FINE ARTS EDUTAINMENT
The Amateur Painter

D. Almeida, S. Nunes, J. Carvalho, V. Brito, J. Rodrigues and J. M. H. du Buf
Vision Laboratory – Inst. for Systems and Robotics (ISR), University of Algarve, 8000-117 Faro, Portugal

Keywords: NPR, painterly rendering, interface, edutainment, fine arts, visual perception.

Abstract: A new scheme for painterly rendering (NPR) has been developed. This scheme is based on visual perception, in particular the multi-scale line/edge representation in the visual cortex. The Amateur Painter (TAP) is the user interface on top of the rendering scheme. It allows to (semi)automatically create paintings from photographs, with different types of brush strokes and colour manipulations. In contrast to similar painting tools, TAP has a set of menus that reflects the procedure followed by a normal painter. In addition, menus and options have been designed such that they are very intuitive, avoiding a jungle of sub-menus with options from image processing that children and laymen do not understand. Our goal is to create a tool that is extremely easy to use, with the possibility that the user becomes interested in painting techniques, styles, and fine arts in general.

1 INTRODUCTION

Digital cameras of good quality have become ubiquitous in the last years. In addition, the use of email and PCs, apart from communication services related to portable devices such as telephones, 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, and many more, users can play with them, they can reveal the *homo ludens* hidden inside them and, eventually, we hope they become interested in painting and the fine arts.

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 sub-menus, even the possibility to create artistic effects by means of filters like the "impressionist" one. However, most people, and especially children, do not have the necessary knowledge—or simply the time—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. There is definitely a need to develop interfaces which are very simple, but in combination with rendering techniques that represent the state-of-the-art in NPR, here painterly rendering.

Two of such interfaces are used by Gertrudis (www.gertrudisgraphics.com) and by 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 adopt the old Polaroid slogan "You push the button, we do the rest," but with the plural "buttons." However, our own solution The Amateur Painter (TAP) is different, because (a) it is automatic in the sense that the user does not select regions, (b) it is not limited to a set of pre-defined styles, (c) it offers sets of options in menus which reflect the logical steps as followed by real painters, and (d) 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, and they partly explain the A in TAP: from a not-really interested amateur who wants good results fast, to an expert amateur who likes to invest a lot of time in optimising results. Therefore, we prefer TAP over YAPP (Yet Another Painting Program)!

Non-photorealistic rendering (NPR) has become a mature research area, with numerous approaches to painterly rendering on the basis of discrete brush

Almeida D., Nunes S., Carvalho J., Brito V., Rodrigues J. and M. H. du Buf J. (2008).
FINE ARTS EDUTAINMENT - The Amateur Painter.
In *Proceedings of the Third International Conference on Computer Graphics Theory and Applications*, pages 262-267
DOI: 10.5220/0001094002620267

strokes. For a recent NPR taxonomy and a review of stroke-based techniques see (Sousa, 2003; Hertzmann, 2003). Most rendering techniques are based on image analysis algorithms from computer vision, although there is a tendency towards including aspects of human vision, for example in constructing saliency maps (DeCarlo and Santella, 2002). In contrast, our own, recently developed technique (du Buf et al., 2006) 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 being complemented with an interface that shows, in one window, available options without a complicated menu structure. This interface allows new users, with absolutely no prior experience, to become familiar with the tool in about half an hour and obtain already good results.

2 THE RENDERING PROCESS

The method has been described in detail in (du Buf et al., 2006). 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.

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 oil-painted brush strokes that have been digitised.

Figure 1 shows an input image (top-left) and the background process using completely random oil strokes and a flat brush (1st and 2nd rows). The three backgrounds shown in the third row were rendered with randomised vertical and horizontal strokes and with diagonal crisscrossing. The bottom row shows foregrounds rendered with flat and round brushes as well as with spray. Figure 2 (top row) illustrates final results obtained with changing saturation and brightness; the bottom row shows an input image and the effect of reducing the colour gamut. Finally, Fig. 3 shows another input image and the use of mixed media, in this case pen and ink on top of a watercolour. For more results see also (du Buf et al., 2006).

Our painting 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. The Amateur Painter does not allow for interactive painting of regions. 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. TAP offers many possibilities with many parameters. Although the user can work with pre-defined and customised stylefiles, for example a default style (sort-of impressionist oil) which is loaded at startup, menus and their parameter options reflect the practice of a real painter with the following logical steps: (a) Select a surface, canvas or paper with a certain texture, and prepare the

Figure 1: Input image (top-left), and the background process using random oil strokes with a flat brush. Third row: backgrounds created with vertical and horizontal strokes as well as diagonal crisscrossing. Bottom row: foreground strokes with a flat brush, a round one, and spray.

surface with one colour. (b) Prepare the palette, i.e. define the colour gamut, for example with unsaturated colours for making a watercolour, or with an emphasis on red-orange for obtaining a “warm” effect. (c) 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. (d) Decide which strokes are going to be applied, for example short, long, straight or curved, and in the latter case the symmetry for cir-

Figure 2: Top: palette effects, changing saturation and brightness. Bottom: input image (left) and final renderings with limited colour gamuts.

cular or S-shaped strokes. (e) Decide how to paint, from very fast (wet-in-wet) to slow such that paint can dry (wet-on-dry). (f) 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 colour of the prepared surface will disappear, or it can be incomplete with a certain density. (g) 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. It is possible to apply mixed media, creating a background in one style and then a foreground in another style. 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 im- 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.

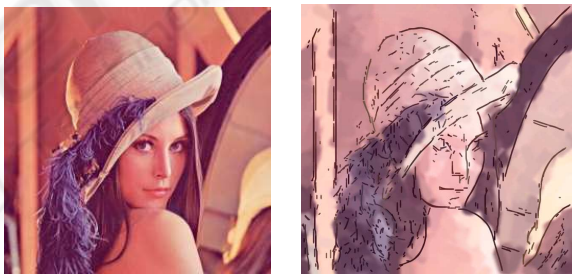

Figure 3: Input image and applying mixed media: pen and ink on top of a watercolour.

3 INTERFACE 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

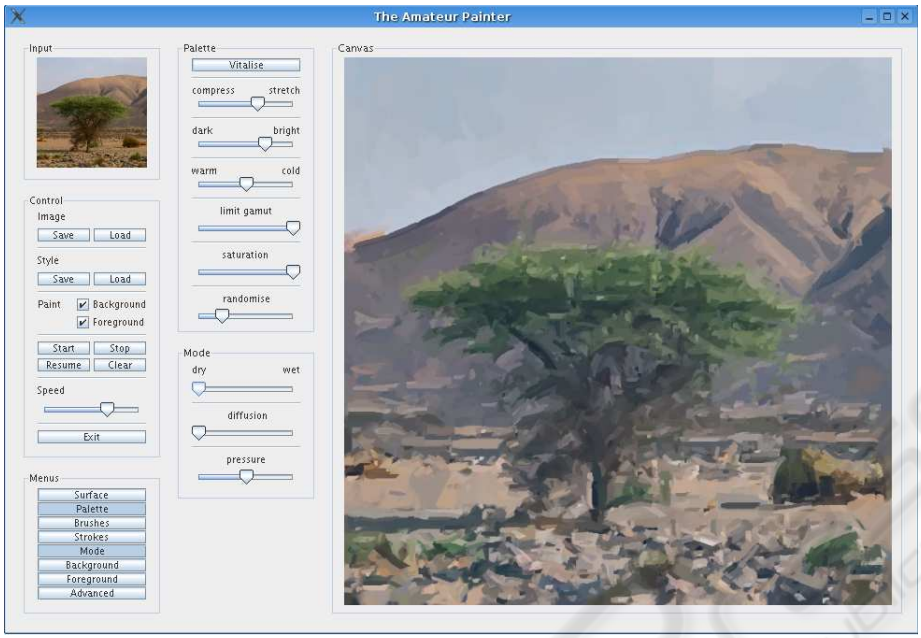

Figure 4: The interface with input image (top-left) and canvas (right). .

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 time stamp that will be updated each time that it is used. The menu with the oldest time stamp will be taken out.

Figure 4 shows a screenshot of the entire TAP window. 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). For a detailed interface explanation see (Nunes et al., 2006).

4 DISCUSSION

In the case of the prototype system with limited functionality, the learning curve of new users was extremely steep. Default settings have been chosen such that, after loading an image and activating the *start* button in the `control` menu, the user saw the painting in progress and could already develop a feeling of what was happening. In about 15 minutes most options could be tried and the corresponding results observed. After about 30 minutes the users were able to select settings such that the final rendering corresponded, more or less, to what was expected.

We deliberately chose not to include an *undo* option that would return the canvas to previously obtained results, step by step. 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.

In our view, TAP is an ideal tool for becoming familiar with painting techniques. The edutainment aspect is not limited to (semi)automatically painting and experimenting with all options. Apart from stimulating *homo ludens*, for example by means of optimised stylefiles for specific effects and painters, the tool could also show real paintings, but this may cause problems with copyrights when the tool is made avail-

able to a wide audience. Much simpler may be to include explanations, by means of *help* options, and/or to create webpages with example results and links to other sources such as museums, techniques, courses etc., for which many portals already exist (for example `www.virtualpainter.com`).

TAP 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 on trajectories in colour space (Chromafile, 2007). The same holds for fluorescent colours for the marker-style brush strokes, using spectral rendering techniques (Garrett and Mark, 1998; Wilkie et al., 2005). 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 (Van Laerhoven et al., 2004). A very realistic watercolour of 640 by 480 pixels took about 7 hours on a 133 MHz SGI workstation (Curtis, C. and Anderson, S. and Seims, J. Fleischer, K. and Salesin, D., 1997). 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 multi-core CPUs or massively parallel GPUs.

The functionality of the current alpha version, of which about 70% has been implemented, has been discussed with a few possible end-users, both experts and non-experts. All are very impressed by the quality of the results and the interface design. Almost every user, after experimenting with TAP-alpha, has ideas for future extensions. These concern mainly special colour effects as used in contemporary art and brush types, for example “paint dripping” à la Jackson Pollock. Of course, in principle it is possible to implement many more options, but the question is what most users would like to use and whether the extreme simplicity of the menu structure can be preserved. *Homo ludens* knows no limitations, but by including many extreme effects it is possible that TAP is going to be seen—or abused—as The Weird Painter’s Secret Attic. This is not our goal.

ACKNOWLEDGEMENTS

This work was partially supported by the Fundação para a Ciência e a Tecnologia (ISR/IST pluri-annual funding) through the POS_Conhecimento Programme which includes FEDER funds.

REFERENCES

Chromafile (2007). Website. `http://www.chromafile.com`.

Curtis, C. and Anderson, S. and Seims, J. Fleischer, K. and Salesin, D. (1997). Computer-generated watercolor. *Computer Graphics*, 31:421–430.

DeCarlo, D. and Santella, A. (2002). Stylization and abstraction of photographs. *Proc. ACM SIGGRAPH02*, pages 769–776.

du Buf, J., Rodrigues, J., Nunes, S., Almeida, D., Brito, V., and Carvalho, J. (2006). Painterly rendering using human vision. *Virtual - Portuguese Journal of Computer Graphics*, AICG 2005:12.

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

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

Nunes, S., Almeida, D., Brito, V., Carvalho, J., Rodrigues, J., and du Buf, J. (2006). Perception-based painterly rendering: functionality and interface design. *Proc. Ibero-American Symp. on Computer Graphics, Santiago de Compostela, Spain*, pages 53–60.

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

Van Laerhoven, T., Liesenborgs, J., and Van Reeth, F. (2004). Real-time watercolor painting on a distributed paper model. *Proc. Computer Graphics Int.*, pages 640–643.

Wilkie, A., Larboulette, C., and Purgathofer, W. (2005). Spectral colour order systems and appearance metrics for fluorescent solid colours. *Proc. Computational Aesthetics in Graphics, Visualization Imaging (Eurographics Dig. Library)*, pages 1–5.