Popup: Reconstructing 3D Video Using Particle Filtering to Aggregate Crowd Responses

Jean Y. Song
University of Michigan
Ann Arbor, MI
jyskwon@umich.edu

Stephan J. Lemmer
University of Michigan
Ann Arbor, MI
lemmersj@umich.edu

Michael Xieyang Liu
Carnegie Mellon University
Pittsburgh, PA
xieyangl@cs.cmu.edu

Shiyan Yan
University of Michigan
Ann Arbor, MI
shiyansi@umich.edu

Juho Kim
KAIST
Daejeon, Republic of Korea
juhokim@kaist.ac.kr

Jason J. Corso
University of Michigan
Ann Arbor, MI
jjcorso@umich.edu

Walter S. Lasecki
University of Michigan
Ann Arbor, MI
wlasecki@umich.edu

ABSTRACT

Collecting a sufficient amount of 3D training data for autonomous vehicles to handle rare, but critical, traffic events (e.g., collisions) may take decades of deployment. Abundant video data of such events from municipal traffic cameras and video sharing sites (e.g., YouTube) could provide a potential alternative, but generating realistic training data in the form of 3D video reconstructions is a challenging task beyond the current capabilities of computer vision. Crowdsourcing the annotation of necessary information could bridge this gap, but the level of accuracy required to obtain usable reconstructions makes this task nearly impossible for non-experts. In this paper, we propose a novel hybrid intelligence method that combines annotations from workers viewing different instances (video frames) of the same target (3D object), and uses particle filtering to aggregate responses. Our approach can leveraging temporal dependencies between video frames, enabling higher quality through more aggressive filtering. The proposed method results in a 33% reduction in the relative error of position estimation compared to a state-of-the-art baseline. Moreover, our method enables skipping (self-filtering) challenging annotations, reducing the total annotation time for hard-to-annotate frames by 16%. Our approach provides a generalizable means of aggregating more accurate crowd responses in settings where annotation is especially challenging or error-prone.

IUI '19, March 17–20, 2019, Marina del Ray, CA, USA

ACM ISBN 978-1-4503-6272-6/19/03…$15.00
https://doi.org/10.1145/3301275.3302305

CCS CONCEPTS

• **Information systems** → **Crowdsourcing**; • **Human-centered computing** → **Human computer interaction (HCI)**; • **Computing methodologies** → *Computer vision.*

KEYWORDS

Crowdsourcing; Human Computation; Answer Aggregation; 3D Reconstruction; Autonomous Vehicle; Particle Filter

ACM Reference Format:
Jean Y. Song, Stephan J. Lemmer, Michael Xieyang Liu, Shiyan Yan, Juho Kim, Jason J. Corso, and Walter S. Lasecki. 2019. Popup: Reconstructing 3D Video Using Particle Filtering to Aggregate Crowd Responses. In *24th International Conference on Intelligent User Interfaces (IUI '19), March 17–20, 2019, Marina del Ray, CA, USA.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3301275.3302305

1 INTRODUCTION

Autonomous vehicles collect large quantities of 3D training data by operating, or being operated, in their target environment. This data is used to teach the vehicle how to adjust to and interact with the physical world [6]. However, training suffers from a lack of realistic data, especially of rare and unusual events such as traffic accidents [19]. To collect sufficient training instances of such events, autonomous vehicles need to run and record billions of miles in the wild, corresponding to decades of operating a car on the roads.

Creating realistic *simulated* 3D scenes from abundant existing traffic accident videos crawled from those available on the Web, such as on YouTube, is a more reasonable method for creating realistic training data of rare events at scale. For example, Waymo's autonomous research vehicles [1] travel and record approximately 25,000 miles every day on public roads [2], while Americans drive a total of nearly three trillion miles every day [19], a factor of 120 million. The process of creating 3D scenes from real-world monocular video is called *3D video reconstruction.* Generally, manual

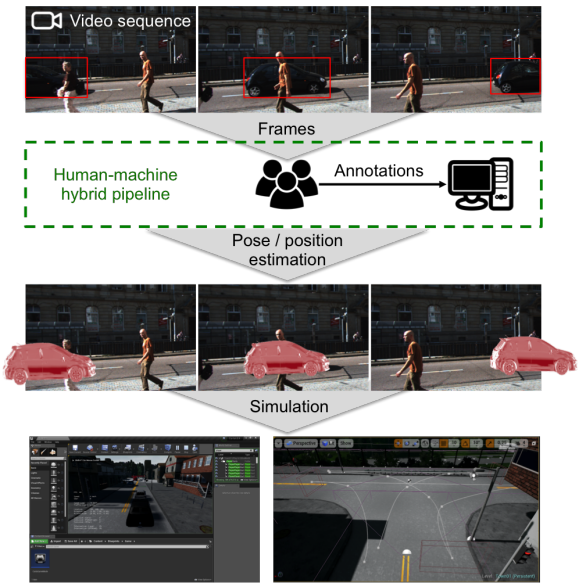


Figure 1: We propose a crowd-powered human-machine hybrid system for collecting and aggregating annotations, which leverages *content diversity* to improve accuracy of 3D state estimates of objects in 2D videos. The system uses particle filtering to aggregate annotations across different content of video frames, which enables generating simulated realistic large 3D datasets even with missing annotations.

annotations are necessary at some point of the process to bridge the sensory and semantic gap between 2D and 3D. To efficiently scale up manual annotation, one can benefit from crowd-powered tools that rapidly leverage human effort.

Even though crowdsourcing has been widely studied in image and video annotation tasks [27, 29, 32, 51], crowdsourcing techniques for 3D video reconstruction have been under explored. This is due to the high degree of difficulty of the task, where even a small error in the annotation results in a significant error when re-projected into 3D. For example, as shown in Figure 2, a three-pixel error in the 2D annotation of vehicle height can result in a 26-meter error in 3D position estimation. Therefore, quality control is a crucial component in both the answer aggregation and 3D state estimation stages to avoid such error amplification. One method known to improve data quality at the time of collection is to allow workers to selectively skip annotations that they have low confidence in the accuracy of (which we call *self-filtering*) [9, 28, 44]. Self-filtering can be particularly useful in visual annotation, as oftentimes it is nearly impossible to generate the correct annotation due to artifacts, such as motion blur, angle of view, truncation, or cropping in individual frames [51]. However, this type of filtering should be handled carefully because it may result in missing annotations, e.g., where *all* the workers self-filtered, resulting in system failure due to the 3D reconstruction problem being underdetermined (there are fewer equations than unknowns).

In this paper, we propose a novel hybrid (human-machine) intelligence pipeline for 3D video reconstruction, as in Figure 1. Our approach leverages *content diversity* from different but related video frames to increase the accuracy of 3D state estimates. We use the term 'content' to denote data shown to workers as part of the task instance (here, a frame of the video). Our method uses a particle filter based aggregation strategy to make use of the temporal dependencies of the different frames to complement dropped out answers from workers, which enables 3D reconstruction even when there are "missing" annotations. This ability allows the requester to set more aggressive filtering thresholds for removing annotations, which improves system output accuracy despite the risk of incomplete per-frame annotations. Our work proposes a generalizable solution for crowdsourcing annotations on series of related content, in which we leverage latent-valued invariants to aggregate annotations across differing content.

We introduce Popup, a crowd-powered system, that collects annotations of 3D dimension lines atop 2D videos, and then aggregates these annotations using particle filtering to generate 3D state estimates of objects of interest. We validate our method on videos from a publicly available and established dataset of traffic scenes [14]. The experimental results show that our proposed approach reduces the relative error by 33% in position estimation compared to a baseline condition which uses L-BFGS-B [55] to optimize re-projection of a 3D cuboid onto each video frame for state estimates. Further, our proposed aggregation method is robust in cases with missing annotations, where the baseline method will fail due to the problem being underdetrermined. Lastly, because Popup enables self-filtering, the annotation time for challenging frames can be reduced by 16%.

The main contributions of the paper are as follows:

- A novel crowdsourcing approach that leverages content diversity as a means of aggregating multiple annotations from different sources.
- Popup, a hybrid intelligence system that estimates 3D states of objects in 2D videos using crowdsourced dimension line annotations on objects and their actual dimension lengths.
- A novel annotation aggregation method that uses particle filtering to integrate annotations from different video frames, enabling more accurate 3D scene reconstruction even with missing annotations.
- Experimental results from 17 videos annotated by 170 Amazon Mechanical Turk crowd workers that shows the improvements in 3D state estimation accuracy, quality control, and efficiency enabled by our proposed annotation aggregation method.

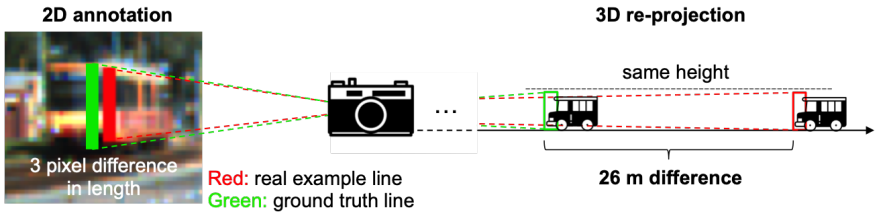


Figure 2: A small pixel error in 2D can be amplified in 3D, resulting in a severe position error. The vehicle image on the left shows a crowdsourced *height entry* dimension line annotation (in red) and the corresponding ground truth (in green). The z-dimension estimate can be calculated from the focal length and the object's actual height, which was 721 pixels and 3.59 meters in our experiment, respectively. The three-pixel difference in dimension line leads to a 26-meter difference in 3D location.

2 RELATED WORK

Crowdsourcing leverages human intelligence via an open call, e.g., to online work platforms like Amazon Mechanical Turk, to help perform tasks that are otherwise difficult for fully automated systems. The task of reconstructing 3D animations from typical RGB videos can benefit from crowd-powered annotation pipelines by using them to help automated systems bridge the semantic and sensory gap between 2D and 3D. Our work proposes novel crowdsourcing strategies for collecting and aggregating annotations for 2D to 3D scene reconstruction, which extends the literatures on crowdsourcing video annotations and improving the quality of aggregated crowd answers.

Crowdsourcing Video Annotations

Crowdsourcing techniques are widely used in visual data annotation in the 2D image-space, such as object segmentation [3, 35, 45], object detection [17, 43, 46], and visual question answering [5, 22, 25] for their efficiency and flexibility. However, the techniques used for static image annotation do not optimally extend to video annotation, as they neglect dependencies in the temporal dimension. This imposes significant additional cost on the task and prevents scaling. Our work focuses on reducing the cost of collecting annotations by improving the aggregation efficiency.

Video annotation systems for tasks like activity recognition [32, 53] or event summarization [54] provide a video stream to crowd workers and ask them to provide a summary of the clip based on the query from a requester. Other systems are designed to detect targeted events from a video stream [4, 23, 29, 41], letting crowd workers refer to the temporal context to decide the specific moment of the targeted events. Several systems have addressed local annotation tasks, such as moving object detection [11], object tracking [51], object annotation [54], control feedback [31], or object segmentation [16, 21]. Our work contributes to this line of research by introducing a novel method to aggregate confined local annotations across video frames to improve the output quality of subsequent processing steps. More specifically, we introduce a system that estimates the 3D state—position and orientation—of objects [47, 48] using novel answer elicitation and aggregation strategies.

Quality Improvement of Aggregated Crowd Answers

Answer aggregation is a challenging problem in crowdsourcing due to the fact that crowds are typically composed of people with unknown and diverse skills, abilities, technological resources, and levels of understanding of the given task. There are two primary classes of methods for improving the quality of crowdsourced annotations: methods for preventing low quality work at the time of annotation collection, and methods for compensating for low quality work post hoc, typically via aggregation [24, 30].

Post-hoc compensation for low quality work, such as majority voting on the result or weighting workers based on expectation maximization [7, 10, 18], is done after results have been submitted, usually in the aggregation stage. Powerful post-hoc techniques can complement poor quality results in crowdsourced datasets by referring to the agreement between multiple workers [34, 45]. However, because answers exist in a large continuous space, agreement may not be possible in generative tasks such as ours. We introduce a novel answer aggregation technique using particle filtering, which leverages coherency in the temporal dimension to handle noisy and even missing annotations.

Another strategy to improve the quality of aggregated answers is to prevent low quality work before it is submitted, e.g., training workers [13], screening workers [20], or applying different incentive strategies [33, 36]. Recently, skip-based annotation techniques [9, 28, 44] have been explored in the labeling domain, which allow crowd workers to self-filter their labels based on their confidence about a question. Revolt [9] introduced a collaborative crowdsourcing system that post-processes self-filtered questions and asks workers to discuss with each other the question to reach a consensus. Shah and Zhou [44] showed that incentivizing workers to

self-filter is the only incentive-compatible payment means possible. Our proposed particle-filtering-based aggregation method enables self-filtering because the missing annotation can be compensated for via temporal coherency.

3 APPROACH

To prevent the amplification of annotation errors in 3D state estimates, filtering as many low quality annotations as possible is a necessary step prior to aggregating and estimating the 3D states of objects. However, missing annotations are difficult to handle in the subsequent steps and a specialized treatment is necessary to avoid system failure. Our work is conceptually motivated by inter-frame prediction techniques in video coding [52], which takes advantage of temporal coherency between neighboring frames to predict pixel values of missing sub-blocks in subsequent frames.

To leverage the temporal coherency, we developed a particle filter which operates on the crowdsourced annotations. Particle filtering is a recursive Bayesian method which estimates a probability density function across a state space by maintaining a large set of particles, each of which represents a potential position ("hypothesis") [49]. Particle filters are commonly used in simultaneous localization and mapping (SLAM) systems [37, 38], as well as face [26], head [40], and hand tracking [8] systems. We selected the particle filter as our state estimation technique for three main reasons: first, particle filters can utilize information from neighboring state estimates in tandem with temporal constraints (e.g., the object has a maximum speed) to refine the state estimate. Second, particle filters can support complex measurement functions, which are required to compare 2D annotations and 3D states. Last, the particle filter does not assume an underlying distribution, which allows it to maintain multimodal and non-Gaussian distributions. This is particularly useful, as incomplete annotations permit multiple correct hypotheses. To validate the efficiency of our particle filtering based answer aggregation and state estimation method, we applied two annotation removal methods to filter out low quality annotations in two different stages: self-filtering at the time of annotation collection and outlier filtering at the time of aggregation.

In this work, we call the act of utilizing temporal dependencies between neighboring video frames *inter-frame referencing*. This lets us leverage content diversity, where the content (frames) have related information. Related information can be invariant or determined, based on how they vary. In this scenario, camera specification and camera position are invariant information, while content's sampling rate is determined by the requester. We define *content diversity* as a measure of a given set of content's variation with respect to selected attributes, e.g., the angle of view.

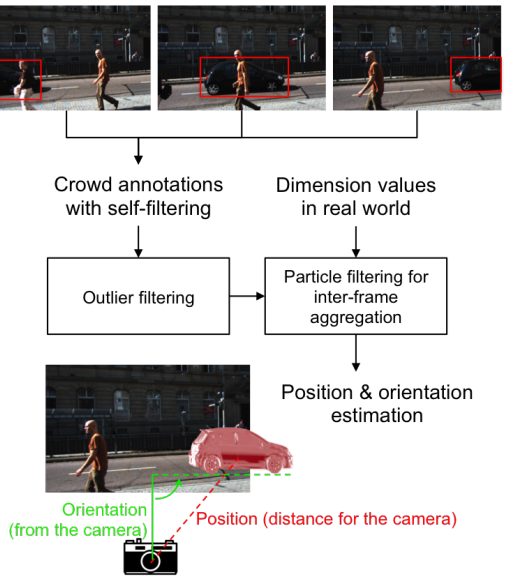


Figure 3: Overview of Popup pipeline. From workers' dimension line annotation input and additional input of real-world dimension values of the target vehicle (looked up from an existing knowledge base), Popup estimates the position and orientation of the target vehicle in 3D.

4 POPUP

Popup leverages dimension line annotations in 2D videos using particle filtering to achieve efficient and accurate 3D position and orientation estimation. Popup's pipeline consists of three main components: (1) dimension line annotation with self-filtering, (2) outlier filtering of submitted annotation sets, and (3) particle filtering based estimation of the 3D position and orientation. The overall pipeline is described in Figure 3. By feeding the output from Popup to a simulator, such as CARLA [12], a user can reconstruct and replay in 3D an event captured in monocular video. Our goal is to make it possible to create large, realistic simulated training datasets to be generated for data-driven algorithms.

Dimension Line Annotation Tool and Self-Filtering

Popup presents crowd workers with a visualization and annotation web application that allows them to crop the object of interest from a video frame and then draw dimension line annotations of the three dimension entries: length, width, and height, on the cropped object. The dimension lines can be directly drawn on objects in video frames (Figure 4(b) ①) to capture the 3D state of an object without any three dimensional interactions, e.g., rotation and scaling of a cuboid, which would require familiarity with interactive 3D tools.

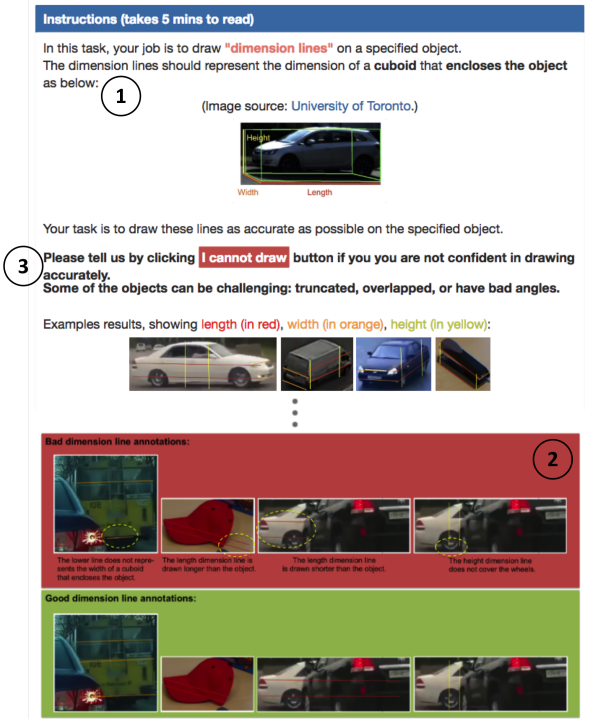
When a crowd worker reviews the dimension line annotation task, an explanation on the goal of the task is first

given (Figure 4(a) ①). Then, step-by-step instructions are provided, along with pictures exemplifying desired and undesired annotations as in Figure 4(a) ②. The instructions ask workers to click the "I cannot draw" button whenever they are not confident in their ability to accurately annotate a particular dimension (Figure 4(a) ③). Once the worker accepts the task, they can perform the first step: cropping the target object. The worker can click and drag on the given video frame to draw a box, and adjust the size and ratio of the box, as needed. The coordinate information of the box is used in the post-hoc outlier filtering step, as explained in the next section. Once the worker is done cropping the target object, she can click the 'Done with Step 1' button and proceed to the next step. Note that a worker annotates one frame at a time. The sampling rate of frames to be annotated by workers can be arbitrarily chosen by the user.

The second step is drawing the dimension line entries (length, width, and height) on the cropped vehicle. The interface has buttons that open a pop-up window to allow workers to draw dimension line annotations for each dimension entry. Workers can choose which they want to draw first. The interactive pop-up window is shown in Figure 4(b). After drawing a line, a message appears at the end of the line and asks workers "Is this end closer to the camera than the other end of the line?" The worker can answer this using a radio button. We initially asked this question to avoid ambiguity due to the Necker cube illusion [39] that occurs when drawing a cube with no visual cues for orientation. The illusion makes it impossible to distinguish the closer ends of the edges of a cube. While we found that answers varied too much to use in our final orientation estimation algorithm, we describe this step because it affects the total time of completing the dimension line annotation task. This variation is likely due to the number of ambiguous cases that arise, such as when a car is nearly 90-degrees to the camera, making it hard to perceive which horizontal end is closer to the camera. The interface asks workers to draw more than one line per dimension in order to proceed to the next step. The interface allows adjusting already drawn dimension lines or redrawing them anytime if needed. Workers are provided with the "I cannot draw" button (Figure 4(b) ②) which they can click on to self-filter dimension line annotations if they are not sure about their answer.

Outlier Removal

Popup is designed to robustly handle aggressively-filtered annotation sets. Popup has two post-hoc filtering modules to control the quality of collected annotations. The post-hoc modules assume multiple submissions per frame so that distribution statistics can be found. The first step uses the median location of the bounding boxes workers cropped from the given frame. The second step uses the standard



(a) Instructions for crowd workers

(b) Interactive UI for dimension line annotation

Figure 4: Crowd worker instructions and the interactive worker UI of Popup. (a) Step-by-step instructions with good and bad examples are provided. (b) Interactive Web UI that crowd workers can use to create, adjust, erase, and redraw *length, width* and *height* dimension lines.

deviation between the dimension lines drawn for the same object in the same frame.

Step 1: Filtering Annotation Sets.
The first step calculates the median bounding box location of submissions to filter incorrect annotation sets (all dimension lines from one annotator). For each target object, the worker crops the object of interest from the given frame. Our assumption is that a malicious worker, careless worker, or bot will fail to crop the correct target object. For width and height independently, if a cropped box does not overlap more than 50% with the median of the cropped boxes, we assume the worker annotated the wrong object and drop the annotation set of all three entries (length, width, and height). This is designed to entirely filter poor submissions.

Step 2: Filtering individual Annotation.
The second step compares the distance of the length and angle of submitted dimension line annotations from the medians. If a dimension line is outside 1.5×Interquartile Range (IQR) from the median, it is filtered. This is useful for filtering out mistakes, e.g., a height entry mistakenly drawn as a length entry or a line added by mistake and not removed, and to filter out low quality annotations. We use relative distances instead of absolute values as filtering criteria because the size of an object can differ from 30 pixels to 300 pixels.

Estimating Position & Orientation via Particle Filter

Popup uses particle filtering as a method for aggregating annotations and estimating 3D states of the target object (vehicle) in the video. A particle filter works by generating many particles, each representing a potential state (hypothesis), and using them to track and combine three probability distributions at each time step (video frame) [49]:
(1) The *previous distribution*: where the object was previously.
(2) The *transition distribution*: where the object could be, given where it was previously.
(3) The *measurement distribution*: where the object could be, given the crowd annotations.

Using these three distributions, the particle filter provides more refined estimates by leveraging temporal constraints. The particle filter used by Popup embodies these three probability distributions as follows:

(1) Previous Distribution. The previous state distribution corresponds to the final distribution of the previous time step. As we do not have any information about the vehicle's initial pose, we set the distribution at $t = 0$ as uniform within the bounds described in Experimental Setting section.

(2) Transition Distribution. The transition distribution describes the probability of a particle being in a new location given its previous location. This distribution allows the filter to maintain knowledge of potential states across time, which has two important implications: first, it means a fully determined system is not necessary at every time step, so the system is tolerant to self-, and post-hoc filtering with aggressive thresholds. Next, it applies a spatiotemporal constraint by limiting how far the vehicle can move in successive frames, narrowing the solution space. Typically the transition distribution is based on knowledge of the vehicle's kinematics and control inputs, but as our system has knowledge of neither, we introduce uncorrelated zero-mean Gaussian noise that spans set of reasonable vehicle motions.

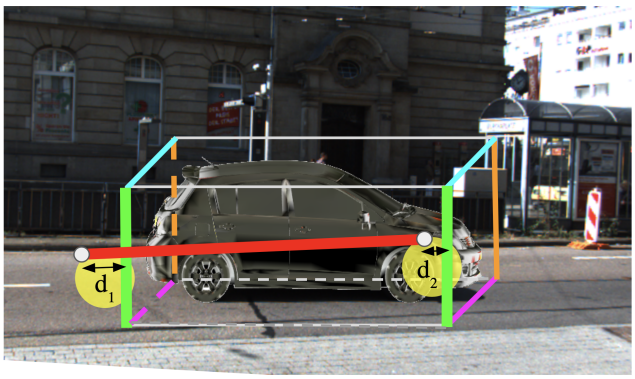


Figure 5: Perceptual distance calculation. The distances (arrows) between endpoints (grey dots of the red line) of an annotation (red line) and corresponding projected hypothesis 3D line pairs (orange, green, blue, pink) are calculated. The distances corresponding to the best-fitting 3D line pair are used to calculate probability. These probabilities are used to determine which hypothesis most closely represents the annotation line, and therefore the position in 3D space.

(3) Measurement Distribution. The measurement distribution utilizes the crowdsourced annotation lines to determine the likelihood of the hypothesis. To test a hypothesis, we create the bounding cuboid in 3D space and project it onto the image. Then, for each annotation, we determine how close its endpoints are to an appropriate pair of edges (Figure 5). This distance is then placed on a normal distribution with a mean of 0 pixels and a standard deviation of 22 pixels. We also calculate the difference between the lengths of the annotation line and corresponding projected hypothesis line, and place that on a normal distribution with a mean of 0 pixels and a standard deviation of 22 pixels. The sum of these two probabilities is used as the probability of an annotation. This function is referred to as *ERR* in Algorithm 1.

Implementation. The pseudocode for our particle filtering implementation is shown in Algorithm 1. Our state space consists of five dimensions: x, y, z, θ, and f, where x, y, z denote the relative 3D position of an object from the camera and θ denotes the orientation as illustrated at the bottom of

Algorithm 1 Particle filter algorithm for Popup

Let $S = \{(s_1, w_1) \ldots (s_N, w_N)\}$ be the set of N particles, where each particle $s_i = \{x_i, y_i, z_i, \theta_i, f_i\}$ is one hypothesis with probability $P(s_i) = w_i$. Let the initial set of particles S_0 be sampled uniformly from the given range for s_i:

for Every Frame t **do**
  $RESAMPLE(S)$
  for Every (s_i, w_i) in S **do**
    Next State Step: $s_{i,t} \leftarrow s_{i,t-1} + \mathcal{N}(0, \sigma)$
    $z \leftarrow 0$
    for Every Annotation Line **do**
      $z \leftarrow z + ERR(AnnotationLine, ParticleState)$
    end for
    $w_{i,t} = w_{i,t-1} \cdot z$
  end for
  $S \leftarrow NORMALIZE(S)$
  $estimate \leftarrow ARGMAX(w_i)$
end for

Figure 3. The last dimension, f, denotes the focal length of the camera. $RESAMPLE(S)$ generates N particles (potential states) based on the existing particles and their probabilities (w). $NORMALIZE(S)$ normalizes all the updated probabilities (w) calculated in the previous for-loop such that the probabilities sum to one. When analyzing across a single frame, we perform the action and resampling steps after iterating through every annotation set.

5 EVALUATION

To evaluate the performance of our proposed 3D video reconstruction strategy, we investigate the accuracy of the collected dimension line annotations before and after both self and outlier filtering. Next, we investigate our proposed annotation aggregation strategy that uses particle filtering to refer to neighboring frames. For the experiments, we recruited 170 workers using LegionTools [15], a toolkit that provides an easy way to recruit and route workers from Amazon Mechanical Turk in real time. We limited the crowd workers to those who are in the U.S. and have an approval rate of over 95%. All workers who accepted our task had to first read the instructions to proceed to the actual task.

Experimental Setting

Our evaluation is done using the KITTI[1] dataset [14], which contains traffic scenes recorded from a moving vehicle using multiple sensor modalities. Along with 2D RGB video scenes, the dataset provides ground truth measurements of distance and orientation of objects relative to the camera. The scenes in the dataset include occluded, truncated, and cropped objects that are challenging and thus appropriate to test the performance of Popup.

In this experiment, we targeted reconstructing the 3D state of one moving vehicle per video clip. There were a total of 21 video clips in the dataset, of which we used 17, as we excluded clips with no vehicle or with no vehicle that spans our sampling range. We sampled 10 frames from each video clip at a rate of two frames per second. For each video clip, we recruited 10 workers to provide annotations. Each worker annotated every other sampled frame, for a total of five frames. That is, for each frame, annotation sets from five different workers were collected. Each worker was paid $1.10 per task, a pay rate of ~$9/hr. To understand the reason why some annotations were self-filtered, we presented the workers with a multiple-choice question when an annotation was self-filtered. The choices were: 1) "The object is heavily occluded", 2) "I don't understand the instruction", and 3) "Other". We asked the workers to still draw the dimension line *after* reporting "I cannot draw" to directly compare accuracy with and without workers' self-filtering. To obtain the true 3D dimensions of the annotated vehicles, we used the ground truth information included in the KITTI dataset. In a real-world deployment of Popup, the dimensions would be found online or in appropriate documentation prior to generating the 3D reconstruction. To reduce computation time and avoid suboptimal estimation, we set bounds for for the 3D pose: $-30 \leq x \leq 30$, $-4 \leq y \leq 4$, $1 \leq z \leq 140$, $0 \leq \theta < \pi$, and $500 \leq f \leq 1000$. Position is given in meters, orientation in radians, and focal length in pixels. We used 50,000 particles for all the particle filtering based conditions.

Results from Dimension Line Annotation

In this section, we present experimental results from collecting 3D dimension line annotations (an example of which is shown in Figure 6) using Popup.

Result of Filtering Annotation Sets.
The first outlier filtering step removed low-quality annotation sets (all dimension lines from one worker) based on bounding box coordinates of each submission. 7% of 850 submissions were filtered in total. We found that few incorrect objects (under 2%) still remained after the filtering step, which occurred when the majority of workers (at least three out of five) annotated an incorrect object.

Result of Self-Filtering.
After the first step of outlier filtering, 793 annotation sets remained in the collection. Each annotation set has three dimension entries (length, width, and height), resulting in a total 2379 entries submitted. Among the submissions, the number of self-filtered entries was 176, which were 7% of the total submitted dimension line entries. Of the self-filtered

[1] Available at: http://www.cvlibs.net/datasets/kitti/

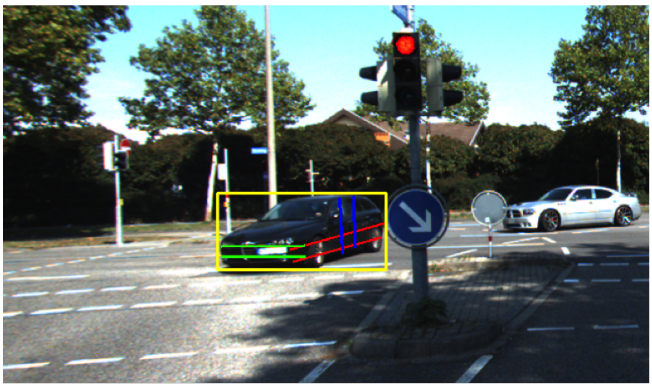


Figure 6: Example of dimension line annotations from one of crowd workers in our experiment. The yellow bounding box is the area that the worker cropped in Step 1, and the red, green, and blue lines are length, width, and height annotations (respectively) drawn in Step 2.

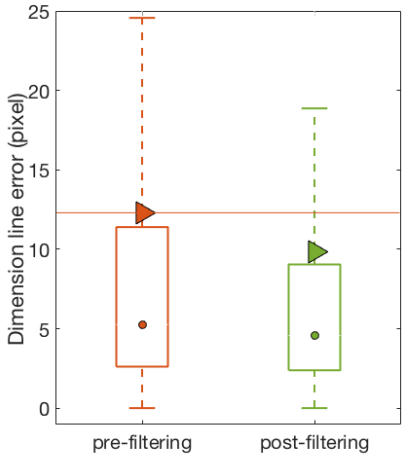


Figure 7: Height dimension-line error (lower is better). The left is without any filtering, and the right is with both outlier and self-filtering. After filtering, the average error was reduced by 20% ($p < .05$). For each box plot, the circle denotes the median and the triangle denotes the mean. The lower and upper edges of boxes denote the 25-th and 75-th percentiles, respectively. The whiskers extend to the most extreme data-points not considered to be outliers.

entries, 34% were filtered for the reason "The object is heavily occluded", and 66% were filtered for the reason "Other". There were no instances where the "I don't understand the instruction" option was chosen. When the "Other" option was chosen, workers could manually enter the reason behind their decision. Most explanations were related to insufficient visual information, e.g., "the object runs off the given image", "it's mostly back view", and "Bad angle, low resolution" as shown in Figure 9. We initially expected a higher self-filtering rate because many of our selected scenes contained objects that are hard to annotate (e.g., truncated or occluded objects). We discuss the potential reasons for the low self-filtering rate further in the Discussion section.

Result of Filtering Individual Annotations.
In the final outlier filtering step, we filtered individual annotations based on the dimension line's length and angular distance from the median. Of the individual annotations, 13% were considered outliers and filtered from the collection. We found that a few (under 3%) outlier annotations did not get filtered with our method. These were cases where the object was relatively small in the scene, and the variance within good annotations was very close to the variance between good and poor annotations.

Accuracy of Dimension Line Annotations.
We examined the effect of pre-processing filtering on the average accuracy of dimension line annotations. Since the dimension line ground truth is not provided by the KITTI dataset, we projected the actual vehicle height of the target vehicle onto the image plane, and compared the difference from the projected height line with the annotated dimension line in pixel units. This analysis was not performed on width and length dimension lines as they are not parallel to the image plane. In our experiment, the distributions were all approximately normal, but with positive skew. Because the distributions were skewed, we computed p-values using Wilcoxon Rank-Sum test. As shown in Figure 7, the pre-processing filtering reduced the average error of dimension lines by 20% ($p < .05$) on average. Note that the mean error after filtering is under 10 pixels (9.8 pixels). Given the frame heights are 375-pixel, the average error is under 3% of the full height of a video frame.

Time Savings from Self-Filtering.
We investigated the average latency of partially- and fully-completed sets of annotations. Because the distributions were skewed normal, we computed p-values using Wilcoxon Rank-Sum test. As shown in Figure 8, we found annotations took approximately 16% longer for annotations for which at least one worker self-filtered ($p < .005$). The result suggests two things: first, self-filtering can reflect a worker's confidence level as we intended in the design stage. This can be inferred by the fact that it took significantly more time for those who did not self-filter the entry, implying that it was also more challenging for them. Second, we can reduce total latency in annotation collection if we encourage workers to self-filter the challenging entries, because they can save time on the drawing activity by skipping them. In this experiment, we

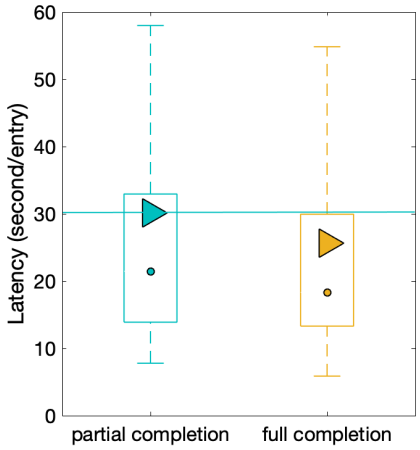


Figure 8: Average latency of partial and full annotation completion. The full completion represents typical entries – entries where no worker self-filtered. The partial completion represents entries that at least one worker self-filtered. The partial completion entries took an average of 16% more time to annotate ($p < .005$). For each box plot, the circle denotes the median and the triangle denotes the mean. The lower and upper edges of boxes denote the $25\text{-}th$ and $75\text{-}th$ percentiles, respectively. The whiskers extend to the most extreme data-points not considered to be outliers.

could save 16% of the annotation time for the hard annotations when workers self-filtered annotations.

Results from Aggregation and State Estimation

In this section, we evaluate our proposed annotation aggregation and state estimation method under different conditions by comparing it against the ground truth from the KITTI dataset [14]. For all evaluations, we dropped outliers; any data point outside 1.5×Interquartile Range (IQR) was removed for fair comparison between conditions.

Evaluation Metrics.
For the evaluation of the accuracy of the state estimates, we used two metrics: a distance difference metric and an angular difference metric. The distance difference metric is the Euclidean distance between the ground truth and the estimate. The angular difference metric corresponds to the smallest angular difference between estimated orientation and the ground truth orientation (Equation 1):

$$\begin{aligned} \text{DistanceDiff} &= \sqrt{(x_g - x_e)^2 + (y_g - y_e)^2 + (z_g - z_e)^2} \\ \text{AngularDiff} &= |(\theta_g - \theta_e) \; mod \; \pi/2| \end{aligned} \quad (1)$$

where x_g, y_g, and z_g are the 3D ground truth, x_e, y_e, and z_e are the 3D state estimate, θ_g is the ground truth orientation, and θ_e is the orientation estimate.

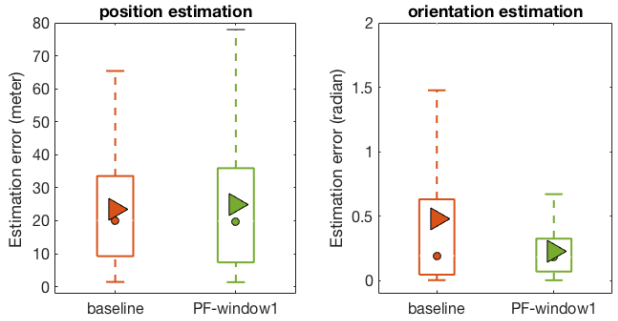

Video #1, frame #3: **4 out of 5 workers self-filtered 'length' entry.**
Reason: No side view.


Video #5, frame #10: **4 out of 5 workers self-filtered 'width' entry.**
Reason: Occluded.

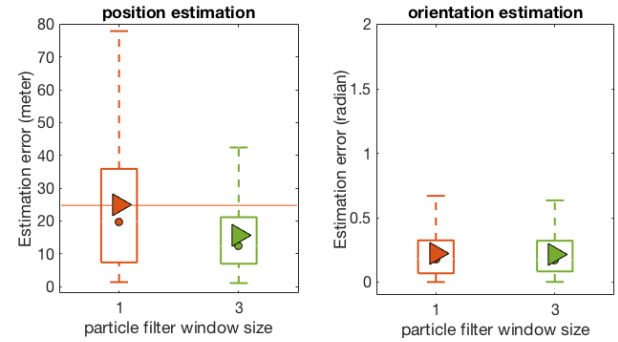

Video #9, frame #2: **all 5 workers self-filtered 'length' entry.**
Reason: Low resolution.

Figure 9: Example of challenging frames where more than three out of five workers self-filtered. The cases include limited side view, occlusion, and low resolution.

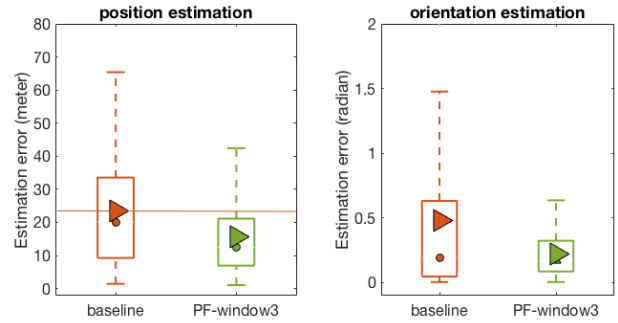
Baseline and Conditions.
To assess the success of the proposed aggregation and 3D state estimation strategy, we compare the performance of our particle filtering based method with a baseline method that uses geometric reprojection and an L-BFGS-B [55] based optimization technique. Note that the baseline cannot leverage content diversity because it does not utilize the temporal dependencies between different video frames (contents). For convenience, we will define the ability to utilize the temporal dependencies between different frames as *inter-frame referencing*, in contrast to intra-frame referencing. The two conditions that are compared to the baseline method are 1) particle filtering without inter-frame referencing and 2) particle filtering with inter-frame referencing. In addition, we look at the performance difference in window sizes for inter-frame referencing: using three, five, and ten frames as window size.



(a) State estimation error of baseline vs. particle filtering without inter-frame referencing



(b) State estimation error of particle filtering without vs. with inter-frame referencing



(c) State estimation error of baseline vs. particle filtering with inter-frame referencing

Figure 10: (a) Without inter-frame referencing, the particle filter's performance is comparable to the baseline. (b) Inter-frame referencing reduced position estimation error 37% ($p < .001$) when window size was 3. Window size 1 indicates without inter-frame referencing. (c) Our proposed method outperforms the baseline for position estimation, 33% error reduction ($p < .001$). The orange horizontal lines indicate significant difference. For each box plot, the circle denotes the median and the triangle denotes the mean. The lower and upper edges of boxes denote the 25-th and 75-th percentiles, respectively. The whiskers extend to the most extreme-most data points that are not considered to be outliers.

Baseline: The baseline method reprojects a 3D cuboid onto a given video frame and compares the corner location of the re-projection with the endpoints of the average dimension lines drawn for the target vehicle. This comparison is used as the cost function, and the L-BFGS-B optimization method [55] is used for minimization. L-BFGS-B is a well-studied optimization algorithm that is used in the state-of-the-art techniques for estimating distributions in various applications, e.g., medical image processing [50] and computer graphics [42]. The baseline method cannot handle cases where a whole entry (e.g., all height, length, and width annotations) is missing, as the problem is underdetermined. Since the baseline method cannot refer to other frames' annotations by utilizing spatiotemporal constraints, the baseline was only run for single frame based estimation.

C1. Particle filtering without inter-frame referencing: For a fair comparison with the baseline, this condition runs the particle filtering algorithm without referring to other frames. Since this condition does not refer to other frames, the solution is underdetermined under the same conditions as the baseline. The comparison between the baseline and this condition emphasizes the effect of using inter-frame referencing which is enabled by the proposed particle filtering based annotation aggregation in the next condition.

C2. Particle filtering with inter-frame referencing: The strength of using particle filtering is that it can leverage content diversity by referring to information from other frames to complement missing annotations. For this condition, we first evaluate the performance of different window sizes for inter-frame referencing: three, five, and ten. After evaluating the performance of different window sizes, we compare the state estimate performance of the best window size with the baseline performance.

Evaluation 1: Particle Filter Without Inter-Frame Referencing. To evaluate the performance of our particle filtering, we compared the performance of condition C1 with the baseline. Figure 10(a) shows the position and orientation performance of the two conditions. The left graph shows position error and the right graph shows orientation error of the estimation results. In each graph, the left box plot shows the baseline condition applied to individual frames, and the right box plot shows condition C1: particle filtering applied to individual frames without inter-frame referencing. Note that we present our results as a box and whisker plot throughout this section, as the distributions were all approximately skewed normal with positive skew. Because they were skewed, we computed p-values using Wilcoxon Rank-Sum test. The summarized result shows that in terms of position estimation, the baseline and the proposed particle filtering method perform

similarly (no significant difference was observed, $p = 0.89$). In terms of orientation estimation, we observed a 53% lower mean for our proposed particle filtering method compared to the baseline. However, while the effect size was medium-large ($d = 0.65$), the results were only approaching statistical significance ($p = 0.11$). We assume that the difference in performance comes from how the two methods incorporate dimension line evidence. The baseline method averages given dimension lines, and considers the average as an edge of the 3D re-projected cuboid to minimize the difference from the projection and the dimension lines. In contrast, the particle filtering based method does not compute an average, but compares each dimension line to the 3D re-projected cuboid to update the orientation estimation. This enables retaining information from all given dimension line annotations. Overall, the result implies that without inter-frame referencing to utilize temporal dependencies, our proposed method is comparable to the state-of-the-art baseline.

Evaluation 2: The effect of Inter-Frame Referencing.
The primary strength of using particle filtering based estimation is in that we can leverage content diversity of different video frames and refer to other frames' state estimates when estimating the current frame's state. This allows us to fill in the missing information caused by either outlier-filtering or self-filtering. We looked at three different window sizes and window size of three frames had the lowest average state estimate error. Therefore, our comparisons against single-frame particle filtering (without inter-frame referencing) was performed with this window size. The summarized result in Figure 10(b) shows that referencing three neighboring frames (including the current frame) results in a 37% improvement in accuracy compared to not referencing neighboring frames in terms of position estimation ($p < .001$). However, orientation estimation accuracy did not improve by referring to neighboring frames.

Evaluation 3: Baseline vs. Proposed Method.
To evaluate the performance of Popup in leveraging the content diversity of different frames to utilize temporal dependencies, we compare the baseline result with Popup using a three-frame reference window—the best performing window size tested in *Evaluation 2*. Figure 10(c) shows the position and orientation estimation results. In terms of position estimation, the average error was reduced by 33% ($p < .001$). In terms of orientation estimation, we observed an average error reduction of 54%, but with low confidence, as the results were only approaching statistical significance ($p = .105$). The result shows that the proposed aggregation and estimation strategy for crowdsourcing image annotations in videos can handle noisy and incomplete annotation sets, and also outperform the state-of-the-art baseline condition in terms of position estimation.

6 DISCUSSION

In this section, we discuss the effect of inter-frame referencing, leveraging content diversity in annotation tasks, factors that can affect workers' self-filtering behavior, and factors affecting the final 3D estimation quality.

The Effect of Inter-Frame Referencing

Our evaluation shows that referencing annotations from multiple neighboring frames can increase estimation accuracy in video annotation tasks. We tested four different window sizes, and found the window size affects the impact of inter-frame referencing. For position estimation, windowing seemed to improve the state estimate accuracy, but the effect was not linear, being maximized at three-frame window. For orientation estimation, the performance was consistent from window size one to five. The performance largely degraded for the 10-frame window. We speculate that the reason we did not observe progressive improvement in accuracy with increased window size is because of propagation of bad annotations. If one frame is poorly annotated, it will affect all other frames within the window. It follows that a larger window size allows local errors to affect more frames, which results in a larger aggregated overall error. For example, a critical error in frame k will affect only frame $k-1$ and $k+1$ in a three-frame window, but will affect all 10 frames in a 10-frame window.

Leveraging Content Diversity in Annotation Tasks

Our proposed method presents a new paradigm for crowdsourced annotation tasks that efficiently collects and aggregates annotations from diverse content to improve the aggregate output accuracy. Content diversity is defined as a property of an input dataset—set of video frames in this case—which measures the mutual information of the set with respect to selected attribute. Less mutual information indicates greater content diversity. While we demonstrate the concept of leveraging content diversity in 3D video reconstruction task, we expect the strategy to benefit other annotation tasks, especially where the task requires complex manual annotations and needs highly accurate aggregation results. For other tasks to benefit from our approach, we suggest the task meets the following conditions: First, the instances to be annotated should share related and invariant information. For example, annotating two random images cannot leverage our approach because they capture different scenes, and have different camera specifications. Second, it must be possible to find content where target attributes vary across instances. In our study, the requester could determine timestamps of instances. Lastly, annotations to the content should be provided given the same request. In this experiment, the task was to "annotate three dimension lines: length,

width, and height". If the task instruction varied between the contents, e.g., "annotate 3D bounding box", it would not be possible to aggregate the annotations due to their different response types.

Factors Affecting Self-Filtering

There are many factors which can affect the self-filtering rate and accuracy, but we believe three to be of primary importance: the quality of instructions, the complexity of the task, and the incentive mechanism. The instruction and task can be designed to help workers clearly understand the benefit of self-filtering. In our post survey, we asked workers who completed our task if they think it is better to provide an answer or not when they are not confident about the answer being correct. One worker answered, "I think an attempt at an answer is better than none at all. Even if you aren't sure an attempt at least shows your [*sic*] trying to help the study and not just wasting everyone's time". Another answered, "Try my level best to satisfy the requester". The survey response tells that crowd workers are willing to help the requester but they might not know what is most helpful, resulting in them submitting low-confidence annotations even when they should be self-filtered. Therefore, providing clear instructions on how to benefit the task would lead to better usage of self-filtering. Another factor that affects the self-filtering rate is the complexity of the task. If a task is too complex for the non-expert crowd workers, they might feel like self-filtering a lot of annotations. On the other hand, workers might submit a lot of low confidence annotations with the expectation to interact with the requester as a follow-up. In our post-study survey, one worker commented, "I think providing some answer is better then none. It creates a line of communication". Last, we believe that the incentive mechanism of the platform affects workers' use of the self-filtering option. Shah et al. [44] gave a clear incentive to the workers, which encouraged them to use the self-filtering option ("I'm not sure" option) wisely. This resulted in the highest data quality in their experiments. Thus, requesters should clearly design an incentive mechanism and mention in the task how they would like workers to use the self-filtering option.

Other Factors Affecting State Estimation Accuracy

As indicated by our experiment on window size, and the counterintuitive result that a window of size three outperforms other window sizes, both the baseline and proposed state estimation method depend on parameters which must be carefully chosen. Improper parameters, especially when state bounds are manually specified, can cause the system to behave poorly in unpredictable ways. Further, we note 3D state estimation accuracy dependents on the consensus between workers. That is, if a majority of workers annotate an incorrect vehicle, we cannot detect them with the proposed methods. This causes error propagation when using temporally-aware methods.

7 CONCLUSION

In this paper, we have introduced a new crowdsourcing approach that leverages content diversity to collect and aggregate annotations more efficiently and accurately. We use particle filtering to aggregate annotations from multiple video frames with different contents and provide an accurate final output even in the presence of incomplete or missing annotations. We introduced Popup, a hybrid intelligence system that implements the proposed methods to reconstruct 3D information from 2D videos. Our results demonstrate that the proposed particle filtering based aggregation method can handle noisy and missing annotations, enabling the generation of more accurate 3D state estimations. Because the proposed method is robust to missing annotations, overally latency can be reduced during data collection by allowing the annotators (here, crowd workers) to self-filter. In the future, output from Popup can potentially be passed to simulation software to enable generating a realistic and large 3D training dataset of rare events for autonomous vehicles and machines to learn.

8 ACKNOWLEDGMENTS

This research was supported in part by the Denso Corporation and MCity at the University of Michigan. Toyota Research Institute ("TRI") provided funds to assist the authors with their research but this article solely reflects the opinions and conclusions of its authors and not TRI or any other Toyota entity. This research was also supported in part by the Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT (NRF-2017M3C4A7065960).

REFERENCES

[1] 2017. Inside Waymo's Secret World for Training Self-Driving Cars. https://www.theatlantic.com/technology/archive/2017/08/inside-waymos-secret-testing-and-simulation-facilities/537648/.

[2] 2018. Waymo Has The Most Autonomous Miles, By A Lot. https://www.forbes.com/sites/davidsilver/2018/07/26/waymo-has-the-most-autonomous-miles-by-a-lot/.

[3] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. 2013. OpenSurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics (TOG)* 32, 4 (2013), 111.

[4] Michael S Bernstein, Joel Brandt, Robert C Miller, and David R Karger. 2011. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 33–42.

[5] Jeffrey P Bigham, Chandrika Jayant, Hanjie Ji, Greg Little, Andrew Miller, Robert C Miller, Robin Miller, Aubrey Tatarowicz, Brandyn White, Samual White, et al. 2010. VizWiz: nearly real-time answers to

visual questions. In *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM, 333–342.

[6] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316* (2016).

[7] Jonathan Bragg, Mausam, and Daniel S. Weld. 2013. Crowdsourcing multi-label classification for taxonomy creation. In *First AAAI conference on human computation and crowdsourcing*.

[8] Matthieu Bray, Esther Koller-Meier, and Luc Van Gool. 2004. Smart particle filtering for 3D hand tracking. In *Automatic Face and Gesture Recognition, 2004. Proceedings. Sixth IEEE International Conference on*. IEEE, 675–680.

[9] Joseph Chee Chang, Saleema Amershi, and Ece Kamar. 2017. Revolt: Collaborative crowdsourcing for labeling machine learning datasets. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. ACM, 2334–2346.

[10] Alexander Philip Dawid and Allan M Skene. 1979. Maximum likelihood estimation of observer error-rates using the EM algorithm. *Applied statistics* (1979), 20–28.

[11] Roberto Di Salvo, Daniela Giordano, and Isaak Kavasidis. 2013. A crowdsourcing approach to support video annotation. In *Proceedings of the International Workshop on Video and Image Ground Truth in Computer Vision Applications*. ACM, 8.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. CARLA: An open urban driving simulator. *arXiv preprint arXiv:1711.03938* (2017).

[13] Ujwal Gadiraju, Besnik Fetahu, and Ricardo Kawase. 2015. Training workers for improving performance in crowdsourcing microtasks. In *Design for Teaching and Learning in a Networked World*. Springer, 100–114.

[14] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[15] Mitchell Gordon, Jeffrey P Bigham, and Walter S Lasecki. 2015. LegionTools: a toolkit+ UI for recruiting and routing crowds to synchronous real-time tasks. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. ACM, 81–82.

[16] Sai R Gouravajhala, Jinyeong Yim, Karthik Desingh, Yanda Huang, Odest Chadwicke Jenkins, and Walter S Lasecki. 2018. EURECA: Enhanced Understanding of Real Environments via Crowd Assistance. (2018).

[17] Kotaro Hara, Jin Sun, Robert Moore, David Jacobs, and Jon Froehlich. 2014. Tohme: detecting curb ramps in google street view using crowdsourcing, computer vision, and machine learning. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 189–204.

[18] Panagiotis G Ipeirotis, Foster Provost, and Jing Wang. 2010. Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*. ACM, 64–67.

[19] Nidhi Kalra and Susan M Paddock. 2016. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice* 94 (2016), 182–193.

[20] Ece Kamar, Severin Hacker, and Eric Horvitz. 2012. Combining human and machine intelligence in large-scale crowdsourcing. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 467–474.

[21] Alexandre Kaspar, Genevieve Patterson, Changil Kim, Yagiz Aksoy, Wojciech Matusik, and Mohamed Elgharib. 2018. Crowd-Guided Ensembles: How Can We Choreograph Crowd Workers for Video Segmentation?. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, 111:1–111:12.

[22] Harmanpreet Kaur, Mitchell Gordon, Yiwei Yang, Jeffrey P Bigham, Jaime Teevan, Ece Kamar, and Walter S Lasecki. 2017. Crowdmask: Using crowds to preserve privacy in crowd-powered systems via progressive filtering. In *Proceedings of the AAAI Conference on Human Computation (HCOMP 2017)., HCOMP*, Vol. 17.

[23] Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J Guo, Robert C Miller, and Krzysztof Z Gajos. 2014. Crowdsourcing step-by-step information extraction to enhance existing how-to videos. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, 4017–4026.

[24] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. 2013. The future of crowd work. In *Proceedings of the 2013 conference on Computer supported cooperative work*. ACM, 1301–1318.

[25] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123, 1 (2017), 32–73.

[26] B Kwolek. [n. d.]. Model Based Facial Pose Tracking Using a Particle Filter. In *Geometric Modeling and Imaging–New Trends (GMAI'06)*.

[27] Gierad Laput, Walter S Lasecki, Jason Wiese, Robert Xiao, Jeffrey P Bigham, and Chris Harrison. 2015. Zensors: Adaptive, rapidly deployable, human-intelligent sensor feeds. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. ACM, 1935–1944.

[28] Walter S Lasecki. 2013. Real-time conversational crowd assistants. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*. ACM, 2725–2730.

[29] Walter S. Lasecki, Mitchell Gordon, Danai Koutra, Malte F. Jung, Steven P. Dow, and Jeffrey P. Bigham. 2014. Glance: Rapidly coding behavioral video with the crowd. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*. ACM, 551–562.

[30] Walter S Lasecki, Christopher Homan, and Jeffrey P Bigham. 2014. Architecting real-time crowd-powered systems. *Human Computation* 1, 1 (2014).

[31] Walter S. Lasecki, Kyle I. Murray, Samuel White, Robert C. Miller, and Jeffrey P. Bigham. 2011. Real-time crowd control of existing interfaces. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. ACM, 23–32.

[32] Walter S. Lasecki, Young Chol Song, Henry Kautz, and Jeffrey P. Bigham. 2013. Real-time crowd labeling for deployable activity recognition. In *Proceedings of the 2013 conference on Computer supported cooperative work*. ACM, 1203–1212.

[33] Walter S Lasecki, Rachel Wesley, Jeffrey Nichols, Anand Kulkarni, James F Allen, and Jeffrey P Bigham. 2013. Chorus: a crowd-powered conversational assistant. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. ACM, 151–162.

[34] Christopher Lin, Mausam Mausam, and Daniel Weld. 2012. Dynamically Switching between Synergistic workflows for Crowdsourcing. In *AAAI Conference on Artificial Intelligence*.

[35] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *European conference on computer vision*. Springer, 740–755.

[36] Andrew Mao, Ece Kamar, Yiling Chen, Eric Horvitz, Megan E Schwamb, Chris J Lintott, and Arfon M Smith. 2013. Volunteering versus work for pay: Incentives and tradeoffs in crowdsourcing. In *First AAAI conference on human computation and crowdsourcing*.

[37] Michael Montemerlo and Sebastian Thrun. 2007. FastSLAM 2.0. *FastSLAM: A scalable method for the simultaneous localization and mapping problem in robotics* (2007), 63–90.

[38] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. 2002. FastSLAM: A factored solution to the simultaneous localization and mapping problem. *Aaai/iaai* 593598 (2002).

[39] Louis Albert Necker. 1832. LXI. Observations on some remarkable optical phenomena seen in Switzerland; and on an optical phenomenon which occurs on viewing a figure of a crystal or geometrical solid. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 1, 5 (1832), 329–337.

[40] Kenji Oka, Yoichi Sato, Yasuto Nakanishi, and Hideki Koike. 2005. Head Pose Estimation System Based on Particle Filtering with Adaptive Diffusion Control.. In *MVA*. 586–589.

[41] Sunghyun Park, Gelareh Mohammadi, Ron Artstein, and Louis-Philippe Morency. 2012. Crowdsourcing micro-level multimedia annotations: The challenges of evaluation and interface. In *Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia*. ACM, 29–34.

[42] Leonid Pishchulin, Stefanie Wuhrer, Thomas Helten, Christian Theobalt, and Bernt Schiele. 2017. Building statistical shape spaces for 3d human modeling. *Pattern Recognition* 67 (2017), 276–286.

[43] Akshay Rao, Harmanpreet Kaur, and Walter S Lasecki. 2018. Plexiglass: Multiplexing Passive and Active Tasks for More Efficient Crowdsourcing. In *Proceedings of the AAAI 2018 Conference on Human Computation*. ACM.

[44] Nihar Bhadresh Shah and Denny Zhou. 2015. Double or nothing: Multiplicative incentive mechanisms for crowdsourcing. In *Advances in neural information processing systems*. 1–9.

[45] Jean Y. Song, Raymond Fok, Alan Lundgard, Fan Yang, Juho Kim, and Walter S. Lasecki. 2018. Two Tools Are Better Than One: Tool Diversity As a Means of Improving Aggregate Crowd Performance. In *23rd International Conference on Intelligent User Interfaces (IUI '18)*. ACM, New York, NY, USA, 559–570.

[46] Alexander Sorokin, Dmitry Berenson, Siddhartha S Srinivasa, and Martial Hebert. 2010. People helping robots helping people: Crowdsourcing for grasping novel objects. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2117–2122.

[47] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. 2015. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE International Conference on Computer Vision*. 2686–2694.

[48] Ryan Szeto and Jason J Corso. 2017. Click Here: Human-Localized Keypoints as Guidance for Viewpoint Estimation. In *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, 1604–1613.

[49] Sebastian Thrun. 2000. Monte Carlo POMDPs. In *Advances in neural information processing systems*. 1064–1070.

[50] Yu-Jung Tsai, Alexandre Bousse, Matthias J Ehrhardt, Charles W Stearns, Sangtae Ahn, Brian F Hutton, Simon Arridge, and Kris Thielemans. 2018. Fast Quasi-Newton Algorithms for Penalized Reconstruction in Emission Tomography and Further Improvements via Preconditioning. *IEEE transactions on medical imaging* 37, 4 (2018), 1000–1010.

[51] Carl Vondrick, Donald Patterson, and Deva Ramanan. 2013. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision* 101, 1 (2013), 184–204.

[52] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. 2003. Overview of the H. 264/AVC video coding standard. *IEEE Transactions on circuits and systems for video technology* 13, 7 (2003), 560–576.

[53] Jinyeong Yim, Jeel Jasani, Aubrey Henderson, Danai Koutra, Steven P Dow, Winnie Leung, Ellen Lim, Mitchell Gordon, Jeffrey P Bigham, and Walter S Lasecki. 2016. Coding Varied Behavior Types Using the Crowd. In *Proceedings of the 19th ACM Conference on Computer Supported Cooperative Work and Social Computing Companion*. ACM, 114–117.

[54] Jenny Yuen, Bryan Russell, Ce Liu, and Antonio Torralba. 2009. Labelme video: Building a video database with human annotations. In *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 1451–1458.

[55] Ciyou Zhu, Richard H Byrd, Peihuang Lu, and Jorge Nocedal. 1997. Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Transactions on Mathematical Software (TOMS)* 23, 4 (1997), 550–560.