Real-time dynamic and pressure-sensitive brush rendering

Liqiang Shi
University of Science and Technology of China, Anhui, China.
Shizhe Zhou
University of Science and Technology of China, Anhui, China.

ABSTRACT: The work develops a real-time simulation interface for dynamic brush sketching effect by combining drawing tablet and GPU. The system takes the drawing dynamics from a consumer-level drawing tablet, and makes full use of the received information about location and pressure to generate different brush styles. The whole task of the brushes' simulation and visualization only uses GPU, taking into consideration both the final diffusion effect of the liquid color oil and the artistry of their intermediate status of detail shapes of the brush area. Our brush system also supports importing of carefully designed textures and moving frame sequence to create stylized dynamic brushes. By using this system, the users can conveniently render images in real time by simulating painting oil and pigment's dynamics.

1 INTRODUCTION

In the field of computer graphics, real-time brush drawing system satisfies users' need by transforming their inputs into visual pleasing displays. These systems often provide the function of simulation sketching and painting effects. Sketching line is one of the basic elements of the creation of graphical content. Generally, a variety of graphic design software provides various kinds of brush libraries as well as parameters for adjusting the contour shape of the brush area. Although the user can specify a brush parameter controlling the jittering along the central curve, it is still a static collage of the specify brush pattern. Recently, in the field of image stylization, There emerges an novel approach: collect the brush database from the brush samples reversely and get the specified pattern based on the synthesis of contour and inner region. Since this process supports the selection of significant contour area from the original image interactively, it maintains the overall drawing style and reproduces the quality lossless in the target figure. For synthesizing the outer contour, this method takes on a "semi dynamic" effect: when the users drag a control point of the curve, they generate a partial result by generating piecewise fragment texture mapping using the brush texture inside the extension area along the central curve. But the generated brush texture has been fixed, so the final output is also a static image.

This paper focuses on how to use the dynamic brushes in a more reasonable way to imitate the color diffusion effect that are ubiquitous in color painting, Chinese painting and other real art forms. In order to achieve real-time process we design a brush computing and rendering system solely based on GPU. The fast parallel computing capability of GPU in mass textures is widely used in computer graphics. For example, in a non-photorealistic rendering field, artists only need to write the code segment-by-pixel (fragment shader) to get complex lighting models suitable for real-time calculation. In addition, these lighting models can customize their parameters and observe the effect of their corresponding rendering in real time.

Overall, there are only few studies on the dynamic diffusion effect of brush pigment. They are all based on a physical model of fluid simulation, the Navier-Stokes equation, which is more complicated in terms of calculation and parameter setting, and their working domain generally equals the whole square canvas area. When removing the divergence in the convection diffusion step using implicit Lagrangian method, the result is quite sensitive to the setting of the iteration step length. Using this method for illustrating dynamic brush effect is not easy to control because of its overly diffusion. We design our diffusion brushes based on the idea of diffusion curve: We set a narrow computational domain for a single brush (determined by the central curve and the width along the direction orthogonal to the central curve), which avoid the dissipation problems of existing methods. Our brushes show various dynamic diffusion effects within a specified period of time, such as homogenization of color, emergence of micro textures, the expansion and the emergence of the micro dynamic structures etc. Our

corresponding author: Shizhe Zhou,shizhezhou2014@gmail.com

system has the same performance advantages as diffusion curve: the whole process of generating and rendering is equivalent to solving a Poisson equation using an iterative method and displaying the data flow during each step. Here the Jacobi iteration method is used for solving the Poisson equation, since it has a simple structure and is friendly to a GPU implementation.

We integrate a drawing tablet for user interaction. This kind of system is usually composed of a digital drawing board and a pressure pen,similar to the actual drawing board and brushes. Besides, the Wintab system offers a programmable interface that provides for the tablet application developers the real-time data flow from the pressure pen in the windows system. We extract the pressure data and then transform it into the brush width.

2 RELATED WORK

In the past ten years,researchers have developed a number of models to simulate brushes by computer, the majority of which are about the simulation of the ink painting, the water color painting and other art styles. We make a brief review of these techniques: Strassmann (1986) models a collection of bristles which evolve over the course of the stroke on rice paper, leaving a realistic image of a sumi brush stroke;Guo & Kunii (1991)further extend the model to the process of the diffusion of ink via the paper fibers. Shi et al. (2003) propose to simulate ink painting based on a particle system. Specifically they employed the ”pseudo-Brownian motion” as the driving force of the flow of ink particles, and finally simulated some effects of the typical Chinese ink painting. Besides its complexity in terms of implementation, this drawing process doesn't include any dynamic diffusion process.Yu & Xu (2000)proposes a three-dimensional gouache brush model, which includes brush unit selection, pigment simulation, pigment diffusion and the whole control.Meier (1996) propose a brush model to simulate the effect of oil painting. Xie et al. (2008) also develop a model to simulate the effect of the pencil drawing based on the algorithm of convolution operator. After analyzing the structural characteristics of the real texture of a pencil, they set up a simple mathematical model of pencil stroke, and successfully realized the real-time pencil-style drawing of video image rendering. In general, the study of the dynamic diffusion effect of brush pigment is less than those for the static brushes.

Diffusion curve is proposed by Orzan et al. (2008). Given a set of planar curves and anchor color values defined at the control points of these curves, they use a Poisson equation to interpolate color for the rest of the canvas domain. Solving the Poisson equation boils down to finding the solution of a large sparse linear system. Classical methods include Jacobi iterative method and matrix decomposition-based methods. Thanks to the recent research on GPU general purpose computation, several parallel algorithms for solving large linear system has been proposed: Goodnight et al. (2003) propose GPU-based multi-grid algorithms, while Jeschke et al. (2009) develop efficient parallel implementation of the Laplacian solving method, whose implementation is simpler than multi-grid method.

We use Wintab drawing tablet for receiving the user input. Wintab is also a well-known interactive device for developing both research and commercial drawing systems. For instances, Quan (2007) developes a system of identity authentication based on dynamic handwritten signature.Vandoren et al. (2009) use the tablet to simulate a watercolor drawing interface. Users can choose the color they prefer for a watercolor pen to draw objects. Experiments show that this system improves the creative expressiveness for the artist.Chu & Tai (2005) also adopted this interface to develop a rice-paper painting drawing system. Their design of the diffusion brushes is also based on this interface. Based on the existing brush models, we adopt the concept of diffusion curves and further develop a brush model focusing on dynamic diffusion effects. We imitate the color diffusion effect in a more reasonable way and simplify the complexity of the realization of the algorithm. At the same time, we take the drawing tablet as our input device and finally get the real-time diffusion effect of the brushes with the real-time solving process entirely on GPU.

3 DRAWING TABLET

Figure 1: Bamboo drawing tablet(product copyright @ Wacom Co.Ltd)

Drawing tablet is favored by the majority of drawing enthusiasts. The main part of the tablet hardware is an electromagnetic-aware touching surface. Its main customizable parameters include pressure sensing, coordinate accuracy, input reading rate, resolution, among which the pressure sensing reflects the instant pressure between the touch surface and the pen. The thickness of the stroke may reflect the users' pressing force, which brings a sense of reality to the users.

In this paper, the brushes we design incorporate drawing tablet developed by Wacom Company. With the interface of Wintab data packet, it is easy to program. When we develop this system, we extract the pressure data and then transform it into the brush

width. Finally, we will get the real-time pressure-sensitive dynamic brushes.

4 MODEL ESTABLISHMENT

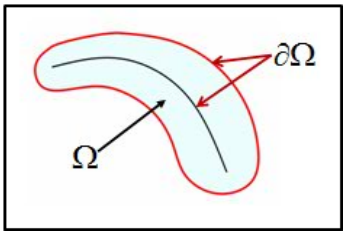


Figure 2: Computing domain of our dynamic diffusion brush.Note its boundary has two parts: the outer border and the central skeletal curve.

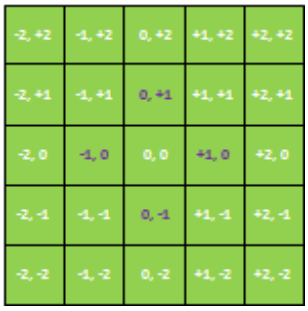


Figure 3: Our difference scheme for solving the PDE using a one ring neighborhood of N_p.

Figure.2 illustrates the computation domain and its boundary condition locations of our diffusion brushes. The red and black curve denoted by $\partial\Omega$ and pointed by the red arrows are the boundaries of the brush, where we set the constraint values for the Poisson equation. The region Ω within the outer red boundary excluding the pixels covered by the central skeletal curve is the computing domain of our brush. Each pixel inside the computing domain correspond to a unknown variable of our linear system for solving the Poisson equation. Contrary to existing diffusion brush systems, we gradually grow the boundary of the brush stroke. During growing the boundary we solve the following equation by interation:

$$\begin{cases} \triangle f = divR \\ f = c(x,y) \quad \text{if pixel(x,y) is on the boundary} \end{cases} \tag{1}$$

In this equation, R represents the background of the drawing domain Ω and $c(x,y)$ the condition of boundary. When the pixel is on the boundary $\partial\Omega$, the value of f is the corresponding color value in $c(x,y)$. The final value of f evaluated according to equation (1) decides the final diffusion effect.

To achieve real-time dynamic diffusion effect require a method to efficiently solve equation(1). We convert the equation into a discrete format and put all the computing process and data flow onto a GPU platform. Note that for any pixel$p$, N_p represents its one radius neighborhood containing four pixels.I.e, for pixel in the figure.3, $N_{0,0}$ represents pixels at a neighborhood with the offset of $(0,1)$,$(0,-1)$,$(-1,0)$,$(1,0)$.

Besides, f_p represents the color value of the pixel p . Our final discrete format of the equation for each pixel $p \in \Omega$ can be expressed as:

$$|N_p| f_p - \sum_{q \in N_p \cap \Omega} f_q = \sum_{q \in N_p \cap \partial\Omega'} c_q + \sum_{q \in N_p} \triangle_{pq} \tag{2}$$

In this discrete format, the value of c_q is the corresponding color value of pixel q when it is on the boundary, $\partial\Omega'$ is the outer boundary of the brushes, and $\triangle_{pq} = R_p - R_q$. According to the radius of the neighborhood used in our format, the value of N_p is four.

According to the discrete format (2) of equation (1), solving the discrete format is equivalent to solve a large sparse matrix. During this process, we will make full use of the parallel computing ability of GPU to solve equation (2) with iterative method.

5 SOLVE ON GPU

In order to achieve real-time diffusion results, we will take advantage of the parallel computing ability of GPU to solve the discrete format in Equation (2). For convenience, we will convert format (2) to the following iterative format:

$$f_p = 0.25 * (\sum_{q \in N_p \cap \Omega} f_q + \sum_{q \in N_p \cap \partial\Omega'} c_q + \sum_{q \in N_p} \triangle_{pq}) \tag{3}$$

When we solve our model on the GPU, drawing domain and boundary will grow larger at each iteration step during the movement of brush. In order to solve the discrete differencing format, we run a Nvidia CG function (Mark et al. 2003) which execute a fragment shader for each pixel within the drawing domain. Figure 4 is the schematic diagram of the solving process on GPU. We go into the process of iteration after calculating divergence field $divR$. During the iteration process, the real-time diffusion effects are saved in Out Texture and showed on the screen simultaneously. At the same time, the corresponding inner boundary pixel color's value of the brush is saved in the In Texture real-time. Thus, the value of the boundary pixels is reassigned for each new iteration. The color values of the inner boundary pixels will not change with iteration. Finally, the whole computing process simulates the dynamic effect of a diffusion brush.

6 DIFFUSION BRUSHES

6.1 *Dynamic diffusion brush*

Our dynamic brush (shown in Figure 6-a) takes the color chosen by the user for the central skeletal curve, such as blue, as the internal color, and take the total

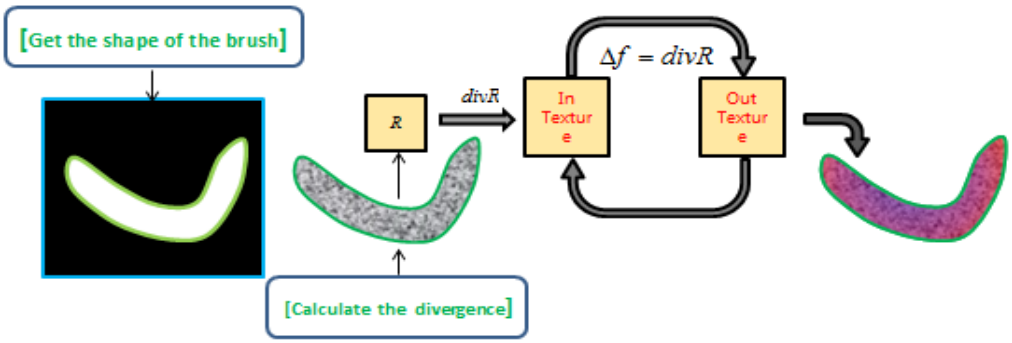


Figure 4: Workflow and GPU Texture layout. We do texture computing by iterating between two textures representing the rendering and hidden buffer for the canvas.

black of canvas as the boundary color. Then we generate diffusion effect by real-timely interpolate between these two colors when solving on GPU. Because the canvas is black, the value of $\sum_{q \in N_p} \triangle_{pq}$ in Equation (3) is set to be zero. Therefore, the iterative format (3) can be simplified as:

$$f_p = 0.25 * (\sum_{q \in N_p \cap \Omega} f_q + \sum_{q \in N_p \cap \partial\Omega'} c_q) \quad (4)$$

6.2 *Colorful brush*

During the moving of the brush tip, we randomly assign color value inside the brush area to achieve various bright-dark fading effects guided by the noise data. We show such results in figure 6-b. Furthermore, as the color is assigned automatically, the users do not need to select the brush color when this brush is in use, which is different from dynamic diffusion brush.

6.3 *Noise brush*

Sometime adding a certain amount of noise into the system can increase the reality since in real world both the painting old and the media contains noise. In our system it is easy to achieve such effect. Instead of the setting divergence field in Equation(1) as total zero, we add noise which are directly loaded from proper texture images and are transferred into GPU texture data before starting the iteration described by Equation (3). Consequently, we achieve several noisy diffusion brush drawing effects using the classic berlin noise (figure 6-c).

6.4 *Dynamic texture brush*

If the noise changes dynamically, it is conceivable that the brush will produce the dynamic diffusion effects. We directly load texture along with their motion data from GIF image files which are ubiquitous in the Internet. For instances, we load wave and stars blinking images. We then bind those pictures to the different attachment point of GPU internal memory. Finally, we can get the dynamic texture brushes after solving the iterative format (3). Figure 6-d shows the dynamic texture brushes developed by the gif image of stars. Our accompanied videos of this paper also demonstrate these effects. The users can choose the brushes they like to paint and can also take their favorite GIF image files to design new dynamic texture brushes.

6.5 *Dynamic growth of the tree brush*

The previously described brushes include an outer boundary and an inner central skeletal curve. Now, we add new inner boundary regularly with the movement of the brushes. In figure 5, while the brush is moving, we add new dynamical boundaries of a sinusoidal function to the inner boundary of the brush. As it still satisfies equation (1), there will be more apparent dynamic diffusion effects after solving the iterative format(4). Figure 6-e shows the final result of simulating the growing effects of wickers.

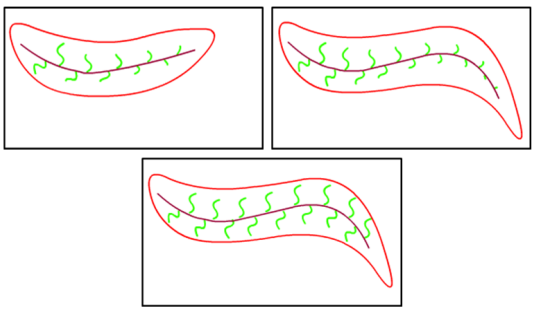


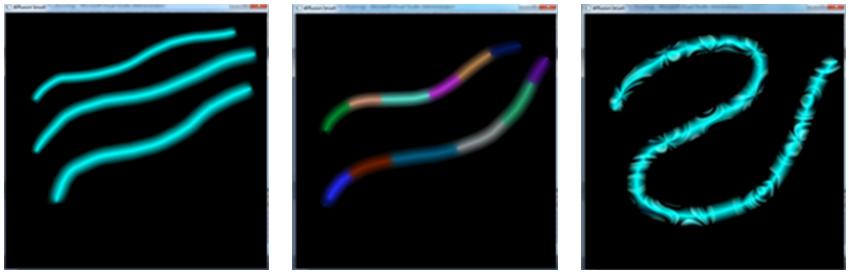
Figure 5: 3 consecutive frames showing motion of the dynamic pattern of the growing wickers.

6.6 *Brush painting on the white canvas*

The background color determines the color of the outer boundary of the brush. In addition to the black background, we can also choose other colors as the background in order to produce various diffusion effects. Figure 6-f shows the diffusion effect with white background.

7 CONCLUSIONS

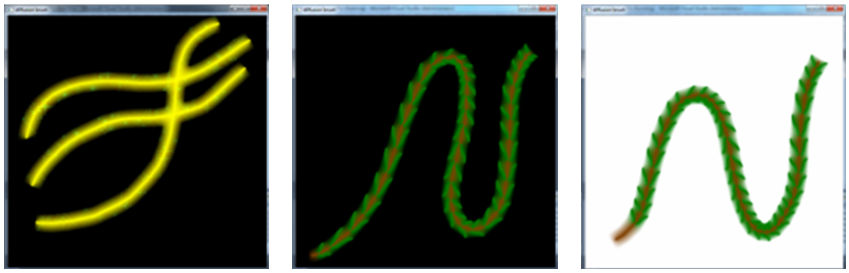
During the experiment, we implement the whole dynamic brush diffusion system on a NVIDIA GTX-660 card, combined with a bamboo drawing tablet for user having natural and realistic drawing experience. Embedded in the system we provide multiple options in



(a) Dynamic diffusion brush

(b) Colorful brush

(c) Noise brush



(d) Dynamic texture brush

(e) Dynamic growth of the tree brush

(f) Brush painting on the white canvas

Figure 6: Our system offer various options on brush dynamic and underlying textures.

terms of brush shape, underlying canvas texture, motion data of the dynamic of the bristles. Users can choose their favorite brush type during painting different art objects. Meanwhile, the pressure data from the sketching pen is reflected during the sketching the central curve of each stroke. Figure 7 shows some works created by our diffusion brushes. The video in this paper also records the real-time diffusion process.

The results show that the brushes we designed bring users choices. The use of dynamic diffusion brush and the colorful brush may help the users outline the contours of the objects; the detailed dynamic motion of the texture loaded in the GPU internal memory both for the shape of the brush and the canvas textures bring new light to the creation. Dynamic growth of the tree brush makes the users feel the vigor and vitality of the works during the painting process.

Thanks to GPU implementation, our diffusion brushes achieve the real-time diffusion dynamics. The main limitation of our system is that we can only create brush in a two-dimensional plane. A natural future work of this system is to extend the brushes our two-dimensional manifold surfaces so that the artists can create dynamic diffusing brushes on the input mesh surfaces.

REFERENCES

Chu, N. & C. L. Tai (2005). Moxi: Real-time ink dispersion in absorbent paper. *ACM Transactions on Graphics. 24(3)*, 504–511.

Goodnight, N., C. Woolley, & G. Lewin (2003). A multigrid solver for boundary value problems using programmable graphics hardware. *Technical Report.*.

Guo, Q. & T. Kunii (1991). Modeling the diffuse paintings ofsumie. *Modeling in Computer Graphics, T Kunii,eds. Tokyo:Springer-Verlag.*.

Jeschke, S., D. Cline, & P. Wonka (2009). A gpu laplacian solver for diffusion curves and poisson image editing. *ACM Transactions on Graphics. 28(5)*, 1–8.

Mark, W. R., R. S. Glanvilie, & K. M. J. (2003). Cg: a system for programming graphics hardware in a c-like language. *ACM Transactions on Graphics. 22(3)*, 896–907.

Meier, B. J. (1996). Painterly rendering for animation. *In: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, New Orleans, Louisiana.*, 477–484.

Orzan, A., A. Bousseau, & H. Winnemoller (2008). Diffusion curves: A vector representation for smooth-shaded images. *Proceedings of SIGGRAPH.*, 1–8.

Quan, Z. H. (2007). Research on identity authentication based on dynamic handwritten signature. *University of Science and Technology of China.*.

Shi, Y. X., J. Z. Sun, & H. J. Zhang (2003). Graphical simulation algorithm for chinese inkwash drawing by particle system. *Journal of computer aided and computer graphics. 15(6)*, 667–672.

Strassmann, S. (1986). Hairy brushes. *In: Computer Graphics Proceedings, Annual Conference Series, ACM SIGGRAPH, Dallas, Texas.*, 225–232.

Vandoren, P., L. Claesen, & T. Vanlaerhoven (2009). Fluidpaint: an interactive digital painting system using real wet brushes. *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces.*, 53–56.

Xie, D. E., Y. Zhao, & D. Xu (2008). A method for generation of pencil filter and its implementation on gpu. *Journal of computer aided and computer graphics.*.

Yu, J. H. & X. G. Xu (2000). Computer generated gouache rendering of 3d polygonal models. *Journal of computer aided and computer graphics.*.

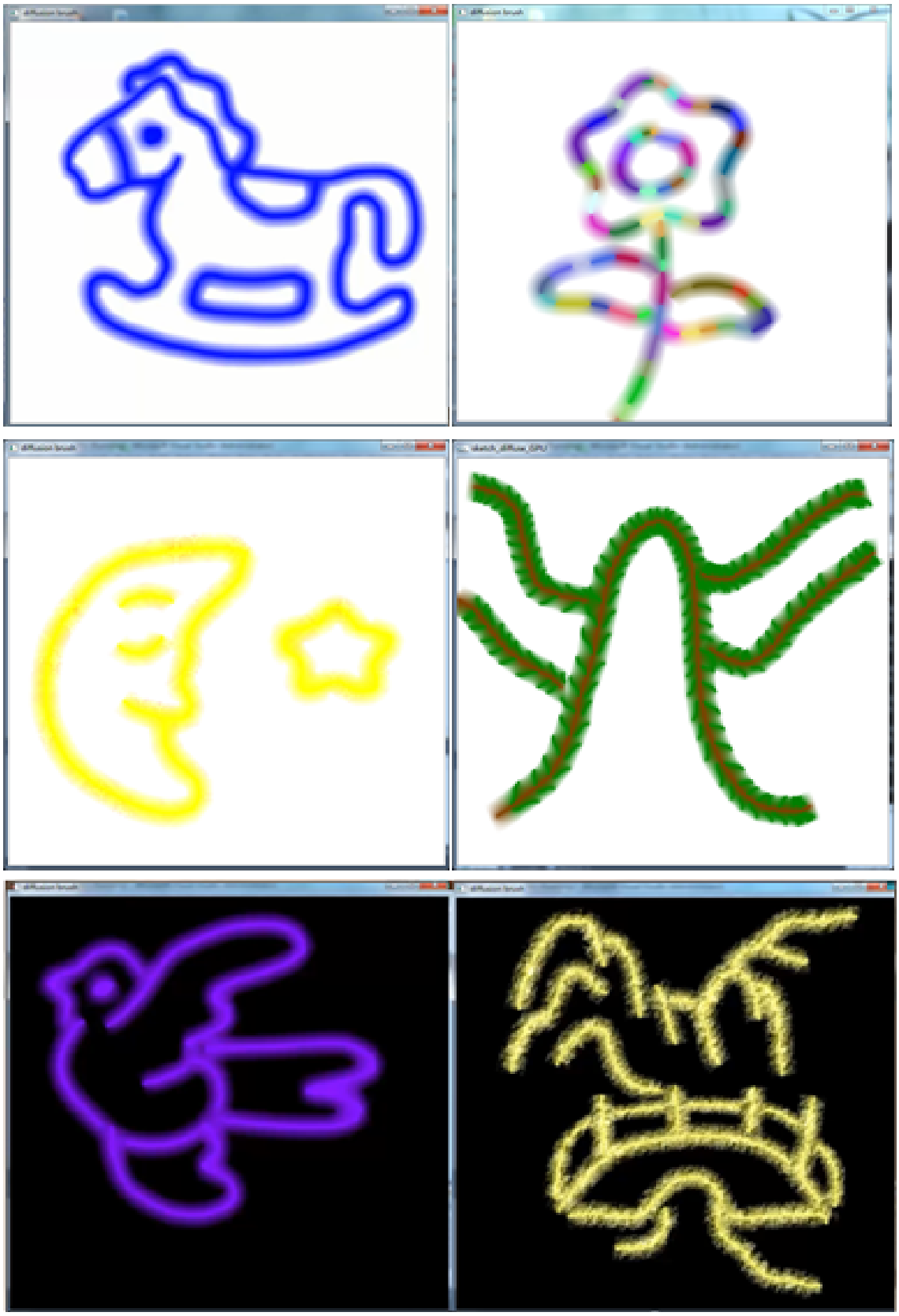


Figure 7: Result Gallery: Artists using our system draw objects on the screen. The dynamic effects are shown in our accompanied video.