Abstract—Over the last few years, UAV applications have grown immensely from delivery services to military use. Major goal of UAV applications is to be able to operate and implement various tasks without any human aid. To best of our knowledge, in the existing works for autonomous navigation for UAV’s, ideal environments (e.g., 2D) are considered instead of realistic or special hardware are used (e.g., nine range sensors) to navigate through an ideal environment. Therefore, in this thesis, we aim to overcome the limitations of the existing works by proposing a model for navigating a drone in an unknown environment without any prior knowledge about the terrain using default drone sensors only. We present a model which is compatible with almost every off-the-shelf drone available in the market. Our methodology utilizes only standard drone sensors which are attached to almost every drone. These include a camera, GPS, IMU, magnetometer, and barometer. Our methodology uses 3D, POMDP, and continuous environment. It also takes into account environmental factors such as winds and rain. Our main contributions are: 1) We are using realistic environment model including factors like rain and wind. 2) We are only using onboard computing resources to run our model instead of some external server. 3) We were able to fly the achieve complete autonomous flight using only standard drone sensors.

Index Terms—Unmanned aerial vehicle, global positioning system, inertial measurement unit, partially observable Markovian decision process.

I. INTRODUCTION

With the rapid advancements in the technology, the fields of robotics and mechatronics have drastically upgraded in terms of their role in the modern business and different industries. UAV also known as a drone is an aircraft that flies without any human pilot and are controlled remotely. For a quadcopter to perform in an indoor environment, it requires to operate with agility and efficient feedback without losing control. Stabilization is needed for an outdoor environment for a drone to operate against external forces. In short, for both situations, a stable flight is immensely important for a drone to function effectively. This can help a quadcopter to survive in extreme external factors e.g., heavy rain or wind as it will be able to prevent crashing. In order to use these applications for navigation and stabilization, UAVs and drones need to be supplied with a series of sensors e.g., depth cameras, accelerometer, gyroscope, magnetometer, etc.

Apart from this, applications of drones can be observed in many fields such as accident reporting, infrastructure inspection, crop monitoring, etc. Fig. 1 is from a survey which presents the growth rate of different UAV applications, it can be seen that aerial surveying and mapping is growing at a higher rate (Murfin) [1]. Quadcopters are a more specific form of an aerial vehicle known as multicopter having an arbitrary number of rotors. Quadcopters are driven by four rotors. In the 1920s, when quadcopters were first introduced, they were unable to gain popularity due to challenges like having comparatively large size and weight, mechanical complexity, control management, and etc. [1], [2].

Data in large volumes are often spawned by drones and UAVs. These UAV applications will only be useful for the consumer if this data is managed efficiently and effectively without requiring extra efforts. Therefore, AI (Artificial Intelligence) seems to cover these challenges as nowadays every other company or industry is using AI techniques, machine learning or deep learning to process immense amount of data. There are many AI based tasks that deal with image recognition, UAVs are also required to perceive and understand the environment, detect and avoid collisions in order to achieve a smooth flight and motion planning is also another task which requires AI and machine learning to play an active part as shown in Fig. 2.

Autonomous navigation in unknown terrain is an exponentially hard task due to unavailability of pre-constructed maps or path planning. In this thesis, a deep RL based framework is developed for UAVs navigation in unknown large-scale complex environments. Specifically, first, an efficient policy based DQN is designed, which comprises of convolutional and fully connected layers to extract important features from images taken through depth vision camera. These features are then
combined with inputs taken directly from other sensors, e.g., IMU or magnetometer, etc. and are passed to a fully connected layer. This is followed by a policy-based Q-learning approach to apprehend UAV navigation.

![Fig. 2. Role of AI in UAVs.](image)

II. RELATED WORK

Relatively much work has been going on with respect to automatic UAV navigation with machine learning algorithms but less or no work has been done on unknown path travel by UAV. Su Yeon Choi and Dowan Cha in their paper “Unmanned aerial vehicles using machine learning for autonomous flight; state-of-the-art” has divided their research areas into three categories parameter tuning, adaptive control and real-time path planning [3].

Smolyanskiy, et al. "Toward low-flying autonomous MAV trail navigation using DNN for environmental awareness’’ proposes a micro air vehicle system to follow trails in an unstructured outdoor environment. Drawbacks are NVIDIA custom hardware chips (TX1+J120) are used and not fully autonomous (Human intervention required) [4]. For non-colliding paths for multi-agent UAV’s cooperative planners have been developed. It has inaccuracies. Geramifard et al. studies on the ICCA for combining cooperative planners and RL techniques as a framework [5]-[9]. Zhang et al. [10] propose a CGLA, which is made for path planning on multiple UAV cooperation [11]. In CGLA, to balance between the economy of paths and collision avoidance, the parameters are trained. The individual weight matrix and cost matrix are proposed for an efficient path planning of both single and multiple UAVs [3]. Based on the geometric distance and risk information shared among UAVs, the individual weight matrix is calculated and adaptively updated. The optimal path from a starting point to a target point is calculated. The simulation results validate the effectiveness and feasibility of CGLA for safe navigation of multiple UAVs. However, it is required to utilize the coarse and fine grid to design a fast path planning algorithm in the proposed framework and apply the algorithm to an actual UAV [3].

There are results to avoid a collision for UAV using laser range finders or Kinect cameras. Vandapel et al. [12] avoid a collision in 3D space using lidars, and Bachrach et al [13], and Bry et al. [14] depict obstacle avoidance in an unknown room using scanning lidars for SLAM [12-[15]. Bachrach et al. use Microsoft Kinect camera in an unknown room [15]. However, these research works are required to improve on problems of power supply, payload, UAVs cost, and the reliability of localization. As a result, Achtelik et al. [16], Wendel et al. [17], Fraundorfer et al. [18] created maps by a single camera, or stereo cameras [15], [16], [18]. Achtelik et al. [19] created sparse maps of the environment using a single, cheap camera [15]. Wendel et al. [9] created dense maps of the environment using a camera [8]. Fraundorfer et al. [18] demonstrated accurate depth estimation and localization using stereo cameras. However, even though the algorithms are reasonably fast to avoid a collision, they were still too computationally expensive for UAVs flight. Recently, many researchers have studied path planning and formation light techniques [20]-[23].

The environmental model defines the space in which drone navigates. In [22], the 2D environmental model is utilized. In [21], a simplified model in a 3D environment is used where the impact of drag forces on the drone is ignored. In [22], the 3D model is utilized for modeling. In our work, we have considered a complete 1D environment with consideration to all the environmental factors, which include winds, rain as well as drag effect created by these elements.

Observational Space defines how the environment behaves when some action is performed. In the deterministic environment, if given a state, action pair next state can be guaranteed while in a stochastic environment, it’s not guaranteed that state, action pair will generate the same output every time. In [21], [22], observational space is set to deterministic while [4], [24], [25] have used stochastic model. We have also utilized the stochastic observational model for our environment as its closer depiction of real-world scenarios.

Observability refers to how much environmental state is available to UAV at any given point in time. In [4], [22], completely observable environment is utilized. In [4], [21], [25], assumed the partially observable environment is assumed where some part of the state is available at any given time instead of whole (which is in case of fully observable). We have taken environment observability to be partial.

In [14], Model based RL along with random shooter model is deployed. In [22], [25], model free RL is used while in [4], [21], standard DNNs are exploited. In our work, we are utilizing model free RL.

In [24], a separate module for image processing is used, but detailed explanations are not specified in the paper. In [21], [22], [25], no camera module is utilized at all. Therefore, there is no image processing module. In [4], a separate module for video processing is utilized. For this purpose, theYOLO module is utilized. In our work, we are not utilizing any separate module for video processing. The data is instead fed into the NN, which utilizes it for generating action probabilities.

Ideally, a UAV should not rely on any other external help for its decision making. In [24], however, an external ROS server is used as a helping node. In [22], the extra onboard PID module is used. In [4], [21], [25], any helper modules are not required. In our work, we are not utilizing any external helper modules except the NN.
Drone type is essential as it plays a huge part in deciding the adaptability of research in real-world scenarios. Off the shelf, drones are the desired scenario as it increases the chances of adaptability to very high. In [21], [22], [24], off the shelf drones are used. In [4], [25], customized drones are used for the experiment. In our work, we are using off the shelf drone.

In [24], an extra ROS server is used, which is deployed on a high-end machine. In [21], [22], no off-board resources are deployed in their work. In [4], TX-1 and AuVedia chips are placed on a drone for processing. In [25], an extra nine range sensors are deployed for the experiment. In our work, we have not utilized any off-board computing resource and have used only off the shelf drone.

When a model is pushed into drone memory which is already trained on some existing dataset, then that model is called pre-trained model and process is called pre-training. In [22], [24], [25], they have not used any pre-training. In [21], 512 initial and 1024 branching trajectories are utilized for pre-training the model. In [4] IDSIA dataset is being utilized for pre-training. In our work, we have not utilized any pre-training.

In [4], state action pairs from the previous 15 iterations are used as an input for the NN. In [22], the 3D environmental state is deployed as its primary data source. In [21], orientation, position, and velocity are exploited as input. In [4], cameras are used as input. In [25], gyroscope, GPS, and range sensors are deployed as input. In our work, we have utilized the camera, IMU, GPS, barometer, and magnetometer as input.

III. PROPOSED METHODOLOGY

The baseline idea is to autonomously navigate a drone in an unknown environment, which means in whatever environment a drone is, it must be able to navigate effectively and efficiently. This is achieved by using policy based DQN, a deep learning technique. There are five major modules in our system named as Input, Output, System, Environment, and Reward. Input deals with the sensor generated values. The reward is calculated as an indicator of an action’s feedback. The more the action is performed accurately, the higher is the reward. Next, this reward is forwarded to the system, where our policy-based DQN model (described in next section) processes all the inputs and generates output in the form of the probability distribution of actions. The most favorable action is performed by the quadcopter; action feedback is recorded, passed to our DNN, and the cycle continues. The environment here represents the terrain where our quadcopter is flying or navigating. Environment generates many parameters like sensor’s readings and some states on which the reward function is based.

We propose a model for navigating a drone in an unknown environment without any human help or aid. The goal is to navigate from location A to location B in unknown terrain without having any prior knowledge about the terrain using default drone sensors only. This work aims to address the limitations of existing research works by taking into account the following parameters as primary contributions of this work are following. A continuous domain is being considered rather than a discrete one. The simulation environment is 3D. 3D navigation is enabled, i.e., the quadcopter can navigate across all three-axis x, y, and z. High model accuracy (High success rate, low stray, and failure rate). The model is completely autonomous. Performance evaluation is done on multiple environment datasets.
with which you aim to interact, in our case, its quadcopter as these APIs are also accessible as an independent cross-platform library. Code written in the simulator can later be executed on the real UAVs. Table III presents the system specifications. We used python 3.6 as a programming language, Tesorflow version 1.13, Cuda version 10.0, cudann version 7.4.2.

<table>
<thead>
<tr>
<th>Components</th>
<th>Name</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation tool/Platform</td>
<td>AirSim</td>
<td>Open-source simulator for UAVs, cars and more</td>
</tr>
<tr>
<td>AirSim APIs</td>
<td>simGetImage, reset, confirm connection, simGetObjectPose, simGetCollisionInfo</td>
<td>To retrieve images, get state, control the vehicle and for many other tasks</td>
</tr>
<tr>
<td>Vendor</td>
<td>Microsoft</td>
<td></td>
</tr>
<tr>
<td>Sensors</td>
<td>Camera, GPS, IMU, magnetometer, barometer</td>
<td>AirSim currently supports these sensors data</td>
</tr>
</tbody>
</table>

**TABLE III: SYSTEM SPECIFICATIONS**

<table>
<thead>
<tr>
<th>Components</th>
<th>Versions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operation System</td>
<td>Linux (x64)</td>
</tr>
<tr>
<td>Python</td>
<td>3.6</td>
</tr>
<tr>
<td>Tensorflow</td>
<td>1.13</td>
</tr>
<tr>
<td>Cuda</td>
<td>10.0</td>
</tr>
<tr>
<td>CudaNN</td>
<td>7.4.2</td>
</tr>
</tbody>
</table>

### V. RESULTS

The drone is initially spawned in a random location on the map, and the final destination is set to a predefined location (usually midpoint in the map). The goal of the drone is to fly from its random initial position to the final destination without any collisions.

The speed and altitude of the UAV range between 0 to 10m/s and 5 to 200m respectively. The episode length (i.e., time step) can be up to 200. The network parameters are learned by using Adam Optimizer with a learning rate of 10-3. The discount factor γ for the experiments is varied in the range of 0.91 to 0.99.

The graph shown in Fig. 2 shows a trend of reward function values when we run the program for 500 epochs. After a few small tweaks and an increasing number of iterations to 1000, we can observe that the reward is maxing out. The reward was capped at 200. As the iterations increase, the reward values show a rapid increase, respectively.

In any navigation, the most important part is a realistic and complete environmental model. It’s a baseline for interaction as a strong environmental model would result in strong results. In [21], [22], complete environments are used. In [22], a 2D environment is used, which in itself reduces the complexity model. It also removes a lot of complication from the problem statement. In [21], the 3D model is used, but a very simplified model is assumed. All the drag forces that act on the drone and make navigation a difficult task are ignored. In [24], [4], [25], completely realistic environment is utilized. In our work, we are also utilizing a completely realistic environment taking into account all the factors that make the environment more stochastic.

Model free RL is not the golden standard for navigation tasks. Its performance is best among all the available methods. In [23], [24], model based RL is utilized. In [4], simple DNNs are deployed. In [5], [21], [22] Deep model free RL is used.

It’s very important that what kind of hardware is utilized for the navigation as it has a direct impact on its adaptability if someone is getting very good results but is utilizing very specific hardware, then its less useful as compared to one which has relatively less impressive results but is using commodity hardware. [21], [22], [24] are using commodity hardware, so their work is extensible. In [4], very specific Nvidia TX1, AuVedia 120, and three cameras are utilized. In [25], an extra nine range sensors were attached to their UAV. In our work, we are using off the shelf drone with all the standard sensors. This makes our work more relevant for implementation by industry.

This refers to the ability to learn from its mistakes as its flies instead of training model explicitly. In [21], [22], [24], [25], deep RL is utilized; therefore, the ability to have a model that learns on the fly exists. In [4], DNN is used; therefore, they lack this ability. We are also utilizing learning on the fly ability as we are also using model free RL.

Onboard processing is an important benchmark in drone navigation. Nowadays, it’s necessary that the drone should
have all the processing onboard to guarantee complete autonomy.

<table>
<thead>
<tr>
<th>Related Work</th>
<th>Evaluation Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Low level control of quadrotor with dynamic multi-rotor model [4]</td>
<td>✓ X ✓ X ✓ X</td>
</tr>
<tr>
<td>Autonomous UAV navigation using RL [12]</td>
<td>✓ ✓ ✓ ✓ X X</td>
</tr>
<tr>
<td>Control of quadrotor with DL [10]</td>
<td>✓ ✓ ✓ X ✓ X</td>
</tr>
<tr>
<td>Search, low-flying autonomous MAV path navigation [13]</td>
<td>✓ X X X ✓ X</td>
</tr>
<tr>
<td>Autonomous navigation in C200 in Large scale environment [16]</td>
<td>✓ X X ✓ X</td>
</tr>
<tr>
<td>Our Approach</td>
<td>✓ ✓ ✓ X X X</td>
</tr>
</tbody>
</table>

This parameter defines whether the authors were able to fly drone autonomously for more than 5 mins. In [4], they were able to achieve 6 seconds flight from 3 minutes of data. If they tweaked with the training data, their results deteriorated. In [22], results were presented for the 2D environment. Therefore, the work was only good enough for the testbed. For an outdoor, real-world 3D environment where drag forces effect, the model couldn’t perform well. In [21], drone stabilization was achieved but not the navigation. Their work shows that the drone can hover for more than 5 minutes but cannot navigate through the environment. Another issue in this work [21] was a simplified model assumption which makes even hovering unstable in a real-world environment.

In [24], an extra ROS server is deployed. In [21], [22], [25], only using onboard processing power is exploited. In [4], extra chips beside the traditional hardware are utilized. TX-1 and Auvedia-120 chips are being used. In our work, we are also not utilizing any off-board processing.

All of these factors are summarized in Table IV. We have concluded from the above considerations that most of the works only cover a few of the above considerations. This makes our work stand out from rest as no one has covered all these factors in their research (till date) along with our state-of-the-art results to the best of our knowledge. The work that was closest to our implementation is presented in [5], but it also misses a few key aspects (e.g., customized hardware, extra sensors, etc.). We were also able to achieve state-of-the-art success rate in our work. In our research work, we have taken into account all these factors and have completed autonomous navigation.

VI. CONCLUSION

We have summarized the critical factors that are important in drone navigation scenario. We have concluded from the above considerations that most of the works only cover a few of the above considerations. This makes our work stand out from rest as no one has covered all these factors in their research (till date) along with our state-of-the-art results to the best of our knowledge. The work that was closest to our implementation is presented in [5], but it also misses a few key aspects (e.g., customized hardware, extra sensors, etc.). We were also able to achieve state-of-the-art success rate in our work. In our research work, we have taken into account all these factors and have completed autonomous navigation.

ACKNOWLEDGMENT

This research was supported by the 2018 scientific promotion program funded by Jeju National University.

REFERENCES


Mudassar Liaq has done B.S from National University of Computer and Emerging Sciences, (NUCES), Islamabad in Computer Science, Pakistan in 2014. For next 3 years, he worked in industry before starting M.S from Jeju National University in September 2017. Currently he is MS student in Jeju National University (JNU). He is also working as a research assistant in ML lab in JNU.

Yung Cheol Byun received his B.S. from Jeju National University, Korea in 1993. M.S and Ph.D. degrees from Yonsei University in 1995 and 2001. He worked as a special lecturer in SAMSUNG electronics in 2000 and 2001. From 2001 to 2003, he was a senior researcher of Electronics and Telecommunications Research Institute and he promoted to join Jeju National University as an assistant professor in 2003. He is currently a professor of the Department of Computer Engineering, Jeju National University. From 2012 to 2014, he had research activities at University of Florida as a visiting professor. His research interests include pattern recognition & image processing, artificial intelligence & machine learning, security based on pattern recognition, home network and ubiquitous computing, u-healthcare and RFID & IoT middleware system.