

LCPP: Low Computational Processing Pipeline for Delivery Robots

Soofiyan Atar¹^a, Simranjeet Singh¹^b, Srijan Agrawal⁴^c, Ravikumar Chaurasia³^d,
Shreyas Sule⁵, Sravya Gadamssetty², Aditya Panwar¹, Amit Kumar¹ and Kavi Arya¹^e

¹Indian Institute of Technology Bombay, Mumbai, India

²Indian Institute of Technology Bhubaneswar, Bhubaneswar, India

³University of Mumbai, Mumbai, India

⁴Shri G. S. Institute of Technology and Science, Indore, India

⁵D Y Patil's Ramrao Adik Institute of Technology, Mumbai, India


Keywords: Delivery Robot, Semantic Segmentation, Object Detection.


Abstract: Perception techniques in novel times have enormously improved in autonomously and accurately predicting the ultimate states of the delivery robots. The precision and accuracy in recent research lead to high computation costs for autonomous locomotion and expensive sensors and server dependency. Low computational algorithms for delivery robots are more viable as compared to pipelines used in autonomous vehicles or prevailing delivery robots. A blend of different autonomy approaches, including semantic segmentation, obstacle detection, obstacle tracking, and high fidelity maps, is presented in our work. Moreover, LCPP comprises low computational algorithms feasible on embedded devices with algorithms running more efficiently and accurately. Research also analyzes state-of-the-art algorithms via practical applications. Low computational algorithms have a downside of accuracy, which is not as proportional as computation. Finally, the study proposes that this algorithm will be more realizable as compared to Level 5 autonomy for delivery robots.


1 INTRODUCTION


The Autonomous Delivery Robot (ADR) is evolving rapidly in terms of scalability and efficiency. ADR shows compelling advantages over a standard delivery system, such as low delivery cost, faster service and higher accuracy than a human. Though ADRs do not have high adoption, they operate well, considering their advantages over other methods. The ADR business is booming in this coronavirus outbreak, which promises contact-less delivery, a mandate in this pandemic. Before the pandemic, ADRs were underused in airports, university campuses, hotels, hospitals, and large corporate campuses. After the pandemic, demand for ADR has increased exponentially as these do not mimic human behaviour, which prevents the spreading of novel coronavirus. ADR market in the USA is valued at 0.35 million USD in 2020, and after half a decade, it is expected to reach 3.82 million USD (Intelligence, 2020).


ADR uses different perception pipelines as compared to outdoor vehicles, including truck delivery robots (Marshall, 2017). The complexity increases in terms of dynamic planning, but on the other side, low driving speed decreases the requirement of computational power (Lamon et al., 2006). The pipeline used in autonomous vehicles cannot be used in ADR because the environment, momentum and path taken are different. ADR requires precise tuning to navigate itself in the crowded street. ADRs are mostly knee-height robots, and they usually navigate in sophisticated areas giving priority to safety (Kümmerle et al., 2013). Nowadays, ADRs requires very complex sensory equipment, which increases the overall complexity and requirement of high computation power. Most of the current autonomous robots consist of sensors covering the 2D plane of the environment for navigation, reducing complexity (Ren et al., 2019). Still, only 2D information of the domain is not enough for navigating in cluttered areas. ADRs need an exact perception stack for navigating in cluttered conditions. Complex perception involves different approaches using depth cameras, stereo cameras, 360-degree cameras, 3D lidar and a combination of many cameras. Perception pipeline in current state-of-the-art robots use 3D point cloud data

^a  <https://orcid.org/0000-0002-0878-9347>

^b  <https://orcid.org/0000-0002-8297-1470>

^c  <https://orcid.org/0000-0002-5080-4184>

^d  <https://orcid.org/0000-0002-1917-8146>

^e  <https://orcid.org/0000-0002-7601-317X>

or server-based High Fidelity Maps (HFM) (Browning et al., 2012; Chipka and Campbell, 2018) approach with complicated architecture for navigation pipeline. The perception pipeline of ADR consists of road/sidewalk segmentation, obstacle/pedestrian avoidance and tracking, and HFM for localization.

The problem with these methods is that they are highly computationally expensive concerning the embedded device used in ADR. Modern techniques use the fusion of the Global Positioning System (GPS) and HFM. The issue with GPS is that signal reception gets weakened near buildings (Kos et al., 2010), which causes problems in navigation. HFM being more accurate and precise gives better localization results, but the current implementation requires server-based computation (Vitali et al., 2019), as HFM cannot be used for onboard processing. This research provides an asynchronous HFM technique for localization with low computation, ignoring less essential features, and prioritizing powerful spatial global features, thus reducing the algorithm's complexity.

Semantic segmentation is a vital segment for local planning of the ADR (Qiu et al., 2019; Lin et al., 2017; Hong et al., 2018; Cheng et al., 2019). Semantic segmentation gives in-depth information about the environment and provides accurate localization of objects in the sensor frame. However, state-of-the-art semantic segmentation algorithms for autonomous vehicles are computationally heavy. Current systems heavily depend on point cloud data for segmentation, which suffers from near-field blind spots (Ghafarzadeh, 2019), resulting in low precision. Using point cloud data also increases computational processing, or server dependency (Wang et al., 2021). Semantic segmentation datasets such as Cityscape (Cordts et al., 2015), or Camvid (Brostow et al., 2009) do not provide accurate data for delivery robots; they do not include rough patches of the road. To solve this issue, we created a custom dataset for pedestrian level height camera feed. LCPP uses semantic segmentation only for traversing. Combining object detection for obstacles along with semantic segmentation reduces computation cost. Object tracking is necessary for tracking and anticipating the movement of various moving objects in the view. Tracking obstacles for future path prediction is also a processing-intensive task that enables more safety in robot's path planning. Every algorithm focuses on low computational processing in this research, which reduces server dependency and high processing power requirements.

LCPP focuses on a low computational perception pipeline that enables onboard computing, viable in communication prone areas and GPS denied areas. The major contribution of the paper is:

- Implementation of less computation expensive road segmentation techniques for efficient sensing
- Implementation of object detection and tracking algorithm for ADR.
- Efficient implementation of HFM for amendments of robot localization.

This research modifies perception stacks used in state-of-the-art delivery robots with increased efficiency and safety. The decrease in computational processing power also minimizes the cost of the sensors stack.

The rest of the paper is structured as follows. Section 2 reviews the existing and related work on perception techniques in ADR. Section 3 focuses on the implementation of the proposed perception pipeline for ADR. Finally, Section 4 provides the conclusion and outlook of this work in brief.

2 RELATED WORK

LCPP is related to an autonomous mobile robot running in outdoor environments. In outdoor environments, delivery robots need to run on sidewalks because of their size and payload capacity. Delivery trucks (Lee et al., 2013) is used for a higher payload capacity, and their challenges, complexity, safety requirements and processing power are the same as autonomous vehicles. Unlike indoor autonomous delivery robots, our implementation does not require mapping (Pereira et al., 2006) the environment, as we use GPS mission planning using Google maps to find global paths. Previous research (Yahja et al., 2000) plans the trajectory in outdoor terrain without mapping the environment. Our approach focuses on low computational processing (Miller and Gat, 1991) for outdoor navigation in GPS denied areas without server communication.

Semantic segmentation takes up a lot of computational processing as it assigns labels pixel-wise. In the past, many researchers have implemented semantic segmentation for outdoor navigation (Hong et al., 2018; Cheng et al., 2019), that are not suitable for embedded devices. There are other implementations based on 3D point cloud (Qi et al., 2017; Balado Frias et al., 2019), which are even more computationally expensive compared to 2D image segmentation. Considering standard deep neural networks for semantic segmentation AlexNet (Krizhevsky et al., 2017), VGG-16 (Simonyan and Zisserman, 2015), GoogleNet (Szegedy et al., 2014), ResNet (He et al., 2015) are mostly used for road segmentation in outdoor navigation. These standard networks consist of

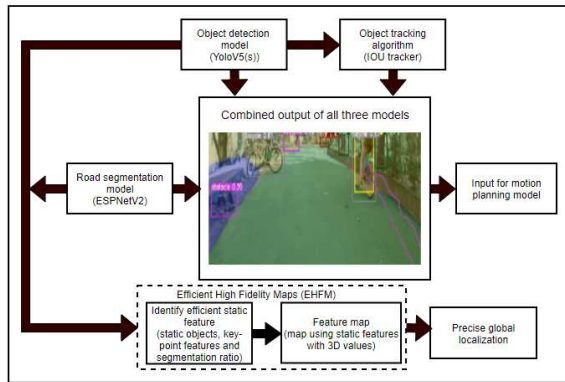


Figure 1: Pipeline for low computational perception. Trained road segmentation and object detection models are used to process the input. The EHFM pipeline would then utilize the combined data to perform global robot precision. Object detection data is used to track objects in the view, which is used for motion planning.

fewer layers for low computation, but the accuracy decreases considerably.

Amongst all these algorithms, ResNet gives the most accurate results, but it contains 156 layers of the deep neural network, which results in high computation. Combining semantic segmentation and object detection (known as multi-task learning (Crawshaw, 2020)) decreases computational requirements (Wang et al., 2007). Object detection is used to detect obstacles such as pedestrians, potholes, trees, and many more (Uçar et al., 2017a; Uçar et al., 2017b). Object detection algorithms are lighter than semantic segmentation; thus, they reduce the computational cost by reducing classes used in semantic segmentation. SimpleNet (Lewis, 2016) has an acceptable accuracy with real-time detection, Fast RCNN (Ren et al., 2016) does not provide real-time performance but gives high accuracy, YOLO (Redmon et al., 2016) is a balanced performer in terms of accuracy and computation. Obstacles after detection are used for tracking to predict future paths for traversing. Multi-object tracking (Luo et al., 2017) for mobile obstacles is used during path planning.

Our approach is similar to the tracking-by-detection approach. The majority of the batch methods (Berclaz et al., 2011; Pirsiavash et al., 2011) use graph-based representation for a global optimizer. Various trackers such as centroid tracker(Nascimento et al., 1999), Intersection Over Union (IOU) tracker(Bochinski et al., 2017), and Kalman tracker(Gunjal et al., 2018) are computationally inexpensive for mobile objects used for motion planning.

HFM (Browning et al., 2012; Chipka and Campbell, 2018) is useful in GPS denied areas such as near

apartments or in groves of trees where the GPS gives inaccurate readings or no readings. HFM helps correct the world coordinate of the ADR with higher accuracy than GPS. HFM is nowadays used with GPS using fusion algorithms for more accurate and precise localization. Implementation of HFM using low computational processing is still under research. LCPP uses Efficient High Fidelity Maps (EHFM) for localizing and is also used in robot kidnap problems (Yi and Choi, 2011), where the robot spawns randomly.

3 IMPLEMENTATION

Road segmentation, object detection, object tracking, and EHFM are all part of LCPP. Figure 1 describes the efficient synchronous pipeline, where the annotated data for road segmentation and object detection is used to train the semantic segmentation model (ESNetV2) and object detection model (YoloV5(s)), respectively. The trained semantic segmentation and object detection models are combined and used for the motion planning module. The integrated model is used in conjunction with object tracking, which is used to track mobile objects such as cars, bicycles, and other vehicles. The segmentation and object detection results are further used to perform feature extraction. The extracted features are taken as an input for the EHFM algorithm to get the global position of the ADR. Oriented FAST and Rotated BRIEF (ORB) (Rubblee et al., 2011) descriptors which use Accelerated Segment Test (FAST) keypoints and Binary Robust Independent Elementary Features (BRIEF). ORB descriptors are used to extract distinct 3D point features, and we built a low-resolution map combining all these features.

3.1 Custom Dataset

We have created a custom dataset to train and test semantic segmentation algorithms in the designed pipeline. This dataset is also available on the Kaggle ([Dataset link](#)). The dataset comprises various scenes from the campus of IIT Bombay with knee heightened scenes. It consists of 4000+ annotated images created by us for road segmentation and 12600+ annotated images for object detection, a mixture of collected datasets from Kaggle (Patel, 2019), and COCO 2017 train dataset (Veit et al., 2016). The remaining images are added from our custom dataset.

The dataset made for road segmentation consists of five classes: road, shallow, footpath, pothole and background. The image portion containing regions that can be traversed at average speed by the ADR are

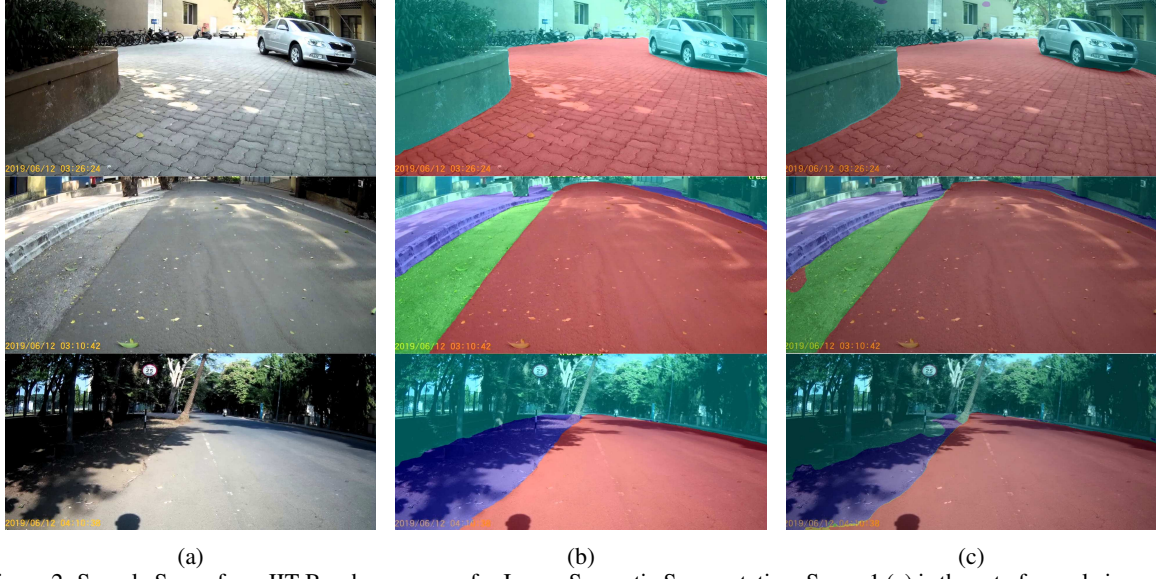


Figure 2: Sample Scene from IIT-Bombay campus for Image Semantic Segmentation, Scene 1 (a) is the set of sample images, Scene 2 (b) is output using MobileNetV3 on sample images, and Scene3 (c) is the output of ESPNetv2 on sample images. In segmentation predicted mask, red color represents road; green color represents shallow paths; violet color represents footpath and cyan color represents the background respectively.

marked as road. The image portions where the ADR has to travel at a slower pace are marked as shallow. The regions marked as footpaths are the sides of the road that cannot be traversed (in the Indian environment, the ramps for the footpath are not adequately set up, which might lead to an abrupt end during the ADR's movement). Regions on the road that are not traversable are categorised as potholes. Similarly, we have limited categories for object detection as well.

Table 1: Road segmentation models comparison.

| Model Name | MobileNetV3_small | ESPNetV2 |
|---------------------------|-------------------|--------------|
| Parameter | 2.5M | 0.79M |
| FPS* | 21 | 31 |
| mIOU on Cityscape dataset | 68 | 66.7 |
| mIOU on IITB dataset | 62 | 68.2 |

*on GTX 1050 (Mobile) (GPU Geekbench link).

3.2 Road Segmentation

Road segmentation means to segment visual input from the camera into different categories like road, footpath, pothole, and background in real-time. Cars, humans, trees and such objects were

Table 2: Object detection models comparison.

| Model Name | mAP | FPS |
|------------|--------------|-----------|
| RetinaNet | 33.5 | 10 |
| ENet | 48.3 | 16 |
| YoloV5(s) | 55.60 | 33 |



Figure 3: Object detection on custom dataset. The colored bounding boxes represent different classes of objects detected by the model in the image.

not included in segmentation because the object detection model handled them. The navigation system uses these insights to traverse on-road and avoid non-traversable paths (potholes, footpaths). Some State-of-the-art segmentation models that were considered are DeeplabV3(Chen et al., 2018), MobileNetV3(Howard et al., 2019), ENet(Paszke et al., 2016) and ESPNetV2(Mehta et al., 2019). The Deeplab model was not real-time (low FPS), hence rejected even though it had the highest accuracy. The ENet model did not perform well on the Cityscape dataset and had mean Intersection Of Union (mIOU) of only 0.18. The remaining two models were trained and compared as shown in Table 1. The results of MobileNetV3 and ESPNetV2 are shown in Figure 2b and Figure 2c respectively. From Table 1 it can be

concluded that the ESPNetV2 model performs best on the given dataset in terms of both high accuracy and low computation as compared to MobileNetV3_small.

3.3 Object Detection and Tracking

Object detection model is essential for avoiding obstacles. RetinaNet(Lin et al., 2018), EfficientNet (ENet)(Tan and Le, 2020), and YoloV5(s)(Jocher et al., 2021) were the models tested. The accuracy and performance of YoloV5(s) was satisfactory as compared to other models on custom dataset as shown in Table 2.

YoloV5(s) was trained on a custom dataset for 50 epochs. Table 2 shows that YoloV5(s) outperformed all other architectures by a wide margin. As the table shows, mean Average Precision (mAP) of YoloV5(s) is 14% higher than that of ENet, and FPS is 100% higher than ENet. The output of YoloV5(s) is shown in Figure 3 which detects obstacles, vehicles, potholes and many more objects with 55.6 mAP. For object tracking, centroid tracker, IOU tracker and Kalman tracker were considered. The centroid tracker and Kalman tracker had a lot of false detections. Centroid and Kalman filter tracking gave FPS of 44, and IOU tracking provided FPS of 46. The IOU tracker outperformed the Kalman and centroid trackers in terms of efficiency, thus tracking objects such as bicycles, humans, cars, and other dynamic objects.

By avoiding the tracking of static objects such as trees and potholes, the computation is further reduced. The outcome of object tracking is shown in Figure 4 where the tracking is obtained with the relative motion of the objects. For example, cars, persons and bicycles are tracked, and their trailing path is displayed in Figure 4. YoloV5(s) and IOU tracker are the best for the delivery application as they perfectly balance precision and efficiency.



Figure 4: Mobile objects are tracked in subsequent images. The colored lines represent various positions of the object being tracked.

3.4 EHFM

Regular HFM is computationally expensive, so it has to run on high-end servers with constant connectivity (Chipka and Campbell, 2018)(Vitali et al., 2019). LCPP uses different techniques to reduce the complexity and to increase the efficiency of HFM. Algorithm 1 explains the method applied to obtain global coordinate used in LCPP.

Algorithm 1: EHFM algorithm for efficient global localization.

Input: RGB-D Image

Output: Global coordinate of ADR

Data: Low-resolution map

- 1 Apply semantic segmentation model
- 2 Extract ratios of the road to background and footpath to background
- 3 Match the segmentation ratio of the input image with the low-resolution map
- 4 Detect static objects from the image using object detection model
- 5 **for every static object in the frame do**
- 6 **if object is in list of stored static objects then**
- 7 calculate distance from static object using Equation (1) and Equation (2)
- 8 Take average of all global coordinates obtain from static objects to calculate g_1
- 9 Extract set of key-point features from input image using ORB
- 10 Match every feature from the set of low-resolution feature maps
- 11 Select strong correspondences from these matches
- 12 **for all strong correspondences matched features do**
- 13 Calculate point depth and apply Equation (1) and Equation (2) to get the exact location of the ADR
- 14 Take average of all the matched points and calculate g_2
- 15 Apply weighted average on g_1 and g_2 to obtain the global coordinate of the ADR

3.4.1 Low Resolution Maps

For building low-resolution maps, LCPP uses four depth cameras, each facing in different principal directions. To reduce computation, while testing, a single depth camera is used and then features obtained from it are used to match features from the pre-built maps. These maps consist of all features (segmentation ratios, static objects and key points) from all four

Table 3: Comparing segmentation pixel ratio of 2 scenes for uniqueness, up-to five decimal. Refer Figure 5

| Segmentation Pixel Count Ratios | Scene 1 | Scene 2 |
|---------------------------------|---------|---------|
| Road to Background | 1.73711 | 5.29886 |
| Footpath to Background | 0.15859 | 0.68700 |
| Shallow to Background | 0.27880 | 0.98892 |

images. These features were stored with a 1 Hz frequency. Also, the world coordinate of the camera's centre is saved along with the 3D information of all the features.

3.4.2 Filtering and Slicing Relevant Features

Orientation of the camera is used to get the relevant image by slicing the four principal direction images extracted from the map. From Figure 7, the image is formed in a 50:50 ratio by slicing half from the first image (0°) and a half from the second image (-90°) marked by black boundary. FOV of camera was set at 90° . As shown in Figure 7, the sliced image is confined to match the features from a section of the pre-built map. Similarly, a maximum of two images from the four images are processed in the pipeline for all the orientations, making the process data-efficient and reducing processing per cycle.

3.4.3 Segmentation Pixel Ratios

Every image/scene has a unique segmentation ratio of road to background and footpath to the background, which is used as a global feature, as shown in Table 3.

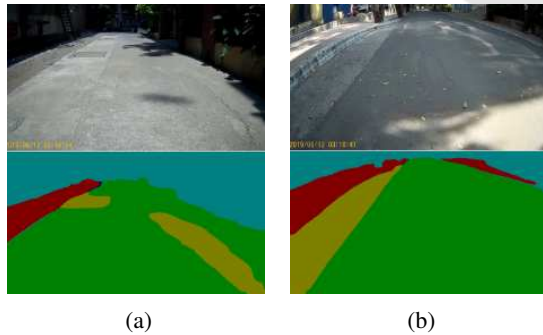


Figure 5: Sample Scene from IIT-Bombay campus, Scene 1 (a) and Scene 2 (b). In segmentation mask (lower half) green, yellow, red and cyan are road, shallow, footpath and background respectively.

The segmented mask of all four principal direction images is stored as features in the pre-built map. The segmentation pixel ratio for any orientation of all the stored masks is obtained using the sliced image. Alternatively, it can also store the segmentation ratio of

every 10° for four principal direction views, lowering the searching computation.

Table 4: Comparison of ORB, SURF, SIFT feature extractor algorithms on sample images.

| Algorithms | Time (sec) | Match (%) | Descriptor size (Bytes) |
|------------|-------------|-----------|-------------------------|
| ORB | 0.03 | 64 | 32 |
| SURF | 0.63 | 69 | 64 |
| SIFT | 0.51 | 72 | 128 |

3.4.4 Static Objects Feature

Static objects such as signboards, fire-hydrants, road-side milestones, and many (Figure 6) are some of the helpful roadside features and significant markers. With the list of static objects from object detection,



Figure 6: Some of the common static object by the side of IIT-Bombay road.

as shown in Algorithm 1, it first identifies the object from the stored static object feature using template matching (opencv dev team, 2021) by ORB descriptor (Rublee et al., 2011).

3.4.5 Efficient Key-point Features

ORB descriptors (Rublee et al., 2011; Tareen and Saleem, 2018) performs best for outdoor scenarios, while SIFT(Lindeberg, 2012) works well for indoor scenarios because of the graffiti-type images. Besides improving the detection speed, it can also detect moving targets accurately in real-time. The sample dataset was tested with all of the feature extractors, and ORB was the fastest and yielded a fairly good match (Ref. Table 4).

As shown in Figure 7, feature matching of a scene (train image from upper right section) with another

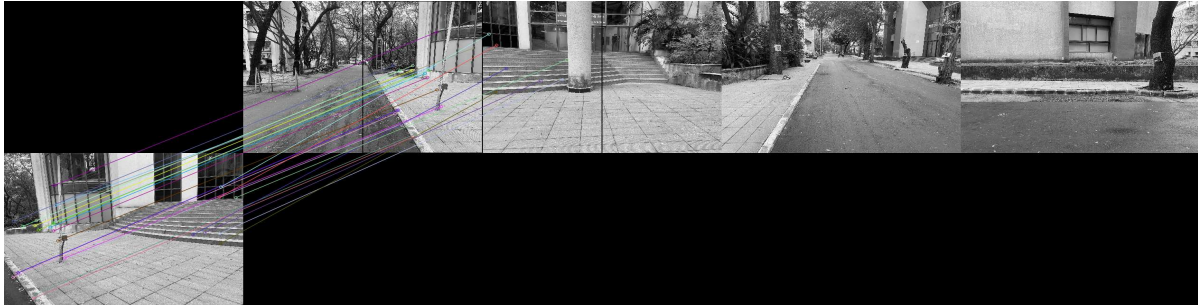


Figure 7: Sample feature matching with four principal direction view.

scene from live data (test image from lower left section), confined to the sliced image from the four principal direction view. Instead of matching the live data with individual stored images, LCPP matches it with the stored key points. The feature key points are stored along with its descriptors and space coordinate using ORB descriptor; this decreases the data to store and the onboard process. To reduce the computational load further, as shown in Algorithm 1, LCPP uses only those descriptors with solid correspondences between the feature key points from the map and the sliced image. Running it at 0.5 Hz yields a fair, accurate prediction of ground truth. Intrinsic data of the 3D camera (cx, cy, fx, fy) is obtained by calibrating the camera. The 3D coordinate of a given pixel is calculated using the Equation 1 and 2 (opencv dev team, 2014):

$$x = \frac{(u - cx)}{fx}, y = \frac{(v - cy)}{fy}, norm = \sqrt{x^2 + y^2 + 1} \quad (1)$$

$$(x, y, z) = \left(\frac{x}{norm}, \frac{y}{norm}, \frac{1.0}{norm} \right) \quad (2)$$

where, (cx, cy) is the centre of curvature; (fx, fy) are the focal lengths; (u, v) is the pixel coordinate of 2D image. The rotation matrix of the camera's origin transforms the calculated feature coordinates to the world frame. This method reduces the error caused by the vibration or noise established from the depth camera. The predicted location was produced using the vector subtraction method by subtracting the world coordinates of the recorded features at the origin and the world coordinates of the same features with respect to the camera frame. Further, efficient key point matching eliminates all the nonessential loads, making EHFM run more effectively and efficiently than standard HFM. This method was tested on IIT-Bombay's road using a ZED2 camera with road segmentation and object detection running for approximately 120 meters of distance.

The execution cycle was set at 0.25 Hz, and the average controlled speed was approx. 2 m/s, i.e., after every 8 meters. In Figure 8, ground truth is marked in

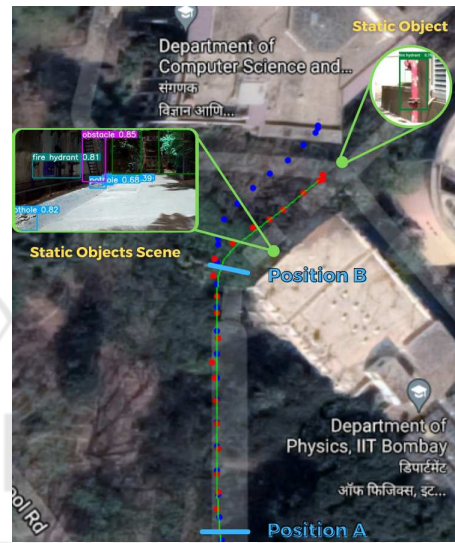


Figure 8: EHFM result on IIT-Bombay campus.

green, prediction is marked in red, and Odom's reading is marked in blue. Position A in Figure 8 represents the straight path; here, EHFM prediction is close to the ground truth. Position B in Figure 8 represents the curve path, here the challenge was the rapid moving of frames and less number of static objects, which caused a slightly higher error in the EHFM prediction, but with the following prediction, it gets much accurate as shown in Figure 9. After 120 meters, the Odom was deflected by 8 meters from the ground truth. However, the EHFM errors were consistently below 2.4 meters.

4 CONCLUSION

LCPP is a highly optimized perception pipeline for outdoor navigation developed for delivery robots. All perception techniques, which comprises most of the computation for delivery robots, including road segmentation, EHFM and object tracking, are realized for onboard computation, which is a requirement

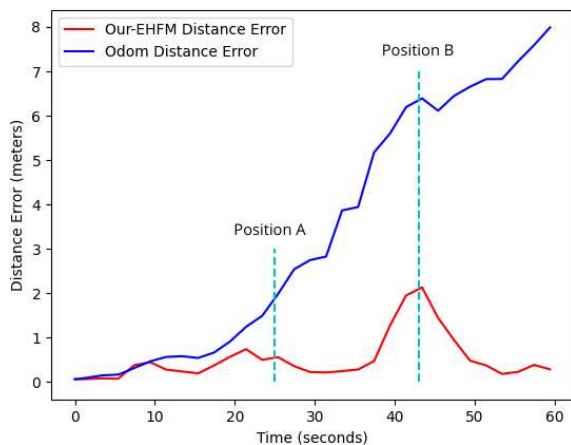


Figure 9: EHFM error vs Odom error.

for autonomy and eliminating server dependencies. LCPP consist of a balance of efficient and precise road segmentation algorithms. ESPNetV2 provided mIOU of 68.2, and YoloV5(s) provided mAP of 55.6 for the custom dataset, indicating that precision was high and can be increased using a more computational heavy backbone for these algorithms. The outcome of EHFM is demonstrated, which runs efficiently and performs better than odometry sensor values using low-resolution pre-built maps. The error of actual location and the coordinates provided through EHFM gave a maximum of 2% error for 120m trajectories. LCPP has been implemented on Jetson Xavier AGX, with all algorithms running parallel.

In future work, LCPP will consist of more efficient dynamic algorithms and implement them on a Graphics Processing Unit (GPU) or Field Programmable Gate Array (FPGA). In addition, LCPP will also be implemented in a more active and cluttered environment to demonstrate highly efficient and effective behaviour.

ACKNOWLEDGEMENTS

This work was supported in part by Ministry of Education (MoE), Govt. of India in the project e-Yantra (RD/0113-MHRD001-021). We acknowledge the support of e-Yantra staff especially Rathin Biswas for reviewing the paper.

REFERENCES

Balado Frias, J., Martínez-Sánchez, J., Arias, P., and Novo, A. (2019). Road environment semantic segmentation with deep learning from mls point cloud data. *Sensors*, 19:3466.

Berclaz, J., Fleuret, F., Turetken, E., and Fua, P. (2011). Multiple object tracking using k-shortest paths optimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1806–1819.

Bochinski, E., Eiselein, V., and Sikora, T. (2017). High-speed tracking-by-detection without using image information. In *2017 14th IEEE (AVSS)*, pages 1–6.

Brostow, G. J., Fauqueur, J., and et al. (2009). Semantic object classes in video: A high-definition ground truth database. *Pattern Recognition Letters*, 30(2):88–97.

Browning, B., Deschaud, J.-E., Prasser, D., and Rander, P. (2012). 3d mapping for high-fidelity unmanned ground vehicle lidar simulation. *The International Journal of Robotics Research*, 31:1349–1376.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*.

Cheng, G., Zheng, J. Y., and Kilicarslan, M. (2019). Semantic segmentation of road profiles for efficient sensing in autonomous driving. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 564–569.

Chipka, J. and Campbell, M. (2018). Autonomous urban localization and navigation with limited information.

Cordts, M., Omran, M., and et al. (2015). The cityscapes dataset. In *CVPR Workshop on the Future of Datasets in Vision*, volume 2.

Crawshaw, M. (2020). Multi-task learning with deep neural networks: A survey.

Ghaffarzadeh, D. K. (2019). Dtechex research: Sidewalk last mile delivery robots: A billion-dollar-market by 2030. .

Gunjal, P. R., Gunjal, B. R., Shinde, H. A., Vanam, S. M., and Aher, S. S. (2018). Moving object tracking using kalman filter. In *2018 ICACCT*, pages 544–547.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

Hong, Z.-W., Yu-Ming, C., and et al. (2018). Virtual-to-real: Learning to control in visual semantic segmentation.

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al. (2019). Searching for mobilenetv3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1314–1324.

Intelligence, M. (2020). Autonomous delivery robots market - growth, trends, covid-19 impact, and forecasts (2021 - 2026). <https://www.mordorintelligence.com/industry-reports/autonomous-delivery-robots-market/>.

Jocher, G., Stoken, A., and et al. (2021). ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations.

Kos, T., Markezic, I., and Pokrajcic, J. (2010). Effects of multipath reception on gps positioning performance. pages 399 – 402.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90.

Kümmerle, R., Ruhnke, M., Steder, B., Stachniss, C., and Burgard, W. (2013). A navigation system for robots

- operating in crowded urban environments. In *2013 IEEE ICRA*, pages 3225–3232. IEEE.
- Lamon, P., Kolski, S., and Siegart, R. (2006). The smartter-a vehicle for fully autonomous navigation and mapping in outdoor environments. In *Proceedings of CLAWAR*.
- Lee, D.-Y. D.-Y., Thomas, V., and Brown, M. (2013). Electric urban delivery trucks: Energy use, greenhouse gas emissions, and cost-effectiveness. *Environmental science & technology*, 47.
- Lewis (2016). Object detection for autonomous vehicles gene.
- Lin, J., Wang, W.-J., Huang, S.-K., and Chen, H.-C. (2017). Learning based semantic segmentation for robot navigation in outdoor environment. In *2017 Joint 17th World Congress of IFSA and 9th SCIS (IFSA-SCIS)*, pages 1–5.
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., and Dollár, P. (2018). Focal loss for dense object detection.
- Lindeberg, T. (2012). Scale Invariant Feature Transform. *Scholarpedia*, 7(5):10491. revision #153939.
- Luo, W., Xing, J., Milan, A., Zhang, X., Liu, W., Zhao, X., and Kim, T.-K. (2017). Multiple object tracking: A literature review.
- Marshall, A. (2017). The us postal service is building a self-driving mail truck. <https://www.wired.com/story/postal-service-office-self-driving-mail-trucks/>.
- Mehta, S., Rastegari, M., Shapiro, L., and Hajishirzi, H. (2019). Espnetv2: A light-weight, power efficient, and general purpose convolutional neural network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9190–9200.
- Miller, D. P. and Gat, E. (1991). Exploiting known topologies to navigate with low-computation sensing. In *Sensor Fusion III: 3D Perception and Recognition*, volume 1383, pages 425–435. International Society for Optics and Photonics.
- Nascimento, J., Abrantes, A., and Marques, J. (1999). An algorithm for centroid-based tracking of moving objects. In *1999 IEEE ICASSP99 (Cat. No.99CH36258)*, volume 6, pages 3305–3308 vol.6.
- opencv dev team (2014). *Camera Calibration and 3D Reconstruction*. OpenCV, 2.4.13.7 edition.
- opencv dev team (2021). *Template Matching*. OpenCV, 4.5.2 edition.
- Paszke, A., Chaurasia, A., Kim, S., and Culurciello, E. (2016). Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*.
- Patel, S. (2019). Pothole image data-set. <https://www.kaggle.com/sachinpatel21/pothole-image-dataset>.
- Pereira, G., Pimenta, L., Chaimowicz, L., Fonseca, A., Almeida, D., Corrêa, L., Mesquita, R., and Campos, M. (2006). Robot navigation in multi-terrain outdoor environments. volume 28, pages 331–342.
- Pirsiavash, H., Ramanan, D., and Fowlkes, C. C. (2011). Globally-optimal greedy algorithms for tracking a variable number of objects. In *CVPR 2011*, pages 1201–1208.
- Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation.
- Qiu, Z., Yan, F., Zhuang, Y., and Leung, H. (2019). Outdoor semantic segmentation for uavs based on cnn and fully connected crfs. *IEEE Sensors Journal*, 19(11):4290–4298.
- Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection.
- Ren, R., Fu, H., and Wu, M. (2019). Large-scale outdoor slam based on 2d lidar. *Electronics*, 8(6):613.
- Ren, S., He, K., Girshick, R., and Sun, J. (2016). Faster r-cnn: Towards real-time object detection with region proposal networks.
- Rublee, E., Rabaud, V., Konolige, K., and Bradski, G. (2011). Orb: an efficient alternative to sift or surf. pages 2564–2571.
- Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.
- Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2014). Going deeper with convolutions.
- Tan, M. and Le, Q. V. (2020). Efficientnet: Rethinking model scaling for convolutional neural networks.
- Tareen, S. A. K. and Saleem, Z. (2018). A comparative analysis of sift, surf, kaze, akaze, orb, and brisk. In *2018 International Conference on Computing, Mathematics and Engineering Technologies (iCoMET)*, pages 1–10.
- Uçar, A., demir, Y., and Güzeliş, C. (2017a). Object recognition and detection with deep learning for autonomous driving applications. *SIMULATION*, 93:003754971770993.
- Uçar, A., demir, Y., and Güzeliş, C. (2017b). Object recognition and detection with deep learning for autonomous driving applications. *SIMULATION*, 93:003754971770993.
- Veit, A., Matera, T., Neumann, L., Matas, J., and Belongie, S. (2016). Coco-text: Dataset and benchmark for text detection and recognition in natural images.
- Vitali, E., Gadioli, D., Palermo, G., Golasowski, M., Bispo, J., Pinto, P., Martinovic, J., Slaninova, K., Cardoso, J., and Silvano, C. (2019). An efficient monte carlo-based probabilistic time-dependent routing calculation targeting a server-side car navigation system. *IEEE transactions on emerging topics in computing*.
- Wang, L., Shi, J., Song, G., and Shen, I.-f. (2007). Object detection combining recognition and segmentation. pages 189–199.
- Wang, X., Mizukami, Y., Tada, M., and Matsuno, F. (2021). Navigation of a mobile robot in a dynamic environment using a point cloud map. *Artificial Life and Robotics*, 26(1):10–20.
- Yahja, A., Singh, S., and Stentz, A. (2000). Efficient on-line path planner for outdoor mobile robots. *Robotics and Autonomous Systems*, 32.
- Yi, C. and Choi, B.-U. (2011). Detection and recovery for kidnapped-robot problem using measurement entropy. volume 261, pages 293–299.