Chemical Kinetics-Assisted, Path-Based Smoke Simulation*

Yoojin Jang†, Insung Ihm‡

† Macrograph

JNS Bldg 23-5, Nonhyun-dong, Gangnam-gu, Seoul, Korea

Tel. +82-2-2142-7201, Fax. +82-2-2142-7299

Email: contact.yj@gmail.com

‡ Corresponding author

Department of Computer Science and Engineering

Sogang University

1 Shinsu-dong, Mapo-gu, Seoul, Korea

Tel. +82-2-705-8493, Fax. +82-2-704-8273

Email: ihm@sogang.ac.kr

* **Accepted for the CASA2009 special issue**

Abstract

Despite recent successes in physics-based fluid animation, generating desired fluid flow with intuitive control of its motion still remains a challenging problem in the special effects industry. In this paper, we propose a novel approach for path-based smoke simulation that explores the theory of chemical kinetics in an aim to provide a useful animation tool. By describing intended smoke effects through chemical reaction equations and adjusting their parameters, our method allows to easily create various interesting smoke effects that were often hard to get with previous techniques. To demonstrate the effectiveness of the presented animation framework, we describe several examples of path-based smoke animations, generated with easily understandable reaction equations and control parameters.

Keywords: Fluid animation, path-based smoke simulation, chemical kinetics, particles, animation control.

Introduction

Background and our contribution

Physically-based fluid simulation has recently become an important element in reproduction of realistic fluid effects in the special effects industry. Unfortunately, synthesizing fluid flows as intended still remains a challenging problem because the employed physics-based equations are sensitive to the choice of parameter values, and hence often produce unpredictable results. It is known that the nonlinearity and complexity of the routinely used partial differential equations like the Navier-Stokes equations, makes it difficult to control the simulation process intuitively.

Among the recently introduced fluid control methods, one effective approach is the path-based smoke simulation in which a natural motion of smoke following given path in three-dimensional space is generated with user controls (for instance, refer to the recent works [1, 2]). By interactively designing a space curve that models the trajectory of smoke to follow, animators can create various desired smoke effects easily.

In this paper, we continue such an effort to provide them another effective way of synthesizing fluid effects with intuitive control. Specifically, we present a novel, path-based smoke simulation framework which, coupled with a chemical reaction mechanism, allows to create a wide range of interesting smoke animation effects that were hard to achieve with previous techniques. In our simulation scheme, we ensure that smoke follows a specified trajectory through space by building adequate velocity fields automatically. Rather than

constructing target velocity fields explicitly as often done previously when the fluid equations are solved, we generate the fields as intended by adding a driving force and correcting them through a divergence parameter, both easily computed from the specified path geometry. Then, a physics-based simulation pipeline is augmented through a chemical reaction mechanism in an effort to provide more flexibility in the design of smoke animation. By defining desired effects by chemical reaction equations, grammar-like rules, that describe how smoke evolves, animators can creatively generate a wide range of realistic or artificial smoke effects. Furthermore, by adjusting the relevant parameters of the reaction equations, they can intuitively control the appearance and behavior of smoke flows in detail. To show the effectiveness of our method, we discuss several examples of path-based smoke simulation.

Related work

There have been several studies on fluid controls in the physically-based fluid simulation. Foster et al. proposed a simulation method based on an embedded controller through which boundary conditions as well as velocity and pressure fields were effectively controlled [3]. In [4], Schpok et al. provided a fluid modeling tool that manipulated high-level behaviors through an intuitive set of extracted simulation features.

In [5], Treuille et al. proposed to use a series of target density fields for keyframe control of smoke simulation. A gradient-based nonlinear optimization technique was applied to

fluid control by McNamara et al. [6], where the presented adjoint method calculated derivatives efficiently even for large simulation problems. Fattal et al. controlled smoke animation by adding a driving force term and a smoke gathering term to the standard flow equations [7]. On the other hand, Hong et al. generated control force from the differentials of geometric potentials [8]. In [9], Shi et al. enforced incompressibility on closed surfaces of objects by using a discrete vector field decomposition algorithm. In [10], the same authors applied a feedback force field and the gradient field of a potential function for liquid control.

In [11], Foster et al. proposed to use control particles taken from given path for effective liquid control. Thürey et al. also used particles in their detail-preserving method for controlling liquids [12]. In [1], Kim et al. presented a path-based linear feedback control method that accurately forced movement along given trajectory, possibly with high curvature. In [2], Angelidis et al. applied a vortex-based control method to smoke simulation where the motion of smoke was controlled by high level tools, such as animated current curves, attractors, and tornadoes. A divergence constraint was used to control the volume of bubbles to compensate any undesired volume loss or gain occurred during a liquid simulation [13]. Recently, Dobashi et al. presented a cloud simulation method that automatically adjusted simulation parameters to generate clouds forming user-specified shapes [14].

Chemical reaction phenomena, like fire and flame, explosion, and catalysis, are abundant in the real world, and are important elements that are routinely generated in the special effects industry. In order to reproduce such reactive fluids, Ihm et al. explored the theory of chemical kinetics for effectively simulating reactive gases containing multiple reacting

species [15]. While generated several interesting fluid effects of reacting gases, the method often required a substantial demand for memory space, especially when a complex reaction mechanism involving many chemical species was simulated. In an attempt to ease the computational overhead, Kang et al. presented a hybrid simulation method, where both 'Eulerian' grids and 'Lagrangian' particles were exploited for efficient modeling of chemical reactions [16].

The proposed smoke simulation pipeline

Figure 1 illustrates the computation pipeline of our chemical kinetics-assisted, path-based smoke simulation method whose basic idea is straightforward. First, given a trajectory to be followed by smoke through three-dimensional space, a driving force is exerted on the path through its tangential direction (*Add driving force*). For this, the given path is discretized into a sequence of *path* particles at which proper driving force is generated automatically. A careless continual application of the driving force, however, may accumulate it too excessively around curved regions with high curvature, forcing smoke deviate from the path. To prevent this problem, we correct the velocity field in the projection stage by adjusting the divergence constraint so that the smoke constricts towards the path curve appropriately (*Adjust divergence*).

In addition to the path particles, we employ *chemical*, *vortex*, and *soot* particles to simulate chemical reactions through which the flow of smoke is controlled intuitively (*Apply*

chemical kinetics), as is the main contribution of this paper. The remaining computation stages are similar to those implemented in the previous works, for instance, in [17, 18].

Addition of driving force

In order to effectively reflect a driving force through given trajectory in the *Update velocity* stage, where velocity field is discretized in volume space, we represent the curve in the form of a series of path particles. At each path particle $\mathbf{p}$, we store local geometric properties of the trajectory such as its position $\mathbf{x_P}$, path vector $\mathbf{v_p}$ to the next particle, curvature $\kappa_{\mathbf{p}}$, and radius $r_{\mathbf{p}}$ of region of influence (ROI) (see Figure 2).

Then, in the beginning of each time step, a driving force is added at each grid point $\mathbf{x}_{ijk}$ that resides in the ROI of path particles $\mathbf{p}$ as follows (*Add driving force*):

$$\mathbf{f}(\mathbf{x}_{ijk}) = \sum_{\mathbf{x}_{ijk} \in ROI(\mathbf{p})} \delta_{\mathbf{p}} \cdot g(||\mathbf{x_p} - \mathbf{x}_{ijk}||, r_{\mathbf{p}}) \cdot \mathbf{v_p},$$

where a Gaussian kernel $g(d, r) = e^{-d^2/r^2}$ is applied for a natural attenuation of the created force. In this equation, the ROI's radius $r_{\mathbf{p}}$ controls the extent to which smoke flow is affected by the driving force, whereas the positive constant $\delta_{\mathbf{p}}$ adjusts the overall strength of the driving force that is added to the velocity field.

Once the driving force, followed by any other external forces like vortex or buoyancy force (*Add external force*), is exerted, the velocity field is updated by integrating the Euler equation $\frac{\partial \mathbf{u}}{\partial t} = -(\mathbf{u} \cdot \nabla)\mathbf{u} - \frac{1}{\rho}\nabla p + \frac{\mathbf{f}}{\rho}$, where p, ρ and $\mathbf{f}$ respectively indicate the fluid pressure, the density, and the external force acting on the fluid (*Advect velocity*).

Correction of velocity field using divergence constraint

While the applied driving force generates a series of velocity fields that guide smoke as intended, it is often observed that the resulting smoke tends to flow away from the trajectory in the regions of curve of high curvature due to an excessive accumulation of the driving force. In order to confine the smoke around the path, the velocity field must be corrected properly.

In [18], Feldman et al. applied a modified divergence equation $\phi = \nabla \cdot \mathbf{u}$ in the projection stage of their simulation pipeline in order to control the expansion ($\phi > 0$) or contraction ($\phi < 0$) of fluids through the divergence constraint ϕ. In fact, this idea is well-suited for resolving the deviation problem. Just before the projection computation, we modify the divergence value at each grid point as follows (*Adjust divergence*):

$$\phi(\mathbf{x}_{ijk}) = \sum_{\mathbf{x}_{ijk} \in ROI(\mathbf{p})} \epsilon_{\mathbf{p}} \cdot g(||\mathbf{x}_{\mathbf{p}} - \mathbf{x}_{ijk}||, r_{\mathbf{p}}) \cdot \kappa_{\mathbf{p}},$$

where the negative parameter value $\epsilon_{\mathbf{p}}$, combined with the Gaussian filter and curvature values, automatically controls the magnitude of confining force. The new divergence values are reflected to the velocity field through the modified Poisson equation $\nabla^2 p = \frac{\rho}{\Delta t}(\nabla \cdot \mathbf{u} - \phi)$ in the projection stage (*Project velocity*). Since the Poisson equation needs to be solved anyway in integrating the Euler equations, there is hardly no additional cost for constricting the smoke. See Figure 3 for an example of two-dimensional slices of velocity fields generated by our method.

Application of chemical kinetics

For a more effective chemical kinetics computation in a path-based smoke simulation (*Apply chemical kinetics*), we slightly modify the computational model that was described in [16]. In the original method, they used both *Lagrangian* particles and *Eulerian* volume densities to represent the reactive fluids. In particular, the major role of the *material* particles was to provoke a specified chemical reaction, creating an overall flow of fluids as designed, whereas the volumetric density fields, generated during the reaction computation, were visualized as fluids in the rendering stage.

In contrast, our method uses only particles, called *chemical* particles, to model the chemical species, both reactants and products, that participate in chemical reactions. As well as triggering given chemical reactions, these particles or selected ones of them are directly rendered as smoke using our particle renderer. This particle-only reaction scheme allows a more compact and efficient computational framework because there is, for instance, no need to deal with entire volumetric datasets to represent 'path-based' smoke that often takes only small space in simulation domain (note that the velocity field is still computed in volume space as we currently find it more stable in our implementation).

We also employ soot and/or vortex particles, as in [16], to improve the rendering quality and to produce a detailed turbulent appearance of smoke, respectively. However, contrary to the previous method that the vortex particles were treated as massless, we find that particles with proper masses create more controllable and interesting animations.

Animation controls through chemical reactions

Whether physically correct or not, we may have various parameters affect given chemical reactions so that intended animation effects can be created. In this section, we briefly summarize those kinetics-related parameters that were frequently used in generating our animation examples (for details on the theory of chemical kinetics, please refer to an introductory textbook in physical chemistry). Consider an example reaction described by an equation $aA + bB \longrightarrow eE + fF$. Above all, the chemical reaction is dominated by the rate of reaction r, which is a time-dependent function of the molar concentrations (in mol/L) of participating species. For a large class of 'forward' reactions, it is experimentally expressed in the form $r = k[A]^{\alpha}[B]^{\beta}$ for some orders α and β (note that these constants are user-controllable parameters in the point of smoke animation), where the rate constant k is usually dependent on temperature T as in the Arrhenius equation $k(T) = \gamma_1 e^{\frac{-\gamma_2}{T}}$ with two parameters γ_1 and γ_2. Controlling the rate function is one of the most fundamental tasks as it decides how fast the reaction takes place, affecting the entire animation deeply.

With the reaction rate determined, the amount of smoke that appears or disappears can be controlled through the stoichiometric coefficients a, b, e, and f as the changes of the species are respectively determined by solving the differential equations $r = -\frac{1}{a}\frac{d[A]}{dt} = -\frac{1}{b}\frac{d[B]}{dt} = \frac{1}{e}\frac{d[E]}{dt} = \frac{1}{f}\frac{d[F]}{dt}$. The numbers and masses of chemical particles of given species, say A, are decided by its molar concentration $[A]$ and molar mass M_A, the mass of one mole of substance A. By controlling the molar mass, it is possible to adjust the speed of smoke

flow along trajectory as the mass is an important factor in the particle advection stage that governs the motion according to Newton's second law of motion (*Advect particles*).

A user-defined heat source function $H_T = f_T(r, \cdots)$ is useful as its reaction-dependent value is reflected during the temperature advection computation to increase or decrease temperature (*Advect temperature*). It controls the change of temperature in the reaction system, which in turn affects the occurring reaction through the rate function $r = \gamma_1 e^{\frac{-\gamma_2}{T}}[A]^\alpha[B]^\beta$, for instance, and/or the buoyancy force. Vortex particles also generate a detailed turbulent appearance of smoke effectively by affecting velocity field through the vorticity confinement force. Its effect is controlled by yet another set of user-specified functions that define the number of generated particles $n_{vort} = f_{vort}(r, \cdots)$ and the initial vorticity vector $\omega_{vort} = f_{vort}(r, \cdots)$. As explained before, the divergence value ϕ is exploited for creating the confining force as well as controlling the expansion or contraction of reacting smoke.

Last but not the least, a variety of interesting chemical reactions, such as the appearing/disappearing and glowing/splashing phenomena of smoke, may be produced by creatively designing composite reactions. This is one of the most prominent advantages of exploring the theory of chemical kinetics for the path-based smoke simulation, as will be demonstrated by examples in the next section.

Animation examples

In order to demonstrate how the theory of chemical kinetics can be applied to the path-based smoke animation, we have implemented our simulation framework, and tested it with several scenarios involving various types of reaction equations.

Simple smoke controls with reaction parameters

Consider the following simple reaction equations, having four chemical species A, B, C and D, and five stoichiometric coefficients σ_1, σ_2, σ_3, λ_1 and λ_2:

$$\sigma_1 A + \sigma_2 B \longrightarrow \lambda_1 C, \quad \sigma_3 C \longrightarrow \lambda_2 D$$

In this example, the reaction was triggered as proper amounts of chemical particles that modelled the reactants A and B were injected into given path via its starting point. Then, chemical particles for the product D, created as a result of the reaction and advected through the path by driving force, were rendered as smoke.

As mentioned in the previous section, the amount of generated smoke can be controlled by adjusting the stoichiometric coefficients of given chemical reaction and the molar masses of participating chemical species. Figure 4 shows two examples of controlling the amount and behavior of smoke advected through given space curve. In (a), the stoichiometric coefficient λ_2 of D was tested to control the volume of produced smoke, whose control effect is easily understandable. On the other hand, in (b), the effect of changing molar mass of species D was investigated. Note that the mass of chemical particle is determined as the

product of its molar concentration and molar mass. Hence, the molar mass is quite useful to control the velocity of smoke flow as the acceleration is inversely proportional to the mass according to Newton's second law of motion. Refer to the caption of Figure 4 for details.

The overall appearance of smoke may be also controlled through the other reaction-related parameters as demonstrated in Figure 5. In the two examples depicted in (a) and (b), we attempted to adjust the amount of produced smoke by controlling the reaction rate of the first reaction equation $\sigma_1 A + \sigma_2 B \longrightarrow \lambda_1 C$ (note that the amount of species C produced directly affects the generation of the smoke substance D). In this experiment, we employed the reaction rate function $r_1 = \gamma_1 e^{\frac{-\gamma_2}{T}} [A]^\alpha [B]^\beta$ where the two constants γ_1 and γ_2 directly affect the speed that the first reaction occurs. It is possible to make the reaction less active by increasing the value of γ_2, for instance, entailing less smoke produced (compare the figures (a) and (b)). In this example, we set the number of vortex particles, determining the amount of circulation in smoke, to be created proportionally to the reaction rate r_1 so more turbulent smoke was also possible by decreasing γ_2. On the other hand, we could also adjust the volume of smoke through the divergence constraint parameter ϕ as demonstrated in the figures (c) and (d) (details are described in the caption of the figures).

Making smoke disappear and then reappear

In addition to adjusting the parameters of reaction equations, various interesting animation effects can be produced by creatively designing appropriate equations. For instance, the

following (artificial) chemical reaction enables to have smoke disappear and then reappear shortly thereafter on the way of travel along a given path (see Figure 6(a)):

$$A + B \longrightarrow 2.0C, \quad 0.01C \longrightarrow 5.0D, \quad D + F \longrightarrow A$$

To generate this animation, chemical particles for species A and B were initially positioned around the starting point of the path. Then, D particles, produced via C (the first and second equations), were rendered as smoke while they flowed through the trajectory. To create the desired effect, we distributed an appropriate amount of F particles around a point on the path, at which the smoke was to disappear. We also positioned B particles around another point, at which the smoke would start to reappear. Here, the B and F particles worked as on-off switches. When the D particles hit the F particles, they turned into the invisible A particles (the third equation). Upon meeting the B particles at the second point, the A particles were transformed into the D particles through the first two equations, making the smoke reappear. Note that the stoichiometric coefficients in the reaction can be used to control the amount of smoke that disappears and reappears.

Creating a firework-like smoke effect

The next example, based on the following reaction, intended that smoke would flow through given path, instantly splashing in all directions like firework at pre-specified path points (see Figure 6(b)):

$$A + B \longrightarrow A + 0.05B + 0.1C, \quad C \longrightarrow 200.0D$$

In this example, A particles, acting as visible smoke, were injected into the starting point of the curve, and forced to flow along the path by applied driving force. When they arrived at the curve points on which B particles had been put, the reaction of the first equation took place, generating the C substance that had a role of determining the locations and the directions of splashing. It was then regenerated instantly as D particles at an exponential rate (the second equation), which sparkled in random directions. To create the momentary glows at the moment of sparkling, we generated light emitting volume photons for realistic rendering, whose number and color were set based on the heat $H_T = f_T(r, \cdots)$ induced by the chemical reaction.

Figure 6(c) displays a blasting fuse-like example, produced by a variant of the previous reaction, where repetitive splashing along given path was intended through the following reaction:

$$A + B \longrightarrow 0.1A + 0.1B + 0.1C + 5.0E, \quad C \longrightarrow 200.0D$$

As in the previous example, the C material triggered the splashing effect. On the other hand, the A and B particles were regenerated by 10% of the original amounts, re-participating in the reaction so that the splashing phenomenon continued while they vanished in a gradual, fading manner. This example also employed another species E, associated with a positive divergence constraint $\phi = 0.1$, to create the secondary remnant smoke resulting from the splashing phenomenon.

Creating a soot-like effect

The fourth example animation in Figure 6(d) was created with a simple reaction $A + B \longrightarrow 5C + D$, where the white smoke and the green soot-like material were modeled by the C and D species, respectively. In order to create the soot-like effect, we used vortex particles whose number and initial velocity vectors were determined by $n_{vort} = \frac{r}{\eta}$ and $\omega_{vort} = c_{vort} \cdot \frac{\Delta r}{||\Delta r||}$, where r is the rate of the reaction, η is a user-controllable constant, and c_{vort} is another parameter that controls the overall strength of vorticity.

Synthesizing more complicated animations

Lastly, Figure 7 shows two more interesting examples that demonstrate how basic reactions can be effectively combined to create complicated animations. Here, the figures (a) and (b) show simulation results, before and after rendering, produced by the following reaction involving eight reacting substances:

$$0.5A + 0.5B \longrightarrow 0.03C + 2.5D + G, \quad 0.2A + 0.2B + 0.5F \longrightarrow 20.0E,$$

$$E + H \longrightarrow 0.5G, \quad C \longrightarrow 200I$$

In this example, the G substance modelled the red smoke, drawing the word *fluid*, whose particles were generated by the first and third equations, and rendered as 'red soot-like material'. Initially, A and B particles were distributed randomly along the locations of the path where the G material was desired to be created. Then, the starting point of the path was heated up enough to provoke the reaction of the first equation that was set, through the rate

function $r_1 = \gamma_1 e^{\frac{-\gamma_2}{T}}[A]^{\alpha}[B]^{\beta}$, to occur actively only at high temperature. This temperature was then advected along the path by the velocity field induced by driving force, causing the G particles were created gradually by the first equation. Here, the C and I substances were used to generate the splashing effect as before, whereas the D material was employed to produce the remnant smoke. In order to get more control over the volume/appearance of the generated G smoke, we added two more reaction equations whose stoichiometric coefficients were another set of control parameters (the second and third equations). The F substance in the second equation functioned as a switch that generated a proper amount of the E substance which in turn caused to generate additional G particles as a result of reaction with H particles that had also been randomly put along the path. All together, this composite reaction mechanism created a fancy sparkling phenomenon in which the word *fluid* appeared gradually.

The final example in the figure (c) was created by combining two different types of chemical reaction equations. First, a simple reaction $\sigma_1 A + \sigma_2 B \longrightarrow \lambda C$ was applied to stir up ground fog as a man runs on the floor. In this example, A particles were sprayed over the skin of the man while B particles were distributed randomly on the floor. As the man started running, the reaction occurred in the contact regions of the man and fog, producing the C material, where both B and C particles were rendered as fog. In this example, vortex particles were additionally generated to depict a natural turbulent appearance of the ground fog. On the other hand, another set of chemical reaction equations, similar to those

described in the previous examples, were applied to generate the blue smoke that chased the man through a pre-specified path. These two examples demonstrate how various simple chemical reactions are creatively combined to synthesize interesting, composite fluid animation effects.

Performance statistics

To demonstrate the effectiveness of the presented approach, we have implemented the computational pipeline illustrated in Figure 1, and measured its computational overhead. Table 1 summarizes the statistics collected on a desktop PC with a 2.4 GHz Intel Core 2 Duo CPU and 3.25 GB RAM for six example scenes described in this section. The obtained runtime results of a single-threaded program, given in average time per simulation frame, clearly reveal that the two steps, *Add driving force* and *Adjust divergence*, newly added to the conventional smoke simulation pipeline to make smoke flow through given trajectory, comprised only a trivial portion of the entire velocity computation time (*Update velocity*). On the other hand, the cost, needed to include chemical kinetics (*Apply chem. kinetics*) in the physics-based smoke simulation, varied depending on the complexity of involved reactions and the number of used/generated particles. Even for the most complicated example (Fig. 7(a)), however, the added expense was quite acceptable considering that several interesting animation effects were created at the same time.

Conclusion

In this paper, we have presented an effective method for path-based smoke simulation that explores the theory of chemical kinetics. We have shown that a wide range of interesting fluid effects could be modelled by creatively specifying a set of grammar-like rules through chemical reaction equations, and their appearance and behavior be intuitively controlled by adjusting the relevant parameters. The experimental results also demonstrated that the computational cost, additionally required to include the chemical reaction, was quite acceptable. While already fast enough for special effects production, the simulation pipeline, inherently containing much parallelism, could be made even faster by performing multi-threaded computations on the easily available multi-core CPUs and GPUs. It would also be interesting to see how remarkable a speedup a Lagrangian, particle-based velocity update computation would exhibit for the entire simulation.

Acknowledgements

This work was supported by the Korea Science and Engineering Foundation grant, funded by the Korea government (MEST) (No. R01-2007-000-21057-0(2008)) and the IT R&D program of IITA, funded by the Korea government (MKE) (No. 2007-S045-01(2008)).

References

[1] Y. Kim, R. Machiraju, and D. Thompson. Path-based control of smoke simulations. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '06*, pages 33–42, 2006.

[2] A. Angelidis, F. Neyret, K. Singh, and D. Nowrouzezahrai. A controllable, fast and stable basis for vortex based smoke simulation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '06*, pages 25–32, 2006.

[3] N. Foster and D. Metaxas. Controlling fluid animation. In *Proc. of Computer Graphics International '07*, pages 178–188, 1997.

[4] J. Schpok, W. Dwyer, and D. Ebert. Modeling and animating gases with simulation features. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '05*, pages 97–105, 2005.

[5] A. Treuille, A. McNamara, Z. Popović, and J. Stam. Keyframe control of smoke simulations. *ACM Transactions on Graphics (ACM SIGGRAPH '03)*, 22(3):716–723, 2003.

[6] A. McNamara, A. Treuille, Z. Popović, and J. Stam. Fluid control using the adjoint method. *ACM Transactions on Graphics (ACM SIGGRAPH '04)*, 23(3):449–456, 2004.

[7] R. Fattal and D. Lischinski. Target-driven smoke animation. *ACM Transactions on Graphics (ACM SIGGRAPH '04)*, 23(3):441–448, 2004.

[8] J. Hong and C. Kim. Controlling fluid animation with geometric potential. *Computer Animation and Virtual Worlds*, 15(3-4):147–157, 2004.

[9] L. Shi and Y. Yu. Controllable smoke animation with guiding objects. *ACM Transactions on Graphics*, 24(1):140–164, 2005.

[10] L. Shi and Y. Yu. Taming liquids for rapidly changing targets. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '05*, pages 229–236, 2005.

[11] N. Foster and R. Fedkiw. Practical animation of liquids. In *Proc. of ACM SIGGRAPH '01*, pages 23–30, 2001.

[12] N. Thürey, R. Keiser, M. Pauly, and U. Rüde. Detail-preserving fluid control. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '06*, pages 7–12, 2006.

[13] B. Kim, Y. Liu, I. Llamas, X. Jiao, and J. Rossignac. Simulation of bubbles in foam with the volume control method. *ACM Transactions on Graphics (ACM SIGGRAPH '07)*, 26(3):Article No. 98, 2007.

[14] Y. Dobashi, K. Kusumoto, T. Nishita, and T. Yamamoto. Feedback control of cumuliform cloud formation based on computational fluid dynamics. *ACM Transactions on Graphics (ACM SIGGRAPH '08)*, 27(3):Article No. 94, 2008.

[15] I. Ihm, B. Kang, and D. Cha. Animation of reactive gaseous fluids through chemical kinetics. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '04*, pages 203–212, 2004.

[16] B. Kang, Y. Jang, and I. Ihm. Animation of chemically reactive fluids using a hybrid simulation method. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation '07*, pages 199–208, 2007.

[17] R. Fedkiw, J. Stam, and H. Jensen. Visual simulation of smoke. In *Proc. of ACM SIGGRAPH '01*, pages 15–22, 2001.

[18] B. Feldman, J. O'Brien, and O. Arikan. Animating suspended particle explosions. *ACM Transactions on Graphics (ACM SIGGRAPH '03)*, 22(3):708–715, 2003.

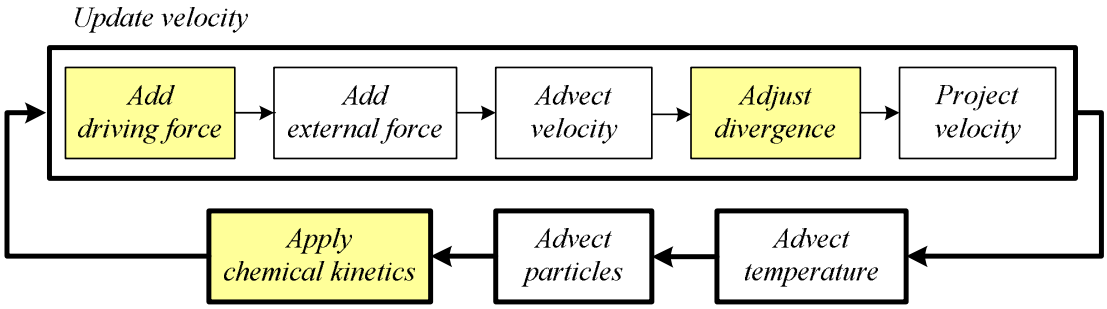


Figure 1: The computational pipeline for our chemical kinetics-assisted, path-based smoke simulation. The shaded boxes indicate the computation modules, newly added to a conventional smoke simulation pipeline.

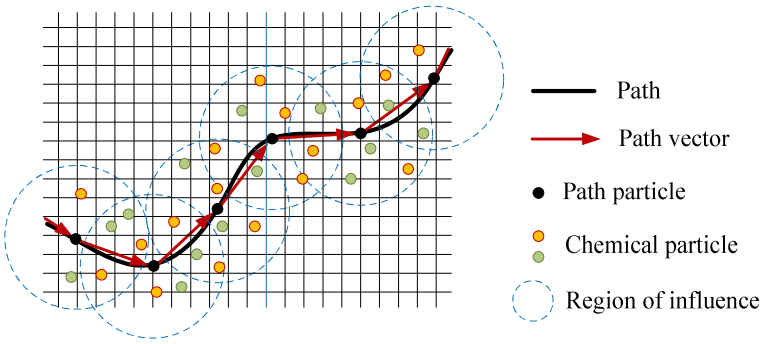


Figure 2: Path and its region of influence. Once a given path is discretized into a sequence of path particles, driving force is added at the grid points within its region of influence in the *Add driving force* stage. Furthermore, in order to guarantee that the resulting smoke is confined to the path without a deviation in high-curvatured path intervals, the velocity field is corrected in the *Project velocity* step by adjusting the divergence constraint based on the path's curvature information in the *Adjust divergence* stage.

(Time unit: sec.)

		Fig. 4(a)	Fig. 6(a)	Fig. 6(b)	Fig. 6(c)	Fig. 6(d)	Fig. 7(a)
Total frames		200	220	280	85	180	180
Path particles		56	56	56	143	143	125
Update velocity		0.6326	6.9436	3.2311	1.1781	0.7057	3.3617
Add driving force		**0.0434**	**0.0370**	**0.0316**	**0.0280**	**0.0820**	**0.0535**
Adjust divergence		**0.0142**	**0.0089**	**0.0041**	**0.0004**	**0.0470**	**0.0160**
Advect temperature		0.1720	0.1703	0.1691	0.2431	0.2012	0.5090
Advect particles		0.9009	1.9361	1.0436	0.5667	0.2625	0.2346
Apply chem. kinetics		**0.2672**	**0.9876**	**0.2582**	**0.1010**	**0.6355**	**4.9884**
Chemical particles	A	6	44,686	121,218	700	2,000	760
	B	6	17	40	700	2,000	760
	C	2	143	6	703	51,457	1,084
	D	114,822	206,962	5,575	29,882	7,409	35,728
	E	–	–	–	25,250	–	5,493
	F	–	90	–	–	–	11,645
	G	–	–	–	–	–	36,071
	H	–	–	–	–	–	104
	I	–	–	–	–	–	24,504
Vortex particles		0	71,484	31,118	10,000	659	32,505

Table 1: Summary of performance statistics. In this experiment, given paths were discretized using proper numbers of path particles (Path particles). The respective simulation times of each example scene were averaged over entire frames. A grid of $100 \times 100 \times 128$ was employed to represent the velocity field for all test scenes except the last one (Fig. 7(a)) that used a $180 \times 128 \times 64$ grid. The *Update velocity* row indicates the entire computation time taken for the velocity computation, while the following two rows (*Add driving force* and *Adjust divergence*) show the respective partial times necessary for the generation of path-based smoke flow. As some particles, participating in the reactions, appeared only for a short period of time, we provide the maximum numbers of them, achieved during the simulations, rather than averaging over the entire frames.

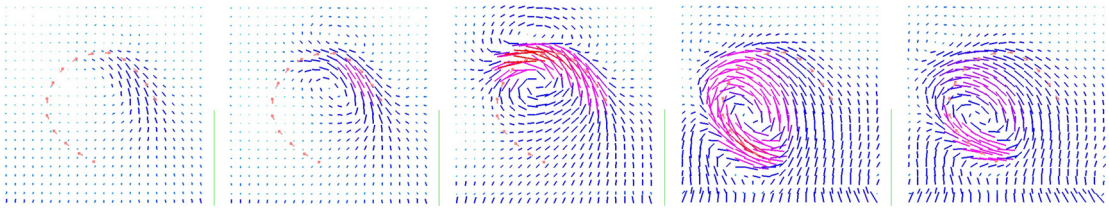
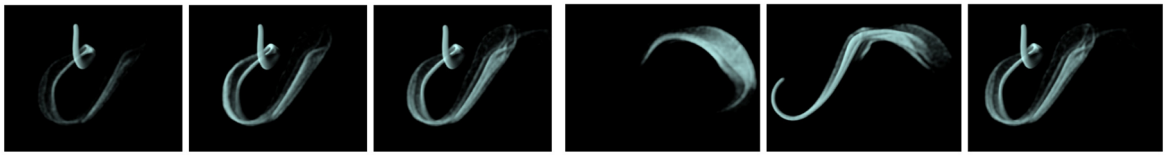
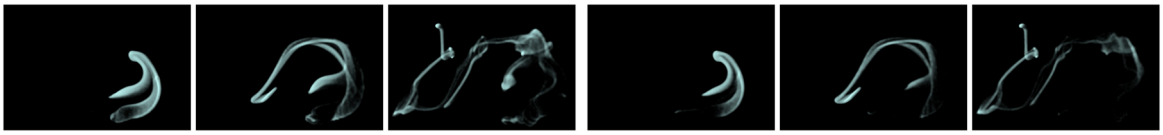


Figure 3: Two-dimensional slices of generated velocity fields. These images illustrate an example of velocity fields induced by the presented velocity update method. By applying a driving force and correcting the velocity through a divergence constraint, desired velocity fields were produced, forcing smoke follow given path faithfully.



(a) Control with stoichiometric coefficients (b) Control with molar masses

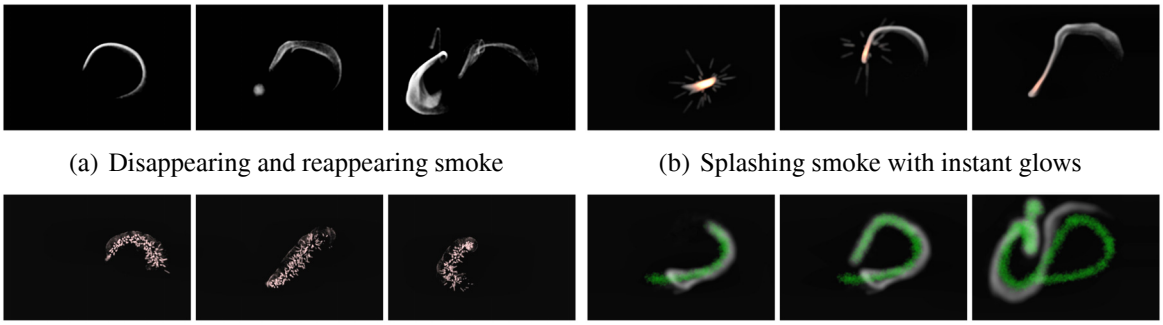
Figure 4: Controls with reaction parameters I. (a) Three different looking animations were produced by varying the stoichiometric coefficient λ_2 of product D, that was visualized as smoke (λ_2 =0.5, 1.5, and 2.0 from left to right). In this example, the number of D particles, increased from 53,067 to 170,209 and then to 185,910, resulting in more abundant smoke. (b) The molar mass is another useful reaction parameter that can easily control the velocity of smoke flow. Three different molar masses, $M_D = 0.2$, 0.1, and 0.05 from left to right, were respectively tried for the smoke substance D. Clearly seen in the three images from the same time frame, lighter smoke moved along the path both faster and longer. See the attached movies.



(a) $(\gamma_1, \gamma_2) = (300, 0.5)$ (b) $(\gamma_1, \gamma_2) = (300, 2.0)$

(c) $\epsilon_{\mathbf{p}} = 1.0$ (d) $\epsilon_{\mathbf{p}} = 1.5$

Figure 5: Controls with reaction parameters II. (a) & (b) The two parameters γ_1 and γ_2 of the rate $r_1 = \gamma_1 e^{\frac{-\gamma_2}{T}}[A]^{\alpha}[B]^{\beta}$ of the first reaction equation can have a great effect to the appearance of produced smoke. For instance, an increase of γ_2 would result in a slower reaction, generating less smoke. Compared to the smoke shown in (a), we can see that thinner smoke was generated in (b). Furthermore, we also set that vortex particles were generated proportionally to the reaction rate r_1, causing a more detailed turbulent appearance in (a). (c) & (d) In addition, $\epsilon_{\mathbf{p}}$ in the divergence adjustment function $\phi(\mathbf{x}_{ijk}) = \sum_{\mathbf{x}_{ijk} \in ROI(\mathbf{p})} \epsilon_{\mathbf{p}} \cdot g(||\mathbf{x}_{\mathbf{p}} - \mathbf{x}_{ijk}||, r_{\mathbf{p}}) \cdot \kappa_{\mathbf{p}}$ is yet another useful control parameter for controlling the contraction/expanstion of created smoke, as explained before. As demonstrated in the figures (c) and (d), a slightly more expanded smoke was generated by increasing $\epsilon_{\mathbf{p}}$ from 1.0 to 1.5. See the attached movies.



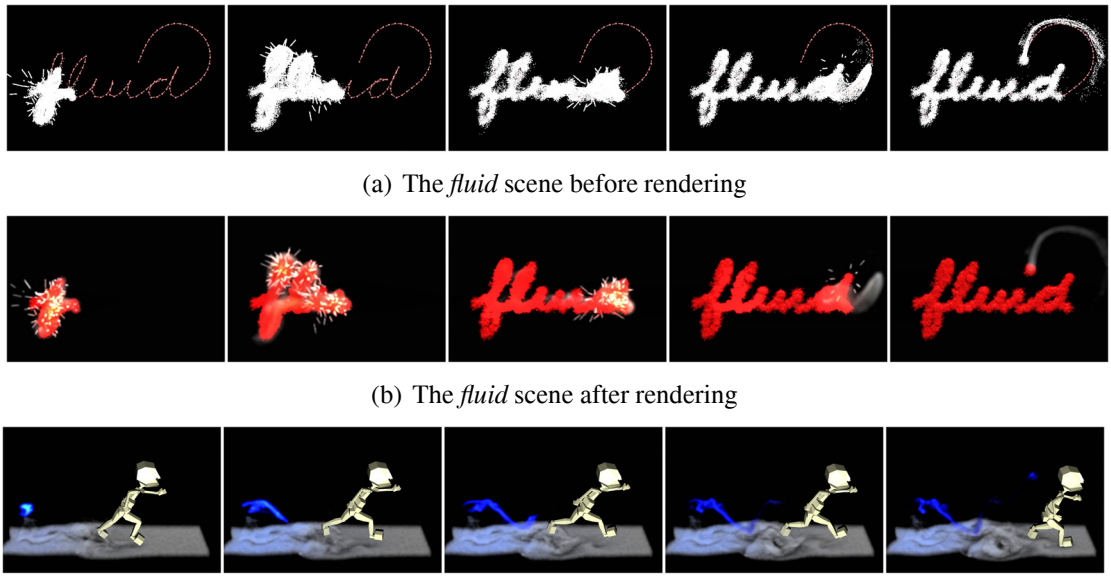
(a) Disappearing and reappearing smoke

(b) Splashing smoke with instant glows

(c) Repeatedly splashing smoke

(d) Smoke and soot

Figure 6: Four path-based smoke animation effects. As explained in the text, a variety of interesting animation effects were possible by designing, through the chemical reaction equations, appropriate grammar-like rules that specify desired smoke animations. See the attached movies.



(a) The *fluid* scene before rendering

(b) The *fluid* scene after rendering

(c) The *running man* scene

Figure 7: More examples. These example scenes demonstrate that a combination of various basic chemical reactions can create interesting path-based smoke animations effectively. See the attached movies.