EUROGRAPHICS 2010 / T. Akenine-Möller and M. Zwicker
(Guest Editors)

Volume 29 (2010), Number 2

Adding Depth to Cartoons Using Sparse Depth (In)equalities

D. Sýkora,[†1] D. Sedlacek,[2] S. Jinchao,[1] J. Dingliana,[1] and S. Collins[1]

[1]Trinity College Dublin
[2]Czech Technical University in Prague, Faculty of Electrical Engineering

Abstract

This paper presents a novel interactive approach for adding depth information into hand-drawn cartoon images and animations. In comparison to previous depth assignment techniques our solution requires minimal user effort and enables creation of consistent pop-ups in a matter of seconds. Inspired by perceptual studies we formulate a custom tailored optimization framework that tries to mimic the way that a human reconstructs depth information from a single image. Its key advantage is that it completely avoids inputs requiring knowledge of absolute depth and instead uses a set of sparse depth (in)equalities that are much easier to specify. Since these constraints lead to a solution based on quadratic programming that is time consuming to evaluate we propose a simple approximative algorithm yielding similar results with much lower computational overhead. We demonstrate its usefulness in the context of a cartoon animation production pipeline including applications such as enhancement, registration, composition, 3D modelling and stereoscopic display.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.4]: Graphics Utilities—Graphics editors, Image Processing and Computer Vision [I.4.8]: Scene Analysis—Depth cues, Image Processing and Computer Vision [I.4.6]: Segmentation—Pixel classification, Computer Applications [J.5]: Arts and Humanities—Fine arts

1. Introduction

Recovering depth from a single image is a challenge, which has remained an open problem after decades of active research. In this paper we focus on a specific variant of the problem where the input image is a hand-made line drawing. As opposed to previous attempts to provide complete 3D reconstruction either by imposing various geometric assumptions [LS96, VM02, LFG08] or using sketch-based interfaces to create the 3D model incrementally [IMT99, KH06, NISA07, JC08, GIZ09], we seek a simple 2.5D pop-up consistent with the observer's perception of depth in the scene. Such representation is crucial for maintaining correct visibility and connectivity of individual parts during interactive shape manipulation [IMH05, WXXC08], deformable image registration [SDC09a] or fragment composition [SBv05] (see Figure 1 top). It can also be used in the context of image enhancement to generate 3D-like shading effects [Joh02], improve perception of depth [LCD06], or produce stereoscopic images which have become increasingly popular due to emerging 3D display technologies (see Figure 1 bottom).

Although in general it is almost impossible to obtain consistent 2.5D pop-up automatically we show that by using a few user-provided hints the task becomes very simple. Our novel approach is motivated by perceptual studies that have tried to understand how a human reconstructs depth information from a single image [KvDK96, Koe98]. These studies show that in contrast to machines, humans have no internal representation of the scene and rather rely on a set of local judgements from which they reconstruct global observation. More specifically they have found that humans typically fail to specify absolute depth values but are much more accurate in telling whether some part of the object is in front of another and vice versa. This observation leads us to the formulation of an optimization problem that tries to emulate the human depth reconstruction process. Its key advantage is that it completely avoids specification of absolute depth values and instead uses depth (in)equalities which better cor-

[†] e-mail: sykorad@cs.tcd.ie

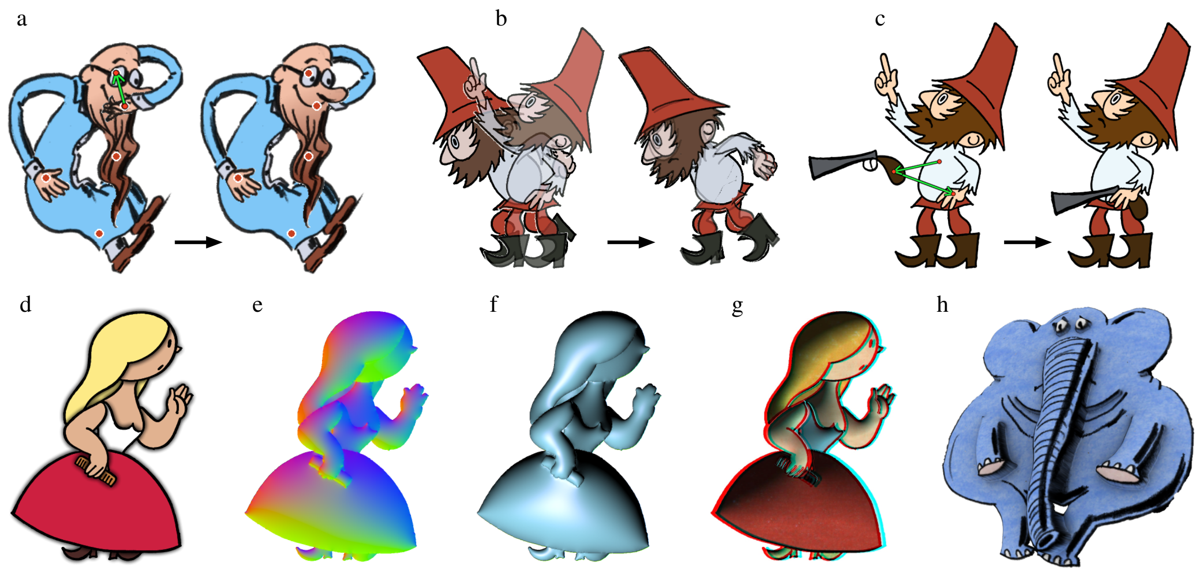

Figure 1: *Applications – depth maps generated using our system can be utilized to maintain correct visibility in (a) as-rigid-as-possible shape manipulation, (b) deformable image registration, and (c) fragment composition. They can also help to (d) enhance perception of depth, (e) interpolate normals with respect to depth discontinuities, and produce (f) 3D-like shading, (g) stereoscopic images, or (h) simple 3D models.*

respond to the user's intuition. Using this framework we implement an interactive user interface that significantly lowers the manual effort needed to create simple cartoon pop-ups.

The rest of the paper is organized as follows. We first summarize previous work in recovering depth from single images and analyze the main drawbacks with respect to our cartoon-oriented scenario. Then we proceed to a description of our novel approach, show some results and discuss implementation issues and limitations. Finally we demonstrate various applications in the context the cartoon animation production pipeline and conclude with possible avenues for future work.

2. Related work

Classical approaches to depth from a single image exploit various photographic depth cues such as shading [Hor90, WSTS08], texture [SB95, For01], depth-of-field [NN94, AW07], or haze [Fat08, HST09]. There are also techniques that treat the problem as a statistical inference and use supervised learning [HEH05, SSN09] or a large database of manually annotated images [RT09] to provide a sample of prior knowledge. However, due to their statistical nature they require the input data to be very similar to that provided in examples or databases and if not they typically produce only very coarse approximations.

When depth cues or statistical inference do not provide sufficient results one can let the user paint with depth directly as with color [Wil90, Kan98]. Although several simple geometric constraints [HiAA97] or custom tailored editing tools [OCDD01] can speed-up the process, it is still very tedious and time consuming.

The problem becomes slightly simpler when the input image is a line drawing with visible contours that delineate object boundaries. When the depicted object consists of several planar surfaces and when some basic geometric constraints are satisfied, a complete 3D model can be generated automatically [LS96, VM02, LFG08]. However, such constraints are typically not satisfied in the case of cartoon images where the aim is to provide highly stylized depiction. A common assumption here is that the object has a blobby surface therefore it is possible to use its contours as an input to some sketch-based 3D modelling tool [IMT99, KH06, NISA07, JC08, GIZ09] to create a simple blobby object that can be further used in various applications [PFWF00, OCN04]. However, such a workflow is not suitable for our 2.5D scenario where the aim is to provide depth values consistent with the internal structure of the cartoon image.

Our technique is most closely related to the work of Zhang et al. [ZDPSS01], who propose a system that allows reconstruction of free-form surfaces from several user-specified hints which form a linearly constrained quadratic optimization problem. Although the method is very successful, there are several drawbacks that make the depth assignment difficult in our cartoon pop-up scenario: (1) they require the user to delineate depth discontinuities manually, (2) it is necessary to specify absolute depth values for regions separated by discontinuities, and (3) the system does not pro-

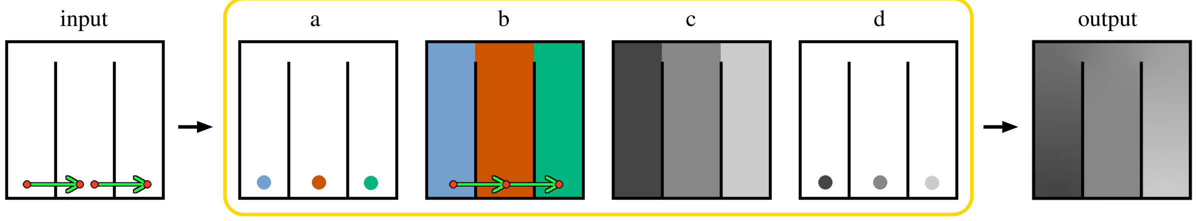

Figure 2: *Depth from sparse inequalities – input image (left) with two user-specified depth inequalities (green arrows), output depth map obtained by solving quadratic program (right), and approximative solution allowing interactive feedback (middle yellow box): (a) multi-label segmentation, (b) topological sorting, (c) intermediate depth map, (d) boundary conditions for Laplace equation.*

vide interactive feedback since the optimization is extremely time consuming even when sophisticated solvers with hierarchical preconditioning are used. Assa and Wolf [AW07] later extended Zhang's approach by adding automatic pre-segmentation and depth inequalities similar to that used in our system. However, their system is still not interactive. The whole pipeline takes several hours per single image. They first generate constraints automatically using various depth cues extracted from an input photograph and then solve a complex optimization problem to obtain the final depth map.

Recently, Ventura et al. [VDH09] presented an interactive sketch-based interface for photo pop-up that also bears some resemblance to our system. They first pre-segment the image and then use distance on a ground plane to assign coarse depths to each layer. After that they manipulate details by painting depth gradients and integrate them to produce the final depth map.

Our approach can also be viewed as an extension of colorization [LLW04] or multi-label segmentation [Gra06] where, instead of colors or labels, the aim is to propagate depth values. The added value of our system is that we do not require the user to specify exact depth values as boundary conditions but instead use depth (in)equalities which make the process more intuitive.

3. Our approach

In this section we present our novel approach to cartoon pop-up based on a set of sparse depth (in)equalities. We first formulate an optimization problem and show how to solve it using quadratic programming, then we proceed to an approximative solution that allows interactive feedback and, finally, we demonstrate how to extend it to handle drawing styles with thick contours and cartoon animations.

3.1. Problem formulation

Let us assume that we have an image $\mathcal{I}$ for which the user has already specified desired depth equalities $\{p,q\} \in \mathcal{U}_=$ and inequalities $\{p,q\} \in \mathcal{U}_>$ (see Figure 2 left), where $p,q \in \mathcal{I}$ are pixels or a set of pixels (scribbles). In the following section we assume that p and q are pixels but it is trivial to extend the approach to work also with scribbles.

Now the task is similar to colorization or segmentation. We want to assign depths to all pixels so that the user-specified constraints are satisfied and discontinuities are preferred at locations where the intensity gradient of $\mathcal{I}$ is high (see Figure 2 right). This can be formulated as an optimization problem where the aim is to find a minimum of the following energy function:

$$\text{minimize:} \quad \sum_{p\in\mathcal{I}} \sum_{q\in\mathcal{N}_p} w_{pq}(d_p - d_q)^2 \qquad (1)$$

$$\text{subject to:} \quad \begin{array}{ll} d_p - d_q = 0 & \forall\{p,q\} \in \mathcal{U}_= \\ d_p - d_q \geq \varepsilon & \forall\{p,q\} \in \mathcal{U}_> \end{array}$$

where d_p denotes the depth value assigned to a pixel p, $\mathcal{N}_p$ is a 4-connected neighborhood of p, and ε is some positive number greater than zero (e.g. $\varepsilon = 1$). The weight w_{pq} is set as in [Gra06]:

$$w_{pq} \propto \exp\left(-\frac{1}{\sigma}(\mathcal{I}_p - \mathcal{I}_q)^2\right),$$

where $\mathcal{I}_p$ is the image intensity at pixel p, and σ is a mean contrast of edges. A closer inspection of (1) reveals that our problem can be rewritten as a quadratic program:

$$\text{minimize:} \quad \frac{1}{2} d^T L d \qquad (2)$$

$$\text{subject to:} \quad \begin{array}{ll} d_p - d_q = 0 & \forall\{p,q\} \in \mathcal{U}_= \\ d_p - d_q \geq \varepsilon & \forall\{p,q\} \in \mathcal{U}_> \end{array}$$

where L is a large sparse matrix of size $|\mathcal{I}|^2$ representing the so called Laplace-Beltrami operator [Gra06]. This program can be solved directly using, for instance, an active set method [GMSW84], however, in practice the computation can take tens of seconds even for very small images. To allow instant feedback we propose an approximative solution that yields similar results with significantly lower computational overhead.

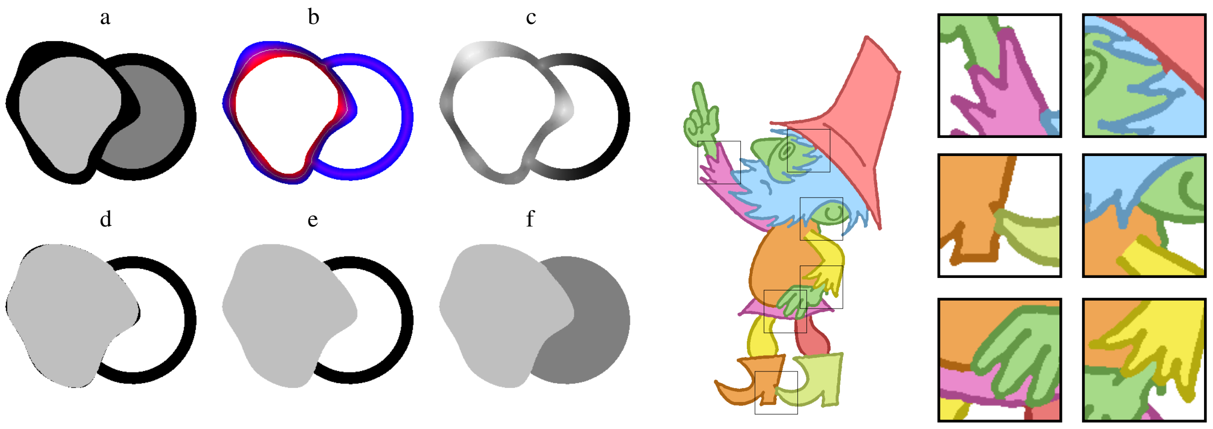

Figure 3: *Depth expansion – synthetic example (left): (a) input depth map with contours, (b) medial axis obtained using two distance transforms computed from active region (red) and all other regions (blue), (c) propagation of contour thickness from medial axis to all contour pixels, (d) depth expansion based on local thickness estimate, (e) filling small gaps, (f) expanded depth map. Practical example (right): several minor artifacts are depicted in selected zoom-ins.*

3.2. Interactive approximation

The key idea of our approximative solution is to decompose the problem into two separate steps: (1) multi-label segmentation (Figure 2a) and (2) depth assignment (Figure 2b). To do so we first treat the user-specified constraints as an unordered set of labels $\mathcal{L}$. Then we exploit an interactive multi-label segmentation algorithm tailored to cartoon images – LazyBrush [SDC09b]. Another alternative is random walker segmentation [Gra06], however, since this algorithm requires us to solve a number of poorly conditioned linear systems it does not provide instant feedback. In contrast LazyBrush exploits fast discrete optimization and furthermore is resistant to imprecise scribble placement.

Once regions are separated we can apply depth inequalities $\mathcal{U}_<$ but now all constraints that fall inside a region are associated with one graph node that corresponds to the underlying region. Such nodes are interconnected by a set of oriented edges representing desired inequalities (see Figure 2b). In general this process can result in an arbitrary oriented graph for which consistent depth assignment (see Figure 2c) exists only if it does not contain oriented loops. The algorithm that solves this problem is known as topological sorting [Kah62] (see Appendix). A part of this algorithm is a detection of oriented loops so that one can easily recognize whether the newly added constraint is consistent. If not, the segmentation phase can be invoked to create a new region and then update the depths with another topological sorting step. Note that since segmentation and depth assignment steps are independent they can be executed incrementally until the desired depth assignment is reached.

When the absolute depths are known, we can easily reconstruct smooth depth transitions (see Figure 2 right) to mimic the result provided by the quadratic program (2). To do so we use a simplified version of (1):

$$\text{minimize:} \sum_{p\in\mathcal{I}} \sum_{q\in\mathcal{N}_p} v_{pq}(d_p - d_q)^2 \qquad (3)$$

$$\text{subject to:} \quad d_p = \hat{d}_p \quad \forall p \in \mathcal{U}_\circ$$

where

$$v_{pq} \propto \begin{cases} \mathcal{I}_p & \text{for } \hat{d}_p \neq \hat{d}_q \\ 1 & \text{otherwise,} \end{cases}$$

$\hat{d}$ denotes depth values from a depth map produced by the topological sorting, and $\mathcal{U}_\circ$ is a subset of pixels used in $\mathcal{U}_<$ or $\mathcal{U}_=$ (see Figure 2d). We let the user decide whether these constraints will be included in $\mathcal{U}_\circ$. The reason we use $\mathcal{I}_p$ instead of 0 is that we have to decide whether the depth discontinuity is real or virtual. Real discontinuities have $\mathcal{I}_p = 0$ but those which are not covered by contours (like upper gaps in Figure 2) have $\mathcal{I}_p = 1$, i.e. they preserve continuity and therefore produce smooth depth transitions. The energy (3) can be minimized by solving the Laplace equation

$$\nabla^2 d = 0$$

with the following boundary conditions ($q \in \mathcal{N}_p$):

$$\begin{array}{lccc} \text{Dirichlet:} & d_p = \hat{d}_p & \Longleftrightarrow & p \in \mathcal{U}_\circ \\ \text{Neumann:} & d'_{pq} = 0 & \Longleftrightarrow & \hat{d}_p \neq \hat{d}_q \wedge \mathcal{I}_p = 0 \end{array}$$

for which a fast GPU-based solver exists [JCW09].

3.3. Depth expansion

Up to this point we have not yet taken care of pixels covered by contours. As the LazyBrush algorithm places region boundaries inside the contour it is not clear to which region the contour actually belongs. Since this information is inevitable for most applications discussed in this paper

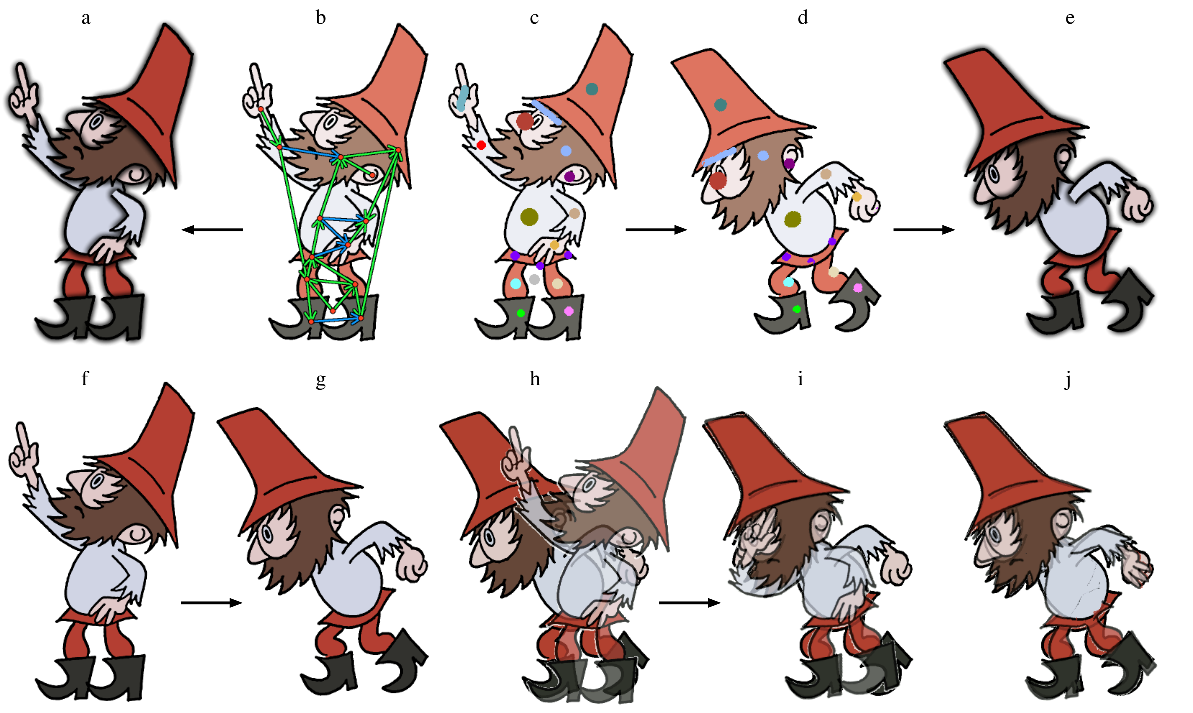

Figure 4: *Depth propagation – transfer of user-provided constraints (top): (a) source image enhanced by a depth map, produced using constraints (b,c), (d) constraints transferred to a target image using as-rigid-as-possible image registration, (e) target image enhanced by a depth map produced using (d). As-rigid-as-possible image registration (bottom): (f) source image, (g) target image, (h) initial overlap, (i) erronous registration without depth map, (j) correct registration using depth map (a) and additional cracks at depth discontinuities specified by blue arrows in (b).*

we have to provide a solution to this depth expansion problem. For drawings containing only thin contours, regions can be consecutively eroded in a front-to-back order. However, such a simple approach can easily lead to noticeable artifacts when used for thick contours. In this case the depth expansion process is equivalent to the figure-ground separation problem described in [PGR99] which is not trivial since it typically requires semantic knowledge that cannot be recovered without additional user intervention.

To solve this we improve upon a previous approach proposed in [SBv05] that in practice produces good results with minor artifacts. We first estimate local contour thickness using two distance maps [FH04]: $\mathcal{D}^1$ computed from the boundaries of regions being expanded (red color in Figure 3b) and $\mathcal{D}^2$ from all other regions (blue color in Figure 3b). Pixels where $\mathcal{D}^1_p = \mathcal{D}^2_p$ form a medial axis from which we can propagate estimates of contour thickness t to all other pixels (Figure 3c) by solving Laplace equation

$$\nabla^2 t = 0$$

using the following boundary conditions ($q \in \mathcal{N}_p$):

$$\begin{array}{lllll} \text{Dirichlet:} & t_p = 2\mathcal{D}^1_p & \iff & \mathcal{D}^1_p = \mathcal{D}^2_p & \\ \text{Neumann:} & t'_{pq} = 0 & \iff & \mathcal{D}^1_p = 0 \vee \mathcal{D}^2_p = 0. & \end{array}$$

Then we expand the region to pixels where $\mathcal{D}^1_p < t_p$ (Figure 3d) and fill in small gaps by removing connected components of which the size is below a predefined threshold (Figure 3e). The expanded region is then removed from the depth map and the same process is applied to all other remaining regions in a front-to-back order to obtain the resulting expansion (Figure 3f).

3.4. Depth propagation

Our depth assignment framework can be easily extended to handle cartoon animations. We can utilize the as-rigid-as-possible image registration algorithm proposed in [SDC09a]. The depth propagation phase is the same as auto-painting: a pre-annotated animation frame (Figure 4c) is registered with a target image (Figure 4g) and user-defined labels $\mathcal{L}$ are automatically transferred (Figure 4d) between corresponding pixels (Figure 4j). After that the LazyBrush algorithm [SDC09b] is used to obtain the final depth map (Figure 4e). Note that since the absolute depth values are already known in the source frame we do not need to execute a topological sorting step.

Known depth values can be also utilized during as-rigid-as-possible image registration to avoid inconsistent configurations which can arise when several parts of the deformed shape overlap each other (Figure 4i). Moreover, the user can additionally specify a subset of depth inequalities (blue arrows in Figure 4b) to mark out discontinuities that will be understood as cracks during the square embedding phase used in the as-rigid-as-possible registration algorithm [SDC09a]. This modification allows more accurate image registration and consequently more accurate depth propagation in cases when several parts of the deformed shape are glued together in the 2D projection although they can move independently (e.g. hands or boots in Figure 4f).

4. Results and Limitations

We have implemented our algorithm in a simple interactive application where the user can freely combine multi-label segmentation with the specification of depth inequalities. A typical workflow suitable for novice users is to first define the segmentation and then add depth inequalities, however, experienced users will tend to combine these two modes to achieve faster progress. They can, for instance, directly generate scribbles with a newly added depth inequality to produce a region and constraint in one step.

Several examples of cartoon pop-ups generated using our algorithm are presented in Figures 5 and 6. User interactions are recorded in two separate layers – one contains equality scribbles and the second contains inequalities. These two layers are hidden during the interaction to avoid clutter. It is typically sufficient to see only the currently added constraint together with an intermediate depth map (Figure 5c) enhanced using a technique that will be discussed later.

Note that the user does not tend to specify a minimal set of inequalities but instead uses a more complicated graph (see Figure 5b). She typically starts with local inequalities and then refines the result by adding more distant constraints. This incremental process nicely corresponds to how the human visual system works. It starts from local cues and then proceeds to more complicated global ones to reach the overall perception of depth in the scene.

As the LazyBrush algorithm [SDC09b] suffers from metrication artifacts when a large part of the contour is missing, we additionally allow the user to specify a set of virtual contours using scribbles or spline strokes (see pink curves with black dots in Figure 6a). However, these curves are required only when the output is expected to be the intermediate discrete depth map (Figure 6c). In the smooth case (Figure 6d), the shape of the virtual contours is not important since it does not correspond to a depth discontinuity and therefore will be smoothed out. Nevertheless, the virtual contours can still be used in order to remove real contours that do not represent depth discontinuities (upper part of trunk in Figure 6) or to introduce them (lower part of dog's head in Figure 6). From this perspective our approximative algorithm brings another important benefit as the pre-segmentation phase automatically removes unimportant details (e.g. trunk wrinkles in Figure 6) that will normally be understood as candidates for depth discontinuities. To obtain the same result using the solution based on quadratic programming (2) it is necessary to mark out all such details as virtual contours.

Limitations. An important limitation of the proposed algorithm is the depth expansion phase described in Section 3.3. Although in general it gives better results as compared to the simple median filter used in [SBv05] it still produces several artifacts (see separate zoomed-in insets in Figure 3). These arise mainly due to the incorrect estimation of contour thickness and/or nontrivial overlapping of contours. Although these small errors are typically imperceptible in most applications it is necessary to correct them manually when a clean depth map is expected. Another drawback lies in the expert annotation workflow when the user starts to combine segmentation with the specification of depth inequalities. Although the proposed approximative algorithm detects cycles automatically it is not able to decide where to put a new scribble in order to resolve the conflict. This limitation makes the interface more complicated as compared to the solution based on quadratic programming, where the the problem is resolved automatically.

5. Applications

In this section we discuss several applications that use depth maps generated by the proposed algorithm. As our research is mainly motivated by the needs of the cartoon animation production pipeline we focus on this field, however, we believe that similar principles are applicable also in a more general image editing context.

Composition and manipulation. Depth maps generated using our system provide invaluable information for interactive shape manipulation [IMH05, WXXC08] where several parts of deformed shapes can overlap each other. Using our approach the user can quickly modify the depth order to enforce the desired visibility (see Figure 1a). The same problem arises in systems where the users extract and compose image fragments [SBv05]. Here depth inequalities allow quick reordering of regions to obtain correct composition (see Figure 1c). Although a similar workflow is used also in [MP09], our approach is more general as it naturally handles self occlusion and does not require any special data structure.

Enhancement and shading. Per-pixel depth values are important also for various image enhancement techniques. In our framework we enhance depth perception by superimposing stacks of regions with blurred boundaries in a back-to-front order (see Figures 1d and 5). This effect is similar to that produced by depth-buffer unsharp masking [LCD06]. Another popular technique that can profit from our algorithm is Lumo [Joh02]. Here per-pixel depth maps help to obtain correct normal interpolation (Figure 1e) which is later used to emulate 3D-like shading (Figure 1f). In the original system the user had to trace region boundaries manually and

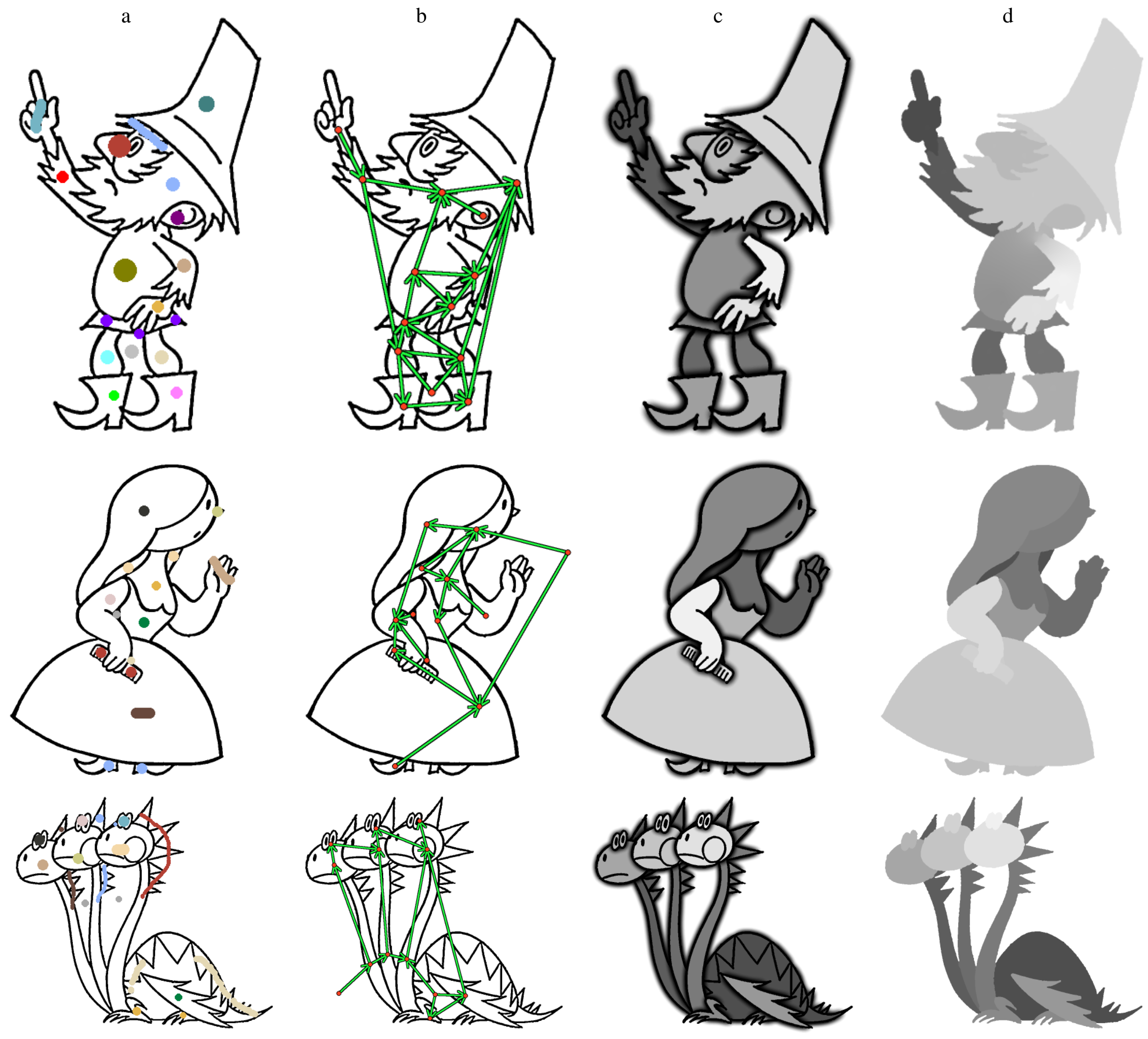

Figure 5: *Examples of cartoon pop-ups generated using our system: (a) user-specified depth equalities, (b) inequalities, (c) depth visualization used during the interactive session to provide visual feedback, (d) final depth map.*

then specify absolute depth values, which is time consuming and counterintuitive. Using our technique correct normal maps can be obtained very quickly.

Modelling and stereo. Using an analogy to shape from shading we can reconstruct an initial blobby surface using the Z-component or normals generated by Lumo [Joh02]. Such surfaces can be further refined using smooth depth maps produced by our framework. We set up a consistent gradient field that respects smooth depth transitions as well as user-provided discontinuities and then integrate it to reconstruct the final height field (see Figure 1h). Such a simple 3D model can be further refined in some modelling tool or rendered from two different viewpoints to obtain stereoscopic images (see Figure 1g).

6. Conclusions and Future work

We have presented a novel interactive approach to cartoon pop-up that enables artists to quickly annotate their handmade drawings and animations with additional depth information. A key benefit of the proposed method as compared to previous depth assignment techniques is that it uses depth (in)equalities instead of absolute depth values. Since this type of constraint better corresponds to the depth recovery mechanism used in the human visual system, it makes the depth assignment more intuitive and less tedious.

As a future work we plan to improve the depth expansion phase and develop an automatic approach for resolving the cycling problem in the expert annotation workflow. We also

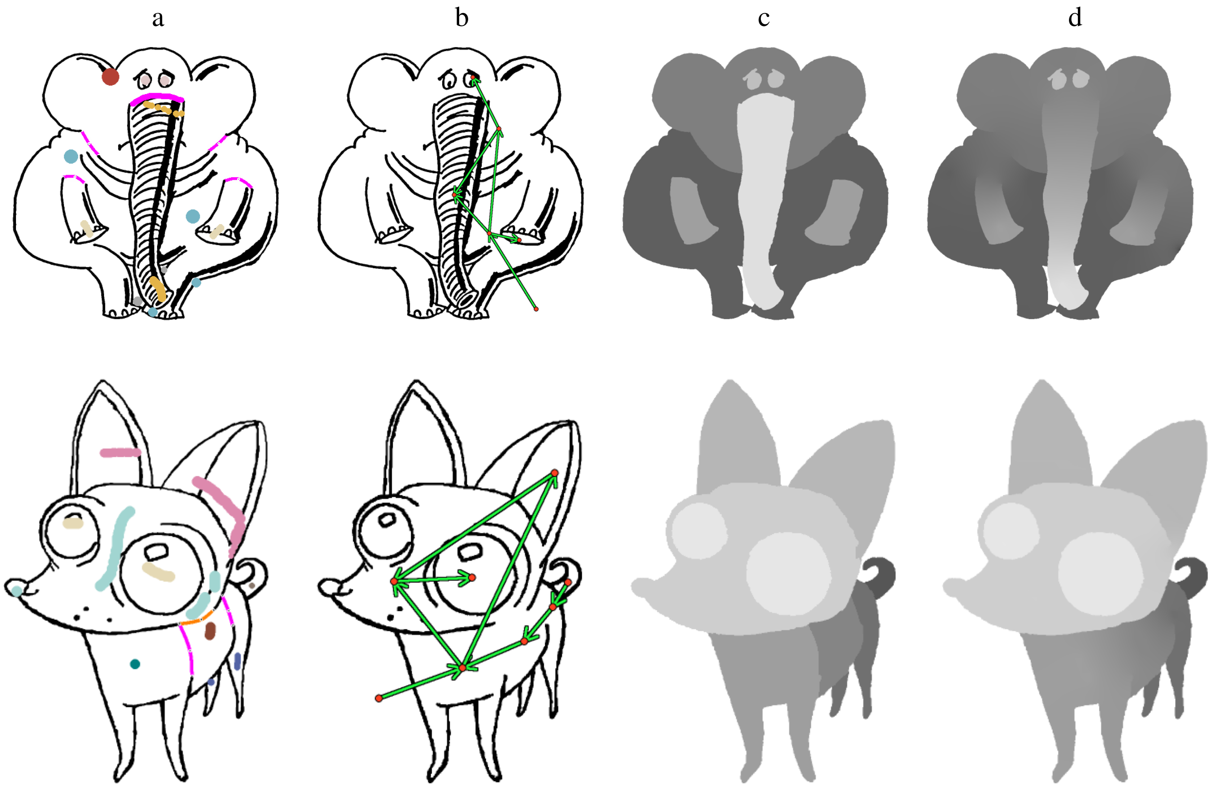

Figure 6: *Examples of cartoon pop-ups generated using our system: (a) user-specified virtual contours (pink) and depth equalities, (b) inequalities, (c) intermediate discrete depth map, (d) final smooth depth map.*

want to extend our approach to work with natural images and find some other applications for using inequalities instead of absolute values in the more general context of interactive image editing.

Acknowledgements

We thank L. Kavan and anonymous reviewers for their fruitful comments. Images used in this paper are courtesy of studios UPP, DMP & Anifilm, P. Koutský, and K. Mlynaříková. This work has been supported by the Marie Curie action IEF, No. PIEF-GA-2008-221320 and partially by the MŠMT under the research program LC-06008 (Center for Computer Graphics).

References

[AW07] ASSA J., WOLF L.: Diorama construction from a single image. *Computer Graphics Forum 26*, 3 (2007), 599–608.

[Fat08] FATTAL R.: Single image dehazing. *ACM Transactions on Graphics 27*, 3 (2008), 72.

[FH04] FELZENSZWALB P. F., HUTTENLOCHER D. P.: *Distance Transforms of Sampled Functions*. Tech. Rep. TR2004-1963, Cornell University, 2004.

[For01] FORSYTH D. A.: Shape from texture and integrability. In *Proceedings of IEEE International Conference on Computer Vision* (2001), pp. 447–453.

[GIZ09] GINGOLD Y., IGARASHI T., ZORIN D.: Structured annotations for 2D-to-3D modeling. *ACM Transactions on Graphics 28*, 5 (2009), 148.

[GMSW84] GILL P. E., MURRAY W., SAUNDERS M. A., WRIGHT M. H.: Procedures for optimization problems with a mixture of bounds and general linear constraints. *ACM Transactions on Mathematical Software 10*, 3 (1984), 282–298.

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

[HEH05] HOIEM D., EFROS A. A., HEBERT M.: Automatic photo pop-up. *ACM Transactions on Graphics 24*, 3 (2005), 577–584.

[HiAA97] HORRY Y., ICHI ANJYO K., ARAI K.: Tour into the picture: Using a spidery mesh interface to make animation from a single image. In *ACM SIGGRAPH Conference Proceedings* (1997), pp. 225–232.

[Hor90] HORN B. K. P.: Height and gradient from shading. *International Journal of Computer Vision 5*, 1 (1990), 37–75.

[HST09] HE K., SUN J., TANG X.: Single image haze removal using dark channel prior. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 1956–1963.

[IMH05] IGARASHI T., MOSCOVICH T., HUGHES J. F.: As-rigid-as-possible shape manipulation. *ACM Transactions on Graphics 24*, 3 (2005), 1134–1141.

[IMT99] IGARASHI T., MATSUOKA S., TANAKA H.: Teddy:

A sketching interface for 3D freeform design. In *ACM SIGGRAPH Conference Proceedings* (1999), pp. 409–416.

[JC08] JOSHI P., CARR N. A.: Repoussé: Automatic inflation of 2D artwork. In *Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2008), pp. 49–55.

[JCW09] JESCHKE S., CLINE D., WONKA P.: A GPU Laplacian solver for diffusion curves and Poisson image editing. *ACM Transaction on Graphics 28*, 5 (2009), 116.

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

[Kah62] KAHN A. B.: Topological sorting of large networks. *Communications of the ACM 5*, 11 (1962), 558–562.

[Kan98] KANG S. B.: *Depth Painting for Image-Based Rendering Applications*. Tech. rep., Cambridge Research Laboratory, 1998.

[KH06] KARPENKO O. A., HUGHES J. F.: SmoothSketch: 3D free-form shapes from complex sketches. *ACM Transactions on Graphics 25*, 3 (2006), 589–598.

[Koe98] KOENDERINK J. J.: Pictorial relief. *Philosophical Transactions of the Royal Society of London 356*, 1740 (1998), 1071–1086.

[KvDK96] KOENDERINK J. J., VAN DOORN A. J., KAPPERS A. M. L.: Pictorial surface attitude and local depth comparisons. *Perception & Psychophysics 58*, 2 (1996), 163–173.

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

[LFG08] LEE S., FENG D., GOOCH B.: Automatic construction of 3D models from architectural line drawings. In *Proceedings of Symposium on Interactive 3D Graphics* (2008), pp. 123–130.

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

[LS96] LIPSON H., SHPITALNI M.: Optimization-based reconstruction of a 3D object from a single freehand line drawing. *Computer-Aided Design 28*, 8 (1996), 651–663.

[MP09] MCCANN J., POLLARD N. S.: Local layering. *ACM Transactions on Graphics 28*, 3 (2009), 84.

[NISA07] NEALEN A., IGARASHI T., SORKINE O., ALEXA M.: FiberMesh: Designing freeform surfaces with 3D curves. *ACM Transactions on Graphics 26*, 3 (2007), 41.

[NN94] NAYAR S. K., NAKAGAWA Y.: Shape from focus. *IEEE Transactions on Pattern Analysis and Machine Intelligence 16*, 8 (1994), 824–831.

[OCDD01] OH B. M., CHEN M., DORSEY J., DURAND F.: Image-based modeling and photo editing. In *ACM SIGGRAPH Conference Proceedings* (2001), pp. 433–442.

[OCN04] ONO Y., CHEN B.-Y., NISHITA T.: 3D character model creation from cel animation. In *Proceedings of International Conference on Cyberworlds* (2004), pp. 210–215.

[PFWF00] PETROVIĆ L., FUJITO B., WILLIAMS L., FINKELSTEIN A.: Shadows for cel animation. In *ACM SIGGRAPH Conference Proceedings* (2000), pp. 511–516.

[PGR99] PAO H.-K., GEIGER D., RUBIN N.: Measuring convexity for figure/ground separation. In *Proceedings of IEEE International Conference on Computer Vision* (1999), pp. 948–955.

[RT09] RUSSELL B., TORRALLBA A.: Building a database of 3D scenes from user annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 2711–2718.

[SB95] SUPER B. J., BOVIK A. C.: Shape from texture using local spectral moments. *IEEE Transactions on Pattern Analysis and Machine Intelligence 17*, 4 (1995), 333–343.

[SBv05] SÝKORA D., BURIÁNEK J., ŽÁRA J.: Sketching cartoons by example. In *Proceedings of Eurographics Workshop on Sketch-Based Interfaces and Modeling* (2005), pp. 27–34.

[SDC09a] SÝKORA D., DINGLIANA J., COLLINS S.: As-rigid-as-possible image registration for hand-drawn cartoon animations. In *Proceedings of International Symposium on Non-photorealistic Animation and Rendering* (2009), pp. 25–33.

[SDC09b] SÝKORA D., DINGLIANA J., COLLINS S.: LazyBrush: Flexible painting tool for hand-drawn cartoons. *Computer Graphics Forum 28*, 2 (2009), 599–608.

[SSN09] SAXENA A., SUN M., NG A. Y.: Make3D: Learning 3D scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence 31*, 5 (2009), 824–840.

[VDH09] VENTURA J., DIVERDI S., HÖLLERER T.: A sketch-based interface for photo pop-up. In *Proceedings of Eurographics Symposium on Sketch-Based Interfaces and Modeling* (2009), pp. 21–28.

[VM02] VARLEY P. A. C., MARTIN R. R.: Estimating depth from line drawings. In *Proceedings of ACM Symposium on Modeling and Application* (2002), pp. 180–191.

[Wil90] WILLIAMS L.: 3D paint. *ACM SIGGRAPH Computer Graphics 24*, 2 (1990), 225–233.

[WSTS08] WU T.-P., SUN J., TANG C.-K., SHUM H.-Y.: Interactive normal reconstruction from a single image. *ACM Transactions on Graphics 27*, 5 (2008), 119.

[WXXC08] WANG Y., XU K., XIONG Y., CHENG Z.-Q.: 2D shape deformation based on rigid square matching. *Computer Animation and Virtual Worlds 19*, 3–4 (2008), 411–420.

[ZDPSS01] ZHANG L., DUGAS-PHOCION G., SAMSON J.-S., SEITZ S. M.: Single view modeling of free-form scenes. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (2001), pp. 990–997.

Appendix

Let us assume we have a graph $G(V,E)$ consisting of nodes V interconnected by a set of oriented edges E. In addition to that we have an empty set L and a set S that contains all nodes having no incoming edges. Topological sorting of V can be obtained using the following algorithm [Kah62]:

```
**while** $S \neq \emptyset$ **do**
    $S := S - \{n\}$   and   $L := L \cup \{n\}$
    **for** $\forall m \in V$ having edge $e : n \rightarrow m$ **do**
        $E := E - \{e\}$
        **if** $m$ has no other incoming edges **then**
            $S := S \cup \{m\}$
    **endfor**
**endwhile**
**if** $E \neq \emptyset$ **then**
    $G$ has at least one oriented cycle **else**
    $L$ contains topologically sorted nodes.
```