Towards Foundation Models for 3D Vision: How Close Are We? Supplementary Materials

Yiming Zuo*, Karhan Kayan*, Maggie Wang, Kevin Jeon, Jia Deng, Thomas L. Griffiths
Princeton University
{zuoym,karhan,maggiewang,kevinjeon,jiadeng,tomg}@princeton.edu

A. Human Annotation Collection

We provide more details of the human annotation interface on MTurk, as shown in Fig. 1.

In Fig. 1a we show the interface that the workers see on MTurk. Instead of completing a multiple-choice question, the workers are asked to click near the center of the marker that they think is closer to the camera. This allows us to effectively filter out the bots/spammers on the MTurk platform. Our quality control is effective as shown in Fig. 1b. With the clicking questions, we are able to boost the human annotation accuracy from 77% to 91%, which is almost the same accuracy as what we get by annotating ourselves.

For the keypoint detection and matching task, we prompt the workers with a pair of co-visible images from the Megadepth-1500 [3, 6] dataset. Our short prompt is "Choose EXACTLY 5 points in the left image and the same 5 points in the right image." See Fig. 4 for the full instructions given to the subjects. Also, see Fig. 1c for the user interface we use to collect annotations.

Fig. 1d shows an example correspondence annotation we collected for the keypoint matching task. To ensure quality, we only keep responses where the subject followed all instructions. For instance, we eliminate responses where the subject did not label exactly 5 keypoints in each image. We also eliminate responses where the subject matched a keypoint to another keypoint in the same image or to a completely random point in the other image.

B. More Results on Keypoint Matching

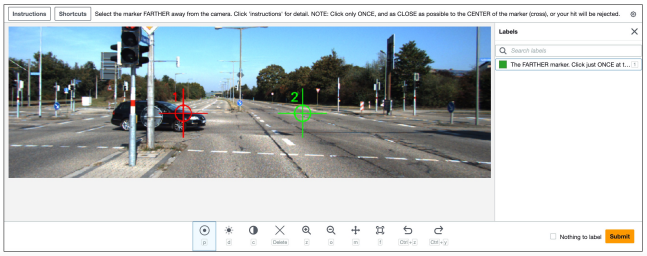
We analyze what causes human subjects to make errors in keypoint matching, which could hopefully inform benchmarks and training datasets that rely on human annotations. Fig. 2 shows that subjects were more accurate around keypoints that are detected by SIFT [4], FAST [5], and Harris Corner detectors. In other words, if a subject matched a keypoint that is close to a corner, for instance, they would be closer to the ground-truth correspondence. We further observe in Fig. 2b and Fig. 2c that the end-point error (EPE) increases logarithmically with the distance to the nearest SIFT and FAST keypoints. However, as the confidence intervals in Fig. 2 reveal, these conclusions are not as clear for very distant keypoints due to the lack of samples. Our linear regression analysis shows that around 15% of the variance in human error is explained by the distance from SIFT and FAST keypoints with $p < 0.001$. We also find that approximately 10% of the variance is explained by the distance from a corner with $p < 0.05$.

We analyze how the human annotations are affected by texture as well. We observe that human subjects tend to overwhelmingly choose textured points compared to random choice. Fig. 3b demonstrates this phenomenon. To measure how textured the patch around a pixel is, we use a combination of Gabor filters [1] with different orientations. We take the variance of these filters to measure the amount of texture around a pixel. We further observe in Fig. 3a that the subjects made less matching errors when matching textured points, meaning that human EPE was lower for textured keypoints.

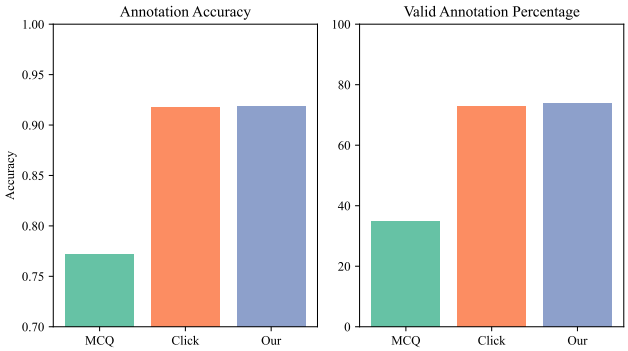
C. Experimental Setup: Camera Pose Estimation

To train the neural network models, we set up the camera pose estimation as a two-way classification problem, with the most prominent axis of movement as input. This is because the most prominent axis is given to VLMs and Humans as input and they are asked to classify the movement direction, so we mimic the same setup in order to ensure fair evaluation. Given a pair of images, we convert the ground-truth relative pose between them into the ground-truth primary move direction. Specifically, given the x,y components of the relative translation vector between the two frames $\mathbf{T} = [T_x, T_y]$, we first compute the absolute values of the components $\mathbf{A} = [|T_x|, |T_y|]$. We then identify the component with the largest magnitude by index $=$ argmax$(\mathbf{A})$, which indicates the axis along which

*These authors contributed equally (random order).

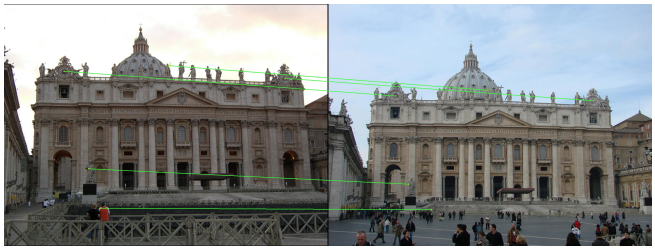


(a) Annotation Interface. The user is asked to click near the center of the maker that they think is farther away from the camera. The user can use the MTurk's built-in functionalities to zoom and move if necessary.

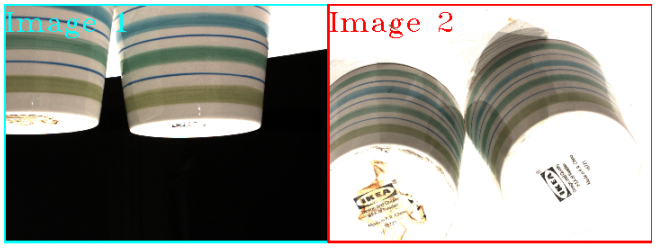


(b) Multiple-choice questions (MCQ) cannot detect the bots, resulting in only 35% valid data and 77% accuracy. In comparison, using the click-based interface boosts the valid percentage and accuracy to 73% and 91% respectively, being very close to the human upper-bound (annotate ourselves).

(c) The user interface we use to collect keypoint detection and matching annotations from human subjects.



(d) Example matches collected from human annotators.



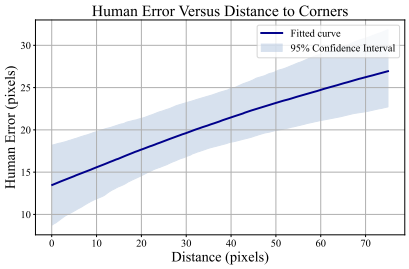
(e) Camera pose estimation user interface. This is an example of a flipped image.

Figure 1. Human annotation interface on MTurk and annotation quality control.

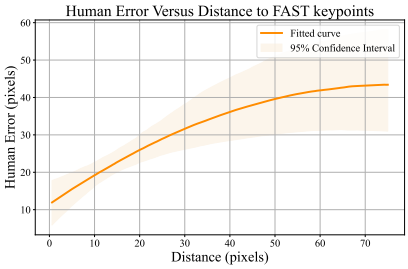
the most significant movement occurs. Based on the sign of this component, the ground truth answer is given as follows:

$$D = \begin{cases} 0 & \text{if index} = 0 \text{ and } T_x > 0 \quad \text{(+x direction)} \\ 1 & \text{if index} = 0 \text{ and } T_x < 0 \quad \text{(-x direction)} \\ 0 & \text{if index} = 1 \text{ and } T_y > 0 \quad \text{(+y direction)} \\ 1 & \text{if index} = 1 \text{ and } T_y < 0 \quad \text{(-y direction)} \end{cases}$$

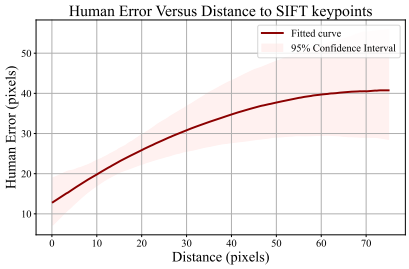
We train Resnet, ViT, and Swin Transformer backbones on this classification task. Given a pair of images, we pass each of them through the backbone and concatenate the two feature vectors. We concatenate this feature vector with the index given above which encodes the primary movement axis. We then pass the concatenated feature vector through an MLP output head two predict the primary movement direction. We use cross-entropy loss to train each network. As our training dataset, we choose the BlendedMVS [7]



(a) Human error and distance to corners.

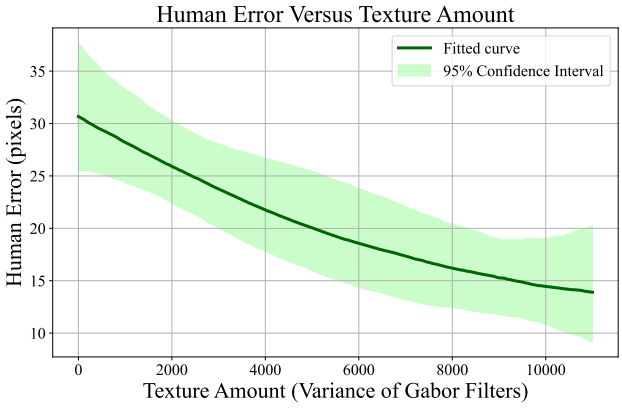


(b) Human error and distance to FAST keypoints.

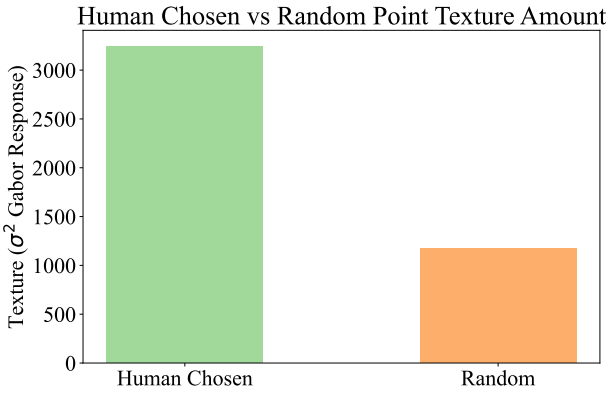


(c) Human error and distance to SIFT keypoints.

Figure 2. Matching errors humans make with respect to the ground truth correspondence. Subjects make fewer mistakes when they match points that are salient. (a) The closer a point is to a corner, the easier it is to match for humans. (b) The closer a point is to a FAST keypoint, the easier it is to match for humans. (c) The closer a point is to a SIFT keypoint, the easier it is to match for humans.



(a) Human error and the amount of texture.



(b) The average amount of texture around keypoints chosen by the subjects versus the amount of texture around a random point.

Figure 3. (a) More textured locations are easier to match for humans. (b) The subjects are more likely to choose textured keypoints

An example annotation is given above. The image below shows the matches.

Your task is to choose matching points between two images. Label EXACTLY 5 points in the left image, and label the corresponding 5 points in the right image. For instance, if you put 'Point 1 - Left Image' on the top of a tower in the left image, you should put 'Point 1 - Right Image' on the top of the same tower in the right image.

An example workflow is: choose whichever point you want in the left image (Point 1 - Left Image), choose the same point in the right image (Point 1 - Right Image). Choose another point you want in the left image (Point 2 - Left Image), choose the same point in the right image (Point 2 - Right Image). And so on...

Every label should be used ONLY ONCE.

Try to be as accurate as possible. You can zoom in and out using the toolbar and undo your annotations with Ctrl+z.

Do not label more or less than 5 points per image, 10 points per task.

Figure 4. Full instruction prompt given to human subjects for keypoint detection and matching annotations.

dataset and train each network for 15 epochs on NVIDIA RTX 3090 GPUs.

During testing, we evaluate both the networks and VLMs on a two-way classification task where the objective is to distinguish between the direction of the movement along the primary movement axis. If the primary movement is along the x-axis, we ask VLMs "Imagine you captured image 1 with your camera. To capture image 2, in what direction do you need to move your camera? A: move left, and rotate to your right; B: move right, and rotate to your left?". If the primary movement is along the y-axis, we ask VLMs "Imagine you captured image 1 with your camera. To capture image 2, in what direction do you need to move your camera? A: move down, and rotate to look up; B: move up, and rotate to look down". Note that these are the same questions asked to the human subjects. For the test dataset, we use DTU [2]. We deliberately choose the test-stage to be zero-shot for the neural networks by not fine-tuning them on DTU. The goal here is to compare humans, VLMs, and neural networks in equal conditions assuming none of the VLMs have been trained on DTU.

References

[1] Hans Feichtinger and Georg Zimmermann. Gabor Analysis and Algorithms. pages 123–170. January 1998. 1

[2] Rasmus Jensen, Anders Dahl, George Vogiatzis, Engil Tola, and Henrik Aanæs. Large scale multi-view stereopsis evaluation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 406–413. IEEE, 2014. 4

[3] Zhengqi Li and Noah Snavely. MegaDepth: Learning Single-View Depth Prediction from Internet Photos, April 2018. 1

[4] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157 vol.2, 1999. 1

[5] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European Conference on Computer Vision*, 2006. 1

[6] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-Free Local Feature Matching with Transformers, April 2021. arXiv:2104.00680 [cs]. 1

[7] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. *Computer Vision and Pattern Recognition (CVPR)*, 2020. 2