Depth-Lift: Edge-Deployable Pedestrian Trajectory Prediction via Lightweight Monocular Depth Estimation

Li Zhang[1], Yi Zhang[1*], Xinyu Wang[1], Zhihao Lin[3], Xiang Luo[1], Fengmin Cheng[2]

***Abstract*— Most pedestrian trajectory prediction methods rely on LiDAR or stereo cameras for 3D localization—hardware that is impractical on resource-constrained edge devices. We propose Depth-Lift, a monocular-only pipeline that replaces dedicated depth sensors with a lightweight depth estimation network. A MobileNetV2-based encoder–decoder estimates dense depth from a single RGB frame, and detected bounding boxes are back-projected into pseudo-3D coordinates to feed a downstream trajectory predictor. On KITTI, the pipeline achieves 38 ms end-to-end latency on NVIDIA Jetson Nano (FP16) and 105 ms on Raspberry Pi 5 (INT8), while pedestrian trajectory prediction accuracy degrades by only 8–12% compared with ground-truth depth input, enabling practical behaviour prediction on resource-constrained platforms.**

I. Introduction

Pedestrian trajectory prediction is a fundamental capability for autonomous systems operating in shared environments. Accurate 3D pedestrian localization is a prerequisite for reasoning about future motion [1], yet LiDAR and stereo systems are impractical on edge platforms such as delivery robots and low-cost ADAS units where only a monocular camera is available, leaving pedestrian behaviour prediction severely under-served in these deployment scenarios.

Although lightweight monocular depth estimation has advanced rapidly [2], its use as a perception module for downstream trajectory prediction under edge constraints has not been systematically studied. Two questions remain open: (i) does depth noise from a lightweight estimator significantly degrade trajectory prediction accuracy, and (ii) can the full perception–prediction pipeline run in real time on representative edge hardware?

We address both questions with **Depth-Lift** (Fig. 1), a two-module pipeline connecting a lightweight MobileNetV2-based [3] depth estimator with a standard trajectory predictor. A *depth-lifting* layer back-projects detected bounding boxes into pseudo-3D coordinates, and the entire pipeline is optimized via INT8/FP16 quantization for deployment on NVIDIA Jetson Nano and Raspberry Pi 5.

II. Method

A. Module 1: Lightweight Depth Estimation

Architecture. We adopt an encoder–decoder structure with MobileNetV2 [3] as the encoder backbone, following the architecture of FastDepth [2]. To improve boundary sharpness with minimal overhead, we insert a lightweight channel attention block between encoder and decoder. The decoder uses depthwise separable transposed convolutions for progressive upsampling; total parameter count is kept under 4 M.

Training & Distillation. The network is supervised on the KITTI Eigen split [4] with the loss:

$$\mathcal{L}_{\text{depth}} = \alpha \, \|\hat{d} - d^*\|_1 \; + \; (1-\alpha)\,\big(1 - \text{SSIM}(\hat{d}, d^*)\big), \quad (1)$$

where $\hat{d}$ and d^* denote predicted and ground-truth depth, and $\alpha = 0.85$. To strengthen geometric priors in the lightweight encoder, we apply feature-level knowledge distillation from a pretrained dense prediction teacher, aligning intermediate decoder features via a layer-wise MSE loss.

Edge Optimization. For Jetson Nano deployment, the trained model is exported to ONNX and compiled with TensorRT under FP16 precision. For Raspberry Pi 5, we convert to TensorFlow Lite (TFLite) with INT8 post-training quantization calibrated on 500 KITTI training images, and accelerate inference with the XNNPACK delegate.

B. Module 2: Depth-Lifting and Trajectory Prediction

Depth-Lifting. Given a pedestrian bounding box (u, v, w, h) detected by YOLOv8-nano, the depth-lifting layer computes the pseudo-3D world coordinates of the pedestrian center:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \hat{d}(u_c, v_c) \cdot \mathbf{K}^{-1} \begin{bmatrix} u_c \\ v_c \\ 1 \end{bmatrix}, \quad (2)$$

where (u_c, v_c) denotes the bottom-center of the bounding box, $\hat{d}(u_c, v_c)$ is the median estimated depth within the lower 30% of the bounding box area (the foot region), and $\mathbf{K}$ is the camera intrinsic matrix. Using the foot region reduces boundary outliers and provides a stable ground-plane contact point.

Trajectory Predictor. The resulting pseudo-3D coordinate sequence $\{(X_t, Y_t, Z_t)\}_{t=1}^{T_{\text{obs}}}$ is fed to a lightweight two-layer LSTM predictor. The predictor observes $T_{\text{obs}} = 8$ frames and forecasts $T_{\text{pred}} = 12$ future frames at 10 Hz. We train the predictor on ground-truth 3D coordinates from KITTI tracking sequences and evaluate under two conditions: ground-truth depth (oracle) and estimated depth (Depth-Lift). The complete pipeline is illustrated in Fig. 1.

III. Experiments

A. Setup and Baselines

All models in this work were trained on an NVIDIA RTX 4070 GPU using PyTorch. The depth estimation network

[1]Hefei University of Technology. lizhang@hfut.edu.cn
[*]Corresponding author: Yi Zhang, 2024180223@mail.hfut.edu.cn
[2]Zhejiang University of Technology.
[3]University of Glasgow.

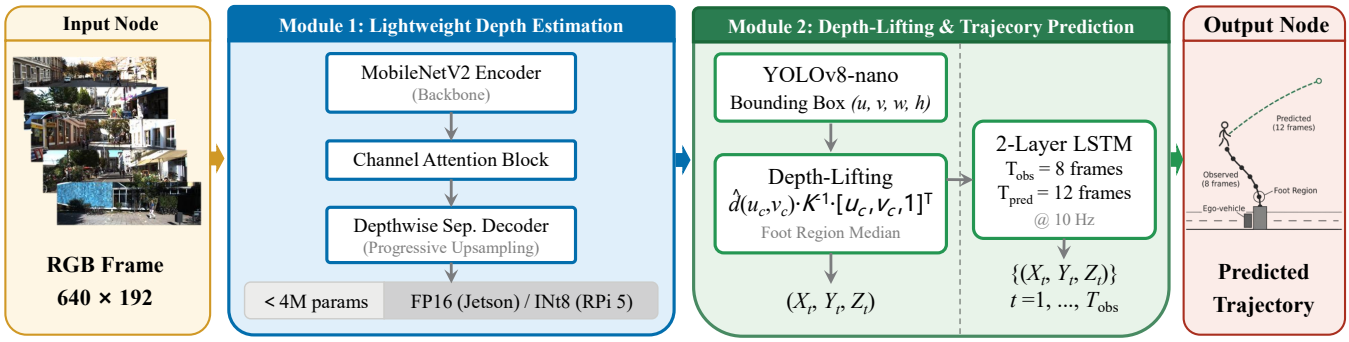

Fig. 1. System overview of Depth-Lift. Module 1 estimates dense depth from a single RGB frame via a lightweight MobileNetV2-based network; Module 2 lifts 2D pedestrian detections to pseudo-3D coordinates and feeds them to an LSTM trajectory predictor.

TABLE I
DEPTH ESTIMATION ACCURACY AND DOWNSTREAM TRAJECTORY PREDICTION PERFORMANCE ON KITTI

Depth Source	AbsRel↓	RMSE↓	ADE (m)↓	FDE (m)↓
GT Depth (Oracle)	—	—	0.43	0.82
Depth-Lift (Ours)	**0.121**	**4.51**	0.48	0.91
FastDepth [2]	0.152	5.14	0.56	1.07
Monodepth2 [5]	0.115	4.86	0.46	0.87

TABLE II
END-TO-END LATENCY AND RESOURCE CONSUMPTION ON EDGE PLATFORMS

Method	Jetson Nano (FP16)			Raspberry Pi 5 (INT8)		
	Lat.	FPS	Mem.	Lat.	FPS	Mem.
Depth-Lift	**38ms**	**26.3**	**210MB**	**105ms**	**9.5**	**85MB**
FastDepth	29ms	34.5	185MB	78ms	12.8	72MB
Monodepth2	112ms	8.9	480MB	385ms	2.6	310MB

was trained on the KITTI Eigen split [4] for 30 epochs with the Adam optimizer (learning rate 10^{-4}). The trajectory predictor was trained on KITTI tracking sequences containing annotated pedestrian trajectories. Edge deployment tests were conducted on two platforms: NVIDIA Jetson Nano (4 GB, FP16, TensorRT 8.x) and Raspberry Pi 5 (8 GB, INT8, TFLite + XNNPACK). All latency measurements were averaged over 200 consecutive frames at 640×192 input resolution. Baselines include **FastDepth** [2], a lightweight embedded model, and **Monodepth2** [5], a high-accuracy self-supervised model.

B. Results

Table I reports depth estimation accuracy and downstream trajectory prediction performance. With ground-truth depth as the oracle, Depth-Lift increased ADE and FDE by only **11.6%** and **11.0%**, respectively—a substantially smaller degradation than FastDepth (30.2%/30.5%). While Monodepth2 achieved slightly better depth metrics (AbsRel 0.115 vs. 0.121), its trajectory prediction gain over Depth-Lift was marginal (ADE: 0.46 vs. 0.48 m), and its computational cost was too high for edge deployment (Table II). These results suggest that once depth accuracy crosses a threshold ($\text{AbsRel} < 0.13$), further improvements yield diminishing returns for downstream prediction, making the lightweight model the practical choice for edge deployment.

Table II reports end-to-end latency on both platforms. Depth-Lift achieved **26.3 FPS** on Jetson Nano and **9.5 FPS** on Raspberry Pi 5, meeting the real-time requirement ($\geq$10 Hz) on the GPU-equipped platform and approaching it on the CPU-only device. Depth estimation accounted for approximately 65% of total pipeline latency, making it the primary bottleneck for future optimization.

IV. CONCLUSION

We present Depth-Lift, a lightweight monocular pipeline for pedestrian trajectory prediction on edge devices, achieving real-time inference on Jetson Nano (26.3 FPS) with less than 12% accuracy degradation relative to ground-truth depth.

At AbsRel ≈ 0.12, depth noise propagates to only $<$12% ADE increase, with errors concentrated at long range ($>$25 m) where collision risk is lower. Future work will incorporate pedestrian crossing intention prediction and validate the pipeline on mobile robots in pedestrian-rich environments.

REFERENCES

[1] L. Neumann and A. Vedaldi, "Pedestrian and ego-vehicle trajectory prediction from monocular camera," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021, pp. 10 199–10 207.

[2] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "FastDepth: Fast monocular depth estimation on embedded systems," in *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019, pp. 6101–6108.

[3] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4510–4520.

[4] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI Vision Benchmark Suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, 2012, pp. 3354–3361.

[5] C. Godard, O. Mac Aodha, M. Firman, and G. Brostow, "Digging into self-supervised monocular depth estimation," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 3828–3838.