EUROGRAPHICS 2009 / P. Dutré and M. Stamminger
(Guest Editors)

Volume 28 (2009), Number 2

LazyBrush: Flexible Painting Tool for Hand-drawn Cartoons

Daniel Sýkora[†], John Dingliana, and Steven Collins

Trinity College Dublin

Abstract

In this paper we present LazyBrush*, a novel interactive tool for painting hand-made cartoon drawings and animations. Its key advantage is simplicity and flexibility. As opposed to previous custom tailored approaches [SBv05, QWH06]* LazyBrush *does not rely on style specific features such as homogenous regions or pattern continuity yet still offers comparable or even less manual effort for a broad class of drawing styles. In addition to this, it is not sensitive to imprecise placement of color strokes which makes painting less tedious and brings significant time savings in the context cartoon animation.* LazyBrush *originally stems from requirements analysis carried out with professional ink-and-paint illustrators who established a list of useful features for an ideal painting tool. We incorporate this list into an optimization framework leading to a variant of Potts energy with several interesting theoretical properties. We show how to minimize it efficiently and demonstrate its usefulness in various practical scenarios including the ink-and-paint production pipeline.*

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.4]: Graphics Utilities—Graphics editors, Image Processing and Computer Vision [I.4.6]: Segmentation—Pixel classification, Computer Applications [J.5]: Arts and Humanities—Fine arts

1. Introduction

Painting, i.e. the process of adding colors to hand-made drawings, is a common operation in standard image manipulation programs starting from simple bitmap editors such as *Paintbrush* to professional digital ink-and-paint solutions like *Animo*, *Toonz*, or *Retas*. In these systems a variant of the flood-fill algorithm is typically used to speed up painting. This algorithm works well for images with homogenous regions and salient continuous outlines. However, many hand-made drawing styles contain more complicated structures (e.g. pencil drawing in Figure 1). For such images it is necessary to perform many detailed manual corrections to get clean results. This additional effort can be very time consuming and cost ineffective in the context of the ink-and-paint pipeline where thousands of frames must be painted.

Recently, significant effort has been devoted to a similar problem – the interactive colorization of gray-scale images [LLW04,YS06]. Although these approaches offer fascinating results on natural photographs and videos, they typically fail when applied to hand-made drawings which do not preserve a smooth image model (see Figure 2). Sýkora et al. [SBv05] addressed this issue by developing an unsupervised segmentation algorithm for black-and-white cartoon animations able to produce segmentation similar to that produced by *connected component analysis* [RK82] on a binary image. The main drawback of their approach is the assumption of large homogenous regions enclosed by distinct continuous outlines. When applied to more complicated styles, they tend to group salient regions due to gappy outlines or produce many small regions (see Figure 2).

Qu et al. [QWH06] proposed manga colorization framework that overcomes forementioned limitations by exploiting both pattern and intensity continuity in conjunction with a level-set optimization. According to user-specified examples of hatching patterns, they extract textural features and compute a similarity map having an intensity profile like a homogeneous region with distinct boundaries. Subsequently they propagate colors from user-specified scribbles until they reach salient barriers. During the propagation they also employ shape regularization to overcome possible leakage through gappy boundaries. Despite the success of this ap-

[†] e-mail: sykorad@cs.tcd.ie

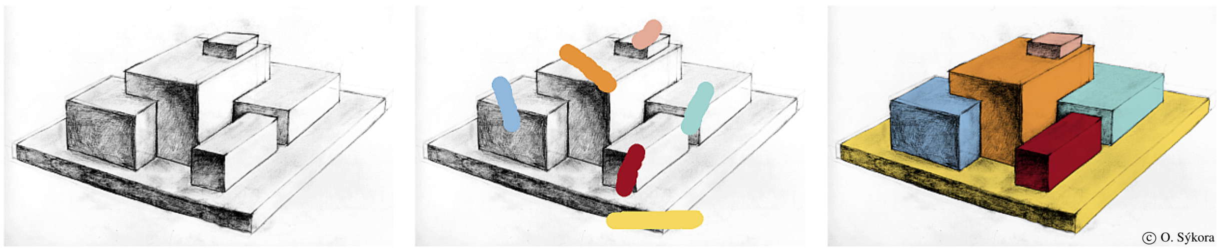

Figure 1: *LazyBrush in action – minimal effort is needed to paint this highly structured pencil drawing with fuzzy outlines and shaded regions (left). See how the algorithm handles imprecise placement of color strokes (middle) and is able to produce high quality anti-aliased output (right).*

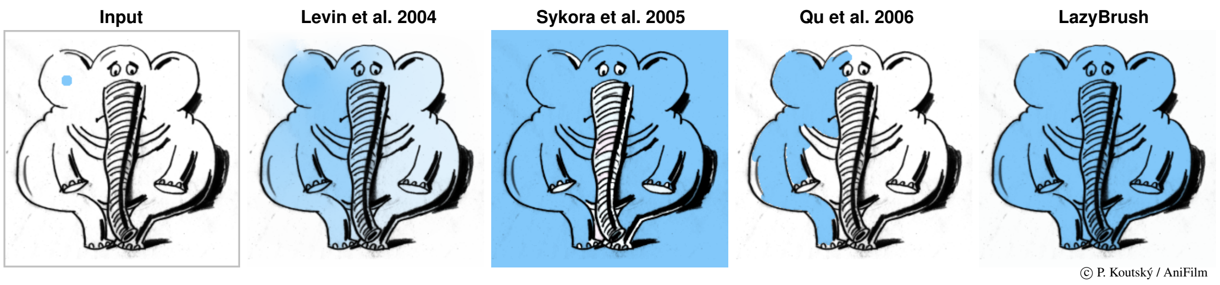

Figure 2: *LazyBrush vs. state-of-the-art – various algorithms applied on the same input data (background seeds around the image border and blue seed inside the elephant's ear): Levin et al. [LLW04] assume improper image model, Sýkora et al. [SBv05] do not handle gaps and produce many small regions, and Qu et al. [QWH06] get stuck in inappropriate local minima so that all remaining regions should be filled individually. In contrast to this, LazyBrush finds an optimal boundary and does not require further effort.*

proach, many important issues remain. Since the level-set optimization is based on gradient descent it can easily get stuck in some inappropriate local minima. This typically occurs when the algorithm is used for images which do not contain repetitive hatching patterns (see Figure 2). In this case the user has to specify many additional scribbles or tweak parameters of level-set optimization to allow crossing salient boundaries during front propagation. Another problem occurs when narrow or small regions are painted. Also in this case many thin scribbles must be drawn and parameters tweaked to achieve desired results. These limitations hinder the practical usability of manga colorization for images which do not contain repetitive patterns.

The aim of this paper is to present a novel flexible painting tool easily applicable to various drawing styles. We demonstrate an approach that is independent of style-specific features but, despite this, requires comparable or less manual effort than previous style-limited approaches. Our key contribution is hidden in a list of previously undiscussed properties presented in Section 3 which redefines behavior of an ideal painting tool. This list arose from a requirements analysis carried out with professional ink-and-paint illustrators. We reformulate it as an energy optimization problem and obtain an interesting and, to our knowledge, unexplored variant of energy function with Potts interaction [Pot52] and special sparse data term. We discuss its interesting theoretical properties and present an efficient approximation algorithm requiring only a few globally optimal decisions to obtain a nearly optimal solution.

The rest of the paper is organized as follows. First we briefly discuss related work, then we analyze some desired properties of a new painting tool, formulate the energy minimization problem and show how to solve it efficiently. Afterwards we use our new algorithm for painting real cartoon images in different drawing styles and analyze its practical strengths and limitations. Finally, we present a couple of promising applications in the cartoon production pipeline and conclude with several new avenues for future research.

2. Related work

Interactive filling of homogenous regions has been studied since several decades ago when large pixel frame-buffers became practical. Lieberman [Lie78] proposed an extension of the flood-fill algorithm for filling with arbitrary black-and-white patterns, Smith [Smi79] showed how to fill regions with shaded boundaries, and Fishkin and Barsky [FB84] presented recoloring of anti-aliased images. Although these approaches can simplify filling in some special cases, they still suffer from limitations of the original flood-fill algorithm, i.e. the inability to cope with gappy boundaries or to reach a salient boundary of a region with complicated hatching.

The same limitations also hold for auto-painting systems [SF00, QST*05] which build upon connected component analysis. This process is equivalent to sequential execution of the flood-fill algorithm with different labels on each unfilled pixel in a thresholded binary image. Sýkora et al. [SBv05] replaced the thresholding by a more sophisticated outline detection algorithm allowing auto-painting of black-and-white cartoon animations. Nevertheless, in the final stage, they still rely on connected component analysis and thus share the aforementioned limitations.

A related operation to filling is colorization based on color seeds. This method was pioneered by Horiuchi [Hor02] who used probabilistic relaxation to propagate colors. Levin et al. [LLW04] popularized this approach with their variant based on a weighted least squares optimization framework. Later Yatziv and Sapiro [YS06] proposed a different solution based on a blending of several nearest color seeds weighted by geodesic distance. Although these approaches require little effort for images satisfying a smooth image model, they become impractical for cartoon images due to color bleeding artifacts. Qu et al. [QWH06] and later Luan et al. [LWCO*07] addressed these issues by employing hard pre-segmentation based on texture classification schemes. However, this approach is applicable only for drawing styles containing repetitive textural patterns.

Painting has much in common with interactive image segmentation. This field was mainly motivated by the seminal work of Boykov and Jolly [BJ01] who demonstrated numerous benefits of a graph cut based solution. Grady [Gra06] later proposed a concurrent approach based on a weighted least squares framework (similar to [LLW04]) which is easily extendable to multi-label segmentation and obtains comparable results to a graph cut framework. Nevertheless, all these approaches do not take into account the specific requirements of painting which differ from those used in image segmentation.

3. Ideal painting tool

In this section, we formulate a set of desired properties for an ideal painting tool. This set arose from discussion with professional ink-and-paint illustrators who are familiar with standard image manipulation tools as well as professional ink-and-paint systems. They typically use a variant of the flood-fill algorithm, providing an effective solution for simple cartoon images with homogenous regions and distinct continuous outlines, but one rarely applicable to more complicated drawing styles.

One of the well-known problems of the flood-fill algorithm is color leakage through outline gaps. To overcome this issue, illustrators typically join problematic gaps manually. This is a tedious task requiring high concentration since the human visual system normally tends to connect weak edges [Kan79]. In professional ink-and-paint systems, automatic outline joining algorithms [SC94] are available. However, this process usually connects all gaps which is often counterproductive since in many drawings this operation removes the simplicity of one-click filling. A similar problem occurs also when the image contains hatching or many small regions. In these cases illustrators typically delineate the region of interest using some edge snapping selection tool (such as *intelligent scissors* [MB99]) and then fill the whole area. This however requires precise positioning of boundary seeds which is a tedious task. Manga colorization [QWH06] partially overcomes these limitations by virtually converting areas with repetitive patterns into homogenous regions with distinct boundaries. Nevertheless, such conversion works only for manga since repetitive patterns are rare in hand-made cartoon drawings.

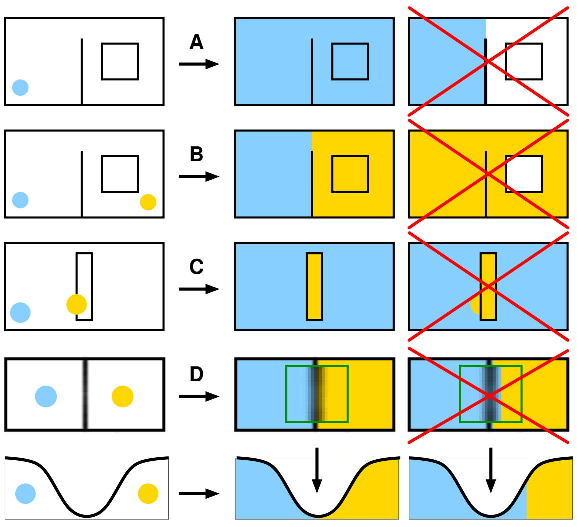

Figure 3: *An ideal painting tool tends to fill as much area as possible (A); when there are concurrent seeds, it finds an optimal boundary regardless of gappy outlines and produces compact regions without holes (B); it supports soft scribbles by preserving rule of majority so it is not necessary to paint precisely inside the region of interest (C); it handles anti-aliasing by pushing color boundaries to pixels with minimal intensity not with maximal gradient (D).*

Optimal boundary. The illustrators' wish is to have a tool that tends to fill as much area as possible by finding an optimal enclosing boundary (regardless of holes and gappy outlines) and then, when necessary, they can refine the interior using additional strokes (see cases A and B in Figure 3). Such workflow is not supported in manga colorization. Although it handles gappy outlines via region shape regularization, it is not able to find and optimal boundary due to getting stuck in inappropriate local minima (see Figure 2 or red crossed example in Figure 3, rule A).

Connected labelling. In manga colorization, user edits can produce color regions with arbitrary topology (i.e. they can consist of several disconnected parts). This functionality brings considerable speed-up in a special case when there is a one-to-one correspondence between color and pattern. However, in a more general setting this behavior can be confusing since it breaks a locality assumption, which is essential for painting and is required by illustrators.

Soft scribble. Another feature which illustrators appreciate is a color brush resistant to imprecise placement. According to naming convention used in colorization and interactive image segmentation, we refer to strokes made with such a brush as *soft scribbles*. Soft scribbles should satisfy the so called *rule of majority*, meaning that a region is filled with a color whose strokes have most of their pixels lying in its interior (see case C in Figure 3). This simple rule can bring significant time savings when painting thin structures or small regions. Due to Fitts' law [Fit54] the time needed to reach thin objects can be greatly reduced by slightly increasing brush radius (see Figure 4). A great speed up can also be achieved in the context of the ink-and-paint pipeline when several aligned animation phases are painted simultaneously (*onion fill*) or when color patches are transferred from already painted frames to new ones (*patch pasting*, see Section 5 and Figure 9). In comparison to the manga colorization, soft scribbles are a completely new feature, however, a similar idea has been explored recently in the context of appearance editing [AP08]. The key difference is that the energy minimization framework used in [AP08] takes into account only coarse edits which are insufficient for painting.

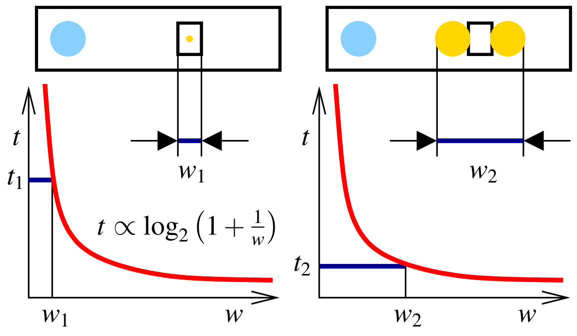

Figure 4: *Soft scribbles and Fitts' law [Fit54] – the task is to fill the small rectangle of width w_1. Using a pixel-wide brush the expected time needed to reach its interior is t_1. By increasing brush radius we can enlarge the target margin to w_2 and obtain considerably lower time t_2.*

Anti-aliasing. Since scanned hand-drawn images contain soft anti-aliased edges, it is necessary to have a mechanism that preserves such anti-aliasing during the painting phase (see case D in Figure 3). This feature can also be formulated as a goal to retrieve boundaries minimizing the visibility of color discontinuities. The reason is that in cartoon images dark outlines are used to emphasize region shape and since the color is typically multiplied by the original intensity, the optimal boundary should be in the place where this intensity is minimal. This finding is inconsistent with standard maximum gradient formulation used in interactive image segmentation [BJ01] (see intensity profiles in Figure 3 bottom). In manga colorization this feature was not discussed since authors considered only binary images.

4. Energy function

In this section, we formulate an energy minimization framework, the aim of which is to satisfy the requirements presented in the previous section.

As an input we have a gray-scale image I consisting of pixels P in a 4-connected neighborhood system $\mathcal{N}$ and a set of user-provided non-overlapping strokes S with colors C. The aim is to find a labelling, i.e. the color-to-pixel assignment c that minimizes the following energy function:

$$E(c) = \sum_{\{p,q\}\in\mathcal{N}} V_{p,q}(c_p,c_q) + \sum_{p\in P} D_p(c_p) \qquad (1)$$

where smoothness term $V_{p,q}$ represents the energy of color discontinuity between two neighbor pixels p and q, and data term D_p the energy of assigning color c_p to pixel p.

4.1. Smoothness term

As discussed in Section 3, the aim is to hide color discontinuities. Since typically multiplicative color modulation is used, the best locations for color discontinuities are at pixels where the original image intensity is low, e.g. inside dark outlines. According to this finding we let the energy $V_{p,q}$ be:

$$V_{p,q}(c_p,c_q) \propto \begin{cases} I_p & \text{for } c_p \neq c_q \\ 0 & \text{otherwise} \end{cases} \qquad (2)$$

However, the absolute values of $V_{p,q}$ should be set carefully since they have a fundamental impact on the resulting labelling. As we want to prefer compact and hole-free regions it is necessary to avoid zeros in $V_{p,q}$ for the case $c_p \neq c_q$, otherwise regions with outlines having zero intensity will not contribute to the minimum of (1). Such regions can be easily disconnected and produce holes in the final labelling. As opposed to that, non-zero smoothness term will lead to compact regions without holes. However, it can also produce unintended shortcuts through white areas. To suppress this shortcoming it is necessary to set high energies for the boundaries going through the white pixels. Theoretically, this energy should be higher than the longest outline in the image I. Nevertheless, a good estimate for this value is a perimeter of I. In most cases this setting effectively ensures that a region boundary will go through white pixels only when there is no other low energy path along dark outlines. Following these ideas, we map an interval of image intensities $\langle 0,1\rangle$ to $\langle 1,K\rangle$, where $K = 2\cdot(w+h)$, w is width and h height of I. For nearly binary images such mapping can be linear, i.e. $I'_p = K\cdot I_p + 1$, however, for black-and-white cartoons or soft pencil drawings (such as "blocks" image in Figure 1 or "robber" in Figure 10) the problem with shortcuts persists due to lower contrast between homogenous areas and outlines.

To alleviate this issue it is possible to use some nonlinear mapping that enhances the contrast (e.g. $I'_p = K \cdot I_p^2 + 1$) or employ a more powerful technique previously used for outline detection in black-and-white cartoon images [SBv05]. Here, outlines are detected using the response of a Laplacian of Gaussian ($\mathbf{L}\circ\mathbf{G}$) filter. This filter corresponds to a light-over-dark mechanism used in the primary stages of the human visual system [MH80]. From a mathematical point of view, $\mathbf{L}\circ\mathbf{G}$ estimates the second order derivative of the image intensity, its zero-crossings correspond to edge locations, and local maxima to places with high curvature (e.g. centers of outlines). According to this we preprocess the image I by filtering with $\mathbf{L}\circ\mathbf{G}$ and produce a new image $I_f = 1 - \max(0, s \cdot \mathbf{L}\circ\mathbf{G}(I))$ where the negative response of $\mathbf{L}\circ\mathbf{G}$ is clamped to zero and positive values scaled by s to match the interval $\langle 0, 1\rangle$. After this preprocessing, the contrast of outlines is emphasized and the interior of homogenous regions are turned to white regardless of their original intensity (see Figure 5). Finally, values in I_f are linearly mapped to the interval $\langle 1, K\rangle$ and used in smoothness term $V_{p,q}$.

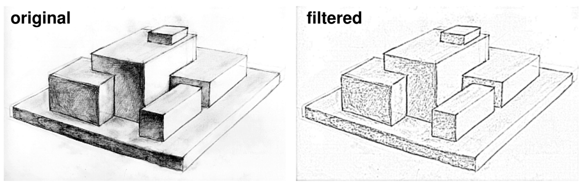

Figure 5: *An example of an image preprocessed by filtering with $\mathbf{L}\circ\mathbf{G}$ – the original image (left); normalized and clipped response of $\mathbf{L}\circ\mathbf{G}$ (right). See the improvement on the contrast of outlines.*

Note, how our smoothness term completely differs from terms used in interactive image segmentation [BJ01, Gra06]. The aim here is to push the segment boundary to pixels with maximal gradient. If the gradient magnitude is high (as in cartoon images), many pixels can have $V_{p,q}$ near or equal zero. As discussed in Section 3 this setting is unsuitable for painting since it reveals color discontinuities on soft edges and produces holes.

4.2. Data term

In manga colorization or interactive image segmentation the data term D_p is usually set to some image-based likelihood such as pattern or intensity similarity. The assumption behind this setting is that there is a one-to-one correspondence between color and pattern/intensity. However, repetitive patterns or intensity variations are not typical for hand-made drawings and even if they are present, one-to-one correspondences are rare. To address this fact *LazyBrush* does not rely on image-based likelihoods but uses completely user-driven data term allowing the implementation of a soft scribbles discussed in Section 3.

The key idea is to relax a common assumption, i.e. that all user-defined seeds are necessarily hard constraints. Instead we let the user to decide how to penalize labelling by setting:

$$D_p(c_p) = \lambda \cdot K, \tag{3}$$

where $\lambda \in \langle 0, 1\rangle$ is a constant given by the user and K is the energy of discontinuity at white pixels that balances the influence of data and smoothness terms (therefore we use the same symbol as in Section 4.1). The value of λ indicates the presence of a brush stroke and its "strength": $\lambda = 1$ is for pixels that have not received a brush stroke, $\lambda = 0$ for hard scribbles, and for soft scribbles λ should satisfy the following inequality: $0 + K \cdot |S| < K \cdot \partial S + \lambda K \cdot |S|$, saying that the energy (1) is lower even if the pixels under a scribble S have not receive its color ($|S|$ is the area and ∂S the perimeter of S). From this constraint we obtain: $\lambda > 1 - \partial S/|S|$ which we can measure for each scribble but in practice most scribbles have $1 - \partial S/|S| < 0.95$ so we set $\lambda = 0.95$.

It is easy to verify that soft scribbles preserve the rule of majority. Imagine several seeded pixels S with $D_p = \lambda \cdot K$ inside a region R where the smoothness is assumed to by constant. Then the labelling with minimal energy should have lowest $\sum D_p = \lambda \cdot K \cdot |S| + K \cdot |R - S|$. After simplification: $\sum D_p = K \cdot (|R| - (1 - \lambda) \cdot |S|)$ yields minimum for the largest $(1 - \lambda) \cdot |S|$, i.e. when all scribbles have equal λ then the winner will have the largest number of seeded pixels $|S|$.

4.3. Minimization

Now we proceed to the minimization of (1). Since the smoothness term $V_{p,q}$ depends only on pixel intensity and not on the color labels, our energy function satisfies Potts model [Pot52]. As shown in [BVZ98] minimizing such a function is equivalent to solving a *multiway cut* problem on a certain undirected graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ where $\mathcal{V} = \{P, C\}$ is a set of vertices and $\mathcal{E} = \{\mathcal{E}_p, \mathcal{E}_c\}$ a set of edges (see Figure 7).

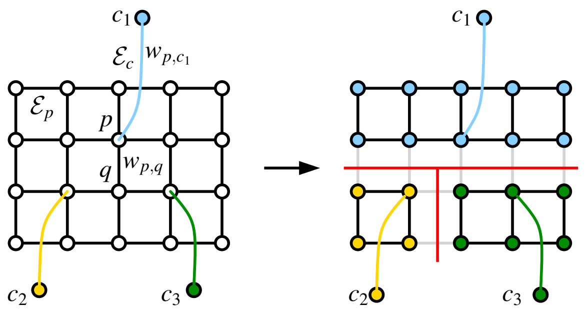

Figure 7: *Multiway cut – basic structure of graph $\mathcal{G}$ (left): pixels P (white dots), color terminals C (color dots), pixel edges $\mathcal{E}_p$ with weight $w_{p,q}$ (black lines), and links to color terminals $\mathcal{E}_c$ with weight $w_{p,c}$ (color lines). Resulting multiway cut and corresponding labelling of pixels (right).*

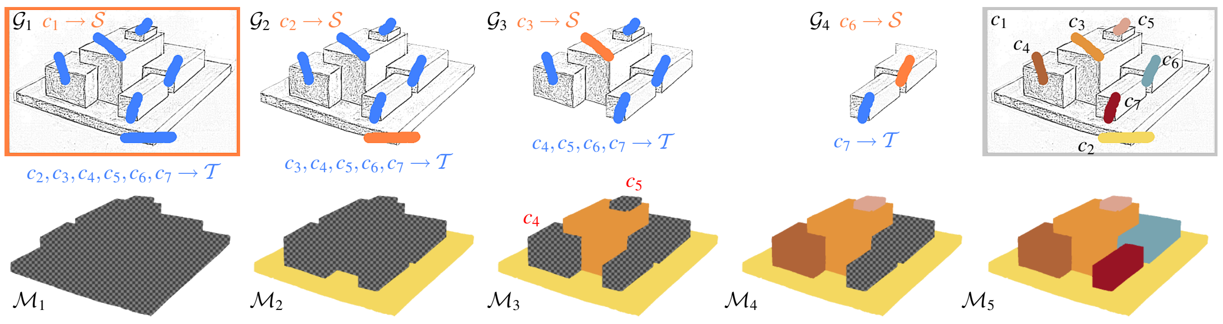
$\mathcal{G}_1$ $c_1 \rightarrow \mathcal{S}$
$c_2, c_3, c_4, c_5, c_6, c_7 \rightarrow \mathcal{T}$
$\mathcal{G}_2$ $c_2 \rightarrow \mathcal{S}$
$c_3, c_4, c_5, c_6, c_7 \rightarrow \mathcal{T}$
$\mathcal{G}_3$ $c_3 \rightarrow \mathcal{S}$
$c_4, c_5, c_6, c_7 \rightarrow \mathcal{T}$
$\mathcal{G}_4$ $c_6 \rightarrow \mathcal{S}$
$c_7 \rightarrow \mathcal{T}$
c_1 c_3 c_5 c_4 c_6 c_7 c_2
c_5 c_4
$\mathcal{M}_1$ $\mathcal{M}_2$ $\mathcal{M}_3$ $\mathcal{M}_4$ $\mathcal{M}_5$

Figure 6: *Multiway cut algorithm in progress – gradually reducing max-flow/min-cut subproblems solved on graphs $\mathcal{G}$ with terminals $\mathcal{S}$ and $\mathcal{T}$ (top), corresponding masks of unlabelled pixels $\mathcal{M}$ (bottom, checkerboard pattern indicates unlabelled pixels). Note how two trivial subproblems c_4 and c_5 were pruned in the third iteration (middle).*

Vertices $\mathcal{V}$ consist of pixels P and color terminals C. Each pixel $p \in P$ is connected to its 4 neighbors via edges $\mathcal{E}_p$ having weight equal to smoothness term $w_{p,q} = V_{p,q}$ for case $c_p \neq c_q$. There are also auxiliary edges $\mathcal{E}_c$ that connect color terminals C to seeded pixels. Each $\mathcal{E}_c$ has weight $w_{p,c} = K - D_p(c)$ (hard scribbles have $w_{p,c} = K$ and soft $w_{p,c} = (1-\lambda) \cdot K$).

Note that in contrast to interactive image segmentation [BJ01] our graph is very sparse (has much less $\mathcal{E}_c$). This is due to the fact that most pixels have $D_p = K$ for all labels so the weight $w_{p,c} = 0$ and thus the corresponding $\mathcal{E}_c$ is redundant. Since there are no other links to terminals besides user-defined the resulting labelling will be always connected to seeds. This is in accordance with properties discussed in Section 3.

A multiway cut with 2 terminals is equivalent to a max-flow/min-cut problem for which efficient algorithms exist [BK04]. However, for 3 or more terminals the problem is NP-hard [DJP*92] even on our sparse graph. Nevertheless, it is interesting that we are very close to P, because if we assume only a set of hard scribbles each with unique terminal (e.g. as in Figure 7), we can always collapse seeded pixels to this terminal and obtain a planar graph for which an exact polynomial algorithm exists [Yeh01]. Nevertheless, we cannot collapse pixels seeded by soft scribbles and so we need to solve the full non-planar problem for which no polynomial approximation scheme exists. The best known approximation [KKS*04] based on geometric embedding and linear programming guarantees an optimal solution within a factor of $1.3438 - \varepsilon_k$, where ε_k goes to zero with increasing number of terminals k (for $k = 3$ the bound is $\frac{12}{11}$). This algorithm is not easy to implement and due to slow performance it is inappropriate for interactive applications. There are also other approximation algorithms based on the max-flow/min-cut subroutine [DJP*92, BVZ01]. Although they guarantee optimality only within a factor of $2 - \frac{2}{k}$ and 2 respectively, they are much easier to implement. The problem is that they are still relatively slow due to many max-flow/min-cut steps. For example it takes more than 11 seconds to compute labelling for 0.5 Mpix image in Figure 1 on a 2.4GHz CPU using α-expansion algorithm described in [BVZ01].

Inspired by the *isolation heuristic* used in [DJP*92] we propose a novel greedy multiway cut algorithm, which takes advantage of our special graph topology guaranteeing connected labelling. In practice, it provides similar results as the widely used α-expansion [BVZ01] but is significantly faster (18x for Figure 1, see also Table 1) and so more suitable for interactive applications. It works in a simple hierarchical fashion by solving less than N one-to-all max-flow/min-cut problems (where N is the number of colors). The significant speed up is obtained thanks to (1) gradually reducing size of max-flow/min-cut subproblems and (2) ability to prune trivial cases. It has the following steps (cf. work-in-progress example in Figure 6):

1. Initialize a set of active color labels C and a mask $\mathcal{M}$ of unlabelled pixels.
2. Find all unlabelled regions R in $\mathcal{M}$ that intersect strokes with only one color label c_r. For each such $r \in R$ set labels in $\mathcal{M}$ to c_r and if there is no other region in $\mathcal{M}$ containing strokes with label c_r, remove c_r from C.
3. If C is empty then stop.
4. Select an arbitrary color label $c \in C$.
5. Build a graph $\mathcal{G}$ from all unlabelled pixels in $\mathcal{M}$.
6. Connect pixels seeded with color label c to terminal $\mathcal{S}$, and pixels seeded with colors $C - \{c\}$ to terminal $\mathcal{T}$.
7. Solve max-flow/min-cut problem [BK04] on $\mathcal{G}$ with source $\mathcal{S}$ and sink $\mathcal{T}$.
8. At pixels where corresponding graph vertices were assigned to terminal $\mathcal{S}$, set label in mask $\mathcal{M}$ to c.
9. Remove color label c from C and go to (2).

Roughly speaking the algorithm selects an arbitrary color as a first terminal and all other colors as a second terminal. Then it solves the binary max-flow/min-cut problem and removes a part of the image assigned to the first terminal. It performs the same operation on the reduced image with reduced set of colors while avoiding max-flow/min-cut computation when there are regions containing only seeds with one color. If there is no other connected component with two different color labels the algorithm stops.

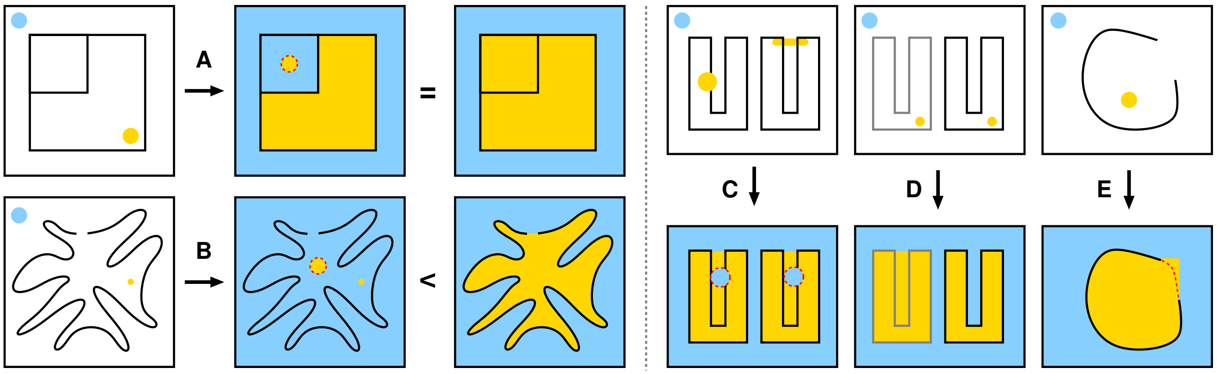

Figure 8: *Limitations – two different minimal solutions with equal energy (A); a shortcut encompassing a small scribble has lower energy than a boundary along the outline (B); the rule of majority is biased by thin creeks (C); low contrast between outlines and homogenous regions causes unintended labelling (D); long gaps or missing outlines can produce jaggy boundaries (E). Additional soft scribbles (marked with red dashed line) are necessary to resolve cases A-C. Case D can be suppressed by contrast enhancement and case E by post-processing using smooth active contour model [XAB07].*

Although such a greedy approach does not guarantee optimality within a factor of 2, in practice it produces labelling with energy close or even slightly better than α-expansion (see Table 1) so that the visual difference is imperceptible. Moreover, when the size of regions corresponding to individual colors is known beforehand (e.g. background seed or dominant color) it is possible to perform a selection of colors from the "largest" to the "smallest" and gain significant subproblem reduction after only a few initial steps. Another great optimization can be achieved if we can predict the topology of the resulting labelling. Then, thanks to the *four color theorem* [AH89], we can group color labels to 4 terminals and use only 4-way cut to obtain a constant time solution for an arbitrary number of colors.

name	resultion	colors	speed up	ΔE [%]
bottle	720x576	3	3x	-0.0452
robber	720x576	6	17x	0.0196
boy	720x576	7	17x	0.0274
picnic	1026x578	7	17x	0.0090
blocks	1026x578	7	18x	0.0038
footman	1026x578	9	9x	0.0025
manga	1026x578	11	16x	0.0395

Table 1: *Our algorithm vs. α-expansion – the speed up increases roughly with the number of colors while the change in labelling is imperceptible (negative ΔE means our algorithm found better local minimum and vice versa). Names correspond to drawings in Figure 10. The drawing "blocks" is shown in Figure 1.*

4.4. Limitations

There are several situations where the energy function (1) does not exactly preserve rules presented in Section 3. Although these cases are rare, the user should be aware of them, know the source of a problem and a way to resolve it.

Ambiguity. The first problematic situation is depicted in Figure 8 (case A). There are two different minimal solutions with equal energy. In this case the structure of the final labelling depends only on the order of labels. This ambiguity can be easily resolved by putting another decisive stroke inside the small square.

Shortcuts. When the user draws thin scribbles (e.g. one pixel wide) inside a region with a very long or gappy outline, the case can easily be that a shortcut encompassing the scribble will have lower energy than a long boundary along the outline (case B in Figure 8). To avoid such degenerate solutions, it is necessary to use wider brushes to ensure that the scribble's perimeter is much longer than the sum of lengths of all gaps.

Majority bias. Another problem is connected with the fact that the rule of majority can be biased by the image content. This bias becomes critical in the case of thin creeks (see case C in Figure 8). Here, the lower energy of soft scribbles can compensate for the high energy of shortcuts and produce unintended labelling. Another soft scribble is necessary to resolve this situation.

Low contrast. Our approach can fail on images where the contrast between outline and homogenous area is low (see case D in Figure 8). For such images it is recommended to use non-linear contrast enhancement or **L**∘**G**-based preprocessing as discussed in Section 4.1. Such modification is necessary only for setting up the smoothness term in (1), the resulting labelling can be then applied on the unmodified image (as done in Figure 1).

Metrication artifacts. When outlines have long gaps, the resulting boundary can have jaggy shape since its length is minimized in the sense of the L^1 norm (see case E in Figure 8). Although an extension exists [BK03], allowing the approximation of the L^2 norm to arbitrary extents, it requires many additional $\mathcal{E}_p$ edges yielding significant slowdown of the optimization. Even if the L^2 norm is minimized, the boundary shape still does not need to be optimal in the sense of higher order continuity (C^1 or C^2). To solve this problem the shape can be post-processed using some active contour model with a more sophisticated smoothness term [XAB07].

5. Results

We implemented *LazyBrush* in a simple interactive application and validated its flexibility on various drawing styles including pen and pencil drawings, black-and-white cartoons [SBv05] and manga [QWH06]. Selected results are presented in Figure 10 (drawings "bottle" and "robber" were preprocessed using $\mathbf{L}\circ\mathbf{G}$ to enhance contrast of outlines).

Note that the user effort required by *LazyBrush* is comparable to previous techniques, even though it does not assume any style-specific features. The roughly positioned soft scribbles also reduce the level of concentration required of the illustrator, which makes painting more enjoyable and thus less tedious. This feature can also be beneficial in applications where specific motor coordination abilities are taken into account (e.g. a painting tool for young children).

The advantage of soft scribbles becomes even more apparent in the context of the ink-and-paint pipeline where thousands of frames must be painted manually. A popular technique here is *onion fill* allowing to paint several superimposed animation in-betweens simultaneously (see Figure 9, top). In contrast to standard approaches *LazyBrush* does not require precise positioning of color seeds. By drawing longer soft scribbles it is possible to cover large movements.

LazyBrush can be also utilized in a more sophisticated auto-painting technique called *patch pasting* [SBv04] where color information is transferred automatically by registering interesting points in already painted and unpainted drawings. The original method requires pre-segmentation in order to compute the most frequently used color inside a region (see Figure 9, bottom). With *LazyBrush* the pre-segmentation is no longer needed since the color patches can be used as soft scribbles and the rule of majority will propagate to the most frequently used color. Thanks to this, patch pasting can be used also for more complicated drawing styles where meaningful segmentation is hard to obtain.

Since *LazyBrush* does not mix colors (as compared to [LLW04]) but produces hard segments, it is possible to use it for filling regions with different content beyond just color, e.g. gradients, patterns or depth information which can be utilized for composition, *unsharp masking* [LCD06] or in *Lumo* [Joh02] where it is necessary to produce correct normal orientation for 3D-like shading.

6. Conclusions and Future work

We have presented a new interactive tool for painting hand-made cartoon drawings called *LazyBrush*. It is based on an optimization framework that tries to mimic the behavior of an ideal painting tool as proposed by professional illustrators. The key advantage of *LazyBrush* is flexibility. It can be used for painting in various drawing styles with comparable user effort to that offered by previous style-limited approaches. We have also demonstrated its usability in the context of the ink-and-paint pipeline for which *LazyBrush* was mainly designated and can provide significant reduction of manual effort.

As future work we plan to integrate *LazyBrush* more into the spatio-temporal domain [WBC*05] and examine possible modifications of our energy function in order to utilize different optimization schemes such as *weighted least-squares* [Gra06, AP08]. Another interesting avenue for future research is *graph coloring algorithms* [RSST96], which allow the grouping of color terminals in such a way that only a fixed number of max-flow/min-cut steps is necessary to obtain a solution for arbitrary numbers of colors.

Acknowledgements

We are grateful to T. Jarkovský, V. Votýpka, and T. Rychecký from AniFilm studio for being initiators and first users of our system. Many thanks also go to L. Kavan for extensive discussions on the earlier version of the paper and to anonymous reviewers for their fruitful comments. Images in this paper are courtesy of P. Koutský / Anifilm, O. Sýkora, L. Vlček, Y. Qu et al., and studios UPP & DMP. This work has been supported by the Marie Curie action IEF, No. PIEF-GA-2008-221320.

References

[AH89] APPEL K., HAKEN W.: Every planar map is four-colorable. *Contemporary Mathematics 98* (1989), 1–741.

[AP08] AN X., PELLACINI F.: AppProp: All-pairs appearance-space edit propagation. *ACM Transactions on Graphics 27*, 3 (2008), 40.

[BJ01] BOYKOV Y., JOLLY M.-P.: Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *Proceedings of Internation Conference on Computer Vision* (2001), pp. I: 105–112.

[BK03] BOYKOV Y., KOLMOGOROV V.: Computing geodesics and minimal surfaces via graph cuts. In *Proceedings of IEEE International Conference on Computer Vision* (2003), pp. I: 26–33.

[BK04] BOYKOV Y., KOLMOGOROV V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence 26*, 9 (2004), 1124–1137.

[BVZ98] BOYKOV Y., VEKSLER O., ZABIH R.: Markov random fields with efficient approximations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (1998), pp. 648–655.

[BVZ01] BOYKOV Y., VEKSLER O., ZABIH R.: Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 23*, 11 (2001), 1222–1239.

[DJP*92] DAHLHAUS E., JOHNSON D. S., PAPADIMITRIOU C. H., SEYMOUR P. D., YANNAKAKIS M.: The complexity

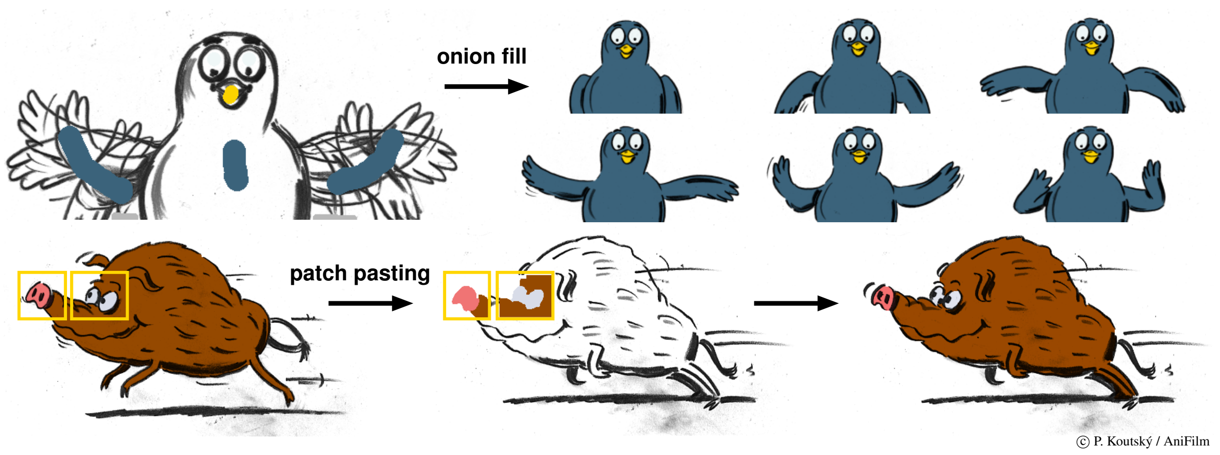

Figure 9: *LazyBrush for the ink-and-paint pipeline – a whole animation can be painted simultaneously in the onion fill setting using a few soft scribbles (top), auto-painting by patch pasting [SBv04] is possible even without pre-segmentation (bottom).*

of multiway cuts. In *Proceedings of ACM Symposium on Theory of Computing* (1992), pp. 241–251.

[FB84] FISHKIN K. P., BARSKY B. A.: A family of new algorithms for soft filling. *ACM SIGGRAPH Computer Graphics 18*, 3 (1984), 235–244.

[Fit54] FITTS P. M.: The information capacity of the human motor system in controlling the amplitude of movement. *Journal of Experimental Psychology 47*, 6 (1954), 381–391.

[Gra06] GRADY L.: Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence 28*, 11 (2006), 1768–1783.

[Hor02] HORIUCHI T.: Estimation of color for gray-level image by probabilistic relaxation. In *Proceedings of IEEE International Conference on Pattern Recognition* (2002), pp. 867–870.

[Joh02] JOHNSTON S. F.: Lumo: Illumination for cel animation. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering* (2002), pp. 45–52.

[Kan79] KANIZSA G.: *Organization in Vision*. Praeger, New York, 1979.

[KKS*04] KARGER D. R., KLEIN P., STEIN C., THORUP M., YOUNG N. E.: Rounding algorithms for a geometric embedding of minimum multiway cut. *Mathematics of Operations Research 29*, 3 (2004), 436–461.

[LCD06] LUFT T., COLDITZ C., DEUSSEN O.: Image enhancement by unsharp masking the depth buffer. *ACM Transactions on Graphics 25*, 3 (2006), 1206–1213.

[Lie78] LIEBERMAN H.: How to color in a coloring book. *ACM SIGGRAPH Computer Graphics 12*, 3 (1978), 111–116.

[LLW04] LEVIN A., LISCHINSKI D., WEISS Y.: Colorization using optimization. *ACM Transactions on Graphics 23*, 3 (2004), 689–694.

[LWCO*07] LUAN Q., WEN F., COHEN-OR D., LIANG L., XU Y.-Q., SHUM H.-Y.: Natural image colorization. In *Proceedings Eurographics Symposium on Rendering* (2007), pp. 309–320.

[MB99] MORTENSEN E. N., BARRETT W. A.: Toboggan-based intelligent scissors with a four parameter edge model. In *Proceedings of IEEE Computer Vision and Pattern Recognition* (1999), pp. II: 452–458.

[MH80] MARR D., HILDRETH E. C.: Theory of edge detection. In *Proceedings of Royal Society* (1980), vol. B207, pp. 187–217.

[Pot52] POTTS R.: Some generalized order-disorder transformation. In *Proceedings of Cambridge Philosophical Society* (1952), vol. 48, pp. 106–109.

[QST*05] QIU J., SEAH H. S., TIAN F., WU Z., CHEN Q.: Feature- and region-based auto painting for 2D animation. *The Visual Computer 21*, 11 (2005), 928–944.

[QWH06] QU Y., WONG T. T., HENG P. A.: Manga colorization. *ACM Transactions on Graphics 25*, 3 (2006), 1214–1220.

[RK82] ROSENFELD A., KAK A. C.: *Digital Picture Processing*, vol. 1. Academic Press, Orlando, USA, 1982.

[RSST96] ROBERTSON N., SANDERS D. P., SEYMOUR P., THOMAS R.: Efficiently four-coloring planar graphs. In *Proceedings of ACM Symposium on Theory of Computing* (1996), pp. 571–575.

[SBv04] SÝKORA D., BURIÁNEK J., ŽÁRA J.: Unsupervised colorization of black-and-white cartoons. In *Proceedings of the 3rd International Symposium on Non-photorealistic Animation and Rendering* (2004), pp. 121–127.

[SBv05] SÝKORA D., BURIÁNEK J., ŽÁRA J.: Colorization of black-and-white cartoons. *Image and Vision Computing 23*, 9 (2005), 767–782.

[SC94] SEAH H. S., CHUA B. C.: A skeletal line joining algorithm. In *Proceedings of the Computer Graphics International* (1994), pp. 62–73.

[SF00] SEAH H. S., FENG T.: Computer-assisted coloring by matching line drawings. *The Visual Computer 16*, 5 (2000), 289–304.

[Smi79] SMITH A. R.: Tint fill. *ACM SIGGRAPH Computer Graphics 13*, 2 (1979), 276–283.

[WBC*05] WANG J., BHAT P., COLBURN R. A., AGRAWALA M., COHEN M. F.: Interactive video cutout. *ACM Transactions on Graphics 24*, 3 (2005), 585–594.

[XAB07] XU N., AHUJA N., BANSAL R.: Object segmentation using graph cuts based active contours. *Computer Vision and Image Understanding 107*, 3 (2007), 210–224.

[Yeh01] YEH W.-C.: A simple algorithm for the planar multiway cut problem. *Journal of Algorithms 39* (2001), 68–77.

[YS06] YATZIV L., SAPIRO G.: Fast image and video colorization using chrominance blending. *IEEE Transactions on Image Processing 15*, 5 (2006), 1120–1129.

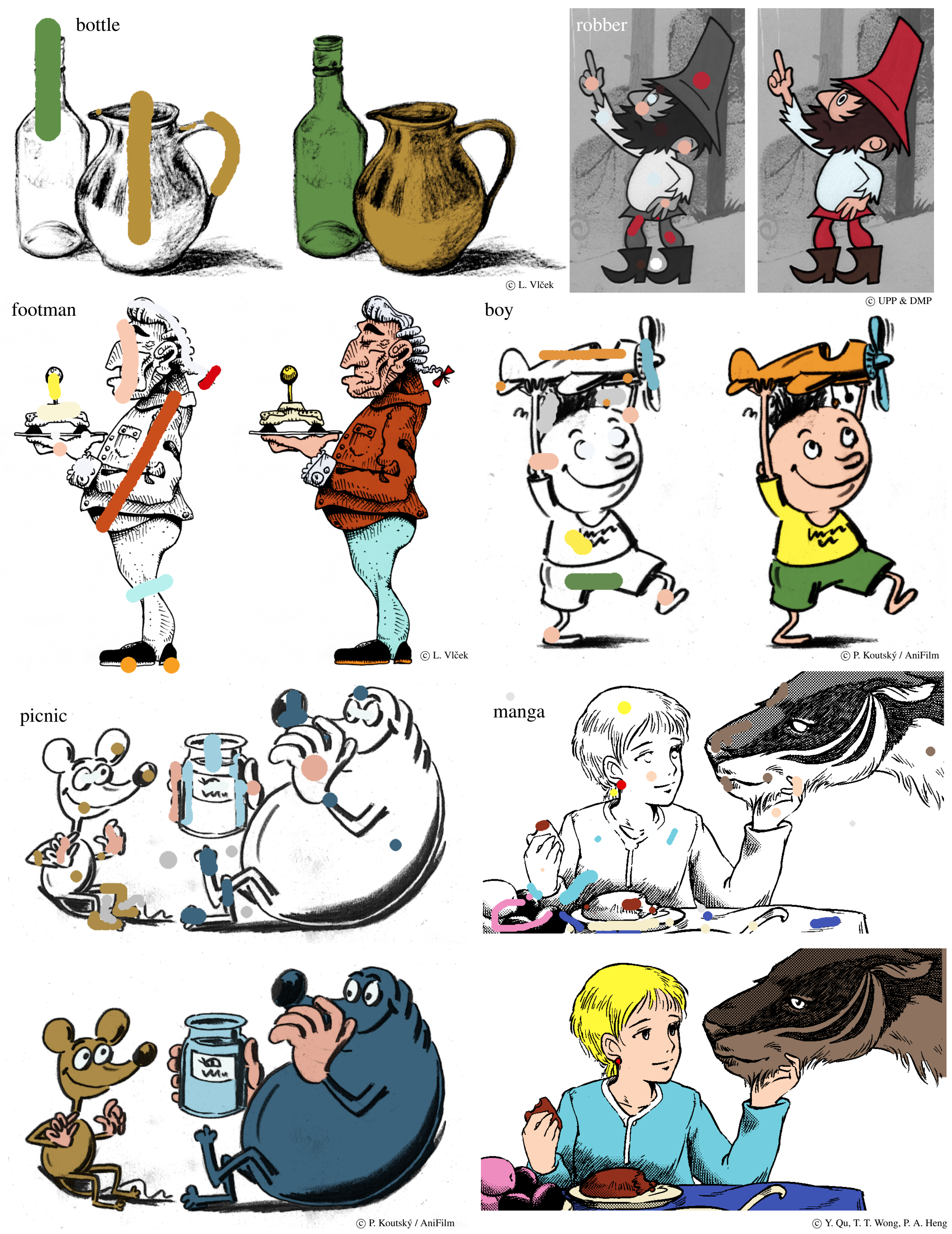

Figure 10: *LazyBrush paintings – in each example see user-specified scribbles superimposed on the original drawing and the corresponding optimized painting (names correspond to Table 1). Background scribbles are sometimes invisible since they are implicitly drawn around the image border. Most color scribbles are soft therefore they can be positioned roughly.*