DeepFly: Towards Complete Autonomous Navigation of MAVs with Monocular Camera

Utsav Shah
Robotics Research Center
KCIS, IIIT Hyderabad

Rishabh Khawad
Robotics Research Center
KCIS, IIIT Hyderabad

K Madhava Krishna
Robotics Research Center
KCIS, IIIT Hyderabad

ABSTRACT

Recently, the interest in Micro Aerial Vehicles (MAVs) and their autonomous flights has increased tremendously and significant advances have been made. The monocular camera has turned out to be most popular sensing modality for MAVs as it is light-weight, does not consume more power, and encodes rich information about the environment around. In this paper, we present DeepFly, our framework for autonomous navigation of a quadcopter equipped with monocular camera. The navigable space detection and waypoint selection are fundamental components of autonomous navigation system. They have broader meaning than just detecting and avoiding immediate obstacles. Finding the navigable space emphasizes equally on avoiding obstacles and detecting ideal regions to move next to. The ideal region can be defined by two properties: 1) All the points in the region have approximately same high depth value and 2) The area covered by the points of the region in the disparity map is considerably large. The waypoints selected from these navigable spaces assure collision-free path which is safer than path obtained from other waypoint selection methods which do not consider neighboring information.

In our approach, we obtain a dense disparity map by performing a translation maneuver. This disparity map is input to a deep neural network which predicts bounding boxes for multiple navigable regions. Our deep convolutional neural network with shortcut connections regresses variable number of outputs without any complex architectural add on. Our autonomous navigation approach has been successfully tested in both indoors and outdoors environment and in range of lighting conditions.

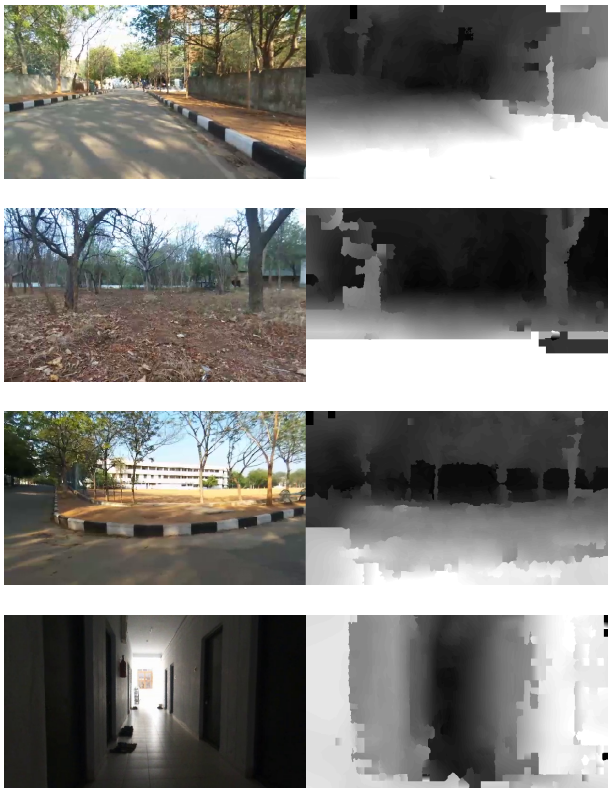

Figure 1: Navigable Space Perception. Left: Detecting Navigable Spaces with Color Images is inherently hard problem as what constitutes such a space depend greatly on environment. Right: In Disparity Space, this detection becomes more tractable.

CCS Concepts

•**Computing methodologies** → **Vision for robotics;** *Computer vision; Machine learning;*

Keywords

Micro Aerial Vehicles; Autonomous Navigation; Deep Learning

ICVGIP, December 18-22, 2016, Guwahati, India

DOI: http://dx.doi.org/10.1145/3009977.3010047

1. INTRODUCTION

Micro Aerial Vehicles(MAVs) that can autonomously navigate through cluttered environments have long been an objective of robotics research [14], [22], [13], [2]. MAVs can be useful in a number of important applications, such as disaster scene surveillance, package delivery, and aerial imaging for entertainment, remote farming, and construction work

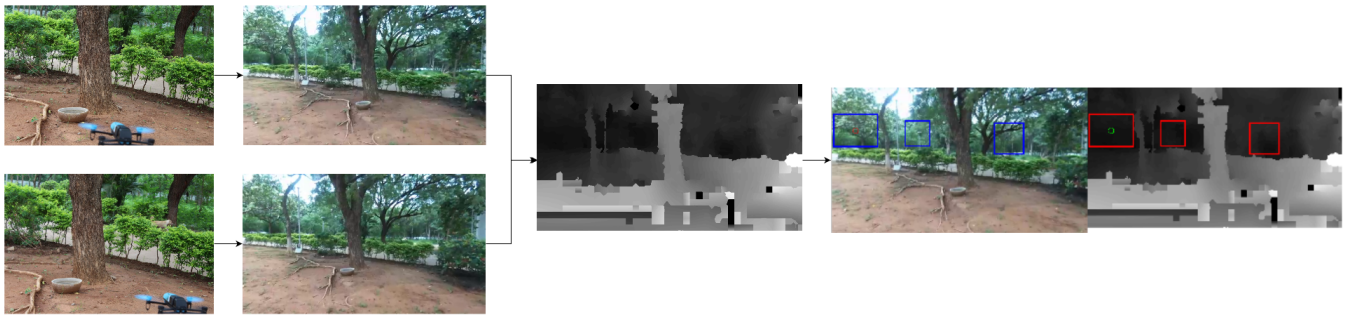

Figure 2: DeepFly Framework. Column 1: Horizontal Translation of Quadcopter. Column 2: Images obtained at endpoints. Column 3: Disparity map computed using two images. Column 4: Output of Network as bounding boxes containing navigable spaces. Center of largest box is next waypoint.

monitoring. In order to fulfil the promise of these applications, MAVs should be able to navigate in unstructured environments with complete autonomy. For this, it is essential that they are able to avoid obstacles and navigate robustly.

Most of the earlier approaches for autonomous operations of MAVs use laser range sensors, Kinect, stereo camera, or combination of multiple sensors. However, these sensors are heavy and very power consuming. They hurt agility of MAVs and make flight-time shorter. Also in most of the publicly available MAVs, mounting such sensors is not feasible. Therefore, we are interested in autonomous navigation method that uses only a monocular camera, which is readily available in most of the MAVs. Much of the existing approaches for autonomous navigation are tailored for either indoors or outdoors conditions. Moreover, the lighting conditions have greater impact on the results of the system. We focus on the architecture that works both indoors and outdoors and also in challenging lighting conditions.

While most of the existing efforts for autonomous navigation of MAVs focus on obstacle detection, we rather focus on detecting good navigable spaces. The navigable spaces can not be segmented efficiently using only color intensities in variety of scenes. As seen in the left column of Figure 1, bounding navigable spaces across diverse colour images can prove to be difficult due to the wide chasm in the semantics of what constitutes such a space. However, the context of the navigable space is more tractable with disparity images (right column of Figure 1). In disparity space, the representation of navigable spaces is more uniform in wide range of scenes. Hence, the task of finding good navigable regions can be posed as finding regions in the disparity map which occupies large area and have approximately same low disparity values.

In this paper we present DeepFly, our framework for autonomous navigation of a quadcopter equipped with monocular camera. Figure 2 shows the architecture of our system, composed of two modules: disparity map generation and navigable space detection. We obtain a pair of images by horizontal translation of quadcopter and pass it to stereo correspondence algorithm. The generated dense disparity map is then fed to our deep convolutional neural network with shortcut connections which performs multivariate regression. The regression target is bounding boxes for one, two, or three navigable spaces depending on the structure in front of the quadcopter. We select next waypoint from these navigable spaces as center of biggest navigable space.

We specify two contributions of our approach: 1) We obtain dense disparity map of scene using quadcopter equipped with monocular camera which works in range of different environments and 2) We present a deep network with a novel ground-truthing scheme which can regress "variable" number of outputs without any complex architectural add on.

It is important to note that we do not claim that the disparity map obtained in our approach is ideal. Instead, we emphasize that we do not need perfect depth structure for autonomous navigation purposes. The learning process can handle small random errors in disparity map. Moreover, our framework is modular: our disparity map generation module can be used with other waypoint selection methods and our navigable space detection and waypoint selection module can be used on disparity map / depth map obtained through some other pipeline.

2. RELATED WORK

Lately, there has been very interesting and impressive research on Autonomous Navigation and Obstacle Avoidance of MAVs. The bouquet of methods include using Range Sensors, Stereo Camera, Monocular Camera, Structure from Motion, and other feature based techniques. The existing approaches also range from purely reactive systems to complete planning based systems. Many of these approaches to autonomous navigation of MAVs are specific to outdoors [4] or indoors [11] conditions. Also many of them depend significantly on lighting conditions [21]. On the other hand, our approach works in both indoors and outdoors environments. It is very robust and works well in range of lighting conditions.

Different Range sensors used with MAVs can include laser range finder, infrared sensors, and such. Nieuwenhuisen et al [17] presented obstacle detection and complete planning method with 3D laser scanner, two stereo camera pairs, and ultrasonic distance sensors. Bry et al [3] presented a state estimation method using an on-board laser range finder and inertial measurement unit and showed aggressive flight in GPS-denied environments. However, the range sensors are not practical to most of the publicly available quadcopters as they are often too heavy for MAVs and consume lots of power. Our work is based on only a monocular camera, which consumes low power and is also light weight. Moreover, a monocular camera is readily built in to most of the quadcopters.

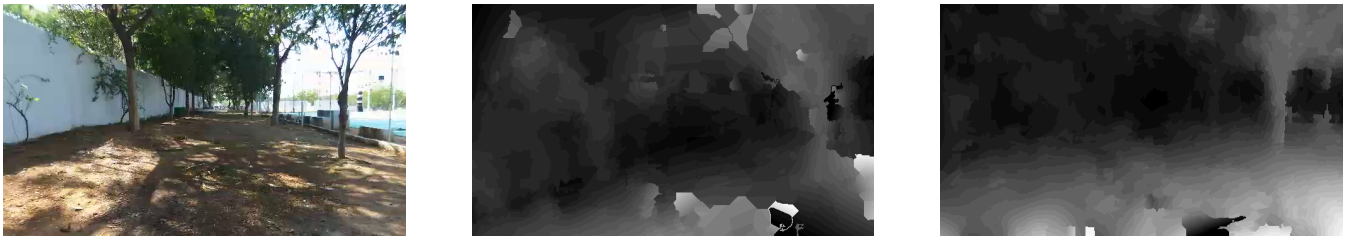

Figure 3: Left: Image of the scene in front. Center: Disparity Map using ZED Stereo Camera. Right: Disparity Map by Horizontal Translation of Quadcopter

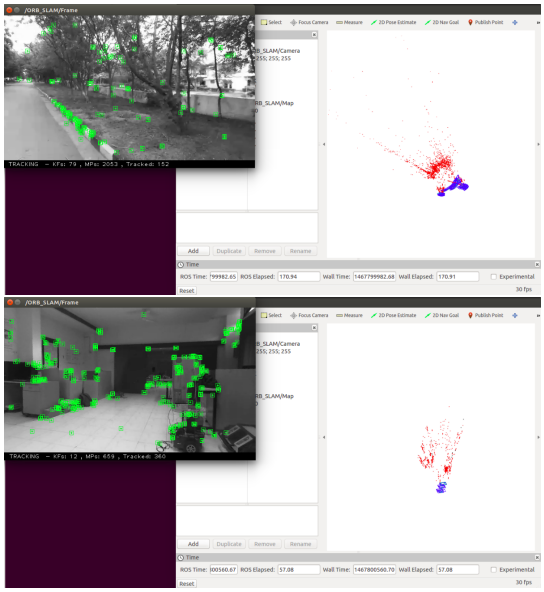

Figure 4: ORB SLAM Output in typical Outdoors (Top) and Indoors (Bottom) Environment

Structure from Motion (SfM) or Monocular Visual SLAM approaches [12], [15] can be used to reconstruct 3D scene geometry from a moving monocular camera. While such SfM approaches are reasonably fast, they produce sparse maps that are not suited for collision-free navigation; the absence of visual features in some region does not necessarily imply free space (Figure 4). Many of the approaches that try to obtain dense depth map use some variant of DTAM [16], [1]. However, DTAM is susceptible to breakage due to insufficient feature tracks if quadcopter pitches or rolls significantly. It also requires high computing power. Moreover, DTAM based approaches needs large number of images (ranging from 10 to 100) of varying views. In contradiction, our method requires only two images captured by quadcopter by performing a horizontal translation. Using this image pair, we obtain a dense disparity map using stereo correspondence algorithm (SPS Stereo [25] in this work). The obtained disparity map is not as accurate as depth map produced by DTAM, but it contains sufficient information for the steps to follow in our pipeline.

Once the structure of the environment is learned, the waypoint can be obtained in multiple ways. Alvarez et al [1] use a horizontal scan line at the center of 3D world map and find a point on that line that has highest depth value. Fraundorfer et al [5] use a 2D slice of 3D occupancy grid map based on MAV's fixed altitude and forwards MAV in the direction of free region. These approaches either do not consider neighbouring information or are computationally expensive. More importantly, since most of these approaches work on a fixed height, the results can be suboptimal. Moreover, they will fail if there is a wide obstacle, like a fencing wall, in front at MAV's fixed height. In our work, we have trained a Convolutional Neural Network with shortcut connections that regresses variable number of outputs. These outputs are bounding boxes containing navigable spaces. We choose center of the biggest box as the next waypoint. As our network can learn navigable spaces in any part of the scene, our approach is not bound by fixed altitude constraints.

Object detection is a well studied problem in the vision community. However, detecting navigable spaces is a very different problem from real world object detection as the shape of these regions in disparity maps can be completely arbitrary. There can be many regions in disparity map representing high depth at various locations and occupying different amount of areas. We trained deep network to predict minimum one to maximum three navigable regions given the disparity map. For object detection, Ren et al [20] use Region Proposal Network and classification network with shared weights to jointly classify the object and predict its bounding box. Redmon et al [19] divide the images in grids and uses two anchor boxes per cell. Then the cells covering objects are merged; classification and detection are optimized together. Our neural network architecture is extremely simple and cleaner compared to Ren et al and Redmon et al. We are looking at the whole image at both training and testing time like Redmon et al, but we do not divide it into grids. Instead, we do multivariate regression on whole image. We achieve the variable number of outputs using a novel ground-truthing scheme. Our network is also different from Tulsiani et al [24] as we do not use any masking for variable number of outputs.

3. DEEPFLY FRAMEWORK

Our architecture is composed of two functional modules: 1) Disparity Map Generation 2) Waypoint Selection. Logically, this can be thought of as obtaining a depth map (as disparities are inversely proportional to depth) and interpreting the depth map.

3.1 Disparity Map Generation

One of the big challenges in our application is to obtain a pair of images which is to be input to stereo correspondence algorithm. These correspondence algorithms usually take rectified pair of images as input. Also most of such image pairs are obtained using binocular stereo camera or precisely

calibrated monocular cameras. But in our case, the pair of images are obtained from quadcopter moving horizontally; it moves right to left to capture two images. These image pairs are not rectified. The baseline, though mostly in specified range, is not always exact. Moreover, there could be minor movements in forward/backward direction or up/down direction as quadcopter moves from right to left. For our work, we have tried to tackle these situations by observing the behavior of quadcopter during horizontal movements and explicitly modelling rectifying commands.

Once we have obtained image pair, we feed it to stereo correspondence algorithm. In our work, we are using SPS Stereo [25] to obtain disparity map. It is open source and produces dense disparity map. Other than SPS Stereo, we also tried SGBM [8] (OpenCV implementation) with fine-tuned parameters. It produces dense disparity map faster, but it is not as accurate as SPS Stereo. Hence we have used SPS Stereo in our work. At this point, it is crucial to investigate the claim that we can produce a meaningful disparity map by translating quadcopter horizontally. Figure 3 shows outputs produced by SPS Stereo of the same scene by using an image pair obtained from ZED Stereo Camera and image pair obtained by quadcopter. The results intimates that disparity maps are of comparable quality. We present accuracy of depth obtained by our method in Section 5. We believe two reasons responsible for good quality of disparity map: baseline is larger in our approach (25 centimeter) compared to baseline of ZED camera (12 centimeter) and the stereo correspondence algorithm is powerful enough to handle images obtained using less than ideal methods. The disparity maps in Figure 1 (right) are obtained by horizontal translations of quadcopter. Figure 5 shows results of disparity generation in extreme lighting conditions.

Figure 5: Left: Color Image of Scene in front in Challenging Lighting Condition. Right: Corresponding Disparity Map.[2]

[2] These images are dark and hence may not look detailed enough on all screens. They look fine with Intel Corporation Broadwell-U Integrated Graphics (rev 09) VGA controller. Also, zoom in for best visualization.

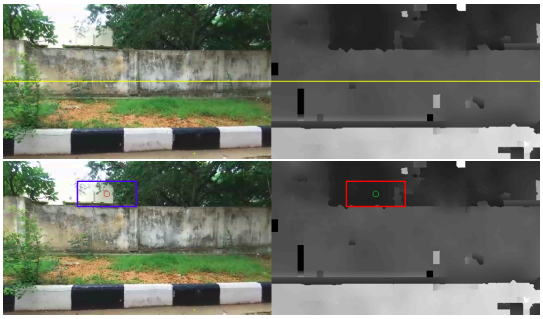

Figure 6: Top: Typical Failure Case for Fixed Altitude Methods. Bottom: Output of Our Deep Network for Same Case.

3.2 Waypoint Selection

The dense disparity map is input to deep network. At test time, our network outputs bounding box for minimum one to maximum three navigable spaces, depending on the structure of environment. The complete details of architecture, training, and dataset for our network can be found in next section. Even though generated disparity map is not as accurate as some other dense methods, we can cope up by learning the free space. The disparity map contains enough knowledge about free space and nearby obstacles. The output of the network are bounding boxes containing navigable spaces. We find the box with biggest area from all available boxes and choose its center as waypoint.

Many existing approaches use a scan line in depth map or a cell in 2D slice of 3D grid map to obtain waypoint. Such methods limit that quadcopter fly only at fixed height. And these methods will fail if there is a wide obstacle in front with height even slightly higher than what quadcopter flies at. Also finding a single point in depth map which has highest depth value is not very meaningful if we do not consider neighboring environment. There can be an immediate obstacle just at side of the point with highest depth. Our learning approach takes this into account and outputs the regions that are large and navigable, even if they are not farthest. This is fine in our approach because we let quadcopter fly for about 5 meters before it repeats the process of obtaining image pair, generating disparity map, and selecting next waypoint. Figure 6 shows one case where typical methods would fail but our method will work fine.

4. DEEP NEURAL NETWORK: DETAILS

4.1 Architecture

Our network is inspired from He et al [6]. It is a 51-layer network with shortcut connections performing multivariate regression. Figure 10 shows three core components of network (visualization using Netscope). The image data first goes through Convolution-BatchNormalization[9]-Scaling-ReLU block with convolution kernel size 7, stride 2 and number of filters 64. This block is followed by Max Pooling layer with kernel size 3 and stride 2 (figure 10, top). Post this point, the network is composed of repetitive structure known as “bottleneck” building block [6]. The bottleneck blocks contain three Convolution-BatchNormalization-Scaling-ReLU modules with convolution kernel size 1, 3, and 1 respectively. The shortcut connections are introduced by

element-wise summation of output of previous block and output of current block (figure 10, center). There are total 16 bottleneck blocks before average pooling through which tensors pass. The bottleneck blocks are followed by Average Pooling with kernel size 7 and stride 1. The first fully connected layer has 100 filters, followed by ReLU. The final fully connected layer has 12 outputs, four for each of possible three bounding boxes containing navigable space (figure 10, bottom).

Other than our 51-layer network, we tried 16 layer VGGNet [23] for our task. However, its performance was not upto level of the former. We could not try network any deeper than 51-layer because of GPU memory limitation. Our 51-layer network has more layers than VGGNet, however, the total number of parameters ($\sim$ 0.8 M) are extremely low compared to total parameters of VGGNet ($\sim$ 138 M). We cannot use any pretrained model available as most of these networks are designed for color images and real world scenes.

4.2 Multivariate Regression

Our network predicts bounding boxes for one, two, or three navigable spaces from given disparity map. We do not use any Region Proposal Networks [20] or divide image into grids and process each grid [19]. Our approach is extremely simple and cleaner. We train the network end to end with some special number when it has to predict less than maximum possible number of outputs. For example, our network can predict maximum three bounding boxes. If the environment in front has two or one navigable spaces, we will use the special numbers once or twice respectively. In our case, we have used 0 as the special number. To make example more concrete, when there are only two navigable spaces in front, we will train network with eight numbers specifying coordinates of two bounding boxes containing navigable spaces and four zeros for the remaining target values.

The network learns this phenomenon quite well. At test time, whenever the number of navigable spaces are less than maximum, the network predicts very small values close to zero for the remaining bounding boxes. These small numbers can easily be thresholded and discarded. Hence, the network learns to regress variable number of outputs without any assistance like masking or region proposals. But there are some caveats here which must be considered. First is that there should be an upper bound on how many outputs there can be or what is maximum number of outputs that we care about. In our case, we chose three regions as maximum possible because it was the maximum number of possible regions in more than 90% of our data. The second thing to note is that there should be enough examples of different number of outputs for network to learn such phenomenon. This is a very fundamental point that is relevant in many scenarios in machine learning.

4.3 Dataset and Training

Because of unavailability of any standard dataset related to our work, we created our own dataset. Our augmented dataset contains about 12000 images of 640 x 368 resolution. The input to the deep neural network is greyscale disparity map. We normalize the disparity map in range 0 to 255 before training and testing. This normalization essentially takes care of varying baselines among different pairs of images obtained by translation of quadcopter. We annotated ground truth boxes manually, from left to right up to three boxes per image. Each box was annotated such that it covers maximal navigable (dark) area without hitting any bright regions (obstacles). We performed two types of data augmentation – mirroring and Gaussian smoothing. The size of the dataset can be boosted greatly using random cropping. However, we observed that the overall structure of the scene is important in navigation task. We ground-truthed data with the philosophy that the regions in the center should be given more preference than regions near edges. Moreover, we maintained to avoid navigable spaces that are close to immediate obstacles. Such knowledge propagation cannot be performed efficiently if we crop image in multiple parts. Hence, we did not augment dataset using cropping. We use the said resolution disparity map and did not scale it down because regression is a hard problem and more details are helpful.

We trained the network using Caffe library [10]. We initialize the weights as in [7] and train our network from scratch. We use Stochastic Gradient Descent with mini batch size of 6. We chose the size to be maximum that we can fit on GPU. We start with learning rate 0.000001 with step size of 25,000 and multiplying factor 0.1. However, we increase learning rate after 50,000 iterations and again follow step policy of decreasing it by factor 0.1 every 25,000 iterations. We use Euclidean Loss for learning. The model is trained for 150,000 iterations. We use weight decay of 0.0001 and momentum of 0.9, and we do not use dropout.

5. EXPERIMENTS

We have evaluated our DeepFly framework on a low cost commercial Bebop quadcopter by Parrot. It is equipped with frontal monocular camera, ultrasound altimeter, and onboard IMU. It transmits frames at 640 x 368 resolution and 30 fps to the host device through WiFi connection. We use Robot Operating System (ROS) [18] as middleware for communication with quadcopter. The whole pipeline of quadcopter making a horizontal translation, disparity map generation, and waypoint selection takes about 1.3 seconds. The time taken by our pipeline is relatively low and quadcopter can safely hover for necessary period. Because of the limited computational power onboard, we perform all calculations on host system.

We perform experiments in wide range of environments such as among the trees, on roads, at open spaces, in corridors, in halls, and on stairs. We present some of our results in Figure 7. We also test our system in challenging lighting conditions (Figure 5). Find the statistical details of experiments performed in Table 1. In the table, we divide the failure cases into two categories: (1) Failure in quadcopter translation and disparity generation (2) Failure by the neural network. We consider the experiment successful if the biggest bounding box from which next waypoint is to be selected indeed represents ideal navigable space. Table 2 presents some statistics about average depth error for various depth ranges using our method. We use IMU data to to measure baseline.

The horizontal translation of the quadcopter will fail if the external forces such as wind are too great. By failure, we mean that the quadcopter will not translate enough or will translate too much. Because of this abnormal translation, the produced disparity map could be different from the disparity maps the network is trained to work upon.

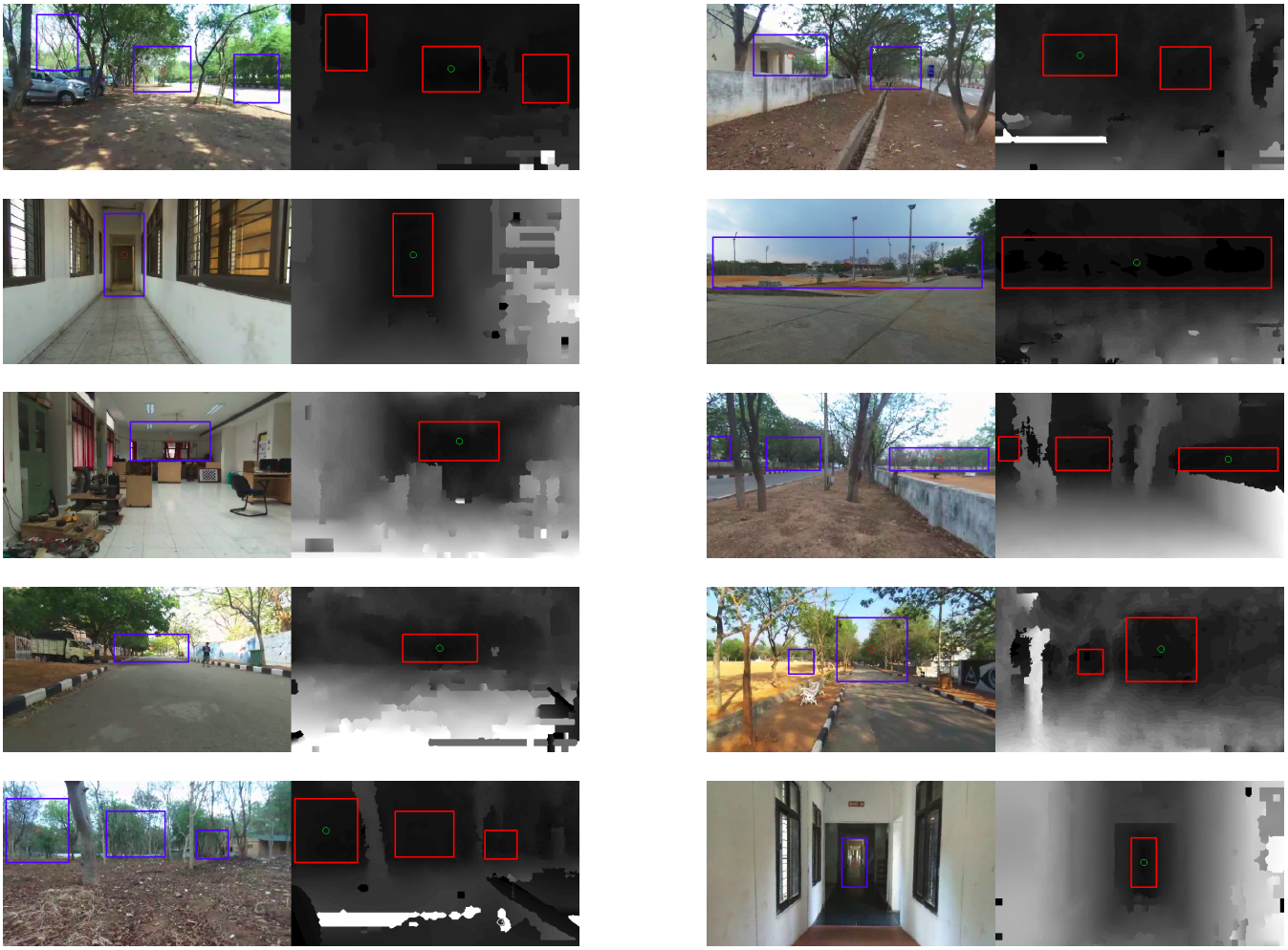

Figure 7: Some Results of Deep Neural Network in Different Environments. The Center of the biggest box is selected as next waypoint (red/green circle).

The failure cases in disparity generation means strong errors in disparity map. These failure case occurs in case of extremely texture less regions. We present couple of failure cases in Figure 8.

It should be noted that it is easy to detect translation failure using monocular SLAM. We use ORB SLAM [15] in our experiments to keep account of the translations. However, the errors in disparity map cannot be detected easily. When the neural network fails, it outputs the bounding boxes in some special ways which can be caught easily, as discussed in next section. Figure 9 shows some more results; these results are fine for navigation purposes since we get the correct waypoint, but they can be improved. We believe that overall results of our system can improve if we have larger dataset.

6. DISCUSSION

6.1 Comments on Learning Capability of Deep Networks

While ground-truthing for training the network, we gave bounding box coordinates in particular direction: first four neurons of last fully connected layer learn bounding box of left most navigable region, next four neurons learn bounding box of second left most (or, equivalently, second right most) navigable region (if any), and last four neurons learn bounding box of right most navigable region (if any). The network successfully learned this directionality. At test time, whenever there are more than one navigable regions, neurons regressed bounding boxes in same direction as they were trained to do (first four neurons predicting left most region and so on). Interestingly, it is easy to detect failure cases of network because we observe that every time it fails to predict the navigable regions, it outputs bounding boxes which are not in strictly left to right order.

One more comment about our deep network is that since it was able to interpret variable number of outputs, we tried to figure out whether it can count. We changed the Euclidean Loss to Softmax Loss and tried to classify images based on number of navigable regions they have – class 0 indicates one region, class 1 indicates two regions, and so on. And the results show that our network can count regions too. This just strengthens our claim that if the maximum possible number of outputs are bound by upper limit, the deep

Table 1: Results

Environment	Total Runs	Successful Runs	Failure by (1)	Failure by (2)
Outdoors, Normal Light	80	71	4	5
Indoors, Normal Light	40	34	5	1
Abnormal Light (Indoors and Outdoors)	20	13	5	2

Table 2: Depth Accuracy

Depth Range	< 2 m	2 - 5 m	5 - 10 m
Average Error (in m)	0.1765	0.3823	0.7392

networks can learn to predict varying number of outputs without any complex architecture. Though, it is important to note that our problem is in grayscale image space and we are interested in multiple instances of one class. It would be interesting to realise these type of results with color images and for multiple classes.

Figure 8: Top: Failure Because of Error in Disparity Generation. Bottom: Failure Because of Neural Network.

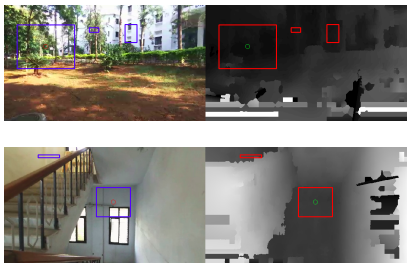

Figure 9: Some Improvable Results: Here, the Waypoint Selected is Fine for Navigation Purposes. Although the Results in Entirety can be Improved.

6.2 Limitations and Future Work

While we obtain working results in large number of experiments, the output of the neural network can still get better. We believe that current limitations on performance are because of the size of our dataset. Augmenting the current dataset can boost the performance of the network, which is one possible next direction.

One other limitation of our pipeline is the quality of generated disparity map. Though it is good enough for our purposes, it can still get better and faster by using more accurate algorithms. Better disparity map can also boost performance of our neural network. We did not train any deeper network than the 51-layer one because of GPU constraints, but that would be an interesting avenue. We are neglecting outer forces (like wind) and dynamic environments in this work. Incorporating this work with a global planner and handling the outer forces seems a fruitful direction.

7. CONCLUSION

We solve the problem of recognizing and bounding variable number of navigable spaces for a quadcopter with monocular camera. Understanding the scene structure in terms of navigable regions is a critical problem in autonomous navigation and can be challenging when the operating environments are heterogeneous and diverse. To this effect, we presented DeepFly framework, our first step towards complete autonomous navigation of quadcopter, that regresses multiple navigable regions. This was accomplished by obtaining dense disparity maps by horizontal translations of quadcopter. We use deep convolutional neural network with shortcut connections to regress multiple navigable regions from given disparity map. We present a novel ground-truthing scheme which can manage variable number of outputs. Making the pipeline more robust and faster, and incorporating it with a global planner are our possible next directions.

8. ACKNOWLEDGMENTS

This work was supported in part from grants made available by Rockwell Collins - IDC under the CSR program.

9. REFERENCES

[1] H. Alvarez, L. Paz, J. Sturm, and D. Cremers. Collision avoidance for quadrotors with a monocular camera. In *Experimental Robotics*, pages 195–209. Springer, 2016.

[2] K. Bipin, V. Duggal, and K. M. Krishna. Autonomous navigation of generic monocular quadcopter in natural environment. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1063–1070. IEEE, 2015.

[3] A. Bry, A. Bachrach, and N. Roy. State estimation for aggressive flight in gps-denied environments using onboard sensing. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1–8. IEEE, 2012.

[4] S. Daftry, S. Zeng, A. Khan, D. Dey, N. Melik-Barkhudarov, J. A. Bagnell, and M. Hebert. Robust monocular flight in cluttered outdoor environments. *arXiv preprint arXiv:1604.04779*, 2016.

[5] F. Fraundorfer, L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, and M. Pollefeys. Vision-based autonomous mapping and exploration using a quadrotor mav. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4557–4564. IEEE, 2012.

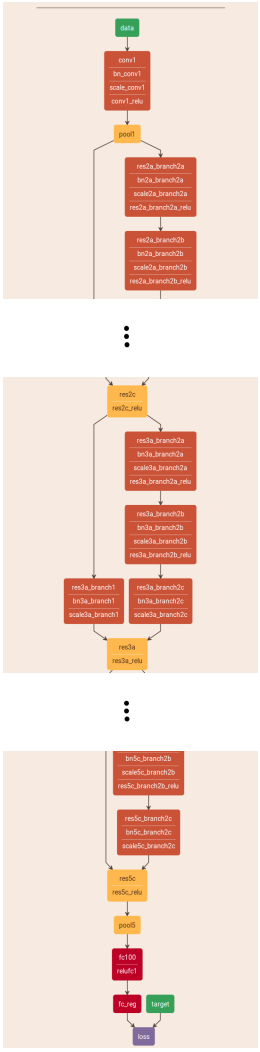

Figure 10: Deep Neural Network: Building Blocks

[6] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[8] H. Hirschmuller. Stereo processing by semiglobal matching and mutual information. *IEEE Transactions on pattern analysis and machine intelligence*, 30(2):328–341, 2008.

[9] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[10] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

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

[12] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, pages 225–234. IEEE, 2007.

[13] T. Krajník, M. Nitsche, S. Pedre, L. Přeučil, and M. E. Mejail. A simple visual navigation system for an uav. In *Systems, Signals and Devices (SSD), 2012 9th International Multi-Conference on*, pages 1–6. IEEE, 2012.

[14] J. Langelaan and S. Rock. Towards autonomous uav flight in forests. In *Proc. of AIAA Guidance, Navigation and Control Conference*, 2005.

[15] R. Mur-Artal, J. Montiel, and J. D. Tardós. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[16] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*, pages 2320–2327. IEEE, 2011.

[17] M. Nieuwenhuisen, D. Droeschel, M. Beul, and S. Behnke. Obstacle detection and navigation planning for autonomous micro aerial vehicles. In *Unmanned Aircraft Systems (ICUAS), 2014 International Conference on*, pages 1040–1047. IEEE, 2014.

[18] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5. Kobe, Japan, 2009.

[19] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. *arXiv preprint arXiv:1506.02640*, 2015.

[20] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[21] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *Advances in Neural Information Processing Systems*, pages 1161–1168, 2005.

[22] M. F. Selekwa, D. D. Dunlap, D. Shi, and E. G. Collins. Robot navigation in very cluttered environments by preference-based fuzzy behaviors. *Robotics and Autonomous Systems*, 56(3):231–246, 2008.

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

[24] S. Tulsiani and J. Malik. Viewpoints and keypoints. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1510–1519. IEEE, 2015.

[25] K. Yamaguchi, D. McAllester, and R. Urtasun. Efficient joint segmentation, occlusion labeling, stereo and flow estimation. In *European Conference on Computer Vision*, pages 756–771. Springer, 2014.