# Modelling in 2D Enabling Fluid Stylised Animation

Fabian Di Fiore* Frank Van Reeth†

Limburgs Universitair Centrum
Expertisecentre for Digital Media
and
Transnationale Universiteit Limburg, School for Information Technology
Universitaire Campus
B-3590 Diepenbeek, Belgium

## Abstract

This paper introduces new techniques and tools to draw, manipulate and animate stylised brush strokes in computer assisted animation production. We focus on eliminating the time-consuming and tedious process of drawing and managing the numerous brushes that are painted on top of each other, and on avoiding temporal aliasing artifacts such as brushes popping up in successive frames. Moreover, we also aim at giving the animator the same freedom of expressing the artistic style he is bearing in mind as if painting in the traditional way.

To establish these goals we first break down the traditional drawing process into a modelling process and an animation process. The first is used to create extreme poses of the character (without the restrictive inconveniences of standard 'point-click-and-drag' metaphors), while the latter is employed to provide for frame-to-frame coherent animation. Aside this, we present some higher-level tools that enable the animator to locally and globally control user selected parts of the drawings. This simplifies the interaction drastically.

The provided solution is intuitive to use and empowers the production of painted animation while not hampering the animation artists' creativity.

**CR Categories:** I.3.m [Computer Graphics]: Miscellaneous—Computer-Assisted Traditional Animation; I.3.4 [Computer Graphics]: Graphics Utilities—Paint Systems

**Keywords:** painterly rendering, non-photorealistic rendering, stylised animation, computer-assisted traditional animation, computer animation

## 1 INTRODUCTION AND MOTIVATION

When we talk about animation in the context of this paper we refer to computer assisted animations where the objects and characters have a hand-painted look. When looking at existing developments, animators still have to make great efforts to interact with the software. Most systems only support the creation of drawings that are made out of curves. For cartoon animations this is relatively easy, since these characters typically are built up just from silhouette lines which are coloured and filled in a uniform way.

Difficulties arise when the animator chooses to replace the simpler cartoon style to a more painterly style such as the one depicted in figure 1. This kind of artistic drawings usually consists of numerous strokes that are painted one over the other. Therefore, when animating these drawings it is difficult for the animator to manage that much brush strokes in a proper and intuitive way. Existing applications use 'point-click-and-drag' procedures. Hence, creating and editing strokes is a tedious and unpleasant task that often results in mistakes being made. In addition, the applied painted strokes may not suddenly appear and disappear, nor move or deform with respect to the object. Without such frame-to-frame coherence, the temporal aliasing makes the animation hard to enjoy.

Figure 1: Animation of a palm tree blown by the wind.

Within the boundaries of this study our goal is to assist the animator throughout this cumbersome process. Besides tackling the mentioned issues, we also aim at giving the animator the same freedom of expressing the artistic style he is bearing in mind as if painting in a traditional way.

To establish these goals we first break down the traditional drawing process into a modelling process and an animation process. The modelling process is used to create the extreme poses of the object (disposing of several brushing and editing tools), while the animation stage is employed to provide for frame-to-frame coherent animation. Second, we present some higher-level tools that enable the animator to locally and globally control user selected parts of the drawings.

This paper is organised as follows. Section 2 describes previous work in the field and indicates the differences with our philosophy. In Section 3 we elaborate on our approach. First, we elucidate the modelling and animation stages in 2D. Then, the creation, (high-level) manipulation and animation of single brush strokes as well as painted characters is explained. Section 4 provides clarifying results, while we end with our conclusions and ongoing future research (Section 5).

*e-mail: fabian.difiore@luc.ac.be
†e-mail: frank.vanreeth@luc.ac.be

## 2 RELATED WORK

In this section we dwell on techniques starting from pure 2D drawings and some approaches found in the painterly rendering and non-photorealistic rendering (NPR) domain.

### 2.1 Pure 2D Approaches

Over the past years, a lot of work has been reported upon creating 2D stylised paintings. The majority of the publications concentrate on how to simulate real brush styles like pencils, airbrushes, watercolour, etc.

In 1990, [Haeberli 1990] demonstrated an interactive painting system for quickly producing a painted representation of a still image. More recently, a new algorithm was presented by [Hertzmann 1998] for producing paintings from images. Brush stroke sizes are selected to convey the level of detail present in the source image using a multi-scale algorithm. Direction normals, stroke curvature and other parameters describe a space of rendering styles that can be created and modified by artists and graphic designers. The painting process ítself is built up in a series of layers and so generates convincing results. Nowadays, a lot of commercial packages provide a wide variety of painterly image filters and brushing tools based on these ideas. However, a major drawback of these techniques is that they are only suitable for creating still images.

We refer the interested reader to [Gooch and Gooch 2001] for an overview of how to simulate artistic media such as brushes and other painting tools.

We can conclude that 2D painting systems are easy and intuitive to use and the way of working very much resembles the traditional way of working. However, these techniques do not take into account the succession of images and so can only be applied to create still images. In order to create stylised animations, new techniques will be required to maintain temporal coherence for each brush stroke.

### 2.2 Starting from 3D: Non-Photorealistic Rendering Techniques

Recently popular, non-photorealistic rendering (NPR) techniques are used to automatically generate stylised renderings and animations [Gooch and Gooch 2001; Strothotte and Schlechtweg 2002].

Barbara Meier [Meier 1996] presented an interesting solution to render animations in a specific artistic style without the need to draw each frame by hand and without suffering from temporal aliasing. She introduced a method to obtain a painterly render style starting from 3D geometrical objects. 2D brush stroke attributes are obtained from reference pictures and particles attached to the 3D model define the brush stroke locations. Meier managed to eliminate the 'shower door' effect (strokes seem to be disconnected from the objects they represent and float around) that disturbs many other approaches to obtain a good frame-to-frame coherence. This approach leads to very impressive results for rigid objects, but requires extensive modelling and animation if it were applied to fully animated characters. In practice, even for rigid objects, a lot of hard work is involved to accurately grasp the appearance an artist is imagining: models have to be built up and all necessary brushes need to be assigned properly.

[Kaplan et al. 2000] discussed an algorithm for rendering subdivision surface models in a variety of artistic styles using an interactively editable particle system. The algorithm is suitable for modelling artistic techniques explicitly by the user, or automatically by the system. Frame-to-frame coherence is maintained due to the use of particles to represent hand drawn strokes. However, a lot of user interventions are needed since the editing of strokes involves the individual editing of each of the corresponding particles.

More recently, [Kalnins et al. 2002] presented a system that lets an animator interactively embellish a 3D model with stylised strokes. In fact, strokes are drawn over the model from one or many viewpoints. The user has a wide variety of brush styles to choose from and so is given a high degree of control over the stylisation. Nevertheless, the results suffer from being either too cold or too '3D-ish'.

To sum up, starting from 3D models has the advantage of automatically generating stylised renderings and/or animations but at the cost of heavy modelling and animating. Moreover, since the underlying 3D geometry is rendered too accurate, a very 3D look is generated as well which we especially want to avoid.

### 2.3 Painterly Rendering Techniques

In 2000, Hertzmann presented a method for painterly video processing [Hertzmann and Perlin 2000]. Basically, successive frames of animation are painted over, applying paint only in regions where the source video is changing. Optionally, brush strokes may be warped between frames using computed or procedural optical flow. This method produces animation with a novel visual style distinct from previously demonstrated algorithms. However, it gives the subjective impression of rather a 'living' painting that is continually being painted over, than a fluent animation that is made in a particular painted style.

More recently, the same author described a system that uses a relaxation algorithm combined with search heuristics to produce painted animations starting from images or video [Hertzmann 2001]. The user keeps control over the process by varying the relative weights of energy terms. The energy function in turn yields an economical style and produces greater temporal coherence than the previous technique. A drawback of this method is that the energy function is very difficult to optimise, which therefore limits the number of animation styles.

To summarise, painterly rendering techniques are promising and deliver quite aesthetic results. On the other hand, from an artistic standpoint a lot of constraints are imposed on the traditional animator such as the automatic generation of brush strokes and the restricted number of available animation styles. Furthermore, in these approaches animation is only possible when a video or a sequence of images as input is available.

## 3 OUR APPROACH

In traditional 2D animation [Blair 1994; Williams 2001], the 'Ink and paint' process could somewhat be regarded as being the equivalent of the rendering stage in 3D animation, but without the explicit presence of the modelling and animation processes. Instead, they are combined into a single drawing process. In this drawing process, the most striking characteristic is the freedom the animator has. He can easily draw all kinds of stylised brushes on paper, he can dynamically alter the width and the curvature of the strokes and, moreover, when the animator is not satisfied with the result, he can make (simple) corrections just by drawing or painting over new strokes along or on top of the 'wrong' ones.

As will be clear from the subsequent subsections, in order to provide the same freedom, we opt for a methodology that clearly distinguishes a modelling phase and an animation phase (hence following 3D computer animation in this respect). This methodology already has been proven to be very useful [Di Fiore et al. 2001] for the purpose of creating convincing 3D-like animations starting from pure 2D vector drawings.

A major difference with pure 2D vector drawings is that painted animation usually consists of much more strokes. Therefore, the emphasis of this paper is on substantially improving on these techniques by managing these numerous brushes in a proper and intu-

itive way so that artists do not need to worry about mistakes being made while editing, or brushes suddenly popping up during the animation.

Following subsections successively describe the modelling and manipulation of brushes, the animation of brushed characters and some higher-level tools to manipulate drawings.

### 3.1 Modelling/Drawing and Manipulating Individual Brushes

Our basic aim in this context is to support the animator in the creation and manipulation of the strokes representing animation characters. Also, we want to provide a user interface that is much more friendly than the standard 'point-click-and-drag' metaphors.

The creation of a stroke is done interactively by sampling a (not necessarily) pressure sensitive stylus along the trail of the stroke. In order to allow for real-time high-level manipulations on the strokes, the individual pixels that make up the character are not used. Instead, a high-level internal representation, using cubic Bézier splines, is made. This happens as follows. While sampling the pressure sensitive stylus we simultaneously perform an iterative curve fitting technique based on least-squared-error estimation. Existing curve drawing solutions mostly take recourse to a 'batch' curve fitting solution, in which the fitting is done after the stroke drawing is finished, whereas our fitting is done on-the-fly while the curve is being drawn.

Once the underlying curve is created, it can be edited. Conventional interaction metaphors are almost always based on explicitly 'point-click-and-drag' the control points. We use the solution of [Vansichem et al. 2001] that mimics the sketching method used by artists when sketching strokes with a pencil on paper: the stroke is edited upon by moving the pressure sensitive stylus alongside (a part of) an existing stroke. These movements are sampled and are consequently interpreted as an attractor transforming the underlying curve representation of the targeted stroke, reflecting the user's intentions.

Manipulations on the splines are actually performed on their control points, but this happens transparently for the user. The control points are never shown in the user interface.

The splines themselves are not drawn as solid lines; instead the animator can choose from a wide range of stylised brush tools including a solid brush tool (figure 2(a)), a paintbrush tool (figure 2(b)) and an airbrush tool (figure 2(c)).

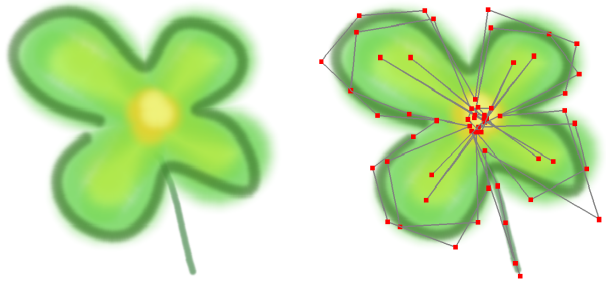

(a) (b) (c)

Figure 2: Different kinds of brush tools. a) Solid brushes. b) Paintbrushes. c) Airbrushes.

Figure 3 depicts a flower modelled with our system using different kinds of brushes.

Aside the creation and editing of free-form strokes, we also added support for performing affine transformations upon (selections of) strokes.

Notice that the animator does not have to worry about picking, clicking and dragging control points associated with the underlying curve primitives. That way we preserve the same freedom of painting as when painting on paper.



(a) (b)

Figure 3: a) Picture of a flower created with our system. b) The same flower, depicted with all underlying control points.

### 3.2 Animating Brushes

In traditional animation [Blair 1994; Patterson and Willis 1994], the drawing/animation process can be broken down into three substages: (i) main animators draw the most significant images, which are referred to as extreme frames or poses, containing the major features of the action; (ii) assistant animators produce key frames between the extreme frames, hence detailing the desired animation action; while (iii) less experienced animators are responsible for creating all the remaining in-between frames of the animation.

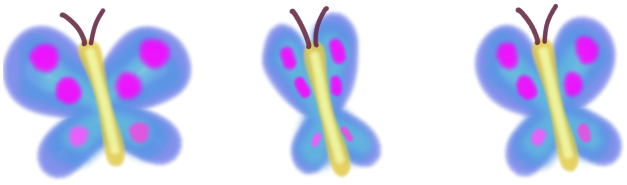
In prior work we defined a framework for automatic in-betweening [Di Fiore et al. 2001]. This is implemented as a multi-layered system starting with basic 2D drawing primitives at level 0. Level 1 manages and processes explicit modelling information and is fundamental in the realisation of transformations outside the drawing plane. That is, characters and objects are modelled as sets of depth ordered primitives with respect to the $X$-axis (horizontal) and $Y$-axis (upstanding) rotations. For each set of 'important' $x$-$y$-rotations of the object/character relative to the virtual camera, the animator draws a set of ordered primitives, functionally comparable to the extreme frames in traditional animation [Blair 1994; Patterson and Willis 1994]. The following level offers the opportunity to include higher-level manipulation tools.

This paper builds further on this framework as follows. The basic 2D drawing primitives at level 0 are in fact the on-the-fly created cubic Bézier splines of previous section whereas the extreme frames are sets of modelled brush strokes that make up a character.

Once the animator has created the extreme frames, he only has to specify key frames in time by entering parameters using the same methods as described in [Di Fiore et al. 2001]. Afterwards, the automatic in-betweening method comes into play and generates the desired animation.

As a result, convincing 3D-like animations starting from pure 2D painted drawings can be made.

Figure 4 shows some extreme frames and an in-between frame of an animation of a butterfly.



(a) (b) (c)

Figure 4: a–b) Extreme frames of an animation of a butterfly. c) Generated in-between frame.

Unlike purely 3D based approaches, our animation still has many lively aspects akin to 2D animation and also preserves frame-to-frame coherence. A rigid 3D look is avoided through varying brush thickness, brush pressure, brush opacity, edge softness and so on (depending on the chosen brush tool) which enable the animator to create subtle outline changes that are either impossible or hard to achieve utilising 3D models.

## 3.3 Manipulating a Drawing

The techniques described in the previous section are all modelling and animation tools that work locally on a single brush stroke. In this section, we will introduce some tools that enable the animator to manipulate the drawings on a higher level.

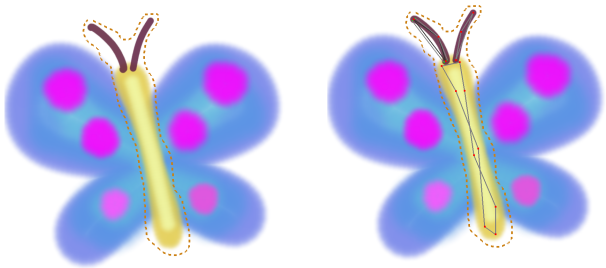
We successively describe a grouping tool, transformation tools, a deformation tool and a hierarchical structure. Each of these tools can operate both on the whole drawing as well as on a user selected part of the character.

### *3.3.1 Grouping*

The grouping tool acts as the starting point of all higher-level tools that manipulate the drawing. Basically, this tool groups together some of the control points that make up the brushes, contrary to other applications that simply group the selected pixels. As will be clear from the following subsections, our approach is preferable.

Since the underlying mathematical representation of the brush strokes needs to be hidden for the user, grouping is done by drawing a selection lasso over the part of the character one wants to change. After selecting, the control points within the lasso will be marked as a group and can be manipulated as will be explained in following sections.

Figures 5(a–b)) show the grouping tool in action. For visualisation purposes we depict the tool without (a) and with (b) the selected control points.



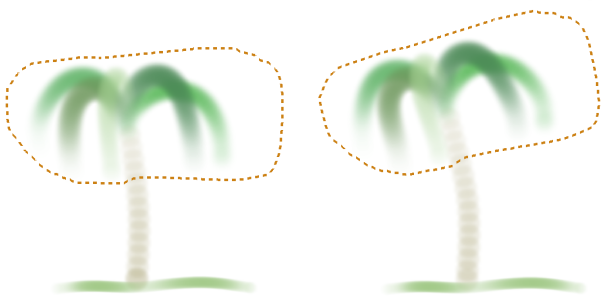
(a) (b)

Figure 5: Grouping tool in action. a) Upon selecting (no control points are shown). b) Upon selecting (only the relevant (i.e. currently selected) control points are displayed).

We provided also the functionality to save and restore groups for reuse in the current or other key-frames. Also, control points can be restricted to be exclusively part of only one group.

### *3.3.2 Transformations*

We also added support for performing transformations on parts of the drawn character. Existing applications transform drawings on a per-pixel basis which results in artifacts because the transformed parts are cut out and then pasted at a new position. In our case the transformation tools ( translate, rotate, scale, ...) only affect the control points selected by the grouping tool and so the animator has local control over the drawing while preserving the continuity and connectivity of brushes. This is illustrated in figure 6. As one can see, rotating the leafs of the palm tree does not create a hole between the rotated and fixed part. Instead, the brushes that make out the stem of the palm tree stay connected to the leafs and are even lengthened in order to preserve the continuity.



(a) (b)

Figure 6: Transformation tool in action. a) Untransformed palm tree. b) The same palm tree after rotating a selected part of the image.

### *3.3.3 Deformations*

In this section we explain the use of a free-form deformation tool as a means to easily and rapidly manipulate (deform) a selected part of the character in a free-form manner.

Existing free-form deformation applications usually impose an evenly spaced $N$-dimensional grid of deformation control points $G_{d_1,..,d_N}$ on the character. Then, the user has to manipulate these deformation control points in order to deform the character. For the 2D case this deformation grid always is rectangular.

This way of deforming imposes major constraints on the animator's way of working. First, manipulating the deformation control points is not intuitive at all since it is hard to estimate how the movement of a single control point will affect the character. Second, because of the rectangular shape of the deformation grid one only can deform rectangular parts of the object and so it is difficult to have local control. However, the most important issue is that existing free-form deformation applications are pixel based and do not take the underlying control curves into account. As a result, connectivity and continuity of the brushes may be lost.

For the second issue, in our approach, first the grouping tool is used to select the part of the character to be deformed. Hence, all control points belonging to the selected group are marked as deformable. Then, our system calculates the bounding box surrounding the deformable control points. The resulting bounding box now acts as deformation grid of which each dimension $d$ gets divided in $X_d$ cells which is a user adjustable parameter. The advantage of this approach is that only the selected part of the image (enclosed by the grouping tool) is eligible for deformation, despite the rectangular shape of the deformation grid. As a result, the animator is assured of having local control. Also, taking into account the new positions (due to the deformation) of the control points all affected brushed are regenerated and therefore the connectivity and continuity are preserved.

Regarding the first issue, [Di Fiore and Van Reeth 2002] presented a sketching tool that enables the animator to intuitively perform free-form deformations just by drawing two strokes: one that represents the initial state of the object, and a second one that indicates how the object should be deformed. The same method can be employed to manipulate our previously generated deformation grid.

As a result, the combination of the grouping tool and generation of a deformation grid together with the use of an intuitive sketching tool makes the direct manipulation and visualisation of the brush's control points and deformation grid superfluous: all of this is hidden and happens transparently for the user. The animator only has to select a part of the character and to draw the source and deformation strokes.

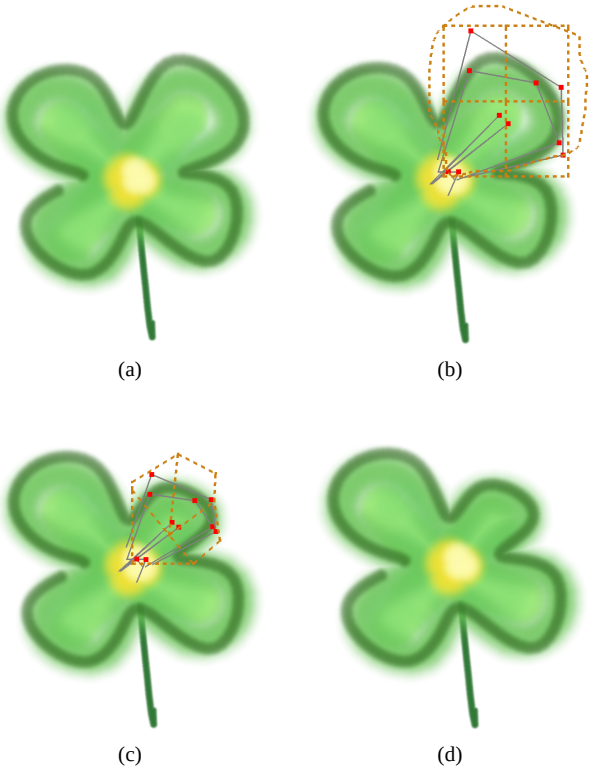



Figure 7: Free-form deformation tool. a) Before deformation. b) After selecting a part to be locally deformed (we also displayed the lasso tool, the control points that will be affected by deformations, as well as the deformation grid). b) After deformation (with deformation grid and control points shown). d) After deformation. Note that the deformation has only affected the selected part of the image while preserving the connectivity and continuity of the brushes.

Figure 7 gives an example of a free-form deformation of a flower's leaflet. For information purposes we also depicted the selected control points as well as the generated deformation grid. As once can see in figure 7(c) our deformation tool has only affected the selected part of the image while preserving the connectivity and continuity of the brushes.

#### *3.3.4 Hierarchical Structures*

We can interconnect several groups by defining an anchor point for each group. This allows the user to create a hierarchical model of a character drawing. Such a model can be used for animating lively characters. Obviously, this kind of control can also be used in combination with forward and inverse kinematic tools or other physics based approaches to control the movement of a character in a more physical way.

## 4 RESULTS

We have used the approach of our paper on some examples. For demonstration reasons, the following examples only address motion seen from the same viewpoint. However, our approach is just as well suitable when the camera is tilted or when the objects turn away from the camera.

Figures 8 and 9 show some snapshots of the animation of a talking man. As one can see, our system easily lends itself to different kind of painterly styles. Figure 8 is painted in a more 'tense' style while figure 9 gives off more subtlety.

For the animation depicted in figure 10, we used the extreme frames of figure 1 to generate a palm tree being blown by the wind.

As can be derived from our examples, our approach enables artists to create visually appealing painted animations without worrying about brushes suddenly popping up in successive frames. Furthermore, since no constraints are imposed on the artist, we preserve the same freedom of painting as when using traditional media.

## 5 CONCLUSIONS AND FUTURE WORK

In this paper, we presented a novel approach to assist the animator throughout the time-consuming process of traditional painted animation. To be more precise, the creation, manipulation, managing and animation of the numerous strokes that are painted one over the other to make up the characters. At the same moment we needed to preserve the freedom of an animator to express the specific artistic style he is bearing in mind.

We accomplished these goals by breaking down the traditional drawing process into a modelling process and an animation process. The modelling process is used to create and manipulate the extreme poses of the character while the animation stage is employed to provide for frame-to-frame coherent animation. We also presented higher-level tools that enable to locally and globally control user selected parts of the drawings.

We believe this work is significant for its novel contribution to computer-assisted traditional animation since it empowers the production of fluent painted animation while not hampering the animation artists' creativity.

From an artistic point of view, the most interesting future work is the exploration of new high-level tools for manipulating brush strokes, drawings and even parts of animations.

## ACKNOWLEDGEMENTS

We gratefully express our gratitude to the European Fund for Regional Development and the Flemish Government, which are kindly funding part of the research reported in this paper.

Part of the work is also funded by the European research project IST-2001-37116 'CUSTODIEV'. In this context, we would like to express our thanks to the various people in the project providing ideas and for helping with the implementation.

Many thanks to Johan Claes for the valuable paper reviews and to 'Xemi' Morales and Bjorn Geuns for their artistic contributions.

Furthermore, we would like to acknowledge Marc Flerackers for helping us implementing the software.

## REFERENCES

BLAIR, P. 1994. *Cartoon Animation*. Walter Foster Publishing Inc., ISBN: 1–56010–084–2.

DI FIORE, F., AND VAN REETH, F. 2002. A multi-level sketching tool for pencil-and-paper animation. In *Sketch Understanding: Papers from*

*the 2002 American Association for artificial Intelligence (AAAI 2002) Spring Symposium. Technical Report SS–02–08*, 32–36.

DI FIORE, F., SCHAEKEN, P., ELENS, K., AND VAN REETH, F. 2001. Automatic in-betweening in computer assisted animation by exploiting 2.5D modelling techniques. In *Proceedings of Computer Animation 2001*, 192–200.

GOOCH, B., AND GOOCH, A. A. 2001. *Non-Photorealistic Rendering*. A. K. Peters Ltd., ISBN: 1568811330.

HAEBERLI, P. 1990. Paint by numbers: Abstract image representations. In *Proceedings of SIGGRAPH 1990*, vol. 24(4), ACM, 207–214.

HERTZMANN, A., AND PERLIN, K. 2000. Painterly rendering for video and interaction. *Symposium on Non-Photorealistic Animation and Rendering (NPAR2000)* (June), 7–12.

HERTZMANN, A. 1998. Painterly rendering with curved brush strokes of multiple sizes. In *Proceedings of SIGGRAPH 1998*, ACM, 453–460.

HERTZMANN, A. 2001. Paint by relaxation. In *Proceedings of Computer Graphics International (CGI 2001)*, 47–54.

KALNINS, R. D., MARKOSIAN, L., MEIER, B. J., KOWALSKI, M. A., LEE, J. C., DAVIDSON, P. L., WEBB, M., HUGHES, J. F., AND FINKELSTEIN, A. 2002. WYSIWYG NPR: drawing strokes directly on 3D models. In *Proceedings of SIGGRAPH 2002*, ACM, 755–762.

KAPLAN, M., GOOCH, B., AND COHEN, E. 2000. Interactive artistic rendering. *Symposium on Non-Photorealistic Animation and Rendering (NPAR2000)* (June), 67–74.

MEIER, B. J. 1996. Painterly rendering for animation. In *Proceedings of SIGGRAPH 1996*, vol. 25(4), ACM, 477–484.

PATTERSON, J. W., AND WILLIS, P. J. 1994. Computer assisted animation: 2D or not 2D? *The Computer Journal 37*, 10, 829–839.

STROTHOTTE, S., AND SCHLECHTWEG, S. 2002. *Non-Photorealistic Computer Graphics. Modeling, Rendering, and Animation*. Morgan Kaufmann Publishers, ISBN: 1-55860-787-0.

VANSICHEM, G., WAUTERS, E., AND VAN REETH, F. 2001. Real-time modeled drawing and manipulation of stylized cartoon characters. In *Proceedings of the IASTED International Conference on Computer Graphics and Imaging*, IASTED, 44–49.

WILLIAMS, R. 2001. *The Animator's Survival Kit*. Faber and Faber Limited, ISBN: 0–571–20228–4, 3 Queen Square London WC1N 3AU.

## About the Authors

Fabian Di Fiore is a Ph. D. student of computer science at Limburgs Universitair Centrum, Expertisecentre for Digital Media (EDM). His contact email is fabian.difiore@luc.ac.be.

Frank Van Reeth is a professor of computer science at Limburgs Universitair Centrum, Expertisecentre for Digital Media (EDM). His contact email is frank.vanreeth@luc.ac.be.

Figure 8: Animation of a talking man drawn in a ‘tense’ style.

Figure 9: More ‘refined’ version of the animation depicted in figure 8.

Figure 10: Some snapshots of an animation sequence generated from the extreme frames in figure 1.