FINO-Net: A Deep Multimodal Sensor Fusion Framework for Manipulation Failure Detection

Arda Inceoglu[1], Eren Erdal Aksoy[2], Abdullah Cihan Ak[1], Sanem Sariel[1]

Abstract— We need robots more aware of the unintended outcomes of their actions for ensuring safety. This can be achieved by an onboard failure detection system to monitor and detect such cases. Onboard failure detection is challenging with a limited set of onboard sensor setup due to the limitations of sensing capabilities of each sensor. To alleviate these challenges, we propose FINO-Net, a novel multimodal sensor fusion based deep neural network to detect and identify manipulation failures. We also introduce FAILURE, a multimodal dataset, containing 229 real-world manipulation data recorded with a Baxter robot. Our network combines RGB, depth and audio readings to effectively detect failures. Results indicate that fusing RGB with depth and audio modalities significantly improves the performance. FINO-Net achieves %98.60 detection accuracy on our novel dataset. Code and data are publicly available at https://github.com/ardai/fino-net.

I. INTRODUCTION

The use of service robots in domestic environments raises some important ethical concerns which must be taken into account in robot designs. One crucial concern is execution safety in such environments where robots work with humans and everyday objects. Although robots are equipped with well designed actions, unintended or unsafe situations may arise from failed executions of these actions in the real world due to either faults in perception, failures in actuation, or unforeseen events. Physical interactions with objects may result in collisions, fall downs, splits, or overturnings. Robots should monitor these types of interactions to protect objects in interaction, humans in collaboration, themselves, and their environments.

The primary step in execution safety is to be equipped with essential skills to detect failures or unsafe situations. After then, predictions and precautions are possible. Execution monitoring and safety have been well investigated research areas [1], [2] and several methods exist for both ground and aerial robots [3]–[5]. However, there still exist several research questions for service robots working in unstructured environments. This is the main motivation behind our work. We investigate how robots can effectively detect manipulation failures as a first step in safe execution. We define a *failure* as an unintended outcome of an action

Fig. 1: Snapshots from a failed place-in-container execution.

execution. Sample failures that we focus on are tabletop manipulation actions: *push*, *pick&place*, *place-in*, *put-on*, and *pour*. A sample *place-in* execution failure of our humanoid robot is given in Fig. 1. In this sample, the robot fails in putting a plastic pear into a bowl full of other items.

It is almost impossible to detect and identify failures without perception. Furthermore, multi sensory integration is more desirable to take advantages of each sensor's perceptual contribution [6]. Multimodal sensor fusion methods are extensively used for manipulation, and execution monitoring also benefits from the use of multimodal data [7]–[10]. However, there exist potential challenges to deal with multimodal sensor fusion: (i) different data formats, (ii) different operating regimes/frequencies, (iii) inter and/or intra conflicting observations of sensors.

To address the above mentioned challenges, we propose a deep multimodal sensor fusion network for manipulation failure detection, named Failure Is Not an Option (FINO)-Net. FINO-Net effectively detects manipulation failures by fusing visual (RGB and depth) and audio modalities. Our network also adopts early fusion to combine RGB and depth frames while employing late fusion to combine vision and audio data. To capture spatio-temporal features in sensory observations, modalities are processed individually with a series of convolutional and convolutional-LSTM layers, then the latent space representations are combined for detecting possible failures.

Our earlier work involves an analysis on contributions of different sensor modalities on failure detection [11]. Based on this analysis, we developed a Hidden Markov Model-based failure detection system where the features

are collected through classifiers [12]. Although our new FINO-Net model uses the same modalities as input, it does not rely on precomputed features. Instead, FINO-Net processes raw sensory data in an end-to-end manner for failure detection.

The main contributions of this paper are as follows:

- We introduce a novel deep neural network model that fuses multimodal sensory readings to detect possible failure types.
- We further analyze the unique contribution of each sensing modality on the failure detection task.
- We release a multimodal (RGB, depth and audio) real-world dataset, named FAILURE, with 5 different manipulation types and in total 229 manipulation scenarios. We evaluate the performance of FINO-Net on this dataset and compare with a VGG-based model.

II. RELATED WORK

We first review the earlier literature on failure detection, execution monitoring, and multimodal sensor fusion. We then discuss some previously published manipulation datasets and elaborate on why we need to introduce our own new dataset.

A. Failure Detection and Execution Monitoring

In the literature, *failure (a.k.a fault or anomaly) detection* and *execution monitoring* keywords are used interchangeably. The work in [13] summarizes deep learning based anomaly detection for various application domains. From the robotics perspective, there are model-based and model-free approaches proposed for execution monitoring [14]. The former approaches compare the already known models with observations, whereas the latter use sensory observation to make predictions [1], [2].

Among earlier model-based approaches, Kalman Filter (KF) [3], kinematic model [4], and residual [5] based systems are proposed to detect and identify mechanical and sensor failures. Execution models [15] and stochastic action models [16] are used to detect and correct [17] execution anomalies. In [18], action models are extended to detect and recover from failures by repairing the plan. Additionally, Description Logic (DL) [19], Temporal Action Logic (TAL) [20] and Metric Temporal Logic (MTL) [7] formulas are defined for execution monitoring. Furthermore, the work in [21] uses a domain-specific language based approach to monitor software components.

On the other hand, model-free execution monitoring methods employ pattern recognition techniques [22]. The recent work in [23] extends planning with a vision-based execution monitoring system to search for target objects in the scene to ensure planning pre-conditions are satisfied. In [24], different preprocessing techniques for introspective data are analysed to detect gearbox failures for industrial robots. Non-parametric Bayesian models are also used for execution monitoring in robotics [25]. Non-parametric Hidden Markov Models (HMMs) representing spatio-temporal dynamics of anomalies are applied in [26] to detect and classify anomalies

by considering only introspective data (i.e., force-torque, velocity, and tactile).

The method is evaluated on pick&place tasks by using 6 varying sized and shaped objects on a Baxter robot. A multimodal execution monitoring system is proposed for the assistive feeding task in [8]. The authors adopt LSTM-based variational autoencoders to process multimodal input from a sensor set including a camera, a microphone, a joint encoder and a force sensor.

In their earlier work, multivariate Gaussian HMMs are explored [9].

Our work differs from the existing studies as we address the problem of detecting manipulation failures which are mainly caused by uncertainties in perception and execution. Unlike the works in [8], [9], where the focus is rather on the failures emerging during human-robot interaction (e.g., failures due to collisions, face occlusions, utensil misses or sound from the user, etc.), we investigate the robot-object interaction failures observed over the course of an object manipulation. Instead of hand-crafted features such as gripper status, audio events, and object displacements used in [11], [12] or sound energy, spoon position, and mouth position used in [8], [9], our proposed perception framework learns feature representations directly from the raw multimodal sensory data in an end-to-end fashion.

B. Multimodal Sensor Fusion

Multimodal fusion has a wide variety of applications with different kinds of input modalities [27]. The work in [28] fuses RGB and depth images for object recognition. RGB-D and audio data are combined in [29] for human action recognition. In [7], RGB-D camera, sonar, microphone and tactile sensor data are integrated, via high level predicates extracted from sensory data, in order to detect different kind of execution failures for mobile robots.

One of the main challenges in multimodal sensor fusion is determining when to combine modalities. Prior works introduce early, intermediate, and late fusion based architectures. Early fusion combines low-level features (i.e., raw observations), whereas late fusion incorporates high-level features (i.e., close to prediction). Intermediate architectures, where early and late fusions are intertwined, cover different strategies to fuse intermediate-level features. Variety of architectures are summarized in [30]. In this work, we adopt a combination of early and late fusion methods to combine different sensing modalities.

C. Manipulation Datasets

Recently, there have been great efforts to collect large scale robot manipulation data. These efforts, however, center around manipulation skill learning tasks and ignore failures emerging during manipulation executions. For example, RoboTurk [31], [32] is a crowdsourced simulation tool to collect teleoperated manipulation data for imitation learning. The dataset also includes real robot RGB-D frames for the following tasks: object search, tower stacking and laundry layout. RoboNet [33] provides 15 million video frames, from

7 different robot platforms for push and pick&place tasks. In addition, the work in [34] presents a large scale grasping dataset with a multi robot platform, and in [35] a pushing dataset is introduced. To the best of our knowledge, there exists no publicly available multimodal datasets on robot manipulation failures. The FAILURE dataset introduced here, as the first of its kind, rather focuses on the failed attempts of various robot manipulation actions (e.g., push, pour, pick&place, etc.) while collecting multimodal sensor readings such as RGB images, depth data, and audio waves.

III. METHOD

In this section, we first describe the problem, then present the details of FAILURE dataset and FINO-Net architecture.

A. Problem Description

In order to detect manipulation failures, the changes in the scene should be monitored using multiple sensory modalities [12] [11]. In this work, we model the failure detection task as a classification problem as follows. Let there be $M \in \{1, 2, ...\}$ sensing modalities where $m \in M$ is modality index. Let also D be the dataset containing $|D| = N$ multimodal observation sequences, $x_{t_m,i}^m$ is observation, t_m is the time index for modality m, i is the recording index and $y \in \{success, fail\}$ is the class label:

$$D = \{\{(x_{1,i}^{(m)}, ..., x_{t_m,i}^{(m)})\}_{m=1}^M, y_i\}_{i=1}^N \qquad (1)$$

The goal is to detect failures by learning a function $\Phi(\cdot)$ that maps multimodal sensory data to a label as either success or failure.

B. Setup and Data Collection

We introduce FAILURE, a multimodal dataset, containing 229 real-world manipulation data recorded with a Baxter robot. The experimental setup is presented in Fig. 2. We use a Baxter humanoid robot with the following equipments: a parallel gripper, an Asus Xtion Pro RGB-D camera mounted on the head, a PSEye microphone mounted on the lower torso and the Baxter's default RGB camera mounted on the left arm which is also used to monitor manipulation execution. The object sets used in the experiments are shown in Fig. 3.

1 Asus Xtion Pro RGB-D Camera   3 Gripper
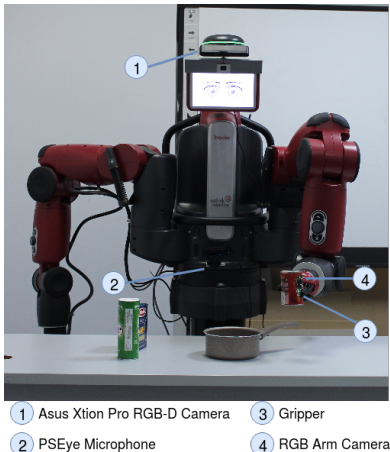2 PSEye Microphone   4 RGB Arm Camera

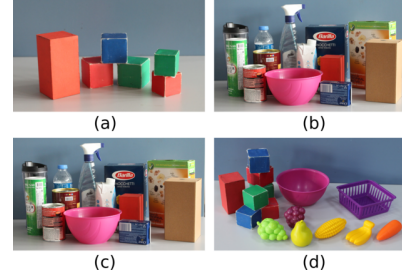Fig. 2: Experimental environment

Fig. 3: Object sets for the (a) Pick&Place and Put-on-top, (b) Push, (c) Place-in-container, (d) Pour manipulations.

During the data collection, the robot is tasked to execute a manipulation scenario, and all synchronized sensor readings (i.e., RGB/RGB-D image streams, audio waves) are then simultaneously recorded.

Among the primitive manipulation types introduced in [36], we mainly work on the following 5 types:

push, *pick&place*, *pour*, *place-in-container*, *put-on-top*. The distribution of the successful and failed trials for each manipulation is presented in Table I.

Sampled frames for each successful and failed case are visualized in Fig. 4. See the supplementary video showing the robot execution of sample manipulations.

Regarding data labeling, we have followed a goal oriented approach. The following cases are labelled as failure since the manipulation goal may not be met: dropping the manipulated object, collision with another object, collapsing a structure, localizing incorrectly, spilling the poured content onto the table. Minor cases, such as slight changes in the intended target location and/or orientation, are still considered as success.

C. Data Preprocessing

The robot plans and executes trajectories online. The length of the recordings vastly vary due to differences in executions of the manipulation and trajectory types. During manipulation, the observability of the scene varies depending on the the robot's arm position. To automatically eliminate such occluded frames, a depth-based thresholding method is applied. After filtering, remaining frames are roughly segmented into three phases corresponding to the *approach*, *manipulate*, and *retreat* primitives, respectively. Next, we randomly sample four frames from each of *approach* and *retreat* primitives. Finally, the 224×224 pixels area corresponding to the table plane is cropped. Fig. 4 shows randomly sampled 8 frames for each manipulation type.

TABLE I: Distribution of the successful and failed executions in our multimodal robot manipulation dataset FAILURE.

Manipulation	#Successes	#Failures	Total
Push	12	19	31
Pick&place	13	32	45
Pour	25	42	67
Place-in-container	23	33	56
Put-on-top	9	21	30
	82	147	229

Fig. 4: Sample successful and failed executions from the proposed FAILURE dataset. Higher resolution animated images can be found at https://github.com/ardai/fino-net.

D. FINO-Net

To address the learning problem described in Section III-A, we propose FINO-Net as a multimodal classifier network to detect manipulation failures using onboard sensory data in real time. The inputs of the network are composed of RGB and depth frames captured from the head camera and audio waves recorded over the course of a robot manipulation. FINO-Net adopts early fusion to combine RGB and depth frames while applies late fusion to combine vision and audio data. The architecture processes visual and audio inputs individually with a series of convolutional and convolutional-LSTM (convLSTM) layers. Finally, in the fusion step, the latent space representations are concatenated into a feature vector and fed to the fully connected layers. The overall FINO-Net architecture is depicted in Fig. 5. In the following subsections, we elaborate more on the network architecture, loss function, and training details.

1) Vision: In order to process spatio-temporal features in RGB and depth frames, we employ convLSTM cells. A typical LSTM cell is implemented using the fully connected layers, while a convLSTM replaces these with convolution operators.

The visual branch consists of three main blocks (see the top branch in Fig. 5). Inspired from [37], each block is composed of two convolutional layers and a convLSTM layer. RGB and depth frames are early fused by stacking on top of each other before feeding to the first visual block. Inside each block, the filter numbers remain the same for all convolutional and convLSTM layers. Each convolutional layer has 3x3 filters. Before convLSTM layers, max pooling is applied to cut in half the number of features. Each block

also has batch normalization and dropout layers.

2) Audition: In our earlier works [11], [12] we present that audition can be employed to monitor manipulation execution as it provides complementary information to other sensing modalities. We also showed that Mel Frequency Cepstral Coefficients (MFCCs) are the suitable representations for auditory monitoring where Support Vector Machines were employed to classify audio features.

In this work, we adopt a convolutional network composed of two convolutional layers followed by a max pooling layer (see the bottom branch in Fig. 5). There are 64 filters in each layer with a filter size of 32. As input, we use single channel audio recordings with 16 KHz sampling rate. The raw audio signal is divided into 32 millisecond windows. For each window, Short Time Fourier Transform is applied to convert signal into frequency domain. Mel filterbank is applied and 20 MFCCs are obtained using the librosa [38] library. The number of the audio windows are fixed by either applying padding or clipping.

3) Fusion: We adopt a late fusion approach to combine visual and auditory modalities. We introduce the following model:

$$y = \Phi(\phi_1(x_1^{(1)}, ..., x_{t_1}^{(1)}) \oplus ... \oplus \phi_m(x_1^{(m)}, ..., x_{t_m}^{(m)})) \ , \quad (2)$$

where ϕ_m is a unimodal convolutional neural network which acts as feature extractor, $\oplus$ is the concatenation operator, and Φ is the late fusion based classifier network. In the fusion step, vision and audition features, obtained from final output of each modality, are concatenated into a single feature vector. The fusion layer is composed of two fully connected layers as shown in Fig. 5. There can be made connections between the proposed hierarchical fusion approach and primate visual cortex [39].

4) Optimizer And Regularization: As an optimizer, we use Adam with a learning rate of $1e-6$. Furthermore, batch normalization is applied after each convolutional layer. To boost the roles of very basic features (e.g., edges and curves), a central dropout approach is adopted with the probability rate of 0.4. After each convolutional, convLSTM, and fully connected layers, a dropout layer is inserted except the first convolutional layer in Block 1 and the last convLSTM layer in Block 3.

To prevent overfitting, we augment the data by applying color augmentation and random flipping. For instance, the brightness, contrast, saturation, and hue values of all images in a sequence are randomly changed with a probability of 0.2. In a similar fashion, each image sequence is flipped vertically with a probability of 0.5.

IV. EXPERIMENTS

We split our dataset into train (70%) and test (30%) sets by preserving the class distribution. Experimental results are presented in terms of class weighted precision, recall and F1-scores. Reported results are the highest scores obtained after applying early stopping.
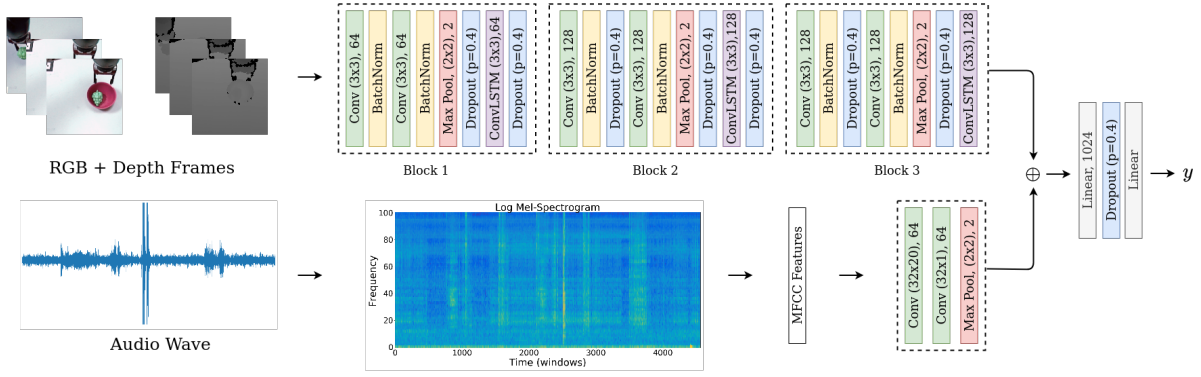
Fig. 5: FINO-Net Architecture

A. Baselines

We compare FINO-Net with a VGG-based network [40], which has 16 convolutional layers extended with an LSTM layer. The VGG baseline network is pretrained on ImageNet to be further used as a feature extractor. Obtained features from the final convolutional layer are fed into a single layer LSTM with 1024 neurons, followed by a fully connected layer. During training, only LSTM and fully connected layer weights are updated. For a fair comparison, this baseline model is trained with the same parameters (e.g. learning rate, batch size, etc.) and the same strategy (e.g., loss function, data augmentation, etc.) used for the training of FINO-Net. Note that the baseline model is also extended with the same fusion structure that FINO-Net has. Following baseline networks are trained with the given modality data:

- VGG-RGB: The input of the network is only the RGB frames obtained from the head camera.
- VGG-D: The input is only the single channel depth (D) frames captured from the head camera.
- VGG-RGB-D: RGB and depth frames are stacked on top of each other to obtain a 4 channel RGB-D input.
- VGG-RGB-D-A: Similar to FINO-Net, stacked RGB-D frames and audio (A) features are first individually processed, and then are concatenated to be fed to the fully connected layer.

B. FINO-Net

To analyse the unique contribution of each sensing modality on the failure detection task, we trained several variations of FINO-Net. To boost the performance, all convolutional and convLSTM layers of FINO-Net are initialized with the VGG weights pretrained on ImageNet. Next, we perform various training operations with the following modality data from our FAILURE dataset:

- FINO-Net-RGB: Network is trained with only the RGB input.
- FINO-Net-D: Network is trained with the depth (D) frames only.
- FINO-Net-A: Only the audio (A) branch is trained. After the convolutional layers, there is a single fully-connected layer with 64 neurons.
- FINO-Net-RGB-D: The visual branch of FINO-Net is trained by stacking the RGB and depth frames as input

to the network.
- FINO-Net-RGB-D-A: The entire network in Fig. 5 is trained with all the given modalities. The visual branch weights are initialized with FINO-Net-RGB-D and updated during training.

In addition to those experiments, we also freeze the FINO-Net weights initialized with that of the VGG baseline model. Next, our dataset is employed to train only the last convLSTM and the subsequent layers. Finally, the following experiments are performed:

- FINO-Net-F-RGB: While training the visual branch with RGB input, the layers are frozen (F) except the final convLSTM and fully connected layers.
- FINO-Net-F-RGB-D: The vision branch is duplicated for the RGB and depth frames. The RGB branch weights are frozen, except for the final convLSTM layer. All layers dedicated to the depth frames are updated. Next, the RGB and depth features are concatenated and fed to the fully connected layers.
- FINO-Net-F-RGB-D-A: In addition to the previous FINO-Net-F-RGB-D, the audio features are concatenated with visual features before passing to the fully connected layers.

Note that in the experiments FINO-Net-F-RGB-D and FINO-Net-F-RGB-D-A, we treat the RGB and depth channels separately by duplicating the visual branch. This is because VGG features are learned from the RGB ImageNet images, and thus, are not compatible with single-channel depth features. By updating the layers in the depth branch, we let the network explore unique depth cues which boost the network performance. This decoupling between the RGB and depth streams can also be interpreted as late fusion which is different than the early fusion strategy employed in FINO-Net-RGB-D and FINO-Net-RGB-D-A.

Note also that since the VGG network is particularly implemented for the structured image data, it cannot be trained with the audio modality. We, therefore, omit the model VGG-A and instead use FINO-Net-A for a fair comparison. The same applies to FINO-Net-F-A. Recalling the fact that VGG features learned from ImageNet images are not compatible with the depth features, we skip FINO-Net-F-D and instead use FINO-Net-D.

TABLE II: Quantitative Evaluation

| | Failure Detection | | |
	Precision	Recall	F1
FINO-Net-RGB	84.59	84.72	84.49
FINO-Net-D	85.34	84.72	84.07
FINO-Net-A	88.07	87.50	87.63
FINO-Net-RGB-D	89.19	88.88	88.97
FINO-Net-RGB-D-A	98.64	98.61	**98.60**
FINO-Net-F-RGB	86.13	86.11	85.82
FINO-Net-F-RGB-D	87.01	86.11	86.29
FINO-Net-F-RGB-D-A	95.90	95.83	95.84
VGG-RGB	90.39	90.27	90.31
VGG-D	89.19	88.88	88.97
VGG-RGB-D	90.39	93.27	90.31
VGG-RGB-D-A	94.44	94.44	94.44

C. Quantitative Results

The obtained quantitative results for the test split are reported in Table II. These results clearly show that our proposed FINO-Net has incremental performance improvement when a new sensor modality is introduced. Thanks to the proposed combination of the early and late fusion approaches, the FINO-Net's accuracy on the RGB data significantly increases with the depth and audio features. The same applies to the FINO-Net-F model where the model weights are mostly frozen. Note that the contribution of the audio data (i.e., FINO-Net-RGB-D-A) is relatively much more than that of the depth cues (FINO-Net-RGB-D). It is because when a failure (such as an object fall down) occurs, there are observed abrupt changes in the sound recordings. Such changes are not explicitly obtained in the captured depth images which are randomly sampled as described in section III-C.

When it comes to the baseline model VGG, we do not have the same observations: the depth modality (VGG-RGB-D) exhibits no contribution to the head camera data, i.e. VGG-RGB. The contribution of the audio modality is still less compared to the one observed in both FINO-Net models (FINO-Net-RGB-D-A and FINO-Net-F-RGB-D-A). These findings clearly show that our proposed FINO-Net has enough capacity to capture the unique contribution of each modality.

In terms of accuracy, our proposed FINO-Net-RGB-D-A (as well as FINO-Net-F-RGB-D-A) considerably outperforms the baseline (VGG-RGB-D-A) by leading to the highest F1-score (98.60%) in the failure detection scenarios where the task is to distinguish the successful and failed manipulation executions in a binary-classification manner.

Table III summarizes the ablation study for the FINO-Net model design choices we have made. Comparing the results, the central dropout applied together with batch normalization layers leads to 11% performance boost. Furthermore, increasing the number of convolutional layers for the audio branch slightly increases the network performance.

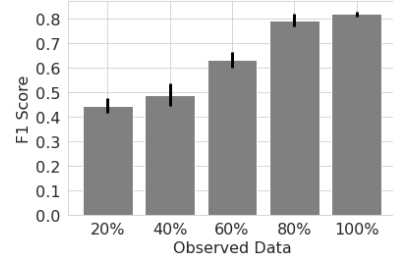Fig. 6 presents FINO-Net inference results in the case

Fig. 6: Fino-Net-RGB-D-A inference results on incomplete observations. For each recording in the test set, 10 different inferences are made with frames randomly sampled from the entire course of a manipulation.

of having incomplete observations. Results indicate that observations right after the robot arm retreats from the scene, i.e., after 80%, are more informative. Therefore, as explained in section III-C, we prefer to sample frames from the *retreat* phase, and to capture differences in the scene during the execution we also sample frames from the early *approach* phase. In addition, Table IV presents the FINO-Net execution time for one single forward pass obtained on Nvidia 2080 Ti GPU. The obtained results are average values of multiple executions. The results in Table IV and Fig. 6 show that even though the model runs in real-time, inferences are more accurate at the end of the execution.

V. DISCUSSION AND CONCLUSION

In this work, we presented a novel deep network model, named FINO-Net, that employs multi-model sensor readings to capture spatio-temporal features of robot manipulation executions to detect possible failures. In addition, we introduced a new multimodal manipulation dataset FAILURE with failed robot execution attempts. We conducted various experiments and compared the performance of FINO-Net with a VGG-based baseline model.

Findings in Table II show that FINO-Net has enough capacity to capture unique contribution of each sensor modality, which is not the case for the baseline model. This plays a crucial role when the robot plans a recovery action to prevent such failures in time. For instance, the robot can autonomously decide what modality type(s) to rely on, in order to react to a failure with the most optimal action.

We are aware of the fact that our proposed dataset has limited number of samples. Therefore, we also plan to extend

TABLE III: Ablation Study for the FINO-Net Design

Approach	Precision	Recall	F1-Score
FINO-Net-RGB-D-A	98.64	98.61	98.60
-Without central dropout	94.44	94.44	94.44
-Without batchnorm	91,93	91.66	91.72
-Without batchnorm and dropout	88.07	87.50	87.63
-Single layer audio branch	95.83	95.83	95.81

TABLE IV: Inference Durations on Nvidia 2080 Ti GPU

Approach	Run time (msec)	Approach	Run time (msec)
FINO-Net-RGB	9.4	FINO-Net-A	1
FINO-Net-D	9.3	FINO-Net-RGB-D-A	9.8
FINO-Net-RGB-D	9.3	VGG-RGB-D-A	3.4

our dataset with more semantically different manipulations listed in [36]. Increasing the number of sensors by including introspective sensors (i.e., force-torque and tactile) and robot hand cameras is another planned task in our agenda.

REFERENCES

[1] C. Fritz, "Execution monitoring – a survey," University of Toronto, Tech. Rep., 2005.

[2] O. Pettersson, "Execution monitoring in robotics: A survey," *Robotics and Autonomous Systems*, vol. 53, no. 2, pp. 73–88, 2005.

[3] P. Goel, G. Dedeoglu, S. I. Roumeliotis, and G. S. Sukhatme, "Fault detection and identification in a mobile robot using multiple model estimation and neural network," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, vol. 3, 2000, pp. 2302–2309.

[4] G. K. Fourlas, S. Karkanis, G. C. Karras, and K. J. Kyriakopoulos, "Model based actuator fault diagnosis for a mobile robot," in *IEEE Int. Conf. on Industrial Technology (ICIT)*, 2014, pp. 79–84.

[5] D. Stavrou, D. G. Eliades, C. G. Panayiotou, and M. M. Polycarpou, "Fault detection for service mobile robots using model-based method," *Autonomous Robots*, pp. 1–12, 2015.

[6] M. Ersen, E. Oztop, and S. Sariel, "Cognition-enabled robot manipulation in human environments: Requirements, recent work, and open problems," *IEEE Robotics Automation Magazine*, vol. 24, no. 3, pp. 108–122, 2017.

[7] M. Kapotoglu, C. Koc, S. Sariel, and G. Ince, "Action monitoring in cognitive robots (in turkish)," in *Signal Processing and Communications Applications Conference (SIU)*, 2014, pp. 2154–2157.

[8] D. Park, Y. Hoshi, and C. C. Kemp, "A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder," *Robotics and Automation Letters*, vol. 3, no. 3, pp. 1544–1551, 2018.

[9] D. Park, H. Kim, and C. C. Kemp, "Multimodal anomaly detection for assistive robots," *Autonomous Robots*, vol. 43, no. 3, pp. 611–629, 2019.

[10] I. Saltali, S. Sariel, and G. Ince, "Scene analysis through auditory event monitoring," in *Proceedings of the International Workshop on Social Learning and Multimodal Interaction for Designing Artificial Agents*. ACM, 2016, p. 5.

[11] A. Inceoglu, G. Ince, Y. Yaslan, and S. Sariel, "Comparative assessment of sensing modalities on manipulation failure detection," in *IEEE ICRA Workshop on Perception, Inference and Learning for Joint Semantic, Geometric and Physical Understanding*, 2018.

[12] A. Inceoglu, G. Ince, Y. Yaslan, and S. Sariel, "Failure detection using proprioceptive, auditory and visual modalities," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 2491–2496.

[13] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," *arXiv preprint arXiv:1901.03407*, 2019.

[14] J. Gertler, *Fault detection and diagnosis in engineering systems*. CRC press, 1998.

[15] J. P. Mendoza, M. Veloso, and R. Simmons, "Focused optimization for online detection of anomalous regions," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014, pp. 3358–3363.

[16] J. P. Mendoza, M. Veloso, and Simmons, "Plan execution monitoring through detection of unmet expectations about action outcomes," in *ICRA*, 2015, pp. 3247–3252.

[17] J. P. Mendoza, R. Simmons, and M. Veloso, "Detection and correction of subtle context-dependent robot model inaccuracies using parametric regions," *The International Journal of Robotics Research*, 2019.

[18] R. Micalizio, "Action failure recovery via model-based diagnosis and conformant planning," *Computational Intelligence*, vol. 29, no. 2, pp. 233–280, 2013.

[19] A. Bouguerra, L. Karlsson, and A. Saffiotti, "Handling uncertainty in semantic-knowledge based execution monitoring," in *IROS*, 2007, pp. 437–443.

[20] P. Doherty, J. Kvarnstrom, and F. Heintz, "A temporal logic-based planning and execution monitoring framework for unmanned aircraft systems," *Autonomous Agents and Multi-Agent Systems*, vol. 19, no. 3, pp. 332–377, 2009.

[21] S. Adam, M. Larsen, K. Jensen, and U. P. Schultz, "Towards rule-based dynamic safety monitoring for mobile robots," in *Simulation, Modeling, and Programming for Autonomous Robots*. Springer, 2014, pp. 207–218.

[22] O. Pettersson, L. Karlsson, and A. Saffiotti, "Model-free execution monitoring in behavior-based robotics," *Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 37, no. 4, pp. 890–901, 2007.

[23] L. Mauro, F. Puja, S. Grazioso, V. Ntouskos, M. Sanzari, E. Alati, and F. Pirri, "Visual search and recognition for robot task execution and monitoring," *arXiv preprint arXiv:1902.02870*, 2019.

[24] V. Sathish, M. Orkisz, M. Norrlof, and S. Butail, "Data-driven gearbox failure detection in industrial robots," *IEEE Transactions on Industrial Informatics*, 2019.

[25] X. Zhou, H. Wu, J. Rojas, Z. Xu, and S. Li, *Nonparametric Bayesian Method for Robot Anomaly Monitoring*. Springer Singapore, 2020, pp. 51–93.

[26] H. Wu, Y. Guan, and J. Rojas, "A latent state-based multimodal execution monitor with anomaly detection and classification for robot introspection," *Applied Sciences*, vol. 9, no. 6, p. 1072, 2019.

[27] D. Ramachandram and G. W. Taylor, "Deep multimodal learning: A survey on recent advances and trends," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 96–108, 2017.

[28] A. Eitel, J. T. Springenberg, L. Spinello, M. Riedmiller, and W. Burgard, "Multimodal deep learning for robust rgb-d object recognition," in *IROS*. IEEE, 2015, pp. 681–687.

[29] A. Pieropan, G. Salvi, K. Pauwels, and H. Kjellstrom, "Audio-visual classification and detection of human manipulation actions," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3045–3052.

[30] D. Feng, C. Haase-Schutz, L. Rosenbaum, H. Hertlein, C. Glaeser, F. Timm, W. Wiesbeck, and K. Dietmayer, "Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges," *IEEE Transactions on Intelligent Transportation Systems*, 2020.

[31] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay *et al.*, "Roboturk: A crowdsourcing platform for robotic skill learning through imitation," in *Conference on Robot Learning*, 2018, pp. 879–893.

[32] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 1048–1055.

[33] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," in *CoRL 2019: Volume 100 Proceedings of Machine Learning Research*, 2019.

[34] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 421–436, 2018.

[35] C. Finn, I. Goodfellow, and S. Levine, "Unsupervised learning for physical interaction through video prediction," in *Advances in Neural Iinformation Processing Systems (NeurIPS)*, 2016, pp. 64–72.

[36] F. Worgotter, E. E. Aksoy, N. Kruger, J. Piater, A. Ude, and M. Tamosiunaite, "A simple ontology of manipulation actions based on hand-object relations," *IEEE Transactions on Autonomous Mental Development*, vol. 5, no. 2, pp. 117–134, 2013.

[37] J. Rothfuss, F. Ferreira, E. E. Aksoy, Y. Zhou, and T. Asfour, "Deep episodic memory: Encoding, recalling, and predicting episodic experiences for robot action execution," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4007–4014, 2018.

[38] B. McFee, C. Raffel, D. Liang, D. P. Ellis, M. McVicar, E. Battenberg, and O. Nieto, "librosa: Audio and music signal analysis in python," in *Python in Science Conference*, vol. 8, 2015, pp. 18–25.

[39] N. Kruger, P. Janssen, S. Kalkan, M. Lappe, A. Leonardis, J. Piater, A. J. Rodriguez-Sanchez, and L. Wiskott, "Deep hierarchies in the primate visual cortex: What can we learn for computer vision?" *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 8, pp. 1847–1871, 2012.

[40] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *Preprint arXiv:1409.1556*, 2014.