Unsupervised Colorization of Black-and-White Cartoons

Daniel Sýkora*
Czech Technical University in Prague

Jan Buriánek†
Digital Media Production

Jiří Žára‡
Czech Technical University in Prague

Figure 1: Color has been applied on the grey-scale image in the middle without user intervention using left color image as an example. Source images courtesy © *Vít Komrzí*, *Universal Production Partners* & *Digital Media Production*.

Abstract

We present a novel *color-by-example* technique which combines image segmentation, patch-based sampling and probabilistic reasoning. This method is able to automate colorization when new color information is applied on the already designed black-and-white cartoon. Our technique is especially suitable for cartoons digitized from classical celluloid films, which were originally produced by a paper or cel based method. In this case, the background is usually a static image and only the dynamic foreground needs to be colored frame-by-frame. We also assume that objects in the foreground layer consist of several well visible outlines which will emphasize the shape of homogeneous regions.

CR Categories: I.2.6 [Artificial Intelligence]: Learning (Analogies); I.3.4 [Computer Graphics]: Graphics Utilities (Graphics editors); I.4.6 [Image Processing and Computer Vision]: Segmentation (Edge and feature detection); J.5 [Computer Applications]: Arts and Humanities (Fine arts);

Additional Keywords: image processing, image registration, image segmentation, image analogies, color-by-example, patch-based sampling, probabilistic relaxation

1 Introduction

In the history of cartoon-making, it is possible to discover really valuable and artistically advanced work which stand up in front of the world-wide, modern cartoon production. They provide an invaluable source of imagination for each new generation of children.

Old cartoons were often shot in black-and-white on classical celluloid film which are now usually stored in depositories with non-optimal humidity conditions. When one wants to restore them, it is necessary to end the progressive destruction and convert them to a digital format.

Digital image processing results in significant reduction of time consuming handmade work connected with movie restoration, and also allows us to apply missing color information into the already designed black-and-white world. The main motivation for this modification is the well known virtue of color enhancement to specific artistic impressions which are well perceived, especially by an adolescent audience.

Unfortunately a semi-automatic toolbox able to simplify color transfer into the sequence of grey-scale images is not available in commercial cartoon authoring systems (e.g. *Toonz*,[1] *FlipBook*, [2] etc.). An artist who wants to color black-and-white cartoon usually has to focus on featureless, repetitive work which prevents him from doing really creative artwork.

In this paper we show that using our novel example-based technique, one is able to automate colorization pipeline, reduce the amount of hand-made intervention and make the whole process temporarily feasible and thus cost effective. We bring forward an interactive performance and straightforward implementation by which our method may be successfully incorporated into the existing cartoon authoring system.

This paper is organized as follows. In the first instance we present an overview of existing colorization approaches and address the main disadvantages which led us to develop our novel approach. Next, we describe in detail our method including implementation details and optimization issues. Finally we present several experiments performed on real cartoon images to confirm the efficiency of the proposed solution.

2 Previous work

Colorization has been extensively studied in the movie industry since 1970's. Brute force or various semi-automatic analogue techniques have been used to accomplish this challenging task [Markle 1984]. In this section we focus on digital colorization only.

2.1 Luminance keying

To transfer color into the grey-scale image it is possible to use *luminance keying* [Gonzalez and Woods 1987]. This simple approach exploits the user defined look-up table which converts each level of

*e-mail: sykorad@fel.cvut.cz
†e-mail: burianek@dmp.cz
‡e-mail: zara@fel.cvut.cz

[1] http://www.toonz.com
[2] http://www.digicelinc.com

grey-scale intensity into the specified hue, saturation and brightness. The problem arises when one wants to apply different colors at the same intensity level. It is usually possible to overcome this limitation using simultaneously a few luminance keys for different manually segmented regions. This tedious process significantly increases the amount of hand driven work. However, luminance keying is usually only one way how to apply color on the grey-scale image in recent commercial post-production systems (e.g. *Combustion*,[3] *After Effects*,[4] etc.) and is extensively used.

2.2 Image analogies

A more advanced technique [Welsh et al. 2002] exploits *textural information*. It is inspired by a method of color transfer between images [Reinhard et al. 2001] and by the framework of image analogies [Hertzmann et al. 2001]. This technique transfers color to the grey-scale image from the already colorized example using local luminance distribution matching in $l\alpha\beta$ color space. A subset of representative swatches in the color and in the grey-scale image is selected manually or using jitter sampling. This technique is surprisingly successful when it is applied to some specific natural scenarios. However homogeneous regions in black-and-white cartoons do not provide textural information. In this instance luminance distribution matching reduces to the single intensity level comparison which has the same properties as luminance keying.

2.3 Motion estimation

Recently *Pan et al.* [2004] introduced a novel method which is based on the motion estimation. They assume that the same objects between adjacent frames move slightly hence it is possible to use optical flow to track this motion and assign the corresponding chromatic information from reference color frames. This method is similar to video compression algorithms where spatial and temporal correlation between consecutive frames provide possibility to store color information only in several key frames.

However in cartoons motion seems to be coarse in contrast to real videos (see Figure 2). The same animation phase is usually shot twice using two consecutive frames. Accordingly the structural differences between the current and the new animation phase becomes really noticeable. It is usually impossible to track this rapid motion using optical field estimation.

Figure 2: Common example of structural differences between two consecutive animation phases. Arrows point to the important topology changes and small holes.

[3] http://www.discreet.com

[4] http://www.adobe.com

2.4 Image segmentation

In order to simplify color transfer into the black-and-white cartoon *Sýkora et al.* [2003] suggest to use an unsupervised *image segmentation*. This method is suitable especially for cartoons which consist of two planar layers (background and foreground). The dynamic foreground layer contains homogeneous regions surrounded by visible outlines and the background layer is usually a more complicated textural image which remains static during the animation. This important property allows us to divide the original grey-scale image into the set of regions using robust outline detector [Sýkora et al. 2003] and classify them roughly as foreground or background via region size thresholding (see Figure 3).

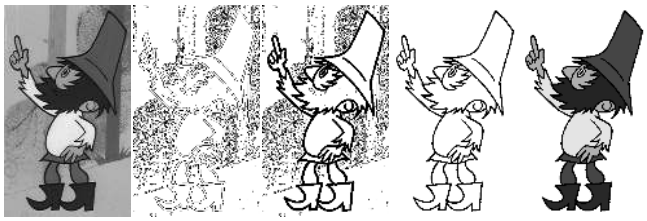

Figure 3: Segmentation in progress (from left to right) – original image, edge detection, outline detection, outline extraction, and the final segmentation.

Information about the scene structure allow us to reconstruct one big image which contains only the visible area of the whole background layer (see Figure 4). On such image the color transfer is applied only once using the standard image manipulation software or some specialized colorization tool (see *BlackMagic*[5]).

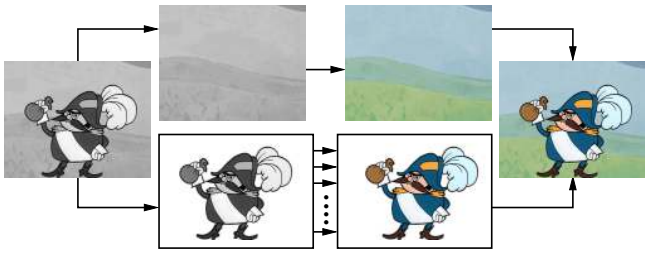

Figure 4: Scene separation – static background layer (top) needs to be managed only once in contrast to dynamic foreground layer (bottom) which is colorized frame-by-frame.

In the dynamic foreground layer color is applied frame-by-frame. However the assumption is that at least one animation frame is correctly colorized by a color expert. This means that each foreground region has associated with one index from the palette of available colors and it is possible to predict color-to-region assignment for the rest of the sequence using already colored frames as an example.

Finally color composition of each animation frame is made by pasting previously extracted and already colorized foregrounds into the correct position on the reconstructed and colorized background (see Figure 4).

2.5 Color prediction

The simplest way how to predict color-to-region assignment in the foreground layer is to use *position-based* prediction. In this method the extent of spatial intersection between target and already colored regions in the example frame determine which color index will be propagated to the target region (see Figure 5). To increase prediction performance it is useful to select the best example phase from the database of already colored frames using the sum of absolute

[5] http://www.neuraltek.com/bmagic

differences and pose alignment based on the *log-polar phase correlation* [Reddy and Chatterji 1996]. This technique is able to estimate optimal shift and in some special cases also the rotation and scale alignment when correlation is applied only to the foreground layer to hide possible structural differences in the background layer.

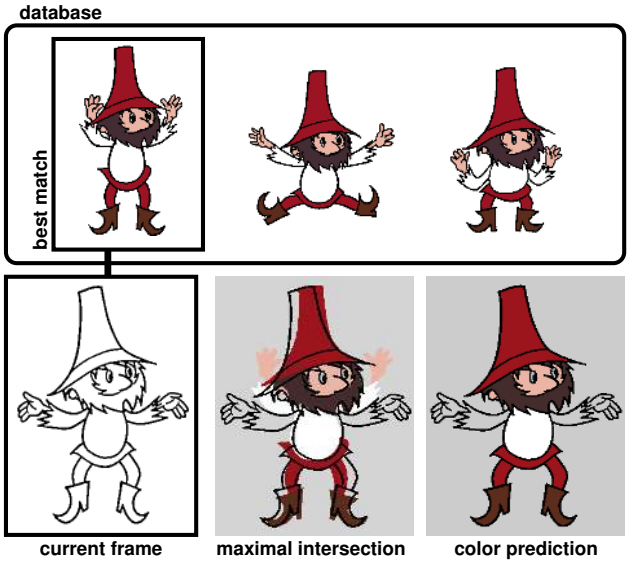

Figure 5: Position-based prediction – the best animation phase is selected from the database and color index from example region which has the largest spatial intersection with target region is propagated.

Usually previous animation frame is selected to be the best example frame. However when the target animation is looped or consists of several phases randomly used during animation sequence, this approach automatically retrieves the best example phase from the sequence of already colored frames and transfers complete color information to the new animation frame. However for completely novel frames this method produce usually large derangement.

Another useful color prediction technique is based on the *grey-scale intensity*. It compares intensity medians of example regions with medians of target regions and propagates color indices between regions which have the minimal median difference. Usually this method produces better results when it is applied on completely new frames in contrast with position-based prediction because of the independence of the region shape and its spatial location. However like luminance keying this method suffers from wrong prediction when different colors are assigned to the similar intensity.

A more advanced color transferring scheme exploit *region-based* prediction which has been originally developed to automate cel painting. This problem is even more complicated in contrast to black-and-white cartoon colorization due to missing grey-scale intensity cue. To match regions in order to transfer color information *Madeira et al.* [1996] use region shape similarity, *Chang* and *Lee* [1997] include topological relations, and *Seah* and *Feng* [2000] proposed position-based prediction where pose alignment is done by motion estimation similarly to *Pan et al.* [2004].

Unfortunately these methods perceive cartoon drawing as a planar image. Occlusion and other topology variations imposed by virtual depth (see Figure 2) usually destroy important region shape features and spatial relations. To overcome this limitation *Qiu et al.* [Qiu et al. 2003] use couple of predefined *master frames* to define pseudo 3D appearance of individual characters. In our case this modification is not available. However it is possible to select couple of dissimilar frames from the whole animation sequence and treat them as pseudo master frames.

3 Unsupervised colorization

In this section we present our novel *feature-based* color transferring scheme which exploit image segmentation, patch-based sampling and probabilistic relaxation. Our goal is to eliminate some of the problems illustrated in the previous section. To understand better we redefine our problem.

Problem statement: *We have two segmented frames. First frame serves as a color example where each region has one color index assigned from the user-defined palette. The second frame contains unlabelled target regions. Our task is to assign color indices to target regions similarly to as they are assigned in the example frame (see Figure 1).*

3.1 Patch-based sampling

To accomplish this task we use patch-based sampling. Our approach is similar to the *segmentation-by-example* technique presented in [Borenstein and Ullman 2002] where an *optimal cover* strategy has been used. This exhaustive approach is not necessary for our purpose. Instead we deterministically select a subset of image patches which belong to important outline features (see Figure 6) and we match correspondent features between the example and the target frame using patch-based structural similarity.

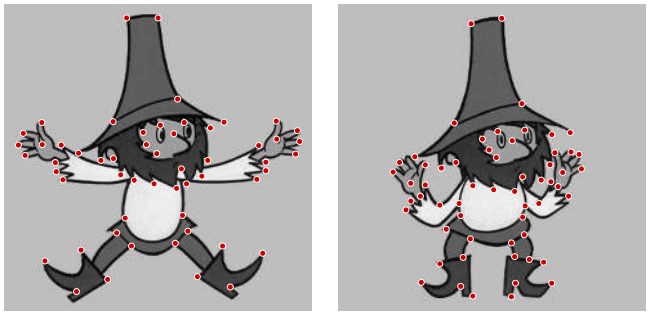

Figure 6: Feature extraction – high curvature points and junctions.

Before feature extraction begins we mask the background layer to avoid any detection of points which are not important for our purpose. Afterwards we apply *Kanade-Lucas-Tomasi* feature extractor [Tomasi and Kanade 1991; Birchfield 1998] to mark out a predefined number of well scattered features. We also exploit the simple bottom-up clustering scheme to reduce any coverage redundancy.

3.2 Structural matching

To find out the best structural correspondences between example and target features we use a *backward mapping*. This single direction matching technique is able to retrieve a good example patch for each target feature.

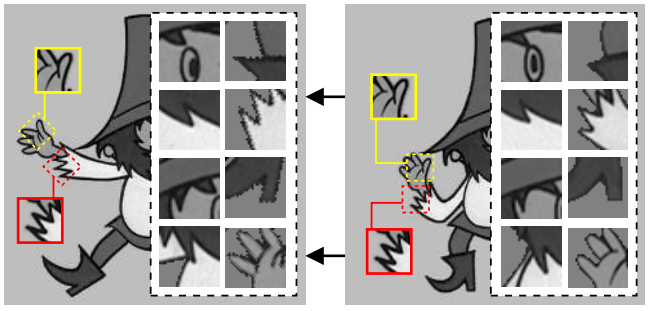

Figure 7: Structural matching – target patches are transformed to fit the best matching location in the example frame. See samples of the best matching pairs (dashed boxes) where example patches (left box) have already been transformed to emphasize structural similarity.

It is also possible to apply reverse *forward mapping* or symmetric *one-to-one mapping*. Unfortunately these two approaches significantly reduce the prediction performance, because forward mapping does not guarantee coverage of all target features and one-to-one mapping avoids any possibility of reusing one example patch several times.

To estimate the degree of structural similarity between two image patches we use normalized sum of squared differences (L_2-norm):

$$D(P_e,P_t)=\frac{1}{w^2}\sum_{x=1}^{w}\sum_{y=1}^{w}(P_e(x,y)-P_t(x,y))^2, \tag{1}$$

where $D(P_e,P_t)$ is structural difference ratio of example P_e and target P_t square patches w pixels wide and high.

Example features serve as a good starting points for the final alignment of target patches. We align each target patch to fit the best matching location close to the corresponding example feature (see Figure 7). Such alignment provides us to transfer color information pixel-to-pixel from the example to the target image.

We assume that the distance between features is able to be smaller than the size of patches. Due to this property, it usually occurs that the target and example patches overlap each other. To avoid the loss of valuable information we introduce a *quality buffer* which we describe later in Section 3.3.

Due to computational feasibility we suppose that arbitrary deformations are large enough in comparison with patch size. We also assume that the scales of the example and the target image are similar. This assumption is realistic since the camera field-of-view usually remains static during animation. When it changes we perform global image re-scaling to reach the same scale. This simplification allows us to use the rigid transformation model:

$$\begin{aligned} x' &= x\cos\alpha - y\sin\alpha + x_0 \\ y' &= x\sin\alpha + y\cos\alpha + y_0 \end{aligned} \tag{2}$$

Unknown rotation (α) is estimated using an exhaustive search over the set of all example patches in all possible orientations. Hierarchical block motion estimation [Nam et al. 1995] is then used to obtain the final translation alignment (x_0,y_0) in the small neighborhood of the pre-selected example feature (for details see Section 4).

3.3 Patch pasting

When the best matching location (x_0,y_0) of each target patch is estimated, rectangular areas with the same patch size are extracted from the same position in the example *color buffer*. Elements of this buffer only contains information about the corresponding regional color.

Figure 8: Patch pasting – quality buffer (left), color buffer after patch pasting (middle), and color buffer after non-maxima suppression (right). For better lucidity we superimpose black outlines onto the color buffer to mark out the original regions. Values in the quality buffer are visualized as follows: pixels with better matching quality have the closer intensity to the same pixel in the original image.

After extraction, we apply inverse rotation ($-\alpha$) and we paste the transformed rectangles into the proper position on the target color buffer (see Figure 8 in the middle). Finally a non-maxima suppression is applied to emphasize the most likely color-to-region assignment. To do this, we accumulate a color histogram for each target region and select the most frequently used color index.

This straightforward approach is not the best due to the possibility of patches overlapping. To overcome this problem we first sort out example patches by a nondecreasing matching quality which guarantees that better matches will be pasted into the color buffer before inferior ones. Additionally to avoid each pixel being overridden by a patch with a lower local matching quality, we use an already introduced quality buffer (see Figure 8 on the left). This technique is similar to *z-buffer* visibility culling:

$$\begin{array}{lll} \textbf{if} & |I_e(x',y')-P_t(x,y)| < Q_t(x,y) & \textbf{then} \\ \{ & Q_t(x,y)=|I_e(x',y')-P_t(x,y)| & \\ & C_t(x,y)=C_e(x',y') & \} \end{array} \tag{3}$$

Before we paste a pixel (x',y') from the transformed color patch C_e we compute the absolute intensity difference between the corresponding pixel (x,y) in the transformed target patch P_t and in the example image I_e. If such differences are smaller than the actual value stored in the quality buffer Q_t, we allow color pasting to be done to the target color buffer C_t and we then update the value in the quality buffer using a lower difference. Otherwise, we simply preserve the original color index and difference.

3.4 Probabilistic relaxation

Proposed patch-based sampling scheme is far limited by the number of local structural correspondences and may completely fail when different colors are applied on a similar structural pattern. In this case the most frequently used color label has a relatively small peak in the final color histogram and consequently two or more color labels have similar frequency and so all of them have similar probability to be assigned for such a region.

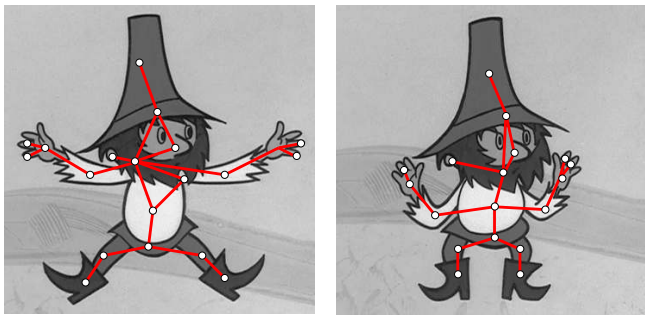

Figure 9: Attributed relational graphs – regions (bullets) and their neighborhood relations (lines) in the example (left) and in the target (right) foreground layer.

However this ambiguity usually disappears if we consider local neighborhood relations between regions in the example frame. To exploit this cue we use *probabilistic relaxation* [Christmas et al. 1995] on planar *attributed relational graphs* (`ARG`) which was proven to be efficient especially for graph matching problems and model-based object recognition [Ahmadyfard and Kittler 2000].

In our case local neighborhood relations in the foreground layer are represented using a single `ARG` (usually discontinuous). `ARG` nodes correspond to individual regions $i \in \mathbf{R}$. If regions $i,j \in \mathbf{R}$ share the same boundary contour then their graph nodes are connected using undirected arc $[i,j]$ (see Figure 9). Additionally each example graph node $i \in \mathbf{R}_E$ has two important attributes: median of region intensity $\hat{I}_i$ and previously assigned color label $\hat{c} \in \mathbf{C}$. This single label is replaced by the vector of a priori probabilities $P_i(c)$

for each color $c \in \mathbf{C}$ in the target graph node $i \in \mathbf{R}_T$. Such vector is initialized immediately after patch-based sampling using normalized frequencies from resulting color histogram.

Our basic relaxation rule computes posterior $P_i^{k+1}(c)$ as a product of a prior $P_i^k(c)$ and neighborhood support function $Q_i^k(c)$ normalized over all possible color-to-region assignments:

$$P_i^{k+1}(c) = \frac{P_i^k(c) \cdot Q_i^k(c)}{\sum_{c_i \in \mathbf{C}} P_i^k(c_i) \cdot Q_i^k(c_i)}. \tag{4}$$

For our purposes we define a novel neighborhood support function $Q_i^k(c)$ as follows:

$$Q_i^k(c) = \sum_{j \in \mathbf{R}_i^+} \sum_{[m,n] \in \mathbf{E}} P_j^k(\hat{c}_n) \cdot S_c([i,j],[m,n]). \tag{5}$$

For target arcs $[i,j] \in \mathbf{T}$ which correspond to local neighborhood of region i we sum up weighted similarity with all example arcs $[m,n] \in \mathbf{E}$. Here the weight factor is a prior probability $P_j^k(\hat{c}_n)$ taken from corresponding neighbor region j, where $\hat{c}_n$ denote to color label assigned in the example node $n \in \mathbf{R}_E$. We use simple arc similarity function $S_c([i,j],[m,n])$ which is defined using this formula:

$$S_c([i,j],[m,n]) = \begin{cases} \exp\left(-\frac{(\hat{I}_i - \hat{I}_m)^2 + (\hat{I}_j - \hat{I}_n)^2}{\sigma_I^2}\right) & \text{iff } c = \hat{c}_m, \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

When the actual color label c is the same one as the label $\hat{c}_m$ assigned to the example node $m \in \mathbf{R}_E$ then the arc similarity is expressed as weighted difference of region intensity medians $\hat{I}$. User defined parameter σ_I affects the tolerance to global or local intensity fluctuation between the example and the target images.

To terminate relaxation it is possible to examine differences between actual and novel posterior probabilities till they fall under some user defined threshold. Afterwards in each region the color label with the maximum posterior probability is selected to produce the most probable color-to-region assignment. Our novel probabilistic relaxation scheme converges very fast by the reason that actually a lot of color labels are assigned with initial probabilities close to one.

4 Implementation issues

In this section we discuss a couple of implementation issues which stand behind the robustness and interactive performance of the proposed method.

First we allow the user to select an optimal patch size. By default we assume the patch size `48x48` which produced an optimal results in our experiments with cartoon images scanned in the full PAL resolution (`720x576`) using standard camera field-of-view where outline thickness varies from 2 to 6 pixels (see Figure 10).

It is also important to extract an optimal number of well scattered features to cover all significant curvature points and junctions in the original image. To fulfil this constraint roughly 200 features per image were sufficient in our experiments. We let the user to trade-off between the computational complexity and the prediction performance (see Figure 11).

Using brute force, backward mapping allows us to perform color prediction in non-interactive times. To reduce the computational overheads we retrieved suboptimal matching pairs in interactive speed using a hierarchical approach based on *quad-tree pyramid* [Liang et al. 2001] and *approximate nearest neighbor* (ANN) search [Arya et al. 1998; Mount and Arya 1998]) where the axis orthogonal *kd*-tree [Friedman et al. 1977] is used as a data partitioning structure.

For each feature in the example frame we extract a patch with the size `68x68` (to fit the horizontal width of the rotated `48x48` rectangle by 45° which equals its diagonal $\lfloor 48\sqrt{2} \rfloor$). Then we compute 16 rotation patches incrementally by the angle of $\frac{\pi}{8} \approx 23°$ radians using bilinear interpolation (see Figure 12 on the right). Such rotated bitmaps are then cropped back to the `48x48` patch and three levels of quad-tree pyramid are precalculated. Finally we store all these patches in the *kd*-tree structure.

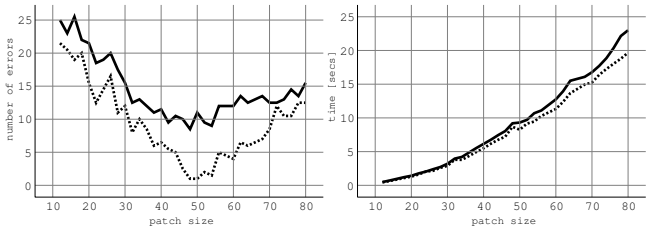

Figure 10: Comparison of the patch size with the number of prediction errors (left) and corresponding execution time for 200 aligned patches (right). Dashed line is for **a)** and solid line for **b)** in Figure 13.

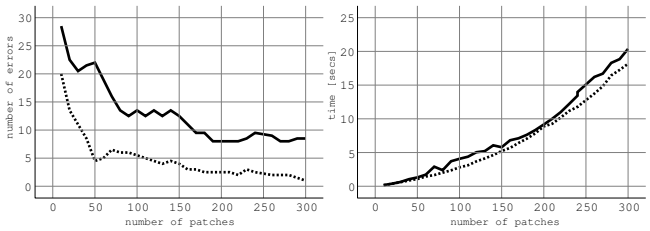

Figure 11: Comparison of the number of patches with the number of prediction errors (left) and corresponding execution time for `48x48` patches (right). Dashed line is for **a)** and solid line for **b)** in Figure 13.

In our experiments we allow displacement of the optimal solution to be ± 1 pixels for translation and $\pm \frac{\pi}{16}$ radians for rotation. An optimal alignment marginally affects the prediction performance, however it significantly increases the computational overhead. This simplification also allows us to precalculate and store only two upper sub-sampled levels of the quad-tree pyramid.

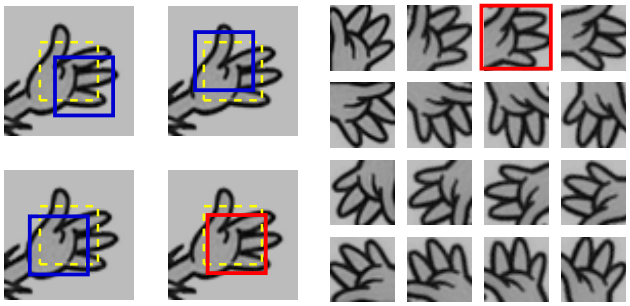

Figure 12: For each target patch we precalculate 16 rotations (right). Rotation patches are aligned on the small area close to the example feature (left). Dashed rectangles depict the search area. Black rectangle (right) indicate the best rotation and white rectangle (left) the best alignment.

Using *k*-ANN search we retrieve eight best example candidates for each target patch. We keep corresponding locations of candidate features in the example image and we align properly oriented target patch to fit the best matching location in a small area (± 8 pixels) close to the example feature. Afterwards we select the best alignment with the lowest structural difference $D(P_e, P_t)$ (see Figure 12 on the left) and we perform inverse patch pasting between color

buffers using patch sorting and quality buffer to avoid overlapping problems.

Notwithstanding that the proposed preprocessing requires negligible computation and storage overheads, it provides a significant speed up during the feature matching phase. On `750MHz CPU` it takes approximately 8 seconds to retrieve the matches and align the 200 patches with size `48x48`, in contrast to 7 minutes of exhaustive searching.

The structural similarity between example and target frame does not depend on the actual color-to-region assignment hence it is possible to execute patch-based sampling as a stand-alone thread during hand driven error correction phase. This parallelism allow us to reach interactive performance in real-time application because in average it takes a couple of seconds to locate and correct possible prediction errors. Patch correspondences for a new animation frame can be pre-calculated during this idle time.

In contrast to patch-based sampling the probabilistic reasoning scheme produce negligible computational overhead. However sometimes resulting error correction is negligible or inferior in contrast to stand-alone patch-based sampling. Due to these circumstances it is useful to let user call probabilistic relaxation on demand as an optional post-processing tool which has ability to correct possible derangement when different colors are applied on a similar structural pattern.

5 Results

Our experiments have been performed on images from old Czech black-and-white cartoon *"O loupežníku Rumcajsovi"* which were originally produced by *Radek Pilař* in 1967. The proposed colorization approach has been implemented as a plug-in for our proprietary semi-automatic PC application that allows us to effectively apply novel color information on the already designed black-and-white cartoons.

The whole colorization pipeline consists of five independent phases: image segmentation, foreground layer color prediction, color brightness modulation, background layer reconstruction and colorization, and the final composition. In this paper we focus on color prediction only. Selected results are presented in Figure 13. They readily confirm that the proposed method significantly reduces the amount of hand driven correction needed in contrast to the pure intensity-based approach.

However prediction performance of our method is greatly limited by the number of local structural matches between the example and target images. It may completely fail when the example and target images are widely dissimilar.

6 Conclusion

In this paper we have introduced a novel color-by-example technique which successfully extends the toolbox of existing example-based image processing approaches. The proposed method automates colorization of already designed black-and-white cartoons. It significantly reduces tedious manual work in contrast to widely used brute force, intensity-based and position-based approaches. Using our color transferring scheme one is able to make colorization of aged black-and-white cartoons cost effective and temporarily feasible.

7 Acknowledgements

The authors would like to thank *Jiří Bittner* and *Tomáš Pajdla* for many useful comments. Images in this paper are published by the courtesy of © *Vít Komrzí*, *Universal Production Partners* and *Digital Media Production*. This work has been partly supported by the Ministry of Education, Youth and Sports of the Czech Republic under the research program No. Y04/98: 212300014 (Research in the area of information technologies and communications), and under the student research program FRVŠ-2004-2067.

References

AHMADYFARD, A., AND KITTLER, J. 2000. Region-based object recognition: Pruning multiple representations and hypotheses. In *Proceedings of British Machine Vision Conference*, 745–754.

ARYA, S., MOUNT, D. M., NETANYAHU, N. S., SILVERMAN, R., AND WU, A. Y. 1998. An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. *Journal of the ACM 45*, 6, 891–923.

BIRCHFIELD, S., 1998. An implementation of the Kanade-Lucas-Tomasi feature tracker. `http://www.vision.stanford.edu/~birch/klt`.

BORENSTEIN, E., AND ULLMAN, S. 2002. Class-specific, top-down segmentation. In *Proceedings of European Conference on Computer Vision*, 109–122.

CHANG, C. W., AND LEE, S. Y. 1997. Automatic cel painting in computer-assisted cartoon production using similarity recognition. *The Journal of Visualization and Computer Animation 8*, 165–185.

CHRISTMAS, W. J., KITTLER, J., AND PETROU, M. 1995. Structural matching in computer vision using probabilistic relaxation. *IEEE Transactions on Pattern Recognition and Machine Intelligence 17*, 8, 749–764.

FRIEDMAN, J., BENTLEY, J., AND FINKEL, R. 1977. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematics Software 3*, 3, 209–226.

GONZALEZ, R. C., AND WOODS, R. E. 1987. *Digital Image Processing*, 2nd ed. Addison-Wesley Publishing, Reading, Massachusetts.

HERTZMANN, A., JACOBS, C. E., OLIVER, N., CURLESS, B., AND SALESIN, D. H. 2001. Image analogies. In *SIGGRAPH 2001 Conference Proceedings*, 327–340.

LIANG, L., LIU, C., XU, Y.-Q., GUO, B., AND SHUM, H.-Y. 2001. Real-time texture synthesis by patch-based sampling. *ACM Transactions on Graphics 20*, 3, 127–150.

MADEIRA, J. S., STORK, A., AND GROB, M. H. 1996. An approach to computer-supported cartooning. *The Visual Computer 12*, 1–17.

MARKLE, W. 1984. The development and application of colorization. *SMPTE Journal*, 632–635.

MOUNT, D., AND ARYA, S., 1998. Approximate nearest neighbor library. `http://www.cs.umd.edu/~mount/ANN`.

NAM, K. M., KIM, J.-S., PARK, R.-H., AND SHIM, Y. S. 1995. A fast hierarchical motion vector estimation algorithm using mean pyramid. *IEEE Transactions on Circuits and Systems for Video Technology 5*, 4, 344–351.

PAN, Z., DONG, Z., AND ZHANG, M. 2004. A new algorithm for adding color to video or animation clips. In *Proceedings of WSCG – International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 515–519.

QIU, J., SEAH, H. S., TIAN, F., CHEN, Q., AND MELIKHOV, K. 2003. Computer-assisted auto coloring by region matching. In *Proceedings of Pacific Conference on Computer Graphics and Applications*, 175–185.

REDDY, B. S., AND CHATTERJI, B. N. 1996. An FFT-based technique for translation, rotation and scale-invariant image registration. *IEEE Transactions on Computers 5*, 8, 1266–1271.

REINHARD, E., ASHIKHMIN, M., GOOCH, B., AND SHIRLEY, P. 2001. Color transfer between images. *IEEE Transactions on Computer Graphics and Applications 21*, 5, 34–41.

SEAH, H. S., AND FENG, T. 2000. Computer-assisted coloring by matching line drawings. *The Visual Computer 16*, 289–304.

SÝKORA, D., BURIÁNEK, J., AND ŽÁRA, J. 2003. Segmentation of black and white cartoons. In *Proceedings of Spring Conference on Computer Graphics*, 245–254.

TOMASI, C., AND KANADE, T. 1991. Shape and motion from image streams: A factorization method part 2. detection and trcking of point features. Tech. Rep. CMU-CS-91-132, Carnegie Mellon University, School of Computer Science.

WELSH, T., ASHIKHMIN, M., AND MUELLER, K. 2002. Transferring color to greyscale images. In *SIGGRAPH 2002 Conference Proceedings*, 277–280.

Figure 13: Unsupervised colorization in progress. From left to right in every row: example color buffer, target color buffer, intensity-based prediction for comparison and final target color buffer after non-maxima suppression. Prediction errors are emphasized using gray regions with dot markers. Interesting properties of individual examples: **a)** Intensity-based prediction is not able to detect holes in contrast to our novel approach. **b)** Intensity-based prediction completely fails when the scene consists of regions with similar intensity levels and when different colors are assigned to the same level. **c)** In this case intensity levels of all shoes are similar, hence only matching based on the scene structure is able to determine which color is more suitable for a particular shoe. **d)** Our algorithm still behaves satisfactorily in the presence of changes that are not modelled by the proposed transformation model (mirroring effect). **e)** In this case the quality buffer has not been used. Example patches were pasted only in decreasing order of matching quality. The result is still superior in contrast to the intensity-based prediction.