2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

MinNav: Minimalist Navigation For Active Tiny Aerial Robots

Aniket Patil[1], Mandeep Singh[1], Uday Girish Maradana[1], Nitin J. Sanket[1,*]

Fig. 1. *MinNav* handles navigation in unstructured and wild scenes including static obstacles, dynamic obstacles and unknown shaped gaps without any prior knowledge of location or scene ordering. All this is performed using a monocular camera and an active strategy using only onboard computation and sensing. *All the images in this paper are best viewed in color on a computer screen at 200% zoom.*

***Abstract*— Navigation using a monocular camera is pivotal for autonomous operation on tiny aerial robots due to their perfect balance of versatility, cost and accuracy. In this paper, we introduce *MinNav*, a navigation stack based on optical flow and its uncertainty to fly through a scene with static and dynamic obstacles and unknown-shaped gaps without any prior knowledge of the scene components and/or their locations/ordering. We further improve success rate by using the activeness of the robot to move around in an exploratory way to find obstacles and navigate. We successfully evaluate and demonstrate the proposed approach in many real-world experiments in various environments with static and dynamic obstacles and unknown-shaped gaps with an overall success rate of 70%. To the best of our knowledge, this is the first solution to tackle all the aforementioned navigation cases without prior knowledge using a monocular camera. Our approach is on par in performance with depth based methods with factors of magnitude less computation required and can readily run onboard tiny aerial robots.**

Supplementary Material

The accompanying video, supplementary material, code and dataset can be found at https://pear.wpi.edu/research/minnav.html.

I. Introduction

Tiny aerial robots have risen in popularity in the last decade due to their utility in humanitarian applications such as plant pollination [1], [2], search and rescue [3]–[5] among others [6]. This rapid rise has been accelerated by the development of better computing hardware at lower power and size [7], [8]. While these robots offer advantages in agility, safety, and scalability, their autonomy remains limited by sensor and computational constraints. The use of Optical Flow rather than a 3D reconstruction of the scene has emerged as a more efficient representation, as remarked by experts in computer vision and robotics [9]–[12]. Using optical flow for navigation is minimalist in operation and works on image pixels rather than metric units for navigating a robot through a scene [2], [13]–[15]. Variants of optical flow are speculated to be used in nature, where bees perform peering and other maneuvers to dodge obstacles or go through a gap using functions of optical flow [16]. Computational algorithms inspired by nature have been used for performing various navigational tasks such as flying through gaps [13], dodging static and dynamic obstacles [17], landing [18] and so on. As some of the prior works point out, using only optical flow for navigation has a few problems: it has a dead spot near the focus of expansion where the flow magnitudes are low [19], flow can be ill-conditioned near color flat regions, large changes in illumination, around object boundaries and so on [14], [20]. Recently, works have proposed to utilize uncertainty in flow predictions [14] as a representation for navigation in various scenes, but this requires prior knowledge of the scene setup or the kind of scene that might be encountered (static or dynamic obstacles).

To this end, we present *MinNav*, which utilizes optical flow and its uncertainty to navigate through a scene that has static obstacles, dynamic obstacles and unknown gaps without any prior knowledge of the ordering of the scene components with only onboard sensing and computation. We utilize a neural network to obtain optical flow and uncertainty prediction onboard a tiny aerial robot. We further use the mathematical properties of our formulation to intuit how we can classify among various scene motifs (types of scene

[1]Perception and Autonomous Robotics (PeAR) Group, Robotics Engineering Department, Worcester Polytechnic Institute. *Corresponding author: Nitin J. Sanket (nitin@wpi.edu).

components). This classification enables a first-of-its-kind unified framework that jointly addresses all three motifs to generate efficient navigation commands, surpassing prior works that treat flow and uncertainty in isolation.

A. Problem Formulation and Contributions

A quadrotor is present in a scene that can have static obstacles, a static unknown-shaped gap and/or dynamic obstacles (called scene motifs). The quadrotor has no prior knowledge of which scene motifs are present and/or their ordering, size, shape or location. The robot is tasked to perform goal-directed navigation (such as heading north) using only a front-facing monocular camera. A down-facing optical flow sensor, laser altimeter, and inertial measurement unit are employed strictly for low-level attitude control, velocity estimation, and altitude regulation. All sensing and processing is done on board without the use of motion capture or GPS. We present an active approach to fly toward the free space while avoiding unknown static and dynamic obstacles. The core concept is built around utilizing optical flow and its uncertainty to find free space regions for navigation.A summary of our contributions are (Fig. 1):

- We propose a generalized navigation stack for a monocular camera-equipped robot to navigate through a scene with unknown static obstacles, gaps and dynamic obstacles. To the best of our knowledge, this is the first work that handles all such scenarios together (§II).
- We present a lightweight multi-scale optical flow and uncertainty network that can run onboard embedded computers (Fig. 2, §II-D).
- We propose an active control policy that combines exploratory motion and moving towards the free space in one formulation (§III).
- We evaluate and demonstrate the proposed approach on a real quadrotor with onboard perception and computation in many real-world experiments and simulation experiments (§IV). To benefit the community, we will release the source code and simulation environments used in this work.

B. Related Work

There are two main trains of thought when it comes to navigation through cluttered spaces using aerial robots.

1) Depth-based/Structured Navigation

In these classes of methods, structure or depth is either directly perceived by a sensor(s) or by reconstructing the environment. The most common way for achieving such navigation is by the use of commonly found LIDAR sensors [21], [22] for larger robots and RGBD sensors for smaller robots [23], [24]. Classically, a 3D representation is used on which paths and trajectories are planned and executed [23], [25], [26], but recent studies have shown that combining or replacing perception, planning and control using deep learning is more efficient and often more performant [27]–[31] either due to better policies learned or finding better policies through reinforcement learning. Although structured approaches seem like the gold-standard solution for navigation, there could be often cases when estimating depth using an active sensor might not be allowed or possible (such as direct sunlight or dusty conditions), or the sensor (or computation) might be too heavy to carry on a tiny robot. This necessitates a structureless approach to navigation. One can also imagine structureless approaches based on a monocular camera always running in the background and taking over during a sensor failure or low battery condition of a structured approach for larger robots.

2) Structureless Navigation

Structureless Navigation refers to methods that perform navigation without explicitly computing metric or relative depth (commonly called structure in computer vision literature). In these methods, either a function of depth is perceived that is directly used for control [32], [33], or lately, in tiny robots, motion is used to obtain additional information cues to infer about functions of depth [15], [33]. One such formulation utilizes optical flow (motion of pixels) to infer the ordinality of depth using the parallax effect (closer objects move more than farther objects). Newer works have also shown the efficacy of using optical flow uncertainty or occlusion masks to perform navigational tasks [14]. Although, in general, structureless approaches are sub-optimal in terms of path length as compared to structured approaches, they often use lesser computation power and lower-quality sensors making them perfectly suited for tiny robots.

In both cases, dynamic obstacles are handled very differently. In structured approaches, the simplest way is to treat the dynamic obstacle as a static obstacle and dodge it when it is closer than some distance. A better way is to model the 3D motion and predict the movement to dodge more efficiently. Structureless approaches either use a motion detection sensor such as an event camera to find dynamic obstacles efficiently [17] or optical flow coupled to Epipolar geometric constraints [34] or optical flow inlier probability (or uncertainty) to find dynamic obstacles [14], [35]. In the latter case, the obstacles are dodged in a best-effort manner.

To the best of our knowledge, no prior work handles all cases of navigation: dodging dynamic and static obstacles and flight through unknown gaps using a single structureless formulation without prior information. To this end, we present a novel method based on optical flow and its uncertainty to achieve navigation in a scene with static and dynamic obstacles and unknown gaps without any prior knowledge using only onboard sensing and computation using a monocular camera without reconstructing the scene.

C. Organization of the paper

The paper is structured into perception and control modules. We present our perception solution in §II where we explain how we distinguish between different scene cases (static obstacles, dynamic obstacles and gaps). After which we talk about our active perception-based control policy to enhance our perception stack while navigating towards the goal direction in §III. Experiments and analysis for both real-world and simulation settings are detailed in §IV and

finally, we conclude the paper in §V with parting thoughts on future work.

II. Scene Motifs For Visual Navigation

When navigating in the wild, an aerial robot can encounter various scene components such as static obstacles, dynamic obstacles and free space. In this work, we will call a collection of such scene components as *scene motifs*. Commonly, five scene motifs are found in both natural and human-made environments: (a) Free space with static obstacles, (b) Free space with dynamic obstacles, (c) Static Gaps, (d) Dynamic Gaps, (e) Dead-end, i.e., no free space. In this work, we propose to navigate through a scene in the first three scenarios where we do not have any prior knowledge of the scene statistics or geometry only using an on-board monocular camera with on-board computation. To enable such a general navigation stack without explicit 3D scene reconstruction, we utilize motion information between image frames in the form of optical flow and its uncertainty. Before we explain our approach further, we will explain a few preliminaries required.

A. Preliminaries

Let two image frames captured at t and $t+\delta t$ be denoted as $\mathcal{I}_t$ and $\mathcal{I}_{t+\delta t}$ respectively. Further, let the 3D linear and angular velocities of the camera in this time be $V = \begin{bmatrix} V_x & V_y & V_z \end{bmatrix}^T$, $\Omega = \begin{bmatrix} \Omega_x & \Omega_y & \Omega_z \end{bmatrix}^T$. Optical flow at a pixel $\mathbf{x} = \begin{bmatrix} x & y \end{bmatrix}^T$ is defined as the apparent image velocity of the corresponding 3D world point and is given by

$$\dot{\mathbf{p}}_{\mathbf{x}} = \frac{1}{Z_{\mathbf{x}}} \begin{bmatrix} xV_z - V_x \\ yV_z - V_y \end{bmatrix} + \begin{bmatrix} xy & -(1+x^2) & y \\ (1+y^2) & -xy & -x \end{bmatrix} \Omega \tag{1}$$

Here, $Z_{\mathbf{x}}$ denotes the depth at a pixel $\mathbf{x}$ which is commonly called the *structure*. Estimating 3D structure or reconstructing the scene is computationally expensive and slow for tiny robots. To navigate through a scene, we wish to distinguish between different motifs which would be simple if the structure was known. In this work, we utilize optical flow and its uncertainty to distinguish between various motifs and take the appropriate control action for navigation. Intuitively, closer objects (obstacles) produce high optical flow (particularly translational flow) by virtue of the low value of $Z_{\mathbf{x}}$. One might ponder, this is very simple then, why not just use the magnitude of optical flow for control action? There are two main issues (a) the rotational flow which is independent of the scene can overshadow the translational flow during fast rotational motion commonly found in aerial robots, and (b) the translational flow has a low magnitude near the Focus Of Expansion (FOE) which cannot be used for control action due to inherent high uncertainty and high noise sensitivity. A common solution for the first problem is either by using a mechanical or an electronic gimbal [19]. For the second problem, past works [19] avoid utilizing the circular area around the FOE for control. To tackle both issues, we propose to utilize the active nature of the robot, i.e., we can make controlled exploratory motions to reason about the scene to take a navigation action using the philosophy of Active Perception [12], [36]. By maintaining small rotational movements in our exploratory motion (cf. §III.), we obtain both low rotational flow and maintain FOE outside the image plane. Our perception stack for each motif is explained in the subsequent subsections.

B. Static Obstacles and Unknown Gaps

Since gaps are special cases of free space with obstacles, we treat these two motifs as the same entity. Technically, gaps are enclosed free spaces surrounded by obstacles. But for all practical purposes, we need to fly to the free space avoiding obstacles. In the case of static obstacles, if we have 'tackled' the FOE and high rotation issues as we discussed previously, then we can simply look for the largest contour (largest free space) of low flow magnitude and fly towards it. Contrary to previous works, our approach trades off perception complexity with planning and control complexity. Our perception stack is very simple due to carefully controlled exploratory active movements. The price to pay here is in terms of exploratory control action which is generally negligible for small agents as we see in nature [37], [38].

C. Dynamic Obstacles

Dynamic obstacles are defined as obstacles whose movement is independent of one's movement (ego-motion). This means that they will generate an optical flow field with a different FOE to the robot's own movement. Prior works have clustered optical flow based on FOE [34], but this is expensive because of the number of dynamic obstacles and other statistics are not known. One might ponder, can we not just look for high optical flow regions for dynamic obstacles? Conceptually, we want to dodge the obstacle only if it is coming toward us, in which case the optical flow magnitudes will be higher as compared to a static obstacle at the same depth $Z_{\mathbf{x}}$ because the relative velocities V, Ω will be higher. But this is not as simple as it sounds when we have both static and dynamic obstacles in the scene with similar optical flow magnitudes. Furthermore, to dodge dynamic obstacles, one has to dodge the 'future' projection not just the current detection since the object is moving. To this end, we will utilize optical flow uncertainty [14] to gain additional cues which we will use to distinguish static from dynamic obstacles with large flow magnitudes. In this work, since we do not know when we might encounter a dynamic obstacle, we utilize a combination of high flow magnitude and high uncertainty to detect dynamic obstacles. High uncertainty is encountered for dynamic obstacles due to large amounts of occlusion presented when objects move.

D. Network Architecture and Training Details

Our network architecture is inspired from previous works [14], [39], [40]. We perform the following modifications: (a) Multiscale pyramidal architecture for predicting optical flow at different levels in a coarse to fine manner, (b) Our multiscale architecture differs from the traditional

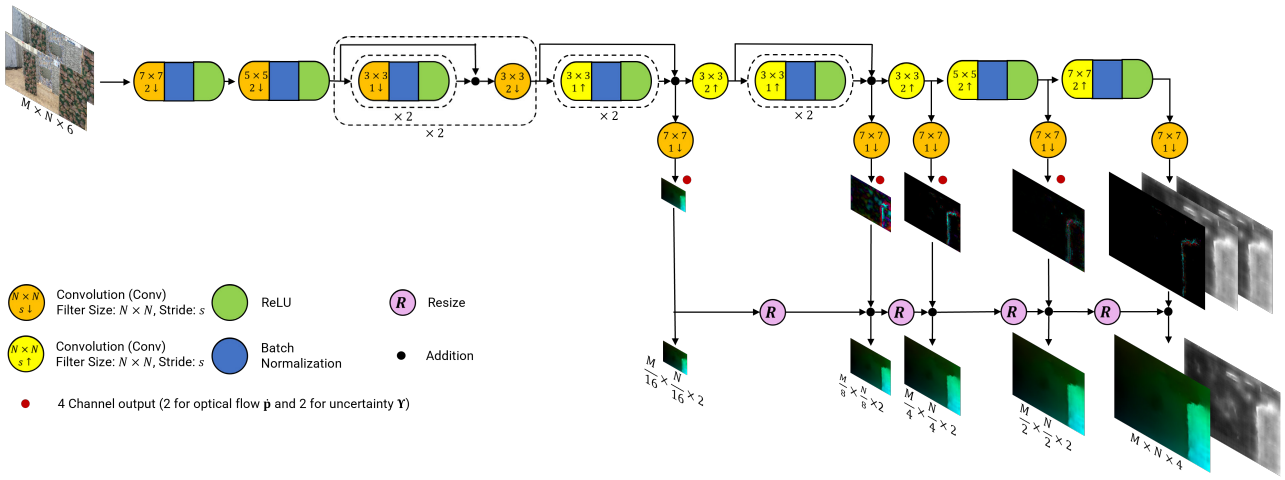

Fig. 2. Lightweight multi-scale pyramidal neural network architecture used to predict optical flow $\dot{\mathbf{p}}$ and its uncertainty $\boldsymbol{\Upsilon}$ used in *MinNav*.

approaches, such as in [41], by only predicting incremental optical flow for an efficient design. For implementation we use five levels ($L = 5$) and the architecture is shown in Fig. 2. Our network is a fully convolutional architecture and can take any arbitrary input size of $M \times N$ and output the same size. Specifically, the input is a stack of temporal images of size $M \times N \times 6$ and the output is optical flow $\dot{\mathbf{p}}$ and its uncertainty $\boldsymbol{\Upsilon}$ with a final output size of $M \times N \times 4$ (including x and y directions).

Our network was trained for 400 epochs on the FlyingChairs2 (FC2) dataset [42], [43] and then fine-tuned for another 50 epochs on the FlyingThings3D (FT3D) dataset [44] for better generalization to the real world. Our *MinNav* network has 2.8M parameters which is small enough to run on-board the Jetson Orin Nano with a 68.2ms forward inference time without any optimizations such as TensorRT on $1\times640\times480\times6px.$ resolution (24.6ms for $1\times320\times240\times 6px.$). The network is trained using a custom multiscale loss function (Eqs. 2, 3) with ADAM optimizer and a learning rate of 10^{-4} for the first 400 epochs (FC2) and then 10^{-5} for the last 50 epochs (FT3D) with a mini-batch size of 32.

$$\mathcal{L} = \underset{\dot{\mathbf{p}}_{\mathbf{x}}, \boldsymbol{\Upsilon}_{\mathbf{x}}}{\operatorname{argmin}} \sum_{l=1}^{L} \left(\mathbb{E}_{\mathbf{x}} \left(\frac{\| \dot{\mathbf{p}}_{\mathbf{x},l} - \mathcal{R}_L^l \left(\dot{\mathbf{q}}_{\mathbf{x}} \right) \|_1}{\log \left(1 + e^{(\boldsymbol{\Upsilon}_{l,\mathbf{x}} + \epsilon)} \right)} + \log \left(1 + e^{\boldsymbol{\Upsilon}_{l,\mathbf{x}}} \right) \right) \right) \tag{2}$$

$$\dot{\mathbf{p}}_{\mathbf{x}} = \sum_{l=2}^{L} \left(\mathcal{R}_{l-1}^{l} \left(\dot{\mathbf{p}}_{\mathbf{x},l-1} \right) + \Delta \dot{\mathbf{p}}_{\mathbf{x},l} \right); \quad \dot{\mathbf{p}}_{\mathbf{x},1} = \Delta \dot{\mathbf{p}}_{\mathbf{x},1} \tag{3}$$

Here, $\dot{\mathbf{p}}_{\mathbf{x}}, \dot{\mathbf{q}}_{\mathbf{x}}, \boldsymbol{\Upsilon}_{\mathbf{x}}$ are the predicted optical flow, ground truth optical flow and predicted uncertainty respectively. $\Delta \dot{\mathbf{p}}_{l,\mathbf{x}}$ denotes the incremental flow prediction at level l and $\mathcal{R}_{l-1}^{l} \left(\dot{\mathbf{p}}_{l-1,\mathbf{x}} \right)$ represents a differentiable bilinear resizing function that resizes optical flow from size at level $l-1$ to l and ϵ is a small constant for numerical stability (we use 10^{-3} in our experiments). Finally, L is the number of levels which is set to 5 in our experiments. At the heart of the perception stack is the exploratory motion that is imbibed in the control stack as described next.

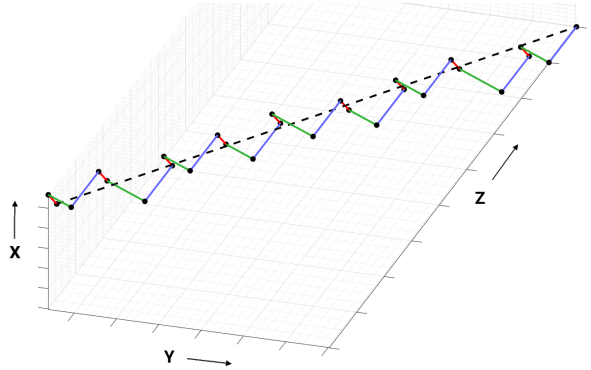

Fig. 3. A toy example showing how our control strategy will "wander" around the perfect line the robot would have followed if metric depth was available (dashed black line, time increasing with increasing Z). The red parts show tiny active exploration motion, green parts show free space alignment motion and blue parts show the forward Z motion.

III. Active Perception Based Control Strategy

The core idea behind our control strategy involves three key steps: (a) a diagonal exploratory motion (in X and Y direction), (b) a movement towards free space (in X and Y direction) and (c) a movement forward towards the global goal direction (Z axis). The concept of exploratory motion is fundamentally inspired by concepts of Active Perception [12], [13], [15], [36] and Honey Bee Peering [37], [38], [45] as explained next. See Fig. 3 for a toy example.

A. Static Obstacle Avoidance And Flight Through Unknown Gaps

We utilize the above strategy to avoid static obstacles and also fly through static unknown gaps by iteratively moving toward the free space region.

Active Exploratory Motion: As we mentioned before, the goal of the exploratory motion is to avoid the issues that arise around FOE. To this end, we employ a diagonal motion in the 3D X-Y plane. The exploratory motion is implemented using an open-loop position controller with a pre-chosen period and velocity profile for the diagonal motion. Let τ be the period of diagonal motion, P^E be the open-loop exploratory positional command, $V = \begin{bmatrix} V_x & V_y & V_z \end{bmatrix}^T$ be the chosen

velocity, then the commanded motion is given by:

$$P^E = \begin{bmatrix} P_x^E & P_y^E & P_z^E \end{bmatrix}^T = \begin{bmatrix} {V_x}^E \tau & {V_y}^E \tau & 0 \end{bmatrix}^T \quad (4)$$

Note that, we alternate the sign of both V_x^E and V_y^E at every iteration to have minimal drift.

Alignment Movement Towards Free Space: Since our goal is to keep moving towards free space, once we obtain image frames after exploration, we take a step towards the free space using a simple velocity controller. Let $\mathbf{x}_0 = \begin{bmatrix} x_0 & y_0 & 1 \end{bmatrix}^T$ and $\mathbf{x}_\mathcal{F} = \begin{bmatrix} x_\mathcal{F} & y_\mathcal{F} & 1 \end{bmatrix}^T$ be the center of the image and the center of the free space region on the image plane, respectively. We implement a alignment open-loop incremental position command $P^\mathcal{F}$ using a velocity controller and a fixed time period as before.

$$P^\mathcal{F} = \begin{bmatrix} P_x^\mathcal{F} & P_y^\mathcal{F} & P_z^\mathcal{F} \end{bmatrix}^T = \begin{bmatrix} K_{p,x}\,(x_0 - x_\mathcal{F})\,\tau & K_{p,y}\,(y_0 - y_\mathcal{F})\,\tau & 0 \end{bmatrix}^T \quad (5)$$

Here, $K_{p,x}, K_{p,y} > 0$ are the user-defined proportional gains of the controller.

Movement Towards Global Goal Direction: In the last step, we align our heading direction (angle associated with Y axis) to orient ourselves towards the global goal direction using a proportional integrative derivative controller. After aligning with the goal direction, we move forward (in the Z direction) for τ seconds using an open-loop positional command like before using a velocity controller.

Finally, all the aforementioned steps which we call an iteration are repeated until we have reached the goal.

Furthermore, the pre-chosen exploratory motion is generally much smaller in magnitude compared to the movement toward the free space. This formulation ensures that we do not spend too much time exploring and the overall path length is still relatively short as compared to having complete knowledge of depth. This is similar to the idea used in [13], [15].

B. High Priority Dynamic Obstacle Dodging

Often in the wild, robots will encounter dynamic obstacles such as falling leaves, branches, or rocks in a disaster scenario. While navigating through a cluttered static scene, one has to prioritize dodging dynamic obstacles to avoid a severe catastrophe. To this end, our dynamic obstacle detection and control algorithm are always running in parallel to the static obstacle avoidance methodology as a separate thread. We detect dynamic obstacles by looking for a large magnitude of optical flow and uncertainty for consecutively two frames. Whenever a dynamic obstacle is detected, an interrupt is raised to invoke a dodging maneuver using an acceleration command in the direction as given in [17]. The choice for the acceleration command is to minimize the latency and utilize the maximum agility of the robot.

IV. Experiments

The robot used in the experiments is a custom-built platform called Corgi210, and has a $210mm$ diagonal wheelbase. All the lower-level control algorithms are run on the Holybro Pixhawk 32 V6 Flight Controller using ArduCopter version 4.5.0-Dev. The ArkFlow optical flow sensor is used for enabling Loiter. All the higher-level perception, decision making and control commands are computed onboard the NVIDIA Jetson Orin Nano running Ubuntu 20.04.6 LTS with JetPack 5.1.2 and TensorFlow 2.12.0. The higher-level control commands are sent to the flight controller using Guided mode. The perception stack takes input from the onboard Arducam OV9281 120fps global shutter camera at a resolution of $640 \times 480px$.

A. Experimental Results

1) Evaluation Metrics

We adapt *Success Rate*, *Path Length Increase* over metric depth, and *Run Time* evaluation metrics from [14]. We perform exhaustive quantitative evaluation both in the real world and in simulation environments under various scenarios.

2) Real World Experiments

We test our *MinNav* framework in the Washburn flying space which is a netted facility of dimension $11 \times 4.5 \times 3.65m$. We construct four different scenarios with a variety of static and dynamic obstacles and an unknown-shaped gap (Fig. 4). The static obstacles are made of cuboidal cardboard boxes of sizes ranging from $1.15-1.28m$ with rock and moss textures stuck on them. The dynamic obstacles are made of an empty milk jug and a cardboard box of sizes ranging from $0.25 - 0.30m$ with the same textures as the static obstacles. These dynamic obstacles are swung (at a maximum speed of $3ms^{-1}$) using a thread towards the quadrotor. We utilize two arbitrarily shaped gaps that are made of foamcore with stone textured paper on it. The gap has an antipodal minimum and a maximum distance of $0.5m$ and $0.85m$ respectively giving us an average minimum clearance of $0.17m$.

In the four scenarios, the position and orientation of the static obstacles, gap shape and direction and velocity of the dynamic obstacles are randomized. Furthermore, we also randomize the ordering of the scene components (gap, static obstacles and dynamic obstacles). Specifically, **Scenario 1:** Randomized Static Obstacles $\rightarrow$ Gap Type 1 $\rightarrow$ Box dynamic obstacle, **Scenario 2:** Gap Type 1 $\rightarrow$ Randomized Static Obstacles $\rightarrow$ Milk carton dynamic obstacle, **Scenario 3:** Gap Type 2 $\rightarrow$ Randomized Static Obstacles $\rightarrow$ Box dynamic obstacle, **Scenario 4:** Randomized Static Obstacles $\rightarrow$ Gap Type 2$\rightarrow$ Milk carton dynamic obstacle.

We performed 20 trials on each scenario. Our method successfully could navigate through complex scenarios 56/80 times leading to an average success rate of 70%. Although this is not sufficient for ready deployment into the wild, we believe it is a first step in the direction of minimal autonomy in complex scenes in the wild. Furthermore, as the robot becomes smaller, its collision probability decreases because a reduced physical scale reduces its spatial footprint and improves operation in cluttered environments [46]. This trend also motivates the development of insect-inspired cognitive systems, such as those inspired by bees [37].

Discussion: A majority of our failure cases are in scenarios when we are trying to dodge dynamic obstacles, as we hit the nets on the side of the robot due to a lack of omnidirectional sensing. Although, we did not explore

Fig. 4. Sequence of images of quadrotor navigating through different scenarios. Green and red arrow shows the path of the robot and the dynamic obstacle respectively.

better data generation regimes as described in [47], [48], we believe they can significantly improve the optical flow quality and would be a great direction for future work. We also note that our latency for dynamic obstacle detection is about $100ms$ including our processing pipeline, which is fairly large for fast and close obstacles. To this end, we plan to explore TensorRT accelerations as a future direction along with better software engineering in C++ to lower latency. Lastly, we aim to accelerate our optical flow predictions by utilizing methods from [7], which will lower latency and improve dynamic obstacle response further.

B. Simulation Experiments

For a comprehensive quantitative evaluation of our approach, we designed a custom simulation environment in Blender® based on [17], [49]. The static components of the environment are represented by randomized trees depicting the challenges posed by natural obstacles. Dynamic obstacles were introduced in the form of randomly thrown soccer balls. The arbitrary-shaped gap was constructed using a planar surface with a randomized rock texture on it. The floor is simulated with a planar texture with bumps on it to mimic a real-world outdoor floor. We further randomize the ordering and location of the static objects and gaps and velocity + location of the dynamic obstacles. This deliberate variability in our generated scenes allowed us to conduct a comprehensive evaluation of *MinNav's* robustness under different conditions. Upon acceptance of this publication, we will release these scenes and code with an open-source license to accelerate further research.

We test our approach in 100 randomly simulated scenarios across various metrics and compare it with state-of-the-art approaches. In particular, we evaluate performance when the robot had access to different amounts of information, i.e., metric depth to relative monocular depth to optical flow (implicit function of metric depth).

For depth-based navigation, we use a simple strategy wherein we perform a potential field-based obstacle avoidance in a receding horizon manner to move towards the closest free space as described in [50]. This has the highest amount of information on the robot and results in the shortest path length, hence we measure all other path lengths as an increase (inefficiency) over the depth-based method. For the relative monocular depth method, we utilize a similar strategy as before but the robot parameters have to be more conservative due to a lack of environment knowledge. Finally, for optical flow-based methods, we utilize the *MinNav* control algorithm on RAFT optical flow as well as our network. Furthermore, we also test our approach with and without exploratory motion to show the utility of active vision under a dearth of high-quality computation and sensing. The results are tabulated in Table I and an example path output is shown in Fig. 5.

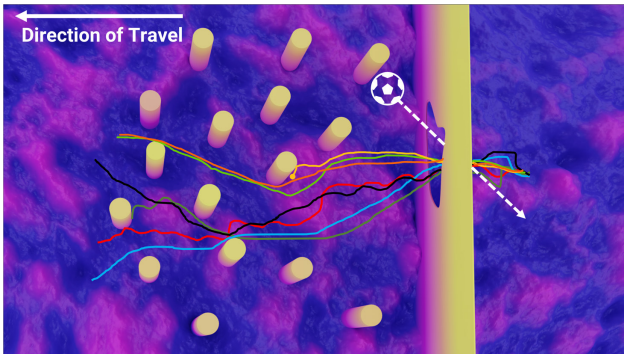

Fig. 5. Comparison of various navigation methods: Ground truth depth, MiDaS-v3.1, MiDaS-v2.1S, RAFT without exploration, **RAFT with exploration**, *MinNav* without exploration, *MinNav* (Ours) with exploration. The dashed white line shows the dynamic obstacle path. See Table I for quantitative evaluation.

Discussion: As one would expect metric depth-based methods such as MorphEyes perform the best overall but require careful tuning of depth threshold parameters for dynamic obstacles which are obvious due to the lack of explicit motion information and can be the go-to choice for navigation if a depth sensor can be placed onboard the robot when coupled with motion information for dynamic obstacles (such as scene flow). Similarly, relative monocular

TABLE I

QUANTITATIVE EVALUATION FOR SIMULATION EXPERIMENTS.

Method	SR (%)↑	Path Length Inc. (%) ↓	Run Time† (ms) ↓	FLOPs (G) ↓	Num. Params (M) ↓
MorphEyes [50]	100	–	9.8	–	–
MiDAS-v2.1S [53]	38	17.16	99.8	43.7	21
MiDAS-v3.1 [51]	**88**	**4.95**	2915.2	1052.9	344.6
RAFT* [54]	78	36.52	615.3	211.01	5.2
RAFT [54]	84	32.91	615.3	211.01	5.2
MinNav (Ours)*	57	57.84	**68.2**	**27.14**	**2.8**
MinNav (Ours)	82	52.49	**68.2**	**27.14**	**2.8**

*Without Exploration. †On NVIDIA Jetson Orin Nano at an input size of $1 \times 640 \times 480 \times 6px$.

depth-based methods are performant but often require a much higher amount of computation to predict high-fidelity depth. This is very evident from the high performance of the MiDAS-v3.1 model [51], [52] which is almost 2 factors of magnitude more expensive to compute than our model. Furthermore, it is important to observe that distilling a monocular depth model is not enough due to a massive loss in depth quality which fails to dodge obstacles as seen in the MiDAS-2.1-S model. This model not only fails to detect dynamic obstacles well due to their small size but also predicts inaccurate depth for large static obstacles resulting in a low success rate. Although optical flow-based methods require more care to be taken when dodging static obstacles, they excel at dynamic obstacles very well due to the virtue of their design. It is remarkable to observe that optical flow models although not more accurate than depth-based models, come close in overall success rate and have a massive potential to be used as a backup or sanity check algorithm for depth model failures. Furthermore, when the optical flow models become smaller, their accuracy reduces but the navigational success rate can be enhanced by adding an activeness component to the planner as seen in our work. Furthermore, it is noteworthy to see that *MinNav* performs almost on par with RAFT in terms of success rate while being $9\times$ faster embracing the active nature of the robot. We remark that smaller robots with a dearth of high-quality sensing and computation can leverage this methodology to enable successful navigation in the wild.

V. CONCLUSIONS

We presented an active vision-based solution to navigate through a scene containing static and dynamic obstacles and unknown shaped gaps without the prior knowledge of scene components, their location and their ordering. We utilize exploratory motion on the quadrotor to alleviate common problems encountered with optical flow-based navigation. We show comparable results to depth-based navigation in both the real world and simulation. We believe such a strategy can help advance the autonomy of tiny aerial robots further. As a parting thought, utilizing optical flow and uncertainty to dodge thin wire-like obstacles is a promising direction we will explore further to enhance the navigational capabilities of small aerial robots that will bring large-scale deployment into the wild one step further.

REFERENCES

[1] Mozhdeh Mazinani, Payam Zarafshan, Mohammad Dehghani, Kourosh Vahdati, and Hamed Etezadi. Design and analysis of an aerial pollination system for walnut trees. *Biosystems Engineering*, 225:83–98, 2023.

[2] Nitin Jagannatha Sanket. *Active Vision Based Embodied-AI Design for Nano-UAV Autonomy*. PhD thesis, University of Maryland, College Park, 2021.

[3] N. Michael et al. Collaborative mapping of an earthquake-damaged building via ground and aerial robots. *Journal of Field Robotics*, 29(5):832–841, 2012.

[4] Kartik Mohta, Matthew Turpin, Alex Kushleyev, Daniel Mellinger, Nathan Michael, and Vijay Kumar. Quadcloud: a rapid response force with quadrotor teams. In *Experimental Robotics: The 14th International Symposium on Experimental Robotics*, pages 577–590. Springer, 2016.

[5] Jeffrey Delmerico, Stefano Mintchev, Alessandro Giusti, Boris Gromov, Kamilo Melo, Tomislav Horvat, Cesar Cadena, Marco Hutter, Auke Ijspeert, Dario Floreano, et al. The current state and future outlook of rescue robotics. *Journal of Field Robotics*, 36(7):1171–1191, 2019.

[6] T. Özaslan et al. Inspection of penstocks and featureless tunnel-like environments using micro UAVs. In *Field and Service Robotics*, pages 123–136. Springer, 2015.

[7] Sai Ramana Kiran Pinnama Raju, Rishabh Singh, Manoj Velmurugan, and Nitin J Sanket. Edgeflownet: 100fps@ 1w dense optical flow for tiny mobile robots. *IEEE Robotics and Automation Letters*, 2024.

[8] Nitin J Sanket, Chahat Deep Singh, Cornelia Fermüller, and Yiannis Aloimonos. Prgflow: Unified swap-aware deep global optical flow for aerial robot navigation. *Electronics Letters*, 57(16):614–617, 2021.

[9] Andrew P Duchon, William H Warren, and L Pack Kaelbling. Ecological robotics: Controlling behavior with optical flow. In *Proceedings of the seventeenth annual conference of the Cognitive Science Society*, volume 17, page 164. Psychology Press, 1995.

[10] Kahlouche Souhila and Achour Karim. Optical flow based robot obstacle avoidance. *International Journal of Advanced Robotic Systems*, 4(1):2, 2007.

[11] Berthold KP Horn and Brian G Schunck. Determining optical flow. 1980.

[12] J. Aloimonos et al. Active vision. *International journal of computer vision*, 1(4):333–356, 1988.

[13] Nitin J Sanket, Chahat Deep Singh, Kanishka Ganguly, Cornelia Fermüller, and Yiannis Aloimonos. Gapflyt: Active vision based minimalist structure-less gap detection for quadrotor flight. *IEEE Robotics and Automation Letters*, 3(4):2799–2806, 2018.

[14] Nitin J Sanket, Chahat Deep Singh, Cornelia Fermüller, and Yiannis Aloimonos. Ajna: Generalized deep uncertainty for minimal perception on parsimonious robots. *Science Robotics*, 8(81):eadd5139, 2023.

[15] Rik J Bouwmeester, Federico Paredes-Vallés, and Guido CHE De Croon. Nanoflownet: Real-time dense optical flow on a nano quadcopter. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1996–2003. IEEE, 2023.

[16] Nicholas P Burnett, Marc A Badger, and Stacey A Combes. Wind and obstacle motion affect honeybee flight strategies in cluttered environments. *Journal of Experimental Biology*, 223(14):jeb222471, 2020.

[17] Nitin J Sanket, Chethan M Parameshwara, Chahat Deep Singh, Ashwin V Kuruttukulam, Cornelia Fermüller, Davide Scaramuzza, and Yiannis Aloimonos. Evdodgenet: Deep dynamic obstacle dodging with event cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10651–10657. IEEE, 2020.

[18] Julien Dupeyroux, Jesse J Hagenaars, Federico Paredes-Vallés, and Guido CHE de Croon. Neuromorphic control for optic-flow-based landing of mavs using the loihi processor. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 96–102. IEEE, 2021.

[19] Jean-Luc Stevens and Robert Mahony. Vision based forward sensitive reactive control for a quadrotor vtol. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5232–5238. IEEE, 2018.

[20] Syed Tafseer Haider Shah and Xiang Xuezhi. Traditional and modern strategies for optical flow: an investigation. *SN Applied Sciences*, 3:1–14, 2021.

[21] Ji Zhang. Online lidar and vision based ego-motion estimation and mapping. *PhD thesis*, 2017.

[22] Yunfan Ren, Fangcheng Zhu, Guozheng Lu, Yixi Cai, Longji Yin, Fanze Kong, Jiarong Lin, Nan Chen, and Fu Zhang. Safety-assured

high-speed navigation for mavs. *Science Robotics*, 10(98):eado6187, 2025.

[23] Xin Zhou, Xiangyong Wen, Zhepei Wang, Yuman Gao, Haojia Li, Qianhao Wang, Tiankai Yang, Haojian Lu, Yanjun Cao, Chao Xu, et al. Swarm of micro flying robots in the wild. *Science Robotics*, 7(66):eabm5954, 2022.

[24] Mihir Kulkarni, Huan Nguyen, and Kostas Alexis. Semantically-enhanced deep collision prediction for autonomous navigation using aerial robots. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3056–3063. IEEE, 2023.

[25] Fei Gao, William Wu, Wenliang Gao, and Shaojie Shen. Flying on point clouds: Online trajectory generation and autonomous navigation for quadrotors in cluttered environments. *Journal of Field Robotics*, 36(4):710–733, 2019.

[26] Sikang Liu, Michael Watterson, Sarah Tang, and Vijay Kumar. High speed navigation for quadrotors with limited onboard sensing. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1484–1491. IEEE, 2016.

[27] Drew Hanover, Antonio Loquercio, Leonard Bauersfeld, Angel Romero, Robert Penicka, Yunlong Song, Giovanni Cioffi, Elia Kaufmann, and Davide Scaramuzza. Autonomous drone racing: A survey. *arXiv e-prints, pp. arXiv–2301*, 2023.

[28] Yunlong Song, Angel Romero, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Reaching the limit in autonomous racing: Optimal control versus reinforcement learning. *Science Robotics*, 8(82):eadg1462, 2023.

[29] Elia Kaufmann, Leonard Bauersfeld, Antonio Loquercio, Matthias Müller, Vladlen Koltun, and Davide Scaramuzza. Champion-level drone racing using deep reinforcement learning. *Nature*, 620(7976):982–987, 2023.

[30] Jonas Eschmann, Dario Albani, and Giuseppe Loianno. Learning to fly in seconds. *arXiv preprint arXiv:2311.13081*, 2023.

[31] Robin Ferede, Guido de Croon, Christophe De Wagter, and Dario Izzo. End-to-end neural network based optimal quadcopter control. *Robotics and Autonomous Systems*, 172:104588, 2024.

[32] Dario Izzo and Guido De Croon. Landing with time-to-contact and ventral optic flow estimates. *Journal of Guidance, Control, and Dynamics*, 35(4):1362–1367, 2012.

[33] N. Sanket et al. GapFlyt: Active vision based minimalist structure-less gap detection for quadrotor flight. *IEEE Robotics and Automation Letters*, 3(4):2799–2806, Oct 2018.

[34] Yiran Zhong, Pan Ji, Jianyuan Wang, Yuchao Dai, and Hongdong Li. Unsupervised deep epipolar flow for stationary or dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12095–12104, 2019.

[35] Anurag Ranjan, Varun Jampani, Lukas Balles, Kihwan Kim, Deqing Sun, Jonas Wulff, and Michael J Black. Competitive collaboration: Joint unsupervised learning of depth, camera motion, optical flow and motion segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12240–12249, 2019.

[36] Ruzena Bajcsy et al. Revisiting active perception. *Autonomous Robots*, pages 1–20, 2017.

[37] Lars Chittka. *The mind of a bee*. Princeton University Press, 2023.

[38] Sridhar Ravi, Olivier Bertrand, Tim Siesenop, Lea-Sophie Manz, Charlotte Doussot, Alex Fisher, and Martin Egelhaaf. Gap perception in bumblebees. *Journal of Experimental Biology*, 222(2):jeb184135, 2019.

[39] Nitin J Sanket, Chahat Deep Singh, Chethan M Parameshwara, Cornelia Fermüller, Guido CHE de Croon, and Yiannis Aloimonos. Evpropnet: Detecting drones by finding propellers for mid-air landing and following. *arXiv preprint arXiv:2106.15045*, 2021.

[40] Chahat Deep Singh, Nitin J Sanket, Chethan M Parameshwara, Cornelia Fermüller, and Yiannis Aloimonos. Nudgeseg: Zero-shot object segmentation by repeated physical interaction. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2714–2712. IEEE, 2021.

[41] Optical flow estimation using a spatial pyramid network, 2016.

[42] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[43] E. Ilg, T. Saikia, M. Keuper, and T. Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *European Conference on Computer Vision (ECCV)*, 2018.

[44] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134.

[45] Sridhar Ravi, Tim Siesenop, Olivier J Bertrand, Liang Li, Charlotte Doussot, Alex Fisher, William H Warren, and Martin Egelhaaf. Bumblebees display characteristics of active vision during robust obstacle avoidance flight. *Journal of Experimental Biology*, 225(4):jeb243021, 2022.

[46] Hang Yu, Guido CH de Croon, and Christophe De Wagter. Avoidbench: A high-fidelity vision-based obstacle avoidance benchmarking suite for multi-rotors. *arXiv preprint arXiv:2301.07430*, 2023.

[47] Chahat Deep Singh, Riya Kumari, Cornelia Fermüller, Nitin J Sanket, and Yiannis Aloimonos. Worldgen: A large scale generative simulator. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9147–9154. IEEE, 2023.

[48] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10093–10102, 2021.

[49] Chethan M Parameshwara, Nitin J Sanket, Chahat Deep Singh, Cornelia Fermüller, and Yiannis Aloimonos. 0-mms: Zero-shot multi-motion segmentation with a monocular event camera. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9594–9600. IEEE, 2021.

[50] Nitin J Sanket, Chahat Deep Singh, Varun Asthana, Cornelia Fermüller, and Yiannis Aloimonos. Morpheyes: Variable baseline stereo for quadrotor navigation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 413–419. IEEE, 2021.

[51] Reiner Birkl, Diana Wofk, and Matthias Müller. Midas v3.1 – a model zoo for robust monocular relative depth estimation. *arXiv preprint arXiv:2307.14460*, 2023.

[52] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. *ICCV*, 2021.

[53] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3), 2022.

[54] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020.