Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Computer Graphics and Interaction

Bridging the Gap Between Computer-Generated and Hand-Drawn Animation

Marek Dvorožňák

A dissertation submitted to
the Faculty of Electrical Engineering, Czech Technical University in Prague,
in partial fulfilment of the requirements for the degree of
Doctor of Philosophy.

Ph.D. programme: Electrical Engineering and Information Technology
Branch of study: Information Science and Computer Engineering

Supervisor: Daniel Sýkora

April 2019

Bridging the Gap Between Computer-Generated and Hand-Drawn Animation

Marek Dvorožňák
dvoromar@fel.cvut.cz
Department of Computer Graphics and Interaction
Faculty of Electrical Engineering
Czech Technical University in Prague
Karlovo nam. 13, 121 35 Prague 2, CZ

Abstract

Animated feature-length movies produced by renowned studios such as Disney Animation or Pixar are nowadays almost exclusively created on computers using various advanced digital techniques such as physically-based rendering or simulation. Those algorithmic solutions make the movies highly appealing to the audience and substantially increase the level of automation in the production process which leads to a visible unification of the motion as well as appearance stylization. In contrast, traditional animation techniques that were used in the production of animated feature films formerly allowed artists to express better their unique artistic vision enabling them to work with physical, artistic media and have full creative freedom over the production process. Nowadays, due to their extreme time demands, traditional animation techniques are utilized only marginally. Nevertheless, the recent release of several critically acclaimed movies which combine digital pipeline with hand-drawn animation indicates an emerging trend to bring features of the traditional techniques back in the game.

In this thesis, we propose a set of tools that enable to combine the benefits of computer-assisted production with the freedom of manual work. With these tools, an artist can create traditional hand-drawn animations that have appearance comparable to computer-generated movies or use automation of repetitive manual tasks while still retaining the original hand-drawn look. To enable this connection, we developed and practically verified several new techniques that allow to process and decompose hand-drawn input into a set of reusable structures. Those can be utilized for the creation of a new content that retains the unique artistic style and preserves essential visual characteristics of the used artistic media. In particular, we developed: (1) a method for cleaning up a time-lapse video of a painting process and decomposing it into a set of translucent layers; and (2) a method for extracting character's appearance and motion characteristics from hand-drawn animations. We show how to utilize those extracted structures in various applications including the synthesis of novel hand-drawn animations and non-trivial editing of real-world paintings. We also present a method for reconstruction of 3D models from a drawing which enables enhancement of hand-drawn animations with physically-based rendering effects.

This thesis is presented as a collection of four research papers which describe the proposed approaches. Three articles were published in a journal with impact factor and one in conference proceedings.

Keywords

hand-drawn animation, computer-generated animation, time-lapse video, decomposition into layers, 3D reconstruction, example-based synthesis, skeletal animation, style transfer, stylization, artistic freedom

Acknowledgements

I would like to express my gratitude to my advisor Daniel Sýkora for his kind guidance and persistent support throughout the studies. Without his precious ideas and advice, my research would have been much more difficult. I am grateful to my internship mentors Pascal Barla, Pierre Bénard, Wil Li, Vova Kim, and also to my other collaborators for trusting in me and helping me with the research. I would like to give special thanks to the artists, especially to Zuzana Studená, who provided us with all the amazing pictures and animations which served as input data for our methods. At last but not least, I want to thank my parents, friends, and colleagues for support and a bit of needed distraction from work.

The research presented in this thesis was conducted in collaboration with, and supported by, Adobe Research and has further been supported by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center), by the Grant Agency of the Czech Technical University in Prague, grants No. SGS13/214/OHK3/3T/13 (Research of Progressive Computer Graphics Methods) and No. SGS16/237/OHK3/3T/13 (Research of Modern Computer Graphics Methods), by Czech Science Foundation under research program P202/12/2413 (OPALIS), and by Research Center for Informatics No. CZ.02.1.01/0.0/0.0/16_019/0000765.

Metody propojení počítačem generované a ručně kreslené animace

Marek Dvorožňák
`dvoromar@fel.cvut.cz`
Katedra počítačové grafiky a interakce
Fakulta elektrotechnická
České vysoké učení technické v Praze
Karlovo náměstí 13, 121 35 Praha 2

Abstrakt

Celovečerní kreslené filmy z dílen renomovaných animačních studií, jako je Disney Animation nebo Pixar, jsou v dnešní době téměř výlučně tvořeny na počítačích s využitím pokročilých digitálních technik, jakými jsou např. realistická syntéza obrazu nebo fyzikální simulace. Tyto algoritmy dodávají filmům vysoce atraktivní vizuální podobu a zároveň zvyšují úroveň automatizace celého výrobního procesu. Dochází tak k viditelnému sjednocení stylizace a to jak na úrovni vzhledu, tak i z hlediska výsledného pohybu animovaných postav. V kontrastu k počítačovým metodám stojí tradiční animační techniky, jež byly používány při výrobě kreslených filmů dříve. Jejich hlavní výhodou oproti digitální animaci byl fakt, že skrze práci s fyzickými výtvarnými médii umožnily animátoru plnou tvůrčí svobodu. V dnešní době jsou však kvůli extrémním časovým nárokům využívány pouze okrajově. Nedávný úspěch animovaných filmů, jež kombinují digitální techniky s ruční animací, ale naznačuje vznikající trend, který přináší tradiční kreslenou animaci zpět do hry.

V této disertační práci představujeme soubor nových algoritmů, které umožní kombinovat výhody počítačem podporované tvorby se svobodou ruční práce. Díky navrženým nástrojům může umělec vytvářet tradiční ručně kreslené animace mající vzhled srovnatelný s filmy generovanými na počítači nebo naopak využít automatizaci opakujících se ručních úkonů při zachování vzhledu původní ruční kresby. K dosažení tohoto cíle bylo nutné vyvinout a prakticky ověřit několik nových algoritmů. Ty umožní dekomponovat ručně nakreslený vstup do souboru elementárních struktur, které lze následně využít ke generování nového vizuálního obsahu. Konkrétně jsme vyvinuli algoritmus pro čištění časosběrného videa zaznamenávajícího proces malby a jeho následné rozložení do sady poloprůhledných vrstev. Dále jsme navrhli algoritmus pro extrakci základních vizuálních charakteristik a jednotlivých pohybových prvků z ručně kreslených animací. S jejich využitím lze následně generovat nové kreslené animace nebo provádět netriviální úpravy na fyzických malbách tak, že zůstane zachován původní výtvarný ráz. Představujeme také metodu rekonstrukce 3D modelů z ruční kresby, jež umožňuje dodat kreslené animaci atraktivní fotorealistický vzhled běžný pro počítačem generované obrazy.

Tato práce je prezentována jako soubor čtyř článků popisujících navrhované přístupy. Tři z těchto článků byly publikovány v impaktovaném časopise a jeden na konferenci ve sborníku.

Klíčová slova

ručně kreslená animace, počítačem generovaná animace, časosběrné video, rozklad na vrstvy, 3D rekonstrukce, syntéza založená na příkladu, skeletální animace, přenos stylu, stylizace, umělecká svoboda

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

Animation is a popular medium for communication and entertainment. Its dynamic nature and the ability to convey stylization makes it attractive for the audience and helps to communicate feelings, emotions, and ideas. Animation techniques evolve over the last century starting with traditional hand-drawn and stop-motion animation. In hand-drawn animation, every frame is created by hand using physical, artistic media whereas in stop-motion animation, pre-created 2D or 3D physical models are manipulated in each animation frame. Despite the fact that both early animation techniques are extremely labor-intensive and require highly experienced artists, their advantage is hidden in the fact that all strokes or manipulations are performed in a physical world which provides unlimited creative possibilities and gives the artists complete freedom to develop their own distinctive ideas.

In the past, traditional techniques were the standard for production of animated feature films; nevertheless, with the advent of computers, a constantly growing effort has been devoted into utilizing computer power to facilitate the process of animation production. Physical tools were slowly replaced by their digital counterparts like a computer tablet or animation software. Advances in computer hardware and algorithms brought new opportunities for employing computer in labor-intensive tasks such as inbetweening (the creation of intermediate poses between keyframes) or filling the hand-drawn outlines with colors. The computer assistance started to make the animation production less time-consuming and boosted its development.

Nowadays, computer-generated (CG) animation is understood as a descendant of traditional animation. It typically uses 2D or 3D digital models and exploits algorithms designed to simplify the creation of models' movement and appearance. There are several reasons for the preference of CG animation by the audience and also the movie industry [A. Jones and Oliff 2006]. Recent advances in algorithms utilized in various stages of CG animation pipeline and also performance boost in computer hardware enabled 3D CG animation to include rich visual effects including stereoscopic vision. Those features have brought a higher level of entertainment into animations and made them more appealing to the audience which naturally resulted in higher cinema attendances. Moreover, the ability to reuse assets, create different views of a scene using virtual cameras, compute physically plausible effects by utilizing a computer simulation, or to transfer

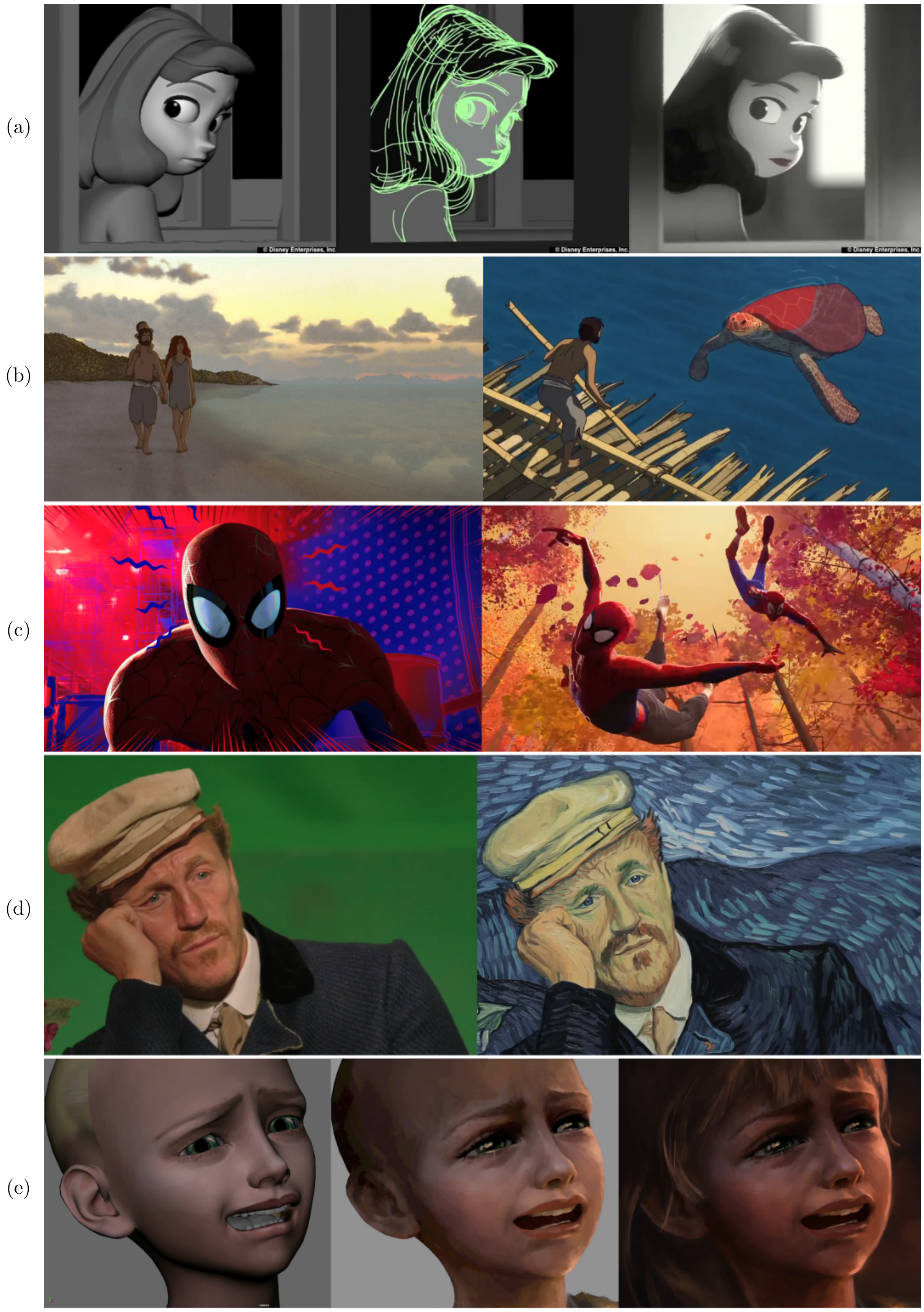

(a)
© Disney Enterprises, Inc
© Disney Enterprises, Inc
© Disney Enterprises, Inc
(b)
(c)
(d)
(e)

Figure 1.1: *Recent movies that combine CG animation (or live action) with hand-drawn animation to benefit from both media: Paperman, The Walt Disney Company (2012) (a), The Red Turtle, Studio Ghibli (2017) (b), Spider-Man: Into the Spider-Verse (2018) (c), Loving Vincent (2018) (d), Annie, Riot Games (2018) (e). (All images come from respective movie trailers or behind-the-scenes documentaries.)*

human motion onto virtual characters notably enlarges possibilities and has a positive effect on the production budget.

On the downside, with the advent of computer-assisted animation production, some of the essential properties of its traditional counterpart started to be suppressed. For example, a physical media like pencil, chalk or watercolor contain specific kind of texture variations which are difficult to simulate and which give the observer an impression that the artwork was created by hand. Their digital counterparts typically lack this richness which invoked a kind of sterile computer-generated appearance. Furthermore, physical animation workflow allowed artists to express their creative intents with notably greater freedom as compared to the computerized approach which requires artists to deal with various technological limitations such as predefined animation rigs.

Due to this reason, production of CG animation is still usually preceded with a hand-drawn concept or rough hand-drawn animation. The reason lies in the artist's capability to capture the moment of inspiration quickly on the paper. The hand-drawn artwork is then transformed into 3D models which usually have to obey certain geometrical and physical constraints to allow for their easier manipulation. The process is, however, prone to causing the resulting models not to precisely conform to the style of the original author which may make their appearance too artificial. There are no such constraints in traditional animation. Visual representation of a character may vary considerably across frames, various deformations are often used to exaggerate character's action [Thomas and Johnston 1981] (see Figure 1.2), and even temporal flickering can be used as a tool of expression [Noris et al. 2011]. Nevertheless, the fact that it requires an enormous amount of time even for an experienced artist to produce a short animation is one of the main reasons for today's preference for CG animation in the industry. Features present in hand-drawn animation are often mimicked in 3D CG animation [Lasseter 1987]. However, it involves laborious manual tweaking of a 3D model for each animation frame.

Maybe right due to the aforementioned reasons, the growing trend of CG animation seems to slow down a bit, and animation studios start to getting back to experiments with traditional techniques. Several recent movies which combine computer animation with traditional techniques (see Figure 1.1) have been recently critically acclaimed. All this indicate a tendency of bringing the traditional techniques back in the game.

1.2 Aim of the Work

The aim of this work is to contribute to bridging the world of hand-drawn and CG animation by developing *tools* that enable to facilitate the creation of hand-drawn animations by utilizing contemporary CG animation pipelines and a *small amount of hand-drawn input*. These tools will be specifically designed to faithfully retain artist's individual style and the characteristics of the utilized physical media (in case such media has been used as input). Moreover, the tools should allow the artist to create with freedom of the hand-drawn workflow while at the same time significantly reducing time demands on the animation production by utilizing computer power.

The resulting animations created using these tools may resemble an entirely hand-drawn content, or they can combine 2D and 3D features as presented in the approach by Sýkora et al. [2014] and they thus may be more appealing to the contemporary audience (see Figure 1.3).

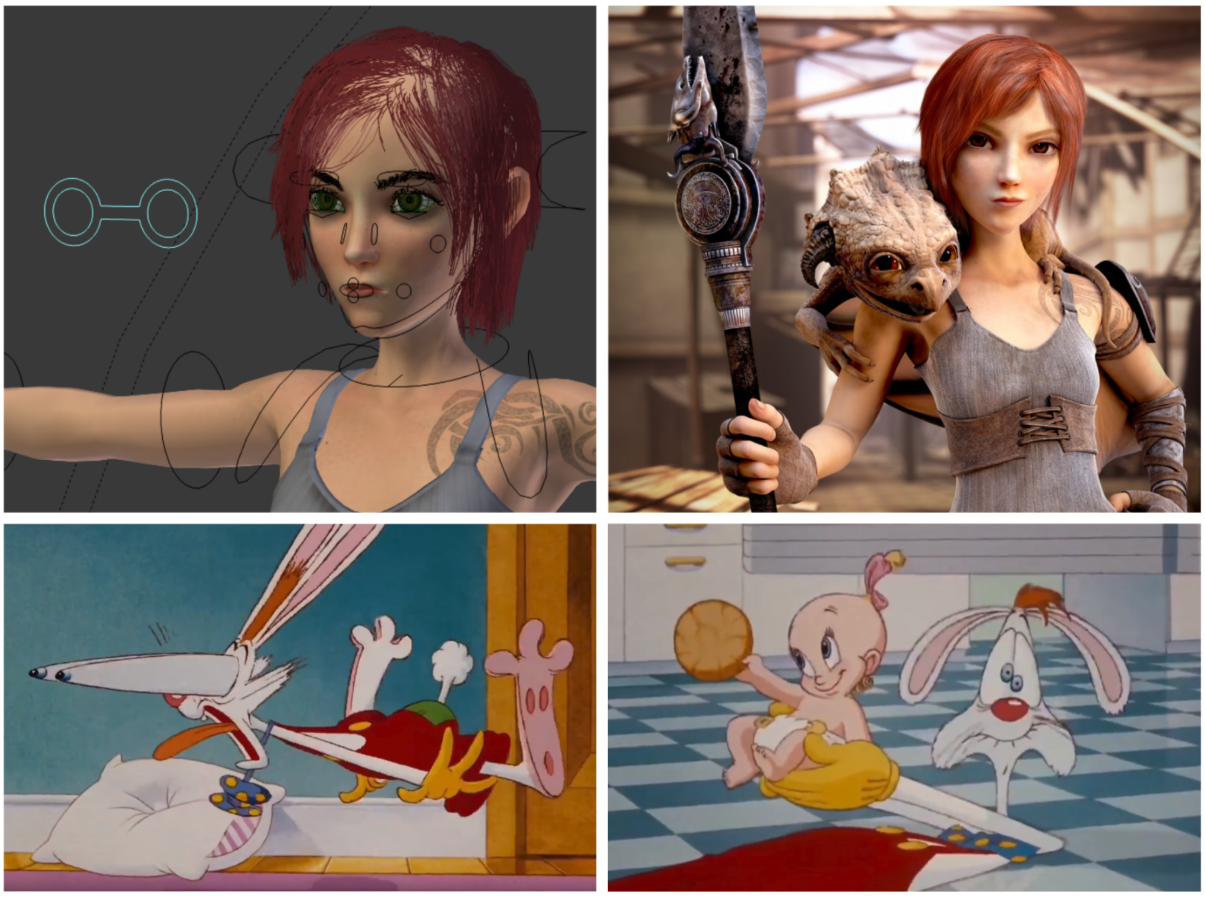


Figure 1.2: *Computer-generated animation utilizes 3D models which usually have to obey certain geometrical and physical constraints to allow for their easier manipulation. On the one hand, this allows the resulting animation to easily include rich visual effects which may be more appealing to the audience (top), on the other hand, it poses a restriction on artistic expressivity. Traditional hand-drawn animation allows artists to create more freely. In such animation, the visual representation of a character may vary across frames to a large extent, and various deformations are often used to exaggerate character's action (bottom). These effects add legibility to the resulting animation, however, the production of hand-drawn animation is extremely demanding. (The top images come from Sintel movie by Blender Institute, the bottom ones from Who Framed Roger Rabbit movie trailer.)*

To address the requirements on the tools mentioned above, we developed new techniques that enable to (1) extract editable structures from the hand-drawn input, and (2) manipulate and reuse the structures to generate a new content that retains the style of the input.

1.3 Overview of the Contributions

To extract the aforementioned structures, we developed several novel approaches: (1) a method enabling to decompose time-lapse videos of painting process into translucent layers together with a technique for cleaning up the videos from unwanted objects such as painter's hand, and (2) techniques for extracting appearance and motion characteristics from hand-drawn animations. We demonstrate the utility of the extracted structures

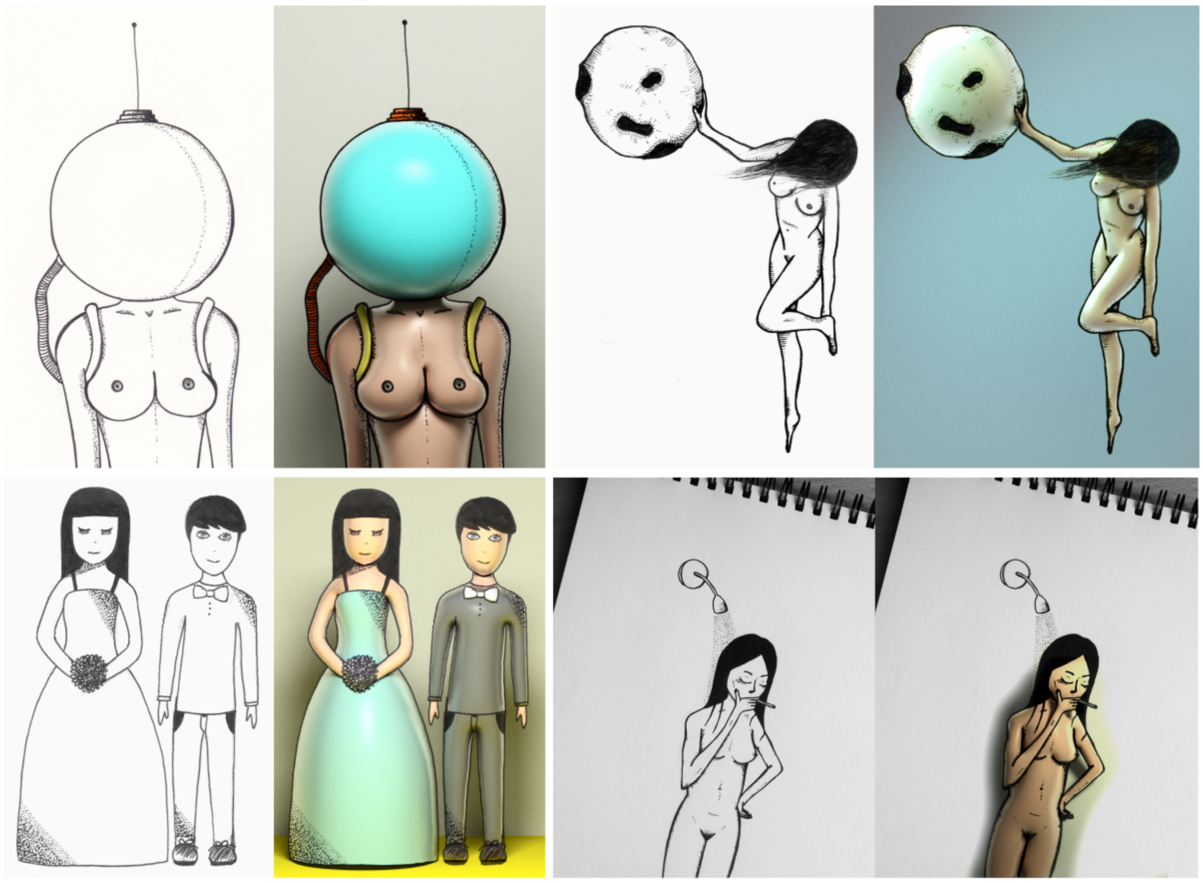


Figure 1.3: *Ink-and-ray [Sýkora et al. 2014] allows to reconstruct a proxy 3D model from a hand-drawn image. The model can be rendered with a texture to enrich the hand-drawn artwork with attractive global illumination effects like shadows and glossy reflections. Hand-drawn inputs by Štěpánka Sýkorová (bottom left) and Lada Brůnová (the rest).*

for various applications including non-trivial editing of real paintings on the computer retaining look of the physical media and also the synthesis of novel hand-drawn animations. To enhance hand-drawn artwork with appealing visual effects that are typical for photorealistic CG artwork, we invented a method for reconstruction of high-relief models from a single hand-drawn image that enables the production of seamless organic models which can be applied in the contemporary CG pipeline.

In the following paragraphs, we overview the techniques and the specific contributions of this work. We group the techniques into the three following categories: (1) decomposition of time-lapse paintings into layers, (2) seamless reconstruction of high-relief models from hand-drawn images, and (3) example-based synthesis of hand-drawn animations.

Decomposition of time-lapse paintings into layers

We start with examining the process of painted artwork creation, for instance, the process of painting a picture or a *single* animation frame. The process itself contains valuable information that is typically lost in the final artwork such as the information about the placement of individual strokes and their overlap. To retain the information, we invented several novel algorithms that are capable of extracting a creation history from digital and real-world time-lapse recordings of the painting process. We capture the creation history

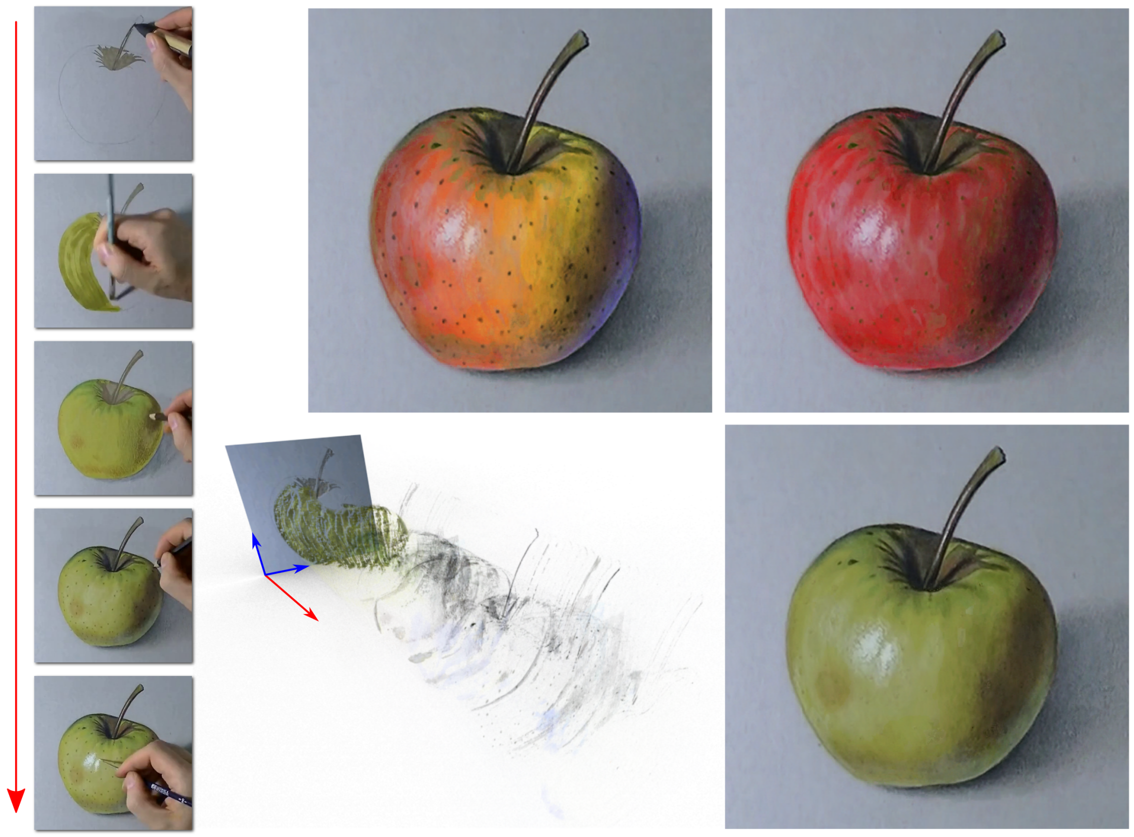


Figure 1.4: *The process of painted artwork creation contains information about the spatiotemporal placement of strokes and their overlap. Such information is typically lost in the final artwork. (A few frames from a recording of painting process are selected on the left.) Our method is capable of extracting a spatiotemporal volume retaining the painting history. (The volume is visualized in the middle column at the bottom, the spatial and temporal domain is depicted using blue and red arrows, respectively.) We also present a method for removing unwanted artifacts from the recording such as occlusions caused by the artist's hand and its shadow (visible on the left). The volume enables artists to perform otherwise arduous edits, such as recoloring or dots removal (right), efficiently. (Input time lapse video by Marcello Barenghi.)*

in the form of a spatiotemporal volume, i.e., paint layers in time – see Figure 1.4 for an illustration of the volume. The algorithms have been designed to reconstruct layers that are useful for later editing. Therefore, instead of seeking opaque solutions, we look for solutions yielding *translucent layers* which increase the number of possible editing operations.

The extracted spatiotemporal volume enables numerous exciting applications such as scribble-based spatiotemporal selection, layers recoloring, color gradient controlled by time or a temporal eraser (see Figure 1.4 for examples). There is a body of previous work dealing with editing history interaction for digital content for which our method may provide a new source of data for physical paintings. (See Section 2.1 for more details about such previous work.) In the context of animation production, the spatiotemporal volume may be utilized to extract a structure of animated objects, for instance by performing a semi-automatic segmentation and depth ordering prediction of object-parts, and the volume may also contain information about the shape or texture of hidden over-

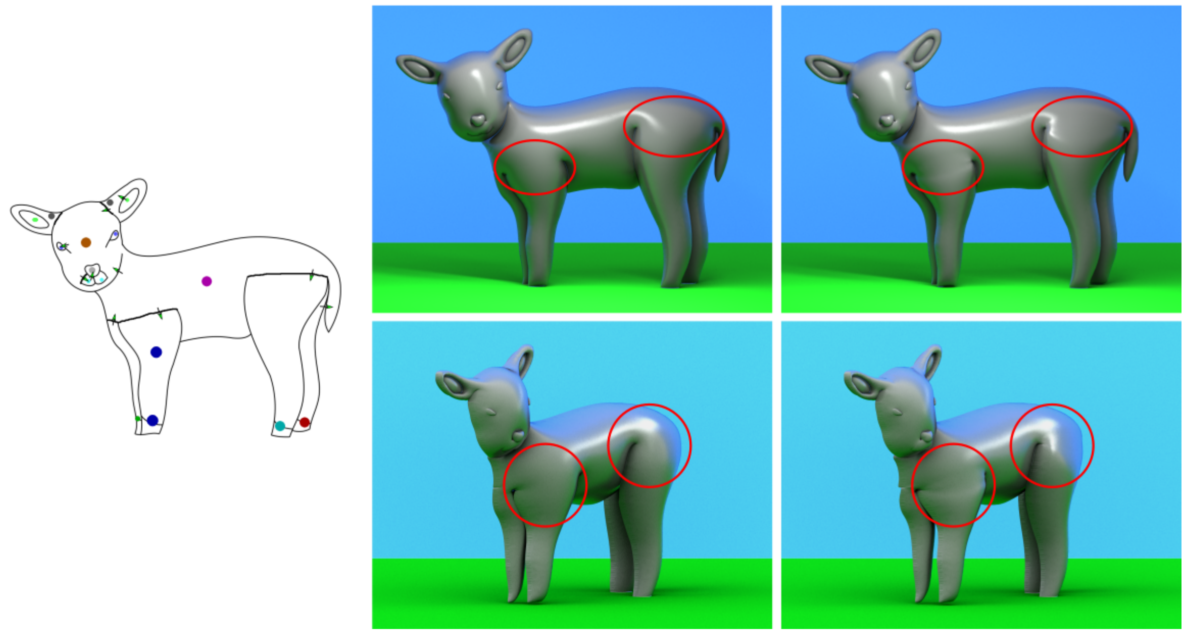


Figure 1.5: *From a single drawing with a small amount of additional input (left), our technique can reconstruct a high-relief model with seamless interconnections of individual parts (middle column). Compared to our approach, the state-of-the-art method of Sýkora et al. [2014] produces results with unwanted seams (right column).*

lapping object-parts. The extracted structures thus may also find their applications in the 3D reconstruction and the hand-drawn animation synthesis which we discuss later in this section.

The decomposition algorithms mentioned earlier require the time-lapse recordings to be free of the typical real-world artifacts such as occlusions caused by the artist's hand and its shadow or lighting and color changes. (See Figure 1.4(left) for an illustration of these artifacts.) To address this, we devised a pipeline for processing the recordings capable of suppressing the unwanted objects. This pipeline may also enable the artist to use physical tools like brush and canvas as an artistic input device (like a natural tablet [Xu et al. 2006]). The above-mentioned methods are thoroughly described in Chapter 3.

Seamless reconstruction of high-relief models from hand-drawn images

Our next contribution is a method capable of reconstructing a high-relief model with seamlessly interconnected parts from a *single* hand-drawn image and a small amount of additional user input (see Figure 1.5). Working in the 2D domain helps the artist to stay focused on the creative process and not be distracted by 3D modeling aspects such as manipulation with geometric primitives in 3D space and working with multiple 2D projections, which is frequent in 3D modeling software like Blender or Autodesk Maya. Even though we do not aim at reconstructing a full 3D model but an approximation, thanks to the bas-relief ambiguity [Belhumeur et al. 1999], such results are sufficient to be rendered with global illumination effects, self shadowing, glossy reflections, and color bleeding, as was demonstrated by Sýkora et al. [2014]. Moreover, thanks to this simplification, our technique takes only a single image as input and does not require side-view information.

Our method is tailored to an interactive modeling scenario where the input drawing can be separated into a set of semantically meaningful parts of which relative depth order is known beforehand. For this kind of input, our technique allows inflating individual components to have a semi-elliptical profile, positioning them to satisfy prescribed depth order, and providing their smooth and seamless interconnection. Opposed to the previous work on this topic which separates the reconstruction process into several subsequent subproblems – such as inflation, shifting and smoothing of object-parts – we propose a novel *unified* non-linear energy functional of which minimization solves for both inflation and shifting simultaneously. This process delivers seamless organic shapes that seem like sculpted from a single block of material. Because direct optimization of the functional is computationally challenging, we propose an approximate solution which yields comparable results orders of magnitude faster and thus enables an interactive user workflow.

The reconstruction method may be applied independently to every frame of a hand-drawn animation, and the resulting models may then be put into the contemporary CG animation pipeline to produce a 2D animation retaining the style given by hand-drawn outlines and textures while enriched with attractive global illumination effects or also stereoscopic features. The method is described in detail in Chapter 4.

Example-based synthesis of hand-drawn animations

The largest part of this thesis is dedicated to methods for example-based synthesis of hand-drawn animations that we invented. We present end-to-end solutions for simplifying the production of hand-drawn animations of colliding stylized 2D rigid body objects and also moving hand-colored characters. Our techniques are based on *animation analogy* concept which extends the established analogy approach of Hertzmann et al. [2001] for style transfer between images to animations. The animation analogy utilizes a guiding *source* animation which prescribes motion (imagine a walk cycle of a skeletal character) and a corresponding stylized hand-drawn *exemplar* animation (imagine a fully hand-colored walking character with exaggerated motion). For a more complex guiding *target* animation, such as jumping or dancing skeletal animation, the aim is to automatically synthesize an *output* animation that follows the target motion while preserving the specific visual appearance and stylized motion of the hand-drawn exemplar animation. (See Figure 1.6 for an illustration of the principle for character animation and Figure 1.7 for 2D rigid body animation.)

The core of our techniques is an analysis and synthesis phase. The analysis phase extracts appearance and motion stylization characteristics contained in the exemplar and source animation, and the subsequent synthesis phase transfers the extracted properties to the target animation. The analysis phase assumes the knowledge of one-to-one correspondence between frames of the source and exemplar animation and captures the stylization characteristics by using a template which may either be a special unstylized frame in a rest pose or a suitable frame taken from the exemplar animation. The template is registered to the corresponding object in each frame of the exemplar animation, which enables extraction of deformations, and the known bijection then allows for capturing various kinds of differences between the source and exemplar animation. Such extracted

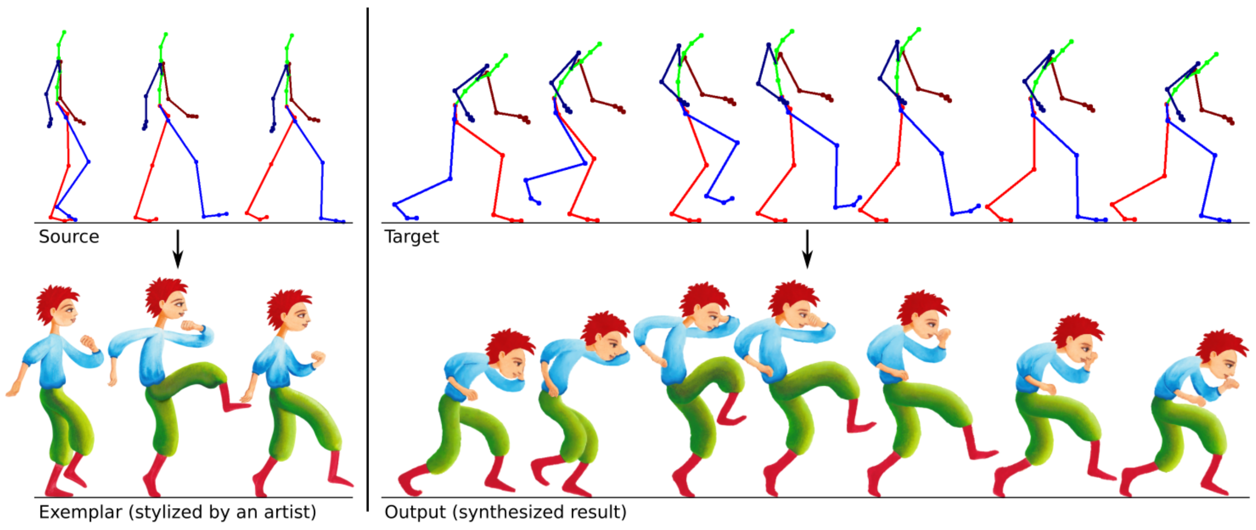


Figure 1.6: *An example of analogy principle for character animations: The analogy between a source skeletal animation (top left) and a corresponding fully stylized hand-colored exemplar animation supplied by an artist (bottom left) is applied to a different target skeletal animation (top right) to produce an output hand-colored animation (bottom right). Notice how the stylized version of the character is different from the skeletal one in terms of both hand-colored appearance and exaggerated motion. (Stylized exemplar animation by Zuzana Studená.)*

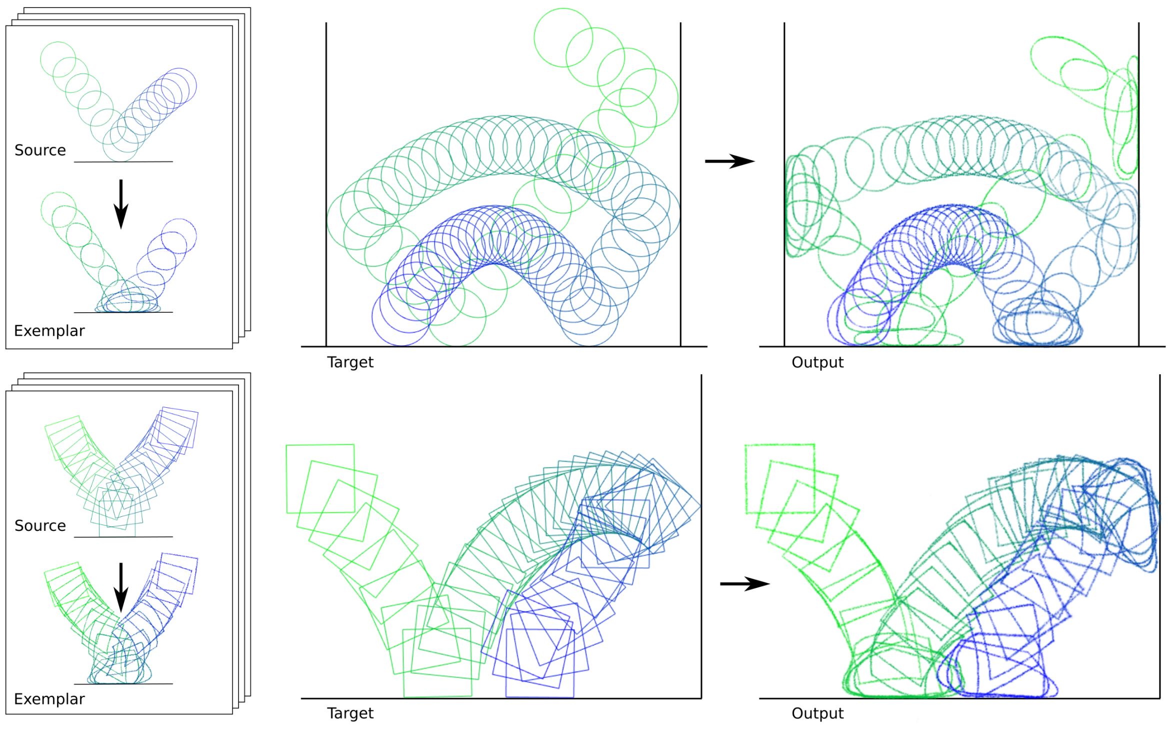


Figure 1.7: *An example of analogy principle for 2D rigid body animations: Pairs of source and exemplar sequences obtained using a physical simulation system and drawn by an artist, respectively (left column), are utilized to transfer the artist-defined style to a different and more complex computer-generated target sequence (middle column) producing the stylized output animation (right column). (Stylized exemplar animations by Zuzana Studená.)*

information is then utilized in the synthesis phase. To make reusing of the exemplar animation feasible for the creation of a different target animation, we partition all animations into shorter ones which we refer to as sub-sequences. For each target sub-sequence, the synthesis phase then searches for the best matching source sub-sequence and utilizes the corresponding differences and deformations to perform a *parametric synthesis*. The result of the parametric synthesis typically contains artifacts due to distortion or blending of textures which invoke unwanted CG appearance. We thus alleviate these artifacts by utilizing the parametric synthesis results only as a *guide* for *non-parametric image synthesis*. This approach allows our methods to perform fully automatic synthesis of convincing hand-drawn cartoon animations from a small number of animation exemplars. Compared to previous work, our techniques are the first to preserve both the detailed visual appearance and stylized motion of the original hand-drawn content. Chapter 5 and Chapter 6 describe the techniques for example-based expressive animations of 2D rigid bodies and example-based synthesis of hand-colored cartoon animations, respectively.

1.4 Structure of the Thesis

The rest of the thesis is structured as follows: Chapter 2 reviews previous work related to our research. Chapters 3, 4, 5, 6 describe the individual contributions of our work in depth. These chapters correspond to individual papers that have been published in journals with impact factor or conference proceedings. Chapter 7 summarizes contributions of this work, briefly describes concurrent work and proposes avenues for future work. Appendix A lists author's publications also with a list of citations to date of submission of this thesis, Appendix B specifies author's contribution to the individual papers, and Appendix C contains supplementary material for "Decomposition of Time-Lapse Paintings into Layers" paper.

Chapter 2

Previous Work

This chapter reviews previous work related to our research. The work is grouped into three sections corresponding to the three fields that this thesis contributes to, as introduced in the previous chapter. This chapter is an illustrated extension of the corresponding sections from our papers.

2.1 Decomposition of Time-Lapse Paintings into Layers

During the painting process, artists paint strokes one after another on a 2D canvas. In time, the strokes typically get superimposed and the information about the spatiotemporal placement of strokes and their overlap is lost in the final artwork. From a video recording of the painting process, our method that is described in Chapter 3 enables to retain this information by decomposing the time-lapse painting into translucent layers, capturing the creation history, which can be later utilized to facilitate editing of the artwork. Moreover, our technique also deals with removal of unwanted real-world artifacts such as painter's hand that may be present in the recording.

In this section, we review related techniques for decomposition of images into editable structures and also related image matting methods. There is a body of work on interaction with editing history for which our method could provide a new source of data for physical paintings. We conclude this section with a brief description of these works.

Decomposition into editable structures

Xu et al. [2006] introduced a technique for decomposition of a real painting into brush strokes and utilized them for the creation of animated paintings. Their method assumes that the painting consists of a smaller number of strokes of which each depicts something specific in the real world (such as leaves, stems, etc.) which is the case for Chinese paintings and some other types of artwork. To extract the geometry and appearance of brush strokes, they exploit a library of strokes, made by an expert on Chinese paintings, which are utilized as shape priors to avoid extracting implausible shapes. To extract the appearance of overlapping strokes after separation, they minimize the color variations along the brush direction parallel to the stroke's skeleton. In our work, we utilize

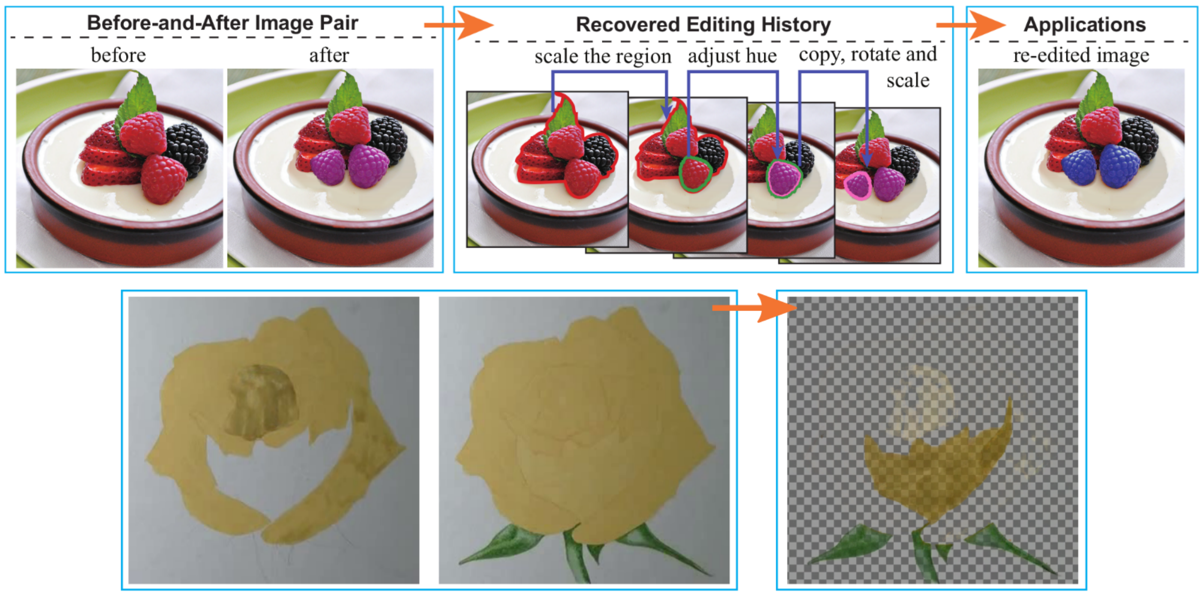


Figure 2.1: *From a pair of images (top left), inverse image editing method [Hu et al. 2013] is able to recover image editing history (top middle). The editing history may be used for various applications such as re-editing (top right) or editing history transfer. Our work solves the orthogonal problem: for each pair of successive frames of a recording of a painting process (bottom left), we recover maximally translucent layers (bottom right).*

time-lapse video to facilitate the segmentation. Moreover, we do not aim at extracting individual strokes but rather translucent paint layers.

A method for decomposing a real painting of a plant into semantically meaningful parts like leaves, petals, and stems was described by Amati and Brostow [2010]. Their work focuses only on Sumi-e, which is a style of monochromatic art with relatively few strokes. Similar to our work, they utilize a video recording of the painting process. They allow using a low-resolution video, which is sufficient to recover the order in which parts were painted, paired with a high-resolution scan of the final artwork that captures stroke details. Their method also handles the removal of real-world objects other than ink such as hand, brush, and shadows. In their technique, a small number of clean frames is found by comparing binary thresholded frames to the binary thresholded final painting. The result of their analysis is a segmentation of the final painting into parts. We take inspiration from this work and share the observation that methods for skin detection and foreground subtraction are unstable and that the paint is far more temporally stable than occlusions.

Fu et al. [2011] presented a method to reconstruct the order of individual strokes and their directions from a single line drawing. Their goal is to derive a drawing order that is visually plausible for a human viewer. To do that, they base the technique on guidelines for drawing order [Van Sommers 1984] and take into account the hierarchical structure of a drawing [Tversky 1999]. Their work is tailored to line art images with cleanly defined lines or curves while our work operates on paintings and utilizes time-lapse videos. Like in their work, we also assume that the drawing order is not random.

An automatic approach to recover semantically-meaningful image editing history from a before-after image pair was proposed by Hu et al. [2013]. (See Figure 2.1 for an illustration.) Their method allows finding editing operations such as geometric transformations,

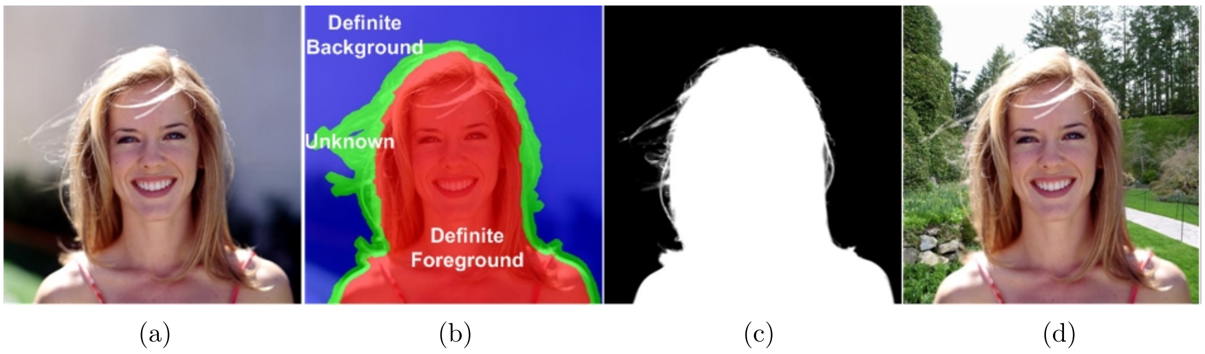


Figure 2.2: *Image matting is a problem of separation of an image (a) into a foreground and background resulting in a matte (c). A user input (b) may be required to make the problem tractable. Image matting allows applications like a substitution for a different background (d). (The illustration is borrowed from [J. Wang and Cohen 2008].)*

color adjustments with spatially-varying brushes, cropping, and object insertion or removal. They, however, do not consider the translucency of edits. In contrast, our method allows for efficient reconstruction of maximally translucent layers between consecutive frames of a time-lapse painting using the traditional Porter-Duff [Porter and Duff 1984] compositing model and also the Kubelka-Munk [Kubelka 1948] model which is best suited for physical paintings.

Richardt et al. [2014] presented an interactive tool for decomposition of an input bitmap image into a set of opaque and semi-transparent vector layers, which can be exploited to make editing of the image easier. Their approach supports vector layers with a linear or radial color gradient. The technique is suited for drawings and studio photographs containing smooth color gradients and may not work well for natural images or images containing complex texture variations.

Image matting

Color separation is also related to layer extraction and image matting. Szeliski et al. [2000] presented a solution to the layer extraction problem in which two independent "images" are layered on top of each other as a result of reflection or transparency. Their approach requires that the layered images are moving with respect to each other, a reasonable assumption for a reflection on a window, but one that does not holds in our scenario. Farid and Adelson [1999] introduced another solution to this problem, but require as input two photographs taken with a polarized light filter. Smith and Blinn [1996] study the related blue-screen matting problem of separating a potentially translucent foreground object from one or two known backgrounds. Their analysis of the problem shows that the problem is, in general, underspecified. Zongker et al. [1999] solve a generalized version of the matting problem which allows for reflections and refractions. A large body of works [J. Wang and Cohen 2008] deals with image matting for natural images that typically contain a background with non-constant colors (see Figure 2.2). Since the problem is ill-posed, additional user input is often required. In our problem, two successive frames are given which makes the problem easier.

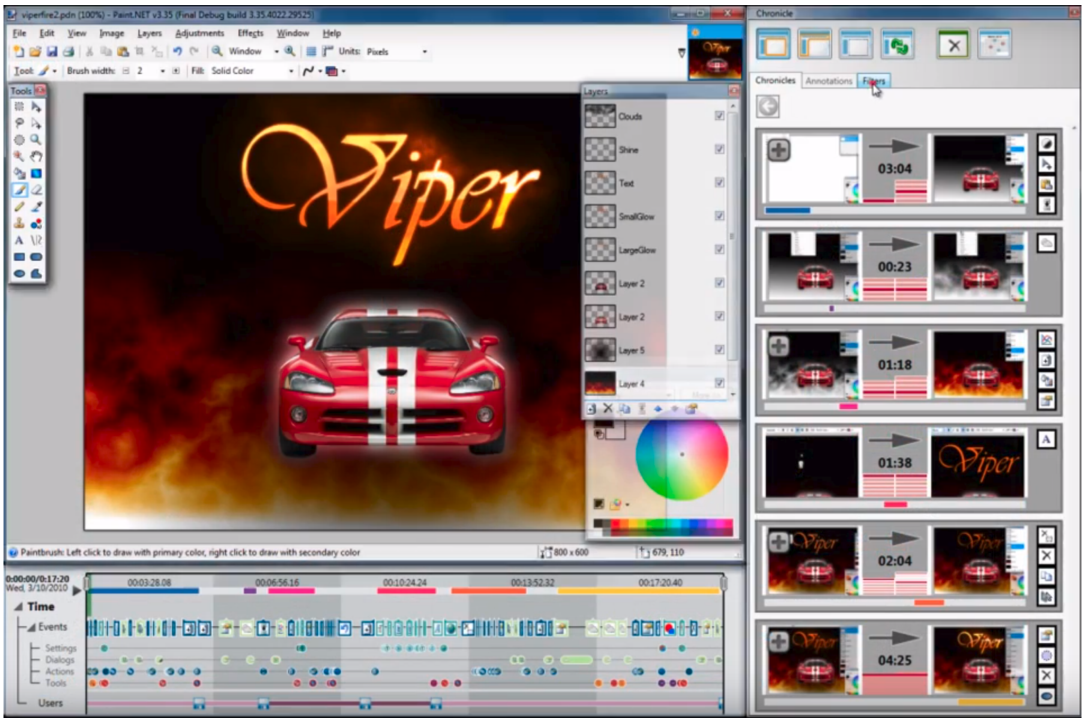


Figure 2.3: *Systems for interaction with editing history like Chronicle [Grossman et al. 2010] allow users to explore document workflow histories. Our method for decomposing time-lapse paintings into layers can provide a new source of data for these systems.*

Applications

Interacting with editing history is a powerful concept in human-computer interaction [Nancel and Cockburn 2014]. This rich literature on history systems extends far beyond undo/redo. For *digital* image editing, this literature includes a generalization of layers for scaling, resizing, and recoloring strokes [Nancel and Cockburn 2014], revision control [H.-T. Chen et al. 2011], grouping command history [H.-T. Chen et al. 2012], learning from or reapplying previous commands [Grossman et al. 2010; Berthouzoz et al. 2011; Xing et al. 2014] (Figure 2.3). Wetpaint [Bonanni et al. 2009] explored a tangible "scraping" interaction to visualize layered information, such as a painting's history. Such powerful interactions are unavailable for physical paintings, even when digitized.

In addition to the history-based interactions for image editing, related interactions have also been proposed for 2D and 3D vector graphics. Su et al. [2009] presented a technique for 2D vector graphics editing which suggests objects to select based on previous selections in the user's command history. Noris et al. [2012] presented a scribble-based approach to segment 2D vector graphics sketches based on time of creation, which helps distinguish nearby strokes drawn at very different times. VisTrails [2009] is a commercial tool for reviewing and reusing command history in the commercial 3D modeling package Maya. Denning and Pellacini [2013] presented algorithms for revision control of 3D meshes. H.-T. Chen et al. [2014] introduced a technique for choosing good views and segmenting 3D models based on the editing history. Two other works analyze and visualize changes in outdoor, urban scenes [Matzen and Snavely 2014] and construction sites [Karsch et al. 2014]. These approaches operate on a collection of photographs from different viewpoints, relying on structure-from-motion to obtain a 3D reconstruction (the former) or a 3D architectural model (the latter). Finally, while not about editing history per se, McCann and Pollard [2009] and McCann and Pollard [2012] introduced two noteworthy generalizations of layers for image editing.

2.2 Seamless Reconstruction of High-Relief Models from Hand-Drawn Images

The traditional way of 3D object modeling using professional 3D modeling software requires skilled users who are comfortable working in the virtual 3D space. Such work is far from natural for 2D artists who prefer to draw. Closer to the 2D workflow are sketch-based modeling systems, such as [Igarashi et al. 1999; Igarashi and Hughes 2003; Tai et al. 2004; Schmidt et al. 2005; Nealen et al. 2007; Borosán et al. 2012], which allow modeling of 3D objects by adding a volume to user-drawn sketches and by manipulation of the created shapes. In such modeling systems, the user typically starts with drawing a closed 2D shape. After finishing the sketch, a volume is added to the 2D shape by utilizing a 3D shape reconstruction method like, e.g., *shape inflation*. The user may then change camera viewpoint, edit the shape using various operations (e.g., by pulling its outline), or draw another shape's outline and attach the newly created shape to the existing one. (See Figure 2.4 for an example of such a modeling system.) To produce 3D models with a more complex topology, the user typically has to view the object from different viewpoints and manually position parts of the object in space. These systems thus still require dealing with the 3D domain which may be perceived as a distraction for the user.

To reduce the distractions, a number of techniques that enable the production of 3D models from a *single drawing*, such as [Gingold et al. 2009; Shtof et al. 2013; T. Chen et al. 2013; Sýkora et al. 2014; Entem et al. 2015; Zeng et al. 2015; Bessmeltsev et al. 2015; Yeh et al. 2017], have been introduced. A more topologically complex 3D object typically consists of several parts which may be positioned in depth and attached or smoothly connected. In a drawing, these parts are usually delineated by outlines. (The 2D projections of these parts are also referred to as *regions*. See Figure 2.5(a,i) for examples of such drawings.) To resolve the absolute depth position of the object-parts, most of these approaches propose an automatic method for doing so, usually based on a small number of user-provided cues.

Aside from the approach of Sýkora et al. [2014] and Entem et al. [2015], the techniques produce models that have object-parts attached, and they do not make any effort to smoothen the interconnections. The method of Sýkora et al. [2014] and Entem et al. [2015] consider smooth interconnections, however, the resulting models still have smoothness-related issues.

In previous work, the reconstruction process is typically separated into a set of individual sub-problems which are solved sequentially:

- First, the input regions are *inflated* to obtain volume.
- Then they are *shifted* in depth to preserve depth ordering.
- Finally, already inflated and shifted components are *stitched together or smoothly interconnected.*

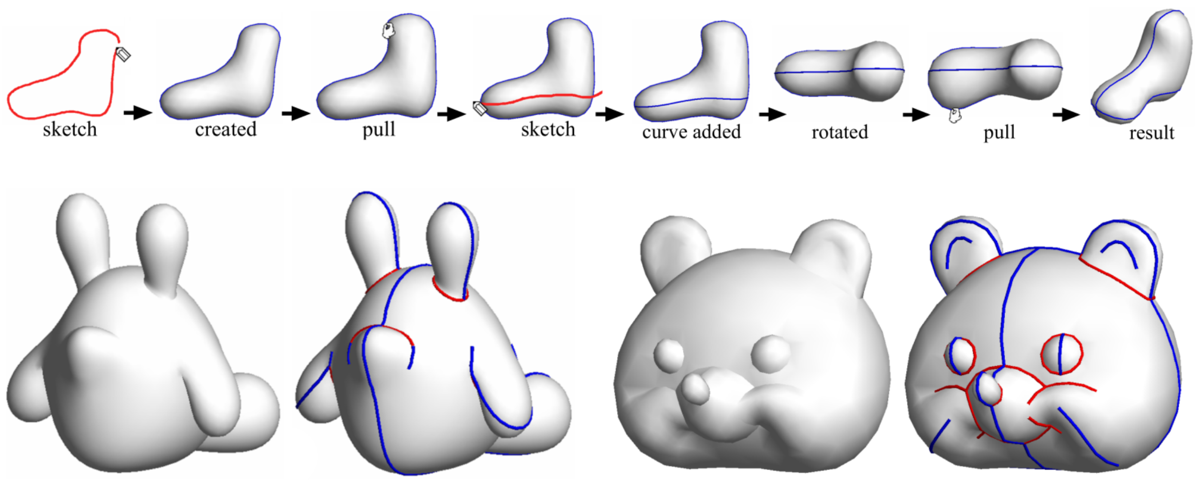


Figure 2.4: *An example of a sketch-based modeling system: By performing a sequence of operations that consists of drawing shapes, changing camera viewpoints and shapes adjustion (an example at the top), it is possible to model 3D objects with a certain level of complexity. The 3D models are obtained by inflating initially flat surfaces and constraining them by the shapes. This figure illustrates the FiberMesh system [Nealen et al. 2007].*

This section reviews advances in solving these sub-problems. We also point out the main issues of the methods that are the most related to our work ([Sýkora et al. 2014], [Entem et al. 2015]) and which we address in our work. The pipeline of these methods is visualized in Figure 2.5 and Figure 2.6, respectively.

Inflation of regions

The idea of adding volume into a 2D drawing by inflation was introduced by Williams [1991]. In the sketch-based modeling system by Igarashi et al. [1999], a 2D shape is inflated by using the distance from a chordal axis of the input 2D region to displace vertices of its triangulated interior. This technique produces surfaces that are coarse and suffer from a low level of smoothness. Igarashi and Hughes [2003] later improved the output model quality by beautification of the coarse mesh using implicit surfaces. Tai et al. [2004] utilized convolution surfaces for inflation. Their method computes a convolution of a linearly weighted kernel along a medial axis of the 2D shape and sums the obtained fields. A procedure of sweeping 2D template scalar field followed by bounding of the resulting 3D field is utilized in [Schmidt et al. 2005] while Nealen et al. [2007] use non-linear optimization to generate a fair surface interpolating the 2D shape. In [Gingold et al. 2009], primitives such as generalized cylinders or ellipsoids are fitted into the user sketches. Shtof et al. [2013] and T. Chen et al. [2013] also utilize such primitives along with performing computer-assisted fitting to outlines of the reconstructed object-part to simplify the placement of the primitive. Borosán et al. [2012] approximate the input sketches using a set of smoothly joined generalized cylinders. Those are also utilized in [Zeng et al. 2015] followed by the application of a simple Laplacian filtering to make the reconstructed surface smoother. Sýkora et al. [2014] perform a two-step inflation procedure to obtain smooth rounded shapes precisely copying the outlines. They first solve a Poisson equation to obtain a shape with a parabolic cross-section and then by taking the square root of the height of the parabolically inflated surface, they obtain a surface with a semi-elliptical profile – this process is visualized in Figure 2.5(c,d). Yeh et al. [2017] follow a similar region inflation approach. They first diffuse user-specified curvature cues (Laplacian values) over the 2D region and then reconstruct the surface by solving a Poisson equation. They note, however, that such inflation procedure results in shapes with parabolic profiles. To obtain fully 3D shapes, Feng et al. [2016] propose a simple extension to the inflation method of Sýkora et al. [2014] that utilizes mirroring. In [Entem et al. 2015], an implicit surface defined by a skeleton of the 2D shape and its radius function is used to reconstruct the volume while Bessmeltsev et al. [2015] represent the 3D shape of 2D regions by a union of generalized surfaces of revolution.

Shifting of inflated regions

The approaches for finding absolute depth position of parts may be categorized as manual and automatic.

The manual approach requires the user to sketch in a different projection or position the parts or some auxiliary structure, such as the skeleton, in 3D space manually. Methods using such an approach include [Igarashi et al. 1999; Igarashi and Hughes 2003; Tai et al. 2004; Schmidt et al. 2005; Nealen et al. 2007; Borosán et al. 2012; Bessmeltsev et al. 2015; Feng et al. 2016].

The automatic approaches first infer the missing relative depth information using various cues and then perform the positioning in an automatic fashion using this information. Gingold et al. [2009] utilize user-supplied connection curve annotations to establish the relative depth between two parts (with optional specification of an intersection curve). Shtof et al. [2013] and T. Chen et al. [2013] make use of semantic geometric constraints such as parallelism, orthogonality, collinear centers, concentricity, and coplanarity to determine the relative depth of primitives in 3D space. To find the absolute depth positions of object-parts, Shtof et al. [2013] formulate a non-linear problem utilizing the geosemantic constraints, which they optimize using L-BFGS solver with automatic differentiation. T. Chen et al. [2013] improve on the speed of the process by decomposing the non-linear problem into two steps – first finding initial depths of all parts at once to satisfy the geometric constraints and then proceeding with full optimization allowing change of shapes of the parts. In [Sýkora et al. 2014], illusory surfaces [Geiger et al. 1998] are employed to predict the relative depth ordering between object-parts with optional correction of prediction error using user-supplied inequalities [Sýkora et al. 2010]. Then they solve a quadratic program to find as-constant-as-possible shifting surfaces that deform object-parts in depth while meeting the inferred relative depth ordering. Zeng et al. [2015] propose an iterative optimization process to find consistent object positions on a supporting plane. As relative depth information, they utilize T-junctions with ground contact cues. Entem et al. [2015] base their method for finding relative depth ordering of regions on structural symmetry and then automatically position the 3D reconstructed regions in depth by considering their thickness. The process may, however, cause some parts to interpenetrate. Yeh et al. [2017] allow deforming regions in depth with the use of user-specified slope cues. To shift the inflated and deformed regions in depth so that the parts do not interpenetrate, they formulate an energy function which aims to minimize additional deformations of the object-parts and also the gaps between them and meet the prescribed relative depths. To estimate the relative depth ordering, they employ T-junctions and overlap area hypotheses [Liu et al. 2013; Yeh et al. 2015].

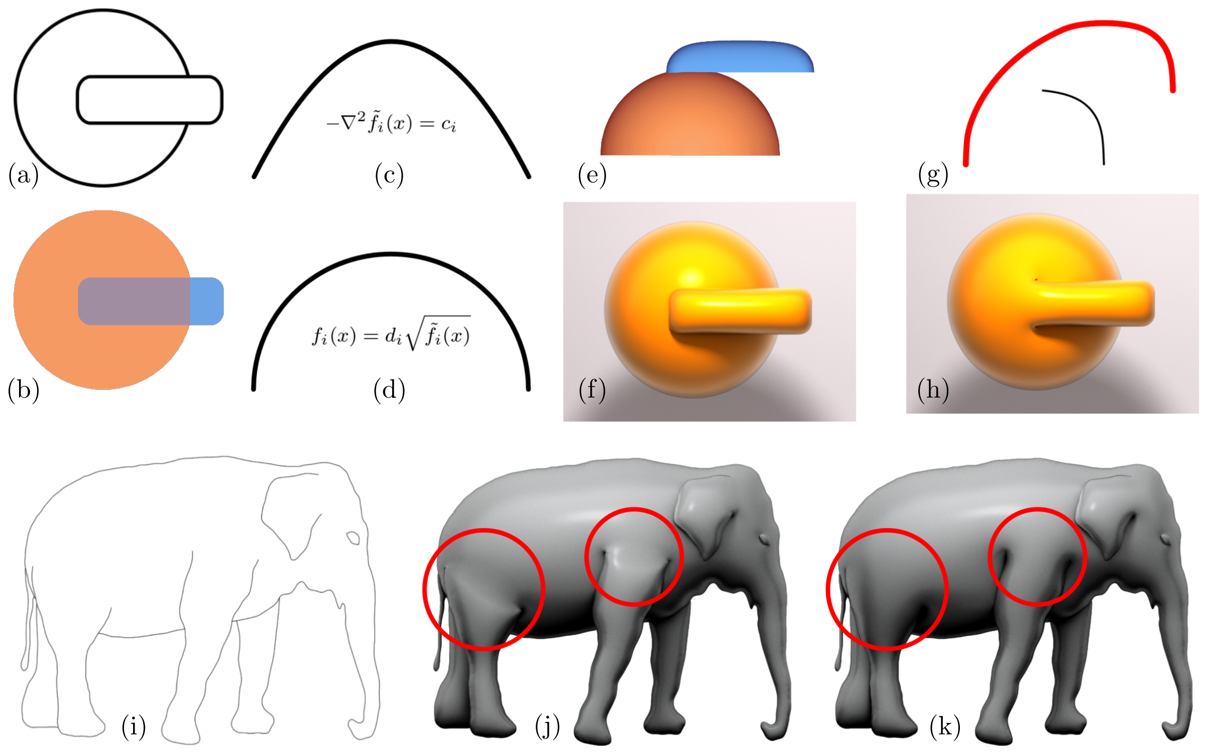

(a)
(c)
$-\nabla^2 \tilde{f}_i(x) = c_i$
(e)
(g)
(b)
$f_i(x) = d_i\sqrt{\tilde{f}_i(x)}$
(d)
(f)
(h)
(i)
(j)
(k)

Figure 2.5: *A brief description of the pipeline of Ink-and-ray method [Sýkora et al. 2014]: A single input image (a) with determined regions and their relative depth ordering (b) is 3D reconstructed by inflating each region separately on a common base plane. The inflation is based on first solving a Poisson equation (c) yielding parabolic profile and then taking the square root of the height to obtain semi-elliptical profile (d). The absolute depth position of regions (e) is found by solving a quadratic program taking into account the relative depth ordering. The resulting model (f) has individual inflated regions attached but not smoothly interconnected. To produce smooth joints, they perform an additional smoothing step by using biharmonic interpolation (g) yielding the final model (h). For a different input drawing (i), the method may produce models that have visible seams at connections of regions (j) while the desired result should rather have seamless interconnections of parts (k).*

Smooth merging of inflated and shifted regions

The works of Gingold et al. [2009], Shtof et al. [2013], T. Chen et al. [2013], Zeng et al. [2015], Bessmeltsev et al. [2015], Feng et al. [2016], and Yeh et al. [2017] do not support smoothly interconnected object-parts.

Igarashi et al. [1999] smoothen the part of the mesh in an area marked by the user using a heuristic method followed by a surface fairing algorithm [Taubin 1995], and a non-linear geometric fairing algorithm [Schneider and Kobbelt 2001] is employed in their follow-up work [Igarashi and Hughes 2003]. A variant of the non-linear fairing algorithm is utilized in [Nealen et al. 2007] for shape reconstruction from control curves which ensures smooth and seamless interconnections of individual modeled object parts. Borosán et al. [2012] perform the Laplacian smoothing around the intersection loop. In [Sýkora et al. 2014], a smoothing area is first computed based on user-given parameters, and bi-harmonic interpolation is then performed in the area. Such smoothing procedure may, however, still produce joints with unwanted seams (see Figure 2.6(j)). Some works [Tai et al. 2004;

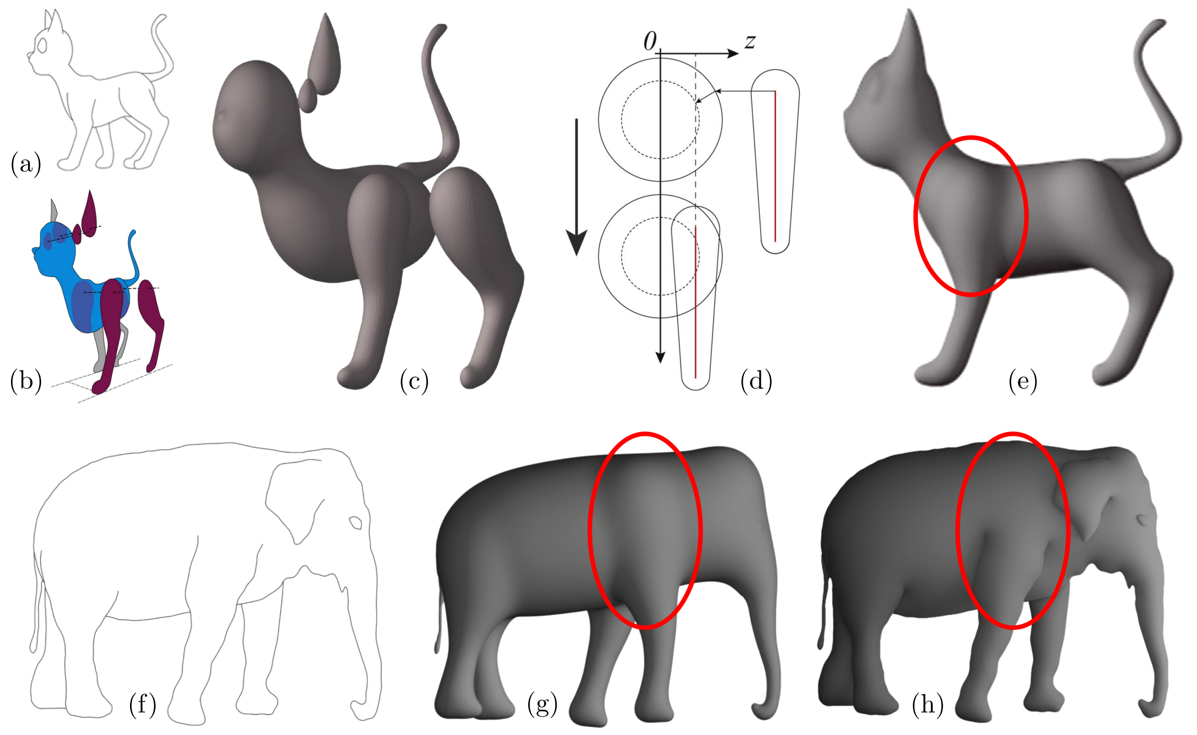


Figure 2.6: *A brief description of the pipeline of a method for 3D reconstruction of animals from a single image [Entem et al. 2015]: From a single input image (a), regions are extracted and their relative depth ordering is found based on structural symmetry (b). Individual regions are 3D reconstructed using skeleton-based convolution surfaces (c), and the absolute depth position of regions is determined based on the thickness of their 3D reconstruction (d). The final 3D model with individual parts smoothly interconnected (e) is obtained by summation. The reconstructed model usually has visible protrusions as depicted inside the red circle. For a different input image (f), this method produces a result with visible bulges (g) while the desired result should rather have truly seamless interconnections of individual parts (h).*

Schmidt et al. 2005; Entem et al. 2015] utilize implicit surfaces which enable smooth blending of parts by using Ricci's operator [Ricci 1973]. This approach allows control over the smoothness of each joint, however, when combined with the shifting technique of Entem et al. [2015], the result usually has visible protrusions in areas where several parts are interconnected. Such behavior may be unwanted as depicted in Figure 2.6(g).

Conclusion

The works mentioned in this section decouple the modeling process into separate subproblems. Because of this sequential process, the quality of the resulting mesh often suffers from the lack of smoothness in the areas where individual parts were stitched together. Our method described in Chapter 4 *unifies* inflation, positioning, and seamless joining of individual components into a single energy minimization framework. From a single hand-drawn image and with minimal user intervention, our approach makes it possible to create organic high-relief models with rounded parts that are automatically correctly positioned in depth and seamlessly interconnected. The resulting models produced using our method resemble objects sculpted from a single piece of material.

2.3 Example-Based Synthesis of Hand-Drawn Animations

Despite today's prevalence of 3D animation, 2D animation remains a popular and engaging medium for storytelling. Production of 2D animation with qualities of traditional hand-drawn animation is, however, extremely demanding process that requires highly experienced artists. Since the pioneering work of Catmull [1978], many follow-up research has been carried out over the last few decades to facilitate the production of traditional hand-drawn animation using computers.

Computer-assisted inbetweening

One of the problems that have received significant attention is computer-assisted inbetweening. It aims to generate intermediate poses between two (or more) extreme key poses (keyframes). The problem is challenging as the extreme poses are projections of a usually 3D object in animator's mind and they thus typically miss some important information, for instance, due to occlusions [Catmull 1978]. Various techniques have been proposed to tackle the problem, achieving impressive results both in the vector [Kort 2002; Baxter and Anjyo 2006; Whited et al. 2010; Yang 2018; Yang et al. 2018] and raster domains [Baxter et al. 2009; Sýkora et al. 2009; Arora et al. 2017] (see Figure 2.7 for an illustration). Some of these methods [Baxter and Anjyo 2006; Baxter et al. 2009; Arora et al. 2017] propose N-way interpolation between multiple keyframes to widen the available pose space. Nevertheless, these techniques are designed to deliver plausible transitions only between keyframes. To produce animation for a new target motion, artists must create additional keyframes by hand.

Simulation of animation principles

One of the long-term goals of computer graphics has been to reproduce the *expressiveness* of traditional hand-drawn 2D animations while reducing the cost and effort of producing it. The fundamental principles of animation developed at Walt Disney Studio play a crucial role in this expressiveness, and many works have tried to adapt them to digital animation tools. The 2D animation principles include squash-and-stretch, timing and spacing, anticipation, follow-through, arc trajectories, and lines of action [Thomas and Johnston 1981]. Lasseter [1987] described how these principles might also be manually applied by an artist to produce expressive 3D keyframe animations. Much subsequent work has been done to automatize the creation of the animation effects using computers. The techniques aiming to do so may be categorized as procedural techniques and techniques that employ physical simulation.

Procedural techniques. To simulate anticipation and follow-through as well as squash-and-stretch deformations, J. Wang et al. [2006] propose a simple temporal filter that enables the production of these effects by delaying parts of an existing animation represented by 2D polygonal shapes or motion captured (MoCap) data. Lee et al. [2012] obtain similar effects on segmented objects in a video by relating a set of representative 2D deformations to the modal analysis of the object motion. Focusing on 3D skeletal animation, Noble and Tang [2006] present a tool to automatically bend the limbs of a

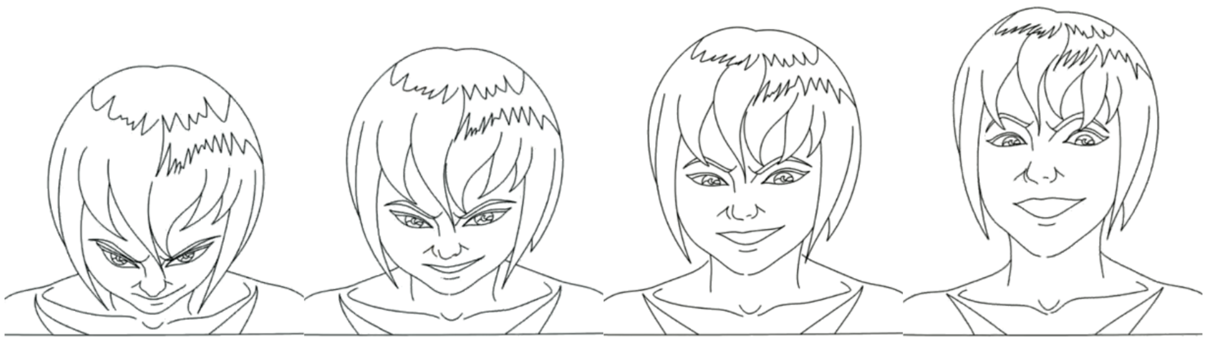


Figure 2.7: *Recent computer-aided inbetweening methods [Yang et al. 2018] allow generating smoothly interpolated animation (frames in the middle) from a set of hand-drawn keyframes (left and right) even for challenging poses with occlusions. However, inbetweening methods are designed to deliver plausible transitions only between keyframes. Based on a short hand-drawn input animation, our animation synthesis methods can generate a longer and more complex novel animation retaining the style of the input.*

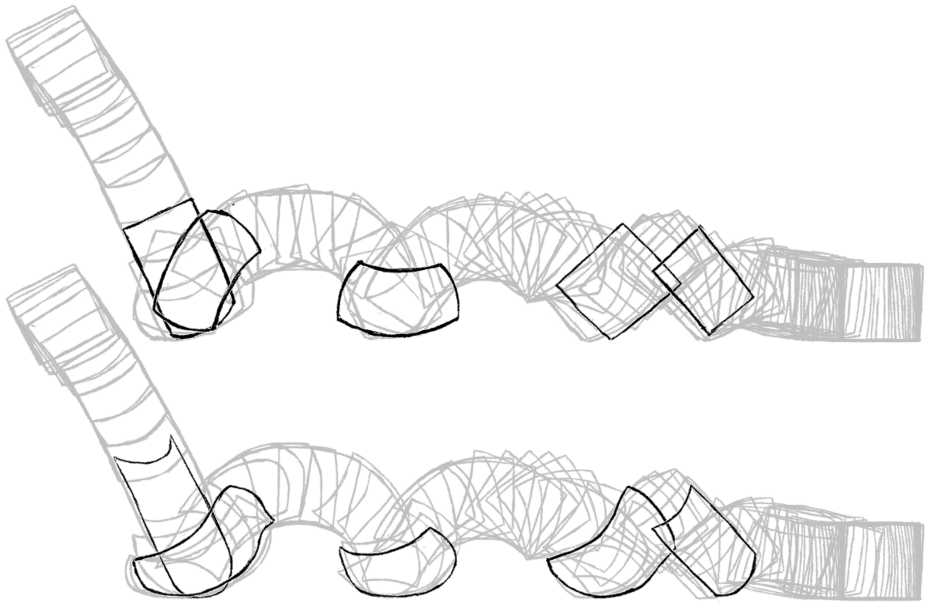
character following lines of actions or arcs. On the artistically controllable side, Y. Li et al. [2003] present a sketching interface that allows a user to guide the deformation of both the input animated skeleton and surface mesh to improve the expressiveness of Mo-Cap data. Recently, several 2D sketching systems [Kazi et al. 2014a; Kazi et al. 2014b; Kazi et al. 2016; Xing et al. 2016] have been developed to simplify the production of dynamic illustrations, reproducing most principles of character or special effects animation. To stylize motion in CG animation, Schmid et al. [2010] proposed programmable motion effects which extend the concept of surface shader and demonstrate their utility for generating motion blur, stroboscopic images, speed lines, and also time shifting which can be used to bend a moving object automatically.

Since artist's style is a highly personalized concept, the stylization of one animation effect may differ considerably between two animators and even a single animator may create many variations. (See Figure 2.8(a) for a short illustration.) The variability of real hand-drawn animation thus cannot be easily parameterized. The works mentioned above employ algorithmic routines to simulate animation effects which inherently limits the stylization variability. Unlike our work, they are not tailored to the specific style given by the artist.

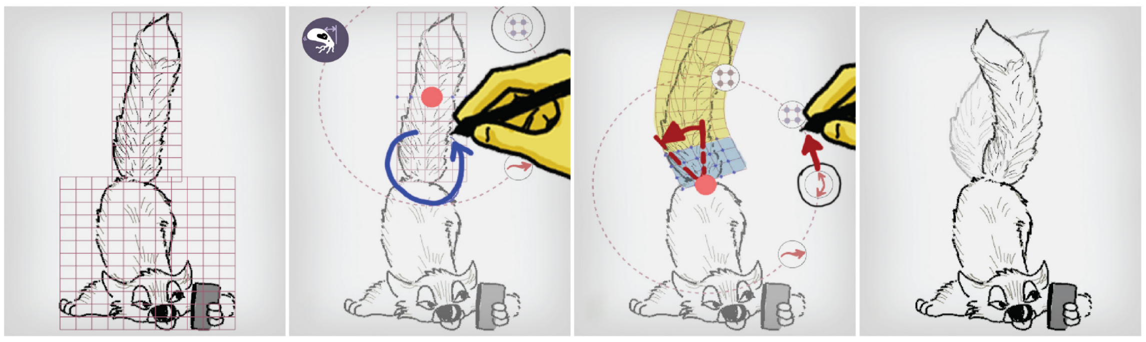
Physics-driven approaches. Physical simulation is a convenient way to animate a large number of 2D or 3D bodies automatically, but the expressiveness of the resulting motion is limited by the degree of complexity modeled by the physical system. For instance, it is common to restrict the simulation to rigid bodies for computational efficiency, while traditional hand-drawn animated objects more often resemble deformable bodies governed by exaggerated physical laws. To enhance basic 3D simulations, multiple works [Chenney et al. 2002; Haller et al. 2004; Garcia et al. 2007] derive automatic procedural rules to generate squash-and-stretch and temporal effects based on motion parameters (velocity, acceleration, etc.) but such methods have limited art-directability.

To allow artistic control, the spacetime [Witkin and Kass 1988] and dynamic [Barzel and Barr 1988] constraint formulations cast physical simulation as a constrained optimization problem. Through those constraints, the artist can direct the simulation to act

(a)



(b)



(c)

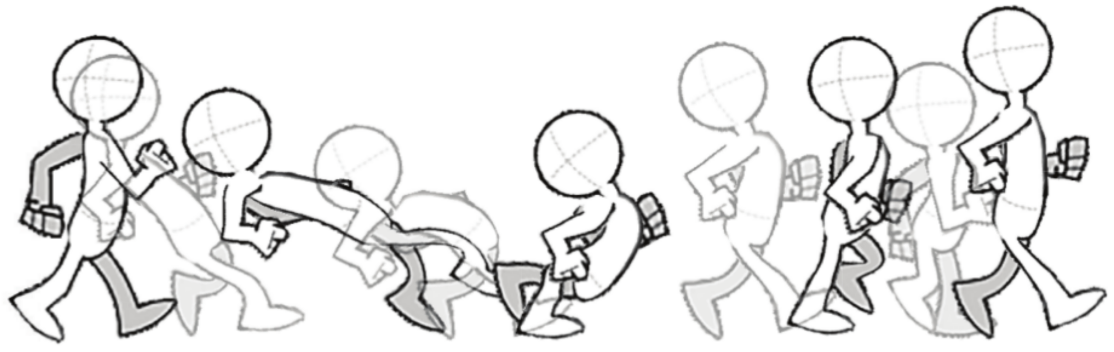


Figure 2.8: *Traditional hand-drawn animation offers almost unlimited stylization variability. For illustration, two different stylizations of squash-and-stretch effect for a simple bouncing square animation were created by a single artist (a). Sketching systems for 2D animation production like Motion Amplifiers [Kazi et al. 2016] (b) enable the creation of 2D dynamic illustrations enriched with various animation effects. The effects are, however, encoded as scripted deformations which inherently limits the variability. 2D animation systems like [Bai et al. 2016] (c) combine keyframing of local deformations with physical simulation and also enable reproducing many of the animation principles. These systems, however, require the user to manipulate higher-level controls instead of simply drawing frames which is unnatural and more constrained. They also do not consider faithful preservation of hand-drawn appearance and produce visual artifacts.*

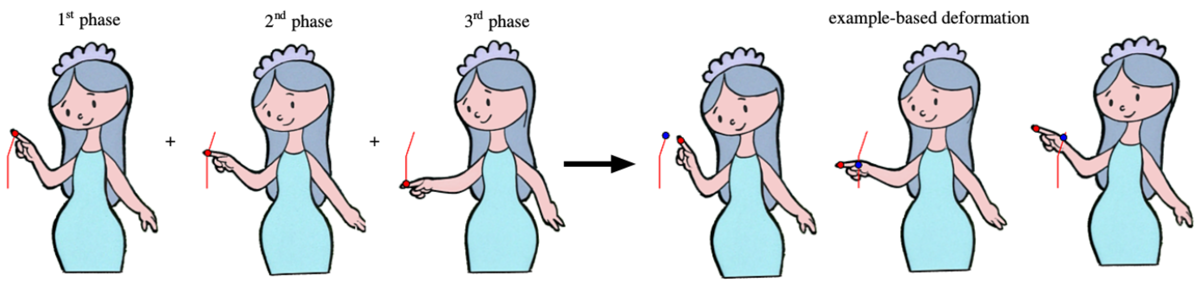


Figure 2.9: *Sýkora et al. [2009] propose an example-based shape deformation technique that enables reusing consecutive frames of original hand-drawn animation (left) for generating novel poses by inbetweening and subsequent deformation (right).*

as a physically-plausible interpolation mechanism between key-poses. Bai et al. [2016] leverage this idea to build a 2D animation system that combines the keyframing of local deformations with physical simulation for powerful inbetweening (Figure 2.8(c)). Although this approach allows an unprecedented level of artistic control and manages to reproduce many of the principles of animation, it requires the user to specify control handles, which are constrained and unnatural to use when compared to simply drawing frames, in particular when the artist desires a precise control over shape outlines.

Physical simulation is also leveraged in [Willett et al. 2017] to simplify the creation of secondary motion effects for layered 2D artwork. Zhu et al. [2017] utilize physics-driven deformation to perform planar interpolation with dynamic secondary motion effects and B. Jones et al. [2015] leverage example-based physical simulation [Koyama et al. 2012] to navigate in a manifold of example poses.

Although the methods mentioned above may achieve the look-and-feel of traditional animation, they might not preserve specific motion details that often characterize a given artist's style. These techniques also do not consider how to faithfully preserve the detailed visual appearance of hand-drawn artwork that is in motion. In most cases, textures are simply stretched and deformed, which causes visual artifacts.

Original content manipulation

To retain more of a hand-drawn appearance, some techniques directly reuse or manipulate existing hand-drawn content. They either use the animation sequences unchanged [Buck et al. 2000; de Juan and Bodenheimer 2004; van Haevre et al. 2005; de Juan and Bodenheimer 2006] and only reorder the animation frames, add more inbetweens, or directly manipulate the appearance on a pixel level [Sýkora et al. 2009; Sýkora et al. 2011; Zhang et al. 2012] to enhance the visual content or change the motion characteristics. (See Figure 2.9 for an illustration). Although these approaches better preserve the notion of hand-colored animation, their potential to make substantial changes to the motion is rather limited. Extensive manual work is typically required when a different animation needs to be produced out of existing footage.

Example-based methods

Example-based techniques provide a natural and intuitive interface, where examples are used to capture the style and intent of an artist. Rather than directly reusing hand-drawn frames, image analogies [Hertzmann et al. 2001] provide a powerful framework for synthesizing new content based on example artwork. In this approach, a guiding image and its stylized version are provided to define the style transfer analogy. The analogy is then used to produce the stylized version of a different guiding image. (See Figure 2.10 for an illustration of the principle). The approach has been improved by utilizing a non-parametric texture optimization technique [Kwatra et al. 2005; Wexler et al. 2007] with uniform usage of source patches to avoid the undesirable wash-out effect [Jamriška et al. 2015]. Fišer et al. [2016] demonstrated that taking object illumination and its stylization into account combined with the encouragement of uniform usage of source patches better preserves the visual richness of hand-created style exemplars. In their follow-up work [Fišer et al. 2017], they proposed specifically designed guidance channels for the synthesis of stylized faces (see Figure 2.11). Gatys et al. [2016] utilized a convolutional neural networks approach to tackle the style transfer problem, however, their method has difficulties to preserve the textural richness and may distort local visual features. The image analogies approach has also been extended to stylize animations [Bénard et al. 2013] with later work adding user control over the amount of temporal flickering [Fišer et al. 2014; Fišer et al. 2017] to preserve better the impression that every animation frame was created by hand independently.

Nevertheless, these analogy-based approaches only support appearance style transfer and do not consider how to represent and apply motion stylizations. There are some exceptions, such as Bregler et al. [2002] who propose to capture and re-target motion from existing cartoon animations by combining a global affine deformation with drawings interpolation using a key-shape model. B. Jones et al. [2015] follows a similar approach, connecting the navigation in a simplicial complex [Ngo et al. 2000] with events of a 2D physical simulation. Pose-space interpolation can produce impressive results, but the quality of the output is highly dependent on a good choice of the key-shapes which an artist has to select and organize manually beforehand.

To guide or enrich 3D simulations, example-based approaches augment the simulated objects with examples of desirable deformations [Martin et al. 2011; Koyama et al. 2012; Coros et al. 2012; B. Jones et al. 2016]. In those approaches, however, exact correspondences between deformation exemplars are known beforehand, and only a simple parametric deformation with a limited number of degrees of freedom is used. Even though this setting may be natural for digital 3D artists, it is again limited and constraining for traditional 2D animators.

Closest to the traditional animation pipeline, Xing et al. [2015] present an interactive system for computer-assisted 2D animation. It combines discrete texture synthesis with an as-rigid-as-possible deformation model to predict and interpolate drawings based on the current and previous frames. This approach is convincing for frame-by-frame animation, but the spatiotemporal locality of the analysis makes it unsuited for pose-to-pose planning. Since the interpolations are solely based on the past drawings using local affine transformations, the predicted motion and deformations tend to be unnatural and cannot easily be edited, unless the artist draws most intermediate frames.

Skeletal animation

Skeletal animation [Burtnyk and Wein 1976] has proven to be an efficient tool for deforming 2D shapes [Hornung et al. 2007; Vanaken et al. 2008]. It has been used to control

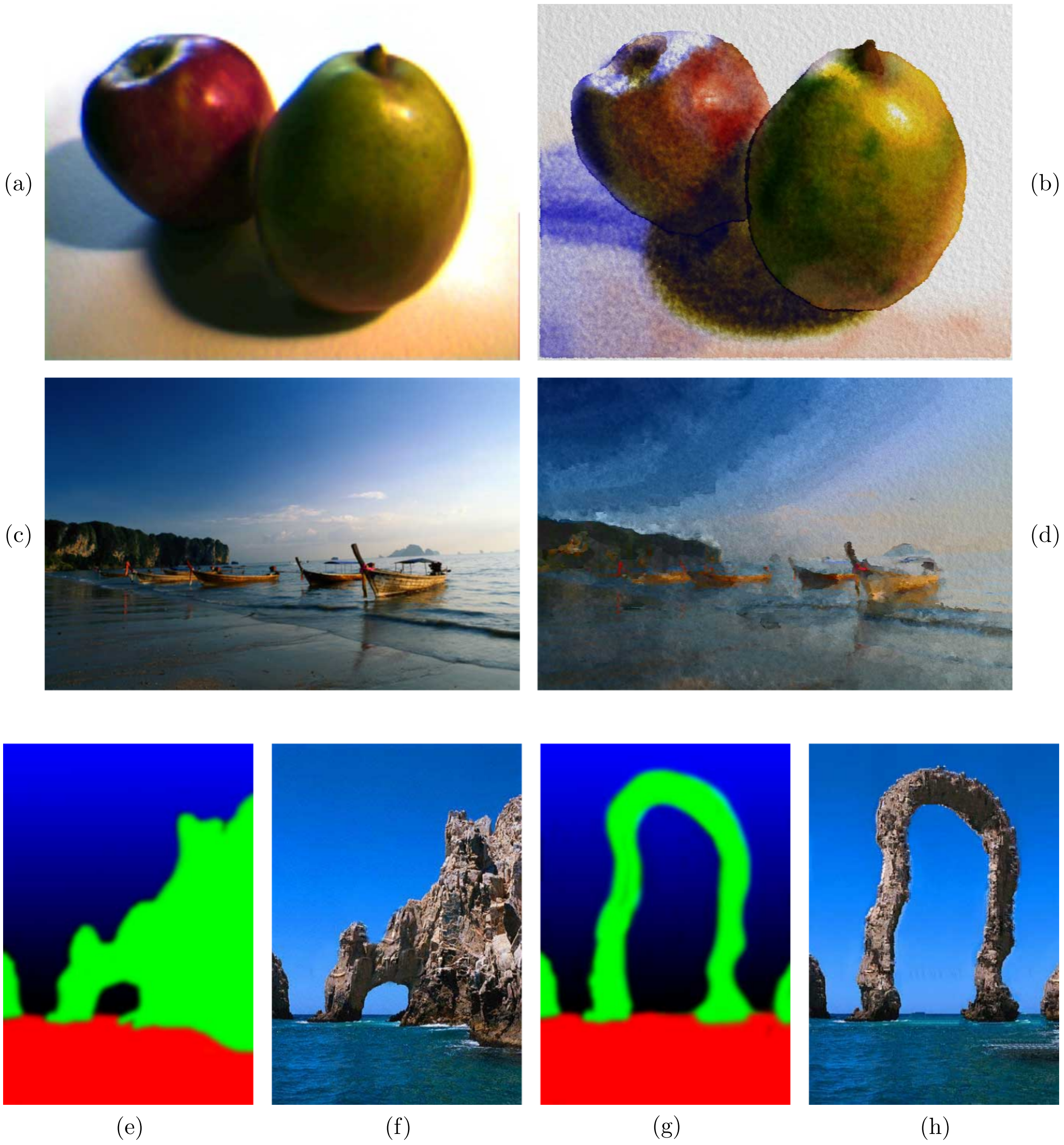


Figure 2.10: *Image Analogies [Hertzmann et al. 2001] make possible to learn a style from a pair of unstylized (a) and stylized (b) images and transfer the learned style to a different image (c) to obtain a stylized version of it (d). In a similar application (Texture by Numbers), an analogy between a pair of a segmented (e) and original (f) image is determined and then applied to a modified segmentation (g) to produce a novel image (h) similar in appearance to the original image.*

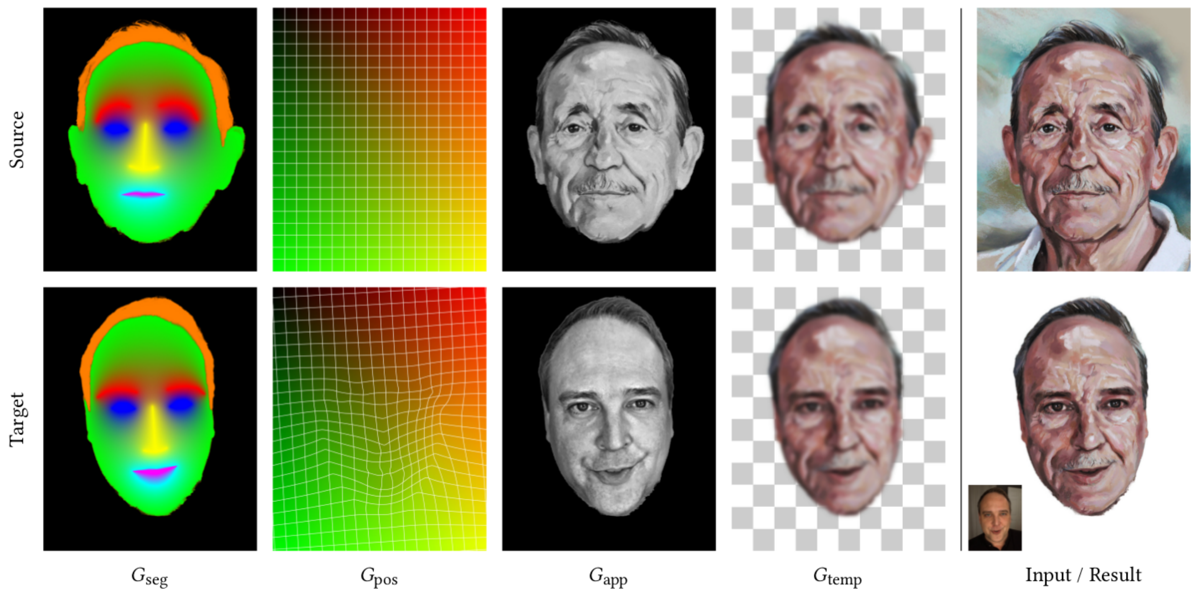


Figure 2.11: *FaceStyle [Fišer et al. 2017] use the concept of Image Analogies [Hertzmann et al. 2001] to transfer the appearance of a source painting of a face (top right) to frames of a target video sequence (bottom right). To achieve this, they use specifically designed guidance channels (all columns except the rightmost). Unlike our work, such analogy-based approaches only support appearance style transfer and do not consider how to represent and apply motion stylizations.*

deformation in the context of cartoon animations [Sýkora et al. 2005; X. Wang et al. 2013] as well as to transfer motion from a sequence of drawings [Bregler et al. 2002; Davis et al. 2003; Jain et al. 2009] or a single pose [Bessmeltsev et al. 2016] onto a 3D model. In our framework described in Chapter 6, we demonstrate that skeletal animation can also be used as an effective guide to performing style transfer between hand-drawn exemplars and target animation.

Conclusion

The above-mentioned 2D animation research mostly focuses on individual subproblems such as inbetweening and motion or appearance transfer. Dealing with the subproblems separately may cause the result missing some important features of the input hand-drawn content. For example, in some cases, shapes are interpolated with the proper motion characteristics, but the result invokes computer-generated appearance due to artifacts caused by distortion or blending of textures [Baxter et al. 2009; Sýkora et al. 2009; Arora et al. 2017]. Or, the appearance is transferred properly, but the underlying motion feels too artificial [Fišer et al. 2014; Fišer et al. 2017]). Our techniques for example-based animation synthesis combine motion and appearance stylization into a unifying framework that reproduces both the appearance and motion characteristics of hand-drawn animations.

Chapter 3

Decomposing Time-Lapse Paintings into Layers

The paper included in this chapter was presented at the SIGGRAPH 2015 conference and has been published in ACM Transactions on Graphics journal:

Tan, J., Dvorožňák, M., Sýkora, D., and Gingold, Y. [2015]. "Decomposing Time-Lapse Paintings into Layers". *ACM Transactions on Graphics* 34.4, p. 61

Decomposing Time-Lapse Paintings into Layers

Jianchao Tan* George Mason University
Marek Dvorožňák CTU in Prague, FEE
Daniel Sýkora CTU in Prague, FEE
Yotam Gingold George Mason University

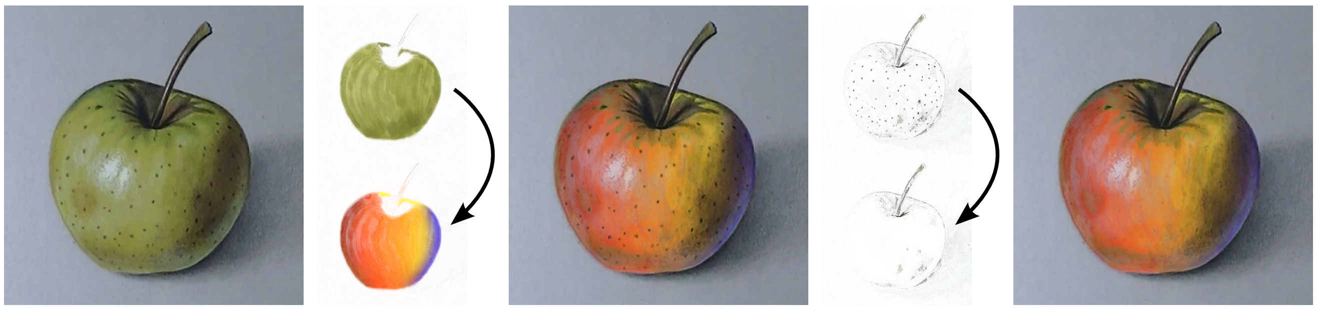
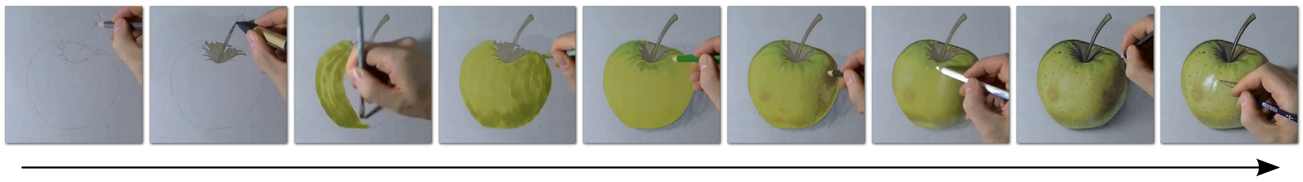



Figure 1: *Our approach enables rich history-based editing operations on physical paintings. From a time lapse recording of the painting process (bottom) we extract translucent paint layers into a temporal creation history. This allows artists to perform otherwise impossible spatio-temporal selections & edits which leverage temporal information to produce complex effects such as color gradients controlled by time (top left) or temporal eraser (top right). (Paint layers in this example use Porter-Duff "over" blending.) Time lapse video © Marcello Barenghi.*

Abstract

The creation of a painting, in the physical world or digitally, is a process that occurs over time. Later strokes cover earlier strokes, and strokes painted at a similar time are likely to be part of the same object. In the final painting, this temporal history is lost, and a static arrangement of color is all that remains. The rich literature for interacting with image editing history cannot be used. To enable these interactions, we present a set of techniques to decompose a time lapse video of a painting (defined generally to include pencils, markers, etc.) into a sequence of translucent "stroke" images. We present translucency-maximizing solutions for recovering physical (Kubelka and Munk layering) or digital (Porter and Duff "over" blending operation) paint parameters from before/after image pairs. We also present a pipeline for processing real-world videos of paintings capable of handling long-term occlusions, such as the painter's hand and its shadow, color shifts, and noise.

CR Categories: I.3.7 [Computer Graphics]: Picture/Image Generation—Bitmap and framebuffer operations I.4.6 [Image Processing and Computer Vision]: Segmentation—Pixel classification;

Keywords: images, surfaces, depth, time, video, channel, segmentation, layers, photoshop, painting

*e-mail:tanjianchaoustc@gmail.com

ACM Reference Format
Tan, J., Dvoroznak, M., Sykora, D., Gingold, Y. 2015. Decomposing Time-Lapse Paintings into Layers. ACM Trans. Graph. 34, 4, Article 61 (August 2015), 10 pages. DOI = 10.1145/2766960 http://doi.acm.org/10.1145/2766960.

1 Introduction

A painting is an arrangement of colors on a 2D canvas. During the painting process, artists deposit color throughout the canvas via a sequence of strokes, often with a real (or simulated, in the case of digital paintings) paint brush. The final painting is, from a computational point of view, a grid of unstructured color values. Extracting structure from the final painting is extremely challenging. Yet the temporal record, which is lost in the final painting, is informative about the scene being painted. Complex drawings are drawn according to a hierarchical structure [Taylor and Tversky 1992]. Objects and parts of objects are drawn together using lower-level rules [Van Sommers 1984; Novick and Tversky 1987; Tversky 1999].

Interacting with editing history is a powerful concept in human-computer interaction. (See Nancel and Cockburn's CAUSALITY [2014] for a recent survey and conceptual model.) This rich literature on history systems extends far beyond undo/redo. For *digital* image editing, this literature includes a generalization of layers for scaling, resizing, and recoloring strokes [Nancel and Cockburn 2014], revision control [Chen et al. 2011], grouping command history [Chen et al. 2012], learning from or reapplying previous commands [Grossman et al. 2010; Berthouzoz et al. 2011; Xing et al. 2014]. Wetpaint [Bonanni et al. 2009] explored a tangible "scraping" interaction to visualize layered information, such as a painting's history. Such powerful interactions are unavailable for physical paintings, even when digitized.

To enable these interactions, we propose to extract editing history from paintings by analyzing time lapse videos of their creation and decomposing them into translucent paint layers. These paint layers correspond to the "commands" necessary for history-based editing applications. Throughout this paper (unless stated otherwise), we use "paint" in a general sense to mean any kind of physical marking, including pencils, pens, markers, watercolor, acrylic, etc. Time lapse videos of paintings are readily available online, often for instructional or demonstrative purposes. Our analysis can also be used to recover a command history for time-lapse videos of digital paintings; this is useful for applying history-based editing operations to popular digital painting applications.

There are two primary challenges to extracting paint layers from time lapse videos. The first challenge is that, given two images from a time lapse, the paint layer which effected the observed difference is ambiguous. For example, an observed change from white to pink could be the result of opaque pink paint or translucent red paint. Because opaque paint completely covers or hides colors underneath, it limits the reuse of covered strokes in history-based editing operations. Therefore, we seek maximally translucent paint in order for the layers to be maximally reusable. In Section 3, we present solutions based on the Kubelka-Munk model of pigment layering [Kubelka 1948] and the standard Porter and Duff [Porter and Duff 1984] "over" blending equation used throughout computer graphics. The second challenge is that time-lapse painting videos contain many changes *not* due to the application of paint. Spurious changes may be caused by occlusions, such as the painter's hand or brush and accompanying shadows; camera motion, refocusing, or color balance changes; compression noise, watermarks, or overlays; and dynamic effects like canvas vibration and the drying of watercolor. Unlike the foreground object subtraction problem in computer vision, some parts of the canvas may be occluded more often than not. In Section 4, we present a solution for processing real-world time-lapse painting videos. Our solution removes even extremely frequent occlusions, noise, and global color shifts. Camera and canvas motion are beyond the scope of this work.

Contributions

- A pipeline for processing real-world time-lapse paintings (Section 4), capable of suppressing long-term occlusions such as the painter's hand and its shadow, short-term occlusions like brushes, noise due to compression and lighting, and color shifts.
- Extremely efficient algorithms for decomposing image sequences into translucent paint layers according to either the Kubelka-Munk model for physical material layering, an extension of their model for mixing, or the standard linear blending algorithm used in digital painting (Section 3).

Our contributions are in the generation of this data, not in its downstream applications (Section 5).

2 Related Work

Interacting with editing history In addition to the previously mentioned history-based interactions for image editing, related interactions have also been proposed for 2D and 3D vector graphics. Su et al. [2009] presented a technique for 2D vector graphics editing which suggests objects to select based on previous selections in the user's command history. Noris et al. [2012] presented a scribble-based approach to segment 2D vector graphics sketches based on time of creation, which helps distinguish nearby strokes drawn at very different times. VisTrails [2009] is a commercial tool for reviewing and reusing command history in the commercial 3D modeling package Maya. Denning and Pellacini [2013] presented algorithms for revision control of 3D meshes. Chen et al. [2014] introduced a technique for choosing good views and segmenting 3D models based on the editing history. Two recent works analyze and visualize changes in outdoor, urban scenes [Matzen and Snavely 2014] and construction sites [Karsch et al. 2014]. These approaches operate on a collection of photographs from different viewpoints, relying on structure-from-motion to obtain a 3D reconstruction (the former) or a 3D architectural model (the latter). Finally, while not about editing history per se, McCann and Pollard [2009; 2012] introduced two noteworthy generalizations of layers for image editing.

Decomposing edits Hu et al. [2013] investigated the related problem of recovering an image editing operation from a pair of images. The editing operations they support are duplicating and geometrically transforming an image region, and adjusting the color of an image region. We solve the orthogonal problem of recovering maximally translucent paint layers using both a physical Kubelka-Munk model and the traditional "over" digital compositing operation. Our work also includes a far more efficient algorithm for finding per-pixel opacity for a single-color layer.

Amati and Brostow [2010] analyzed videos of sumi-e paintings, a style of monochromatic (shades of black) art with relatively few strokes. They find a small number of clean frames by comparing binary thresholded frames to the binary thresholded final painting. The result of their analysis is a segmentation of the final painting into parts (e.g. leaves and flowers). We are inspired by this work and share the observation that paint is far more temporally stable than occlusions, and that algorithms for skin detection and foreground subtraction are unsuitably unstable for analyzing painting videos.

Fu et al. [2011] extract an animated stroke order from static line drawings based on such cognitive and geometric properties. They operate at the part level and take as input a drawing segmented into objects. In contrast, we operate on paintings and are already given a time lapse sequence of its creation. Both our work and Fu et al. [2011] rely on a similar assumption, that the order of markings made when drawing or painting is not random.

Xu et al. [2006] introduced an algorithm that decomposes a single image of a Chinese painting into a collection of layered brush strokes. This comprises image segmentation, detecting the curves of painted strokes, and separating colors for overlapping strokes. Richardt et al. [2014] presented a semi-automatic approach to decompose single images into a mix of transparent and opaque vector graphic layers, where the vector graphics can be filled with linear and radial gradients. In contrast, our approach relies on a video of the creation process, which simplifies segmentation though not color separation. We treat color separation in terms of both the Kubelka-Munk physical layering model and the digital compositing "over" operation, and transform the problem into a simple geometric one in RGB-space.

Image matting Color separation is also related to layer extraction and blue-screen matting. Szeliski et al. [2000] presented a solution to the layer extraction problem in which two independent "images" are layered on top of each other as a result of reflection or transparency. Their approach requires that the layered images are moving with respect to each other, a reasonable assumption for a reflection on a window, but one that does not holds in our scenario. Farid and Adelson [1999] introduced another solution to this problem, but require as input two photographs taken with a polarized light filter. Smith and Blinn [1996] study the related blue-screen matting problem of separating a potentially translucent foreground object from one or two known backgrounds. Their analysis of the problem

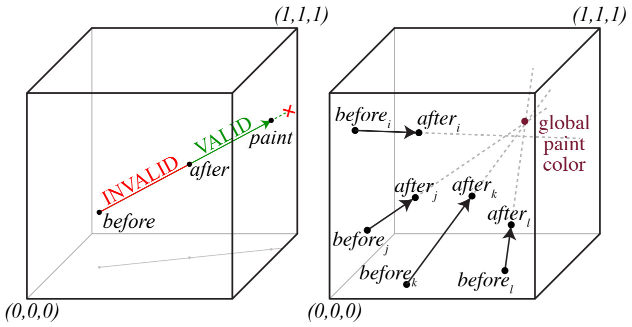
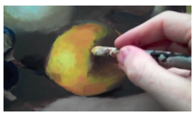


Figure 2: *The cube of valid RGB colors. Left: The color of a pixel before and after a modification to a painting defines a line. With Porter-Duff's "over" compositing, the paint color must lie on the portion of the line beyond after and within the cube. Right: In RGB-space, the after color of every pixel (here, i, j, k, l) affected by a stroke is the result of pulling its before color towards the stroke's color.*

shows that the problem is, in general, underspecified. Zongker et al. [1999] solve a generalized version of the matting problem which allows for reflections and refractions.

3 Decomposing Time Lapse Paintings

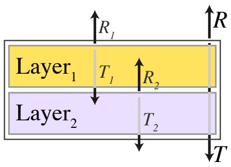

© Will Kemp

Viewed as a time lapse, the darkening of a lemon (right) has an ambiguous interpretation. The darker color could either be the result of painting with an opaque, darker shade of yellow, or else it could be the result of painting with a "translucent" black or brown pigment. (Translucency could be the result of pigment layering or mixing; see below.) The opaque interpretation is always possible, but completely hides all previous information, preventing its use in later editing applications. For history-based editing operations, we claim that the translucent interpretation is more useful and, in general, a better conceptual match to the painting process. In other words, we wish not to distinguish between paint mixed on the easel and paint mixed directly on the canvas, and to view even "opaque" acrylic paint as potentially translucent.

Input We take as input a time lapse recording of a painting in the form of a sequence of albedo or reflectance images $I_{t_1}, I_{t_2}, \ldots, I_{t_n}$. (We describe our process for obtaining such images from real-world videos in Section 4.) An image I_{t_i} stores the state of the painting at time t_i. Each RGB color channel of a pixel store a value in $[0, 1]$ representing the overall fraction of incident light that is reflected at that wavelength. Given two such images $I_{t_{i-1}}$ and I_{t_i}, our goal is to recover the per-pixel parameters of "paint" P_{t_i} such that applying P_{t_i} to $I_{t_{i-1}}$ results in I_{t_1}. We do this using a physical Kubelka-Munk model of material layering (Section 3.1, $P_{t_i} = (R_{t_i}, T_{t_i})$) and using the linear blending equation [Porter and Duff 1984] commonly used for digital compositing and painting (Section 3.2, $P_{t_i} = \mathrm{RGB}_{t_i}$).

3.1 Recovering Physical Paint Parameters

Kubelka and Munk [Kubelka and Munk 1931; Kubelka 1948] modeled the reflectance R and transmittance T of a layer of homogeneous material in terms of the material's absorption and scattering coefficients K and S. (All parameters are per-wavelength.) The model is widely used in paint, plastic, paper, and textiles and has previously been used in computer graphics for accurately simulating paint [Konieczny and Meyer 2009; Curtis et al. 1997; Baxter et al. 2004; Haase and Meyer 1992; Lu et al. 2014; Budsberg 2007; Dannen 2012]. A summary of useful formulae related to the Kubelka-Munk model can be found in [Barbarić-Mikočević and Itrić 2011]. When multiple materials are present, the Kubelka-Munk model can be used in two ways: layering and mixing.



Layering Kubelka [1948] presented formulae for the overall reflectance and transmittance of a stack of non-opaque layers, given each layer's individual reflectance R and transmittance T coefficients.[1] R and T can be expressed in terms of absorption, scattering, and thickness parameters, but it is more straightforward and involves fewer parameters to simply consider R and T. In our scenario, we observe a sequence of reflectance images I_{t_i}. Each I_{t_i} stores the overall reflectance underneath every pixel at time t_i. I_{t_0}, the initial frame, stores the reflectance of the backing material, such as a blank canvas or sheet of paper. We wish to find the reflectance R_{t_i} and transmittance T_{t_i} of the paint layer that results in I_{t_i} when placed above $I_{t_{i-1}}$. The equation is [Kubelka 1948; Kubelka 1954]:

$$I_{t_i} = R_{t_i} + \frac{T_{t_i}^2 I_{t_{i-1}}}{1 - R_{t_i} I_{t_{i-1}}} \tag{1}$$

$R, T \in [0, 1]$ and, due to the conservation of energy, $T + R \leq 1$. (The fraction of illumination *absorbed* by the layer is $1 - T - R$.) In general, there are infinitely many solutions, including the opaque solution $T_{t_i} = 0, R_{t_i} = I_{t_i}$. To maximize the reusability of recovered layers, we find the solution that maximizes the transmittance T:

$$\begin{array}{llll}
\text{if} & I_{t_{i-1}} = 0: & R_{t_i} = I_{t_i}, & T_{t_i} = 1 - I_{t_i} \\
\text{else if} & I_{t_i} = 0: & R_{t_i} = 0, & T_{t_i} = 0 \\
\text{else if} & I_{t_i} + \frac{1}{I_{t_{i-1}}} \leq 2: & R_{t_i} = 0, & T_{t_i} = \sqrt{\frac{I_{t_i}}{I_{t_{i-1}}}} \\
\text{else if} & \frac{I_{t_i}}{I_{t_{i-1}}} \leq 1: & R_{t_i} = 0, & T_{t_i} = \sqrt{\frac{I_{t_i}}{I_{t_{i-1}}}} \\
\text{else:} & & R_{t_i} = X = \frac{\frac{I_{t_i}}{I_{t_{i-1}}} - 1}{I_{t_i} + \frac{1}{I_{t_{i-1}}} - 2} & \\
 & & T_{t_i} = 1 - X &
\end{array}$$

To the best of our knowledge, these transmittance-maximizing solutions have not previously been presented. See Appendix A for a derivation. In our setting, where the I are RGB reflectance images, R and T are also RGB images (with values in $[0, 1]$) and easily visualized (Figure 4). R is an additive image (transparent R is black) and T is a multiplicative image (transparent T is white). Our solution continuously transitions between multiplicative blending with bright backgrounds and additive blending with dark ones. Nearly all changes are partially translucent, and more extreme changes gradually become opaque. We compare our maximally transmissive solutions to scanned watercolor layers in Figure 3.

Mixing The Kubelka-Munk mixing model, while suitable for homogeneous mixtures such as wet paint, is not as suited for our purposes as layering. The mixing model can only approximate a completely opaque layer of paint via very large mixing coefficients (see supplemental materials). Moreover, the scattering and absorption parameters are less intuitive. They have no upper bound, unlike transmittance and reflectance, which range from 0 to 1, and can be visualized as an additive and multiplicative image. For completeness,

[1]Kubelka [1948] points out that the layering equation we use was independently derived and presented by Gurevic in 1930 and by Judd in 1934.

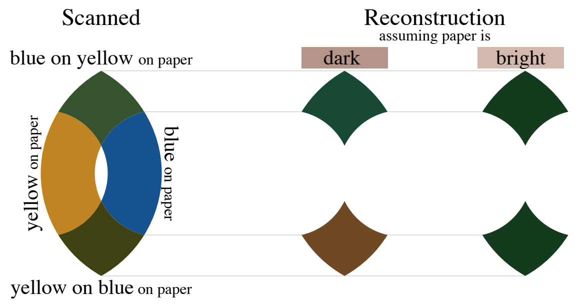


Figure 3: *Our Kubelka-Munk layering model recovers different reflectance and transmittance layers depending on the backing paper's brightness. To illustrate this difference, we scan blue and yellow watercolor paint and their overlap (left column). By assuming dark or bright backing paper, we can correspondingly recover different pairs of reflectance and transmittance layers for the blue and yellow paint, and reconstruct the overlap colors (middle and right columns). When assuming dark backing paper (middle column), recovered layers are both reflective (additive) and transmissive (multiplicative), and the overlap color will depend on the layer order. When assuming bright backing paper (right column), the recovered paint layers are purely transmissive (multiplicative). Purely transmissive layers commute; the overlap color is independent of the layer order, and the two reconstructed overlap colors are the same.*

we include solutions for minimal modification of current mixing parameters in the supplemental materials.

3.2 Recovering Digital Paint Parameters

In digital painting and compositing, the standard blending equation is Porter and Duff [1984]'s "over" operation:

$$\textit{after} = (1-\alpha)\textit{before} + \alpha \cdot \textit{paint} \tag{2}$$

In our setting, we treat *before* and *after* as the observed per-pixel RGB reflectance in $I_{t_{i-1}}$ and I_{t_i}. The layer's translucency $\alpha \in [0,1]$ is the interpolation parameter between *before* and *paint*.

To determine α and *paint*, we view the blending equation geometrically in RGB space (Figure 2). For every changed pixel, possible *paint* colors lie on the line passing through *before* and *after*, and, for a given *paint* color on the line, $\alpha = \frac{\textit{after}-\textit{before}}{\textit{paint}-\textit{before}}$. There are additional constraints: $0 < \alpha \leq 1$ and *paint*'s RGB components must all lie within $[0,1]$. Note that *paint* cannot deviate from this line, or else *before* would not exactly reconstruct *after*.

This is still, however, an under-constrained problem. We propose two techniques for choosing *paint* and α, one that minimizes α and one that computes a consistent *paint* color among pixels. A comparison of these techniques can be seen in Figure 4. To minimize α, each pixel's *paint* is chosen to be the intersection of its line with the RGB cube itself. This is equivalent to choosing the most extreme *paint* possible. This SMALL-ALPHA approach is extremely efficient and results in "minimally" opaque layers.

When minimizing α, however, pixels' *paint* colors aren't necessarily consistent (Figure 4). For short time-lapse intervals, however, we expect that the artist will only have used one color. Our CLOSEST-PAINT approach finds the color that minimizes the squared distance (in RGB-space) to every changed pixel's line. The minimizing color is then projected onto each pixel's $\overleftrightarrow{\textit{before after}}$ line. Solving the least squares problem entails solving a simple 3×3 linear system

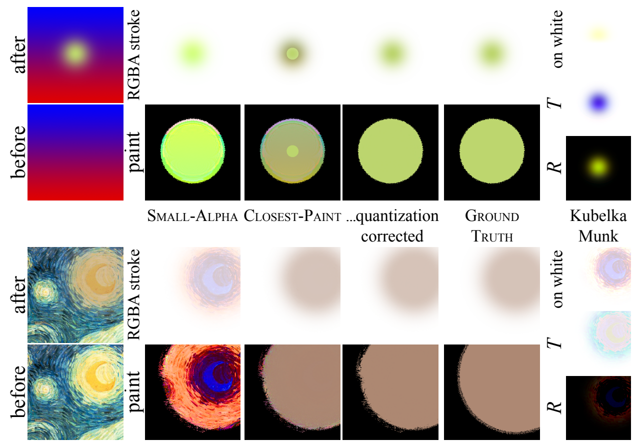


Figure 4: *Before and after a stroke is applied. The RGBA layers recovered by our algorithms* SMALL-ALPHA, CLOSEST-PAINT, CLOSEST-PAINT *with quantization correction, and the ground truth stroke itself. The Kubelka-Munk R and T values for the layer, and the R, T values composited with white.*

of equations (3 being the number of color channels):

$$E = \left\| \frac{u_i \cdot (\textit{paint} - \textit{before}_i)}{\|u_i\|^2} u_i + \textit{before}_i - \textit{paint} \right\|^2, \tag{3}$$

where $u_i = \textit{after}_i - \textit{before}_i$. Building the 3×3 system requires a summation over all n changed pixels. This approach is far more efficient than the $n \times n$ approaches presented in Xu et al. [2006] and Hu et al. [2013] and assumes nothing about α's smoothness.

The intuition behind the CLOSEST-PAINT approach is that every pixel affected by a stroke is pulled towards the stroke's *paint* color, by an amount determined by each pixel's α parameter (Figure 2). After projecting the global paint color onto the valid region of each pixels' line, we compute per-pixel α values.

Two minor improvements to CLOSEST-PAINT account for the quantization of (typically 8-bit) color component values. Taking quantization into account, we seek *paint* and α such that

$$\textit{after} = \text{ROUND}\left((1-\alpha)\textit{before} + \alpha \cdot \textit{paint}\right)$$

Because *after* is the result of rounding, the line that *paint* must lie on can pass through any part of the RGB "pixel cube" that rounds to *after*. The accuracy of each line's direction is proportional to its length, so our first improvement is to weight each term in Equation 3 by $\|\textit{after}_i - \textit{before}_i\|^2$. Once we solve the least squares problem and find the global paint color, our second improvement is to project *after* onto the line between *before* and the minimizing paint color—or as far towards the line as possible without crossing the boundary of *after*'s "pixel cube." We perform this quantization-correcting projection prior to the final projection of the minimizing color onto the valid region of the line passing through *before* and $\widehat{\textit{after}}$.

4 Processing time lapse videos

Time lapse painting videos are readily available online, often for instructional or demonstration purposes. Before these videos can be processed by our layer decomposition algorithms (Section 3), they must first be filtered to eliminate changes *not* due to the application of paint and finally converted to reflectance (albedo) images. Spurious changes can be caused by the environment (occlusions

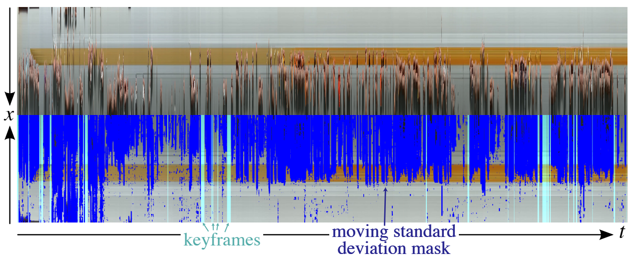


Figure 6: *A slice of the* egg *video in time, and its annotated reflection. Many pixels are occluded for the majority of frames. Video © Marcello Barenghi.*

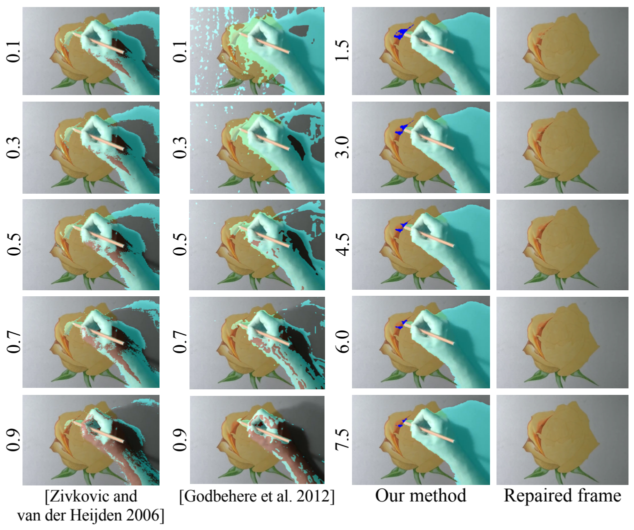


Figure 7: *Background estimation algorithms are challenged by time-lapse painting videos. We train Zivkovic and van der Heijden [2006] and Godbehere et al. [2012] on 200 frames and vary their respective thresholds. Both algorithms classify painted strokes as foreground (or, with a liberal threshold, misclassify the hand and shadow). Our algorithm's keyframe mask (cyan) suppresses the hand and shadow; the moving standard deviation (pure blue), with varying thresholds, masks the pencil tip and little additional paint. Video © Marcello Barenghi.*

of the canvas, such as the painter's hand or instruments, and accompanying shadows); camera (motion or color balance changes); post-processing (compression noise, permanent watermarks or overlays, and other visual effects); and dynamics (canvas vibration or motion and the drying of watercolor). In this work, we process time lapse videos with long-term occlusions, global color shifts, compression noise, and a mild amount of watercolor paint dynamics. We assume that the camera and canvas are fixed. Camera motion, canvas vibration and motion, and permanent post-processing watermarks and overlays are beyond the scope of this work. We further assume that the input video has been manually cropped to the canvas.

Existing background/foreground estimation algorithms are not well-suited for our particular input (a finding shared by Amati and Brostow [2010]). To begin, many pixels are occluded or in shadow for the *majority* of frames (Figure 6). The "background" in our case is the painting, which is constantly changing in tandem and exactly underneath the foreground object. The occluder includes a paintbrush whose tip is exactly the same color as the paint we are interested in. Likewise, we do not want to use a model of human skin, because the hand may be similar to the paint colors. Background estimation algorithms [Zivkovic and van der Heijden 2006; Godbehere et al. 2012] are challenged by our data and would produce large numbers of spurious changes (Figure 7). Finally, watercolor and markers take some time to dry, so they include some dynamics. If we waited until they were completely dry, layer order would be strongly affected.

The recent time lapse motion compensation algorithm of Rubinstein et al. [2011] is limited to very short sequences (300 frames took 50 hours, whereas our input sequences have ~5000 frames). They search for motion in space as well as time; we do not need this significant computational overhead. Nearly all steps in our video processing pipeline are per-pixel (and embarrassingly parallel).

4.1 Our pipeline

Our video processing pipeline (Figure 5) handles global color-shifts, removes noise, and is capable of ignoring frequent or majority occlusions. To motivate our approach, consider the changes to a horizontal slice of a painted egg in time (Figure 6). The color of an unpainted background pixel drifts significantly. The color of all pixels exhibit significant noise (some due to video compression and some due to global illumination effects). Many pixels, especially towards the right side of the painting (the painter's dominant handedness) are occluded in the *majority* of frames.

Our approach is based on two key observations. **I.** The value of an unoccluded pixel should be piecewise constant in time; equivalently, changes to a pixel on the canvas should be sparse in time. (The stability of paint versus occluders has also been noted by Amati and Brostow [2010].) Our approach is based on detecting and ignoring unstable pixels via moving standard deviation. **II.** Identical sequences of frames, which indicate that no occluders are present, provide crucial checkpoints for the progress of the painting. We call such identical frames **keyframes**. We use keyframes to mask and eliminate spurious changes in the intervening frames.

Color correction The first step in our pipeline corrects color drift between adjacent frames. We do not include all pixels in the calculation, as many pixels may have changed due to occlusion or paint. We solve for the per-channel linear function (offset and slope) that minimizes the color difference in a least-squares sense, considering only pixels whose magnitude of change lies between first and second octiles. (Pixels which changed the least may be due to outliers, e.g. oversaturated pixels.) We found this approach to be stabler than the non-linear color correction described in [Hu et al. 2013].

Keyframe detection The second step in our pipeline searches for keyframes, or sequences of repeated frames. We assume that a sequence of repeated frames implies no foreground occluders. We consider two frames to be repeated if they differ in less than n pixels; we detect differences with an L*a*b*-space threshold. (See Table 1 for all parameters.) After a keyframe is detected, we reduce noise by replacing its pixels with the per-pixel average in time. With these known good frames, we perform our color correction step a second time, this time aligning every frame with the most recent keyframe.

Inter-keyframe processing Keyframes allow us to reduce the number of pixels under consideration. A pixel which does not change from one keyframe to another should not change in the intervening frames; any such changes should be ignored. We compute a difference mask between adjacent keyframes for use in inter-keyframe processing. We again detect differences with a L*a*b*-space threshold and close pixel-sized gaps in the mask with a 3×3 topological closing operation. Outside of the mask, we linearly interpolate colors between keyframes. Inside of the mask, we detect and repair

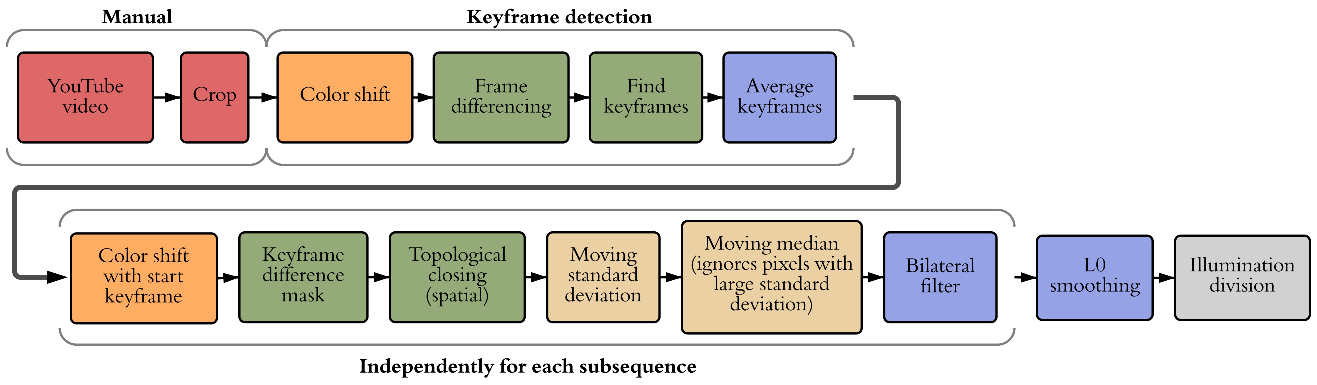

Manual
Keyframe detection
YouTube video
Crop
Color shift
Frame differencing
Find keyframes
Average keyframes
Color shift with start keyframe
Keyframe difference mask
Topological closing (spatial)
Moving standard deviation
Moving median (ignores pixels with large standard deviation)
Bilateral filter
L0 smoothing
Illumination division
Independently for each subsequence

Figure 5: *Our pipeline for processing time lapse painting videos. See Section 4.1 for details.*

	rose	egg	eye	apple	cube	candy	cola	graffiti
detection L*a*b* threshold	10	10	10	7	10	10	12	8
detection # differing pixels	50	50	50	50	50	50	80	50
detection # repeated frames	4	4	4	4	4	4	4	2
mask L*a*b* threshold	8	8	8	9	9	8	8	4
standard deviation threshold	1.5	0.5	0.5	0.6	1	0.8	1	1.5

Table 1: *Keyframe parameters in our video processing pipeline.*

occlusions. Painted marks are sparse in time and smoothly varying or piecewise constant; we detect unstable values—occluders—at each pixel with a thresholded L*a*b*-space moving standard deviation (in time, window size 7). We repair unstable pixels with a moving median filter (in time) [Cheung and Kamath 2004] whose window is made up of the $m = 9$ most recent stable pixels. The moving standard deviation threshold is the most sensitive parameter of our entire video processing pipeline. A small threshold minimizes false positives (occlusions) but degrades the temporal resolution of layers in the processed video. We have not found this to be a problem in our history-editing based applications (Section 5). See the supplemental materials for processed videos in which we manually selected thresholds for each subsequence to be as large as possible while still suppressing occluders.

Smoothing To suppress temporal noise and enforce temporal sparsity, we perform adaptive bilateral filtering [Tomasi and Manduchi 1998] on each inter-keyframe sequence (in time, with window size 15 and maximum $\sigma = 200$), and then we perform L_0 smoothing [Xu et al. 2011] (in time, $\kappa = 1.1, \lambda = 0.001$) for each pixel across all frames in the entire video. In 1D, Xu et al.'s L_0 smoothing algorithm amounts to repeatedly solving a tridiagonal system of equations; we constrain the very first and last pixels' values (with a hard constraint) to ensure that they remain unchanged.

Illumination division To convert our scrubbed videos into albedo (reflectance) images, we must divide each pixel's value by the incoming illumination. We assume that the spatial variation in illumination is unchanged throughout the entire sequence (e.g. the light source does not move) and divide each pixel's value by an illumination estimate computed from the first frame. In the first frame, the canvas is blank or nearly blank, so we perform a large, spatial median (window size 55) and use each pixel's maximum value (in any channel) as the unnormalized illumination divisor. Although the canvas material's true reflectance is unknown, estimates for the reflectance of paper placed on a non-white surface [Hubbe et al. 2008] typically range from 0.5 to 0.7. We normalize all pixels' illumination divisors with the globally brightest value encountered at any point in the video. This value is typically around 0.6; we cap it at 0.7. This step ensures that the brightest value in our albedo images is less than 1.

Input videos and the result of our pipeline can be seen in Figure 8 and in the supplemental material.

5 Applications

The layer decomposition for a painting stores its time lapse history as a sequence of RGBA or reflectance and transmittance images containing the individual layers (see Figure 8 and the supplementary materials) extracted from the original time lapse sequence using one of the algorithms from Section 3. As these layers typically affect only a small portion of the entire painting and are spatially coherent, they can be represented compactly in memory using simple run-length encoding. To achieve interactive response, we also store bounding boxes of layers and compare them with the bounding box of the edit so that the compositing operations can be done on a small fraction of the total pixels. Moreover, when the edit affects a certain interval in time, we can pre-composite prior and subsequent layers into two RGBA images or four reflectance and transmittance images, which are then blended together with the modified content.

Translucency-maximizing solutions are stable, general, and useful. The CLOSEST-PAINT method, which seeks to recover the true color and transparency, is appropriate for clean data where the difference between frames truly is a Porter-Duff over composite with a single color, such as digital painting sequences (supplemental materials). Physical painting sequences are noisy, so we use translucency-maximizing solutions, which make no assumptions about the input data. The Kubelka-Munk layering model is a physically-based approach. When editing, the reflectance and transmittance layers can be hue-shifted or modulated (increasing transparency and decreasing the reflectance). The linear (Porter-Duff) model is "correct" for extracting digital layers. It also works well for physical paintings. It has the advantage that it is simple to work with, as it is built into digital editing tools and GPU's.

The apple in Figures 1 & 9 and the rose in Figure 9 were edited with the Porter-Duff model. The eye in Figure 12 was edited with the Kubelka-Munk layering model. The Kubelka-Munk model typically produces layers that are more "translucent" than those produced in the Porter-Duff model. The graffiti example in Figure 10 provides a side-by-side comparison of the two models.

For videos with approximately 500×500 resolution, our entire pipeline runs at around 2 frames per second on a single core of an Intel Core i7-3520M CPU (except for the whole-video L_0 smoothing stage). Most stages are pixelwise, and the algorithms are embar-

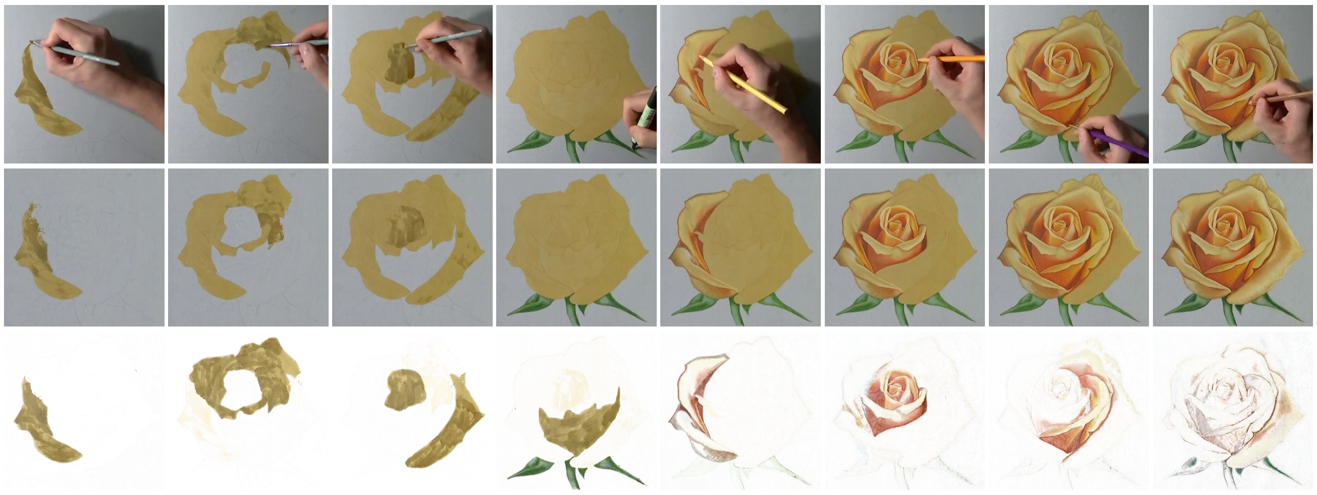


Figure 8: *Examples of layers (bottom) reconstructed from a time lapse video (top) which was pre-processed using our pipeline (middle). Individual layers were grouped into larger clusters to enhance their visibility. Time lapse © Marcello Barenghi.*

rassingly parallelizable. The final whole-video L_0 smoothing stage was parallelized and run on a 60-core cluster of Intel Xeon E5-2670 CPU's; this stage took around 10 minutes for a ~5000 frame video. The decomposition of the ~5000 frame rose example takes ~60 MB to store as a sequence of RGBA images (PNG format) and ~140 MB as a sequence of Kubelka-Munk reflectance and transmittance images (as gzipped double-precision floating point values).

Given a painting and its layer decomposition, we can perform various selection and editing operations (see Figure 1, 9, 10, 11, 12, and the supplementary materials) that generalize the notion of layers in digital painting programs.

Spatio-temporal selection & layering There are several possibilities for using temporal information for selection. Users can position the mouse at a specific spatial location and pick the temporal value of the pixel underneath. They can also alter the selected temporal value by scrubbing through time with the mouse wheel. When two or more locations are specified, the tool can select layers which lie in the specified time interval. A more complex selection can be achieved using selection strokes—scribbles. In this scenario algorithms based on temporal information such as [Noris et al. 2012] can be used to improve segmentation results. In Figure 11 we present a simplified solution where temporal statistics of all pixels underneath the scribble are analyzed and then a set of dominant temporal clusters is retrieved which can then be offered to the user as possible candidates for selection. This approach can also be understood as a supervised layer decomposition that helps artist convert the layer decomposition into a smaller set of meaningful layers. The user can perform various edits which composite nicely with prior and subsequent layers, as in a digital image editing workflow (see Figure 1). The great advantage of our approach is that the user can create different sets of layers *ex post* even for physical paintings, in contrast with the traditional, digital workflow which involves planning in advance.

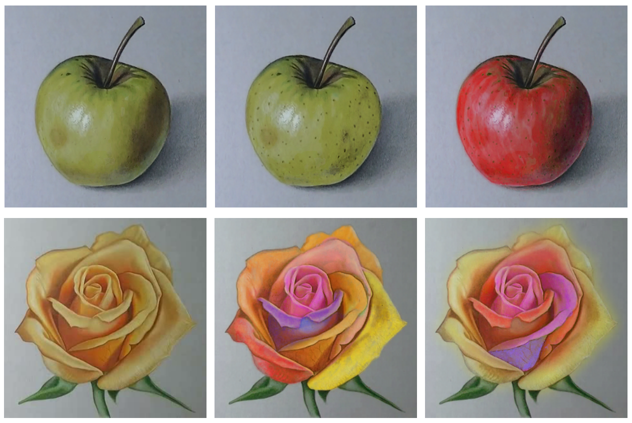


Figure 9: *Edits created using our Porter-Duff layer decomposition on the apple sequence from Figure 1 (top), and on the rose sequence (bottom, original image is on the left). Time lapse sequences © Marcello Barenghi.*

Modification of extracted layers Besides recoloring extracted layers or clustering them, the user can also modify their structure directly or use temporal information to perform additional operations. It is possible to erase or re-order selected layers, which has the effect of cloning, undo-ing or replaying a stroke at a different spatial or temporal location. Existing algorithms can be used to perform local layering [McCann and Pollard 2009] as well as soft stacking [McCann and Pollard 2012] to rearrange recovered layers and let them appear partially at multiple time frames. The user may also draw new strokes at specific times, create a spatial selection at a specific time and then move this selection to a different time to perform edits, or use time-of-creation as a parameter which can affect edits, such as the creation of color gradients (see Figure 1, 9, 10, 12, and supplementary material for illustrations of these operations).

6 Conclusions and Future Work

Our video processing pipeline tailored to time lapse painting videos can clean the videos of even persistent occlusions, and produce reflectance images suitable for paint layer decomposition and editing. The less-used Kubelka-Munk layering equations are a convenient alternative to their material mixing model and more suitable for real-world data than the "over" blending operation used throughout computer graphics. These equations can be "solved" efficiently to find maximally transparent stroke layers. With a single-stroke assumption for digital images, our 3×3 system can recover the

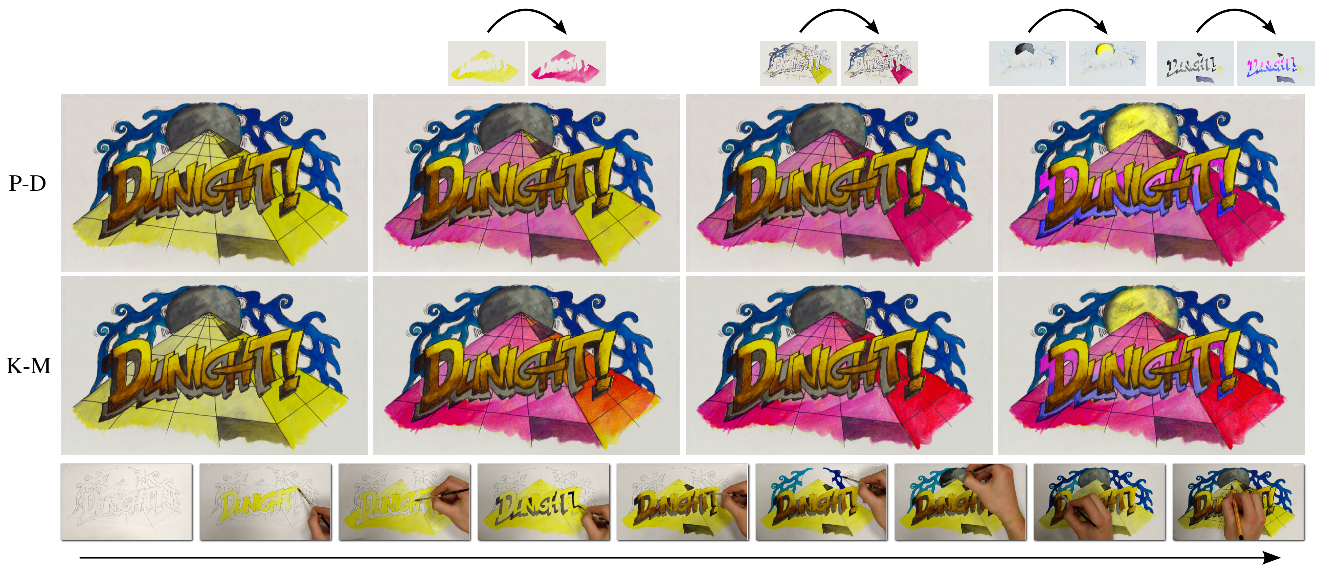


Figure 10: *Comparing edits created with our layer decomposition using the Porter-Duff (first row) and Kubelka-Munk (second row) models. In each model, similar clusters of layers were recolored. Note the difference in layer translucency and the effect of recoloring. The original unmodified painting is on the left while the final composition is on the right. Thumbnails of individual edits are shown on top of the Figure. Time lapse video © Matyáš Veselý.*

original stroke color and its alpha channel extremely robustly (within $\frac{1}{255}$ in each channel).

The history of a painting contains valuable information that can be stored efficiently using run-length encoding and used for a variety of history-based editing applications to generalize the notion of layers. A live implementation for decomposing on-the-fly would allow artists to use physical tools as their artistic input device.

One limitation of our approach is that we do not extract vector strokes. This is challenging because physical tools may create quite complex brush “shapes,” and an approximation, when replayed, would not exactly match the final painting.

If an input video has too-low temporal resolution, multiple overlapping strokes could be painted between adjacent frames, and the original painted colors may not be recoverable. Moreover, the greater the magnitude of change, the less transparent the solution. Extreme noise manifests as spurious strokes that appear and then disappear. Artist drawing order will affect applications such as selection based on time ranges (including our clustering). This is true for all applications of editing history, regardless of the problem domain. Our work provides a new data source, but does not aim to solve problems in the literature on interacting with editing history.

In the future, we plan to replace software instrumentation in systems such as Chronicle [Grossman et al. 2010] and Chen et al. [2011; 2012] with our framework, in order to deliver similar functionality for both digital and real-world paintings. One can imagine, for example, an interactive gallery where visitors can inspect the creation process of individual exhibited paintings using visualization systems like WetPaint [Bonanni et al. 2009]. Recovered layers of real world paintings can also be used to generate painting tutorials in the spirit of Grabler et al. [2009]. Our technique can also enrich the tool set of appearance operators for inverse image editing [Hu et al. 2013]. Such operators can be used for content-adaptive macros [Berthouzoz et al. 2011], to predict the final appearance by example, and to perform automatic completion as in [Xing et al. 2014]. We wish to explore applications in forensics and artwork restoration. Finally, we wish to expand the class of videos we are able to process. Canvas vibrations are a challenge unique to our domain.

A Kubelka-Munk Layering Derivation

Recall Equation 1 and our solution. The first two solutions, when $I_{t_i} = 0$ or $I_{t_{i-1}} = 0$, are self-evident. For $I_{t_{i-1}} \neq 0$, we can express Equation 1 as $T_{t_i}^2 = (R_{t_i} - I_{t_i})(R_{t_i} - \frac{1}{I_{t_{i-1}}})$. Recall our physical constraints $0 \leq R_{t_i}, T_{t_i}, I_{t_i}, I_{t_{i-1}} \leq 1$ and $R_{t_i} + T_{t_i} \leq 1$. The latter is equivalent to $T_{t_i}^2 \leq (1 - R_{t_i})^2$. Thus we have an equality for $T_{t_i}^2$ which is a parabola in R_{t_i}, and an inequality for $T_{t_i}^2$, also a parabola in R_{t_i}. Both parabola open upwards, and both reach their minima when $R_{t_i} > 0$ (because neither I_{t_i} nor $I_{t_{i-1}}$ equal 0), and so are already increasing in $T_{t_i}^2$ when $R_{t_i} = 0$. The remaining solutions follow algebraically by intersecting the parabolas with each other and with the line $T_{t_i}^2 = 1$.

Acknowledgements

We would like to thank Bob Sumner, Paula Crawford, Jay Hendrick, Nathan Loda, and all anonymous reviewers for their constructive comments. We are grateful to Stephen Marschner for an illuminating discussion about the Kubelka-Munk mixing model and to Chris Kim, Will Xu, and Jae Yun for initial experiments. Authors Tan and Gingold were supported in part by the United States National Science Foundation (IIS-1451198 and IIS-1453018) and a Google research award. Authors Dvorožňák and Sýkora were supported by the Technology Agency of the Czech Republic under the research program TE01020415 (V3C – Visual Computing Competence Center), by the Czech Science Foundation under research program P202/12/2413 (OPALIS), and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS13/214/OHK3/3T/13 (Research of Progressive Computer Graphics Methods). Several experiments were run on ARGO, a research computing cluster provided by the Office of Research Computing at George Mason University, VA. (URL:`http://orc.gmu.edu`).

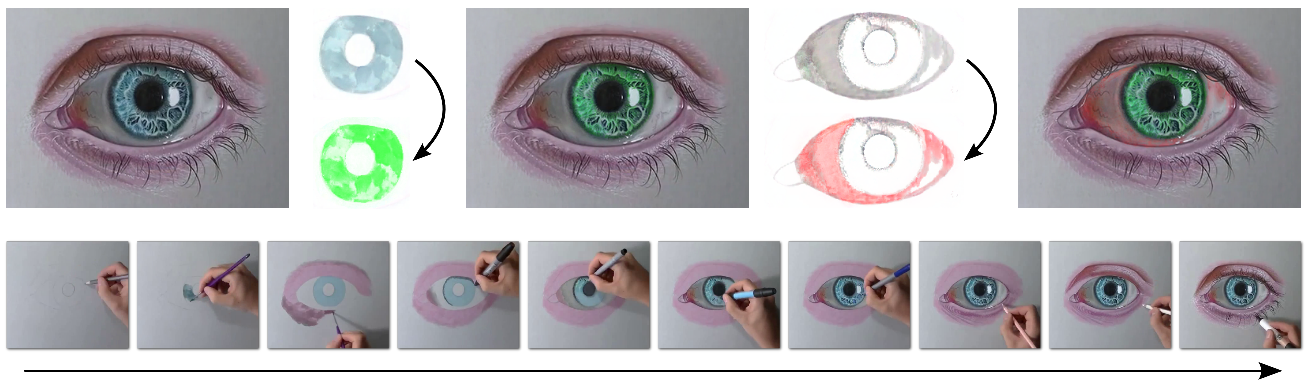


Figure 12: *Editing a decomposed painting with the Kubelka-Munk model. The user modifies colors in two extracted layers to adjust the original painting. Unprocessed time lapse recording of the painting process is depicted below. Time lapse video © Marcello Barenghi.*

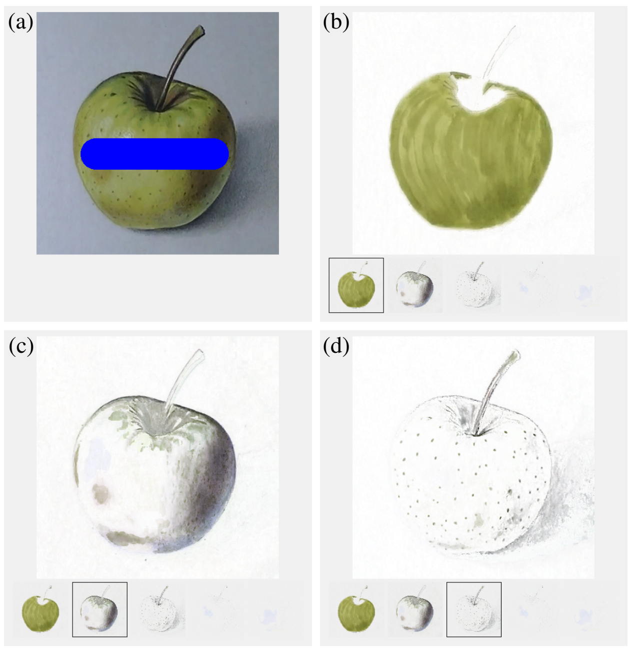


Figure 11: *Image selection & layering: the user draws a selection scribble (a), the system then suggests several temporal clusters containing pixels underneath the scribble (b-d). The user can select the appropriate cluster and use it as a layer for further editing. Source time lapse video © Marcello Barenghi.*

References

Amati, C., and Brostow, G. J. 2010. Modeling 2.5D plants from ink paintings. In *Proceedings of Eurographics Symposium on Sketch-Based Interfaces and Modeling Symposium*, 41–48.

Barbarić-Mikočević, V. D.-M. Ž., and Itrić, K. 2011. Kubelka-Munk theory in describing optical properties of paper (i). *Technical Gazette 18*, 1, 117–124.

Baxter, W. V., Wendt, J., and Lin, M. C. 2004. IMPaSTo: A realistic, interactive model for paint. In *Proceedings of the International Symposium on Non-Photorealistic Animation and Rendering*, 45–56.

Berthouzoz, F., Li, W., Dontcheva, M., and Agrawala, M. 2011. A framework for content-adaptive photo manipulation macros: Application to face, landscape, and global manipulations. *ACM Transactions on Graphics 30*, 5, 120.

Bonanni, L., Xiao, X., Hockenberry, M., Subramani, P., Ishii, H., Seracini, M., and Schulze, J. 2009. Wetpaint: Scraping through multi-layered images. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 571–574.

Budsberg, J. B. 2007. *Pigmented Colorants: Dependency on Media and Time*. Master's thesis, Cornell Univrsity, Ithaca, New York, USA.

Chen, H.-T., Wei, L.-Y., and Chang, C.-F. 2011. Nonlinear revision control for images. *ACM Transactions on Graphics 30*, 4, 105.

Chen, T., Wei, L.-Y., Hartmann, B., and Agrawala, M. 2012. Data-driven history list for image editing. Tech. Rep. TR-2012-07, The University of Hong Kong.

Chen, H.-T., Grossman, T., Wei, L.-Y., Schmidt, R. M., Hartmann, B., Fitzmaurice, G., and Agrawala, M. 2014. History assisted view authoring for 3D models. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 2027–2036.

Cheung, S.-c. S., and Kamath, C. 2004. Robust techniques for background subtraction in urban traffic video. In *Proceedings of SPIE*, vol. 5308, 881–892.

Curtis, C. J., Anderson, S. E., Seims, J. E., Fleischer, K. W., and Salesin, D. H. 1997. Computer-generated watercolor. In *ACM SIGGRAPH Conference Proceedings*, 421–430.

Dannen, C., 2012. The magical tech behind Paper for iPad's color-mixing perfection, Nov. Accessed: January 10th, 2015.

Denning, J. D., and Pellacini, F. 2013. MeshGit: Diffing and merging meshes for polygonal modeling. *ACM Transactions on Graphics 32*, 4, 35.

Farid, H., and Adelson, E. H. 1999. Separating reflections from images by use of independent component analysis. *Journal of the Optical Society of America A 16*, 9, 2136–2145.

Fu, H., Zhou, S., Liu, L., and Mitra, N. J. 2011. Animated construction of line drawings. *ACM Transactions on Graphics 30*, 6, 133.

GODBEHERE, A., MATSUKAWA, A., AND GOLDBERG, K. 2012. Visual tracking of human visitors under variable-lighting conditions for a responsive audio art installation. In *Proceedings of American Control Conference*, 4305–4312.

GRABLER, F., AGRAWALA, M., LI, W., DONTCHEVA, M., AND IGARASHI, T. 2009. Generating photo manipulation tutorials by demonstration. *ACM Transactions on Graphics 28*, 3, 66.

GROSSMAN, T., MATEJKA, J., AND FITZMAURICE, G. 2010. Chronicle: Capture, exploration, and playback of document workflow histories. In *Proceedings of ACM Symposium on User Interface Software and Technology*, 143–152.

HAASE, C. S., AND MEYER, G. W. 1992. Modeling pigmented materials for realistic image synthesis. *ACM Transactions on Graphics 11*, 4, 305–335.

HU, S.-M., XU, K., MA, L.-Q., LIU, B., JIANG, B.-Y., AND WANG, J. 2013. Inverse image editing: Recovering a semantic editing history from a before-and-after image pair. *ACM Transactions on Graphics 32*, 6, 194.

HUBBE, M. A., PAWLAK, J. J., AND KOUKOULAS, A. A. 2008. Paper's appearance: A review. *BioResources 3*, 2, 627–665.

KARSCH, K., GOLPARVAR-FARD, M., AND FORSYTH, D. 2014. Constructaide: Analyzing and visualizing construction sites through photographs and building models. *ACM Transactions on Graphics 33*, 6, 176.

KONIECZNY, J., AND MEYER, G. 2009. Airbrush simulation for artwork and computer modeling. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, 61–69.

KUBELKA, P., AND MUNK, F. 1931. An article on optics of paint layers. *Zeitschrift für Technische Physik 12*, 593–601.

KUBELKA, P. 1948. New contributions to the optics of intensely light-scattering materials. Part I. *Journal of the Optical Society of America 38*, 5, 448–448.

KUBELKA, P. 1954. New contributions to the optics of intensely light-scattering materials. Part II: Nonhomogeneous layers. *Journal of the Optical Society of America 44*, 4, 330–334.

LU, J., DIVERDI, S., CHEN, W. A., BARNES, C., AND FINKELSTEIN, A. 2014. RealPigment: Paint compositing by example. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, 21–30.

MATZEN, K., AND SNAVELY, N. 2014. Scene chronology. In *Proceedings of European Conference on Computer Vision*. 615–630.

MCCANN, J., AND POLLARD, N. 2009. Local layering. *ACM Transactions on Graphics 28*, 3, 84.

MCCANN, J., AND POLLARD, N. 2012. Soft stacking. *Computer Graphics Forum 31*, 2, 469–478.

NANCEL, M., AND COCKBURN, A. 2014. Causality: A conceptual model of interaction history. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1777–1786.

NORIS, G., SÝKORA, D., SHAMIR, A., COROS, S., WHITED, B., SIMMONS, M., HORNUNG, A., GROSS, M., AND SUMNER, R. 2012. Smart scribbles for sketch segmentation. *Computer Graphics Forum 31*, 8, 2516–2527.

NOVICK, L. R., AND TVERSKY, B. 1987. Cognitive constraints on ordering operations: The case of geometric analogies. *Journal of Experimental Psychology: General 116*, 1, 50.

PORTER, T., AND DUFF, T. 1984. Compositing digital images. *ACM SIGGRAPH Computer Graphics 18*, 3, 253–259.

RICHARDT, C., LOPEZ-MORENO, J., BOUSSEAU, A., AGRAWALA, M., AND DRETTAKIS, G. 2014. Vectorising bitmaps into semi-transparent gradient layers. *Computer Graphics Forum (Proceedings of EGSR) 33*, 4 (July), 11–19.

RUBINSTEIN, M., LIU, C., SAND, P., DURAND, F., AND FREEMAN, W. T. 2011. Motion denoising with application to time-lapse photography. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 313–320.

SMITH, A. R., AND BLINN, J. F. 1996. Blue screen matting. In *ACM SIGGRAPH Conference Proceedings*, 259–268.

SU, S. L., PARIS, S., AND DURAND, F. 2009. QuickSelect: History-based selection expansion. In *Proceedings of Graphics Interface*, 215–221.

SZELISKI, R., AVIDAN, S., AND ANANDAN, P. 2000. Layer extraction from multiple images containing reflections and transparency. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, 246–253.

TANGE, O. 2011. GNU Parallel: The command-line power tool. *;login: The USENIX Magazine 36*, 1 (Feb), 42–47.

TAYLOR, H. A., AND TVERSKY, B. 1992. Descriptions and depictions of environments. *Memory & Cognition 20*, 5, 483–496.

TOMASI, C., AND MANDUCHI, R. 1998. Bilateral filtering for gray and color images. In *Proceedings of IEEE International Conference on Computer Vision*, 839–846.

TVERSKY, B. 1999. What does drawing reveal about thinking? *Visual and Spatial Reasoning in Design*, 93–101.

VAN SOMMERS, P. 1984. *Drawing and cognition: Descriptive and experimental studies of graphic production processes.* Cambridge University Press.

VISTRAILS, I., 2009. Vistrails provenance explorer for maya. `http://www.vistrails.com/maya.html`.

XING, J., CHEN, H.-T., AND WEI, L.-Y. 2014. Autocomplete painting repetitions. *ACM Transactions on Graphics 33*, 6, 172.

XU, S., XU, Y., KANG, S. B., SALESIN, D. H., PAN, Y., AND SHUM, H.-Y. 2006. Animating chinese paintings through stroke-based decomposition. *ACM Transactions on Graphics 25*, 2, 239–267.

XU, L., LU, C., XU, Y., AND JIA, J. 2011. Image smoothing via L0 gradient minimization. *ACM Transactions on Graphics 30*, 6, 174.

ZIVKOVIC, Z., AND VAN DER HEIJDEN, F. 2006. Efficient adaptive density estimation per image pixel for the task of background subtraction. *Pattern Recognition Letters 27*, 7, 773–780.

ZONGKER, D. E., WERNER, D. M., CURLESS, B., AND SALESIN, D. H. 1999. Environment matting and compositing. In *ACM SIGGRAPH Conference Proceedings*, 205–214.

Chapter 4

Seamless Reconstruction of Part-Based High-Relief Models from Hand-Drawn Images

The paper included in this chapter was presented at the Expressive 2018 conference and has been published in the conference's proceedings:

Dvorožňák, M., Sepehri Nejad, S., Jamriška, O., Jacobson, A., Kavan, L., and Sýkora, D. [2018]. "Seamless Reconstruction of Part-Based High-Relief Models from Hand-Drawn Images". In *Expressive '18: The Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*

Seamless Reconstruction of Part-Based High-Relief Models from Hand-Drawn Images

Marek Dvorožňák
Czech Technical University in Prague,
Faculty of Electrical Engineering
dvoromar@fel.cvut.cz

Saman Sepehri Nejad
University of Utah
samansepehri@gmail.com

Ondřej Jamriška
Czech Technical University in Prague,
Faculty of Electrical Engineering
jamriond@fel.cvut.cz

Alec Jacobson
University of Toronto
jacobson@cs.toronto.edu

Ladislav Kavan
University of Utah
ladislav.kavan@gmail.com

Daniel Sýkora
Czech Technical University in Prague,
Faculty of Electrical Engineering
sykorad@fel.cvut.cz

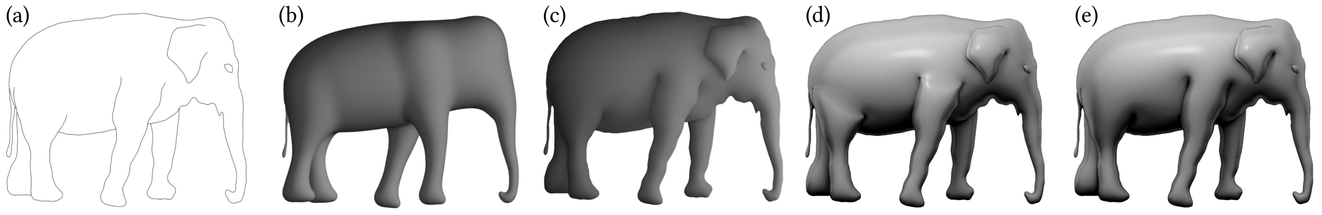


Figure 1: A comparison of our approach with the current state-of-the-art: the original input drawing (a); the result of Entem et al. [2015] (b) in contrast to the result of our technique (c) that produces a more natural transition between individual parts; the result of Sýkora et al. [2014] suffers from visible seams between individual parts (d) whereas our approach delivers smooth transition (e). (Images (a) and (b) come from [Entem et al. 2015].)

ABSTRACT

We present a new approach to reconstruction of *high-relief* surface models from hand-made drawings. Our method is tailored to an interactive modeling scenario where the input drawing can be separated into a set of semantically meaningful parts of which relative depth order is known beforehand. For this kind of input, our technique allows inflating individual components to have a semi-elliptical profile, positioning them to satisfy prescribed depth order, and providing their seamless interconnection. Compared to previous methods, our approach is the first that formulates this reconstruction process as a single non-linear optimization problem. Because its direct optimization is computationally challenging, we propose an approximate solution which delivers comparable results orders of magnitude faster enabling an interactive user workflow. We evaluate our approach on various hand-made drawings and demonstrate that it provides state-of-the-art quality in comparison with previous methods which require comparable user intervention.

CCS CONCEPTS

• **Computing methodologies → Mesh models**; **Reconstruction**; *Non-photorealistic rendering*;

KEYWORDS

sketch-based 3D modeling, high-relief models, hand-drawn images

ACM Reference Format:
Marek Dvorožňák, Saman Sepehri Nejad, Ondřej Jamriška, Alec Jacobson, Ladislav Kavan, and Daniel Sýkora. 2018. Seamless Reconstruction of Part-Based High-Relief Models from Hand-Drawn Images. In *Expressive '18: The Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering, August 17–19, 2018, Victoria, BC, Canada.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3229147.3229153

1 INTRODUCTION

Recent advances in interactive 3D modeling from a single image [Entem et al. 2015; Feng et al. 2016; Li et al. 2017; Sýkora et al. 2014; Xu et al. 2014; Yeh et al. 2017] make the creation of 3D models less demanding for artists and also more accessible for novice users who do not have sufficient experience with professional 3D modeling tools. Such tools need complex manipulation with geometric primitives in 3D space which requires working with multiple 2D projections. A key advantage of staying in the 2D domain is that it allows the user to remain entirely focused on the creative process and not be distracted by resolving consistency in depth. This important aspect also explains why 2D sketches are often used as concept art or in early stages of 3D model design.

In this paper, we focus on a branch of single image modeling methods that are suitable for organic structures composed of several rounded parts that are positioned in depth and attached or smoothly connected to each other [Entem et al. 2015; Feng et al. 2016; Sýkora et al. 2014; Yeh et al. 2017]. In the original image, these parts are usually delineated by outlines or have distinct boundaries that can be easily extracted. Thanks to the 2.5D layered structure, this kind of input usually requires only little user intervention while the resulting meshes still have a certain level of complexity which would be more difficult to achieve using standard 3D modeling tools. The desired result is akin to a *high-relief* sculpture, defined in classic sculpture as a relief where more than half of the shape projects from the background at full depth (see, e.g., [Read 1961]).

With previous methods, however, the reconstruction process is usually separated into a set of individual sub-problems which are solved sequentially, e.g., the input regions are first inflated and then shifted to preserve the relative depth. Finally, already inflated and shifted components are stitched together or smoothly interconnected. Due to this sequential process, the quality of the resulting mesh often suffers from the lack of smoothness in the areas where individual parts were stitched together. In this paper, we formulate a single optimization problem that unifies all of the above-mentioned sub-problems within a single energy minimization framework, delivering seamless organic shapes that seem like sculpted from a single block of material.

The contributions of our work are as follows: (1) We formulate the reconstruction of high-relief models from a single hand-drawn image as a minimization of a unified non-linear energy functional. Thanks to this joint formulation, our technique naturally produces meshes where the individual parts are interconnected smoothly and seamlessly. (2) We propose an efficient approximate method to our non-linear solution which enables interactive modeling workflow.

2 RELATED WORK

Igarashi et al. [1999] introduced the concept of modeling by inflation. They add volume to a 2D shape procedurally by triangulating the shape and setting vertex heights proportional to chordal axis distance. This concept was later extended by others using convolutional surfaces [Tai et al. 2004], sweeping 2D template scalar field [Schmidt et al. 2005], using mass-spring system [Karpenko and Hughes 2006], non-linear optimization [Nealen et al. 2007], generalized cylinders [Borosán et al. 2012; Zeng et al. 2015], surfaces of revolution [Bessmeltsev et al. 2015], level set method [Levi and Gotsman 2013], and finally by an implicit surface which is defined by a skeleton of the inflated region's shape and its radius function [Entem et al. 2015].

Other methods provide an extension that allows specification of cross-sectional functions for individual components [Olsen and Samavati 2010] or define a set of primitives that can be used to approximate their shape [Chen et al. 2013; Gingold et al. 2009; Shtof et al. 2013]. Sýkora et al. [2014] and Feng et al. [2016] obtain inflated shapes with semi-elliptical profiles by solving Poisson equation of the squared height (recovering the height by taking the square root). Yeh et al. [2017] and Jayaraman et al. [2018] utilize user-specified curvature annotations to infer gradient fields and obtain inflated shapes with parabolic profiles by solving Poisson equation

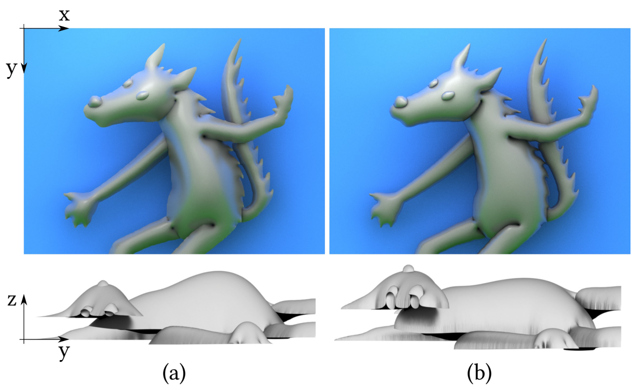


Figure 2: Comparison of inflation with a parabolic (a) and a semi-elliptical (b) profile for a frontal (top) and a side (bottom) view. Notice how the semi-elliptical inflation is steeper at boundaries and more evenly rounded which gives the frontal render (top right) a more natural appearance.

of the (unsquared) height (see Fig. 2). Li et al. [2017] use an iterative process guided by several types of user-provided curves.

As the inflation process is usually applied only to a single 2D region, the resulting 3D object has only limited structural complexity. To produce more complex 3D objects, individual components need to be inflated separately and then joined together. In this "piecewise" workflow, correct absolute depth values need to be specified in order to preserve the overall 3D structure and avoid potential penetration of the individual parts. A typical approach how to resolve this problem is to let the user to view the object from side-views and specify absolute depths manually [Borosán et al. 2012; Feng et al. 2016; Igarashi et al. 1999; Nealen et al. 2007]. Gingold et al. [2009] use manually constructed intersection curves to guide the positioning of parts in depth, and Bessmeltsev et al. [2015] delegate depth specification to an underlying 3D skeleton positioned by the user. Finally, Yeh et al. [2017] allows the user to specify a set of sparse slope cues which locally define surface increase in depth.

Other approaches try to infer the missing depth information using various cues and perform the positioning in an automatic fashion. Sýkora et al. [2010] use a set of sparse depth (in)equalities given by the user. Sýkora et al. [2014] utilize illusory surfaces to predict depth (in)equalities. Liu et al. [2013] use relative depth cues represented by T-junctions while Zeng et al. [2015] combines them also with ground contact cues. In [Yeh et al. 2015], angles at junctions and region overlaps are used as a layering metric while Shtof et al. [2013], as well as Chen et al. [2013], utilize geosemantic constraints as relative depth cues. Finally, Sýkora et al. [2014] use quadratic programming to automatically find smooth surfaces that translate and deform parts along axis perpendicular to the image so that the relative depth ordering is satisfied.

After determining the relative depth positioning of the individual components, the remaining task is to join the components together, ideally in a smooth and seamless way. Igarashi et al. [1999] lets the user specify a region for subsequent surface fairing by low-pass filtering [Taubin 1995]. In a follow-up work [Igarashi and Hughes

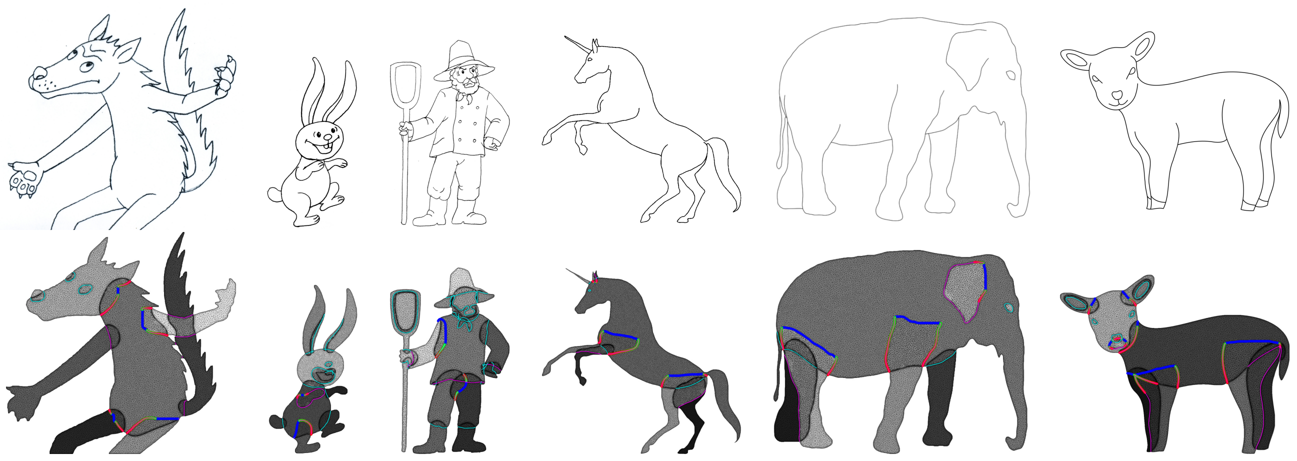


Figure 3: Original drawings (top) and the input to our method (bottom): pre-tesselated regions with completed occluded parts have their relative depth ordering visualized in grayscale (the lighter is the closer). Equality and inequality constraints are visualized using cyan and magenta, respectively, and interconnection lines using blue color. The Robin boundary constraints (green-to-red gradient) may be automatically determined by our system. (Original drawings: wolf, bunny, farmer © Anifilm; unicorn, elephant come from [Entem et al. 2015]; lamb © Marek Dvorožňák.)

2003], simple filtering is replaced by a non-linear method [Schneider and Kobbelt 2001]. A similar approach is used by Nealen et al. [2007], utilizing a fairing interpolation of surfaces defined by control curves. Borosán et al. [2012] use Laplacian smoothing around intersection loops, and Levi and Gotsman [2013] utilize a heuristic where a boolean union of spheres and reconstructed parts at joints is followed by bi-Laplacian smoothing. Blending of implicit primitives based on Ricci's operator [Ricci 1973] is used in Schmidt et al. [2005] as well as in Entem et al. [2015]. This approach allows control over the smoothness of each joint. To produce smooth joints, Sýkora et al. [2014] perform additional smoothing step by performing biharmonic interpolation in regions corresponding to connections of the individual components.

Despite the progress made in the above-mentioned work, the modeling process remains decoupled into separate steps. In this paper, we *unify* inflation, positioning, and seamless joining of individual components. Due to this unified formulation, hand-drawn images can be converted into high-quality meshes with minimal user intervention.

In this section, we have described methods that deal with the reconstruction of 3D shapes from sketches. The results are either full 3D models or some of their approximations, like high-relief models. There are also different techniques [Arpa et al. 2015; Cignoni et al. 1997; Schüller et al. 2014; Weyrich et al. 2007] that aim at bas- or high- relief generation out of a full 3D model. However, those deal with depth compression of the model as opposed to the model reconstruction.

3 OUR APPROACH

The aim of our approach is to reconstruct a high-relief model from a hand-drawn image. As an input to our method, we expect a set of semantically meaningful regions with completed occluded parts of which relative depth ordering is known. In addition to that, we assume (in)equality constraints for region boundaries and boundary vertices where two regions should merge (see Fig. 3). All this can be obtained using a semi-automatic process described in [Sýkora et al. 2014].

We would like the final high-relief model to satisfy the following requirements:

- regions should be inflated in a way so that the resulting shapes have semi-elliptical profiles,
- they should be shifted in depth to satisfy the prescribed relative depth ordering,
- interconnection of regions should be seamless, and
- the resulting model should closely match contours of the input 2D drawing when using orthographic projection.

Although the inflation with parabolic profile is solvable using linear system [Sýkora et al. 2014], non-linear semi-elliptical inflation is more desirable because it produces shapes that are steeper at boundaries and more evenly rounded. This is especially important for organic models such as cartoon characters (see Fig. 2 for comparison).

To restrict the high-relief model to stay within the boundaries given by the outlines of the original drawing, we restrict the inflation and shifting of individual parts to take place only in a direction that is perpendicular to the original image plane, i.e., we will optimize only z coordinates of the final 3D mesh as in [Sýkora et al. 2014].

In the rest of this section, we first show how to inflate a single region to have a desired semi-elliptical profile by formulating a non-linear inflation functional ($E_{inflation}$). Then, we consider joint inflation of multiple regions and satisfaction of their relative depth order. For this, we combine $E_{inflation}$ with a shifting functional (E_{shift}) which gives us the final non-linear energy E that expresses the whole reconstruction problem. Finally, we show how to linearize E and get an approximate solution which can be solved using a quadratic program. This enables substantial speedup while retaining similar quality as the original non-linear solution.

3.1 Non-linear Formulation

Inflation of a single region. An initial planar region Ω_i can be inflated to a semi-elliptical profile by finding a function $f(\mathbf{x}) : \Omega_i \to \mathbb{R}$ that minimizes the following energy functional

$$E^s_{inflation} = \int_{int(\Omega_i)} \left(\Delta f^2(\mathbf{x}) - c\right)^2 d\mathbf{x} \tag{1}$$

subject to

$$f(\mathbf{x}) = 0 \quad \forall \mathbf{x} \in \partial\Omega_i \tag{2}$$

where $int(\Omega_i)$ and $\partial\Omega_i$ is the interior and the boundary of Ω_i, respectively, Δ is the Laplacian operator and c is a scalar specifying a user-controllable amount of inflation. The energy is non-linear because the Laplacian is applied to the square of f and therefore, the result is not invariant to translation of the boundary conditions. To be able to move the parts in the z direction, which may be necessary to meet the relative depth order, we introduce a separate shifting function.

Simultaneous inflation and stitching of multiple regions. Given n initially planar regions Ω_i, we topologically interconnect those individual overlapping components at areas where seamless connection is desired (see Fig. 3). This enables us to achieve seamless transitions among them. Now, the task is to find an inflation and a shifting function $f(\mathbf{x}) : \Omega \to \mathbb{R}$ and $g(\mathbf{x}) : \Omega \to \mathbb{R}$, respectively, that minimize the following energy functional:

$$E = E_{inflation} + \lambda_{shift} E_{shift} \tag{3}$$

where

$$E_{inflation} = \int_{int(\Omega)} \left(\Delta f^2(\mathbf{x}) - c\right)^2 d\mathbf{x} + \lambda_{bnd} \int_B (f(\mathbf{x}))^2 \, d\mathbf{x}, \tag{4}$$

$$E_{shift} = \int_{\Omega} \|\nabla g(\mathbf{x})\|^2 d\mathbf{x}, \tag{5}$$

$\Omega = \Omega_1 \cup \ldots \cup \Omega_n$ is a unifying region that contains all the regions Ω_i and λ_{shift} is a regularization parameter controlling the balance between the inflation and the shifting of parts in the optimization.

As compared to the inflation of a single region where the boundary of the region is fixed at the plane $z = 0$ (Formula 1 and 2), the inflation energy $E_{inflation}$ is extended by a term (controlled by the parameter λ_{bnd}) that allows movement of the boundary of f on a subset $B \subseteq \partial\Omega$. This relaxation proves useful in alleviating unwanted reconstruction artifacts visually resembling *pits*. The influence of λ_{bnd} is visualized in Fig. 4 and also in our supplementary video. Fig. 9 (a) and (c) shows how the results with and without these artifacts look like when rendered.

The aim of the shifting energy E_{shift}, where ∇ stands for the gradient operator, is to find a function g that deforms the inflated shape f in a way that the result $h = f + g$ satisfies the relative depth conditions while encouraging g to be as flat as possible. The energy

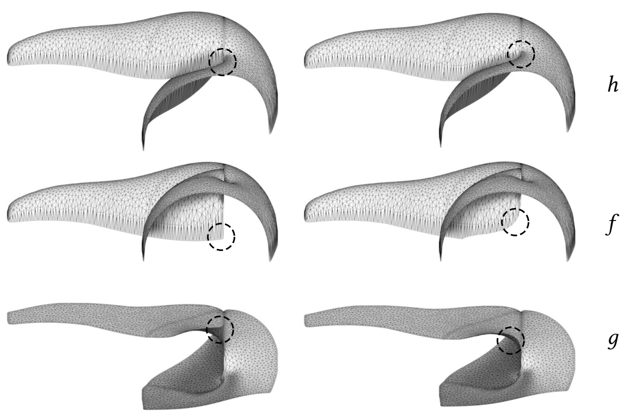


Figure 4: Converged results of the non-linear optimization with visualization of the inflation (f) and the shifting (g) function for the final result $h = f + g$, and also of the influence of the inflation boundary relaxation term λ_{bnd}. The result in the right column has its boundary more flexible ($\lambda_{bnd} = 0.01$) than the result in the left column ($\lambda_{bnd} = 100$) which mitigate formation of pits (depicted inside circles). See accompanying video for an animation.

E is minimized subject to the following relative depth conditions:

$$\begin{aligned} f_i(\mathbf{x}) + g_i(\mathbf{x}) &= f_j(\mathbf{x}) + g_j(\mathbf{x}) \quad \forall \mathbf{x} \in C^{=}_{i,j}, \\ f_i(\mathbf{x}) + g_i(\mathbf{x}) &\le f_j(\mathbf{x}) + g_j(\mathbf{x}) \quad \forall \mathbf{x} \in C^{\le}_{i,j}, \\ f_i(\mathbf{x}) + g_i(\mathbf{x}) &\ge f_j(\mathbf{x}) + g_j(\mathbf{x}) \quad \forall \mathbf{x} \in C^{\ge}_{i,j}, \end{aligned} \tag{6}$$

where $C^{=}_{i,j}, C^{\le}_{i,j}, C^{\ge}_{i,j} \subseteq \Omega_i \cap \partial\Omega_j$ are sets of points that specify relative depth order for two overlapping regions Ω_i and Ω_j. The resulting function h is then simply $h = f + g$.

3.2 Efficient Reformulation of Non-linear Energy

Although the non-linear solution that we have described fulfills our reconstruction requirements, depending on the mesh resolution and chosen numerical method, the convergence of the non-linear optimization may be relatively slow. In our implementation, the optimization often lasts hours for input with moderate complexity (see more details in Section 4). Even though more sophisticated numerical methods could offer higher performance, in this section we propose a different approach based on the observation that for a suitable λ_{shift}, when the optimization converges, the inflation energy $E_{inflation}$ is minimized and f is therefore completely inflated. To obtain a solution that is orders of magnitude faster while producing results that are comparable in quality, we build on this observation and separate the problem into the two subsequent steps: inflation and shifting.

As shown by Sýkora et al. [2014], the inflation with a semi-elliptical profile that corresponds to the Formula 1 can be obtained efficiently by solving the Poisson equation

$$\Delta\tilde{f}(\mathbf{x}) = \mathbf{c} \tag{7}$$

which produces a shape $\tilde{f}$ with a parabolic profile, and then the following cross-section function is used to change the shape of f to a semi-elliptical profile:

$$f(\mathbf{x}) = d\sqrt{\tilde{f}(\mathbf{x})} \tag{8}$$

where d is a scaling factor allowing to obtain flatter profile or reverse inflation.

To allow for greater modeling flexibility, we extend this inflation method to support transition between fixed and free boundaries via Robin boundary conditions for Equation 7:

$$\alpha(\mathbf{x})f(\mathbf{x}) + (1-\alpha(\mathbf{x}))\frac{\partial f}{\partial \mathbf{n}}(\mathbf{x}) = 0 \quad \forall \mathbf{x} \in \partial\Omega_i \tag{9}$$

where $\alpha(\mathbf{x}) \in [0,1]$ specifies the interpolation between Dirichlet and Neumann boundary constraints. This extension allows us to mimic the behavior of non-linear solution where the boundary can be shifted (Formula 4).

To obtain the final surface h that satisfies the specified relative depth order, we minimize:

$$\int_\Omega \|\nabla h(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 d\mathbf{x} \tag{10}$$

subject to (in)equality constraints representing the relative depth conditions:

$$\begin{aligned} h_i(\mathbf{x}) &= h_j(\mathbf{x}) \quad \forall \mathbf{x} \in C^{=}_{i,j}, \\ h_i(\mathbf{x}) &\le h_j(\mathbf{x}) \quad \forall \mathbf{x} \in C^{\le}_{i,j}, \\ h_i(\mathbf{x}) &\ge h_j(\mathbf{x}) \quad \forall \mathbf{x} \in C^{\ge}_{i,j}. \end{aligned} \tag{11}$$

This formulation is mathematically equivalent to the one used in [Sýkora et al. 2014]. However, it allows us to directly optimize for the final mesh h as opposed to the two-step procedure in [Sýkora et al. 2014].

3.3 Implementation Details

We discretize our continuous formulation using the finite element method. We assume that each planar region Ω_i is converted into a triangular mesh with additional interior vertices for boundary vertices of each region Ω_j that overlaps with Ω_i. This instantly gives us pairs of corresponding vertices that are used to satisfy relative depth ordering, i.e., the sets $C^{=}_{i,j}$, $C^{\le}_{i,j}$ and $C^{\ge}_{i,j}$. Please refer to [Sýkora et al. 2014] for more details about the procedure. Then, we reconnect the meshed regions at vertices where seamless transition is expected. These are visualized using blue lines in Fig. 3.

Non-linear formulation details. The inflation and shifting non-linear energies (Formula 4 and 5) are discretized as follows

$$\begin{aligned} E_{inflation} &\approx \left(\mathbf{M}_{in}^{-1}\mathbf{L}_{in}\mathbf{f}_{in}^2 - \mathbf{c}\right)^\mathsf{T} \mathbf{M}_{in} \left(\mathbf{M}_{in}^{-1}\mathbf{L}_{in}\mathbf{f}_{in}^2 - \mathbf{c}\right) \\ &\quad + \lambda_{bnd}\mathbf{f}_{bnd}^\mathsf{T}\mathbf{M}_{bnd}\mathbf{f}_{bnd}, \\ E_{shift} &\approx (\mathbf{Gg})^\mathsf{T}\,\mathbf{T}\,(\mathbf{Gg}), \end{aligned}$$

where $\mathbf{c}$ is a column vector of scalars c (see Formula 1), $\mathbf{T}$, $\mathbf{M}$ and $\mathbf{M}^{-1}$ are diagonal matrices representing areas of mesh triangles, the mass matrix and its inverse, $\mathbf{G}$ and $\mathbf{L}$ are sparse matrices representing discretization of the gradient operator and the usual cotangent discretization of the Laplacian operator [Meyer et al. 2003]. The square of a vector is understood as element-wise operation and the subscripts *in* and *bnd* denote a part of a matrix or a vector corresponding to interior and boundary vertices, respectively.

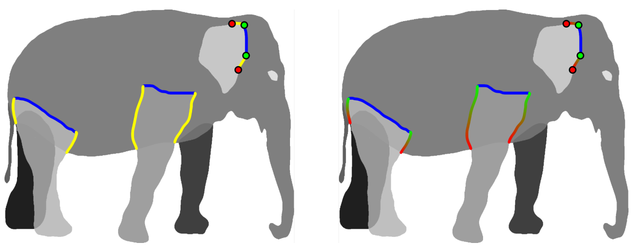


Figure 5: Example of boundary conditions used for mitigation of pits. Movable boundary utilized in our non-linear solution is visualized using yellow color (left) and Robin boundary conditions that we employ in our approximate solution using green-to-red gradient (right). User input is visualized using green and red points which specify the range of the specified conditions.

Efficient reformulation details. The Poisson equation for obtaining a parabolic inflation (Formula 7) is discretized as

$$\mathbf{L}\tilde{\mathbf{f}} = \mathbf{Mc}$$

and the minimization of functional in Formula 10 with (in)equalities (Formula 11) is discretized as a quadratic program, i.e., we minimize:

$$\frac{1}{2}\mathbf{h}^\mathsf{T}\mathbf{L}\mathbf{h} - \mathbf{h}^\mathsf{T}\mathbf{L}\mathbf{f} \tag{12}$$

subject to (in)equality constraints:

$$\begin{aligned} \mathbf{h}_i(p) &= \mathbf{h}_j(p) \quad \forall p \in C^{=}_{i,j}, \\ \mathbf{h}_i(p) &\le \mathbf{h}_j(p) \quad \forall p \in C^{\le}_{i,j}, \\ \mathbf{h}_i(p) &\ge \mathbf{h}_j(p) \quad \forall p \in C^{\ge}_{i,j}. \end{aligned} \tag{13}$$

4 RESULTS

We implemented both the original non-linear method and the linearized approximation in C++. Our implementation relies on the Eigen library [Guennebaud et al. 2010] for linear algebra routines, libigl [Jacobson et al. 2013] for discrete differential operators, solvers of quadratic problems and programs, and L-BFGS non-linear solver [Liu and Nocedal 1989]. To compute the gradient of the energy used in the non-linear solver, we used reverse-mode automatic differentiation from the Stan Math Library [Carpenter et al. 2015].

For all results included in this paper, we use the following parameters: $c = 4$ to obtain an inflation with hemispherical profile for circular regions and a semi-elliptical profile for regions with different shapes, $\lambda_{shift} = 1$ to equally balance inflation with shifting, $\lambda_{bnd} = 100$ for results that contain pits, and $\lambda_{bnd} = 0.01$ for results that mitigate pits.

According to our experiments, including the entire boundary of the function f into the subset B in Formula 4 may result in unwanted shifting of parts of the boundary that should stay fixed. This is caused by the competition between the individual terms in energy E (Formula 3). We resolve this by restricting B as follows: For each two overlapping regions that are interconnected and one is supposed to be above the other one, we include only boundary

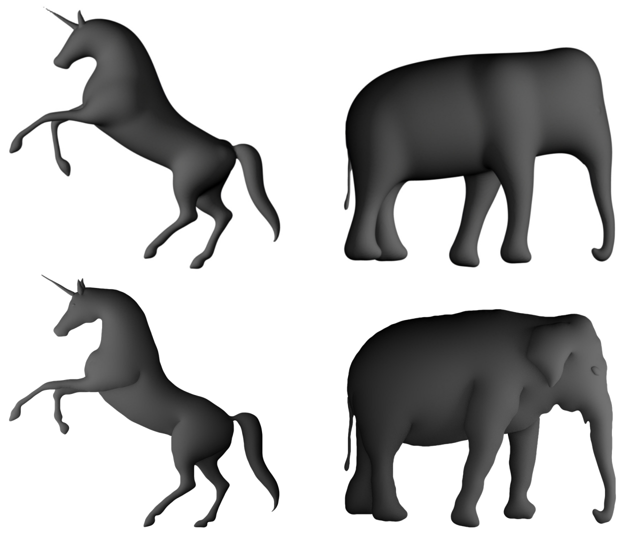


Figure 6: Comparison of results produced by Entem et al. [2015] (top) with our approach (bottom); for the unicorn and the elephant example. Our results more closely reproduce outlines of the original input drawings and contain seamlessly merged body parts without bulges. (Images located at the top come from [Entem et al. 2015].)

vertices of the top region that are overlapping with the bottom region into B (an example of the resulting subset B is depicted using yellow color in Fig. 5, left). For top regions that are entirely surrounded by the bottom components (e.g., elephant's ear), a user intervention may be needed to obtain satisfactory result. For these cases, we provide a simple two-click interface to specify the range manually.

We use the same technique for specification of Robin boundary conditions for the top region (Formula 9) in our approximate solution. We set $\alpha(\mathbf{x}) \in [0, 0.2]$ for boundary points that are adjacent to the interconnection line (blue line in Fig. 5), $\alpha(\mathbf{x}) = 1$ for points that are at the borderline of the bottom region and linearly interpolate α between the points. For all other boundary points, we assume $\alpha(\mathbf{x}) = 1$. For obtaining a reverse inflation (e.g., rabbit's ears or farmer's shovel in Fig. 7), the parameter d was set to -1, otherwise we used $d = 1$.

We ran our implementation on a quad-core CPU (Core i7, 2.7 GHz, 16 GB RAM). Summarized running times for our approximate solution are presented in Table 1.

Our implementation of non-linear solution often required hours to converge (see the supplementary material for a time-lapse video which illustrates convergence of the non-linear optimization). In contrast, the approximate version is significantly faster while the results look nearly identical (see Fig. 9 for a comparison).

We also compared the results produced by our method with the results produced by the two most closely related previous works: Ink-and-Ray [Sýkora et al. 2014] and a method by Entem et al. [2015].

The results of Ink-and-Ray with visible seams at connections of regions are shown in Fig. 7 and are even more pronounced when the lighting conditions changes, see Fig. 8 and the supplementary material. Those seams are caused by bi-Laplacian smoothing in a user-specified area around the connection that is performed in a post-processing grafting phase. Since we reconnect the overlapping regions before the reconstruction phase, our results naturally reproduce seamless connections that respect the specified inflation profile.

Table 1: Running times of our approximate solution.

model	# vertices	# faces	time
wolf	32 k	52 k	10.1 s
bunny	19 k	33 k	2.7 s
farmer	33 k	55 k	1.5 s
unicorn	22 k	36 k	3.8 s
elephant	39 k	68 k	12.4 s
lamb	30 k	50 k	7.1 s

The method of Entem et al. [2015] represents each part of a reconstructed 3D model by a skeleton-based convolution surface. These parts are then positioned in depth based on thickness of their 3D reconstruction and then smoothly blended together by simple summing operation. The quality of this blend strongly depends on the shape of completed regions as well as on their positions in depth. Due to this reason, unwanted bulges may appear in the final solution (see Fig. 6) as opposed to our method which guarantees to produce seamless connections between individual parts.

In addition to the rendered results presented in this paper, we include the resulting meshes for all models present in Fig. 7 and 9 in our supplementary material.

5 LIMITATIONS AND FUTURE WORK

Our method enables reconstruction of smooth high-relief models for a variety of different input drawings. However, we would like to point out some limitations of our approach which can serve as motivation for future work.

In some configurations where there are two regions smoothly interconnected and one is assumed to be above the other one, our solution may pull the top region down a bit and produce unwanted deformation of the lower region (see Fig. 10 for an example). As a future work, we plan to incorporate additional user-assisted compensation for such kind of configurations.

Our method assumes that the resulting model consists of a set of rounded shapes which have semi-elliptical profiles. Although this assumption is realistic for most organic shapes, there can still be situations which would require local modifications of the shape profile. For those, one may incorporate, e.g., the concept of curvature cues used in [Yeh et al. 2017] and modify the ∇f in Formula 10.

Although we provide a quick user-assisted mitigation of pits, an interesting avenue for future work would be an automatic approach such as determining suitable $\alpha(x)$ without user intervention. As future work, we plan to extend our technique from high-relief models to full 3D models by taking into account shape mirroring extension as used in [Feng et al. 2016].

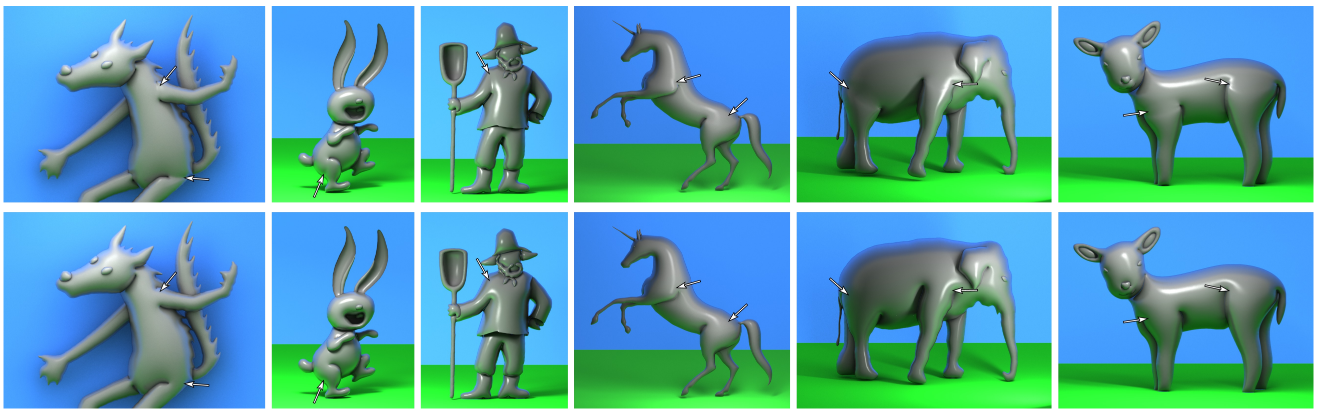


Figure 7: Comparison of results produced by Ink-and-Ray system [Sýkora et al. 2014] (top row) and the results of our method (bottom row). The differences in smoothness are pointed out with arrows. The reverse inflation of rabbit's ears and mouth, and farmer's shovel may be obtained by setting the parameter d to -1 for these parts.

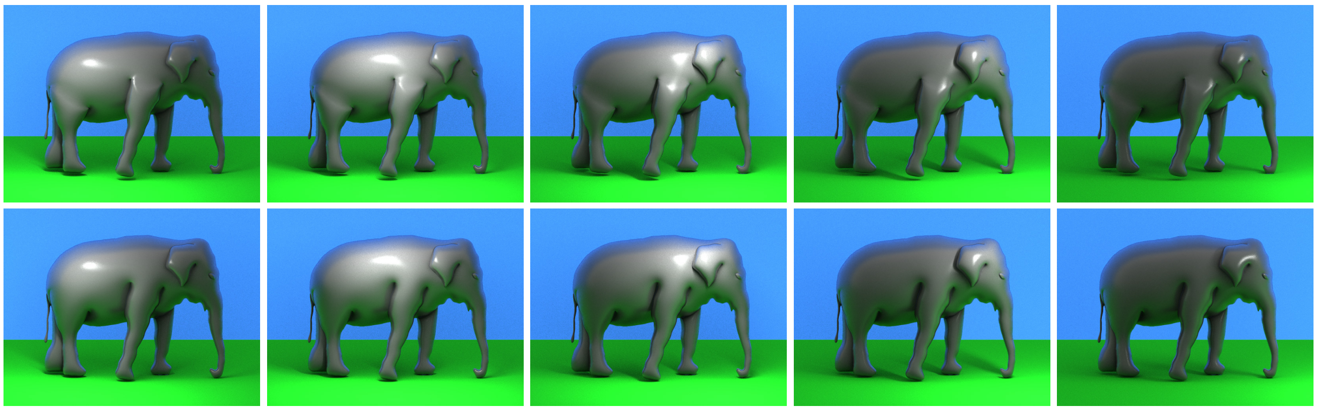


Figure 8: Comparison between a sequence of light variation on results produced by Ink-and-Ray method [Sýkora et al. 2014] (top) and our method (bottom). See accompanying video for an animation.

6 CONCLUSION

We presented a method to reconstruct high-relief part-based models from a single hand-drawn image. In contrast to previous techniques where the modeling process was subdivided into several independent steps, we proposed a unified non-linear energy minimization formulation which enables joint inflation and shifting of individual parts. In addition, we also proposed an efficient approximate method which delivers comparable solution as the original non-linear formulation but is notably faster. This enables us to create an interactive 3D modeling framework that enables production of high-quality meshes where individual parts are interconnected seamlessly. We confirmed the improvement in quality by comparing renderings of our resulting models and those obtained with the current state-of-the-art techniques.

ACKNOWLEDGMENTS

We would like to thank all anonymous reviewers for their fruitful comments and suggestions. We gratefully acknowledge the support of Activision, Adobe, and hardware donation from NVIDIA Corp. This research has been supported by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center), the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/237/OHK3/3T/13 (Research of Modern Computer Graphics Methods), Research Center for Informatics No. CZ.02.1.01/0.0/0.0/16_019/0000765, the Fulbright Commission in the Czech Republic, the NSERC Discovery Grants (RGPIN-2017-05235 & RGPAS-2017-507938), a Canada Research Chair award, the Connaught Fund, and the National Science Foundation under Grant Numbers IIS-1617172 and IIS-1622360. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

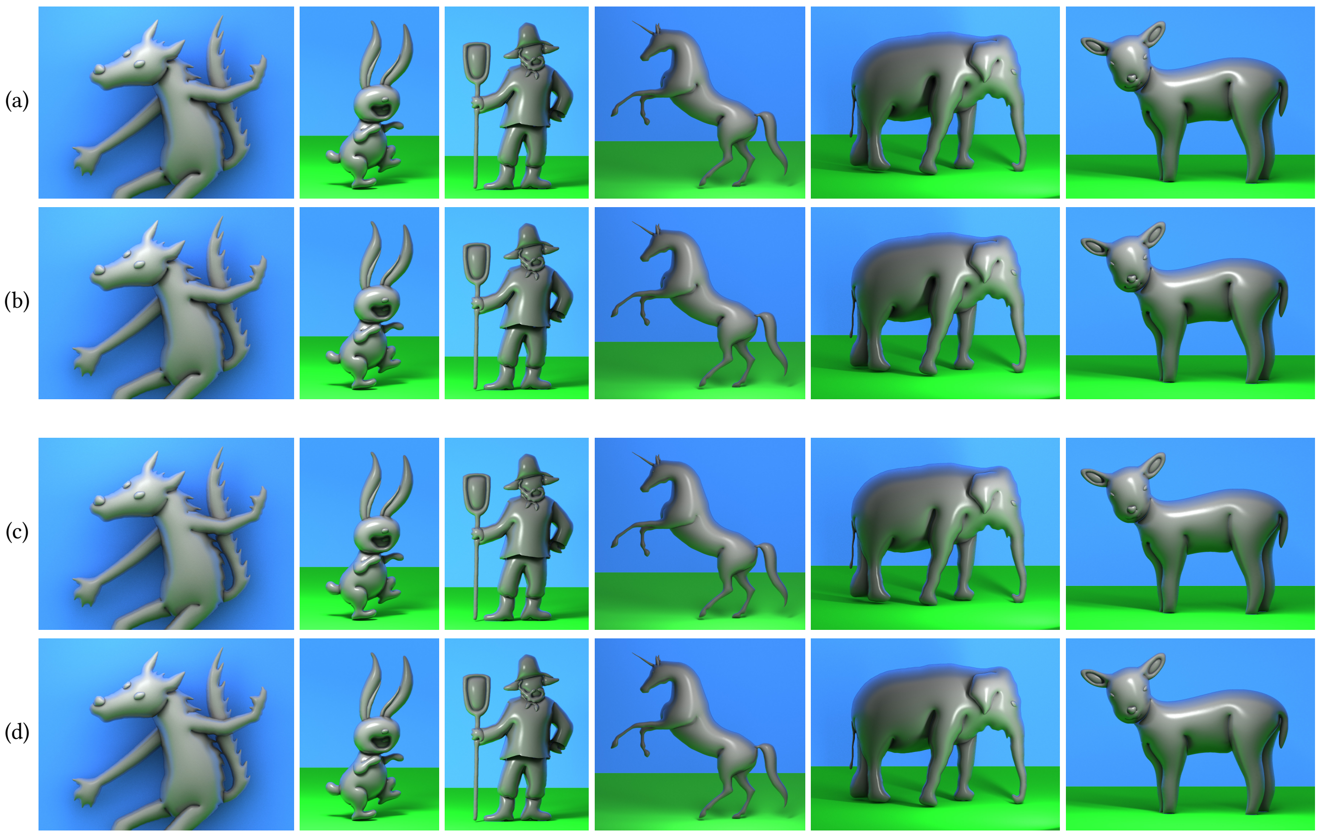


Figure 9: Comparison between the non-linear solution (a and c) and the approximate solution (b and d)—(a) results of the non-linear solution with pits, (b) corresponding approximations, (c) results of the non-linear solution with mitigated pits, (d) corresponding approximations (our final results).

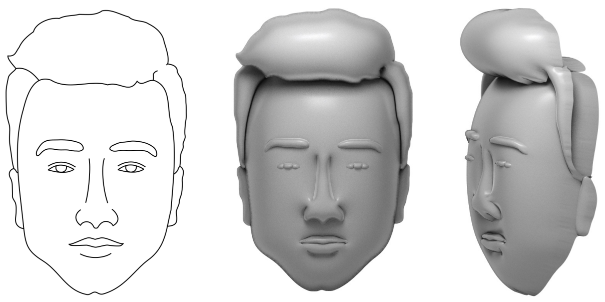


Figure 10: Limitation of our method: A top region (nose) which is interconnected with a bottom region (head) may pull the bottom region down a bit which may produce unwanted deformation. This deformation is more evident when rendered from a side view (right).

REFERENCES

Sami Arpa, Sabine Süsstrunk, and Roger D. Hersch. 2015. High Reliefs from 3D Scenes. *Computer Graphics Forum* 34, 2 (2015), 253–263.

Mikhail Bessmeltsev, Will Chang, Nicholas Vining, Alla Sheffer, and Karan Singh. 2015. Modeling Character Canvases from Cartoon Drawings. *ACM Transactions on Graphics* 34, 5 (2015), 162.

Peter Borosán, Ming Jin, Doug DeCarlo, Yotam Gingold, and Andrew Nealen. 2012. RigMesh: Automatic Rigging for Part-Based Shape Modeling and Deformation. *ACM Transactions on Graphics* 31, 6 (2012), 198.

Bob Carpenter, Matthew D. Hoffman, Marcus Brubaker, Daniel Lee, Peter Li, and Michael Betancourt. 2015. The Stan Math Library: Reverse-Mode Automatic Differentiation in C++. *CoRR* abs/1509.07164 (2015).

Tao Chen, Zhe Zhu, Ariel Shamir, Shi-Min Hu, and Daniel Cohen-Or. 2013. 3-Sweep: Extracting Editable Objects from a Single Photo. *ACM Transactions on Graphics* 32, 6 (2013), 195.

Paolo Cignoni, Claudio Montani, and Roberto Scopigno. 1997. Computer-Assisted Generation of Bas- and High-Reliefs. *Journal of Graphics Tools* 2, 3 (1997), 15–28.

Even Entem, Loïc Barthe, Marie-Paule Cani, Frederic Cordier, and Michiel van de Panne. 2015. Modeling 3D animals from a side-view sketch. *Computers & Graphics* 46 (2015), 221–230.

Lele Feng, Xubo Yang, Shuangjiu Xiao, and Fan Jiang. 2016. An Interactive 2D-to-3D Cartoon Modeling System. In *Proceedings of International Conference on Technologies for E-Learning and Digital Entertainment*. 193–204.

Yotam Gingold, Takeo Igarashi, and Denis Zorin. 2009. Structured Annotations for 2D-to-3D Modeling. *ACM Transactions on Graphics* 28, 5 (2009), 148.

Gaël Guennebaud, Benoît Jacob, et al. 2010. Eigen v3. http://eigen.tuxfamily.org. (2010).

Takeo Igarashi and John F. Hughes. 2003. Smooth Meshes for Sketch-based Freeform Modeling. In *Proceedings of Symposium on Interactive 3D Graphics*. 139–142.

Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. 1999. Teddy: A Sketching Interface for 3D Freeform Design. In *SIGGRAPH Conference Proceedings*. 409–416.
Alec Jacobson, Daniele Panozzo, et al. 2013. libigl: A simple C++ geometry processing library. (2013).
Pradeep Kumar Jayaraman, Chi-Wing Fu, Jianmin Zheng, Xueting Liu, and Tien-Tsin Wong. 2018. Globally Consistent Wrinkle-Aware Shading of Line Drawings. *IEEE Transactions on Visualization and Computer Graphics* (2018).
Olga A. Karpenko and John F. Hughes. 2006. SmoothSketch: 3D free-form shapes from complex sketches. *ACM Transactions on Graphics* 25, 3 (2006), 589–598.
Zohar Levi and Craig Gotsman. 2013. ArtiSketch: A System for Articulated Sketch Modeling. *Computer Graphics Forum* 32, 2pt2 (2013), 235–244.
Chang-Jian Li, Hao Pan, Yang Liu, Xin Tong, Alla Sheffer, and Wenping Wang. 2017. BendSketch: Modeling Freeform Surfaces Through 2D Sketching. *ACM Transactions on Graphics* 36, 4 (2017), 125.
D. C. Liu and J. Nocedal. 1989. On the Limited Memory BFGS Method for Large Scale Optimization. *Mathematical Programming* 45, 3 (1989), 503–528.
Xueting Liu, Xiangyu Mao, Xuan Yang, Linling Zhang, and Tien-Tsin Wong. 2013. Stereoscopizing Cel Animations. *ACM Transactions on Graphics* 32, 6 (2013), 223.
Mark Meyer, Mathieu Desbrun, Peter Schröder, and Alan H. Barr. 2003. Discrete Differential-Geometry Operators for Triangulated 2-Manifolds. In *Visualization and Mathematics III*, Hans-Christian Hege and Konrad Polthier (Eds.). 35–57.
Andrew Nealen, Takeo Igarashi, Olga Sorkine, and Marc Alexa. 2007. FiberMesh: Designing Freeform Surfaces with 3D Curves. *ACM Transactions on Graphics* 26, 3 (2007), 41.
Luke Olsen and Faramarz F. Samavati. 2010. Image-assisted Modeling from Sketches. In *Proceedings of Graphics Interface*. 225–232.
H Read. 1961. *The Art of Sculpture.*
A. Ricci. 1973. A constructive geometry for computer graphics. *Comput. J.* 16, 2 (1973), 157–160.
R. Schmidt, B. Wyvill, M. C. Sousa, and J. A. Jorge. 2005. ShapeShop: Sketch-based Solid Modeling with BlobTrees. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*. 53–62.
Robert Schneider and Leif Kobbelt. 2001. Geometric fairing of irregular meshes for free-form surface design. *Computer Aided Geometric Design* 18, 4 (2001), 359–379.
Christian Schüller, Daniele Panozzo, and Olga Sorkine-Hornung. 2014. Appearance-Mimicking Surfaces. *ACM Transactions on Graphics* 33, 6 (2014), 216:1–216:10.
Alex Shtof, Alexander Agathos, Yotam Gingold, Ariel Shamir, and Daniel Cohen-Or. 2013. Geosemantic Snapping for Sketch-Based Modeling. *Computer Graphics Forum* 32, 2 (2013), 245–253.
Daniel Sýkora, Ladislav Kavan, Martin Čadík, Ondřej Jamriška, Alec Jacobson, Brian Whited, Maryann Simmons, and Olga Sorkine-Hornung. 2014. Ink-and-Ray: Bas-Relief Meshes for Adding Global Illumination Effects to Hand-Drawn Characters. *ACM Transactions on Graphics* 33, 2 (2014), 16.
Daniel Sýkora, David Sedláček, Sun Jinchao, John Dingliana, and Steven Collins. 2010. Adding Depth to Cartoons Using Sparse Depth (In)equalities. *Computer Graphics Forum* 29, 2 (2010), 615–623.
Chiew-Lan Tai, Hongxin Zhang, and Jacky Chun-Kin Fong. 2004. Prototype modeling from sketched silhouettes based on convolution surfaces. *Computer Graphics Forum* 23, 1 (2004), 71–83.
Gabriel Taubin. 1995. A Signal Processing Approach to Fair Surface Design. In *SIGGRAPH Conference Proceedings*. 351–358.
Tim Weyrich, Jia Deng, Connelly Barnes, Szymon Rusinkiewicz, and Adam Finkelstein. 2007. Digital Bas-Relief from 3D Scenes. *ACM Transactions on Graphics* 26, 3 (2007), 32.
Baoxuan Xu, William Chang, Alla Sheffer, Adrien Bousseau, James McCrae, and Karan Singh. 2014. True2Form: 3D Curve Networks from 2D Sketches via Selective Regularization. *ACM Transactions on Graphics* 33, 4 (2014), 131.
Chih-Kuo Yeh, Shi-Yang Huang, Pradeep Kumar Jayaraman, Chi-Wing Fu, and Tong-Yee Lee. 2017. Interactive High-Relief Reconstruction for Organic and Double-Sided Objects from a Photo. *IEEE Transactions on Visualization and Computer Graphics* 23, 7 (2017), 1796–1808.
Chih-Kuo Yeh, Pradeep Kumar Jayaraman, Xiaopei Liu, Chi-Wing Fu, and Tong-Yee Lee. 2015. 2.5D Cartoon Hair Modeling and Manipulation. *IEEE Transactions on Visualization and Computer Graphics* 21, 3 (2015), 304–314.
Qiong Zeng, Wenzheng Chen, Huan Wang, Changhe Tu, Daniel Cohen-Or, Dani Lischinski, and Baoquan Chen. 2015. Hallucinating Stereoscopy from a Single Image. *Computer Graphics Forum* 34, 2 (2015), 1–12.

Chapter 5

Example-Based Expressive Animation of 2D Rigid Bodies

The paper included in this chapter was presented at the SIGGRAPH 2017 conference and has been published in ACM Transactions on Graphics journal:

Dvorožňák, M., Bénard, P., Barla, P., Wang, O., and Sýkora, D. [2017]. “Example-Based Expressive Animation of 2D Rigid Bodies”. *ACM Transactions on Graphics* 36.4

Example-Based Expressive Animation of 2D Rigid Bodies

MAREK DVOROŽŇÁK, Czech Technical University in Prague, Faculty of Electrical Engineering
PIERRE BÉNARD, LaBRI (UMR 5800, CNRS, Univ. Bordeaux), France
PASCAL BARLA, Inria Bordeaux Sud-Ouest, France
OLIVER WANG, Adobe Research
DANIEL SÝKORA, Czech Technical University in Prague, Faculty of Electrical Engineering

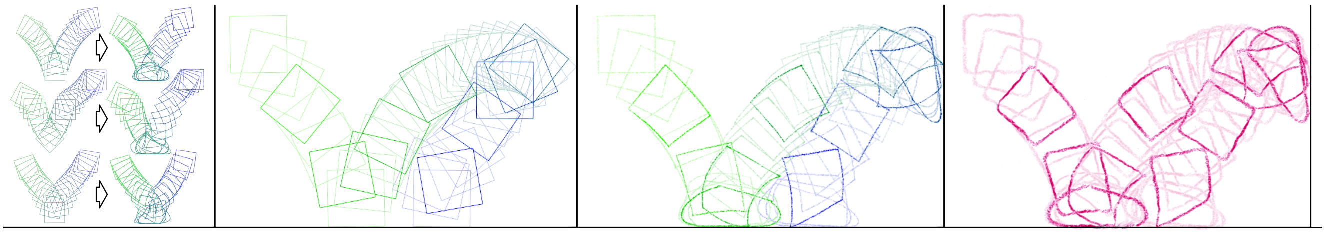


Fig. 1. Given a small set of exemplars consisting of computer-generated and hand-drawn 2D animation pairs **(a)**, our method transfers to a new target sequence produced by physical simulation **(b)** both the high-level deformations and fine-scale appearance variations **(c)** present in the example animations. Optionally, the final appearance of the drawings can be modified by re-synthesizing different stroke textures **(d)**.

We present a novel approach to facilitate the creation of stylized 2D rigid body animations. Our approach can handle multiple rigid objects following complex physically-simulated trajectories with collisions, while retaining a unique artistic style directly specified by the user. Starting with an existing target animation (e.g., produced by a physical simulation engine) an artist interactively draws over a sparse set of frames, and the desired appearance and motion stylization is automatically propagated to the rest of the sequence. The stylization process may also be performed in an off-line batch process from a small set of drawn sequences. To achieve these goals, we combine parametric deformation synthesis that generalizes and reuses hand-drawn exemplars, with non-parametric techniques that enhance the hand-drawn appearance of the synthesized sequence. We demonstrate the potential of our method on various complex rigid body animations which are created with an expressive hand-drawn look using notably less manual interventions as compared to traditional techniques.

CCS Concepts: • **Computing methodologies** → *Motion processing*; *Non-photorealistic rendering*;

Additional Key Words and Phrases: 2D animation, example-based synthesis

ACM Reference format:
Marek Dvorožňák, Pierre Bénard, Pascal Barla, Oliver Wang, and Daniel Sýkora. 2017. Example-Based Expressive Animation of 2D Rigid Bodies. *ACM Trans. Graph.* 36, 4, Article 127 (July 2017), 10 pages.
DOI: http://dx.doi.org/10.1145/3072959.3073611

1 INTRODUCTION

Despite the recent success of computer-generated animations, traditional hand-drawn approaches often yield more expressive and stylized looks than those produced with the currently available digital tools. However, creating hand-drawn animations is a tedious process that requires years of training by an artist and countless hours of labor. Furthermore, style is a highly personalized concept, and two different artists never animate exactly in the same way. As a result, example-based stylization has been a long-standing goal in computer graphics.

In this work we focus on rigid bodies, which are particularly challenging to animate by hand, since multiple objects may collide and rebound in ways that are difficult to plan in advance. Conversely, using physical simulation, computer-based methods can quickly give rise to rigid body animations with realistic trajectories, but ones that lack *expressiveness*. Our main goal is therefore to combine the ease of use of computer-simulated 2D rigid body animations with the expressive qualities of hand-drawn techniques.

To accomplish this goal, we have a number of added requirements. First, editability is of paramount importance to animators, and an ideal solution should work iteratively, always providing the artist with the ability to refine the current solution. Second, producing each hand-drawn frame is time consuming, so a practical example-based 2D animation system should be able to generalize from a *very* limited set of artistic inputs, while being able to apply these edits seamlessly into the dense set of final target frames. These two requirements make example-based 2D animation out of reach of current data-driven machine learning techniques, due to the scarcity of data (tens of exemplars, rather than tens of thousands), and uniqueness of each style.

Instead, our approach is inspired by a workflow that is widespread among both traditional and digital animators. A 2D animation is successively produced and refined at three different temporal scales (see Figure 3): the full animation scale at which timing, contacts and trajectories are planed; the pose-to-pose scale, at which the overall dynamics and deformations between contacts are considered; and the frame-to-frame scale at which the actual drawings with secondary deformations and precise collisions are produced.

At the full animation scale, we split the input computer-generated sequences based on collision events, and we independently analyze and stylize each sub-sequence around a hit point, which we call a key pose. Next, for every sub-sequence, we estimate the spatial deformations of the hand-drawn exemplars before and after the key pose; these are then transferred using a parametric synthesis algorithm. Correspondences between sub-sequences are estimated by leveraging the physical properties of each frame, which ensures preservation of appropriate stylistic effects given the forces applied at each frame. The final frame-by-frame drawings are then synthesized from the artist drawings using a non-parametric technique, that captures residual deformations and appearance details.

We show that this organization is necessary to capture both the long- and short-range stylistic choices made by the artist, while producing results that have the desired hand-drawn look and feel. When taken as a whole, our method considerably reduces the amount of work needed to create the entire sequence by hand.

In summary, we present the following contributions:

- a careful analysis of traditional hand-drawn animations, especially focusing on the correlation between physical parameters and deformations,
- a parametric motion synthesis algorithm capable of transferring deformations from exemplars,
- an example-based non-parametric stylization technique capturing the fine-scale drawing appearance.

2 PREVIOUS WORK

We present prior work related to general stylized computer animation, followed by those driven by physical simulations, and finally example-based solutions.

Techniques inspired by the principles of animation. From its beginning, one of the goals of computer graphics has been to reproduce the expressiveness of traditional hand-drawn 2D animations, while reducing the cost and effort of producing it. The fundamental principles of animation, developed from the late 1920's to the 1930's at Walt Disney Studio [Thomas and Johnston 1981], play a crucial role in this expressiveness, and many works have tried to adapt them to digital animation tools. Lasseter [1987] describes these 2D principles – including squash and stretch, timing and spacing, anticipation and follow-through, arc trajectories and lines of action – and how they can be manually applied by an artist to produce expressive 3D keyframe animations.

Subsequent work has aimed at fully or partially automatizing those effects. Wang et al [2006] describe a simple temporal filter that produces anticipation and follow-through as well as squash-and-stretch deformations by delaying parts of an existing animation, represented by 2D polygonal shapes or motion captured (MoCap) data. Lee et al. [2012] obtain similar effects on segmented objects in a video by relating a set of representative 2D deformations to the modal analysis of the object motion. Focusing on 3D skeletal animation, Noble and Tang [2006] present a tool to automatically bend the limbs of a character following lines of actions or arcs. On the artistically controllable side, Li et al. [2003] present a sketching interface that allows a user to guide the deformation of both the input animated skeleton and surface mesh to improve the expressiveness of MoCap data. Recently, several 2D sketching systems [Kazi et al. 2014a,b, 2016; Xing et al. 2016] have been developed to simplify the production of dynamic illustrations, reproducing most principles of character or special effects animation. However these principles are essentially encoded as scripted deformations or animated loops triggered by events, and unlike our work are not tailored to the specific style of a given artist.

Physics-driven approaches. Physical simulation is a convenient way to automatically animate a large number of 2D or 3D bodies, but expressiveness of the resulting motion is restricted by the degree of complexity modeled by the physical system. For instance, it is common to restrict the simulation to rigid bodies for computational efficiency, while traditional hand-drawn animated objects more often resemble deformable bodies governed by exaggerated physical laws. To enhance basic 3D simulations, multiple works [Chenney et al. 2002; Garcia et al. 2007; Haller et al. 2004] derive automatic procedural rules to generate squash-and-stretch and temporal effects based on motion parameters (velocity, acceleration, etc.) but such methods have limited art-directability.

To allow artistic control, the spacetime [Witkin and Kass 1988] and dynamic [Barzel and Barr 1988] constraint formulations cast physical simulation as a constrained optimization problem. Through those constraints, the artist can direct the simulation to act as a physically-plausible interpolation mechanism between key-poses. Bai et al. [2016] leverage this idea to build a 2D animation system that combines keyframing of local deformations with physical simulation for powerful inbetweening. Although this approach allows an unprecedented level of artistic control and manages to reproduce many of the principles of animation, it requires the user to specify control handles, which are constrained and unnatural to use when compared to simply drawing frames, in particular when the artist desires a precise control over shape outlines.

Example-based methods. This family of techniques provides a natural and intuitive interface, where examples are used to capture the style and intent of an artist. Such approaches have already produced impressive results for static images and videos, either using non-parametric texture synthesis [Bénard et al. 2013; Fišer et al. 2016; Hertzmann et al. 2001; Lu et al. 2012] or more recently with neural networks [Gatys et al. 2016]. Yet these methods are mostly restricted to appearance stylization, leaving motion largely untouched.

There are some exceptions, such as Bregler et al. [2002] who propose to capture and re-target motion from existing cartoon animations by combining a global affine deformation with drawings interpolation using a key-shape model. Jones et al. [2015] follows a similar approach, connecting the navigation in a simplicial complex [Ngo et al. 2000] with events of a 2D physical simulation. Pose-space interpolation can produce impressive results, but the quality

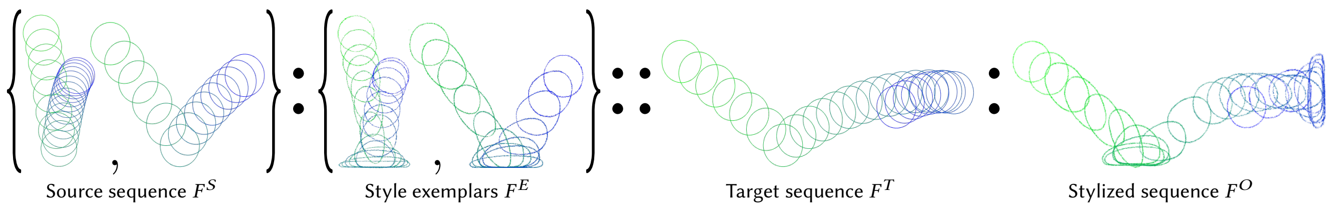


Fig. 2. Stylization analogy setup — given a set of frames F^S coming from reference 2D rigid body source animations, corresponding hand-animated exemplars F^E, and a new target animation F^T, the synthesis algorithm relates physical parameters in F^S and F^T to produce the output stylized sequence F^O that resembles F^E. (Animations are depicted with onionskins, colored from green to blue according to frame numbers.)

of the output is highly dependent on a good choice of the key-shapes which an artist has to select and organize manually *beforehand*.

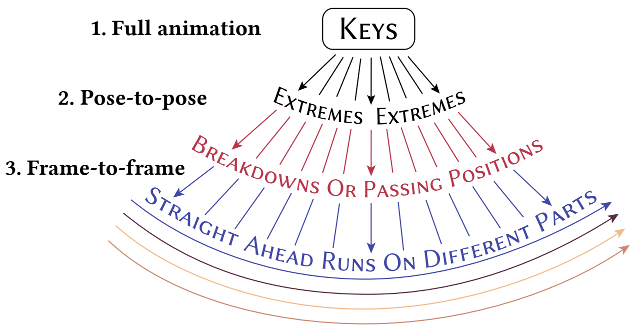
To guide or enrich 3D simulations, example-based approaches augment the simulated objects with examples of desirable deformations [Coros et al. 2012; Jones et al. 2016; Koyama et al. 2012; Martin et al. 2011]. In those approaches, however, exact correspondences between deformation exemplars are known beforehand and only a simple parametric deformation with limited number of degrees of freedom is used. Even though this setting may be natural for digital 3D artists, it is again limited and constraining for traditional 2D animators.

Closest to the traditional animation pipeline, Xing et al. [2015] present an interactive system for computer-assisted 2D animation. It combines discrete texture synthesis with an as-rigid-as-possible deformation model to predict and interpolate drawings based on the current and previous frames. This approach is convincing for frame-by-frame animation, but the spatio-temporal locality of the analysis makes it unsuited for pose-to-pose planning. Since the interpolations are solely based on the past drawings using local affine transformations, the predicted motion and deformations tend to be unnatural and cannot easily be edited, unless the artist draws most intermediate frames.

3 OVERVIEW

Similar in spirit to Image Analogies [Hertzmann et al. 2001], our algorithm transforms a *target* 2D rigid body animation F^T into an *output* stylized version F^O using an example-based transformation defined by a set of *source* sequences F^S and a corresponding set of hand-drawn *exemplars* F^E (Fig. 2). Sequences F^S and F^T can be computer-generated using, e.g., physical simulation. The style exemplars F^E are created by an artist digitally or physically, by redrawing a small subset of the source frames F^S. In one application, the untouched frames of F^S can be added to F^T, in which case our method can be seen as an interactive *style propagation* tool. This is shown in the accompanying video where the artist first draws over a few frames, sees the intermediate result, identifies parts which have not been successfully stylized, provides additional examples, and iterates this procedure until she is satisfied by the stylized result.

The key challenge here comes from the fact that F^T will typically not contain sub-sequences exactly like those in F^S, and thus stylized frames from F^E cannot simply replace original rigid frames in F^T. To tackle this problem, we take inspiration from guidelines in traditional 2D animation books [Thomas and Johnston 1981; Williams



1. Full animation
2. Pose-to-pose
3. Frame-to-frame
Keys
Extremes
Extremes
Breakdowns Or Passing Positions
Straight Ahead Runs On Different Parts

Fig. 3. Three scales hierarchical decomposition of the animation process, based on [Williams 2001, p.67].

2001], especially from Richard Williams' hierarchical decomposition of the animation process (see Figure 3). We identify three main stages or temporal scales: (1) the full animation scale, at which timing and spacing are planned by choosing the key events, (2) the pose-to-pose stage, at which the main dynamics and deformations are defined between two key poses by drawing "extremes" and "breakdowns", and (3) the frame-to-frame scale, corresponding to final straight-ahead "runs" (or drawing passes) during which subtle variations and details are added.

Each of the three stages need to be analyzed for transferring the style of a hand-drawn animation and our method thus follows this organization. First, timing and spacing are specified by the input sequences F^S and all animations are subdivided into overlapping *sub-sequences* around key events (Section 4). The style pairs $F^S : F^E$ are then decomposed into a coarse geometric deformation $\mathcal{D}$ and a fine-scale "residual" stylization $\mathcal{R}$ (Section 5.1). Our aim is to transfer both $\mathcal{D}$ and $\mathcal{R}$ to the target sequence F^T. For each target sub-sequence independently, a new parametric deformation is synthesized by selecting and blending together multiple deformations $\mathcal{D}$ coming from similar sub-sequences of the style pairs (Section 5.2). Finally, sub-sequences are blended together, the fine-scale details are reintroduced on a frame-by-frame basis by morphing the residual changes $\mathcal{R}$, and the appearance of individual strokes is changed to have a desired look of particular artistic media (Section 6). In the following, we use the classical "bouncing ball" animation to illustrate the various steps of our algorithm; results on more complex sequences (collisions between objects, bouncing square, textured objects) are shown in Section 7 and the supplemental material.

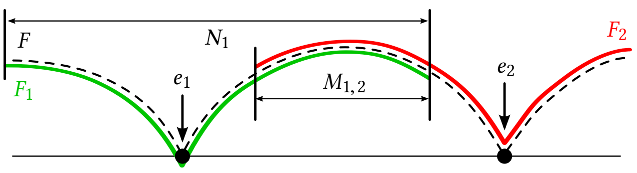


Fig. 4. Decomposition into sub-sequences — an input sequence F is subdivided into sub-sequences (F_i) of N_i frames around key events ($e_i \in \mathcal{E}$) with an overlap of $M_{i,i+1}$ frames with the following sub-sequence.

4 TIMING AND SPACING

The source F^S and target F^T input sequences consist in 2D rigid body animations produced using physical simulation. In practice, we use Box2D [Catto 2007]. Similar to prior work [Kazi et al. 2016], an arbitrary target rigid body is represented by its proxy geometry, e.g., bounding circle, square, or any other closed polygon. In addition to images, the simulation generates, at each frame and for each object, a set of physical parameters including the object velocity v and the rotation angle α around the object centroid with respect to the local trajectory frame. The timing and spacing are dictated by the simulation; the artist draws over existing frames, establishing one-to-one temporal correspondences between F^S and F^E.

The simulation also identifies frames at which semantically important events $\mathcal{E}$ occur, such as contact points or collisions. Following the guidelines of Richard Williams [2001], these frames represent key poses. Analyzing the hand-drawn exemplars F^E, we also observed that those frames, and their immediate neighbors in time, are the ones most stylized by the artist, whereas distant frames are less modified. In addition, we noticed that the physical parameters along the simulated trajectories before and after these key events largely influence the artist's stylization choices, e.g., the magnitude of the deformation.

These observations motivate us to subdivide F^S, F^T and F^E into a set of smaller overlapping sub-sequences F_i^S, F_i^T and F_i^E around every key event e_i of $\mathcal{E}$. Each sub-sequence F_i contains N_i consecutive animation frames and overlap with the next sub-sequence on $M_{i,i+1}$ frames. As shown in Figure 4, the overlapping part between two events resides at frames where there are no abrupt changes in physical parameters and moderate artistic stylization, making them most suitable for stitching.

5 POSE-CENTERED DEFORMATIONS

At this stage, we consider each sub-sequence independently, and focus on the coarse deformations used by artists when hand-animating rigid bodies to reproduce effects described in the principles of animation (squash-and-stretch, arc trajectories, lines of action). Residual deformations and appearance variation that are not captured by this coarse deformation will be reintroduced in Section 6.

5.1 Parametric deformation analysis

For each frame of a style pair $F_i^S : F_i^E$, we first estimate a coarse parametric deformation $\mathcal{D}$ (see Figure 5(a)) using the registration algorithm of Sýkora et al. [2009] which aligns bitmap images with an as-rigid-as-possible grid deformation. However, instead of the deformable grid matching, we use a single quadratic transformation as in Müller et al. [2005]. The main advantage of this quadratic model is that besides shear and stretch, it also captures twist and bending modes (see Figure 6) which better represent the larger scope of deformations used in traditional animation.

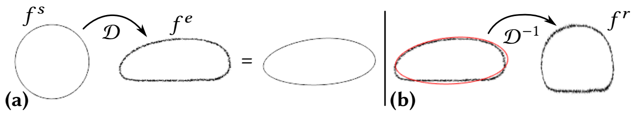


Fig. 5. Deformation analysis — **(a)** The parametric deformation $\mathcal{D}$ is estimated using as-rigid-as-possible registration between the source f^s and exemplar f^e frames. **(b)** The residual frame f^r is then computed by applying the inverse deformation $\mathcal{D}^{-1}$ on f^e.

The output of the image registration phase consists of 12 parameters describing the corresponding quadratic deformation that warps pixels $\mathbf{p} = (x, y)^\top$ from the source frame $f^s \in F_i^S$ to match pixels $\mathbf{p}' = (x', y')^\top$ of the stylized frame $f^e \in F_i^E$:

$$\begin{aligned} x' &= a_{11}x + a_{12}y + q_{11}x^2 + q_{12}y^2 + m_1 xy + t_1 \\ y' &= a_{21}x + a_{22}y + q_{21}x^2 + q_{22}y^2 + m_2 xy + t_2 \end{aligned} \tag{1}$$

Written in matrix form:

$$\mathbf{p}' = \underbrace{\begin{bmatrix} A & Q & \mathbf{m} & \mathbf{t} \end{bmatrix}}_{\mathcal{D}} \tilde{\mathbf{p}} \tag{2}$$

where $\tilde{\mathbf{p}} = (x, y, x^2, y^2, xy, 1)^\top$ is $\mathbf{p}$ expressed in extended homogeneous coordinates, and $\mathcal{D}$ is a quadratic transformation matrix composed of affine A, purely quadratic Q, mixed $\mathbf{m}$, and translation $\mathbf{t}$ parts:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad Q = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} \quad \mathbf{m} = \begin{pmatrix} m_1 \\ m_2 \end{pmatrix} \quad \mathbf{t} = \begin{pmatrix} t_1 \\ t_2 \end{pmatrix}.$$

5.2 Parametric deformation synthesis

Based on traditional hand-drawn animation resources as well as our own observations and discussions with 2D animators, we make the key hypothesis that deformation is closely tied to motion. As a result, to perform the deformation transfer, we search for correspondences between source F_i^S and target F_j^T sub-sequences using physical parameters that describe the frame's *motion* (velocity, trajectory orientation, and the object's rotation), and we assume that the matching sub-sequences should undergo similar deformations $\mathcal{D}$ as the source ones.

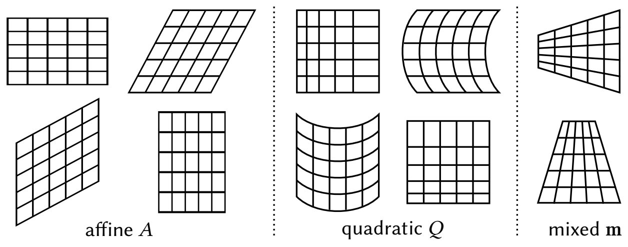


Fig. 6. Visualization of the 10 modes defined by the quadratic deformation of Müller et al. [2005].

Source-target sub-sequence matching. Practically, we define the following difference metric between a source sub-sequence F_i^S and a target sub-sequence F_j^T:

$$\begin{aligned}\mathrm{Diff}(F_i^S, F_j^T) = {} & \lambda_{\mathrm{vel}}\mathrm{Vel}(F_i^S, F_j^T) \\ & + \lambda_{\mathrm{dir}}\mathrm{Dir}(F_i^S, F_j^T) \\ & + \lambda_{\mathrm{rot}}\mathrm{Rot}(F_i^S, F_j^T),\end{aligned} \tag{3}$$

where weights λ_{vel}, λ_{dir}, λ_{rot} are used to balance the influence of individual terms:

- $\mathrm{Vel}(F_i^S, F_j^T)$ measures the difference between rigid body centroid velocities v:

$$\mathrm{Vel}(F_i^S, F_j^T) = \sum_{n=1}^{N} ||v_n(F_i^S) - v_n(F_j^T)||^2, \tag{4}$$

- $\mathrm{Dir}(F_i^S, F_j^T)$ penalizes discrepancy of the trajectory orientation δ:

$$\mathrm{Dir}(F_i^S, F_j^T) = \sum_{n=1}^{N} ||\delta_n(F_i^S) \ominus \delta_n(F_j^T)||^2, \tag{5}$$

  where $\ominus$ computes the smallest difference between two angles,
- $\mathrm{Rot}(F_i^S, F_j^T)$ accounts for differences in the rotation α of the rigid body around its centroid:

$$\mathrm{Rot}(F_i^S, F_j^T) = \sum_{n=1}^{N} ||\alpha_n(F_i^S) \ominus \alpha_n(F_j^T)||^2. \tag{6}$$

When computing the metric we assume that both sub-sequences are centered at a key event and have the same number of frames N. This can be done by resampling the original trajectories to have equidistant samples according to their arc length. The longest sub-sequence is trimmed to have the same length as the shortest one.

Deformation blending. Since it is unlikely that any source sub-sequence perfectly matches a given target sub-sequence F_j^T, we retrieve K nearest neighbor sub-sequences $F_1^S \dots F_K^S$ instead of a single one. For each frame in F_j^T, we then compute its stylized version as a combination of K quadratic transformations $\mathcal{D}_1 \dots \mathcal{D}_K$ from the K best corresponding frames in source sub-sequences using weights $w_1 \dots w_K$ proportional to their similarity:

$$w_k = \frac{1/\mathrm{Diff}(F_k^S, F_j^T)}{\sum_{\kappa=1}^{K} 1/\mathrm{Diff}(F_\kappa^S, F_j^T)}, \quad k \in [1 \dots K]$$

where normalization is used to obtain the partition of unity. See Figure 7 for an overview of this blending scheme (with $K = 3$) which adds robustness to the matching process and gives more stable results than simply using the single best match ($K = 1$).

To perform a meaningful interpolation of the rotational part of the transformation, the affine part A of the matrix $\mathcal{D}$ is factorized using polar decomposition [Higham and Schreiber 1990] into a linear stretch U (not used directly) and a rotation matrix R_α, from which the rotation angle $\alpha = \arctan(r_{11}/r_{21})$ is extracted. A weighted blend is computed on $\alpha_1 \dots \alpha_K$:

$$\hat{\alpha} = w_1\,\alpha_1 \oplus \dots \oplus w_K\,\alpha_K \tag{7}$$

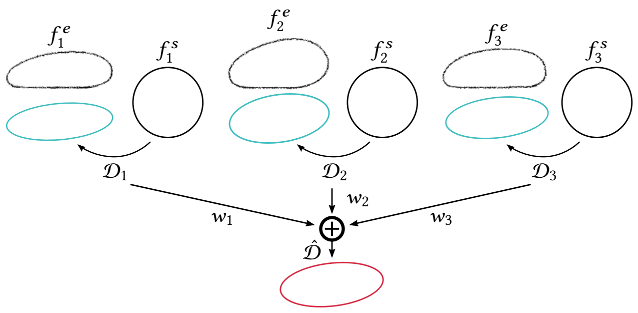


Fig. 7. Blending quadratic deformations $\mathcal{D}$ — individual quadratic deformations $\mathcal{D}_1$, $\mathcal{D}_2$, $\mathcal{D}_3$ estimated from the source frames f_1^s, f_2^s, f_3^s and their corresponding stylized counterparts f_1^e, f_2^e, f_3^e are blended together using the weights w_1, w_2, w_3 to obtain the resulting quadratic deformation $\hat{\mathcal{D}}$.

where $\oplus$ computes a weighted average of circular quantities. The remaining coefficients of rotation-free quadratic transformations $\mathcal{D}'_1 \cdots \mathcal{D}'_K$ are similarly computed:

$$\hat{\mathcal{D}}' = w_1\,\mathcal{D}'_1 + \dots + w_K\,\mathcal{D}'_K \tag{8}$$

where $\mathcal{D}' = R_{-\alpha}\mathcal{D}$. Finally, the blended quadratic transformation matrix $\hat{\mathcal{D}}$ is constructed from $\hat{\alpha}$ and $\hat{\mathcal{D}}'$:

$$\hat{\mathcal{D}} = R_{\hat{\alpha}}\hat{\mathcal{D}}' \tag{9}$$

Data augmentation. To generate plausible results even when the global orientation or scale of the target trajectory departs considerably from the available exemplars, we enrich the set of source analogies by scaling, rotating, and flipping the input sequences. We can directly extract the set of required rotation angles γ by analyzing the target simulation. Based on our experiments, we also use 5 scaling factors ρ between 0.2 to 1 and allow symmetries with respect to the vertical axis only (to preserve gravity effects). For rotationally symmetric objects of order n, we modify the $\ominus$ operator in a way that it outputs zero difference for angles $k\frac{360°}{n}$ where the apperance of the rotated object is the same. For the circle the order of rotational symmetry is infinite so instead we set $\lambda_{\mathrm{rot}} = 0$.

The drawback of the augmentation is that it may lead to incorrect stylizations when the source exemplars are far from the target motion. For example, a source exemplar corresponding to a small jump will not be equivalent to a source exemplar with a higher jump. To account for this, we dampen the resulting quadratic transformation $\hat{\mathcal{D}}$ by computing a weighted blend of $\hat{\mathcal{D}}$ with the identity matrix I using weight ξ, proportional to the ratio of the average source and target velocities:

$$\hat{\mathcal{D}} = \xi\hat{\mathcal{D}} + (1-\xi)I \quad \text{with} \quad \xi = \frac{\sum_{n=1}^{N} v_n(F_i^S)}{\sum_{n=1}^{N} v_n(F_j^T)} \tag{10}$$

However, if rotational invariance is not (even approximately) satisfied, orientation augmentation cannot be used, and the artist will need to prepare a set of additional exemplars corresponding to the correct rotational motion.

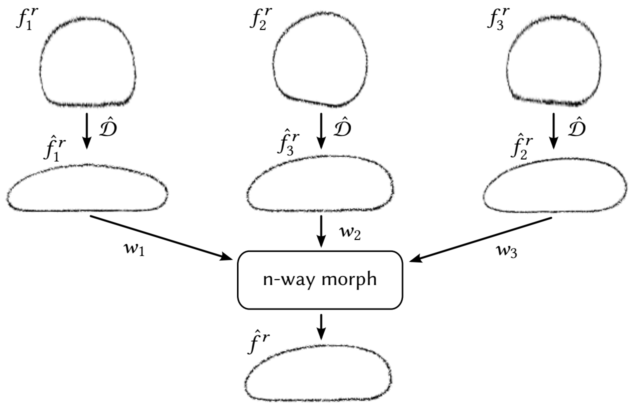


Fig. 8. Synthesis of fine-scale details $\mathcal{R}$ — the synthesized quadratic deformation $\hat{\mathcal{D}}$ is applied to individual residual frames f_1^r, f_2^r, f_3^r producing their deformed counterparts $\hat{f}_1^r, \hat{f}_2^r, \hat{f}_3^r$. Those are then blended together using n-way morphing [Lee et al. 1998] to produce the resulting frame $\hat{f}^r$.

6 FRAME-BY-FRAME STYLIZATION

Although the parametric transformations $\mathcal{D}$ capture most of the dominant global deformations, there are still small residual deformations and appearance variations $\mathcal{R}$ (e.g., sketch lines of the drawings) which cannot be simply described by the quadratic deformation model. These residual changes represent a very important part of the stylization, as they provide much of the uniqueness of traditional hand-drawn, as opposed to computer-generated animation.

Extraction of the residual. Due to the parametric deformation model, extracting $\mathcal{R}$ from the source exemplars is straightforward. We compute and store the residual frames in F_i^R by "rectifying" the example frames in F_i^E using the inverse transformation to the deformation $\mathcal{D}$ estimated in Section 5.1 (see Figure 5(b)).

Synthesis of fine-scale details. For a given target frame in F_j^T, we now want to re-introduce the residual variations to the synthesized parametric transformation $\hat{\mathcal{D}}$. As illustrated in Figure 8, we deform the corresponding residual frames in $f_1^r, \ldots, f_K^r$ using $\hat{\mathcal{D}}$ to produce a set of deformed example frames $\hat{f}_1^r \ldots \hat{f}_K^r$. We then compute a set of pairwise pixel-level warping fields $\phi_{\kappa,\kappa'} : \hat{f}_\kappa^r \to \hat{f}_{\kappa'}^r, \forall(\kappa,\kappa') \in \{1 \ldots K\}^2$ using deformable image registration [Glocker et al. 2008]. Finally we apply a weighted n-way morph [Lee et al. 1998] to produce a single output frame $\hat{f}^r$ by displacing and blending pixels in $\hat{f}_1^r \ldots \hat{f}_K^r$ according to the same weights $w_1 \ldots w_K$ as in Section 5.2 and the warping fields $\phi_{\kappa,\kappa'}$.

Sub-sequence stitching. Since the style transfer is applied independently on each animation sub-sequence, stylized overlapping subsequences need to be stitched together to avoid abrupt changes. We apply the same approach as described in previous paragraphs, but now only with two frames and with blending weights proportional to the temporal distance τ: to stitch overlapping sub-sequences i and $i+1$, we use $w_i = \tau/M_{i,i+1}$ and $w_{i+1} = 1 - \tau/M_{i,i+1}$, where $M_{i,i+1}$ is the number of frames in the transition.

Contact points adjustment. Our synthesis process does not guarantee that the resulting animation preserves the alignment with obstacles at contacts. This issue bears resemblance to the foot step detection/correction mechanism used when blending motions in skeletal animation. Yet our problem is simpler since we know the position of contact points; we can easily verify whether the spatial alignment with obstacles is preserved. If not, we simply shift or slightly rotate the synthesis result so that it aligns perfectly with the obstacle at collision time. To estimate the corresponding translation and rotation we use again image registration algorithm of Sýkora et al. [2009]. To avoid ambiguity in translation along the obstacle during the registration, we restrict the centroid of the synthesized drawing to move perpendicularly to (along the normal of) the nearest obstacle.

Texturing. We support two options to apply a texture inside the deformed drawings. The first one takes as input a static image whose content is constant during the full sequence. This image is first rotated according to the target sequence orientation, then it is registered with every residual frame f_i^r using [Glocker et al. 2008], and finally replaces those during the subsequent fine-scale synthesis steps (see Figure 13(c)). If the content of the texture varies in time, the artist needs to provide two versions of the style exemplar: one only showing the outline of the drawing and another with the full texture (see Figure 17). The former is used for quadratic registration whereas the latter is copied during the frame-by-frame synthesis.

Stroke appearance transfer. To offer additional artistic controls, we optionally allow the stroke appearance to be re-synthesized by exemplar using *StyLit* [Fišer et al. 2016]. This can also help suppress resampling artifacts that may occur when applying the quadratic and free-form deformations. In practice we replace the complex illumination-based guidance with a simple gray-scale guiding channel G that softly demarcates positions of strokes in the source and in the target image (see Figure 9).

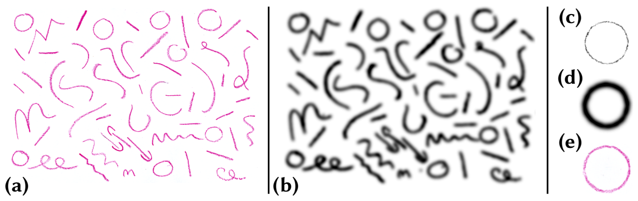


Fig. 9. Stroke appearance transfer based on StyLit algorithm [Fišer et al. 2016] — it takes as input an exemplar drawing containing few strokes created using the desired drawing media, e.g., color pencil **(a)**, its corresponding guiding channel **(b)**, the target frame synthesized using our method **(c)** and its guiding channel **(d)**, and produces the resulting appearance transfer **(e)**.

To generate G we filter the input stroke style exemplar as well as the output frame using the *FDoG* filter [Kang et al. 2007] which suppresses local discrepancies caused by physical properties of the artistic medium (in the stroke style exemplar) or by errors caused by warping and blending pixels (in the target stylized frame) and produces clean stroke outlines. Then a Gaussian blur is applied on this clean outline mask and the resulting bitmap image is normalized and used as a guiding channel for the StyLit algorithm.

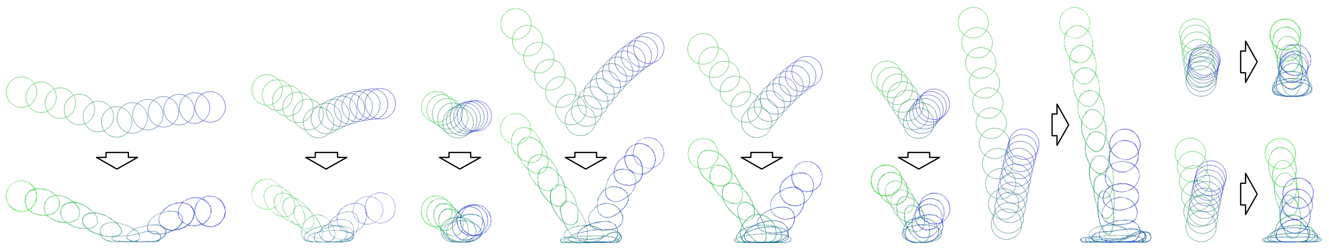


Fig. 10. Style pair database — overview of the source animations and their corresponding stylizations of a bouncing ball drawn by a professional artist.

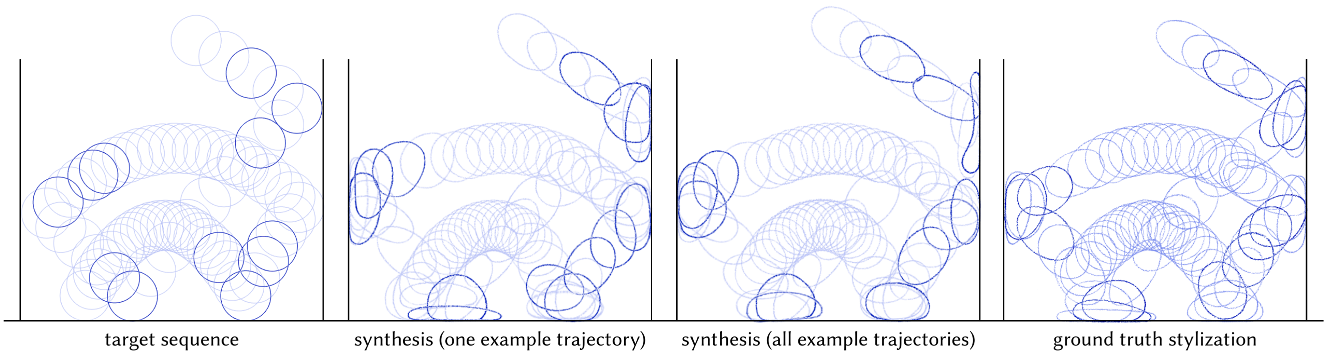

target sequence
synthesis (one example trajectory)
synthesis (all example trajectories)
ground truth stylization

Fig. 11. Results of our synthesis algorithm — (from left to right) target animation computed using physical simulation, synthesis using only one exemplar trajectory, synthesis using all available exemplar trajectories shown in Figure 10, ground truth stylization drawn by a professional artist.

7 RESULTS

We implemented our approach using a combination of Matlab/C++ code, and integrated it as a plug-in into "TVPaint Animation", a professional 2D animation software. In all our experiments, the input sequences were split into sub-sequences of $N_i = \frac{2}{3}\#(e_{i-1}, e_{i+1})$ frames, where $\#(e_i, e_j)$ is the number of frames between the key events e_i and e_j. The overlap $M_{i,i+1}$ was set to $\frac{2}{3}\#(e_i, e_{i+1})$. For source-target sub-sequence matching, N was set to 20 and the weights were set to $\lambda_{\text{vel}} = 1$, $\lambda_{\text{dir}} = 1$, $\lambda_{\text{rot}} = 50$ to encourage consistent rotation.

To validate our method we designed two practical usage scenarios: (1) an *off-line scenario* where an existing database of pre-made style exemplars is used to perform stylization of new more complex sequences and (2) an *interactive scenario* where an animator specifies a set of sparse style exemplars in the target sequence and those are then used as a source for the synthesis of the entire sequence.

In the case of the off-line scenario, to create the database, we asked a professional animator to stylize a few physical simulations with a single ball (see Figure 10) and square bouncing on the ground (see Figure 1). This database of exemplars was then used to stylize more complex physical simulations containing multiple interacting objects in different environment (see Figures 1, 11, 12, 16, 13, 17 and the supplementary video). For this scenario, we precompute parametric deformations for each exemplar in the database. Our implementation can precompute 10 frames in about 13 seconds on average. Using the precomputed deformations, synthesis of an animation containing 100 frames lasts approximately 56 seconds. The most time consuming parts of the pipeline are the image registration phases. Parallel processing could be used to accelerate the block

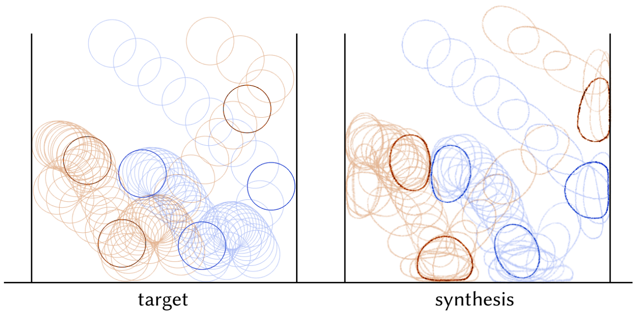


Fig. 12. Result of our synthesis algorithm for two bouncing balls — target sequence computed using physical simulation (left); resulting synthesis computed using all available exemplar trajectories shown in Figure 10 (right).

matching step of Sýkora et al. [2009] and the graph-cut computation in Glocker et al. [2008]. In Figure 14 we also demonstrate the synthesized sequences with additional stroke appearance transfer which further enhances the hand-drawn look and feel.

For the interactive scenario, we let the artist work in TVPaint Animation, a professional animation software, using our plug-in to iteratively improve the stylization of a target sequence. The artist first selects a few frames where a more pronounced stylization is required and draws over them. The target sequence is then re-synthesized taking into account those modified frames, and the artist can inspect the result, tweak already modified frames, or provide stylization for additional frames (see the supplementary video for the recording of a live editing session). In this scenario, we compute

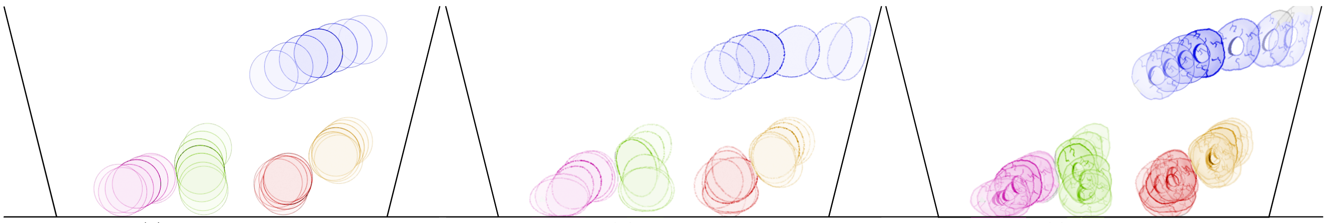


Fig. 13. Result of our synthesis algorithm for more colliding balls — **(a)** target sequence computed using physical simulation; **(b)** resulting synthesis computed using all available exemplar trajectories shown in Figure 10; **(c)** "stone wheel" texture applied during the frame-by-frame synthesis.

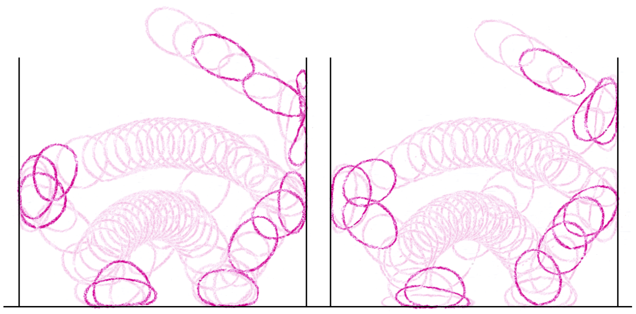


Fig. 14. Stroke appearance transfer applied to our stylized result (left), and on the same sequence drawn by a professional artist (right).

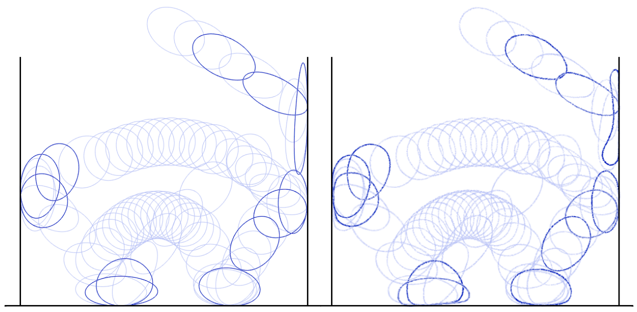


Fig. 15. Comparison — synthesis using only the quadratic deformation model $\mathcal{D}$ (left), and with the full processing pipeline which includes both the quadratic deformation $\mathcal{D}$ and the residual changes $\mathcal{R}$ (right). Note how the addition of the residual changes $\mathcal{R}$ significantly improves the expressiveness of the resulting animation.

parametric deformations only for newly stylized frames and then the synthesis is executed with comparable computational overhead as in the off-line scenario. In the captured session, the artist stylized only 5 percent of the frames until she was satisfied with the result. The method could even work with a single stylized frame, but it will obviously produce visually pleasing results only when a few frames before and after a collision event are provided.

To confirm that our approach reproduces the hand-drawn style well, we asked the professional artist to stylize the entire sequence frame-by-frame (ground truth), and we also let our algorithm synthesize the same sequence using exemplars (1) from the database and (2) from a fraction of the stylized frames in the ground truth sequence (see Figure 11). Then, we asked two professional animators and three casual observers to compare the three results side-by-side and assess the overall motion stylization quality. In most cases the results were almost indistinguishable from ground truth for non-professional observers. Professional artists were able to see some discrepancies namely in frames of which assumed stylization diverted significantly from the available exemplars, however, the overall response was very positive and they highly appreciate the advantage of having a trade-off between the final visual quality and the time required to produce the exemplars.

In addition to the comparison with a ground truth sequence, we also provide an additional comparison showing the necessity of the decomposition into a quadratic deformation $\mathcal{D}$ and residual changes $\mathcal{R}$. We synthesize sequences using only our parametric model and then compare with those synthesized using the full processing pipeline (see Figure 15).

8 LIMITATIONS AND FUTURE WORK

We have shown that our approach can produce good results using only a small subset of exemplar frames, and as a result can notably lower the amount of manual work required to create a variety of 2D animations. However, those are currently restricted to simple rigid bodies, and some practical – rather than conceptual – limitations need to be addressed to allow the stylization of more complex sequences.

First, as mentioned in Section 4, for simplicity, our method does not explicitly model the spacing variations that an artist may introduce compared to the physical simulation (e.g., ease-in / ease-out effects). This is especially noticeable in the dynamics of collisions. To a certain extent, those variations are implicitly captured by the translational part of the parametric transformation, but only as a linear frame-by-frame approximation when it should maybe be modeled as a continuous function along the rigid body trajectory. This can be seen in instances where a shape deforms along the contact plane just before or right after the collision (levitation effect).

Besides, our approach takes advantage of rigid body symmetries which are helpful namely for the synthesis of rotations. In particular, exemplars of symmetric counterparts can be reused in situations where there is no corresponding sequence available in the original orientation. However, for asymmetric objects this simplification cannot be used and thus more exemplars need to be provided in order to be able synthesize consistent orientations. To further reduce

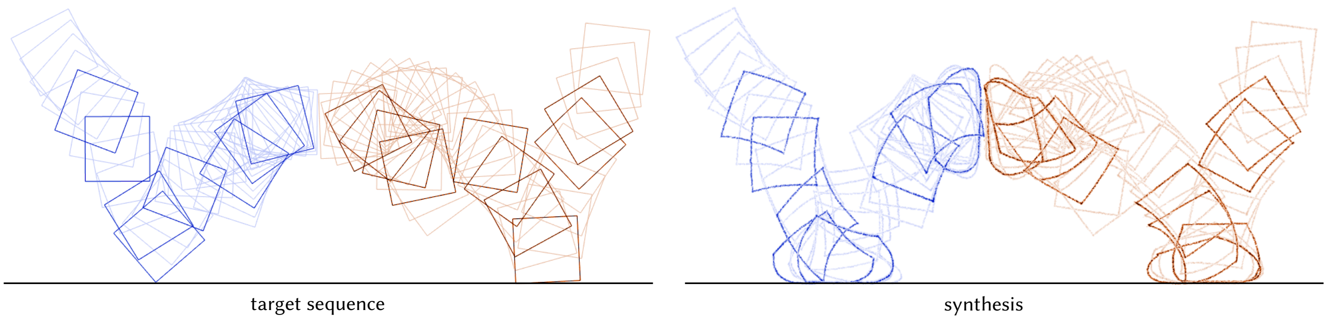


Fig. 16. Result of our synthesis algorithm for two interacting squares — the style pairs of Figure 1 are used, these only contain collisions with the ground plane.

the burden on the artist in such cases, it would be required to model the rotation of the example object separately and allow some amount of re-targeting during the transfer to the novel sequence.

The success of our method also strongly depends on the result of the automatic image registration phase, during which the quadratic deformation model is fit to the stylized exemplar. For elongated objects such as the car in Figure 17, it would be interesting to explore the use of multiple quadratic deformation clusters such that the front of the object may be stylized earlier than its rear along the motion trajectory. More complex or excessive stylization may not fit well to the available degrees of freedom and consequently the resulting residual changes may become overly large. This can cause difficulties during the deformable image registration and the subsequent n-way morphing phase. As future work, we would like to explore different parametric models which would be able to describe a larger variety of deformations occurring in hand-drawn animations. Those more complex models will probably require a better image registration algorithm with a hierarchical processing that is able to adaptively add individual degrees of freedom to avoid getting stuck in some erroneous configurations. It would also require a more advanced blending scheme.

When the exemplar animations differ considerably as compared to the target motion our method just picks a set of most similar ones which likely would not be appropriate. Moreover, when there is a large difference in velocities, the dampening phase tends to remove the stylization effect. Both effects are undesirable and serve as a signal for the artist to provide some more appropriate exemplars.

A last drawback of our technique is that it requires multiple pixel-based warping operations to produce the final shape. Although their number can be minimized by applying compound quadratic transformations, subsequent image re-sampling may still introduce artifacts that can alter the genuine appearance of the used artistic media. This limitation can be partially alleviated by additional post-processing steps such as the proposed appearance re-synthesis or vectorization.

Combining all these improvements, we plan to extend our method to articulated objects such as cartoon characters. In this case a more complex, potentially hierarchical parametric model will be needed.

9 CONCLUSION

We have presented a method allowing to perform style transfer from an existing 2D hand-drawn animation exemplar to a more complex rigid body animation. To the best of our knowledge this is the first attempt to provide a solution for such a challenging task. Despite the recent success of example-based stylization techniques focusing on appearance transfer, animation is still a mostly unexplored area and we believe that the results provided in this paper will motivate other researchers to explore and solve the large variety of challenges that emerges when considering transfer of motion style to more complex articulated objects such as cartoon characters.

ACKNOWLEDGEMENTS

We would like to thank Zuzana Studená for preparation of hand-drawn animations used in the paper, Petra Hapalová for integrating our algorithm into TVPaint, Vojtěch Votýpka for the recording of interactive sessions, and all anonymous reviewers for insightful comments and suggestions. This research was funded by Adobe and has been supported by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center) and by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/237/OHK3/3T/13 (Research of Modern Computer Graphics Methods).

REFERENCES

Yunfei Bai, Danny M Kaufman, C. Karen Liu, and Jovan Popović. 2016. Artist-directed dynamics for 2D animation. *ACM Transactions on Graphics* 35, 4 (2016), 145.

Ronen Barzel and Alan H. Barr. 1988. A Modeling System Based on Dynamic Constraints. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '88)*. ACM, 179–188.

Pierre Bénard, Forrester Cole, Michael Kass, Igor Mordatch, James Hegarty, Martin Sebastian Senn, Kurt Fleischer, Davide Pesare, and Katherine Breeden. 2013. Stylizing animation by example. *ACM Transactions on Graphics* 32, 4 (2013), 119.

Christoph Bregler, Lorie Loeb, Erika Chuang, and Hrishi Deshpande. 2002. Turning to the Masters: Motion Capturing Cartoons. *ACM Transactions on Graphics* 21, 3 (2002), 399–407.

Erin Catto. 2007. Box2d – a 2D physics engine for games. http://www.box2d.org. (2007).

Stephen Chenney, Mark Pingel, Rob Iverson, and Marcin Szymanski. 2002. Simulating Cartoon Style Animation. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. 133–138.

Stelian Coros, Sebastian Martin, Bernhard Thomaszewski, Christian Schumacher, Robert Sumner, and Markus Gross. 2012. Deformable Objects Alive! *ACM Transactions on Graphics* 31, 4 (2012), 69.

Jakub Fišer, Ondřej Jamriška, Michal Lukáč, Eli Shechtman, Paul Asente, Jingwan Lu, and Daniel Sýkora. 2016. StyLit: Illumination-guided Example-based Stylization of 3D Renderings. *ACM Transactions on Graphics* 35, 4 (2016), 92.

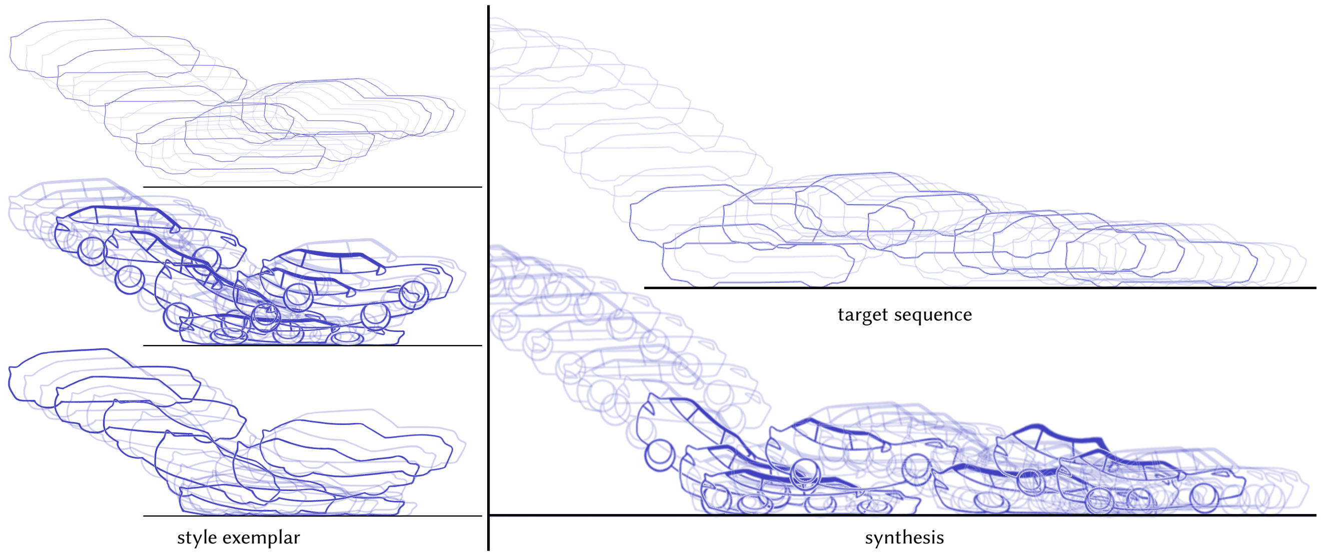


Fig. 17. Result of our synthesis algorithm for a cartoon car – a more complex polygonal proxy is used for the simulation (top left) and registered to the outline of the stylized exemplar (bottom left). During synthesis, the exemplar with texture varying in time (middle left) is used instead.

Marcos Garcia, John Dingliana, and Carol O'Sullivan. 2007. A Physically Based Deformation Model for Interactive Cartoon Animation. In *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2007)*. Eurographics Association.

Leon A Gatys, Alexander S Ecker, and Matthias Bethge. 2016. Image style transfer using convolutional neural networks. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 2414–2423.

Ben Glocker, Nikos Komodakis, Georgios Tziritas, Nassir Navab, and Nikos Paragios. 2008. Dense Image Registration Through MRFs And Efficient Linear Programming. *Medical Image Analysis* 12, 6 (2008), 731–741.

Michael Haller, Christian Hanl, and Jeremiah Diephuis. 2004. Non-photorealistic Rendering Techniques for Motion in Computer Games. *Computers in Entertainment* 2, 4 (2004).

Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. 2001. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 327–340.

Nicholas J. Higham and Robert S. Schreiber. 1990. Fast polar decomposition of an arbitrary matrix. *SIAM J. Sci. Statist. Comput.* 11, 4 (1990), 648–655.

Ben Jones, Jovan Popovic, James McCann, Wilmot Li, and Adam W. Bargteil. 2015. Dynamic sprites: Artistic authoring of interactive animations. *Journal of Visualization and Computer Animation* 26, 2 (2015), 97–108.

Ben Jones, Nils Thuerey, Tamar Shinar, and Adam W. Bargteil. 2016. Example-based Plastic Deformation of Rigid Bodies. *ACM Transactions on Graphics* 35, 4 (2016), 34:1–34:11.

Henry Kang, Seungyong Lee, and Charles K. Chui. 2007. Coherent Line Drawing. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. 43–50.

Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, and George W. Fitzmaurice. 2014a. Kitty: Sketching dynamic and interactive illustrations. In *Proceedings of ACM Symposium on User Interface Software and Technology*. 395–405.

Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, Shengdong Zhao, and George W. Fitzmaurice. 2014b. Draco: Bringing life to illustrations with kinetic textures. In *Proceedings of ACM Conference on Human Factors in Computing Systems*. 351–360.

Rubaiat Habib Kazi, Tovi Grossman, Nobuyuki Umetani, and George Fitzmaurice. 2016. Motion Amplifiers: Sketching Dynamic Illustrations Using the Principles of 2D Animation. In *Proceedings of ACM Conference on Human Factors in Computing Systems*. 4599–4609.

Yuki Koyama, Kenshi Takayama, Nobuyuki Umetani, and Takeo Igarashi. 2012. Real-time Example-based Elastic Deformation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '12)*. Eurographics Association, 19–24.

John Lasseter. 1987. Principles of Traditional Animation Applied to 3D Computer Animation. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*. ACM, 35–44.

Seungyong Lee, George Wolberg, and Sung Yong Shin. 1998. Polymorph: Morphing Among Multiple Images. *IEEE Computer Graphics and Applications* 18, 1 (1998), 58–71.

Sun-Young Lee, Jong-Chul Yoon, Ji-Yong Kwon, and In-Kwon Lee. 2012. CartoonModes: Cartoon stylization of video objects through modal analysis. *Graphical Models* 74, 2 (2012), 51–60.

Yin Li, Michael Gleicher, Ying-Qing Xu, and Heung-Yeung Shum. 2003. Stylizing Motion with Drawings. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '03)*. Eurographics Association, 309–319.

Jingwan Lu, Fisher Yu, Adam Finkelstein, and Stephen DiVerdi. 2012. HelpingHand: example-based stroke stylization. *ACM Transactions on Graphics* 31, 4 (2012), 46.

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus Gross. 2011. Example-based Elastic Materials. *ACM Transactions on Graphics* 30, 4 (2011), 72:1–72:8.

Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. 2005. Meshless Deformations Based on Shape Matching. *ACM Transactions on Graphics* 24, 3 (2005), 471–478.

Tom Ngo, Doug Cutrell, Jenny Dana, Bruce Donald, Lorie Loeb, and Shunhui Zhu. 2000. Accessible Animation and Customizable Graphics via Simplicial Configuration Modeling. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '00)*. ACM, 403–410.

Paul Noble and Wen Tang. 2006. Automatic Expressive Deformations for Stylizing Motion. In *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia (GRAPHITE '06)*. ACM, 57–63.

Daniel Sýkora, John Dingliana, and Steven Collins. 2009. As-Rigid-As-Possible Image Registration for Hand-Drawn Cartoon Animations. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering*. 25–33.

Frank Thomas and Ollie Johnston. 1981. *The illusion of life : Disney animation.* Disney Editions, New York.

Jue Wang, Steven M. Drucker, Maneesh Agrawala, and Michael F. Cohen. 2006. The Cartoon Animation Filter. *ACM Transactions on Graphics* 25, 3 (2006), 1169–1173.

Richard Williams. 2001. *The animator's survival kit.* Faber and Faber, London, New York.

Andrew Witkin and Michael Kass. 1988. Spacetime Constraints. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '88)*. ACM, 159–168.

Jun Xing, Rubaiat Habib Kazi, Tovi Grossman, Li-Yi Wei, Jos Stam, and George Fitzmaurice. 2016. Energy-Brushes: Interactive Tools for Illustrating Stylized Elemental Dynamics. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, 755–766.

Jun Xing, Li-Yi Wei, Takaaki Shiratori, and Koji Yatani. 2015. Autocomplete Hand-drawn Animations. *ACM Transactions on Graphics* 34, 6 (2015), 169:1–169:11.

Chapter 6

ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations

The paper included in this chapter was presented at the SIGGRAPH 2018 conference and has been published in ACM Transactions on Graphics journal:

Dvorožňák, M., Li, W., Kim, V. G., and Sýkora, D. [2018]. "ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations". *ACM Transactions on Graphics* 37.4

ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations

MAREK DVOROŽŇÁK, Czech Technical University in Prague, Faculty of Electrical Engineering
WILMOT LI, Adobe Research
VLADIMIR G. KIM, Adobe Research
DANIEL SÝKORA, Czech Technical University in Prague, Faculty of Electrical Engineering

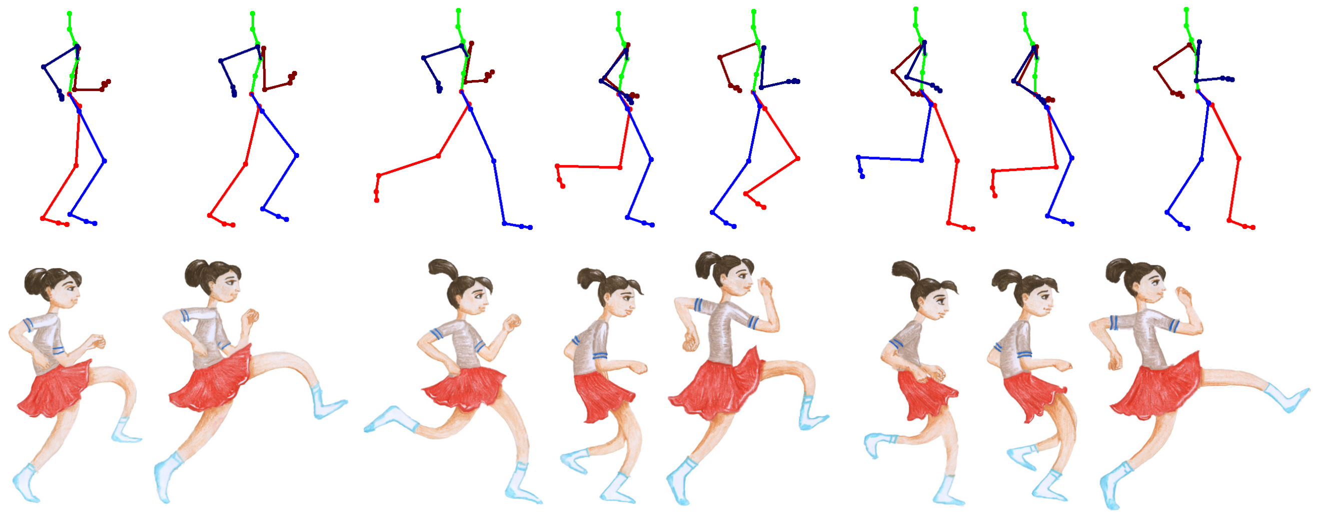


Fig. 1. An example of hand-colored animation synthesized using our approach (bottom row) following the user-specified skeletal animation (top row) and preserving the motion as well as appearance style prescribed by an artist (see a corresponding style exemplar in Fig. 10). Note how the synthesized images still resemble the hand-colored original.

We present a new example-based approach for synthesizing hand-colored cartoon animations. Our method produces results that preserve the specific visual appearance and stylized motion of manually authored animations without requiring artists to draw every frame from scratch. In our framework, the artist first stylizes a limited set of known source skeletal animations from which we extract a *style-aware puppet* that encodes the appearance and motion characteristics of the artwork. Given a new target skeletal motion, our method automatically transfers the style from the source examples to create a hand-colored target animation. Compared to previous work, our technique is the first to preserve both the detailed visual appearance and stylized motion of the original hand-drawn content. Our approach has numerous practical applications including traditional animation production and content creation for games.

CCS Concepts: • **Computing methodologies → Motion processing**; **Image processing**;

Additional Key Words and Phrases: style transfer, skeletal animation

ACM Reference Format:
Marek Dvorožňák, Wilmot Li, Vladimir G. Kim, and Daniel Sýkora. 2018. ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations. *ACM Trans. Graph.* 37, 4, Article 167 (August 2018), 11 pages. https://doi.org/10.1145/3197517.3201326

Authors' addresses: Marek Dvorožňák, Czech Technical University in Prague, Faculty of Electrical Engineering, dvoromar@fel.cvut.cz; Wilmot Li, Adobe Research, wilmotli@adobe.com; Vladimir G. Kim, Adobe Research, vokim@adobe.com; Daniel Sýkora, Czech Technical University in Prague, Faculty of Electrical Engineering, sykorad@fel.cvut.cz.

1 INTRODUCTION

While advances in computer graphics have contributed to the evolution of 3D animation as an expressive, mature medium, 2D animation remains an extremely popular and engaging way to tell stories. One common workflow for creating 2D animations is to decompose characters, objects and the background into separate layers that are transformed (either rigidly or non-rigidly) over time to produce the desired motion. A key advantage of this layer-based approach is that a single piece of artwork (i.e., layer) can be reused across many animated frames. As long as the appearance of the layer does not change dramatically (e.g., a character's torso turning from a front to side view), the artist does not need to redraw from scratch. Compared to drawing and coloring every frame by hand, animating with layers greatly reduces the authoring effort, which is one reason why many modern cartoon series (e.g., Archer, BoJack Horseman, Star vs the Forces of Evil) are created in this manner.

Unfortunately, this increase in efficiency comes at a cost. While hand-created animations give artists complete freedom to specify the appearance of each frame, many styles of artwork are hard to animate using a typical layer-based workflow. Since layers are reused and transformed across several frames, painterly artwork can look awkward as textured regions are compressed and stretched. In addition, rendering styles with visible brush strokes often appear somewhat "dead" when the pattern of strokes remains fixed from frame to frame. Beyond the appearance of the artwork, the motion of layers is also constrained since commercial tools typically enable a limited set of transformations that do not directly support many secondary effects or exaggerated bending and bulging of moving parts. As a result, most layer-based animations are rendered in simple, flat-shaded styles and exhibit relatively stiff or jerky motion.

In this work, we propose an example-based layered animation workflow that allows artists to customize the appearance and motion of characters by specifying a small set of hand-colored example frames for one or more specific source motions. Our system automatically captures and applies the style of the example to new target motions. The key difference between our approach and standard layered animation is that target animation frames are generated by synthesizing each layer based on the set of example frames rather than transforming a single drawn layer. Since the synthesis procedure preserves stylistic aspects in the appearance and motion of the hand-colored source animation, our method supports a much wider range of animation styles. Compared to traditional frame-by-frame drawing, our approach allows artists to get much greater use out of their artwork, since a relatively small set of drawings can be leveraged to produce animated results for a variety of related motions (e.g., a drawn walk cycle can be used to generate a fast angry walk, slow sneaky walk, etc.).

Existing example-based techniques for 2D animation mostly focus on individual sub-problems such as 2D shape interpolation, motion, or appearance transfer. However, focusing on individual steps separately leads to noticeable discrepancies between the real hand-drawn artwork and computer generated output: either the motion characteristics or visual appearance lack quality. For example, in some cases shapes are interpolated with the proper motion characteristics, but the appearance includes artifacts due to distortion or blending of textures [Arora et al. 2017; Baxter et al. 2009; Sýkora et al. 2009]. Or, the appearance is transferred properly, but the underlying motion feels too artificial [Fišer et al. 2017, 2014]). Thus, a key remaining challenge is to combine motion and appearance stylization into a holistic framework that produces synthesis results with all the characteristics of hand-drawn animations. To our best knowledge, our approach is the first that provides such a joint solution and enables fully automatic synthesis of convincing hand-colored cartoon animations from a small number of animation exemplars.

We tailor our method to handle in-plane motions with occlusions, which are typical for cartoon animations and gaming scenarios. Focusing on such motions allows us to apply a relatively simple algorithm that still produces effective results supporting a range of practical applications. For out-of-plane motions that involve more complex depth order changes as well as topological variations, additional manual intervention would be necessary.

Our paper makes the following specific contributions. We define the concept of a layered *style-aware puppet* that is flexible enough to encode both the appearance and motion stylization properties exemplified by the artist's hand-colored animation frames. We also present a mechanism to combine the information captured by this puppet to transfer motion and appearance style to target animations prescribed by skeletal motion. A key benefit of our technique over previous work is that we specifically designed our pipeline to preserve the visual characteristics of the original artistic media, including a user-controllable amount of temporal incoherence.

2 RELATED WORK

Pioneered by Catmull [1978], there has been a concerted effort over the last few decades to simulate or simplify the production of traditional hand-drawn animation using computers.

Computer-assisted inbetweening [Kort 2002] — i.e., generating smoothly interpolated animation from a set of hand-drawn keyframes — is one of the problems that has received significant attention. Various techniques have been proposed to tackle it, achieving impressive results both in the vector [Baxter and Anjyo 2006; Whited et al. 2010; Yang 2017] and raster domains [Arora et al. 2017; Baxter et al. 2009; Sýkora et al. 2009]. Some of these techniques propose N-way morphing between all available frames to widen the available pose space. Nevertheless, inbetweening is designed to deliver plausible transitions only between keyframes. To produce animation for a new target motion, artists must create additional keyframes by hand.

Another large body of research focuses on the simulation of basic motion principles seen in traditional animations, including squash-and-stretch, anticipation, and follow-through [Lasseter 1987]. Existing work proposes customized procedural techniques [Kazi et al. 2016; Lee et al. 2012; Schmid et al. 2010; Wang et al. 2006] as well as controllable physical simulation [Bai et al. 2016; Jones et al. 2015; Willett et al. 2017; Zhu et al. 2017]. Although these methods are capable of achieving the look-and-feel of traditional animation, they do not in general preserve specific motion details that often characterize a given artist's style. These techniques also do not consider how to faithfully preserve the detailed visual appearance of hand-drawn artwork that is in motion. In most cases, textures are simply stretched and deformed, which leads to visual artifacts.

To retain more of a hand-drawn appearance, some techniques directly reuse or manipulate existing hand-drawn content. They either use the animation sequences unchanged [Buck et al. 2000; van Haevre et al. 2005; de Juan and Bodenheimer 2004, 2006] and only reorder the animation frames, add more inbetweens, or directly manipulate the appearance on a pixel level [Sýkora et al. 2011, 2009; Zhang et al. 2012] to enhance the visual content or change the motion characteristics. Although these approaches better preserve the notion of hand-colored animation, their potential to make substantial changes to the motion is rather limited. Extensive manual work is typically required when a different animation needs to be produced out of existing footage.

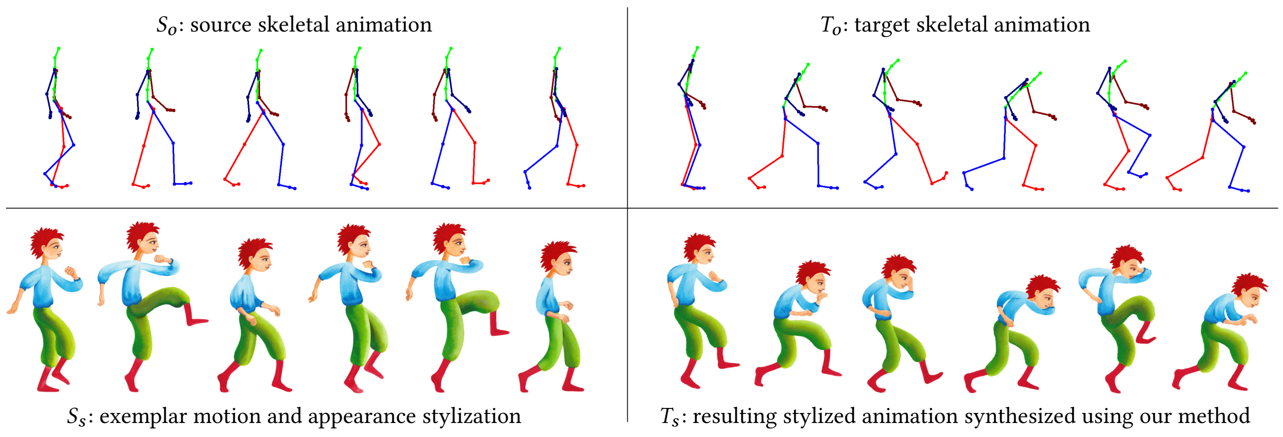

S_o: source skeletal animation
T_o: target skeletal animation
S_s: exemplar motion and appearance stylization
T_s: resulting stylized animation synthesized using our method

Fig. 2. The animation analogy concept: for a given source skeletal animation (S_o), an artist prepares a corresponding hand-colored animation which jointly expresses stylization of character's motion and appearance (S_s). Then for a different target skeletal animation (T_o), our system produces a synthetically-generated hand-colored animation (T_s) that respects the provided analogy $S_o : S_s :: T_o : T_s$ and transfers the motion and appearance style to (T_o).

Rather than directly reusing hand-drawn frames, image analogies [Hertzmann et al. 2001] provides a powerful framework for synthesizing new content based on example artwork. In this approach, a guiding image and its stylized version are provided to define the style transfer analogy. This approach has been extended to stylize animations [Bénard et al. 2013] with later work adding user control over the amount of temporal flickering [Fišer et al. 2017, 2014] to better preserve the impression that every animation frame was created by hand independently. However, these analogy-based approaches only support appearance style transfer and do not consider how to represent and apply motion stylizations.

Recently, Dvorožňák et al. [2017] presented a motion style analogy framework that has similar motivations to our pipeline. In their workflow, an artist prepares a set of hand-drawn animations that stylize input rigid body motion (of circles or squares) computed using physical simulation. Then they analyze the style by registering a quadratic deformation model as well as a residual deformation. Finally, for a given target rigid body animation, they synthesize a hand-drawn animation by blending the deformation parameters from similar exemplar trajectory segments. One key difference in our work is that we focus not only on motion stylization but also appearance synthesis for fully colored drawings. While Dvorožňák et al.'s method does synthesize simple outline drawings, our approach is designed to support a wide range of hand-colored rendering styles. In addition, the previous technique only handles simple rigid body scenarios where each object in the scene can be represented by a single artwork layer and one set of deformation parameters. In contrast, we describe an analogy framework that works for complex, multi-layered, articulated characters.

Skeletal animation [Burtnyk and Wein 1976] has proven to be an efficient tool for deforming 2D shapes [Hornung et al. 2007; Vanaken et al. 2008]. It has been used to control deformation in the context of cartoon animations [Sýkora et al. 2005; Wang et al. 2013] as well as to transfer motion from a sequence of drawings [Bregler et al. 2002; Davis et al. 2003; Jain et al. 2009] or a single pose [Bessmeltsev et al. 2016] onto a 3D model. In our framework, we demonstrate that skeletal animation can be used also as an effective guide to perform style transfer between hand-drawn exemplars and target animation.

3 OUR APPROACH

The primary goal of our work is to help artists create hand-colored animations of characters without having to draw every frame from scratch.

Motivated by the abundance of available motion capture data thanks to recent advances in pose estimation [Mehta et al. 2017], professional MoCap systems (Vicon, OptiTrack, The Captury), and existing motion databases (CMU, HumanEva, HDM05), we assume skeletal animation is easily accessible and can serve as a basic tool to convey motion characteristics. Moreover, tools such as MotionBuilder allow users to combine and extend existing MoCap data using forward/inverse kinematics to create skeletal motions suitable for our method.

Thus, we focus on the challenge of generating colored animations that match a given target skeletal motion while at the same time follow the visual appearance and motion style of an artist-created analogy where a few hand-colored frames serve as an example of how the artist would stylize a particular skeletal animation. Inspired by previous analogy-based techniques [Dvorožňák et al. 2017; Hertzmann et al. 2001] we call our approach *animation analogies*.

In our framework, the artist first chooses a short *source skeletal animation* S_o and creates a *source stylized animation* S_s by authoring hand-colored frames that express the stylization of the source skeletal animation. We call this pair a *source exemplar* $S_o : S_s$. In the source exemplar, we assume that frames of S_s roughly follow the motion in S_o, but the details of the motion can be different due to stylization effects. For example, if S_o is a walk cycle, we assume the foot steps in S_s are synchronized, but the legs themselves may

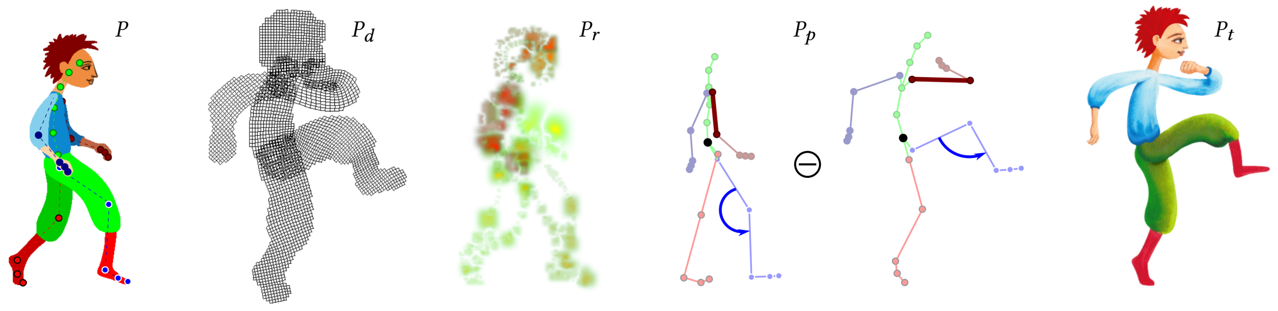


Fig. 3. A style-aware puppet P_s consists of a layered template puppet P, coarse deformation of individual puppet shapes P_d, their residual elastic deformations captured by multi-layer residual motion field P_r (layers and the magnitude of deformation are color-coded), the difference between the source and stylized skeletal pose P_p, and the stylized texture of the character P_t.

bend and stretch in an exaggerated way. We also assume that each stylized frame $S_s(i)$ can be separated into a consistent set of layers that are associated with the skeleton bones in $S_o(i)$ and that the depth order of the layers matches that of the corresponding bones. The shape of occluded parts in those layers can be either automatically reconstructed [Sýkora et al. 2014; Yeh et al. 2017] or manually inpainted. An artist can also specify detailed appearance stylization of those occluded parts. However, this step is optional as our appearance transfer technique can be used to fill the missing areas automatically. We then analyze the appearance and motion stylization given by the source exemplar $S_o : S_s$ and let the artist or another user provide multiple novel *target skeletal animations* T_o that represent the desired motions of the character for the final animation. Finally, our method uses the analogy $S_o : S_s :: T_o : T_s$ to automatically generate the corresponding hand-colored output frames T_s (see Fig. 2).

While the target skeletal motions T_o can differ considerably from the source S_o, we expect some similarities for our analogy-based framework to work. For example, the artist might stylize a standard walk cycle and transfer the stylization to a sneaky walk, drunk walk, or running cycle. However, a jumping motion might be too dissimilar from the source to stylize successfully, in which case a different style exemplar can be created.

To enable this analogy-based workflow, we propose a guided synthesis technique that uses the style exemplar $S_o : S_s$ to generate stylized frames T_s for the target skeletal motion T_o. Our method has two main stages. First, we analyze the source animations to determine the relationship between the skeletal animation S_o and the corresponding hand-colored data S_s. Specifically, we construct a *style-aware puppet* P_s that encodes the pose, shape, and appearance stylization S_s for every frame from S_o. Once we have this encoding, we can automatically apply the stylization to frames in T_o and generate a new hand-colored animation T_s. The following sections describe these two stages in detail.

3.1 Representing Source Stylization

Given the source skeletal S_o and stylized S_s animations, we construct a style-aware puppet P_s that describes the pose, shape and appearance properties of the exemplars with respect to a layered template puppet P. The template puppet P represents the character in a "neutral" pose; it has the same set of layered parts as the source artwork where each part is associated with a corresponding portion of the source skeleton (see Fig. 4). In case some parts are occluded in the original artwork, we ask the artist to complete their shapes and also specify important semantic details that needs to be preserved (e.g., facial features or cloth draping). We then encode the stylization by registering the template puppet P to every hand-colored source frame $S_s(i)$. This allows us to extract the deformed skeletal pose as well as detailed shape deformation of the character with respect to the neutral pose of P. We also encode the appearance of the character in the form of a texture. More formally, a style-aware puppet P_s consists of a layered template puppet P and a tuple $[P_d, P_r, P_p, P_t]$ for each stylized frame i (see Fig. 3): $P_d(i)$ captures the coarse deformation of individual puppet shapes, $P_r(i)$ their residual elastic deformation, and P_p the difference between the source skeletal pose $S_o(i)$ and stylized skeletal pose $S_p(i)$. $P_t(i)$ is the stylized texture of the character. We use these tuples to stylize novel skeletal animations T_o.

Layered Template Puppet Creation. To create a layered template puppet P, we can either use a special unstylized frame in a rest

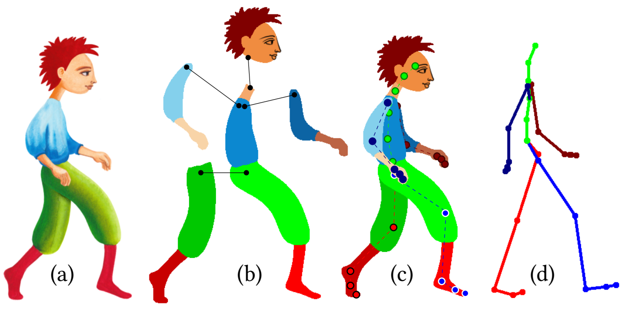


Fig. 4. An example of a layered template puppet: for a single existing hand-colored frame (a), we create a set of semantically meaningful layers which are interconnected at junctions (b) and assign joints of the source skeleton (d) to corresponding locations on each individual layer (c).

pose created by the artist or one of the frames taken from the input hand-drawn animation S_s. It consists of a set of semantically meaningful layers (e.g., head, body, hands, and legs) manually stitched together at locations where they naturally connect. Each layer must be attached to the underlying skeleton at one or more user-specified joints. These attachments define the correspondence between bones and layers (see Fig. 4).

Registration. To register the template puppet P to every frame i of the segmented hand-colored animation S_s, we use a similar approach as in Dvorožňák et al. where a coarse deformation is estimated first and then a more detailed residual motion is extracted. This coarse-to-fine strategy improves the robustness of the registration algorithm while still allowing us to encode very accurate deformations. While Dvorožňák et al. use a single as-rigid-as-possible (ARAP) mesh, a key improvement of our approach is that we use a layered ARAP model with multiple piecewise connected meshes defined by our layered template puppet P.

We compute the coarse deformation using the ARAP image registration algorithm [Sýkora et al. 2009], which iteratively applies two steps: the *pushing phase* shifts every point on the ARAP mesh towards a better matching location in the target image using a block-matching algorithm; and the *regularization phase* keeps the ARAP mesh consistent. To use this approach with our multi-mesh ARAP model, we adapt the pushing phase so that the block-matching only uses the content of the corresponding layer to shift each mesh (see Fig. 6, left). This concept is similar to the depth-based separation used in [Sýkora et al. 2010], which avoids clutter caused by occlusion and improves the overall accuracy of the final registration. The registration process as described is automatic. Nevertheless, there can be challenging configurations (e.g., when the deformation is large compared to the template) where manual intervention (dragging a control point to the desired location) can help to speed up the registration process or correct possible misalignments.

Once we obtain a coarse deformation of our layered template puppet $P_d(i)$, we rectify each hand-colored part by removing the computed coarse deformation and perform a more accurate elastic registration between the template and the rectified frame using the method of Glocker et al. [2008]. The result of this step is a multi-layer residual motion field $P_r(i)$ that encodes subtle shape changes of individual body-parts (Fig. 6, right).

To compute $P_p(i)$ we need to infer the stylized skeletal pose $S_p(i)$ from the configuration of the registered puppet layers. We aim to only obtain a 2D projection of the stylized pose. To do so, we use a topologically equivalent 2D representation of the skeleton that is specified by a root joint position, lengths of skeleton bones and their rotations in the ancestor bone's reference frame. Since each layer is attached to the template skeleton at specific joints, the stylized position of those joints can be directly obtained from the position of the corresponding attachment points on the deformed mesh. $P_p(i)$ is then computed as a difference between root joint positions, bone lengths and their rotations: $P_p(i) = S_p(i) \ominus S_o(i)$.

Finally, $P_t(i)$ is obtained by storing pixels from the hand-colored artwork.

3.2 Style Transfer of Motion and Appearance to Target Skeletal Animation

Synthesis of Motion. We use the extracted style-aware puppet represented by the puppet template P and the per-frame tuples $[P_d, P_r, P_p, P_t]$ to stylize new skeletal animations. We assume that the target skeleton has the same topology as the source skeleton, which is generally true for most MoCap systems.

The transfer of motion style is analogous to patch-based texture synthesis [Kwatra et al. 2005; Wexler et al. 2007] which involves two alternating steps: *search* and *vote*. In our context, instead of texture patches, these steps operate on small sub-sequences of $2N+1$ consecutive skeletal poses around each frame in the source and target animations. The search step finds the closest matching sub-sequence in the source exemplar for each frame in the target and then the voting step averages the content over all intersecting sub-sequences to obtain the final frame pose (see Fig. 5).

More formally, in the search step, we find the closest source sub-sequence $S(i) = S_o[(i-N)\dots(i+N)]$ for each target sub-sequence $T(k) = T_o[(k-N)\dots(k+N)]$ using the pose similarity metric of Kovar et al. [2002], which exploits the sum of distances between point clouds formed by the trajectories of corresponding skeleton joints in each sub-sequence after removing global translation.

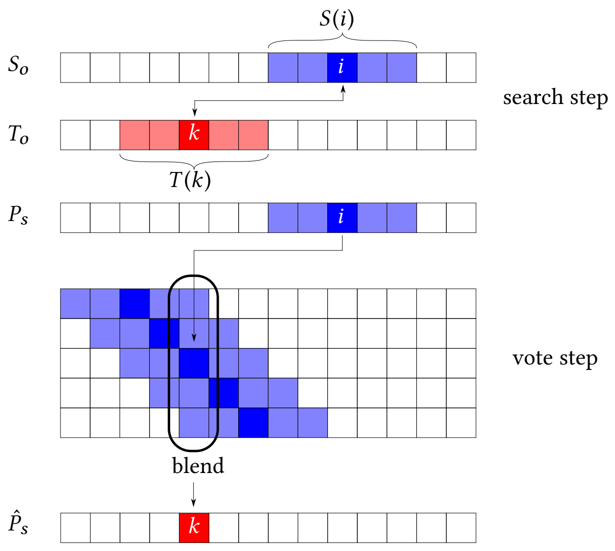


Fig. 5. Obtaining a blended style-aware puppet $\hat{P}_s$ for a target frame: for a sub-sequence of the target skeletal animation $T(k)$, the closest sub-sequence of the source skeletal animation $S(i)$ is found (search step) and then the corresponding sub-sequence of style-aware puppets $P_s(i)$ is blended with other intersecting sub-sequences (vote step).

Once we have found the best matching source sub-sequence for each target frame, we are left with a set of overlapping source sub-sequences (see Fig. 5). At this point, we perform the voting step to blend over all the source frames (using the information encoded in the associated style-aware tuples) that correspond to each output target frame. This step results in a blended style-aware tuple $[\hat{P}_d, \hat{P}_r, \hat{P}_p, \hat{P}_t]$ for each target frame which is obtained using an

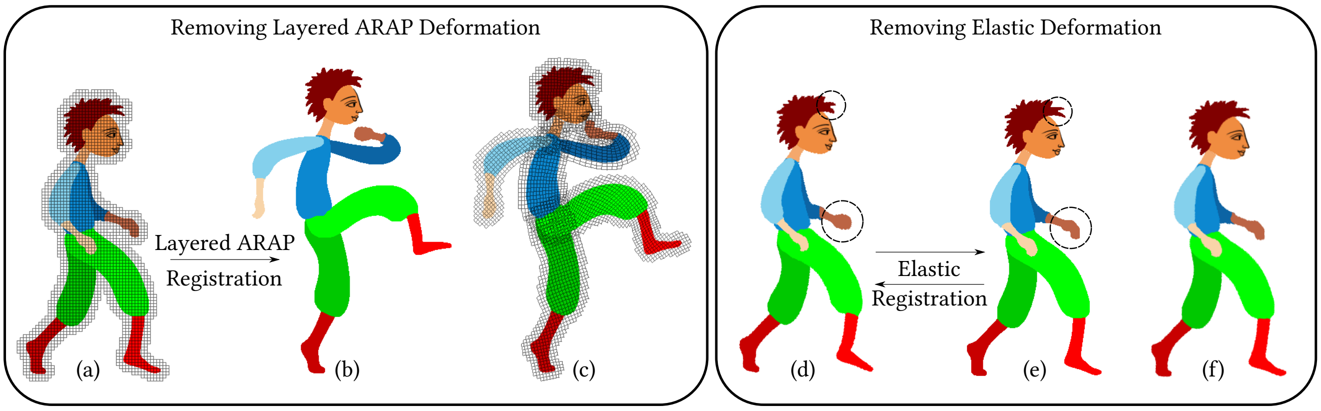

Removing Layered ARAP Deformation
Layered ARAP
Registration
(a)
(b)
(c)
Removing Elastic Deformation
Elastic
Registration
(d)
(e)
(f)

Fig. 6. An example of capturing motion stylization: a layered template puppet (a) is first registered with the segmented version of the stylized animation frame (b) with as-rigid-as-possible (ARAP) image registration [Sýkora et al. 2009] using a layered piecewise connected ARAP deformation model (c). Then, the coarse deformation is removed (d) and the rectified animation frame is registered to the template (e) using the elastic registration method of Glocker et al. [2008] resulting in a segmented stylized animation frame that has both the coarse deformation and the elastic deformation removed (f). Notice the subtle difference in the shape of the hand and hair, which the coarse deformation alone was not able to capture.

N-way ARAP interpolation [Baxter et al. 2009] of the coarse part deformations P_d and a linear blend of the residual shape deformations P_r [Lee et al. 1998] and skeletal pose differences P_p. The blended texture $\hat{P}_t$ is obtained by first rectifying the textures P_t (i.e., removing P_d as well as P_r) and then linearly blending the pixel colors. Finally, we apply the resulting blended skeletal pose difference $\hat{P}_p(k)$ to the target skeleton $T_o(k)$ to obtain its stylized pose (see Fig. 7).

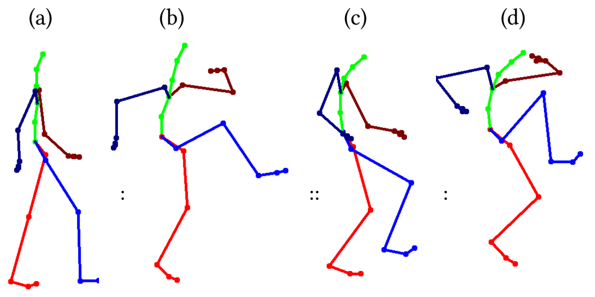


Fig. 7. Style transfer to the target skeletal animation: differences in root joint positions, bone lengths and their rotations between the source skeleton pose (a) and its stylized counterpart (b) are transferred to the target skeleton pose (c) to obtain its stylized pose (d).

Synthesis of Appearance. Once the stylized deformation of the target frame is known, a straightforward way to transfer the stylized appearance would be to deform the blended shapes using the new skeleton joint locations on $T_o(k)$ and warp the blended textural information accordingly. This straightforward solution, however, gives rise to numerous artifacts. Linear blending often smooths away visual details in the original hand-colored frames that are critical to the style of the artwork (see Fig. 8 and the supplementary video for comparison). This is caused mainly by the fact that high-frequency details of individual blended frames are not perfectly aligned and

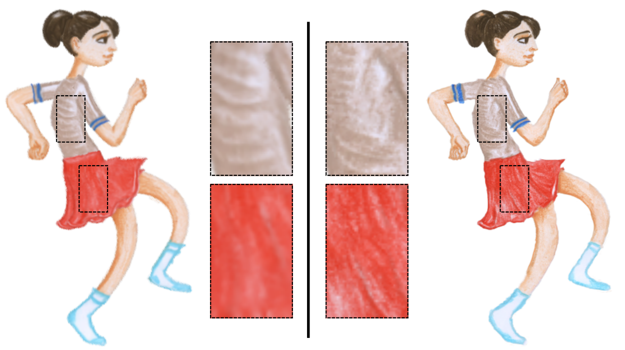


Fig. 8. When the pixel colors of textures of multiple example poses are linearly blended, the result often smooths away subtle details from the original textures (left). This is caused by the blending of slightly different textural content stored in the exemplar frames. The richness of the original textures may be preserved using guided texture synthesis (see the result on the right). See also supplementary video for an animation.

thus simple averaging suppresses them. Moreover, in the case where the artist specifies only the shape of the occluded layers in the style exemplar frames, the stylized target may include regions that do not contain textural information, which need to be filled as well. Finally, blending and warping typically does not produce the same type of temporal variation (i.e., “boiling”) that characterizes many hand-colored animations. Ideally, we would like to support controllable temporal flickering as in [Fišer et al. 2014].

To alleviate all these issues, we replace image warping with guided texture synthesis [Fišer et al. 2017], which creates coherent, detailed texture content and has the flexibility to fill-in newly visible regions. For this technique to work properly, we need to prepare a set of guiding channels that define how texture from the source stylized frames should transfer to the deformed target frames.

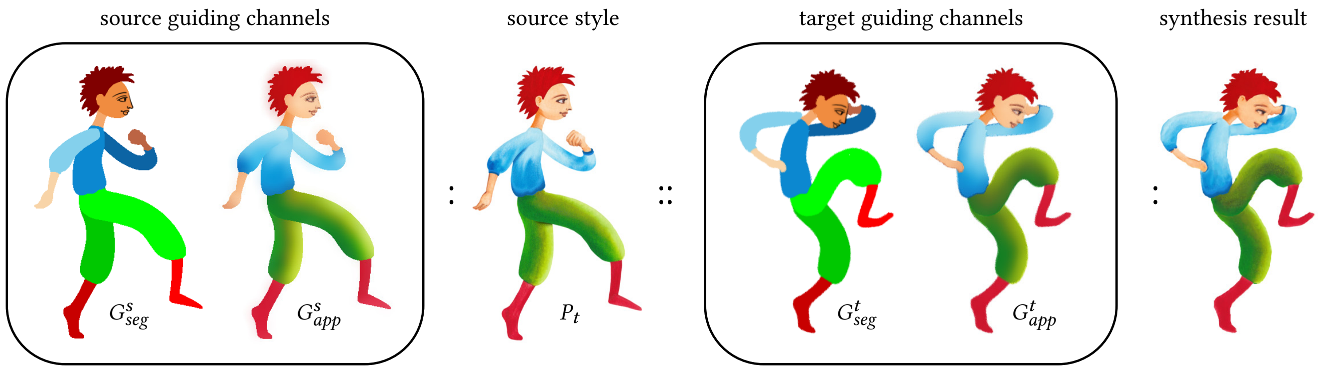


Fig. 9. An example of guiding channels produced by our method to constrain appearance style synthesis: segmentation G_{seg} and temporal appearance G_{app}. The StyLit algorithm [Fišer et al. 2016] is used to perform the actual synthesis using both guiding channels and style exemplar P_t to produce the final animation frame. The amount of blur in the G_{app} controls the amount of temporal flickering in the final animation.

Since the textures for various parts of the character are usually distinct, we want to avoid texture transfer across different parts. To this end, we introduce a segmentation-based guidance channel G_{seg} that represents each segmented part using a separate color label (see Fig. 9). Since the segmentation also contains important semantic details like eyes, nose, and mouth, G_{seg} ensures that these details will be preserved at the appropriate locations.

In addition, we would like to preserve temporal coherence in the synthesized target textures in a controllable fashion. To do so, we introduce a temporal appearance guide G_{app} that influences how consistently the texture is synthesized from one frame to the next. We define G_{app} as the original texture P_t for source frames, and the blended texture $\hat{P}_t$ for target frames. The details in these guiding textures encourage frame-to-frame consistency by restricting a set of matching exemplar patches. To control the amount of consistency, we use a similar strategy as in [Fišer et al. 2017, 2014], we smooth P_t and $\hat{P}_t$. However, contrary to Fišer et al. who uses simple Gaussian blur, we employ the joint bilateral filter [Eisemann and Durand 2004] with the joint domain G_{seg}, i.e., we avoid blurring over part boundaries which allows to better preserve consistency of individual segments. Increasing the amount of blur in G_{app} reduces restrictions on the synthesis, thereby increasses the amount of temporal flickering in the resulting synthesized target animation.

To generate the guides for the source animation, we simply render the segmentation labels and texture (with the specified amount of smoothing) for every stylized frame $S_s(i)$. For the target frames, we apply the deformations $\hat{P}_r$ and $\hat{P}_d$ to the template puppet P and warp the puppet to the stylized pose using the skeleton obtained in the motion stylization step. We then render the segmentation labels for G_{seg} and the smoothed texture $\hat{P}_t$ for G_{app}. Finally, we run the synthesis using StyLit [Fišer et al. 2016] to produce the final stylized target frames (see Fig. 9).

4 RESULTS

We implemented our approach using a combination of C++ and CUDA. We set $N = 4$ in all our experiments. To smoothen the texture using joint bilateral filter for the appearance guide G_{app}, we set $\sigma_{space} = 5$ and $\sigma_{intensity} = 1$. For the appearance transfer, the segmentation guide G_{seg} has weight 2 and G_{app} is set to 1. For the previously published methods utilized in our pipeline, we set parameters according to recommendations in the corresponding papers.

On a quad-core CPU (Core i7, 2.7 GHz, 16 GB RAM), the analysis phase (namely the registration) takes on average 15 seconds per frame (6 seconds for ARAP registration, 9 seconds for elastic registration). Synthesizing new target animation frames takes roughly 9 seconds per frame (1 second for the motion synthesis, 8 seconds for the appearance transfer). The appearance transfer is parallelized on the GPU (GeForce GTX 750 Ti) using CUDA. Moreover, every animation frame can be synthesized independently, i.e., the synthesis process can be executed in parallel on a cluster.

To assess the effectiveness of our method, we asked an artist to prepare a set of hand-drawn exemplars for different skeletal motions selected from the CMU motion capture database[1] (walking, running, jumping, and window cleaning) using different artistic media (watercolor, pencil, and chalk, see Fig. 10 and 14). Then we selected a set of target sequences from the same motion capture database that have similar overall types of movement as the source animations, but different detailed characteristics. For instance, we include slower, faster and "sneaky" walking motions, and sequences that combine running and jumping motions. We also tested slow motion versions of the source skeletal animations to demonstrate that our technique can also be used for inbetweening. Figures 1, 12, 13, and 14 show static frames from some of our results, more synthesized animations can be found in the supplementary video.

Overall, the results demonstrate that our method successfully captures important aspects of the appearance and motion stylization from the different source examples. For example, the appearance synthesis preserves important characteristics of used artistic media including color variations in the water color style, the high-frequency texture in the chalk renderings, and fine shading in the pencil drawings. These characteristics persist throughout the target animations, even when the pose is significantly different from any of the example frames. The artist also added several motion

[1]http://mocap.cs.cmu.edu/

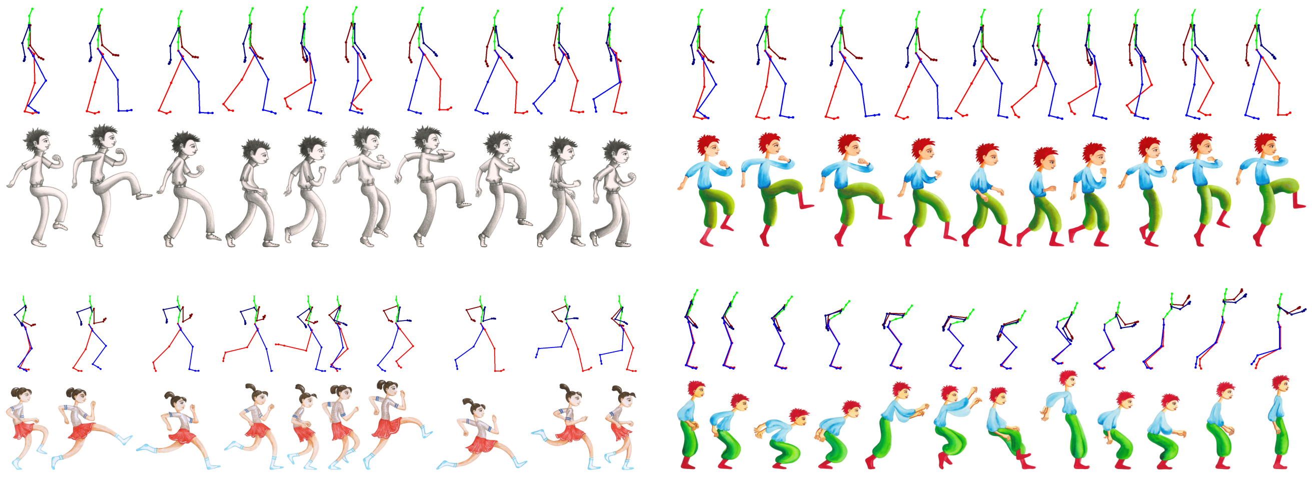


Fig. 10. An overview of exemplar animations created by an artist which we used for most results presented in this paper and in the supplementary video. In each example, we show source skeletal animation (top) and its stylized hand-colored counterpart (bottom). Style exemplars: © *Zuzana Studená*

stylizations, such as the exaggerated arm swings and knee raises in the walking motions, and the secondary effects (e.g., squash and stretch) in the jumping and running animations. Our technique transfers these characteristics to the new target motions, as shown, e.g., in Fig. 1.

Our method has several components that together contribute to the quality of the final synthesized animation. To demonstrate the impact of these components, we generated comparison where we add key steps in our pipeline (ARAP deformation, residual deformation, replacing static textures with blended textures, and appearance synthesis) one-by-one, starting from a simple skeleton-based deformation of the source puppet as the baseline. We also generate results with different amounts of temporal coherence by modifying the strength of the joint bilateral blur in the guidance texture. Please refer to our supplemental videos to see these comparisons.

5 LIMITATIONS AND FUTURE WORK

Our results demonstrate that the proposed method can effectively transfer a range of stylizations to new target motions. However, the technique as it stands does have some limitations.

Motion constraints. The current version of our method does not enforce explicit constraints on the stylized target motion. As a result, artifacts like foot slip or over-exaggerated bending of joints are possible (see Fig. 11, left). It would be a relatively straightforward extension to preserve such constraints by adjusting the stylized target skeletal pose after we apply the per-frame pose deformation $\hat{P}_p(k)$.

Sub-skeleton matching. When finding the closest matching source sub-sequence to a given target sub-sequence, we currently incorporate all skeletal joints into the similarity metric. A possible extension for future work would be to consider only partial matches, e.g., to find separate sub-sequence matches for the upper and lower parts of the skeleton. This could provide more flexibility in adapting existing animation exemplars to a larger variety of target motions.

Out-of-plane motions. There are two challenges in handling out-of-plane motions with our method. First, since we project 3D skeletal poses to 2D representations, out-of-plane motions can introduce ambiguities in the search phase of the motion synthesis step (see Fig. 11, right). For example, rotating an arm towards the camera may have a similar 2D projection as rotating away from the camera, which can make it hard to automatically select the appropriate source sub-sequence to use for synthesis. To address this, we can extend our approach to use the 3D skeletal information in the source and target sequences. The second challenge involves out-of-plane motions that do not preserve a consistent depth order across the layered parts (e.g., a pirouette). Handling such motions is an interesting direction for future work.

Reducing input requirements. Our approach enables artists to leverage a relatively small set of hand-colored frames to synthesize many new target motions. However, there are opportunities

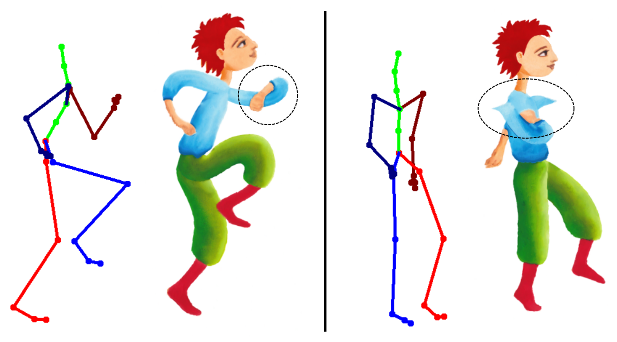


Fig. 11. Limitations: Our method does not enforce explicit constraints on the stylized target motion which may produce over-exaggerated bending of limbs (left). Combined with out-of-plane motions, the deformation may become highly inconsistent and produce visible artifacts (right).

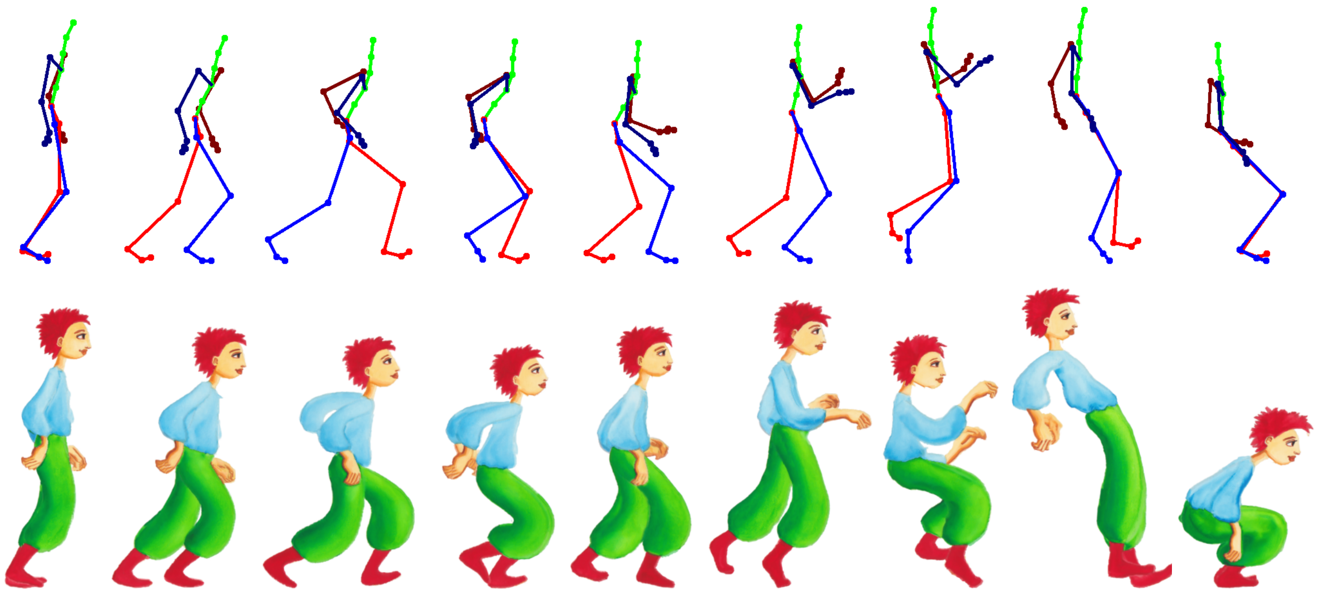


Fig. 12. An example of animation synthesized using our method: target skeletal animation (top), resulting synthesis (bottom). See the original style exemplar in Fig. 10.

to further reduce the input requirements. For example, rather than stylizing every frame of the source skeletal motion, perhaps artists could choose a few key frames to provide hand-colored examples. To support this reduced input, the analysis phase of our framework could potentially interpolate the intervening frames using a guided synthesis method similar to what we currently use to generate stylized target frames. In addition, we could try to augment our existing puppet registration method to avoid the need for a segmented version of each stylized source frame.

Inconsistent motion or skeletal structure. In theory, an artist can provide any pose stylization to the input sequence (e.g., mapping a jump motion to a walking sequence or using artwork that has notably different structure from the original skeleton). However, in this situation the closest match is typically very different and thus the algorithm may produce an N-way morph that is far from the expected shape prescribed by the target skeletal pose (e.g., over-exaggerated stretching). In such situations, the artist may need to provide additional stylization frames that capture the desired pose.

6 CONCLUSION

In this paper, we presented ToonSynth, a novel method for synthesizing hand-colored cartoon animations for target skeletal motions. Our approach leverages artist-created exemplars drawn in reference to source skeletal motions. We create a style-aware puppet that encodes the artist-specific stylization into a skeletal pose, coarse as-rigid-as-possible warp, fine elastic deformation, and texture. Using this represetation we can transfer stylization to many new motions by generating guiding channels that capture basic motion properties as well as provide control over the amount temporal dynamics and are used to produce the final appearance using guided patch-based synthesis. This approach enables us to provide the look and feel of hand-colored animation where each frame is drawn independently from scratch.

ACKNOWLEDGMENTS

We would like to thank Zuzana Studená for preparation of hand-drawn exemplars, Ondřej Jamriška for help on guided texture synthesis part, and all anonymous reviewers for their fruitful comments and suggestions. This research began as an internship by Marek Dvorožňák at Adobe. It was funded by Adobe and has been supported by the Technology Agency of the Czech Republic under research program TE01020415 (V3C – Visual Computing Competence Center), by the Grant Agency of the Czech Technical University in Prague, grant No. SGS16/237/OHK3/3T/13 (Research of Modern Computer Graphics Methods), by Research Center for Informatics No. CZ.02.1.01/0.0/0.0/16_019/0000765, and by the Fulbright Commission in the Czech Republic.

REFERENCES

Rahul Arora, Ishan Darolia, Vinay Namboodiri, Karan Singh, and Adrien Bousseau. 2017. SketchSoup: Exploratory Ideation Using Design Sketches. *Computer Graphics Forum* 36, 8 (2017), 302–312.
Yunfei Bai, Danny M Kaufman, Karen Liu, and Jovan Popović. 2016. Artist-directed dynamics for 2D animation. *ACM Transactions on Graphics* 35, 4 (2016), 145.
William Baxter and Ken-ichi Anjyo. 2006. Latent Doodle Space. *Computer Graphics Forum* 25, 3 (2006), 477–485.
William Baxter, Pascal Barla, and Ken Anjyo. 2009. N-way morphing for 2D animation. *Journal of Visualization and Computer Animation* 20, 2–3 (2009), 79–87.
Pierre Bénard, Forrester Cole, Michael Kass, Igor Mordatch, James Hegarty, Martin Sebastian Senn, Kurt Fleischer, Davide Pesare, and Katherine Breeden. 2013. Stylizing animation by example. *ACM Transactions on Graphics* 32, 4 (2013), 119.
Mikhail Bessmeltsev, Nicholas Vining, and Alla Sheffer. 2016. Gesture3D: posing 3D characters via gesture drawings. *ACM Transactions on Graphics* 35, 6 (2016), 165.
Christoph Bregler, Lorie Loeb, Erika Chuang, and Hrishi Deshpande. 2002. Turning to the Masters: Motion Capturing Cartoons. *ACM Transactions on Graphics* 21, 3

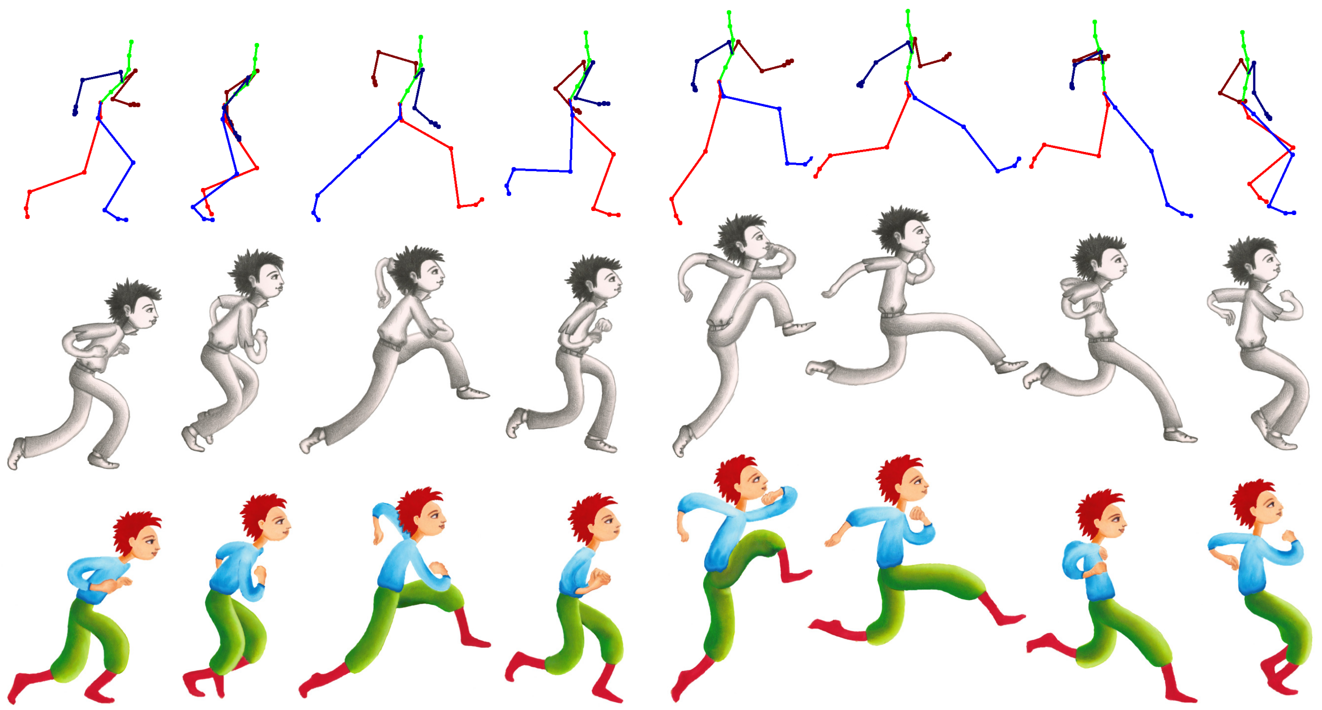


Fig. 13. An example of two hand-colored animations produced using our method (bottom) for the same target skeletal motion (top). See the original pencil and watercolor exemplars in Fig. 10.

(2002), 399–407.
Ian Buck, Adam Finkelstein, Charles Jacobs, Allison Klein, David Salesin, Joshua Seims, Richard Szeliski, and Kentaro Toyama. 2000. Performance-Driven Hand-Drawn Animation. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*. 101–108.
Nestor Burtnyk and Marceli Wein. 1976. Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation. *Commun. ACM* 19, 10 (1976), 564–569.
Edwin Catmull. 1978. The Problems of Computer-Assisted Animation. 12, 3 (1978), 348–353.
James Davis, Maneesh Agrawala, Erika Chuang, Zoran Popovic, and David Salesin. 2003. A sketching interface for articulated figure animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 320–328.
Marek Dvorožňák, Pierre Bénard, Pascal Barla, Oliver Wang, and Daniel Sýkora. 2017. Example-Based Expressive Animation of 2D Rigid Bodies. *ACM Transactions on Graphics* 36, 4, Article 127 (2017).
Elmar Eisemann and Frédo Durand. 2004. Flash photography enhancement via intrinsic relighting. *ACM Transactions on Graphics* 23, 3 (2004), 673–678.
Jakub Fišer, Ondřej Jamriška, Michal Lukáč, Eli Shechtman, Paul Asente, Jingwan Lu, and Daniel Sýkora. 2016. StyLit: Illumination-guided Example-based Stylization of 3D Renderings. *ACM Transactions on Graphics* 35, 4 (2016), 92.
Jakub Fišer, Ondřej Jamriška, David Simons, Eli Shechtman, Jingwan Lu, Paul Asente, Michal Lukáč, and Daniel Sýkora. 2017. Example-Based Synthesis of Stylized Facial Animations. *ACM Transactions on Graphics* 36, 4, Article 155 (2017).
Jakub Fišer, Michal Lukáč, Ondřej Jamriška, Martin Čadík, Yotam Gingold, Paul Asente, and Daniel Sýkora. 2014. Color Me Noisy: Example-Based Rendering of Hand-Colored Animations with Temporal Noise Control. *Computer Graphics Forum* 33, 4 (2014), 1–10.
Ben Glocker, Nikos Komodakis, Georgios Tziritas, Nassir Navab, and Nikos Paragios. 2008. Dense Image Registration Through MRFs And Efficient Linear Programming. *Medical Image Analysis* 12, 6 (2008), 731–741.
William van Haevre, Fabian di Fiore, and Frank van Reeth. 2005. Uniting Cartoon Textures with Computer Assisted Animation. In *Proceedings of International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*. 245–253.
Aaron Hertzmann, Charles E Jacobs, Nuria Oliver, Brian Curless, and David H Salesin. 2001. Image analogies. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*. ACM, 327–340.
Alexander Hornung, Ellen Dekkers, and Leif Kobbelt. 2007. Character Animation from 2D Pictures and 3D Motion Data. *ACM Transactions on Graphics* 26, 1 (2007).
Eakta Jain, Yaser Sheikh, and Jessica Hodgins. 2009. Leveraging the talent of hand animators to create three-dimensional animation. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 93–102.
Ben Jones, Jovan Popovic, James McCann, Wilmot Li, and Adam Bargteil. 2015. Dynamic sprites: Artistic authoring of interactive animations. *Journal of Visualization and Computer Animation* 26, 2 (2015), 97–108.
Christina de Juan and Bobby Bodenheimer. 2004. Cartoon Textures. In *Proceedings of Eurographics Symposium on Computer Animation*. 267–276.
Christina de Juan and Bobby Bodenheimer. 2006. Re-using traditional animation: Methods for semi-automatic segmentation and inbetweening. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 223–232.
Rubaiat Habib Kazi, Tovi Grossman, Nobuyuki Umetani, and George Fitzmaurice. 2016. Motion Amplifiers: Sketching Dynamic Illustrations Using the Principles of 2D Animation. In *Proceedings of ACM Conference on Human Factors in Computing Systems*. 4599–4609.
Alexander Kort. 2002. Computer Aided Inbetweening. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*. 125–132.
Lucas Kovar, Michael Gleicher, and Frédéric Pighin. 2002. Motion Graphs. *ACM Transactions on Graphics* 21, 3 (2002), 473–482.
Vivek Kwatra, Irfan Essa, Aaron Bobick, and Nipun Kwatra. 2005. Texture optimization for example-based synthesis. *ACM Transactions on Graphics* 24, 3 (2005), 795–802.
John Lasseter. 1987. Principles of Traditional Animation Applied to 3D Computer Animation. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*. 35–44.
Seungyong Lee, George Wolberg, and Sung Yong Shin. 1998. Polymorph: Morphing Among Multiple Images. *IEEE Computer Graphics and Applications* 18, 1 (1998), 58–71.
Sun-Young Lee, Jong-Chul Yoon, Ji-Yong Kwon, and In-Kwon Lee. 2012. CartoonModes: Cartoon Stylization of Video Objects Through Modal Analysis. *Graphical Models* 74, 2 (2012), 51–60.
Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. 2017. VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera. *ACM*

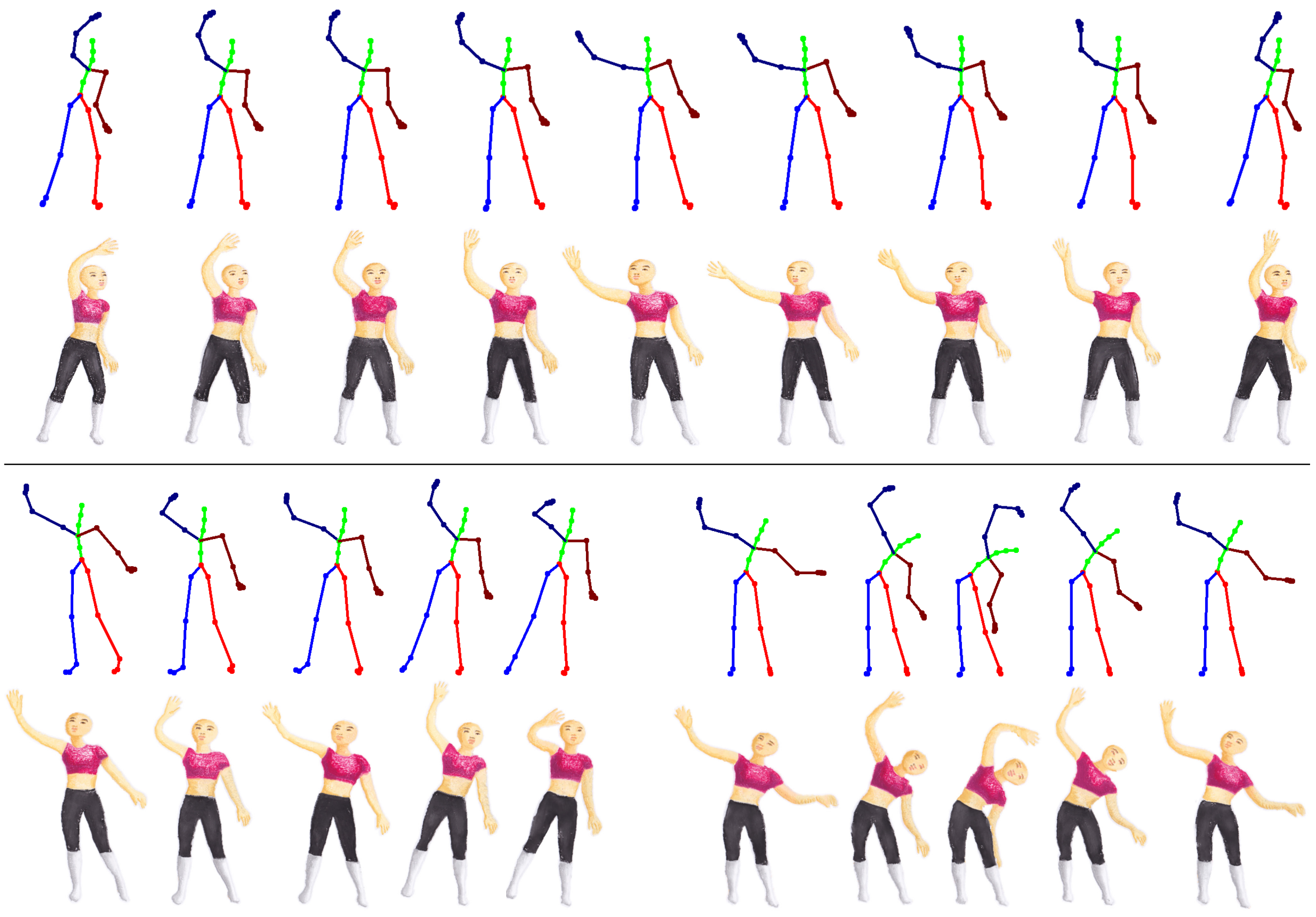


Fig. 14. A hand-colored exemplar animation created by an artist for the specific source skeletal motion (top) has been used to produce the animations at the bottom for a different target skeletal motions. Style exemplars: © *Zuzana Studená*

Transactions on Graphics 36, 4 (2017), 44:1–44:14.
Johannes Schmid, Robert Sumner, Huw Bowles, and Markus Gross. 2010. Programmable Motion Effects. *ACM Transactions on Graphics* 29, 4 (2010), 57.
Daniel Sýkora, Mirela Ben-Chen, Martin Čadík, Brian Whited, and Maryann Simmons. 2011. TexToons: Practical Texture Mapping for Hand-drawn Cartoon Animations. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*. 75–83.
Daniel Sýkora, Jan Buriánek, and Jiří Žára. 2005. Sketching Cartoons by Example. In *Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling*. 27–34.
Daniel Sýkora, John Dingliana, and Steven Collins. 2009. As-Rigid-As-Possible Image Registration for Hand-Drawn Cartoon Animations. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*. 25–33.
Daniel Sýkora, Ladislav Kavan, Martin Čadík, Ondřej Jamriška, Alec Jacobson, Brian Whited, Maryann Simmons, and Olga Sorkine-Hornung. 2014. Ink-and-Ray: Bas-Relief Meshes for Adding Global Illumination Effects to Hand-Drawn Characters. *ACM Transactions on Graphics* 33, 2 (2014), 16.
Daniel Sýkora, David Sedlacek, Sun Jinchao, John Dingliana, and Steven Collins. 2010. Adding Depth to Cartoons Using Sparse Depth (In)equalities. *Computer Graphics Forum* 29, 2 (2010), 615–623.
Cedric Vanaken, Chris Hermans, Tom Mertens, Fabian Di Fiore, Philippe Bekaert, and Frank Van Reeth. 2008. Strike a Pose: Image-Based Pose Synthesis. In *Proceedings of the Conference on Vision, Modeling and Visualization*. 131–138.
Jue Wang, Steven Drucker, Maneesh Agrawala, and Michael Cohen. 2006. The Cartoon Animation Filter. *ACM Transactions on Graphics* 25, 3 (2006), 1169–1173.
Xun Wang, Wenwu Yang, Haoyu Peng, and Guozheng Wang. 2013. Shape-aware skeletal deformation for 2D characters. *The Visual Computer* 29, 6-8 (2013), 545–553.
Yonatan Wexler, Eli Shechtman, and Michal Irani. 2007. Space-Time Completion of Video. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29, 3 (2007), 463–476.
Brian Whited, Gioacchino Noris, Maryann Simmons, Robert Sumner, Markus Gross, and Jarek Rossignac. 2010. BetweenIT: An Interactive Tool for Tight Inbetweening. *Computer Graphics Forum* 29, 2 (2010), 605–614.
Nora Willett, Wilmot Li, Jovan Popovic, Floraine Berthouzoz, and Adam Finkelstein. 2017. Secondary Motion for Performed 2D Animation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology (UIST '17)*. ACM, New York, NY, USA, 97–108.
Wenwu Yang. 2017. Context-Aware Computer Aided Inbetweening. *IEEE Transactions on Visualization and Computer Graphics* (2017).
Chih-Kuo Yeh, Shi-Yang Huang, Pradeep Kumar Jayaraman, Chi-Wing Fu, and Tong-Yee Lee. 2017. Interactive High-Relief Reconstruction for Organic and Double-Sided Objects from a Photo. *IEEE Transactions on Visualization and Computer Graphics* 23, 7 (2017), 1796–1808.
Lei Zhang, Hua Huang, and Hongbo Fu. 2012. EXCOL: An EXtract-and-COmplete Layering Approach to Cartoon Animation Reusing. *IEEE Transactions on Visualization and Computer Graphics* 18, 7 (2012), 1156–1169.
Yufeng Zhu, Jovan Popović, Robert Bridson, and Danny Kaufman. 2017. Planar Interpolation with Extreme Deformation, Topology Change and Dynamics. *ACM Transactions on Graphics* 36, 6 (2017), 213.

Chapter 7

Conclusion

This thesis has presented four novel approaches contributing to bridging the gap between computer-generated and hand-drawn animation and advancing the current state-of-the-art. These approaches allow for raw user input preprocessing and enable to significantly reduce demands on the production of hand-drawn animation while retaining its essential characteristics and also making it more appealing to the contemporary audience. In this chapter, we summarize the contributions of our work, briefly discuss related concurrent work and propose directions for future work.

7.1 Summary

In Chapter 3, we presented several methods for preprocessing of raw user input – a set of efficient methods for reconstruction of a temporal creation history from a video capturing the process of a digital or real-world artwork creation, and a technique for cleaning up the video of unwanted real-world objects such as painter's hand. We presented solutions suitable for both digital and real-world artwork, which utilize the standard Porter-Duff and physically motivated Kubelka-Munk blending operations, respectively. Our methods are designed to extract translucent layers which increase utility for later applications.

Chapter 4 described our technique enabling reconstruction of part-based high-relief models from a single hand-drawn image. We presented a unified formulation of the reconstruction problem that expresses inflation and depth positioning of individual parts using single non-linear energy and enables seamlessly interconnected object parts. Furthermore, we proposed a significantly faster approximate method yielding results of comparable quality.

In Chapter 5, we proposed an example-based approach for simplifying the production of hand-drawn animations of colliding stylized rigid body objects. We introduced the animation analogy concept which extends the established analogy approach for style transfer between images to animations. From a few hand-drawn frames provided by an artist, our approach successfully transfers both motion and appearance stylization to a longer and more complex animation. In addition to this, we demonstrated an interactive application which enables iterative work and refinement of the result on-the-fly.

Finally, Chapter 6 introduced ToonSynth, an example-based method for facilitating the production of stylized hand-colored character animations, which proposes a way of applying the animation analogy concept to character animation. We introduced a style-aware puppet that encodes motion and appearance characteristics and a technique of its extraction from an input hand-drawn exemplar animation. We also proposed a synthesis method that utilizes the extracted puppets. Compared to the technique for the rigid body animation analogy, we simplified the parametric synthesis technique by using more general fixed-length overlapping sub-sequences instead of ones defined around events, and employed a non-parametric synthesis method to faithfully transfer the hand-colored appearance of the input exemplars and to introduce a user-controllable amount of temporal dynamics.

7.2 Concurrent and Future Work

Following up on our work, Tan et al. [2016] proposed a method for tackling the challenging problem of reconstruction of layers, corresponding to a user-specified palette, from a single image. Their approach has been recently extended to increase the reconstruction efficiency significantly [Tan et al. 2018]. Even though this is an exciting direction of research enabling impressive applications, in the future, we would like to focus more on exploring applications of the spatiotemporal volume reconstructed from the time-lapse paintings. The information about a shape or texture of hidden overlapping object-parts that is present in the volume may serve as a valuable input for our 3D reconstruction and animation synthesis techniques. The volume may also be informative about the temporal behavior of physical media during the painting process. An interesting future work would be to utilize such information for synthesis.

Entem et al. [2018] presented an automatic approach for recovering depth-ordered structure in a contour drawing of an organic object. They allow for automatic extraction of possibly occluded object-parts and their ordering in depth. Our high-relief reconstruction method could benefit from employing their approach to eliminate the need for the additional user input which would decrease the number of user distractions even more. Recently, Ramos et al. [2018] proposed a method for 3D reconstruction from side-view sketches which follows-up on the work of Entem et al. [2015] to which we compared our reconstruction method. Similar to our work, the technique of Ramos et al. [2018] allows for more precise outline reproduction then [Entem et al. 2015], but unlike our approach, theirs does not support seamless interconnections between object-parts.

Recently, artificial neural network approaches have been applied to various challenging problems yielding impressive results. C. Li et al. [2018] proposed a single-view freeform surface modeling method employing convolutional neural networks. Their technique enables the user to incrementally explore the emerging shape by drawing intuitive and sparse 2D sketches. To deform the shape in depth, they introduce depth strokes or allow profile-view sketches which, however, requires viewing the shape from a different viewpoint. Their method supports the creation of full 3D models. This process, however, requires additional manual intervention in the 3D domain.

In the future, we plan to extend our high-relief reconstruction technique to allow for full 3D model reconstruction. To obtain full 3D models, Feng et al. [2016] proposed a simple extension of the semi-elliptical inflation method [Sýkora et al. 2014] that utilizes mirroring. Reconstruction of full 3D models with seamless interconnections between object-parts from a single image remains an unsolved problem.

Another interesting direction for future work is an extension of our reconstruction method to generate animation-ready models, or allowing for incremental modeling combined with automatic rigging [Borosán et al. 2012] or modeling and animation interleaving workflow [Jin et al. 2015].

Our hand-drawn animation synthesis methods make a few simplifying assumptions to make the synthesis feasible. The methods, for instance, assume that one-to-one correspondence between frames of source guidance and hand-drawn exemplar animations is established, and that object's motion stays in camera plane. Adapting our methods for scenarios where these assumptions are not met would enable support for synthesis of more types of animation.

Our synthesis techniques could be improved to require even less amount of user input. Instead of a sequence of consecutive hand-drawn frames, the user could draw only several keyframes, and the rest of the input sequence could be synthesized based on the keyframes. To also reduce the number of fully hand-colored frames, the available exemplars could be utilized to generate missing texture exemplars by applying geometric transformations as in [Lukáč et al. 2013].

Approaches which employ convolutional neural networks for image synthesis have proven to be able to generalize the input data very well. They could also be employed for the hand-colored animation synthesis. These techniques, however, require an extensive learning dataset. Manual acquisition of such dataset for hand-drawn animation would be extremely labor intensive. Our synthesis methods could be employed for such a task, i.e., to generate a large number of animation frames from a small amount of user input.

References

Amati, C. and Brostow, G. J. (2010). “Modeling 2.5D Plants from Ink Paintings”. In *Proceedings of Eurographics Symposium on Sketch-Based Interfaces and Modeling Symposium*, pp. 41–48.

Arora, R., Darolia, I., Namboodiri, V., Singh, K., and Bousseau, A. (2017). “Sketch-Soup: Exploratory Ideation Using Design Sketches”. *Computer Graphics Forum* 36.8, pp. 302–312.

Bai, Y., Kaufman, D. M., Liu, K., and Popović, J. (2016). “Artist-directed dynamics for 2D animation”. *ACM Transactions on Graphics* 35.4, p. 145.

Barzel, R. and Barr, A. (1988). “A Modeling System Based on Dynamic Constraints”. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 179–188.

Baxter, W. and Anjyo, K. (2006). “Latent Doodle Space”. *Computer Graphics Forum* 25.3, pp. 477–485.

Baxter, W., Barla, P., and Anjyo, K. (2009). “N-way morphing for 2D animation”. *Journal of Visualization and Computer Animation* 20.2–3, pp. 79–87.

Belhumeur, P. N., Kriegman, D. J., and Yuille, A. L. (1999). “The Bas-Relief Ambiguity”. *International Journal of Computer Vision* 35.1, pp. 33–44.

Bénard, P., Cole, F., Kass, M., Mordatch, I., Hegarty, J., Senn, M. S., Fleischer, K., Pesare, D., and Breeden, K. (2013). “Stylizing animation by example”. *ACM Transactions on Graphics* 32.4, p. 119.

Berthouzoz, F., Li, W., Dontcheva, M., and Agrawala, M. (2011). “A Framework for Content-adaptive Photo Manipulation Macros: Application to Face, Landscape, and Global Manipulations”. *ACM Transactions on Graphics* 30.5, p. 120.

Bessmeltsev, M., Chang, W., Vining, N., Sheffer, A., and Singh, K. (2015). “Modeling Character Canvases from Cartoon Drawings”. *ACM Transactions on Graphics* 34.5, p. 162.

Bessmeltsev, M., Vining, N., and Sheffer, A. (2016). “Gesture3D: posing 3D characters via gesture drawings”. *ACM Transactions on Graphics* 35.6, p. 165.

Bonanni, L., Xiao, X., Hockenberry, M., Subramani, P., Ishii, H., Seracini, M., and Schulze, J. (2009). “Wetpaint: Scraping Through Multi-layered Images”. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 571–574.

Borosán, P., Jin, M., DeCarlo, D., Gingold, Y., and Nealen, A. (2012). “RigMesh: Automatic Rigging for Part-Based Shape Modeling and Deformation”. *ACM Transactions on Graphics* 31.6, p. 198.

Bregler, C., Loeb, L., Chuang, E., and Deshpande, H. (2002). “Turning to the Masters: Motion Capturing Cartoons”. *ACM Transactions on Graphics* 21.3, pp. 399–407.

Buck, I., Finkelstein, A., Jacobs, C., Klein, A., Salesin, D., Seims, J., Szeliski, R., and Toyama, K. (2000). “Performance-Driven Hand-Drawn Animation”. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, pp. 101–108.

Burtnyk, N. and Wein, M. (1976). “Interactive Skeleton Techniques for Enhancing Motion Dynamics in Key Frame Animation”. *Communications of the ACM* 19.10, pp. 564–569.

Catmull, E. (1978). “The Problems of Computer-Assisted Animation”. 12.3, pp. 348–353.

Chen, H.-T., Grossman, T., Wei, L.-Y., Schmidt, R. M., Hartmann, B., Fitzmaurice, G., and Agrawala, M. (2014). “History Assisted View Authoring for 3D Models”. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2027–2036.

Chen, H.-T., Wei, L.-Y., and Chang, C.-F. (2011). “Nonlinear Revision Control for Images”. *ACM Transactions on Graphics* 30.4, p. 105.

Chen, H.-T., Wei, L.-Y., Hartmann, B., and Agrawala, M. (2012). *Data-driven History List for Image Editing.* Tech. rep. TR-2012-07. The University of Hong Kong.

Chen, T., Zhu, Z., Shamir, A., Hu, S.-M., and Cohen-Or, D. (2013). “3-Sweep: Extracting Editable Objects from a Single Photo”. *ACM Transactions on Graphics* 32.6, p. 195.

Chenney, S., Pingel, M., Iverson, R., and Szymanski, M. (2002). “Simulating Cartoon Style Animation”. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, pp. 133–138.

de Juan, C. and Bodenheimer, B. (2004). “Cartoon Textures”. In *Proceedings of Eurographics Symposium on Computer Animation*, pp. 267–276.

de Juan, C. and Bodenheimer, B. (2006). “Re-using traditional animation: Methods for semi-automatic segmentation and inbetweening”. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 223–232.

Coros, S., Martin, S., Thomaszewski, B., Schumacher, C., Sumner, R., and Gross, M. (2012). “Deformable Objects Alive!” *ACM Transactions on Graphics* 31.4, p. 69.

Davis, J., Agrawala, M., Chuang, E., Popovic, Z., and Salesin, D. (2003). “A sketching interface for articulated figure animation”. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 320–328.

Denning, J. D. and Pellacini, F. (2013). “MeshGit: Diffing and Merging Meshes for Polygonal Modeling”. *ACM Transactions on Graphics* 32.4, p. 35.

Entem, E., Barthe, L., Cani, M.-P., Cordier, F., and Panne, M. van de (2015). “Modeling 3D animals from a side-view sketch”. *Computers & Graphics* 46, pp. 221–230.

Entem, E., Parakkat, A. D., Cani, M.-P., and Barthe, L. (2018). “Structuring and Layering Contour Drawings of Organic Shapes”. In *Proceedings of the Joint Symposium on Computational Aesthetics and Sketch-Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering*. Expressive ’18. ACM, 4:1–4:14.

Farid, H. and Adelson, E. H. (1999). “Separating reflections from images by use of independent component analysis”. *Journal of the Optical Society of America A* 16.9, pp. 2136–2145.

Feng, L., Yang, X., Xiao, S., and Jiang, F. (2016). “An Interactive 2D-to-3D Cartoon Modeling System”. In *Proceedings of International Conference on Technologies for E-Learning and Digital Entertainment*, pp. 193–204.

Fišer, J., Jamriška, O., Lukáč, M., Shechtman, E., Asente, P., Lu, J., and Sýkora, D. (2016). “StyLit: Illumination-guided Example-based Stylization of 3D Renderings”. *ACM Transactions on Graphics* 35.4, p. 92.

Fišer, J., Jamriška, O., Simons, D., Shechtman, E., Lu, J., Asente, P., Lukáč, M., and Sýkora, D. (2017). “Example-Based Synthesis of Stylized Facial Animations”. *ACM Transactions on Graphics* 36.4.

Fišer, J., Lukáč, M., Jamriška, O., Čadík, M., Gingold, Y., Asente, P., and Sýkora, D. (2014). “Color Me Noisy: Example-Based Rendering of Hand-Colored Animations with Temporal Noise Control”. *Computer Graphics Forum* 33.4, pp. 1–10.

Fu, H., Zhou, S., Liu, L., and Mitra, N. J. (2011). “Animated Construction of Line Drawings”. *ACM Transactions on Graphics* 30.6, p. 133.

Garcia, M., Dingliana, J., and O'Sullivan, C. (2007). "A Physically Based Deformation Model for Interactive Cartoon Animation". In *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2007).*

Gatys, L. A., Ecker, A. S., and Bethge, M. (2016). "Image style transfer using convolutional neural networks". In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2414–2423.

Geiger, D., Pao, H., and Rubin, N. (1998). "Salient and Multiple Illusory Surfaces". In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition.* CVPR '98. IEEE Computer Society, pp. 118–124.

Gingold, Y., Igarashi, T., and Zorin, D. (2009). "Structured Annotations for 2D-to-3D Modeling". *ACM Transactions on Graphics* 28.5, p. 148.

Grossman, T., Matejka, J., and Fitzmaurice, G. (2010). "Chronicle: Capture, Exploration, and Playback of Document Workflow Histories". In *Proceedings of ACM Symposium on User Interface Software and Technology*, pp. 143–152.

Haller, M., Hanl, C., and Diephuis, J. (2004). "Non-photorealistic Rendering Techniques for Motion in Computer Games". *Computers in Entertainment* 2.4.

Hertzmann, A., Jacobs, C. E., Oliver, N., Curless, B., and Salesin, D. H. (2001). "Image analogies". In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques.* ACM, pp. 327–340.

Hornung, A., Dekkers, E., and Kobbelt, L. (2007). "Character Animation from 2D Pictures and 3D Motion Data". *ACM Transactions on Graphics* 26.1.

Hu, S.-M., Xu, K., Ma, L.-Q., Liu, B., Jiang, B.-Y., and Wang, J. (2013). "Inverse Image Editing: Recovering a Semantic Editing History from a Before-and-after Image Pair". *ACM Transactions on Graphics* 32.6, p. 194.

Igarashi, T. and Hughes, J. F. (2003). "Smooth Meshes for Sketch-based Freeform Modeling". In *Proceedings of Symposium on Interactive 3D Graphics*, pp. 139–142.

Igarashi, T., Matsuoka, S., and Tanaka, H. (1999). "Teddy: A Sketching Interface for 3D Freeform Design". In *SIGGRAPH Conference Proceedings*, pp. 409–416.

Jain, E., Sheikh, Y., and Hodgins, J. (2009). "Leveraging the talent of hand animators to create three-dimensional animation". In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 93–102.

Jamriška, O., Fišer, J., Asente, P., Lu, J., Shechtman, E., and Sýkora, D. (2015). "LazyFluids: Appearance Transfer for Fluid Animations". *ACM Transactions on Graphics* 34.4, p. 92.

Jin, M., Gopstein, D., Gingold, Y., and Nealen, A. (2015). “AniMesh: Interleaved Animation, Modeling, and Editing”. *ACM Transactions on Graphics* 34.6, 207:1–207:8.

Jones, A. and Oliff, J. (2006). *Thinking Animation: Bridging the Gap Between 2D and CG*. Thomson Course Technology PTR.

Jones, B., Popovic, J., McCann, J., Li, W., and Bargteil, A. (2015). “Dynamic sprites: Artistic authoring of interactive animations”. *Journal of Visualization and Computer Animation* 26.2, pp. 97–108.

Jones, B., Thuerey, N., Shinar, T., and Bargteil, A. (2016). “Example-based Plastic Deformation of Rigid Bodies”. *ACM Transactions on Graphics* 35.4, p. 34.

Karsch, K., Golparvar-Fard, M., and Forsyth, D. (2014). “ConstructAide: Analyzing and Visualizing Construction Sites Through Photographs and Building Models”. *ACM Transactions on Graphics* 33.6, p. 176.

Kazi, R. H., Chevalier, F., Grossman, T., and Fitzmaurice, G. (2014a). “Kitty: Sketching dynamic and interactive illustrations”. In *Proceedings of ACM Symposium on User Interface Software and Technology*, pp. 395–405.

Kazi, R. H., Chevalier, F., Grossman, T., Zhao, S., and Fitzmaurice, G. (2014b). “Draco: Bringing life to illustrations with kinetic textures”. In *Proceedings of ACM Conference on Human Factors in Computing Systems*, pp. 351–360.

Kazi, R. H., Grossman, T., Umetani, N., and Fitzmaurice, G. (2016). “Motion Amplifiers: Sketching Dynamic Illustrations Using the Principles of 2D Animation”. In *Proceedings of ACM Conference on Human Factors in Computing Systems*, pp. 4599–4609.

Kort, A. (2002). “Computer Aided Inbetweening”. In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, pp. 125–132.

Koyama, Y., Takayama, K., Umetani, N., and Igarashi, T. (2012). “Real-time Example-based Elastic Deformation”. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 19–24.

Kubelka, P. (1948). “New Contributions to the Optics of Intensely Light-Scattering Materials. Part I”. *Journal of the Optical Society of America* 38.5, pp. 448–448.

Kwatra, V., Essa, I., Bobick, A., and Kwatra, N. (2005). “Texture optimization for example-based synthesis”. *ACM Transactions on Graphics* 24.3, pp. 795–802.

Lasseter, J. (1987). “Principles of Traditional Animation Applied to 3D Computer Animation”. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 35–44.

Lee, S.-Y., Yoon, J.-C., Kwon, J.-Y., and Lee, I.-K. (2012). “CartoonModes: Cartoon Stylization of Video Objects Through Modal Analysis”. *Graphical Models* 74.2, pp. 51–60.

Li, C., Pan, H., Liu, Y., Sheffer, A., and Wang, W. (2018). “Robust Flow-Guided Neural Prediction for Sketch-Based Freeform Surface Modeling”. *ACM Transactions on Graphics* 37.6, 238:1–238:12.

Li, Y., Gleicher, M., Xu, Y.-Q., and Shum, H.-Y. (2003). “Stylizing Motion with Drawings”. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 309–319.

Liu, X., Mao, X., Yang, X., Zhang, L., and Wong, T.-T. (2013). “Stereoscopizing Cel Animations”. *ACM Transactions on Graphics* 32.6, p. 223.

Lukáč, M., Fišer, J., Bazin, J.-C., Jamriška, O., Sorkine-Hornung, A., and Sýkora, D. (2013). “Painting by Feature: Texture Boundaries for Example-based Image Creation”. *ACM Transactions on Graphics* 32.4, p. 116.

Martin, S., Thomaszewski, B., Grinspun, E., and Gross, M. (2011). “Example-based Elastic Materials”. *ACM Transactions on Graphics* 30.4, p. 72.

Matzen, K. and Snavely, N. (2014). “Scene Chronology”. In *Proceedings of European Conference on Computer Vision*, pp. 615–630.

McCann, J. and Pollard, N. (2009). “Local Layering”. *ACM Transactions on Graphics* 28.3, p. 84.

McCann, J. and Pollard, N. (2012). “Soft Stacking”. *Computer Graphics Forum* 31.2, pp. 469–478.

Nancel, M. and Cockburn, A. (2014). “Causality: A Conceptual Model of Interaction History”. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1777–1786.

Nealen, A., Igarashi, T., Sorkine, O., and Alexa, M. (2007). “FiberMesh: Designing Freeform Surfaces with 3D Curves”. *ACM Transactions on Graphics* 26.3, p. 41.

Ngo, T., Cutrell, D., Dana, J., Donald, B., Loeb, L., and Zhu, S. (2000). “Accessible Animation and Customizable Graphics via Simplicial Configuration Modeling”. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 403–410.

Noble, P. and Tang, W. (2006). “Automatic Expressive Deformations for Stylizing Motion”. In *Proceedings of the 4th International Conference on Computer Graphics and Interactive Techniques in Australasia and Southeast Asia*, pp. 57–63.

Noris, G., Sýkora, D., Shamir, A., Coros, S., Whited, B., Simmons, M., Hornung, A., Gross, M., and Sumner, R. (2012). "Smart Scribbles for Sketch Segmentation". *Computer Graphics Forum* 31.8, pp. 2516–2527.

Noris, G., Sýkora, D., Coros, S., Whited, B., Simmons, M., Hornung, A., Gross, M., and Sumner, R. (2011). "Temporal Noise Control for Sketchy Animation". In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, pp. 93–98.

Porter, T. and Duff, T. (1984). "Compositing Digital Images". *ACM SIGGRAPH Computer Graphics* 18.3, pp. 253–259.

Ramos, S., Trevisan, D. F., Batagelo, H. C., Sousa, M. C., and Gois, J. P. (2018). "Contour-aware 3D reconstruction of side-view sketches". *Computers & Graphics* 77, pp. 97–107.

Ricci, A. (1973). "A constructive geometry for computer graphics". *The Computer Journal* 16.2, pp. 157–160.

Richardt, C., Lopez-Moreno, J., Bousseau, A., Agrawala, M., and Drettakis, G. (2014). "Vectorising Bitmaps into Semi-Transparent Gradient Layers". *Computer Graphics Forum (Proceedings of EGSR)* 33.4, pp. 11–19.

Schmid, J., Sumner, R., Bowles, H., and Gross, M. (2010). "Programmable Motion Effects". *ACM Transactions on Graphics* 29.4, p. 57.

Schmidt, R., Wyvill, B., Sousa, M. C., and Jorge, J. A. (2005). "ShapeShop: Sketch-based Solid Modeling with BlobTrees". In *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, pp. 53–62.

Schneider, R. and Kobbelt, L. (2001). "Geometric fairing of irregular meshes for free-form surface design". *Computer Aided Geometric Design* 18.4, pp. 359–379.

Shtof, A., Agathos, A., Gingold, Y., Shamir, A., and Cohen-Or, D. (2013). "Geosemantic Snapping for Sketch-Based Modeling". *Computer Graphics Forum* 32.2, pp. 245–253.

Smith, A. R. and Blinn, J. F. (1996). "Blue Screen Matting". In *ACM SIGGRAPH Conference Proceedings*, pp. 259–268.

Su, S. L., Paris, S., and Durand, F. (2009). "QuickSelect: History-based Selection Expansion". In *Proceedings of Graphics Interface*, pp. 215–221.

Sýkora, D., Ben-Chen, M., Čadík, M., Whited, B., and Simmons, M. (2011). "TexToons: Practical Texture Mapping for Hand-drawn Cartoon Animations". In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, pp. 75–83.

Sýkora, D., Buriánek, J., and Žára, J. (2005). "Sketching Cartoons by Example". In *Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling*, pp. 27–34.

Sýkora, D., Dingliana, J., and Collins, S. (2009). "As-Rigid-As-Possible Image Registration for Hand-Drawn Cartoon Animations". In *Proceedings of International Symposium on Non-Photorealistic Animation and Rendering*, pp. 25–33.

Sýkora, D., Kavan, L., Čadík, M., Jamriška, O., Jacobson, A., Whited, B., Simmons, M., and Sorkine-Hornung, O. (2014). "Ink-and-Ray: Bas-Relief Meshes for Adding Global Illumination Effects to Hand-Drawn Characters". *ACM Transactions on Graphics* 33.2, p. 16.

Sýkora, D., Sedláček, D., Jinchao, S., Dingliana, J., and Collins, S. (2010). "Adding Depth to Cartoons Using Sparse Depth (In)equalities". *Computer Graphics Forum* 29.2, pp. 615–623.

Szeliski, R., Avidan, S., and Anandan, P. (2000). "Layer extraction from multiple images containing reflections and transparency". In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition.* Vol. 1, pp. 246–253.

Tai, C.-L., Zhang, H., and Fong, J. C.-K. (2004). "Prototype modeling from sketched silhouettes based on convolution surfaces". *Computer Graphics Forum* 23.1, pp. 71–83.

Tan, J., Echevarria, J., and Gingold, Y. (2018). "Efficient palette-based decomposition and recoloring of images via RGBXY-space geometry". *ACM Transactions on Graphics* 37.6, 262:1–262:10.

Tan, J., Lien, J.-M., and Gingold, Y. (2016). "Decomposing Images into Layers via RGB-space Geometry". *ACM Transactions on Graphics* 36.1, 7:1–7:14.

Taubin, G. (1995). "A Signal Processing Approach to Fair Surface Design". In *SIGGRAPH Conference Proceedings*, pp. 351–358.

Thomas, F. and Johnston, O. (1981). *The illusion of life : Disney animation.* New York: Disney Editions.

Tversky, B. (1999). "What does drawing reveal about thinking?" *Visual and Spatial Reasoning in Design*, pp. 93–101.

Van Sommers, P. (1984). *Drawing and cognition: Descriptive and experimental studies of graphic production processes.* Cambridge University Press.

Vanaken, C., Hermans, C., Mertens, T., Fiore, F. D., Bekaert, P., and Reeth, F. V. (2008). "Strike a Pose: Image-Based Pose Synthesis". In *Proceedings of the Conference on Vision, Modeling and Visualization*, pp. 131–138.

VisTrails, I. (2009). *VisTrails Provenance Explorer for Maya.* `http://www.vistrails.com/maya.html`.

Wang, J. and Cohen, M. F. (2008). “Image and video matting: a survey”. *Foundations and Trends® in Computer Graphics and Vision* 3.2, pp. 97–175.

Wang, J., Drucker, S., Agrawala, M., and Cohen, M. (2006). “The Cartoon Animation Filter”. *ACM Transactions on Graphics* 25.3, pp. 1169–1173.

Wang, X., Yang, W., Peng, H., and Wang, G. (2013). “Shape-aware skeletal deformation for 2D characters”. *The Visual Computer* 29.6-8, pp. 545–553.

Wexler, Y., Shechtman, E., and Irani, M. (2007). “Space-Time Completion of Video”. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 29.3, pp. 463–476.

Whited, B., Noris, G., Simmons, M., Sumner, R., Gross, M., and Rossignac, J. (2010). “BetweenIT: An Interactive Tool for Tight Inbetweening”. *Computer Graphics Forum* 29.2, pp. 605–614.

Willett, N., Li, W., Popovic, J., Berthouzoz, F., and Finkelstein, A. (2017). “Secondary Motion for Performed 2D Animation”. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology.* UIST ’17. New York, NY, USA: ACM, pp. 97–108.

van Haevre, W., Fiore, F. di, and Reeth, F. van (2005). “Uniting Cartoon Textures with Computer Assisted Animation”. In *Proceedings of International Conference on Computer Graphics and Interactive Techniques in Australasia and South East Asia*, pp. 245–253.

Williams, L. (1991). “Shading in two dimensions”. In *Proceedings of Graphics Interface ’91.*

Witkin, A. and Kass, M. (1988). “Spacetime Constraints”. In *Proceedings of the 15th Annual Conference on Computer Graphics and Interactive Techniques*, pp. 159–168.

Xing, J., Chen, H.-T., and Wei, L.-Y. (2014). “Autocomplete Painting Repetitions”. *ACM Transactions on Graphics* 33.6, p. 172.

Xing, J., Kazi, R. H., Grossman, T., Wei, L.-Y., Stam, J., and Fitzmaurice, G. (2016). “Energy-Brushes: Interactive Tools for Illustrating Stylized Elemental Dynamics”. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 755–766.

Xing, J., Wei, L.-Y., Shiratori, T., and Yatani, K. (2015). “Autocomplete Hand-drawn Animations”. *ACM Transactions on Graphics* 34.6, p. 169.

Xu, S., Xu, Y., Kang, S. B., Salesin, D. H., Pan, Y., and Shum, H.-Y. (2006). "Animating Chinese Paintings Through Stroke-based Decomposition". *ACM Transactions on Graphics* 25.2, pp. 239–267.

Yang, W. (2018). "Context-Aware Computer Aided Inbetweening". *IEEE Transactions on Visualization and Computer Graphics.*

Yang, W., Seah, H.-S., Chen, Q., Liew, H.-Z., and Sýkora, D. (2018). "FTP-SC: Fuzzy Topology Preserving Stroke Correspondence". *Computer Graphics Forum* 37.8.

Yeh, C.-K., Huang, S.-Y., Jayaraman, P. K., Fu, C.-W., and Lee, T.-Y. (2017). "Interactive High-Relief Reconstruction for Organic and Double-Sided Objects from a Photo". *IEEE Transactions on Visualization and Computer Graphics* 23.7, pp. 1796–1808.

Yeh, C.-K., Jayaraman, P. K., Liu, X., Fu, C.-W., and Lee, T.-Y. (2015). "2.5D Cartoon Hair Modeling and Manipulation". *IEEE Transactions on Visualization and Computer Graphics* 21.3, pp. 304–314.

Zeng, Q., Chen, W., Wang, H., Tu, C., Cohen-Or, D., Lischinski, D., and Chen, B. (2015). "Hallucinating Stereoscopy from a Single Image". *Computer Graphics Forum* 34.2, pp. 1–12.

Zhang, L., Huang, H., and Fu, H. (2012). "EXCOL: An EXtract-and-COmplete Layering Approach to Cartoon Animation Reusing". *IEEE Transactions on Visualization and Computer Graphics* 18.7, pp. 1156–1169.

Zhu, Y., Popović, J., Bridson, R., and Kaufman, D. (2017). "Planar Interpolation with Extreme Deformation, Topology Change and Dynamics". *ACM Transactions on Graphics* 36.6, p. 213.

Zongker, D. E., Werner, D. M., Curless, B., and Salesin, D. H. (1999). "Environment Matting and Compositing". In *ACM SIGGRAPH Conference Proceedings*, pp. 205–214.

Appendix A

Author's Publications

Publications Related to the Thesis

In Journals with Impact Factor

The following publications were co-authored by the author of this thesis and published in ACM Transactions on Graphics journal which has an average impact factor of 4.23 for the year range 2015–2017 (based on Journal Citation Reports database).

Tan, J., Dvorožňák, M., Sýkora, D., and Gingold, Y. [2015]. "Decomposing Time-Lapse Paintings into Layers". *ACM Transactions on Graphics* 34.4, p. 61 (SIGGRAPH, Los Angeles, USA, August 2015).

Cited in:

Aharoni-Mack, E., Shambik, Y., and Lischinski, D. [2017]. "Pigment-based recoloring of watercolor paintings". In *Proceedings of the Symposium on Non-Photorealistic Animation and Rendering.* ACM, p. 1.

Di Cicco, F., Wiersma, L., Wijntjes, M., Dik, J., Stumpel, J., and Pont, S. [2019]. "A Digital Tool to Understand the Pictorial Procedures of 17th Century Realism". In *Computer Vision – ECCV 2018 Workshops.* Springer International Publishing, pp. 671–675.

Favreau, J.-D., Lafarge, F., and Bousseau, A. [2017]. "Photo2clipart: image abstraction and vectorization using layered linear gradients". *ACM Transactions on Graphics* 36.6, p. 180.

Koyama, Y. and Goto, M. [2018a]. "Decomposing Images into Layers with Advanced Color Blending". *Computer Graphics Forum* 37.7, pp. 397–407.

Lu, S.-P., Dauphin, G., Lafruit, G., and Munteanu, A. [2015]. "Color retargeting: Interactive time-varying color image composition from time-lapse sequences". *Computational Visual Media* 1.4, pp. 321–330.

Mohr, P., Mandl, D., Tatzgern, M., Veas, E., Schmalstieg, D., and Kalkofen, D. [2017]. "Retargeting Video Tutorials Showing Tools With Surface Contact to

Augmented Reality". In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems.* ACM, pp. 6547–6558.

Peng, M., Xing, J., and Wei, L.-Y. [2018]. "Autocomplete 3D sculpting". *ACM Transactions on Graphics* 37.4, p. 132.

Tang, F., Dong, W., Meng, Y., Mei, X., Huang, F., Zhang, X., and Deussen, O. [2017]. "Animated Construction of Chinese Brush Paintings". *IEEE Transactions on Visualization and Computer Graphics.*

Willett, N. S., Kazi, R. H., Chen, M., Fitzmaurice, G., Finkelstein, A., and Grossman, T. [2018]. "A Mixed-Initiative Interface for Animating Static Pictures". In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology.* UIST '18. ACM, pp. 649–661.

Zhao, Q. and Lee, W.-H. [2018]. "The Application of Traditional Chinese Painting Technique and Stroke Effect in Digital Ink Painting". *TECHART: Journal of Arts and Imaging Science* 5.2, pp. 35–42.

Dvorožňák, M., Bénard, P., Barla, P., Wang, O., and Sýkora, D. [2017]. "Example-Based Expressive Animation of 2D Rigid Bodies". *ACM Transactions on Graphics* 36.4 (SIGGRAPH, Los Angeles, USA, July 2017).

Cited in:

Furukawa, S., Fukusato, T., Yamaguchi, S., and Morishima, S. [2017]. "Voice Animator: Automatic Lip-Synching in Limited Animation by Audio". In *International Conference on Advances in Computer Entertainment.* Springer, pp. 153–171.

Koyama, Y. and Goto, M. [2018b]. "OptiMo: Optimization-Guided Motion Editing for Keyframe Character Animation". In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems.* ACM, p. 161.

Dvorožňák, M., Li, W., Kim, V. G., and Sýkora, D. [2018a]. "ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations". *ACM Transactions on Graphics* 37.4 (SIGGRAPH, Vancouver, Canada, August 2018).

In Conference Proceedings

Dvorožňák, M., Sepehri Nejad, S., Jamriška, O., Jacobson, A., Kavan, L., and Sýkora, D. [2018b]. "Seamless Reconstruction of Part-Based High-Relief Models from Hand-Drawn Images". In *Expressive '18: The Joint Symposium on Computational Aesthetics and Sketch Based Interfaces and Modeling and Non-Photorealistic Animation and Rendering* (Victoria, Canada, August 2018).

Other Publications

Dvorožňák, M. [2014]. “Interactive As-Rigid-As-Possible Image Deformation and Registration”. In *Proceedings of CESCG 2014: The 18th Central European Seminar on Computer Graphics* (Smolenice, Slovakia, May 2014).

Cited in:

DeVito, Z., Mara, M., Zollhöfer, M., Bernstein, G., Ragan-Kelley, J., Theobalt, C., Hanrahan, P., Fisher, M., and Nießner, M. [2017]. “Opt: A domain specific language for non-linear least squares optimization in graphics and imaging”. *ACM Transactions on Graphics* 36.5, p. 171.

Appendix B

Authorship Contribution Statement

This statement describes the specific contributions of the author of this thesis to the publications presented therein.

Decomposing Time-Lapse Paintings into Layers (33%)

I contributed to this paper by performing initial experiments with alpha values estimation from a sequence of successive frames of the digital artwork creation process. Then I mainly focused on the demonstration of applications of the extracted layers. I developed a tool enabling interactive editing of the final artwork using layers with both Porter-Duff and Kubelka-Munk models. Using the tool, I created all the results included in the paper and the supplementary video.

Seamless Reconstruction of Part-Based High-Relief Models from Hand-Drawn Images (50%)

For this paper, I carried out initial experiments with methods for inflation of 2D surfaces which led to the formulation of an inflation method producing models with seamlessly interconnected parts. Later, I figured out how to mitigate undesirable artifacts (pits) caused as a side effect of the seamless inflation and implemented the mitigation into both non-linear and linearized inflation method. I implemented a prototype application with which I created most of the results and figures in the paper and generated supplementary material. I wrote an initial version of the paper and contributed significantly to finalizing it. I also made the supplementary video clip.

Example-Based Expressive Animation of 2D Rigid Bodies (44%)

I performed a series of initial experiments dealing with exaggeration of motion through deformation – from experiments with the stylization of 3D rigid body animation based on stylized animations provided by traditional hand-drawn animation artist, to example-based 2D rigid body animation synthesis. These experiments led to concretization and simplification of our goals. Based on the experiments, I proposed a method to augment exemplar dataset by stylization dampening. I worked closely with our animator to specify exemplar animations necessary for the project. I implemented prototype applications using which I created the resulting animations and figures in the paper. I helped to finalize the paper and create the supplementary video clip.

ToonSynth: Example-Based Synthesis of Hand-Colored Cartoon Animations (55%)

In ToonSynth, I extended the pipeline of our method for 2D rigid body stylized animation synthesis to animations of stylized moving characters by developing a technique for motion and appearance stylization transfer. In particular, I proposed a way of motion stylization extraction and transfer by considering differences between skeletal animations and template-to-exemplar deformations. Based on my concurrent experiments on texture synthesis, I simplified our previous parametric synthesis technique by utilizing more general fixed-length overlapping sub-sequences instead of ones defined around events. I helped to figure out proper guidance channels for non-parametric texture synthesis, which allowed automatic selection of appropriate samples from the input exemplar animation and also controlling the temporal coherence in the output animation. I worked closely with our animator to specify source skeletal and exemplar animations necessary for the project. I implemented a prototype application which I utilized to create the resulting animations and figures in the paper. I wrote a draft of the method section and helped to finalize the paper.

Appendix C

Supplementary Material

Decomposing Time-Lapse Paintings into Layers: Kubelka-Munk Mixing Parameters

Jianchao Tan* George Mason University
Marek Dvorožňák CTU in Prague, FEE
Daniel Sýkora CTU in Prague, FEE
Yotam Gingold George Mason University

1 Kubelka-Munk Mixing Parameters

For a (homogeneous) mixture, the overall absorption and scattering coefficients are the weighted sum of constituent materials' coefficients. For an opaque mixture with constituent material proportions c_i and absorption and scattering coefficients K_i and S_i, the overall reflectance R is [Duncan 1940; Barbarić-Mikočević and Itrić 2011]: $\frac{\sum c_i K_i}{\sum c_i S_i} = \frac{(1-R)^2}{2R}$. K and S are both non-negative quantities with no upper bound. The c_i parameters must be non-negative, but need not sum to one. (The use of c_i in the numerator and denominator is self-normalizing.) In our scenario, we can consider each pixel in the time lapse reflectance images to be the result of mixing unknown paint. Between frame $I_{t_{i-1}}$ and I_{t_i}, a new unknown paint may have been added. Solutions for the new paint parameters $c_{t_i} K_{t_i}$ and $c_{t_i} S_{t_i}$ that result in minimal modification of current parameters are as follows.

Because R is expressed as a ratio of K and S, we can assume (without loss of generality) that in the initial reflectance image I_{t_0}, either $c_{t_0} K_{t_0} = 1$ or $c_{t_0} S_{t_0} = 1$. Solving for the other is trivial. (Note that for perfectly reflective or absorptive pixels, one of the two must be zero.) In subsequent frames, we observe a new $R_{t_{n+1}}$ and wish to find $c_{t_{n+1}} K_{t_{n+1}}$ and $c_{t_{n+1}} S_{t_{n+1}}$ such that

$$\frac{c_{t_{n+1}} K_{t_{n+1}} + \sum_{i=0}^{n} c_{t_i} K_{t_i}}{c_{t_{n+1}} S_{t_{n+1}} + \sum_{i=0}^{n} c_{t_i} S_{t_i}} = \frac{\Delta K + \bar{K}}{\Delta S + \bar{S}} = \frac{(1-R)^2}{2R}$$

where $\Delta K = c_{t_{n+1}} K_{t_{n+1}}$, $\bar{K} = \sum_{i=0}^{n} c_{t_i} K_{t_i}$, $\Delta S = c_{t_{n+1}} S_{t_{n+1}}$, and $\bar{S} = \sum_{i=0}^{n} c_{t_i} S_{t_i}$. There are infinitely many solutions for ΔK and ΔS. To solve for the "smallest change" or "most transparent" solution, we seek the solution which minimizes ΔK and ΔS. (Recall that both must be non-negative.) The relationship between ΔK and ΔS is linear with non-negative slope:

$$\Delta K + \bar{K} = (\Delta S + \bar{S}) \frac{(1-R)^2}{2R}$$

Therefore, the minimizer of $\Delta K + \Delta S$ is also the minimizer for ΔK and ΔS individually and occurs at either the y-intercept or x-intercept (whichever is non-negative):

$$\begin{array}{lcl} \Delta S = 0 & & \Delta K = 0 \\ \Delta K = \bar{S}\frac{(1-R)^2}{2R} - \bar{K} & \text{or} & \Delta S = \bar{K}\frac{2R}{(1-R)^2} - \bar{S} \end{array}$$

There is no solution when $R = 0$ or $R = 1$, unless $\bar{S}$ or $\bar{K}$, respectively, is already 0.

This additive mixing model can only approximate completely opaque paint that hides all previous paint (via a very large c coefficient for the new paint). A 100% reflective pixel can never become 0% reflective and vice-versa. An undesirable threshold would be needed to determine when a change is so large that previous frames' K and S parameters should be "forgotten."

*e-mail:tanjianchaoustc@gmail.com

References

BARBARIĆ-MIKOČEVIĆ, V. D.-M. Ž., AND ITRIĆ, K. 2011. Kubelka-Munk theory in describing optical properties of paper (i). *Technical Gazette 18*, 1, 117–124.

DUNCAN, D. R. 1940. The colour of pigment mixtures. *Proceedings of the Physical Society 52*, 3, 390.

Decomposing Time-Lapse Paintings into Layers: Supplemental Figures

Jianchao Tan[*]
George Mason University

Marek Dvorožňák
CTU in Prague, FEE

Daniel Sýkora
CTU in Prague, FEE

Yotam Gingold
George Mason University

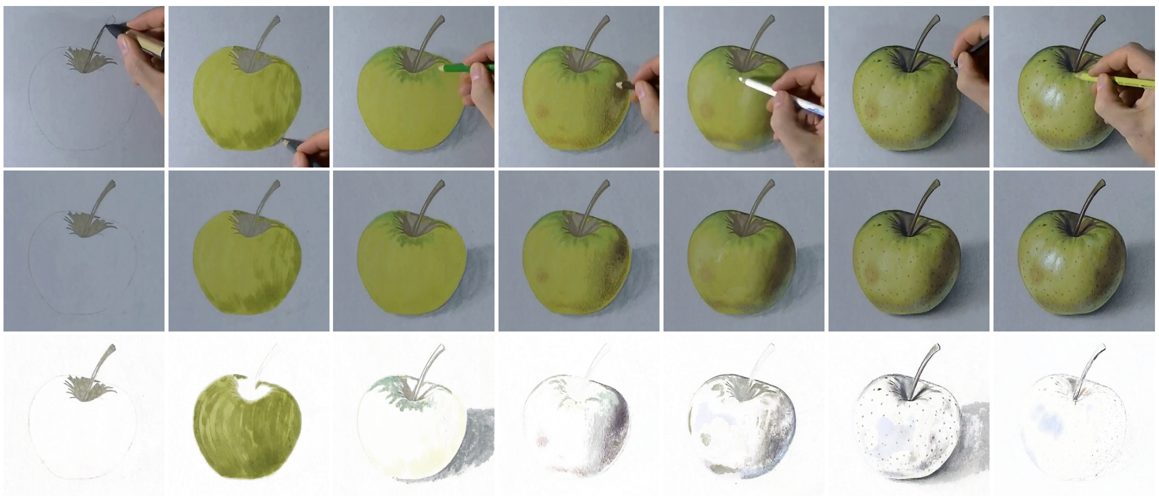


Figure 1: *Examples of layers (bottom) reconstructed from a time lapse video (top) which was pre-processed using our pipeline (middle). Individual layers were grouped into larger clusters to enhance their visibility. Time lapse video © Marcello Barenghi.*

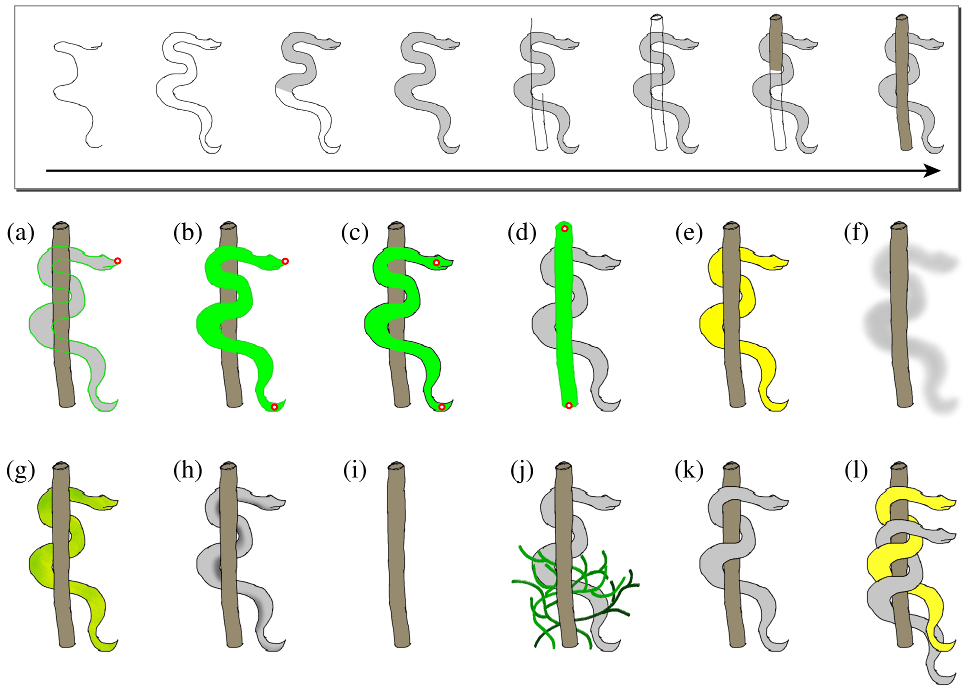


Figure 2: *Image selection & editing using layers obtained from a digital time-lapse painting by our decomposition technique. Given a time lapse recording of an image creation process (top row), layer decomposition is performed with which the user can quickly perform complex spatio-temporal selections (a–d) by clicking on specific pixels located in space and time (red circles). To refine the selection in cluttered scenes, the user can use the mouse wheel to scrub through important events in time to find a desired temporal value. When a range of layers is selected, the user can perform a variety of edits, e.g.: change color (e), blur (f), add texture (g), or darken (h). It is also possible to erase strokes (i), draw new strokes in time (j), move strokes in time (k), or clone strokes in space & time (l).*

[*]e-mail:tanjianchaoustc@gmail.com