Word Count: 3264

Plagiarism Percentage 3%

Matches

1 **World Wide Web Match**
View Link

2 **World Wide Web Match**
View Link

3 **World Wide Web Match**
View Link

4 **World Wide Web Match**
View Link

5 **World Wide Web Match**
View Link

6 **World Wide Web Match**
View Link

7 **World Wide Web Match**
View Link

Suspected Content

Multimodal Navigation for Autonomous Drones Pruthvish E

Department of Computer Science Engineering PES University Bengaluru, India umapruthvish @gmail.com Vishwas N S Department of Computer Science Engineering PES University Bengaluru, India vishwas3151999 @gmail.com Prof. K S Srinivas Department of Computer Science Engineering PES University Bengaluru, India srinivasks@pes.edu Abstract— 1

Drones have versatile use cases from doorstep delivery to finding witnesses in crime scenes. Making the complete working of the drones autonomous is the problem at hand , autonomously making optimal decisions by taking actions based on the state of the surroundings , while decisions differ from state to state and from one surrounding to another, we can broadly classify the surroundings has external and internal which covers most types of surroundings , and agents way of learning is different in this type of surroundings and the transition from one surrounding to another is rather a complex task to make it completely autonomous , in this paper we have implemented ways to autonomously traverse in this surroundings and make suitable transitions from one surrounding to another optimally using reinforcement learning and other deep learning routines. Keywords—Multimodal, pv tradeoff , pv modes , delicacy point , switch optimality ratio , SLAM, frontier exploration, autonomous robots, drone . I. INTRODUCTION We usually come across two types of environments or surroundings around us namely outdoors and indoors and these environments possess different properties and states. Outdoors usually have lesser obstacles and has more exploration opportunity compared to indoors which is usually restricted in space having lot of obstacles and less degrees of movement, localisation in outdoors can be done using a coordinate system or a gps localisation will suffice where we can afford some error in the position or localisation of the drone whereas in indoors the localisation should have minimum error rate as the movement is restricted , therefore we need to come up with a better localisation method for the drone. Simultaneous localisation and mapping (SLAM) is needed for the drone in all the environments to effectively make decisions and navigate through the environment to achieve the desired goals, having various sensors like Barometer , Gps, Imu, Distance Sensor, Magnetometer, Camera, Lidar ,..etc. we can solve this problem of SLAM. Using simulation methods we have simulated the working of the drone in these environments , we have used airsim and ROS’s gazebo to simulate the outdoor and indoor environments. We have trained the drone in these two environments separately using different algorithms and strategies for the specific environment. Transitioning from one environment to another is a complex process as the exact point of transition is unknown and is hard to automate and generalize for all cases. This point as we call it , the delicacy point, is very sensitive and controls the optimal decision making process of the drone. In this paper we have implemented a method to optimally make this transition from outdoor to indoor with minimum loss. Training the drone for SLAM and navigation is a tedious task in both these environments and requires high computing devices, In this paper we have used some optimizations to reduce these computations. Multimodal navigation of autonomous drones is the process of navigating in any environment and optimally switching between various modes (outdoor, indoor) based on the surrounding states. In this paper we have implemented the complete task of navigation in both outdoors, indoors and transition between these environments as required for smooth navigation of the drone with minimum collisions and utilisation of resources to attain a certain goal. II. OUTDOOR ENVIRONMENT A. Properties Outdoor Environments have more margin for error compared to the indoors, outdoors have more exploration opportunities, outdoors have a set of states usually consisting of obstacles, available paths and degrees of freedom, There is no constraint or little constraint on the height of the drone as sky's the limit, localisation can be easily done through a gps sensor, there are available google APIs for mapping and planning of the path of the drone. B. Technologies Used Microsoft's open source application airsim with

unreal engine is used to simulate the outdoor environment, the simulation

tries to replicate with highest accuracy the dynamics and physics of the drones and surrounding environment having weather support and various other external features like air resistance ,..etc. Input to the network is the state of the environment at the C. Algorithm present time step , policy network is used to choose the optimal action for the present state which increases the We have used Deep Q networks for reinforcement returns in the future and attains the goal(in our use case learning of the drone to navigate in the environment with reaching a destination coordinate with minimum collisions optimal policy function with 6 degrees of action and we and time) , we use the target network to calculate the loss have used masked R-CNNs to detect and identify objects in using the huber loss function by computing reward the environment. (fig b) evaluation function. Masked R-CNNs are implemented as specified by [8] using Reward evaluation function returns the reward for a tensorflow and keras , we are using a pre-trained model with particular action while -100 suggests a failure(end of that state of the art accuracy to segment the image and detect and episode and drone resets to initial state) and 100 suggests a identify objects. win (drone reaches the destination with minimal collisions and time), each timestep the drone is ideal we give it a -1 Our Deep Q network has used a replay memory with a reward and every time the drone crashes to an obstacle we history length of 7, keeping track of the latest 7 (state, give it a reward of -10 , every time the drone reaches closer action, reward) experience. to the destination point (distance measured using euclidean distance) we reward the drone by +1 while going away from Our policy network has an input layer , followed by 3 the destination point we give a -1 reward , we end the CONV2d layers with a relu activation function followed by episode if we either reach the destination or have -100 a flatten layer and 2 dense layers and an output layer with a reward which indicates poor performance of the drone, as linear activation function(fig a). the reward decreases every time step of idleness we do not end up in a stagnant state. We have another cloned network of the policy network called the target network used for finding losses from the Based on the rewards our target network which has an given output of the policy network and the optimal Q value earlier weights or un updated weights is used to calculate the or returns in the future states. return from the future states

we use the huber loss as the loss function. 5

We use Adam optimizer with experimentally chosen parameters for optimal learning of momentum values and learning rate, we then backpropagate to update the weights. While choosing the action we have two possibilities one of exploiting the present state and exploring the present state , this tradeoff of exploration and exploitation we tweak the policy network to Exploration policy using Linear Epsilon Greedy method where the rate of exploring new states decreases as we progress in learning the environment from each time step. Our actions are defined with a scaling factor which changes the velocity with which the drone travels to a particular point , this scaling factor is increased everytime we choose a similar action, the set of actions defined are (+/-

scaling factor, 0, 0) , (0, +/- scaling factor, 0) , (0, 0, +/- scaling factor) 4

we choose one of this actions after each forward propagation through the policy network. We have trained the model for 50000 episodes , where each episode has an average of 10000 time steps, and have attained really good results. Mask R-CNN gives a list of objects with their coordinates in the present state, processing the input taken from the camera on the drone. Fig a. Policy Network Fig b.Outdoor Algorithm Object detection is essential in decision making of the drone and can be used for various applications. D. Results Our DQN model for the outdoor environment performs really well in our simulated environment with little or no crash , and reaches a particular target point in optimal time steps.Obstacles are avoided and the drone explores other routes to reach the destination in case of an obstacle , we have trained our model in an environment simulation with just obstacles for which the drone has to have utmost accuracy and control to avoid collisions , see fig c, d, e and f for some screenshots of the environment and drone actions.Graph of collisions vs episodes is a concave upward decreasing graph as collision count of the drones in various episodes of the game decreases has the agent learns, we can observe the collision count being almost zero in case of later episodes after the drone learns the weights. Fig c. Obstacle course - 1 Fig d.Obstacle course - 2 Fig e.Outdoor Environment - 1 Fig f. Outdoor Environment - 2 III. INDOOR ENVIRONMENT A. Properties Unlike external environments, indoor navigation requires precise map building and path planning due to complex building architectures. Using drones to navigate in such environments requires a 3D

representation of the building to generate a collision free path. 3

In this project, we have explored the possibility of autonomous navigation in indoor environments using drones, and also autonomous generation of 3D maps. B. Technologies Used This part of the project uses ROS (Robot Operating System) as the base framework. ROS is a framework that provides the necessary tools for developers to simulate and build various kinds of robots. It also acts as a community for different institutions and individuals to collaborate and create useful applications in the field of robotics. Gazebo

is an open-source simulator used to design, develop and 2

test different types of robots. It's easy integration with ROS, and availability of different robot and environment models makes it ideal for this project. Simulation of an indoor environment with multiple rooms and corridors, and an open-source quadrotor model (hector quadrotor) has been used to test our algorithm on the Gazebo simulator. C. Algorithm In this project, we implement two features that are crucial for autonomous drones to work in indoor environments, i.e., navigation and mapping. Both these processes need to be automated in order to have a truly autonomous robot. Autonomous navigation - Navigating a drone autonomously to a destination involves path generation such that the drone does not collide with obstacles and also sticks to the path while constantly avoiding dynamic obstacles. Move-It

is a ROS package that allows 7

us to generate paths through a given map by performing collision checks with the simulated model of the robot. RRT Connect is a variation of RRT (Rapidly exploring Random Tree) that searches from both the source and destination until the paths meet at a common point. This combined path is then used to traverse the map and reach the destination. The current occupancy map and a 3D mesh of the robot is provided as the input to this algorithm and a list of waypoints to be traversed is returned as the output. This path is referred to as the global plan. Once a global plan is received, the goal of the robot is to stick to this plan as much as possible while avoiding any obstacles that might have not been considered in the map. Such obstacles are referred to as dynamic obstacles. Dynamic obstacle detection is achieved by constantly monitoring the depth camera feed and other sensors placed on the robot. This task is performed by a local planner. The below figure shows a brief overview of the algorithm. See fig g for the indoor algorithm flow chart. Autonomous mapping - Creating a map of an environment is often performed manually due to a cyclic dependency between mapping and navigation. Navigating to a point requires a map of the area in order to plan a path, and creating a map autonomously requires the bot to navigate to different areas. However, manual mapping of an environment is often time consuming, and in some cases (such as space exploration), impossible. Hence automating the process of mapping is a crucial step in having a truly autonomous robot. In this project, we use the “frontier exploration” technique combined with heuristic based greedy selection to explore unknown regions and incrementally build a map. The occupancy grid created contains three types of regions, i.e., free, occupied, and unknown. A frontier can be defined as the border between free, and unknown regions of a map. The exploration algorithm keeps track of all such frontiers available in the current point, and iteratively visits them, till there are no frontiers left or a specified amount of the map has been explored. Frontier selection is based on an estimation of the amount of space that can be discovered if that frontier is explored, which is assumed to be directly proportional to the size of the frontier. This is determined by performing BFS on the frontier cells and grouping them into different frontiers. This combined with the

distance of a frontier from the robot’s current position 6

(in order to encourage the exploration of frontiers that are

closer to the robot’ s position) is used as the 3

heuristic. Fig g. Indoor Algorithm D. Results Using the above mentioned algorithms, we were able to IV. TRANSITION FROM ONE ENVIRONMENT TO ANOTHER successfully map a complex indoor environment Outdoor and Indoor environment navigation have been autonomously while minimising the amount of time required described previously. The transition from one to another to explore the entire region, and also navigate to different plays a crucial role in performance and overall accuracy of locations inside the map. The below figure shows the indoor the drone, optimal transitions provide optimal results and a environment simulation on the right with its corresponding means of multimodal navigation of the drone. map being visualised on the left. A sample path is also shown in the figure which the drone follows to explore the We use probabilistic methods to attain optimal transition map. to find the delicacy point. Fig h. Indoor mapping and navigation A. Algorithm Based on the coordinates specified by the destination in an outdoor environment we use probabilistic methods to find the way to the indoor environment and later change the mode of the drone to indoor navigation and mapping. We will define the two modes of the drone as the pv modes one for outdoor navigation and the other for indoor navigation, we will define the tradeoff in the transition point between the modes as pv tradeoff , pv tradeoff is the use of an appropriate pv mode during the transition period. We train our model during this transition point by randomly choosing the pv modes and based on the performance if the model we define a switch optimality ratio , which the ratio of each pv modes in a particular episode , we calculate the overall switch optimality ratio and keep track of the state of the drone before the transition at the delicacy point, we can store the visited destinations delicacy points for future applications , based on the state of the drone and the environment we compute the switch optimality ratio if this ratio tends to >1 then we switch to the other mode else stay in the same mode till the next state transition has this point is not a delicacy point. While indoor navigation is a suitable pv mode in the vicinity of the destination , it is computationally expensive while outdoor mode is less expensive to make the next action, if the delicacy point is far-off from the destination then outdoor mode is the preferred pv method hence there is tradeoff between this modes known has pv tradeoff. Based on experimental results we choose a random number between 1 to 100 if the number is >20 we switch to indoor pv mode. B. Results As our pv tradeoff is very low only 20% our model performs well in the overall process but the computation power increases. In our episodes the delicacy point is found roughly 60% of the time. V. OVERALL

RESULTS Outdoor navigation has an average of 4 collisions per episode and takes optimal paths 70% of the time. While being ideal or making wrong decisions on an average of every 50 time steps. Our model takes around roughly 50000 episodes with an average of 10000 time steps to learn. Delicacy point is found roughly 60% of the time while in transition. VI. ● ● ● ● ● ● ● ● ● APPLICATIONS End to End delivery In forensics to collect evidence. Mapping of an unknown or partially known environment To assist factory workers. To assist in tasks where the conditions are really bad for people to explore. Cave and unknown places exploration. To assist officials in case of hostages. To find missing objects. To help in collecting data for various purposes like making a count of the average number of people through a particular subway or other survey details. VII. FUTURE WORK We are yet to integrate the mask R-CNN with the drone to better understand the environment, optimizing the probability of accurately transitioning and finding the delicacy point with utmost perfection, training the model with other simulation environments so our model is not environment specific or reducing the bias in our model. Since we can say that such a model is feasible from the results of our project, the next step would be to implement this model on a real drone and test it under various real world scenarios. ACKNOWLEDGMENT We would like to thank Prof. K S Srinivas for guiding us throughout the project and helping us overcome various roadblocks that we faced. We would also like to thank PES UNIVERSITY for providing us the right environment and encouraging us to complete our project on time. REFERENCES [1] Durrant-Whyte, Hugh, and Tim Bailey. "Simultaneous localization and mapping: part I." IEEE robotics & automation magazine 13, no. 2 (2006): 99-110. [2] Bailey, Tim, and Hugh Durrant-Whyte. "Simultaneous localization and mapping (SLAM): Part II." IEEE robotics & automation magazine 13, no. 3 (2006): 108-117. [3] Shah, Shital, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. "Airsim: High-fidelity visual and physical simulation for autonomous vehicles." In Field and service robotics, pp. 621-635. Springer, Cham, 2018. [4] Madaan, Ratnesh, Nicholas Gyde, Sai Vemprala, Matthew Brown, Keiko Nagami, Tim Taubner, Eric Cristofalo, Davide Scaramuzza, Mac Schwager, and Ashish Kapoor. "AirSim Drone Racing Lab." arXiv preprint arXiv:2003.05654 (2020). [5] Mnih, Volodymyr, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves et al. "Human-level control through deep reinforcement learning." Nature 518, no. 7540 (2015): 529-533. [6] Van Hasselt, Hado, Arthur Guez, and David Silver. "Deep reinforcement learning with double q-learning." In Thirtieth AAAI conference on artificial intelligence. 2016. [7] Conte, Gianpaolo, and Patrick Doherty. "An integrated UAV navigation system based on aerial image matching." In 2008 IEEE Aerospace Conference, pp. 1-10. IEEE, 2008. [8] He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask r-cnn." In Proceedings of the IEEE international conference on computer vision, pp. 2961-2969. 2017. [9] Stanford Artificial Intelligence Laboratory et al., 2018. Robotic Operating System, Available at: https://www.ros.org [10] Meyer J., Sendobry A., Kohlbrecher S., Klingauf U., von Stryk O. (2012) "Comprehensive Simulation of Quadrotor UAVs Using ROS and Gazebo." In Noda I., Ando N., Brugali D., Kuffner J.J. (eds) Simulation, Modeling, and Programming for Autonomous Robots. SIMPAR 2012. Lecture Notes in Computer Science, vol 7628. Springer, Berlin, Heidelberg [11] S. S. Belavadi, R. Beri and V. Malik, "Frontier Exploration Technique for 3D Autonomous SLAM Using K-Means Based Divisive Clustering," 2017 Asia Modelling Symposium (AMS), Kota Kinabalu, 2017, pp. 95-100.