Sensor Modality Fusion with CNNs for UGV Autonomous Driving in Indoor Environments

Naman Patel[1], Anna Choromanska[1], Prashanth Krishnamurthy[1], Farshad Khorrami[1]

***Abstract*— We present a novel end-to-end learning framework to enable ground vehicles to autonomously navigate unknown environments by fusing raw pixels from a front facing camera and depth measurements from LiDAR. A new deep neural network architecture is introduced for mapping the depth and vision from LiDAR and camera, respectively, to the steering commands. The network effectively performs modality fusion and reliably predicts steering commands even in the presence of sensor failures. The proposed network in trained on our own dataset, which we will publicly release, of LiDAR depth measurements and camera images taken in an indoor corridor environment. Comprehensive experimental evaluation to demonstrate the robustness of our network architecture is performed to show that the proposed deep learning neural network is able to fully autonomously navigate in the corridor environment. Furthermore, we demonstrate that the fusion of the camera and LiDAR modalities provides further benefits beyond robustness to sensor failures. Specifically, the multimodal fused system shows a potential to navigate around obstacles placed in the corridor environment and to handle changes in environment geometry (e.g., having additional paths such as opening of doors that were closed during training) without being trained for these tasks.**

I. Introduction

There have been significant advances in machine learning based approaches for robotic applications in recent years due to the advancements in deep learning techniques. Deep learning approaches have the ability to leverage large amounts of labeled and contextually rich data to give desired outputs. Some recent applications of deep learning that are relevant to this paper include autonomous car driving systems [1], [2]. This paper addresses the robust sensor fusion problem (Figure 1) in the context of building deep learning frameworks for self-driving vehicles equipped with multiple sensors (mainly camera and LiDAR) although the same methodology may be utilized for fusing a larger number of sensors. This work is motivated by two primary objectives. The first objective is related to the observation, which we also empirically verify, that the deep network trained jointly with camera and LiDAR data (i.e., without considering possibility of sensor failures) performs very poorly when one of the sensors is suddenly not available, i.e., one of the sensors intermittently going off-line. Hence, we seek to introduce a learning methodology that can handle intermittent sensor failures during testing. The second primary objective is to study the possibility of obtaining better performance characteristics with the multimodal fused system than with either sensor modality separately (e.g., see Figure 3). In other words, the underlying goal addressed by both the motivating objectives is to train the system to properly merge data to leverage both (or more) sensors and to be robust to sensor failures.

Our work focuses on the problem of navigating of an autonomous unmanned ground vehicle (UGV) using vision from camera and depth measurements from LiDAR in an indoor environment with deep learning. A novel approach to modality fusion is presented to generate steering commands for autonomous navigation of a ground vehicle. The proposed methodology naturally extends to the setting with multiple sensors, where one or more sensor data might be missing.

We propose a deep learning architecture for the sensor fusion problem that consists of two convolutional neural networks (CNNs), each consisting of a different input modality, which are fused with a gating mechanism. The gating mechanism is realized as a fully-connected network that is trained to generate environment-appropriate scalar weights for LiDAR and camera using the CNN-generated feature vectors. These scalar weights are then utilized to obtain the fused embedding including both modalities. The fused embedding is then passed through additional network layers to generate the steering command for the vehicle. The training of the network relies on the introduction of corrupted data in the training batches (to mimic sensor failures). This synthetic sensor failure introduction enables, in effect, the network to generalize better. The novel aspects of this paper are as follows:

- application of deep learning for the problem of indoor corridor tracking with a ground vehicle registering camera and LiDAR data,
- proposing a new deep learning architecture and training method for sensor fusion that leads to a system for autonomous driving of ground vehicles indoors that is robust to the presence of partial data from a single modality,
- experimental demonstration of the efficacy of the proposed system on our in-house developed ground vehicle that includes a real-time autopilot, single board computer with graphics processing unit (GPU), and integrated camera and LiDAR sensors,
- releasing a new dataset dedicated to the problem of autonomous driving of ground vehicles indoors.

The paper is organized as follows. Related literature is briefly summarized in Section II. The problem formulation is presented in Section III. The architectures for sensor fusion developed in this paper are discussed in Section IV, including an architecture based on a gating mechanism as well as two other architectures more similar to prior literature (though in a different context than in this paper). The training mechanisms are also discussed in Section IV. Empirical verification studies are presented in Section V. Finally, concluding remarks are provided in Section VI.

[1]All authors are with the Department of Electrical and Computer Engineering, NYU Tandon School of Engineering, 2 MetroTech Center, USA. naman.patel@nyu.edu, ac5455@nyu.edu, prashanth.krishnamurthy@nyu.edu, khorrami@nyu.edu

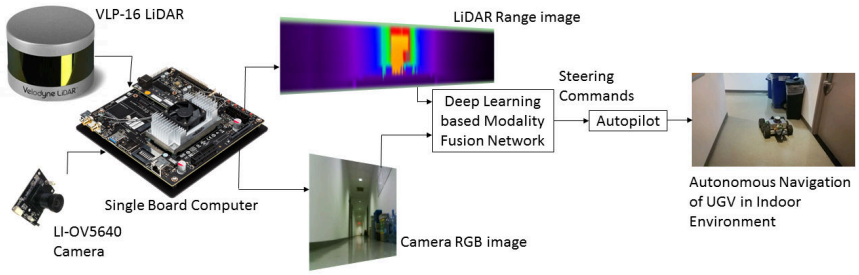

Fig. 1: End-to-end learning framework for autonomous navigation in indoor environment.

II. Related Work

Various aspects of robot autonomy has been extensively studied in the literature [3], [4]. For example, using Simultaneous Localization and Mapping (SLAM) based approaches, [5], [6], autonomous navigation in both indoor and outdoor environments has been studied using vision and depth based sensors, such as camera, stereo camera, and LiDAR. Obstacle avoidance and navigation in uncertain environments has been studied using various approaches [7]–[9]. Vision processing for indoor wall detection and corridor following has been studied using techniques such as optical flow [10] and visual servoing [11]. Reinforcement learning techniques have also been utilized to teach the mobile robot to avoid obstacles and navigate through the corridor using sensors such as a laser range finder [12].

With the advances in neural networks over the last few years, new toolsets are emerging for autonomous navigation of robots. For example, an online navigation framework relying on object recognition was presented in [13]. CNNs have been successfully used for learning driving decision rules for autonomous navigation [14] and for end-to-end navigation of a car using a single front facing camera [1] (also some debugging tools were developed for these autonomous systems to understand the visual cues that the network uses to produce a steering command, e.g. [15]). Visual navigation in simulated environment has been addressed using deep reinforcement learning in [16], [17]. Generative adversarial networks have also been used for aiding in the autonomous navigation tasks [18], [19]. Neural network based navigation in indoor environments has also been studied in multiple works including [20]–[24].

In the deep learning literature, fusion of different modalities has been studied for various applications in recent years such as in [25] for object detection using images and depth maps. Deep learning for a recurrent neural network [26] was applied to implicitly learn the dependencies between RGB images and depth map to perform semantic segmentation. In [27], RGB image and its corresponding 3D point cloud are used as inputs for 3D object detection. RGB image, optical flow, and LiDAR range images are combined to form a six channel input to a deep neural network [28] for object detection. The same network can also be used for different modalities to learn a joint representation [29]. RGB images and depth maps (HHA images) were fused in [30] for an indoor scene recognition application using a multi-modal learning framework and the learned features were classified using a support vector machine.

Compared to the prior works summarized above, the proposed system introduces several novel aspects as summarized in the introduction. Specifically, we introduce a new gating mechanism based architecture that enables modality fusion for robust end-to-end learning of autonomous corridor driving and improved training techniques that enable resiliency to sensor failure. The efficacy of the proposed approach is demonstrated through experimental studies on our UGV platform (Figure 2).

III. Problem Formulation

We address the problem of end-to-end learning of appropriate steering commands for a UGV to drive autonomously through an indoor environment using camera and LiDAR sensors. The proposed deep learning based system is trained using data recorded under human tele-operation of the UGV. Within this context, the objective of this paper is to explore effective network architectures and training techniques for fusion of the camera and LiDAR modalities to obtain robustness to a sensor failure and also to achieve performance characteristics superior to either what is achieved by each sensor separately (e.g., Figure 3).

The sensory inputs considered here for indoor navigation are vision (RGB image) and depth (LiDAR range image). The visual RGB image gives information about the type of environment and information regarding texture and colour of the objects present in the nearby environment whereas the depth range image gives complementary information to RGB channels in the form of the structure of the environment via depth measurements to points in the environment. In order to successfully navigate through a corridor, the ground vehicle has to use the most relevant information from the camera RGB image and depth range image in order to fuse it to predict the steering command for adjusting its heading. As illustrated in Figure 3, each sensor separately can have limitations in environment perception. There are also other complementary sensory performance characteristics of camera and LiDAR, e.g., sensitivity of a camera to lighting conditions, limitations of a LiDAR in detecting small objects due to typically significantly lower resolutions than a camera.

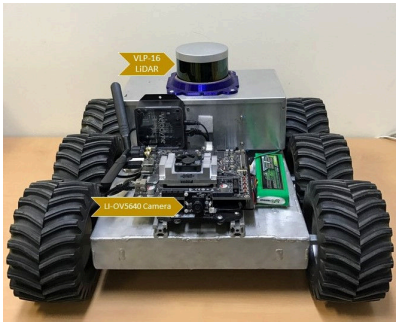

Fig. 2: Our unmanned ground vehicle system with integrated LiDAR and camera sensors.

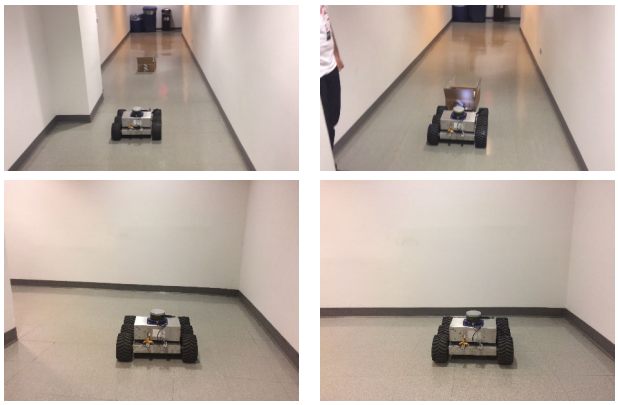

Fig. 3: Examples where using only camera or LiDAR generates undesirable behavior. In the top row, a LiDAR-only system does not detect the low-profile object (trash can) in front which would register only a few points in the LiDAR scan. In the second row, a camera-only system is not able to disambiguate left/right turns when approaching near a corner without any other discriminating object/lighting features (the pictures above are from behind the UGV; the onboard camera would just see a featureless wall). The fused system is able to successfully function in both these instances (i.e., moving around an obstacle and making an appropriate turn when approaching a featureless wall).

IV. Proposed Sensor Fusion Framework

In this section, we introduce our proposed sensor fusion framework, its deep learning based network architecture, and training implementation details.

A. System Framework

The LiDAR sensor (light detection and ranging) provides accurate range measurements to points in the environment at various azimuth and elevation angles relative to the sensor. Hence, the LiDAR sensor effectively provides three dimensional information on points in the local environment. While our LiDAR sensor provides 360 degree azimuth measurements, the forward-facing 180 degree part of the measurements is utilized since that is sufficient for indoor corridor navigation. The LiDAR measurements are encoded as a grayscale depth range image. RGB images are obtained from the camera. The depth range images are updated 10 times every second and the camera images are updated 30 times every second. Our framework uses the most recent depth range image and the camera image to predict a suitable steering command for navigating through the corridor. This steering command is used by the onboard autopilot to send appropriate signals to the motor controllers of the ground vehicle. A simple PID controller is used to control the heading of the ground vehicle, given the steering command and feedback from the motor encoders.

B. Network Architectures

Three network architectures are considered for the sensor fusion task described above. The primary architecture (Figure 4) which we denote *NetGated* is a gating based architecture described further below. We also consider two other architectures (which we denote as *NetEmb* and *NetConEmb*) that are more similar to prior literature; these networks are also described further below.

The architectures of *NetEmb*, *NetConEmb*, and *NetGated* are described in Tables I, II, and III, respectively. In *NetEmb* (which shares the same first 20 layers as *NetConEmb* and *NetGated*), feature maps from RGB image and LiDAR depth range image are extracted through a series of convolutional layers. Next, the features extracted from the convolutional layers in both the parallel networks are embedded into a feature vector using a fully connected network. The intuition behind embedding features is that the features extracted from image and depth range image will have the same dimension. This ensures that one modality does not have a greater effect on the result than the other due to unequal size. In *NetConEmb*, the convolutional feature maps are passed into a fully connected network. As shown in Table I, the network architecture consists of 8 convolutional layers and 1 fully connected network for each modality and 2 fully connected networks for information fusion from the two modalities. Each convolutional layer consists of 3x3 kernels which convolve through the input with a stride of 1 to generate feature maps which are then passed through Rectification (ReLU) non-linearity. The inputs are padded during convolution to preserve the spatial resolution. The feature maps are downsampled after every two convolution layers by max-pooling operation with a window size and stride of 2x2. All hidden layers including the fully connected layers are equipped with Rectification (ReLU) non-linearity. The network learns its parameters by minimizing the Huber loss (δ=1) between the predicted steering command and the command of the human driver. In *NetGated*, the embedded features constructed as in *NetEmb* are passed through a gating network to fuse the information from both the modalities which is then used to generate the steering command. The gating network takes the two embeddings obtained from RGB image and range image as input and outputs the corresponding two weights which are then used to perform a weighted sum of the embeddings. This weighted sum is then passed through two fully connected networks to obtain the steering command. Each of the considered network architectures is an end-to-end deep learning system that takes an RGB image and a LiDAR depth range image as input and fuses the modalities using a deep neural network to predict the appropriate steering command of the ground vehicle for autonomous navigation.

C. Implementation and Training

The inputs to the networks are the normalized RGB image with a field of view of 72° and the LiDAR range image which is cropped such that the front half with a field of view of 180° is visible. Both the modalities are normalized by making each channel of the modality in the training dataset zero mean with a standard deviation of 1. At testing time, the mean and standard deviation calculated during training are used to normalize the input.

To train the networks, camera and LiDAR datasets were obtained by manually driving the vehicle (with constant speed) through the corridor environment obtaining approximately the same amount of training data for straight motion, left turns, and right turns. A Leopard Imaging LI-OV5640 camera running at 30 frames per second is used as the vision sensor and a Velodyne VLP-16 LiDAR running at 10 rotations per second is used as the depth sensor. The data from the camera and LiDAR are logged at 30 and 10 frames per second, respectively.

The network was trained on a dataset of 14456 images and their corresponding range images. The images and range images were preprocessed by making all channels zero mean

	Layer Name	Layer Input (For RGB Image)	Layer Output	Kernel Size	Stride	No. Kernels	Layer Name	Layer Input (For LiDAR Range Image)	Layer Output	Kernel Size	Stride	No. Kernels
1	Spatial Convolution	3x120x160	16x120x160	3x3	1	16	Spatial Convolution	1x900x16	16x900x16	3x3	1	16
2	Rectified Linear Unit	16x120x160	16x120x160	-	-	-	Rectified Linear Unit	16x900x16	16x900x16	-	-	-
3	Spatial Convolution	16x120x160	16x120x160	3x3	1	16	Spatial Convolution	16x900x16	16x900x16	3x3	1	16
4	Rectified Linear Unit	16x120x160	16x120x160	-	-	-	Rectified Linear Unit	16x900x16	16x900x16	-	-	-
5	Max Pooling	16x120x160	16x60x80	2x2	2	-	Max Pooling	16x900x16	16x450x8	2x2	2	-
6	Spatial Convolution	16x60x80	32x60x80	3x3	1	32	Spatial Convolution	16x450x8	32x450x8	3x3	1	32
7	Rectified Linear Unit	32x60x80	32x60x80	-	-	-	Rectified Linear Unit	32x450x8	32x450x8	-	-	-
8	Spatial Convolution	32x60x80	32x60x80	3x3	1	32	Spatial Convolution	32x450x8	32x450x8	3x3	1	32
9	Rectified Linear Unit	32x60x80	32x60x80	-	-	-	Rectified Linear Unit	32x450x8	32x450x8	-	-	-
10	Max Pooling	32x60x80	32x30x40	2x2	2	-	Max Pooling	32x450x8	32x225x4	2x2	2	-
11	Spatial Convolution	32x30x40	48x30x40	3x3	1	48	Spatial Convolution	32x225x4	48x225x4	3x3	1	48
12	Rectified Linear Unit	48x30x40	48x30x40	-	-	-	Rectified Linear Unit	48x225x4	48x225x4	-	-	-
13	Spatial Convolution	48x30x40	48x30x40	3x3	1	48	Spatial Convolution	48x225x4	48x225x4	3x3	1	48
14	Rectified Linear Unit	48x30x40	48x30x40	-	-	-	Rectified Linear Unit	48x225x4	48x225x4	-	-	-
15	Max Pooling	48x30x40	48x15x20	2x2	2	-	Max Pooling	48x225x4	48x113x2	2x2	2	-
16	Spatial Convolution	48x15x20	64x15x20	3x3	1	64	Spatial Convolution	48x113x2	64x113x2	3x3	1	64
17	Rectified Linear Unit	64x15x20	64x15x20	-	-	-	Rectified Linear Unit	64x113x2	64x113x2	-	-	-
18	Spatial Convolution	64x15x20	64x15x20	3x3	1	64	Spatial Convolution	64x113x2	64x113x2	3x3	1	64
19	Rectified Linear Unit	64x15x20	64x15x20	-	-	-	Rectified Linear Unit	64x113x2	64x113x2	-	-	-
20	Max Pooling	64x15x20	64x8x10	2x2	2	-	Max Pooling	64x113x2	64x57x1	2x2	2	-
21	Flatten	64x8x10	5120	-	-	-	Flatten	64x57x1	3648	-	-	-
22	Fully Connected	5120	512	-	-	-	Fully Connected	3648	512	-	-	-
23	Rectified Linear Unit	512	512	-	-	-	Rectified Linear Unit	512	512	-	-	-
24	Concatenate	512,512	1024	-	-	-	-	-	-	-	-	-
25	Fully Connected	1024	32	-	-	-	-	-	-	-	-	-
26	Rectified Linear Unit	32	32	-	-	-	-	-	-	-	-	-
27	Fully Connected	32	10	-	-	-	-	-	-	-	-	-
28	Rectified Linear Unit	10	10	-	-	-	-	-	-	-	-	-
29	Fully Connected	10	1	-	-	-	-	-	-	-	-	-

TABLE I: *NetEmb*: Deep learning based modality fusion architecture using embeddings. The left side of the table is for processing of the RGB image from the camera and the right side of the table is for processing of the depth range image from the LiDAR. The feature vectors (of length 512) constructed from camera and LiDAR are concatenated at layer 24.

	Layer Name	Layer Input	Layer Output	Layer Name	Layer Input	Layer Output
1 .. 20	Same as Table 1					
21	Flatten	64x8x10	5120	Flatten	64x57x1	3648
22	Concatenate	5120,3648	8768	-	-	-
23	Fully Connected	8768	1024	-	-	-
24	Rectified Linear Unit	1024	1024	-	-	-
25	Fully Connected	1024	32	-	-	-
26	Rectified Linear Unit	32	32	-	-	-
27	Fully Connected	32	10	-	-	-
28	Rectified Linear Unit	10	10	-	-	-
29	Fully Connected	10	1	-	-	-

TABLE II: *NetConEmb*: Fusion architecture where the convolutional feature maps are directly passed through a fully connected network instead of first converting them into feature embeddings as done in *NetEmb*. The first 20 layers are identical to *NetEmb*.

	Layer Name	Layer Input	Layer Output	Layer Name	Layer Input	Layer Output
1 .. 20	Same as Table 1					
21	Flatten	64x8x10	5120	Flatten	64x57x1	3648
22	Fully Connected	5120	512	Fully Connected	3648	512
23	Rectified Linear Unit	512	512	Rectified Linear Unit	512	512
24	Concatenate	512,512	1024	-	-	-
25	Fully Connected	1024	64	-	-	-
26	Rectified Linear Unit	64	64	-	-	-
27	Fully Connected	64	2	-	-	-
28	Split	2	1,1	-	-	-
29	Multiplication with output 23	1	512	Multiplication with output 23	1	512
30	Addition	512,512	512	-	-	-
31	Fully Connected	512	32	-	-	-
32	Rectified Linear Unit	32	32	-	-	-
33	Fully Connected	32	1	-	-	-

TABLE III: *NetGated*: Fusion architecture with gating mechanism based on computing scalar weights from the feature embeddings and then constructing a combination of the feature embeddings based on the scalar weights. The first 20 layers are identical to *NetEmb*.

with a standard deviation of 1. The network was trained using Adagrad optimizer with a learning rate of 0.01. The learning rate is decreased to 0.005 after 30 epochs and to 0.001 after 60 epochs. Bias terms for all the layers in the networks are disabled.

Our end-to-end learning framework learns to predict the appropriate steering command by learning the weights of the network which minimize the Huber loss between the predicted steering commands and the recorded human steering commands. We use the Huber loss instead of mean square error since an instability due to divergence of the gradients was noted with mean square error loss. The Huber loss was introduced in [31] for bounding box regression and is given by

$$L(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2, & \text{for } \|y - f(x)\| \leq 1 \\ \|y - f(x)\| - 0.5, & otherwise \end{cases} \quad (1)$$

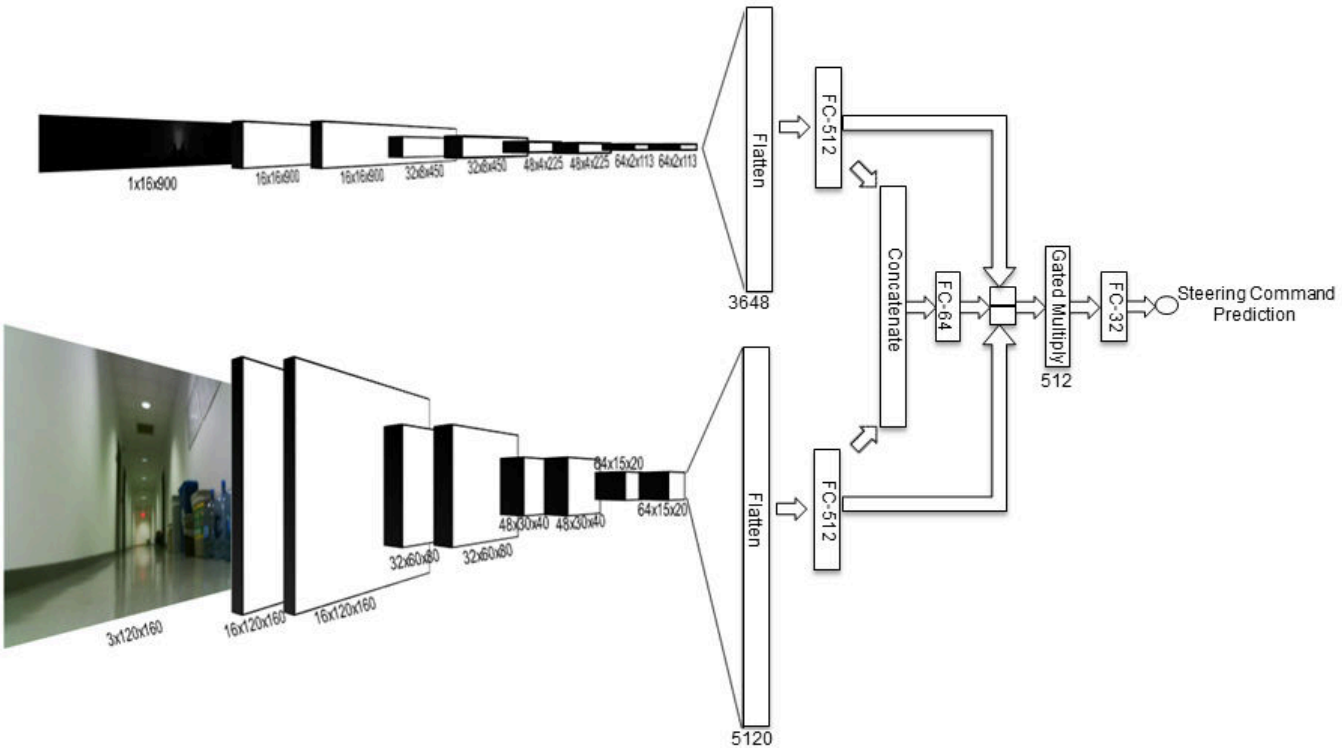

Fig. 4: *NetGated*: Our proposed architecture for deep learning based fusion of camera and LiDAR sensors.

To train the network to be able to utilize both sensors when available and also to be robust to the possibility of sensor failure, the training of the network was performed in two stages. In the first stage of training, the network is trained with the corresponding LiDAR depth range images and camera RGB images for each time step as input. In the second stage, the training of the network is continued with corrupted data (i.e., with one modality shut down to mimic sensor failure). Specifically, the network is trained with 40% corrupted data for each epoch out of which 50% data is with the camera shut off (i.e., zero values for all elements in the RGB image) and 50% is with the LiDAR shut off. The same training procedure as described above was applied to each of the network architectures described in Section IV-B and the training was stopped for each network at the same final accuracy for the training set. It is seen in Section V that the proposed network architecture and training approach provides robust performance under sensor failures and implicitly learns to use the relevant information from both modalities to generate steering angle predictions. We compare the networks trained only on the original dataset and the networks retrained with the corrupted dataset, and show that the networks retrained with the corrupted dataset provide superior performance when one of the modalities fail and retains the performance of the originally trained networks when both the sensors are present.

D. Differential drive of the vehicle

The predicted steering commands are mapped to the motor actuation commands to drive the ground vehicle. This mapping is done as part of the in-house developed firmware on our real-time autopilot system which takes the steering commands from the networks and outputs the control signals to the motor drivers on the ground vehicle. The speeds of the DC motors on the vehicle are controlled by Pulse Width Modulated (PWM) pulses generated by the autopilot with a period of $20ms$ and an ON period varying from $1ms$ to $2ms$. The steering commands output by the network lie in the interval [-100, 100] and are mapped linearly to PWM signal ON periods of $1ms$ (5%) to $2ms$ (10%) and utilized as an additive differential drive actuation to the motors on the left and right sides of the vehicle, which essentially results in a left turn when the steering command is negative, a right turn when the steering command is positive, and straight movement when the steering command is zero. The magnitude of the steering command controls the sharpness of the turn.

V. Experimental Studies

In this section, experimental results are presented for the three previously described architectures (NetGated, NetConEmb, and NetEmb) first with training using both camera and LiDAR and then with retraining using the corrupted data as discussed above.

A. Performance of the Different Network Architectures

In order to evaluate the performance of the proposed architectures (namely NetEmb, NetConEmb and NetGated as described in Table I, II and III, respectively), steering command predictions of each network were compared with the steering commands of a human controller. This evaluation was done using a different dataset (test dataset) than the one used for training. The results of each of the architectures compared to the human controller are shown in Figure 5 where the steering commands given by a human controller (during tele-operation of the UGV) are denoted as the ground truth.

As shown in Figure 5, the utilization in *NetEmb* of an equal-size embedding (constructed using a fully connected

layer) for each modality after the last convolution layer provides better performance than *NetConEmb* as hypothesized in Section IV-B. The *NetEmb* architecture performs better when one of the modalities is switched off and also oscillates less compared to the *NetConEmb* architecture. As discussed in Section IV-B, the much larger number of features for the camera after the last convolution layer than for LiDAR causes the output to become more dependent on one modality in *NetConEmb* resulting in unbalanced fusion which makes the steering commands oscillate more, similar to the behavior of the camera only network. We also observe that the fusion architectures *NetEmb* and *NetConEmb* are biased towards moving right(negative steering command) compared to the ground truth human command.

Motivated by the observations above, fully connected layer based embeddings for each modality were also used in the *NetGated* architecture. An additional advantage of using an equal-size embedding for each modality is that it is then easier and more natural to fuse the embeddings by the learned gated weights by simply taking a weighted linear combination. As shown in Figure 5, the *NetGated* architecture based network learns to move straight with fewer oscillations than even the human controller. The fusion of camera and LiDAR results in a smoother output than a LiDAR only system as shown in the Figure 5.

Since a desirable characteristic of motion in the indoor corridor environment is that the ground vehicle should approximately track the center of the corridor and should not come too close to walls when turning, a useful metric for performance of the system is the distance of the vehicle to the left side and right side walls/objects. Since there are several objects such as trash cans and also empty spaces and open office doors at some locations, the closest distances on the left and right sides varies quite significantly even for an "ideal" motion. To remove such "noise" effects, an effective performance metric is the variance (rather than mean) of distances to the left side and right side walls/objects. These variances were recorded under fully autonomous mode (i.e., with the network providing the commands to the autopilot) with the different networks for both clockwise and counterclockwise directions. The measured variances for a clockwise motion through the building corridor environment (one complete floor of the building) are shown in Table IV and it is noted that the *NetGated* network architecture provides the best (lowest) variance; a similar observation was also noted for a counterclockwise motion.

	Network Type	Network Input	Left Wall Distance Variance (in m)	Right Wall Distance Variance (in m)
1	NetConEmb network	Camera and LiDAR	0.2306	0.1876
2	NetEmb	Camera and LiDAR	0.1416	0.1245
3	NetGated	Camera and LiDAR	0.1008	0.0575

TABLE IV: Variance of minimum distances to the wall for a clockwise trajectory under fully autonomous mode.

For all the considered network architectures, it is noted in Figure 5 that the system trained on a dataset with both camera and LiDAR data is not directly robust to the possibility of a sensor failing (i.e., only one sensor modality available and the other zeroed out). For example, *NetGated* places much more trust on the LiDAR input than on the camera input and does not provide any reasonable performance in the event of a LiDAR failure. Hence, in order to achieve robustness to sensor failure, we introduce the training strategy described in Section sec:training to continue retraining of the network with corrupted data generated by synthetically turning off either of the two modalities. The performance of the retrained *NetGated* network (after retraining with this corrupted data based technique) is compared below with the original trained *NetGated* network and the human controller.

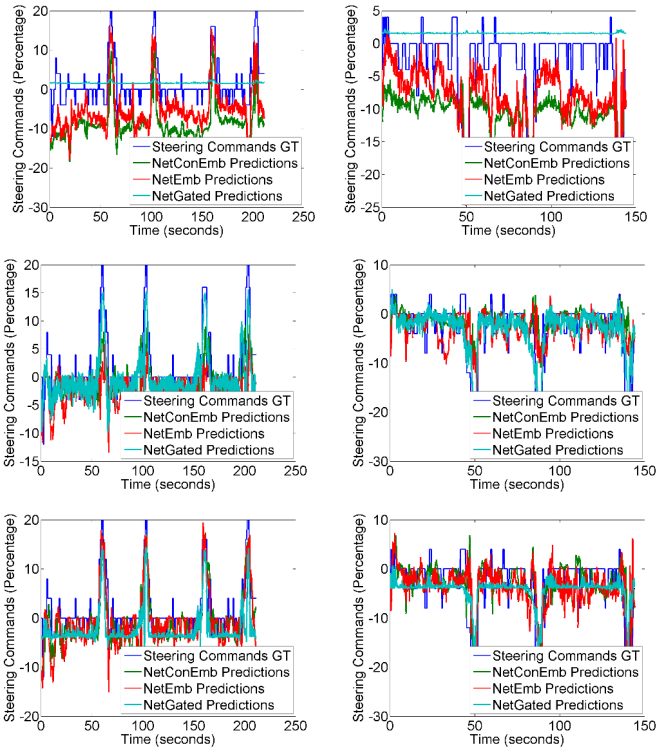

Fig. 5: Steering command predictions using the different network architectures under cases of only camera working (top row), only LiDAR working (middle row), and both camera and LiDAR working (bottom row). In each row, the left side and right side pictures show clockwise (right turns) and counterclockwise (left turns) navigations of a complete floor of a corridor environment. Ground truth (GT) refers to the recorded human inputs.

B. Performance of Network Retrained with Corrupted Data

The *NetGated* architecture when retrained with corrupted data as explained in IV-C achieves better performance than the network only trained with the original dataset. As shown in Figure 6, when both camera and LiDAR are working, both the original and retrained networks perform well and have very similar performance; but, when one of the modalities is shut off, the retrained network performs better. The performance characteristics of the retrained *NetGated* network was also evaluated (under the possibilities of both camera and LiDAR available, only camera available, and only LiDAR available) using the distance variance based metric as discussed above under fully autonomous operation of the UGV (Figure 7). It is noted in Table V that the

retrained *NetGated* network achieves autonomous navigation through the corridor although the camera-only and LiDAR-only situations provide lower performance (i.e., higher distance variances) than the camera+LiDAR situation.

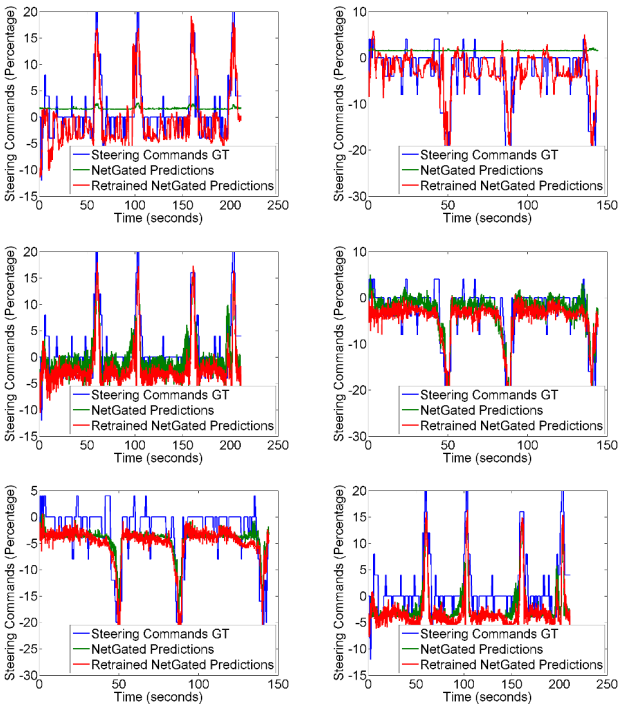

Fig. 6: Steering command predictions using the *NetGated* trained only with the camera+LiDAR dataset and the *NetGated* retrained with corrupted data under cases of only camera working (top row), only LiDAR working (middle row), and both camera and LiDAR working (bottom row). As in Figure 5, the left side and right side pictures in each row show clockwise and counterclockwise navigations of the corridor environment.

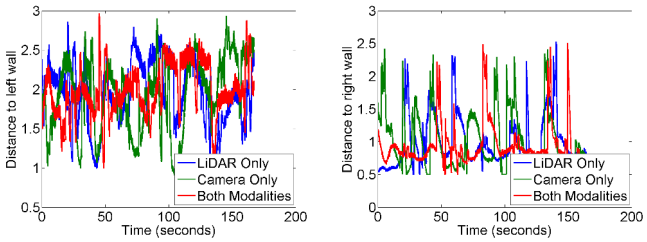

Fig. 7: Distances of the ground vehicle from the left wall and right wall for clockwise (left) and counterclockwise (right) autonomous navigations in the corridor environment.

C. Autonomous Indoor Navigation of the Ground Vehicle

The proposed deep learning based system is able to fully autonomously navigate through the indoor corridor environment. With the retraining procedure discussed above, the system is robust to failure of either of the camera or LiDAR sensor modalities. Auutonomous navigation through corridors is shown in Figure 8. The ground vehicle is able to appropriately make turns at corners enabling it to be equidistant from the walls after the turn. It is also able to navigate through narrower spaces (e.g., between trash cans) as shown in the middle two rows of Figure 8.

Furthermore, the system is able to implicitly learn to avoid static and dynamic obstacles as shown in Figure 9 without

	Network Type	Network Input	Left Wall Distance Variance (in m)	Right Wall Distance Variance (in m)
1	NetGated Architecture	Camera	0.1843	0.1539
2	NetGated Architecture	LiDAR	0.1358	0.1046
3	NetGated Architecture	Camera and LiDAR	0.1013	0.0964

TABLE V: Variance of minimum distances to the walls using the retrained NetGated architecture when various modalities are turned off for a clockwise trajectory.

Fig. 8: Examples of autonomous navigation in an indoor environment: left turn (top two rows), straight motion (middle two rows), right turn (bottom two rows). These pictures were taken from behind the UGV.

ever being specifically trained for this purpose, i.e., the training dataset did not include any specific demonstrations of moving around obstacles. Also, the fused camera+LiDAR network performs better in several scenarios than the camera-only or LiDAR-only situations. While a LiDAR-only network can enable avoiding of obstacles such as humans, it does not typically detect small (low-profile) objects since these register only a few points in the LiDAR scan. In such situations, the camera image enables the fused network to avoid the obstacle. When approaching a visually featureless wall, a camera-only system can not disambiguate between left and right turns while the LiDAR enables the fused network to detect the appropriate turn. When passing an open door or other open spaces (such as a short corridor leading to a dead end), the LiDAR being a more geometric sensor measuring distances to points tends to make a LiDAR-only system move towards the open space. However, the visual features implicitly detected from the camera enable the fused network to completely ignore such an "unintended" open space and remain at the center of the corridor (Figure 10).

VI. Conclusion

An end-to-end CNN based framework was developed for fusing vision and depth measurements from camera and LiDAR, respectively, for autonomous navigation of a ground robot in an indoor environment. Multiple network

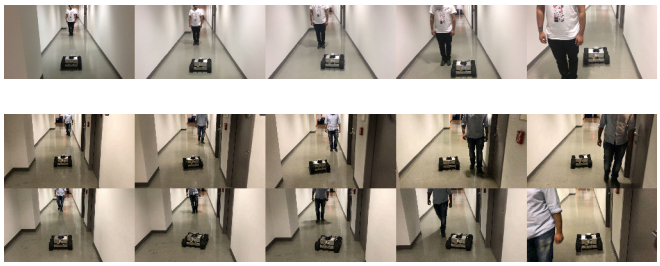

Fig. 9: Examples of avoidance of static obstacles (top row) and dynamic obstacles (bottom row) by the UGV. These pictures were taken from behind the UGV.

Fig. 10: Comparison of LiDAR-only network vs fused network in presence of "spurious" open spaces (e.g., open doors). With only LiDAR, there is marked deviation (from the center of the corridor) towards open doors or other open spaces.

architectures were considered including a novel gating based network architecture. A two-stage training methodology was proposed to achieve robustness to the possibility of sensor failure and to properly leverage the complementary strengths of the two sensors so as to achieve better performance than with either sensor separately. It was experimentally demonstrated that the proposed deep learning based system is able to fully autonomously navigate in the indoor environment with robustness to failure of either the camera or the LiDAR.

Topics for future work include improvements to the network architectures (and training algorithms) to make them more robust to additive random noise, environment perturbations caused due to vibration of the camera, and sensor placement. We also plan to experiment with recurrent neural networks due to their inherent property of capturing temporal dependencies between inputs and extend the system to outdoor environments.

REFERENCES

[1] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, *et al.*, "End to end learning for self-driving cars."

[2] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," in *NIPS Workshop on Learning, Inference and Control of Multi-Agent Systems*, 2016.

[3] S. Thrun, W. Burgard, and D. Fox, "Probabilistic robotics (intelligent robotics and autonomous agents)," 2005.

[4] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 2, pp. 237–267, 2002.

[5] Y. Ono, H. Uchiyama, and W. Potter, "A mobile robot for corridor navigation: a multi-agent approach," in *Proceedings of the 42nd annual Southeast regional conference*. ACM, 2004, pp. 379–384.

[6] H. Lategahn, A. Geiger, and B. Kitt, "Visual slam for autonomous ground vehicles," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1732–1737.

[7] P. Krishnamurthy and F. Khorrami, "GODZILA: A low-resource algorithm for path planning in unknown environments," *Journal of Intelligent and Robotic Systems*, vol. 48, no. 3, pp. 357–373, March 2007.

[8] ——, "A hierarchical control and obstacle avoidance system for Unmanned Sea Surface Vehicles," in *Proceedings of the IEEE Conference on Decision and Control/ European Control Conference*, Dec 2011, pp. 2070–2075.

[9] G. Brooks, P. Krishnamurthy, and F. Khorrami, "Humanoid robot navigation and obstacle avoidance in unknown environments," in *Proceedings of the Asian Control Conference*, June 2013.

[10] A. Dev, B. Krose, and F. Groen, "Navigation of a mobile robot on the temporal development of the optic flow," in *Intelligent Robots and Systems, 1997. IROS'97., Proceedings of the 1997 IEEE/RSJ International Conference on*, vol. 2. IEEE, 1997, pp. 558–563.

[11] F. Pasteau, V. K. Narayanan, M. Babel, and F. Chaumette, "A visual servoing approach for autonomous corridor following and doorway passing in a wheelchair," *Robotics and Autonomous Systems*, vol. 75, pp. 28–40, 2016.

[12] W. D. Smart and L. P. Kaelbling, "Effective reinforcement learning for mobile robots," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 4. IEEE, 2002, pp. 3404–3410.

[13] Z. Zheng, X. He, and J. Weng, "Approaching camera-based real-world navigation using object recognition," *Procedia Computer Science*, vol. 53, pp. 428–436, 2015.

[14] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2722–2730.

[15] M. Bojarski, A. Choromanska, K. Choromanski, B. Firner, L. D. Jackel, U. Muller, and K. Zieba, "Visualbackprop: visualizing cnns for autonomous driving," *CoRR*, vol. abs/1611.05418, 2016.

[16] A. A. Rusu, M. Vecerik, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, "Sim-to-real robot learning from pixels with progressive nets," *arXiv preprint arXiv:1610.04286*, 2016.

[17] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," *arXiv preprint arXiv:1609.05143*, 2016.

[18] E. Santana and G. Hotz, "Learning a driving simulator," *arXiv preprint arXiv:1608.01230*, 2016.

[19] A. Ghosh, B. Bhattacharya, and S. B. R. Chowdhury, "Sad-gan: Synthetic autonomous driving using generative adversarial networks," *arXiv preprint arXiv:1611.08788*, 2016.

[20] G. Chronis and M. Skubic, "Experiments in programming by demonstration: Training a neural network for navigation behaviors," in *Proceedings of the International Symposium on Robotics and Automation (ISRA)*, 2000.

[21] M. Jonsson, P.-A. Wiberg, and N. Wickström, "Vision-based low-level navigation using a feed-forward neural network," in *International Workshop on Mechatronical Computer Systems for Perception and Action (MCPA'97), Pisa, Italy, Feb. 10-12, 1997*, 1997, pp. 105–111.

[22] M. Meng and A. C. Kak, "Neuro-nav: a neural network based architecture for vision-guided mobile robot navigation using non-metrical models of the environment," in *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*. IEEE, 1993, pp. 750–757.

[23] V. N. Murali and S. T. Birchfield, "Autonomous navigation and mapping using monocular low-resolution grayscale vision," in *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW'08. IEEE Computer Society Conference on*. IEEE, 2008, pp. 1–8.

[24] K. K. Narayanan, L.-F. Posada, F. Hoffmann, and T. Bertram, "Situated learning of visual robot behaviors," in *International Conference on Intelligent Robotics and Applications*. Springer, 2011, pp. 172–182.

[25] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *European Conference on Computer Vision*. Springer, 2014, pp. 345–360.

[26] C. Hazirbas, L. Ma, C. Domokos, and D. Cremers, "Fusenet: Incorporating depth into semantic segmentation via fusion-based cnn architecture," in *Proc. ACCV (Vol. 2)*. Springer, 2016.

[27] S. Song and J. Xiao, "Deep sliding shapes for amodal 3d object detection in rgb-d images," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[28] M. Giering, V. Venugopalan, and K. Reddy, "Multi-modal sensor registration for vehicle perception via deep neural networks," in *High Performance Extreme Computing Conference (HPEC), 2015 IEEE*, Sept 2015.

[29] L. Castrejon, Y. Aytar, C. Vondrick, H. Pirsiavash, and A. Torralba, "Learning aligned cross-modal representations from weakly aligned data," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[30] H. Zhu, J.-B. Weibel, and S. Lu, "Discriminative multi-modal feature fusion for rgbd indoor scene recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[31] R. Girshick, "Fast r-cnn," in *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.