Four Dimensional Embedding for Augmented Optimal Transport Between Human Body Regions

Quy-Dzu Do
qndo@ucsd.edu

Matt Tokunaga
m2tokunaga@ucsd.edu

Alex Cloninger
acloninger@ucsd.edu

Rayan Saab
rsaab@ucsd.edu

Abstract

Optimal transport is an active area of mathematics and data science research that focuses on transporting one probability distribution to another. Viewed through the right lens, this can also apply to point clouds. These transports plans can then be thought of as point-to-point correspondences. This paper focuses on using methods in optimal transport to establish pointwise correspondences between point clouds of human body surface mesh data. In particular, we establish an algorithm based on embedding three dimensional data into a four dimensions using a left-right disambiguation scheme that helps distinguish between otherwise very similar body parts.

Website: `https://qndo1.github.io/motion_capture_ot_website/`
Code: `https://github.com/qndo1/motion_capture_ot`

1 Introduction . . . 2
2 Data . . . 3
3 Methods . . . 3
4 Metrics . . . 6
5 Results . . . 7
6 Discussion . . . 7
7 Conclusion and Possible Extensions . . . 8
References . . . 9
Appendices . . . A1
B Contributions . . . A1

1 Introduction

Algorithms for matching points play a crucial role in analyzing and comparing geometric and temporal data, particularly in domains such as object recognition in autonomously driving vehicles, where Zhang and Singh (2015) used the idea of point registration to combine information from both visual and lidar systems. In a domain such as Motion Capture (MoCap), it's easy to use the idea of point registration, since the points are labeled and are positioned on the same point in the body across different frames. With general lidar style data, however, which is high spatial density but lacks a ground truth frame-to-frame point registration, matching individual points can be harder, if it even makes sense to do at all. Thus we're interested not as much in the specific point to point registration, but a region to region registration. Instead of asking if a particular point got mapped to the correct point, we ask only if it got mapped to the correct region.

One way of getting around this is to use a more abstract type of matching. In Rodolà et al. (2015), instead of matching points in one point cloud to points in another point cloud, they find a function that maps functions on one surface to functions on the other surface by way of a linear map between Fourier coefficients of an eigenfunction basis. However, this method is computationally and temporally expensive, requiring about five minutes to establish a map between two point clouds using an alternating optimization scheme where they optimize over the linear map as well as the mask of missing points. This works very well for the partial maps that they have (where there's large holes or cuts in the data), but we restrict ourselves to a cleaner dataset, which will be described in the data section.

For our solution to the problem of region to region matching, we draw from the field of optimal transport. From a theoretical point of view, optimal transport is an area of mathematics focused on distances and transport plans between two probability distributions. Transport plans in this context are couplings between probability distributions. Thus, the optimal transport plan is given by:

$$\inf_{\gamma \in \Gamma_{\mu,\nu}} \int c(x,y)\gamma(dx,dy)$$

In words, this is the transport plan γ that minimizes the cost function $c(x,y)$ over the whole space. For our purposes, and in fact in many practical uses, our probability distributions will be point clouds. Given N points $x_i \in \mathbb{R}^n$, one can always define a probability measure in the following way:

$$\frac{1}{N}\sum_{i=1}^{N} \delta_{x_i}$$

This is a discrete probability measure consisting of a Dirac at each point in our point cloud (and weighed accordingly so that total measure is 1). A transport plan in this context, assuming a equal number of points in each point cloud and no regularization, is a (scaled) permutation matrix.

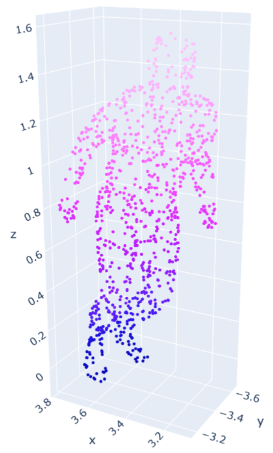


Figure 1: Example sampled point cloud (1000 Points)

2 Data

The best use case for our algorithm is lidar-style data, which has very high spatial density. Since we don't have access to that, we generated synthetic data using the ACCAD (Advanced Computing Center for the Arts and Design (2199)) dataset, obtained from the AMASS (Mahmood et al. (2019)) website. Using the files in the dataset, we were able to construct mesh objects. Using these mesh objects we were able to sample points from the surface of the mesh. While sampling an arbitrary number of points is possible, we used 1000 points for the entirety of our analysis. Our methods also rely on the two point clouds having the same number of points, but in a real world scenario the clouds could be downsampled to a common number.

The temporal resolution of the data is also quite high. The meshes are 120 frames per second for each action. There are dozens of separate actions, including male/female running, walking, turning, etc, as well as more exotic actions like martial arts and lifting boxes. We limit our analysis to the more simple ones of male/female walking and running.

Each point has an associated face index that it was sampled from, and using a calibration file we were able to manually define 16 body regions based on the face indices to use for our accuracy metric. Note that this is only for ground truth comparisons, and our algorithm does not have access to these face indices, only the x, y, and z coordinates of each point.

3 Methods

The central problem that our algorithm solves is a left/right disambiguation. As will be discussed more in the results section, a baseline of simply finding an optimal transport map between the two point clouds performs generally well, especially if the two frames aren't very far apart temporally. However, one thing that it consistently struggles with is

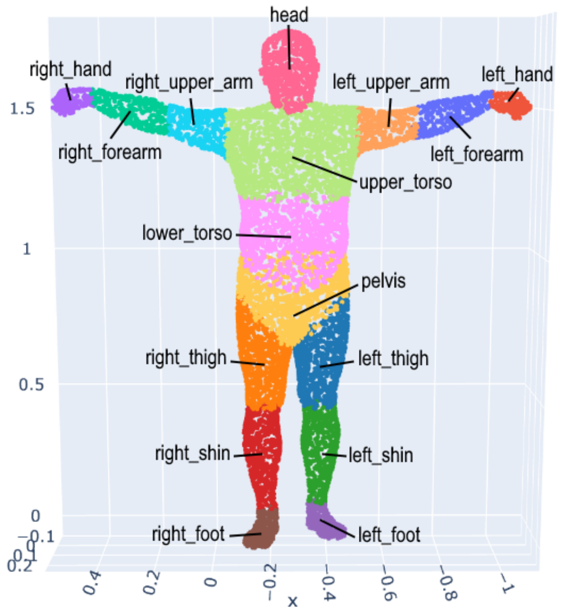


Figure 2: Region labels on a calibration pose with 10,000 sampled points

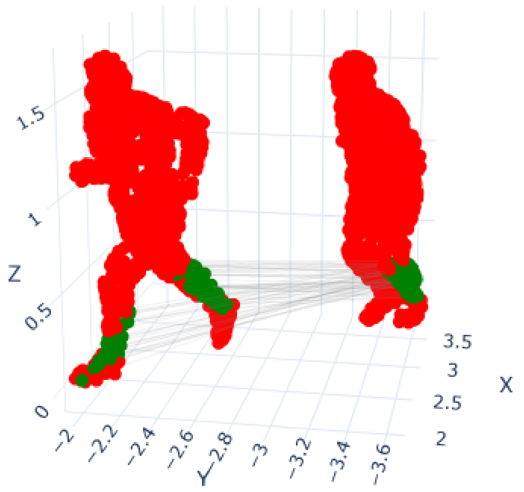
consistently mapping the lower body. The positions of the two legs vary wildly, and in many instances points from one leg get mapped to points on the other leg. An example of this phenomenon is shown in Figure 3.

The general strategy is to, instead of doing an optimal transport between the two point clouds, first embed the points into four dimensions, where the fourth dimension captures some notion of left vs. right, and then do an optimal transport matching on the four dimensional point clouds.

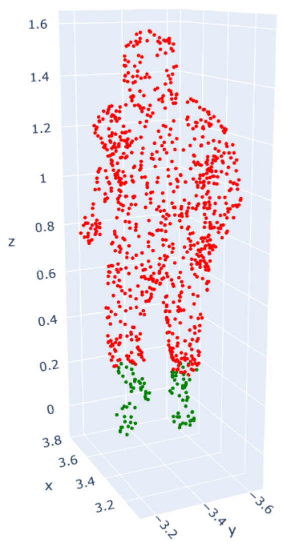
The first step is to isolate the bottom 10% of points by z coordinate. In most normal cases, this will isolate the two feet. However, it does mean we are making a relatively big assumption that the point clouds represent people in a roughly vertical pose. An example of the points captured by this process is given in Figure 4.

After isolating the bottom 10% of points in a point cloud, we project them down to the x-y plane and run a K-means algorithm with $k = 2$ to find the center of each foot (the center being defined as the point closest in projected space to the cluster mean). To assign one of these as the ”left” cluster and one as the ”right” cluster, we make another assumption: that the person is moving in a generally forward direction. We take the vector from the mean of the points in the first frame to the mean of the points in the second frame. Taking the cross product between that direction vector and the purely vertical vector, we can define a consistent axis where positive is left and negative is right. We then take the most positive cluster center to be the left cluster and vice versa.

The important part of the preceding section is that we have a point representing the left cluster, and a point representing the right cluster. We’ll call these points our anchor points. We can then define any other point’s ”leftness” and ”rightness” as their relative distance



(a) Figure 3a: A baseline matching where points on the left shin get mapped to both the right and left shin.



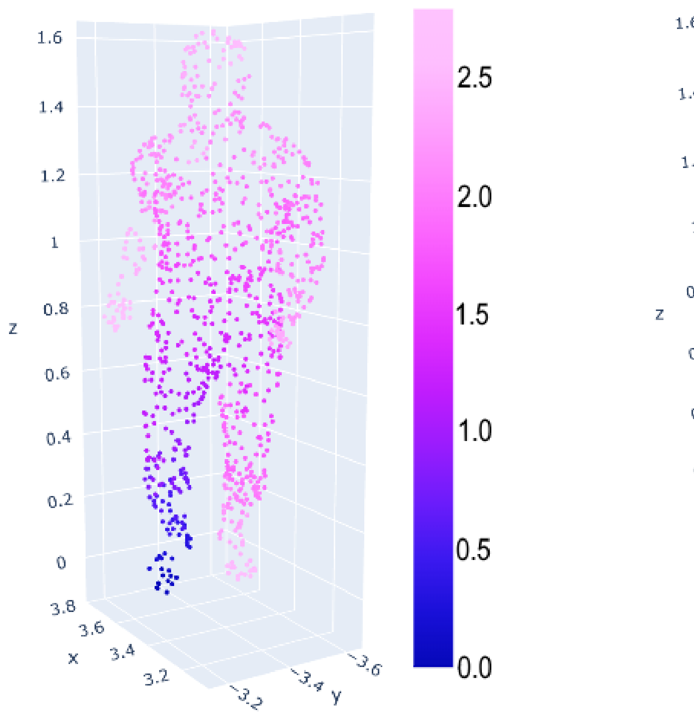
(b) Figure 3b: Bottom 10 percent of points in green

from these anchor points. A point will be considered to have high rightness if it is much closer to the right anchor point than the left, and vice versa. To define these distances correctly, however, it's important to have a notion of distance that respects the geometry of the human body.

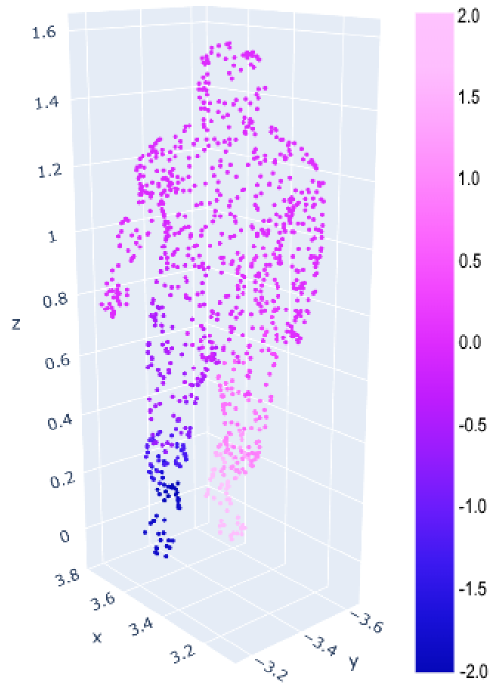
To do this, we constructed a graph of the human body in the following way. First, connect each sampled point to its 3 nearest neighbors. Then, if the graph is not already connected, make the shortest connections possible (by Euclidean distance) to connect two components and repeat until the graph is fully connected. Figure 5 shows the graph distances between every point and a point on the right foot.

Once we have, for each sampled point, the distances from both the right and left anchor points, we can subtract the distances. Thus if a point is much closer to the right anchor point, it will have a negative relative distance, and if a point is much closer to the left anchor point its relative distance will be positive. Note that for most points above the thighs, the relative distance will be basically zero. We can then use this relative left/right distance as another variable, essentially embedding each point into four dimensions.

Once the data is embedded into four dimensions, we run a simple optimal transport matching between the two point clouds. We run this without any sort of regularization so that a true one to one matching is obtained. With regularization, a point in the first frame might have half of its mass sent to one point and the other half sent to another point, which we're trying to avoid.



(a) Figure 4a: Distance from a particular point on the right foot to each other point.



(b) Figure 4b: Example of the automatically generated left/right spectrum.

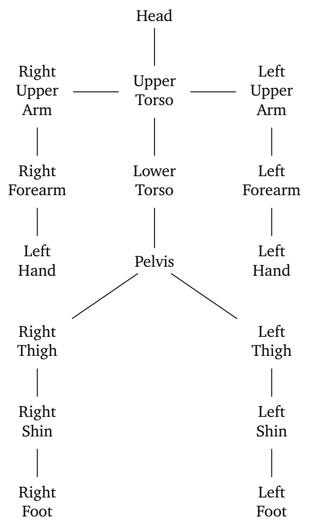
4 Metrics

To evaluate our algorithm, we use two metrics. The first, called *Adjusted Accuracy*, measures whether points are matched to the same anatomical region they originated from. Using the 16 manually defined body regions shown in Figure 2, we determine the region of each sampled point in both the first and second frames. For every matched pair of points, we check whether the two points belong to the same region and count how many matches satisfy this condition. Because the number of sampled points in a region may differ between frames, a perfect matching is not always possible. For example, if 3 points are sampled from the left hand in frame 1 but 4 points are sampled from the left hand in frame 2, at most 3 correct matches can exist. To account for this, we normalize by the maximum possible number of correct matches. This quantity is computed by summing, over all regions, the minimum of the number of sampled points from that region in each frame. Thus, adjusted accuracy is the fraction of region-consistent matches out of the maximum number that could possibly be correct:

$$\text{Adjusted Accuracy} = \frac{\#\ \text{same-region matches}}{\sum_{r \in R} \min\left\{n_r^{(1)}, n_r^{(2)}\right\}}$$

where $n_r^{(1)}$ is the number of points in region r in frame 1, and the same is true for $n_r^{(2)}$.

The other metric we used is called *Average Region Distance*. The idea is that one incorrect matching can still be better than another incorrect matching if the regions are closer together. For example, mapping a point from the left hand to the left forearm is ”incorrect,” but it’s still better than mapping a point from the left hand to the right hand. This Average Region Distance accounts for this by calculating the distance from each point to the region it was supposed to be mapped to. The distances are defined as being discrete natural numbers based on the following graph:



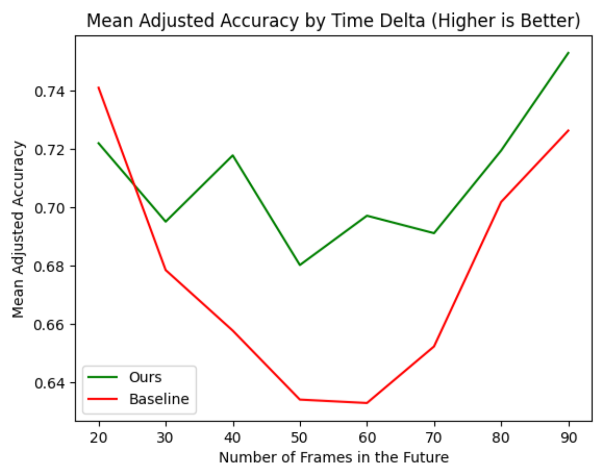
So if a point originates from the left hand and is mapped to a point on the upper torso, that pairing gets a Region Distance of 3. The Average Region Distance is then an average over all of the pairings. Note that here a lower number is better, while for Adjusted Accuracy a bigger number is better.

5 Results

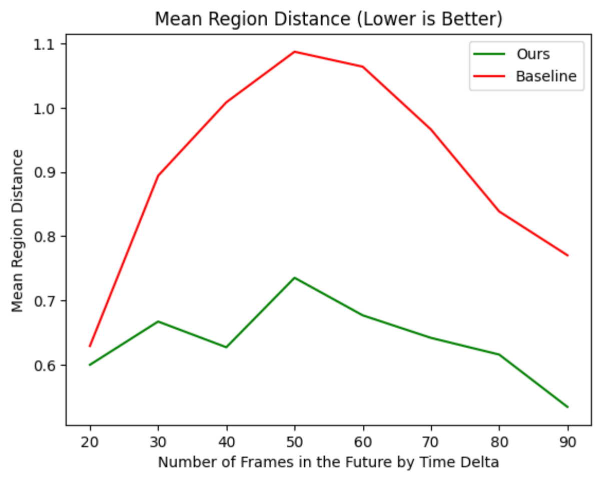
We compared our algorithm to a baseline of an optimal transport mapping on the unaugmented 3 dimensional coordinates. To compare, we randomly sampled 20 frames from our actions dataset. Then, for each frame, we get mapped between that frame and the one 20 frames in the future, then 30 frames in the future, then 40, etc until 90 frames in the future. With 20 starting frames, we get a total of $20 \cdot 8 = 160$ baseline matchings and 160 augmented matchings. We average the metrics across different starting frames. So all the +20 frame matchings get averaged, all the +30 frame matchings get averaged, etc. The results are in Figure 5.

6 Discussion

We see that our algorithm outperforms the baseline. Optimal transport tends to be very good when the point clouds are extremely close together, so it is not surprising that the



(a) Our method outperforms the baseline approach once the frames are more than about 20 frames apart.



(b) Our method consistently outperforms the baseline algorithm.

Figure 5: Overall, our method tends to outperform the baseline. In terms of accuracy, the baseline approach works better for frames that are very close together, but in the 40-70 frame range our algorithm performs much better. The rise in accuracy for both methods around 80 frames in the future is explained by the cyclical nature of walking and running.

baseline performs very well when the frames are not very far apart. However, after about 20 frames in the future, out algorithm consistently outperforms the baseline. The difference is most noticeable in the 40-70 frame range. We see that our algorithm is more stable and less prone to performance decay than the baseline. Both the Adjusted Accuracy and Mean Region Distance get worse only slightly for our algorithm when looking further into the future.

We also note that, for both methods, the performance drops and then goes back up after about 80 frames. While this is unintuitive, it makes sense when one considers the fact that walking and running, which is what our data consists of, is cyclic. Thus we would expect the point clouds to, during certain frames, be very similar (except translated forwards). Even when that occurs, however, our algorithm still outperforms the baseline.

7 Conclusion and Possible Extensions

We have successfully showed that distinguishing between the left and right sides of the body is an effective way to improve the performance of an optimal transport based matching between point clouds. The biggest issue with the naive approach tends to be the left and right sides of the lower body getting mapped to each other, and our 4-dimensional embedding technique gets around that issue.

There are a number of ways that this work could be extended. We addressed the lower body issue, but the arms still present a unique challenge. The positions of the arms vary more than the legs, and the algorithm has a particularly difficult time distinguishing them from the torso when the arms are held against the chest/stomach. A graph based structure could potentially help with this, but for our graphs the arms tended to be connected to the torso, so it's somewhat delicate and would require a novel solution.

Given that the general geometry of the body doens't change, it just deforms, one might expect a graph-only method to perform quite well. Geodesic distances would hopefully not change and could be used. We investigated using graph based methods, including graph distances with Gromov Wasserstein and Fused Gromov Wasserstein, but they did not seem to perform better than the baseline. Still though, we suspect that they could help in some way.

Finally, we believe our Average Region Distance metric is a bit coarse. The body region cutoffs are somewhat arbitrary, so it would be nice to have a distance metric that varies continuously. Using our graph, we could have done some sort of graph geodesic distance, but its complicated by the fact that the points sampled on each frame are random and unregistered. Thus there is really no ”ground truth” point to find the distance between. Given the face indices, it could be possible to use the face centers as the ground truth and create a distance based on that.

References

Advanced Computing Center for the Arts and Design., "ACCAD MoCap Dataset." [Link]

Mahmood, Naureen, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. 2019. "AMASS: Archive of Motion Capture as Surface Shapes." In *International Conference on Computer Vision*.

Rodolà, Emanuele, Luca Cosmo, Michael M. Bronstein, Andrea Torsello, and Daniel Cremers. 2015. "Partial Functional Correspondence." [Link]

Zhang, Ji, and Sanjiv Singh. 2015. "Visual-lidar odometry and mapping: low-drift, robust, and fast." In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. [Link]

Appendices

A.1 Project Proposal A1

A.1 Project Proposal

Motion capture data has become the gold standard when it comes to translating actions done by a person into computer software. Hollywood and games animation studios, having used this technique for decades, have generated a large amount of motions and animations that can be used for data science applications. However, a problem that can often be encountered with uncleaned or unprocessed motion capture data is how to effectively relabel points automatically. In practice, most commercial practices would use strict setups to have auto-labeling on the key frames as they are being captured, but archival data does not have that luxury. Thus, it would be useful to make a reliable system that can label Motion Capture data without the use of dedicated systems to auto-label points or manually go through the points in archival data.

Other papers have sought to achieve similar goals. One particular paper (https://arxiv.org/pdf/2110.04431) opted to use a Neural Network to classify points from a singular frame of Motion capture. Though it achieved very good F1 scores, the authors noted that the model would have difficulty identifying poses not included in the training set though this could be alleviated by having a larger more diverse dataset. We seek to find a method that would be less dependent on relying on known data, instead focusing on the rigid structure of the human body as a key measure in finding similarities between points thus tracking them through a temporal dimension. We will work with analysis with many frames to utilize the rigidity of certain bones to find the skeletal structure of a person as well as how upsampled data that may come from LiDAR can be used to help find the skeletal structure in single frames.

In this paper, we will empirically test our models, utilizing the AMASS dataset with its vast array of meshes to serve as our LiDAR and Motion capture data. We will also provide an action classification to show uses in autonomous vehicles where LiDAR and video feed would be a more common sensor than a motion capture setup.

B Contributions

Quy-Dzu Do was responsible for researching and testing graph construction methods, as well as graph-based mapping methods. He also ran experiments to compare different optimal transport-based methods.

Matt Tokunaga was responsible for data loading and preparation, as well as graphics generation. He also ran experiments on different left/right disambiguation methods as well as methods utilizing Wasserstein and fused Gromov-Wasserstein.