PREPRINT: Accepted at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018

Learning to Fly by MySelf: A Self-Supervised CNN-based Approach for Autonomous Navigation

Alexandros Kouris[1] and Christos-Savvas Bouganis[1]

***Abstract*— Nowadays, Unmanned Aerial Vehicles (UAVs) are becoming increasingly popular facilitated by their extensive availability. Autonomous navigation methods can act as an enabler for the safe deployment of drones on a wide range of real-world civilian applications. In this work, we introduce a self-supervised CNN-based approach for indoor robot navigation. Our method addresses the problem of real-time obstacle avoidance, by employing a regression CNN that predicts the agent's distance-to-collision in view of the raw visual input of its on-board monocular camera. The proposed CNN is trained on our custom indoor-flight dataset which is collected and annotated with real-distance labels, in a self-supervised manner using external sensors mounted on an UAV. By simultaneously processing the current and previous input frame, the proposed CNN extracts spatio-temporal features that encapsulate both static appearance and motion information to estimate the robot's distance to its closest obstacle towards multiple directions. These predictions are used to modulate the yaw and linear velocity of the UAV, in order to navigate autonomously and avoid collisions. Experimental evaluation demonstrates that the proposed approach learns a navigation policy that achieves high accuracy on real-world indoor flights, outperforming previously proposed methods from the literature.**

I. Introduction

In the past decade, significant progress has been made in the field of aerial robotics, driven by the rapid development of inexpensive off-the-shelf drones. UAVs are employed in a wide range of applications, spanning from search and rescue operations [1] to precision agriculture [2] and infrastructure inspection [3]. At the same time, with Deep Learning (DL) demonstrating state-of-the-art performance in various Machine Vision tasks, recent research has focused on enhancing the autonomy of UAVs by employing DL algorithms [4], such as Convolutional Neural Networks (CNNs) [5].

Obstacle detection and avoidance methods form an important step towards safe autonomous UAV navigation, especially when deployed in real-world environments. Towards this goal, several methods have been proposed in the literature using sophisticated sensors such as RGB-D [6] and LIDAR [7] and complex algorithms such as SLAM [8] to construct a 3D model of the environment (map) and extract depth information to deduce the navigable space. However, such methods present high computational cost and usually require heavyweight expensive sensors that restrict their applicability on lightweight commercial drones. Moreover, such systems may fall to unrecoverable errors when operating in real-world dynamic environments or confronting textureless surfaces which are often present in indoor scenes.

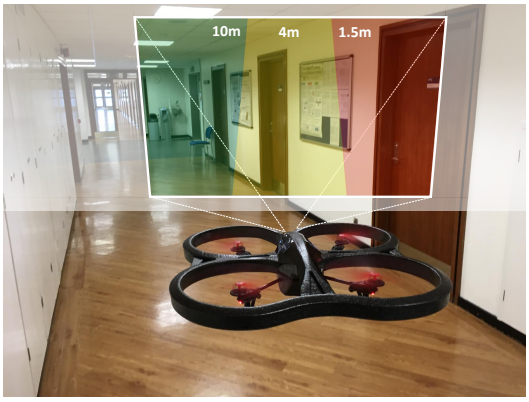

Fig. 1. The proposed regression CNN, trained on our custom dataset collected and annotated in a self-supervised manner, predicts the distance-to-collision towards three directions within the camera's field of view, based solely on visual input, to navigate autonomously in indoor environments. **Supplementary Video:** https://youtu.be/43dmXKprIVQ

Recently, vision-based navigation has gained a lot of interest, owing to its applicability on commercially available quadcopters, which are commonly equipped with a forward-looking camera, in the absence of external power-hungry or heavyweight sensors. Simultaneously, machine learning's tremendous advancement has enhanced the capabilities of visual navigation, as Deep Neural Networks (DNNs) enabled the development of end-to-end learning approaches [9] in which the feature extraction is performed using a large set of learnable parameters in place of handcrafted feature selection. Since the latter approach suffers from low generalisation capabilities, DNNs act as an enabler for visual navigation in real-world environments that inherently demonstrate significant variation in visual appearance.

In this direction, various approaches have been introduced targeting to correlate raw sensor inputs with control commands by employing learning by demonstration [10] and reinforcement learning (RL) [11] algorithms. However, the former require human expert demonstrations during the data collection and annotation process, which acts as a limiting factor to the volume of training data, undermining the generalisation capabilities of the resulting models. As regards the latter setting, the trial-and-error nature of RL policy search raises safety concerns and suffers from catastrophic crashes when operating in real-world environments. Learning control policies in simulation has also been studied recently [12], however the gap between real and virtual environments limits their applicability in the physical world [13].

The support of the EPSRC Centre for Doctoral Training in High Performance Embedded and Distributed Systems (HiPEDS, Grant Reference EP/L016796/1) is gratefully acknowledged.

[1]The authors are with the Department of Electrical and Electronic Engineering, Imperial College London.
Email: {a.kouris16,christos-savvas.bouganis}@imperial.ac.uk

In this paper we address the problem of autonomous UAV navigation in indoor environments using solely monocular input, by employing a Deep Learning approach. The key contributions of this work are summarised by the following:

- The deployment of a two-stream CNN for robot perception, trained on a custom dataset to fit a regression model that predicts the real distance-to-collision in view of the robot's forward-looking camera input. The employed network architecture, consisting of two merging streams that concurrently process the current and previous raw visual input, demonstrates enhanced learning capacity by extracting spatio-temporal features that incorporate both static appearance and robot motion information in the learning process.
- A novel local motion planning policy that modulates the robot's yaw and linear velocity based on the CNN's distance predictions towards three diverging directions within the field of view (FoV) of the robot's camera (Fig. 1), to accomplish collision-free navigation.
- A real-flight custom dataset of indoor UAV trajectories, used to train the proposed CNN model, annotated with continuous distance-to-collision values towards the front, left and right direction with respect to the on-board camera's FoV. The dataset is collected and annotated in a self-supervised manner by a drone equipped with external distance sensors mounted on its hull.

Extensive experiments on a commercial drone demonstrates that the proposed system for autonomous indoor navigation, achieves considerably higher performance in various navigation scenarios compared to related approaches introduced in recent literature. Distance-to-collision predictions performed by the proposed CNN allow informed motion decisions to be made by the introduced local motion planning policy, accomplishing safe robot navigation on real-world environments. Moreover, it is shown that the employed two-stream CNN architecture, incorporating spatio-temporal features in its learning process, significantly boosts the accuracy on the undertaken regression task compared to other CNN architectures, commonly used in relevant literature.

II. Related Work

Autonomous navigation is highly correlated with obstacle detection and avoidance tasks. Monocular obstacle detection methods in literature are based either on scene analysis using classical computer vision algorithms or on machine learning.

Most of the latter recently introduced approaches, employ CNNs due to their state-of-the-art performance in various Machine Vision tasks. These works can be further divided into those mainly targeting the robot perception task, that extract meaningful information about the relative state of the robot to its environment from sensory inputs, and end-to-end learning approaches which learn a direct mapping of raw sensor measurements to control commands. End-to-end learning approaches usually rely on imitation learning, where a human expert controls the agent in a real-world environment to provide *<input image, pilot's choice-of-action>* pairs that are used for training a model to mimic the pilot's behaviour [14] [15]. Alternatively, automated generation of ground truth labels for each input has been proposed [16], using for example predicted depth and surface normal data with a 3D cost function to couple each input image into a desired trajectory from a pool of predefined paths.

Lately, a state-of-the-art transfer learning approach has been introduced by Loquercio et al. in which a CNN trained on data collected by cars and bicycles is successfully applied on a drone to fly autonomously in urban environments [9]. The key principles of this concept are closely related to the work of [17], in which a dataset collected by human hikers with head mounted cameras (IDSIA) was used to train a DNN for drone navigation on forest/mountain trails, formulated as a classification task across three high-level planning decisions (go straight, turn left, turn right). Building on this work, TrailNet [18] forms a recently introduced model, trained on an enhanced version of IDSIA dataset, to predict the drone's orientation and lateral offset with respect to the centre of the trail, allowing it to control both its z-axis rotation (yaw) and y-axis translation in order to retain its path on the trail. Interestingly, in the work of [9] autonomous UAV navigation in outdoor urban environments is reduced to a regression problem, directly predicting the desired yaw-velocity of the drone considering each input frame. The regression CNN model was trained on an autonomous driving dataset with commanded steering angle labels. Simultaneously, a collision probability is predicted by a classification branch of the CNN, sharing the same feature extractor with the regression stream, to control the UAV's linear velocity. The classification branch was trained on data collected by cyclists, labelled as positive or negative based on the bicycle's distance to its surrounding vehicles/objects.

On the other side of the spectrum, various approaches have focused on the perception task, employing custom motion planning schemes to determine the robot's action based on the perception output. In [19], for example, a depth map is predicted for each monocular image captured by the drone's on-board camera, using a CNN trained on RGB-D data. Then, a deterministic arbitration scheme is employed to steer the UAV away from obstacles by controlling its angle on two rotational degrees of freedom (DoF), based on the generated depth map.

In the state-of-the-art work for indoor navigation by Gandhi et al. [20], which is most closely related to our approach, a large dataset of drone trajectories has been collected, with all of its paths resulting to a collision that was identified by monitoring the accelerometer data. In this self-supervised data collection setting, a predefined number of samples close to the collision are labelled as negative (i.e. samples that were met moments before a crash), while the rest as positive (i.e. samples collected when flying towards navigable space). The resulting binary classification problem is addressed by training a CNN model to performs inference on three segments of each input image, with the prediction probabilities being fused by a deterministic algorithm to conclude on a single yaw-angle control command.

In this paper, to detect navigable space on the UAV's environment, we train a CNN to run inference on three overlapping windows of each input image independently, similar to the work of [20]. However, instead of classifying every input window to "collision-free" or "non-navigable", we treat robot perception as the more information-rich regression task of predicting the real distance-to-collision towards each direction, inspired by recent literature that introduces continuous-value CNN-based predictions on various robotic applications [21] [22]. Trained on a custom dataset of real-flight images, collected and annotated in a self-supervised way by an UAV with external active distance sensors attached, distance-to-collision can be predicted solely from monocular visual input. To fuse the CNN predictions for every input frame's windows, a custom local motion planning policy is implemented, making an informed decision about the robot's motion, controlled via its yaw and linear velocity.

The proposed method differs from the CNN classification-based approach of [20] in that the introduced regression CNN model predicts fine-grained distance-to-collision values for each input, trained on continuous distance-labelled data, considering the closest obstacle on the robot's path towards three diverging directions. Conversely, the binary classification model of [20] is trained on UAV images that are coarsely labelled into two discrete classes thresholded based on each frame's position in its trajectory (assuming that all trajectories result to a collision). This classification approach (also employed by [9] to predict collision probability) demonstrates a rapidly escalating probability value within a small portion of consecutive input frames of each trajectory, located around the predefined class separation threshold, while maintaining an overconfident prediction probability (0 or 1) during all other parts of navigation; as demonstrated in our experiments (Sec. IV-B). Instead, the proposed regression model generates a more insightful prediction of the actual distance to the closest obstacle in the drone's candidate trajectories, that smoothly scales with the UAV's motion. This information-richer approach enables the development of a more informed local motion planning policy, which not only avoids collisions during navigation, but also considers the longer range traversability of space in the robot's environment to produce motion commands. Moreover, by avoiding encapsulation of thresholding values in the dataset's labelling process, the proposed system demonstrates larger flexibility in the sense of tunability for different navigation scenarios and/or UAV platforms, without the need of retraining the model.

Our method also differs from the approaches of [9] and [14] that employ imitation learning to determine the UAV's steering angle (in the continuous and discrete domain respectively), in that the proposed local motion planning system determines robot's velocity and orientation, based solely on the CNN's distance prediction, without requiring a task-specific end-to-end annotated dataset. Moreover, to the best of our knowledge, this is the first work that employs a two-stream CNN architecture to fuse spatio-temporal features of sensory data for the task of autonomous navigation.

III. Methodology

A. Self-Supervised Dataset Collection and Annotation

Deep Learning methods require large amount of data in order to train models that would generalise well in differing real-world environments. To minimise human effort, self-supervised methods can be adopted to automate the collection and annotation process of large scale datasets [23] [24].

Employing a self-supervised methodology, we create the first, to the best of our knowledge, indoor flight dataset annotated with real distance labels to the closest obstacle towards three diverging directions in the field of view of the drone's forward-looking camera. The need for a relevant dataset, has also been identified in recent literature [9].

In the proposed approach, three pairs of Ultrasonic and Infra-Red (IR) distance sensors are mounted on an UAV pointing towards different directions within its camera's field of view, to allow automatic annotation of all data samples. Since the utilised drone is equipped with a wide lens (92°) camera, the selected sensor alignment is pointing towards [-30°, 0°, 30°] across the image width, w.r.t. the centre of the captured frame. The drone executes straight-line trajectories in multiple environments within real buildings, while camera images are recorded along with raw data from the external distance sensors. As illustrated in Fig. 2, each trajectory terminates when the forward-looking sensor detects an obstacle at the minimum distance that the drone requires to stop without colliding. A random rotation is performed to determine the flight direction of the next trajectory, which should be towards a navigable area of the environment. This constraint is evaluated by using the external distance sensors of the UAV, minimising the human supervision requirements of the data collection process, and hence making the dataset easily extensible.

Ultrasonic sensors support longer range distance sensing, at the cost of providing many noisy samples due to scattering and reflection effects when obstacles are positioned too close or on a narrow angle to the sensor. Conversely, Infra-Red sensors demonstrate better behaviour on a wide range of sensing angles, however their scope is limited to notably shorter distances. Having used pairs of both sensor technologies towards each direction, the recorded distance data from each pair are used to automatically annotate the corresponding input images of each trajectory by being aligned and fused, using the Ultrasonic data in samples with large-distance obstacles and the IR data when available due to presence of obstacles in close proximity. The fused annotations of each trajectory are also processed by a low-pass filter to eliminate sensor noise that would affect the training process.

After numerous iterations of that process in various real-world indoor environments (including seminar room, hallway, office space, kitchen,...), at the moment of writing a large data set of more than 300,000 samples, organised in 2,000 trajectories, has been collected. Each input image captured by the UAV's forward looking camera, is divided into three overlapping windows, each being annotated with the fused measurements of the corresponding sensor pair.

Fig. 2. Self-supervised collection of an indoor flight-trajectory dataset in real-world buildings, automatically annotated with distance-to-collision labels.

B. CNN Architecture and Training

The introduced distance-annotated dataset is used for training a CNN to fit a regression model on the visual perception task of predicting the distance to the closest obstacle.

From each input image captured by the forward-looking on-board camera of the UAV, three overlapping rectangular windows are extracted and separately processed by the CNN. A similar approach was introduced in the work of [20], however, the proposed model incorporates more information about the scene, being trained to predict the continuous real distance-to-collision towards each direction, instead of the probability distribution of a binary classification between collision-free and non-navigable space classes.

Although image sensors provide a sequence of visual data, related research has focused solely on learning spatial (i.e. static appearance) features by independently processing each input sample. In contrast, the proposed approach also leverages the temporal information encapsulated between consecutive frames, which enables the extraction of temporal features such as relative motion, local proximity etc. This is achieved by employing a two-stream CNN, driven by the performance that similar architectures have demonstrated on a wide span of applications with temporal or multimodal data [25] [26]. Initially, contiguous-in-time frames are processed concurrently in pairs by two separate streams that extract scene appearance information from both static images. The two streams are then fused to a single stream (deeper in the network) to extract temporal features incorporating information regarding the relative motion between the two frames. Hence, both spatial and temporal components of the input sequence are exposed in the trained model.

Selecting the appropriate method and depth for fusing the two streams of such network architectures to combine static appearance and motion information has been widely studied in literature [27]. In our case, experimental evaluation suggested that fusing the two streams in an intermediate depth of the feature extractor achieves the highest performance in the undertaken task of distance prediction. The resulting network fuses the extracted spatial feature maps of intermediate abstraction level from each pair of consecutive input frames by concatenation, to determine their temporal relation in the form of temporal features learned deeper in the network. More details on the findings of this evaluation are presented in Sec. IV-B.

The proposed two-stream CNN architecture is illustrated in Fig. 3a. Each network's stream consists of two layers with identical structure to the first two Convolutional (CONV) layers of AlexNet [28], which is a widely employed and computationally low-demanding CNN model. After fusing the two streams by concatenating their output channels, AlexNet's third CONV layer (with modified number of input channels) is inserted to process the fused feature-map volume. Successively, the two last CONV layers of AlexNet's feature extractor are adopted to form the single stream section of the architecture. The classifier originally used in AlexNet, consisting of three Fully-Connected (FC) layers, is replaced by a regression unit, consisting of a FC layer connected to a single-neuron FC layer that produces the network's output (i.e. the predicted distance-to-collision). Similar to AlexNet's input layer, each stream of the proposed CNN receives an input image of size 227x227x3, denoting height, width and number of channels respectively. As UAV cameras typically provide larger RGB images, we scale down all collected input samples without affecting their aspect ratio, to meet the CNN's input-height constraint. Subsequently, three overlapping windows satisfying the input-width requirement are extracted and associated with the corresponding distance labels of the dataset. Each window is treated as an independent training sample, linked with the matching image region of its previous in time frame, to form a 6-channel input for the two-stream network. Multiple data augmentation techniques are applied on the training samples, including random rotations, horizontal flipping (with corresponding switching of the leftmost and rightmost window labels) and lighting variations (focusing mainly on brightness, contrast and saturation).

The CONV layers of the proposed network were pre-trained on the imitation learning dataset for indoor navigation of [14], to enhance the generalisation capability of the CNN by exposing its feature extractor to environments with diverse appearance. The training of the regression CNN was performed using Stochastic Gradient Descent with Momentum, in 30 epochs, employing a mini-batch size of 128 with the training set being shuffled on each epoch to form

Fig. 3. (a) Two-stream CNN architecture for spatio-temporal feature extraction on regression tasks. (b) Instance of local motion planner.

diverse batches. Momentum was set to 0.9, with a starting learning rate of 0.001 being reduced by a factor of 10 after every 10 epochs. Moreover, regression response values were normalised (in the range of [0,1]; with their scaling back to the predicted distance values during inference being trivial) to prevent an exploding behaviour of loss values that would affect the convergence of the training method.

C. UAV Local Motion Planning for Obstacle Avoidance

During inference, the trained model is used to regress three distance-to-collision values $|\tilde{d}_l|, |\tilde{d}_c|, |\tilde{d}_r|$ towards the leftmost, central and rightmost directions of the forward-looking camera's field of view, corresponding to [-30 , 0 , 30] angle w.r.t. the centre of the input image respectively. To achieve that, three overlapping windows of the robot's visual input, along with the corresponding image regions of the previous frame, are extracted in similar way as in the training stage (described in Sec. III-B) and processed independently by the proposed network architecture.

The outputs of the proposed regression model are fed to a custom local motion planning algorithm to conclude into a single control command that modulates the robot's yaw (u^z_{rot}) and forward linear velocity (u^x_{lin}). These information-rich predictions of the regression CNN, provide a fine-grained and accurate distance estimation, gradually escalating across a wide range of real distance-to-collision values. The proposed motion planning policy is leveraging this distance information to perform continuous fine adjustments of the robot's motion. This allows commanding timely manoeuvres that result to smoother navigation and prompt interaction with the environment, as well as insightful longer-range planning decisions by selecting to move towards the direction that is considered to contain the largest amount of traversable space.

The linear velocity of the robot (u^x_{lin}) is defined to be inversely proportional to the predicted distance-to-collision for the central window of the visual input (Eq. 1), to avoid collisions in the direction of flight of the UAV.

$$u^x_{lin} = \begin{cases} \frac{u_{max}}{d_H - d_L}(|\tilde{d}_c| - d_H), & \text{if } |\tilde{d}_c| < d_H \\ u_{max}, & \text{otherwise} \end{cases} \quad (1)$$

where d_L, d_H form tunable, task- and drone-specific parameters of the control scheme representing the minimum distance that the UAV is allowed to get close to detected obstacles, and the minimum detection distance that the drone is capable to decelerate and avoid collision when flying on its maximum allowable velocity $u_{max} \in [0, 1]$ accordingly.

The UAV's rotational (yaw) velocity $u^z_{rot} \in [-1, 1]$ is adjusted to guide the UAV towards the direction that is predicted to have the largest amount of navigable space within the current frame's FoV. This is determined by the angle of the resultant distance vector of all consecutive examined directions that exceed a tunable threshold value d_{th}. This UAV control strategy is described by Alg. 1.

Algorithm 1 Calculate Rotational Yaw Velocity

```
Input: {d̃_l, d̃_c, d̃_r}, d_th
Output: u^z_rot
if {|d̃_l|, |d̃_c|, |d̃_r|} > d_th then
    z = ∠(d̃_l + d̃_c + d̃_r)
else if |d̃_l| − |d̃_r| then
    z = ∠(d̃_l + d̃_c)
else if |d̃_l| < |d̃_r| then
    z = ∠(d̃_c + d̃_r)
end if
u^z_rot = z / ∠(d̃_r)
```

where $\angle()$ indicates the angle w.r.t vector $\tilde{d}_c$

Under the control of this policy, the UAV instantly reacts to the visual input of its sensors, based on the trained model's prediction, as also illustrated in Fig. 3b. Low-pass filtering of commanded velocities is employed to eliminate undesirable oscillations in robot motion. Treating perception as a regression task that predicts distance-to-collision, rather than a classification task between navigable and non-navigable space, permits enhanced tunability of the navigation policy as a result of the most primitive nature of information that is fed to the motion planner. As discussed, the proposed policy can be tuned by a number of thresholding parameters whose values may differ significantly between different UAV platforms and/or navigation scenarios. For example, different drones demonstrate varying levels of manoeuvrability, leading to unalike reaction time and stopping distance. Moreover, tuning can be performed to meet task-specific requirements, such as keeping the drone on a large safety distance from obstacles in scenarios where failures are unrecoverable. Conversely, the classification-based setting is less condescending in such tuning as it deals with a higher-level task and its behaviour is mainly dictated ahead of the training of the model.

IV. Experimental Evaluation

In this section, the experimental evaluation of the proposed approach is discussed, both for the perception task undertaken by the CNN and for the introduced end-to-end navigation method, by performing comparisons with other commonly employed CNN architectures and relevant state-of-the-art works from the recent literature, showing both quantitative and qualitative results.

A. Experimental Setup

CNN design and training are performed using MATLAB R2017b and the Neural Network Toolbox[1] on a desktop server equipped with an Intel Xeon E5-2630 processor, 64GB of RAM and a GTX1080 GPU. Parrot AR-Drone 2.0[2] is used for the UAV-related experiments, equipped with a 720p forward-facing camera, with a wide-angle lens (92 degrees) that captures images at 30 fps. During the experiments WiFi communication is established between the drone and a laptop equipped with an Intel Core i7-6700HQ, 16GB of RAM and a GTX1070 GPU, executing the CNN inference and motion planning, to send velocity commands to the drone. Ardrone_autonomy package[3] and ROS Kinetic Kame were used. Moreover, during the data collection process, external distance sensors[4] were attached on the UAV's hull connected on an Arduino Pro Mini micro-controller.

B. Distance Prediction Results

Two-stream vs Single-stream CNN: Initially, we evaluate the performance of the trained two-stream CNN model on the regression task of distance prediction from raw visual input, comparing it with a commonly used single-stream AlexNet architecture. Additionally, an experimental study between various configurations of the proposed two-stream network architecture (Fig. 3), in terms of: (a) fusing depth (i.e. after which layer the two streams are fused) and (b) fusing method (i.e. if the streams are fused by performing channel-wise addition or concatenation), is presented. All CNN models are trained on the contributed dataset (Sec. III-A) following the previously described methodology (Sec. III-B) and evaluated in terms of Root-Mean-Squared-Error (RMSE) on a test dataset containing 20,000 real-flight pictures from various indoor environments of the dataset, which were excluded from the training set of Sec. III-B.

The results of this comparison are listed in Table I. As can be seen from the table, all the variations of the proposed two-stream CNN architecture significantly overperform the single-stream approach, achieving smaller prediction error on normalised distance values across the test set. Hence, it is deduced that employing CNN architectures that are capable of learning spatio-temporal features boosts the achieved accuracy on the distance prediction task by capturing relative motion information between consecutive input frames. A comparison between the error range in the predicted distance of the proposed two-stream network configuration and the single-stream architecture is illustrated in Fig. 4 (left y-axis). As demonstrated in this plot, the proposed two-stream CNN consistently demonstrates a narrower confidence interval compared to the single-stream approach across the test set, independently of the input frame's actual distance range.

TABLE I

CNN-Regression Error on Distance Prediction Task

Network Architecture			RMSE
Streams	Fusing Depth	Fusing Method	(Normalised [0,1])
Single-stream*	-	-	0.083642
Two-Stream	Early (Input)	Concatenation	0.024926
Two-Stream**	Middle (Conv2)	Concatenation	0.023307
Two-Stream	Late (Conv5)	Concatenation	0.025041
Two-Stream	Early (Input)	Addition	0.029317
Two-Stream	Middle (Conv2)	Addition	0.043447
Two-Stream	Late (Conv5)	Addition	0.043642

*Based on AlexNet architecture [28] with regression unit **This work

Moreover, the proposed CNN architecture (Sec. III-B), that fuses its two streams in intermediate network depth using channel concatenation, performs slightly better than all other concatenation-based settings. This configuration initially extracts spatial features of intermediate level of abstraction from both input frames, before fusing the two streams to extract temporal relations between them. Finally, it is observed that, in general, network configurations employing channel concatenation as their fusing method achieve higher accuracy on the regression task, compared to those using channels-wise addition. This can be explained by the greater flexibility provided by concatenation, as output channels of both streams are combined in the next CONV layer using learnable parameters (exposed on training), instead of being summed up as in the case of additive fusion.

Regression vs Classification CNN: In Fig. 4 we also depict the output of the binary classification task proposed in [20], predicting the probability of each input frame to correspond to collision-free or non-navigable space, along with the regressor output of the proposed architecture. As demonstrated in reference with the right y-axis of the plot, the probability output of the binary classifier is correctly predicting the existence of navigable space in samples with high-distance labels without false-positives. However, the CNN is mostly concluding to over-confident predictions, maintaining a probability of 0 or 1 for most of the distance values in the examined spectrum. A highly escalating behaviour with wide error range is observed within a short range of distance values (70-120cm), positioned around the labelling threshold of the two classes, which is determined ahead of training and cannot be adapted afterwards. Conversely, the proposed regression-based approach demonstrates a more information-rich, smoothly escalating behaviour across the whole range of the examined distances with significantly lower error range, bounded only by the maximum sensing range of the external distance sensors used during data collection (500cm). This complementary information is exploited by the proposed local motion planning policy, to make more informed longer-range decisions for autonomous navigation.

[1] https://www.mathworks.com/products/neural-network.html
[2] https://www.parrot.com/us/drones/parrot-ardrone-20-power-edition
[3] http://wiki.ros.org/ardrone_autonomy
[4] GP2Y0A60SZLF Analog IR Sensor and HC-SR04 Ultrasonic Sensor

TABLE II

COMPARISON WITH RELATED WORK

Method	Seminar Room		Hallway	
	Time(s)*	RMSE **	Time(s)*	RMSE **
Straight Path Policy	4.8	0.5643	9.2	0.42872
Human Pilot	96.2	-	107.9	-
Imitation Learning [14]	11.7	0.1359	14.1	0.1298
Classification-based [20]	27.4	0.1177	38.2	0.0996
This work	51.4	0.0679	68.0	0.0775

*Mean time between collisions/human interventions

**Compared to human pilot's choice of action

Fig. 4. Actual vs Predicted distance-to-collision from proposed and baseline regression CNNs and navigable-space probability from the classifier of [20].

C. Quantitative Results on UAV Navigation

In this section the proposed method is evaluated on an end-to-end manner in the task of autonomous UAV navigation. Four different indoor scenarios are examined, including traversing a Hallway and navigating within a seminar room with or without real-world obstacles (such as chairs, boxes, poster-stands, bins etc) in various physical configurations.

The system's performance is compared with a Straight-Path Policy acting as a weak baseline and a human pilot controlling the drone using a joystick, based solely on its forward-looking camera inputs (without having the drone on sight), acting as a strong baseline. We have also implemented two state-of-the-art methods for CNN-based autonomous indoor navigation from the literature:

- The CNN classification-based model of [20] trained on AlexNet architecture, which feeds its predictions to an arbitration scheme that controls the UAV's yaw in accordance with the predicted collision probabilities towards each direction
- The Imitation Learning based method of [14] which employs an AlexNet-based end-to-end classifier to map each input image to a flight-command from a discrete motion pool (move forward, move right, spin right,...)

All policies are compared in terms of average time between collisions after navigating in all four test-scenarios for 5 minutes each, with minimal human intervention (Table II). Moreover, the normalised commanded yaw velocities of all methods are compared in terms of RMSE, with the human pilot's choice of action (annotated offline). The results of this comparison (also listed in Table II) demonstrate that the proposed approach outperforms the other autonomous baselines, managing to navigate without coming in contact with obstacles present in the testing environments for 4.60 more time (mean) compared to the imitation learning approach of [14] and 1.78 compared to the CNN classification-based work of [20], reaching up to 0.53 and 0.63 the flight time of a human pilot, and overpassing by 10.7 and 7.4 the weak baseline of Straight-Path flight, on the seminar room and hallway scenario respectively.

Representative input instances from the UAV camera, along with the commanded yaw by all examined methods, are illustrated in Fig. 5. The proposed regression-based approach combined with the developed motion planning policy, is shown to take finer-grained actions to continuously adjust the robot's orientation, resulting to smoother path.

D. Qualitative Results on UAV Navigation

Discussion and Limitations: From the experiments described in the previous section, we deduce that the proposed two-stream regression architecture generally manages to make more insightful decisions in cases of high ambiguity (such as when dealing with reduced number of trackable features due to close proximity to obstacles) by utilising the learned, richer in information, spatio-temporal representation of the visual input. Moreover, it is observed that the fine granularity of predicted distance values fed to the motion planner enables continuous slight adjustments of the robot's track, resulting to a considerable reduction in oscillations on the yaw axis. In the extreme case that a texture-less surface covers the camera's field of view, resulting to a complete loss of features, the accuracy of the predicted distance-to-collision drops considerably. However, in such cases, a value close to the minimum observed distance on the dataset is inferred, allowing the motion planner to avoid collisions. Additionally, in the case of flying in extremely close proximity to glass surfaces, the accuracy of the distance prediction demonstrates a significant drop. Such cases that were evident in the dataset being annotated with the real distance captured by the external proximity sensors downgraded the performance of the trained model, by increasing its error range and were removed from the training set. However, both cases are rarely encountered during flight in indoor cluttered environments, due to the wide field of view of the cameras used in UAVs.

Motion Planning Case Study: Finally, we provide a representative case study that demonstrates the contribution of employing a regression model to obtain fine-grained distance-to-collision predictions across a trajectory and exploiting them by a custom motion planning policy to make informed action decisions for autonomous navigation. Our method is compared with the coarser classification-based approach of [20]. Fig. 6 illustrates the flight trajectory of a drone on an indoor environment, navigating autonomously by employing both approaches. In the examined case, the informed local motion planning policy, leverages the richer distance information provided by the regression model, which enables insightful, longer-range planning decisions to be made based on the ratio of the predicted distances towards the examined directions. Exploiting this information, the proposed approach demonstrates higher level of environmental awareness, following a track towards the navigable corridor, in contrast to the classification-based approach which successfully avoided collisions but made shorter-range motion planning decisions based on the less informative probability of collision and eventually got trapped.

Fig. 5. Samples cases from comparison between: imitation learning [14], classification-based CNN [20], the proposed regression CNN and Human Pilot.

Fig. 6. Case study comparison of regression- and classification-based approaches on a representative autonomous navigation scenario.

V. Conclusion

In this work the problem of autonomous UAV navigation is addressed by the deployment of a two-stream CNN that leverages spatio-temporal information from visual input sequence to predict the distance-to-collision between the robot and its environment towards multiple directions. Trained to fit a regression model, the proposed CNN along with a novel local motion planning policy that translates these distance predictions to velocity commands, demonstrates considerable performance improvement on real-world indoor navigation scenarios, compared to state-of-the art approaches.

References

[1] M. B. Bejiga, A. Zeggada, A. Nouffidj, and F. Melgani, "A convolutional neural network approach for assisting avalanche search and rescue operations with uav imagery," *Remote Sensing*, vol. 9, no. 2, 2017.

[2] S. W. Chen, S. S. Shivakumar, S. Dcunha, J. Das, E. Okon, C. Qu, C. J. Taylor, and V. Kumar, "Counting apples and oranges with deep learning: a data-driven approach," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, April 2017.

[3] T. Ikeda, S. Yasui, M. Fujihara, K. Ohara, S. Ashizawa, A. Ichikawa, A. Okino, T. Oomichi, and T. Fukuda, "Wall contact by octo-rotor uav with one dof manipulator for bridge inspection," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 5122–5127.

[4] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy, "A review of deep learning methods and applications for unmanned aerial vehicles," *Journal of Sensors*, vol. 2017, 2017.

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-Based Learning Applied to Document Recognition," in *Proc. IEEE*, 1998.

[6] K. Schmid, T. Tomic, F. Ruess, H. Hirschmller, and M. Suppa, "Stereo vision based indoor/outdoor navigation for flying robots," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Nov 2013, pp. 3955–3962.

[7] S. Shen, Y. Mulgaonkar, N. Michael, and V. Kumar, "Multi-sensor fusion for robust autonomous flight in indoor and outdoor environments with a rotorcraft mav," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, May 2014, pp. 4974–4981.

[8] K. Çelik and A. K. Somani, "Monocular vision slam for indoor aerial vehicles," *Journal of electrical and computer engineering*, 2013.

[9] A. Loquercio, A. I. Maqueda, C. R. del Blanco, and D. Scaramuzza, "Dronet: Learning to fly by driving," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, April 2018.

[10] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 1527–1533.

[11] N. Imanberdiyev, C. Fu, E. Kayacan, and I. Chen, "Autonomous navigation of uav by using real-time model-based reinforcement learning," in *2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Nov 2016.

[12] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2016.

[13] 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," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 3357–3364.

[14] D. K. Kim and T. Chen, "Deep neural network for real-time autonomous indoor navigation," *arXiv preprint arXiv:1511.04668*, 2015.

[15] S. Ross, N. Barkhudarov, K. S. Shankar, A. Wendel, D. Dey, J. A. Bagnell, and M. Hebert, "Learning monocular reactive uav control in cluttered natural environments," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, May 2013, pp. 1765–1772.

[16] S. Yang, S. Konam, C. Ma, S. Rosenthal, M. Veloso, and S. Scherer, "Obstacle Avoidance through Deep Networks based Intermediate Perception," *arXiv preprint arXiv:1704.08759*, 2017.

[17] A. Giusti, J. Guzzi, D. C. Cireşan, F.-L. He, J. P. Rodríguez, F. Fontana, M. Faessler, C. Forster, J. Schmidhuber, G. Di Caro *et al.*, "A machine learning approach to visual perception of forest trails for mobile robots," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, 2016.

[18] N. Smolyanskiy, A. Kamenev, J. Smith, and S. Birchfield, "Toward low-flying autonomous mav trail navigation using deep neural networks for environmental awareness," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017.

[19] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, and L. V. Eycken, "Cnn-based single image obstacle avoidance on a quadrotor," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 6369–6374.

[20] D. Gandhi, L. Pinto, and A. Gupta, "Learning to fly by crashing," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 3948–3955.

[21] T. Naseer and W. Burgard, "Deep regression for monocular camera-based 6-dof global localization in outdoor environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 1525–1530.

[22] E. Johns, S. Leutenegger, and A. J. Davison, "Deep learning a grasp function for grasping under gripper pose uncertainty," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 4461–4468.

[23] S. Zhou and K. Iagnemma, "Self-supervised learning method for unstructured road detection using fuzzy support vector machines," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2010, pp. 1183–1189.

[24] S. Pillai and J. J. Leonard, "Towards visual ego-motion learning in robots," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 5533–5540.

[25] R. Zhao, H. Ali, and P. van der Smagt, "Two-stream rnn/cnn for action recognition in 3d videos," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 4260–4267.

[26] N. Patel, A. Choromanska, P. Krishnamurthy, and F. Khorrami, "Sensor modality fusion with cnns for ugv autonomous driving in indoor environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2017, pp. 1531–1536.

[27] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2014, pp. 1725–1732.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances in Neural Information Processing Systems 25*. Curran Associates, Inc., 2012, pp. 1097–1105.