We thank the reviewers for their constructive feedback. Overall, the reviewers (especially R1 and R2) appreciated our algorithm's simplicity, its effectiveness on challenging real-world 3D human pose estimation, and our thorough experiments. R1 and R3 do point out that the technical novelty of the method is minor: indeed, most of the techniques we use to extract motion and build our dataset are familiar. R1 rightly points out that these techniques are not really the goal of the paper: instead, our goal was to investigate which pieces of the synthetic training pipeline are most important for success, and we found surprising results with a relatively simple, understandable methodology.

R3, on the other hand, mentions problems with "generality" and "viability" as significant weaknesses. On generality, R3's first concern is that the learning method is limited to video data. This concern is puzzling to us. Many algorithms are developed solely for video (e.g. self-supervised methods like 'Shuffle and Learn[1]'), and for many domains of interest in sim2real (e.g. robotics), video is readily available. But more fundamentally, our algorithm allows pseudo-labeling of individual frames, which means that a single-frame model can be learned from unlabeled data. This is indeed future research, but it's premature to say that our approach doesn't apply at all to single-frame situations.

R3's second concern is that our 2D keypoint detector has been trained from annotations on real images. We are not trying to hide this fact. We are asserting that we train the full 3D pose estimator only on synthetic humans; that is, we use no ground-truth depth for real images. We are not wishing to claim (as stated by R3) that "the method completely relies on simulated human data." We have identified a few places where our original prose may have been unclear about this, and we will rewrite them. Importantly, while our ultimate goal is to remove manual annotation from the pipeline completely, the field is currently far from this goal. Therefore, progress will require gradually reducing reliance on labeled real data, starting with the most difficult parts of the pipeline like depth. Our work is a core step on this path.

Finally, in evaluating our results, R3 almost completely ignores our contributions showing *how* to get good performance when using 2D keypoints as input, seemingly treating data generation and optical flow as the only contributions of our method. Apparently this is because the "Novelty of the method over [44] is not major", where [44] also uses keypoints. In response, we point out that our algorithm better than halves the error achieved by [44] on 3DPW. The main difference is that [44] does not use *motion* information, the importance of which is the core thesis of our work. Finally, even if focusing on flow, an improvement of 5.5 on 3DPW is hardly marginal: [6] for example, shows an improvement of 5.0 with their weak labeling method. It only appears marginal relative to the enormous loss in performance that comes from naïve use of synthetic data, and the performance recovery from keypoints.

Detailed responses: R1: FlowNet vs TVL1: We used two different algorithms only for implementation convenience. We had TVL1 features pre-computed for Kinetics, and so it was straightforward to use these to estimate camera motion. However, we didn't have TVL1 flow precomputed for the composited videos, so we did this on-the-fly, and FlowNet was easiest to implement in our Tensorflow scripts. If required, we can re-run using FlowNet for everything; we expect results to be very similar.

R1: references 59 and 69 are swapped: This is indeed an error. Thanks for pointing it out; we'll fix it in the final manuscript.

R2: Fig. 5 is hard to understand: For each test image, we show 3 displays: the original image with the detection (which all have bounding boxes), the extracted box with the inferred mesh, and the rotated mesh. In retrospect, we agree that this presentation makes it difficult to group the images. We'll improve it in the final version. However, there should also be no empty blue boxes; this suggests an issue with the reader. We can debug if you tell us which one you used.

R2: Differences between Temporal HMR [35] and Motion HMR: Architecturally, the methods are intentionally similar for comparability. However, one difference is that [35] uses a 3D convnet, while we use an LSTM to aggregate in time: this is because we expect optical flow to capture short-range information, and care more about propagating pose information long-term across frames without motion. However, between the methods, the more fundamental difference is the training data (synthetic vs. real), and we expect these differences to be more important for the overall behavior of the model.

R2: Title is general: We wrote the title thinking that our method can be readily applied to other types of 3D pose estimation whenever synthetic data is available, and would therefore be of interest to all 3D pose estimation researchers. However, we are happy to update the title as requested.

R3: Reference to Shrivastava, Ashish, et al.: Thanks for this reference; we weren't aware of this paper at submission time, but will be happy to add it.

References

[1] I. Misra, C. L. Zitnick, and M. Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *ECCV*, 2016.