Analysis of Speed in Traditional Animation

Tyler B. Nowicki* University of Waterloo
William B. Cowan† University of Waterloo
Stephen Mann‡ University of Waterloo

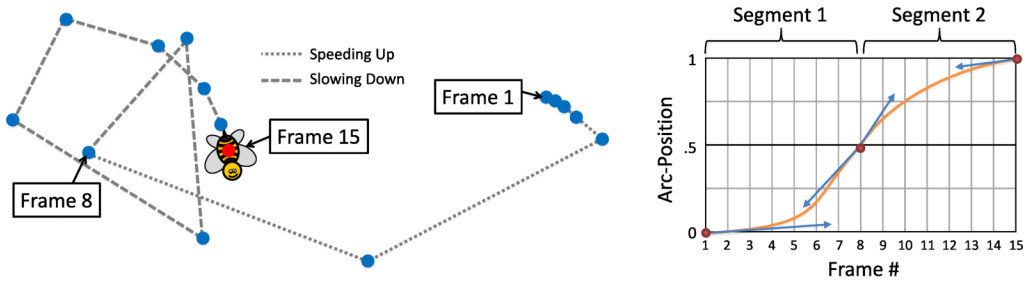

Figure 1: The bee speeds up from frame 1 and slows down to frame 15. The time and place of the change point is a *key-frame* at frame 8. The cubic Bézier spline (orange line) requires two 2D control-points (blue arrows) per segment.

Abstract

Frame-to-frame movement (speed) in animation is commonly described as easing in, easing out, and moving evenly between key-frames. We examined the 2D movement of the salient parts of characters in freehand animation and identified a pattern in speed resembling these descriptions. To identify this pattern we first developed a manual annotation procedure to identify trajectories, their speed related key-frames and their intermediate *subsequences* in a variety of animations. We found that the speed of a subsequence is related to its average acceleration. Our analysis indicates that a cubic polynomial best approximates the subsequence speed over time and each of the polynomial coefficients are related to average acceleration by four additional polynomials of degree 2, 3, 2 and 3. We develop a polynomial model for speed with least squares polynomial regression and validate our results and annotations with several statistical tests that use 10-fold cross-validation. Our experimental animation interface demonstrates that this relationship has the potential to ease the burden of controlling speed by replacing the control points that otherwise must be specified with a single parameter for average acceleration.

Keywords: 2D animation, key-frame annotation, speed control

Index Terms: Computing Methodologies—Computer graphics—Animation—Motion processing

1 Introduction

Appealing character animation requires the careful coordination of multi-part objects. Each part is controlled by a *trajectory* determined in part by its speed, i.e., frame-to-frame distance over time. To control speed the animator must manipulate the value of numerous parameters, such as the eight parameters of a cubic Bézier curve. Consider, for example, animating the flight of the bee in Figure 1 on a computer. The animator specifies the bee's trajectory followed by its speed. Careful control over the number of intermediate frames and their positions is required to properly depict the intent of the movement [9]. Consequently, specifying the speed of each part of a character is a time consuming task. The usual approach to simplify this task is to reduce the number of degrees-of-freedom by manually specifying the kinematic and dynamic relationships between different parts of the character. Although such techniques have been demonstrated, little advancement has been made in identifying and modeling patterns in the speed of individual parts.

A common strategy for overcoming the difficulty involved in hand-animating divides the trajectory into *subsequences* that are more manageable in terms of complexity and length. This is done by creating key-frames that mark important moments in the movement [18, p. 48]. The animator can then focus on creating the intermediate movements (subsequences) to make them accelerate away from the key-frames, *ease-out*; decelerate into the key-frames, *ease-in*; or move at a constant speed, *even* [13]. The goal of this research is to identify, model, and validate the relationship between speed and average acceleration over time by analyzing character movement in a data set of 2D freehand animation.

We examine the speed of trajectories found in animations sampled from different animators, styles, and eras for patterns that may be able to simplify the task of specifying speed. We found that the speed of a trajectory is related its average acceleration by a non-linear relationship. To identify this pattern we captured a data set of trajectories from existing 2D freehand animation and developed a procedure for annotating by eye the speed of each trajectory as accelerating, decelerating, or constant velocity. Computer animation is specifically excluded to ensure we are studying the patterns expressed by hand-animators and not those imposed by splines. Our analysis indicates that a cubic polynomial best approximates the speed over time of our subsequences and each of the polynomial coefficients is related to average acceleration by four additional polynomials of degree 2, 3, 2 and 3. We develop a polynomial model for speed with least squares polynomial regression and validate our polynomial model and annotations with several statistical tests. Our polynomial speed model eliminates a large subset of speed functions from the solution space leaving only those found in freehand animation. Specifically, we replace the control points that otherwise must be specified with a single parameter for average acceleration. Our experimental animation interface demonstrates that our polynomial speed model has the potential to ease the burden of controlling speed.

The key-frames of an animation are not often available and when they are it is not known which key-frames mark an abrupt change in acceleration. To train our polynomial model we identify subsequences by annotating trajectories with key-frames where we saw discontinuous changes in acceleration (jerk) in the movement. Speed is perceived in the visual system by a spatio-temporal integration

*e-mail:tyler.nowicki@uwaterloo.ca
†e-mail:wbcowan@uwaterloo.ca
‡e-mail:smann@uwaterloo.ca

over multiple frames simultaneously [1, 11]. Consequently, when a slow moving object momentarily moves a large distance the integration smooths over the jerk creating the illusion of some acceleration in the resulting movement. Our manual annotation procedure allows the viewer to stop the movement on any frame to identify the jerks between subsequences by eye. We address the obvious potential for inaccuracy and bias by validating our results using several statistical tests and by comparing with statistical methods for classification and change-point detection. Our annotated key-frames may not occur on the same frames as those created by animators making generalization of our results more difficult. For example freehand animators may create key-frames in the middle of a subsequence rather than at the discontinuities between subsequences. In computer animation, however, the key-frames are the breakpoints of a piecewise curve; consequently our model is well matched with computer animation workflows.

We did not encounter prior work that found speed-related constraints in 2D freehand animation. However, much speculation has created a variety of methods for controlling speed that we discuss in section 2. We selected and annotated trajectories from a wide variety of cinematic animations for analysis, discussed in section 3. Our analysis methods, discussed in section 4, model the relationship between speed and average acceleration over time. In section 5 we discuss an experimental animation system inspired by the process followed by hand-animators. Our system uses an interface inspired by the spacing-charts drawn by hand-animators to roughly indicate speed between key-frames [18, p. 47]. This chart-based interface replaces the cumbersome graph editor and eliminates the need to directly manipulate a spline to specify speed. We also discuss the various sources of error that may influence our results; our validation of our data set, annotations and our polynomial model; and the difficulties in generalizing our results beyond freehand animation. Conclusions and future work follow in sections 6 and 7, respectively.

Our contributions include a procedure for annotating key-frames related to speed on a trajectory that mitigates the effects of distractions, a set of statistical tests for evaluating polynomial speed models, a general polynomial speed model characterizing the relationship between speed and acceleration in 2D freehand animation, and an experimental chart-based interface for controlling speed.

2 Related Work

Cubic splines are provided by most animation systems to control speed. Perhaps hundreds of thousands of hours of animation have been produced with cubic splines. Curiously, animation tools have yet to provide any higher order curves. Whether cubic curves are sufficient or even appropriate for controlling movement between key-frames remains unexamined. Most sources cite the cubic spline's low computational complexity and optional continuity, which is thought to be necessary, to explain its use and popularity [10, Ch.3].

Cubic splines are also commonly used to edit the trajectory and arc-length parameterization of a recorded performance. Performance capture uses an input device, such as a video camera, to record the animator's movements and apply them to the object [15]. This eliminates the need for a complex user interface. However, recording devices capture an excessive number of samples that must be simplified with a spline before the movement can be edited [4].

A simple approach to edit existing character movement is to reparameterize time. For example, a common Hollywood trick is to speed up a slow, weak looking punch by reducing the time it takes; increasing duration decreases speed and decreasing duration increases speed. A cubic spline controls the passage of time [17]. Local control is obtained by warping time within a defined region of space. The parts of an object that happen to occupy the space are sped-up or slowed-down as desired [19]. However, the viewer may perceive a change in the video playback, such as the slow-motion effect, rather than a change in speed. Selectively removing sequential frames has been proposed to overcome this problem, but is limited to still portions of animations [6].

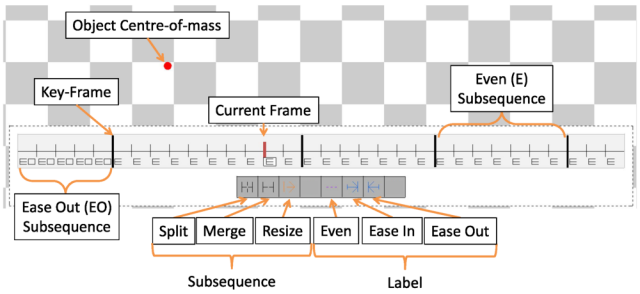

Figure 2: Our annotation program.

Alternatively, an existing animation can be edited by convolving its speed or trajectory with a signal processing filter that is tuned by the animator. The animation bilateral filter exaggerates how the parts of an object slow down and speed up as they enter and leave a turn by reparameterizing the curve's parametric parameter with the result of a modified bilateral filter [8]. The cartoon animation filter produces a similar exaggeration by convolving the trajectory with an inverted Laplacian of a Gaussian filter. This modifies the trajectory producing effects described as anticipation, follow-through and spatial exaggeration [16]. However, these techniques emphasize existing phenomenon in the movement and do not provide control over speed explicitly.

Averaging over many different motion captured poses or movements can be used to determine the constraints on human movement [3]. These are applied to a less detailed avatar to act as an animation interface. Avatar skeletons, although simpler than those of humans, consist of many parts and constraints. Reducing the high-dimensionality requires measuring the similarity between poses with a smaller number of parameters that are predefined or discovered with a principle component analysis. These parameters are the inputs to a model for generating human poses and movements [2, 5]. Our research is similar in that we model time-series data; however, our focus is on lower-level animation data.

3 Trajectory Selection and Annotation

In this section we discuss the selection of a data set of trajectories and our annotation procedure for identifying key-frames. In section 4 we analyze the subsequences to identify and model relationships with speed.

Any salient part of a character is a candidate for analysis, such as a hand, foot, or head. Its movement is the result of the part occupying a sequence of positions, each for a brief period of time. These are samples of its trajectory. Although we cannot be certain of the trajectory that is perceived from these samples by the human visual system, it is no doubt smoother than the polyline that connects the samples. This is due to the spatio-temporal integration of the frames and an unknown amount of error on the part of the animator. The selection of a set of trajectories for analysis is discussed in section 3.1.

To identify and model speed the key-frames that mark changes in speed on the subsequences must be identified. However, the large amount of variance in speed obscures the abrupt changes in acceleration making key-frames difficult to identify automatically. We discuss the sources of error that contribute to this variance in section 5.2. We use a manual procedure to identify key-frames and classify their subsequences as either easing-in, easing-out or moving evenly. We discuss our annotation procedure and how it mitigates illusions due to spatio-temporal integration and distractions caused by multiple parts moving at once in section 3.2.

Table 1: Trajectories extracted from seven traditional animations for analysis.

#	Animation	Studio	Produced	Paths	Subsequences	Frames	Sec.
1	Fresh Hare	Leon Schlesinger Prod.	1942	23	88	736	56.8
2	What's Opera Doc	Warner Bros.	1957	27	89	730	46.8
3	The Thief and the Cobbler (Clip/Trailer)	R. Williams Anim. Prod.	1964-1995	7	21	240	10.4
4	Titan A.E.	Fox Anim. Studio	2000	69	275	2126	154.6
5	Benny's Yawn (Educational)	R. Williams Anim. Prod.	2001	2	10	66	6.0
6	The Princess and the Frog (Clip/Trailer)	Walt Disney Anim. Studio	2009	21	93	732	43.0
7	Get a Horse! (Clip/Trailer)	Walt Disney Anim. Studios	2013	16	29	341	13.5
			1942-2013	165	605	4971	333.1

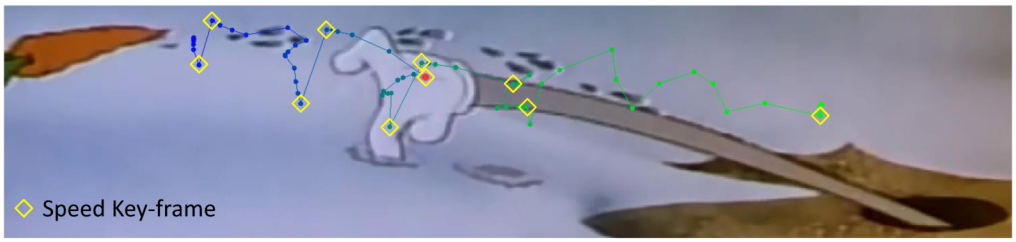

Figure 3: Example of a trajectory and key-frame annotations. The solid line is the trajectory of the hand. Its colour transitions from green at the first frame to blue at the last frame and it is dotted with points marking the hand's centre-of-mass in each frame. The red dot is the location of the hand in the current frame. The animation "Fresh Hare" that includes this frame entered the public domain in 1970.

3.1 Trajectory Selection

For this study, we captured movements from the seven 2D freehand animations listed in Table 1. We chose character animations that were created by professional hand-animators without the aid of computers, although computer methods were utilized for background rendering and shading in some of the recent animations. From these we identified character movements and manually marked the centre-of-mass of its salient parts in each non-duplicate frame. Duplicate frames occur when the animator reuses the previous frame or more commonly when the frame rate is increased from 12 to 24 frames-per-second, the frame-rate of the standard cinematic projector. A sequence of centre-of-masses are the samples of a trajectory. Each sample is located by its arc-length w_i and time t_i, where $1 \leq i \leq N$ and N is the number of samples on the trajectory.

Not all trajectories are suitable for this research. Many animations, such as Saturday morning cartoons, include quick transitions between nearly static poses. In the animations we identified trajectories longer than 5 samples bounded by scene changes, occlusion or stillness. Trajectories longer than 4 seconds were divided on key-frames to make them a manageable length for our annotation interface.

Each salient feature's centre-of-mass was manually marked to avoid perturbations due to changes in orientation. The automated feature tracking methods we considered did not track centre-of-mass and were limited to registering similar pixels [12] or line intersections [7]. In Figure 1 the centre-of-mass of the bee is indicated with a red point at frame 15.

3.2 Annotation Program and Procedure

To identify the key-frames of each trajectory and classify its subsequences we developed an annotation program and procedure that mitigates the effects of unwanted illusions and distractions. Figure 2 depicts the annotation program.

We eliminate distractions by depicting the position of the centre-of-mass as a red dot measuring half a centimetre in diameter on a grey and white chequerboard background. The playback occurs at the intended rate of the animation. The viewer can play forward, backward and pause at any time. And between playbacks the red dot is not drawn for a period of half a second. We specifically do not display the frames of the animation or depict the trajectories during annotation.

We fix the distance of the viewer's head to the screen at 66 cm ± 1 cm resulting in a viewing angle of approximately 31.3 to 41.9 degrees ± 1 degree. These viewing angles are similar to those required by THX for cinema certification; 36 degrees for a theater and 40 degrees for a home theater [14].

The annotation procedure consists of several steps that are repeated until the entire trajectory has been divided by key-frames into labeled subsequences. The viewer first observes the moving dot looking for the frame of the earliest suspected key-frame. Once identified the playback is restricted to the subsequence that runs from the previous key-frame or the beginning of the movement. This mitigates illusions due to discontinuous changes in acceleration between subsequences. To ensure the key-frame is correctly located the viewer experiments with longer and shorter subsequences, aiming to identify the longest subsequence that gives the appearance of continuity in speed. Finally the subsequence's speed category is decided and the key-frame is created. The viewer can choose *even* for constant speed, *ease-in* for decelerating and *ease-out* for accelerating. The process is repeated to annotate the entire trajectory. An example of an annotated trajectory is given in Figure 3.

In total our data set consists of 165 trajectories divided into 605 subsequences: 148 ease-in, 310 even, 147 ease-out by 420 key-frames. Outliers consisting of subsequences with fewer than three frames were removed by splitting trajectories on either side of short subsequences. The order of the trajectories and subsequences was randomized for analysis and validation. A histogram of the subsequence lengths is given in Figure 4. Notice that many subsequences have a duration of less than half a second. This indicates that a feature length animation requires a large number of subsequences.

4 Analysis

In this section we analyze the subsequence speeds over time, depicted in Figure 5, to identify the relationship between speed and acceleration. Formalizing this relationship as a polynomial model

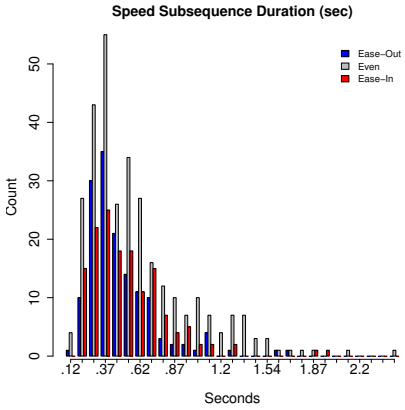

Figure 4: Histogram of subsequence durations. The mean duration is .55 seconds and 80% of the subsequences have a duration of less than .7 seconds.

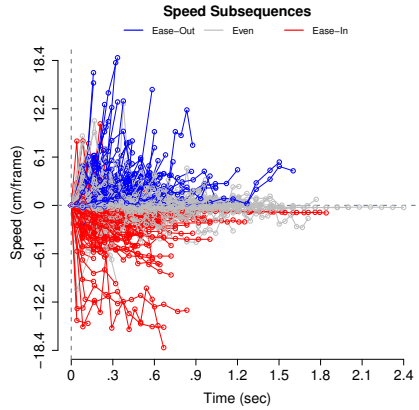

Figure 5: Plot of subsequence speed over time. Note the considerable variation and lack of separation of the data.

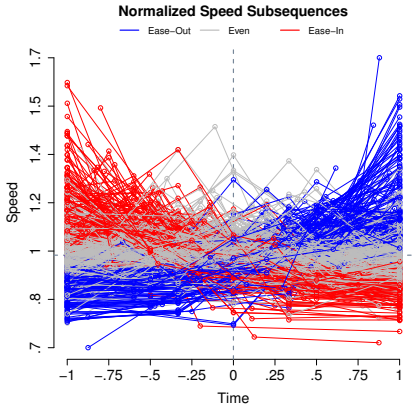

Figure 6: Normalization places constant speed at one and centres time at zero bounded between -1 and 1. Notice that the speed categories each follow a different trend.

allows the arc-length of an object on its trajectory to be computed given its desired acceleration.

Constant speed is normalized to one, acceleration to a transition from values below one to values above one, and the opposite transition for deceleration. Given the arc-length w_k of each sample on its subsequence, where $1 \leq k \leq N-1$ and N is the number of samples on the subsequence, we compute its normalized speed $\hat{\delta}_k$:

$$\hat{\delta}_k = \frac{w_{k+1} - w_k}{\sum_{j=1}^{N-1}(w_{j+1} - w_j)} - \frac{N}{N-1}. \tag{1}$$

Time is centered at zero and bound between negative one and one. Given the sample time t_k, where $1 \leq k \leq N-1$, we compute its normalized time $\hat{t}_k$:

$$\begin{aligned} \tilde{h}_k &= t_k - \frac{\sum_{j=1}^{N-1} t_j}{N-1} \\ \hat{t}_k &= \frac{\tilde{h}_k}{\max(|\tilde{h}_k|)}. \end{aligned} \tag{2}$$

Note that k does not include the last frame N of the subsequence. For example, samples occurring at $h = (0,1,2,3,4,5,6,7,8,9)/12$ seconds are normalized to $t = (-1,-.75,-.5,-.25,0,.25,.5,.75,1)$. At each normalized time $\hat{t}_k$ the normalized speed $\hat{\delta}_k$ indicates how far the object moves from w_k to w_{k+1} relative to the length of the subsequence. Time $\hat{t}$ and speed $\hat{\delta}$ are normalized for the remainder of this section.

Figure 6 depicts the normalized subsequences. The goal of our normalization is to cause subsequences with similar shapes to superpose on a speed over time plot, regardless of the number of samples. The coincidence of similarly shaped subsequences is discussed further in section 4.1. Several other transformations were examined including filtering low frequency noise and log-scaling. However, we found that filtering smoothed over the sudden changes in acceleration that indicate the presents of a key-frame and introduced oscillations, and log-scaling had no significant effect on the results.

To create a polynomial model for speed we first fit polynomial curves to the $\hat{\delta}$ (speed) over $\hat{t}$ (time) polyline of each subsequence. These approximation curves are used for modeling in place of the polylines. We model the relationships between the coefficients of these polynomial approximation curves and the normalized average acceleration (acceleration) of the subsequences with additional polynomials (coef.-accel. polynomials), one per coefficient. Each coef.-accel. polynomial computes a coefficient of the approximation curve. Substituting the trained coef.-accel. polynomials for the coefficients of the approximation polynomial gives our polynomial model of speed.

We discuss the approximation of speed over time for polynomial modeling in section 4.1, the relationship between the coefficients of the approximation curve and acceleration in section 4.2, and polynomial model selection in section 4.3. The polynomial models for each coefficient are included in section 4.2.

Each polynomial model is denoted first by the label **poly.** followed by the degree of the approximation curve. A tuple of numbers follow that indicate the degree of each coef.-accel. polynomial. The label is completed with the total number of terms of the polynomial in brackets. For example, **poly. 3 - 2, 3, 2, 3 (14)** is a degree 3 poly. model where the constant, linear, quadratic and cubic coefficients are given by coef.-factor polynomials of degree 2, 3, 2 and 3, respectively for a total of 14 coefficients in the polynomial model.

4.1 Approximation Curves

The approximation curve is an estimate of the speed the animator had in mind while drawing a subsequence. Consider the example in Figure 7. A subsequence's speed is fit by polynomials of degree one through three and each is used to compute an approximation of the subsequence's $\hat{\delta}$ (speed). The animator may have had one of the curves in mind while drawing the subsequence it is from. Figure 7 also depicts the approximation of $\hat{\delta}$ over time. Linear, quadratic, and cubic polynomials are fit to $\hat{\delta}$ over time of the subsequences. The colour of each cubic polynomial indicates acceleration. The coincidence of curves with a similar colour and shape indicate a relationship between acceleration and $\hat{\delta}$. To identify the type of curve used in freehand animation we compared polynomial curves of degree zero through three in section 4.3. The polynomial curves are fit by least squares.

The quality of the fit (modeling error) is the mean absolute difference between the actual speed and the approximate speed at each frame-time. The speed of a single subsequence can be fit arbitrarily well by a high degree polynomial, however, when applied to many subsequences over fitting or under fitting can occur. Speed is poorly modeled if a polynomial of a different degree would produce a lower modeling error for an independent test set of subsequences. By exhaustively computing the modeling error of many different polynomial models with cross-validation we identify the polynomial that best fits subsequence $\hat{\delta}$ (speed). We found $\hat{\delta}$ is best approximated

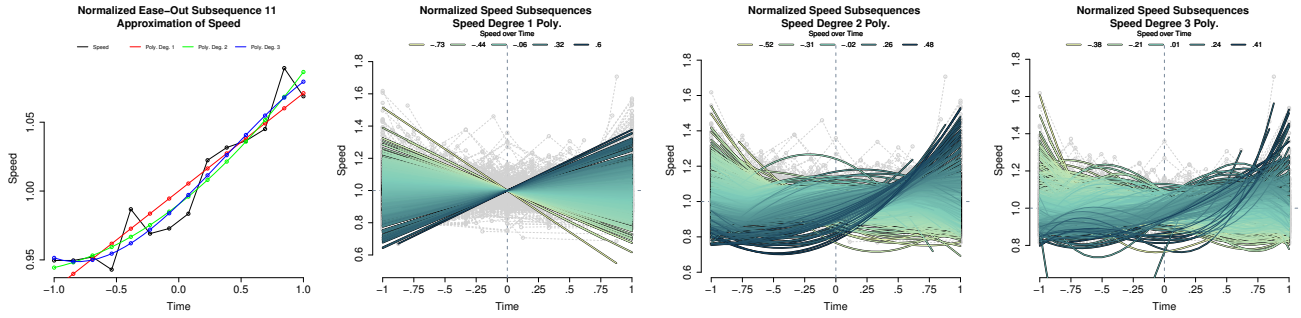

Figure 7: On the left a subsequence's $\hat{\delta}$ (speed) over time is fit by polynomials of degree 1, 2 and 3. The $\hat{\delta}$ (speed) over time of all subsequences fit by polynomials of degree 1 (left-middle), degree 2 (right-middle), and degree 3 (right). Curves are coloured by their acceleration. Curves with higher acceleration are plotted in front of curves with lower acceleration.

with cubic polynomials, discussed in section 4.3.

4.2 Speed and Acceleration

By experimentally modeling the relationships with different factors we found that acceleration is strongly related to the coefficients of the approximation curve. We examined several other factors independently including the min, max, mean, median, and skewness of speed as well as the min, max, median, mean and skewness of each dimension of the subsequence's polyline. Most of these factors did not correlate with the coefficients of the approximation curve. The speed's max, median, and skewness have a weak correlation with the constant and quadratic coefficients of a cubic approximation curve and the subsequence's polyline x-axis skewness has a weak correlation overall. However, incorporating these factors into our polynomial speed model did not reduce the modeling error significantly.

The relationship between the shape of a subsequence and its acceleration is demonstrated in Figure 8. We found that the constant, linear, quadratic, and cubic coefficients of the approximation polynomials of $\hat{\delta}$ (speed) over subsequence acceleration ($\Delta_{\hat{\delta}} = \text{slope}(\hat{\delta}, \hat{t})$) is best fit by polynomials of degree 2, 3, 2 and 3, respectively, discussed in section 4.3. Figure 8 depicts the modeling of the coefficients of the cubic approximation of $\hat{\delta}$ (speed) over time. Our cubic polynomial model of subsequence $\hat{\delta}$ is

$$\begin{aligned}
\hat{\delta}_k(t_k, \Delta_{\hat{\delta}}) &= \dot{\delta}_1(\Delta_{\hat{\delta}}) + \dot{\delta}_2(\Delta_{\hat{\delta}})t_k + \dot{\delta}_3(\Delta_{\hat{\delta}})t_k^2 + \dot{\delta}_4(\Delta_{\hat{\delta}})t_k^3 \\
\dot{\delta}_1(\Delta_{\hat{\delta}}) &= \ddot{\delta}_{1,1} + \ddot{\delta}_{1,2}\Delta_{\hat{\delta}} + \ddot{\delta}_{1,3}\Delta_{\hat{\delta}}^2 \\
\dot{\delta}_2(\Delta_{\hat{\delta}}) &= \ddot{\delta}_{2,1} + \ddot{\delta}_{2,2}\Delta_{\hat{\delta}} + \ddot{\delta}_{2,3}\Delta_{\hat{\delta}}^2 + \ddot{\delta}_{2,4}\Delta_{\hat{\delta}}^3 \\
\dot{\delta}_3(\Delta_{\hat{\delta}}) &= \ddot{\delta}_{3,1} + \ddot{\delta}_{3,2}\Delta_{\hat{\delta}} + \ddot{\delta}_{3,3}\Delta_{\hat{\delta}}^2 \\
\dot{\delta}_4(\Delta_{\hat{\delta}}) &= \ddot{\delta}_{4,1} + \ddot{\delta}_{4,2}\Delta_{\hat{\delta}} + \ddot{\delta}_{4,3}\Delta_{\hat{\delta}}^2 + \ddot{\delta}_{4,4}\Delta_{\hat{\delta}}^3.
\end{aligned} \tag{3}$$

and the coefficients of the polynomial model are given in Table 2. The mean modeling error of $\hat{\delta}$ is .04 with a standard deviation of .04 computed by 10-fold cross-validation. Figure 9 depicts the speed of the subsequences in our data set as computed by our $\hat{\delta}$ polynomial model.

4.3 Model Selection

We exhaustively evaluated models with every combination of approximating curve polynomials of degree zero through three and coef.-accel. relationship polynomials, degree zero through four. Note that a degree zero or constant curve has only a single coefficient. In total 775 polynomial models were considered. We evaluated the modeling error of each by computing the mean absolute difference between the actual speed and the computed speed using 10-fold cross-validation. Subsequences of insufficient length were excluded as necessary. Cross-validation divides the entire set of subsequences into 10 subsets of subsequences (folds). Each fold is used once to test a polynomial model that is first trained on the 9 remaining folds. This results in 10 pairs of data sets for training and testing. The modeling error is computed over the test data. The polynomial with the lowest modeling error is the best model for $\hat{\delta}$ (speed) over $\hat{t}$ (time).

Table 2: Kinematic $\hat{\delta}$ polynomial model poly. 3 - 2, 3, 2, 3 (14) coefficients. Each row gives the coefficients of the polynomial fit to the c^{th} term of the approximating cubic as well as the standard deviation (SD) of the polynomial fit computed via cross-validation.

$\dot{\delta}_c(\Delta_{\hat{\delta}})$	$\ddot{\delta}_{c,1}$	$\ddot{\delta}_{c,2}$	$\ddot{\delta}_{c,3}$	$\ddot{\delta}_{c,4}$	**SD**
c=1	1.01	−.02	−1.19	-	.04
c=2	0	1.39	.23	−9.33	.14
c=3	−.03	.04	2.58	-	.1
c=4	.01	−.69	−.13	11.23	.21

The results of our polynomial model evaluation are summarized in Figure 10. The horizontal axis of the right plot indicates the total number of coefficients from one to twenty, given by counting the coefficients of each coef.-accel. polynomial of a model. For a given number of coefficients the test and training modeling error of the best model is indicated. Notice that as the number of coefficients increases the testing error drops at first then due to overfitting the testing error increases again. Fluctuations are due to under or over fitting where one or more coef.-accel. polynomials are fit by low or high degree polynomials, respectively. The best model for speed is reported in section 4.2.

5 Discussion

The relationship between speed and acceleration is a pattern in freehand animation that occurs between our key-frames. Our polynomial speed model captures this pattern giving animators a way to produce the speed functions that are found in freehand animation with only a single parameter. There are many ways these results can be used to improve 3D animation software. For example, non-linear projection functions might be used to tune speed functions algorithmically. At another extreme 3D animators might be taught how to produce 3D speed functions that project to the good 2D ones under conventional projection. Between these extremes, one can imagine improving the user interface of programs used to create 3D animation. In section 5.1 we present an experimental animation system that uses our polynomial speed model rather than a graph editor. We explore the

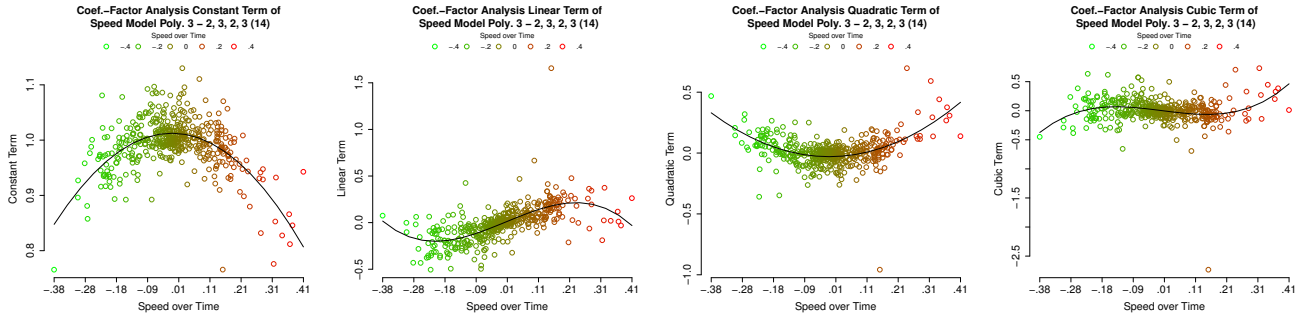

Figure 8: A polynomial (black curve) is fit to each coefficient of each term of the cubic approximation polynomials. Each point is the value of the coefficient over the subsequence's acceleration. The polynomials are of degree 2, 3, 2 and 3 from left to right.

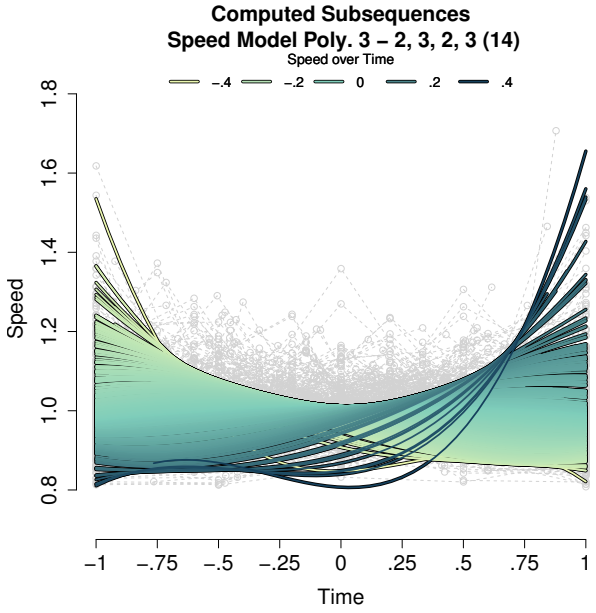

Figure 9: The δ (speed) over time computed by the polynomial model given by Equation 3. The colour of the cubic polynomials indicates their acceleration.

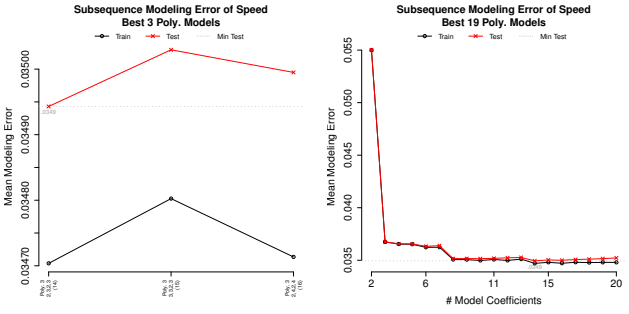

Figure 10: Best 3 and 19 polynomial models identified by exhaustive evaluation for $\hat{\delta}$ (speed).

potential sources of error in section 5.2, briefly discuss our statistical methods for validation in section 5.3 and discuss the difficulties in generalizing our results in section 5.4.

5.1 Animation System Implementation

In this section we demonstrate how our polynomial speed model, given by Equation 3, can be used in an animation system to compute the position of a single rigid object given its average acceleration by the animator. The object's position is given by the arc-length parameterization of its trajectory $p(w)$. Modern animation programs provide a variety of splines for specifying trajectories. For simplicity and realism we created trajectories by tessellating Catmull-Rom splines fitted to the trajectory samples in our data set. The Catmull-Rom spline pass through the positions marked for each salient part's centre-of-mass.

Modern animation programs provide a separate graph editor tool similar in appearance to that in Figure 1 to specify the object's arc-length $w(h)$ over time h using a spline. Evaluating $p(w(h_i))$ for frames $1 \leq i \leq N$ gives the object's position in each frame.

Our example interface, depicted in Figure 11, replaces the graph editor with a chart-based interface, inspired by the spacing-charts drawn by hand-animators to indicate the overall speed of a character between key-frames [18, p. 47]. The arc-length of the object at frame k with normalized speed $\hat{\delta}(t_k, \Delta_{\hat{\delta}})$ is computed with

$$\begin{aligned} \dot{s}_k &= \hat{\delta}(t_k, \Delta_{\hat{\delta}}) + \frac{1 - \sum_{j=1}^{N-1} \hat{\delta}(t_j, \Delta_{\hat{\delta}})}{N-1} \\ \dot{w}_i &= \sum_{k=1}^{i-1} \dot{s}_k \\ w_i &= \dot{w}_i l + m. \end{aligned} \tag{4}$$

In Equation 4 the normalization is first reversed to compute speed $\dot{s}_k$ that is proportional to the arc-length of the subsequence. Cumulative sum over the frames $1 \leq k < i$ gives the proportional arc-length $\dot{w}_i$. Scaling and shifting by the subsequence's length l and position m computes the arc-length w_i of the object on its trajectory. The object's position in frame i is given by evaluating $p(w_i)$.

Key-frames are created using our example interface in Figure 11 by marking their positions on the trajectory, indicated by the red dots, and specifying the frame number they occur on. Frame-times are determined by multiplying the frame number by the duration of each frame in seconds, usually $1/24^{\text{th}}$ of a second. Clicking on a subsequence in the chart causes it to be selected. The number of frames in the selected subsequence is specified by pressing the 'F+' button and dragging right to add frames and left to remove frames. Shortening the selected subsequence shortens the whole animation, causing subsequent key-frames to move to earlier times.

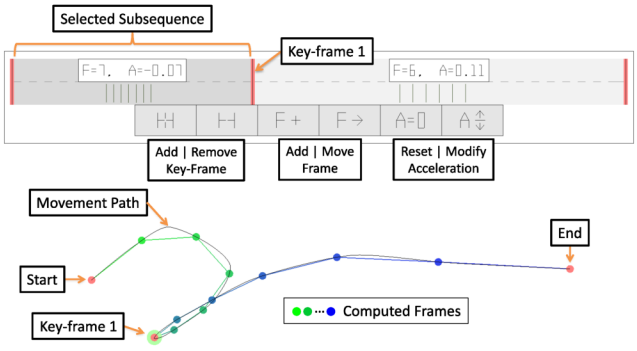

Figure 11: UI for controlling the speed of the example trajectory by specifying the number of frames (F) and acceleration (A).

Alternatively frames can be moved across a key-frame by pressing the 'F→' button and dragging right to send frames into the next subsequence and left to receive frames from the next subsequence. Each subsequence's average normalized acceleration is increased or decreased by pressing 'A⇳' and dragging up or down, respectively.

Our animation system would easily support the traditional approaches: straight-ahead action, pose-to-pose action, and their combination [18, p. 61-63]. The animator works in a straight-ahead manner by creating subsequences in succession, tuning each before creating the next. Pose-to-pose action is accomplished by specifying the duration of the entire animation then recursively specifying its key-frames. This is accomplished by breaking long subsequences into shorter ones by splitting the subsequence at an intermediate frame to create a new key-frame. The combination of these two approaches is also supported by first recursively breaking the trajectory into key-poses, then in succession identifying and tuning the intermediate subsequences.

5.2 Sources of Error

Variance can be observed in the subsequences in Figure 6 and in the correlations in Figure 8. There are several potential sources of error that may contribute to this variance. Hand-animators draw from frame-to-frame, referring back only two to three frames when deciding on where to make the next stroke. Further they usually do not use rulers or other guides. Consequently, there is a cumulative error present in the placement of each stroke. This would prevent animators from depicting the desired speed perfectly.

Our manual identification of the centre-of-masses may also contribute to the variance. Unlike the animator's drawing error our selection error does not accumulate from sample to sample on the trajectory. However, the variance of subsequences with low speeds could be inflated due to our normalization. At a mean subsequence speed of .5 and .25 cm/frame the variance is increased by 2 and 4 times, respectively. In our data set 26% and 1% of our subsequences have a mean subsequence speed of less than .5 and .25 cm/frame, while 50% of subsequences are larger than 1 cm/frame with a long tale to 10 cm/frame. Consequently, our manual identification of the centre-of-masses is not likely to influence our results.

5.3 Validation

To ensure that our polynomial speed model is accurate and representative we verified that our data set of subsequences is sufficiently large by analyzing how modeling error changes with respect to the size of the training set. By repeatedly computing the modeling error using cross-validation on larger and larger training sets we can address the question of whether more data would result in a better model. The analysis indicates that our test sets is sufficiently large to train the polynomial speed model given by Equation 3. However, the high level of variance in the analysis of makes it difficult to observe the stereotypical downward trend and plateau of the modeling error.

We want our polynomial speed model to be accurate and stable. The accuracy of a model is its bias, the difference between its mean and the actual mean in the data set. The stability of a model is its variance or standard deviation squared. We analyze the bias and variance by computing the linear regression line of the computed speed over the actual speed. We found that the linear regression slope is .795 and the correlation coefficient is .758 indicating that our polynomial model is stable and accurate. The computed speed is consistently accurate over its range of values except at the extremes where it is less accurate. The output does not explode or exceed the expected range for accelerations from -.73 to .6.

We analyzed the subsequence category annotations to ensure they are representative by examining the distribution of average accelerations to ensure that subsequences are assigned systematically. If the distributions are not distinct this may indicate a significant number of miscategorizations indicating that bias has influenced the annotations and the resulting polynomial model. Our analysis computes a pair of thresholds for classifying the categories automatically. The analyses indicates that our speed annotations match those automatically classified for 79.6%, 78.1% and 78.5% of subsequences for the ease-out, even and ease-in categories, respectively. The thresholds (mean / SD) of $-.1$ / .005 and .11 / .002 are also narrow compared to the range of accelerations -.73 to .6 overall indicating that subsequences with nearly zero average acceleration are usually categorized as even (constant velocity). The standard deviations of the thresholds, computed with cross-validation, are also not significant enough to affect the results.

We use a change-point detection method that makes use of our polynomial speed model to identify samples that mark significant changes in the parameters of a statistical model. Our algorithm iteratively scans through the frames identifying suspected key-frames. Comparing these computed key-frames to those annotated allows us to evaluate our model. We found that the computed key-frames match 76% of the annotated key-frames and the computed key-frames are within .075 seconds on average of the annotated key-frames with a standard deviation of .136 seconds. The category of the computed subsequences, determined by classification with the slope thresholds computed above, match 61.1% of those annotated. The agreement of the annotated and computed key-frames and categories indicates that the annotations are similar to those computed.

A potential issue is that the ANOVA used by our algorithm for change-point detection requires error to be normally distributed. Although we did not evaluate the error it is unlikely to be normally distributed. Another issue is that the time-window we computed for change-point detection, .46 seconds, is longer than 50% of our subsequences. However, the algorithm has the opportunity to identify short subsequences and appears to do so in practice.

Our analysis does not include a test with an independent data set. Although cross-validation is used to evaluate our polynomial model the selection of the trajectories and annotations is a potential source of bias. Unfortunately, we are unable to create our own independent data set or locate an existing data set of freehand animation trajectories with annotated key-frames.

5.4 Generalization

Our annotated key-frames, made in Section 3, match descriptions of key-frames [18, p. 48] and subsequences [13]. However, many different key-frame placement schemes are possible and actual key-frames were not available for this research. Consequently, it is not known if our key-frames occur near those created by animators. However, placement of key-frames may differ without invalidating our results if the break-points inferred from these key-frames are

similar to those we annotate. We did not examine the problem of inferring a set of break-points from a set of key-frames.

Our data set consist of freehand animations selected from different styles, animators, and eras. Our polynomial speed model is consequently a global average of the speed patterns used throughout this data set. The patterns found in individual animations, styles, and eras may differ from this global average. Additionally, our data set does not include animations produced with different methods such as computer, clay, paint-on-glass, sand or stop-motion. Consequently, our speed model may not generalize to other methods for creating animation.

6 Conclusion

We have identified, modeled and validated a relationship between speed and average acceleration in a data set of existing 2D freehand animation. As a consequence of training on freehand animation the behavior of our polynomial model is indirectly rooted in the principles of animation. We conclude from our results that changes in average acceleration occur in freehand animation where the speed, characterized by Equation 3, abruptly changes. We propose that these change-points are speed related key-frames. We also conclude that speed between a pair of key-frames is well represented by a cubic polynomial with coefficients that are related to average acceleration by polynomials of degree 2, 3, 2 and 3. This does not match the current intuition that a piecewise cubic is sufficient to control position over time on a trajectory.

The key-frames we identified tend to occur where the acceleration changes significantly. This describes a placement scheme for key-frames that may be useful for creating animation. For example, a single curve segment should not be used to make an object that increases in speed then decreases in speed; it would be difficult to separately control the acceleration and deceleration. In advance animators can use this placement scheme to determine that a key-frame should occur where the object's acceleration changes.

We can also conclude that the speed of a subsequence computed after specifying only one of the two control-points of a Bézier-based interface is an incorrect misleading distraction to the animator and not a useful intermediate. To control an object's speed it is clear from speed's relationship with average acceleration that both the linear and quadratic terms of the polynomial must be set simultaneously. However, the animator is limited to manipulating one control-point of a cubic Bézier curve one at a time.

An interface based on our model may ease the burden of controlling speed. Such an interface can be easily adopted by existing animation systems.

7 Future Work

We have analyzed the speed of subsequences identified by our annotation process. We did not however compare our annotated key-frames to actual key-frames created by animators. Such a test could falsify the speed model presented here. Such a comparison is an important step for any future research. Repeating our annotation, analysis and modeling on individual animations, styles, or eras as well as animation by different methods such as clay, paint-on-glass, or 3-D computer animation would be useful for determining how well our model generalizes.

Our polynomial speed model characterizes one aspect of animation. Further research can also be made into other dependent variables such as direction, orientation, scale or non-linear shape deformations. Identifying the factors that influence speed and other dependent variables will enable new user-interfaces and algorithm approaches for creating animation.

Further research can also be made into the simultaneous movements in a scene. For example, scenes often create a focal point to draw the audience's attention by moving parts toward or away from a specific point on the screen. It may be possible to identify and model this phenomenon and others like it so that similar results could be achieved with fewer inputs from the animator.

Acknowledgments

We thank the reviewers for the helpful feedback.

References

[1] E. Castet. Apparent speed of sampled motion. *Vision Research*, 35(10):1375–1384, 1995.

[2] J. Chai and J. K. Hodgins. Constraint-based motion optimization using a statistical dynamic model. In *ACM SIGGRAPH 2007 Papers*, SIGGRAPH '07. ACM, New York, NY, USA, 2007.

[3] L. Ciccone, M. Guay, M. Nitti, and R. W. Sumner. Authoring motion cycles. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '17. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2017.

[4] P. Coleman, J. Bibliowicz, K. Singh, and M. Gleicher. Staggered poses: A character motion representation for detail-preserving editing of pose and coordinated timing. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pp. 137–146. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2008.

[5] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović. Style-based inverse kinematics. *ACM Transactions on Graphics (TOG)*, 23(3):522–531, August 2004.

[6] E. Hsu, M. da Silva, and J. Popović. Guided time warping for motion editing. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '07, pp. 45–52. Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007.

[7] A. Kort. Computer aided inbetweening. In *2nd international symposium on Non-photorealistic animation and rendering*, NPAR '02, pp. 125–132, 2002.

[8] J.-y. Kwon and I.-K. Lee. An animation bilateral filter for slow-in and slow-out effects. *Graphics Models*, 73(5):141–150, September 2011.

[9] J. Lasseter. Principles of traditional animation applied to 3d computer animation. *SIGGRAPH Computer Graphics*, 21(4):35–44, August 1987.

[10] R. Parent. *Computer Animation: Algorithms and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, USA, 2nd ed., 2007.

[11] V. S. Ramachandran and S. M. Anstis. The perception of apparent motion. *Scientific American*, 254(6):102–109, 1986.

[12] D. Sýkora, J. Dingliana, and S. Collins. As-rigid-as-possible image registration for hand-drawn cartoon animations. In *7th International Symposium on Non-Photorealistic Animation and Rendering*, NPAR '09, pp. 25–33, 2009.

[13] F. Thomas and O. Johnston. *The illusion of life: Disney animation*. Hyperion, New York, USA, 1995.

[14] THX. Thx certified cinema screen placement. `http://www.thx.com/professional/cinema-certification/thx-certified-cinema-screen-placement/`, 2017.

[15] B. Walther-Franks, M. Herrlich, T. Karrer, M. Wittenhagen, R. Schröder-Kroll, R. Malaka, and J. Borchers. Dragimation: Direct manipulation keyframe timing for performance-based animation. In *Graphics Interface 2012*, GI '12, pp. 101–108, 2012.

[16] J. Wang, S. M. Drucker, M. Agrawala, and M. F. Cohen. The cartoon animation filter. *ACM Transactions on Graphics (TOG)*, 25(3):1169–1173, July 2006.

[17] D. White, K. Loken, and M. van de Panne. Slow in and slow out cartoon animation filter. In *ACM SIGGRAPH 2006 Research posters*, SIGGRAPH '06. ACM, New York, NY, USA, 2006.

[18] R. Williams. *The animator's survival kit*. Faber and Faber Limited, Bloomsbury House, London, UK, expanded ed., 2009.

[19] I. Yoo, M. Abdul Massih, I. Ziamtsov, R. Hassan, and B. Benes. Motion retiming by using bilateral time control surfaces. *Computers & Graphics*, 47:59–67, April 2015.