Translating Pedestrian Indoor Images Into Maps

Manthan Joshi
School of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, USA
mjoshi78@gatech.edu

Ethan Haarer
College of Computing
Georgia Institute of Technology
Atlanta, USA
ehaarer3@gatech.edu

Saba Karimi
School of Materials Science and Engineering
Georgia Institute of Technology
Atlanta, USA
skarimi30@gatech.edu

***Abstract*—Obtaining a Bird's Eye View (BEV) map of the first-person perspective is a critical task in enabling effective mobile robot navigation in dynamic and crowded environments. This requirement could be extended to not just drones, but also marine bots. As such, having an effective knowledge of the surroundings of the robot does help create intuitive and understandable scene maps for human comprehension. Numerous research efforts have tackled the First Person View (FPV) to BEV conversion challenge leveraging various state-of-the-art learning algorithms heavily based on the convolutional neural networks. However, many of these approaches are heavy on computations and require a powerful rotating LiDAR sensor to construct a 3-dimensional view of the scene.**

In this study, we present two approaches to realize the FPV to BEV conversion of an image which could very well be extended to process videos to benefit from the realized real-time performance. Our approach utilizes minimalistic intrinsic parameter considerations and inverse pixel projection in one approach and leverages the depth map obtained from an iPhone 12 Pro clicked- 2-dimensional FPV image in the other. We then gather ground truth data in a real-world environment by ground marking measured distances and compare the performance of each of these approaches against the real-world data collected.

We qualitatively establish that both these methods give approximately similar results with error margins of plus-minus 2 feet. We also form a covariance ellipsoid plot explaining the relation between the ground truth and the estimated distances for one of these methods. We thus report the results of our graphical visualization of the algorithms' performances, demonstrating their potential for real-world applications.

***Index Terms*—FPV, BEV, LiDAR, depth map, inverse pixel projection**

I. Introduction

In the realm of computer vision, the interpretation of complex visual scenes through automated systems has been a focal point of significant research efforts. Among these, object detection remains a critical foundational task, wherein algorithms aim to identify and locate objects within a 2-dimensional image with high precision. This capability not only powers applications ranging from autonomous driving to surveillance but also serves as a stepping stone for more intricate visual understanding tasks.

Coupled with object detection, depth estimation introduces an additional layer of sophistication for an accurate understanding of crowded scenarios by determining the distance of each object from the observer device. This is achieved through various methods such as stereo vision, structured light, or learning-based approaches using convolutional neural networks. Depth estimation provides the spatial context necessary for accurately interpreting scenes, which is crucial for tasks requiring a three-dimensional understanding from 2D inputs. The availability of a decentralized system with multiple agents equipped with vision sensors establishes a very powerful method of understanding a 3-dimensional crowded scene wherein the inputs are a number of 2-dimensional images.

The culmination of these technologies is embodied in the top-down mapping module, which transforms the input from a first-person view into a comprehensive top-down perspective for all the objects as they are detected in the device's perspective. This transformation is pivotal for applications such as robotics navigation, where a top-down map significantly enhances the robot's ability to plan and execute movements within an environment while making the path-planning process intuitive to the human brain. By integrating object detection and depth estimation, the top-down mapping module synthesizes a navigable, bird's-eye view map that may highlight both-the geometry as well as the semantics of the environment, enabling more effective spatial reasoning and decision-making, especially in systems where human beings are supposedly involved in the loop.

Together, these three modules - object detection, depth estimation, and top-down mapping - form a general framework for understanding and navigating complex visual environments based on 2D perspective input images. Nonetheless, the integration of these modules not only advances the field of robot automation but also opens up new possibilities in the field of computer vision at large.

II. Related Work

YOLO is a CNN-based object detector that predicts bounding boxes and class probabilities for objects in images. It efficiently propagates low-level features through deep convolutional layers, enabling feature extraction. YOLO operates by making predictions within grid cells, eliminating the need for a separate region proposal step and enabling real-time processing. While less accurate than two-stage detectors, which include a region proposal network followed by classification, YOLO is computationally less intensive. [1].

Depth estimation enhances 3D understanding and acts as a link between 2D and 3D environment. Monocular depth

estimation (MDE) provides depth information from a single image, serving as a cost-effective alternative to traditional technologies like LiDAR. Supervised learning formulation of depth estimation has been explored via Markov Random Fields [2], [3], and pixel-level regression using CNNs [4]–[8]. When trained on a sufficiently large-scale dataset, the models learn the joint statistics of scene geometry and appearance. However, they require depth ground-truth, commonly expensive to acquire, while also generalizing poorly to unseen scenes. Unsupervised methods prove CNN-based depth and ego-motion networks can be trained solely on monocular video [9], or stereo images [10], [11]. Self-supervised approaches also emerged, utilizing a variety of supervisory signals, e.g., photometric loss, [10], online refinement [12] and network architecture design [13]. Indoor environments presents unique challenges compared to outdoors: (1) They exhibit varied depth across frames, unlike outdoor scenes where the maximum distance (sky) often remains consistent. (2) They involve complex camera ego-motion, demanding robustness to arbitrary poses and scene complexities, unlike translational motion in driving with a fixed camera on the vehicle. (3) Indoor scenes lack strong structure priors, with irregularly arranged objects. (4) Large untextured surfaces indoors, like walls and carpets, hinder photometric loss-based training due to less meaningful supervision. [14]–[16]

To address challenges, Moving Indoor [14] proposes an unsupervised training paradigm to handle textureless regions and camera ego-motion by optical flow targets supervised by a sparse-to-dense flow estimation network. P^{2}Net [17] combines a point with its local window and minimizing patch-based multi-view photometric consistency error, while also leveraging superpixels to extract homogeneous-color regions. ViTs replaced convolutional networks as the encoder backbone in Dense Prediction Transformer (DPT) [18] to address the loss of feature resolution and granularity caused by downsampling. The global receptive field of the transformer and the consistent dimensionality across stages contribute to creating fine-grained and globally coherent depth predictions.

In the self-supervised realm, MonoIndoor [15] factorizes depth into global and relative depth maps, and a residual pose estimation that adapts the model to changes in depth scale in training, and improves rotation prediction accuracy. DistDepth [16] is a metrically accurate depth for zero-shot cross-dataset generic indoor scenes in real-time that combines a relative depth estimator with learning metrics from left-right consistency. Structural regularities and co-planar constraints are levereged as supervision in [19], increasing the accuracy of predicted depth. MiDaS [20] introduced a loss functions invariant to incompatibility between datasets. In MiDaS v3.1 [20], the impact of integrating the SOTA pre-trained vision backbones on depth estimation quality and runtime is explored. More recently, Depth Anything [21] proposed high-quality depth estimation via coupling a large public unlabeled dataset with a smaller labeled one, using a rigorous optimization target and auxiliary semantic segmentation supervision.

III. Problem Statement

Navigation of autonomous ground robots through dense environments with different static and dynamic objects requires the bot to have an idea of the unknown terrain momentarily before making navigating decisions. As such, the perspective camera view to top-down mapping helps plan the trajectory that is intuitive to a human being.

We strive to collect ground truth data and test the performance of the proposed model for estimating object distances from a stationary ego agent to achieve intuitive top-down mapping of the ego agent's perspective of the 3D scene. We then qualitatively visualize the ellipsoid plots addressing the covariance of the collected ground truths against the estimated values.

Let X be the ground truth and Y be the estimation. The covariance matrix Σ for these variables is defined as:

$$\Sigma = \begin{pmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{pmatrix}$$

where:

- $\sigma_{xx} = \text{Var}(X)$ is the variance of the ground truth,
- $\sigma_{xy} = \sigma_{yx} = \text{Cov}(X, Y)$ is the covariance between the ground truth and the estimation,
- $\sigma_{yy} = \text{Var}(Y)$ is the variance of the estimation.

We plot the Cov(X, Y) as ellipsoid plots to visualize the difference in the estimates of the actual distances such that, solving to achieve $sigma_{xy}$ lesser than but nearly close to $sigma_{xx}$ and $sigma_{yy}$ which would qualitatively mean the covariance scatter plot of the considered variables will be along Var(Y) = Var(X) axis.

Given the dynamics of the camera, a number of research ventures strive to solve the aforementioned problem for time-varying systems, but, by leveraging the use of extrinsic and intrinsic parameters along with LiDAR data of the captured surroundings along with deploying heavy DL algorithms [translating images into maps]. Also, the estimation is subjected to the constraints on the coverage operability of the LiDAR sensor.

Considering this constraint, we in this research, propose two pipelines to solve the mentioned problem statement for monocular and stereo input images with minimal use of mentioned techniques. We first propose a model based on statistically estimated parameters of the homogeneity matrix to obtain object coordinates. We then utilize only the iPhone LiDAR data along with certain mathematical considerations for obtaining plottable 2-dimensional coordinates to realize a top-down map of the captured scene.

IV. Methodology

The two approaches we explore are: Statistical Estimation of Homoginity Matrix Constants for Inverse Pixel Projection and LiDAR-based depth mapping.

A. Statistical Estimation of Homoginity Matrix Constants for Inverse Pixel Projection

1) Depth Estimator: We first employ YOLOv5 pre-trained weights and configurations to detect objects of interest. We chose YOLOv5 as it is a considerably efficient model for the task of object detection. Also, realizing results with YOLOv5 establishes the fact that later versions of YOLO will perform on par with, if not better than the model under consideration, all while limiting memory usage.

Next, to seamlessly obtain mapped obstacle distances, we learn the diagonal distance of an object from the camera and the midpoint of the camera axis, based on pixel projectors to realize object positions in the real world. Statistical Inference of homogeneity matrix constants based on the distance-size indirect proportionality hypothesis states that a person of average known height will appear smaller as the distance of the person/ people increases from the camera. Having dynamically calculated the image resolution using the OpenCV package, the distance of the bounding box is estimated based on the hypothesis considered. The image is divided into two halves based on the midpoint on the x-axis of the image, and the horizontal distance of each detected object from the 90-degree line passing through the midpoint of the image is calculated to then know the diagonal distance of the object from the device based on the Pythagoras formula.

In Fig. 1, the pixel projector block encapsulates the working as explained above once the object(s) are detected and bounding box coordinates are obtained. The coordinates are then mapped in a top-down map (TD map).

The homogeneity matrix incorporating intrinsic as well as extrinsic parameters is typically denoted as:

$$H = \begin{bmatrix} h_{11} & h_{12} & h_{13} & h_{14} \\ h_{21} & h_{22} & h_{23} & h_{24} \\ h_{31} & h_{32} & h_{33} & h_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Here, as the camera is still, the use of extrinsic parameters is discarded as the camera is not rotating or translating. The intrinsic parameters including focal length were considered for a pixel 3a camera which is realized to capture FPV images.

As shown below, with a statistical estimate of the constants that should be obtained by considering the homography matrix of the pixel 3a device, the distance of the object from the horizontal camera axis is obtained. This is done in tandem with obtaining the detected object coordinates with every frame captured.

Computing bounding box x co-ordinate incorporating the corresponding statistically estimated intrinsic constant:

$$x = \frac{(x_1 + x_2)}{2} - K_1 \tag{1}$$

The equation for calculating diagonal distance then, is:

$$\text{diagonal distance} = \frac{K_2}{(y_2 - y_1)} \tag{2}$$

The equation for calculating vertical distance then, is:

$$\text{vertical distance} = \sqrt{\frac{\text{diagonal distance}^2}{1 + \left(\frac{x}{K_3}\right)^2}} \tag{3}$$

The equation for calculating distance from the midpoint then, is:

$$\text{distance from midpoint} = \left(\frac{\text{vertical distance}}{K_3}\right)^x \tag{4}$$

where: K1, K2, and K3 are intrinsic constants estimated to be 800, 7500, and 1300 respectively, and x1, x2, y1, and y2 are bounding box coordinates for each object detected.

In the proposed method, we use a Google pixel 3a to capture images and collect ground truth by tape-marking distances in a well lit indoor setup containing flooring tile dimensions of 2x2 feet.

With angle and polar coordinate considerations, the distance of the objects from the camera is achieved and augmented for representation.

B. LiDAR for Depth Mapping

For comparison with the first pipeline wherein the depth is estimated based on the statistically estimated parameters to the homogeneity matrix, we lay a pipeline leveraging an iPhone 12 pro-LiDAR sensor-equipped camera with a coverage of 5 meters for obtaining the depth map.

Algorithm 1 Object Detection, Depth Estimation, and Top-Down Mapping

```
Object Detection:
Load YOLO network model
Preprocess input image
Run forward pass to detect objects
Apply Non-Maximum Suppression
Depth Estimation:
Extract depth information from depth map
Compute depth for each detected object
Top-Down Mapping:
Compute position of detected objects
Plot top-down map =0
```

Algorithm 1, above, shows the holistic working of the pipeline that leverages a depth map for distance estimation. With angle and polar coordinate considerations, based on the segmented depth, a distance of the objects from the camera is achieved.

1) Object Detector: Given the additional complexity of the algorithm, in order to realize the objective of maintaining the pipeline light on memory, We here leverage the lightweight YOLOv4 pre-trained weights for object detection and map the depth of the iPhone 12 Pro captured still leveraging the LiDAR sensor that the device has.

The RGB image and the corresponding depth map are pre-processed to ensure the alignment is synchronized. The

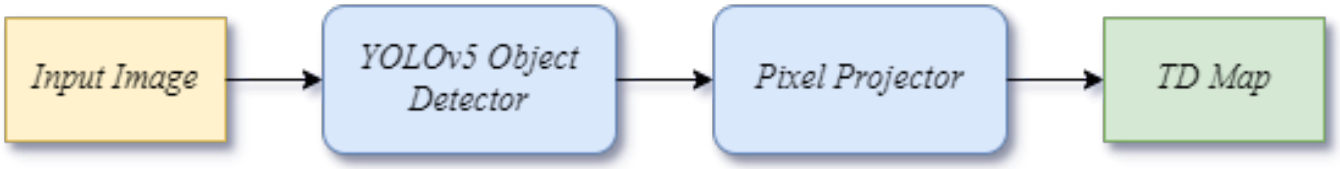

Fig. 1. Pipeline leveraging intrinsic parameters for top-down mapping based on inverse pixel projection

object(s) in the image are detected with the YOLO object detector wherein the object ID is set to 0 to exclusively identify stationary or dynamic pedestrians in the still.

For each detected object, the bounding boxes are operated upon such that 2 main calculations are performed over this data as elaborated herewith.

2) Distance and Angle Estimator:

a) Grayscale Homogenization: A single detected object was seen to show differing distance readings in real-world scenarios. We considered 2 approaches to deal with the situation. The most common approach is the averaging of pixel values. We, however, avoid this approach to account for possible errors in YOLOv4 detection while encapsulating the environment around each object into the detecting bounding box. This approach seemed to condition the model to overshoot the estimated distance from the camera which does not hold good with the ground truth data collected.

Algorithm 2 Algorithm for processing depth maps based on bounding boxes

boxes, depth_map Process boxes to find mode depth values from a depth map
for each box in boxes **do**
  Assign dimensions x, y, w, h from box
  if $w > 0$ & $h > 0$ **then**
    EXTRACT subsection of depth map from dimensions
    CALCULATE mode value in subsection
    if a most frequent value exists **then**
      ASSIGN this value to $mode\_depth$
    else
      CONTINUE to the next box (no frequent value found)
    end if
  else
    CONTINUE to the next box (dimensions are zero)
  end if
end for

Alternatively, a method for homogenizing grayscale values within boxes to determine depth in an image is calculating the $mode$ of the value as shown in Algorithm 2. The approach involves finding the most frequently occurring grayscale value within each box to estimate the depth of the object bounded by the box under consideration. This method aims to mitigate errors, providing depth readings with an error margin of ±2 feet for each box containing detected objects.

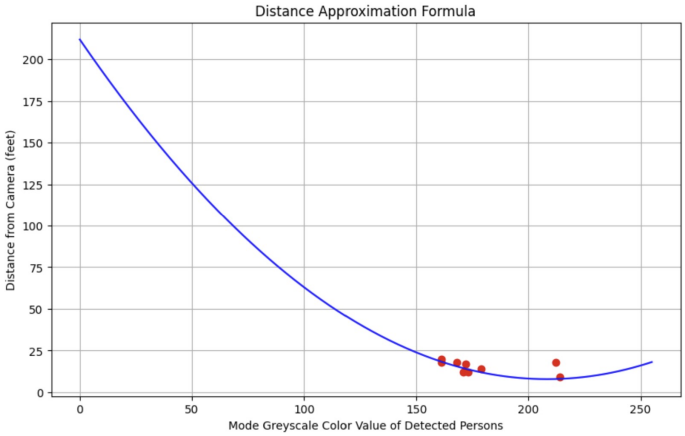

Fig. 2. This graph depicts our distance approximation function, generated through calibrating the lidar depth estimator via mapping the mode of a person to their ground truth distance. A handful of calibration points are also graphed.

To estimate the distance of the object from the camera, the obtained grayscale value for each of the detected objects is then fed into a 2nd-degree polynomial function as seen in similar studies where a quadratic best-fit function was found to best represent LiDAR depth maps. An observation though is that the object being too close to the camera led the model to overshoot the distance estimation. It should be noted that the best-fit function as seen in Fig. 2 was refined through calibration testing based on the non-linear regression conducted against ground truth values, and is shown below, where x is the calculated mode and y is the predicted distance:

$$y = (4.7 * 10^{-3})x^{-2} - 1.9582x + 211.833 \quad (5)$$

b) Estimating Diagonal Object Distance(s): A major challenge for top-down mapping is not just estimating the distance of the object from the horizontal axis of the camera but also estimating the angle of the object from the point of situation of the camera.

Considering the field of view of the camera, we leverage geometrical constructions, to account for the angular distance of the object from the device.

A vertical column, across each of the bonding boxes, is situated at 90 degrees from the horizontal axis through the midpoint of the bounding box. We then have this center line divide the image in two. Knowing the angle by which the camera captures images, a one-to-one mapping of the percentage of the pixels in the image on either side of the line

to the percentage of the total angle captured by the camera is calculated. This is used as the angle by which the polar coordinate with respect to the camera is generated. This is as seen in the equation below. This is also based on the mathematical consideration that the number of pixels is either known or dynamically calculated by utilizing the OpenCV package.

$$\text{Angle} = \frac{1}{2}\text{FOV}\left(\frac{X_{\text{Center Line}} - \frac{1}{2}\text{Image Width}}{\frac{1}{2}\text{Image Width}}\right)$$

Further expounding upon the formula: in order to find the angle for every person in a given image, we first find the center of each box, then use the x-coordinate to create a vertical line that perfectly splits every box into two. For each line, we find the percentage of pixels away from the leftmost edge of the image which is, then subtracted by ½ the image width and multiplied by ½ the FoV of the camera.

This is done to convert the proportion of pixels on either side of a given line into the percentage of the total FOV captured by the image.

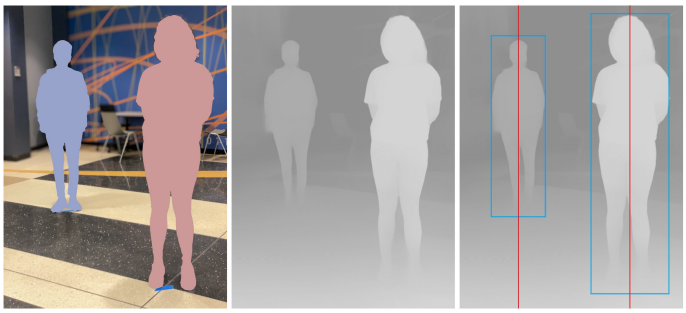

Fig. 3. Above illustrates the input (individuals masked due to privacy concerns) and processing done by our model. The first two images, the RGB image and Associated depth map are fed in. The pretrained weights of YOLOv4 identify and draw boxes around each person (depicted in blue on the rightmost image) and then for each box a center bisecting line is generated (depicted in red) to estimate the angle of the person with respect to the camera orientation.

For an iPhone 12 Pro in portrait mode, the vertical FoV is known to be 45 degrees, effectively this angle is used as the basis for setting the angular constraints for obtaining estimation for iPhone 12 Pro. Offset in the image, is also accounted for, based on this consideration.

Knowing both the object angle from the camera and the distance of the object relative to the camera, we then use these to generate a polar coordinate about the device. In single still stereo captures, the camera is by default considered to be at (0, 0). It should be noted that we further rotate these results by 90 degrees counterclockwise to achieve a more intuitive frame of reference in our generated BEV map, such that the camera frame denotes the FPV. These calculations are visualized in Fig. 3, and Fig. 4. The angle is calculated with respect to the center of the image for ease of translating the points to top-down mapping.

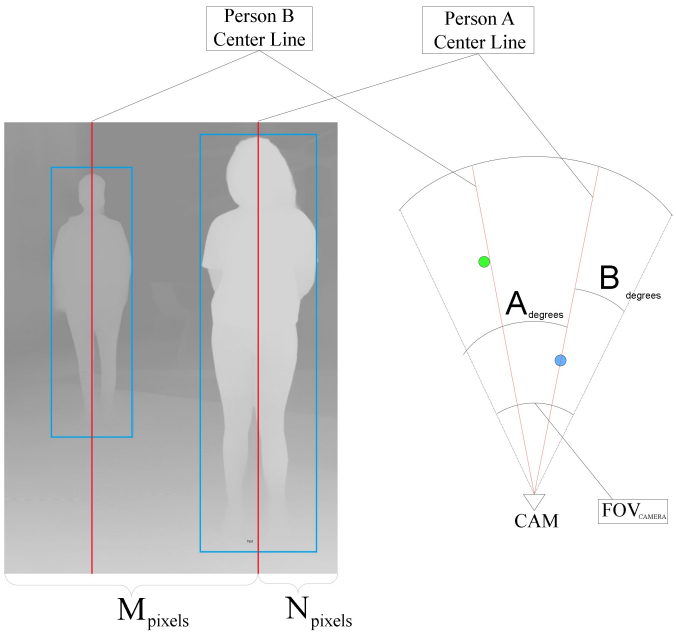

Fig. 4. For every box generated by YOLOv4 we must estimate the angle of each individual. By taking the center bisecting line of each individual, we know that the ratio of pixels M:N is equal to the ratio of A:B, allowing us to convert the pixels into approximating a person's angle in the camera's FOV.

V. Experiments & Results

A. Distance Estimation Based on Statistical Constant Inference for Pixel Inverse Projection

Using the ground truth information collected, as an evaluatory metric, we obtain the accuracy of the proposed pipeline. The average accuracy obtained was 95.736 percent with an observed average loss of 0.3048 meters for well-lit, indoor settings.

The pipeline plots the object distance in approximately 173.8 ms and the estimator requires no training which can also process videos to obtain object distance from the camera in every timeframe in the real world.

From Fig. 5, ground truth diagonal distance readings are- person 1 is 11 ft., person 2 is 24 ft., and person 3 is 19 ft. The obtained distance estimates are- person 1 is 10 ft., person 2 is 23ft., person 3 is 20 ft., approximately. The estimates are verified using Pythagoras theorem and visualized as seen in Fig. 6.

By running the YOLOv5 in parallel with the depth estimator using LiDAR, the readings are $<$ 94 percent accurate and are at par when the pixel projection technique is employed to obtain object coordinates.

The above figure is obtained by augmenting the perspective FPV to BEV and the depth is estimated and plotted. We establish that the FPV to BEV for an image or a video can be achieved by employing pixel projection and depth coordinate mapping for the detected objects to estimate the precise distance from the camera. Next, we research later versions of YOLO to lighten the algorithm further in this approach.

Fig. 5. Estimated coordinates for picture clicked in pixel 3a

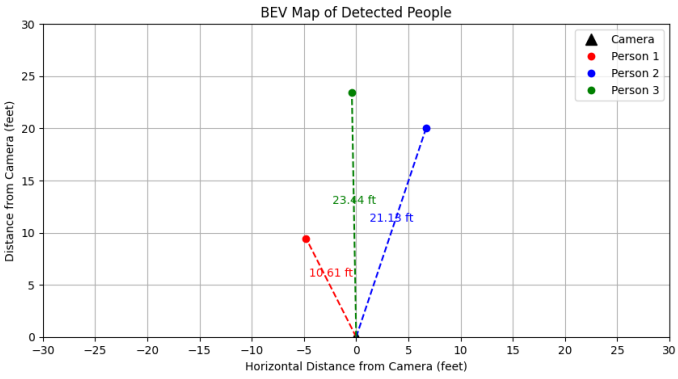

Fig. 6. TD view based on inverse pixel projection used for DE

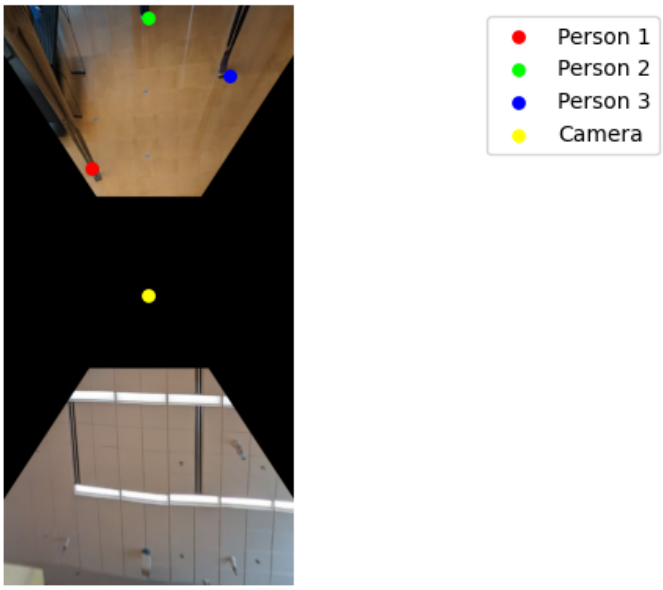

Fig. 7. Augmented TD view with estimated coordinates based on inverse pixel projection

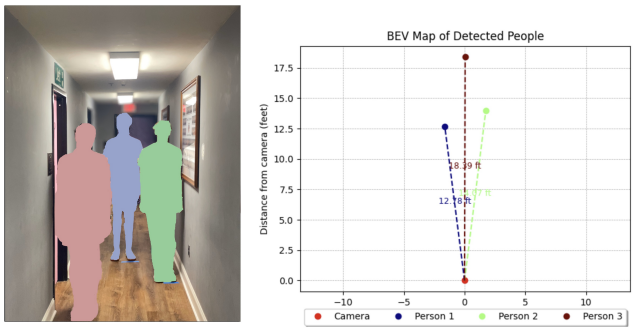

Fig. 8. These images represent the experimental testing of our model with 1, 2, and 3 people in frame. The ground truth for image 1 is 12 feet from the camera, for image 2 12 feet and 9 feet for person 1 and 2 respectively, and 14 feet, 17 feet and 20 feet for each respective person in image 3. We see the error in distance is all within 3 feet of the ground truth.

B. Distance Estimation Based on LiDAR Depth

Before starting formal tests, we needed to calibrate the LiDAR by correlating the grayscale values from ground truth images to estimated distances to create a model that predicts distance based on these grayscale values.

We found that the greatest error for the model was found within single-person images, in which the model repeatedly severely overshot or undershot the distance to the individual person. This is similar to other distance estimators, including MiDaS, where a lack of a frame of reference between different detected entities within the image led to inconsistent readings. A possible contributor to this error could very well be the employment of YOLOv4. We plan to utilize YOLOv5 and later versions to check for improvement in the performance.

The model is most adept at predicting the bird's eye position of two or more people in the image. As seen in Fig. 8 we see that the model is consistent with the Cartesian approximation of each individual within 1-2 ft of the ground truth measurement. Additionally, Fig. 8 also shows the robustness of the model in dealing with an overlap of persons, such that it is able to accurately guess the distance of person 3 despite the fact they are partially obscured by person 1.

Throughout our tests, despite the LiDAR providing further real-world data to calibrate our tests, we found that differing light conditions, especially the introduction of natural sunlight would contaminate the device readings and thus, provide relatively inaccurate depth maps. In particular, the distance estimation of those individuals furthest away from the camera is affected the most. Furthermore, the camera used in the study has an effective lidar range of 5 meters. This limited the range of tests we supposedly performed along with conducting even the model calibration. Both these issues warrant continued study.

Fig. 9, shows that though there exist some data point outliers, the majority of the scattered data points lie along the axis Var(Y) = Var(X). The error in estimation, however,

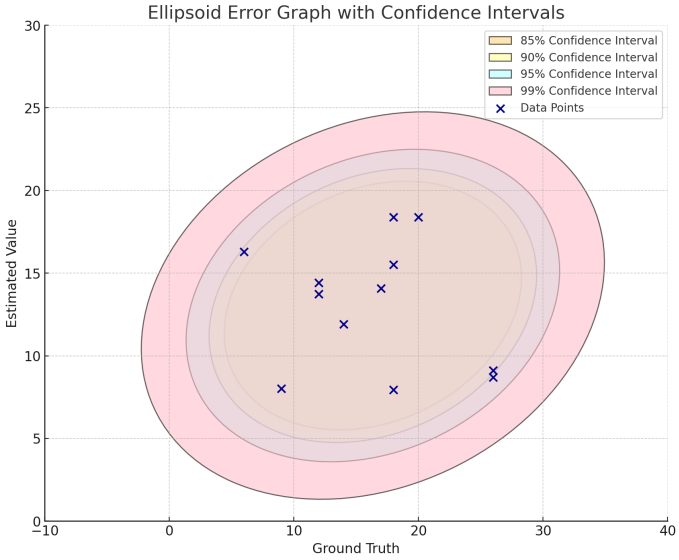

Fig. 9. Ellipsoid graph for TD view based on depth map formed using LiDAR

is within acceptable limits. This denotes a strong positive linear relationship with a high covariance and qualitatively symbolizes that the changes in the values in Var(X) are closely mirrored by Var(X) data points.

This also means that the problem of close to accurate distance estimation can be tackled with minimal use of techniques. That is: either by utilizing a LiDAR-based depth map or by estimating the Homoginity matrix constants.

VI. Discussion

For a generalized case of the FPV to BEV still or movie conversion, focusing on the locations of the agents in the capture, the quickness in pre-processing, processing, inference, and NMS ensures the mapping is achieved for the objects within the device frame, at all times irrespective of the resolution.

Qualitatively, it can be inferred that the algorithms works close to accuracy. Thus, we establish that the FPV to BEV for an image or a video can be achieved by employing inverse pixel projection and depth coordinate mapping for the detected objects to estimate the precise distance from the camera.

To build upon the current work, as future scope of the study, we seek to account for persons captured in the LiDAR depth map that may not be facing the camera straight on or may be greatly obscured by the environment or other persons, which leads the center of the box to not correlate to the true center of the individual. To address this issue, we wish to integrate segmentation into the model to identify the individual parts of a person to better approximate their center, and thus their angle with respect to the camera. As such, even the requirement of an established decentralized system with multiple agents equipped with vision sensors is rendered unnecessary.

Additionally, the efficient use of mathematical concepts and state-of-the-art computer vision algorithms make way for the realization of light model structures. This is sought to be achieved to enable model integration in devices with humble memory sizes and processing powers. As the domain of AI continues to evolve; the research in enabling deep learning model runs, on smaller devices will seek a rise in scope and application. This will greatly benefit studies such as ours to realize lighter solutions that may be utilized in refining applications such as ground robot navigation.

References

[1] A. Vijayakumar and S. Vairavasundaram, "Yolo-based object detection models: A review and its applications," *Multimedia Tools and Applications*, pp. 1–40, 2024.

[2] A. Saxena, S. Chung, and A. Ng, "Learning depth from single monocular images," in *Advances in Neural Information Processing Systems* (Y. Weiss, B. Schölkopf, and J. Platt, eds.), vol. 18, MIT Press, 2005.

[3] A. Saxena, M. Sun, and A. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE transactions on pattern analysis and machine intelligence*, vol. 31, pp. 824–40, 06 2009.

[4] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," 2015.

[5] R. A. Güler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, and I. Kokkinos, "Densereg: Fully convolutional dense shape regression in-the-wild," 2017.

[6] H. Fu, M. Gong, C. Wang, K. Batmanghelich, and D. Tao, "Deep ordinal regression network for monocular depth estimation," 2018.

[7] J. Jiao, Y. Cao, Y. Song, and R. Lau, "Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[8] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab, "Deeper depth prediction with fully convolutional residual networks," 2016.

[9] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," 2017.

[10] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 270–279, 2017.

[11] Z. Yang, P. Wang, Y. Wang, W. Xu, and R. Nevatia, "Every pixel counts: Unsupervised geometry learning with holistic 3d motion understanding," in *Proceedings of the European conference on computer vision (ECCV) workshops*, pp. 0–0, 2018.

[12] Y. Chen, C. Schmid, and C. Sminchisescu, "Self-supervised learning with geometric constraints in monocular video: Connecting flow, depth, and camera," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 7063–7072, 2019.

[13] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[14] J. Zhou, Y. Wang, K. Qin, and W. Zeng, "Moving indoor: Unsupervised video depth learning in challenging environments," 2019.

[15] P. Ji, R. Li, B. Bhanu, and Y. Xu, "Monoindoor: Towards good practice of self-supervised monocular depth estimation for indoor environments," 2021.

[16] C.-Y. Wu, J. Wang, M. Hall, U. Neumann, and S. Su, "Toward practical monocular indoor depth estimation," 2022.

[17] B. Wang, C. Chen, Z. Cui, J. Qin, C. X. Lu, Z. Yu, P. Zhao, Z. Dong, F. Zhu, N. Trigoni, and A. Markham, "P2-net: Joint description and detection of local features for pixel and point matching," 2021.

[18] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," 2021.

[19] B. Li, Y. Huang, Z. Liu, D. Zou, and W. Yu, "Structdepth: Leveraging the structural regularities for self-supervised indoor depth estimation," 2021.

[20] R. Ranftl, K. Lasinger, D. Hafner, K. Schindler, and V. Koltun, "Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer," 2020.

[21] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, "Depth anything: Unleashing the power of large-scale unlabeled data," 2024.

[22] S. F. Bhat, R. Birkl, D. Wofk, P. Wonka, and M. Müller, "Zoedepth: Zero-shot transfer by combining relative and metric depth," 2023.

APPENDIX

As an additional experiment, ZoeDepth pretrained model ZoeD-X-N [22] was utilized to estimate the metric depth of the detected objects from YOLO. This model has been fine-tuned on NYUv2 dataset for indoor settings to predict metrically accurate depth values. We observe that the model gives differing results based on the device used. As such, the model uses device-specific intrinsic parameters for the depth estimation. The models' performance could be enhanced with intrinsic parameter consideration per device [10]. However, in order to generalize the pipeline and avoid intrinsic parameter considerations, further fine-tuning and additional images are required to train the model accordingly. On the basis of preliminary tests done, we deduce that with initial metric calibration of a few images taken with one camera, the model gives decent results. However, to achieve generalization, we plan to deploy ZoeDepth for depth mapping without leveraging the parameters. Upon generalizing and algorithmic changes based on the object segmentation, this model can very well substitute the depth map formed based on the iPhone's LiDAR sensor which has an operability range limit.

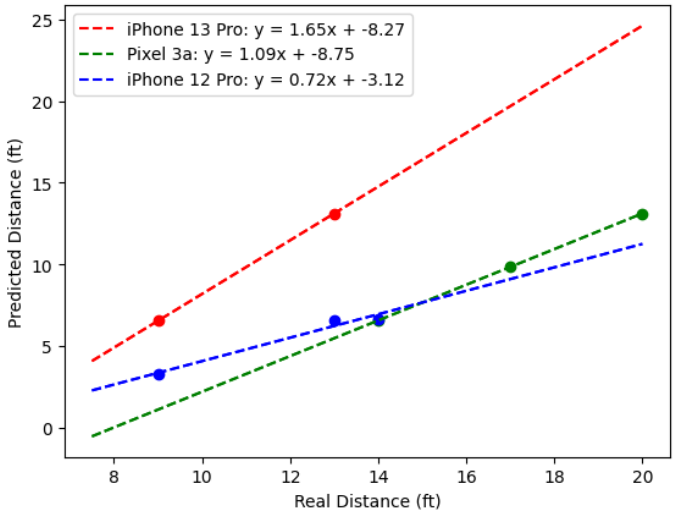

Fig. 10. ZoeD-X-N pretrained model's sensitivity to camera.

With minor changes, we see that ZoeDepth can be utiized to create a 3D mesh of the object(s) with the generated depth map. Using the generated 3D mesh, the BEV of the scene may be constructed without requiring angle estimation. Visually, the generated 3D mesh seems to be relatively decent as shown in figures 11 and 12 of side-view and top view. However, more experimentation is required to confirm the metric and angular accuracy of this map. Upon confirmation of this, the model can be developed further to not leverage device-specific intrinsic parameters, in order to realize real-time FPV to TD mapping for videos.

Fig. 11. Side-view of the generated 3D mesh from 8. Similar color masks are used for corresponding people.

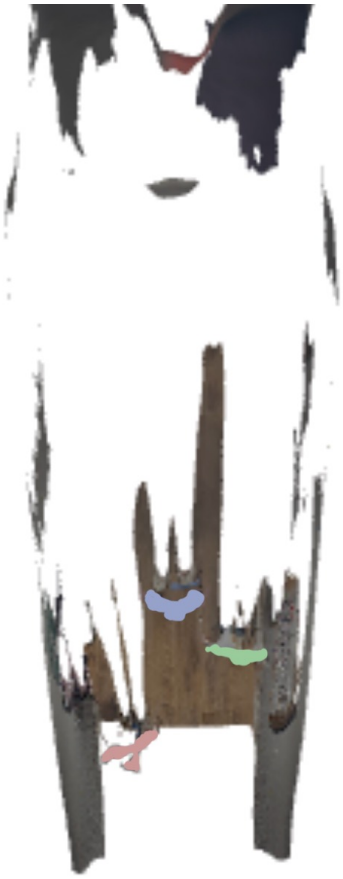

Fig. 12. Top-view of the generated 3D mesh from figure 8. Similar color masks are used for corresponding people.