Accuracy and Usability of Smartphone-Based Distance Estimation Approaches for Visual Assistive Technology Development

Giles Hamilton-Fletcher, Mingxin Liu, Diwei Sheng, Chen Feng, Todd E. Hudson, John-Ross Rizzo, and Kevin C. Chan

Abstract*—*Goal: **Distance information is highly requested in assistive smartphone Apps by people who are blind or low vision (PBLV). However, current techniques have not been evaluated systematically for accuracy and usability. *Methods:* We tested five smartphone-based distance-estimation approaches in the image center and periphery at 1–3 meters, including machine learning (CoreML), infrared grid distortion (IR_self), light detection and ranging (LiDAR_back), and augmented reality room-tracking on the front (ARKit_self) and back-facing cameras (ARKit_back). *Results:* For accuracy in the image center, all approaches had $<\pm$2.5 cm average error, except CoreML which had $\pm$5.2–6.2 cm average error at 2–3 meters. In the periphery, all approaches were more inaccurate, with CoreML and IR_self having the highest average errors at $\pm$41 cm and $\pm$32 cm respectively. For usability, CoreML fared favorably with the lowest central processing unit usage, second lowest battery usage, highest field-of-view, and no specialized sensor requirements. *Conclusions:* We provide key information that helps design reliable smartphone-based visual assistive technologies to enhance the functionality of PBLV.**

***Index Terms*—Assistive technology, sensory substitution, blindness, low vision, navigation.**

***Impact Statement*— We compared five smartphone distance-estimation approaches suitable for visual assistive technologies. LiDAR and augmented reality approaches were the most accurate, distance errors increased toward peripheries, and machine learning had advantages of usability and accessibility.**

Manuscript received 15 August 2023; revised 8 December 2023, 15 January 2024, and 22 January 2024; accepted 22 January 2024. Date of publication 25 January 2024; date of current version 23 February 2024. This work was supported in part by the U.S. Department of Defense Vision Research Program under Grant W81XWH2110615 (Arlington, Virginia), in part by the U.S. National Institutes of Health under Grant R01-EY034897 (Bethesda, Maryland), and in part by unrestricted grant from Research to Prevent Blindness to NYU Langone Health Department of Ophthalmology (New York, New York). The review of this article was arranged by Editor Esteban J. Javier Pino. *(Corresponding author: Kevin C. Chan.)*

Giles Hamilton-Fletcher is with the Department of Ophthalmology, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA, and also with the Department of Rehabilitative Medicine, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA (e-mail: giles.hamilton-fletcher@nyulangone.org).

Mingxin Liu is with the Department of Ophthalmology, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA (e-mail: ml7324@nyu.edu).

Diwei Sheng and Chen Feng are with the Department of Civil and Urban Engineering & Department of Mechanical and Aerospace Engineering, New York University Tandon School of Engineering, Brooklyn, NY 11201 USA (e-mail: diweisheng1@gmail.com; cfeng@nyu.edu).

Todd E. Hudson is with the Department of Rehabilitative Medicine, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA (e-mail: hudson.te@gmail.com).

John-Ross Rizzo is with the Department of Rehabilitative Medicine, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA, and also with the Department of Biomedical Engineering, Tandon School of Engineering, New York University, New York, NY 11201 USA (e-mail: johnross.rizzo@nyulangone.org).

Kevin C. Chan is with the Department of Ophthalmology, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA, also with the Department of Biomedical Engineering, Tandon School of Engineering, New York University, New York, NY 11201 USA, and also with the Department of Radiology, NYU Grossman School of Medicine, NYU Langone Health, New York University, New York, NY 10017 USA (e-mail: chuenwing.chan@fulbrightmail.org).

This article has supplementary downloadable material available at https://doi.org/10.1109/OJEMB.2024.3358562, provided by the authors.

Digital Object Identifier 10.1109/OJEMB.2024.3358562

I. Introduction

ASSISTIVE technology can make visuospatial information accessible to people with blindness or low vision (PBLV) through visual enhancements or audio/tactile feedback. Images can be conveyed at the semantic-level (e.g., objects to speech; text to braille), or at the sensory-level, where the distribution of light, color, or distance values in an image can be preserved within abstract patterns of audio/tactile feedback using sensory substitution devices (SSDs). For instance, SSDs can convey a bright diagonal line as a sweeping auditory tone ascending in pitch, or as a diagonal fizzing sensation on the tongue [1]. From this cross-modal information, the user reconstructs the original image in the mind's eye, enhancing both image understanding and their ability to act in the visual world [2]. As a result, these biomedical devices enhance the 'visual function' of PBLV.

Visual assistive technologies employ a wide variety of approaches for representing the environment to PBLV. In the simplest form, these devices can convert a single value of sensory information (e.g., a single pixel or object's distance) to the user. This can be provided either verbally ("1 meter"), or abstractly,

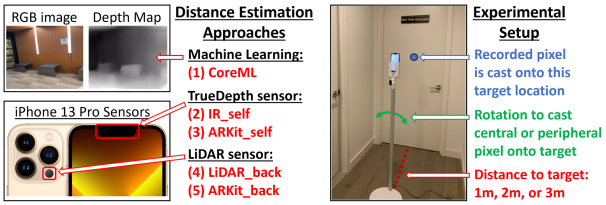

Fig. 1. Distance estimation approaches and experimental setup. Left panel shows five distance estimation approaches, including CoreML, which converts RGB images to depth maps (proximity displayed as brightness). The iPhone 13 Pro can estimate distances using its TrueDepth sensor via infrared grid distortion, or LiDAR sensor via infrared time-of-flight. ARKit methods also use visual-inertial odometry. Right panel shows the experimental setup of measuring distance estimations using a target door in a simple hallway.

with increasing proximity conveyed by increasing audio/tactile intensity [3]. Abstract methods can represent full images using complex auditory/tactile patterns. This involves using multiple sensory dimensions simultaneously and giving feedback in real-time [1]. These devices typically build on intuitive cross-sensory mappings and auditory psychology to facilitate more accurate user image reconstructions [4], [5].

Advancements in the processing power, machine learning (ML) support, and sensors available on smartphones can make both sensory and semantic tools cheaper, more accessible, and portable, without the need of bespoke hardware designs. When PBLV are interviewed about these tools, the most highly sought-after sensory information is distance for objects or people [6]. Recent iPhone Apps like 'LiDAR Sense', 'Super Lidar', Apple's 'Magnifier' and 'SoundSight' convey distances for PBLV – for either a single pixel, object or person, or for conveying an entire depth map respectively [7].

There are now multiple ways to gather distance information on modern smartphones, each technique impacting spatial accuracy and usability differently. However, to date, there has not been a systematic evaluation of approaches suitable for informing visual assistive App development. In the present study, we focus on Apple's iPhone series, as it is the primary choice of both assistive technology companies and Western PBLV [8]. The iPhone 13 Pro model supports multiple distance estimation approaches, making it a suitable platform for comparison. This includes distance estimation from ML approaches on red-green-blue (RGB) images, infrared (IR) grid distortion, light detection and ranging (LiDAR), and hybrid approaches for augmented reality (AR) that combine IR or LiDAR with visual-inertial odometry. Each approach produces a depth map showing the estimated distances across an image (see Fig. 1). Here, for each approach, we assess the reliability of distance estimation in the image center and periphery of such a depth map, as well as key usability metrics such as central processing unit (CPU) usage, battery usage, and field-of-view (FoV).

II. Materials and Methods

A. Experimental Protocol

Materials. Applications used – To examine the utility of different sensors and ML approaches, we altered open-source Apps to gather distance estimates from cameras/sensors which were on either the glass side facing the user ('self') or the back side ('back'). The Apps were: CoreML, iOS Depth Sampler, Real Depth Streamer, and Apple's ARKit 'fog' demo. Codes are available at: https://github.com/KOJILIU/DepthApps

CoreML: Depth prediction on iOS with CoreML is a software-based approach to generate depth values based on real-time RGB images captured by the camera. The ML model is a fully convolutional residual network based on ResNet 50 but provides up-sampling blocks to give higher resolutions with fewer parameters. CoreML is trained on the NYU Depth V2 dataset, which provides RGB images and their depth maps from a Microsoft Kinect RGB-Depth (RGBD) camera. CoreML outputs a 128 × 160 pixel resolution in meter units, at 24 frames per second (FPS), with a 12.47 ms (±1.11 ms) average image processing time (IPT) on the iPhone 13 Pro.

Real Depth Streamer: This provides live-streamed depth data from either the front-facing IR 'TrueDepth' camera (640 × 480, 24FPS, 7.14±0.71 ms IPT), or back-facing LiDAR camera (320 × 240, 24FPS, 6.14±1.41 ms IPT). These options are selected using '.builtInTrueDepthCamera' (and '.front' position) or '.builtInLiDARDepthCamera' (and '.back'). The self-facing TrueDepth camera projects a grid of IR dots which are detected using an IR camera. Spatial distortions in this grid indicate distances. The back-facing LiDAR scanner emits IR pulses and measures their reflection time. This time-of-flight measurement calculates distances. Irrespective of the sensor, the code provides depth values for the selected pixels in the 'f32Pixel' variable, with the data measured in meters. This provides the 'IR_self' and 'LiDAR_back' approaches.

iOS Depth Sampler: This provides a self-facing augmented reality session (ARKit) with face-tracking that primarily uses the TrueDepth camera. The depth map is generated in an augmented reality session using ARFrame, unlike 'IR_self' which live streams raw depth maps. Depth values (in meters) are in variable 'depthDataMap' (640 × 480, ∼60FPS, 18.74 ±11.08 ms IPT). This supports our 'ARKit_self' approach.

ARKit Fog Demo: This sample App is from Apple's ARKit environmental analysis documentation. It combines the back-facing LiDAR camera with room-tracking using visual-inertial odometry from depth-from-motion and inertial measurement unit [IMU] readings to create a stable 3D representation of the environment. Depth map values are stored in the 'sceneDepth' variable (256 × 192, 60FPS, 16.43±0.60 ms IPT) with distance in meters. This supports our 'ARKit_back' approach.

Procedure – To gather distance measurements, we ran each App on an iPhone 13 Pro, secured in a smartphone holder, placed at either 1, 2, or 3 meters from a solid white door at the end of a white corridor. The scene was simple, evenly lit, and largely symmetrical (see Fig. 1). The active pixel providing distance values was cast on the door. For the central condition, the central pixel in the image was cast onto the door. For the peripheral condition, the leftmost or rightmost pixel in the middle row of the image was selected and cast onto the center of the door by rotating the iPhone. Thirty distance values were recorded for each combination of approach and location within the first 5 seconds of use. A further comparison following 3 minutes of use is shown in supplementary materials. Accuracy and usability

TABLE I
MEAN VALUES OF EACH APPROACH FOR DISTANCE ESTIMATION ERROR, CPU USAGE, BATTERY USAGE, AND FIELD-OF-VIEW

Approach	Central Distance Error (cm) 1m	2m	3m	L/R Peripheral Distance Error (cm) 1m	2m	3m	CPU Usage (%)	Battery Usage (%) 10 min	30 min	60 min	Field-of-View (°)
CoreML	1.66	6.22	5.18	(7.16,9.31)	(64.99,34.22)	(83.29,75.57)	**44**	4	13	25	**52**
IR_self	1.30	1.48	1.96	(15.88,25.35)	(17.76,39.36)	(65.94,35.36)	50	**3**	**10**	**21**	40
LiDAR_back	1.36	0.87	**0.81**	(9.66,6.68)	(18.77,8.65)	(27.87,25.47)	48	7	20	40	40
ARKit_self	**0.86**	1.13	2.44	(4.81,9.88)	(13.04,23.65)	(28.54,30.23)	62	6	20	37	35
ARKit_back	1.37	**0.48**	1.40	**(7.08,4.75)**	**(1.88,23.73)**	**(7.21,20.78)**	74	5	14	27	30

Central and peripheral distance errors refer to absolute error values in cm relative to the ground truth distance (1 m, 2 m, 3 m). Underlined values indicate that the mean values were underestimations while regular values indicate overestimations. Bolded values indicate best-in-class performance for that specific metric.

data was gathered using Xcode v14. Analysis was done using IBM SPSS v29 and GraphPad PRISM v9.

III. RESULTS

The five distance estimation approaches were evaluated on metrics relevant to visual assistive technologies for PBLV, which include: (1) distance estimation accuracy in the central region; (2) distance estimation accuracy in the left and right peripheral regions; (3) CPU usage; (4) battery usage over 1 hour; and (5) field-of-view in the portrait orientation (Table I). Average errors are reported here in cm, while maximum errors, 90th percentile errors, and average errors expressed as a percentage are reported in the supplementary materials.

A. Central Distance Estimation

To evaluate the central pixel distance estimation accuracy, for each approach we compared 30 measurements of a door at the end of a corridor at 1, 2, and 3 meters from the iPhone camera/sensor. Comparing actual and estimated distances produced absolute error scores (cm). These were analysed using a five (approach) by three (distance) mixed ANOVA. Values reported for PostHocs are mean absolute error in cm±1 standard deviation, with Bonferroni-corrected statistical tests.

The ANOVA revealed a significant main effect of approach, $F(4,145) = 91.88$, $p<.001$, $\eta_p^2 = .717$, with PostHocs revealing that when averaging error sizes at the three distances, only CoreML (4.4±0.1) was significantly different, with higher absolute errors than IR_self (1.6±0.1), LiDAR_back (1.0±0.1), ARKit_self (1.5±0.1), and ARKit_back (1.1±0.1) ($p<.001$) and no other comparisons reaching significance. This finding indicates that this ML approach was significantly less accurate than alternative methods using depth sensors (see Fig. 2).

This ANOVA also revealed a significant main effect of distance, $F(1.61,232.99) = 20.75$, $p<.001$, $\eta_p^2 = .125$. This means that when all approaches are combined into an average score for each distance, the absolute error measured significantly differed across distances. PostHocs show significant differences with 1 m vs 2 m ($p<.001$), 1 m vs 3 m ($p<.001$), but not 2 m vs 3 m. This indicates that at 2-3 m, when combining approaches, the average absolute error scores did not vary significantly. Instead, only 2 m (2.0±0.1) and 3 m (2.4±0.2) had significantly larger mean errors than 1 m (1.3±0.1) overall.

The mixed ANOVA also revealed a significant interaction effect between approach and distance, $F(6.43,232.99) = 19.63$, $p<.001$, $\eta_p^2 = .351$. This indicates that changes in absolute error across different distances were not uniform across the different approaches. To fully investigate this effect, we conducted a series of follow-up ANOVAs for each approach across the three distances, and for each distance across all approaches. For comparisons of each approach across distances:

- *CoreML:* Significant effect of distance on error, $F(1.48,42.81) = 26.61$, $p<.001$, $\eta_p^2 = .479$, with 1 m (1.7±0.2) being significantly more accurate than 2 m (6.2±0.4) and 3 m (5.2±0.6) at $p<.001$.
- *IR_self:* there was no significant difference in absolute errors across different distances.
- *LiDAR_back:* Significant effect of distance on error, $F(2,58) = 9.54$, $p<.001$, $\eta_p^2 = .248$, with 1 m (1.4±0.1) being significantly more inaccurate than 2 m (0.9±0.1, $p = .003$) and 3 m (0.8±0.1, $p<.001$).
- *ARKit_self:* Significant effect of distance on error, $F(1.61, 46.69) = 10.73$, $p<.001$, $\eta_p^2 = .270$, with 3 m (2.4±0.4) being significantly more inaccurate than 1 m (0.9±0.2, $p = .001$) and 2 m (1.1±0.2, $p = .013$).
- *ARKit_back:* Significant effect of distance on error, $F(1.24, 35.88) = 29.77$, $p<.001$, $\eta_p^2 = .507$, with 2 m (0.5±0.1) being significantly more accurate than 1 m (1.4±0.0, $p<.001$) and 3 m (1.4±0.1, $p<.001$).

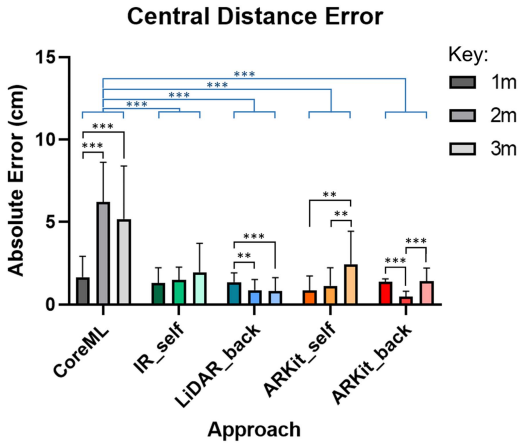

Fig. 2. Central distance error. Graph shows the average absolute error in distance estimation (in cm) for each of five approaches from the ground truth distance (1 m, 2 m, 3 m). Absolute error values summate the magnitude of all errors, irrespective of directionality (under/over-estimations). Higher values indicate larger distance estimation errors, with larger error bars indicating higher variability in mis-estimations. The results of statistical tests comparing approaches are shown in blue, and distances within each approach are shown in black. Error bars = 1SEM; ** = $p<.01$, *** = $p<.001$.

Overall, CoreML and ARKit_self became less accurate with distance, LiDAR_back became less accurate with proximity, ARKit_back was the most accurate at 2 m, and IR_self did not change significantly across the measured distances.

For comparisons of approaches at each distance:

- *1 m:* The approaches differ significantly in their accuracy, $F(4,62.89) = 2.89$, $p = .029$, with **ARKit_self** being the most accurate (0.9±0.9), and significantly more accurate than CoreML (1.7±1.3, $p = .045$) and ARKit_back (1.3±0.2, $p = .026$).
- *2 m:* The approaches differ significantly in their accuracy, $F(4,66.69) = 50.62$, $p<.001$, with **ARKit_back** being the most accurate (0.5±0.3), and significantly more accurate than CoreML (6.2±2.4, $p<.001$), IR_self (1.5±0.8, $p<.001$), LiDAR_back (0.9±0.6, $p = .037$), and ARKit_self (1.1±1.1, $p = .028$). Also, CoreML was significantly more inaccurate than all other approaches (all p's$<.001$), and LiDAR_back was significantly more accurate than IR_self ($p = .014$).
- *3 m:* The approaches differ significantly in their accuracy, $F(4,69.29) = 16.56$, $p<.001$, with **LiDAR_back** being the most accurate (0.8±0.8) and significantly more accurate than CoreML (5.1±3.2, $p<.001$), IR_self (1.9±1.7, $p = .018$), ARKit_self (2.4±2.0, $p = .002$), and ARKit_back (1.4±0.8, $p = .048$). CoreML was significantly more inaccurate than all other approaches (all p's$\leq.002$).

Overall, for each distance, a different approach was the most accurate, with ARKit approaches faring well at 1 m and 2 m, and LiDAR-based approaches being the most accurate at 3 m.

B. Peripheral Distance Estimation

To determine whether the level of accuracy is consistent across central and peripheral regions, for each approach we compared the average error in the periphery for all three distances against the average error in the center for all three distances using a series of paired t-tests:

- *CoreML:* $t(29) = 91.52$, $p<.001$, d = 16.7, $\text{Mean}_{\text{diff}} = 41.4$
- *IR_self:* $t(29) = 98.80$, $p<.001$, d = 18.0, $\text{Mean}_{\text{diff}} = 31.7$
- *LiDAR_back:* $t(29) = 128.31$, $p<.001$, d = 29.4, $\text{Mean}_{\text{diff}} = 15.2$
- *ARKit_self:* $t(29) = 91.21$, $p<.001$, d = 20.9, $\text{Mean}_{\text{diff}} = 16.9$
- *ARKit_back:* $t(29) = 74.39$, $p<.001$, d = 17.1, $\text{Mean}_{\text{diff}} = 9.8$

The results indicate that all approaches were significantly more accurate in the center relative to the periphery, with mean difference ($\text{Mean}_{\text{diff}}$) values indicating that CoreML had the largest difference between central and peripheral errors (41.4 cm) while ARKit_back was the most uniform with a 9.8 cm difference. A between-group ANOVA showed that the size of central-to-peripheral errors varied across approaches, $F(4,70) = 1883.93$, $p<.001$, with PostHocs showing that all approaches significantly differed from one another (all p's$<.001$). Asymmetric results for CoreML and IR_self are discussed in supplementary materials. Overall, ARKit_back is significantly more uniform in its accuracy relative to all other approaches with the lowest mean difference (see Fig. 3).

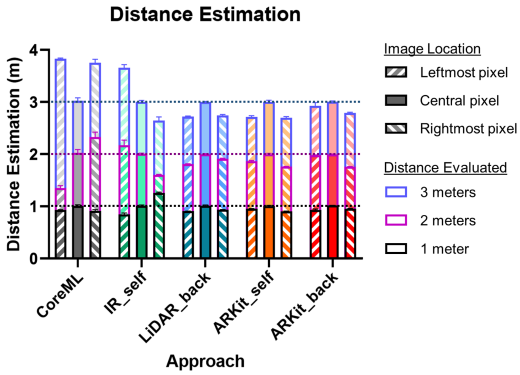

Fig. 3. Mean distance estimations for all approaches and distances (1 m, 2 m, 3 m) across central and peripheral (left, right) locations. Mean distance values are reported for the leftmost, central, and rightmost locations in the image while focused on a flat door surface at various distances from the camera. Color of border and dotted line indicate estimated and ground truth distances, respectively (1 m = black, 2 m = fuchsia, 3 m = blue). Error bars = 1SEM.

C. Descriptive Statistics: CPU Usage, Battery Drain, Field-of-View

While accuracy in estimating distances across the image is a key factor in evaluating the utility of different approaches, other aspects need to be considered for practical applications.

CPU usage is important to understand how computationally demanding a process is during distance estimation. This has implications for performance on more computationally limited smartphones, and how well they may integrate with other computationally demanding processes like object recognition and segmentation [9]. The average CPU usage for the initial 50 readings during operation for each approach are reported in Table I. It shows that CoreML was the least demanding process on smartphone CPUs (44%), with other sensor-only approaches (IR_self, LiDAR_back) at similar levels (50%, 48%). For ARKit processes, which add room or face tracking, CPU usage was higher (62%, 74%). Overall, CoreML and sensor-only approaches use less CPU resources than ARKit – leaving more resources for other processes helpful for PBLV.

Battery drain is important to consider for assistive Apps as smartphones serve as an all-in-one hub for safety, navigation, and productivity information for PBLV [10]. Starting from 100%, we tracked battery percentage while each App viewed the stimulus scene, tracking timepoints at 10, 30, and 60 minutes. It was found that IR_self, CoreML, and ARKit_back have the lowest battery consumption over 1 hour (21%, 25%, and 27% respectively). This is interesting because they have different underlying approaches via sensors, ML, and room-tracking. By contrast, the constant use of LiDAR or face-tracking results in stronger battery consumption. It is also interesting that battery usage is not in the same order as CPU usage. Despite LiDAR_back and ARKit_back both using the iPhone's LiDAR system, ARKit uses less battery. This may be due to ARKit using less LiDAR pulses and more room-tracking processes, which may be more battery efficient.

Field-of-View (FoV) is also a core consideration for visual assistive technologies. Narrower FoVs allow fewer objects in an image with reduced context, and require users to be more precise with the camera/sensor to capture specific objects in the image. Different distance estimation approaches can vary in their FoVs due to the camera/sensor used or supplementary processes like room-tracking. In the portrait orientation, ARKit approaches had the narrowest FoV at ∼35°, while sensor-only approaches had slightly wider FoVs at 40°. Since CoreML uses the standard iPhone camera, it had a much wider FoV at 52°. CoreML offers the widest FoV by default, does not require additional sensors, and has the option of even wider FoVs by using a wide-angle camera/lens. However, substantially increasing the FoV creates image distortions, which could reduce the similarity between live and training images from the NYU Depth V2 dataset, which may degrade distance estimation accuracy. Further performance metrics of CoreML in naturalistic scenarios and at different visual angles (30°, 35°, and 40°) are reported in supplementary materials.

IV. Discussion

We found that LiDAR and both ARKit approaches have the highest accuracy for distances. Even though CoreML is the most inaccurate approach at all distances in the center, the size of inaccuracy (1.7, 6.2, and 5.1 cm) is small enough that it should still effectively assist many activities. CoreML also fares well in terms of usability factors and is unique in not requiring the use of additional sensors (IR, LiDAR, IMUs) to estimate depth, but instead only uses a standard RGB image input. This makes local ML approaches viable on a wider range of smartphone hardware, and opens the door to conducting these ML processes remotely. RGB images can be uploaded to the cloud for processing, with distances, objects, and potentially feedback reported back to the App. While cloud processing requires network connectivity and bandwidth, and adds network latency, it can still be beneficial for users in terms of battery or CPU usage, accuracy, and even overall latency (e.g., 84 ms) when local systems cannot perform the same computations within a time-efficient manner [11]. Finally, it should also be noted that our results reflect the performance of approaches available at the time of testing, and that further hardware or software revisions by the developers may alter their accuracy and usability metrics in the future.

V. Conclusion

We evaluated a variety of distance estimation approaches on smartphones using metrics relevant to visual assistive technologies. For estimating distances in the image center, all approaches were highly accurate at 1–3 m. However, machine learning methods on RGB images such as CoreML became significantly more inaccurate at 2 m and 3 m relative to other approaches. All approaches were significantly more inaccurate in the periphery, with CoreML and self-facing IR showing the greatest increases in error. ARKit and LiDAR were the most accurate approaches in both the image center and periphery. As such, ARKit and LiDAR approaches may be the best choice for assistive technology development when spatial accuracy is a top priority. However, CoreML had several other advantages, with the lowest CPU usage, second lowest battery consumption, and highest FoV, without the need to rely on IR or LiDAR sensors. Here, machine learning approaches may be the best in terms of accessibility to the user, on a wider range of smartphones or even remotely by using cloud processing on smartphone RGB images. Overall, we show the strengths and weaknesses of a variety of distance estimation approaches and discuss their implications. These findings can help guide the development of visual assistive technologies to effectively and reliably deliver information of key interest to blind and low vision communities, all accessible on modern smartphones.

Supplementary Materials: In the supplementary materials, we show: (1) CoreML's peripheral accuracy at additional visual angles used by other distance estimation approaches (30°, 35°, 40°); (2) maximum FPS and average IPT for all approaches; (3) accuracy of all approaches in the first 5 seconds vs. after 3 minutes of continual use; (4) maximum error and 90th percentile errors for all approaches, sides, and distances; (5) CoreML's accuracy for natural scenes; (6) discussion of asymmetric inaccuracies for CoreML and IR_self; and (7) estimation errors of the white door expressed as a percentage of total distance.

*Conflicts of Interest:*The authors declare no financial interests related to the subject of the manuscript.

Authors' Contributions: Study conception and design: G.H., M. L., J. R. R., K.C.C.; Data collection: G.H., M. L.; Data analysis and interpretation: G.H., M. L., J. R. R., K. C. C.; Manuscript writing: G.H., M. L., D. S., C. F., T. E. H., J. R. R., K. C. C.. All authors read and approved the final manuscript.

References

[1] C. Jicol et al., "Efficiency of sensory substitution devices alone and in combination with self-motion for spatial navigation in sighted and visually impaired," *Front. Psychol.*, vol. 11, 2020, Art. no. 1443.

[2] M. Auvray, S. Hanneton, and J. K. O'Regan, "Learning to perceive with a visuo—Auditory substitution system: Localisation and object recognition with 'the voice,'," *Perception*, vol. 36, no. 3, pp. 416–430, 2007.

[3] S. Maidenbaum et al., "The 'EyeCane', a new electronic travel aid for the blind: Technology, behavior & swift learning," *Restorative Neurol. Neurosci.*, vol. 32, no. 6, pp. 813–824, 2014.

[4] G. Hamilton-Fletcher, T. D. Wright, and J. Ward, "Cross-modal correspo ndences enhance performance on a colour-to-sound sensory substitution device," *Multisensory Res.*, vol. 29, no. 4/5, pp. 337–363, 2016.

[5] G. Hamilton-Fletcher and K. C. Chan, "Auditory scene analysis principles improve image reconstruction abilities of novice vision-to-audio sensory substitution users," in *Proc. IEEE Eng. Med. Biol. Soc.*, 2021, pp. 5868–5871.

[6] G. Hamilton-Fletcher, M. Obrist, P. Watten, M. Mengucci, and J. Ward, "'I always wanted to see the night sky': blind user preferences for sensory substitution devices," in *Proc. CHI Conf. Hum. Factors Comput. Syst.*, 2016, pp. 2162–2174.

[7] G. Hamilton-Fletcher et al., "SoundSight: A mobile sensory substitution device that sonifies colour, distance, and temperature," *J. Multimodal User Interfaces*, vol. 16, no. 1, pp. 107–123, 2022.

[8] N. Martiniello et al., "Exploring the use of smartphones and tablets among people with visual impairments: Are mainstream devices replacing the use of traditional visual aids?," *Assistive Technol.*, vol. 34, no. 1, pp. 34–45, 2022.

[9] "Apple models – machine learning – Apple developer," Jan. 31, 2023.[Online]. Available: https://developer.apple.com/machine-learning/models/

[10] K. Locke et al., "Developing accessible technologies for a changing world: Understanding how people with vision impairment use smartphones," *Disabil. Soc.*, vol. 37, no. 1, pp. 111–128, 2022.

[11] Z. Yuan et al., "Network-aware 5G edge computing for object detection: Augmenting wearables to 'see' more, farther and faster," *IEEE Access*, vol. 10, pp. 29612–29632, 2022.