Ibero-American Symposium on Computer Graphics - SIACG (2006)
P. Brunet, N. Correia, and G. Baranoski (Editors)

Perception-based painterly rendering: functionality and interface design

S. Nunes[1], D. Almeida[1], V. Brito[1], J. Carvalho[1], J. Rodrigues[2] and J.M.H. du Buf[1]

[1]University of Algarve, Vision Laboratory – FCT, Faro, Portugal,
[2]University of Algarve, Escola Superior de Tecnologia, Faro, Portugal,

Abstract

Painterly rendering (non-photorealistic rendering or NPR) aims at translating photographs into paintings with discrete brush strokes, simulating certain techniques (im- or expressionism) and media (oil or watercolour). Recently, our research into visual perception and models of processes in the visual cortex resulted in a new rendering scheme, in which detected lines and edges at different scales are translated into brush strokes of different sizes. In order to prepare a version which is suitable for many users, including children, the design of the interface in terms of window and menu system is very important. Discussions with artists and non-artists led to three design criteria: (1) the interface must reflect the procedures and possibilities that real painters follow and use, (2) it must be based on only one window, and (3) the menu system must be very simple, avoiding a jungle of menus and sub-menus. This paper explains the interface that has been developed.

Categories and Subject Descriptors (according to ACM CCS): J.5 [Computer Applications]: Arts and Humanities; I.3.3 [Computer Graphics]: Picture/Image Generation; H.5.2 [Information Interfaces and Presentation]: User Interfaces

1. Introduction

Non-photorealistic rendering (NPR) has become a mature research area, with numerous approaches to painterly rendering on the basis of discrete brush strokes. For a recent NPR taxonomy and a review of stroke-based techniques see [Sou03, Her03]. Most rendering techniques are based on image analysis algorithms from computer vision, see for example [GCS02], although there is a tendency towards including aspects of human vision, for example in constructing saliency maps [DS02]. In contrast, our own, recently developed technique [dBRN*06] is completely based on human vision. It employs four models of processes in our visual cortex: (1) colour constancy, (2) coarse background level construction in brightness perception, (3) multi-scale representation of lines and edges, and (4) saliency maps based on multi-scale keypoint detection. The rendering engine (in OpenGL) is now being complemented with a user interface.

In order to be successful, the interface must be suitable for a very wide audience, because digital cameras of good quality have become ubiquitous in the last years. In addition, the use of email and PCs has become normal routine for many people. The exchange of snapshots may be common practice, but many people also experiment with special effects, for example when producing electronic or printed invitations, flyers and webpages. A real challenge is to create software which allows to automatically translate photographs into paintings with real brush strokes, in oil, watercolour or wax crayon, and with styles that approximate impressionism, expressionism or cubism. By offering the options mentioned above, with many parameters to select, users can play with them and they can (re)discover the *homo ludens* within them.

Standard software packages such as Adobe's Photoshop, Corel's Paint Shop Pro and GNU's GIMP provide many built-in features by means of a jungle of menus and submenus. However, most people, and especially children, do not have the necessary knowledge to experiment with all options in order to obtain the best effects. As a result, only a few basic features are used by trial-and-error, and the "undo" option that returns the tool to a previous state is used most often. Our aim is to develop an interface which is very simple and intuitive, but in combination with a rendering tech-

nique that represents the state-of-the-art in NPR, in our case painterly rendering.

Two different interfaces are used by Gertrudis (www.gertrudisgraphics.com) and Virtual Painter (www.vpainter.com). Gertrudis allows the user to select parameters (colours, brushes) and then the user can apply brush strokes in regions that are interactively selected with mouse and cursor. Virtual Painter is automatic, because it offers a set of pre-defined painting styles (virtual painters), but together with parameters that the user can change. As for Virtual Painter, we aim at automatic production of paintings, but with many more parameters and possibilities that allow to control and influence the painting process. Our own solution is different, because (a) it is not limited to a set of pre-defined styles, (b) it offers sets of options in menus which reflect the logical steps as followed by real painters, and (c) the user can develop optimised styles by customising option parameters. The last two aspects are perhaps most important, because the user can undergo a training process, much like the one real painters underwent.

The educational aspect became important when we asked artists and non-artists about their preferences and requirements, assuming the possibility of commercialising the software in the future. Non-artists were more concerned about special effects (we call this the colour circus), because they want to obtain nice results fast, if possible automatically after only two or three mouse clicks. Artists, on the other hand, emphasised completely different aspects: working with a real palette with not too many pigments, using brushes with different shapes and bristles, preparing first a surface with a constant colour, then painting a coarse background with a big brush and finish important structures with smaller brushes. The infrastructure of the latter procedure was already—unconsciously or intuitively—included in our rendering scheme. There were two aspects where artists and non-artists agreed: the menu system must be extremely easy and intuitive, without a jungle of (sub)menus, and the interface must be based on one window, rather than many windows that pop-up and must be closed again. Apparently, most were not quite impressed by the user-friendliness of Paint Shop Pro and similar products. The user interface that is being developed tries to combine all aspects mentioned above.

It should be stressed that we do not aim at developing a tool for professional painters. Professionals working with oil paint and other media are not likely to accept artificial tools, however sophisticated they may be. A completely interactive scheme like IMPaSTo [BWL04] uses a stylus and tablet or a haptic interface as a virtual brush. This state-of-the-art technique may only be suitable for painters who are willing to undergo a new training process, in order to achieve the same level of interactivity and expressivity that they can achieve with the real media. Nevertheless, in the future we may apply IMPaSTo's advanced modelling of real oil and acrylic paint, but in the context of (semi-)automatic painterly rendering.

The rest of this paper is organised as follows: Section 2 presents a brief introduction to the image-analysis and rendering engines. Section 3 explains the logical steps of the painting process. The user interface, both window and menu design, are explained in Sections 4 and 5. In the final Discussion (Section 6) a summary is presented along with ideas concerning future extensions.

2. Image-analysis and rendering engines

The method has been described in detail in [dBRN*06]. Image analysis is separated from the rendering process, because the analysis cannot (yet) be done in realtime—that is, in a few seconds necessary for fast interactivity—whereas the rendering is much faster. Image analysis is therefore done by means of a pre-processing program that the user must apply to the images to be rendered, which takes a few minutes per image. After that, the image file is complemented with files that contain information about the local image content: detected lines and edges at different scales (level of detail), their positions, orientations and local contrast. Basically, the output consists of coordinate lists, which serve to apply object-related brush strokes where the size of the brush is coupled to the scale of image analysis. This is called the *foreground process*. A *background process* is normally necessary, because there often are homogeneous image regions (in sky etc.) where no lines and edges can be detected. For the latter process two files are prepared: the local contrast for modulating the pressure of brush strokes, and the local dominant orientation for steering the strokes. This information originates from detected lines and edges in non-homogeneous areas, but it is spread into homogeneous areas by lowpass filtering and a diffusion process. All files, except for the one with the input image, are hidden and the user cannot access them.

The rendering engine starts with the background process, applying for example random strokes with a big brush. For each stroke a colour is picked in the input image, at the stroke's centre point. After completion of the background process, the foreground process applies brush strokes at positions where lines and edges have been detected, from coarse scales (big brushes) to small scales (small brushes). Each coordinate list can be rendered as one stroke, but long lists can be split into smaller ones for getting discrete strokes with a pre-defined length. As for the background process, for each stroke a colour is picked in the input image.

The rendering of back- and foreground strokes is the same: coordinate lists are used to create triangle lists, which are rendered with the picked colour in OpenGL. The size of the triangles is determined by the selected brush size. The brush type is defined by opacity maps: in the case of "spray" this map is about elliptic with a gradual decay towards the

edge; in the case of "oil" the opacity maps are constructed by random combinations of sets of heads, bodies and tails of real, oil-painted brush strokes that have been digitised.

Figure 1 shows an input image (at top-right) and the background process using random strokes (left column). The three backgrounds in the right column were rendered with (randomised) vertical and horizontal strokes; and with horizontal-vertical crisscrossing. We note that the peak of the turret is not rendered when using horizontal strokes, because such strokes will introduce wrong colours at both sides and the forground process (with vertical strokes) may not be able to correct them. Figure 2 illustrates final results obtained with different palette settings. The left column shows, top to bottom, the use of all colours and the effect of limiting the gamut to 27 colours. The right column shows the effects of increasing and decreasing saturation, colour rotation in HSV space and brightening. For more results see also [dBRN*06].

Figure 1: *Left column: background rendering with random oil strokes. Right column: input image and backgrounds created with vertical and horizontal strokes as well as horizontal-vertical crisscrossing.*

3. Functionality

Our algorithms aim at automatic production of paintings, in contrast to other solutions like Gertrudis and the more common mouse/cursor-controlled drawing and filling of regions in PSP and GIMP. Detected lines and edges are automatically translated into discrete brush strokes and applied to the "canvas," i.e. the foreground process. In regions where no lines and edges have been detected, a background must be created by the background process. The user can decide not to paint all foreground strokes, even not to cover the entire canvas with background strokes. In this case the surface to be painted can be prepared with any colour. The user can decide to paint the fore- and background with different palettes and brushes. It must be possible to stop the painting process, to change parameter settings (colour manipulations, brush and stroke types), and to resume the process. Although the user can work with pre-defined and customised stylefiles, for example a default style which is loaded at startup, the functionality in terms of menus and their parameter options reflects the practice of a real painter with the following logical steps:

1. Select a smooth surface, or a textured canvas or paper, and prepare the surface with one colour. This "grounding" may disappear—or not—after painting the back- and foreground "layers."
2. Prepare the palette, i.e. define the colour gamut, for example with unsaturated colours for making a pastel, watercolour or gouache, or with an emphasis on red-orange for obtaining a "warm" effect.
3. Select a brush type, which can include traditional ones (for oil paintings, pastels, wax crayons) but also more recent possibilities like spray, felt pen and colour marker. Along with the brush type, the size can be selected.
4. Decide which strokes are going to be applied, for example short, long, straight or curved, and in the latter case the symmetry for circular or S-shaped strokes.
5. Decide how to paint, from very fast (wet-in-wet with colour mixing) to slow such that paint can dry before new strokes are applied (wet-on-dry).
6. Create a background, using only a big brush but then perhaps also a smaller one, with approximately horizontal strokes or diagonal crisscrossing à la Bob Ross. The painted background can be complete, in which case the

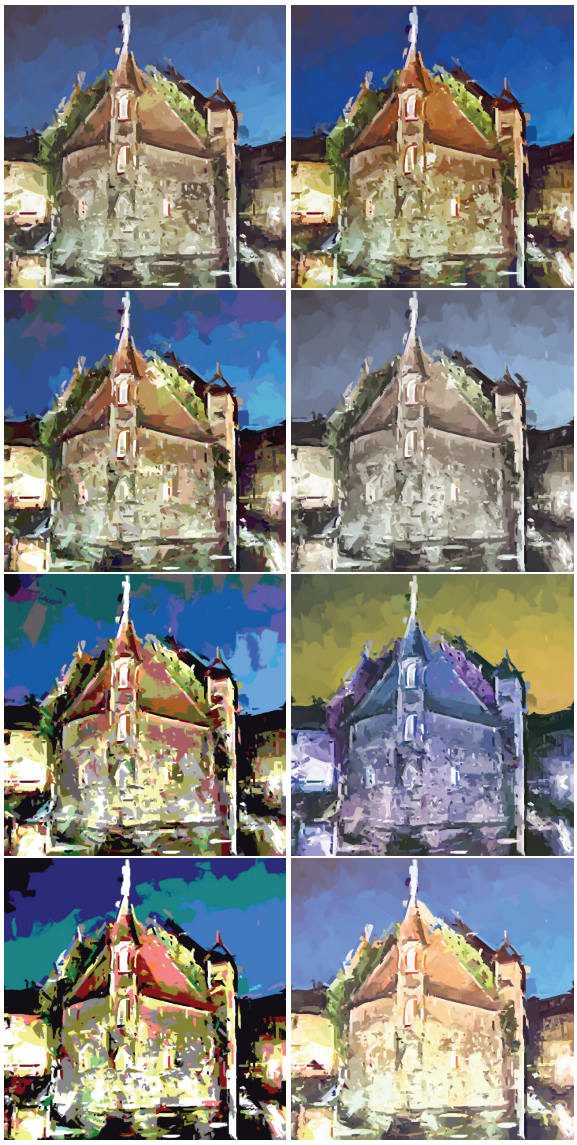

Figure 2: *Final results obtained by limiting the colour gamut (left column) and special effects (right column): increased and decreased saturation, rotated colour space and brightening.*

colour of the prepared surface will disappear, or it can be incomplete with a certain density.

7. Then create the foreground with smaller brushes, in order to accentuate important structures that must be recognised in the final painting. Normally, foreground brush strokes are painted with the orientation of detected lines and edges, but for obtaining expressionistic and cubistic effects they can be rotated—partially or completely—to horizontal, vertical and diagonal orientations.

As in real practice, it is possible to change selections during the painting process. For example, it is possible to apply mixed media, creating a background in one style and then a foreground in another style, like pen and black ink on top of a watercolour to accentuate precise edges in a diffuse, unsaturated background. The user must develop experience in order to realise a style that suits the application. The user can use and develop stylefiles, for example for obtaining impressionistic or expressionistic effects in oil or watercolour, and can optimise the parameters. In combination with future extensions this process may lead to higher-level stylefiles for simulating a late van Gogh or a pointillist Seurat.

4. Window design

The application window must show the input image, the "canvas" during and after the painting process, and necessary user menus. In addition, colour manipulations of the input image must be shown such that the user can see what the final colour impression will be. This is solved by applying `palette` settings to the input image. The main `control` menu must be visible at all times, whereas other menus like `palette` and `surface` must be present only when necessary. We therefore chose to show, in addition to the `control` menu, a permanent `menus` menu in the first column. Selected menus are shown in the second column: after clicking on a menu, it will be shown at the top. After clicking on another menu, this will be shown below the first one. If the first menu is clicked again, it will be removed and the second one will move to the top. If there is not sufficient space for a new menu in the second column, the LRU (least recently used) menu will be removed. For this purpose, each opened menu receives a timestamp that will be updated each time that it is used. The menu with the oldest timestamp will be taken out.

Figure 3 shows a screenshot of the entire window, and Fig. 4 only the two menu columns. A small version of the image to be painted is shown in the top-left corner. It is part of the first column with the permanent `control` and `menus` menus. Selected menus can be seen in the second column. Colours of the window (background, borders) can be customised by the user, using standard Linux and Windows desktop tools. The top-right corner shows normal symbols to iconise, maximise and close the window (the window can also be resized using mouse and cursor).

5. Menus

In this section we describe the menus (indicated in `type font`) and their functions (buttons and rulers etc. in *italics*). As mentioned above, only the `control` and `menus` menus are permanent and shown in the left column, whereas all other menus can be activated and will be shown in the second column, however with LRU replacement if there is not enough space.

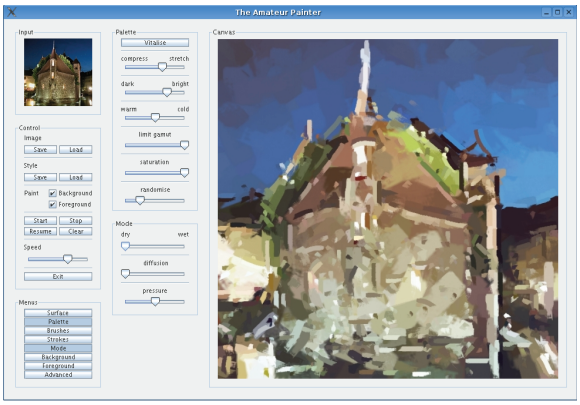

Figure 3: *The interface with input image (top-left) and canvas (right).*

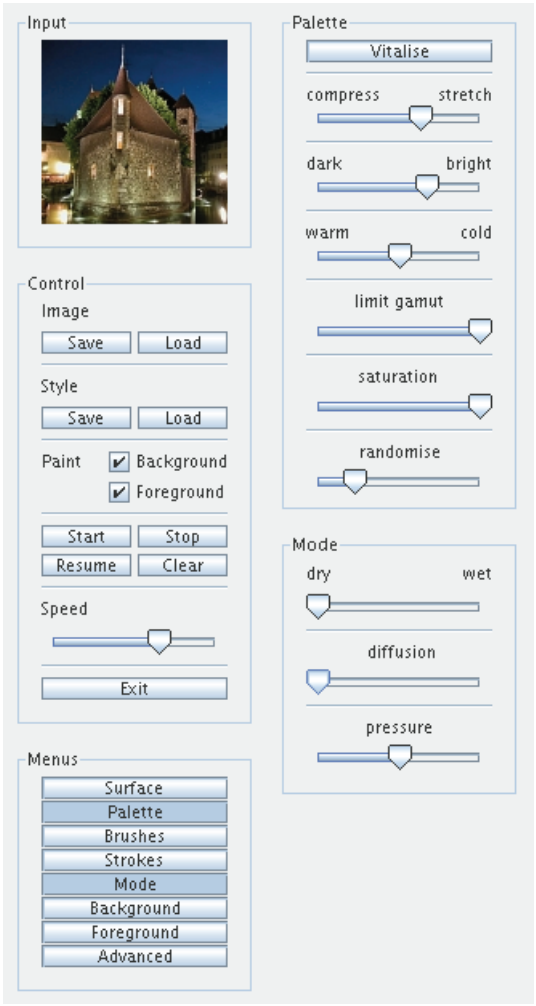

Figure 4: *The permanent* `control` *and* `menus` *menus occupy the leftmost column, whereas selected menus are displayed only in the second column with LRU replacement.*

5.1. CONTROL

IMAGE options *load* and *save* open an external browser window for changing the default directory, filtering file types (tiff, jpg, etc.), selecting or modifying file names, and to load an image file or to store a painted canvas.

STYLE options *load* and *save* do the same as IMAGE options, but concern parameter files. At startup, a default parameter file will be loaded and the menus will show the parameters. After experimenting with parameters, a user can save them in a stylefile for later use.

PAINT options comprise two on/off buttons *background* and *foreground*, which enable or disable the painting of these, the trivial command buttons *start*, *stop*, *resume* and *clear*, and a slider *speed* to delay the painting of brush strokes. Together with options from other menus like `background`, these controls allow the user to change colour and other options during the painting, with the possiblity to save results for later use. In order not to oblige the user to re-position the mouse and cursor each time that the painting process is stopped and resumed, short-cuts can be used: for example, the left mouse button is used for selecting menus, clicking on buttons and sliders, whereas the right mouse button can be reserved for stopping and resuming the painting process. This significantly increases user friendliness. The *exit* button will close the window.

5.2. MENUS

This menu shows the buttons for activating the other menus: *surface* for selecting a canvas or paper texture and colour, *palette* for manipulating colours of brush strokes, *brushes* for selecting the brush type (oil, felt pen, etc.), *strokes* for modifying the type of stroke (length, curvature), *mode* for controlling the balance between painting wet-on-dry and wet-in-wet, *background* with selections for creating different backgrounds and the similar *foreground*. The *advanced* menu offers options for special actions, such as stacking and mixing images. These menus are explained below.

5.3. SURFACE

This menu allows to select the type of surface with *canvas* and *paper* buttons together with a *fine–coarse* slider to define texture scale. If neither canvas nor paper is selected, the surface will be smooth. The *craquelure* and *age* sliders serve to simulate cracked varnish and yellow/brown dirt on the final painting (these have not yet been implemented and do not belong in this menu; they may disappear in the beta version). Three rulers *colour*, *saturation* and *lightness* allow to define

a global background colour—in HSV space—in the case that the user wants to limit the number of the brush strokes in the `background` and `foreground` menus. The resulting colour is shown in a separate, small *colour* window. After clicking the *apply* button the canvas will be coloured. In the current version, textures are applied after painting all brush strokes, using global opacity maps. This is not very realistic in the case of making a watercolour on textured paper, and requires further research into fast approximations of the physical processes; see also Discussion. The *load file* button allows to initialise the surface by using an image file. This image may differ from the one to be painted, it may be the one to be painted but after colour manipulations, or it can be an already painted image. The latter is useful in the case of using mixed media: for example a foreground with pen and black ink on top of a watercolour.

5.4. PALETTE

This menu has the following options: The *vitalise* button allows to normalise the dynamic ranges of R, G and B values to the interval 0–255, which often produces more vivid colours. This approximates the last step in the ACE model of colour constancy [RGM04], but it can destroy the mood of the photograph (a reddish sunset gets a blueish sky; see Fig. 19 in [dBRN*06]). The *compress–stretch* slider can be used to adjust the dynamic range, with or without normalisation. The *dark–bright* slider controls V in HSV space. The *warm–cold* slider allows to apply a shift of all colours in the direction of red-orange (warm) or blue-green (cold). The *limit gamut* slider serves to select all colours or a smaller number. In the current version this is a quantisation of the RGB colour cube into equally-sized blocks, with 8, 27, ... 125 blocks (125 can hardly be distinguished from using all colours). In a future version, this option may be replaced by a sub-menu for advanced features (see Discussion). The *saturation* slider (S in HSV) is typically used to "dilute" colours for obtaining whitish pastel and watercolour effects. The *randomise* slider allows to introduce random variations of stroke colours.

5.5. BRUSHES

This menu shows implemented brush strokes, in the current version: *oil*, *felt pen* and *spray*. In future versions these will include pastel, wax crayon, colour marker, pencil, ink pen and oriental brush. For the pencil option see also [YMI04] for advanced rendering techniques.

5.6. STROKES

The default brush strokes are about linear with a pre-defined lenght. This menu allows to change the shape: the *lenght* ruler can be used to create very short dashes, circular or rectangular strokes (for pointillism) and long strokes. In the case of long strokes, their shape can be influenced by changing the curvature with the *symmetry* and *curvature* rulers, producing strokes that look like $\sim$ or $\frown$, plus the *taper* ruler for changing the default "ellipsoidal" shape into tapered shapes which are narrow on the one side and broad on the other. The *default* button resets the parameters.

5.7. MODE

This menu allows to influence the brush strokes with the following options: the *dry–wet* slider adjusts the way that new brush strokes will be mixed with previous ones. Dry implies that new colours replace old colours, whereas wet means that a new colour is mixed "50/50" with old ones. This option is also useful for simulating drying paper in the case of making a watercolour. The *diffusion* slider can be used to apply a diffusion effect, again for simulating watercolours. Isotropic diffusion is applied, which is not very realistic, but see Discussion. The *pressure* slider modulates brush size: if the contrast along a line or edge varies from small to big to small, a bigger brush will be used in the centre and a smaller one at the two sides. This option works best when using spray and felt-pen brush types, and will be useful in the future after implementing drawing with e.g. pencils. The interaction with global opacity maps that simulate canvas and paper textures requires further research in order to obtain more realistic effects: leaving the texture more "open" with small pressure and "filling in" the texture with more pressure.

5.8. BACKGROUND

This menu offers the following options for creating different backgrounds: the *one* and *two* buttons imply the use of one or two, perpendicular stroke orientations, the latter for "criss-crossing." The *orientation* slider adjusts the orientation to any angle, whereas the *randomise* slider can be used to add small or big random deviations; when the slider is pulled to the right, the background brush strokes will be completely random. In addition to the *randomise* slider, the *slow–wild* slider can be used to control the "wildness." Basically, it sets the type of distribution, from triangular and Gaussian with a predominant orientation to uniform with equal orientation probabilities. The *1/2–auto* slider adjusts stroke orientations between the ones selected above and automatic orientation detection. The *density* slider can be used to limit the number of brush strokes, for example in the case that a user wants to create a background with few strokes on a coloured surface, or, at the other end, to overpaint the background such that many small parts of previously painted strokes remain. Finally, the *big*, *medium* and *small* buttons allow to select one or more brush sizes. Normally, a background is created by only using a big brush, but smaller brushes can be added later. Using increasingly smaller brush sizes approximates painterly rendering on the basis of multi-scale computer vision [Her98], in which case the user can decide *not* to paint the foreground.

5.9. FOREGROUND

This menu controls the way that detected lines and edges are painted. Five on/off buttons from *big* to *medium* to *small* allow to select the brush sizes that are linked to the scales at which the image is analysed. The *density* slider is used to paint all detected lines and edges or a randomly selected subset. Each brush stroke is strongly connected to the local orientation of object contours etc., but the *auto–cubism* slider allows to rotate the orientations of the strokes. In order to circumvent the problem of rotating strokes too much, which would cause e.g. horizontal strokes at vertical edges, four orientations can be emphasised (horizontal, vertical and the two diagonal). Pulling the slider to the right implies that all strokes are converted to these orientations.

5.10. ADVANCED

This menu will offer at least two sub-menus in the future: *user colours* for further customising the palette, for example by selecting a set of specific colours that can approximate real pigments (see also Discussion), and *image mixing* for combining (mixing or stacking) different images. Stacking refers to putting one painted canvas on top of another, where the user must select the colour that will be transparent.

6. Discussion

Default settings have been chosen such that, after loading an image and activating the *start* button in the `control` menu, the user sees the painting in progress and can already develop a feeling of what is happening. In about 15 minutes most options can be tried and the corresponding results observed. Interactivity is very efficient, because changing one parameter may require only two mouse clicks when the parameter menu is already shown (one slider, start) or three when it is not present (menu, slider, start). In addition, shortcuts can be defined such that basic control functions (start, stop etc.) and other important functions are available on the keyboard, which allows the user to work with two hands simultaneously.

The interface uses one configuation file (in ASCII) that allows to define keyboard short-cuts, but also which stylefile is loaded at startup and (not necessarily) one or two option menus that will be initialised in the second window column. Stylefiles are also defined in ASCII, such that the user can modify parameters interactively using the interface, or edit parameters using any editor. The interface of the rendering engine allows for scripting, such that many images or video frames can be processed automatically. However, this is rather slow because of the time-consuming image-analysis engine.

In principle, it is possible to create a window that shows three columns for menus, such that more menus remain visible and less menus must be activated. However, it is possible that a dynamic layout of more menus, with no pre-defined positions, is confusing and decreases efficiency. This possibility can be tested when the implemented functionality is more complete, but one window that shows *all* menus will be impossible.

We deliberately chose not to include an *undo* option that would return the canvas to previously obtained results. There are three reasons for this choice: (a) The *undo* option is typical in applications like Paint Shop Pro and GIMP, which are closely related to image processing with a multitude of menus and parameters that users, who do not understand image processing, must experiment with by trial-and-error. (b) The user must develop a feeling for techniques and effects, such that it is possible to anticipate and obtain results which are expected (the user must undergo a training phase; real painters also had to learn techniques). (c) The user can load, adapt and save stylefiles, also intermediate results. Once the major part of the functionality has been implemented and tested, the beta version will be made available to artists and non-artists for evaluation, where the *undo* option may—or may not—appear.

The rendering engine is the result of a long-term research project. Many improvements and extensions are possible in the future. One possibility is to approximate real colour pigments, such as cadmium yellow, cobalt green and burnt umber, and to translate picked colours in the input image to these. The challenge here is to approximate mixed pigments [BWL04]. The same holds for fluorescent colours for the marker-style brush strokes, using spectral rendering techniques [GM98, WLP05]. A big challenge is to obtain real watercolour effects, i.e. drybrush, edge darkening, backruns, granulation, flow effects (instead of just applying anisotropic diffusion) and glazing. The problem is to approximate these effects, also for gouaches, in real-time [VLLVR04, BKTS06]. A very realistic watercolour of 640 by 480 pixels took about 7 hours on a 133 MHz SGI workstation [CAS*97]. On a current high-end PC this may take about one hour, which is not acceptable for interactive use, even after further accelerations by using dual-core or future SMP-on-chip processors.

With respect to future research, two other lines can be pursued. The first project may concern the development of an expert system for painting styles. Although a user can modify parameters by using the interface or stylefiles, an expert system could ask a set of questions like “Do you prefer long, flowing brush strokes or short, discrete ones [1–10]:” and “Do you like impressionism or expressionism [1-10]:”, and the expert system can determine parameters for a new stylefile. The second project may address questions related to the subjective perception of paintings. The interface is—to the best of our knowledge—the first one based on a real painting process which allows to systematically change parameters, i.e. colours, stroke lengths and orientations. By preparing sets of paintings and by asking experts and non-

experts which one of two paintings is more impressionistic, for example, we can discover which parameters are optimal for impressionism and where precisely the boundary lies between im- and expressionism. Without doubt, more ideas will be developed, because we just started freeing the *homo ludens* within us!

Acknowledgements: This investigation is partly financed by FCT program POSI, framework QCA III, and by PRODEP III Medida 5, Action 5.3. The castle image is courtesy of Doug DeCarlo.

References

[BKTS06] BOUSSEAU A., KAPLAN M., THOLLOT J., SILLION F.: Interactive watercolor rendering with temporal coherence and abstraction. *Proc. ACM/SIGGRAPH-Eurographics NPAR, Annecy (France)* (2006), to appear.

[BWL04] BAXTER W., WENDT J., LIN M.: IMPaSTo: A realistic, interactive model for paint. *Proc. ACM/SIGGRAPH-Eurographics NPAR, Annecy (France)* (2004), 45–56.

[CAS*97] CURTIS C., ANDERSON S., SEIMS J., FLEISCHER K., SALESIN D.: Computer-generated watercolor. *Computer Graphics 31* (1997), 421–430.

[dBRN*06] DU BUF J., RODRIGUES J., NUNES S., ALMEIDA D., BRITO V., CARVALHO J.: Painterly rendering using human vision. *To appear in: VIRTUAL, Advances in Computer Graphics in Portugal.* (2006), 1–12. http://virtual.inesc.pt.

[DS02] DECARLO D., SANTELLA A.: Stylization and abstraction of photographs. *Proc. ACM SIGGRAPH02* (2002), 769–776.

[GCS02] GOOCH B., COOMBE G., SHIRLEY P.: Artistic vision: painterly rendering using computer vision techniques. *Proc. ACM/SIGGRAPH-Eurographics NPAR, Annecy (France)* (2002), 83–91.

[GM98] GARRETT M., MARK D.: Computer synthesis of spectroradiometric images for color imaging systems analysis. *Proc. 6th Color Imaging Conf.: Color Science, Systems and Appl., Scottsdale, Arizona* (1998), 150–153.

[Her98] HERTZMANN A.: Painterly rendering with curved brush strokes of multiple sizes. *Proc. ACM SIGGRAPH98* (1998), 453–460.

[Her03] HERTZMANN A.: A survey of stroke-based rendering. *IEEE Comp. Graphics Appl. 23*, 4 (2003), 70–81.

[RGM04] RIZZI A., GATTA C., MARINI D.: From retinex to automatic color equalization: issues in developing a new algorithm for unsupervised color equalization. *J. Electronic Imaging 13* (2004), 75–84.

[Sou03] SOUSA M.: Theory and practice of non–photorealistic graphics: Algorithms, methods, and production systems. *Course Notes for SIGGRAPH03.* (2003). http://pages.cpsc.ucalgary.ca/~mario/.

[VLLVR04] VAN LAERHOVEN T., LIESENBORGS J., VAN REETH F.: Real-time watercolor painting on a distributed paper model. *Proc. Computer Graphics Int.* (2004), 640–643.

[WLP05] WILKIE A., LARBOULETTE C., PURGATHOFER W.: Spectral colour order systems and appearance metrics for fluorescent solid colours. *Proc. Computational Aesthetics in Graphics, Visualization Imaging (Eurographics Dig. Library)* (2005), 1–5.

[YMI04] YAMAMOTO S., MAO X., IMAMIYA A.: Colored pencil filter with custom colors. *Proc. 12th Pacific Conf. Comp. Graphics and Appl. (PG'04)* (2004), 329–338.