Flow Synthesis Based Visual Servoing Frameworks for Monocular Obstacle Avoidance Amidst High-Rises

Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science
in
Computer Science and Engineering by Research

by

Harshit Kumar Sankhla
2018900050
harshit.sankhla@research.iiit.ac.in

International Institute of Information Technology, Hyderabad
(Deemed to be University)
Hyderabad - 500 032, INDIA
May 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " **Flow Synthesis Based Visual Servoing Frameworks for Monocular Obstacle Avoidance Amidst High-Rises** " by **Harshit Kumar Sankhla**, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Dr. K. Madhava Krishna

To *posterity*

Acknowledgments

First and foremost, I would like to thank my guide, Prof. Madhava Krishna, for allowing me to become a part of the Robotics Research Center (RRC), IIIT Hyderabad. As a research scholar, he nurtured me with his unparalleled guidance. Also, he enabled me with resources and opportunities to build my interest in robotics and develop skills to deliver at the highest level.

At the RRC, I was actively involved in an outdoor passenger drone project sponsored by MeitY, which I also led during the latter years of my program. This project introduced me to the field of aerial robotics. I want to thank the students here, Gaurav, Pravin, and Animesh, who have been active co-working members and guided me in my initial phase of this project, and Rishabh Dev, Sudarshan, and Bhanu, who are now carrying it forward. I would also like to thank the interns Ritik, Bharath, Rishab, Pintu, Deepak, and Anuj for their cameos in making our drones fly higher and faster. Also, Saharsh, Pawan, and Sai Shankar, who were interns on other projects and great company while working.

I thank my co-authors Nomaan, Shankara, Vedansh, Gunjan, Harit Panday, Ayappa, Tanu, and Mukul. Together we have had the opportunity to work on challenging research problems and publish our work at prestigious conferences.

Writing this thesis, hence completing my journey at IIIT, I want to mention some special people close to my heart who have been a tremendous support system throughout my toiling years and with whom I've experienced a life that transcends beyond an academic environment. Shantanu, Udit, Kaustubh, Mithun, Shubodh, Aasheesh, Dhagash, Kinal, Amit, Krishna, Omama, Shanthika, Mounika, and Ihita. Each of you has had a significant impact on my life, and I thank almighty for all the lovely memories I have made with you guys.

Being on campus, we come across many people we connect with through sports, fests, music, and all sorts of events. I can not name all of them, but I would like to acknowledge their contribution to making my life here genuinely splendid.

Last but not least, I would like to thank my parents, my brother, and my entire family, for being a source of solid emotional support that has gotten me through this time on campus. Thank you for being there for me.

Abstract

The goal of this thesis is to design a flow synthesis based visual servoing framework enabling long-range obstacle avoidance for Micro Air Vehicles (MAV) flying amongst tall skyscrapers. Recent deep learning based frameworks use optical flow to do high-precision visual servoing. This work explores the question: can a surrogate flow be designed for these high-precision visual-servoing methods which leads to obstacle avoidance? The concept of saliency is explored for identifying high-rise structures in/close to the line of attack amongst other competing skyscrapers and buildings as a collision obstacle. A synthesised flow is used to displace the salient object segmentation mask. This flow is computed in a way that ensures that the visual servoing controller maneuvers the MAV safely around the obstacle. In this approach, a multi-step Cross-Entropy Method (CEM) based servo control is employed to achieve flow convergence, resulting in obstacle avoidance. This novel pipeline is deployed on an MAV to successfully and persistently maneuver amongst high-rises and reach the goal in simulated and photo-realistic real-world scenes. Extensive experimentation is conducted and results are presented to compare the proposed approach with optical flow and short-range depth-based obstacle avoidance methods hence conclusively demonstrating the proposed framework's merit. Additional visualisations can be found here.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

As Urban Air Mobility (UAM) in the context of smart cities gains popularity, drones and Micro Aerial Vehicles (MAVs), or, Unmanned Aerial Vehicles (UAVs), in the general sense, navigating through high-rises become imminent. Low-cost off-the-shelf RGBD sensors are noisy [16] and detect obstacles reliably only when they are in the typical range of 7-10m. Thus long-range obstacle detection and maneuvering become essential. In this context, this work proposes a novel flow synthesis-based deep visual servoing framework for monocular obstacle avoidance, wherein urban high-rises/skyscrapers are indicated as obstacles. The task above becomes additionally more challenging, given the UAVs' limited payload and compute capacity. The work carried out in this thesis tackles this problem by using a monocular camera as the only sensing modality and combining methods from flow synthesis, visual servoing, and salient object detection to build the overall pipeline. Independently and otherwise flow-based approaches to monocular obstacle avoidance [6, 38] have been popular in literature; nonetheless, their outcomes have tended to be unreliable due to often inaccurate flow estimates and controllers based on such flow. Instead, by synthesizing the desired flow, this approach preempts the challenges associated with the noisy flow, and the multi-step CEM-based control achieves superior goal convergence than a single-step reactive control based on flow.

Recent Visual Servoing frameworks [15, 25, 34] have leveraged deep optical flow to do high precision visual servoing. This work is poised on the question: If these frameworks can utilise optical flow to reach a goal image with such high precision, can a surrogate flow be designed for the obstacle which can push, said obstacle, out of the scene?

Given an urban high-rise scene as a monocular image input, this novel pipeline segments the Building of Concern (BoC) amongst other competing high-rises through a modified saliency segmentation network. The BoC is typically the building that will collide first with the UAV if it continues along its current trajectory. The pixels contained in the segmentation mask of the BoC are subject to a flow that quintessentially is the pixel error that needs to be minimized by the multi-step visual servoing based CEM controller. Fig.1.1 showcases a high-level overview of our idea.

The logic is to mimic the flow pattern or flow trajectory, an obstacle is subject to, when a camera performs an avoidance maneuver around that obstacle. The optical flow sequence of an obstacle as the

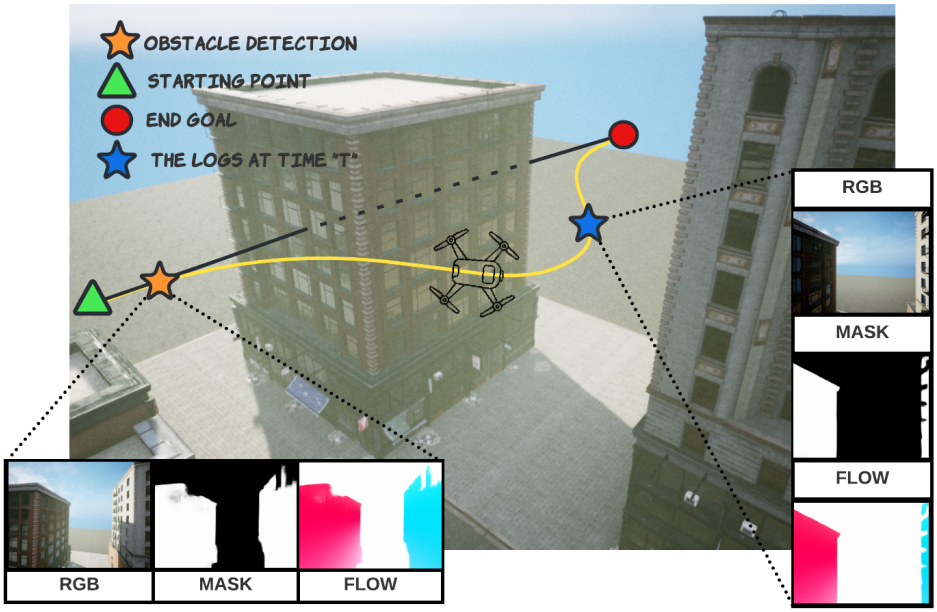

Figure 1.1 The proposed system explicitly segments the building to generate a surrogate desired flow for the high precision Flow-Based visual servoing algorithms. The 'pink' obstacle flow pushes the building leftward, and hence a leftward drone trajectory is generated by the servoing framework and vice-versa for 'blue' flow, which pushes the building rightward.

camera maneuvers around it is shown in Fig.3.3. The net displacement of the pixels of the obstacle due to the avoidance maneuver is typically the synthesized flow imparted to the obstacle. Hence, when the multi-step CEM controller strives to achieve this synthesized flow or minimizes this flow error, it realizes obstacle avoidance.

1.1 Contributions

1. Believably, this is the first approach to tackling the problem of navigation amongst urban high-rises with a single monocular camera as the essential sensing modality.

2. Repurpose the concept of salient object detection as collision obstacle detection. This deep model segments only the Building of Concern (BoC), thereby ensuring the relevance of our servoing based avoidance mechanism for an identified obstacle.

3. In contrast to previous methods of synthesizing control from estimated flow, which is often noisy, this thesis proposes noiseless surrogate flow synthesis and demonstrates its efficacy for monocular obstacle avoidance.

4. Benchmark analysis to demonstrate tangible performance gain with respect to popular flow-based obstacle avoidance methods [6] and other prior art when evaluated on photo-realistic simulation environments in AirSim.

5. In the context of UAM, this work emphasises the importance of image-based flow synthesis as an enabler of long-range avoidance maneuvering and portrays its advantage vis-a-vis short-range depth-based obstacle avoidance pipelines [41].

1.2 Thesis Organization

This thesis is divided into seven chapters. First chapter motivates the problem of navigation for UAVs among high-rise buildings in urban environments and the contributions of this work to address this task. Second chapter provides the literature survey on obstacle avoidance, specifically monocular camera based, methods that can be deployed on UAVs. Third chapter elaborates the various modules of the system and the overall pipeline. Fourth chapter describes the dataset, benchmark environment, results, accuracy metrics and comparative studies with other competing methods. Fifth chapter concludes the thesis and motivates further plausible work in the direction of visual servoing based obstacle avoidance methods for faster and more robust flight. Additionally, chapter six covers the work done in developing an outdoor passenger carrying drone, a sponsored project by MeitY, which also stemmed the research work done under this thesis. Chapter seven mentions the resulting publications.

Chapter 2

Related Works

Obstacle avoidance is one of the most fundamental tasks for any robot navigating an environment. It is an essential capability for robots in various applications, including search and rescue, inspection, and manufacturing. Many methods are available that use different kinds of environment representations, sensing modalities, and cost functions and are selected based on criteria of speed, computational requirements, and environment (indoor/outdoor), to name a few. There are several known approaches to obstacle avoidance in robotics. One approach is to use sensors, such as laser scanners, ultrasonic sensors, or cameras, to detect the presence and location of obstacles. The robot can then use this information to plan a path around the obstacle or to stop and wait for the obstacle to move. Another approach is to use machine learning algorithms to teach the robot to recognize and avoid obstacles based on previous experiences, using methods from reinforcement learning, where the robot receives a reward for successfully avoiding obstacles and a penalty for colliding with them. Here we briefly cover relevant research on different methodologies of obstacle avoidance available for MAVs.

2.1 Classical Methods

Obstacle avoidance can be considered a re-planning problem in light of an obstacle appearing on the robot's trajectory. Early obstacle avoidance methods relied on identifying obstacles through primitive features, such as edges, and maneuvering the robot around the obstacle by pushing it away from the visible edges[3]. Another approach works on building certainty grids of the robot environment using sensor information that probabilistically maintains a certainty value of an obstacle at a specific grid location [9]. On these environment representations, graph-search methods[17, 8] can effectively find collision free paths to a specified goal location, which in this case, would be a waypoint avoiding the object. These methods, however, are only applicable in static environments and would have to recompute the entire plan from scratch for any change in the environment. For real-time obstacle avoidance, artificial potential fields(APF)[14] and vector field histogram(VFH)[40] based methods have been popular. APFs use "virtual force fields" to guide the robot away from obstacles, where obstacles are represented as repulsive forces, and goals are represented as attractive forces. However, this method can cause the robot

to oscillate under specific scenarios, which is overcome in VFHs, in which the robot uses its sensors to measure the vector field in its immediate vicinity and selects a preferred direction of motion based on the vectors in the field. However, VFHs require an extremely good estimate of the robot's state and can not adapt to new obstacles.

2.2 Monocular Obstacle Avoidance Methods

While the literature is sufficiently populated with methods relating to obstacle avoidance with depth sensors such as RGBD cameras [41, 37] or stereoscopy [32, 31], prior art relating to monocular obstacle avoidance has been sparse. In [10] a Monocular Direct SLAM framework popularly called LSD SLAM is used to construct occupancy maps over which navigation and exploration is performed. However, monocular SLAM based occupancy maps are typically noisy that entail uncertainty-based obstacle avoidance formulation along the lines of [16] as an overhead. Moreover, as the authors mentioned, maps often need to be regenerated by intermittent hovering maneuvers that can prove costly as UAVs work with limited onboard power and compute budget. In [11] a similar SLAM-based approach using feature-based ORB-SLAM is presented. The sparse map is interpreted in terms of planes, and an RRT-based trajectory planner is presented. Once again, such sparse point cloud methods are subject to uncertainty-based overheads and need extra modules beyond SLAM to reinterpret the sparse point clouds.

2.3 Optical Flow and Feature Based Avoidance

In contrast to methods that explicitly build a map, there exists a class of methods that make use of optical flow cues to compute obstacle avoidance behavior. Flow balancing is a popular method of obstacle avoidance where the difference in flow between the horizontal and vertical sections of the image are balanced by inducing an appropriate yaw and height changes in the vehicle [6, 38]. Elsewhere flow is used to compute depth or time to collision that guides the control performance [2]. In [2] correlation-based motion detectors were also tried out as a mechanism for obstacle detection taking cues from biology. Whereas in [1] growth in feature sizes were the cue to detect the presence of an obstacle, and heuristic controls were used for avoidance. A primary challenge with optical flow-based control lies in the flow estimates being noisy. Often successive flow estimates can cause conflicting control actions such as back and forth motion primarily due to inaccuracies or noisy flow estimates.

2.4 Data Driven Methods

Data-driven methods for monocular obstacle avoidance have recently become popular. Black-box neural nets are used to learn disparity [5] followed by a traditional controller to avoid the perceived

obstacles over the depth maps. Depth maps learned from monocular vision are noisy and need further retraining in new scenes and entails that the controller resort to uncertainty-based formulations for obstacle avoidance such as in [16]. Recent advances in deep learning has given rise to end-to-end approaches which can directly predict control policies from raw images [26, 24, 39]. These methods can be broadly classified into two categories-

1. **Supervised Learning Based** [13, 36, 26]: These methods provide an easy way to train control policies, however performance is subject to quality of the expert data which is used to train the model. Also domain translation between the expert and agent is not direct in all cases.

2. **Reinforcement Learning Based** [24, 42, 28]: These methods are effective in learning generalized policies to navigate a drone in a given environment, however, require a large sample of operational data which is not easy to acquire.

2.5 Visual Servoing Based Methods

Visual servoing is the problem of active camera control that guides the camera to the pose corresponding to the desired image from the current image viewed at the current pose [21]. Literature abounds with a variety of servoing frameworks [23, 7, 30] while recently data-driven methods have gained in popularity [15, 25, 34]. In this paper, we use a multi-step CEM-based servoing controller that outputs control that minimizes the synthesized flow errors.

Chapter 3

Methodology

With a camera and a GPS sensor, the task addressed through this work is to generate the optimal velocity control commands to reach a desired goal GPS location g^* in a collision-free manner. Existing deep learning-based approaches like DroNet[29] and Reinforcement Learning based methods map the pixels directly to control commands. However, these methods lack interpretability, making failure modes challenging to understand and overcome. This poses several problems in the real-world deployment of these methods. Recently Deep Learning-based servoing methods [15, 25] use optical flow for high-precision visual servoing. The work carried out in this thesis extends on their capabilities and presents the design of a surrogate flow for obstacles that can push them out of the image. We first utilize the saliency-based architecture described in [33] to identify the obstacle or building in the line of attack of the UAV. We then introduce the Radial Flow in Sec. 3.2, which combines with the obstacle mask obtained using BASNet[33] to provide the desired flow for the servoing frameworks introduced in [15, 25]. We use MPC+CEM[25] to optimize for velocity commands; however, this only serves as the design choice.

3.1 Saliency Based Obstacle Avoidance

Saliency detection methods [4] aim to mimic the human visual attention mechanism to identify objects more attentive than the surroundings. Saliency detection has a range of applications, such as foreground-background segmentation, object tracking, identification, reidentification, detection, especially in cluttered environments. A relevant yet sparsely explored application is collision obstacle detection during navigation. A MAV traversing in an urban environment moves near and around many different yet significantly large, high-rise structures. For the task of autonomous navigation, we contextualize the idea of salient object detection as collision obstacle detection and propose a method for long-range obstacle avoidance in real-time flight as salient object detection methods perform well in detecting contextually relevant objects, even at large distances. Further, our downstream avoidance task utilises the generated saliency map to synthesise flow and generate velocity commands that can steer the

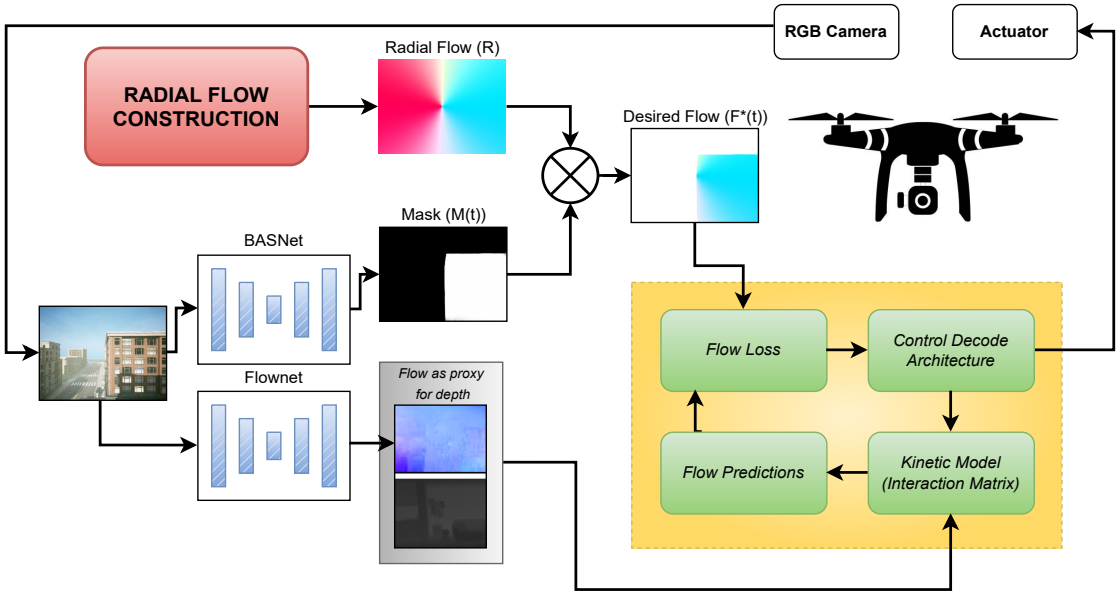

Figure 3.1 Here we summarize our obstacle avoidance pipeline. We use the saliency-based detection module BASNet[33] to get the obstacle mask $\mathcal{M}(t)$. We then combine it with the radial flow $\mathcal{R}$ to get the desired flow $\mathcal{F}^*(t)$. We use the synthesized desired flow $\mathcal{F}^*(t)$ along with a flow-based Visual Servoing pipeline to generate a control command using Cross Entropy Method.

drone clear from a collision. It can also efficiently work with imperfect object segmentation masks, and hence salient object detection methods justify our case for detecting objects of collision.

The salient object detection method we are using is BASNet[33], a U-Net[35] like supervised encoder-decoder based method with an additional residual refinement module which improves the coarse prediction, extending it to the object boundary. We train BASNet for identifying obstacles that appear in the line of sight of collision. Compared with popular segmentation methods [12, 18, 35] which would segment all competing object instances in the entire field of view without any criteria of relevance and making no distinction for a particular object of concern, and depth-based methods [41, 20] which are not reliable beyond stereo camera ranges and would detect large obstacles like buildings very late as compared to when they are visible, salient object detection methods handle the task well for long-range collision obstacle detection from a single RGB image.

We train BASNet on a custom dataset of UAV FPV images of skyscrapers from around the world and similar-sized model buildings in UrbanScene3D and AirSim (building 99 environment). We augment this dataset with transformations such as ColorJitter, MotionBlur, RandomBrightness, and RandomRotate to attain 4000 images. We follow the training methodology of the original authors, using the same hyperparameters, where the initial learning rate=1e-3, betas=(0.9, 0.999), eps=1e-8, and weight decay=0. The training loss converges after 400 iterations on a computer with an 8-core AMD Ryzen 7 2700X CPU, 64 GB RAM, and an Nvidia GTX 1070 Ti GPU.

3.2 Radial Flow

Recent visual servoing methods [15, 25] calculate the flow between the current image and desired image to do high precision visual servoing. However, during the monocular obstacle avoidance, the desired image is unavailable. Here we explain how we can design a surrogate flow, which can help us robustly avoid obstacles. Consider an image of dimension $(H \times W)$. For any pixel coordinate (i, j), we want to assign a flow value that pushes the pixel out of the image. This turns out to be a flow in a radially outward direction from the image center. This "radial" flow R and for any pixel coordinate can be described using :

$$R[i,j] = [i - \frac{H}{2}, \Lambda * (j - \frac{W}{2})] \tag{3.1}$$

Where Λ is a parameter and is set to 10. Λ remains same for each scene in our benchmark. We normalize the Radial Flow as:

$$N[i,j] = \sqrt{(i - \frac{H}{2})^2 + (j - \frac{W}{2})^2)} \tag{3.2}$$

$$R[i,j] = \frac{R[i,j]}{N[i,j]} \tag{3.3}$$

We use the mask $M(t)$ obtained using our saliency based obstacle avoidance module to get the obstacle mask. This mask $M(t)$ is then multiplied with Normalised Radial Flow R to get a desired flow for servoing as described in the next section.

$$\mathcal{F}^*(t) = R \cdot M(t) \tag{3.4}$$

The motivation behind the design of the radial flow can be explained using Fig.3.3. We want the desired flow as calculated in Eq. 3.4 to be biased to take the MAV away from the obstacle while also taking the MAV forward. However, we cannot assign the pixel-wise optical flow in a radially outward direction since it is the same as when the drone moves forward. Hence, we scale the component along the (left, right) direction, forcing the optimisation to produce a velocity that takes it away from the building while also moving forward. Additionally, if the center patch of the image is covered with an obstacle, we damp the forward velocity by a factor μ. This provides a robust surrogate desired flow $\mathcal{F}^*(t) = R * M(t)$ which can be used with a flow-based servoing algorithm to reliably generate optimal velocity commands as explained in section 3.3.

3.3 Avoidance Using Visual Servoing Framework

We now explain our algorithmic pipeline, which combines the saliency mask predicted using BASNet[33] and the radial flow to give a velocity control command to avoid the obstacles. The salient obstacle mask is obtained using the BASNet module. The mask at time t is combined with Radial Flow R to give a desired flow $F^*(t)$ for the current scene using Eq.3.4. We then calculate the 2-View Flowdepth Z_T

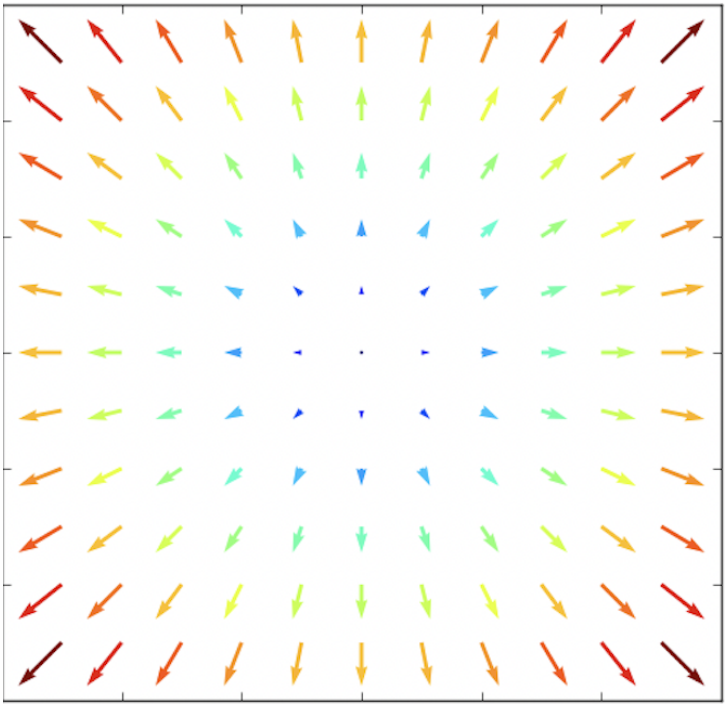

Figure 3.2 Qualitatively understanding the radial flow : For every pixel coordinate (i, j), we assign it a flow which pushes it in a radially outward direction. Instead of generating a whole new scene and generating a flow for that, we utilise the approximate optical flow obtained to induce a control law.

introduced in [15] which considers the optical flow between time step $t-1$ and t as a proxy estimate for depth. We use the flow network FlowNet2[22] to construct the interaction matrix $L(Z_t)$.

$$L(Z_t) = [r]\frac{-1}{Z_t}0\frac{x}{Z_t}xy - (1+x^2)y0\frac{-1}{Z_t}\frac{y}{Z_t}1 + y^2 - xy - x. \tag{3.5}$$

The interaction matrix $L(Z_t)$ maps the drone velocity to the velocity of pixels (or optical flow) on the image plane. We generate the flow predictions using the interaction matrix $L(Z_t)$.

$$\widehat{\mathcal{F}}(\mathbf{v}_{t+1:t+T}) = \sum_{k=1}^{T}[L(Z_t)\mathbf{v}_{t+k}] \tag{3.6}$$

We can then optimize the velocity for the desired flow:

$$\mathcal{L}_{flow} = \|\widehat{\mathcal{F}}(\widehat{\mathbf{v}}_t) - \mathcal{F}^*\| = \|[L(Z_t)\widehat{\mathbf{v}}] - \mathcal{F}^*\| \tag{3.7}$$

Optimising the loss function in Eq.3.7 gives the desired velocity for the avoidance of obstacle. We use Cross-Entropy Method (CEM) to solve for the desired velocity. At each time step we sample 'N' velocities from a Gaussian distribution $g(\mu, \sigma^2)$ and calculate the loss (Eq. 3.7) for each of them. The losses are then sorted, and the top 'K' velocities (with least losses) are used to update the parameters of

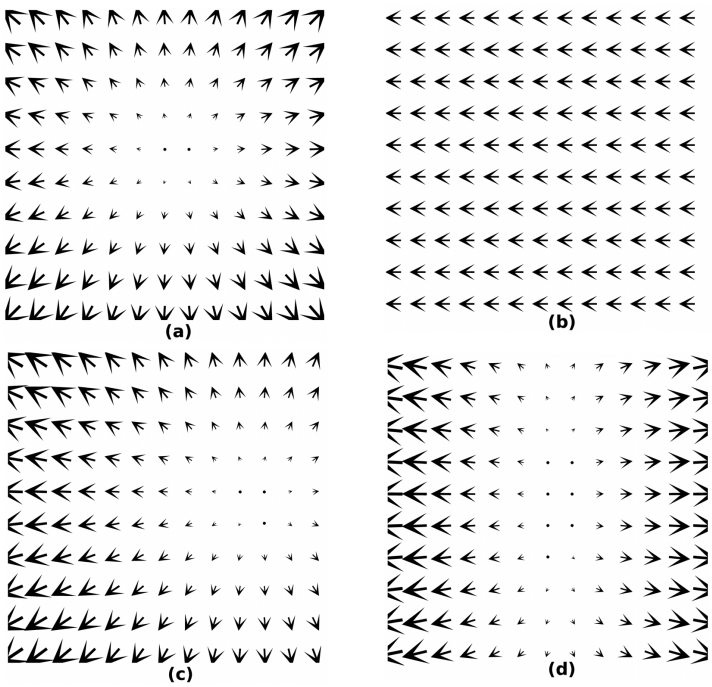

Figure 3.3 Visualisation of flow vectors according to how the drone moves : (a), (b) show the flow obtained when the drone moves forward and right, respectively. (c) shows the flow when the drone moves forward and right simultaneously. (d) shows a visualisation of the synthesized radial flow using Eq.3.1. We leverage the fact that when (d) is combined with the obstacle mask $\mathcal{M}(t)$, it forces the CEM optimisation to generate a velocity that moves the obstacle out of the scene. So if the mask shows that the building is dominantly in the right direction, the servoing generates the velocity, which takes it in (forward+left) direction, towards the side of the building, and vice-versa if the mask is dominantly on the left side.

Gaussian distribution $g(\mu, \sigma^2)$. We run CEM for several iterations before convergence is achieved. Our obstacle pipeline can be summarised using Fig.3.1.

3.4 Overall Pipeline and Implementation Details

We now describe our overall goal-reaching pipeline. We divide our algorithm into two parts, a) Goal Reaching mode: Here, we use GPS location to orient and move towards the goal location g^*. b) Obstacle avoidance mode: We give the velocity commands obtained (as explained in Section 3.3) to the drone. We use the obstacle mask $\mathcal{M}(t)$ obtained using our Saliency Based Obstacle detection algorithm to decide whether to follow Obstacle avoidance mode or not. If the center patch of the image is covered with an obstacle, we damp the forward velocity by a factor μ. Our algorithm can be used to do 6-DOF obstacle avoidance compared to other monocular obstacle avoidance algorithms, which typically use a 2-DoF control command. However, while comparing against these algorithms, we also use a 2-DoF velocity structure.

3.5 Flow Balancing with Radial flow

In this section, we describe how we can use the desired radial flow $\mathcal{F}$ obtained using Eq.3.4 to improve the performance of the Flow Balancing Method. Flow balancing has been classically used to avoid obstacles using a monocular camera. It uses the flow between the current image and the previous image to get a yaw angle for the drone. The yaw rate is given using the equation :

$$\dot{\theta} = (\frac{\omega_L - \omega_R}{\omega_L + \omega_R}) \times k \tag{3.8}$$

Here ω_L and ω_R represent the sum of magnitudes in the left and right half of the Optical flow between the current image I_t and previous image I_{t-1} and k is a constant which is an adjustable parameter. The reasoning behind the approach is that closer obstacles have higher flows, so we should move to the direction that has less flow. However, this method fails to work on our benchmark because, in long-range avoidance, the optical flow fails to provide enough information to give an optimal velocity command. To have a fair comparison against flow-balancing, we use the formulated radial flow $F^*(t)$ described in Eq.3.4 to get the yaw rate. Our Radial Flow-based Flow balancing gives much better results when compared to naive flow balancing.

Chapter 4

Experimentation and Results

The main objective of our paper is to present an off-the-shelf and easy to debug monocular obstacle avoidance algorithm which can avoid high-rise buildings. Most existing approaches for monocular obstacle avoidance focus on directly calculating the control command using a deep network. However, this leads to several practical problems while applying these methods in the real world. These methods suffer from a lack of interpretability, and failure modes can become hard to debug. Our approach explicitly segments the obstacle that needs to be avoided and uses the radial flow with servoing to avoid the building robustly. Additionally, our radial flow-based algorithm also has the ability to output the control command in a 6DOF space. To test the applicability and robustness, we present a new benchmark on synthetic environments and the real-world dataset UrbanScene3D. We compare our algorithm against a) Naive Flow-Balancing: It uses the difference of magnitudes of optical flow between the left half and right half of the image as a control command to do 2DOF control. b) Radial Flow-based Flow-Balancing: We use radial flow to improve the flow balancing approach instead of the optical flow between the current and previous image. We present several robust indicators which show that our proposed algorithm is much more reliable when compared to other baselines.

4.1 Simulation Benchmark

Our simulation benchmark consists of 10 photo-realistic scenes from the building 99 environment and 6 real-world reconstruction scenes from the UrbanScene3D dataset. These scenes span across a varying difficulty level depending on the number of buildings on the path to the goal location, the amount of free space available to the MAV while navigating between buildings, and the angle at which it approaches the building. We have a start and a random goal GPS position for each scene. In easy scenes, the robot has to follow a straight-forward trajectory to reach the desired goal location, while the medium scenes and hard scenes involve reaching the goal location after navigating across buildings by following a complicated trajectory. Additionally, hard scenes are designed to have space constraints, such that the agent has to make some clever decisions while navigating in narrow spaces. The urban scene 3D dataset contains real-world urban environments for cities like Shanghai, New York, etc., with

Scenes	Naive Flow Balance		Radial Flow based Flow Balance		Our Method	
	Min Dist	Traj Length	Min Dist	Traj Length	Min Dist	Traj Length
$Building1$	×	×	4.9801	65.733	**8.2812**	78.651
$Building2$	×	×	×	×	**8.4453**	89.152
$Building3$	×	×	0.2807	81.730	**1.3222**	58.425
$Building4$	×	×	×	×	**3.4189**	98.542
$Building5$	×	×	×	×	**9.6503**	96.384
$Building6$	×	×	×	×	**8.7850**	74.818
$Building7$	×	×	×	×	**12.273**	79.469
$Building8$	×	×	×	×	**2.9671**	77.462
$Building9$	1.2713	50.617	3.6186	50.186	**7.4885**	53.414
$Building10$	×	×	×	×	**3.0070**	59.624
$Residence1$	×	×	0.1072	38.636	**11.818**	66.058
$Residence2$	×	×	×	×	**11.937**	123.748
$Sci-Art1$	6.6581	108.258	13.007	108.386	**25.484**	103.597
$Sci-Art2$	×	×	×	×	**15.039**	167.726
$Campus1$	×	×	×	×	**10.265**	108.532
$Campus2$	×	×	×	×	×	×

Table 4.1 Minimum Distance and Trajectory Length: Min Dist signifies the minimum distance of MAV from any building in the image. We show that our method has the highest Min Dist among all the methods, which empirically proves that our method maintains the safest distance from the buildings. It is clear from Fig. 4.1 trajectory plots that our method takes the safest path to the goal while other methods either collide with the building or graze past it, and hence our trajectory length is highest. "×" indicates that the maneuver was not completed, and the drone collided.

realistic textures. This benchmark further tests the generalization of our methods by putting them in near-realistic scenes. We have chosen to conduct our experiments on the real scenes: Sci-Art, Residence, and Campus. Our method is able to navigate to the goal GPS location provided in all the scenes and is able to make intelligent choices required to navigate in a medium and hard scene. Fig. 4.1 shows the obstacle avoidance sequence for several scenes on our benchmark.

We show that our algorithm is able to navigate to the goal location in scenes across varying levels of difficulty. For Building 1, BASNet segments only the building in the line of attack amongst multiple competing buildings. For Building 4 and Building 10, the algorithm has to navigate in a narrow passage between two skyscrapers and make several key decisions to reach the respective desired goal locations. We are also able to navigate in scenes where a large part of the scene is covered with an obstacle. For e.g., in Building 5, our visual servoing optimisation pipeline is able to find the correct direction to move in order to avoid the obstacle. On the photo-realistic UrbanScene3D dataset, our algorithm generalises and gives similar performance. Our algorithm is able to work on 15 out of 16 scenes. Compared to our proposed novel algorithm, naive Flow Balancing works on only 2 scenes out of 16 scenes from our simulation benchmark. It is clear that the optical flow between the current and previous image is not able to provide any meaningful direction for the MAV to follow. However, our radial-flow guided

flow-balancing algorithm shows a stark improvement compared to naive flow balancing. Still, it fails to generalise and can successfully reach goal location only on 5 out of 16 scenes in our simulation benchmark.

The success-rate results are summarised in Table 4.2, and it is clear that our method shows robust performance and is able to generalise across the Building-99 and UrbanScene3D datasets. Table 4.1 shows the minimum distance of the camera from any building. Our algorithm consistently performs better than other algorithms and maintains the highest safe distance from the obstacles. This is also evident from Fig. 4.1 where we plot the trace of trajectories obtained using different algorithms.

Approaches	**UrbanScene3D**	**Building99**	**Total Success Rate**
Naive Flow-Balancing [38]	1/6	1/10	12.5%
Radial-Flow + Flow-Balancing	2/6	3/10	31.25%
Ours	**5/6***	**10/10***	**93.75%***

Table 4.2 Success Rate Comparison: Our method generalises across different scenes in the Building-99 and UrbanScene3D dataset to give a consistent controller performance.

4.2 Salient Object Detection as Collision Obstacle Detection

In this section, we evaluate and present our findings of using a Salient Object Detection method, in our case BASNet [33], to select obstacle(s) in/close to the line of attack of the drone. Selecting the right obstacle is critical to our avoidance method as velocity commands are generated based on desired flow obtained by combining radial flow with the obstacle mask. We train BASNet to learn to segment high-rise structures that pose as obstacles during flight, which is how we rethink saliency in the context of autonomous navigation.

We present some qualitative results of BASNet in detecting obstacles during flight. Fig. 4.2a)-d) show detection of obstacles in/close to the line of attack of the drone, except in Fig. 4.2d) owing to its small size and distance from the drone. However, it must be noted that this is not a collision scenario, and such behavior is actually desirable so the drone does not remain in a perpetual state of detecting and avoiding an obstacle and can continue moving on the original trajectory. Fig. 4.2a) and c) also show how imperfect/partial segmentation masks do not hinder downstream flow synthesis and subsequent avoidance, as the velocity computed from the desired flow still moves the drone away from the obstacle.

For selecting the Building of Concern in the presence of competing objects, we compare the performance of salient object detection and classical semantic segmentation. For this experiment, we choose BASNet and DeepLabv3[12], respectively. Both models are trained on our custom dataset, where the DeepLabv3 (ResNet50 backbone) is fine-tuned from COCO train2017 for 100 epochs and keeping *Obstacle* as the single-class label. Both networks learn fundamentally different tasks, where salient object detection focuses more on relevance in terms of location, proximity, and separation from the background, the semantic segmentation model, learns more structural properties hence providing cluttered

masks segmenting all object instances, which is clearly depicted in examples Fig. 4.3a)-d). Further, examples Fig. 4.3b) and d) show results that even when the buildings have relative depth, BASNet is able to detect the collision building correctly.

4.3 Stereo Depth based Avoidance

We compare the performance of our obstacle detection and avoidance method with a stereo depth based avoidance method. We use Fast-Planner[41] a kinodynamic path searching method using a 3D point cloud as perception information on a UAV in a simulated environment in Gazebo and approaching a building 30m wide and 60m tall. This method works on voxelised occupancy mapping for environment representation, making it agnostic to the scene/object texture. Fig. 4.4 shows the experiment setup, obstacle occupancy map, and UAV trajectory. We highlight the result in terms of obstacle detection range and subsequent failure for avoidance. Stereo range based depth cameras work reliably in the range of 7-10m, which is essentially how close the UAV gets to the building before seeing it as an obstacle. At this point, for a typically wide building, it becomes unfeasible to view around the sides and plan any path for avoidance, resulting in the drone halting in its track with no further course of action other than the nondeductive exploration of a vast structure to find a safe passage. This becomes inefficient for UAVs by wasting limited flight endurance.

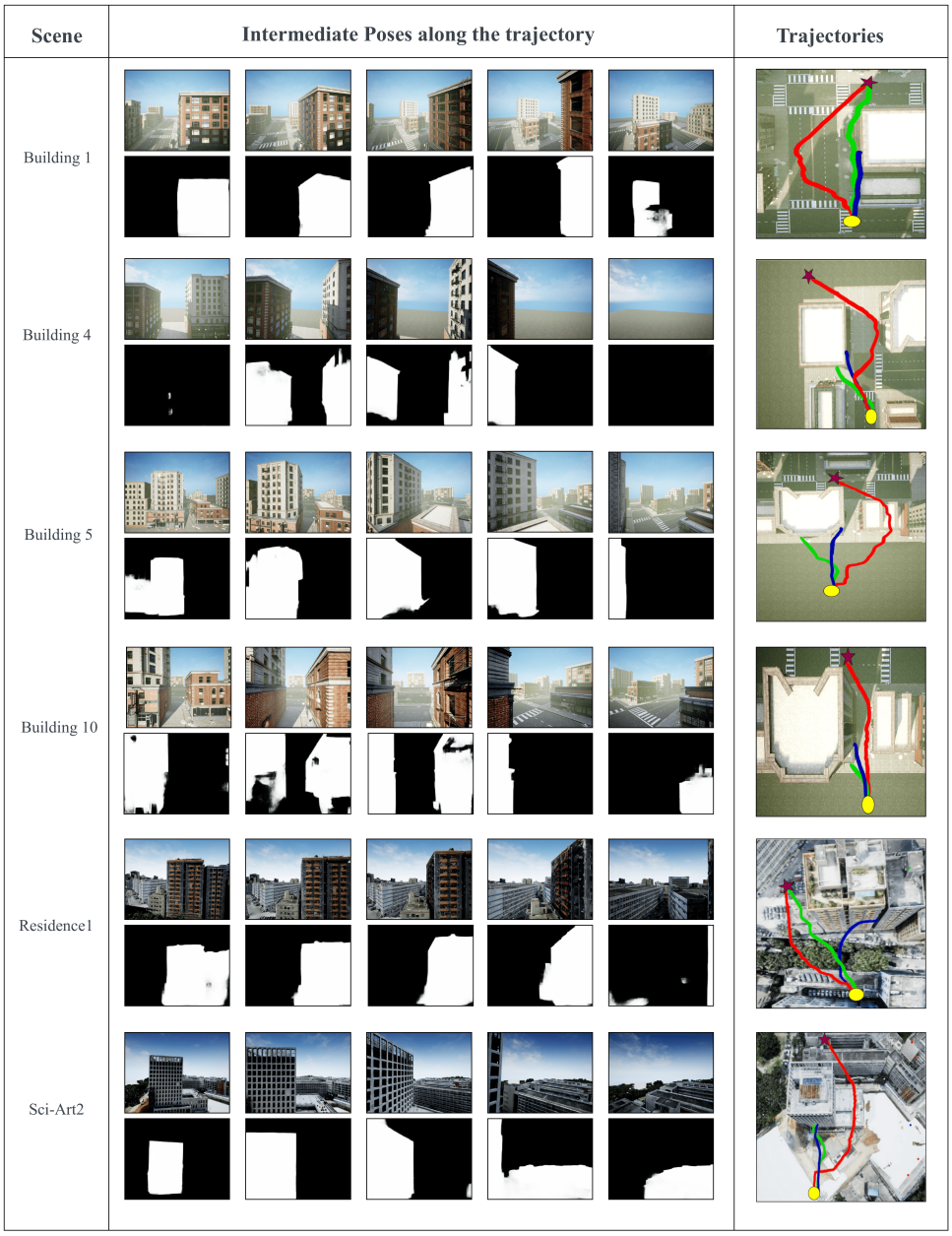

Figure 4.1 Qualitative results on the benchmark for selected scenes: Our method successfully avoids the obstacles on all the 10 scenes in the building 99 environment and 5 of 6 in the UrbanScene3D environments. Here we show images for 6 intermittent poses captured during the obstacle avoidance for selected scenes in the simulation benchmark. In Building 1, we are able to segment and avoid the building even though there are several buildings behind it. Building 4 covers a large part of the image, but our algorithm is able to move in the correct direction. In Building 10, the MAV can navigate the narrow path between the two buildings successfully. We also present results on certain challenging configurations from the real-world dataset UrbanScene3D. The lines in the trajectory column indicate the following: (RED) –> Our Method, (BLUE) –> Flow Balancing, (GREEN) –> Flow Balancing with Radial Flow. The goal and start positions are marked with a red star and a yellow circle, respectively.

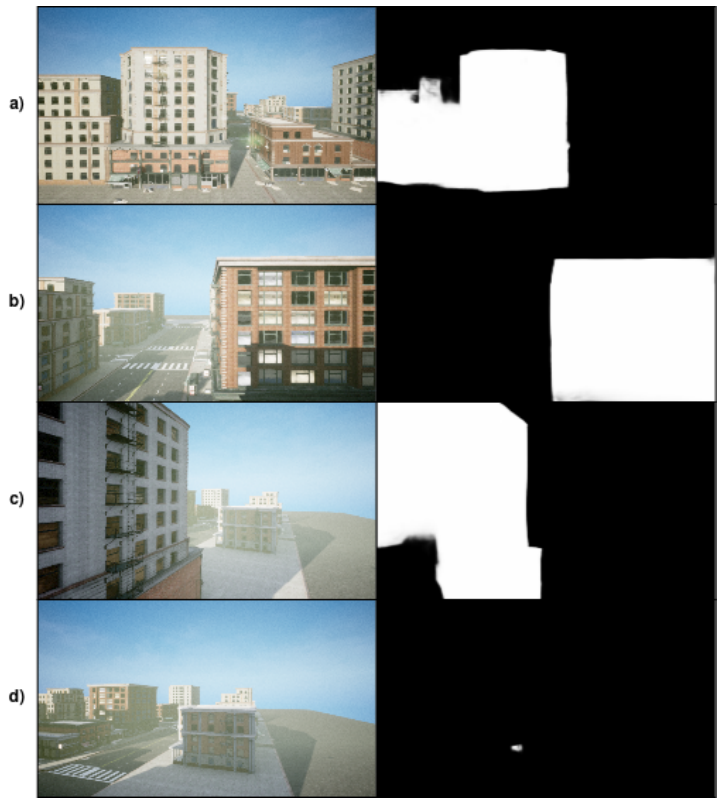

Figure 4.2 Obstacle segmentation masks for scenes in the AirSim Building 99 environment computed from BASNet

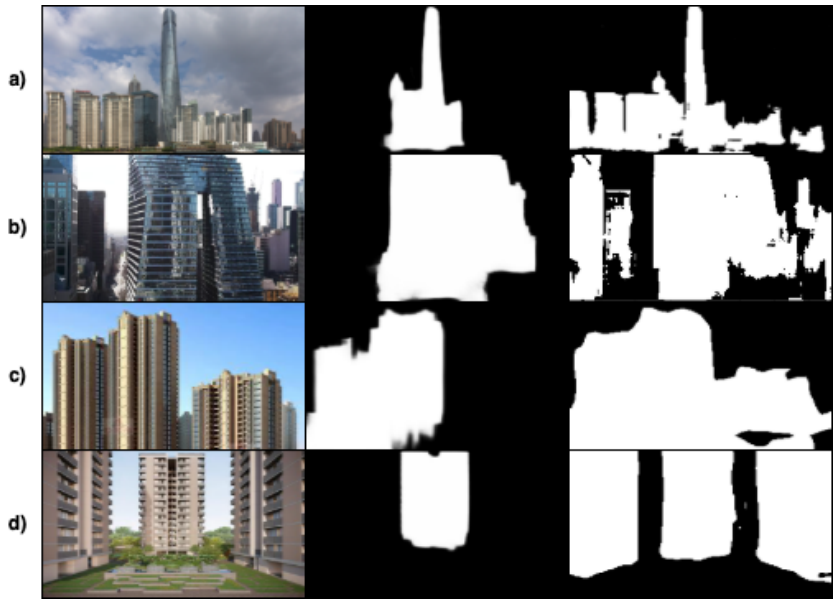

Figure 4.3 ***From Left to Right*****: Input Scene Image, Salient Object Detection (BASNet), Semantic Segmentation (DeepLabv3).** The 4 examples shown are randomly chosen real-world images of typically encountered scenarios in urban air mobility and are not sampled from our dataset.

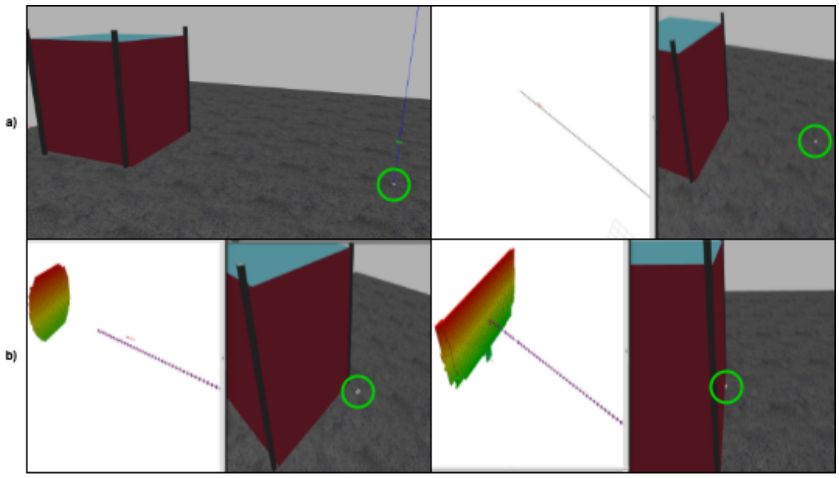

Figure 4.4 Evaluation of stereo depth avoidance for a large structure. The UAV is enclosed in a white bounding box, position highlighted with a green circle. **a)** shows the simulation setup in Gazebo consisting of a building with a 30mx30m front and an Iris drone approaching it from 30m distance and 10m altitude. **b)** shows that the detection of a building (voxel map) starts when the UAV is 9m away from the front and ends up being 1m close till the whole front is detected. The planner fails to find a trajectory to avoid the obstacle; subsequently, the UAV is halted.

Chapter 5

Conclusion

This thesis proposes a robust and interpretable monocular obstacle avoidance framework for a UAV navigating an outdoor urban environment. The framework leverages saliency-based segmentation to identify the Building of Concern. The obtained segmentation mask is combined with the proposed radial flow to get the desired flow, which can be fed into the high precision flow-based visual servoing methods to avoid the obstacles successfully. Conclusively, this work presents extensive experimentation and comparative studies with baseline methods to establish the merit of this approach. This work shows how a visual servoing framework and a saliency-based obstacle detection can be combined to avoid obstacles robustly. Hopefully, this servoing based avoidance pipeline inspires further investigation into this promising direction. This saliency and visual servoing-based framework can be enhanced in future work to avoid dynamic obstacles at a higher velocity.

Chapter 6

Outdoor Passenger Carrying Drone

6.1 Objectives

Towards the development of the passenger carrying drone, the team here at the Robotics Research Center of IIIT Hyderabad has been actively working in developing systems and modules for localisation, state estimation, planning and mapping using RGB-D cameras and other low-cost sensors. We have also been actively building drone prototypes to demonstrate and evaluate these modules in outdoor environments. We have developed systems GPS based localization as well as fusion of Inertial Estimates with Vision based sensing. We showcased real time obstacle avoidance with two state of the art planners, Fast Planner and EWOK on real drones avoiding vegetation and small buildings. Further we proposed a novel uncertainty based trajectory planner, CCO-VOXEL and showcased it to be more robust than the State of the Art Fast Planner by the virtue of its ability to handle uncertainty and noise in depth data from the RGBD sensor. The CCO-VOXEL was successfully integrated with the drone hardware and implemented in real time avoiding vegetation and other obstacles. We also developed a novel Flow Guided Visual servoing framework for obstacle avoidance and showed avoidance of high rises in simulations.

6.2 Drone Prototypes

For live deployment of our developed systems, several Multirotors with various sensing modalities were developed both in-house and contractually.

1. Lucifer, Fig.6.1, is a heavy payload carrying drone which was contractually built through Thanos Pvt Ltd., a custom drone manufacturer in Hyderabad. Lucifer is primarily used for outdoor navigation, and has sensors and compute capability sufficient enough to run our obstacle avoidance system in real-time. Table 6.1 contains the bill of materials.

2. Phoenix, Fig.6.2, is a lightweight quadrotor built in our lab for indoor testing of autonomous navigation using low-cost low-power sensor systems. Table 6.2 contains the bill of materials.

.

Figure 6.1 Lucifer is an advanced quadrotor at RRC (custom built by THANOS Pvt Ltd.) running the latest PX4 firmware. It is our premier ROS-enabled drone capable of running Vision, Mapping, State Estimation, and SLAM algorithms online on an Intel-i7 NUC Board installed as a companion computer. We use MAVROS and MAVSDK for interfacing the flight controller.

6.3 State Estimation Systems

State estimation is a critical task for a robot operating in any environment. For a UAV, state estimation is the process of determining essential parameters, like, but not limited to, position, orientation, velocity, altitude, in three-dimensional space based on data from sensors like accelerometers, gyroscopes, barometers and GPS. State estimation is an important aspect of the control and navigation of aerial robots, as it allows the robot to accurately determine its own position and orientation, as well as the position and orientation of objects in its environment. This information is used to control the robot's motion and achieve its desired goals, such as following a specific path or avoiding obstacles. We have developed and tested multiple state estimation methods on our drone prototypes which consume different sensor modalities and bench-marked their performance. The methods tested are listed below -

1. GPS Only

COMPONENT	MODEL
Flight Controller + GPS	PixHawk 4 (Holybro) with Neo M8N GPS
Companion Computer	Intel NUC8i7BEH
Camera - 1 (Depth)	Intel Realsense D415
Camera - 2 (VINS)	mvBlueFOX 200wC
IMU (VINS)	MTI-30-2A5G4
Motors	Gartt ML 5010 300kv
ESC	Readytosky 80A
RC	Taranis X9D Plus
Telemetry	XRock

Table 6.1 Bill of Materials for Lucifer drone

COMPONENT	MODEL
Flight Controller	Cube Orange
Companion Computer	Raspberry Pi 4
Camera - 1 (State Estimation)	Realsense T265
Camera - 2 (Depth)	Realsense D435
GPS	HERE 2
Frame	s500
Motors	EMAX MT3515 650kv
ESC	DYS SimonK
RC	FlySky FS-i6
Telemetry	Generic
Propellors	Orange 1238

Table 6.2 Bill of Materials for Phoenix drone

2. GPS + IMU (EKF Fusion)

3. RGB Camera + IMU (VINS Mono)

4. RGB Camera + IMU + GPS (VINS Fusion)

Pure GPS-based state estimation methods are naive techniques for tracking drone position outdoors; however, they are susceptible to inaccuracies in cloudy weather and when the UAV is flying in dense environments such as among high-rise buildings. IMU-based positioning is independent of the environment as they respond purely to the system's motion. They perform robustly under fast motions, and state updates are available at a higher frequency. However, they suffer from a high amount of drift that accumulates over time. Hence IMU and GPS data are fused to produce robust state estimates. The fusion technique, which is also the current gold standard for UAV state estimation, is called Extended Kalman Filtering. Autopilot software, by default, use EKF for state estimation. Recently, newer optimization-based techniques have emerged to be quite robust, especially Visual-Inertial navigation systems, which use both image and IMU data and optimize for state estimates as a tightly coupled system. These meth-

Figure 6.2 Phoenix is an advanced quadrotor at RRC running the latest ArduPilot firmware. The Phoenix platform has been developed for testing high-speed camera only applications. The interface is non-ROS. We use pymavlink to communicate with the FC and other standard libraries for development.

ods can also consume GPS data to achieve global fusion. We have also built our own custom VINS sensor setup, Fig. 6.3, and have deployed it on our drone prototypes.

6.4 Mapping and Planning Systems

A mapping system for a UAV allows it create a map of its environment while its in flight. This representation of the environment is important for a variety of purposes such as navigation, localization, planning and obstacle avoidance. There are several different types of mapping systems that can be deployed based on the type of sensor information available on the UAV, each with their own unique characteristics and capabilities. Some common sensor mapping systems include-

1. **Lidar**: A Lidar (light detection and ranging) system uses lasers to directly measure the distance to objects in the environment, based on Time-of-Flight(ToF). Lidar systems can generate long-range high-resolution 3D maps of the environment in real-time, however are power and compute intensive.

Figure 6.3 Custom built setup for Visual-Inertial state estimation systems, which consists of a BlueFox Monocular Global Shutter Camera and an XSens Industry-Grade IMU. The camera is hardware triggered by the IMU to provide millisecond-level synchronization and prevent long-term drift. GPS data is used separately for trajectory refinement through global fusion.

2. **Stereo vision**: Stereo vision systems use two or more cameras to capture images of the environment from different viewpoints and calculating depth of objects in the scene by triangulation. Using this information, the system can generate a 3D map of the environment.

3. **RGB-D cameras**: RGB-D cameras are cameras that capture both color and depth information. Depth and color information combined can be used to generate dense and semantic maps of the environment.

A planning system for a UAV enables the vehicle to generate a plan or trajectory for achieving a specific goal or task. In the context of autonomous navigation, it would mean reaching a goal location safely without any collision. This plan considers the UAV's current state, capabilities, and any constraints or limitations it may face while navigating through the environment. Motion planning algorithms are used to generate paths and/or trajectories for the UAV to follow while avoiding obstacles and satisfying any kinematic or dynamic constraints. These algorithms can be based on techniques such as path planning, motion primitives, or sampling-based approaches.

In building our outdoor drone navigation stack, we have tested and deployed combinations of various mapping and planning methods, each having its own constraints and capabilities. For mapping, we have two systems ready, namely-

1. **RTAB-Map[27]** (Real-Time Appearance-Based Mapping) is an open-source visual SLAM (Simultaneous Localization and Mapping) library for mobile robots and augmented reality applications. It uses a combination of visual features and depth data from a camera or laser rangefinder to create a map of the environment and to simultaneously localize the robot within the map.

2. **OctoMap[19]** a 3D probabilistic occupancy grid mapping library for robotics. It represents the environment as a voxel grid (a 3D grid with each cell or voxel representing a small volume of space) and uses a probabilistic approach to estimate the occupancy of each voxel.

For planning, we have again two methods of computing safe trajectories, namely-

1. **RRT*[27]** (Rapidly-exploring Random Tree*) algorithm is a sampling-based motion planning algorithm that is used to find a path for a robot to follow between a start and goal location in a given environment. It is an extension of the RRT (Rapidly-exploring Random Tree) algorithm and is designed to find a high-quality, asymptotically optimal path in a reasonable amount of time. When using the RRT* algorithm on an OctoMap, the OctoMap is used to represent the environment in which the robot is operating. The algorithm works by sampling random points in the environment and connecting them to the nearest point in the tree using a motion model for the robot. It then repeats this process until a path to the goal is found or until the maximum number of iterations is reached.

2. **FastPlanner[41]** FastPlanner is a motion planning algorithm that was developed by the Hong Kong University of Science and Technology (HKUST) and is designed to find a collision-free path through a given environment in a fast and efficient manner. It is based on the A* (A-star) algorithm and uses a combination of heuristics and search techniques to quickly find a path that is close to optimal.

6.4.1 Offline Mapping and Planning

A map of the environment can be generated by collecting the 3D environment information and registering it spatially using the poses at which the individual measurements were taken. This gives a consistent map from which way-points can be extracted to create paths from a start point to a goal point. For this approach, we have demonstrated the usage of a RRT* planner. We have also implemented a faster and more robust method called the Fast-Planner. This method, also discussed further, will be extended to a more general, real-time obstacle avoidance. In this approach, the full OctoMap is converted to a Costmap in the form of a dynamic EDT which is faster to plan on. The way-points are extracted in a Local NED frame attached to the environment, and can be passed to the flight controller that executes the trajectory by flying the vehicle in off-board mode. This approach is very naive and limited in the sense it will not be able to handle dynamic changes to the environment (obstacles that were not recorded while the map was being constructed, or changed location). Fig.6.4 can be referred to understand the pipeline.

6.4.2 Online Mapping and Planning

This is a more generic approach for obstacle avoidance built on top of the previously described method. The key concept is to build and plan on the map simultaneously. For tractability, a fixed size

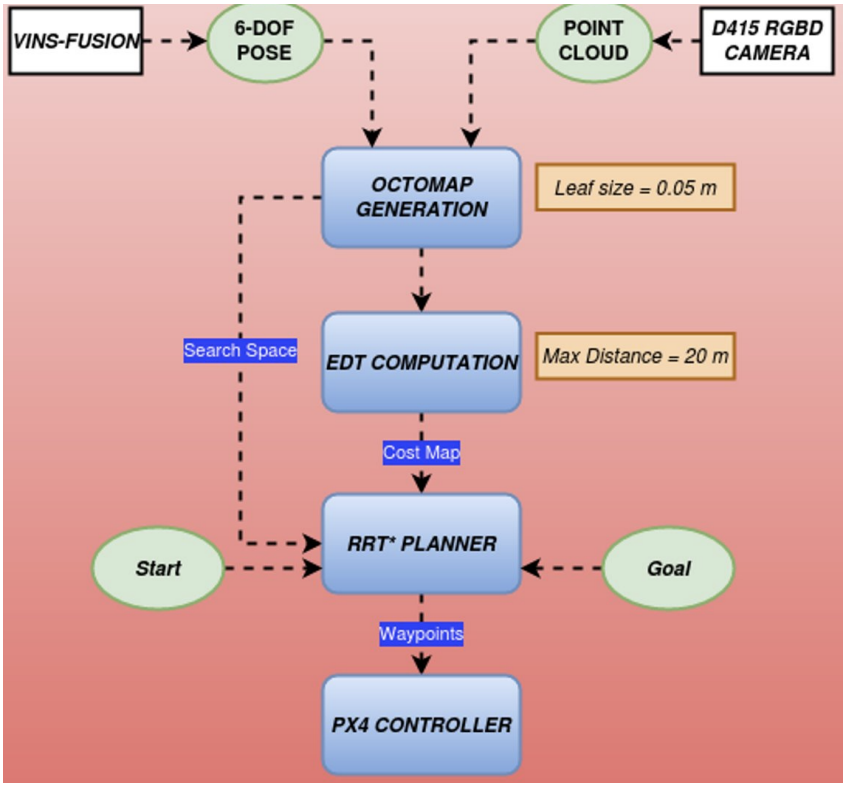

Figure 6.4 Offline Mapping and Planning system overview.

window of the environment around the drone is maintained in which waypoints are extracted. The map is expanded as the vehicle moves towards the goal point. The trajectory is re-planned as new obstacles keep coming up on the map. This type of approach is agnostic to the environment in which the drone flies, and also handles obstacles that appear dynamically. This method is also repeatable with the same configuration, as all maps are computed afresh with every flight. All sensor information and transformations are preserved from the offline mapping approach. We have successfully implemented and demonstrated this method on a quadrotor and achieved avoidance around trees, fences, and other common objects encountered in outdoor urban areas. This method, however, can be fragile when the drone moves at higher speeds as map generation and planning is resourcefully and computationally taxing. Fig.6.5 can be referred to understand the pipeline. Fig.6.6 shows a live demo of the UAV maneuvering over a fence to reach the goal location on the other side.

6.5 Obstacle Avoidance

All the above modules combine to become, what is called, the outdoor navigation stack for our UAV. This stack is enables the UAV for autonomous point-to-point navigation, maneuvering around obstacles in the path and reaching the final goal point within a reasonable time frame. Fig6.7 shows a successful

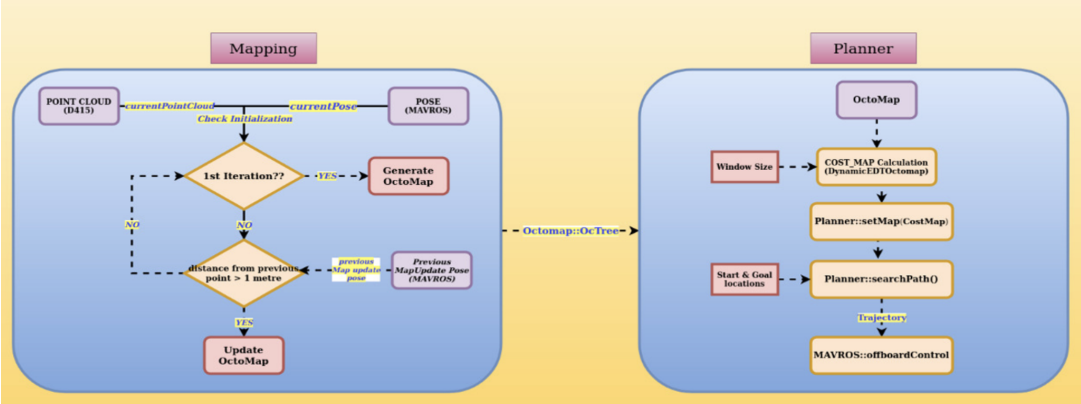

Figure 6.5 Online Mapping and Planning system overview.

demonstration of the UAV running the navigation stack, and maneuvering over an obstacle in its path, to reach the goal point.

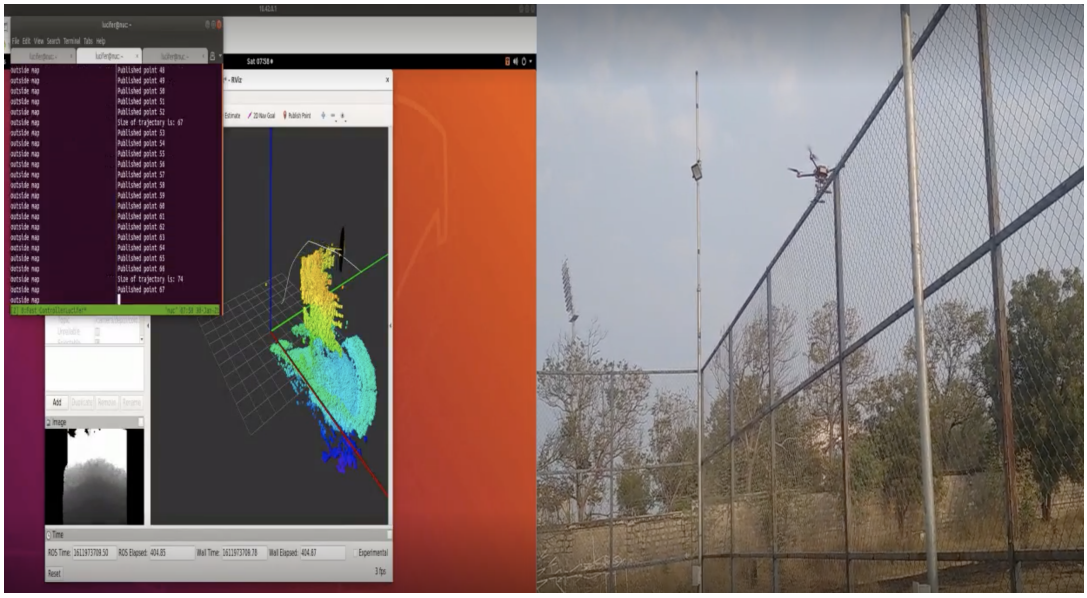

Figure 6.6 UAV, running the full online mapping and planning navigation stack, avoiding the fence of the lawn tennis court in the IIIT-H campus. The UAV trajectory can be understood as the trace provided in the left half of the image.

Figure 6.7 UAV maneuvering above the entrance of a building in the IIIT-H campus.

Chapter 7

Publications

7.1 Relevant Publications

1. **H. K. Sankhla** et al., *"Flow Synthesis Based Visual Servoing Frameworks for Monocular Obstacle Avoidance Amidst High-Rises"*, **2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)**, 2022, pp. 1339-1345, doi: 10.1109/CASE49997.2022.9926593.

7.2 Other Publications during M.S

1. Ayyappa Swamy Thatavarthy, Tanu Sharma, **H. K. Sankhla**, Mukul Khanna and K. Madhava Krishna, *"Multi-view Planarity Constraints for Skyline Estimation from UAV Images in City Scale Urban Environments"*, **16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2021) - Volume 5: VISAPP**, pp. 852-860, doi: DOI: 10.5220/0010208408520860.

Bibliography

[1] A. Al-Kaff, Q. Meng, D. Martín, A. de la Escalera, and J. M. Armingol. Monocular vision-based obstacle detection/avoidance for unmanned aerial vehicles. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 92–97, 2016.

[2] O. J. N. Bertrand, J. P. Lindemann, and M. Egelhaaf. A bio-inspired collision avoidance model based on spatial information derived from motion detectors leads to common routes. *PLOS Computational Biology*, 11(11):1–28, 11 2015.

[3] J. Borenstein and Y. Koren. Obstacle avoidance with ultrasonic sensors. *Robotics and Automation, IEEE Journal of*, 4:213 – 218, 05 1988.

[4] A. Borji, M.-M. Cheng, H. Jiang, and J. Li. Salient object detection: A benchmark. *IEEE transactions on image processing*, 24(12):5706–5722, 2015.

[5] P. Chakravarty, K. Kelchtermans, T. Roussel, S. Wellens, T. Tuytelaars, and L. Van Eycken. Cnn-based single image obstacle avoidance on a quadrotor. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6369–6374, 2017.

[6] G. Cho, J. Kim, and H. Oh. Vision-based obstacle avoidance strategies for mavs using optical flows in 3-d textured environments. *Sensors*, 19(11):2523, 2019.

[7] P. I. Corke. Visual control of robot manipulators–a review. In *Visual Servoing: Real-Time Control of Robot Manipulators Based on Visual Sensory Feedback*, pages 1–31. World Scientific, 1993.

[8] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.

[9] A. Elfes. Sonar based real-world mapping navigation. *Robotics and Automation, IEEE Journal of*, RA-3:249 – 265, 07 1987.

[10] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 834–849, Cham, 2014. Springer International Publishing.

[11] O. Esrafilian and H. D. Taghirad. Autonomous flight and obstacle avoidance of a quadrotor by monocular slam. In *2016 4th International Conference on Robotics and Mechatronics (ICROM)*, pages 240–245, 2016.

[12] L.-C. Florian and S. H. Adam. Rethinking atrous convolution for semantic image segmentation. In *Conference on Computer Vision and Pattern Recognition (CVPR). IEEE/CVF*, 2017.

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

[14] J. Guldner and V. Utkin. Sliding mode control for gradient tracking and robot navigation using artificial potential fields. *IEEE Transactions on Robotics and Automation*, 11(2):247–254, 1995.

[15] Y. V. S. Harish, H. Pandya, A. Gaud, S. Terupally, N. S. Shankar, and K. M. Krishna. Dfvs: Deep flow guided scene agnostic image based visual servoing. *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9000–9006, 2020.

[16] S. S. Harithas, R. D. Yadav, D. Singh, A. K. Singh, and K. M. Krishna. CCO-VOXEL: chance constrained optimization over uncertain voxel-grid representation for safe trajectory planning. *CoRR*, abs/2110.02904, 2021.

[17] P. Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[19] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 2013. Software available at `https://octomap.github.io`.

[20] J. Hu, Y. Niu, and Z. Wang. Obstacle avoidance methods for rotor uavs using realsense camera. In *2017 Chinese Automation Congress (CAC)*, pages 7151–7155. IEEE, 2017.

[21] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *Robotics and Automation, IEEE Transactions on*, 12:651 – 670, 11 1996.

[22] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox. Flownet 2.0: Evolution of optical flow estimation with deep networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1647–1655, 2017.

[23] M. Jagersand, O. Fuentes, and R. Nelson. Experimental evaluation of uncalibrated visual servoing for precision manipulation. In *Proceedings of ICRA*, volume 4, pages 2874–2880. IEEE, 1997.

[24] G. Kahn, A. Villaflor, V. Pong, P. Abbeel, and S. Levine. Uncertainty-aware reinforcement learning for collision avoidance. *CoRR*, abs/1702.01182, 2017.

[25] P. Katara, H. Y V S, H. Pandya, A. Gupta, A. Sanchawala, G. Kumar, B. Bhowmick, and M. K. Deepmpcvs: Deep model predictive control for visual servoing. *2020 Conference on Robot Learning (CoRL)*, 05 2021.

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

[27] M. Labbé and F. Michaud. Appearance-based loop closure detection for online large-scale and long-term operation. *IEEE Transactions on Robotics*, 29(3):734–745, 2013.

[28] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[29] A. Loquercio, A. I. Maqueda, C. R. del Blanco, and D. Scaramuzza. Dronet: Learning to fly by driving. *IEEE Robotics and Automation Letters*, 3(2):1088–1095, 2018.

[30] E. Malis, G. Chesi, and R. Cipolla. 212d visual servoing with respect to planar contours having complex and unknown shapes. *The International Journal of Robotics Research*, 22(10-11):841–853, 2003.

[31] K. McGuire, G. de Croon, C. De Wagter, K. Tuyls, and H. Kappen. Efficient optical flow and stereo vision for velocity estimation and obstacle avoidance on an autonomous pocket drone. *IEEE Robotics and Automation Letters*, 2(2):1070–1076, 2017.

[32] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto. Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning. pages 1366–1373, 09 2017.

[33] X. Qin, Z. Zhang, C. Huang, C. Gao, M. Dehghan, and M. Jagersand. Basnet: Boundary-aware salient object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[34] M. N. Qureshi, P. Katara, A. Gupta, H. Pandya, Y. V. S. Harish, A. Sanchawala, G. Kumar, B. Bhowmick, and K. M. Krishna. Rtvs: A lightweight differentiable mpc framework for real-time visual servoing. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3798–3805, 2021.

[35] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[36] S. Ross, N. Melik-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*, pages 1765–1772. IEEE, 2013.

[37] M. C. Santos, L. V. Santana, A. S. Brandao, and M. Sarcinelli-Filho. Uav obstacle avoidance using rgb-d system. In *2015 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 312–319. IEEE, 2015.

[38] K. SOUHILA and A. Karim. Optical flow based robot obstacle avoidance. *International Journal of Advanced Robotic Systems*, 4, 03 2007.

[39] L. Tai, S. Li, and M. Liu. A deep-network solution towards model-less obstacle avoidance. In *2016 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 2759–2764. IEEE, 2016.

[40] I. Ulrich and J. Borenstein. Vfh+: Reliable obstacle avoidance for fast mobile robots. In *Proceedings. 1998 IEEE international conference on robotics and automation (Cat. No. 98CH36146)*, volume 2, pages 1572–1577. IEEE, 1998.

[41] B. Zhou, F. Gao, L. Wang, C. Liu, and S. Shen. Robust and efficient quadrotor trajectory generation for fast autonomous flight. *IEEE Robotics and Automation Letters*, PP:1–1, 07 2019.

[42] 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)*, pages 3357–3364. IEEE, 2017.