x \approx F^{-1}(0.25)$, so that about 25% of photons is absorbed within a distance Δx. Second, Δx should be smaller than some typical length scale l_{src} for the photon source term. The reason for this is the geometric falloff of the absorption density, which goes like r^{-2} for a point source.

The above requirements on Δx ensure that its value is *small enough*, so that we get an accurate solution close to the photon sources. However, a small grid spacing will lead to significant noise in regions farther away, where few photons are absorbed. Fluctuations have a typical size of $\sqrt{N}$, where N is the number of photons absorbed in some region. We can increase Δx, so that some accuracy is lost near the sources, but a smoother profile is obtained farther away from them.

Absorption using a photon-dependent grid spacing

With a constant grid spacing we cannot get a good approximation of the absorption profile in all regions. Near the source we need a small grid spacing, but farther away we would like to use a coarser grid, to reduce fluctuations. This could be achieved by creating a mesh that is suitably refined, but ideally we would like to use the same mesh as is already used for the plasma fluid model. Assuming that this is a sufficiently refined mesh with multiple levels, we use the following procedure.

For each photon, determine its distance until absorption r as described above. Then compute the level l at which the photon should be absorbed as

$$l = \log_2(\kappa r / \Delta x_1), \tag{11.10}$$

where Δx_1 is the mesh spacing at the coarsest grid (level one) and κ is a factor suitably chosen between zero and one, see section 11.4. The number l is then rounded to an integer such that $l_{\min} \leq l \leq l_{\max}$, with the minimum and maximum chosen such that photons are not absorbed on a too coarse or fine mesh for the application at hand. The rounding is performed by drawing a random number $U(0,1)$: if $l - \lfloor l \rfloor > U$ round up, else round down. This randomization is done to reduce mesh artifacts; without it, there would be sudden transitions between mesh levels.

All photons are now absorbed at their levels l, and the resulting absorption densities are added from coarse to fine grids, by linear interpolation. Near the boundaries, some type of boundary conditions must be specified in order to do linear interpolation. For the examples in the next section, we have used Neumann zero boundary conditions.

11.3.4 Physical fluctuations

In the description of the absorption methods, we assumed that the number of discrete photons used is much smaller than the number of physical photons. The discrete photons then become 'super photons', which increases the stochastic fluctuations in their absorption profile. The goal of the absorption methods described above is to reduce these stochastic fluctuations.

Sometimes, the fluctuations produced by a limited number N_{phys} of physical photons are of interest. If N_{phys} is small enough, one can represent the photons individually to obtain a realistic sample of the absorption profile. Note that equation (11.3) does not take such fluctuations into account – it effectively assumes that $N_{\text{phys}} = \infty$.

11.4 Examples

We will now present examples to illustrate the properties of the proposed method. The absorption function from the Zheleznyak [46] model is used for these examples, see equation (11.5). As a gas, nitrogen with 20% oxygen (air) is used at room temperature.

For the examples a 2D cylindrical (r, z) grid of $8\,\text{mm}^2$ is used. The coarsest and finest grid have a spacing of 1 mm and $\sim 7.8\,\mu$m, so that there are 8 levels of refinement (each with a refinement factor of two). A point source of photons is present at $\boldsymbol{r}_c = (0, 4\,\text{mm})$ and $N = 5 \times 10^4$ discrete photons are produced.

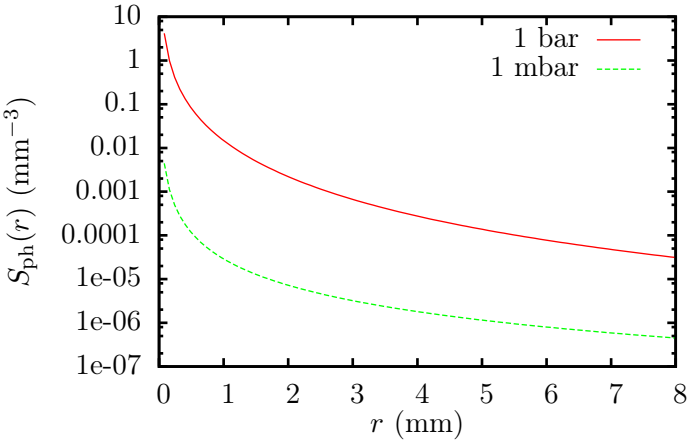

Figure 11.2: The photoionization profile given by equation (11.11) for air at a pressure of 1 bar and 1 mbar, on a logarithmic scale. At 1 bar the mean absorption length is about 0.45 mm, at 1 mbar it is 0.45 m.

With a point source $\delta(\boldsymbol{r} - \boldsymbol{r_c})$ the integral in equation (11.3) simplifies to

$$S_{\text{ph}}(\boldsymbol{r}) = \frac{f(|\boldsymbol{r} - \boldsymbol{r_c}|)}{4\pi|\boldsymbol{r} - \boldsymbol{r_c}|^2}. \tag{11.11}$$

We compare against this solution to evaluate the accuracy of our method, see figure 11.2.

11.4.1 Results for air at 1 bar

In figure 11.3 results are shown for a gas pressure of 1 bar, using 5×10^4 discrete photons. The photons are mapped to densities using either a constant grid spacing Δx or a varying grid spacing, see section 11.3.3.

With a constant grid spacing, the photoionization profile is quite noisy when a small Δx is used. This can be improved by using more photons or by increasing Δx. With a larger grid spacing the Monte Carlo approximation improves away from the point source, but close to it the approximation gets worse. The reason is that the grid becomes to coarse to resolve the absorption profile, as discussed in section 11.3.3. (This does not always matter, see the remark in section 11.2.1.)

With a varying grid spacing for absorption, each photon is absorbed on a grid with a spacing of approximately κ times the photons' distance until absorption. The resulting profiles are generally smoother than those using a constant Δx. With small κ's, there is more noise, but with larger κ's mesh artifacts become visible around the source term. These mesh artifacts occur when the grid for absorption is too coarse to resolve the fast decay of the absorption profile, see figure 11.2. The best value for κ thus depends on the number of photons used:

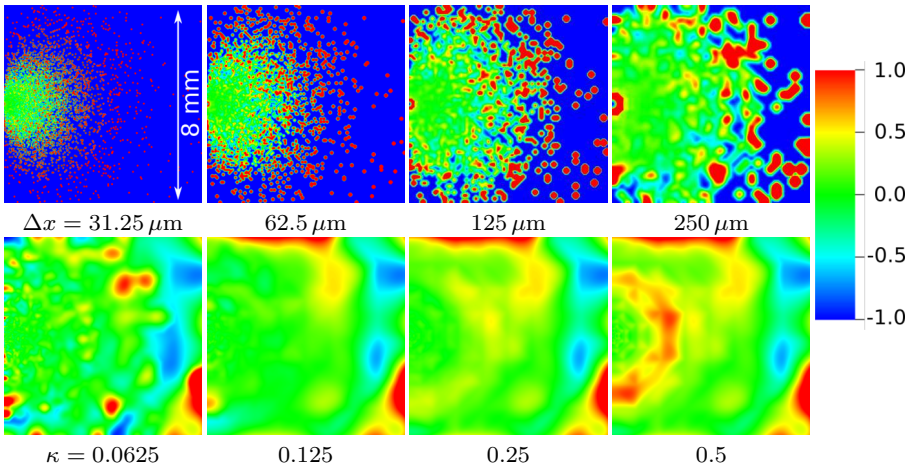

Figure 11.3: The relative error of the Monte Carlo method for air at 1 bar. Top row: results with a constant Δx for absorption (see section 11.3.3). Bottom row: results with varying Δx, controlled by the parameter κ (see section 11.3.3). The same random sample of photons is used for all cases.

with more photons, one can use a smaller value to reduce mesh artifacts while still having reasonably low noise levels.

11.4.2 Results for air at 1 mbar

In figure 11.4 results are shown for the same conditions as in figure 11.3, but now at a pressure of 1 mbar. At this pressure most photons are not absorbed within the domain, so we have used the procedure described in section 11.3.2 to modify the lookup table. Only photons with absorption distances less than twice the domain size were produced.

With a constant grid spacing, there is less noise than for the 1 bar case, because of the larger values of Δx that are used. As before, we see that the solution near the source term cannot be resolved when Δx gets too large. With a varying grid spacing, the results are typically close to the analytic solution in the whole domain. The mesh artifacts for larger values of κ are smaller than for the 1 bar case.

11.5 Conclusion

We have presented a Monte Carlo approach for photoionization in discharge simulations, in which a number of discrete photons is used to approximate the photoionization profile. An efficient sampling method for these photons was described, which can also be used when absorption lengths are much larger than

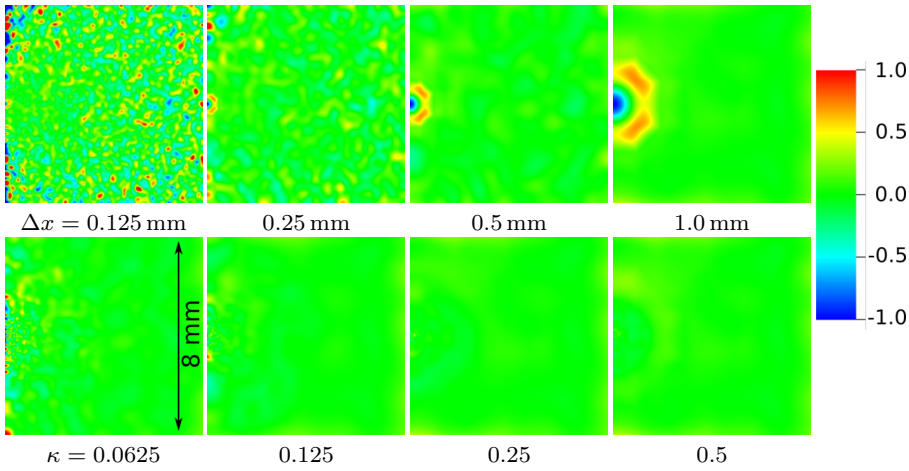

Figure 11.4: The relative error of the Monte Carlo method for air at 1 mbar. Top row: results with a constant Δx for absorption (see section 11.3.3). Bottom row: results with varying Δx (see section 11.3.3). The same random sample of photons is used for all cases.

the computational domain. Two strategies for photon absorption are considered: the photons can be absorbed on a single uniform grid, or on a grid with multiple levels, with the level chosen for each photon. The method presented in this paper has the following advantages:

- The same approach can be used for 2D and 3D simulations and those in cylindrical symmetry, with relatively minor modifications.
- By changing the number of photons and the parameters in the method, one can control the noise in the solution and the accuracy.
- The method is quite flexible and can be used for almost any absorption function $f(r)$.
- Because the start and end location of each photon is known, the interaction of photons with objects and surfaces can be included.
- When adaptive mesh refinement is used, the photoionization profile can be computed on the same mesh as is used for the plasma fluid model.

Although the amount of noise can be controlled in our method, it is difficult to reproduce physically realistic noise levels in all regions. On the other hand, simulations that require such accurate noise levels should probably not be performed with a fluid model.

Our method was not designed to give the best accuracy in the whole solution domain, because that is typically not necessary for photoionization. When an

equation like (11.3) has to be solved with high accuracy, one could use the Fast Fourier Transform (FFT) to perform a convolution with a general kernel, see for example chapter 6-5-4 of [67]. On a Cartesian grid with N grid points, the cost of such an operation is of order $O(N \log N)$.

Chapter 12

Conclusions & outlook

12.1 Conclusions

This thesis deals with various topics, all related to the modeling of streamers and nanosecond pulsed discharges. Below, the main conclusions on these topics are summarized, with references to the relevant chapters.

A general conclusion could be that the simulation of streamer discharges is quite challenging, due to their transient and three-dimensional nature, their steep density gradients, their thin space charge layers which generate strong electric field enhancement and their strongly non-linear propagation – see chapter 2.2.1. The same arguments hold in general for nanosecond pulsed discharges that exhibit strong space-charge effects in a non-trivial geometry. Adaptive mesh refinement (AMR) is an important technique for the simulation of such discharges, especially in three dimensions. This is illustrated in chapter 3, in which the simulations lacked mesh refinement and were limited to a small domain and rather unrealistic conditions. One way to reduce computational costs is to restrict a model to two (or even one) spatial dimensions. However, many discharges do not have the symmetry that is in this way imposed on them, as illustrated in chapters 5–9. For a better understanding of such discharges, three-dimensional simulations are valuable. After this quite general conclusion, we summarize the most important conclusions from individual chapters below:

- **Chapter 3** Both particle and fluid models can be used for streamer simulations. Particle models include more physics, but are computationally more expensive. A *hybrid* model can combine advantages of both models, but the implementation of such a model is challenging. Without adaptive mesh refinement, three-dimensional streamer simulations are limited to small domains and unrealistic conditions.

- **Chapter 4** – To speed up simulations, the weights of simulation particles can be changed adaptively by merging and splitting them. Using random

numbers, pairs of particles can be merged in such a way that energy and momentum are *on average* preserved. The usage of a k-d tree allows to efficiently search for pairs of particles that are close in both position and velocity.

- **Chapter 5** A three-dimensional particle-in-cell model has been constructed, with adaptive mesh refinement, adaptive particle weights, parallelized particle routines and the possibility of including a needle electrode. With this model, we show that the formation of a pulsed discharge depends on the nitrogen/oxygen ratio, which affects the photoionization density.

- **Chapters 6,8** In atmospheric air, we do not expect isolated streamer discharges when the electric field is above the breakdown threshold. Due to background ionization and electron detachment, electron avalanches start to grow in the whole overvolted region. After the 'ionization screening time', these avalanches together screen the electric field. The formation of isolated double-headed streamers, observed in several previous studies, is therefore unlikely.

- **Chapter 7** By taking into account electron impact ionization, the Maxwell time ε_0/σ can be generalized to the 'ionization screening time' τ_{is}. In the limit of negligible ionization, τ_{is} again reduces to ε_0/σ. The predicted screening times are compared with one- and three-dimensional simulations, and a simple criterion for the homogeneity of overvolted discharges is given.

- **Chapter 9** Positive streamers can be guided by weak pre-ionization, i.e., pre-ionization that has a negligible space charge effect. This guiding is demonstrated by experiments and a simulation, which show that a streamer can move almost perpendicular to the background electric field. A requirement for guiding is that the pre-ionized region contains a significantly higher electron density than the bulk gas.

- **Chapter 10** For doing numerical computations on adaptively refined grids, quadtree and octree meshes have attractive properties. We present Afivo, a framework suitable for modest-scale parallel computations on quadtree/octree grids. Afivo can be a simpler alternative for some of the already existing frameworks aimed at large scale parallel computations. Geometric multigrid is highly efficient for solving elliptic equations on structured grids, and is therefore implemented in Afivo.

- **Chapter 11** Photoionization plays an important role in many discharges, but including this process in simulations can be challenging. A Monte Carlo approach for photoionization is presented, suitable for plasma fluid models. By absorbing photons on different grid levels, the photoionization profile can efficiently be approximated.

12.2 Outlook

Using the Afivo framework described in chapter 10, several types of three-dimensional simulations can now be performed, to investigate for example streamer branching, the interaction between two streamers or the propagation of streamers close to flat dielectrics. Many two-dimensional simulations can be performed much more quickly than with previous models, allowing for an interactive exploration of parameter regimes. An important algorithmic development would be the inclusion of curved electrodes in the multigrid method. This probably requires the implementation of a direct solver for the coarse-grid equations. The development of an advection scheme that can handle refinement boundaries would also be valuable.

Compared to plasma fluid models, particle models are computationally often more expensive. This computational expense comes with a big advantage, however: the particle distribution function $f(\boldsymbol{x}, \boldsymbol{v}, t)$ can directly be approximated. Fluid models have to make assumptions about the shape of $f(\boldsymbol{x}, \boldsymbol{v}, t)$, potentially limiting their validity. Particle models can therefore be attractive for problems in which $f(\boldsymbol{x}, \boldsymbol{v}, t)$ behaves in a complicated way – for example near physical boundaries or when electron runaway occurs. Particle models will also be an important tool for the development and validation of fluid approximations, as they have been in the past.

As discussed above, the microscopic modeling of streamer discharges is computationally challenging. An interesting development is the use of so-called 'tree' models [65], see section 2.2.4. In a tree model, the streamers are characterized as macroscopic channels with a velocity, radius, conductivity and branching probability. The development of such tree models is important, because most discharges in nature and in the lab contain many streamer channels, making microscopic simulations computationally unfeasible. The strategy could be to use microscopic simulations of single streamers to characterize the behavior of individual channels in a tree model.

Appendix A

Becoming a computational scientist

I have now been working on computational science and computational physics problems for more than six years. What has surprised me, in hindsight, is the great number of things that one has to be familiar with in order to be productive. The reason is probably the relatively large amount of DIY (do it yourself) in computing, compared to other disciplines. Below I will try to summarize what skills I have found to be generally important.

A.1 Selecting problems

For doing research, the most important skill is perhaps the ability to pick the right problems. This is a rather difficult skill to master, and I am confident that I have not done so yet. Still, there are a couple of simple questions that I find useful for selecting problems:

- Are you interested in the problem?
- Are others interested in the problem?
- Do you expect to learn something useful when studying the problem?
- Does the problem seem feasible to you? If this is not clear, how much time do you approximately have to invest to answer this?
- Suppose that everything works out: you solve the problem. What would that mean to you? And what could you do next?
- How long do you give yourself? And suppose that you are unable to solve the full problem, is there then an intermediate result that could be of value?

- How hard will it be to write about the results? For example, for certain types of results a carefully written introduction, motivation, discussion or analysis might be required.

Another question that becomes more relevant towards the end of a PhD is whether it is possible to obtain (future) grants or funding for a topic.

A.2 Theoretical skills

Below, I briefly discuss some of the theoretical topics that I believe to be important for a computational scientist. The most important topic is missing however, namely knowledge of the domain that you are working in. Such knowledge will help in selecting the right problems and in making the right approximations.

A.2.1 Applied mathematics

These are some of the topics in applied mathematics that I think are important for a computational scientist:

1. Linear algebra: many problems can be written as a system of linear equations.

2. Calculus: ordinary differential equations and Taylor series are important for many numerical methods. Also helps for knowing what can be calculated analytically or for being able to construct reference solutions.

3. Statistics: Monte Carlo methods are quite common; to work with them, at least a basic understanding of statistics is required. The same goes for problems that are probabilistic in nature or contain data with noise.

A.2.2 Computer science

When we want to solve a problem on a computer, we have to select the appropriate *algorithm*. Algorithms can be classified by their 'difficulty' or computational cost, which is the main topic of *computational complexity theory*. Knowing and understanding the computational cost of algorithms is not only important for efficiently solving a problem, but also for predicting what problems are feasible. For example, if you recognize that you are trying to solve an *NP-hard*[1] problem, then you immediately know that you are limited to small problem sizes. With parallel computing, it is usually possible to go to larger problem sizes. To what extent this is the case depends on how well the algorithmic components can be parallelized, i.e., on the amount of local computation versus global communication.

[1] See https://en.wikipedia.org/wiki/NP-hard

The practical cost of algorithms also has to do with the device that performs the algorithmic steps or computations. Modern processors operate in a rather complicated way, but knowledge of the cost of typical operations is important when you have to develop an efficient numerical method. The hardware in a processor also determines what integer and floating point numbers you can use. Understanding floating point arithmetic and its subtleties can save you a lot of time debugging 'weird' behavior.

A.2.3 Computational science

Although there are many types of computations, most of them can be categorized into just a few categories:

- Solving linear systems of equations, i.e., solve $Ax = b$ for a given matrix A and vector b. Surprisingly many problems can be transformed into such a linear system.
- Optimization, for example: find the shortest path between N cities, find the ground state energy of a quantum system or find the minimum of a function.
- Ordinary and partial differential equations. Many (physical) systems can be described by such equations. Different types of partial differential equations require quite different solution strategies.

A computational scientist should probably be familiar with the basic methods for solving problems from these categories, so that one is able to find and select the best methods when the need arises. To prevent reinventing the wheel, some knowledge of the available libraries and codes is valuable.

A.3 Practical skills

The best strategy for solving a problem depends on what tools are already available. If sufficiently many other people have worked on a (similar) problem, software might be available that you can directly use. Take for example CFD (computational fluid dynamics), for which there are many different simulation tools. Selecting the right one then becomes one of the most important aspects of solving your problem.

The other extreme would be that no existing software exists for your problem, so that you have to develop everything yourself. There are of course also many cases in between, for example when existing tools have to be modified to suit your needs. This means that it is often necessary to write computer code. Below, some of the practical aspects of writing your own code and reusing others' code are discussed.

A.3.1 Computer basics

For computing, the `*nix` operating systems appear to be most popular. Being familiar with a variant of e.g., GNU/Linux, BSD or OS X is therefore quite helpful – this allows you to quickly use the code and tools that others have written.

Good command of a text editor such as `vim` or `emacs`, or a suitable IDE (integrated development environment) will speed up your code and text editing. This might also reduce the risk of developing RSI (repetitive strain injury), because most editors can be operated without a mouse[2]. There are many useful tools included in a `*nix` system, but `ssh` gets a special mention, because it allows you to work on remote systems.

There exist a number of software suites for doing numerical or symbolic computations. Commercial packages are for example Matlab and Mathematica, whereas Octave [206] or SageMath [207] are examples of free software alternatives. The many built-in functions can help you to quickly develop a computational method. Even if you eventually have to implement this solver in a different environment, it can be helpful to start from a simple proof-of-concept. The generality of such suites is also their drawback: typically they will not be as efficient as a special purpose solution.

A.3.2 Programming

When you develop a method from scratch, you can use your preferred programming language – this is of course not possible when you have to modify an existing method. The traditional languages for computing are C and Fortran. Especially C is quite 'low-level', so that experience with C will be useful for understanding how a computer and other languages work. Fortran was specifically designed for numerical computing, which can make code development more convenient. Another popular *compiled* language is C++, which allows for many programming styles. This flexibility can be good for the expert but is sometimes hard for the beginner. Performance wise, there are no major differences between these languages as long as you know what you are doing.

For certain tasks, scripting or interpreted languages such as Python can be more convenient. Such languages can for example be used to glue together other programs, to process data or to visualize results. Python can also be used for computations, although the numerical work is then typically performed by routines written in C or Fortran, which are made available by Python modules such as `numpy`.

Numerical code is no different from other code: many things can go wrong. Sometimes a program simply does not compile or run, but at other times it might

[2] In my experience, the combination of stress and mouse usage is most likely to cause physical discomfort.

not be clear whether there is a *bug* or whether there is a failure for another reason. Code often depends on (particular versions of) libraries, which is a source of compilation errors; understanding how code is compiled will help in figuring out what is required. Another example are the *Makefiles*[3] included with numerical software: they might not work on your machine, in which case you need to know how to modify them. As most programs contain bugs, basic debugging skills are very valuable. The larger a project grows, the more important these skills become.

Being familiar with a version control system such as git has various benefits: you can keep tracks of your changes, get the latest version of a code, collaborate with others etcetera. Perhaps even more important is being able to visualize your results. There exist many tools for this, examples of popular open source packages are gnuplot, Visit [135] and Paraview [200].

[3]Makefiles contain rules that describe how a collection of source files should be compiled. Another common build system is `CMake`.

Appendix B

Implementing numerical algorithms: the Koren flux limiter

B.1 Introduction

An important aspect of scientific computing is the actual implementation of an algorithm. Here we demonstrate this for a relatively short algorithm: the Koren flux limiter [49]. Flux limiters are used to prevent oscillations in flow simulations, by switching gradually between a low order scheme (for the difficult regions) and a higher order scheme. The goal of this appendix is to show that a robust implementation of an algorithm can look quite different from the original definition. We also describe some of the general numerical problems that one encounters when implementing algorithms.

B.2 Koren limiter

We discuss the implementation of the *Koren limiter*. The original paper [49] gives the following definition for the flux at a cell face

$$f_{i+\frac{1}{2}} = \begin{cases} u_{i+\frac{1}{2}} \left(c_i + \frac{1}{2}\phi(r^+_{i+\frac{1}{2}})(c_i - c_{i-1}) \right) & \text{for } u_{i+\frac{1}{2}} \geq 0 \\ u_{i+\frac{1}{2}} \left(c_{i+1} + \frac{1}{2}\phi(r^-_{i+\frac{1}{2}})(c_{i+1} - c_{i+2}) \right) & \text{for } u_{i+\frac{1}{2}} < 0, \end{cases} \tag{B.1}$$

where $u_{i+\frac{1}{2}}$ is the velocity at the cell face and c_i is the density at the center of cell i. The flux limiter ϕ is defined as follows, see figure B.1:

$$\phi(r) = \max\left(0, \min\left(2r, \min\left(\frac{1}{3} + \frac{2r}{3}, 2\right)\right)\right), \tag{B.2}$$

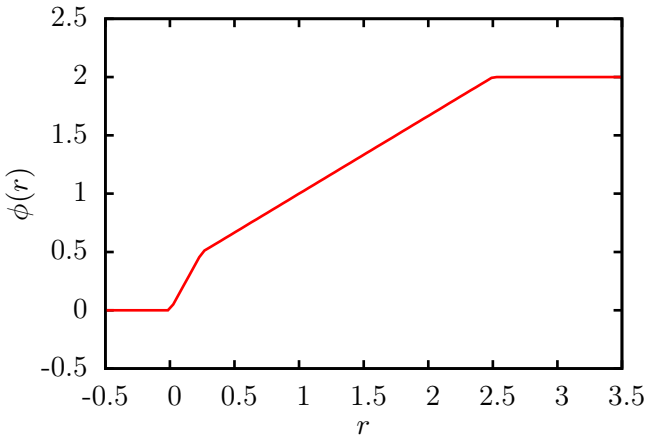

Figure B.1: The Koren limiter function $\phi(r)$.

and r is the *upwind ratio of consecutive solution gradients*, which for $u_{i+\frac{1}{2}} \geq 0$ is given by

$$r^{+}_{i+\frac{1}{2}} = \frac{c_{i+1} - c_i + \epsilon}{c_i - c_{i-1} + \epsilon}, \tag{B.3}$$

and for $u_{i+\frac{1}{2}} < 0$, the definition is

$$r^{-}_{i+\frac{1}{2}} = \frac{c_i - c_{i+1} + \epsilon}{c_{i+1} - c_{i+2} + \epsilon}. \tag{B.4}$$

Here ϵ is 'some very small number (e.g. $\epsilon = 10^{-10}$), introduced to avoid, e.g., division by zero in uniform flow regions' [49].

B.3 Implementation

In principle, we could directly implement the algorithm described in the previous section, because all operations can be translated to instructions in `C`, `Fortran` or some other language. That implementation will not be the most robust and efficient, however. For example, we would like to remove ϵ's from equations (B.3) and (B.4). The denominator in these equation can still be zero when the terms cancel out. This can be prevented by automatically changing the sign of ϵ, but then the value of ϵ is still somewhat arbitrary. Furthermore, for small densities $c_i \sim \epsilon$, the ϵ term could cause inaccuracies.

B.3.1 Potentially unsafe operators

In a way, most floating point operation can be regarded as *unsafe*, because with the 'wrong' arguments, the result might be undefined or $\pm\infty$. However, some operations are more likely to fail than others. For example, $x + y$ or $x \cdot y$ works

fine as long as the result can be represented, but operations like y/x, $\sqrt{x}$ or $\log(x)$ have to be treated more carefully. The reason is that these functions have a domain that excludes some 'typical' floating point values, such as zero or negative numbers.

For example, $1/0$ gives ∞ on most systems (a special floating point value), and $0/0$ gives `NaN`, or Not a Number. However, division by a non-zero number is also unsafe. If $z = y/x$ is smaller or larger than the numbers that can be represented in the floating point system, one still ends up with $\pm\infty$.

When one works with real numbers, $\sqrt{x}$ gives `NaN` for $x < 0$, and logarithms require positive arguments. There are of course more functions that are defined on a limited domain, e.g., the arcsine. The occurrence of `NaN`'s or ∞'s in numerical calculations is often caused by calling such functions with an invalid argument, often caused by rounding errors. For example, on most computers evaluating $0.3 - 3x$ in double precision for $x = 0.1$ gives about -5.55×10^{-17}. A detailed discussion of floating point arithmetic can be found in e.g., [208, 209].

Note that we just consider robustness here, not accuracy. Different ways of calculating the same value can differ in their accuracy. And a practical tip: it can be convenient to enable trapping on floating point exceptions in your compiler, to detect the problems described in this section.

B.3.2 Actual implementation

We first define a function ϕ_1 that incorporates the factor $1/2$ from equation (B.1)

$$\phi_1(r) \equiv \tfrac{1}{2}\phi(r) = \max\left(0, \min\left(r, \min\left(\frac{1+2r}{6}, 1\right)\right)\right). \tag{B.5}$$

If we write $r_{i+\frac{1}{2}} = a/b$, where a and b are the terms of equations (B.3) and (B.4) without the ϵ, then equation (B.1) becomes

$$f_{i+\frac{1}{2}} = \begin{cases} u_{i+\frac{1}{2}}\,(c_i + b\phi_1(a/b)) & \text{for } u_{i+\frac{1}{2}} \geq 0 \\ u_{i+\frac{1}{2}}\,(c_{i+1} + b\phi_1(a/b)) & \text{for } u_{i+\frac{1}{2}} < 0. \end{cases} \tag{B.6}$$

For ϕ_1 four cases should be considered (see figure B.1, which shows ϕ)

$$\phi_1(r) = \begin{cases} 0 & \text{for } r \leq 0 \\ r & \text{for } 0 < r \leq 1/4 \\ (1+2r)/6 & \text{for } 1/4 < r \leq 5/2 \\ 1 & \text{for } r > 5/2. \end{cases} \tag{B.7}$$

We now define a new function

$$\phi_2(a, b) = b\phi_1(a/b), \tag{B.8}$$

```
real function phi2(a, b)
  real, intent(in) :: a, b
  real, parameter  :: sixth = 1/6.0
  real             :: aa, ab

  aa = a * a
  ab = a * b

  if (ab <= 0) then
     phi2 = 0
  else if (aa <= 0.25 * ab) then
     phi2 = a
  else if (aa <= 2.5 * ab) then
     phi2 = sixth * (b + 2 * a)
  else
     phi2 = b
  end if
end function phi2
```

Figure B.2: Fortran implementation of $\phi_2(a,b) = b\phi_1(a/b)$, see equation (B.5). Note that for a double precision version the constants should also be modified.

which we can compute without any division, for example as in figure B.2. A small trick is used: $a/b < x$ is not equivalent to $a < bx$, because b can be negative or zero. However, if $ab > 0$ then $a/b < x \iff a^2 < abx$.

Our implementation of the Koren flux limiter is then as follows: Depending on the sign of the velocity, define three variables a, b and c in the following way

$$a, b, c = \begin{cases} c_{i+1} - c_i, c_i - c_{i-1}, c_i & \text{for } u_{i+\frac{1}{2}} \geq 0 \\ c_i - c_{i+1}, c_{i+1} - c_{i+2}, c_{i+1} & \text{for } u_{i+\frac{1}{2}} < 0, \end{cases} \tag{B.9}$$

after which the flux can be computed as

$$f_{i+\frac{1}{2}} = u_{i+\frac{1}{2}} \left(c + \phi_2(a,b)\right),$$

where $\phi_2(a,b)$ is the routine from figure B.2.

On an Intel `i7-3770S` CPU, this implementation of $\phi_2(a,b)$ is about twice as fast as a similar implementation of $b\phi_1(r)$, see equation (B.5).

B.4 Conclusion

We have presented a robust implementation of the Koren flux limiter, which looks quite different from the original description of the algorithm. Besides conditional

statements (`if` constructs), our implementation uses only multiplication and addition.

Carefully implementing an algorithm takes some effort, but this effort might save considerable debugging time. This is especially important for large and complex computations. For example, the occurrence of `NaN` (not a number) after several hours of parallel computing can easily take more than a day to debug.

Bibliography

[1] V. P. Pasko. *Sprites, Elves and Intense Lightning Discharges*, volume 225 of *NATO Science Series II: Mathematics, Physics and Chemistry*, chapter Theoretical modeling of sprites and jets, pages 253–311. Springer Netherlands, 2006. URL http://www.ee.psu.edu/Directory/FacultyInfo/Pasko/Publications/PaskoSpringer2006.pdf.

[2] Ute Ebert, Sander Nijdam, Chao Li, Alejandro Luque, Tanja Briels, and Eddie van Veldhuizen. Review of recent results on streamer discharges and discussion of their relevance for sprites and lightning. *Journal of Geophysical Research*, 115, 2010. doi:10.1029/2009ja014867.

[3] Z Bonaventura, A Bourdon, S Celestin, and V P Pasko. Electric field determination in streamer discharges in air at atmospheric pressure. *Plasma Sources Sci. Technol.*, 20(3):035012, Jun 2011. doi:10.1088/0963-0252/20/3/035012.

[4] U. Ebert and D.D. Sentman. *Cluster Issue: Streamers, Sprites and Lightning*. Journal of physics: Applied physics. IOP Publishing, 2008.

[5] S. Nijdam, P. Bruggeman, E.M. van Veldhuizen, and U. Ebert. *Plasma Chemistry and Catalysis in Gases and Liquids*, chapter An introduction to nonequilibrium plasmas at atmospheric pressure. Wiley-VCH, 2012.

[6] E.M. van Veldhuizen. *Electrical Discharges for Environmental Purposes: Fundamentals and Applications*. Nova Science Publishers, New York, 2000.

[7] Gregory Fridman, Gary Friedman, Alexander Gutsol, Anatoly B. Shekhter, Victor N. Vasilets, and Alexander Fridman. Applied plasma medicine. *Plasma Process. Polym.*, 5(6):503–533, Aug 2008. doi:10.1002/ppap.200700154.

[8] Andrei Y. Starikovskii, Nikolai B. Anikin, Ilya N. Kosarev, Eugeny I. Mintoussov, Maria M. Nudnova, Aleksandr E. Rakitin, Dmitry V. Roupassov, Svetlana M. Starikovskaia, and Victor P. Zhukov. Nanosecond-pulsed discharges for plasma-assisted combustion and aerodynam-

ics. *Journal of Propulsion and Power*, 24(6):1182–1197, Nov 2008. doi:10.2514/1.24576.

[9] D. V. Rose, D. R. Welch, R. E. Clark, C. Thoma, W. R. Zimmerman, N. Bruner, P. K. Rambo, and B. W. Atherton. Towards a fully kinetic 3D electromagnetic particle-in-cell model of streamer formation and dynamics in high-pressure electronegative gases. *Phys. Plasmas*, 18(9):093501, 2011. doi:10.1063/1.3629989.

[10] Chao Li, Ute Ebert, and W. J. M. Brok. Avalanche-to-streamer transition in particle simulations. *IEEE Trans. Plasma Sci.*, 36(4):910–911, Aug 2008. doi:10.1109/tps.2008.922487.

[11] O. Chanrion and T. Neubert. A PIC-MCC code for simulation of streamer propagation in air. *Journal of Computational Physics*, 227(15):7222–7245, Jul 2008. doi:10.1016/j.jcp.2008.04.016.

[12] Alejandro Luque, Valeria Ratushnaya, and Ute Ebert. Positive and negative streamers in ambient air: modelling evolution and velocities. *J. Phys. D: Appl. Phys.*, 41(23):234005, Nov 2008. doi:10.1088/0022-3727/41/23/234005.

[13] Victor P. Pasko, Umran S. Inan, and Timothy F. Bell. Spatial structure of sprites. *Geophys. Res. Lett.*, 25(12):2123–2126, Jun 1998. doi:10.1029/98gl01242.

[14] Ningyu Liu. Effects of photoionization on propagation and branching of positive and negative streamers in sprites. *Journal of Geophysical Research*, 109(A4), 2004. doi:10.1029/2003ja010064.

[15] Jianqi Qin, Sebastien Celestin, and Victor P. Pasko. Formation of single and double-headed streamers in sprite-halo events. *Geophys. Res. Lett.*, 39 (5), Mar 2012. doi:10.1029/2012gl051088.

[16] V P Pasko. Red sprite discharges in the atmosphere at high altitude: the molecular physics and the similarity with laboratory discharges. *Plasma Sources Sci. Technol.*, 16(1):S13–S29, Jan 2007. doi:10.1088/0963-0252/16/1/s02.

[17] A Bourdon, V P Pasko, N Y Liu, S Célestin, P Ségur, and E Marode. Efficient models for photoionization produced by non-thermal gas discharges in air based on radiative transfer and the helmholtz equations. *Plasma Sources Sci. Technol.*, 16(3):656–678, Aug 2007. doi:10.1088/0963-0252/16/3/026.

[18] D Bessières, J Paillol, A Bourdon, P Ségur, and E Marode. A new one-dimensional moving mesh method applied to the simulation of streamer

discharges. *J. Phys. D: Appl. Phys.*, 40(21):6559–6570, Oct 2007. doi:10.1088/0022-3727/40/21/016.

[19] A. Luque and U. Ebert. Density models for streamer discharges: Beyond cylindrical symmetry and homogeneous media. *Journal of Computational Physics*, 231(3):904–918, Feb 2012. doi:10.1016/j.jcp.2011.04.019.

[20] Chao Li, Ute Ebert, and Willem Hundsdorfer. Spatially hybrid computations for streamer discharges with generic features of pulled fronts: I. planar fronts. *Journal of Computational Physics*, 229(1):200–220, Jan 2010. doi:10.1016/j.jcp.2009.09.027.

[21] Chao Li, Ute Ebert, and Willem Hundsdorfer. Spatially hybrid computations for streamer discharges: II. fully 3D simulations. *Journal of Computational Physics*, 231(3):1020–1050, Feb 2012. doi:10.1016/j.jcp.2011.07.023.

[22] Chao Li, Ute Ebert, and Willem Hundsdorfer. 3D hybrid computations for streamer discharges and production of runaway electrons. *J. Phys. D: Appl. Phys.*, 42(20):202003, Sep 2009. doi:10.1088/0022-3727/42/20/202003.

[23] U Ebert, C Montijn, T M P Briels, W Hundsdorfer, B Meulenbroek, A Rocco, and E M van Veldhuizen. The multiscale nature of streamers. *Plasma Sources Science and Technology*, 15(2):S118–S129, Apr 2006. doi:10.1088/0963-0252/15/2/s14.

[24] U Ebert, F Brau, G Derks, W Hundsdorfer, C-Y Kao, C Li, A Luque, B Meulenbroek, S Nijdam, V Ratushnaya, and et al. Multiple scales in streamer discharges, with an emphasis on moving boundary approximations. *Nonlinearity*, 24(1):C1–C26, Dec 2010. doi:10.1088/0951-7715/24/1/c01.

[25] Manuel Arrayás, Ute Ebert, and Willem Hundsdorfer. Spontaneous branching of anode-directed streamers between planar electrodes. *Physical Review Letters*, 88(17), Apr 2002. doi:10.1103/physrevlett.88.174502.

[26] Carolynne Montijn, Ute Ebert, and Willem Hundsdorfer. Numerical convergence of the branching time of negative streamers. *Phys. Rev. E*, 73(6), Jun 2006. doi:10.1103/physreve.73.065401.

[27] C.-Y. Kao, F. Brau, U. Ebert, L. Schäfer, and S. Tanveer. A moving boundary model motivated by electric breakdown: II. initial value problem. *Physica D: Nonlinear Phenomena*, 239(16):1542–1559, Aug 2010. doi:10.1016/j.physd.2010.03.011.

[28] A. Luque and U. Ebert. Electron density fluctuations accelerate the branching of positive streamer discharges in air. *Phys. Rev. E*, 84(4), Oct 2011. doi:10.1103/physreve.84.046411.

[29] C. Montijn, W. Hundsdorfer, and U. Ebert. An adaptive grid refinement strategy for the simulation of negative streamers. *Journal of Computational Physics*, 219(2):801–835, Dec 2006. doi:10.1016/j.jcp.2006.04.017.

[30] J. Adams, P. Swarztrauber, and R. Sweet. Fishpack90, http://www.cisl.ucar.edu/css/software/fishpack90. [Online; accessed 2011].

[31] N.H. Malik. A review of the charge simulation method and its applications. *IEEE Trans. Elect. Insul.*, 24(1):3–20, 1989. doi:10.1109/14.19861.

[32] She Chen, L C J Heijmans, Rong Zeng, S Nijdam, and U Ebert. Nanosecond repetitively pulsed discharges in N_2-O_2 mixtures: inception cloud and streamer emergence. *J. Phys. D: Appl. Phys.*, 48(17):175201, Mar 2015. doi:10.1088/0022-3727/48/17/175201.

[33] T T J Clevis, S Nijdam, and U Ebert. Inception and propagation of positive streamers in high-purity nitrogen: effects of the voltage rise rate. *J. Phys. D: Appl. Phys.*, 46(4):045202, Dec 2012. doi:10.1088/0022-3727/46/4/045202.

[34] S Nijdam, G Wormeester, E M van Veldhuizen, and U Ebert. Probing background ionization: positive streamers with varying pulse repetition rate and with a radioactive admixture. *J. Phys. D: Appl. Phys.*, 44(45): 455201, Oct 2011. doi:10.1088/0022-3727/44/45/455201.

[35] CWI multiscale dynamics webpage. http://cwimd.nl. Accessed: 2014-08-16.

[36] O. Chanrion and T. Neubert. Production of runaway electrons by negative streamer discharges. *Journal of Geophysical Research*, 115, 2010. doi:10.1029/2009ja014774.

[37] A A Kulikovsky. Positive streamer between parallel plate electrodes in atmospheric pressure air. *J. Phys. D: Appl. Phys.*, 30(3):441–450, Feb 1997. doi:10.1088/0022-3727/30/3/017.

[38] S. Dhali and P. Williams. Numerical simulation of streamer propagation in nitrogen at atmospheric pressure. *Phys. Rev. A*, 31(2):1219–1221, Feb 1985. doi:10.1103/physreva.31.1219.

[39] Achi Brandt and Oren E. Livne. *Multigrid Techniques*. Society for Industrial & Applied Mathematics (SIAM), Jan 2011. ISBN http://id.crossref.org/isbn/978-1-61197-075-3. doi:10.1137/1.9781611970753.

[40] U. Trottenberg, C.W. Oosterlee, and A. Schuller. *Multigrid*. Elsevier Science, 2000. ISBN 9780080479569.

[41] William L. Briggs, Van Emden Henson, and Steve F. McCormick. *A Multigrid Tutorial (2^{nd} Ed.)*. Society for Industrial & Applied Mathematics, Philadelphia, PA, USA, 2000. ISBN 0-89871-462-1.

[42] P. Colella, D. T. Graves, J. N. Johnson, N. D. Keen, T. J. Ligocki, D. F. Martin, P. W. McCorquodale, D. Modiano, P. O. Schwartz, T. D. Sternberg, and B. Van Straalen. Chombo software package for AMR applications - design document, 2011. URL `https://apdec.org/designdocuments/ChomboDoc/ChomboDesign/chomboDesign.pdf`.

[43] Ann S. Almgren et al. Boxlib. URL `https://ccse.lbl.gov/BoxLib/index.html`. [Online; accessed 22-July-2015].

[44] Peter MacNeice, Kevin M. Olson, Clark Mobarry, Rosalinda de Fainchtein, and Charles Packer. Paramesh: A parallel adaptive mesh refinement community toolkit. *Computer Physics Communications*, 126(3):330–354, Apr 2000. doi:10.1016/s0010-4655(99)00501-9.

[45] Sergey Pancheshnyi. Photoionization produced by low-current discharges in O_2, air, N_2 and CO_2. *Plasma Sources Sci. Technol.*, 24(1):015023, Dec 2014. doi:10.1088/0963-0252/24/1/015023.

[46] M. B. Zheleznyak, A. K. Mnatsakanian, and S. V. Sizykh. Photoionization of nitrogen and oxygen mixtures by radiation from a gas discharge. *Teplofizika Vysokikh Temperatur*, 20:423–428, Nov 1982.

[47] Alejandro Luque, Ute Ebert, Carolynne Montijn, and Willem Hundsdorfer. Photoionization in negative streamers: Fast computations and two propagation modes. *Appl. Phys. Lett.*, 90(8):081501, 2007. doi:10.1063/1.2435934.

[48] P Ségur, A Bourdon, E Marode, D Bessieres, and J H Paillol. The use of an improved eddington approximation to facilitate the calculation of photoionization in streamer discharges. *Plasma Sources Sci. Technol.*, 15 (4):648–660, Jul 2006. doi:10.1088/0963-0252/15/4/009.

[49] B. Koren. A robust upwind discretization method for advection, diffusion and source terms. In C.B. Vreugdenhil and B. Koren, editors, *Numerical Methods for Advection-Diffusion Problems*, pages 117–138. Braunschweig/Wiesbaden: Vieweg, 1993.

[50] T M P Briels, E M van Veldhuizen, and U Ebert. Positive streamers in air and nitrogen of varying density: experiments on similarity laws. *J. Phys. D: Appl. Phys.*, 41(23):234008, Nov 2008. doi:10.1088/0022-3727/41/23/234008.

[51] T M P Briels, J Kos, G J J Winands, E M van Veldhuizen, and U Ebert. Positive and negative streamers in ambient air: measuring diameter, velocity and dissipated energy. *J. Phys. D: Appl. Phys.*, 41(23):234004, Nov 2008. doi:10.1088/0022-3727/41/23/234004.

[52] A. Luque and U. Ebert. Sprites in varying air density: Charge conservation, glowing negative trails and changing velocity. *Geophys. Res. Lett.*, 37(6), Mar 2010. doi:10.1029/2009gl041982.

[53] G. V. Naidis. Positive and negative streamers in air: Velocity-diameter relation. *Phys. Rev. E*, 79(5), May 2009. doi:10.1103/physreve.79.057401.

[54] Chao Li. *Joining particle and fluid aspects in streamer simulations*. PhD thesis, Technische Universiteit Eindhoven, 2009.

[55] Y.P. Raizer. *Gas Discharge Physics*. Springer, 1991. ISBN 3540194622.

[56] Walter A. Lyons. The meteorology of transient luminous events – an introduction and overview. *NATO Science Series II: Mathematics, Physics and Chemistry*, pages 19–56, 2006. doi:10.1007/1-4020-4629-4_2.

[57] S. Pancheshnyi. Role of electronegative gas admixtures in streamer start, propagation and branching phenomena. *Plasma Sources Sci. Technol.*, 14 (4):645–653, Aug 2005. doi:10.1088/0963-0252/14/4/002.

[58] E M van Veldhuizen and W R Rutgers. Pulsed positive corona streamer propagation and branching. *J. Phys. D: Appl. Phys.*, 35(17):2169–2179, Aug 2002. doi:10.1088/0022-3727/35/17/313.

[59] A. A. Kulikovsky. Comment on "spontaneous branching of anode-directed streamers between planar electrodes". *Physical Review Letters*, 89(22), Nov 2002. doi:10.1103/physrevlett.89.229401.

[60] A. V. Gurevich and A. N. Karashtin. Runaway breakdown and hydrometeors in lightning initiation. *Physical Review Letters*, 110(18), May 2013. doi:10.1103/physrevlett.110.185005.

[61] Gregory D. Moss, Victor P. Pasko, Ningyu Liu, and Georgios Veronis. Monte Carlo model for analysis of thermal runaway electrons in streamer tips in transient luminous events and streamer zones of lightning leaders. *Journal of Geophysical Research*, 111(A2), 2006. doi:10.1029/2005ja011350.

[62] J.S. Clements, A. Mizuno, W.C. Finney, and R.H. Davis. Combined removal of SO_2, NO_x, and fly ash from simulated flue gas using pulsed streamer corona. *IEEE Trans. on Ind. Applicat.*, 25(1):62–69, 1989. doi:10.1109/28.18870.

[63] Mark J Kushner. Hybrid modelling of low temperature plasmas for fundamental investigations and equipment design. *J. Phys. D: Appl. Phys.*, 42 (19):194013, Sep 2009. doi:10.1088/0022-3727/42/19/194013.

[64] L. Niemeyer, L. Pietronero, and H. Wiesmann. Fractal dimension of dielectric breakdown. *Physical Review Letters*, 52(12):1033–1036, Mar 1984. doi:10.1103/physrevlett.52.1033.

[65] Alejandro Luque and Ute Ebert. Growing discharge trees with self-consistent charge transport: the collective dynamics of streamers. *New Journal of Physics*, 16(1):013039, Jan 2014. doi:10.1088/1367-2630/16/1/013039.

[66] Robert Buschauer. Magnetic field due to a finite length current-carrying wire using the concept of displacement current. *Phys. Teach.*, 52(7):413–414, Oct 2014. doi:10.1119/1.4895357.

[67] R. W. Hockney and J. W. Eastwood. *Computer simulation using particles.* IOP Publishing Ltd., Bristol, England, 1988.

[68] Paul N. Swarztrauber and Roland A. Sweet. Algorithm 541: Efficient Fortran subprograms for the solution of separable elliptic partial differential equations [D3]. *TOMS*, 5(3):352–364, Sep 1979. doi:10.1145/355841.355850.

[69] John C. Adams. MUDPACK: Multigrid portable Fortran software for the efficient solution of linear elliptic partial differential equations. *Applied Mathematics and Computation*, 34(2):113–146, Nov 1989. doi:10.1016/0096-3003(89)90010-6.

[70] Robert D. Falgout and Ulrike Meier Yang. Hypre: A library of high performance preconditioners. In *Proceedings of the International Conference on Computational Science-Part III*, ICCS '02, pages 632–641, London, UK, UK, 2002. Springer-Verlag. ISBN 3-540-43594-8. URL `http://dl.acm.org/citation.cfm?id=645459.653635`.

[71] Patrick R. Amestoy, Iain S. Duff, Jean-Yves L'Excellent, and Jacko Koster. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, Jan 2001. doi:10.1137/s0895479899358194.

[72] Timothy A. Davis. Algorithm 832. *TOMS*, 30(2):196–199, Jun 2004. doi:10.1145/992200.992206.

[73] Youcef Saad and Martin H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. and Stat. Comput.*, 7(3):856–869, Jul 1986. doi:10.1137/0907058.

[74] Van Emden Henson and Ulrike Meier Yang. BoomerAMG: A parallel algebraic multigrid solver and preconditioner. *Applied Numerical Mathematics*, 41(1):155–177, Apr 2002. doi:10.1016/s0168-9274(01)00115-5.

[75] Chao Li, Jannis Teunissen, Margreet Nool, Willem Hundsdorfer, and Ute Ebert. A comparison of 3D particle, fluid and hybrid simulations for negative streamers. *Plasma Sources Science and Technology*, 21(5):055019, Sep 2012. doi:10.1088/0963-0252/21/5/055019.

[76] Chao Li, W. J. M. Brok, Ute Ebert, and J. J. A. M. van der Mullen. Deviations from the local field approximation in negative streamer heads. *Journal of Applied Physics*, 101(12):123305, 2007. doi:10.1063/1.2748673.

[77] Sebastien Celestin and Victor P. Pasko. Energy and fluxes of thermal runaway electrons produced by exponential growth of streamers during the stepping of lightning leaders and in transient luminous events. *Journal of Geophysical Research: Space Physics*, 116(A3), Mar 2011. doi:10.1029/2010ja016260.

[78] G. J. Fishman, P. N. Bhat, R. Mallozzi, J. M. Horack, T. Koshut, C. Kouveliotou, G. N. Pendleton, C. A. Meegan, R. B. Wilson, W. S. Paciesas, and et al. Discovery of intense gamma-ray flashes of atmospheric origin. *Science*, 264(5163):1313–1316, May 1994. doi:10.1126/science.264.5163.1313.

[79] Joseph R. Dwyer, Brian W. Grefenstette, and David M. Smith. High-energy electron beams launched into space by thunderstorms. *Geophys. Res. Lett.*, 35(2), 2008. doi:10.1029/2007gl032430.

[80] Michael S. Briggs, Valerie Connaughton, Colleen Wilson-Hodge, Robert D. Preece, Gerald J. Fishman, R. Marc Kippen, P. N. Bhat, William S. Paciesas, Vandiver L. Chaplin, Charles A. Meegan, and et al. Electron-positron beams from terrestrial lightning observed with fermi gbm. *Geophys. Res. Lett.*, 38(2), Jan 2011. doi:10.1029/2010gl046259.

[81] Natalia Yu Babaeva, Ananth N Bhoj, and Mark J Kushner. Streamer dynamics in gases containing dust particles. *Plasma Sources Sci. Technol.*, 15(4):591–602, Jul 2006. doi:10.1088/0963-0252/15/4/001.

[82] S. Pancheshnyi, P. Ségur, J. Capeillère, and A. Bourdon. Numerical simulation of filamentary discharges with parallel adaptive mesh refinement. *Journal of Computational Physics*, 227(13):6574–6590, Jun 2008. doi:10.1016/j.jcp.2008.03.020.

[83] S. Kacem, O. Eichwald, O. Ducasse, N. Renon, M. Yousfi, and K. Charrada. Full multi grid method for electric field computation in point-to-plane streamer discharge in air at atmospheric pressure. *Journal of Computational Physics*, 231(2):251–261, Jan 2012. doi:10.1016/j.jcp.2011.08.003.

[84] Max Duarte, Zdeněk Bonaventura, Marc Massot, Anne Bourdon, Stéphane Descombes, and Thierry Dumont. A new numerical strategy with space-time adaptivity and error control for multi-scale streamer discharge simulations. *Journal of Computational Physics*, 231(3):1002–1019, Feb 2012. doi:10.1016/j.jcp.2011.07.002.

[85] V.I. Kolobov and R.R. Arslanbekov. Towards adaptive kinetic-fluid simulations of weakly ionized plasmas. *Journal of Computational Physics*, 231 (3):839–869, Feb 2012. doi:10.1016/j.jcp.2011.05.036.

[86] A P Papadakis, G E Georghiou, and A C Metaxas. New high quality adaptive mesh generator utilized in modelling plasma streamer propagation at atmospheric pressures. *J. Phys. D: Appl. Phys.*, 41(23):234019, Nov 2008. doi:10.1088/0022-3727/41/23/234019.

[87] S Nijdam, F M J H van de Wetering, R Blanc, E M van Veldhuizen, and U Ebert. Probing photo-ionization: experiments on positive streamers in pure gases and mixtures. *J. Phys. D: Appl. Phys.*, 43(14):145204, Mar 2010. doi:10.1088/0022-3727/43/14/145204.

[88] G Wormeester, S Pancheshnyi, A Luque, S Nijdam, and U Ebert. Probing photo-ionization: simulations of positive streamers in varying N_2:O_2-mixtures. *J. Phys. D: Appl. Phys.*, 43(50):505201, Dec 2010. doi:10.1088/0022-3727/43/50/505201.

[89] Gideon Wormeester, Sander Nijdam, and Ute Ebert. Feather-like structures in positive streamers interpreted as electron avalanches. *Jpn. J. Appl. Phys.*, 50(8):08JA01, Aug 2011. doi:10.1143/jjap.50.08ja01.

[90] J. M. Meek. A theory of spark discharge. *Phys. Rev.*, 57(8):722–728, Apr 1940. doi:10.1103/physrev.57.722.

[91] Carolynne Montijn and Ute Ebert. Diffusion correction to the raether-meek criterion for the avalanche-to-streamer transition. *J. Phys. D: Appl. Phys.*, 39(14):2979–2992, Jul 2006. doi:10.1088/0022-3727/39/14/017.

[92] R. E. Robson, R. D. White, and Z. Lj. Petrović. Colloquium: Physically based fluid modeling of collisionally dominated low-temperature plasmas. *Rev. Mod. Phys.*, 77(4):1303–1320, Nov 2005. doi:10.1103/revmodphys.77.1303.

[93] S Dujko, A H Markosyan, R D White, and U Ebert. High-order fluid model for streamer discharges: I. derivation of model and transport data. *J. Phys. D: Appl. Phys.*, 46(47):475202, Oct 2013. doi:10.1088/0022-3727/46/47/475202.

[94] Ute Ebert and Wim van Saarloos. Front propagation into unstable states: universal algebraic convergence towards uniformly translating pulled fronts. *Physica D: Nonlinear Phenomena*, 146(1-4):1–99, Nov 2000. doi:10.1016/s0167-2789(00)00068-3.

[95] G. V. Naidis. Effects of nonlocality on the dynamics of streamers in positive corona discharges. *Technical Physics Letters*, 23(6):493–494, Jun 1997. doi:10.1134/1.1261717.

[96] Giovanni Lapenta. Particle rezoning for multidimensional kinetic particle-in-cell simulations. *Journal of Computational Physics*, 181(1):317 – 337, 2002. doi:10.1006/jcph.2002.7126.

[97] T. Bagdonat and U. Motschmann. 3D hybrid simulation code using curvilinear coordinates. *Journal of Computational Physics*, 183(2):470–485, Dec 2002. doi:10.1006/jcph.2002.7203.

[98] D.R. Welch, T.C. Genoni, R.E. Clark, and D.V. Rose. Adaptive particle management in a particle-in-cell code. *Journal of Computational Physics*, 227(1):143 – 155, 2007. doi:10.1016/j.jcp.2007.07.015.

[99] Ute Ebert and Willem Hundsdorfer. Ebert and Hundsdorfer Reply:. *Physical Review Letters*, 89(22), Nov 2002. doi:10.1103/physrevlett.89.229402.

[100] Ute Ebert, Wim van Saarloos, and Christiane Caroli. Propagation and structure of planar streamer fronts. *Phys. Rev. E*, 55(2):1530–1549, Feb 1997. doi:10.1103/physreve.55.1530.

[101] D. Dubrovin, S. Nijdam, E. M. van Veldhuizen, U. Ebert, Y. Yair, and C. Price. Sprite discharges on venus and jupiter-like planets: A laboratory investigation. *Journal of Geophysical Research*, 115, 2010. doi:10.1029/2009ja014851.

[102] Chao Li, Ute Ebert, and Willem Hundsdorfer. Simulated avalanche formation around streamers in an overvolted air gap. *IEEE Trans. Plasma Sci.*, 39(11):2256–2257, Nov 2011. doi:10.1109/tps.2011.2163528.

[103] S. Nijdam, K. Miermans, E. M. van Veldhuizen, and U. Ebert. A Peculiar Streamer Morphology Created by a Complex Voltage Pulse. *Plasma Science, IEEE Transactions on*, PP(99):1–2, 2011. doi:10.1109/TPS.2011.2158661.

[104] Jannis Teunissen and Ute Ebert. Controlling the weights of simulation particles: adaptive particle management using k-d trees. *Journal of Computational Physics*, 259:318–330, Feb 2014. doi:10.1016/j.jcp.2013.12.005.

[105] Charles K. Birdsall and A. Bruce. Langdon. *Plasma physics via computer simulation*. McGraw-Hill, New York, 1985. ISBN 0070053715.

[106] Giovanni Lapenta and Jeremiah U. Brackbill. Dynamic and selective control of the number of particles in kinetic plasma simulations. *Journal of Computational Physics*, 115(1):213 – 227, 1994. doi:10.1006/jcph.1994.1188.

[107] Giovanni Lapenta and J.U. Brackbill. Control of the number of particles in fluid and MHD particle in cell methods. *Computer Physics Communications*, 87(1-2):139 – 154, 1995. doi:10.1016/0010-4655(94)00180-A. Particle Simulation Methods.

[108] F. Assous, T. Pougeard Dulimbert, and J. Segré. A new method for coalescing particles in PIC codes. *Journal of Computational Physics*, 187(2): 550 – 571, 2003. doi:10.1016/S0021-9991(03)00124-4.

[109] G. Lapenta. Adaptive Multi-Dimensional Particle In Cell. *ArXiv e-prints*, June 2008. URL `http://arxiv.org/abs/0806.0830`.

[110] A. B. Sun, J. Teunissen, and U. Ebert. Why isolated streamer discharges hardly exist above the breakdown field in atmospheric air. *Geophys. Res. Lett.*, 40(10):2417–2422, May 2013. doi:10.1002/grl.50457.

[111] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Commun. ACM*, 18(9):509–517, September 1975. doi:10.1145/361002.361007.

[112] C.H. Shon, H.J. Lee, and J.K. Lee. Method to increase the simulation speed of particle-in-cell (PIC) code. *Computer Physics Communications*, 141(3):322 – 329, 2001. doi:10.1016/S0010-4655(01)00417-9.

[113] M. B. Kennel. KDTREE 2: Fortran 95 and C++ software to efficiently search for near neighbors in a multi-dimensional Euclidean space. *ArXiv Physics e-prints*, August 2004. URL `http://arxiv.org/abs/physics/0408067`.

[114] J.A. Bittencourt. *Fundamentals of Plasma Physics*. Springer-Verlag, New York, 2004.

[115] T. Yamamoto, K. Ramanathan, P. A. Lawless, D. S. Ensor, J. R. Newsome, N. Plaks, and G. H. Ramsey. Control of volatile organic compounds by an AC energized ferroelectric pellet reactor and a pulsed corona reactor. *Industry Applications, IEEE Transactions on*, 28(3):528–534, 1992. doi:10.1109/28.137430.

[116] E J M van Heesch, G J J Winands, and A J M Pemen. Evaluation of pulsed streamer corona experiments to determine the O* radical yield. *J. Phys. D: Appl. Phys.*, 41(23):234015, Nov 2008. doi:10.1088/0022-3727/41/23/234015.

[117] T. Huiskamp, S. J. Voeten, E. J. M. van Heesch, and A. J. M. Pemen. Design of a subnanosecond rise time, variable pulse duration, variable amplitude, repetitive, high-voltage pulse source. *IEEE Trans. Plasma Sci.*, 42 (1):127–137, 2014. doi:10.1109/tps.2013.2293841.

[118] E M van Veldhuizen and W R Rutgers. Inception behaviour of pulsed positive corona in several gases. *J. Phys. D: Appl. Phys.*, 36(21):2692–2696, Oct 2003. doi:10.1088/0022-3727/36/21/015.

[119] C. K. Birdsall. Particle-in-cell charged-particle simulations, plus Monte Carlo collisions with neutral atoms, PIC-MCC. *Plasma Science, IEEE Transactions on*, 19(2):65–85, 1991. doi:10.1109/27.106800.

[120] S Nijdam, E Takahashi, J Teunissen, and U Ebert. Streamer discharges can move perpendicularly to the electric field. *New Journal of Physics*, 16 (10):103038, Oct 2014. doi:10.1088/1367-2630/16/10/103038.

[121] Anbang Sun, Jannis Teunissen, and Ute Ebert. The inception of pulsed discharges in air: simulations in background fields above and below breakdown. *J. Phys. D: Appl. Phys.*, 47(44):445205, Oct 2014. doi:10.1088/0022-3727/47/44/445205.

[122] Jannis Teunissen, Anbang Sun, and Ute Ebert. A time scale for electrical screening in pulsed gas discharges. *J. Phys. D: Appl. Phys.*, 47(36):365203, Aug 2014. doi:10.1088/0022-3727/47/36/365203.

[123] Katsuhisa Koura. Null-collision technique in the direct-simulation monte carlo method. *Physics of Fluids*, 29(11):3509, 1986. doi:10.1063/1.865826.

[124] Pitchford L.C. and Boeuf J.P. Siglo database, retrieved april 2013. URL `www.lxcat.net`.

[125] A. Okhrimovskyy, A. Bogaerts, and R. Gijbels. Electron anisotropic scattering in gases: A formula for Monte Carlo simulations. *Phys. Rev. E*, 65 (3), Feb 2002. doi:10.1103/physreve.65.037402.

[126] LXCat Team. The lxcat project. URL `http://www.lxcat.net`. [Online; accessed 26-July-2015].

[127] O Zatsarinny and K Bartschat. B-spline Breit-Pauli R-matrix calculations for electron collisions with argon atoms. *Journal of Physics B: Atomic, Molecular and Optical Physics*, 37(23):4693–4706, Nov 2004. doi:10.1088/0953-4075/37/23/010.

[128] M. Allan, O. Zatsarinny, and K. Bartschat. Near-threshold absolute angle-differential cross sections for electron-impact excitation of argon and xenon. *Physical Review A*, 74(3), Sep 2006. doi:10.1103/physreva.74.030701.

[129] Loup Verlet. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Phys. Rev.*, 159:98–103, 1967. doi:10.1103/PhysRev.159.98.

[130] Michael S. Barnes, Tina J. Cotler, and Michael E. Elta. Large-signal time-domain modeling of low-pressure rf glow discharges. *Journal of Applied Physics*, 61(1):81, 1987. doi:10.1063/1.338804.

[131] Jeroen Wackers. A nested-grid direct poisson solver for concentrated source terms. *Journal of Computational and Applied Mathematics*, 180(1):1–12, Aug 2005. doi:10.1016/j.cam.2004.09.054.

[132] P. M. Ricker. A direct multigrid poisson solver for oct-tree adaptive meshes. *Astrophys J. Suppl. S.*, 176(1):293–300, May 2008. doi:10.1086/526425.

[133] Jingfang Huang and Leslie Greengard. A fast direct solver for elliptic partial differential equations on adaptively refined meshes. *SIAM Journal on Scientific Computing*, 21(4):1551–1566, Jan 1999. doi:10.1137/s1064827598346235.

[134] Timm H. Teich. Emission gasionisierender strahlung aus elektronenlawinen. *Zeitschrift fur Physik*, 199(4):378–394, Aug 1967. doi:10.1007/bf01332287.

[135] Hank Childs, Eric Brugger, Brad Whitlock, Jeremy Meredith, Sean Ahern, David Pugmire, Kathleen Biagas, Mark Miller, Cyrus Harrison, Gunther H. Weber, Hari Krishnan, Thomas Fogal, Allen Sanderson, Christoph Garth, E. Wes Bethel, David Camp, Oliver Rübel, Marc Durant, Jean M. Favre, and Paul Navrátil. VisIt: An End-User Tool For Visualizing and Analyzing Very Large Data. In *High Performance Visualization–Enabling Extreme-Scale Scientific Insight*, pages 357–372. Chapman and Hall/CRC, Oct 2012.

[136] P Tardiveau, N Moreau, S Bentaleb, C Postel, and S Pasquiers. Diffuse mode and diffuse-to-filamentary transition in a high pressure nanosecond scale corona discharge under high voltage. *J. Phys. D: Appl. Phys.*, 42(17): 175202, Aug 2009. doi:10.1088/0022-3727/42/17/175202.

[137] A. Pedersen. On the electrical breakdown of gaseous dielectrics-an engineering approach. *Conference on Electrical Insulation and Dielectric Phenomena*, 1989. doi:10.1109/ceidp.1989.69523.

[138] Gianne Derks, Ute Ebert, and Bernard Meulenbroek. Laplacian instability of planar streamer ionization fronts – an example of pulled front analysis. *J Nonlinear Sci*, 18(5):551–590, Jun 2008. doi:10.1007/s00332-008-9023-0.

[139] Ute Ebert, Bernard Meulenbroek, and Lothar Schäfer. Convective stabilization of a laplacian moving boundary problem with kinetic undercooling. *SIAM Journal on Applied Mathematics*, 68(1):292–310, Jan 2007. doi:10.1137/070683908.

[140] A. Luque and F. J. Gordillo-Vázquez. Mesospheric electric breakdown and delayed sprite ignition caused by electron detachment. *Nature Geoscience*, 5(1):22–25, Nov 2011. doi:10.1038/ngeo1314.

[141] V. I. Ermakov, G. A. Bazilevskaya, P. E. Pokrevsky, and Y. I. Stozhkov. Ion balance equation in the atmosphere. *Journal of Geophysical Research*, 102(D19):23413, 1997. doi:10.1029/97jd01388.

[142] E. O. Hulburt. Atmospheric ionization by cosmic radiation. *Phys. Rev.*, 37(1):1–8, Jan 1931. doi:10.1103/physrev.37.1.

[143] I.G. Usoskin, O.G. Gladysheva, and G.A. Kovaltsov. Cosmic ray-induced ionization in the atmosphere: spatial and temporal changes. *Journal of Atmospheric and Solar-Terrestrial Physics*, 66(18):1791–1796, Dec 2004. doi:10.1016/j.jastp.2004.07.037.

[144] A Bourdon, Z Bonaventura, and S Celestin. Influence of the pre-ionization background and simulation of the optical emission of a streamer discharge in preheated air at atmospheric pressure between two point electrodes. *Plasma Sources Sci. Technol.*, 19(3):034012, May 2010. doi:10.1088/0963-0252/19/3/034012.

[145] I A Kossyi, A Yu Kostinsky, A A Matveyev, and V P Silakov. Kinetic scheme of the non-equilibrium discharge in nitrogen-oxygen mixtures. *Plasma Sources Sci. Technol.*, 1(3):207–220, Aug 1992. doi:10.1088/0963-0252/1/3/011.

[146] F. J. Gordillo-Vázquez and A. Luque. Electrical conductivity in sprite streamer channels. *Geophys. Res. Lett.*, 37(16), Aug 2010. doi:10.1029/2010gl044349.

[147] Ningyu Liu. Multiple ion species fluid modeling of sprite halos and the role of electron detachment of O^- in their dynamics. *Journal of Geophysical Research*, 117(A3), 2012. doi:10.1029/2011ja017062.

[148] Jianqi Qin, Sebastien Celestin, and Victor P. Pasko. On the inception of streamers from sprite halo events produced by lightning discharges with

positive and negative polarity. *Journal of Geophysical Research*, 116(A6), 2011. doi:10.1029/2010ja016366.

[149] Alejandro Luque and Ute Ebert. Emergence of sprite streamers from screening-ionization waves in the lower ionosphere. *Nature Geoscience*, 2(11):757–760, Nov 2009. doi:10.1038/ngeo662.

[150] H C Stenbaek-Nielsen and M G McHarg. High time-resolution sprite imaging: observations and implications. *J. Phys. D: Appl. Phys.*, 41(23):234009, Nov 2008. doi:10.1088/0022-3727/41/23/234009.

[151] Ningyu Liu, Burcu Kosar, Samaneh Sadighi, Joseph R. Dwyer, and Hamid K. Rassoul. Formation of streamer discharges from an isolated ionization column at subbreakdown conditions. *Physical Review Letters*, 109(2), Jul 2012. doi:10.1103/physrevlett.109.025002.

[152] S.T. Surzhikov. *Computational Physics of Electric Discharges in Gas Flows*. De Gruyter Studies in Mathematical Physics. De Gruyter, 2012. ISBN 9783110270419.

[153] Z.M. Raspopovic, S. Sakadzic, S.A. Bzenic, and Z.Lj. Petrovic. Benchmark calculations for Monte Carlo simulations of electron transport. *IEEE Trans. Plasma Sci.*, 27(5):1241–1248, 1999. doi:10.1109/27.799799.

[154] A. Jay Palmer. A physical model on the initiation of atmospheric-pressure glow discharges. *Appl. Phys. Lett.*, 25(3):138, 1974. doi:10.1063/1.1655412.

[155] Jeffrey I. Levatter and Shao-Chi Lin. Necessary conditions for the homogeneous formation of pulsed avalanche discharges at high gas pressures. *Journal of Applied Physics*, 51(1):210, 1980. doi:10.1063/1.327412.

[156] G. Herziger, R. Wollermann-Windgasse, and K. H. Banse. On the homogeneization of transverse gas discharges by preionization. *Appl. Phys.*, 24 (3):267–272, Mar 1981. doi:10.1007/bf00899768.

[157] W.J.M. Samaranayake, Y. Miyahara, T. Namihira, S. Katsuki, T. Sakugawa, R. Hackam, and H. Akiyama. Pulsed streamer discharge characteristics of ozone production in dry air. *IEEE Transactions on Dielectrics and Electrical Insulation*, 7(2):254–260, Apr 2000. doi:10.1109/94.841818.

[158] Ryo Ono and Tetsuji Oda. Ozone production process in pulsed positive dielectric barrier discharge. *J. Phys. D: Appl. Phys.*, 40(1):176–182, Dec 2006. doi:10.1088/0022-3727/40/1/011.

[159] L. R. Grabowski, E. M. van Veldhuizen, A. J. M. Pemen, and W. R. Rutgers. Corona above water reactor for systematic study of aqueous phenol degradation. *Plasma Chemistry and Plasma Processing*, 26(1):3–17, Feb 2006. doi:10.1007/s11090-005-8721-8.

[160] G.J.J. Winands, K. Yan, A.J.M. Pemen, S.A. Nair, Z. Liu, and E.J.M. Van Heesch. An industrial streamer corona plasma system for gas cleaning. *IEEE Trans. Plasma Sci.*, 34(5):2426–2433, Oct 2006. doi:10.1109/tps.2006.881278.

[161] S M Starikovskaia. Plasma-assisted ignition and combustion: nanosecond discharges and development of kinetic mechanisms. *J. Phys. D: Appl. Phys.*, 47(35):353001, Aug 2014. doi:10.1088/0022-3727/47/35/353001.

[162] Jingbo Li and Steven Cummer. Relationship between sprite streamer behavior and lightning-driven electric fields. *Journal of Geophysical Research*, 117(A1), 2012. doi:10.1029/2011ja016843.

[163] Won J Yi and P F Williams. Experimental study of streamers in pure N_2 and N_2/O_2 mixtures and a ≈ 13 cm gap. *J. Phys. D: Appl. Phys.*, 35(3): 205–218, Jan 2002. doi:10.1088/0022-3727/35/3/308.

[164] F. Hegeler and H. Akiyama. Spatial and temporal distributions of ozone after a wire-to-plate streamer discharge. *IEEE Trans. Plasma Sci.*, 25(5): 1158–1165, 1997. doi:10.1109/27.649640.

[165] Sebastien Celestin, Zdenek Bonaventura, Barbar Zeghondy, Anne Bourdon, and Pierre Ségur. The use of the ghost fluid method for Poisson's equation to simulate streamer propagation in point-to-plane and point-to-point geometries. *J. Phys. D: Appl. Phys.*, 42(6):065203, Feb 2009. doi:10.1088/0022-3727/42/6/065203.

[166] Fabien Tholin and Anne Bourdon. Simulation of the stable "quasi-periodic" glow regime of a nanosecond repetitively pulsed discharge in air at atmospheric pressure. *Plasma Sources Sci. Technol.*, 22(4):045014, Jul 2013. doi:10.1088/0963-0252/22/4/045014.

[167] Yoav Yair, Yukihiro Takahashi, Roy Yaniv, Ute Ebert, and Yukihiro Goto. A study of the possibility of sprites in the atmospheres of other planets. *Journal of Geophysical Research*, 114(E9), 2009. doi:10.1029/2008je003311.

[168] A. Luque, U. Ebert, and W. Hundsdorfer. Interaction of streamer discharges in air and other oxygen-nitrogen mixtures. *Physical Review Letters*, 101(7), Aug 2008. doi:10.1103/physrevlett.101.075005.

[169] N L Aleksandrov and E M Bazelyan. Simulation of long-streamer propagation in air at atmospheric pressure. *J. Phys. D: Appl. Phys.*, 29(3): 740–752, Mar 1996. doi:10.1088/0022-3727/29/3/035.

[170] George V. Naidis. Simulation of a single streamer traveling along two counterpropagating helium jets in ambient air. *IEEE Trans. Plasma Sci.*, 40(11):2866–2869, Nov 2012. doi:10.1109/tps.2012.2215311.

[171] N.Yu. Babaeva and G.V. Naidis. Two-dimensional modeling of positive streamer propagation in flue gases in sphere-plane gaps. *IEEE Trans. Plasma Sci.*, 26(1):41–45, 1998. doi:10.1109/27.659531.

[172] Birdsall C.K. and Langdon A.B. *Plasma physics via computer simulation*. Bristol: Institute of Physics Publishing, 1991.

[173] K. Nanbu. Probability theory of electron-molecule, ion-molecule, molecule-molecule, and coulomb collisions for particle modeling of materials processing plasmas and cases. *IEEE Trans. Plasma Sci.*, 28(3):971–990, Jun 2000. doi:10.1109/27.887765.

[174] A. Luque and F. J. Gordillo-Vazquez. Sprite beads originating from inhomogeneities in the mesospheric electron density. *Geophys. Res. Lett.*, 38 (4), Feb 2011. doi:10.1029/2010gl046403.

[175] S. Pancheshnyi. Personal communication, 2009.

[176] Gregory H. Wannier. Motion of gaseous ions in strong electric fields. *Bell System Technical Journal*, 32(1):170–254, Jan 1953. doi:10.1002/j.1538-7305.1953.tb01426.x.

[177] F J Gordillo-Vázquez. Air plasma kinetics under the influence of sprites. *J. Phys. D: Appl. Phys.*, 41(23):234016, Nov 2008. doi:10.1088/0022-3727/41/23/234016.

[178] N. A. Popov. Evolution of the negative ion composition in the afterglow of a streamer discharge in air. *Plasma Phys. Rep.*, 36(9):812–818, Sep 2010. doi:10.1134/s1063780x10090084.

[179] J Koppitz. Nitrogen discharges of large cross section at high overvoltage in a homogeneous field. *J. Phys. D: Appl. Phys.*, 6(12):1494–1502, Aug 1973. doi:10.1088/0022-3727/6/12/312.

[180] E.E. Kunhardt. Generation of large-volume, atmospheric-pressure, nonequilibrium plasmas. *IEEE Trans. Plasma Sci.*, 28(1):189–200, 2000. doi:10.1109/27.842901.

[181] M Ghasemi, P Olszewski, J W Bradley, and J L Walsh. Interaction of multiple plasma plumes in an atmospheric pressure plasma jet array. *J. Phys. D: Appl. Phys.*, 46(5):052001, Jan 2013. doi:10.1088/0022-3727/46/5/052001.

[182] S. Nijdam, C G C Geurts, E M van Veldhuizen, and U. Ebert. Reconnection and merging of positive streamers in air. *J. Phys. D: Appl. Phys.*, 42(4): 045201, 2009. doi:10.1088/0022-3727/42/4/045201.

[183] D. Z. Pai, D. A. Lacoste, and C. O. Laux. Transitions between corona, glow, and spark regimes of nanosecond repetitively pulsed discharges in air at atmospheric pressure. *J. Appl. Phys.*, 107(9):093303, 2010. doi:10.1063/1.3309758.

[184] T Shao, G S Sun, P Yan, J Wang, W Q Yuan, Y H Sun, and S C Zhang. An experimental investigation of repetitive nanosecond-pulse breakdown in air. *J. Phys. D: Appl. Phys.*, 39(10):2192–2197, MAY 21 2006. doi:10.1088/0022-3727/39/10/030.

[185] S. Nijdam, E. Takahashi, A. Markosyan, and U. Ebert. Investigation of positive streamers by double pulse experiments, effects of repetition rate and gas mixture. *Plasma Sources Sci. T.*, 23:025008, 2014. doi:10.1088/0963-0252/23/2/025008.

[186] S. Nijdam, J. S. Moerman, T. M. P. Briels, E. M. van Veldhuizen, and U. Ebert. Stereo-photography of streamers in air. *Appl. Phys. Lett.*, 92: 101502, 2008. doi:10.1063/1.2894195.

[187] Takashi Fujii, Megumu Miki, Naohiko Goto, Alexei Zhidkov, Tetsuo Fukuchi, Yuji Oishi, and Koshichi Nemoto. Leader effects on femtosecond-laser-filament-triggered discharges. *Phys. Plasmas*, 15(1):013107, 2008. doi:10.1063/1.2830647.

[188] S. B. Leonov, A. A. Firsov, M. A. Shurupov, J. B. Michael, M. N. Shneider, R. B. Miles, and N. A. Popov. Femtosecond laser guiding of a high-voltage discharge and the restoration of dielectric strength in air and nitrogen. *Phys. Plasmas*, 19(12):123502, 2012. doi:10.1063/1.4769261.

[189] V D Zvorykin, A O Levchenko, and N N Ustinovskii. Control of extended high-voltage electric discharges in atmospheric air by UV KrF-laser radiation. *Quantum Electron.*, 41(3):227–233, Mar 2011. doi:10.1070/qe2011v041n03abeh014477.

[190] Erick Wong. Name of the generalization of quadtree and octree? URL `http://math.stackexchange.com/questions/644032/name-of-the-generalization-of-quadtree-and-octree`. [Online; accessed 17-July-2015].

[191] Richard D. Hornung, Andrew M. Wissink, and Scott R. Kohn. Managing complex data and geometry in parallel structured amr applications. *Engineering with Computers*, 22(3-4):181–195, Aug 2006. doi:10.1007/s00366-006-0038-6.

[192] Rahul S. Sampath, Santi S. Adavani, Hari Sundar, Ilya Lashuk, and George Biros. Dendro: Parallel algorithms for multigrid and amr methods

on 2:1 balanced octrees. *2008 SC - International Conference for High Performance Computing, Networking, Storage and Analysis*, Nov 2008. doi:10.1109/sc.2008.5218558.

[193] Tobias Weinzierl. *A Framework for Parallel PDE Solvers on Multiscale Adaptive Cartesian Grids*. Technische Universität München, 2009. ISBN 9783868531466.

[194] Stéphane Popinet. Gerris: a tree-based adaptive solver for the incompressible euler equations in complex geometries. *Journal of Computational Physics*, 190(2):572–600, Sep 2003. doi:10.1016/s0021-9991(03)00298-5.

[195] R. Teyssier. Cosmological hydrodynamics with adaptive mesh refinement. *A&A*, 385(1):337–364, Apr 2002. doi:10.1051/0004-6361:20011817.

[196] Donna Calhoun. Adaptive mesh refinement resources. URL http://math.boisestate.edu/~calhoun/www_personal/research/amr_software/index.html. [Online; accessed 17-July-2015].

[197] John Bell. Block Structured AMR Short Course: Lecture 1. URL http://math.boisestate.edu/~calhoun/www_personal/research/amr_software/jbb_shortcourse/. [Online; accessed 17-July-2015].

[198] Kevin Olson. Alpha version of paramesh multigrid support. URL http://www.physics.drexel.edu/~olson/paramesh-doc/Users_manual/multigrid.html. [Online; accessed 17-July-2015].

[199] G.M. Morton. A computer oriented geodetic data base; and a new technique in file sequencing. *IBM Research Report*, 1966.

[200] Kitware. Paraview. URL http://www.paraview.org/. [Online; accessed 22-July-2015].

[201] Wolfgang Hackbusch. Multi-grid methods and applications. *Springer Series in Computational Mathematics*, 1985. doi:10.1007/978-3-662-02427-0.

[202] D. Bai and A. Brandt. Local mesh refinement multilevel techniques. *SIAM Journal on Scientific and Statistical Computing*, 8(2):109–134, Mar 1987. doi:10.1137/0908025.

[203] G J M Hagelaar and L C Pitchford. Solving the boltzmann equation to obtain electron transport coefficients and rate coefficients for fluid models. *Plasma Sources Science and Technology*, 14(4):722–733, Oct 2005. doi:10.1088/0963-0252/14/4/011.

[204] Saša Dujko, Ute Ebert, Ronald D. White, and Zoran Lj. Petrović. Boltzmann equation analysis of electron transport in a N_2-O_2 streamer discharge. *Japanese Journal of Applied Physics*, 50(8):08JC01, Aug 2011. doi:10.1143/jjap.50.08jc01.

[205] George Marsaglia. Choosing a point from the surface of a sphere. *The Annals of Mathematical Statistics*, 43(2):645–646, Apr 1972. doi:10.1214/aoms/1177692644.

[206] John W. Eaton et al. GNU Octave. URL `http://www.octave.org`.

[207] W. A. Stein et al. *Sage Mathematics Software*. The Sage Development Team. URL `http://www.sagemath.org`.

[208] David Goldberg. What every computer scientist should know about floating point arithmetic. *ACM Computing Surveys*, 23(1):5–48, 1991.

[209] Jean-Michel Muller, Nicolas Brisebarre, Florent de Dinechin, Claude-Pierre Jeannerod, Vincent Lefèvre, Guillaume Melquiond, Nathalie Revol, Damien Stehlé, and Serge Torres. *Handbook of Floating-Point Arithmetic*. Birkhäuser Boston, 2010.

Acknowledgments

During my PhD I have often thought: 'work' cannot be this good later. Receiving a salary for studying topics at your own pace feels like a big privilege. Add to that the interesting and friendly colleagues, the considerable freedom in choosing research topics, the opportunities for attending meetings and conferences, and all the other things that I now forget.

So who should be acknowledged for this privilege? I think it was mostly luck: luck that there is a universe, luck that our earth exists and luck that eventually humans came to be – these peculiar biochemical machines with a feeling of identity. But that is not where the luck ends, because I could have been born for a life of hardship. Instead I was born the Netherlands, in a warm family.

Besides all that luck, I would like to acknowledge Ute Ebert: I hope that some day I can transfer the advice, support and motivation that you have given me to my own students.

Curriculum Vitae

Herman Jan 'Jannis' Teunissen was born in Amsterdam, The Netherlands, on July 8$^{\text{th}}$, 1987. After finishing his secondary education at the Barlaeus Gymnasium in Amsterdam in 2005, he received a bachelor's degree in Physics & Astronomy (*cum laude*) at the University of Amsterdam in 2008. Later that year, he enrolled in a master's program in theoretical physics, but after a few months he decided to pursue a master's degree in computational science instead. In 2011, he received this degree (*cum laude*) at the University of Amsterdam. His master's research project was carried out in the Multiscale Dynamics group at the CWI (Centrum Wiskunde & Informatica) in Amsterdam. He started a PhD project in the same group in 2011, the results of which are presented in this thesis.

List of publications

1. C. Li, J. Teunissen, M. Nool, W. Hundsdorfer, and U. Ebert. A comparison of 3D particle, fluid and hybrid simulations for negative streamers. *Plasma Sources Sci. Technol.*, 21(5):055019, 2012. (Chapter 3)
2. J. Teunissen and U. Ebert. Controlling the weights of simulation particles: adaptive particle management using k-d trees. *J. Comp. Phys.*, 259:318-330, 2014. (Chapter 4)
3. A.B. Sun, J. Teunissen, and U. Ebert. Why isolated streamer discharges hardly exist above the breakdown field in atmospheric air. *Geophys. Res. Lett.*, 40(10):2417-2422, 2013. (Chapter 6)
4. J. Teunissen, A.B. Sun, and U. Ebert. A time scale for electrical screening in pulsed gas discharges. *J. Phys. D: Appl. Phys.*, 47(36):365203, Aug 2014. (Chapter 7)
5. A.B. Sun, J. Teunissen, and U. Ebert. The inception of pulsed discharges in air: simulations in background fields above and below breakdown. *J. Phys. D: Appl. Phys.*, 47(44):445205, 2014. (Chapter 8)
6. S. Nijdam, E. Takahashi, J. Teunissen, and U. Ebert. Streamer discharges can move perpendicularly to the electric field. *New Journal of Physics*, 16(10):103038, 2014. (Chapter 9)

7. A.B. Sun, J. Teunissen, and U. Ebert. 3-D particle modeling of positive streamer inception from a needle electrode in supercritical nitrogen. *IEEE Trans. Plasma Sci.*, 42(10):2416-2417, 2014.
8. A. Dubinova, J. Teunissen, and U. Ebert. Propagation of a positive streamer toward a dielectric tip in pure nitrogen and in air under voltage pulses with subnanosecond rise time. *IEEE Trans. Plasma Sci.*, 42(10):2392-2393, 2014.

Not yet published

1. *Accepted* A.H. Markosyan, J. Teunissen, S. Dujko, U. Ebert. Comparing plasma fluid models of different order for 1D streamer ionization fronts. *Plasma Sources Sci. Technol.*
2. *Submitted* N. Mascini, J. Teunissen, R. Noorlag, S. Willems, R. Heeren. Predicting head and neck cancer metastasis and disease-specific survival from MALDI-MSI data: feasible or not? *Anal. and Bioanal. Chemistry.*
3. *In prep.* J. Teunissen, U. Ebert. Simulating the inception of nanosecond pulsed discharges in nitrogen/oxygen mixtures. (Chapter 5)
4. *In prep.* J. Teunissen, U. Ebert. Afivo: a framework for finite volume simulations on adaptively refined quadtree and octree grids. (Chapter 10)
5. *In prep.* J. Teunissen, U. Ebert. A Monte Carlo approach for photoionization in discharge simulations. (Chapter 11)

Summary

3D Simulations and Analysis of Pulsed Discharges

There exists a wide variety of electrical discharges in nature, in the lab and in technological applications. Well known examples are lightning (nature), Tesla coils (lab) and fluorescent lamps (technology). In this thesis, the focus is on *nanosecond-pulsed* discharges, and in particular on *streamers*. Streamers are rapidly growing ionized channels, which are often the precursors for other discharges. They for example occur in corona discharges used for gas processing and plasma medicine, or high above thunderclouds in the form of *sprites*.

In nanosecond-pulsed discharges there are steep gradients in the electron and ion density, which generate thin space charge layers. These charge layers modify the electric field, and because the electric field largely determines the growth of such discharges, their propagation is strongly non-linear. This makes it for example difficult to predict how fast streamers will grow or how their radius will change in time. To help answer such questions, computer simulations are valuable: one can control what physics to include, there is complete information about the system, and simulations can be performed under conditions that are experimentally hard to realize. However, simulating nanosecond pulsed discharges can be quite challenging. Because these discharges are transient phenomena that typically lack cylindrical symmetry, computationally costly three-dimensional time-dependent simulations are often required. This is illustrated by the results from chapter 3 and chapters 5 to 9. For future studies on streamer branching or the interaction between multiple streamers, three-dimensional simulations will also be required.

In the chapters of this thesis a suite of numerical approaches and physical predictions is developed, as summarized below. In chapter 3, four models are compared for the simulation of a negative streamer: a *particle* model, two *fluid* models and a *hybrid* model, which couples the particle and fluid approach in space. Particle models include more physics, but they are computationally expensive. The hybrid model can combine advantages of both models, but its implementation is more challenging. The comparison demonstrated that particle, fluid and hybrid models can be in good agreement, at least up to the moment of front destabilization. Without *adaptive mesh refinement* three-dimensional streamer simulations are limited to small domains and unrealistic conditions,

however.

To speed up particle simulations, the weights of simulation particles can be changed adaptively by merging and splitting them. In chapter 4, we demonstrate that a k-d tree can be used to efficiently search for pairs of particles that are close in both position and velocity. Using random numbers, such pairs can be merged so that energy and momentum are *on average* preserved. In chapter 5, the implementation of a three-dimensional particle-in-cell code is described. This code includes adaptive mesh refinement, adaptive particle weights, parallelized particle routines and the possibility of including a needle electrode. We show how the formation and destabilization of an 'inception cloud' around an electrode tip depends on the nitrogen/oxygen ratio, which affects the non-local photoionization density.

In chapters 6 and 8, the formation of discharges far from electrodes is investigated. If the electric field in such a region rapidly increases to a value above breakdown, then background ionization plays an important role: due to electron detachment, electron avalanches start to grow in the whole overvolted region. After the 'ionization screening time', these avalanches together screen the electric field. The formation of isolated double-headed streamers, observed in several previous studies, is therefore unlikely. In chapter 7, an analytical approximation for the ionization screening time is derived. This time scale is a generalization of the Maxwell time that takes into account electron impact ionization. Predicted screening times are compared with one- and three-dimensional simulations, and a simple criterion for the homogeneity of overvolted discharges is given.

In chapter 9, experiments and simulations demonstrate that the growth of positive streamers is not only determined by the electric field. With weak pre-ionization (having a negligible space charge effect), such streamers can be *guided* so that they move almost perpendicular to the background electric field. Guiding only happens when the pre-ionized region contains a significantly higher electron density than the bulk gas. This is the reason we observe guiding in nitrogen but not in air, in which more non-local photoionization is produced.

Even using adaptive particle weights, particle models are computationally quite expensive compared to fluid models. In chapter 10 we present Afivo, a framework suitable for modest-scale parallel computations on adaptively refined quadtree/octree grids. Afivo can be a simpler alternative for some of the already existing frameworks aimed at large scale parallel computations. For discharge simulations, one of the major computational challenges is Poisson's equation, for which geometric multigrid has been implemented in Afivo. Examples of discharge simulations in two and three dimensions are presented.

Photoionization plays an important role in many discharges, but including this process in simulations can be challenging. In chapter 11, a Monte Carlo approach for photoionization is presented, suitable for plasma fluid models on adaptively refined grids. By absorbing photons on different grid levels, the photoionization profile can efficiently be approximated.