LEARNING RIEMANNIAN METRICS FOR INTERPOLATING ANIMATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Supplementary materials to the paper: "Learning Riemannian Metrics for Interpolating Animations". The code will be made available upon publication.

1 VISUAL COMPARISON OF INTERPOLATIONS

We visualize and qualitatively evaluate the accuracy of our interpolations for different sampling rates s to complement the results shown in Figs. 6-9 in the main text. Our visualizations show the ground truth animation, with the interpolation layered transparently over it. This allows us to highlight where the interpolation deviates from the ground truth (GT) in Figs. 1-5. We observe that our interpolation is accurate even for smaller sampling rates. These static visualizations are further complemented by the supplemental video.

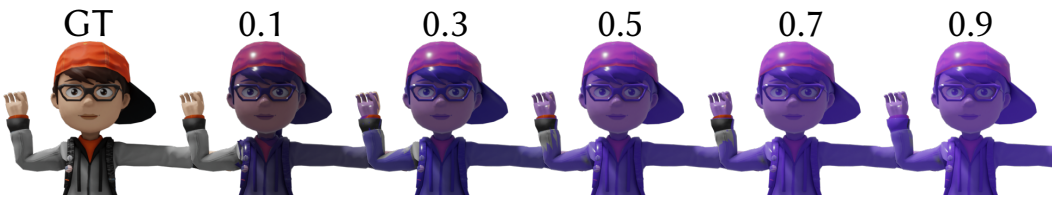

Figure 1: Qualitative evaluation of the geodesic interpolation for the `Pitching` animation. Left: Frame from ground truth (GT) animation. Right: Frames for rates: $s \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

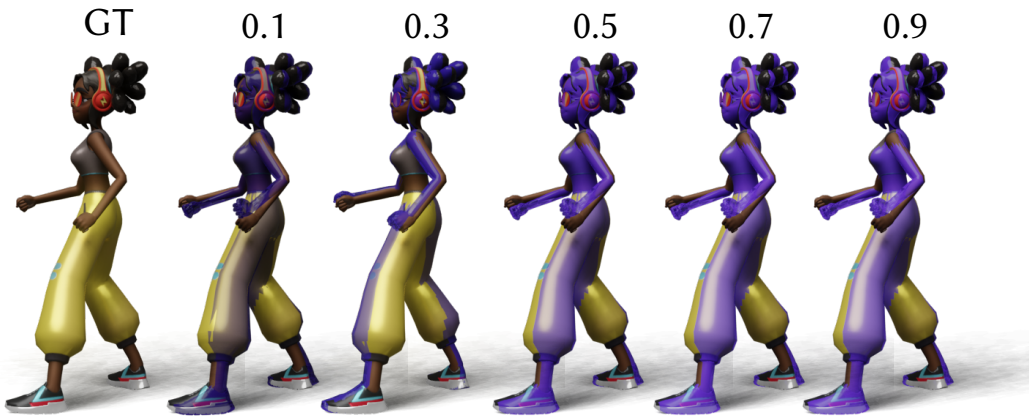

Figure 2: Qualitative evaluation of the geodesic interpolation for the `Punching` animation. Left: Frame from ground truth (GT) animation. Right: Frames for rates: $s \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

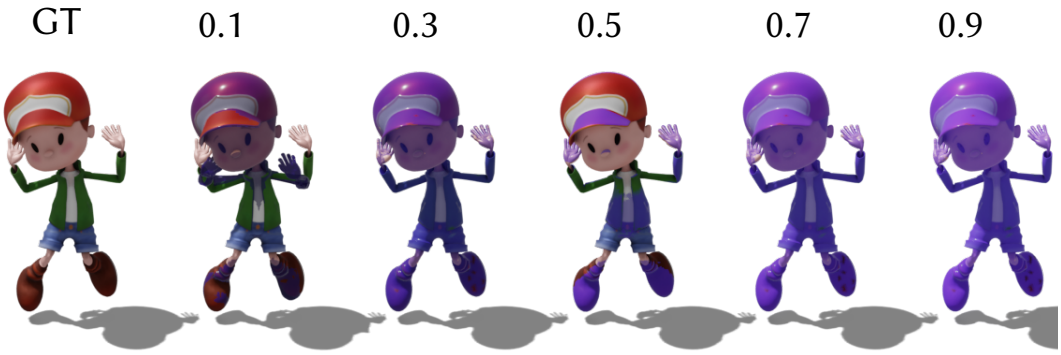

Figure 3: Qualitative evaluation of the geodesic interpolation for the `Jumping` animation. Left: Frame from ground truth (GT) animation. Right: Frames for rates: $s \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

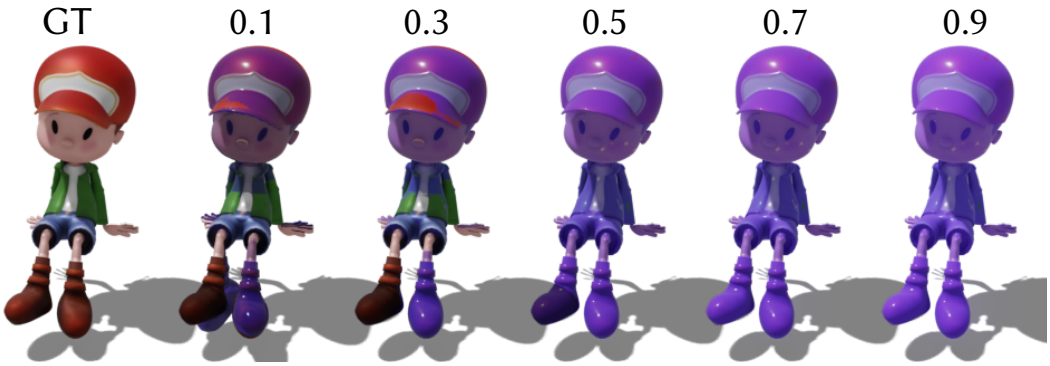

Figure 4: Qualitative evaluation of the geodesic interpolation for the `Sitting` animation. Left: Frame from ground truth (GT) animation. Right: Frames for rates: $s \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

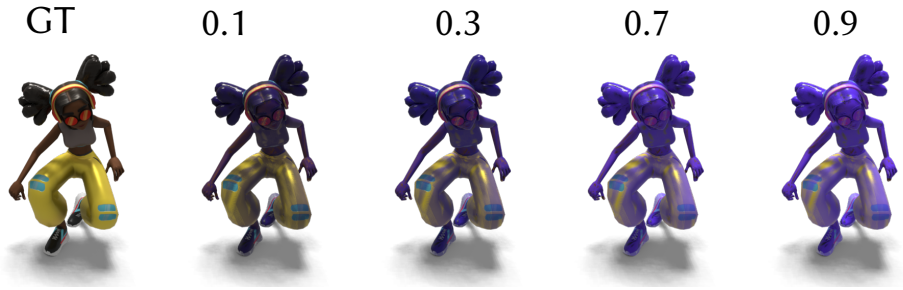

Figure 5: Qualitative evaluation of the geodesic interpolation for the `Rolling` animation. Left: Frame from the ground truth (GT) animation. Right: Frames for rates: $s \in \{0.1, 0.3, 0.5, 0.7, 0.9\}$.

2 Quantitative Comparison of Interpolations

We quantitatively compare the accuracy of each interpolation scheme as a function of the sampling rate s. Fig. 6 plots the interpolations' errors:

$$Q_{hyb} = 0.5Q_{loc} + 0.5Q_{rot},$$

for the `Punching` animation. This complements the results for the `Pitching` animation presented in Fig. 5 in the main text. As for the `Pitching` animation, our approach applied to the `Punching` animation consistently presents the lowest error just in front of slerp's. Despite the

seemingly small quantitative difference between slerp's and ours in Fig. 6, we note from the supplementary videos and qualitative evaluations above that the interpolated animations show significant perceptual differences.

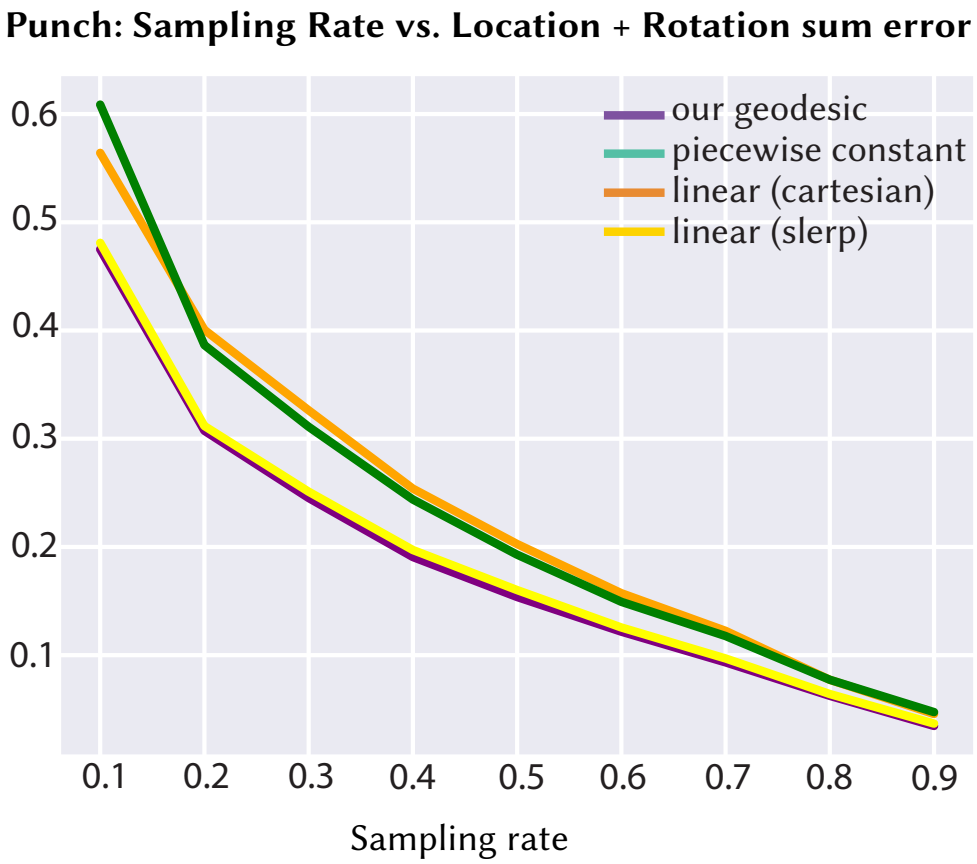

Figure 6: As the sampling rate for the `Punching` animation increases—which means a higher number of ground truth frames in the initial animation—the error metric Q_{hyb} decreases.

To confirm these visual insights and better express these perceptual differences, we use an additional quantitative evaluation: the Hausdorff distance. The Hausdorff distance is a common measure that compares two sets of points: e.g., two meshes at a given frame in between two animations. Given two sets of points, A and B, the Hausdorff distance $d_H(A, B)$ is defined as:

$$d_H(A, B) = \max\{\sup_{a \in A} \inf_{b \in B} ||a - b||, \sup_{b \in B} \inf_{a \in A} ||a - b||\},$$

where $||a - b||$ is the distance between points a and b in Euclidean space, and sup is the supremum.

The lower the Hausdorff distance, the more similar the two meshes on a given frame are. The Hausdorff distance measures the maximum distance from any point in one character mesh to the closest point in the other character mesh. In other words, the Hausdorff distance measures the similarity between two character meshes by quantifying the distance between their farthest points: this is interesting to quantify our own perception of the differences between animations, as our eyes often catch on the most important differences when running the animation.

We compute the Hausdorff distance between each interpolated animation (across interpolation schemes) and the ground truth animation for a fixed sampling rate of $s = 0.3$. Figs.7-9 show our results for the remaining three animations: `Sitting`, `Jumping` and `Rolling` respectively. Each interpolation scheme presents a plot with "peaks", where the distance goes to 0 at regular intervals. This is expected, since each interpolation scheme keeps frames from the ground-truth animations, at regular intervals, and only interpolates in-between them. We observe that our geodesic

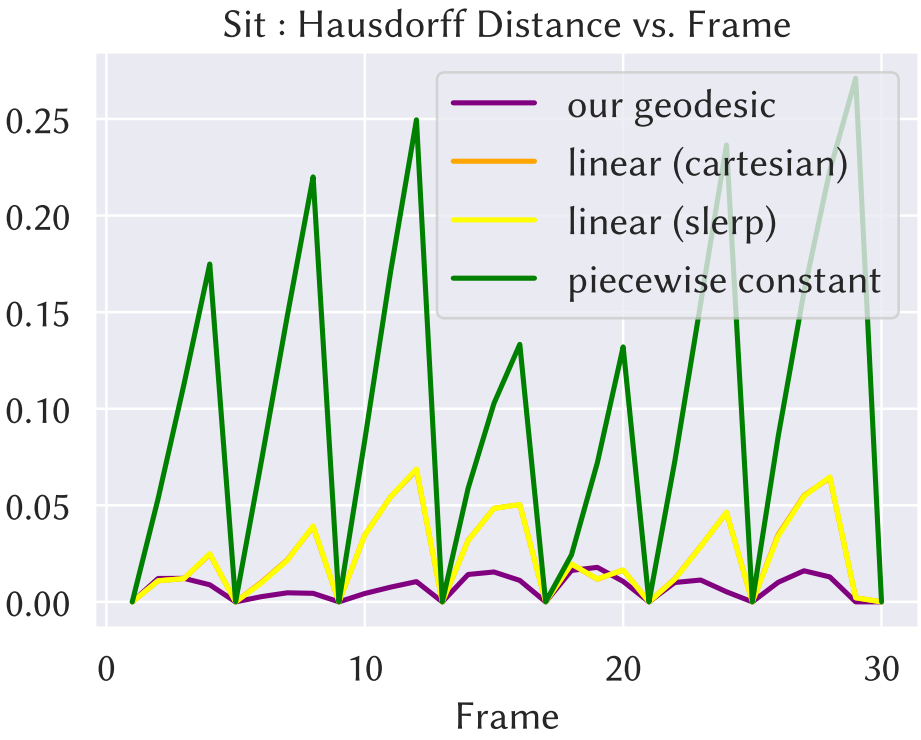

Figure 7: Quantitative comparison per frame using the Hausdorff distance to the ground truth for the `Sitting` animation (sampling rate: $s = 0.3$).

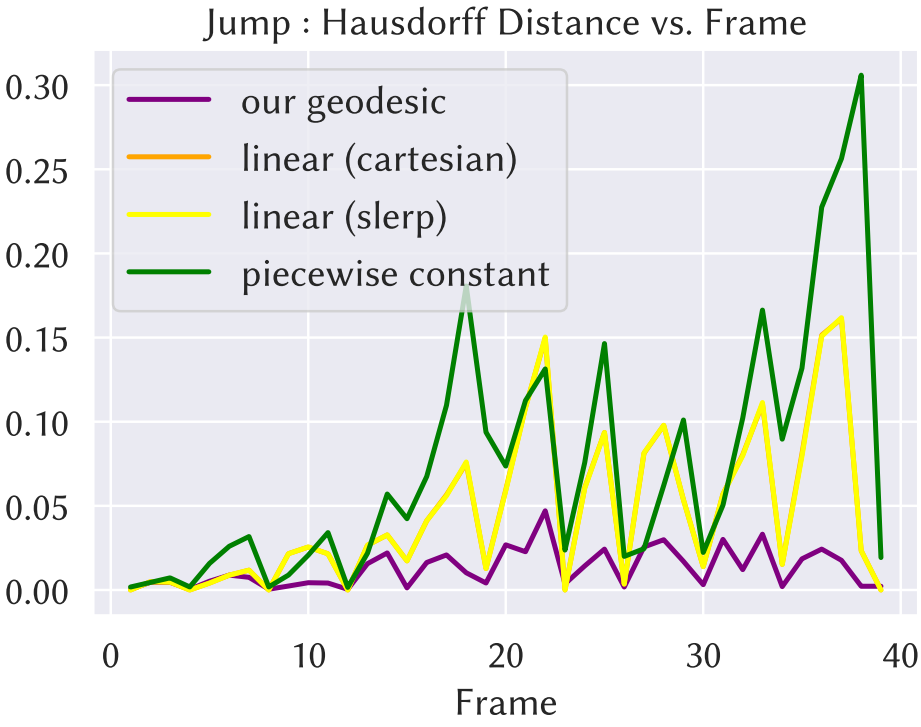

Figure 8: Quantitative comparison per frame using the Hausdorff distance to the ground truth for the `Jumping` animation (sampling rate: $s = 0.3$).

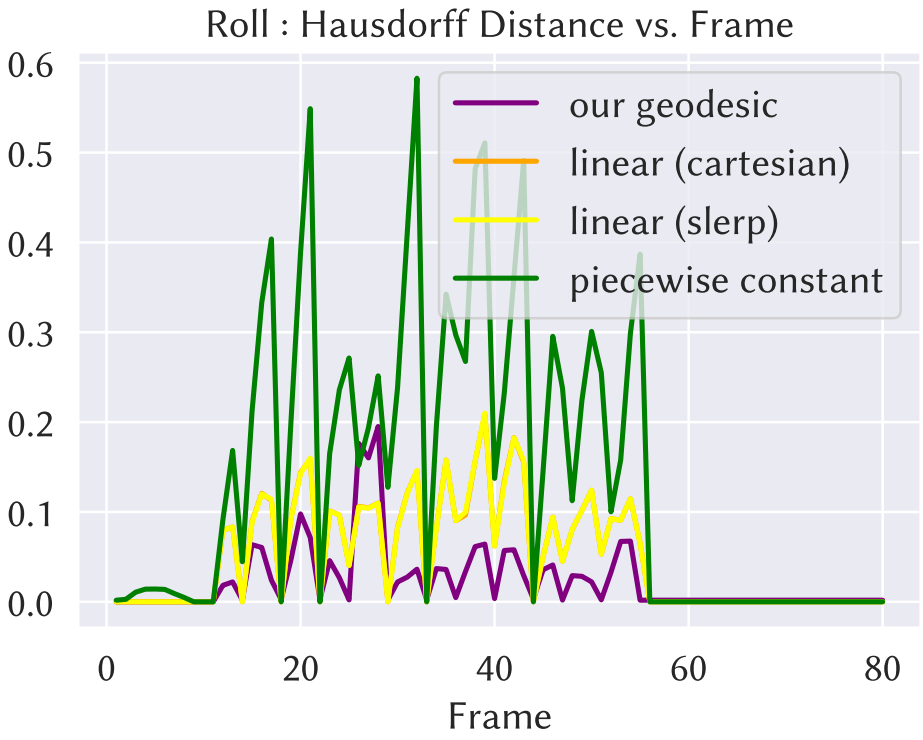

Figure 9: Quantitative comparison per frame using the Hausdorff distance to the ground truth for the `Rolling` animation (sampling rate: $s = 0.3$).

interpolation (shown in purple) systematically provides a lower distance to the ground-truth animation, quantifying what we observe qualitatively in the video. We also observe that the two schemes for linear interpolation, Cartesian and slerp, provide very close (although distinct) values of Hausdorff distances. This highlights an interesting property of our approach: while our interpolation is still "linear" in the sense that it is "geodesic" which is the generalization of "linear" for manifolds, changing the geometry of the manifold does provide a significant perceptual difference compared to only changing from linear space (Cartesian) to nonlinear space (spherical).

Furthermore, we confirm that the frames where our interpolation performs best are indeed the frames where the perceptual difference is the most visible in the supplemental video. For example, Fig. 6 of the main text shows frame 24 of the `Sitting` animation. In Fig. 7, this frame corresponds to one of the high peaks of the Hausdorff distance for the linear interpolations (Cartesian and slerp) and the piece-wise constant interpolation, while our distance remains low. Likewise, Fig. 9 of the main text shows frame 44 of the `Rolling` animation. In Fig. 9, this frame also corresponds to one of the frames where our geodesic interpolation performs significantly better that the alternatives.

Lastly, we evaluate how our approach performs under different sampling rates s for animations `Sitting`, `Jumping` and `Rolling`. Figs. 10-12 show our results. As expected, the distance to the ground truth animation decreases as we increase the sample rate, specifically as a decreasing exponential. We explicitly illustrate the relationship between the error of our method, expressed as Hausdorff distance, and the sampling rate s. Fig. 13 shows the Hausdorff distance averaged over frames in log-scale as a function of s. We confirm a consistent trend among the three animations: decreasing line in log-scale, i.e., decreasing exponential.

3 Compression

We quantitatively compare the file sizes for a given animation under the different compression schemes. We take an error threshold using the Q_{hyb} metric to set a threshold on the desired quality of the animation. For each interpolation scheme, we pick the lowest sampling rate that allows

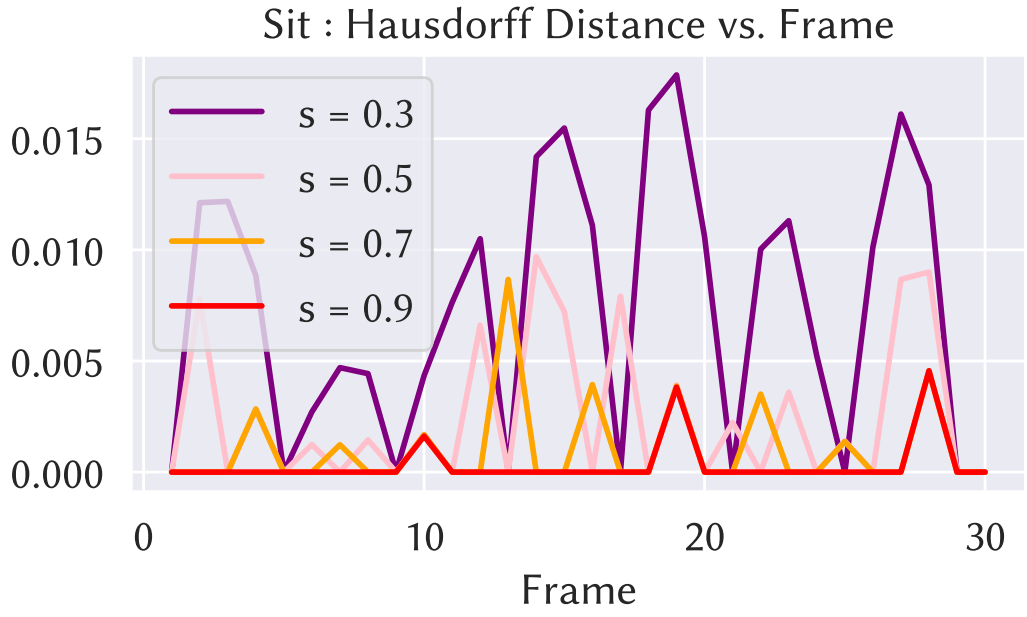

Figure 10: Evaluation of the geodesic interpolation per frame using the Hausdorff distance to the ground truth animation, for the `Sitting` animation.

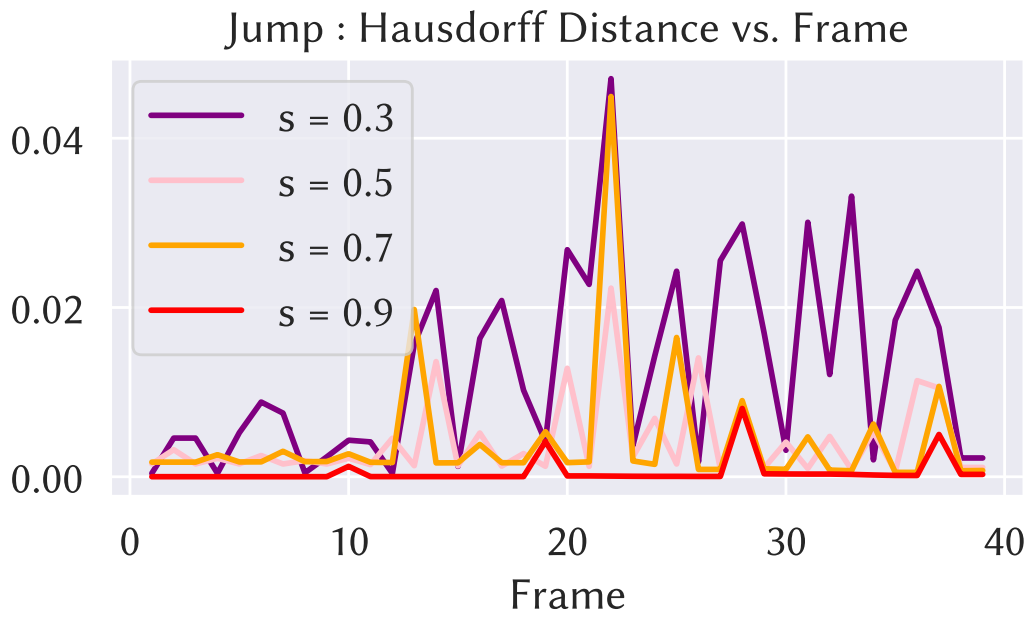

Figure 11: Evaluation of the geodesic interpolation per frame using the Hausdorff distance to the ground truth animation, for the `Jumping` animation.

the interpolation to reach that quality threshold. We then compare the file sizes generated by each subsampled animation. To this aim, we consider the number of frames, bones, and the amount of memory required to represent each bone at each frame. Specifically, since one floating-point number is typically 4 bytes in size and each bone can be represented using 3 coordinates (i.e., 3 floats) at

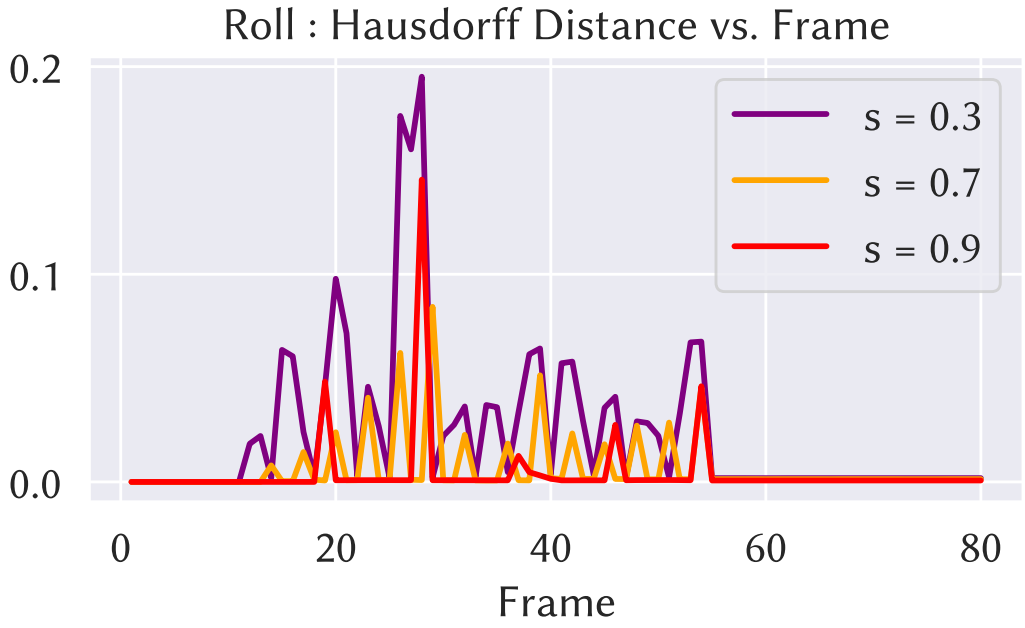

Figure 12: Evaluation of the geodesic interpolation per frame using the Hausdorff distance to the ground truth animation, for the `Rolling` animation.

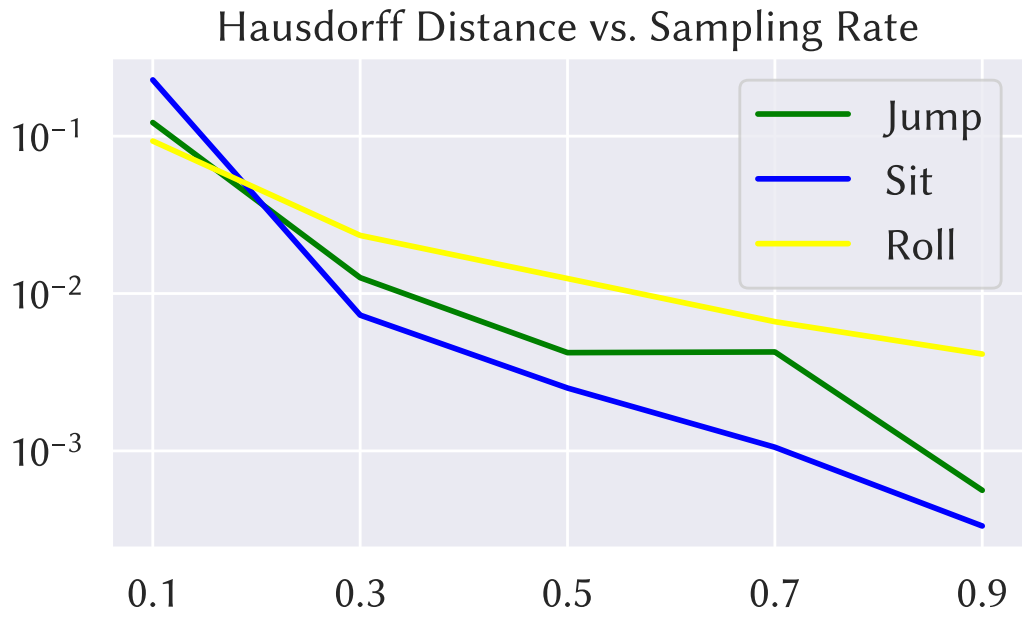

Figure 13: Hausdorff distance (in log-scale) between our interpolated animation and the ground truth animation, averaged over the frames, as a function of the sampling rate, for the three main animations: `Jumping`, `Sitting`, and `Rolling`.

each frame, we estimate the file size using the following formula:

$$S = F \cdot B \cdot 3 \cdot 4 \text{ bytes}, \tag{1}$$

where F is the number of frames and B is the number of bones. To evaluate the file size generated by our method, the two additional parameters α and β must also be saved for each bone. They occupy an additional:

$$S_{\alpha,\beta} = B \cdot 2 \cdot 4 \text{ bytes}. \tag{2}$$

		Sampling Rate	**File Size**
Pitch	our geodesic	0.42	**30.72KB***
	piecewise constant	0.57	40.32KB
	linear (cartesian)	0.57	40.32KB
	linear (slerp)	0.46	31.68KB
Punch	our geodesic	0.38	**184 KB***
	piecewise constant	0.49	234 KB
	linear (cartesian)	0.51	249 KB
	linear (slerp)	0.4	195 KB

Table 1: We show the compression capability of our approach, by giving the file sizes it achieves on two animations with different number of bones: the `Pitch` animation with 24 bones and the `Punch` animation with 65 bones. In both animations, and for each interpolation method, we select the sampling rate that allows us to reach an interpolation error of $Q_{hyb} < 0.2$.

We note that the formula for S only provides a proxy of the file size, as it assumes that each bone is represented using only 3 floats at each frame. Indeed, it does not take into account other memory requirements that are necessary to store the artistic components of an actual character animation in a software such as Blender. Yet, our estimation of file size provides a useful guideline to get intuition on the memory requirements of a given animation, which is crucial for efficient storage.

Table 1 shows compression achieved by the interpolation methods in terms of file size S as defined in the equation above. To stay below a given interpolation error of $Q_{hyb} < 0.2$, we observe that our method requires a lower sampling rate s, which in turn yields a lower file size. The difference in file sizes naturally increases when the character rig is large: that is, for an animation that has a higher number of bones. For example, in the animation `Punch` with 65 bones, even a "small" improvement in sampling rate provides an important improvement in terms of file sizes. Considering the fact that animations do not only store the location of the bones, but also store a very high number of artistic features, the file sizes presented in Table 1 should be multiplied by a few order of magnitudes. In this context, our method can yield substantial improvements in terms of memory requirements.