Generating Animations by Sketching in Conceptual Space

Tom White*, Ian Loh*
School of Design
Victoria University of Wellington
Wellington, New Zealand
{tom.white@vuw.ac.nz, lohjun@myvuw.ac.nz}
* equal contribution

Abstract

We introduce a new sketch based interface for generating animations. Unlike traditional digital tools, ours is parameterized entirely by a neural network with no preprogrammed rules or knowledge representations. The capability of our sketching tool to support visual exploration and communication is demonstrated within the context of facial images, though our framework is domain independent. Our recorded sketches serve not only as a means for generating a specific animation, but also a standalone visual encapsulation of an animation's semantic operation which can be reused and refined.

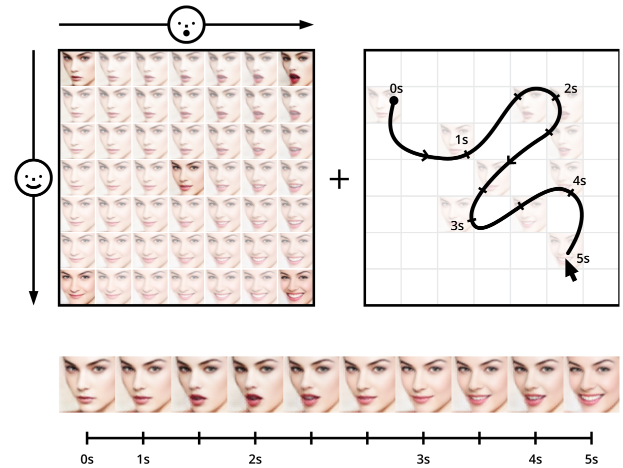

Figure 1: TopoSketch animation process

Introduction

Sketching is the process of quickly exploring an idea through rough designs that focus on key details. In animation, drawings such as thumbnail sketches and pencil tests are used to study the flow of movement. These drawings are often gestural in nature, with loose lines that capture qualities of the animation's structure and its movement. The affordance of speed while still communicating essential qualities makes this process of gestural sketching an ideal tool for supporting digital animation workflows, where high fidelity, production-orientated interfaces may impose obstacles to ideation (Rettig 1994).

Many proposed systems utilise gesture based input in animation sketching by mapping the user's actions to character movement. Building on this, we have developed TopoSketch, a tool for prototyping the animation of faces by sketching gestures using a tablet pen or mouse (Figure 1). A notable difference is that we have displayed gestures as a drawing, intended as a form of encoding that visually communicates qualities of the movement. Gestures generate a path through a vector space of faces generated by a neural network - called a conceptual space - that abstracts the complex task of posing faces into a simple animation controller. TopoSketch animations are generated by alternatively exploring and charting a path through a pre-defined topographic conceptual space.

There are aesthetic and functional motivations to exploring gestural input for digital animation sketching. Abstract gestural lines have been used by animators as a mental model for studying and planning movement, such as "whips" or "lines of action" (Williams 2001; Blair 1994). The director of the hand drawn movie Persepolis (Paronnaud and Satrapi 2007) describes that the 'vibrations of the hand make the drawings come to life' (Education 2007), using these natural variations as a storytelling device. The ability digitally impart these gestures may lend a reflexive element to animation, facilitating a more expressive intuition of the work (Power 2009).

Our system is unique in that it is parameterized not by a rule based system, but instead by the geometric representation layer of the conceptual space. The conceptual space is built from a neural network trained to reconstruct images. This parameterization is reconfigurable and the artifacts produced support and extend sketching as a medium of visual exploration and communication. Additionally those sketches become reusable components of the overall workflow. The sketch's appearance serves as a visual mnemonic to suggest the types of animations that will result when used as an operation in the conceptual space, and the operation the sketch represents can be also reused to achieve a similar animation on different input images.

Background

Sketch and Gesture Animation Interfaces

TopoSketch builds upon other systems that utilise sketch and gesture based input for animation. These applications fall broadly into two categories: the posing of objects or characters, and the generation of movement of those objects.

Conventional animation systems are production orientated, enabling control over many small aspects of an animation. However, a high degree of control is overwhelming and is not ideal for earlier stages of ideation (Rettig 1994). Sketch and gesture based input offer a unique approach to this problem, as its looser precision and intuitive mode of input are suitable qualities for informal tools, letting users focus on the larger picture (Igarashi 2003; Zeleznik, Herndon, and Hughes 1996).

It should be noted the word ”sketch” is used in this paper to generally refer to any form of mark making done quickly to explore an idea. This can include, but should not be confused with a ”design sketch”, which typically focuses on structural representation, such as the shape of a car. Our definition also includes our gesture visualisation as a type of sketch, along with other notations with less literal representations.

Numerous sketch based interfaces have been proposed to make the task of posing characters easier and more accessible. Typically this involves mapping a series of drawn guides as a set of deformations applied to an object. Some enable granular adjustments through familiar notations such as stick figure drawings (Davis et al. 2003; Matthews and Vogts 2011), or by direct drawing on the model itself (Chang and Jenkins 2006). However, the design intention of our system more closely resembles the level of detail afforded by Guay, Cani, and Ronfard, where a single stroke or ”line of action” is used to control the pose of the entire character (Guay, Cani, and Ronfard 2013). The authors argue that this abstraction is less time consuming, and allows users to focus on the overall expressiveness of the animation, better reflecting cognitive workloads involved in early stages of animation. TopoSketch also shares similarities to Sketch Express, where a 2D control interface consisting of a window containing separate drawing regions control different parts of a face (Miranda et al. 2011). While Sketch Express' standardisation lets the same poses to be reused on different faces, our grid based sketches retargets animation timings to different sets of facial expressions.

Similar to posing, sketch based interfaces for generating movement involve recording a path traced by the user - such as a mouse or tablet - and mapping it to an object's transformation. One approach is based on direct manipulation - such as dragging an object across the screen - while the system records these changes (Moscovich and Hughes 2001; Davis, Colwell, and Landay 2008).

Another technique is performance based timing, where the movement of a user drawn path is used to progress through a set of predefined keyframes (Terra and Metoyer 2004) and (Walther-Franks et al. 2012). This allows users to act out their animations, while retaining the precision of keyframes.

Spatial keyframes incorporate aspects of both techniques - users place keyframes with different poses within a scene. Animations are created by moving a cursor in between the spatial keyframes, blending the different poses together based on their distances (Igarashi, Moscovich, and Hughes 2005). The effect of mapping user's movements to complex poses give a ”puppeteering” feel to the system. The authors note that resulting animations are able to make apparent the user's natural sense of timing, contributing to a unique aesthetic. The feeling of creating an animation in TopoSketch is similar in spirit to spatial keyframes, as our animation window can be thought of as having four ”keyframes”, one in each corner. However, we do not allow the creation of new ”keyframes” for more customised controls.

In all approaches, human factors such fatigue, physical limits and acting ability all affect the complexity, quality and length of the animation. For example, it is unreasonable to expect a person to act out a five minute animation in one go. We address this by allowing users to scrub through the timeline to overwrite an area of animation, or continue where they left off.

Another style of sketch based animation generation is through the use of notations. Users draw symbols on top of the scene, which are then parsed into a series contextual animations for a character based on the symbol's position and shape (Thorne, Burke, and van de Panne 2004; Jang et al. 2014; Kazi et al. 2014). While the notations are limited to the number of available symbols, they are often iconic in nature (such as arrows and loops), providing a meaningful visual record of the animation. Although less descriptive in comparison, we argue that our visualised gestures are still able to describe high level aspects of an animation.

In the domain of neural network based tools, sketch input has been used to facilitate searching and exploration of latent spaces. In image manipulation tools by Zhu et al. and Brock et al., instead of directly changing pixels, users are given a ”contextual paintbrush” to draw guiding marks on the image. This rough drawing is used to indirectly navigate a space of generated images within a smaller manifold of coherent results. These assisted interfaces act as ”safety wheels” that allow novice users to make unsupervised changes while maintaining plausible outputs (Zhu et al. 2016; Brock et al. 2016).

Conceptual Spaces

Generative models are a popular approach to unsupervised machine learning. Generative neural network models are trained to produce data samples that resemble the training set (Karpathy et al. 2016). Because the number of model parameters is significantly smaller than the training data, the models are forced to discover efficient data representations. These models are sampled from a set of latent variables in a high dimensional space, called a latent space. Latent space can be sampled to generate observable data values. Learned latent representations often also allow semantic operations with vector space arithmetic (Figure 2), a phonomenon discovered previously in the latent space of language models (Mikolov et al. 2013).

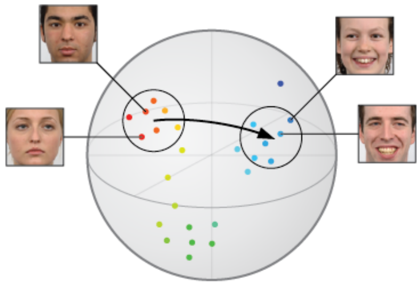

Figure 2: Schematic of the latent space of a generative model. In the general case, a generative model includes an encoder to map from the feature space (here images of faces) into a high dimensional latent space. Vector space arithmetic can be used in the latent space to perform semantic operations. The model also includes a decoder to map from the latent space back into the feature space, where the semantic operations can be observed. If the latent space transformation is the identity function we refer to the encoding and decoding as a reconstruction of the input through the model.

Generative models are often applied to datasets of images. Two popular generative models for image data are the Variational Autoencoder (Kingma and Welling 2013) (VAE) and the Generative Adversarial Network (Goodfellow et al. 2014) (GAN). VAEs use the framework of probabilistic graphical models with an objective of maximizing a lower bound on the likelihood of the data. GANs instead formalize the training process as a competition between a generative network and a separate discriminative network. Though these two frameworks are very different, both construct high dimensional latent spaces that can be sampled to generate images resembling training set data. Moreover, these latent spaces are generally highly structured and can enable complex operations on the generated images by simple vector space arithmetic in the latent space (Larsen, Sønderby, and Winther 2015).

In the latent space of generative models, many high level attributes can be represented as a vector (Figure 3). Using techniques from (White 2016), multiple attributes can be decoupled further to create a visualization of possible states across multiple semantic vectors (Figure 4). For example, when trained on a dataset of portraits, latent vectors can be computed for "smiling" and "mouth open" which then applied to new face images.

Figure 3: Traversals along the smile vector using a GAN model from (Dumoulin et al. 2016)

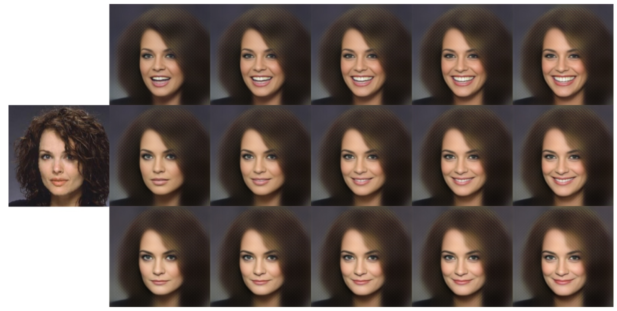

Figure 4: Decoupling attribute vectors for smiling (x-axis) and mouth open (y-axis) allows for more flexible latent space transformations. Input shown at left with reconstruction adjacent. Using a VAE model from (Lamb, Dumoulin, and Courville 2016)

Prior to the discovery of neural network latent spaces supporting semantic operations, cognitive science had hypothesized the existence of knowledge representations that were primarily geometric instead of symbolic. One primary proponent was Gärdenfors who proposed a framework of "Conceptual Spaces" as structured multi-dimensional feature spaces to support modeling information processes such as concept learning and prototype theory (Gärdenfors 2011). Notably, conceptual spaces were proposed as a model of how people structure concepts, independent of any proposed computational implementation of how they might come about.

We adapt the terminology and claim that latent spaces of generative neural networks function as conceptual spaces which can be used as a non-symbolic knowledge representation layers in other tools. Conceptual spaces are a useful medium for building human-centered tools as compared with the "black-box" neural network systems which lack useful substructures or the more brittle symbolic approaches of rule based systems.

With this framework, we examine the ability of a geometric representation layer built from the latent space of a generative neural network model to support a new type of sketching interface tool. In our initial iteration, we explore the conceptual space of human faces, but the tool itself is domain independent and could be used on other similar domains. In exploring the domain of human faces, our tool constructs subspaces of the larger conceptual space of human portraits as a parameter space of a sketch driven animation tool.

TopoSketch

TopoSketch is a sketch based facial animation tool that uses neural networks to navigate a plausible animation manifold. Posing and animating a believable face is a complex process, due to the interrelation of different facial features (eg: the eyes narrowing during a smile). While other systems allow posing of individual facial features, TopoSketch uses utilities a higher level control grid based on expressions such as 'smiling', or 'opening mouth'. These expressions represent changes of many separate features simultaneously. Drawn gestures are then used to control the interpolation and timing between these different expressions.

The current TopoSketch prototype is based on a VAE model described in (Lamb, Dumoulin, and Courville 2016). Our model is initially trained in an unsupervised fashion on images from the CelebA training set (Liu et al. 2015) resized to 256x256. After training, concept vectors are built using attributes from both the CelebA and the Radboud Faces Database (Langner et al. 2010). We have extended the techniques of constructing concept vectors as described in (White 2016) by also using vectors orthoginal to an SVM hyperplane in latent space, which we have found gives superior results when given sufficient training data.

Extended information covering our current TopoSketch implementation including videos is available online.[1]

Workspace

The main TopoSketch workspace is organized into two windows side by side: the animation window and preview window (Figure 5). When an image of a face is loaded into the tool, it is processed by TopoSketch and an animation grid of generated faces is displayed in the animation window. The x and y axis of this grid each represent the results of different operations applied to the input face, increasing in effect along the axis. For example, the face's mouth widens and smile gets larger along x and y axis respectively. Animations are recorded by drawing gestures within the animation window. Animation playback is controlled using a timeline located beneath both windows. Buttons above the windows provide additional functions for exporting, loading and clearing animations or grids.

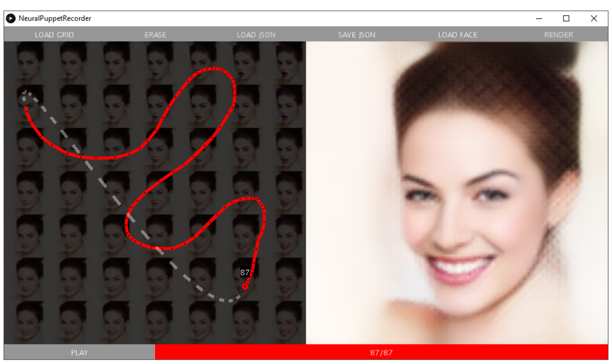

Figure 5: TopoSketch in use

Creating Animations

When the cursor is placed within the animation window, the face closest to the cursor on the grid is shown in the preview window. The user can move the cursor over the animation window to create gestures, "scrubbing" through the animation grid to create transisitions between the faces. Moving along a single axis will only affect the corresponding operation (eg. only smiling) while moving in both axes changes the face with both operations. Once a suitable gesture has been found, the cursor's movement can be recorded by click-dragging within the animation window. The movement is recorded in real time, with the cursor position recorded 25 times a second. This recording is displayed as a line over the animation grid as the user draws, allowing them to see how the animation has progressed. Releasing the cursor stops the recording. To create a smooth loop, the start and end points of the recording are automatically joined with a Bezier curve.

Editing Animations

TopoSketch currently supports basic editing capabilities such as erasing the recording, jumping to any particular time, and continuing the recording from that point. Animations are stored in a modular path file and the animation grid image is an interchangeable element. Paths can be exported for use in another animation or fruther refinement in another program. Path files can also be rendered offline by fully interpolating and sampling in the model's latent space for more temporal resolution. Different animation grids can also be loaded into TopoSketch to reuse the effect of an existing animation with either new faces or conceptual spaces.

Discussion

TopoSketch proposes a method of creating facial animations through a very high level, sketch based interface. Neural network generated conceptual spaces provide an underlying "intuition" that allows simple gestural strokes to be translated into feasible looking transitions between different face expressions. We aspire that the affordances of this style of tool could be useful in a number of practical and aesthetic exploratory applications. The combination of quick low precision gestures, simple representation, and low investment of face posing creates an environment that supports weak filters for quality, encouraging experimentation (Kim, Bagla, and Bernstein 2015).

Many aspects of our system are modular, as both gestures and faces are interchangeable. While naïve gesture and face combinations may not yield practical results, a similar system by (Igarashi, Moscovich, and Hughes 2005) suggests transferring gestures would be useful within similar classes of motion. Examples of these classes can be seen in guides used to plan animation timing, such as "whips" and "waves", that are general enough to be applied to a variety of use cases. For example: both batting eyelashes and an expressive laugh both use an underlying "whip" gesture (Figure 6). The same expression can also vary based on factors such as age and or stress. By changing our conceptual space, we are able to compare the nuances present in these different situations (eg: stressed smile versus a relaxed one). The effects can also be made more extreme, for a caricature-like effect. This can be used as an underlying guide in animation workflows, and for exploring more diverse expressions. While our faces are photorealistic, different stylistic results may be obtained by employing a different model, such as one trained on line drawings. We currently do not provide a way in the tool for users to customise the parameters of the animation grid. However, we envision a mature version of this tool could have a library of expressions that users can

[1]https://vusd.github.io/toposketch/

browse from, or custom expression creation using a webcam.

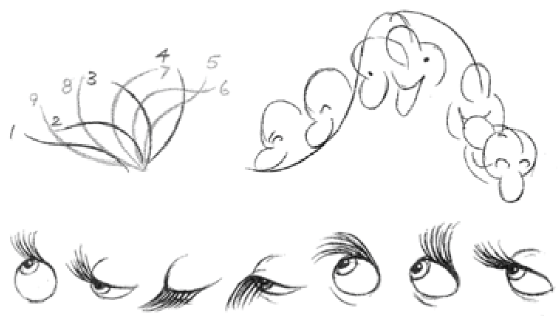

Figure 6: A sketch demonstrating a ”whip” action in motion (left) being applied to the head of a laughing character (right) and batting eyelashes (bottom). (Williams 2001)

Our animation workflow is much more reflexive compared to conventional systems, where animators go back and forth between setting keyframes, and playing back the animation changes. In TopoSketch, animation is created in real-time and viewed in tandem, allowing many different gestures to be explored quickly and practiced, before committing to a final recording. Being able to ”act out” or ”puppeteer” faces using gestures allows users to make reactive adjustments as they are sketching, leading to some stylistic affordances that are not easy to do in conventional tools. For example, start-stop movements that are based on the previous position, or repetitive actions that vary naturally over time. This animation style can potentially be compared to motion capture, or techniques such as straight-ahead animation, which encourage more spontaneous movements (Lasseter 1987).

Displaying recorded gestures as a drawing may have potential applications as a communication aid. While ours does not specifically describe the contents of an animation such as (Thorne, Burke, and van de Panne 2004) or (Kazi et al. 2014), our gestures can still provide some context on the type of the movement. For example, a jagged drawing indicates sudden changes in expression while smoother gestures indicate gentler transitions. Animators already employ similar abstract gestures as guides to study motion. Exposing the visual qualities of more types of animations may lead to serendipitous ideas by way of gestalt effect, or seeing images within the drawings (Owen 2012).

Neuroscience research suggests that being able to see artifacts such as brushstrokes can evoke empathetic responses in viewers (Freedberg and Gallese 2007). While the ”mark”, or underlying structure is quite visible with traditional animation processes (such as guidelines), there is a lack of such in computer animation. Our displayed gestures can been seen to facilitate such a mark, by exposing the construction of movement. Power suggests these indexical artifacts may enable animators to ”feel the movement behind the mark” (Power 2009), opening up a different way to perceive animations. Applications of this can include comparing work from different artists, or as a classification technique for large animation sets. In practice, we have adopted drawn gestures into a notation for planning and describing animations on paper, a formalized version of which is employed in the figures.

Future Work

We plan to expand the concept of the animation grid to accommodate more user customization and potential operation combinations. As an alternative to the grid format, we are exploring using geometric shapes to define how operations are distributed. Adapting TopoSketch to a more freeform interface used in our previous research (Loh 2017) could allow for customized layouts supporting a wider range of animation possibilities.

TopoSketch is able to create a wide range of operations that go beyond facial expressions (eg. getting older or putting on sunglasses). In addition to encouraging users to provide their own images to be used as the subject of an animation, we are also exploring allowing one-shot training of custom facial operations by accepting reference image pairs to define new concept vectors.

References

Blair, P. 1994. *Cartoon Animation*. Tustin, CA: Walter Foster Publishing.

Brock, A.; Lim, T.; Ritchie, J. M.; and Weston, N. 2016. Neural photo editing with introspective adversarial networks. *CoRR* abs/1609.07093.

Chang, E., and Jenkins, O. C. 2006. Sketching articulation and pose for facial animation. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA ’06, 271–280. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association.

Davis, J.; Agrawala, M.; Chuang, E.; Popović, Z.; and Salesin, D. 2003. A sketching interface for articulated figure animation. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA ’03, 320–328. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association.

Davis, R. C.; Colwell, B.; and Landay, J. A. 2008. K-sketch: A ’kinetic’ sketch pad for novice animators. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’08, 413–422. New York, NY, USA: ACM.

Dumoulin, V.; Belghazi, I.; Poole, B.; Mastropietro, O.; Lamb, A.; Arjovsky, M.; and Courville, A. 2016. Adversarially Learned Inference. *ArXiv e-prints*.

Education, F. 2007. Persepolis Production Process.

Freedberg, D., and Gallese, V. 2007. Motion, emotion and empathy in esthetic experience. *Trends in Cognitive Sciences* 11(5):197–203.

Gärdenfors, P. 2011. Semantics based on conceptual spaces. In *Logic and Its Applications - 4th Indian Conference, ICLA 2011, Delhi, India, January 5-11, 2011. Proceedings*, 1–11.

Goodfellow, I. J.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A. C.; and Bengio, Y. 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, 2672–2680.

Guay, M.; Cani, M.-P.; and Ronfard, R. 2013. The line of action: An intuitive interface for expressive character posing. *ACM Trans. Graph.* 32(6):205:1–205:8.

Igarashi, T.; Moscovich, T.; and Hughes, J. F. 2005. Spatial keyframing for performance-driven animation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, 107–115. New York, NY, USA: ACM.

Igarashi, T. 2003. Freeform user interfaces for graphical computing. In *Proceedings of the 3rd International Conference on Smart Graphics*, SG'03, 39–48. Berlin, Heidelberg: Springer-Verlag.

Jang, H.; Jeon, J.; Sohn, E.; Lim, S.-B.; and Choy, Y.-C. 2014. A sketch-based interface to modify and reproduce motion sequences. *Multimedia Tools Appl.* 72(1):591–612.

Karpathy, A.; Abbeel, P.; Brockman, G.; Chen, P.; Cheung, V.; Duan, R.; Goodfellow, I.; Kingma, D.; Ho, J.; Houthooft, R.; Salimans, T.; Schulman, J.; Sutskever, I.; and Zaremba., W. 2016. Generative models. Technical Report 1, OpenAI.

Kazi, R. H.; Chevalier, F.; Grossman, T.; Zhao, S.; and Fitzmaurice, G. 2014. Draco: Sketching animated drawings with kinetic textures. In *ACM SIGGRAPH 2014 Studio*, SIGGRAPH '14, 34:1–34:1. New York, NY, USA: ACM.

Kim, J.; Bagla, A.; and Bernstein, M. S. 2015. Designing creativity support tools for failure. In *Proceedings of the 2015 ACM SIGCHI Conference on Creativity and Cognition*, C&C '15, 157–160. New York, NY, USA: ACM.

Kingma, D. P., and Welling, M. 2013. Auto-encoding variational bayes. *CoRR* abs/1312.6114.

Lamb, A.; Dumoulin, V.; and Courville, A. 2016. Discriminative Regularization for Generative Models. *ArXiv e-prints*.

Langner, O.; Dotsch, R.; Bijlstra, G.; Wigboldus, D.; Hawk, S.; and van Knippenberg, A. 2010. Presentation and validation of the radboud faces database. In *Cognition and Emotion*.

Larsen, A. B. L.; Sønderby, S. K.; and Winther, O. 2015. Autoencoding beyond pixels using a learned similarity metric. *CoRR* abs/1512.09300.

Lasseter, J. 1987. Principles of traditional animation applied to 3d computer animation. *SIGGRAPH Comput. Graph.* 21(4):35–44.

Liu, Z.; Luo, P.; Wang, X.; and Tang, X. 2015. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*.

Loh, I. 2017. Anisketch: Alternative aesthetics for computer animation tools. Master's thesis, School of Design, Victoria University of Wellington, Wellington, NZ.

Matthews, T., and Vogts, D. 2011. A sketch-based articulated figure animation tool. In *Proceedings of the South African Institute of Computer Scientists and Information Technologists Conference on Knowledge, Innovation and Leadership in a Diverse, Multidisciplinary Environment*, SAICSIT '11, 151–160. New York, NY, USA: ACM.

Mikolov, T.; Chen, K.; Corrado, G.; and Dean, J. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781.

Miranda, J. C.; Alvarez, X.; Orvalho, J. a.; Gutierrez, D.; Sousa, A. A.; and Orvalho, V. 2011. Sketch express: Facial expressions made easy. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM '11, 87–94. New York, NY, USA: ACM.

Moscovich, T., and Hughes, J. F. 2001. Animation sketching: An approach to accessible animation. Master's thesis, Department of Computer Science, Brown University, Providence, RI.

Owen, A. 2012. Neuroanimatics: Hyperrealism and digital animations accidental mark. *Animation Practice, Process and Production* 1(2):315–345.

Paronnaud, V., and Satrapi, M. 2007. Persepolis.

Power, P. 2009. Animated expressions: Expressive style in 3d computer graphic narrative animation. *Animation* 4(2):107–129.

Rettig, M. 1994. Prototyping for tiny fingers. *Commun. ACM* 37(4):21–27.

Terra, S. C. L., and Metoyer, R. A. 2004. Performance timing for keyframe animation. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '04, 253–258. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association.

Thorne, M.; Burke, D.; and van de Panne, M. 2004. Motion doodles: An interface for sketching character motion. *ACM Trans. Graph.* 23(3):424–431.

Walther-Franks, B.; Herrlich, M.; Karrer, T.; Wittenhagen, M.; Schröder-Kroll, R.; Malaka, R.; and Borchers, J. 2012. Dragimation: Direct manipulation keyframe timing for performance-based animation. In *Proceedings of Graphics Interface 2012*, GI '12, 101–108. Toronto, Ont., Canada, Canada: Canadian Information Processing Society.

White, T. 2016. Sampling generative networks. *CoRR* abs/1609.04468.

Williams, R. 2001. *The Animator's Survival Kit*. London: Faber.

Zeleznik, R. C.; Herndon, K. P.; and Hughes, J. F. 1996. Sketch: An interface for sketching 3d scenes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, 163–170. New York, NY, USA: ACM.

Zhu, J.-Y.; Krähenbühl, P.; Shechtman, E.; and Efros, A. A. 2016. Generative visual manipulation on the natural image manifold. In *Proceedings of European Conference on Computer Vision (ECCV)*.

Appendix: Example Animations

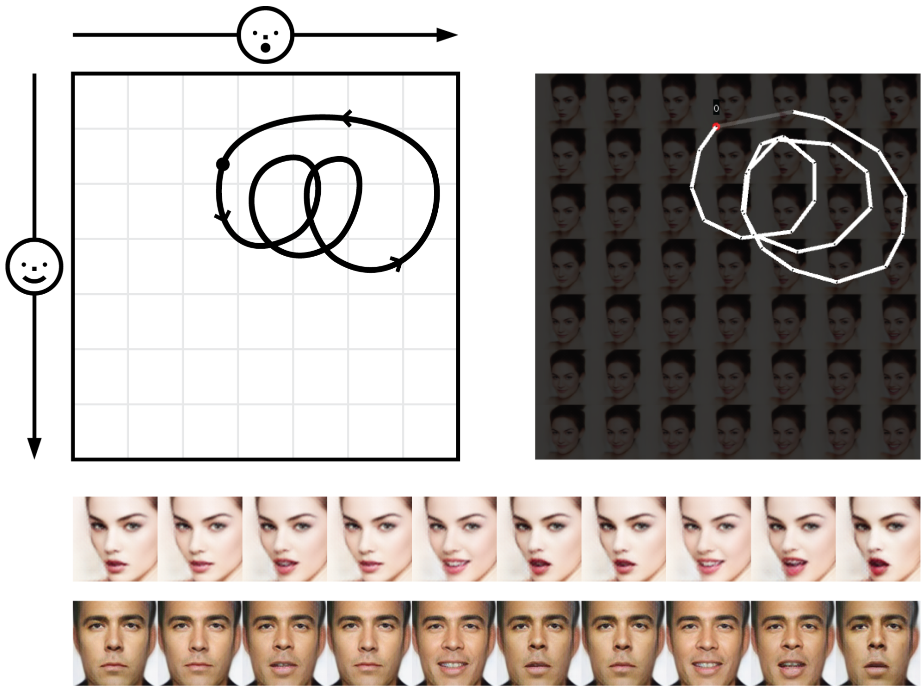

Figure 7: Chewing Animation. The loops in the sketched sequence indicate repeated motions in the animation. The sketch gesture can also be transferred to a second input image without modifying the path.

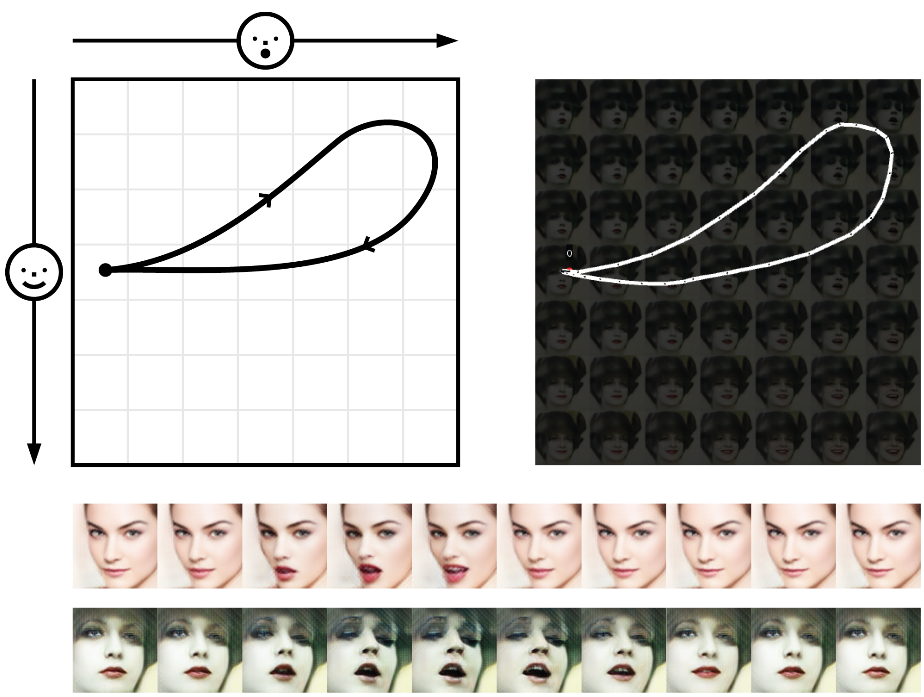

Figure 8: Kissing Animation shows how a sketched sequence is reused on a second input image. The captured sketch is a reusable component that can be applied to other inputs.

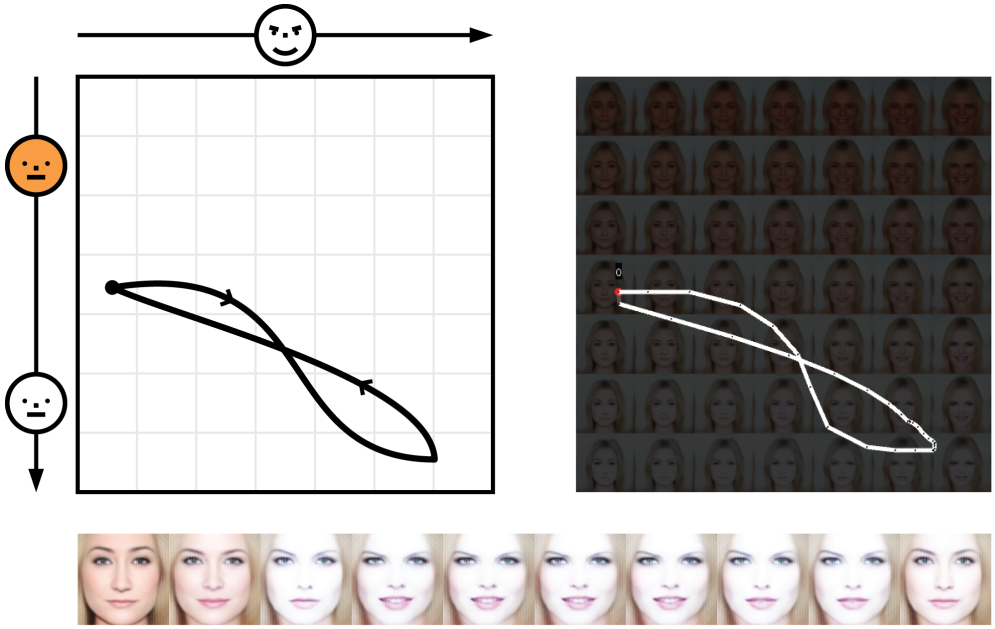

Figure 9: Evil Grin Animation. Depending on the intent, the attributes represented in the conceptual subspace can be changed. In this example, a subspace is created which combines disgust, smiling, and skin tone.

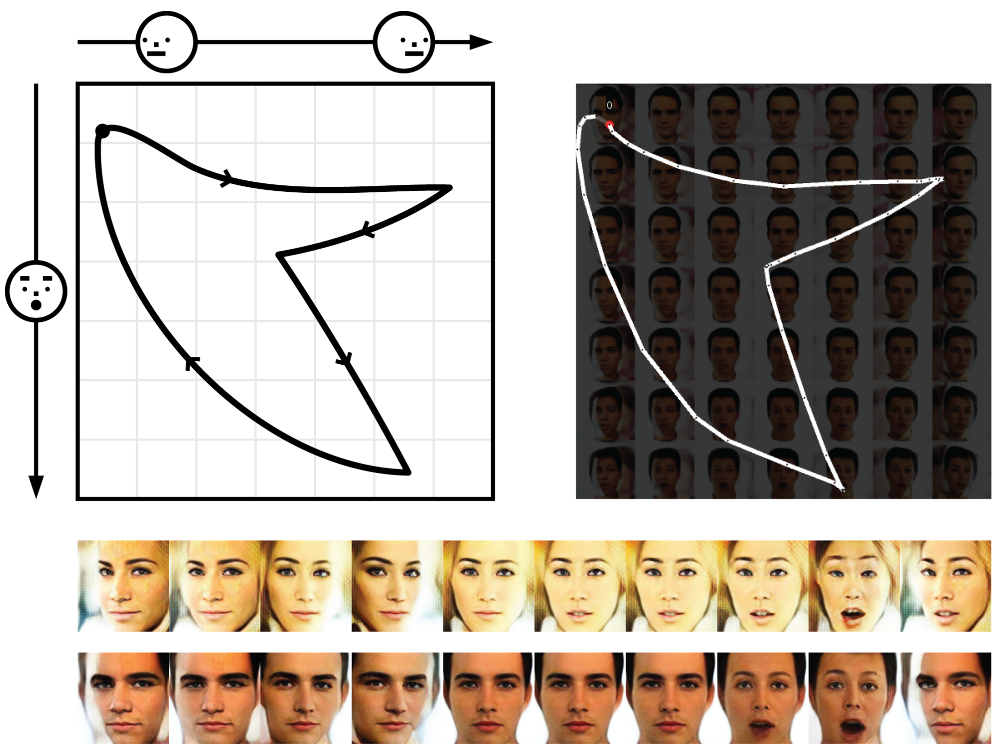

Figure 10: Parameters for animations can combine facial expressions with other changes such as orientation and lighting. Here a ”double take” animation is constructed from face rotation and an expression of surprise.