Improving Conditioned Human Motion Synthesis via Latent Noise Optimization

Kai Zhang*, Öykü Irmak Hatipoğlu*, Ross Roessler*, Bingjie Xue*
ETH Zurich, Switzerland
{zhakai, ohatipoglu, rroessler, binxue}@student.ethz.ch

Abstract

This project extends Diffusion Noise Optimization (DNO) to real-world scenarios with multiple, complex-shaped obstacles by implementing more generalized methods to compute SDFs. We propose three methods to calculate SDFs for diverse obstacles and integrate them into a differentiable loss function respectively. Using motion in-betweening from DNO, our experiments show that our proposed loss functions effectively enable obstacle avoidance in real-world scenarios while maintaining the content preservation and foot skating performance of DNO. Additionally, our methods achieve nearly comparable performance in goal-reaching and joint intersection metrics compared to the Reinforcement Learning-based method DIMOS.

1. Introduction

In recent years, diffusion models have had a prominent performance in the text-to-motion field. Diffusion Noise Optimization (DNO) [11] can optimize the diffusion latent noise of an existing motion diffusion model. It can be edited to realize different tasks, including reaching target locations and avoiding obstacles. This is made possible by propagating the gradient from a target loss function through the whole denoising process. By changing the loss, which is a function of motion and obstacles, DNO can control the generated motions.

DNO can only realize avoiding spherical obstacles, which are represented as a Signed Distance Function (SDF). The SDF for a sphere can be calculated by the distance of joints from the center of the sphere minus the radius of the sphere. **The goal** of this project is to extend it to a real scene, which means that there will be multiple obstacles with complex shapes, which requires implementing a more generalized method to compute the SDF. We aim to achieve obstacle avoidance in real-world scenarios while maintaining the content preservation performance of DNO.

*These authors contributed equally to this work.

We propose 3 methods to calculate a SDF which represents obstacles in real scenes, and add it into the differentiable loss function. We use motion in-betweening from DNO, whose inputs are the starting pose and the ending pose. Our experiments show that our proposed loss functions fulfill obstacle avoiding in real scenes, as well as showing nearly-comparable goal reaching and joint intersection performance with Reinforcement Learning-based method DIMOS [32]. In terms of content preservation and foot skating performance, our results also show comparable performance with original DNO results.

2. Related Work

Human Motion Generation. Recent advances in human motion generation tasks have introduced various models for synthesizing conditioned human motion [33]. One of the main goals of human motion generation is to generate diverse human motion [2] [23] [11] [12] [20]. On the contrary, there are also some subtasks of human motion generation that do not focus on diversity. Motion infilling emphasizes the accuracy of the motion, instead of variety [8] [13]. This method aims to complete the motion when only sparse keyframes are given. Most generation methods focus on human motion conditioned on text input [30] [6] [29] [27] [3] [16] which is made possible by text encoders like CLIP [19]. Other methods can generate motion with additional conditions such as scene [24] [25] [10], action class [7] [15] [1], and even music [4] [14] [22]. However, for realistic human motion synthesis, achieving accurate motion semantics is not enough. Another important aspect of motion generation is the spatial information provided by trajectories and keyframe locations. Guided Motion Diffusion (GMD) [12] incorporates spatial constraints into motion generation to connect isolated human motion to an environment. To further increase the realness of the motion, interactions with the scene should also be considered.

Scene-aware Motion Synthesis. While generating realistic motion is one of the key concerns of human motion generation, many methods neglect the necessity of human-scene interactions. Models that can work with realistic en-

vironments such as scene meshes are crucial for more natural motion synthesis. Due to the sparsity of the datasets regarding human-scene interaction, Wang et al. proposed the HUMANISE dataset [27]. Annotated by textual data, HUMANISE is used to train a scene-and-language conditioned generative model that can synthesize humans interacting with the scene. Other methods exploited scene affordance information [24] [26] or controlled motion synthesis by optimizing path finding algorithms [23] [31] [10] to predict plausible human-scene interaction. There are several ways to represent scenes in scene-aware motion synthesis. Spheres are commonly used to replace the obstacles in several methods such as GMD [12] and DNO [11]. In GMD, the spatial information is used in the denoising process of the diffusion model whereas in DNO, a SDF between the spheres and joints is used to optimize the diffusion latent noise. Wang et al. represented the scene as a point cloud and used depth maps to represent the scene geometrically and measure the non-collision ratio [25]. In contrast, DIMOS calculated the SDF between scene meshes and human joints as collision avoidance rewards in its reinforcement learning-based training [32].

Ours vs. Others. Our model improves the DNO model by using scene mesh SDFs instead of simple sphere obstacles. Similar to DIMOS, we calculate the SDF between the mesh and human joints but we use the calculated SDF loss to optimize the diffusion latent noise. Since making calculations directly on a scene mesh is expensive, we propose 3 different sampling algorithms to represent the scene mesh as compactly and accurately as possible. Our first method samples the mesh randomly and constructs spheres around the sample points. Then, we calculate the distance of the human joints to the closest sphere at that current frame. Our second method evenly samples the mesh and constructs spheres on the samples for a more global approach. Finally, our third method represents the scene mesh with voxels. It divides the entire scene into several little cubes and stores the SDF and gradient information at each cube. Leveraging DNO as our foundation allows us to preserve contextual motion, distinguishing our approach from traditional scene-aware motion synthesis techniques.

3. Method

3.1. Signed Distance Function (SDF)

In order to achieve the goal of avoiding obstacles in a scene, we need to explore a way that can decide whether the digital human penetrates the scene. Here, we exploit the idea of SDF that maps a point in 3D to its corresponding distance to the nearest surface,

$$\mathrm{SDF}(x) \coloneqq s : x \in \mathbb{R}^3,\ s \in \mathbb{R} \tag{1}$$

The SDF value is positive if the 3D point is out of the object and vice versa. If we penalize the model when the SDF function gives a negative value, we are forcing the person to avoid coinciding with the object. However, our real-world scene mesh is based on 3D scans. The mesh is not closed, and there is no clear definition of inside or outside the object. The following three methods explore different ways that try to tackle this problem.

3.2. 1st Method - Random Sample Sphere

DNO paper demonstrates that it can avoid sphere obstacles on the ground while basically keeping the original semantics by optimizing the diffusion latent space [11].

One naive idea is to represent the scene with spheres and leverage this property, i.e., a point cloud representation. However, a scene can have millions of vertices, and it is computationally infeasible to consider every vertex as a sphere to calculate the SDF value in each optimization step. We decided to compromise and took 1000 points from all vertices of the mesh randomly.

Fig. 3a shows the visualization of the points taken from the scene mesh. For each joint, we look for a sample point closest to the joint and treat this point as a sphere so that we can apply the SDF formula for spheres. Here, the SDF value is the negative distance to the surface if a person's joint is inside the sphere and 0 if the joint is outside. To achieve zero contact with the mesh, the distance between the joint and the sphere's center must be at least equal to the pre-set sphere's radius. Therefore, we add the difference between this distance and the sphere's radius to the loss function and optimize it iteratively.

$$\mathcal{L}_{\text{mesh}}(\mathbf{z}) \coloneqq \sum_{(j)\in J} -\min\left[\|\hat{\mathbf{x}}_j(\mathbf{z}) - \mathbf{p}_{min}\|_2 - \text{rad}, 0\right] \tag{2}$$

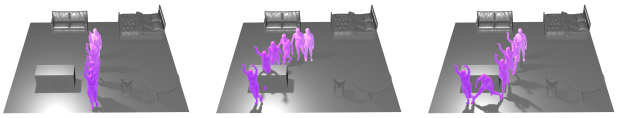
where J is the set of all the joints in the sequence, $\hat{\mathbf{x}}_j(\mathbf{z})$ denotes the position of the joint predicted by our model, $\mathbf{p}_{min}$ denotes the position of the closest sample point, and rad is a predefined hyper-parameter, deciding the radius of spheres.

Formula 3 illustrates the final total loss function employed during the optimization process. The term $\mathcal{L}_{\text{pose}}$ is utilized to ensure that the motion data adheres to a specific trajectory or pose by minimizing the average distance between the generated joint positions and the corresponding target positions. Additionally, the term $\mathcal{L}_{\text{mesh}}$ is designed to prevent collisions with obstacles within the real scene mesh.

$$\mathcal{L}(\cdot) = \mathcal{L}_{\text{pose}}(\mathbf{z}) + \lambda_{\text{mesh}}\mathcal{L}_{\text{mesh}}(\mathbf{z}) \tag{3}$$

where $\mathcal{L}_{\text{pose}}$ follows the convention of DNO

$$\mathcal{L}_{\text{pose}}(\mathbf{z}) \coloneqq \frac{1}{|O|}\sum_{(j)\in O} \|\hat{\mathbf{x}}_j(\mathbf{z}) - \mathbf{x}_j\|_2 \tag{4}$$



(a) MDM (b) DNO (c) Ours

Figure 1. "Walking forward with hands raised above head"

O is the set of joints that we cared about the specific positions, $\mathbf{x}_j$ denotes the target position of a joint, $\hat{\mathbf{x}}_j(\mathbf{z})$ denotes the position of the joint predicted by our model.

Fig. 1 presents the qualitative results obtained using the first method. Specifically, Fig. 1a illustrates the original motion generated by the human motion diffusion model. Without any optimization, the individual simply performs the motion, such as raising their hand and walking forward simultaneously. 1b displays the result of the original DNO. By specifying the start and end points, the individual follows the provided coordinates while executing the original motion generated by the human motion diffusion model. Fig. 1c shows the outcome of our first method under the same conditions. It is evident that the individual selects an alternative path from the start point to the end point, successfully avoiding obstacles while retaining the original motion to a relatively high degree.

3.3. 2nd Method - Even Sample Spheres

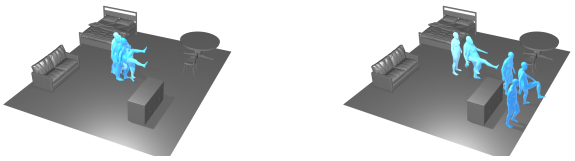
The evaluation results of our initial approach did not meet our expectations, prompting us to explore a second method. In our initial sampling strategy, vertices were randomly sampled across the entire mesh, which often resulted in some vertices being positioned too closely to each other, thereby diminishing the informativeness of the samples. To address this, we revised our sampling strategy to achieve a more uniform distribution of the 1000 sampled vertices by rejecting samples that were in close proximity to one another.

Another key distinction of our second method is the consideration of all sampled points, rather than focusing solely on the nearest vertex. For each joint, we treated all sampled vertices as spheres, calculating the distance between each joint and the center of each sphere, minus the sphere's radius. These distances were then aggregated in the loss function to enhance the robustness of the optimization process.

$$\mathcal{L}_{\text{mesh}}(\mathbf{z}) \coloneqq \frac{1}{|J|} \sum_{(j)\in J} - \sum_{k=1}^{1000} \min\left[\|\hat{\mathbf{x}}_j(\mathbf{z}) - \mathbf{p}_k\|_2 - \text{rad}, 0\right] \tag{5}$$

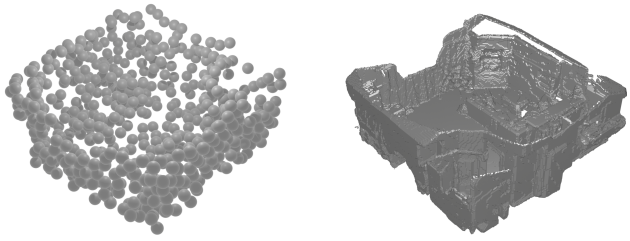
where $\mathbf{p}_k$ denotes the position of one sample point among 1000 instead of the closest one.

Fig. 2 presents the qualitative results obtained using the second method. Specifically, Fig. 2a illustrates the original motion generated by the human motion diffusion model. In the absence of any optimization, the individual performs a kick near the starting point. Conversely, Fig. 2b depicts the outcome of applying our second method under identical conditions. Notably, even when the input prompt does not explicitly include the phrase "walk forward," the individual traverses from the start point to the destination while maintaining the kicking motion, upon specifying the start and end points.



(a) MDM (b) Ours

Figure 2. "Kicking"



(a) point cloud (b) voxel

Figure 3. Reconstruction of the scene with different methods

3.4. 3rd Method - Voxel Scene

In the illustration provided in Fig. 3a, it is evident that using large spheres to represent the entire scene leads to a significant loss of detail, making it difficult to recall specific elements from the reconstructed scene. The quantity of sphere samples is also a critical hyper-parameter. If there are too few sample points, an individual might find their way through the gaps between the spheres. However, determining the ideal number of sample points depends heavily on the specific scene and cannot be decided based solely on the number of vertices in a scene. While a more complex scene requires a greater number of vertices for definition, increasing the number of sample points will only lead to overlapping spheres and wasting computational resources. Thus, it is essential to find a method that allows for a more faithful representation of the scene.

One possible strategy involves using smaller spheres for all the vertices to represent the scene. However, it is still computationally impossible to calculate the relative distance to each vertex in every optimization step. To address this, we can adapt the concept from Stochastic Gradient Descent. Instead of sampling once in the beginning, we can sample a different partial set of small spheres in each optimization step and compute the SDF loss based on these spheres. This allows us to progressively gather information from all the vertices. Nonetheless, it seems that this approach does not yield the desired results. One potential

explanation is that there is significant variation in the gradient between the two spheres, which fails to provide a comprehensive view of the scene. Consequently, we are seeking a method that minimizes distortion to the original scene while maintaining a smooth gradient for the loss. This is where voxel representation comes into play, and the scene reconstruction can be observed in Fig. 3b.

To construct this voxel representation, we start by enclosing the entire scene within a cubic space. We then partition the space into multiple smaller cubes, or "voxels," each representing a $3\,\text{cm} \times 3\,\text{cm} \times 3\,\text{cm}$ section. A SDF value is computed based on the distance from the center of each voxel to the nearest surface of the scene and is assigned to the voxel. This SDF follows the convention of having a positive value when the point is outside the object and a negative value when inside the object. Furthermore, a normalized vector pointing in the quickest direction to decrease the distance function is also assigned to each voxel. In essence, for each voxel, we establish a function that maps the center of the voxel to its SDF value and normalized gradient

$$f(\tilde{x}) \coloneqq (s(\tilde{x}), g(\tilde{x})) : \tilde{x} \in \mathbb{N}^3,\ s(\tilde{x}) \in \mathbb{R},\ g(\tilde{x}) \in [-1, 1]^3 \tag{6}$$

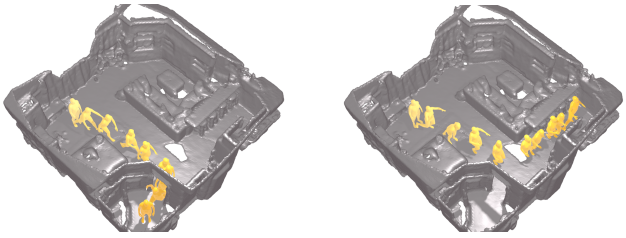
where $\tilde{x}$ denotes the position of the center of a voxel, $s(\tilde{x})$ is its SDF value, and $g(\tilde{x})$ is its normalized gradient.

The next step involves utilizing the SDF value and the normalized gradient to prevent the person from intersecting with obstacles. For each joint of the person, we determine the corresponding voxel the joint resides and extract its SDF value and normalized gradient. A joint is classified as safe if its SDF value exceeds a specific threshold, denoted as τ. If the SDF value falls below the threshold, we proportionally adjust the normalized gradient based on the deviation from the threshold, with a greater impact on joints closer to the surface. Subsequently, we compute the average of the adjusted gradients for all joints and incorporate the (x, y) positional gradient to ensure consistent editing on the motion sequence without altering the motion semantics.

$$\mathcal{L}_{\text{mesh}}(\mathbf{z}) \coloneqq \frac{-1}{\tau |J|} \sum_{(j) \in J} \hat{\mathbf{x}}_j(\mathbf{z}) \frac{1}{N_j} \sum_{k=1}^{N_j} \max\left[\tau - s(\tilde{x}_k), 0\right] \cdot g(\tilde{x}_k) \tag{7}$$

where τ denotes the minimum distance threshold, J denotes the set of all the joints in the sequence, $\hat{\mathbf{x}}_j(\mathbf{z})$ denotes the (x, y) position of the joint, N_j is the number of joints in a single frame, and $\tilde{x}_k$ denotes the position of the center of the voxel for joint $\hat{x}_k(z)$.

Fig. 4 shows the qualitative results obtained from the third Method. Fig. 4a illustrates the results achieved by providing the starting point, destination, and a text prompt "A person is walking forward." The destination is a confined area surrounded by walls with a narrow door only wide enough for the person to pass through. Despite these limitations, the person managed to reach the target without colliding with the walls. Fig. 4b showcases the outcome of using the prompt "A person is jumping forward." In this scenario, the destination is much further from the starting point. Nevertheless, the model successfully identified a path while maintaining the original "jump" semantics.



(a) "Walking forward" (b) "Jumping forward"

Figure 4. Results in a more complicated and clustered scene

4. Experiments

Datasets. For our experiments, we used the HumanML3D dataset which consists of various types of motion such as daily activities (e.g., 'walking', 'raising hands', 'jumping', 'dancing') and sports (e.g., 'swimming'). This dataset has a total of 14,616 motion sequences and 44,970 descriptions [6].

Scenes. We evaluate using two different scenes. The first scene is ScanNet [5] Scene 0000 and it is chosen because it offers a variety of obstacles and paths to travel to. The scene's intricate layout mirrors real-world scenarios, making it an ideal choice for analysis and evaluation. The second scene is the test scene from DIMOS [32], chosen because it still has obstacles but the mesh is smoother and represents ideal conditions. We made the ScanNet scene watertight using point-cloud-utils [28] package.

Motion Diffusion Model. We used the diffusion model provided by DNO [11], which is the Motion Diffusion Model (MDM) [20] retrained using Exponential Moving Averaging [9]. We made no modifications to this model.

Paths. One current limitation of DNO [11] that we couldn't figure out how to solve is that the motion has to start from (0, 0) and move in the positive (x, y) direction. Therefore, we translated and rotated both scenes so that (0, 0) was near the edge of the scene, with most of the scene extending into positive (x, y) coordinates.

We chose 12 diverse paths that required moving in different directions and avoiding obstacles. All started at (0, 0). The longest straight-line distance was about 6 meters, and the average straight-line distance was about 4 meters.

SDF. We use three different methods to create a SDF. For our sampling methods, 1st and 2nd methods, we use the sample_surface and sample_surface_even functions in trimesh [21] respectively. For our 3rd voxel method, we use the mesh_to_voxels method in the mesh-to-sdf library [18]. Voxel size is 3cm x 3cm x 3cm and the resolution is 244 x

244 x 244 for the scene from ScanNet or 235 x 235 x 235 from DIMOS.

Implementation Details. We use most default settings as listed in the DNO paper, such as Adam optimizer with a learning rate of 0.05. We optimize the diffusion noise for 1000 steps, but only use 10 unroll steps as opposed to 100 in original DNO code to reduce runtime [11]. We did not notice any loss of motion quality as a result of this change. The optimization takes approximately 20 minutes on one Nvidia GTX 1080 Ti GPU.

We use the same implementation as Karunratanakul et al. in Guided Motion Diffusion [12] for text prompts, using classifier-free guidance and the CLIP model [19] for text encoding.

4.1. Evaluation Metrics

We focus on three main aspects: (1) realistic motion: is the output motion realistic? (2) accurate navigation: does the generated motion accurately reach the goal while avoiding scene penetrations? and (3) semantic accuracy: does the generated motion reach the desired semantic behavior given in the text prompt? We hypothesize that metrics (1) and (3) are both slightly in conflict with metric (2), since more realistic motion and more accurately preserving semantics should limit the degrees of freedom of motion and could make it more challenging to avoid obstacles and reach the goal.

For all metrics, we evaluated a total of twelve different paths across two different scenes. All paths except one require avoiding at least one obstacle. For each path, we used four different prompts corresponding to different semantic motions and generated three results. This is a total of 144 different results for each of our implementations.

Realistic motion. To measure the realism of the generated output, we measure the Foot Skating Ratio following prior work [11] [12]. This metric serves as an indicator of the incoherence between the trajectory and human motion. It quantifies the proportion of frames in which a foot slides more than a specified distance (2.5 cm) while maintaining ground contact (foot height $<$ 5 cm).

Results. Our results are shown in Table 1. For all motions, our method maintains comparable Foot Skating ratio with the original DNO [11] paper.

Successful Navigation. To evaluate successful navigation, we used the same metrics defined in DIMOS [32], specifically the percentage of paths that successfully reached the goal within a small radius, and the overall percentage of joint intersections with the scene across all frames. We use joint intersections instead of vertex intersections of the SMPL model [17] because there are over 10,000 vertices, which made the code infeasible to run.

Table 1. Results for realistic motion and successful navigation, compared against the DNO [11] baseline and the DIMOS [32] baseline separately. All experiments were run with 1000 optimization steps, with 12 different paths, 2 different scenes, and a total of 144 results per metric. ("*" denotes the best result among our methods)

	Reached Goal % ↑	Joint Intersection % ↓	Foot skating ratio ↓
DIMOS (baseline)	**1.0**	**.0060**	-
DNO (baseline)	-	-	**0.074**
Random Sample	0.736	.0100*	0.098
Even Sample	0.937	0.0101	0.087*
Voxels	0.986*	.0120	0.100

Results. Our results are shown in Table 1. Our best method achieved comparable results to DIMOS [32] in successfully reaching the target, although our best joint intersection percentage (.010) was 66% worse than DIMOS [32]. However, we consider these results promising. Since our method is diffusion-based, we are able to have more motion diversity both in terms of path taken and semantic motions (discussed below), at the cost of more joint intersections.

If we calculate these evaluation metrics only for our generated "walking" motion, the intersection percentage drops by up to 17% for a best score of 0.0083. This seems to indicate that our algorithm may be performing better for "walking", as it's a simpler motion with more training data, than other motions.

DIMOS [32] uses offline path planning by creating a navigation mesh, and then calculating intermediate target waypoints between the start and the goal. One interesting follow-up would be to use this same approach in our method when optimizing the diffusion noise, to see if it provides a stronger guidance signal and reduces collisions. We have implemented this waypoints guidance in our code, but unfortunately we did not have time to do a separate evaluation for it.

Semantic Accuracy. Our diffusion model comes from MDM [20], and a big strength of this model is the ability to generate a variety of realistic human motions. We control the motion via a text prompt, and we wish to measure if this motion is accurately generated. We use the same motions defined in DNO [11], specifically "jumping", "crawling", and "walking with raised hands", chosen because it is easy to define relative joint positions that correspond to these actions. For example, "jumping" is defined as having both feet more than 5 cm off the ground. "Crawling" is defined as head below one meter, which is a lower position than what was defined in DNO [11].

Karunratanakul et al. measured motion editing and re-

Table 2. Results for semantic accuracy matching a text prompt, compared against the DNO [11] baseline. All experiments were run with 1000 optimization steps, with 12 different paths, 2 different scenes, and 36 results per motion. "Walking" was also used as a motion. ("*" denotes the best result among our methods)

	Crawling F1 ↑	Jumping F1 ↑	Raised Hands F1 ↑
N=1			
DNO (baseline)	**1.0**	**0.48**	0.96
Random Sample	0.96	0.44	0.88
Even Sample	**1.0**	0.45*	**0.97**
Voxels	0.61	0.44	0.14
N=10			
DNO (baseline)	**1.0**	**0.39**	0.78
Random Sample	0.99	0.34	0.72
Even Sample	**1.0**	0.35*	**0.79**
Voxels	0.47	0.33	0.10
N=30			
DNO (baseline)	**0.78**	0.0	**0.56**
Random Sample	0.40	0.03	0.33
Even Sample	0.50*	0.0	0.33*
Voxels	0.19	0.0	0.05
N=60			
DNO (baseline)	**0.62**	0.0	**0.20**
Random Sample	0.24	0.0	0.0
Even Sample	0.36*	0.0	0.11*
Voxels	0.15	0.0	0.0

ported the percentage of frames with the same motion [11]. Since we are generating novel motions, we instead measure if that motion was present in at least N frames. We initially only used N=1 frame and reported this in our final project presentation. However, we realized that this made our results look misleadingly good, and now report for several values of N. We report in this way instead of an average frame count as some results may preserve the content very well and some not at all, and we want to understand this.

We also realized that because we condition our model on the start and end frame joint positions, the generated motion sometimes begins in the correct motion and quickly loses semantics. Therefore, we separately evaluate all but the first and last 10 frames of all videos.

Results. Our results are shown in Tables 2 and 3. For all motions, our best method, Even Sample, maintains comparable Semantic Accuracy with the original DNO [11] paper. Our method generally does worse than DNO at preserving Semantic Accuracy for a large number of frames, which may be because it's more challenging to preserve the same motion for a long time in a complex scene.

Table 3. Results for semantic accuracy matching a text prompt, with the first and last ten frames excluded, compared against the DNO [11] baseline. Results only shown for our Even Sampling method, which achieved the best results. The default parameters we choose for our method are marked with *.

	Crawling F1 ↑	Jumping F1 ↑	Raised Hands F1 ↑
N=1			
DNO (baseline)	**1.0**	**0.50**	0.82
Even Sample	0.96	**0.50**	**0.83**
N=10			
DNO (baseline)	**0.78**	**0.43**	0.64
Even Sample	0.53	0.40	**0.71**
N=30			
DNO (baseline)	**0.62**	0.0	**0.36**
Even Sample	0.44	0.0	0.20
N=60			
DNO (baseline)	**0.53**	0.0	**0.11**
Even Sample	0.33	0.0	**0.11**

Evaluating all frames except the first and last 10 frames reduced F1 scores for both methods, as can be seen in Table 3, but not excessively. For some percentage of samples, our model is indeed preserving the semantic motion throughout the videos, and not just focusing on the beginning and end frames where the joint positions are explicitly defined.

We include jumping for completeness, though it's important to note that the jumping motion should not be seen in every frame, otherwise this would be unrealistic levitation. A reasonable baseline is around 10 frames, so we only highlight best results for N=1 and N=10.

Although we did not have time during our project, we believe a good next step would be to add intermediate targets for joint positions, and see if that helps the model preserve motion semantics through the entire generated motion. However, if the positions are defined too precisely, this could lead to unnatural motion and/or a loss of motion diversity.

4.2. Ablation Studies

We conducted several experiments on the Even Sample method to give a justification for our design choices made for this method. We experimented on the weight of the collision loss (2) on our total loss function (3), number of optimization steps and the sphere radius for sampled points. We evaluated each selected hyperparameter individually by keeping others on their default values. The default values we used are 5.0 for collision weight, 1000 steps for optimization and 0.20 for sphere radius, which perform

Table 4. Ablation study on the evenly spaced spheres method. Metrics are calculated with 12 different paths, 2 different scenes, 4 different motions.

	Reached Goal % ↑	Joint Intersection % ↓	Foot skating ratio ↓
Collision loss weight:			
– 0.0	1.0	0.0124	0.075
– 0.5	1.0	0.0120	0.086
– 5.0*	0.937	0.0101	0.087
– 50.0	0.879	0.0150	0.101
Optimization steps:			
– 300	0.715	0.0117	0.080
– 500	0.861	0.0124	0.083
– 1000*	0.937	0.0101	0.087
Sphere radius:			
– 0.05	1.0	0.0114	0.083
– 0.20*	0.937	0.0101	0.087
– 0.40	0.694	0.0067	0.070

Table 5. Ablation study regarding the semantic accuracy metric across different motions on the evenly spaced spheres method. Metrics are calculated with 12 different paths and 2 different scenes. We calculated the metrics by discarding the first and last 10 frames and measured if the motion is preserved for more than 10 frames.

	Crawling F1 ↑	Jumping F1 ↑	Raised hands F1 ↑
Collision loss weight:			
– 0.0	0.780	0.427	0.642
– 0.5	0.615	0.457	0.500
– 5.0*	0.530	0.400	0.710
– 50.0	0.435	0.465	0.604
Optimization steps:			
– 300	0.630	0.474	0.691
– 500	0.326	0.510	0.759
– 1000*	0.530	0.400	0.710
Sphere radius:			
– 0.05	0.691	0.489	0.642
– 0.20*	0.530	0.400	0.710
– 0.40	0.364	0.420	0.531

optimally through the various metrics we used, according to our experiments.

Collision Loss Weight. In Equation (3), it can be seen that our total loss consists of the loss from the motion trajectory and pose, and our mesh loss added together. To justify the effect of the mesh loss on our model, we changed the weight of the mesh loss to see its effect on the metrics. We tried weights 0, 0.5, 5, and 50. In Table 4, it can be seen that as the weight of the collision loss increases, the percentage of reaching the goal decreases and foot skating metric increases, both of which are not desired. This is expected because on the total loss function, as the weight of the mesh loss increases, pose loss gets less important and the quality of motion decreases. Our chosen collision weight of 5.0 strikes a good balance of avoiding intersections but still successfully reaching the goal.

Sphere Radius. To represent the scene mesh accurately by sampling, ideally we would need a vast number of spheres with infinitely small radius. However, due to computational constraints, this is not possible. Therefore, we approximate the scene with bigger spheres. Using 1000 spheres, we tried different spheres with radius 5 cm, 20 cm, and 40 cm, to see which one results in better performance metrics. As the radius of the spheres increase, the percentage of reaching the target decreases as big spheres limit the freedom of movement. However, we also see that too small of a radius leads to increased joint intersections, presumably because it allows the motion to get too close to the scene boundary, leading to accidental intersections. A radius of 20 cm balances the metrics nicely.

Semantic Accuracy. Ablation results on semantic accuracy in Table 5 are mixed, and it is difficult to draw conclusions. This may be because the semantic accuracy metric cannot be fine-tuned for all possible motions. For example, for the motion "jumping", we would need to check less than 10 frames because it is a short motion. In contrast, for crawling, all frames of the motion should be checked. This may be a reason why we are getting non-informative numbers for the ablation study. One observation is that "crawling" and "raised hands" seem to have an inverse relationship. Increasing the collision loss weight generally decreases semantic accuracy, as the quality of motion degrades, but it also generally decreases joint intersection, which is ideal. Increasing the number of optimization steps does not necessarily seem to increase semantic accuracy, but it does help improve the motion by reaching the goal and reducing intersections. We would like to gather more results and investigate in further detail as future work.

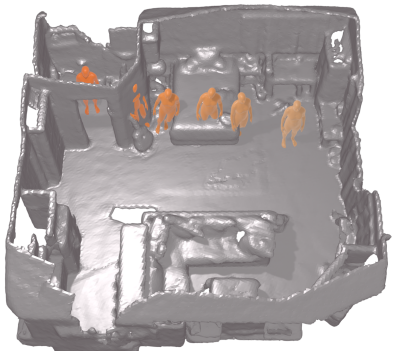
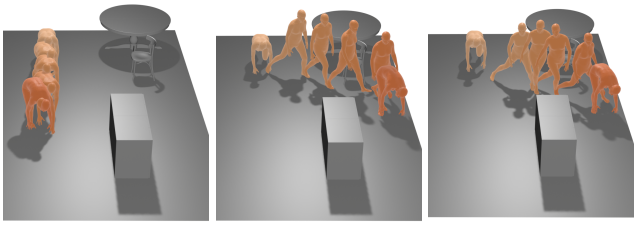


Figure 5. “A person is walking forward” (failure case)



(a) MDM (b) DNO (c) Ours

Figure 6. “A person is crawling forward”

4.3. Failure Cases

While our model is generally able to navigate through the environment successfully, there are also some known failure cases. Since our model reuses the pose loss from DNO, any failure cases of DNO are also present on our model. DNO sometimes fails to preserve the semantics as shown in Fig. 6. When given the input prompt “crawling forward,” the DNO replaces most of the crawling semantics with walking after optimization. Another failure case concerns the sphere-based sampling methods. Since we try to represent points with spheres that may have additional volume compared to obstacles, our model may fail to navigate through narrow spaces as shown in Fig. 5. Here, the model does not see the door opening as an obstacle free path. Therefore the model opts for the shortest path to the target position instead, since it cannot find any obstacle-free pathway. In these scenarios, the Voxels method works better than our sphere-based sampling methods, which can be seen on Fig. 4a.

5. Conclusion

In this work, we demonstrate the ability to generate human motion interacting with real-world scenes by optimizing the latent space of the diffusion model. We expand on the sphere obstacles in the DNO paper by incorporating more complex, uneven obstacles through an additional SDF loss for the scene mesh in the loss function. Importantly, by not explicitly modifying the relationship of joints generated by the diffusion model, we are able to largely preserve the original motion semantics while reaching very challenging destinations.

The existing diffusion model has its limitations. It can only replicate movements it has been trained on. For instance, if the destination is behind the person, the model will make him step back rather than turn around. It’s crucial to ensure that the destination aligns with the person’s facing direction. The underlying DNO also cannot preserve semantics perfectly. As shown in Fig. 6b, the DNO losses the semantics of crawling while optimizing the latent space. Moreover, the diffusion model has a restricted capability to generate long clips, making it difficult to navigate in large environments. Furthermore, our SDF is a discrete function and cannot accurately represent intricate details of the original scene. For example, the wall is very thin and our model may ignore the wall and pass through it directly if the weight for $\mathcal{L}_{\text{mesh}}$ is not large enough. If a more accurate definition is given for the inside and outside of scene to make the scene closed, and a neural network is trained accordingly, the person’s performance should improve. It is anticipated that an enhanced diffusion model and a more detailed SDF will yield improved results and facilitate the accomplishment of more complex tasks.

6. Contributions of team members

The methods mentioned in the work is developed by different team members. Random Sample method belongs to Bingjie Xue, Even Sample method is developed by Öykü Irmak Hatipoğlu and Kai Zhang developed the Voxels method. Ross Roessler is responsible for the evaluation code and the general software engineering work of getting things running. Some sections of this report also belongs to certain team members. Introduction is written by Bingjie Xue, Related Work belongs to Öykü Irmak Hatipoğlu, Method is written by Bingjie Xue and Kai Zhang, Experiments part is written by Ross Roessler and Öykü Irmak Hatipoğlu, and conclusion part belongs to Kai Zhang. All remaining parts of the project are divided equally among all team members.

7. Code

Our code, with a README with full instructions for running and reproducing our results, can be found here: https://github.com/rroessler1/digital_humans_team12

Please note that it’s a private repo as the DNO code is not public yet, and currently the only TA with access is Siwei Zhang. Please reach out to her or to a project 12 team member for access.

References

[1] Pablo Cervantes, Yusuke Sekikawa, Ikuro Sato, and Koichi Shinoda. Implicit neural representations for variable length human motion generation, 2022. 1

[2] Wenheng Chen, He Wang, Yi Yuan, Tianjia Shao, and Kun Zhou. Dynamic future net: Diversified human motion generation, 2020. 1

[3] Xin Chen, Biao Jiang, Wen Liu, Zilong Huang, Bin Fu, Tao Chen, Jingyi Yu, and Gang Yu. Executing your commands via motion diffusion in latent space, 2023. 1

[4] Rishabh Dabral, Muhammad Hamza Mughal, Vladislav Golyanik, and Christian Theobalt. Mofusion: A framework for denoising-diffusion-based motion synthesis, 2023. 1

[5] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017. 4

[6] Chuan Guo, Shihao Zou, Xinxin Zuo, Sen Wang, Wei Ji, Xingyu Li, and Li Cheng. Generating diverse and natural 3d human motions from text. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5142–5151, 2022. 1, 4

[7] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20. ACM, Oct. 2020. 1

[8] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Transactions on Graphics*, 39(4), Aug. 2020. 1

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. 4

[10] Siyuan Huang, Zan Wang, Puhao Li, Baoxiong Jia, Tengyu Liu, Yixin Zhu, Wei Liang, and Song-Chun Zhu. Diffusion-based generation, optimization, and planning in 3d scenes, 2023. 1, 2

[11] Korrawe Karunratanakul, Konpat Preechakul, Emre Aksan, Thabo Beeler, Supasorn Suwajanakorn, and Siyu Tang. Optimizing diffusion noise can serve as universal motion priors, 2024. 1, 2, 4, 5, 6

[12] Korrawe Karunratanakul, Konpat Preechakul, Supasorn Suwajanakorn, and Siyu Tang. Guided motion diffusion for controllable human motion synthesis, 2023. 1, 2, 5

[13] Manuel Kaufmann, Emre Aksan, Jie Song, Fabrizio Pece, Remo Ziegler, and Otmar Hilliges. Convolutional autoencoders for human motion infilling. In *2020 International Conference on 3D Vision (3DV)*. IEEE, Nov. 2020. 1

[14] Nhat Le, Thang Pham, Tuong Do, Erman Tjiputra, Quang D. Tran, and Anh Nguyen. Music-driven group choreography, 2023. 1

[15] Taeryung Lee, Gyeongsik Moon, and Kyoung Mu Lee. Multiact: Long-term 3d human motion generation from multiple action labels. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(1):1231–1239, Jun. 2023. 1

[16] Junfan Lin, Jianlong Chang, Lingbo Liu, Guanbin Li, Liang Lin, Qi Tian, and Chang Wen Chen. Being comes from not-being: Open-vocabulary text-to-motion generation with wordless training, 2023. 1

[17] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Smpl: a skinned multi-person linear model. *ACM Trans. Graph.*, 34(6), oct 2015. 5

[18] PyPI. mesh-to-sdf 0.0.15. https://pypi.org/project/mesh-to-sdf/, 2024. 4

[19] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 1, 5

[20] Guy Tevet, Sigal Raab, Brian Gordon, Yonatan Shafir, Daniel Cohen-Or, and Amit H. Bermano. Human motion diffusion model, 2022. 1, 4, 5

[21] trimesh. trimesh.sample - trimesh 4.4.1 documentation_2022. https://trimesh.org/trimesh.sample.html, 2022. 4

[22] Jonathan Tseng, Rodrigo Castellon, and C. Karen Liu. Edge: Editable dance generation from music, 2022. 1

[23] Jingbo Wang, Yu Rong, Jingyuan Liu, Sijie Yan, Dahua Lin, and Bo Dai. Towards diverse and natural scene-aware 3d human motion synthesis, 2022. 1, 2

[24] Jiashun Wang, Huazhe Xu, Jingwei Xu, Sifei Liu, and Xiaolong Wang. Synthesizing long-term 3d human motion and interaction in 3d scenes, 2021. 1, 2

[25] Jingbo Wang, Sijie Yan, Bo Dai, and Dahua LIn. Scene-aware generative network for human motion synthesis, 2021. 1, 2

[26] Zan Wang, Yixin Chen, Baoxiong Jia, Puhao Li, Jinlu Zhang, Jingze Zhang, Tengyu Liu, Yixin Zhu, Wei Liang, and Siyuan Huang. Move as you say, interact as you can: Language-guided human motion generation with scene affordance, 2024. 2

[27] Zan Wang, Yixin Chen, Tengyu Liu, Yixin Zhu, Wei Liang, and Siyuan Huang. Humanise: Language-conditioned human motion generation in 3d scenes, 2022. 1, 2

[28] Francis Williams. Point cloud utils. https://www.github.com/fwilliams/point-cloud-utils, 2022. 4

[29] Yiming Xie, Varun Jampani, Lei Zhong, Deqing Sun, and Huaizu Jiang. Omnicontrol: Control any joint at any time for human motion generation, 2024. 1

[30] Mingyuan Zhang, Zhongang Cai, Liang Pan, Fangzhou Hong, Xinying Guo, Lei Yang, and Ziwei Liu. Motiondiffuse: Text-driven human motion generation with diffusion model, 2022. 1

[31] Yan Zhang and Siyu Tang. The wanderings of odysseus in 3d scenes, 2022. 2

[32] Kaifeng Zhao, Yan Zhang, Shaofei Wang, Thabo Beeler, and Siyu Tang. Synthesizing diverse human motions in 3d indoor scenes, 2023. 1, 2, 4, 5

[33] Wentao Zhu, Xiaoxuan Ma, Dongwoo Ro, Hai Ci, Jinlu Zhang, Jiaxin Shi, Feng Gao, Qi Tian, and Yizhou Wang. Human motion generation: A survey, 2023. 1