SimpledCLIP: Prompt-based Human 3D Pose Generation using CLIP

Team Cloud-CVLab: Farouk Boukil, Albin Vernhes, Juliette Parchet
Supervisor: Benoît Guillard
Computer Vision Laboratory (CVLab), EPFL
December 22^{nd}, 2022

***Abstract*—In this paper, we introduce SimpledCLIP, a composite model for recovering human 3D poses from text using CLIP. We provide an overview of the models we use in building SimpledCLIP, and how we use it to try to solve the task at hand. We build from the ground up, starting with the most basic settings. We also shed light on what hinders the performance of our model.**
We would like to thank our supervisor, Benoît Guillard, for orienting and advising us throughout this project.

I. Introduction

Training extremely large and deep neural networks for supervised learning tasks often requires tremendous amounts of adequate quality labeled data on top of huge computational resources. To mitigate both issues, we explore the possibility of exploiting state-of-the-art 2D pre-trained such gigantic neural networks to solve 3D tasks via transfer learning.

In this paper, we focus on the particular task of recovering human 3D poses from prompts, that is the retrieval of 3D human body parameters that are described in text. We do so by maximizing the proximity in a joint embedding space of the prompt and a 2D image of the body. We use SMPL as a low-dimension parameterization of 3D human bodies, a differentiable renderer to get 2D images of a 3D body mesh, and CLIP to embed text prompts and images in a joint space. Since all components are differentiable, we can use gradient descent to directly optimize the body parameters.

Although SimpledCLIP does not end up living up to our expectations compared to other related works such as Text2Mesh [1] since it is not yet able to recover 3D poses, we believe that our work may pave the way for further research in this area.

II. Models

A. SMPL: Skinned Multi-Person Linear model [2]

SMPL models the human body in 3D. It represents the body as a set of joints linked by vertices and wrapped in a closed volume of connected triangular faces that materialize the body's morphology. To create a mesh using SMPL, one needs to specify the pose parameter $\theta \in \mathbb{R}^{72}$ that controls the angles of the joints and the shape parameter $\beta \in \mathbb{R}^{10}$ that controls the enclosed volume and the vertices' lengths. SMPL is data-driven: it constructs meshes from scans of real people, in contrast with a handcrafted approach. We use a PyTorch implementation of SMPL that allows automatic differentiation through it with respect to these parameters (θ, β).

B. DR: Differentiable Renderer[3]

A renderer takes a snapshot of a 3D scene in the form of a 2D image, just like a camera does in real life. In a nutshell, it uses a shading algorithm to figure out the effect of light sources on the colors of the objects in the scene depending on their material, followed by a rasterization algorithm to compute the color of each image pixel after projecting the 3D scene on the plane of the camera. We use a PyTorch3D implementation of this renderer so as to automatically differentiate through it.

C. CLIP[4]: Contrastive Learning Image Pretraining model

CLIP is a state-of-the-art model that learns a joint euclidean space for both 2D images and texts. The embedding of a text is close to that of an image as long as the former describes the content of the latter.

We rely on CLIP as our 2D component to recover 3D poses due to its astonishing zero-shot performance on a variety of image recognition tasks on ImageNet: it still provides close embeddings for text/image inputs unseen at training time. This property is particularly interesting for transfer learning since the renderings of our 3D scenes and the prompts of interest that describe the human 3D pose may be unseen at CLIP's training.

Since CLIP is a combination of two neural networks implemented in PyTorch, we can differentiate through them. The version we use operates on 244x244 images.

Adapting CLIP's Input: CLIP was originally designed to take PIL images, not image tensors, which hinders the backpropagation through the composition of the three models (SMPL, Renderer, CLIP). To avoid that issue, we reimplemented CLIP's image preprocessing step so that it directly operates on image tensors through which we can automatically differentiate.

D. SimpledCLIP

We call SimpledCLIP the pipeline of the three models, corresponding to the design in figure 1 below. SimpledCLIP takes a prompt describing the pose of interest and the constituents of a 3D scene. The scene, which contains the mesh, is then rendered as a 2D image which is fed together with the prompt to CLIP. The latter outputs the prompt and image embeddings which we denote throughout this paper by $u \in \mathbb{R}^{512}$ and $v \in \mathbb{R}^{512}$ respectively.

SimpledCLIP has many hyperparameters, regrouped under the notation Ω, mainly consisting of:

- Mesh Hyperparameters: texture, gender.
- Scene: lights, materials, blending.
- Rendering Camera(s): type, position, the field of view, frustum ...etc.
- Rendering Shader: Phong, Gouraud, Flat, or Silhouette.

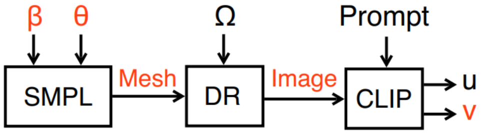


Fig. 1. SimpledCLIP's design.

III. Recovering The Prompted 3D Pose

A. Method

Given the strong motivation to rely on CLIP and what is expected from it, namely that the cosine similarity between the prompt embedding u and the rendered scene's embedding v is large if the prompt describes the pose in the image among other decor details, recovering the 3D pose prompted to SimpledCLIP amounts to solving the following optimization problem:

$$\theta^* = argmin_\theta L(u^*, v(\theta; \Omega, \beta^*))$$

CLIP is pretrained and is used as such, so its weights are not optimized with respect to.

The prompt embedding u can be regarded as a hyperparameter since it solely depends on the prompt which is fixed during optimization, and so we denote it u^*.

Although in theory, we could optimize with respect to both θ and β, we noticed that $|\beta|_F$ did not vary much during optimization unlike $|\theta|_F$, aside from not causing any significant improvement during optimization. As we aim to recover 3D poses, our interest shifted to optimizing with respect to θ only for a fixed β^* portraying a reasonable human silhouette.

We optimize using SGD-based algorithms, where we start with an all-zero θ that is updated every full pass (forward pass into backward pass through the entire model).

B. The Optimization Objective L

In our experiments, we use the following optimization objectives:

- Cosine Distance, $L_{CD}(v, u^*) = 1 - \frac{<v, u^*>}{|v|_F |u^*|_F}$, which is by CLIP's definition assumed to be smaller the more the image and prompt are semantically identical as explained in section II-C.
- Inner Product Loss, $L_{IP}(v, u^*) = \frac{\lambda}{2}|v|_F - < u^*, v >$ for $\lambda > 0$: any positively co-linear with u^* non-zero vector of finite norm v maximizes the 2 vectors' cosine similarity. Basic calculus shows that $\nabla_v L_{IP} = 0 \leftrightarrow v^* = \frac{u^*}{\lambda}$ and $\nabla_v^2 L_{IP} > 0$ since $\lambda > 0$.

  L_{IP} is a smooth and strictly convex loss in terms of v in contrast with L_{CD} at the cost of fixing $|v^*|_F$ for an "optimal" v^*.

IV. Prompt Construction Scheme

As of today, there is no generic way to build effective prompts for CLIP, or at least none that we are aware of. A few tricks shared by the community that have been empirically "proven" to work better include describing the scene of the image, not just the details we aim to recognize within that image. In light of that, we build our prompts as a concatenation of two descriptions:

- A scene description that describes the mesh and the background, such as: *"This is a brown person wearing a grey T-shirt and dark pants. The background is white"*. Indeed, that is the scene description we use in all of our experiments using a human texture.
- A description of the pose to be recovered in 3D as θ^* through optimization.

We start by assessing how well CLIP can distinguish different prompts following the above construction with different basic pose descriptions. Figure 2 shows the cosine similarity between a few publicly available prompts.

The first three prompts describe the same pose in different orderings of specifications, the reason why their cosine similarity is higher than for other prompts, as it should be. The remaining prompts describe different poses or no pose at all. We extrapolate that CLIP is slightly invariant to the ordering of pose specifications and that it distinguishes relatively well different such descriptions under this prompt construction scheme.

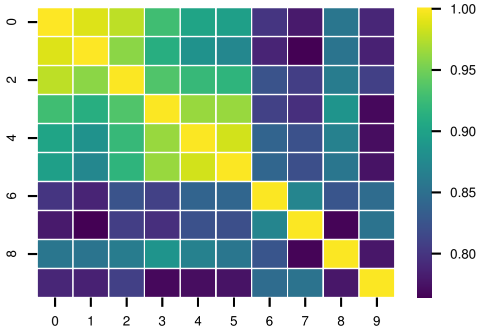


Fig. 2. The cosine similarity between different prompts describing the same scene but different poses.

V. SimpledCLIP's Performance: Single Camera Rendering

In this section, the model uses a single camera positioned on a centric 3D sphere, with a field of view centered at the origin, to render one image per pass. The required pose is given by the following pose prompt: *"She is looking upfront and is standing up straight with both feet and hands flat. Both her arms are straight above her head."*. Figure 3 depicts the evolution of the loss in both cases where the mesh has a grey texture or a human texture. Figure 4 shows the best poses of both meshes, loss-wise, which are also highlighted in Figure 3. The visualization parameters are identical in both cases. In particular, the meshes are seen with null azimuthal rotation.

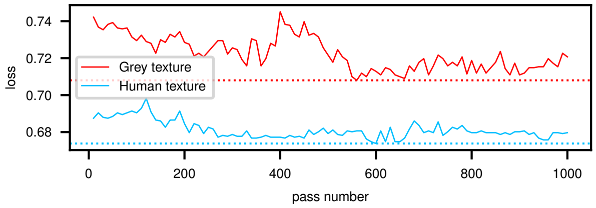


Fig. 3. L_{CD} loss plot for two meshes with different textures but in the same settings. The minimum loss is highlighted for each mesh.

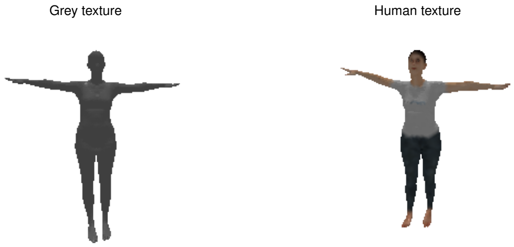


Fig. 4. The minimal loss poses with grey-textured or human-textured meshes.

In most of our experiments with a grey-textured mesh, it has had large variations of the loss during optimization and the longer we optimized the more the mesh tended to take inhuman poses, not close to the required pose. Moreover, the minimal loss pose does not correspond to the required one. On the other hand, even if the human mesh does not correspond to the required pose either, it is closer to it and the loss is smaller overall. Note a few unwanted joint rotations, such as the ankles. This behavior generalizes to other prompts in non-degenerative cases, so we deduce that adding a texture improves the performance.

VI. SimpledCLIP's Performance: Multi-Camera Rendering

In this section, SimpledCLIP uses five cameras positioned on a centric 3D sphere, on a circle parallel to the equator plane of that sphere, with a field of view that is centered at the origin as previously, to render five images per pass. The reason we use more cameras is on one hand to capture full information about the 3D pose of the mesh, and on the other hand, as an effort to compensate for bad minima that may arise when fitting a 3D pose via 2D information only.

A. Adaptation of the Optimization Objectives

Having n renders per pass results in as many image embeddings, each corresponding to a different view angle. We denote those embeddings by the set $V = \{v_i, 1 \leq i \leq n\}$. Inspired by Text2Mesh [1], we define two loss aggregations to account for the information within each image embedding in V during the optimization:

- Loss on Average Embedding (LoAE):
$$L(u^*, V) = L(u^*, \tfrac{1}{n}\textstyle\sum_{i=1}^{n} v_i)$$
- Average Loss on Embeddings (ALoE):
$$L(u^*, V) = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} L(u^*, v_i)$$

B. Results

Following up with the same prompt from section V, figure 5 depicts the evolution of L_{IP} and L_{CD} respectively under the ALoE and LoAE loss aggregations. Just like previously, figure 6 shows the resulting minimal loss poses. The visualization parameters are identical in the pose quad plot. In particular, the azimuthal rotation is 45°.

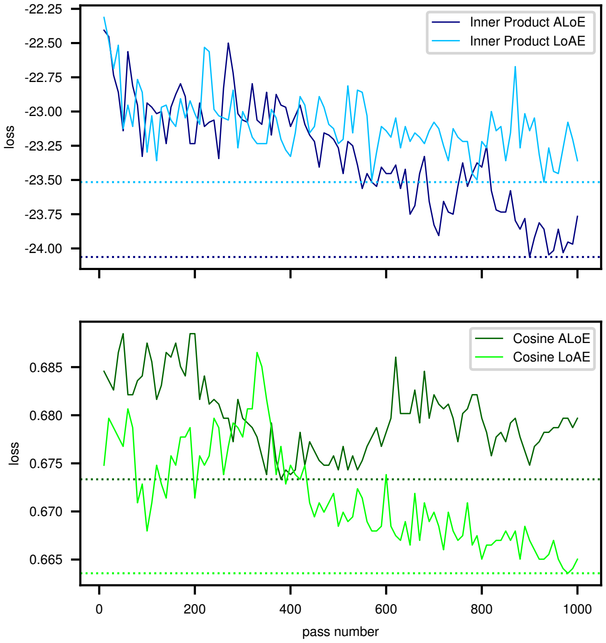


Fig. 5. L_{IP} and L_{CD} ALoE/LoAE respectively with five cameras rendering a human-textured mesh.

Depending on the optimization objective, ALoE or LoAE may perform better loss-wise. Although there is no significant improvement loss-wise for L_{CD} compared to the result of the previous section, the minimal loss pose of L_{CD} LoAE gets a little closer to the required pose. L_{IP} ALoE has a not-so-bad minimal loss pose either. In both cases, the unnecessary joint rotations persist.

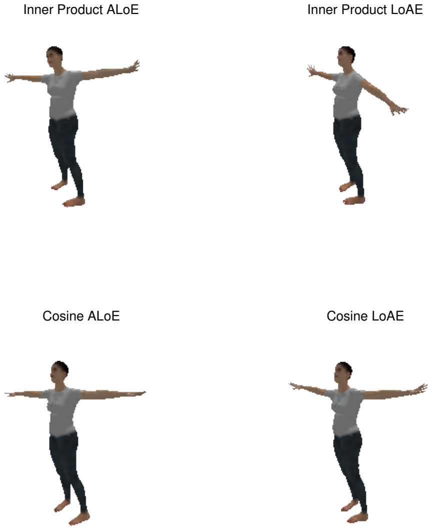


Fig. 6. L_{CD} and L_{IP} LoAE/ALoE minimal loss pose plots, with five cameras rendering a human-textured mesh.

VII. SimpledCLIP's Performance: Partial Optimization on Multi-Camera Rendering

L_{CD} diminishes scarcely (order of 10^{-1} at most) in our previous experiments. Thus, we assist SimpledCLIP by freezing, in a correct pose, the joints that are independent of the required movement with respect to the initial pose. Following up with the same prompt as previously, the frozen joints are at least those of the lower half of the body. The results are shown in figures 7 and VII. The visualization parameters are as usual identical in the quad plot. The azimuth angle is 55°.

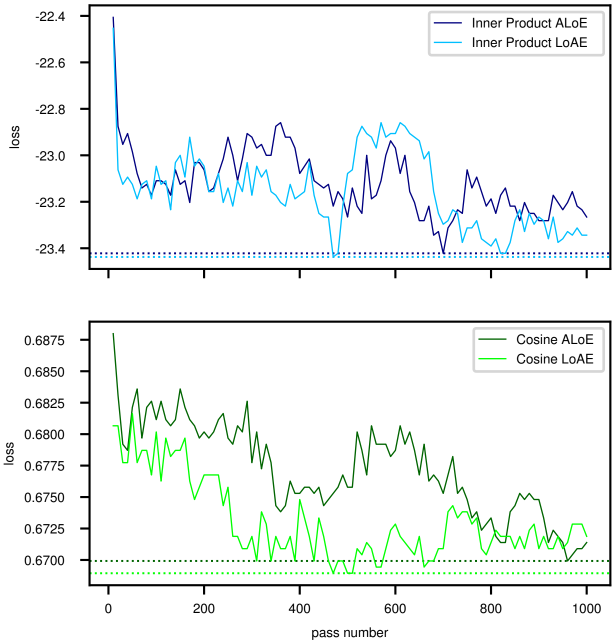


Fig. 7. L_{CD} and L_{IP} LoAE/ALoE minimal loss pose plots, with five cameras rendering a human-textured mesh, when the lower body is frozen.

The losses diminish faster in the first few passes compared to previously, however, surprisingly, the minimal loss is almost always as large if not larger for both losses with either aggregation. Moreover, the minimal loss poses are still far from the required pose. We deduce that SimpledCLIP may rely on unnecessary joint rotations that minimize the loss further, which does not contribute to recovering the required pose.

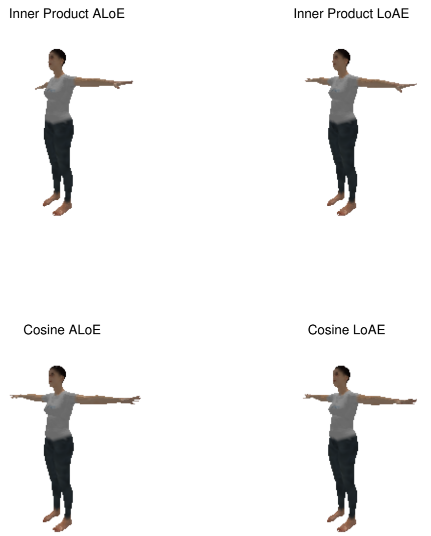


Fig. 8. L_{CD} and L_{IP} LoAE/ALoE minimal loss pose plots when the lower half of the body is frozen.

VIII. Why is SimpledCLIP not able to recover the pose?

A. The composition of SMPL and DR does not fit a target image

In this section, we look at the composition of SMPL with the differentiable renderer (DR) only.

We aim to determine if the pose on a given reference image is recoverable. We choose a $\tilde{\theta}$ encoding a simple plausible human pose and compute its corresponding render $\tilde{w}(\tilde{\theta}; \Omega, \beta^*)$. Similarly, we denote the render of an arbitrary θ as $w(\theta; \Omega, \beta^*)$. Consequently, we have the following optimization problem, where we choose the simplest loss which is the euclidean distance:

$$\bar{\theta} = argmin_{\theta}||w - \tilde{w}||_F^2$$

Figure 9 below is an example result we get by gradient descent optimization, using the resulting θ of the previous pass in the following:

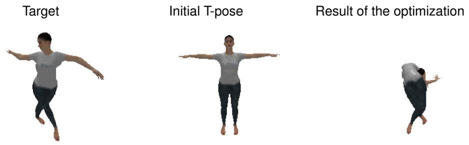
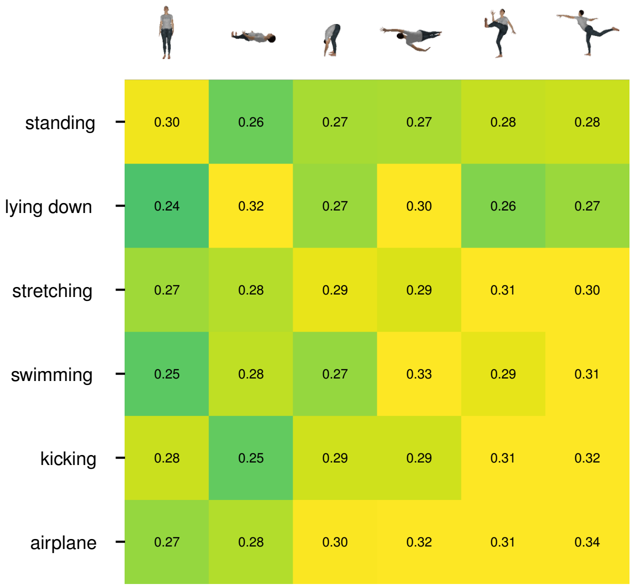


Fig. 9. Trying to fit a target image starting with an initial pose. The result is very far from the target.

As can be seen above, the composition of SMPL and DR is not able to fit the target image, at least not with the euclidean distance as an optimization objective. Indeed, the mesh folds so that the render minimizes the pixel-to-pixel euclidean distance between the target and the resulting image, but it does so in a bad way that gets it stuck in a local minimum. We have noticed similar results with other target poses. Consequently, SimpledCLIP may not be able to recover the required pose.

B. CLIP does not recognize well enough different poses of our human-textured mesh

Figure 10 shows that CLIP does not recognize well enough our human-textured mesh in different poses, since the cosine similarity for matching textual pose and 3D pose render (2D image) which are along the diagonal are very low, given that the cosine similarity is a confidence score that the text matches the content of the image. Occasionally, these values may even be larger for incorrect textual pose and render pairs, as we can see for the pose ”kicking” with the render corresponding to ”airplane”. This may be the biggest hindrance to SimpledCLIP so far.



standing	0.30	0.26	0.27	0.27	0.28	0.28
lying down	0.24	0.32	0.27	0.30	0.26	0.27
stretching	0.27	0.28	0.29	0.29	0.31	0.30
swimming	0.25	0.28	0.27	0.33	0.29	0.31
kicking	0.28	0.25	0.29	0.29	0.31	0.32
airplane	0.27	0.28	0.30	0.32	0.31	0.34

Fig. 10. The cosine similarity of different 3D poses and their respective pose descriptions.

IX. Summary

All in all, SimpledCLIP as we have developed and studied it so far is not successful at retrieving prompted human 3D poses. A few ways to improve SimpledCLIP include adding at least a ”ground” to help CLIP capture the orientation of the body, as well as constraining SMPL’s pose parameters to the humanly plausible ones using for instance Adversarial Paramteric Pose Prior[5].

X. Reproducibility

We provide the following publicly available file that details our model version choices and hyperparameters Ω, in addition to optimization methods and parameters, to help reproduce our results.

References

[1] O. Michel, R. Bar-On, R. Liu, S. Benaim, and R. Hanocka, “Text2mesh: Text-driven neural stylization for meshes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

[2] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, “Smpl: A skinned multi-person linear model,” *ACM transactions on graphics (TOG)*, vol. 34, no. 6, pp. 1–16, 2015.

[3] S. Liu, T. Li, W. Chen, and H. Li, “Soft rasterizer: A differentiable renderer for image-based 3d reasoning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7708–7717, 2019.

[4] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, "Learning transferable visual models from natural language supervision," in *International Conference on Machine Learning*, pp. 8748–8763, PMLR, 2021.

[5] A. Davydov, A. Remizova, V. Constantin, S. Honari, M. Salzmann, and P. Fua, "Adversarial parametric pose prior," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10997–11005, 2022.