Temporal Distortion for Animated Transitions

Pierre Dragicevic, Anastasia Bezerianos, Waqas Javed, Niklas Elmqvist, Jean-Daniel Fekete

Temporal Distortion for Animated Transitions

Pierre Dragicevic[1], **Anastasia Bezerianos**[2], **Waqas Javed**[3],
Niklas Elmqvist[3] **and Jean-Daniel Fekete**[1]
dragice@lri.fr, anastasia.bezerianos@ecp.fr, {wjaved, elm}@purdue.edu, fekete@inria.fr
[1]INRIA Orsay, France [2]École Centrale Paris Paris, France [3]Purdue University West Lafayette, IN, USA

(a) Constant speed. (b) Slow-in/Slow-out. (c) Fast-in/Fast-out. (d) Adaptive speed.

Figure 1. Different temporal distortion strategies for animated transitions. Solid shapes show original positions, faded shapes show ending positions.

ABSTRACT

Animated transitions are popular in many visual applications but they can be difficult to follow, especially when many objects move at the same time. One informal design guideline for creating effective animated transitions has long been the use of slow-in/slow-out pacing, but no empirical data exist to support this practice. We remedy this by studying object tracking performance under different conditions of temporal distortion, i.e., constant speed transitions, slow-in/slow-out, fast-in/fast-out, and an adaptive technique that slows down the visually complex parts of the animation. Slow-in/slow-out outperformed other techniques, but we saw technique differences depending on the type of visual transition.

ACM Classification Keywords

H.5.1 Multimedia Information Systems: [animations]; H.5.2 User Interfaces: [Graphical user interfaces (GUI)]

Author Keywords

Animated transitions, animation, information visualization.

General Terms

Design, Human Factors

INTRODUCTION

Animated transitions, where the transformation between visual states is conveyed with smooth rather than abrupt visual changes [17], are increasingly being used in modern interaction design. For example, the photo viewer Picasa smoothly expands and collapses image galleries, the Pivot browser [25] fluidly moves visual entities during faceted browsing, and different information visualization systems use animated transitions when switching between data dimensions [10, 14], visual representations [16, 38], or when navigating in time [9, 12, 26]. Research suggests that animated transitions not only improve the aesthetics of a user interface, but also helps users to understand the underlying data [5, 16, 35]. However, there are many parameters involved in designing effective animations, including motion paths, staging, scheduling, and timing. In this paper, we focus on the latter: timing aspects of animated transitions.

Rather than having objects move or change at a fixed rate during an animation (Figure 1(a)), cartoon animators sometimes use a "slow in" or "slow out" effect [8, 18], causing more frames to be dedicated to the beginning or end of the animation (Figure 1(b)). Essentially, slow-in and slow-out *distort* time throughout the animation. Computer applications have been quick to adopt this idea [20], and many graphical toolkits (e.g., [6, 15]) and animation packages (e.g., Microsoft PowerPoint and Autodesk Maya) use a combination of slow-in and slow-out (SI/SO) as their default animation pacing. Its use has also been advocated for optimizing animations in user interfaces [2, 8, 17, 33, 38].

There are several arguments for using SI/SO, one being realism. However, physical realism is generally less crucial in graphical user interfaces than in cartoon animation. Another, more practical reason often cited for using SI/SO pacing is that it helps users to anticipate the beginning and ending of the animation. However, no perceptual studies have been performed to confirm this informal design rule. In particular, SI/SO dedicates less frames to the middle segment — effectively accelerating it – so it is not clear whether it should be used in all cases, especially when the middle animation segment is visually complex or particularly important.

In this paper, we address this lack of empirical data by comparing object tracking performance in visually cluttered animations under different temporal distortion strategies, and show how effective these strategies are for important low-

CHI 2011, May 7–12, 2011, Vancouver, BC, Canada.

level tasks. We expect our results to help improve the animated transitions used in user interfaces and information visualization systems so that they become more useful to users.

BACKGROUND

Animation is the rapid display of static sequences to create the illusion of movement. It has been extensively studied by psychologists and applied to both films and computer graphics as well as for improving user interfaces.

Perception of Movement

Perceiving and interpreting motion is a fundamental capability of human perception with deep roots in our evolution: moving objects stand out in our visual field, and the Gestalt principle of common fate states that entities moving in the same direction are seen as a unit [28]. Perception research suggests that the human visual system is capable of tracking multiple objects simultaneously [7, 30]. The actual tracking is performed by the visual system using a mechanism known as *smooth pursuit* [28]. However, many factors influence this capability, including the number of distractors, object speed, occlusions, and motion paths of the objects being tracked.

Cartoon Animation

Designing animations was an active practice long before computers, and there is a wealth of literature, tradition, and design guidelines to draw upon for designing effective animations. In particular, Johnston and Thomas presented the "12 basic principles of animation" in their seminal work *The Illusion of Life: Disney Animation* [18], discussing effective ways of making animations — particularly character animations — as realistic and as lifelike as possible.

Because most animation nowadays is done with computers, much effort has been devoted to transfering these principles to computer animation [24]. Accordingly, major computer animation packages such as 3D Studio MAX, AutoDesk Maya, and even Microsoft PowerPoint, today adopt most of the above basic principles of animation in their algorithms.

Animation in User Interfaces

Animations have been used in interactive applications since the advent of graphical user interfaces [2]. Before GUIs, programmers would animate terminal output, e.g., to show progress in a time-consuming task. In 1993, Chang and Ungar [8] as well as Hudson and Stasko [17] concurrently proposed applying cartoon animation to interfaces. Thomas and Calder [33, 34] have since further improved upon this idea.

Similarly, efforts towards evaluating animated transitions date back to the early 1990s. Gonzalez [13] performed an experiment to show that use of animated transitions in graphical user interfaces can improve the user's decision making process. Later, Bederson and Boltman [5] empirically measured the effect of animated transitions on user capability to build mental maps of spatial information. Interestingly, Tversky et al. [35] study the use of animation in graphical interfaces, and find that its use is mostly detrimental — except for congruence of motion. Building on this result, Heer and Robertson [16] present results from two user studies that emphasize the importance of animated transitions for highlighting changes between related statistical data graphics. In a similar study, Shanmugasundaram et al. [32] found that smooth transitions dramatically improved performance for memorizing graphs using node-link diagrams.

However, all of the above studies investigate the benefits of adding animation to an interactive application, whereas there exists very little work that compares the performance of different *types* of animations. Notable exceptions focus on using animations as notification mechanisms [1, 4]. In particular, we are aware of no existing work studying how temporal pacing for animation affects object tracking performance. While best practice design guidelines suggest the use of smooth animations with slow-in/slow-out extrema, no formal evaluation exists that verifies that this is indeed the optimal pacing strategy for animated transitions.

TEMPORAL DISTORTION IN ANIMATION

Computer animations can be modelled as parametric graphics with time-varying parameters (object positions, size, color, opacity, etc.). The simplest way to animate parametric graphics between two visual states is by linear interpolation, i.e., for each parameter p changing from p^0 to p^1:

$$p(t) = p^0 + t(p^1 - p^0), \quad t \in [0, 1]$$

where t is a global parameter that controls the speed and duration of the entire animation. When p is an object's position, this formulation produces trajectories that follow straight paths. If objects follow non-straight paths [18, 24], arc-length parametrization can be used instead.

On a computer screen the animation must be sampled. Assuming a constant frame rate f and given a total time T for the animation, we need $n = f \times T$ frames to complete the animation. Typical values for f are 60 Hz and typical durations of T are 0.5–1.0 seconds for transitions [16]. The pacing of an animation is defined by a list of n values of t:

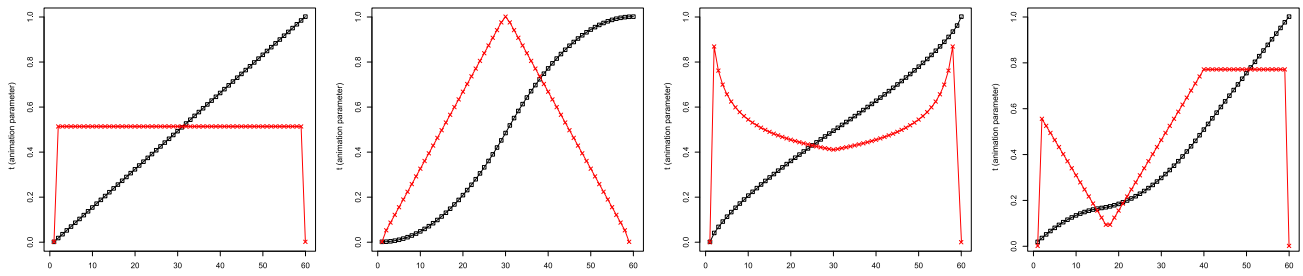
$$t \in \{t_1, t_2, \cdots, t_n\}$$

Setting $t_1 = 0$ and $t_n = 1$, the increments $\Delta t(i) = t_i - t_{i-1}$ for $i \in [2, n-1]$ are left in the hands of the animation designer. Below we will discuss different strategies for determining these increments of the animation parameter t.

Constant Rate

Choosing a fixed $\Delta t = 1/(n-1)$ results in an animation with constant rate throughout its duration (Figure 2(a)). When an object's position is animated, this pacing yields a constant object velocity. In a collection of moving objects, all objects start and stop at the same time, but move with different velocities depending on the distance they must travel.

Constant rate animation has several advantages: it is easy to implement and yields predictable motion because the initial speed of an object suggests its final destination [31]. However, it also produces high accelerations and decelerations on the first and last frames, and has a mechanical look that has been referred to as the "computer signature" [20].



(a) Constant rate. (b) Slow-in/Slow-out. (c) Fast-in/Fast-out. (d) Adaptive rate (frame 18).

Figure 2. Evolution of the animation parameter t (black squares) and animation rate Δt (red crosses) for 4 different temporal pacing strategies in an animation consisting of $n = 60$ frames. The animation rate is normalized across all 4 techniques to allow for comparison. For adaptive speed, the technique has detected a complexity peak at frame 18. Note that all techniques except SI/SO have abrupt speed changes at the start and end.

Using a varying $\Delta t(i)$ – or equivalently, applying a transformation $t(i) \mapsto t'(i)$ to the constant pacing [17] – can be used instead in order to emphasize different parts of the animation. Below we describe some approaches to achieve this.

Slow-in/Slow-out

The concept of slow-in/slow-out (SI/SO) [18, 20] uses $t(i)$ values that devote more frames (smaller increments of t) to the endpoints of the animation, causing the motion to gradually speed up then slow down. Although many SI/SO pacings are possible, we use the following quadratic transformation of the linear pacing, illustrated in Figure 2(b):

$$t' = \begin{cases} 2t^2 & \text{if } t \leq 0.5 \\ 1 - 2(1-t)^2 & \text{if } t > 0.5 \end{cases}$$

SI/SO is a recommended animation pacing for transitions in user interfaces [8, 17, 38] and is used by default in several applications — e.g., MS PowerPoint, 3D Studio MAX and AutoDesk Maya — as well as graphical libraries like Piccolo [6] and Prefuse [15]. Beyond aesthetics, the supposed benefits of SI/SO are (i) that the gradual speed increase at the beginning of the animation helps users to start tracking an animated object, and (ii) that the decreasing speed at the end allows users to predict when an object will stop moving.

Fast-in/Fast-out

One potentially harmful effect of SI/SO is that it accelerates the middle of the animation, which can be problematic if something important occurs at this point. The middle of an animation can also be more cluttered and more difficult to follow, which happens for example when many points move from a random location to another [16]. In this case, one intuition would be to slow down the midpoint of the animation instead of slowing down its endpoints. This not only reduces object velocity, but it also smoothens object motion, which can help visually resolving cluttered animations [33].

We define fast-in/fast-out (FI/FO) animation as the dual of SI/SO, slowing down the animation in the middle as opposed to at its extrema. Again, many designs are possible but we use the following function to achieve this effect (Fig. 2(c)):

$$t' = \begin{cases} \frac{(2t)^{0.75}}{2} & \text{if } t \leq 0.5 \\ 1 - \frac{(2(1-t))^{0.75}}{2} & \text{if } t > 0.5 \end{cases}$$

Of course, this strategy has the unfortunate side effect that objects will move at their fastest speed at the beginning and end of the animation, presumably making it difficult to start and stop tracking any of them. In other words, this idea completely abandons the predictability property of SI/SO animation in favor of reducing visual complexity in the middle.

Adaptive Rate

SI/SO slows down the animation to emphasize its endpoints whereas FI/FO slows down its middle. To strike a balance between these two, we designed an adaptive technique that dynamically selects the frames with the highest visual complexity and slows down the animation around these frames. Figure 2(d) showcases this idea with one visual complexity peak at frame 18. We use an exponential function to slow down (but not completely stop) the animation around this frame. The technique is a generalization of both SI/SO and FI/FO: with peaks at the extrema or at the midpoint of the animation, the technique will reduce to the former or latter.

The remaining design parameters in this scheme are the calculation of visual complexity and the selection of peaks. For the former, complexity depends on the particular type of animated transition and the type of task. In our study, we focus on tracking points in animated point clouds, so for complexity we use a simple distance metric calculated on a per-frame basis for the constant rate animation:

$$D(t) = \sum_i \min_{j \neq i} ||p_i(t) - p_j(t)||$$

where $p_i(t)$ is the position of the point i at t in the constant rate animation. The intuition is that a frame t with low D is complex to follow because a large portion of the objects are located very close to each other, resulting in high degrees of occlusion and making single object tracking difficult.

The final step is to select a small set of frames representing visual complexity peaks, and to slow down the animation speed around these. Bear in mind that given a finite animation time T, slowing down one moment of an animation means speeding up another, so this should be done sparingly. We use a greedy scheme that keeps selecting peaks of high visual complexity (to a particular maximum) as long as they fall within a specific proximity (90-95%) of the most com-

plex frame. This will result in at least one peak — the most complex one — being selected for slow-down. Because visual complexity often changes smoothly throughout an animation, we also enforce a minimum peak separation to avoid one portion of an animation to dominate the peak selection.

USER STUDY BACKGROUND

Here we present several decisions we made in the design of our user study, as well as some first exploration results.

Task Rationale

The goal of our user study is to determine the best way to present animated transitions in terms of temporal distortion, and we thus evaluated how accurately users understood different types of transitions under different temporal distortion schemes. Although what is exactly meant by "understanding" a visual transition is a matter of debate, a common experimental task is to have subjects track an object among others [9, 16]. This is an elementary low-level task, ensuring that if users are unable to perform it, then more complex tasks — e.g. following multiple independent objects or groups of objects — will be equally or more difficult. It is also safe to assume that many higher-level tasks will be difficult to perform if single objects cannot be tracked. For example, reading a scrollable text aloud is likely to be difficult if individual words cannot be tracked during scrolling.

Since our task is tracking an object among others, we focus on visual transitions that involve moving objects. Object translation is the most basic operation in animations. Other examples are changes in color or shape, but to keep the task simple we focused on moving objects that do not change during transitions. To further ensure that perceptual phenomena not directly related to animations such as object recognition and preattentive processing [37] will not interfere with the task, all objects were visually identical. One can assume that if an animation is effective for identical objects, it will also be effective for dissimilar objects. Arguably, in real applications users can highlight objects to facilitate tracking, but interactive highlighting is an orthogonal and complementary approach to animation: effective animations with highlighting support are preferable to poor animations with highlighting support, because they better help users follow objects that are not of immediate interest to them — hence providing context and facilitating incidental discoveries — and saves time by not requiring explicit object selection.

We therefore chose as visual transitions large sets of small objects that move from one location to another, i.e., *point cloud transitions*. This type of transition captures common visualization applications, such as scatterplot-based data exploration (e.g. [10, 12, 26]) and faceted browsing of object collections (e.g. [14, 25]). Finally, since we focus on time and not space, points move on straight paths. Although other trajectories have been proposed [10, 24], this is a simple approach that has been widely employed so far [12, 14, 25].

Datasets

We use two datasets of point cloud transitions. The first one is randomly generated. User studies sometimes involve randomly generated data (e.g., synthetic graphs [27] or images [11]) because compared to real-world data, it is easier to describe in a reproductible way, eliminates the problem of selecting unbiased case scenarios — which is challenging when real-world data is very heterogeneous — and allows for better experimental control. Since we do not know of a method for randomly generating structured point cloud transitions, we introduce such a method. We also use another set of point cloud transitions taken from a real information visualization dataset (described in the *User Study* section).

A *random point cloud* is defined by 6 parameters:

- $\boldsymbol{r}$ is the random seed,
- $\boldsymbol{n}$ is the number of points in the cloud,
- $\boldsymbol{c}$ is the amount of point clustering,
- $\boldsymbol{p}$ is the clustering power,
- $\boldsymbol{s}$ is the separation between points, and
- $\boldsymbol{i}$ is the number of iterations.

Figure 3 illustrates some values for $\boldsymbol{c}$ and $\boldsymbol{s}$. A *random point cloud transition* is defined by the parameters above — which serve to generate the initial and final point clouds — plus the parameter $\boldsymbol{m}$ or *motion coherence* that affects how the final point indices are mapped to the initial ones (Figure 4). Details on the dataset generation are described in an appendix.

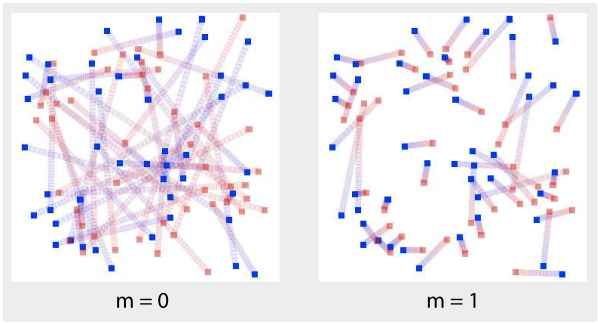


Figure 4. Two transitions between the same point clouds, one with low motion coherence ($m = 0$) and one with high motion coherence ($m = 1$). Point trajectories are shown with trails going from blue to red.

Task Difficulty

We first set out to discover what makes a point cloud animation easy or hard to follow with regards to tracking single points. We experimented with various randomly generated point cloud transitions and although we did not find any perfectly reliable predictor of task difficulty, we found that the number of object crossings strongly affected difficulty. We define *distractor count* or NDIST as the number of objects which cross the object of interest over the course of the animation. Incidentally, NDIST also captures the density of the point cloud (in denser clouds we expect more crossings), as well as the distance traveled by the target (the longer the distance, the more distractors are likely to cross the target).

Pilot Study

We conducted a pilot study investigating the traditional pacing profiles: constant speed and SI/SO. For this study we generated 100 random point cloud transitions by varying the

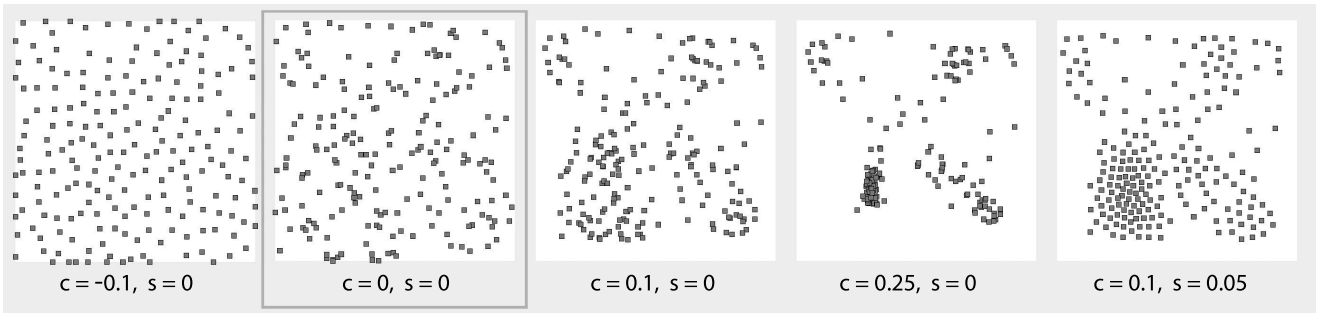


Figure 3. Examples of point clouds generated with parameter values *r*=0, *n*=200, *p*=-2, *i*=60, and various values of *c* (clustering amount) and *s* (point separation). The second point cloud shows a true pseudo-random distribution.

parameters ***r***, ***n***, ***c***, ***p***, ***s***, ***i***, and ***m***. We ensured that the generated trials had different instances of NDIST values. All animated transitions were limited to 1 second, a common duration for even complex animated transitions [16]. The task consisted of tracking a point, and was identical to the one used in the full study. Four subjects participated in our pilot.

We explored the data using ScatterDice [10] and first observed that all tasks for which all 4 users were successful using constant rate were also successfully completed by all 4 users using SI/SO. We then examined tasks for which SI/SO yielded more successful trials than constant rate. These were all tasks where the target point (i.e., the point to follow) was surrounded by other points (e.g., inside or next to a cluster) at the beginning and/or at the end of the animation. We hence postulated that SI/SO worked well primarily because it allowed users to better see what happens when the animation was the most difficult to follow. Inspired by this finding, we designed the adaptive speed technique and decided to also include the FI/FO technique (the dual of SI/SO) in the study.

Distractor Profiles

To test whether temporal distortion helps in following complex parts of an animation, we decided to explicitly include trials where animations are the most complex in the middle or at the endpoints. We thus introduce the measure *distractor profile* or DISTPROF, which captures the evolution of animation complexity in time, i.e., whether it is mostly complex at the endpoints or the middle (for a given target point to track):

$$\text{DISTPROF} = \frac{(n_1 + n_3)/2 + 1}{n_2 + 1}$$

where n_1 is the NDIST measure for $t \in [0, \frac{1}{3}[$, n_2 is NDIST for $t \in [\frac{1}{3}, \frac{2}{3}]$ and n_3 is NDIST for $t \in]\frac{2}{3}, 1]$. An animation that is mostly complex at the endpoints (e.g., the target goes from a dense cluster to another) yields DISTPROF > 1. An animation that is mostly complex at the middle (e.g., the target point goes through a dense cluster) yields DISTPROF < 1.

Error Metric

As our transitions were of fixed duration, we chose to not measure completion time but rather accuracy in tracking the target. Our *selection error* or ERROR metric is defined as:

$$\text{ERROR}(p_s, p_t^1) = \frac{err(p_s, p_t^1)}{E(err(p_i^1, p_t^1))}, \quad err(a, b) = ||a - b||$$

where p_t^1 is the position of the target in the final point cloud, p_s that of the user selection, and $E(err(p_i^1, p_t^1))$ is the expected error that would have been measured had the participant given a random answer, and is equal to the average distance of all points from the target. Thus, an average value of 1 means users guessed the answer whereas a value well above 1 could mean users were misled by the animation.

The definition of the ERROR metric is such that if the user selects the correct target, her error is 0. If she selects an object close to the target, her error is smaller than when selecting an object far from the target. Thus, good approximate selections are penalized less than completely random ones. Moreover, the normalization with the expected error ensures that in cases where most objects are close to the target, as in a tight object cluster, an arbitrary object selection within the cluster does not give too small an error.

USER STUDY

Given the above background, we designed a controlled experiment to formally evaluate object tracking performance under different temporal distortion strategies. We describe the details of this experiment in this section.

Task Generation

A *task* consists of a *point cloud transition* and a *target*, i.e., a particular point to follow. We included two task datasets (one generated, one real) with different properties in terms of structure. The generated dataset (see Figures 3 and 4) was used to ensure we could fully test the different techniques under different distractor profiles. The use of the real dataset ensured that our findings were generalizable to real life data.

Generated: We generated a set of random transitions with $\boldsymbol{n} = 200$, $\boldsymbol{c} \in [-0.25, 0.5]$, $\boldsymbol{p} = -2$, $\boldsymbol{s} \in [0, 0.05]$, $\boldsymbol{i} = 60$ and $\boldsymbol{m} \in [0, 1]$. For each of these transitions, we generated a task by randomly selecting a target point that (i) is not occluded by other points at the first and last animation frame and (ii) travels a minimum distance of 0.5 — i.e., half of the point cloud size. We then pruned tasks for which NDIST < 15 in order to further ensure that they were not too easy. We grouped these tasks into into three bins (d_e, d_m, d_o). Tasks in the first bin d_e are complex at the two endpoints (DISTPROF > 1), and we chose ones with DISTPROF $\in [5, \infty[$. In bin d_m tasks are complex in the middle ($0 <$ DISTPROF < 1) with DISTPROF $\in [0, 0.5]$. Finally,

tasks in d_o are close to constant complexity (DISTPROF ~ 1) and we chose ones with DISTPROF $\in [0.8, 1.2]$. The chosen margins for the values of DISTPROF between bins ensures that tasks differ enough in their distractor profile. We randomly selected 12 tasks per bin and obtained a total of 36 randomly-generated tasks.

Real: The second dataset is a high-dimensional dataset of digital cameras[1] (1,038 objects and 8 dimensions per object). A point cloud transition in this dataset depicted the change between two scatterplots of different dimension pairs. To select tasks that follow the distractor profiles we wanted to explore, we generated about 800 potential tasks by combining (i) random transitions (combinations of dimensions), and (ii) random objects as targets, always ensuring that the target was not covered by other objects at the first and last animation frame. Since values for DISTPROF were very close to 1, we used only two bins (d_e, d_m) for which we selected the 12 tasks with highest DISTPROF (complexity at the extrema) and the 12 with lowest DISTPROF. The selected bins verified DISTPROF > 1.2 and DISTPROF < 0.8 respectively.

Procedure and Apparatus

Participants were first shown the initial point cloud. They were asked to press the Space bar to highlight the target in red. After releasing Space, the highlighting of the target would disappear (but it persisted for at least 1 second). When subjects were ready to begin the trial they pressed Space again, after which all objects were animated to their final state. Participants were then asked to use the mouse to select the target in this final state as accurately as possible.

The experiment was conducted on a desktop computer equipped with a mouse, keyboard, and a 19" LCD monitor (1280×1024 resolution, 60 Hz refresh). Point clouds were shown in a 800×800 rectangular area, with points being dark gray 16-pixel squares. Animations lasted one second.

Experimental Design and Procedure

12 participants (paid university students) were randomly assigned to one of 4 groups. Each group used all 4 techniques described in the design space section in an ordering balanced using a Latin square. Tasks were selected as described previously, and were repeated across techniques. To avoid learning, task order was randomized across techniques, and point clouds were rotated by 90^o between techniques.

Prior to each technique users were given brief instructions, without explaining the implementation details of each technique, and performed a short warm-up session (2 trials) to familiarize themselves with the technique. The experiment lasted on average 45 minutes and had the following design:

	12	participants
	4	TECH (*C, SI/SO, FI/FO, A*)
	2	DATASET (*Generated*, *Real*)
	3, 2	DISTPROF (d_e, d_m, d_o)
$\times$	12	repetitions
	2280	total trials

[1]Collected from `http://www.dpreview.com/`.

RESULTS

Trials were marked as outliers when ERROR was beyond 3 standard deviations from the mean for a given subject, TECH and DISTPROF (1% of all trials), and were removed from further analysis. The remaining trials were aggregated per subject for each combination of conditions, and followed closely the normal distribution.

Error (Figure 5)

Generated Dataset

ANOVA showed a significant effect of TECH on ERROR ($F_{3,33} = 35.1$, $p < .0001$). Post-hoc pair-wise means comparison (all adjustments Bonferroni) showed that *SI/SO* and *C* were significantly different from each other and from all other techniques (all $p < .01$). Mean ERROR was lowest for *SI/SO* (0.28), then *C* (0.49), *A* (0.63), and *FI/FO* (0.65).

A significant TECH $\times$ DISTPROF interaction was present ($F_{6,66} = 4.4$, $p < .0001$). Pair-wise means comparison (all $p < .05$) showed that *SI/SO* was significantly better than all other techniques across distractor profiles. However, the results differ for the remaining techniques. Specifically under the d_m and d_o distractor profile, *A* performs significantly worse than *C*, but not in the d_e distractor profile case.

Real Dataset

ANOVA on the real dataset yielded a significant effect of TECH on ERROR ($F_{3,33} = 17.6$, $p < .0001$). Pair-wise means comparison (all $p < .05$) showed that *SI/SO* was significantly more accurate than *C* and *FI/FO*, with no significant difference between *SI/SO* and *A*. Contrary to the generated dataset, mean ERROR was less for *SI/SO* (0.25), followed this time by *A* (0.32), *C* (0.39), and *FI/FO* (0.51).

A significant TECH $\times$ DISTPROF interaction was present ($F_{3,33} = 2.9$, $p < .05$) for the 2 distractor profiles. Pair-wise means comparison (all $p < .05$) showed that for the d_e distractor profile, the trends follow that of the main analysis (*SI/SO* not different from *A*, but better than *C* and *FI/FO*). However, for the d_m distractor profile, we found that *SI/SO* was significantly better than *A*, but not *C*.

Correct Answers (Figure 6)

Although we were mainly interested in the selection error ERROR, we investigated perfectly correct trials (trials where ERROR $= 0$) to see the percentage of correct answers per technique and distractor profile.

Error rates were relatively high compared to the many user studies that focus on completion time. This is because we only measure errors: had we measured low error rates (by giving trivial tracking tasks) we would have seen little or no difference between techniques. Note that the relatively high difficulty of our tasks is not artificial, as many graphical applications display large numbers of objects and animate them very rapidly so as not to slow users down (e.g., [25]).

Generated Dataset

The number of correct answers (303 overall) was higher for *SI/SO* (38%), followed by *C* (31%), *FI/FO* (15%), and *A*

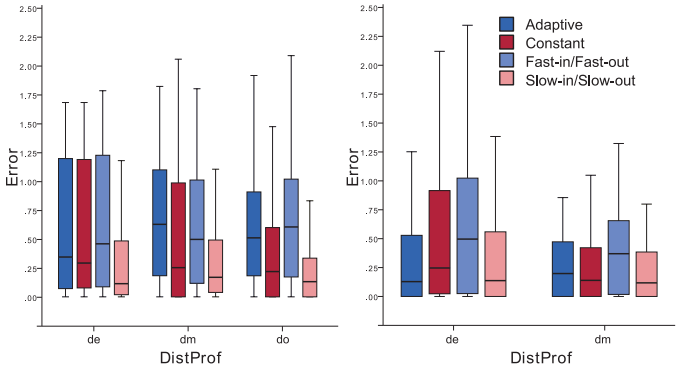


Figure 5. Mean selection error ERROR **grouped by distractor profile** DISTPROF **for all** TECH**. Generated (left) and real** DATASET **(right).**

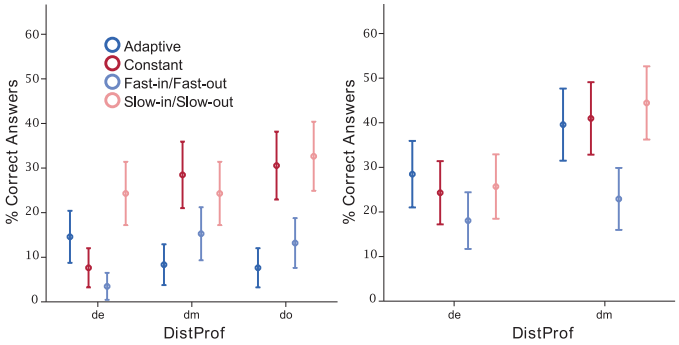


Figure 6. Percent of correct answers grouped by DISTPROF **for all** TECH**. Generated (left) and real** DATASET **(right).**

(14%). There was a significant effect of TECH ($F_{3,33} = 21$, $p < .0001$), with *SI/SO* and *C* having more completely correct trials than *A* and *FI/FO* (all $p < .05$). A significant TECH $\times$ DISTPROF ($F_{6,66} = 5.1$, $p < .0001$) interaction showed that this trend was not followed in the d_e distractor profile, where *SI/SO* was significantly better than *C* as well, but not *A* (all $p < .05$). By examining the correct trials in detail, we found that if a task was completed correctly in any of the techniques (303 correct trials corresponding to 17 tasks), it was also performed correctly using *SI/S0* in 95% of tasks.

Real Dataset

The number of correct answers (352 overall) was higher for *SI/SO* (36%), then for *A* (34%), *C* (32%), and finally *FI/FO* (20%). There was a significant effect on TECH ($F_{3,33} = 12.2$, $p < .0001$), where *SI/SO* had significantly more correct trials than *C* and *FI/FO*, but not *A* (all $p < .05$). No interaction effect with distractor profile was present. We further found that if a task was completed correctly in any of the techniques (352 trials corresponding to 24 tasks), it was also performed correctly with *SI/SO* in 84% of tasks.

DISCUSSION

We can summarize the results from the study as follows:

- The SI/SO animation pacing technique has better performance (i.e., significantly higher accuracy) than any of the other techniques tested for all distractor profiles;
- The other pacing techniques, while all less accurate than SI/SO, have different performance depending on the distractor profile; in particular, we see the following:
- Adaptive speed pacing works best for transitions with high complexity at the extrema of the animation (d_e), where it basically reduces to SI/SO; and
- Constant speed motion is better than adaptive speed and FI/FO for all other distractor profiles (d_m and d_o), second only to SI/SO.

In the following section, we try to explain and generalize these results. We also try to offer some recommendations for designers planning to use animation in interaction design.

Explaining the Results

There seems to be two primary and conflicting principles at work for how to effectively design the temporal pacing of linear animated motion of point clouds:

P1 **Allocate frames to endpoints:** Spend the majority of the frame budget on the beginning and end of an animation to allow users to anticipate motions; or

P2 **Allocate frames to complex segments:** Spend the frame budget on segments of an animation that are visually complex, either by calculating the complexity (adaptive) or by observing typical point cloud transitions (fast-in/fast-out).

Principle P1 has so far been the dominant approach in animation literature, practice, and tradition, whereas P2 has barely received any attention at all. One of the conclusions of the present work should be that this emphasis on the endpoints of an animation has been justified, and that a strategy based on adapting animation speed depending on frame complexity will only be successful when those complex frames happen at the animation endpoints. In other words, easing in and out of an animation seems much more important than slowing down and speeding up around frames of high visual complexity. The question is of course why this is the case.

A common explanation is that gradually accelerating and then decelerating motion aids tracking the object as well as helps the user in anticipating the beginning and ending of the animation. In particular, if the user is not expecting the animation to happen in the first place, a slow and gradual start will help the user to detect that the animation is starting and to adjust to the tracking task accordingly. In other words, *predictability* seems to be one of the key features named for slow-in/slow-out pacing, *not* that the slow movement allows users to better decipher complex parts of the animation (P2 above). To begin to find an explanation for why predictability is so dominant in this task, it is necessary to delve into the workings of the human visual system.

Human Vision and Smooth Pursuit

Voluntary eye movement can be performed in only two separate ways: *saccadic movement* and *smooth pursuit* [21, 28], where the latter is the method employed during object tracking (at least for objects moving slowly enough that catch-up saccades are not necessary [3]). Research in smooth pursuit shows that it consists of two stages [22]: *open-loop* and

closed-loop pursuit. Open-loop pursuit is the first visuomotor response to motion and typically lasts 100 ms; it is ballistic and thus not attuned to the velocity or direction of the visual stimulus. Closed-loop pursuit then takes over until the motion ends, and is characterized by uniform pursuit gain, i.e., the ratio between angular velocity of the eye and the target is close to 1 (the target's retinal velocity is zero).

This suggests two things: First, that a slow and gradual start of an animated motion will help mitigate any inaccuracies caused by the ballistic behavior of the first open-loop response. This presumably prevents the user from losing tracking of an object in the first few instants of the animation. Second, a gradual slow-down will help the closed-loop stage in dynamically adjusting the angular velocity of the eye to maintain zero retinal velocity of the moving object. In other words, this presumably prevents the user's eye from overshooting a target as it reaches the end of its motion path.

Note that recent evidence in vision science suggests that overshooting generally does not occur, even if the motion stops unexpectedly [23], and that the eye, after a latency of about 100 ms, is capable of decreasing its velocity to zero in a constant time of 100 ms [29]. Nevertheless, overshooting the target as it stopped was a common complaint among participants for other pacing techniques than slow-in/slow-out in our experiment. We speculate that because our point clouds involved many distractor objects that often ended up in the same vicinity at the end of an animation, the final few instants of an animation were crucial for successfully distinguishing the target, and thus that the $\sim$200 ms response may cause tracking loss for abrupt stops.

It should also be noted that given the time intervals involved in smooth pursuit, i.e., 100 ms for the ballistic open-loop response, 100 ms latency for detecting motion termination, and 100 ms for slowing down the eye velocity to zero, our one-second animations are highly taxing for the visual system — around 30% of the duration of the animation is spent in visuomotor response to the motion! Nevertheless, one-second transitions remain an informal guideline for interaction design [16], and the fact that the error rate in our experiment was so low for such comparably difficult tasks attributes to the capabilities of the human visual system.

Scientific findings on smooth pursuit may also explain why principle P2 is not more significant than it is. While it is generally difficult to initiate smooth pursuit without visual stimulus [21], research shows that it is possible to continue smooth pursuit if a target is momentarily occluded [3]. This suggests that visual complexity in the form of overlapping and crossing motion paths may not be as serious a problem as we may have initially thought, and that frame budgets are better spent on the extrema of the motion.

Finally, it is possible that abrupt changes in object velocity must be avoided not only at animation endpoints, but also during the animation itself in order to facilitate object tracking. If true, this would suggest a third design principle (P3): keep velocity variations as low as possible, from which P1 (SI/SO) would simply be a consequence. This is an intriguing possibility that needs further investigation in the future.

Limitations

Our work makes a number of assumptions that may limit its broad applicability to other areas. For example, we focus on animated transitions of point clouds where each point has the same visual appearance, whereas it can be argued that in many real-world animations, the objects have a unique visual identity which would simplify object tracking. However, this is an orthogonal aspect of temporal distortion, and we think that our results should generalize to real-world tasks as well.

Another potential limitation is that our study only measured tracking of a single object, but many realistic tasks involve several objects moving simultaneously; in fact, perceptual research suggests that most humans are capable of tracking up to four or more objects at the same time [7, 30]. Our motivation here is that object tracking of a single object is clearly a task component of tracking multiple objects, so our results should give an indication of the general case. Nevertheless, more research is needed to study this in full detail.

Furthermore, recent results in vision science show that it is possible to initiate smooth pursuit even *before* the target starts to move, especially if the person knows exactly when it will start [3]. This was of course the case for our experiment, where participants initiated the transition by pressing the space bar. More research is needed to see whether unexpected animated transitions will cause different results for animation pacing on object tracking than those we observed.

Finally, given the perceptual nature of our evaluation, it would be interesting to also study these effects using a high-precision eye tracker. For example, is there an overshooting effect when an object moving at constant speed stops, and is there a marked difference for slow-in/slow-out motion?

Generalizations

Despite the above limitations, we still believe that our task and our findings are general enough to apply to a wide spectrum of interaction and visualization scenarios. In other words, our recommendation for short-duration animated transitions to show state changes in an interactive application is to use slow-in/slow-out animation pacing — not only does this result in more realistic and aesthetically pleasing motion, it also provides the high predictability necessary for reliably tracking individual objects in the animation.

It is hard to say whether our findings also generalize to other animation durations or to non-linear motion paths. They should hold for slightly different animation durations, but with very different durations we would have observed either a ceiling effect or a floor effect given our task difficulties. As for non-linear paths, animation practice suggests using them [18, 24], but our intuition suggests that this would again decrease the predictability of the motion. Future research should address this question.

Finally, it is important to note that our work has not addressed the question whether or not to use animation in the first place, but rather which pacing methodology should be chosen if animation is adopted in an interactive application. For the former question, we refer the reader to existing literature on the topic, such as that of Tversky et al. [35].

CONCLUSION AND FUTURE WORK

We have presented results from a formal user study evaluating object tracking accuracy in animated point cloud transitions under different temporal distortion strategies. These results provide solid empirical data on the use of animation for graphical user interfaces, an area that so far has largely been dominated by design principles from general animation that may not necessarily transfer to interaction design. Our findings show that slow-in/slow-out, i.e., smoothly stretching time at the endpoints of an animation, is the most accurate temporal distortion strategy, and we speculate that this is because it maximizes the predictability of the motion.

In future work, we plan to design temporal distortions that support both design principle P1 — slowing down around the endpoints of the animation — and principle P2 — slowing down around visually complex animation frames. We are also interested in mathematically optimizing temporal distortion functions similar to van Wijk and Nuij's approach to pan-and-zoom animation [36]. Finally, we would like to explore more complex time distortion schemes, such as staging animation [16] so that all objects move at the same velocity.

Animation for interaction design is a large topic, and we plan to continue to study differences between this domain and cartoon animation. For example, slow-in/slow-out is just one of Disney's 12 basic principles of character animation [18], and it would be useful to explore the other principles in equal depth as our present work. In addition, our results also open up an array of new questions on human perception that needs further investigation, including impacts of momentary occlusion, curved motion paths, and the number of distractors.

REFERENCES

1. S. Athènes, S. Chatty, and A. Bustico. Human factors in ATC alarms and notifications design: an experimental evaluation. In *Proceedings of the USA/Europe Air Traffic Management R&D Seminar*, 2000.
2. R. Baecker and I. Small. Animation at the interface. In B. Laurel, editor, *The Art of Human-Computer Interface Design*, 251–267. Addison-Wesley, 1990.
3. G. R. Barnes. Cognitive processes involved in smooth pursuit eye movements. *Brain and Cognition*, 68(3):309–326, Dec. 2008.
4. L. Bartram, C. Ware, and T. Calvert. Moticons: detection, distraction and task. *International Journal of Human-Computer Studies*, 5(58):515–554, 2003.
5. B. B. Bederson and A. Boltman. Does animation help users build mental maps of spatial information? In *Proceedings of the IEEE Symposium on Information Visualization*, 28–35, 1999.
6. B. B. Bederson, J. Grosjean, and J. Meyer. Toolkit design for interactive structured graphics. *IEEE Transactions on Software Engineering*, 30(8):535–546, 2004.
7. P. Cavanagh and G. A. Alvarez. Tracking multiple targets with multifocal attention. *Trends in Cognitive Sciences*, 9(7):349–354, 2005.
8. B.-W. Chang and D. Ungar. Animation: From cartoons to the user interface. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 45–55, 1993.
9. F. Chevalier, P. Dragicevic, A. Bezerianos, and J.-D. Fekete. Using text animated transitions to support navigation in document histories. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, 683–692, 2010.
10. N. Elmqvist, P. Dragicevic, and J.-D. Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics*, 14:1141–1148, 2008.
11. N. Elmqvist, P. Dragicevic, and J.-D. Fekete. Color lens: Adaptive color scale optimization for visual exploration. *IEEE Transactions on Visualization and Computer Graphics*, 2011. To appear.
12. GapMinder, 2006. `http://www.gapminder.org/`.
13. C. Gonzalez. Does animation in user interfaces improve decision making? In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, 27–34, 1996.
14. Health Visualizer, 2009. `http://www.ge.com/visualization/health_visualizer/`.
15. J. Heer, Card, S. K., Landay, and J. A. prefuse: a toolkit for interactive information visualization. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, 421–430, 2005.
16. J. Heer and G. Robertson. Animated transitions in statistical data graphics. *IEEE Transactions on Visusalization and Computer Graphics*, 13(6):1240–1247, 2007.
17. S. E. Hudson and J. T. Stasko. Animation support in a user interface toolkit: Flexible, robust, and reusable abstractions. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 57–67, 1993.
18. O. Johnston and F. Thomas. *The Illusion of Life: Disney Animation*. Disney Editions, 1981.
19. C. Klein and B. B. Bederson. Benefits of animated scrolling. In *Extended Abstracts of the ACM CHI Conference on Human Factors in Computing Systems*, 1965–1968, 2005.
20. D. H. U. Kochanek and R. H. Bartels. Interpolating splines with local tension, continuity, and bias control. In *Computer Graphics (SIGGRAPH '84 Proceedings)*, volume 18, 33–41, July 1984.
21. R. J. Krauzlis. The control of voluntary eye movements: New perspectives. *Neuroscientist*, 11(2):124–137, Apr. 2005.
22. R. J. Krauzlis and S. G. Lisberger. Temporal properties of visual motion signals for the initiation of smooth pursuit eye movements in monkeys. *Journal of Neurophysiology*, 72(1):150–162, July 1994.

23. R. J. Krauzlis and F. A. Miles. Transitions between pursuit eye movements and fixation in the monkey: Dependence on context. *Journal of Neurophysiology*, 76:1622–1638, 1996.
24. J. Lasseter. Principles of traditional animation applied to 3D computer animation. In *Proceedings of the ACM Conference on Computer Graphics and Interactive Techniques*, 35–44, 1987.
25. Microsoft Pivot, 2009. `http://www.getpivot.com/`.
26. MicroStrategy, 2010. `http://www.microstrategy.com/dashboards/`.
27. T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J.-D. Fekete. Topology-aware navigation in large networks. In *Proceedings of the ACM CHI Conference on Human Factors in Computing Systems*, 2319–2328, 2009.
28. S. E. Palmer. *Vision science: Photons to phenomenology*. MIT Press, 1999.
29. J. Pola and H. J. Wyatt. Offset dynamics of human smooth pursuit eye movements: effects of target presence and subject attention. *Vision Research*, 37(18):2579–2595, Sept. 1997.
30. Z. W. Pylyshyn and R. W. Storm. Tracking multiple independent targets: Evidence for a parallel tracking mechanism. *Spatial Vision*, 3:179–197, 1988.
31. D. A. Rosenbaum. Perception and extrapolation of velocity and acceleration. *Journal of Experimental Psychology: Human Perception and Performance*, 1(4):395–403, 1975.
32. M. Shanmugasundaram, P. Irani, and C. Gutwin. Can smooth view transitions facilitate perceptual constancy in node-link diagrams? In *Proceedings of Graphics Interface*, 71–78, 2007.
33. B. H. Thomas and P. Calder. Applying cartoon animation techniques to graphical user interfaces. *ACM Transactions on Computer-Human Interaction*, 8(3):198–222, Sept. 2001.
34. B. H. Thomas and P. R. Calder. Animating direct manipulation interfaces. In *Proceedings of the ACM Symposium on User Interface Software and Technology*, 3–12, 1995.
35. B. Tversky, J. B. Morrison, and M. Bétrancourt. Animation: can it facilitate? *International Journal of Human-Computer Studies*, 57(4):247–262, 2002.
36. J. J. van Wijk and W. A. A. Nuij. A model for smooth viewing and navigation of large 2D information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 10:447–458, 2004.
37. C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004.
38. K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst. Animated exploration of dynamic graphs with radial layout. In *Proceedings of the IEEE Symposium on Information Visualization*, 43–50, 2001.

APPENDIX

To generate a random point cloud from the parameters listed in the *Datasets* section (in bold here), we first initialize the random seed to $\boldsymbol{r}$ and choose $\boldsymbol{n}$ points $p_i \in \mathcal{S}$, with $\mathcal{S}$ being the unit square. We then move them using a force-directed algorithm and $\boldsymbol{i}$ iterations. The force exerted on each p_i is:

$$\vec{F}(p_i) = [\frac{1}{\boldsymbol{n}} \sum_{j \neq i} \vec{F}_C(p_i, p_j)] - \vec{F}_{SQ}(p_i) + [\sum_{j \neq i} \vec{F}_R(p_i, p_j)] + \vec{F}_{ER}(p_i) + \vec{F}_F(\vec{v}_i)$$

• $\vec{F}_C$ is the *clustering force*, defined as:

$$\vec{F}_C(p_i, p_j) = \begin{cases} f_{C_0} ||p_i - p_j||^{\boldsymbol{p}} \cdot \widehat{p_i p_j} & \text{if } ||p_i - p_j|| \geq d_{Cmin} \\ f_{C_0 min} \cdot d_{Cmin}{}^{\boldsymbol{p}} \cdot \widehat{p_i p_j} & \text{otherwise} \end{cases}$$

with:

$f_{C_0} = \frac{\boldsymbol{c}}{1000} \cdot \frac{k^2}{k - \boldsymbol{p}}$ being the force magnitude. We use $k = 7$ because it yields similar degrees of visual clustering for different values of $\boldsymbol{p}$.

$\widehat{u}$ is the unit vector $\frac{\vec{u}}{||u||}$

d_{Cmin} is the minimum distance above which the force applies. We use $d_{Cmin} = 0.1$ if $\boldsymbol{c} > 0$ and $= 0.05$ otherwise.

$f_{C_0 min}$ is the force applied below the minimum distance. We use $f_{C_0 min} = f_{C_0}$ if $\boldsymbol{c} > 0$ and $= 0$ otherwise.

• $\vec{F}_{SQ}$ is the *clustering force in an homogeneous square*, a term that prevents the whole point cloud from collapsing:

$$\vec{F}_{SQ}(p) = \iint_{p' \in \mathcal{S}} f_{C_0} \cdot ||p - p'||^{\boldsymbol{p}} \cdot \widehat{p_i p_j}$$

• $\vec{F}_R$ is the *point repulsion force* that locally reduces cluster density and point overlap:

$$\vec{F}_R(p_i, p_j) = \begin{cases} -f_{R_0} \cdot (\frac{\boldsymbol{s}}{||p_i - p_j||} - 1) \cdot \widehat{p_i p_j} & \text{if } ||p_i - p_j|| < \boldsymbol{s} \\ 0 & \text{otherwise} \end{cases}$$

f_{R_0} being the amount of repulsion. We use $f_{R_0} = 0.001$.

• $\vec{F}_{ER}$ is the *edge repulsion force* that prevents points from going past the unit square:

$$\vec{F}_{ER}(p_i) = \begin{cases} f_{ER_0} \cdot \overrightarrow{p_i \Omega} & \text{if } p_i \notin \mathcal{S} \\ 0 & \text{otherwise} \end{cases}$$

Ω is the point on $\mathcal{S}$'s edge closest to p_i. We use $f_{ER_0} = 2$.

• $\vec{F}_F$ is the *fluid friction force* that prevents points from moving too fast: $\vec{F}_F(v_i) = -f_{F_0} \cdot \vec{v}_i$, with $\vec{v}_i$ being the speed of p_i. We use $f_{F_0} = 0.1$.

To generate a transition, we first generate two point clouds $P^0 = (\boldsymbol{r}, \boldsymbol{n}, \boldsymbol{c}, \boldsymbol{p}, \boldsymbol{s}, \boldsymbol{i})$ and $P^1 = (\boldsymbol{r} + 1, \boldsymbol{n}, \boldsymbol{c}, \boldsymbol{p}, \boldsymbol{s}, \boldsymbol{i})$ with the previous method. At this point, every point of index i moves from the location $p_i^0 \in P^0$ to the location $p_i^1 \in P^1$.

We then increase motion coherence by taking two random indices i and j and swapping p_i^1 and p_j^1 if this yields a lower value of $\Delta_i + \Delta_j$. This operation is repeated $\boldsymbol{n}^2 \boldsymbol{m}^2$ times.

Δ_i is the *relative motion* of the point of index i, defined as:

$$\Delta_i = \frac{\sum_{j | d_{i,j} \neq 0} \frac{(p_j^1 - p_j^0)}{d_{i,j}}}{\sum_{j | d_{i,j} \neq 0} \frac{1}{d_{i,j}}} \text{ with } d_{i,j} = (p_i^0 p_j^0)^2 + (p_i^1 p_j^1)^2$$