Stroke based Painterly Inbetweening

Nicolas Barroso[1†], Amélie Fondevilla[2], David Vanderhaeghe[1]

1. IRIT, Université de Toulouse, CNRS, UT3, INP, Toulouse, France.
2. Les fées spéciales, Montpellier, France.

Abstract

Creating a 2D animation with visible strokes is a tedious and time consuming task for an artist. Computer aided animation usually focus on cartoon stylized rendering, or is built from an automatic process as 3D animations stylization, loosing the painterly look and feel of hand made animation. We propose to simplify the creation of stroke-based animations: from a set of key frames, our methods automatically generates intermediate frames to depict the animation. Each intermediate frame looks as it could have been drawn by an artist, using the same high level stroke based representation as key frame, and in succession they display the subtle temporal incoherence usually found in hand-made animations.

CCS Concepts

• ***Computing methodologies*** → ***Non-photorealistic rendering;*** *Animation;*

1. Introduction

We present an automatic paint method to create 2D animations in a painterly style. Our approach focuses on the automatic creation of intermediate frames given a set of key frames. We target professional animator artists, hence we need to provide fine control over each frame, even those computed automatically. The artist draw key frames using tablet with a paint simulation software. Our approach generates intermediate frames using the same stroke representation as in stroke based rendering [VC12], rather than keeping only the pixels representation of the frames.

Our work is close to Chen et. al. [CZBB20] that builds strokes of intermediate frames by rigid transformation of the strokes of key frames. We share the same approach, but in our case, the strokes' transformation are guided by an underlying motion field that gives the expected motion.

The main contributions of our method are:

- the extrapolation of 3D rendered motion field to obtain a 2D motion field suitable for stroke propagation ,
- a frame generator that leverage advected strokes from up to two key frames to produce intermediate frames.

2. Overview

Key frames refer to frame drawn, by the artist, and *intermediate frames* are automatically rendered by our approach. The animation workflow is as follow: the artist draws two or more key frames which convey the example style and appearance to reproduce. The artist also provides a motion field that encompass the animation motion. Our approach generates intermediate frames taking key frames and motion, one frame after the other. At any time, the artist can decide that frame do not corresponds to his wishes and modify or redraw it, this introduce a new key frame in the animation. Intermediate frames are updated to take into account this new key frame. Intermediate frames are surrounded by two key frames, one before and one after along the timeline.

Key frames and intermediate frames are defined by an ordered list of strokes. A stroke is defined by a curve represented as a polyline, and a set of parameters, i.e. the quantity and color of paint on the virtual tool and the pressure of the tool on the canvas along the curve. The final image is obtained by rendering the strokes with a paint simulator.

3. Propagation

Our method needs a motion field for each frame of the animation as input. This motion field can come from different sources, for instance it can be handcrafted by the artist or computed from a video. In practice, for the examples shown here, we start from a simple 3D animation capturing the motion to convey and render a motion field using Blender [Com18].

For each frame, the motion field contains two motion vectors for each pixel of the image, one vector to next frame position of this pixel and one vector to previous frame position. Since the motion vectors are zeros outside of the animated object's surface, we extend the motion information for background pixels to let the artist paint over the background as well. To this end we compute biharmonic weights as describe by Baster et. al. [BBA09]. The main

† E-mail: nicolas.barroso@irit.fr

step of the algorithm is to build a triangular 2D mesh over the image plane with evenly distributed vertices. Each vertex over a surface is assigned the underlying motion vector and becomes a seed point to the interpolation. The motion vector of vertices falling on background pixels are computed using bi-harmonic weight computed for each vertices according to the seeds. Finally, the motion vector of each pixels of the image plane is computed using barycentric interpolation from the motion field of the triangular mesh.

We compute an advected stroke for each strokes of a key frame. An advected stroke captures both key frame content, i.e. the curve, color, and pressure variation, and the motion information, i.e. the motion fields from stroke time to intermediate frame time. As stated before, the stroke's curve is described by a poly-line. The advected curve is computed by moving each poly-line vertex according to the underlying motion field.

4. Generation

The stroke generation consists in creating the set of stroke of an intermediate frame given the advected strokes. To do so, we make three straightforward assumptions:

1. The more an intermediate frame is close in time to key frame, the more it should look like this key frame.
2. An intermediate frame should reflect the content of the key frames before and after it.
3. The intermediate frame should have been done by hand, or at least look likes a hand made drawing.

We propose to generate the strokes of an intermediate frame as follow:

We select a subset of the advected strokes from the two key frames. To go from key frame A at time t_a to key frame B at time $t_b = t_a + N$ we progressively select advected strokes generated from key frame B while deselecting stroke generated from key frame A. To ensure a complete representation of the animated object, we choose to have at least the advected strokes from one of the key frame fully selected at each intermediate frame. To this end we define a selection ratio for intermediate frame at time $t_a + i$ as $\min\left(1, \frac{2(N-i)}{N}\right)$. Selected advected strokes generated from key frame A is 100% for intermediate frames from time t_a to $t_a + N/2$, and decrease to 0% at time t_b. The ratio of advected strokes from key frame B is computed similarly, but reversed in time. We design this selection scheme to ensure correct coverage in the intermediate frames.

We use a strategy that randomizes the list of advected strokes once for all, then selects the strokes according to the ratio following the list order. When rendering the intermediate frame, the stroke drawing order always follows the order in the key frames: We assign a scalar value as *draw time* between 0 and 1 to each of the strokes of a key frame, according to the drawing order of the artist. The selected strokes of both key frames are sorted by this draw time before rendering.

5. Implementation

Our C++/OpenGL prototype use Radium Engine [MRB*21] as main rendering engine. We use our own implementation of the paint simulator presented by Baxter et. al. [CBWG10]. This paint simulator compute bi-directional paint exchanges between the brush and the canvas, on the GPU. Strokes advection is done on the GPU using compute shader. We extract motion fields through Blender AOV rendering [Com18], we use triangle lib in python [She96,R*20] and our implementation of bi-harmonic weight interpolation to obtain the interpolated motion field, as a pre-process.

6. Conclusion

The main limitation of the approach is about the complexity of the motion we can depict. For instance, when two moving objects cross each others on the same layer, some point of the poly-line of stroke of one object will follow the other object motion. To solve this issue, we think a better registration of each stroke with the underlying motion field could be envisioned.

References

[BBA09] BAXTER W., BARLA P., ANJYO K.: N-way morphing for 2d animation. *Computer Animation and Virtual Worlds 20*, 2-3 (2009), 79–87. doi:10.1002/cav.310. 1

[CBWG10] CHU N., BAXTER W., WEI L.-Y., GOVINDARAJU N.: Detail-preserving paint modeling for 3d brushes. In *Proceedings of the 8th International Symposium on Non-Photorealistic Animation and Rendering* (New York, NY, USA, 2010), NPAR '10, Association for Computing Machinery, p. 27–34. doi:10.1145/1809939.1809943. 2

[Com18] COMMUNITY B. O.: *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL: http://www.blender.org. 1, 2

[CZBB20] CHEN J., ZHU X., BÉNARD P., BARLA P.: Stroke Synthesis for Inbetweening of Rough Line Animations. In *Pacific Graphics Short Papers, Posters, and Work-in-Progress Papers* (2020), Lee S.-h., Zollmann S., Okabe M., Wuensche B., (Eds.), The Eurographics Association. doi:10.2312/pg.20201233. 1

[MRB*21] MOURGLIA C., ROUSSELLET V., BARTHE L., MELLADO N., PAULIN M., VANDERHAEGHE D., ET AL.: Radium-engine, July 2021. URL: https://storm-irit.github.io/Radium-Engine/, doi:10.5281/zenodo.5101334. 2

[R*20] RUFAT D., ET AL.: Triangle, 2020. URL: https://rufat.be/triangle/. 2

[She96] SHEWCHUK J. R.: Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, Lin M. C., Manocha D., (Eds.), vol. 1148 of *Lecture Notes in Computer Science*. Springer-Verlag, May 1996, pp. 203–222. From the First ACM Workshop on Applied Computational Geometry. 2

[VC12] VANDERHAEGHE D., COLLOMOSSE J.: Stroke Based Painterly Rendering. In *Image and Video-Based Artistic Stylisation*, Rosin P., Collomosse J., (Eds.), vol. 42 of *Computational Imaging and Vision*. Springer, London, 2012, pp. 3–21. doi:10.1007/978-1-4471-4519-6_1. 1