Implementation of a momentum-based distance metric for motion graphs

Student: Alessandro Di Domenico (st.no 3775682),
Supervisor: Nicolas Pronost

April 3, 2014

Abstract

This report presents the procedures and results of the performed experiments relative to the construction of a motion graph for character animation, using a momentum-based extension of a kinematics-based distance metric. The graph is automatically generated from an input set of motion clips and used to animate a human-like character in a virtual environment. The aim of the experiments is to generate motion graphs with the new metric and evaluate their performances. The results show that the new hybrid distance metric offers improvements with respect to the kinematics-based one, in terms of average distance between the poses in the transitions, while the kinematics-based distance metric performs better in terms of graph density.

Contents

1 Introduction **2**

2 Related work **3**

3 Momentum-based metric **5**

3.1 Method 5

3.2 Experiment 8

3.2.1 Point cloud distance metric 11

3.2.2 Momentum-based distance metric 12

3.2.3 Hybrid distance metric 13

4 Conclusion **17**

1 Introduction

Today's technology allows the creation of extremely realistic and interactive virtual worlds. To preserve the illusion and enhance the sense of immersion in these environments, the need comes for the characters that inhabit them to be animated in the most authentic way possible. Animating human-like models represents a huge challenge, because taking in consideration all the subtleties of human movements, necessary to obtain satisfying and convincing results, is often hard and time-consuming.

Many approaches have been proposed to deal with this problem; one of the most widely used animation techniques is motion capture, which allows the recording of a person performing specific actions. Motion capture is a reliable way to reproduce human movements but comes with one big downside: editing the motion capture data, aside from small changes, has proven to be very difficult. If the recorded data do not meet the quality requirements expected by the user, the only solution is another motion capture session; this greatly increases costs and time consumption.

Kovar et al. [2002] presented a method that allows the creation of new streams of motion based on a set of pre-recorded movements while preserving the overall quality of the original data. The result is a structure called a motion graph, which allows the rearrangement of the original motion clips in different ways, thus allowing the creation of new sequences.

In order to construct these new movements, the set of original data must be examined for transition points, i.e. points in the clips in which the character's motions are sufficiently similar between each other. One efficient way to determine such similarity is the comparison of the character's pose: if the poses are close enough, blending techniques can be used to create a transition between the data segments.

In this report we propose a different method to find transition point between clips: instead of just using a kinematics-based metric to evaluate the pose similarities between the characters, we implemented a physics-based extension that will evaluate the angular and linear momentum of the characters and establish the presence of a similarity according to the distance between these values. The purpose is to compare the graphs obtained using the these different metrics and analyze the results in terms of connectivity and distance between the poses in the transition phase.

The remainder of the report is organized as follows. In section 2 we describe related work. In section 3 we describe the method in detail and present the experiment results. In Section 4 we draw conclusions and suggest areas of improvement and future work.

2 Related work

Motion graph construction requires the use of a distance metric to individuate good candidates for the transitions and many different metrics have been proposed. In Kovar et al. [2002], detection of candidate transition points is performed using a point cloud distance metric. In Gleicher et al. [2003], the authors use point clouds to compute distances between frames of different clips. Point clouds are formed attaching markers to the skeleton of the character in the two candidate frames and in a small window of frames adjacent to them; subsequently, the weighted sum of squared distance between the points forming the point clouds is determined and stored in a database. Transition points are selected by searching for local minima in a function that associates every frame with its distance towards other frames of a motion clip. These minima are evaluated against a user-defined threshold and if they are below such threshold, a transition can be established between the examined frames.

In Arikan and Forsyth [2002] a metric is defined by calculating the difference between joint positions and velocities, as well as the difference between the torso velocities and acceleration. These values are then used to calculate a weight that is assigned to the edges of the motion graph; the weight represents the degree to which two frames can be concatenated.

A similar metric is used in Lee et al. [2002], where a distance function is computed as the sum of two terms, describing the weighted difference of joint angles and the weighted difference of joint velocities. Subsequently, the function is used to calculate the expectation of transition from one frame to another, by means of a probability function which takes the previously calculated distance function as input.

Forbes and Fiume [2005] propose a different way to calculate the distance between frames. The animation dataset is transformed with the Principal Component Analysis technique; the resulting principal components vectors represent the axes in space. Points are represented by a combination of these principal components thus allowing the description of different poses. The distance is then computed as a scaled L2 norm of these PC coordinates. The same concept is also used in Egges et al. [2004].

Li et al. [2008] take a different approach on the evaluation of transition points in motion capture data. Using the conjecture that in the comparison of two similar trajectories, the most natural looking is the one that implies less effort, they develop a new metric to measure said effort. They are then able to compute an L-score for the different motions that allows them to select the optimal transition points.

The idea of using linear and angular momentum for motion synthesis is not new; In Abe et al. [2004] the authors introduce a system for the editing of highly dynamic character motion that allows animators to create smooth animations maintaining at the same time a good amount of interactive control over the character. The preservation of the behaviour of the input motion is guaranteed by a technique that matches the linear and angular momentum patterns of the data; by solving a constrained optimization problem for every spline that is used to define the momentum characteristics of a motion, a set of control points and knots are computed. The points and knots are used to make sure that the edited motions follow the momentum path of the original data, thus resulting in physically realistic movements.

3 Momentum-based metric

3.1 Method

The purpose of this experiment was the evaluation of the performance of a motion graph constructed with a metric that was not only based on kinematic parameters and pose similarity. We initially implemented one based on the difference between a character's linear and angular momentum in different poses and we studied the quality of the resulting motion graph. The decision to also research on a metric that was a combination of both the point cloud and the momentum-based one came from the observation that while two frames may have similar values of momentum, it is not necessary true that they correspond to similar movements of a character.

In fact, angular and linear momentum are physical properties of moving bodies but, by themselves, they do not offer a visual description of a performed movement; linear momentum is a vector quantity defined as the product of the body's mass with its velocity. It quantifies the force that is necessary to stop a moving body in the unit of time. Its analogous quantity for spatial rotations is the angular momentum; the angular momentum of a body is defined as the product of a body's moment of inertia with its angular velocity; it measures the tendency of a rotating body to persist in its rotary motion.

Thus, it appears evident the need to incorporate some sort of pose recognition phase while building the transitions between clips. The implementation was followed by an experimentation phase in which we compared the outcome of the process using three different configurations: the first used the original point cloud distance metric; the second configuration was purely based on the momentum difference in the animation frames; finally, the third configuration was the combination of the first two, with the use of different user-defined thresholds.

The initial stage of the implementation process was focused on gathering and examining relevant studies on animation techniques and documentation about calculation of linear and angular momentum of human bodies. This was followed by a structural analysis of the given framework aimed at understanding how to modify the already existing system in order to substitute the former distance metric with the new one. In fact, the framework was already provided with a module for the creation of a motion graph generated from a set of motion clips, using the point cloud distance metric.

Prior to the actual implementation of the momentum-based metric, we spent some time analyzing the point cloud distance metric to better understand its operating principles. Soon it appeared evident that the most straightforward way to develop the new distance metric would be to set up a structure to attach to the character's skeleton in order to keep track of the physical properties that were being considered; thus a physicsSkeleton structure was created. This structure mainly acted as container of physicsBones; each bone was approximated as a cylinder object with several defining attributes attached to it. Namely, every bone had a name, a mass, a width and a length that were used to calculate their moment of inertia and center of mass.

Moment of inertia is a physical property of a body that defines its resistance to a variation of angular velocity about a certain rotation axis. The physicsSkeleton contained as many bones as the number of body parts of which the character's skeleton is composed, a total of twenty-seven; the inertia tensor of each bone is given by:

$$I = \begin{bmatrix} \frac{1}{12}m(3r^2 + h^2) + mr_x^2 & 0 & 0 \\ 0 & \frac{1}{12}m(3r^2 + h^2) + mr_y^2 & 0 \\ 0 & 0 & \frac{1}{2}mr^2 + mr_z^2 \end{bmatrix}$$

The mass and girth data for the limbs were extrapolated by experimental data found in Plagenhoef et al. [1983]. The whole character had a mass of 75 kg, distributed along the parts that composed it; every part had a weight associated to it that was used to determine its mass.

We used the pre-existent code that already took care of aligning the character's position in order to create a coherent animation, with a major distinction. In the original code, the transitions were generated only if the computed point cloud distance was below a certain user-defined threshold; this mechanism acted as a filter and allowed the creation of a relatively small database. The new version, instead, created every possible transitions between all the input frames and the resulting database was much bigger than before; this served for the construction of a unique database that could be used with the three different metrics, instead of building a database for each of them. The filtering was thus moved to a later phase.

In detail, the whole experimental process can be divided in the following four points:

1. The motion files were loaded into the framework, which was also used to set the virtual environment in which the character moved and its corresponding physicsSkeleton object.

2. Subsequently, the files were examined frame by frame: for each of them, linear and angular momentum of the character were computed. Each frame of the first motion was compared to each frame of the second one and the measured difference was temporary stored.

3. After this stage, the points to build the point clouds were gathered, for every frame of both the files, and the point cloud distance was stored.

4. Finally, the translation and rotation data for the model's alignment were calculated and the point cloud distance, the linear momentum distance and the angular momentum distance were stored in the database.

Formulas used for the calculation of the momentum are:

$$\mathbf{p} = m\mathbf{v}$$

and

$$\mathbf{L} = I\omega$$

The first one refers to the linear momentum, while the second one refers to the angular momentum. Since the linear and angular momentum are strictly dependent on the linear and angular velocity,respectively, they were calculated frame by frame as :

$$\mathbf{v}(t_2) = \frac{\mathbf{x}(t_2) - \mathbf{x}(t_1)}{\Delta t}$$

and

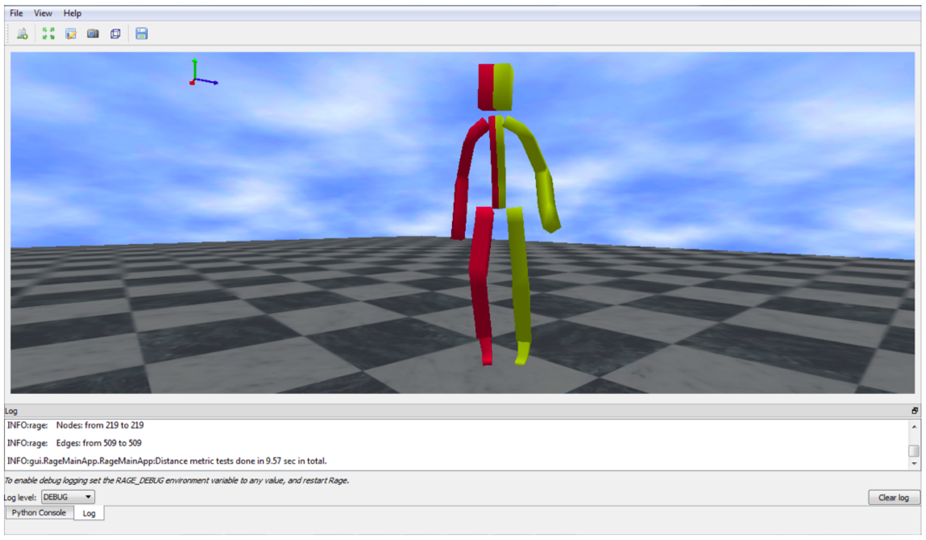
$$\omega(t_2) = \frac{\theta(t_2) - \theta(t_1)}{\Delta t}$$

The final database contained, as precedently stated, all the possible transitions between the frames of the two motion files; once it was completed, we built the motion graph from it. In order to do so, we used different combinations of three different thresholds; these thresholds, applied on the database, filtered out the transitions and the resulting subset was used to build the graph.

3.2 Experiment

The framework runs on a pc that mounts a 2.40 GHz dual core processor, 8 GB of ram, in a Windows environment with the Visual Studio IDE.

The virtual environment used for the simulation is composed of a square plane on which a stylized human model is able to walk. Figure 1 shows us the environment and the model used in the experiment.



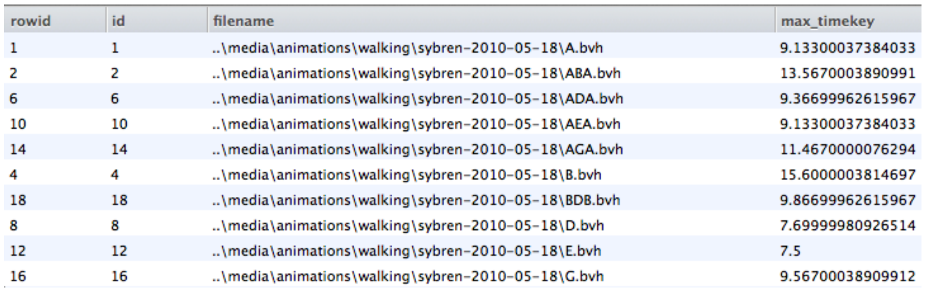
```
INFO:rage:   Nodes: from 219 to 219
INFO:rage:   Edges: from 509 to 509
INFO:gui.RageMainApp.RageMainApp:Distance metric tests done in 9.57 sec in total.
```

To enable debug logging set the RAGE_DEBUG environment variable to any value, and restart Rage.

Figure 1: A close-up of the model used in the experiment

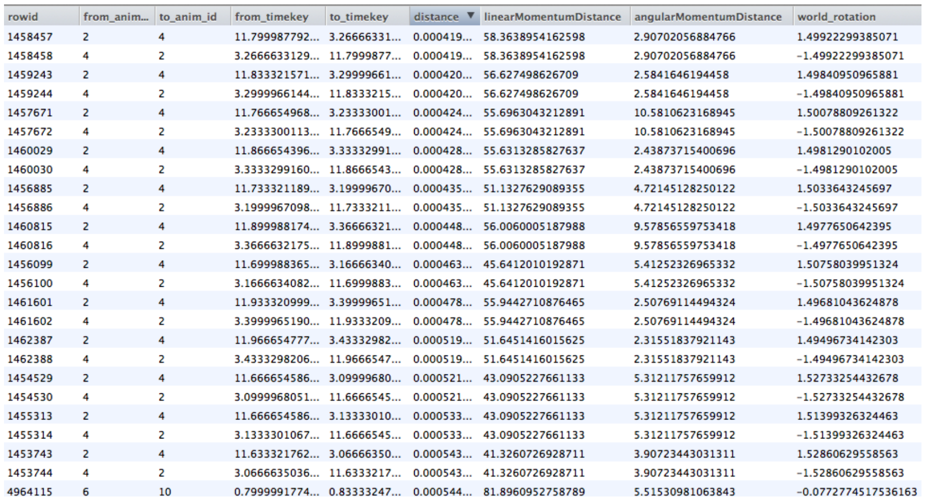
In order to obtain a motion graph of adequate dimensions, a total of 10 animations were used; the animations show a character walking at different gait speed.

The output database is composed by a total of 910 MB worth of data. Its structure is shown in Figure 2 and Figure 3. The former shows the animations table with the *max_timekey* column being the duration time of each video; the latter shows the transitions table in which we can see the columns *distance*, which is referring to the point cloud distance, and the columns that indicate to the linear momentum distance and the angular momentum distance. The columns *from_timekey* and *to_timekey* point out the keyframes in the motions in which the transitions happen; from this image it can be noticed that each transition is repeated in two directions, because it can happen in both ways. This design choice was already part of the framework and we decided not to modify it. As previously specified, the cause for the big size of the database is that it contains all the possible transitions between the different frames composing the motion clips. We chose the approach of filtering valid transitions in a second stage for reasons of computing speed. In fact, the creation of different databases for the three different metrics (point cloud, momentum-based and hybrid) would have taken more time than the about 150 hours of calculation required for one big comprehensive database.



rowid	id	filename	max_timekey
1	1	..\media\animations\walking\sybren-2010-05-18\A.bvh	9.13300037384033
2	2	..\media\animations\walking\sybren-2010-05-18\ABA.bvh	13.5670003890991
6	6	..\media\animations\walking\sybren-2010-05-18\ADA.bvh	9.36699962615967
10	10	..\media\animations\walking\sybren-2010-05-18\AEA.bvh	9.13300037384033
14	14	..\media\animations\walking\sybren-2010-05-18\AGA.bvh	11.4670000076294
4	4	..\media\animations\walking\sybren-2010-05-18\B.bvh	15.6000003814697
18	18	..\media\animations\walking\sybren-2010-05-18\BDB.bvh	9.86699962615967
8	8	..\media\animations\walking\sybren-2010-05-18\D.bvh	7.69999980926514
12	12	..\media\animations\walking\sybren-2010-05-18\E.bvh	7.5
16	16	..\media\animations\walking\sybren-2010-05-18\G.bvh	9.56700038909912

Figure 2: The transition table of the output database



rowid	from_anim...	to_anim_id	from_timekey	to_timekey	distance ▼	linearMomentumDistance	angularMomentumDistance	world_rotation
1458457	2	4	11.799987792...	3.26666331...	0.000419...	58.3638954162598	2.90702056884766	1.49922299385071
1458458	4	2	3.2666633129...	11.7999877...	0.000419...	58.3638954162598	2.90702056884766	-1.49922299385071
1459243	2	4	11.833321571...	3.29999661...	0.000420...	56.627498626709	2.5841646194458	1.49840950965881
1459244	4	2	3.2999966144...	11.8333215...	0.000420...	56.627498626709	2.5841646194458	-1.49840950965881
1457671	2	4	11.766654968...	3.23333001...	0.000424...	55.6963043212891	10.5810623168945	1.50078809261322
1457672	4	2	3.2333300113...	11.7666549...	0.000424...	55.6963043212891	10.5810623168945	-1.50078809261322
1460029	2	4	11.866654396...	3.33332991...	0.000428...	55.6313285827637	2.43873715400696	1.4981290102005
1460030	4	2	3.3333299160...	11.8666543...	0.000428...	55.6313285827637	2.43873715400696	-1.4981290102005
1456885	2	4	11.733321189...	3.19999670...	0.000435...	51.1327629089355	4.72145128250122	1.5033643245697
1456886	4	2	3.1999967098...	11.7333211...	0.000435...	51.1327629089355	4.72145128250122	-1.5033643245697
1460815	2	4	11.899988174...	3.36666321...	0.000448...	56.0060005187988	9.57856559753418	1.4977650642395
1460816	4	2	3.3666632175...	11.8999881...	0.000448...	56.0060005187988	9.57856559753418	-1.4977650642395
1456099	2	4	11.699988365...	3.16666340...	0.000463...	45.6412010192871	5.41252326965332	1.50758039951324
1456100	4	2	3.1666634082...	11.6999883...	0.000463...	45.6412010192871	5.41252326965332	-1.50758039951324
1461601	2	4	11.933320999...	3.39999651...	0.000478...	55.9442710876465	2.50769114494324	1.49681043624878
1461602	4	2	3.3999965190...	11.9333209...	0.000478...	55.9442710876465	2.50769114494324	-1.49681043624878
1462387	2	4	11.966654777...	3.43332982...	0.000519...	51.6451416015625	2.31551837921143	1.49496734142303
1462388	4	2	3.4333298206...	11.9666547...	0.000519...	51.6451416015625	2.31551837921143	-1.49496734142303
1454529	2	4	11.666654586...	3.09999680...	0.000521...	43.0905227661133	5.31211757659912	1.52733254432678
1454530	4	2	3.0999968051...	11.6666545...	0.000521...	43.0905227661133	5.31211757659912	-1.52733254432678
1455313	2	4	11.666654586...	3.13333010...	0.000533...	43.0905227661133	5.31211757659912	1.51399326324463
1455314	4	2	3.1333301067...	11.6666545...	0.000533...	43.0905227661133	5.31211757659912	-1.51399326324463
1453743	2	4	11.633321762...	3.06666350...	0.000543...	41.3260726928711	3.90723443031311	1.52860629558563
1453744	4	2	3.0666635036...	11.6333217...	0.000543...	41.3260726928711	3.90723443031311	-1.52860629558563
4964115	6	10	0.7999991774...	0.83333247...	0.000544...	81.8960952758789	5.51530981063843	-0.0772774517536163

Figure 3: The animation table of the output database

The experiment was conducted as follows: after the creation of the motion graph, the resulting animation ran while a background process calculated the average distance between the frames chosen for the transitions. This was necessary because the framework did not include any motion blending, making the evaluation of naturalness of the motion very difficult. The average distance allowed us to judge the performances of the different metrics with objective data. Secondly, the graph density was measured and compared for the different thresholds and metrics; a dense graph is a graph in which the number of edges is close to the maximal number of edges and graph density is the property that define if a graph is a dense or sparse one, with values in the interval [0,1]. The denser the graph is, the more connected are the different motion clips. Graph density is calculated as:

$$D = \frac{2|E|}{|V|(|V| - 1)}$$

where E is the number of edges and V is the number of vertices of the graph. The results showed an advantage of the hybrid metric over the other two that were tested in terms of average distance. The time needed to build a motion graph varied greatly according to the threshold values selected by the user and by the type of metric utilized, ranging between 80 and 17400 seconds. The average distance considered for the results represents the distance between the poses, in meters; the smaller it is, the more similar are the poses, which leads to better blending results.

3.2.1 Point cloud distance metric

First of all, the experiment was carried out on the point cloud distance metric; thus the only parameter that was modified was the point cloud distance threshold. The values of this threshold ranged between $0.003m$ and 0.030 m, with a pace of 0.003 m, for a total of 10 different values. The average time to build the motion graph was 2977 seconds, with the values towards the end of the interval taking considerably more time than those at the beginning; a bigger threshold means more transitions and thus a bigger build time for the graph. The results observed using this metric showed that the average distance between the transition frames increased with the growth of the threshold value, ranging from 0.00223927 m to 0.0184824 m. The graph density is also in a relationship of direct proportionality with the thresholds, with higher thresholds having a higher value for the edge density parameter, as a result of the less restrictive conditions to create a transition. The results are showed in the following charts.

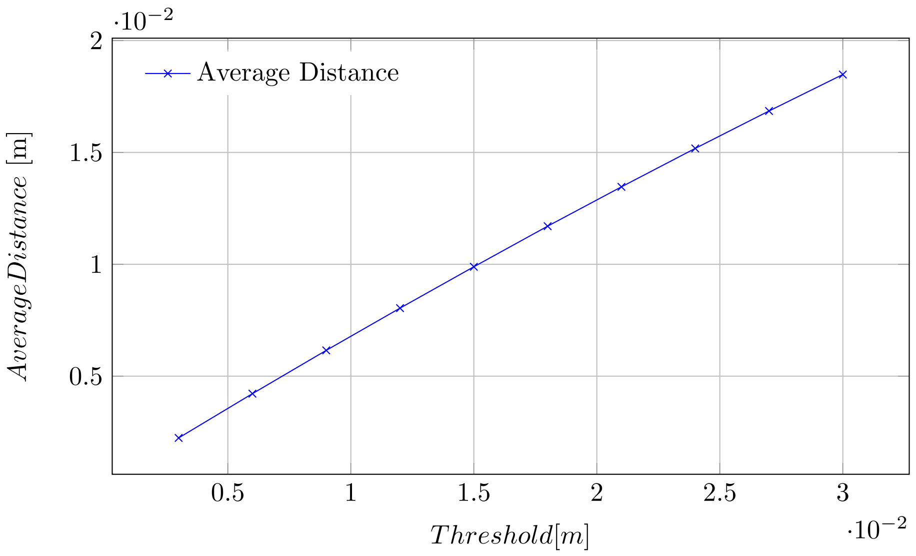


Figure 4: Average distance chart that shows the growth of the average distance with the threshold for the point cloud distance metric.

From the results we can deduce that low values of the thresholds will guarantee better results in terms of visual quality, but with more restriction on the number of animations that can be created.

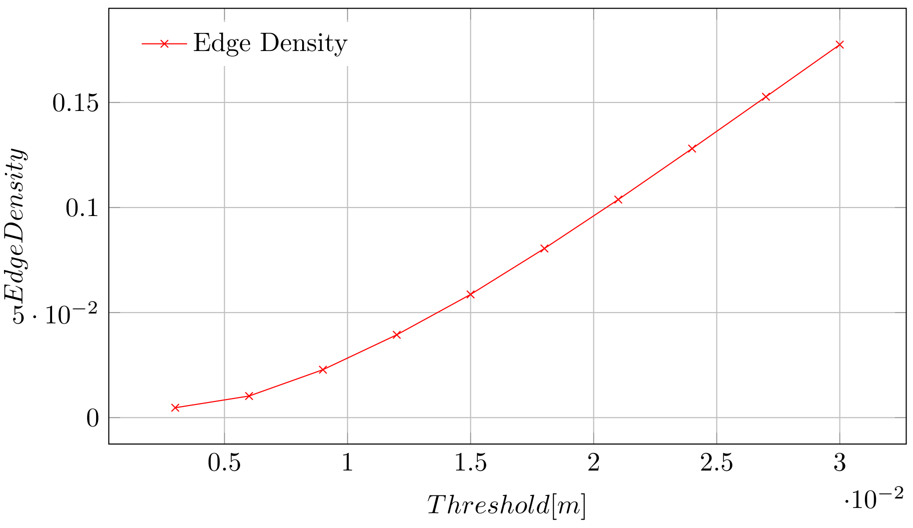


Figure 5: Edge density chart that shows the growth of the edge density with the threshold for the point cloud distance metric.

3.2.2 Momentum-based distance metric

The point cloud distance metric experiment was followed by the experiment on the new momentum-based metric. In this case, the variables were represented by the linear momentum distance and the angular momentum distance thresholds. The former varied between 10 $\frac{Kgm^2}{s}$ and 100$\frac{Kgm^2}{s}$, with a pace of 10$\frac{Kgm^2}{s}$, while the latter went from 1.5$\frac{Kgm}{s}$ to 15$\frac{Kgm}{s}$, with a pace of 1.5$\frac{Kgm}{s}$. Every value of the linear momentum threshold was tested with every value of the angular momentum threshold. The average time to construct the graph was about 6448 seconds, more than twice the time required by the point distance metric. Additionally, an unforeseen low memory error was raised when the value of the linear momentum distance threshold was set between 80$\frac{Kgm^2}{s}$ and 100$\frac{Kgm^2}{s}$. For this reason, some of the cases were considered incomplete and discarded in the final evaluation. The results observed with this metric can be considered quite unsatisfactory; the following chart shows the results when the angular momentum threshold is fixed at 1.5 $\frac{Kgm}{s}$.

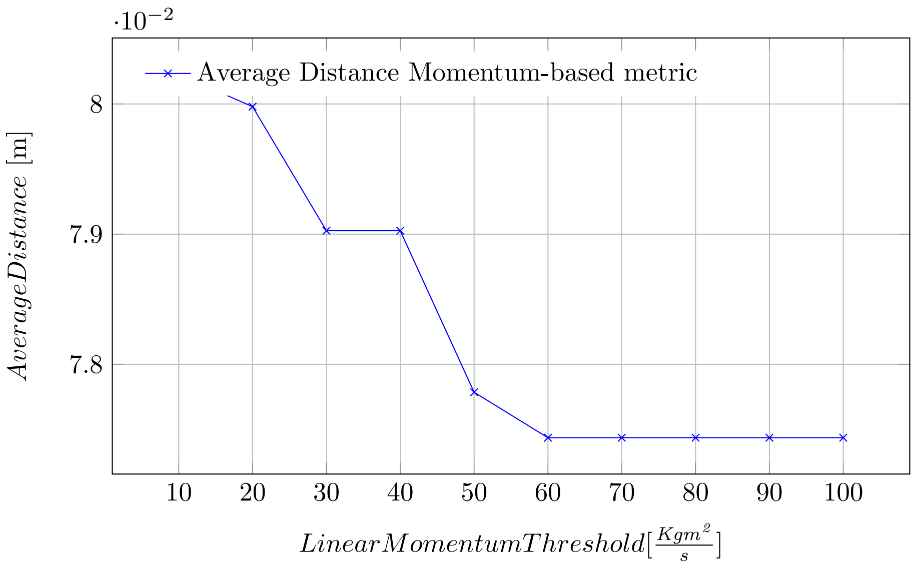


Figure 6: Average distance chart that shows the reduction of the average distance with the growth of threshold for the momentum-based distance metric.

From the chart it can be noticed that, in this case, the average distance between the poses decreases as the momentum thresholds grows. In addition to that, if we look at the numbers, we can actually see that even for the thresholds that have the lowest average distance, the actual value is much higher with respect to the point cloud distance metric values. Obviously this is not the behaviour that we were looking for and it confirms the idea that a pure momentum-based distance metric is not adequate to build motion graphs that result in satisfying animations. On the contrary, if we take a look at the data for the edge density, we can see that the beaviour does not differ from the expected one, with bigger thresholds having bigger density values. In this chart we show the result when the angular momentum threshold is fixed at 1.5 $\frac{Kgm}{s}$.

3.2.3 Hybrid distance metric

Finally, we tested the last distance metric, a combination of the first two. In practical terms, this means that the transitions were created using the point cloud distance thresholds and the momentum distance thresholds together. As in the previous experiments, the values of the thresholds varied between 0.003m and 0.030 m for the point cloud distance threshold, between 10 $\frac{Kgm^2}{s}$ and 100$\frac{Kgm^2}{s}$ for the linear momentum distance threshold and between 1.5$\frac{Kgm}{s}$ and 15$\frac{Kgm}{s}$ for the angular momentum distance threshold. This brought the final number of tested values to 100. An excerpt of the collected data can be seen in Figure 8.

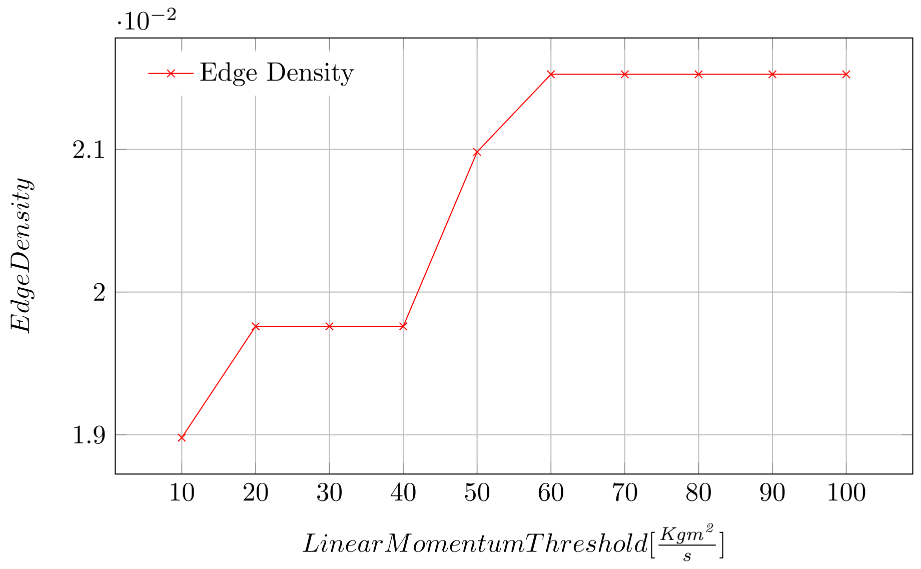
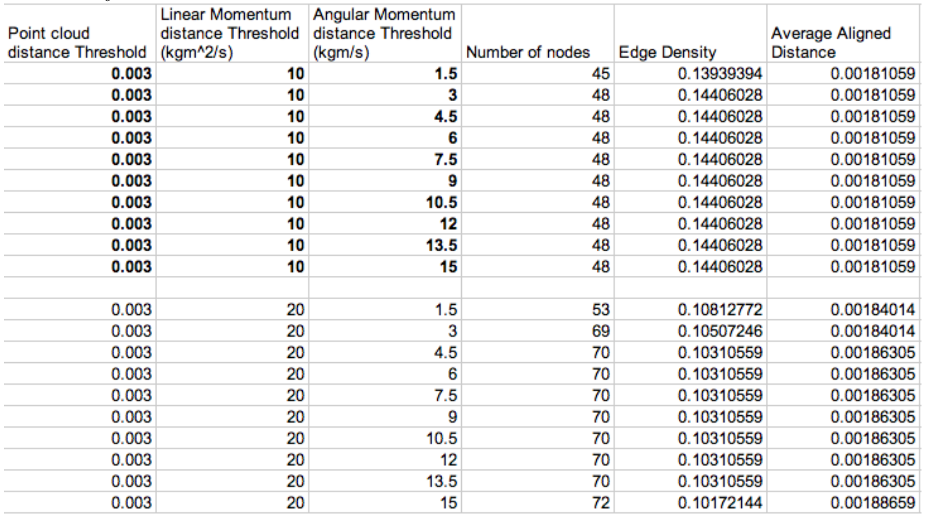


Figure 7: Edge density chart that shows the growth of the edge density with the linear momentum threshold and a fixed thresholf for the angular momentum.

The average time for the graph construction was 10600 seconds. Generally we noticed a reduction of the average distance between transitions; comparing it with the point cloud distance metric, we found that in the best case, in which the results showed the lowest average distance for both the metrics, the percentage reduction was up to 19%. Comparing the hybrid metric with the momentum-based metric showed that, in the best case, the contraction was up to 97%. The following chart shows the results when the linear momentum threshold is fixed at 10 $\frac{Kgm^2}{s}$ and the angular momentum threshold is at 1.5$\frac{Kgm}{s}$, and compares them with the point cloud distance curve showed before:

Figure 8: Values for the thresholds used in the generation of the motion graph for the hybrid distance metric.



Point cloud distance Threshold	Linear Momentum distance Threshold (kgm^2/s)	Angular Momentum distance Threshold (kgm/s)	Number of nodes	Edge Density	Average Aligned Distance
0.003	**10**	**1.5**	45	0.13939394	0.00181059
0.003	**10**	**3**	48	0.14406028	0.00181059
0.003	**10**	**4.5**	48	0.14406028	0.00181059
0.003	**10**	**6**	48	0.14406028	0.00181059
0.003	**10**	**7.5**	48	0.14406028	0.00181059
0.003	**10**	**9**	48	0.14406028	0.00181059
0.003	**10**	**10.5**	48	0.14406028	0.00181059
0.003	**10**	**12**	48	0.14406028	0.00181059
0.003	**10**	**13.5**	48	0.14406028	0.00181059
0.003	**10**	**15**	48	0.14406028	0.00181059
0.003	20	1.5	53	0.10812772	0.00184014
0.003	20	3	69	0.10507246	0.00184014
0.003	20	4.5	70	0.10310559	0.00186305
0.003	20	6	70	0.10310559	0.00186305
0.003	20	7.5	70	0.10310559	0.00186305
0.003	20	9	70	0.10310559	0.00186305
0.003	20	10.5	70	0.10310559	0.00186305
0.003	20	12	70	0.10310559	0.00186305
0.003	20	13.5	70	0.10310559	0.00186305
0.003	20	15	72	0.10172144	0.00188659

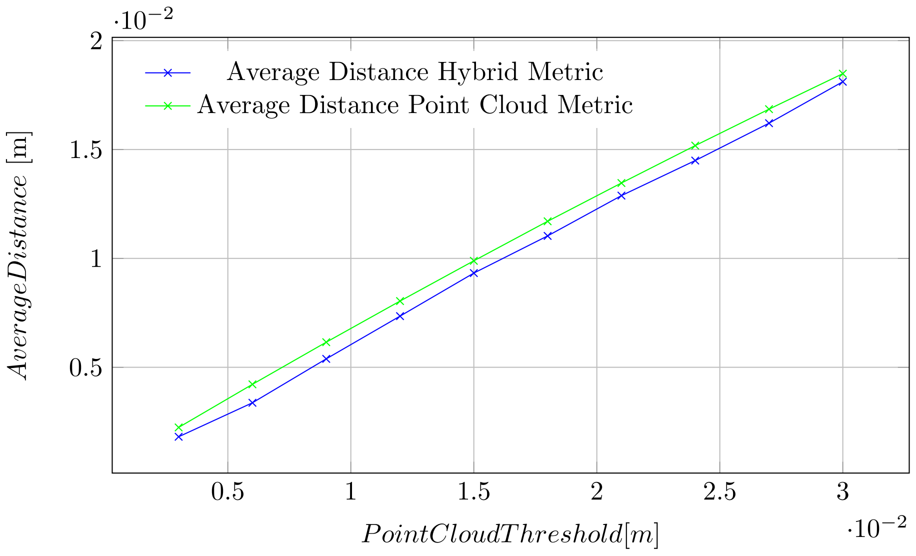


Figure 9: Average distance chart that compares the point cloud distance metric curve with the hybrid distance metric curve .

We can clearly see the advantage of the hybrid distance metric curve that always stays under the point cloud distance metric one. Unfortunately, we did not find the same positive results in terms of graph density: if we look at the charts for the graph density with the same parameters, we can see something unexpected:

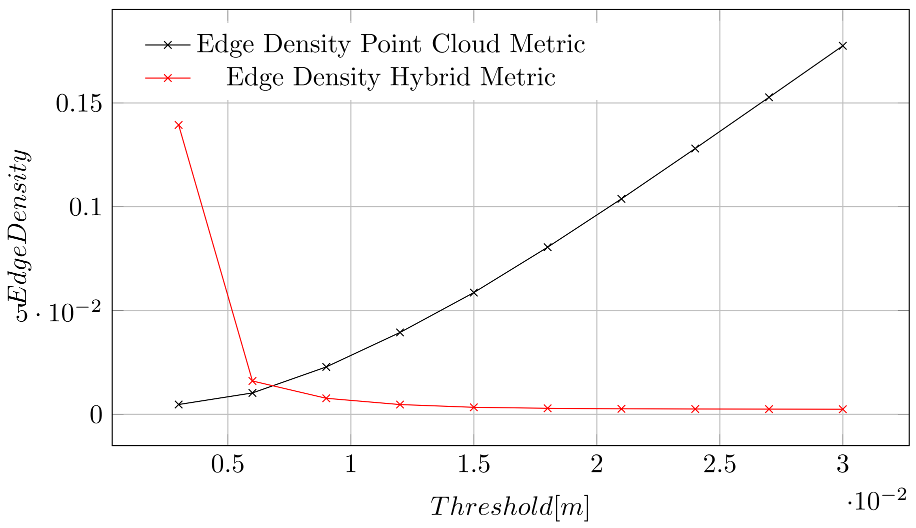


Figure 10: Edge density chart that compares the edge density curves for the point cloud distance metric and the hybrid distance metric.

We can immediately notice that the density curve for the hybrid distance metric is decreasing, instead of increasing like it would be expected. In its maximum value, the hybrid curve is still well under the maximum density value reached by the point cloud distance, which puts the hybrid distance metric at disadvantage. Comparing the numbers, we actually have a cutback of about 18.85%. The same can be observed if we compare the hybrid distance metric with the pure momentum-based one: the contraction reaches the value of 70%, in the best cases, when the density values are the highest for both of the metrics. The reasons for this behaviour are still not clear and are being investigated.

4 Conclusion

In conclusion, a distance metric based on physical properties of a moving character was developed and tested to evaluate possible improvements against one of the metrics that are currently used to generate motion graphs.

In particular, it was compared to the point-cloud distance metric, a technique also discussed by Mémoli and Sapiro [2004]. The analysis showed that a pure momentum-based metric by itself did not produce interesting results; the main problem was that angular and linear momentum are properties that, by themselves, do not describe movements in a way that is suitable to the task of building a motion graph. However, the implementation of a metric that takes into account both the distance between the poses and the similarities in momentum showed promising results that could help improving the final quality of the animations.

Further research should be aimed at improving the graph connectivity, which would guarantee more freedom in the creation of different animations. Moreover, further physical properties could be used to increase the precision of the metric, like torques and forces; also, it would be worth exploring different kinds of technique for pose recognition and doing a comparative study on the quality of the resulting animations.

References

Yeuhi Abe, C. Karen Liu, and Zoran Popović. Momentum-based parameterization of dynamic character motion. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, SCA '04, pages 173–182, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association. ISBN 3-905673-14-2. doi: 10.1145/1028523.1028546. URL `http://dx.doi.org/10.1145/1028523.1028546`.

Okan Arikan and D. A. Forsyth. Interactive motion generation from examples. *ACM Trans. Graph.*, 21(3):483–490, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566606. URL `http://doi.acm.org/10.1145/566654.566606`.

Arjan Egges, Tom Molet, and Nadia Magnenat-Thalmann. Personalised real-time idle motion synthesis. In *Computer Graphics and Applications, 2004. PG 2004. Proceedings. 12th Pacific Conference on*, pages 121–130. IEEE, 2004.

K. Forbes and E. Fiume. An efficient search algorithm for motion data using weighted pca. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '05, pages 67–76, New York, NY, USA, 2005. ACM. ISBN 1-59593-198-8. doi: 10.1145/1073368.1073377. URL `http://doi.acm.org/10.1145/1073368.1073377`.

Michael Gleicher, Hyun Joon Shin, Lucas Kovar, and Andrew Jepsen. Snap-together motion: assembling run-time animations. SIGGRAPH '08, pages 52:1–52:9, New York, NY, USA, 2003. ACM. doi: 10.1145/1401132.1401203. URL `http://doi.acm.org/10.1145/1401132.1401203`.

Lucas Kovar, Michael Gleicher, and Frédéric Pighin. Motion graphs. SIGGRAPH '02, pages 51:1–51:10, New York, NY, USA, 2002. ACM. doi: 10.1145/1401132.1401202. URL `http://doi.acm.org/10.1145/1401132.1401202`.

Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, July 2002. ISSN 0730-0301. doi: 10.1145/566654.566607. URL `http://doi.acm.org/10.1145/566654.566607`.

Lei Li, James McCann, Christos Faloutsos, and Nancy Pollard. Laziness is a virtue: Motion stitching using effort minimization. *Computer Science Department*, 2008.

Facundo Mémoli and Guillermo Sapiro. Comparing point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, SGP '04, pages 32–40, New York, NY, USA, 2004. ACM. ISBN 3-905673-13-4. doi: 10.1145/1057432.1057436. URL `http://doi.acm.org/10.1145/1057432.1057436`.

S. Plagenhoef, F.G. Evans, and T. Abdelnour. Anatomical data for analyzing human motion, 1983.