Appendix

A. Implementation details

A.1. Pre-trained Generative Models

Full-body pose prior. For our full-body generative model $\mathcal{P}$, we use the GitHub implementation of VPoser [33]. VPoser is a variational autoencoder [73] trained on data obtained by applying MoSh [13] on three publicly available human motion capture datasets: CMU [6], training set of Human3.6M [74], and the PosePrior dataset [15].

Hand object grasping model. For the right-hand grasping model $\mathcal{G}$, we use the GitHub implementation of GrabNet [19]. GrabNet consists of two networks: **1) CoarseNet** for coarse grasps and **2) RefineNet** for refining the coarse grasps. CoarseNet takes as input a latent vector w, and the object $\mathcal{O}$, represented by its BPS representation [75], and generates a hand pose, including its translation and rotation, to be used as input to MANO. RefineNet takes as input the CoarseNet grasp and the distances D from the coarse MANO vertices to the object mesh. It then refines the grasp through 3 iterations as in [76] to give the final grasp. RefineNet has been trained by sampling CoarseNet grasps as ground truth and perturbing the hand pose parameters to simulate noisy input estimates.

Object representation. The object to-be-grasped $\mathcal{O}$ is represented in GrabNet using the Basis Point Set (BPS) [75] representation which is capable of encoding arbitrary 3D object shapes. Given any object vertices, an approaching angle α and N_b fixed basis points, the BPS representation involves rotating the object by the angle α, placing it in the center of the points and calculating the minimum distance from each point to the nearest surface of the object. The outputs of our model are rotated by the inverse of the rotation matrix given by α. We use the implementation from the bps_torch library on GitHub.

A.2. Training Details

- For every example (scene and object), we optimize $N = 500$ latent vectors z, with different initializations, and at the end of the optimization process we select the ones that result in the smallest loss. During training, we periodically discard the 50% of the latent vectors that produce the largest losses. At the end of the optimization process, we end up with the best 16 samples out of the 500. The parameters of the mapping network (consisting of a 2-layer MLP) are shared across the N latent vectors.
- We constrain the value of w by normalizing it such that its norm is always one, following the density of a high-dimensional Gaussian prior, thus making sure w is within the distribution of the latent space of $\mathcal{G}$.
- We train with Adam optimizer with a learning rate of $1e-3$ for z, and $1e-4$ for the mapping network. Additionally, we found that the translation parameters have a much stronger gradient than the rest, especially the latent v, so we divide the gradient that goes through t_b^{xy} by 3, and multiply the gradient that flows through v by 10. We train for 500 iterations.
- Empirically, we found that the pre-trained GrabNet model was much more sensitive to approaching angle α then it was to the latent w, hence we set w to the zero vector in our experiments.
- For the hand matching loss, we weigh the vertices around the wrist more ($\times 3$) than the rest, as the alignment around the wrist is less noisy than in the fingers.
- The values for the loss weights λ in the total loss are set to: $\lambda_{\text{hm}} = 20$, $\lambda_{\text{o}} = 1000$, $\lambda_{\text{g}} = 0.01$. These do not necessarily reflect the importance given to each loss, as the loss values are in completely different scales.
- We additionally found that scaling the output of the MLP differently for every parameter was helpful. Specifically, we scale v by 5 (giving more flexibility to the human pose generator), the translation parameters by 10, and the angle and orientation by 20.
- We found that some obstacles have very thin walls, so we make the obstacle mesh $\mathcal{M}_{\text{obstacle}}$ thicker by 5mm, which allows us to model the intersections better.

B. ReplicaGrasp Dataset

Receptacles. We use a total of 48 receptacles from the ReplicaCAD dataset [34]. Some of the static rigid object receptacles include: apartment chair, sofa, table top, TV stand and wall cabinet. The receptacles from articulated objects include: refrigerator top, middle and bottom; top, middle and bottom drawers of kitchen counter on both right and left sides, and kitchen sink; as well as top, middle and bottom compartments of both sides of the kitchen cupboard. Many of the receptacles are visible in Fig. 1.

Objects. We obtain 50 everyday object meshes from the GRAB dataset and use the Habitat Simulator [35] to get the final locations of the objects on the receptacles. We use the GitHub v0.2.2 release. The simulator runs dynamics for 5 seconds to check for stability of newly placed objects.

C. Quantitative Analysis

We conduct a detailed analysis of the results in Table 1.

Performance as a function of object height. We demonstrate the need of having a benchmark like ReplicaGrasp

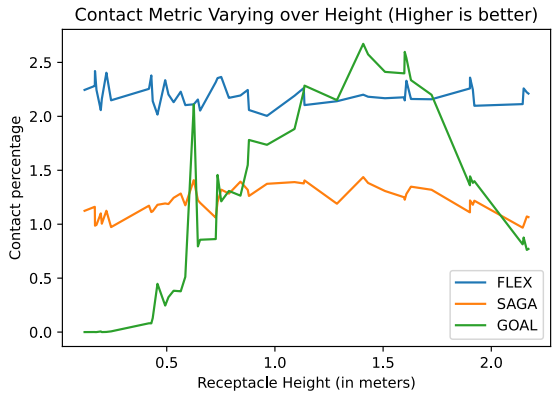


Figure 11. **Object contact percentage varying by height.** FLEX performs more consistently than both baselines and is best on average.

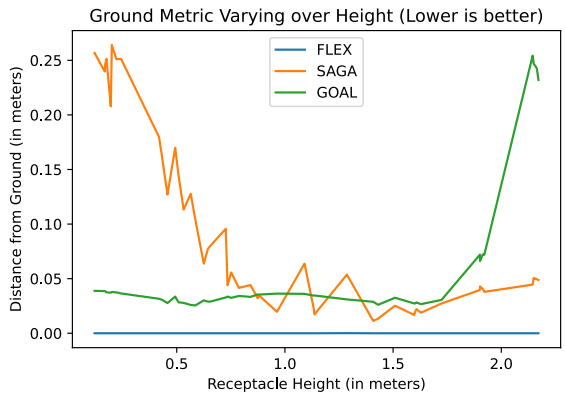


Figure 12. **Ground distance varying by height.** SAGA performs worst at lower heights while GOAL performs worst when the objects are high up.

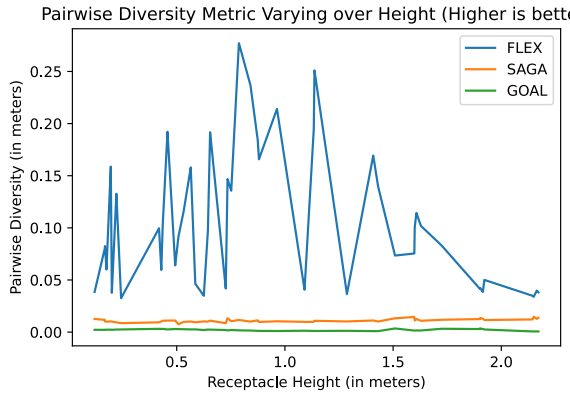


Figure 13. **Diversity varying by height.** GOAL and SAGA both consistently fail at generating diverse outputs at all heights. FLEX can generate most diverse grasps at medium heights.

that allows evaluating grasps at different heights. In GRAB, the object heights varies from a minimum of 0.75 meters to a maximum of 1.38 meters, with a mean of 1 meter. In ReplicaGrasp, our object heights have a much larger range

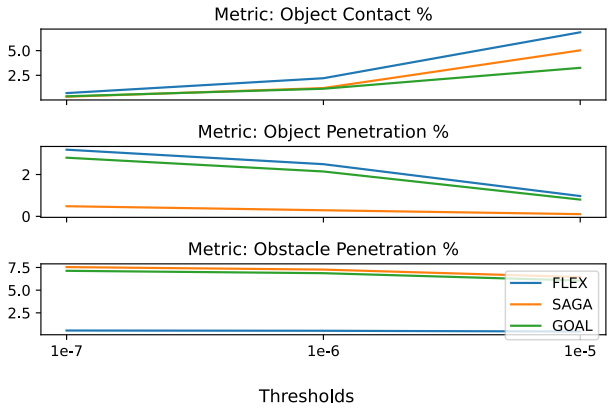


Figure 14. **Sensitivity to threshold σ.** Trends stay the same for all metrics across different thresholds.

from a minimum of 0.12 meters to a maximum of 2.2 meters, with a mean of 1 meter. We show how performance along different metrics changes by varying the heights of the objects.

- **Object contact percentage** - Fig. 11 shows that GOAL performs well when objects are at heights that have been seen during training, but sees drops in performance at other heights. SAGA is more consistent than GOAL even at varying heights. FLEX outperforms both baselines on average across all heights without showing much variation across height changes.
- **Ground distance** - Fig. 12 shows that when the object is at a low height, SAGA fails by generating humans with legs buried below the ground. SAGA is better at higher heights, although qualitatively the humans appear elongated. GOAL generates humans that try to fly up to grasp objects at larger heights. GOAL performs better at lower heights, although qualitatively the humans look unnatural with awkwardly bent legs.
- **Sample diversity** - In Fig. 13, we show the average pairwise diversity across pairs of samples generated for an instance, averaged across all instances of the dataset. This quantifies the method's ability to generate a range of complex human poses. FLEX outperforms both baselines by a large margin despite having additional constraints of avoiding obstacles.

Sensitivity of the metrics to the threshold. In order to compute the metrics, we set a threshold σ that determines the boundary between contact and penetration with an object or an obstacle. In Fig. 14 we report results of our metrics for different values of this threshold, and show that the metrics are not sensitive to its value, and that the trends shown in the main paper (where we use $\sigma = 1e-6$) hold.

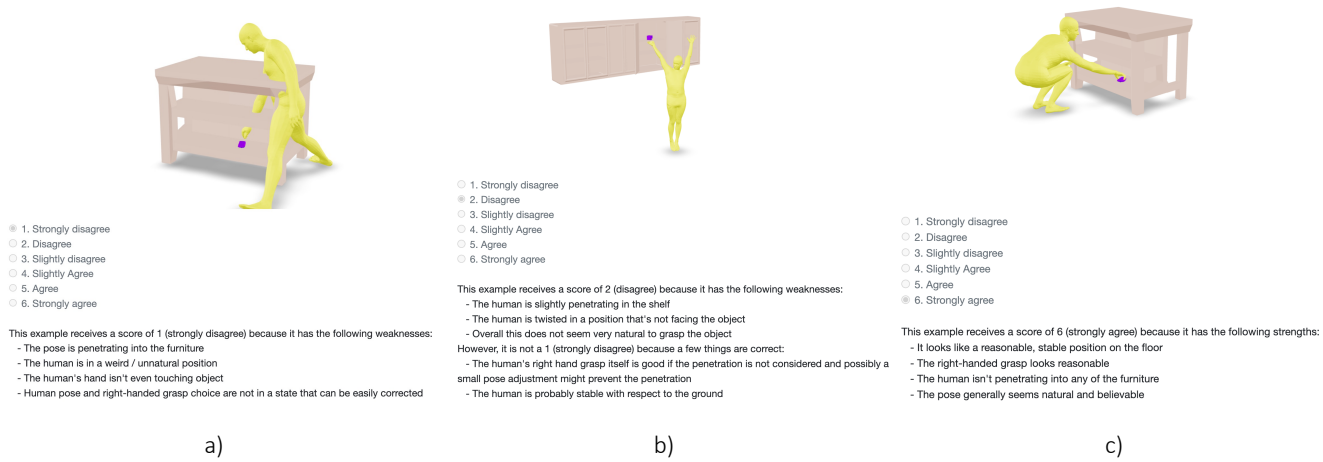


Figure 15. **Examples shown to Amazon Mechanical Turk subjects**. We provide these three examples to the subjects, which range from very bad (strongly disagree with the statement) to very good (strongly agree) results.

Figure 16. **Human studies.** Example of a HIT shown to subjects on Amazon Mechanical Turk.

D. Evaluation Metrics

We provide equations for each of the metrics described in Section 5.3.

Object contact percentage.

$$M_{\text{obj}}^{\text{contact}} = \frac{100}{|\mathcal{V}_{\text{o}}|} \sum_{i=1}^{|\mathcal{V}_{\text{o}}|} \mathbb{1}\left(|d_{\text{vm}}(\mathcal{V}_{\text{o}_i}, \mathcal{M}_{\text{human}})| \leq \sigma\right), \quad (5)$$

where $\mathcal{V}_{\text{o}}$ are the vertices of the object, $\mathcal{M}_{\text{human}}$ is the human mesh, d_{vm} is the signed vertex-to-mesh distance, σ is a small threshold and $\mathbb{1}$ is the indicator function.

Object penetration percentage.

$$M_{\text{obj}}^{\text{penet}} = \frac{100}{|\mathcal{V}_{\text{o}}|} \sum_{i=1}^{|\mathcal{V}_{\text{o}}|} \mathbb{1}\left(d_{\text{vm}}(\mathcal{V}_{\text{o}_i}, \mathcal{M}_{\text{human}}) < -\sigma\right) \quad (6)$$

Obstacle penetration percentage.

$$M_{\text{obst}}^{\text{penet}} = \frac{100}{|\mathcal{V}_{\text{b}}|} \sum_{i=1}^{|\mathcal{V}_{\text{b}}|} \mathbb{1}\left(d_{\text{vm}}(\mathcal{V}_{\text{b}_i}, \mathcal{M}_{\text{obstacle}}) < -\sigma\right), \quad (7)$$

where $\mathcal{V}_{\text{b}}$ are the vertices of the human body and $\mathcal{M}_{\text{obstacle}}$ is the obstacle mesh. In contrast to Eq. (6), here we average over the human body (not the obstacle) vertices, because we care about how much of the human is penetrating an obstacle, not how much of the obstacle is being penetrated by the human.

Ground distance.

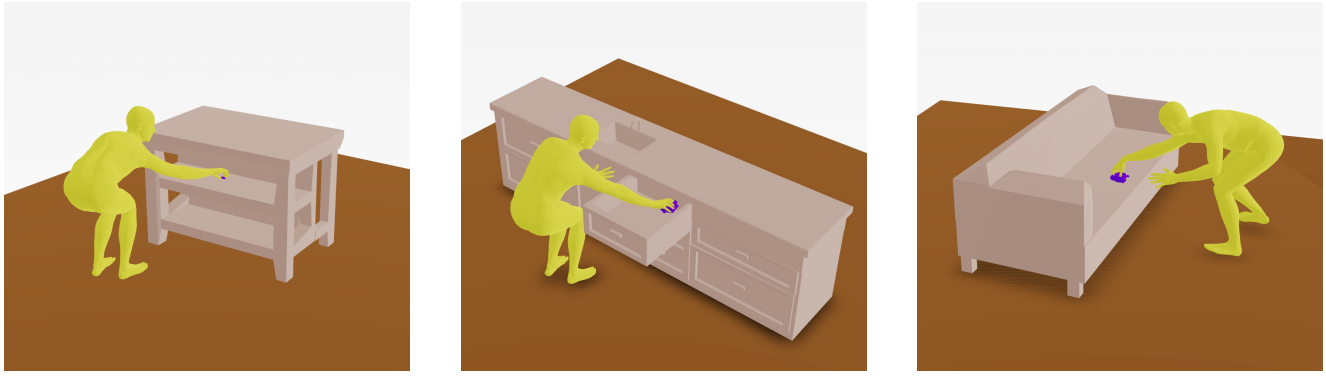
$$M_{\text{ground}} = \left|\min(\mathcal{V}_{\text{b}}^{z})\right|, \quad (8)$$

where $\mathcal{V}_{\text{b}}^{z}$ is the z component of all vertices in $\mathcal{V}_{\text{b}}$.

Sample diversity.

$$M_{\text{Div}_{\text{samp}}} = \frac{2}{\mathcal{N}_{\text{s}} \cdot (\mathcal{N}_{\text{s}} - 1)} \sum_{\substack{i,j \in \mathcal{N}_{\text{s}} \\ i \neq j}} d_{\text{vv}}(\mathcal{V}_b^{(i)}, \mathcal{V}_b^{(j)}), \quad (9)$$

where $\mathcal{N}_{\text{s}}$ is the number of samples for a single example and d_{vv} is the L^2 vertex-to-vertex distance in the 3D space. $\mathcal{V}_b^{(i)}$ represents the human body vertices corresponding to the i-th sample.



a) Strongly agree b) Agree c) Slightly disagree

Figure 17. **Examples of human ratings.** The figure shows three samples and their corresponding human ratings. See Appendix E for comments from subjects.

Overall diversity.

$$M_{\text{Div}_{\text{all}}} = \frac{2}{\mathcal{N}_{\text{d}} \cdot (\mathcal{N}_{\text{d}} - 1)} \sum_{\substack{i,j \in \mathcal{N}_{\text{d}} \\ i \neq j}} d_{\text{vv}}(\mathcal{V}_b^{(i)}, \mathcal{V}_b^{(j)}), \quad (10)$$

where $\mathcal{N}_{\text{d}}$ is the total number of instances in the dataset.

E. Human Studies

We conducted perceptual evaluation studies on Amazon Mechanical Turk (AMT) with a prompt as shown in Fig. 16. The specific instructions were as follows:

```
We want to evaluate the realism of the
    humans. Some questions to ask yourself
    while solving the task:
1) Would you expect to see a human like
    this in real-life?
2) Is the hand grasp going to result in a
    natural grasp?
3) Is the human stable on the ground?

The scene can be navigated by:
1) Clicking and dragging the mouse, to
    rotate the scene.
2) Zooming in and out with the scroll
    wheel.
3) Clicking at a point in the scene, to
    position that point in the center of
    the scene.

See examples for a better intuition.
```

Further, we showed subjects three examples of how to successfully perform the task by showing an example that deserves the ratings of 1, 2 and 6 respectively with explanations for the reasoning as shown in Figure 15. We randomly selected 96 examples of ReplicaGrasp covering objects in all 48 receptacles in both upright and fallen orientations. We showed each example to 5 different subjects and we had 30 unique subjects solve the task. We filtered out cases which saw high inter-subject disagreement.

	Sample-wise ↑		Overall ↑	
Method	Full Body	Right-hand	Full Body	Right-hand
GOAL	0.11	0.04	6.01	12.14
SAGA	1.14	0.09	15.29	13.79
FLEX (ours)	**26.91**	**0.36**	**39.98**	**16.40**

Table 3. Diversity analysis on GRAB (cm)

Fig. 17 shows some examples of FLEX generations evaluated by subjects. Participants generally found the results realistic – for example, for Fig. 17 b, a participant wrote: *"The stretch of the hand inside the drawer is very realistic"*. In some cases where the subjects gave a low rating, for instance in Fig. 17 c, we received interesting comments: *"Doesn't need to squat to grab item"*. This reveals a shortcoming of our system wherein we do not measure the effort required to grasp an object. Explicitly modeling physical effort and its effect on the choice of the human's pose is an interesting direction that we leave as future work.

F. Computational Budget

We performed speed and memory comparisons (averaged across 10 runs) for generating 16 different samples on a single RTX 2080 Ti GPU. FLEX involves using pre-trained models simultaneously, the memory consumption is 3x (4.8 GB vs 1.4 GB). FLEX takes around 8.5 minutes to generate 16 samples, while SAGA and GOAL take 6 and 1 minute respectively. We sacrifice computational budget for significantly better results.

G. Diversity Analysis

Tab. 3 shows diversity metric computed for hand (no-full-body) and for full-body (no-hand) for all 3 methods. FLEX has higher diversity in both, but the gains are significant for full-body.

Method	Obj Cont (%) ↑	Obj Penet (%) ↓	Obs Penet (%) ↓	Ground (cm) ↓
Random Init	0.15	35.06	2.02	60.32
CMA-ES	0.03	17.39	2.03	58.25
FLEX	**2.20**	**2.50**	**0.53**	**0.00**

Table 4. Comparison with Optimization methods.

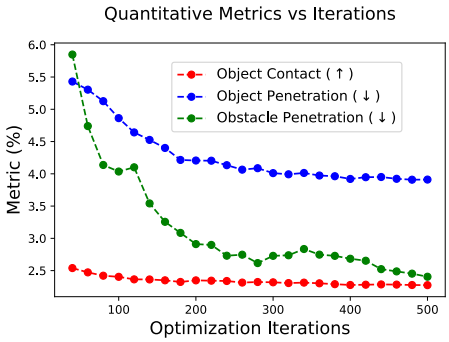


Figure 18. Object contact, penetration and Obstacle penetration metrics varying by iterations.

H. Choice of Optimization Framework

FLEX is agnostic to the choice of the optimization method. We used the recent Liu *et al.* [72] which smooths the loss landscape for better convergence. To validate this choice of a gradient-based optimization framework, we conduct experiments with non-gradient based methods described below:

- **Ranking:** Instead of optimization, we simply rank a large the batch of whole-body grasps produced by randomly sampling the optimization parameters.
- **CMA-ES:** Covariance matrix adaptation evolution strategy implemented from PyPI.

Results are shown in Tab. 4. As expected, FLEX is superior to both the baselines.

I. Performance as a function of the number of optimization steps

Fig. 18 shows average optimization metrics over different iterations. Object and obstacle penetration go down with training. Object contact stays largely unchanged.