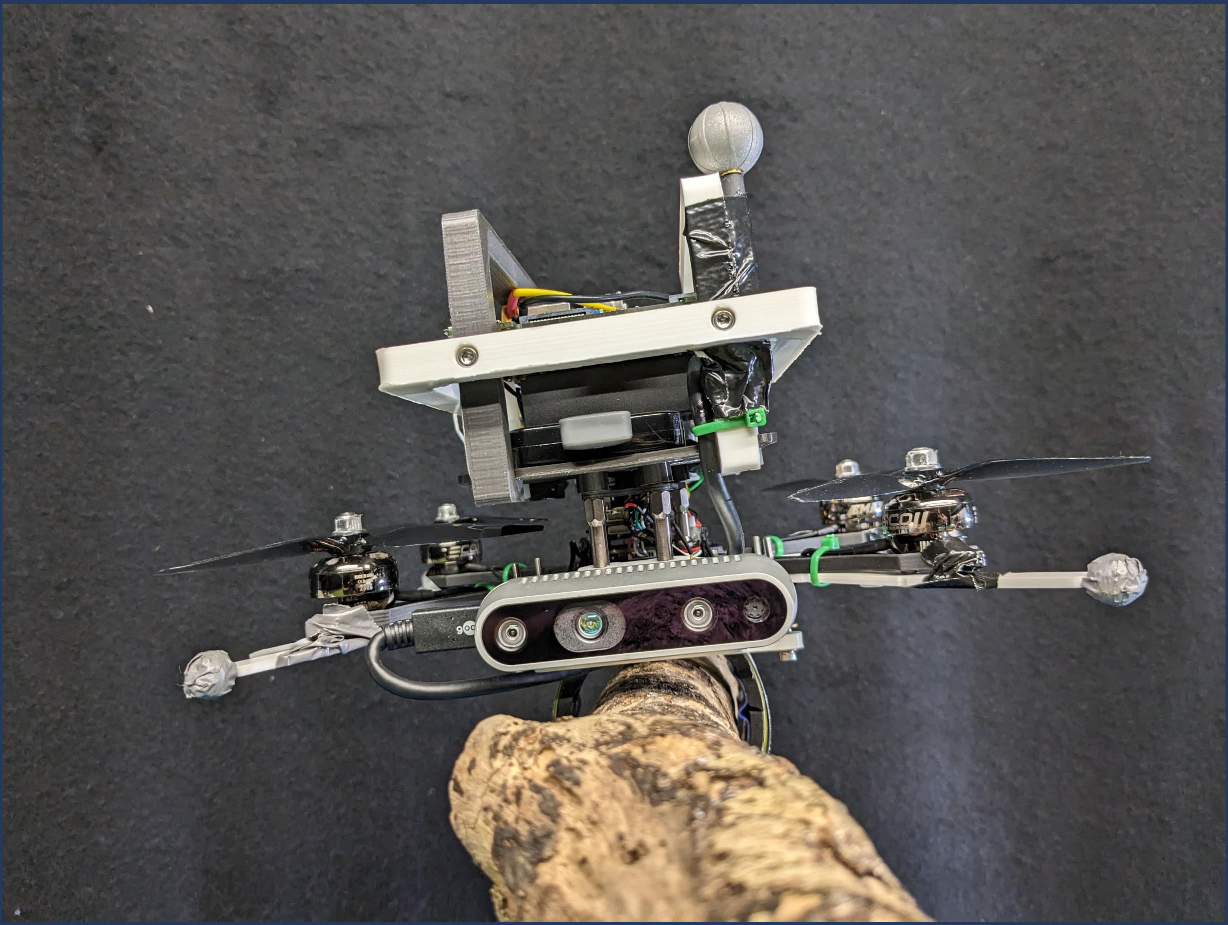
Control & Simulation Masters Thesis

Vision-guided Quadrotor Perching on Imperfectly Cylindrical Structures

Seamus McGinley



TU Delft

Control & Simulation Masters Thesis

Vision-guided Quadrotor Perching on Imperfectly Cylindrical Structures

Thesis report

by

Seamus McGinley

to obtain the degree of Master of Science
at the Delft University of Technology
to be defended publicly on September 29, 2023 at 13:00

Thesis committee:

Chair:	Dr. G.C.H.E. de Croon
Supervisors:	Dr. S. Hamaza
External examiner:	Prof. Julian Kooij
Place:	Faculty of Aerospace Engineering, Delft
Project Duration:	August, 2022 - September, 2023
Student number:	4670957

Faculty of Aerospace Engineering · Delft University of Technology

Preface

This thesis is dedicated to my mother. And Tod.

Contents

1 Introduction 1

1.1 Introduction to thesis report 1

I Scientific Article 2

2 Vision-guided Quadrotor Perching on Imperfectly Cylindrical Structures 3

2.1 Introduction 3

2.2 System Design 4

2.3 Experiments and Results. 6

2.4 Discussion and Conclusion 9

2.5 References 10

2.6 Appendices 10

II Literature Review 12

3 Literature Review 13

3.1 Contents 15

3.2 List of Figures. 16

3.3 List of Tables 17

3.4 Nomenclature. 18

3.5 Introduction 19

3.6 Perching Mechanism 22

3.7 Collision Resilience. 28

3.8 Perching Point Detection. 34

3.9 Guidance, Navigation & Control 39

3.10 Integration 44

3.11 Conclusions and Future Work 47

3.12 Bibliography. 51

1 Introduction

1.1. Introduction to thesis report

This document contains a thesis report written for a masters degree in Control and Simulation at the aerospace faculty of the Delft University of Technology. The goal of this project was to design, build and test a novel concept for an aerial robotic system capable of autonomous perching. The contributions to the current body of research are threefold: a RANSAC-based algorithm for detecting suitable (imperfectly) cylindrical perch structures in complex environments; a lightweight control system that utilises perch detections for onboard flight planning; and a passive grasper design based on bistable shell structures that uses the impact force generated during the perching manoeuvre for gripper actuation. This technology is uniquely suited for use in many field applications, such as wildlife monitoring without disturbing the fauna, surveillance from a fixed elevated position and mission extension through power generation in the perched state.

Although this report is credited to a single author, the work it describes could not have been completed without the aid and advise from a host of intelligent people. It may take a village to raise a child, but once that child decides to pursue a masters degree, they better be comfortable with seeking help from distant shores. However, to begin more close to home, the many hours that went into the research, building and writing behind this thesis project would not have been possible without the unwavering support of friends and family. It would be unfair not to thank Ana McGinley, Maurice McGinley and Kate Williamson by name for the hours each of them put into reading and reviewing the many draft deliverables produced during this project. The members of the Delft University of Technology MAVlab were instrumental during the building of the drone. This thesis would not have been completed in a single year without the near constant help of Liming Zheng, who was able to teach the fundamentals of quadrotor design in a couple months and always seemed to have answers when no one else did. Finally, Salua Hamaza must also be thanked for her supervision and guidance throughout the project and the incredible opportunities she was able to provide throughout its duration.

The structure of this report is broken into three sections. Following this introduction, in Part I the scientific article that details the work completed during this thesis is presented. Here, in the format of an IEEE scientific journal article, the hardware and software of this project can be found along with the various experiments and results used to validate the work. This new content was written specifically for this report and is the final deliverable of the masters program. In Part II, the literature review written at the beginning of the project can be found. This details the initial months of research required to develop the objectives and scope of this thesis. This has been previously graded as part of an earlier course. Each of these documents contain their own references which can be used for further reading if one wishes to delve deeper into any of the discussed topics.

Part I

Scientific Article

Vision-guided Quadrotor Perching on Imperfectly Cylindrical Structures

Seamus McGinley

***Abstract*—The design of aerial robots capable of perching poses significant challenges, from requiring pilots to master precise manoeuvres, to devising hardware and software capable of adapting to diverse perch structures and complex field environments. The Slapper drone presented in this paper tackles these challenges through three main innovations. First, a lightweight, vision-based system for autonomous perch detection using onboard flight hardware detects (imperfect) cylindrical objects found in both natural and artificial environments. Second, an onboard flight planning algorithm autonomously handles the detection, approach and perching flight phases, removing the need for a pilot. Third, a completely passive gripper utilises bistable shell structures to allow for perching on general long narrow features without any precise control inputs or power consumption. This design was successfully validated through both simulation and multiple indoor flights to result in reliable autonomous quadrotor perching in real-world environments.**

***Index Terms*—Perception-aware Motion Planning, Vision-guided Detection, Compliant Mechanism, Aerial Perching, UAV Applications**

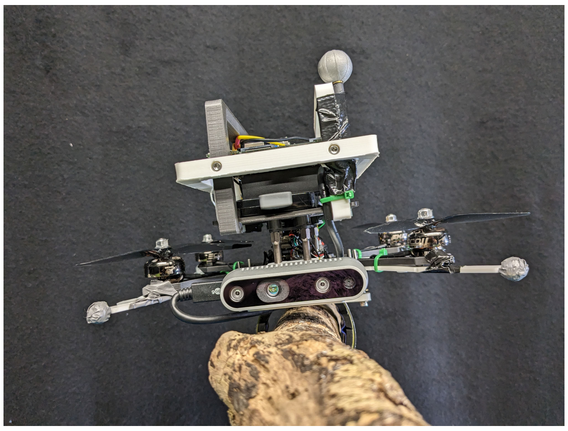


Fig. 1. Assembled Slapper drone passively perched on an imperfectly cylindrical tree branch

I. Introduction

THE ability to perch Unmanned Aerial Vehicles (UAVs) from flight provides a number of advantages that have attracted interest in recent aerospace research [1]–[3]. Perching extends mission times for aerial robots with limited battery capacity since, during a stable perch, lift-generating motors can be switched off. Furthermore, the perched position provides sensing opportunities not possible from hovering flight [4]. For example, a perched drone could monitor wildlife, collecting audio and visual data from an elevated position, without motor noise scaring fauna or obscuring bioacoustics [5]. However, to realistically allow a UAV to reap the benefits of a successful perch, especially beyond the pilot's visual line of sight, the perching manoeuvre must be automated due to the prohibitively difficult nature of manual perching.

The Slapper drone presented in this paper contributes to the growing body of research on perching drones by combining autonomous perch detection and perch manoeuvre control with a novel passive perching mechanism. The Slapper drone is designed to land on (imperfect) cylindrical structures common in both the natural and artificial worlds, e.g. branches and (warped) pipes. Using only a single stereo camera and an onboard computer running a Random Sample Consensus (RANSAC) algorithm, the quadrotor is able to detect the relative position of suitable perch structures and execute an approach to autonomously perch upon the structure. This is done without any prior knowledge of the environment. The perch detection algorithm has been designed and tested for obstacle-dense, real-world environments. The entirely passive grasper design facilitates perching using only the impact between the perch and the grasper for actuation, removing the need for additional servos that increase the weight and power consumption and that can require prohibitively precise control inputs to operate. The Slapper quadrotor can be seen passively perching on a tree branch in Figure 1.

Many different perch detection systems have been experimented with before and the Slapper drone design is built from this body of pre-existing work [6]–[9]. RANSAC algorithms have found wide use in a variety of applications, from image registration to tree modelling [10]–[12]. Prior work has used RANSAC for onboard perch detection, but this was restricted to perfect cylinders in front of a large plane and was thus not suited to realistic perching environments such as the branch of a tree high in the air [13]. A RANSAC algorithm built specifically for the detection of imperfect cylinder structures for UAV perching without this limitation has also been presented, but was never run on hardware suited for aerial robotics [14].

Bistable structures which utilise the collision force during perching to transition between the stable open and closed configurations have received much attention [15]–[18]. Recently, a gripper design using bistable cylindrical shell structures was shown to be distinctly well suited for UAV perching on structures of different dimensions and shapes [19]. The Slapper gripper design uses the same bistable structures to achieve similar performance in these aspects.

To the best of the author's knowledge, the Slapper drone is the first to achieve vision-guided UAV perching on imperfectly cylindrical structures using a completely passive gripper and onboard perch detection suited for real-world environments.

Section II describes both the unique gripper hardware and perch detection algorithms used for this drone. Section III presents the experiments conducted to validate the hardware and software and the results of these tests. Finally, Section IV discusses the potential for future work and conclusions taken from the Slapper project. The appendix contains the algorithm parameters used during testing and links to videos from each experiment.

II. System Design

A. Hardware

Much of the Slapper drone hardware consists of off-the-shelf racing drone components. A SpeedyBee FS225 V2 quadrotor frame was used, paired with a Pixracer R15 autopilot. To facilitate autonomous perch detection and vision-guided motion planning, an Intel RealSense D435 camera and an Nvidia Jetson Xavier NX are present on board. The forward-facing RealSense camera is used to generate a 3D RGB point cloud of the environment used for perch detection. The Jetson computer runs the perch detection algorithm and, simultaneously, any other programs needed for the control of the drone. The Jetson was chosen for its high computational power and compact, lightweight design; characteristics critical to allow the algorithm to be run on board the drone.

The gripper design is centred around the stacked bistable shell structures of several steel tape measure segments, previously shown to allow for passive perching on a variety of perch geometries [19]. Seven of the approximately 0.2m long and 0.02m wide steel segments were extracted from slap bracelets and stacked on top of each other. Stacking the segments in this manner improves the grasping force during perching, but increases the force needed to transition between the two stable states. Foam segments were attached to the inside of the stacked segments and a non-slip material, Dycem Non-Slip, was glued to this foam. The foam allows for the gripper to conform to any irregularities in the perch structure (e.g. the bark of a branch), and the non-slip material increases friction between the gripper and perch, improving perched stability, see Figure 2.

To attach the steel segments to the drone body, a custom frame was 3D printed from PLA. This frame consists of a top and bottom segment which are connected using metal bolts and linear springs. The two sections are designed so that, upon contact with a perch structure, the bottom section will be pushed up towards the top section which holds the steel segments. A pointed end of the bottom section is then pushed into these shell structures causing the transition from the open to the closed state. The pointed end concentrates the force of the impact into a smaller area meaning that less total impact force is needed for the transition. The linear springs serve to dampen the impact force passed to the drone frame, preventing the landing impact from causing the drone to bounce off the perch. Assembled, the gripper weighs 83g.

B. Perch Detection

The algorithm designed for detecting (imperfect) cylindrical perches from the 3D point clouds is centred around cylinder fitting using *random sample consensus*, or RANSAC.

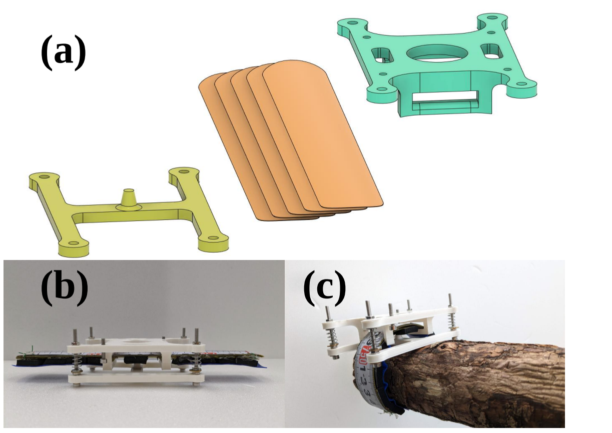


Fig. 2. Passive gripper uses bistable cylindrical shell structures. (a) Exploded view of renderings of gripper components. (b) Gripper in open configuration. (c) Gripper in closed configuration.

RANSAC algorithms have been proven to be a fast and robust method of fitting shape primitives to point cloud data and have been used for this purpose in many previous projects [13], [14], [20].

Due to their large size and noise content, the raw point clouds from the RealSense camera cannot be used directly. Instead, they are first run through decimation, spatial and temporal post-processing filters built into the RealSense firmware [21]. This helps remove noisy outlier points and intelligently scale down total point cloud size. The cleaned point cloud is then passed to a C++ ROS node for further processing using various functions from the Point Cloud Library [22]. Once received by the node, the point cloud message is downsampled again using a voxel grid filter, groups of points within a 3D voxel of a given leaf size are replaced by the centroid of those points, and then a pass through filter is applied to crop the point cloud. The point cloud is split into smaller point clusters using a conditional Euclidean clustering algorithm which clusters points based on their distance to neighbouring points in both physical space and the CIELAB colour space. The CIELAB colour space was chosen as the colour distance in this space was found to more closely represent the colour differences perceived by the human eye than the other colour spaces tested. The normals of each point are estimated and RANSAC cylinder fitting is then applied to each individual cluster.

The goal of the perch detection system is to detect the most suitable perch within the input point cloud. This is defined as the most cylindrical, but not necessarily perfectly cylindrical, structure that falls within the dimension and angle limits defined by the perching mechanism. This is represented in the code by enforcing limitations during the cylinder fitting process on the minimum and maximum perch radius and roll, pitch and yaw of the cylinder's axis in the camera's reference frame. The fit of each cluster is given a score based on the percentage of that cluster's points that fall within a threshold distance from the cylinder's surface. This scoring helps account for noise in the point cloud, but also means that imperfect cylinders, the majority of perch structures in the real

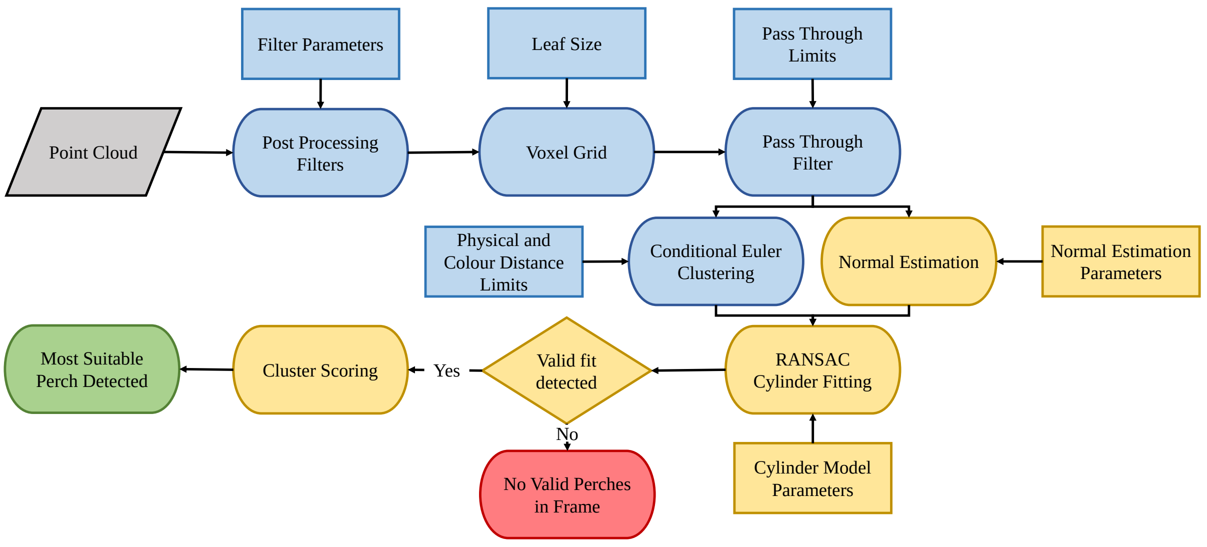


Fig. 3. Flow chart detailing the steps in the autonomous perch detection algorithm. Here, the point cloud cleaning steps are shown in blue and the cylinder fitting steps are shown in yellow.

world, can be detected as suitable perches. Figure 3 shows a flowchart representing this entire algorithm.

The noise content of the RealSense point clouds was found to be one of the largest challenges in the detection of suitable perches and, as with the filters and scoring mentioned above, many steps were taken to overcome this. The noise paired with the, by definition, random nature of a RANSAC-based detection method meant that even when the camera is pointed at an unchanging scene, it may return two separate locations as the most suitable perching point in subsequent time steps. Often, due to differences in the clustering or randomness in the fitting, the algorithm returns different clusters along the same cylindrical object as the best perching point. This is, however, not an issue. As long as the fitting parameters have been tuned correctly, all detections should represent suitable perching points and the final choice can be determined during the onboard flight planning. Again, with correct parameter tuning, the number of false detections can be greatly reduced and any others can be disregarded through additional processing. This is discussed further in the following section.

C. *Onboard Flight Planning*

A typical Slapper drone perching mission consists of three distinct phases: i) detection, ii) approach and iii) perch. As the name suggests, the first phase consists of the detection of a suitable perching location. This could be done from the ground prior to takeoff, but, in most cases, detection will be performed from flight. Whether the perch detection is performed from a stationary hover or from a ”scanning” flight pattern, detection from flight is generally preferred as it provides a broader view of the environment and potential perch locations. The detection of suitable perches will start as soon as the perch detection algorithm is run, but an additional validation step can be added between the detection and approach phases. In this validation step, the detected perching location is manually checked and either validated as a correct detection or discarded and the algorithm detects a new location. This step is not strictly necessary, but it does provide a simple way to combat the false detections mentioned in the previous section.

Once a suitable perch location has been detected, the approach phase can begin. In its simplest form, this phase entails flying in a straight line to a point directly above the detected perching point. This only works in uncluttered environments free of obstacles that the drone could collide with during its approach. If this is not the case, then the point cloud returned by the camera can be used to check whether the approach path to the perch is unobstructed. By checking if there are any points in the box defined by opposite corners at the approach point and the drone’s position, both with an additional safety margin, the drone can determine if an approach is feasible. If this is not the case, the flight remains in the detection phase and detects a new approach point.

Due to noise in the initial detection, errors in drone state estimation or changes in the environment such as wind, once the drone has flown to the initial approach point it may not be directly above the perch structure. All of these factors can be compensated for by continuing to run the perch detection algorithm as feedback for the controller. By using the cylinder fit to the initial perch structure as a reference, subsequent detections of the same structure can be used to create an updated and improved detection of the perch’s position. To ensure that the initial detection is correct, it can either be manually validated or chosen as the detection with the best score from a set number of early point cloud frames. After this, subsequent detections are verified to be of the same perch structure as the initial detection by comparing position and cylinder axis angle. The fitting score can be used as a weighing factor to create a weighted average of the perch’s pose, providing an improved perch position estimate that remains accurate over time despite changes in the environment. This updated perch position detection can be used to make final adjustments during the approach phase and ensure that the drone remains directly over the perch structure before transitioning to the final perching phase.

Due to the completely passive design of the gripper, the perching phase can be as simple as switching off the rotors and allowing the drone to free-fall onto the perch. Upon contact with the perch, the steel segments will transition from the open to the closed state and grip the perch. This requires the drone to be directly above the perch at a suitable distance, but given successful execution of the previous phases, this should be the case. It would also be possible to perform a controlled descent towards the perch without completely switching the rotors off, but the descent would have to be fast enough to generate the force needed to close the gripper. This may be more desirable in the presence of gusts that might blow the drone away from the perch during a free-fall.

III. Experiments and Results

A. *Perch Detection Tests*

To validate that the perch detection system could be used in real-world environments, it was tested outside with a real tree branch. Point clouds were generated using the RealSense camera, but the algorithm was run in a virtual machine and not the Jetson board. The choice of computer running the algorithm should not affect detection accuracy, so the choice of not using the Jetson was made to allow for a simpler testing setup. A suitable tree branch was propped up in a small tree to mimic a scenario of the quadrotor searching for a perch in a natural environment where leaves and other natural objects result in complex point clouds. During the test, the camera was moved around manually to simulate the drone in flight and to investigate the effect this has on detections.

This test showed that the algorithm was indeed capable of identifying imperfectly cylindrical features that would make suitable perches in a natural environment. Running at approximately 14Hz on the virtual machine, the algorithm returned in real-time the cluster with the highest score along with a visualisation of the corresponding cylinder fit to that cluster. The algorithm worked while the camera was in motion as long as the branch remained in frame. One of the frames of this test can be seen in Figure 4.

B. *Gripper Tests*

Drop tests were performed with the Slapper hardware to evaluate the gripper design and its ability to support the quadrotor's weight on perch structures of varying sizes when dropped from different heights. Three tree branches, with bark and other surface irregularities intact, were chosen as perch structures. The branches had average diameters of approximately 48mm, 55mm and 77mm. Note that the aim of this experiment was solely to qualitatively determine whether the gripper design is suited for different branch sizes and geometries and, therefore, other branch variables such as the looseness of the bark were not controlled for.

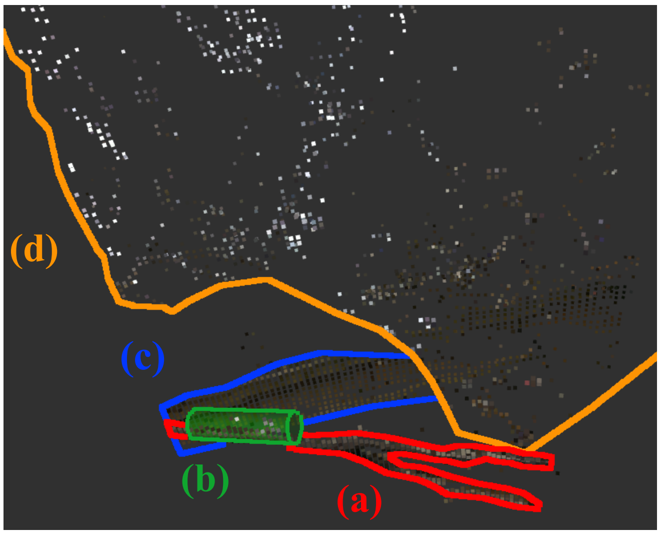


Fig. 4. Detection of suitable perch with fitted cylinder in outdoor environment. (a) Branch. (b) Cylinder fit by algorithm. (c) Fence. (d) Tree.

TABLE I
Successful Perch Drop Heights Ranges

Average Perch Diameter [mm]	77	55	48
Height Range [cm]	3 - 5	3 - 25	3 - 5

These tests showed that, while the gripper is robust to different perch sizes, performance improves when the perch geometry is within certain limits. The height at which the perching free-fall is initiated also impacts whether a successful perch can be achieved. As seen in Table I, the branch with an average diameter of 55mm resulted in stable perches from a larger range of drop heights than the smallest and largest branches, which both resulted in identically poor performance. This medium branch allowed for the most surface contact between the gripper and branch, resulting in the best recoil resistance and successful perches from greater heights. It was observed that the larger branch resulted in more frequent partial perches where, although the drone would remain in place above the branch after the drop, the gripper would not completely wrap around the branch leading to a less stable perching position. An example of this can be seen in Figure 5. Finally, the lowest height which would result in a successful perch for any of the branches was about 3cm since this was the height of a fall needed to generate the necessary force for passive gripper actuation.

C. *Simulation*

Before tests with the actual drone were performed, the algorithm and flight planning were tested in simulation. For these tests, the Gazebo simulation environment was used along with the PX4 simulation-in-the-loop firmware. The Pixracer autopilot also uses the PX4 autopilot firmware so these choices significantly reduced the simulation-to-reality gap. A drone was simulated along with a depth camera capable of generating RGB point clouds, much like the RealSense camera. The stream from this camera could be visualised during testing

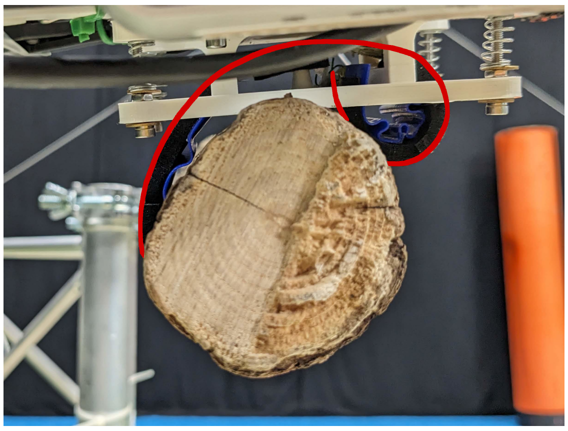


Fig. 5. Partial perch during gripper test. The shape of the steel segments are denoted in red and it can be seen that they do not correctly wrap around the branch.

TABLE II
THE TRACKING ALGORITHM USES NEW DETECTIONS TO ADJUST THE DRONE'S POSITION AFTER CHANGES IN THE ENVIRONMENT

	Position Error [m]		Angle Error [rad]		
	Initial	Adjusted	Initial	Adjusted	Score
Initial Position	N/A	0.01	N/A	0.00	0.47
Second Position	0.24	0.21	0.0	0.0	0.09
Third Position	0.21	0.05	0.2	0.06	0.87

using RViz. The gap between the real and simulated devices was nontrivial since the simulated depth camera had no noise and generated perfectly accurate point clouds. The algorithm was also run on a laptop more powerful than the Jetson, but the computational load on this laptop was higher since it also had to run the simulation.

Initial experiments in simulation were performed to determine the extent to which the point cloud images could be used to determine the feasibility of a direct approach to a perching point. The experiment was set up as follows: the drone was set to hover with a valid cylindrical perch structure in its view and a box was drawn over the RViz camera stream from the camera's origin to an approach point directly above where the algorithm detected the perch. This is the box described in Section II, where one corner is defined by the drone's position and the opposite corner is defined by the detected approach point. This box was visualised as green if the approach was unobstructed (no points inside the box) and red if the path was obstructed, as can be seen in Figure 6. To be of use in an actual flight scenario, this method would have to be fast enough to update in real time and accurate with a changing environment and drone position. This was found to be the case and the box changed colour and dimensions based on whether simulated objects were placed in the drone path or if the drone was moved by simulated disturbance forces. This happened at a rate of approximately 9Hz. This experiment shows that the real-time point cloud stream from a single depth camera could be used to determine the feasibility of a direct approach and validates this part of the onboard flight planning methodology.

To test the ability of the drone to track the perch and make any final adjustments during the approach phase, a separate

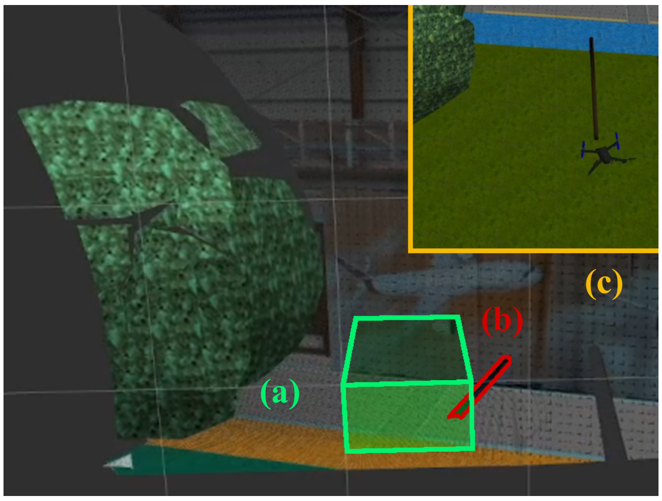


Fig. 6. Visualisation from the direct approach validation experiment. The planning box is green here which indicates that a direct approach would be possible. (a) Planning box. (b) Perch structure. (c) Gazebo view showing the drone's position in the environment.

set of experiments were conducted. In these, the simulated quadrotor starts from a hover with the perch in view, but not directly above it and with an offset between the perch axis and the drone's heading. Based on the ideal performance of the control developed algorithm, the drone should detect the perch structure and then correct the distance and angle offsets and align itself over the branch. Once the drone has done this, the simulated perch is then moved or rotated slightly and the drone should again detect this and update its position correspondingly. Note that the drone should only correct itself for small changes in perch angle or position since if the difference is too large this could represent a false detection or an entirely different perch structure. This functionality was tested by including decoy perches at large angle and position offsets from the correct perch to test if the drone would erroneously track these. This testing setup can be seen in Figure 7.

This experiment validated the perch tracking, but limitations were discovered. The drone would detect the initial angle and position offset and correct its pose and would do the same once the perch was moved. The decoy branches posed no issues during the tracking experiments. The drone, however, tended to fly forward when correcting its position. This can be explained by the 45° vertical field of view of the depth camera. Since the camera cannot see directly below or behind it, each new detection must be in front of the drone, causing the drone to fly forward when adjusting itself based on new detections. This was partially solved by not having the drone adjust itself if it is already along the detected cylinder axis. However, rotation of the perch still resulted in the drone approaching the perch's end. In some cases this would lead to the perch then being completely outside of the camera's field of view and the drone no longer being able to make any subsequent adjustments. This could be addressed further by having the drone rotate 180° in place if it can no longer detect the original feature, as it would likely be able to detect it again after the

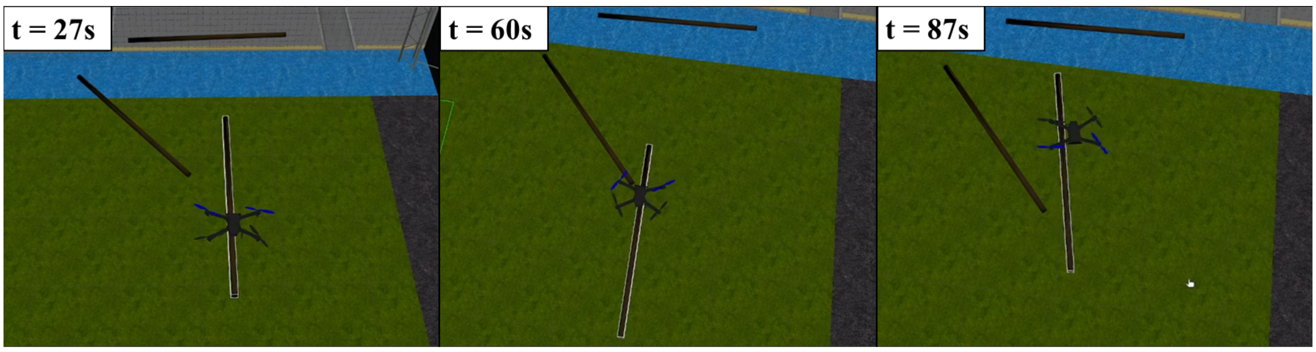


Fig. 7. Figure demonstrating how the drone can track a perch structure despite changes in the environment and in the presence of decoy perch structures. The tracked perch is highlighted in white in this figure. At t=27s, the drone is initially aligned over the perch, at t=60s the drone realigns itself over the perch after it has been shifted 0.2m to the left and, at t=87s, the drone aligns itself again over the perch after it has been moved along its axis by 1m to remain in the camera's field of view and then rotated by 0.2rad.

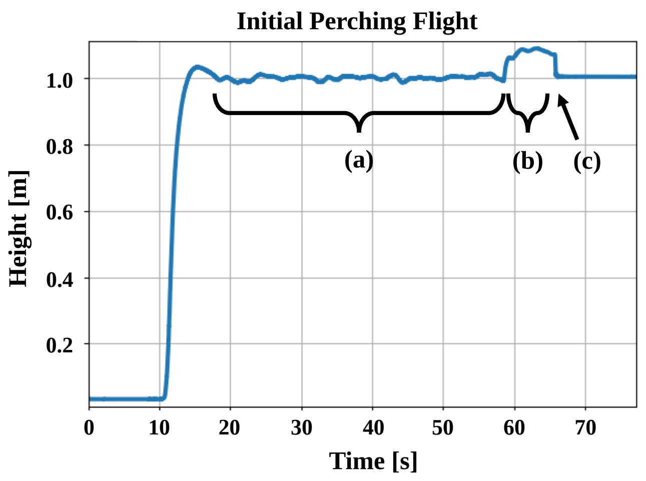


Fig. 8. Plot of initial successful perching flight experiment which clearly shows the different phases of a perching mission. (a) Detection Phase. (b) Approach Phase. (c) Perch Phase.

rotation. Since the algorithm uses the score of a detection to update its estimate for the perch position, detections with a lower score result in the offsets being closed less quickly. This can be seen in Table II where the errors are shown for the three perch positions shown in Figure 7 before and after the drone's position is adjusted using a new detection. The low score of the second detection results in the positional offset only being corrected slightly. This issue is minor, however, as future detections with higher scores can be used to correct this error, as seen with the third detection which drastically reduces the position error from the second detection while also correcting the newly introduced angle error. Overall, this experiment was considered a success as it showed that new perch detections could be used as feedback to the controller and help deal with noise, errors in the initial detection or disturbances in the environment.

D. Autonomous Perching

Indoor flight experiments were conducted using the Slapper quadrotor hardware inside the TU Delft Cyberzoo flight area. For these experiments, the drone had to autonomously detect and perch on a tree branch approximately 1m above the ground. The chosen branch was 0.93m long (of which 0.59m extended beyond the triangular truss frame), still had its bark attached and was neither perfectly straight nor cylindrical, accurately representing a branch on a living tree. The radius of the branch varied between 0.05m to 0.06m, excluding knobs which locally increased this radius. The branch had a maximum angle of 10° at its tip with respect to the level horizon. An Optitrack motion capture system was used to estimate the drone's pose, but not that of the branch. Detection of the branch and perching location occurred autonomously on board the drone using the RealSense camera and Jetson companion computer. Control of the drone took place on board its hardware using the PX4 position controller. Additional steps were taken to protect the hardware including printing a protective exoskeleton from PLA to surround the computer and some manual validation checks at key points during the mission. The total takeoff weight of the drone was equal to 1.135kg.

The mission plan for these experiments followed the same detection, approach and perch phases detailed previously. The perch was to be detected from a stationary hover with the drone facing the branch and, before transitioning into the approach phase of the mission, the detection would be validated manually. This was a simple check to ensure that the drone had correctly detected the perch and prevent a collision in the event of a false detection. The detection was not modified in any way during this check. If the detection was rejected, the algorithm would attempt to detect the branch again. If the detection was accepted, the drone would directly fly to an approach point approximately 0.05m above the detected perch. Due to the obstacle-free testing environment, it was not necessary to run the direct approach feasibility check described earlier. The final adjustment phase was also not used. This was because the narrow vertical field of view and large minimum depth distance of the camera, 58° and 0.3m respectively, and the short length of the branch meant that the branch was not visible from the approach point. Thus, detections from the approach point could not be used as controller feedback for final positional adjustments. Once the

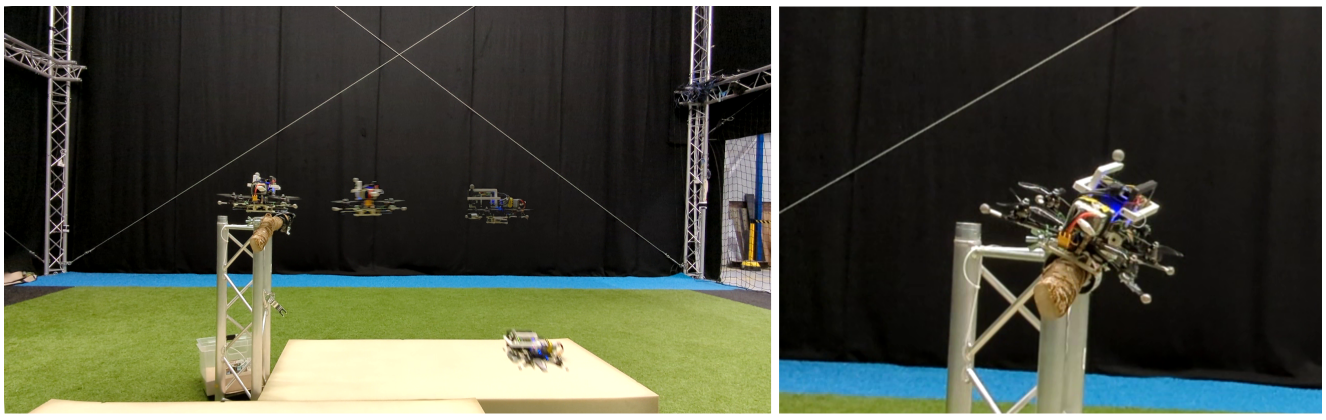

(a) (b)

Fig. 9. Initial successful autonomous perch detection and perching flight with the Slapper drone. (a) Flight time-lapse. (b) Drone in perched position after mission.

drone had reached the approach point, the perching phase was initiated by manually engaging the kill switch. Much like the perch validation step, the choice to do this manually instead of automatically switching off the rotors was to allow for the mission to be aborted in the case of a software failure that would result in a collision and damaged hardware.

Once all the algorithm parameters had been tuned, the indoor experiments showed that the Slapper drone is capable of autonomous perch detection and passive perching using a single depth camera and an onboard companion computer. For a successful perch, the drone is required to be positioned directly above the branch and, although it was shown to work using just a single detection, this would be much more reliable if it had been possible to use subsequent detections for final adjustments as described above and as tested in simulation. The perch detection algorithm ran an average rate of 5Hz on board the drone and this would have been sufficient for these adjustments with this testing setup. A more reliable system would also reduce the need for the manual validation steps and thus increase total system autonomy. A time-lapse of the first successful perching flight up until the point at which the motors were disabled can be seen in Figure 9a along with an image of the final perched position in 9b. Figure 8 plots the height of the drone during this flight against time and each of the three mission phases are clearly distinguishable.

To determine the reliability of the system, 9 additional perching flights were conducted after the initial successful mission validated that the algorithm and motion capture parameters had been tuned correctly. These tests followed the same mission plan and an identical experiment setup to the first successful flight. Two parameters were adjusted, the parameters denoting the maximum and minimum perch radii used during the perch detection, since it was discovered that these parameters had been set slightly too large for the chosen branch. An overview of the results of these tests can be found in Table III.

Of the 10 total tests conducted to validate the full pipeline, most of the failures were attributed to the gripper slipping away from the branch - although the drone was correctly

TABLE III
SUCCESS RATES IN AERIAL PERCHING FLIGHT EXPERIMENTS

Vision-based Detection	Path Planning	Branch Grasp
100 %	80 %	60 %

aligned above the branch proving the validity of the detection and planner methods. The slippage problem could be solved by adjusting the gripper design to increase the friction between the gripper and the perch, for example by lengthening the steel segments. It was also observed that bark debris would stick to the non-slip material between flights, decreasing the gripper's adhesion over time. Thoroughly cleaning the non-slip surface between flights solves this, but further exploration in the gripper design could prevent manual cleaning operations altogether. Furthermore, 2 failed perches were due to path planning errors in the approach phase; one saw the drone align itself correctly with the branch's axis but slightly too far towards the end of the branch, resulting in the gripper missing the branch after the rotors were switched off. In the second case, the drone approached the part of the branch that was being held by the truss structure and this obstructed the perching manoeuvre. These cases could be addressed by including a simple check in the algorithm to ensure that the perching point is in the lengthwise centre of a perch structure.

IV. Discussion and Conclusion

The Slapper drone presented in this paper utilises a passive perching mechanism, RANSAC-based perch detection and autonomous planning algorithms ran on board the drone to facilitate perching on imperfectly cylindrical structures. Simulations demonstrated how the perch detection algorithm could use the point cloud stream from a single forward-facing stereo camera for flight planning and as feedback for a position controller to achieve accurate positioning above a perch. Flights with the Slapper hardware and a real tree branch were performed, validating that the detection and planning algorithms can be run using on-board processing, resulting in reproducible successful perching missions without prior knowledge of suitable perch locations. This technology could be used for UAV perching in real-world environments allowing for persistent sensing from an elevated position, surveillance or mission length extension.

Future work should focus on incorporating onboard state estimation robust enough to work in outdoor environments. This will likely increase sensor weight and required computational power, but will allow for the drone to be tested in a much wider range of environments. This would provide the possibility to further develop and test the detection and planning algorithms. Finally, the work presented here could also be used to aid in developing an autonomous machine-learning-based perch detection and control system for aerial robots in real-world environments. Such a system could eventually be trained to be extremely versatile and efficient, but, to the best of the author's knowledge, no suitable training data currently exists. Autonomous outdoor flights performed by the Slapper drone could facilitate the rapid gathering of this data and lead to such a model becoming a reality in the future.

References

[1] W. Stewart *et al.*, "Passive perching with energy storage for winged aerial robots," *Advanced Intelligent Systems*, vol. 5, no. 4, p. 2100150, 2023.

[2] Y. Zou *et al.*, "Perch a quadrotor on planes by the ceiling effect," *arXiv preprint arXiv:2307.00861*, 2023.

[3] W. R. T. Roderick *et al.*, "Touchdown to take-off: at the interface of flight and surface locomotion," *Interface Focus*, vol. 7, no. 1, p. 20160094, Feb. 2017.

[4] T. W. Danko *et al.*, "Robotic rotorcraft and perch-and-stare: Sensing landing zones and handling obscurants," in *ICAR'05. Proceedings., 12th International Conference on Advanced Robotics, 2005.* IEEE, 2005, pp. 296–302.

[5] S. Kirchgeorg and S. Mintchev, "Hedgehog: Drone perching on tree branches with high-friction origami spines," *IEEE Robotics and Automation Letters*, vol. 7, no. 1, pp. 602–609, 2021.

[6] L. Li *et al.*, "Supervised Fitting of Geometric Primitives to 3D Point Clouds," 2019, pp. 2652–2660.

[7] S. Ubellacker *et al.*, "Aggressive aerial grasping using a soft drone with onboard perception," *arXiv preprint arXiv:2308.06351*, 2023.

[8] C. Luo *et al.*, "A Vision-Aided Approach to Perching a Bioinspired Unmanned Aerial Vehicle," *IEEE Transactions on Industrial Electronics*, vol. 65, no. 5, pp. 3976–3984, May 2018. [Online]. Available: https://ieeexplore.ieee.org/document/8074761/

[9] R. Figueiredo *et al.*, "A robust and efficient framework for fast cylinder detection," *Robotics and Autonomous Systems*, vol. 117, pp. 17–28, Jul. 2019.

[10] L.-R. Dung *et al.*, "Implementation of RANSAC Algorithm for Feature-Based Image Registration," *Journal of Computer and Communications*, vol. 01, no. 06, pp. 46–50, 2013.

[11] K. Olofsson *et al.*, "Tree Stem and Height Measurements using Terrestrial Laser Scanning and the RANSAC Algorithm," *Remote Sensing*, vol. 6, no. 5, pp. 4323–4344, May 2014.

[12] J. M. Martínez-Otzeta *et al.*, "RANSAC for Robotic Applications: A Survey," *Sensors*, vol. 23, no. 1, p. 327, Jan. 2023.

[13] K. M. Popek *et al.*, "Autonomous Grasping Robotic Aerial System for Perching (AGRASP)," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).* Madrid: IEEE, Oct. 2018, pp. 1–9.

[14] F. von Frankenberg and S. Nokleby, "Detection of Long Narrow Landing Features for Autonomous UAV Perching," in *2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON).* Vancouver, BC, Canada: IEEE, Nov. 2020, pp. 0565–0570.

[15] S. Guest and S. Pellegrino, "Analytical models for bistable cylindrical shells," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 462, no. 2067, pp. 839–854, Mar. 2006.

[16] C. Geckeler and S. Mintchev, "Bistable Helical Origami Gripper for Sensor Placement on Branches," *Advanced Intelligent Systems*, p. 2200087, Aug. 2022.

[17] S.-W. Kim *et al.*, "Towards a bio-mimetic flytrap robot based on a snap-through mechanism," in *2010 3rd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics.* Tokyo, Japan: IEEE, Sep. 2010, pp. 534–539. [Online]. Available: https://ieeexplore.ieee.org/document/5627994

[18] R. Zufferey *et al.*, "How ornithopters can perch autonomously on a branch," *Nature Communications*, vol. 13, no. 1, p. 7713, 2022.

[19] P. H. Nguyen *et al.*, "A Soft-Bodied Aerial Robot for Collision Resilience and Contact-Reactive Perching," May 2022.

[20] K. D. Katyal *et al.*, "A collaborative BCI approach to autonomous control of a prosthetic limb system," in *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC).* San Diego, CA, USA: IEEE, Oct. 2014, pp. 1479–1482.

[21] A. Grunnet-Jepsen and D. Tong, "Depth Post-Processing for Intel® RealSense™ D400 Depth Cameras." [Online]. Available: https://dev.intelrealsense.com/docs/depth-post-processing

[22] "Point Cloud Library." [Online]. Available: https://pointcloudlibrary.github.io/

Appendix A
Algorithm parameters used during testing

Here, the parameters that were used for the perch detection algorithm during the outdoor test and the perching flight tests can be found. First a definition of each of the parameters is presented and then the parameter values are given in Table IV.

Perch detection algorithm parameter definitions:

- **vg_leaf**: Leaf size [m] used in voxel grid filter (along all axes)
- **pt_min**: Minimum limit [m] on pass through filter (in camera z-axis)
- **pt_max**: Maximum limit [m] on pass through filter (in camera z-axis)
- **LAB_threshold**: Threshold distance in CIELAB colour space for clustering
- **cluster_tolerance**: Threshold spatial distance [m] for clusteting
- **cluster_size_perc_min**: Minimum cluster size as percentage of cloud size
- **cluster_size_perc_max**: Maximum cluster size as percentage of cloud
- **normal_k**: k-nearest neighbours considered during normal estimation
- **cylinder_normal_distance**: Relative weight given to the angular distance between point normals and the plane normal during cylinder fitting
- **cylinder_max_iterations**: Maximum number of iterations used in RANSAC cylinder fitting
- **cylinder_inlier_distance**: Threshold distance [m] for a point to be considered an inlier during RANSAC fitting
- **cylinder_radius_min**: Minimum cylinder radius [m] used during RANSAC fitting. Note, this was changed after the first flight.
- **cylinder_radius_max**: Maximum cylinder radius [m] used during RANSAC fitting. Note, this was changed after the first flight.
- **cylinder_scoring_distance**: Factor multiplied by the cylinder radius, if points within resulting distance they are considered inliers during the cluster scoring
- **optimise_cylinder**: Flag denoting whether RANSAC cylinder model coefficients are optimised after initial fit
- **cylinder_min_roll**: Minimum accepted cylinder axis roll angle [deg] in camera frame

TABLE IV
ALGORITHM PARAMETERS

	Outdoor Test	Perching Tests
vg_leaf	0.01	0.01
pt_min	0.0	0.0
pt_max	4.0	2.0
LAB_threshold	52.0	22.0
cluster_tolerance	0.016	0.04
cluster_size_perc_min	0.01	0.05
cluster_size_perc_max	0.5742	0.2673
normal_k	50	50
cylinder_normal_distance	0.21	0.1
cylinder_max_iterations	10000	1000
cylinder_inlier_distance	0.01	0.02
cylinder_radius_min	0.026	0.05, 0.02
cylinder_radius_max	0.05	0.07, 0.045
cylinder_scoring_distance	0.25	0.5
optimise_cylinder	False	False
cylinder_min_roll	N/A	0.0
cylinder_max_roll	N/A	90.0
cylinder_min_pitch	N/A	0.0
cylinder_max_pitch	N/A	90.0
cylinder_min_yaw	N/A	0.0
cylinder_max_yaw	N/A	20.0

- **cylinder_max_roll**: Maximum accepted cylinder axis roll angle [deg] in camera frame
- **cylinder_min_pitch**: Minimum accepted cylinder axis pitch angle [deg] in camera frame
- **cylinder_max_pitch**: Maximum accepted cylinder axis pitch angle [deg] in camera frame
- **cylinder_min_yaw**: Minimum accepted cylinder axis yaw angle [deg] in camera frame
- **cylinder_max_yaw**: Maximum accepted cylinder axis yaw angle [deg] in camera frame

APPENDIX B
EXPERIMENT VIDEOS

Videos from each of the experiments are given here along with a short description of the contents of each video.

Outdoor Perch Detection Demo [*https://youtu.be/Vmw2R6O6Z4k*]: In this video, the perch detection algorithm is tested outside in a complex environment. First, the environment is shown in the form of a point cloud stream taken from a stereo camera, and then the same scene is shown again, but this time with the algorithm running in real-time. The algorithm isolates the cluster in each frame that it finds to be the most suitable perch and visualises a segment of the cylinder that it has fit to this perch structures. The single colour point groups that are visible at certain points are a visualisation of other clusters resulting from the conditional clustering segmentation process.

Passive Gripper Test [*https://youtu.be/Rle8jImwDVU*]: This video includes clips from the gripper tests validating the Slapper drone gripper to allow for passive perching on branches of different geometries. Successful perches are shown for each of the different branches used during the testing. These tests showed that the performance of the gripper is dependent on both perch structure geometry and drop height.

Direct Approach Validation Test [*https://youtu.be/hoQh4WtlFPY*]: This video shows whether a direct approach to a detected perch structure is feasible. This is visualised using a box marker that is coloured green if a direct approach is feasible and red if it is not. The video shows that the feasibility is determined in real time and updates with changes in the environment and disturbances to the drone.

Perch Tracking with Decoy Perches [*https://youtu.be/9_x5Jpg2do0*]: Here, the algorithm that uses subsequent perch detections to improve the drones position above a perch after the initial detection is demonstrated. This is shown to work for changes in the perch's position and orientation and despite the presence of several decoy perch structures. Note that there is a delay before the drone updates its position due to the need to pause the simulation environment to change the perch position. The flight controller does not reconnect immediately once simulation is resumed and this causes the short delay.

Slapper Perching Flight [*https://youtu.be/ZXeYG8QJnfo*]: The entire perching flight test with the Slapper drone is shown here. The algorithm is visualised with a video overlay which shows perch detections in real time. This was the same visualisation that was used for the manual approach point validation. This experiment successfully demonstrated the capability of the Slapper drone to autonomously detect a suitable perch structure and then perch with no prior knowledge of the environment.

Part II

Literature Review

*This part has been assessed for the course AE4020 Literature Study.

Literature Review

August 2022 - November 2022

Completed By:
4670957 Seamus McGinley

TUDelft

Literature Review

August 2022 - November 2022

Preliminary literature study for an aerospace Master of Science
at the Delft University of Technology.

Student:	S. McGinley	4670957
Supervisor:	Asst. Prof. S. Hamaza	
Project duration:	August 1, 2022 – November 16, 2022	

Contents

Nomenclature **iv**

1 Introduction **1**
1.1 Background . . . 1
1.2 Research questions and objective . . . 2
1.3 Document outline . . . 3

2 Perching Mechanism **4**
2.1 Comparison of active and passive perching mechanisms . . . 4
2.2 Passive perching mechanism requirements. . . . 5
2.3 Determining perching mechanism design principles . . . 5
2.4 Implementation of perching mechanism design principles . . . 8

3 Collision Resilience **10**
3.1 Benefits of mechanical collision resilience . . . 10
3.2 Comparison of mechanical collision resilience methods . . . 10
3.3 Incorporation of origami into bumpers . . . 12
3.4 Validating mechanical collision resilience measures . . . 14

4 Perching Point Detection **16**
4.1 Perching point detection system requirements . . . 16
4.2 Comparison of different perching point detection systems . . . 17
4.3 Implementation of RANSAC for detection of long narrow features. . . . 19

5 Guidance, Navigation & Control **21**
5.1 Guidance, navigation & control system requirements . . . 21
5.2 Guidance strategy . . . 22
5.3 Navigation system . . . 23
5.4 Control strategy. . . . 23

6 Integration **26**
6.1 Required hardware . . . 26
6.2 Subsystem integration . . . 27

7 Conclusions and Future Work **29**
7.1 Conclusion . . . 29
7.2 Project plan . . . 31

List of Figures

2.1 1. Zhang et al.'s claw gripper [23]. 2. Nguyen et al.'s slapper gripper [9]. 3. Geckler and Mintchev's helical gripper [10]. 6
2.2 Possible slapper mechanism configurations 9

3.1 The three different collision resilience measures: 1. exoskeleton, 2. soft drone body and 3. rotor bumpers. 11
3.2 Soft bumpers out-perform rigid structures, as show in: 1. Sareh et al. [13], 2. Dilaveroglu and Ozcan [14]. 12
3.3 Four common crease patterns: (a) waterbomb base, (b) Miura-ori patter, (c) Yoshimura pattern and (d) Diagonal pattern [33]. 13
3.4 Dilaveroglu and Ozcan's origami bumper design [14]. 15

4.1 Perch dimension check using bounding cylinders. 17
4.2 Flow chart of perching point detection algorithm [15] 20

5.1 Block diagram of GNC system 25

6.1 Interaction between subsystems during operation 28

List of Tables

2.1 Advantages and disadvantages of active and passive perching mechanisms 5
2.2 How well each of the design concepts meet the requirements 7

Nomenclature

List of Abbreviations

1. UAV - Unmanned Aerial Vehicle
2. RANSAC - Random Sample Consensus
3. MAV - Micro Air Vehicle
4. CFRP - Carbon Fiber Reinforced Prepreg
5. IMU - Inertial Measurement Unit
6. GNC - Guidance, Navigation and Control
7. INS - Inertial Navigation System
8. GPS - Global Positioning System
9. PCB - Printed Circuit Board

1 Introduction

1.1. Background

The design of unmanned aerial vehicles (UAVs) with the ability to perch from flight is an area of growing interest in the aerospace industry and this has resulted in a large number of unique approaches to the problem [1]. This interest is motivated by the numerous benefits the ability to perch provides. To name a few examples, operation times can be extended through power generation in the perched state; monitoring and surveillance can be performed from a fixed elevated position and surface samples can be extracted from otherwise hard to reach locations. The perching problem presents its own set of unique challenges. Successful perching requires the design of a smart perching mechanism to grasp onto the environment; a method to determine which structures in the environment are suitable to perch on; ideally, some form of obstacle avoidance or collision robustness to make operating in obstacle dense environments feasible a control algorithm for the perching manoeuvre; and, finally, a controller and drone body that enable the perching manoeuvre. Each of these challenges has been tackled separately, or in smaller groups, but to the author's knowledge no research has been done that solves each of these problems within a single drone design. By combining state-of-the-art design principles for innovative perching mechanisms, mechanical collision resilience measures and perching point detection systems with a quadrotor frame and control system that facilitate autonomous perching, this project looks to be the first to do just that. A short background for each of these main research focuses is presented below.

The most appropriate perching mechanism design varies based on the drone being used and the surface that the drone is attempting to perch on. For example, a fixed-wing drone perching onto a wall will require a vastly different perching mechanism than a quadrotor perching onto a branch. To start with the former, two of the most common methods of perching to flat surfaces are micro-spines and adhesion. For example, in Lussier Desbiens et al. micro-spines are used to perch and then take off from vertical walls with a small fixed-wing aircraft [2]. In Stanford's Biomimetics and Dexterous Manipulation Lab a PhD student has taken this micro-spline concept a step further by building a quadrotor capable of perching to ceilings [3]. One limitation of both these methods is that they require rough perching surfaces for the micro-spines to function correctly. Methods that work around this limitation have also been found, such as in Anderson's design, where rat glue was used as an adhesive to stick to smooth surfaces or in Tsukagoshi et al.'s work, where suction cups were used [4] for an aerial manipulator design that enabled perching and door opening. Both come with their own limitations, such as requiring glue to be reapplied for multiple perching cycles or requiring vacuum pumps for the suction cups. Additional methods have focused on perching by grasping onto long narrow structures. Many of these methods are centered around an additional grasper like the designs presented by Chi et al. and Kitchen et al. where claw-like apparatuses are used to attach to cylindrical structures [5] [6]. Xie et al. and Doyle et al. take the bio-inspired claw method a step further by paring the claws with a leg mechanism which enables passive perching without requiring the use of servos for closing the claws [7] [8]. Other methods also exist, such as in Nguyen et al., which uses a mechanism inspired by slapper bracelets to grasp narrow structures [9], or Geckeler and Mintchev's work that uses a helical origami gripper to place sensors on branches [10]. This paper will focus on this second category of perching mechanisms as long narrow structures are common in both man-made and natural environments. The problem of detecting structures such as these that can be perched to is also much better defined than detecting surfaces onto which adhesives or micro-spines may allow perching.

Collision resilience measures for drones typically fall under one of three categories. The first of these are exoskeletons that completely encompass the drone to protect it from any impacts. A recent example of this can be found in the work done by Azambuja et al. where they present a soft-exoskeleton constructed from inexpensive materials inspired by the exoskeleton of an insect [11]. Exoskeletons offer great protection during collisions but tend to be bulky which leads to them easily getting caught on the environment. The second category is soft-bodied drones and, also inspired by insects, Mintchev et al. incorporated a soft drone body to protect their drone in the event of a crash [12]. The performance of soft-bodied drones can, however, be affected by the body deforming under aerodynamic loads which complicates control and can ultimately lead to poorer performance. The final category of mechanical collision resilience measures are propeller guards or bumpers. Rigid propeller guards are already very common in commercial drones, but soft bumpers are much better at absorbing the actual impact force. This is demonstrated by Sareh et al. in their work that uses origami engineering techniques to design a freely rotating, lightweight and soft bumper [13]. This design is exceptional at mitigating the moments experienced by a drone during collision, but is also quite bulky which can affect both aerodynamics and interactions with the environment. Dilaveroglu and Ozcan similarly use origami engineering techniques to design much smaller soft bumpers with minimal impact on the overall form-factor or aerodynamics of the drone [14]. These smaller bumpers are, however, not quite as effective at absorbing all the collision forces and moments.

Several methods exist to detect long and narrow features suitable for perching. Random sample consensus (RANSAC) can be used to determine such features from three-dimensional point clouds. This is exemplified by von Frankenburg and Nokleby in their RANSAC-based algorithm [15] built to allow the autonomous detection of perches for drones. Rabbani also presented a method for efficient automatic cylinder detection in point clouds which could be used to detect cylindrical perches [16]. This work was extended upon by Figueiredo et al. who added a neural network and RGB data to improve robustness to outliers [17]. Spica et al. used visual servoing to determine the structure of spherical and cylindrical targets [18] and Thomas et al. showed that similar visual servoing could be used to first detect cylindrical structures and then perch a quadrotor underneath these structures [19].

1.2. Research questions and objective

The aim of this literature review is to document the research conducted as part of a master's thesis project in preparation of the design and construction of a new perching UAV concept. This research should result in determining the best design principles for a lightweight and mechanically collision resilient quadrotor capable of autonomously perching onto long narrow features using a passive perching mechanism without prior knowledge of the location of suitable perching points. The research objective was to analyse current literature to find these design principles as well as how to combine them in a single design. This objective is equivalent to answering the research question, given below, that guided this research process.

> How can a passive perching mechanism be combined with mechanical collision resilience to achieve perching on long narrow features with a lightweight quadrotor without prior knowledge of the location of suitable perching points?

This is a very broad research question that cannot be answered without first breaking it down into smaller pieces. To this end, the following list of sub-questions was created where each set of sub-questions relates to a separate component or subsystem. Answering each sub-question set was a necessary step along the path towards the overarching research goal. Each of these sub-questions was answered in the order presented below, allowing the answer to an earlier question to inform the research behind later questions. These questions not only provided structure to the research but also to this document, as will be shown in the following section.

1. What are the advantages and disadvantages of passive perching mechanisms in comparison to active perching mechanisms? What are the requirements for a passive perching mechanism to

facilitate lightweight quadrotor perching to long narrow features? Which perching mechanism design principles are best suited to meet these requirements? How can these design principles be incorporated into a perching mechanism design that meets all the requirements?

2. What are the benefits of mechanical collision resilience for a lightweight quadrotor performing autonomous perching? What are the advantages and disadvantages of the following common mechanical collision methods for drones: exoskeletons, soft drone bodies and rotor bumpers? How can origami be incorporated into bumpers to improve collision resilience? How can the collision resilience improvement gained by the incorporation of these bumpers be tested in real world tests?
3. What are the requirements for perching point detection for autonomous perching of a lightweight quadrotor on long narrow features? What are the advantages and disadvantages of RANSAC, Hough Transform and Neural Network based methods for perching point location? How can a RANSAC-based system be implemented on board a lightweight quadrotor for the detection of suitable perching points?
4. What are the requirements for a guidance, navigation and control system to facilitate autonomous perching of a lightweight quadrotor equipped with a passive perching mechanism on long and narrow features? Which guidance strategy is most suitable for autonomous perching with a lightweight quadrotor? Which navigation system is most suitable for a lightweight perching quadrotor? What type of controller is most suitable for a lightweight perching quadrotor?
5. What is the necessary hardware for a lightweight collision resilient quadrotor capable of autonomous perching without prior knowledge of the location of suitable perching points? How can the separate subsystems of this perching quadrotor be integrated into a single complete design?

1.3. Document outline

The layout of this document was chosen to mirror the research process behind it. This research process was informed by the research questions presented in the following section. Each of these sub-questions were answered systematically, one after another, and often the answer to an earlier question informs the research behind a later question. This is also how the findings are presented in this paper, findings earlier in the paper are used to support later sections. Much like each set of sub-questions, each chapter relates to a separate subsystem or component of the design. Each of the sections in these chapters presents the answer to a single sub-question in that set and the order of these sections is the same as that of the sub-questions. This document layout should help link different sections and clearly represent the linearity of the research process.

After this introduction, chapter 2 begins by presenting the research process behind the perching mechanism. This is then followed by the mechanical collision resilience measures in chapter 3. The next chapter, chapter 4, rounds off the main subsystems for this research and discusses the perching point detection system. In chapter 5, the choice of guidance, navigation and control system that should be used to allow autonomous perching is justified and, in chapter 6, the interaction between subsystems, both physical and electronic, is described. To conclude, a summary of all findings along with the project plan for the continuation of this work can be found in the final chapter, chapter 7.

2 Perching Mechanism

In this chapter, the research of design principles of a perching mechanism to allow a lightweight quadrotor to perch on long and narrow features is discussed. Each section of this chapter aims to answer one of the perching mechanism sub-questions presented in the introduction of this paper. Ultimately, a slapper-bracelet inspired design is determined to be the best fit.

2.1. Comparison of active and passive perching mechanisms

An active perching mechanism is a mechanism that needs to be actively controlled to grasp onto a perch structure. Active perching mechanisms have been used before to perch UAVs onto long and narrow objects, for example in Luo et al. [20] and Hang et. al [21], but this approach does have its limitations. To successfully perch, these grippers require a control input to be precisely timed for the perching structure to be correctly grasped. This requires a high level of precision and adds an extra layer of complexity to the control algorithm. Additionally, for the actuation of the perching mechanism an actuator, like the servos used by both Luo and Hang, is required. This results in extra weight that must be carried by the quadrotor hence impacting the performance and endurance of the drone. Active actuation with a servo does allow for the closing torque to be controlled meaning that the gripping force produced by the mechanism can be controlled. Most servos also allow for the torque direction to be reversed which can be used as an easy method for re-opening the perching mechanism.

In contrast to active perching mechanisms, passive perching mechanisms do not require an active control input to close. Instead, these kinds of grippers use interaction with their environment and morphologically intelligent design to grasp onto the perching point. Perching mechanisms in this style often take inspiration from nature where birds use the landing impact force to perch on a branch, as seen in Zufferey et al. [22]. Passive perching mechanisms, like the one in this paper, do not require any additional servos to close which makes them less complex and lighter than their active alternatives. In addition, the control algorithm is simplified because no precisely timed control signal is needed to close the mechanism around the perching point. The grasping force, however, cannot be directly controlled and is predetermined by the configuration of the mechanism and the structure of the perching point. Finally, reopening these passive perchers cannot simply be achieved by reversing the servo torque direction. This means that for mechanisms such as that in Zuffery et al. a motor had to be built into the mechanism solely for the purpose of reopening. This negates one of the advantages of this kind of system.

In Table 2.1 an overview of the advantages and disadvantages of both active and passive perching mechanisms is shown. Linking this back to the original research question, the final row can be discarded since re-opening the perching mechanism to take off from the perch is not considered important. Discarding this also means that the passive perching mechanism doesn't require an additional re-opening mechanism, resulting in a lighter design that does not require servos. The controllable grasping force is then the only advantage that the active mechanism still holds over its passive counterpart. Even this can be regarded as irrelevant as long as the mechanism can generate enough force to support the drone when perched to a long narrow feature. This highly feasible with a passive mechanism of a suitable configuration and smart choices of perching locations.

	Active Perching Mechanism	Passive Perching Mechanism
Low complexity and weight	False	True
No precise control required	False	True
Controllable grasping force	True	False
Simple re-opening	True	False

Table 2.1: Advantages and disadvantages of active and passive perching mechanisms

2.2. Passive perching mechanism requirements

To determine the requirements for a perching mechanism designed for perching a lightweight UAV on long narrow features, one must first define what counts as a successful perch. For the purposes of this research, a successful perch is defined as a manoeuvre which results in the drone being attached either directly above or below the perch structure firmly enough to withstand minor disturbances without detaching. Being able to perch either above or below the perch structure will allow more freedom in the placement of the perching mechanism in later stages of the design process. Since a passive perching mechanism will be used, the perching manoeuvre should only require collision of the correct point of the perching mechanism with the perch structure to initiate the grasping of the mechanism. This means that the mechanism will have to consist of some form of bistable mechanism that switches from the open to the closed stable states on contact. The perching mechanism should also be fairly robust to the exact dimensions and structure of the perch to avoid requiring excessive modification for different perches. Finally, the perching mechanism should be as lightweight as possible to ensure that its impact is minimal to drone performance.

Since this research is being conducted with the goal of helping build a new drone design in the future, other requirements are placed upon the perching mechanism's design. These are not as vital to the function of the mechanism as the previous requirements, but are nonetheless important to ensure that the design and testing phases proceed as smoothly as possible. The perching mechanism should be easy to construct and made from inexpensive and easily available materials since multiple iterations of the design will likely have to be constructed before the optimal configuration is found. Drone testing is often destructive due to the high velocities and small delicate parts, so being able to easily reconstruct the perching mechanism in the event of it breaking during this testing process is extremely beneficial. As a final requirement, the ability to mathematically model the gripper would benefit the design process of the gripper. This model should already exist as creating a new model takes time and focus away from other aspects of the design process.

To summarise, the requirements are listed below in order of importance.

1. The mechanism should firmly attach the UAV directly above or the below perch.
2. The bistable mechanism should close on contact with the perching point.
3. The mechanism should be robust to perch structure.
4. The perching mechanism should be lightweight.
5. The perching mechanism should be easy to construct.
6. A mathematical model should exist for the perching mechanism.

2.3. Determining perching mechanism design principles

Passive perching mechanisms come in many forms. Zhang et al. designed a claw-like passive compliant gripper that uses Von Mises truss structures to convert pressure against a contact pad into gripper actuation [23]. With this gripper the researchers were able to perch underneath a cylindrical rod with a Crazyflie micro air vehicle (MAV), the grip force being more than sufficient to support the weight of the drone. Another example of a similar claw-like perching mechanism design can be found in the bio-inspired claw and leg mechanism presented by Doyle et al. [8]. These claw-like designs were by far the most common design approaches identified in the current literature. For a gripper design inspired by a different animal part, Geckler and Mintchev designed a passive helical origami gripper inspired by the tail of a chameleon [10]. They used this to attach sensors to tree branches with a UAV. The gripper was robust to the surface structure of the perching point and could be customised to achieve the desired level of performance for different branch orientations and diameters. As a final example, Nguyen et al. used a contact reactive perching mechanism that utilised the bistable mechanics of sections of tape measure to attach to long narrow features [9]. This elegantly simple approach takes inspiration from the snap bracelets that similarly use stored energy to convert contact force into a coiling movement. These three examples exemplify the different gripper design groups considered in this literature review: claw grippers, helical grippers and slapper grippers. As previously stated, it was found that the vast majority of grippers fall under the first group, but the performance for grasping to long narrow features and the unique design of the helical and slapper grippers warranted their separate inclusion.

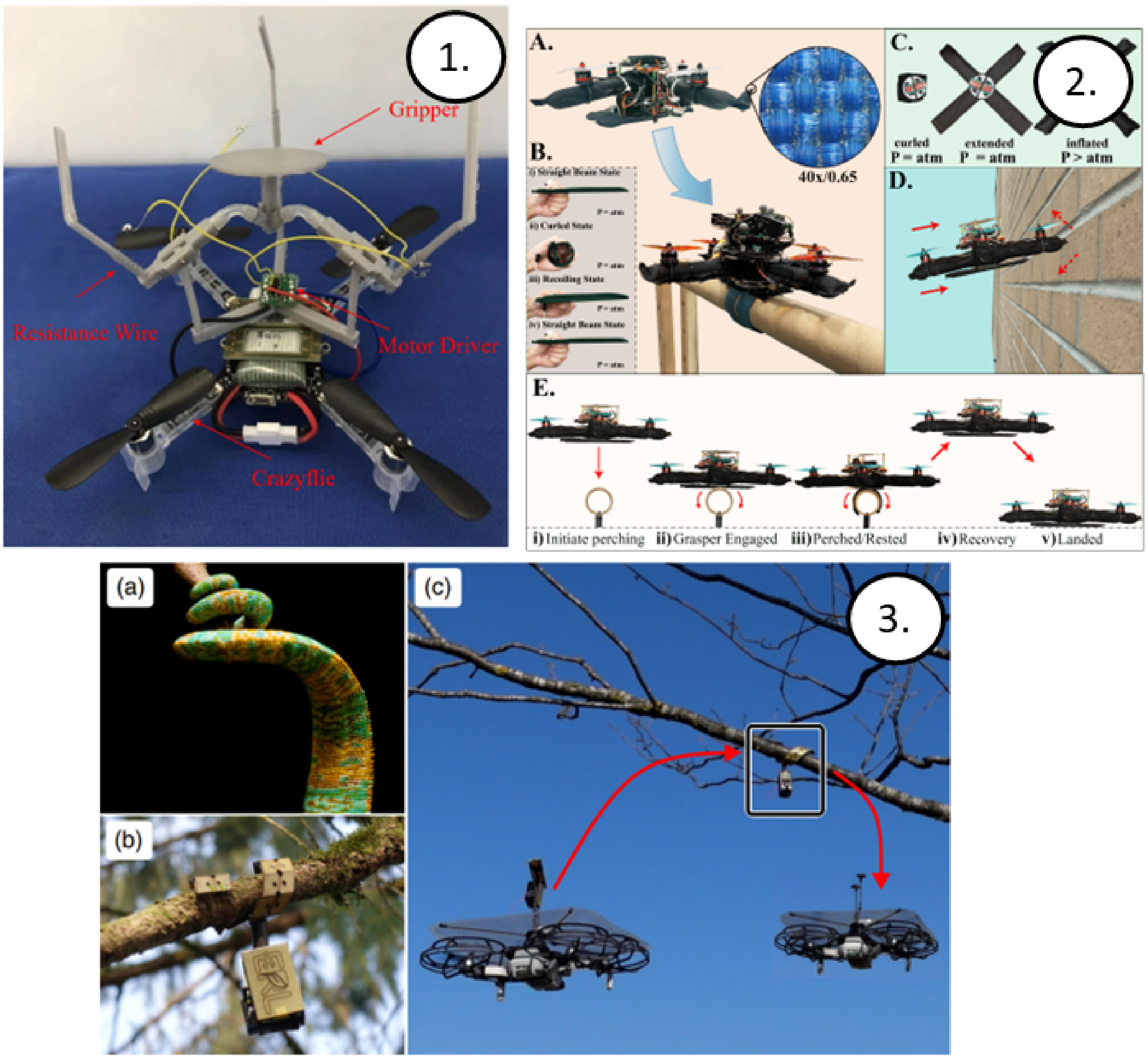


Figure 2.1: 1. Zhang et al.'s claw gripper [23]. 2. Nguyen et al.'s slapper gripper [9]. 3. Geckler and Mintchev's helical gripper [10].

To determine which of these concepts is the best fit for the novel drone design that is the objective of this paper, they must be evaluated based on how well they meet each of the requirements. For the sake of this comparison, the simplest version of each class of gripper is taken as a representative of the group: Zhang et al.'s claw, Geckler and Mintchev's helical gripper and Nguyen et al.'s slapper gripper, shown in Figure 2.1. In Table 2.2 an overview of this comparison is shown where a green ”++” corresponds to the best performance of the three, an orange ”+” corresponds to the second best performance and a ”-” corresponds to the worst performance of the three.

Requirement #	Claw Gripper	Helical Gripper	Slapper Gripper
1.	-	+	++
2.	+	+	+
3.	-	+	++
4.	+	++	-
5.	+	-	++
6.	++	-	+

Table 2.2: How well each of the design concepts meet the requirements

For the first requirement, that the perching mechanism should firmly attach the UAV either directly above or below the perch, the slapper concept emerges as the best performer of the three. This evaluation is mostly based on the fact that this gripper design allows for the most surface contact with the perch surface subsequently allowing for the greatest amount of friction. As such it is the only perching mechanism of the three that allows for the UAV to perch on top of the perching point instead of just hanging below it. This design also had the highest gripping force of all concepts, but this cannot be directly compared between concepts as will be elaborated on later. The helical gripper performed the second best of the concepts since it still allows for a reasonable amount of surface contact and friction with the perch. The claw gripper is the worst performer of the three because the drone simply ends up hanging underneath the perch with little to no lateral stability. It should be noted that other passive claw grippers are able to increase their grasping force through the incorporation of extra elements. This could lead to a better score in this category, but the increase in weight and complexity would negatively affect later categories. Furthermore, as seen in Doyle et al.'s work these design can still be susceptible to stability issues in the presence of gusts or other disturbances.

All concepts perform identically for the second requirement. All concepts are passive, bistable mechanisms that close upon contact with the perching structure, hence meeting this requirement. This is the result of deciding ahead of time to only consider passive perching mechanisms and therefore excluding active grippers from the comparison.

To compare the robustness of each gripper to perch structure, the experiments conducted in each paper had to be relied on. Both Ngyuyen et al. and Geckler and Mintchev conducted grip-force tests for different perch structure diameters. Geckler et al. even took this a step further by performing outdoor tests on natural branches. Despite this, the slapper bracelet was deemed to be the most robust to perch structure since it was found to successfully grip cylinders over a larger range of diameters, from 55 to $115mm$, whereas the helical gripper was only shown capable of gripping cylinders between a range of 8 to $36mm$. Next to the cylindrical perch tests, tests were also conducted that showed that the slapper had the capacity to even perch on a rectangular perch. The slapper gripper was not tested on natural tree branches but this should be possible given sufficient branch stiffness, a prerequisite for the helical gripper as well. The claw gripper is the worst performer in this category. No tests for gripping force for varying cylinder dimensions were conducted in Zhang et al., but it stands to reason that with this being the least compliant of the grippers under consideration, that it would encounter the most issues with variance in perch structure. It was also deemed that this structure would require the most reconfiguration for different perch structures since each of the lower and upper fingers along with each of the beams would need to be redesigned. The other grippers could simply be adjusted by varying the length of the tape measure section or changing the number of origami sections which further contributed to their better score in this category. This relative inflexibility to perch structure was found to be characteristic among claw-like grippers.

All three grippers were designed for different-sized drones of different sizes and weights. The lighter drones naturally resulted in lighter perching mechanisms as less weight had to be supported during perching. This meant that the weights of each perching mechanism could not be directly compared to one another. Instead, for this comparison, the ratio of gripper weight to grasping force was utilised. Using this, the helical gripper performed best with its gripper being able to support $56x$ its own weight. The claw gripper followed, being able to support $10x$ its own weight, and the slapper gripper performed worst, with a grasping force around $5x$ its own weight. It should be noted that this comparison is far from perfect since the relation between grasping force and gripper weight is not necessarily linear, but short of constructing versions of each mechanism identical in weight, this was considered to be the best possible comparison using just the available data.

To determine the ease of construction, the availability of materials used as well as the complexity of the manufacturing process was considered. Furthermore, any complexity added to this process by the re-opening mechanism was neglected since this functionality will not be incorporated into the final design. With this in mind, the slapper gripper scored best in this category. The actual gripper portion consisted simply of a number of tape measure sections. The pneumatic re-opening complicated the system considerably, but, as stated, this was not taken into account. The claw gripper was the second-best performer as it was constructed entirely from either 3D printed or commercially available elements. The layered origami structure of laser-cut fiberglass and Kapton layers heat-pressed together of the helical gripper resulted in it scoring lowest in this category.

Finally, the models are compared based on how well they meet the last requirement: a mathematical model should exist for the perching mechanism. The claw mechanism is the only mechanism for which a specific model exists in the paper. The Von Mises truss structure means that the opening and closing forces can be determined beforehand with a reasonable degree of certainty. Claws are a relatively well researched field resulting in more methods of modelling their behaviour. For both the slapper and helical gripper no specific models exist as of yet. However, there has been some previous work performed towards modelling bistable cylindrical shell structures which are the basis for the slapper mechanism. Kebadze et al. present a model for the stresses in these structures [24] and Guest and Pellegrino extend this work and provide an analytical method for determining the equilibrium configurations, including their stability [25]. More recently, Chen et al. extended the large deformation theory of plates and sheets with a theoretical method that does not require prior knowledge of the curvature [26]. In contrast, the novel structure of the helical gripper means that no prior research on this is available. This leads to higher scoring of the slapper gripper in this area.

Taking all the above into account, the slapper mechanism is determined to be the concept best suited to meet this set of requirements. Therefore, these are the design principles chosen for this drone design. This will be further expanded upon in the coming section.

2.4. Implementation of perching mechanism design principles

The next question that must be answered is how to incorporate the slapper design principles into a preliminary design concept that meets all the requirements. Perhaps a better way of phrasing this question would be: how should the design presented by Nguyen et al. be modified to best meet the specific set of requirements presented earlier in this chapter. The first, and most significant, change results from the fact that no re-opening mechanism is required for this drone design. This immensely reduces the complexity of the design since it leads to the pneumatic elements of the design no longer being required. The concept can essentially be reduced down to just the bistable tape measure sections.

Nguyen et al. present a number of possible configurations for the perching mechanism design, ranging from a two-finger configuration with two of the bistable slapper structures aligned in parallel and positioned directly next to each other, to a five-finger configuration with five of the same structures aligned in parallel in a zipper configuration. The zipper configuration was shown to result in better perching performance for larger perch sizes, but this may not be relevant here due to the focus on a lighter weight drone. The perching mechanism design presented by Nguyen et al. weighed only $115g$ but had to support a total drone weight of $1.14kg$. This included elements specific to that drone's design such as pneumatic pumps and soft robotic components. Without these elements, it is reasonable to assume that the perching mechanism and drone would weigh less. This lighter weight will allow the UAV to perch on smaller perches since these perches will not have to be as stiff. This means that the simpler two-finger configuration, or possibly even a single finger, should be able to produce sufficient grasping force on the smaller perches. This simpler configuration will also lead to a snowball effect where drone weight is further reduced resulting in a lower grasping force requirement, making the double- or single-finger configuration an even safer choice. On top of all this the grip force can be enhanced by including a non-slip material on the outside of the gripper.

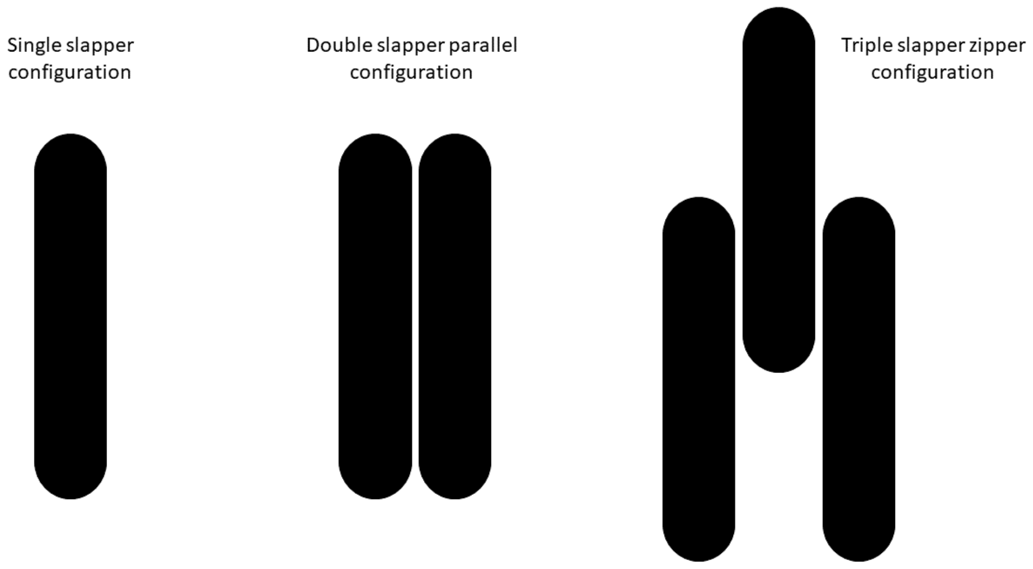

Single slapper configuration
Double slapper parallel configuration
Triple slapper zipper configuration

Figure 2.2: Possible slapper mechanism configurations

Although the original slapper bracelet mechanism used tape measure sections for the bistable shell structure, this is not the only possible way to achieve this structure. For example, Kim et al. use an asymmetrically layered carbon fiber reinforced prepreg (CFRP) to achieve a similar structure in their flytrap-inspired design [27]. This would require a much more involved manufacturing process, but would allow for more customisation of the final product than a design that uses off-the-shelf tape measure sections. It is likely that this would lead to a lighter-weight final perching mechanism. Another possible option is to use the actual snap or slapper bracelets after which this concept was named. These are commonly available and can result in very tight coils when compared to the designs shown in both Nguyen et al. and Kim et al. They could also be layered on top of each other, much like the tape measure sections, to modify performance. The tape measure segments had to be pre-formed to behave as desired, which might be avoidable when using slapper bracelets. This and the fact that slapper bracelets can be purchased in bulk at low cost also makes them the best out of all options mentioned especially when considering the fifth requirement, the ease of construction.

To conclude, the slapper design concept needs slight modification to best meet the design requirements for this lightweight passive perching mechanism. The re-opening mechanism will be removed as it leads to higher complexity and is not needed for this concept. Unlike the design for larger perching structures, a simpler lightweight configuration of either a single or two parallel slapper structures will be used and these will be covered in a non-slip surface to increase grasping force. Finally, instead of tape measure sections or the lighter-weight CFRB, slapper bracelets, will be used for the bistable shell structure to ease the construction process.

3 Collision Resilience

This chapter discusses the benefits of collision resilience for a perching drone and how to best implement mechanical collision resilience measures into a lightweight quadrotor design. Each section in this chapter answers one of the sub-questions from the second set presented in the introduction. Origami bumpers are found to be the most appropriate choice and this chapter finishes by considering how this choice can be validated through experiments.

3.1. Benefits of mechanical collision resilience

Due to the nature of the problem, perching requires a drone to come into close contact with rigid structures. In a real-world scenario, this extends beyond the perch structure and includes any other obstacles in the vicinity of this perch. For example, during the scenario of manually perching a drone on a tree branch, a pilot has the difficult job of manoeuvring the drone around the tree's trunk and any branches obstructing the path. In the unfortunate but not unlikely case of a collision with either of these, the mission would almost certainly end in failure due to critical damage to UAV components or loss of control. Drone components can be expensive and a drone falling out of the sky can be a safety hazard so special care should be taken to avoid this outcome. To achieve this, some form of collision resilience or obstacle avoidance is needed.

Obstacle avoidance systems use sensors to detect obstacles and then manoeuvre the drone to avoid these obstacles. Current approaches rely on cameras and vision-aided techniques [28], or measure distance to obstacles directly using distance sensors [29]. Both of theses approaches add considerable complexity to the system, both computationally and through requiring additional sensors that may not have been needed otherwise. Even when implemented correctly, these systems are far from perfect and may be unable to reliably detect all obstacles in all conditions. It is therefore beneficial to include some form of mechanical collision resilience into a UAV design, especially for UAVs required to fly in close proximity to other objects, as is the case for a perching quadrotor. Mechanical collision resilience may also add weight to the drone, but it does not necessarily have to add any computational complexity to the system. It has the further benefit of working even in cases where an obstacle could not be detected ahead of time. Finally, mechanical collision resilience can also allow for faster flight speeds as it will function at all speeds and is not limited by the sampling and processing speeds of a software-based obstacle detection system to function.

3.2. Comparison of mechanical collision resilience methods

Three different forms of mechanical collision resilience will be considered: exoskeletons, soft drone bodies and rotor bumpers. These were deemed to be the three most relevant forms of collision resilience. An example of an exoskeleton was designed by Klaptocz et al. who investigated the use of an Euler-spring-based external structure to achieve collision resilience [30]. This soft exoskeleton concept was expanded by Azambuja et al. who designed and built a soft, insect-inspired exoskeleton from inexpensive materials around a CogniFly MAV [11]. A rigid exoskeleton frame paired with a gimbal system surrounding a simple flying robot has also been designed and tested by Briod et al. [31]. Nguyen et al. is again a great example of a soft-bodied drone [9]. Using pneumatic pumps they could control the rigidity of the arms of their drone's body and through this manipulate how their drone reacted to collisions. Mintchev et al. built and tested a soft drone body inspired by the collision resilience of insect wings [12]. Finally, numerous examples of rotor bumpers exist and simple versions of these are included on many commercial drones to protect the rotors in the event of a collision. For a more interesting implementation of this concept, consider the work done by Dilaveroglu and Ozcan who built soft rotor bumpers using origami techniques which they attached to the end of the arms of their quadrotor [14].

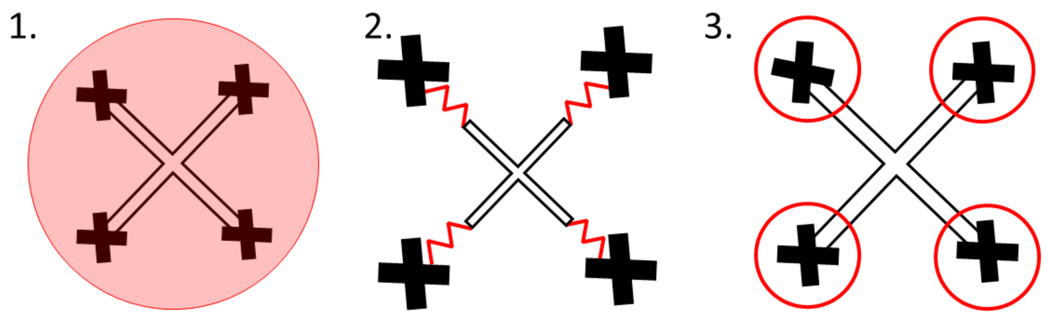


Figure 3.1: The three different collision resilience measures: 1. exoskeleton, 2. soft drone body and 3. rotor bumpers.

An exoskeleton built to completely encircle the drone's body can protect delicate components from collisions with objects from any approach angle. None of the other concepts offer this same level of coverage. The larger size of exoskeletons also leads to a greater distance between the object that is collided with and the drone body. This reduces the chance of damage to the drone due to a moving component, like a rotor, coming into contact with the environment. In the case of soft exoskeletons, such as those designed by Klaptocz et al. and Azambuja et al., this greater distance is taken advantage of to better absorb the collision force. Exoskeletons do have certain disadvantages when compared to other approaches. The larger form factor of exoskeletons results a significant increase in total drone weight, which directly affects performance and endurance. When flying in dense real-world environments, structures such as branches can get caught inside the exoskeletons which can result in the exoskeleton becoming more of a hazard than a benefit in these cases. Finally, these external structures could make perching more complicated as they might limit the angles from which the perch can be approached. Due to this, in combination with the other listed reasons, it was decided that exoskeletons are not the best mechanical collision resilience strategy for this design.

The soft body approach to collision resilience is uniquely elegant in not requiring any additional external structures to absorb impact. Instead the collision resilience is achieved through deformation of the drone's body itself. This is especially attractive for design concepts such as the lightweight quadrotor being researched for this project where keeping the overall weight of the drone to a minimum is a priority. Not having any additional external structures also keeps a sleek overall drone profile, beneficial for aerodynamics and performance. In contrast to the exoskeleton concept, the significantly smaller form factor would result in a much easier time flying through obstacle dense environments without snagging onto any objects. The soft body concept does, however, have its own set of unique disadvantages.

The largest of these disadvantages being that the deformations in the soft frame of the drone due to aerodynamic forces such as rotor thrust can lead to unpredictable in-flight behaviour. Ideally, the frame would be rigid in certain directions to prevent this, but, as seen in both Nguyen et al. and Mintchev et al.'s designs, current concepts aiming to achieve this have not yet been completely successful. This deformation of the frame could also influence perching with a passive perching mechanism. These mechanisms function through converting the collision force with the perch to mechanism actuation and, if the frame were to absorb part of this impact, larger collision forces may be required for this actuation and these higher forces may be hard to generate using a small UAV.

Bumpers can combine the sleek profile of the soft body approach and maintain the performance benefits of a rigid quadrotor frame. They can also be viewed as a stripped back exoskeleton that only encompasses the propellers of the drone. Since they are not as large as an entire exoskeleton they can be much lighter and do not inhibit flight within a tight environment as much. Much like exoskeletons, both rigid and flexible bumpers exist. The former is the most common and is widely available as an add-on to many commercially available drones [32]. The latter is less common, but can absorb the collision impact force more effectively, in much the same way as a soft exoskeleton or soft drone body. This improvement in collision absorption can be seen in Figure 3.2. An example of this are the origami bumpers designed by Dilaveroglu and Ozcan. These extended the arms of their quadrotor with bumpers that could deform to absorb and dissipate the impact, improving collision resilience. Such bumpers could be mounted onto an otherwise rigid drone frame and thus avoid the in-flight and perching consequences of a soft frame. The main disadvantage of bumpers, especially when compared to an exoskeleton, is that they only protect the drone over a limited range of collision approach angles. They typically offer no protection for collisions from above or below. This is a real drawback from a collision resilience perspective, but it does mean that they do not interfere with quadrotor perching manoeuvres approached from above or below such as would be necessary for the mechanism described in chapter 2. For the design of a lightweight perching drone, these sets of advantages and disadvantages are the most favourable, therefore it was decided to use soft bumpers to achieve mechanical collision resilience.

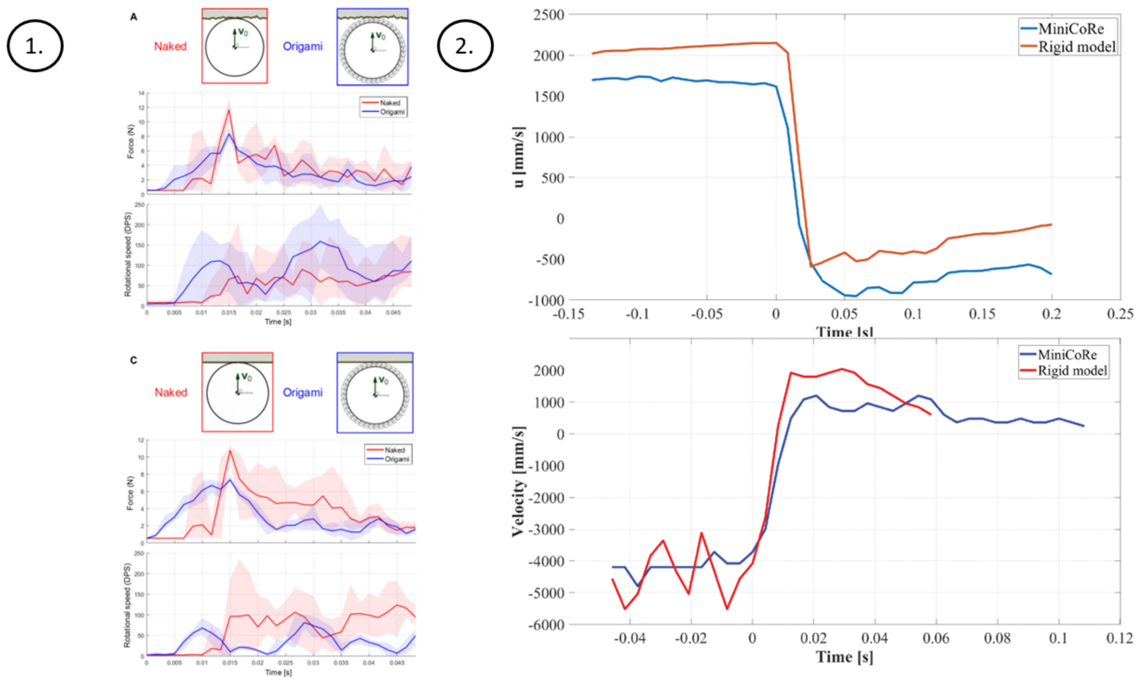


Figure 3.2: Soft bumpers out-perform rigid structures, as show in: 1. Sareh et al. [13], 2. Dilaveroglu and Ozcan [14].

3.3. Incorporation of origami into bumpers

Origami refers to the ancient Japanese art of folding thin sheets of paper to create three dimensional models. In the past fifty years engineering has increasingly explored the same principles used in origami—more accurately, kirigami which also allows for cutting—to construct structures with unique mechanical properties from thin sheets [33]. Instead of paper, sheets of materials, from composites to metals, are plastically deformed through bending beyond their yield point to the desired crease pattern. These structures can be designed to have unique dynamic properties where the folds result in different kinematics in different sections of the origami structure. A few common crease patterns are the Miura-ori pattern, the waterbomb base, the Yoshimura pattern and the diagonal pattern [34]. The first two of these have the interesting property of having a negative Poisson's ratio, meaning that the when the pattern is expanded in one direction, it also expands in an orthogonal direction. The Yoshimura pattern results in a curve or cylinder by folding a diamond pattern into the sheet and the diagonal pattern can be used in folded cylinders to allow for contraction through rotation. These crease patterns can be seen in Figure 3.3.

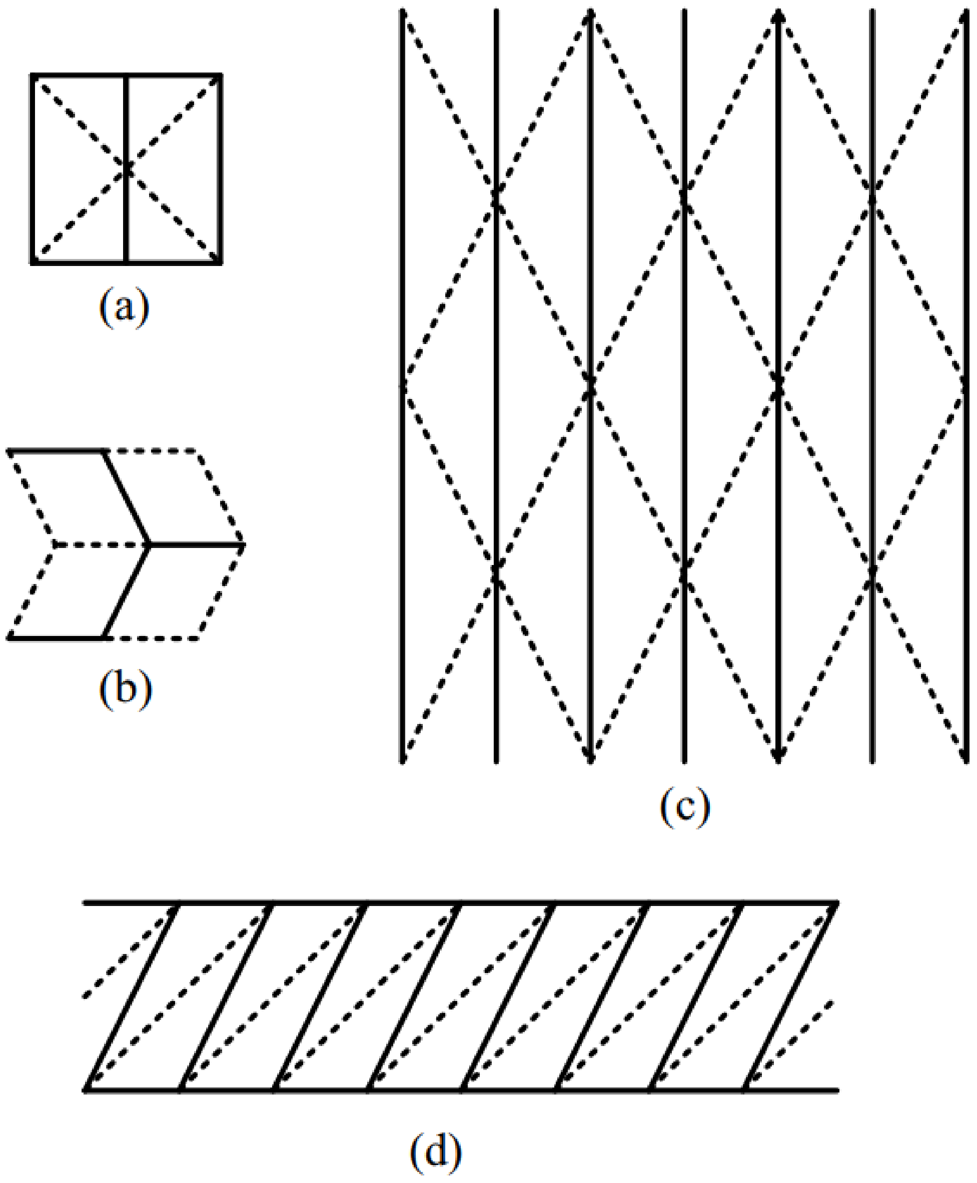


Figure 3.3: Four common crease patterns: (a) waterbomb base, (b) Miura-ori patter, (c) Yoshimura pattern and (d) Diagonal pattern [33].

The main advantage of using origami structures in drones is their light weight. As stated previously, this is hugely advantageous as it allows for better drone performance and endurance. Any weight saved here also allows for other components of the drone to be heavier, such as the perching mechanism and the computational hardware. The flexibility of certain origami structures can also be exploited to obtain mechanical collision resilience. This is the same principle used by soft exoskeletons and bumpers. Through the utilisation of smart folding patterns, a lightweight sheet can be folded to elastically deform upon collision and thus protect the drone. Origami manufacturing can also have the benefit of being a relatively inexpensive and simple process. The cost depends mostly on the material used and often the material is an inexpensive and commercially available polymer. Also depending on the material used, the folding of the structure can often be performed manually by hand which means that the only machine required during the manufacturing process is for the cutting of the sheet. In most scenarios, this can be performed quickly and precisely using a laser cutter.

Many possible ways exist to incorporate origami engineering into drone bumper design. One of the more creative was designed by Sareh et al. with their freely rotating bumper that surrounds the drone in the horizontal plane to protect it from collisions [13]. The manually-folded origami bumper deforms to absorb the normal forces during a collision and the free rotation of the bumper helps mitigate the tangential forces. This design is exceptional at minimising the yaw moment to the drone resulting from a collision. In contrast to this intricate design, an elegantly simple approach to origami bumpers can be found in Dilaveroglu and Ozcan's paper [14]. Their bumpers consist of a single laser-cut piece of cellulose acetate with a single half-fold which is attached to the end of each of the drone's arms. These bumpers deform elastically on collision and then release the stored energy by moving the drone away from the impact point.

For the purposes of a lightweight perching drone, a approach more akin to that of Dilaveroglu and Ozcan is the better choice. This might seem counter-intuitive as the ”Rotorigami” design presented by Sareh et al. leads to better performance, especially at reducing yaw moments during collisions, yet it also has certain disadvantages that must be considered. Although the rotating origami bumper was designed to be as lightweight as possible, constructed from a thin polypropylene sheet, its $4.5g$ still ends up making 9.3% of the total drone weight. The circular frame and connector make this number even higher. This is a big proportion of the total weight and restricts the other hardware components that could be added to the drone. This is especially relevant in the case of a lightweight perching drone since it could affect the perching mechanism. The larger form factor of a design like the ”Rotorigami” also impacts the aerodynamic characteristics of the drone resulting in more drag, especially when flying upwards or downwards. This may lead to lower attainable speeds in both these directions. Againm this is especially relevant to a perching drone since certain speeds will need to be reached in these directions to close the passive perching mechanism. These reasons make simpler and lighter weight implementations like that of Dilaveroglu and Ozcan the best option.

3.4. Validating mechanical collision resilience measures

Any form of mechanical collision resilience needs to be tested after implementation to determine how well it functions. These experiments should determine how well the collision forces are absorbed and how large an impact can be sustained without damage to the sensitive drone components. For a drone, it is also important to determine under which collision conditions the collision resilience measures now enable the drone to withstand the collision while remaining in flight. Unfortunately, due to the nature of drone collisions, in-flight tests with a full drone will likely be destructive, especially in the cases where the mechanical collision resilience is not sufficient to completely mitigate impact forces. Due to this it is a good idea to first conduct some initial, more tightly controlled experiments to determine the boundaries of the collision resilience measures and then expand upon these tests with in-flight experiments.

Perhaps the most common experiment used to measure collision resilience is the drop test [9][11][12]. The height and the orientation of the drone before the drop can be controlled so that the collision conditions are as desired. The rotors of the drone should remain stationary to minimise damage to these delicate components. Accelerations can be measured by equipping the drone with an inertial measurement unit (IMU). A more expansive version of these tests can be found again in the work done by Sareh et al. where they tested the performance of their origami bumper through a series of pendulum tests. For these tests, a drone equipped with the bumper was attached to the end of a rope and then swung against a collision surface. This was done in such a fashion that the collision speed and angle could all be controlled and easily replicated between tests. A tangential velocity component could be added

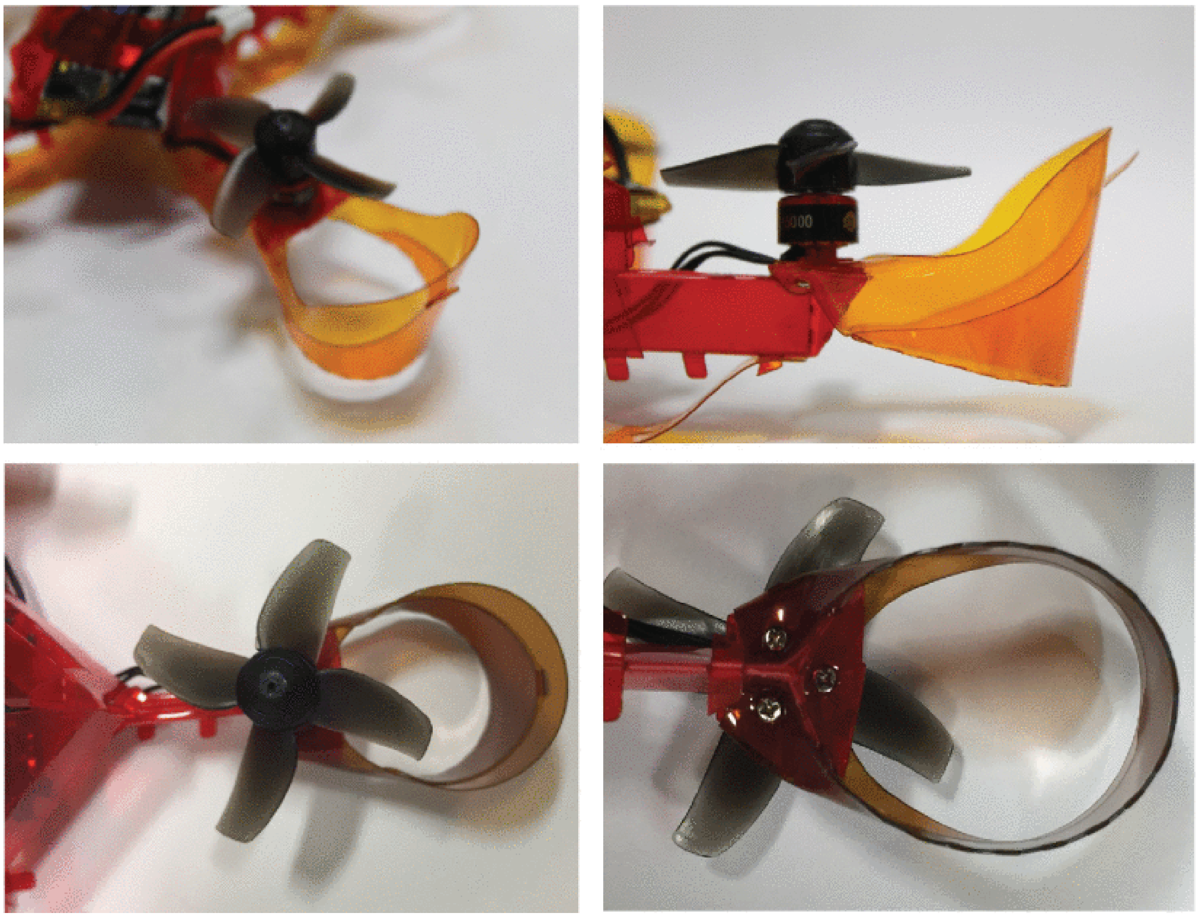


Figure 3.4: Dilaveroglu and Ozcan's origami bumper design [14].

to the collision and the collision surface could be adjusted to determine their effects on the collision resilience. The experiments were conducted with and without the origami bumper equipped to determine the difference made by the bumpers. The drone's rotors remained stationary during these tests and the IMU on board the drone was used to measure the forces and angular velocities experienced during collision. This expanded experimental setup is deemed preferable over typical drop tests since it allows for much finer control.

Being able to recover in flight from a collision is the ultimate goal of incorporating collision resilience into a drone's design and in-flight tests are needed to determine how well the added collision resilience measures allow for this. These tests will involve flying the drone towards an obstacle with a certain velocity and approach vector and seeing whether it is able to regain stability after a collision. These tests are usually conducted in a controlled environment where the collision behaviour can be filmed clearly for later analysis [35] [14]. For complete validation of the system, however, flights in the real-world operating environment of the drone should also be conducted. For a perching drone this could mean flying amongst the tree branches that it might perch on. Final tests could involve collisions with these tree branches to see how the limited space affects recovery. Since these tests will almost certainly lead to the destruction of the drone, they should only be conducted after the other tests have determined the boundaries of impact speeds and approaches from which the collision resilience measures provide the drone a reasonable chance of recovery.

4 Perching Point Detection

Each section of this chapter answers one of the sub-questions from the third set, pertaining to the perching point detection system. Three different common approaches are analysed and their unique sets of strengths and limitations are discussed. It is shown that a modified version of a random sample consensus (RANSAC) algorithm is the best option for determining the relative position of suitable long and narrow structures which can be perched on.

4.1. Perching point detection system requirements

A successful implementation of a perching point detection system would enable the quadrotor to detect the position of suitable perching features relative to its own position. For this, it would first have to be able to determine what is in fact a suitable perching point. For this design, these points would be long and narrow features that can be grasped by the perching mechanism. The exact dimensions of suitable features would be determined during the final design and testing of the perching mechanism, but could be generalised as features that fit within certain bounding cylinders, see Figure 4.1. After such structures have been detected, their position would also need to be determined. Their absolute position is not immediately relevant, instead their position relative to the drone is. The UAV will have to detect suitable perches from flight and then manoeuvre to a certain location and attitude relative to the perch to initiate the landing procedure. This means that the algorithm should be able to run fast enough that it can be used as a real-time, in-flight input to the flight controller.

Since this system is to be used on board a lightweight quadrotor, there are other essential requirements for the system to be feasible. The algorithm used by the system should be as computationally inexpensive as possible. This allows it to be run at a fast enough rate to interact with the flight controller, and keeps the total weight of the system down. A more computationally complex algorithm would require additional hardware to run this algorithm. This could also lead to a snowball effect where this additional hardware requires more power and thus a larger battery, further increasing the weight. To this same goal of minimising the total weight of the system, the number of additional sensors required for perching point detection should also be kept as low as possible. Each additional piece of hardware will take up a valuable portion of the total weight budget for the quadrotor which is already very limited. If too much of this budget is taken up by the perching point detection system, other functionalities may have to be sacrificed.

An overview of the requirements for the perching point detection system design are listed below:

1. The system should detect the relative position of perching points of suitable dimensions.
2. The system should run fast enough on board the drone to be used as a control input.
3. The additional weight required by the system should be kept as low as possible.

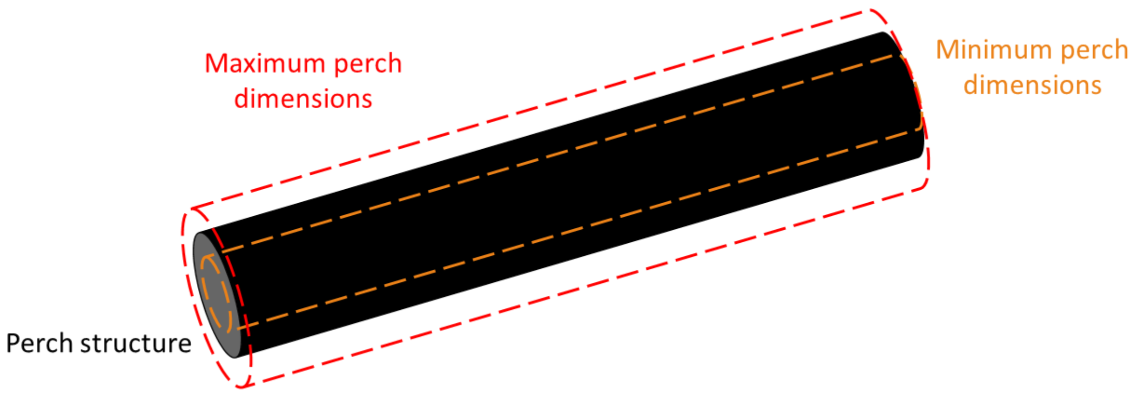


Figure 4.1: Perch dimension check using bounding cylinders.

4.2. Comparison of different perching point detection systems

The three approaches considered for the perching point detection system are those based on the random sample consensus (RANSAC) algorithm, Hough transforms and neural networks. The RANSAC algorithm is one of the most common approaches for fitting shape primitives to three-dimensional point cloud data and this is reflected in the UAV perching literature. The AGRASP drone designed and built by Popek et al. uses a RANSAC algorithm to detect suitable cylindrical perches using only on-board processing [36]. An algorithm using RANSAC has also been developed by von Frankenberg and Nokleby that can be used to detect more general long narrow features viable for UAV perching [15]. Hough transforms have also seen wide use in fitting cylinders in point clouds. A great example of this can be seen in the work done by Rabbani and van den Heuvel, where they show how Hough transforms can be used to efficiently fit cylinders to objects in an industrial site [16]. Suitable perches could be determined in a similar manner by finding structures where the bounding cylinder falls between certain boundaries. Finally, neural network methods also exist for fitting primitives to point clouds, such as in the work done by Li et al. [37]. This could be adopted to determine suitable perches within certain geometric bounds much in the same way as the Hough transform method. Combinations of neural networks and Hough transforms have also been explored by, for example, Figueiredo et al. who use a neural network to first detect cylinders from an image and then perform Hough transforms to fit these cylinders [17].

It should be mentioned that some other methods that could be used to determine suitable perches do exist, but it was decided to focus on RANSAC, Hough transforms and neural networks because they were deemed to be most common and thus most proven. The only other method that came close in these measures were visual-servoing-based approaches. Seo et al. used a combination of RANSAC and visual servoing to fit cylinders that could be grasped with a gripper, but this failed when both ends of the cylinder were not in frame [38]. This would not work for perching where the ends of the perch structures are rarely visible. Spica et al. developed a visual servoing model that they used to determine the structure of spherical and cylindrical objects. Their algorithm relied on these objects being easily distinguishable from the rest of the environment beforehand which would again not be the case for the perching use-case [18]. Other papers have used visual servoing for perching but these did not extend to actually determining which structures were suitable to perch on [20] [19]. For these reasons it was decided to exclude visual-servoing based-approaches, even though they have been shown to work in certain restricted problem spaces.

Random sample consensus is an iterative algorithm that can be used to fit data in the presence of outliers. The RANSAC algorithm functions by randomly sampling a subset of the data and then fitting a model to this smaller subset. All the other data points outside this subset are checked to see how many of them fall as inliers for the model and this is then used to score the model. This process is repeated a given number of times with, ultimately, the highest scoring model being taken as the best fit. RANSAC can be utilised for many purposes as outliers are present in the outputs of most real-world sensors, but it has found particularly widespread use in computer vision applications such as video stabilisation and fundamental matrix estimation [39]. For perching point detection, RANSAC could be used to detect structures of suitable dimensions from the three-dimensional point cloud data returned by a depth camera. These sensors are prone to outliers, so RANSAC is a good choice for fitting shape primitives such as cylinders to their outputs. As previously mentioned, the work done by von Frankenberg and Nokleby presents a method on how this can then be extended to find long and narrow features that a UAV can perch on. The main disadvantages of a RANSAC-based approach is that it can be computationally expensive, von Frankenberg and Nokleby's algorithm ran on a Intel i7-4770 3.40 GHz 4 core processor which is much more powerful than the processors that can fit on board a lightweight quadrotor, as well as the algorithm's susceptibility to the masking effect where certain data points only emerge as influential after the deletion of other points [40]. Both of these can, however, be at least partially mitigated through tuning of the algorithm's parameters.

Much like RANSAC algorithms, Hough transforms are also commonly used in computer vision. The Hough transform was originally used to find lines in images, but has since been expanded to also find other classes of shapes in images. Hough transforms are robust to outliers and can detect imperfect instances of shapes according to an incorporated voting procedure. Here each point votes for the set of shape parameters possible including that point and then, after the procedure is finished, the set of parameters with the highest score is chosen. The three-dimensional Hough transform has been used on point clouds to fit cylinders in the environment (see again the well-known work done by Rabbani and van den Heuvel) and this is also how it could be used for perching point detection. In this work, the cylinder detection is split into two stages: first the cylinder orientation is estimated using the normals of each point and then the position and radius of the cylinder are estimated by rotating the point cloud. Hough transforms tend to be computationally expensive, and this increases exponentially with the dimensionality of the problem. Rabbani and van den Heuvel's method significantly increases efficiency by reducing the cylinder estimation problem from a single five-dimensional Hough transform to a two-dimensional Hough transform followed by a three-dimensional Hough transform. Despite this, the method still requires considerable optimisation to be able to run on board drone hardware and, similar to RANSAC, the masking effect could affect results.

Neural networks are extremely versatile and this extends to perching point detection. A neural-network-based approach would require a network to be trained to recognise perches of suitable dimensions and inference would then need to be run on board the drone using this network. Although, the training process would require a large amount of data, solutions could be devised to expedite this process, such as by automatically creating large amounts of point cloud data with a computer program or expanding the size of a set of images through image manipulation techniques. Running inference on board the drone would also necessitate a separate processor, but lightweight options designed specifically for this purpose do exist such as the Nvidia Jetson and Coral boards [41][42]. The Supervised Primitive Fitting Network built by Li et al. uses three-dimensional point cloud data to determine object structure and is shown to be robust to noise. Training a similar model or transfer learning this model for a depth sensor that can be used in the drone design would likely result in good performance, but would be very computationally expensive. This is not necessarily a deal-breaker as the training could be performed on a separate computer, but the resulting model would need to be compatible with the processor used on board the drone, which is not guaranteed. Taking inspiration from Figueiredo et al. a neural network could be used as a pre-processing step for another algorithm, such as a Hough transform. This may help with some of the limitations of the other method, but could quickly become too large a computational load to run on board a small quadrotor, directly influencing the rate at which the algorithm can be run.

Considering the strengths and weaknesses of each of the approaches and the requirements for the perching point detection system, it was decided that a RANSAC algorithm is the most appropriate choice. Each of the approaches could be used to autonomously detect the relative location of suitable perching points if implemented correctly, hence they all meet the first requirement. It is in the second requirement, relating to the rate at which the algorithm can be run on the drone, that the RANSAC approach starts to rise above the others. This is the only approach already used in a comparable perching point detection system run on board a drone [36], and this offers the most definite proof possible of the feasibility of such a system. While true that Luo et al. built a system that ran a neural network on board their drone to land on cylindrical perches, this system did not actually determine the dimensions of the perch so would be unable to sufficiently meet the first requirement. The extra hardware needed for each of the approaches would likely also be similar. A depth camera capable of generating three-dimensional point clouds would be needed for each approach in addition to some form of extra processor to run the algorithm. Since, to the author's knowledge, no on-board implementations exist for neural-network or Hough-transform-based systems which can detect the relative locations of perching points of suitable dimensions, it is unfortunately not possible to make an informed decision on the exact computational hardware that would be required. Popek et al. did, however, show that the hardware does exist for RANSAC with their design. A final advantage of the RANSAC-based approach that is not directly reflected in the requirements is the ease of implementation. The ROS Point Cloud Library [43] would simplify the process of building a RANSAC-based system allowing extra time to focus on other aspects of the project.

4.3. Implementation of RANSAC for detection of long narrow features

The implementation of the perching point detection system should be based on the algorithm presented by von Frankenberg and Nokleby. This is one of the few systems that allows for the detection of not just cylindrical perches, but general long narrow features. The algorithm used in this paper can be broken down into eight steps, summarised here in text and shown below in Figure 4.2. The first step is to throttle the frame rate received by the depth camera to a rate that can be handled by the hardware. In the original setup, a frame rate of 10Hz was used. The second step is to down-sample the resolution of the point cloud. This is done using the Point Cloud Library and an input leaf size. After this, the point cloud is cropped to further reduce the number of points considered. This could simply mean taking a region in the center of the point cloud, or, more intelligently, taking a region-of-interest based on the location of previously detected features. The fourth step is line segment detection where RANSAC is used to detect line segments based on the inputs of line length, line width, minimum number of points lying on the line and number of iterations. In step five, an orientation check is performed where lines are disqualified if their angle with the horizontal plane is larger than the input maximum angle. This angle is defined according to the perch orientations that the UAV with equipped perching mechanism is capable of perching on. Next, gaps are checked for in the line by moving a small rectangular prism, the size of which is determined by the maximum diameter that the perching mechanism can grasp, over the length of the detected lines. If there are any parts of the line where no points are inside the prism, this is detected as a gap and the line is disqualified. Step 7 is the removal of all points which were located inside the prism in the following step. This is in preparation for the eighth and final step, the proximity check. Here a larger rectangular prism is moved across the length of the remaining lines. The size of this prism is determined by the size of the perched UAV and is used to ensure that there are no obstacles near the perch to obstruct perching. If any points are detected inside the prism, the landing point is disqualified. Note that this base form of the algorithm will be slightly modified when implemented, as is discussed in the future work section of the original paper. Cylindrical regions will replace the rectangular prisms in steps six and eight and the cropping will be performed based on regions of interest.

The hardware that required for this approach is a depth camera that is capable of generating three-dimensional point clouds and an extra on-board processor to run the algorithm. In von Frankeberg and Nokleby's paper a Microsoft Kinect camera is used to generate point clouds. This was not designed to be flown on board a lightweight drone, subsequently its form factor and weight would pose some issues. Alternative depth camera options do exist, such as the Tara USB 3.0 stereo camera and the Intel RealSense cameras [44][45]. Both options can be used with ROS to generate point clouds, with the latter having been used in the system presented by Popek et al. For the companion computer, Popek et al.'s system utilised the Mastermind 3b board equipped with a Intel i7 four core processor.

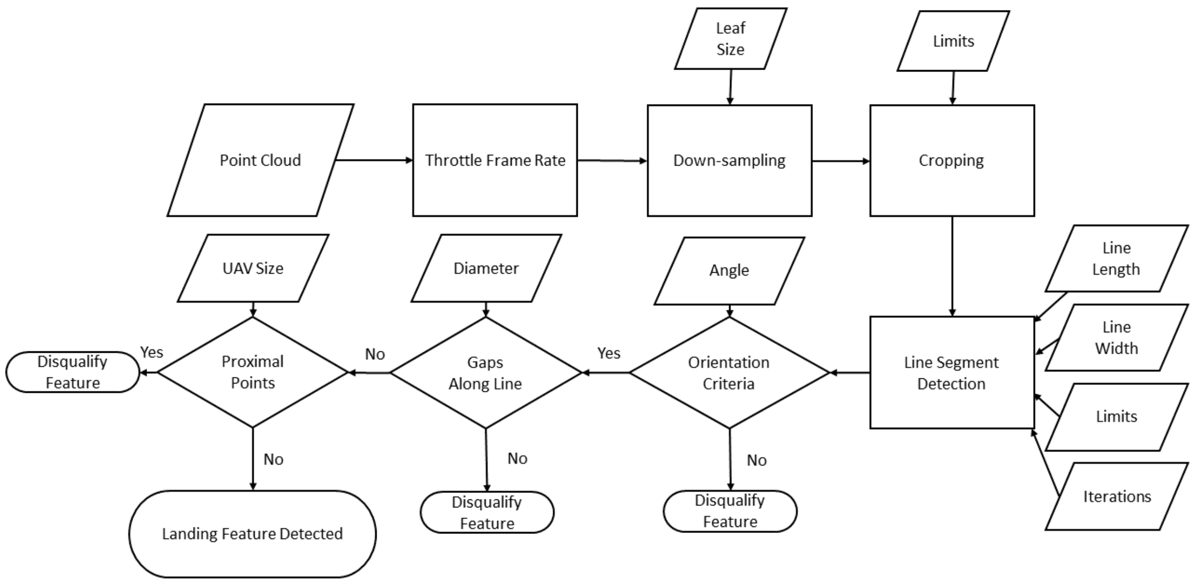


Figure 4.2: Flow chart of perching point detection algorithm [15]

This is the same kind of processor used by von Frankeberg and Nokleby to test their algorithm, so it can be said with a reasonable degree of certainty that this same setup would allow the algorithm to run on board a drone. Unfortunately, this board is no longer available so an alternative must be chosen. This alternative could be the Raspberry Pi 4, compatible with many flight controllers such as the PX4 [46]. This lightweight board supports ROS, but it is less powerful than the Intel i7. With correct tuning it should still be capable of running the perching point algorithm, albeit at a lower frame rate. In their tests, von Frankenberg and Nokleby were able to run their algorithm at 10Hz on the $3.4GHz$ processor. Some back-of-the-napkin, and admittedly over-simplified, calculations suggest that the $1.5GHz$ Raspberry Pi board may be able to run the same algorithm at a little over $4Hz$ [47], which should be fast enough for initial tests. The best choice of drone hardware is, of course, also affected by external factors such as price and availability, but this should suffice to show as proof that such a system is currently feasible.

5 Guidance, Navigation & Control

Although not a priority for this project, a smart choice for the guidance, navigation and control system is crucial to allow the quadrotor to perform as desired. This chapter discusses how this choice was made for each component of this subsystem. Once again, each section of this chapter answers a single research sub-question, mirroring the research process.

5.1. Guidance, navigation & control system requirements

Quadrotor guidance, control and navigation (GNC) systems are a well-established and well-0researched field. For a perching drone there is no need to re-invent the wheel in this area, but instead it is more appropriate to implement what has already been proven to work in a way that facilitates autonomous perching. For this, one must first determine the requirements for an autonomous control system for a perching drone. For any perching manoeuvre, the drone must first fly towards the location of the perch. The assumption is made that the environment is obstacle-free, apart from the perch structure, so the perch can be approached directly. Work has been done in building perching control systems where this is not the case, such as the perception-aware perching system presented by Paneque et al. [48], but this was deemed outside the scope of this project. The relative location of the perch is determined using the perching point detection system outlined in chapter 4, but this relative position can only be determined while the perch is in view of the depth camera, which will typically not be the case throughout the whole perching procedure. Here, the first requirement must be imposed on the navigation system of the quadrotor. Namely, that it allows the drone to continuously determine its own position and orientation in a fixed reference frame during flight. This means that once the relative position of the perch has been obtained from the perching point detection system, the perch's position in the fixed reference frame can also be determined. Since the perch is stationary, the perch no longer needs to be kept in frame and flying towards the perch now just involves manoeuvring towards this predetermined location. Finally, once the drone is near the perch, the choice of a passive perching mechanism means that the actual perching part of the manoeuvre is as simple as approaching the perch with a given velocity and orientation. This is determined by what is required for the perching mechanism to successfully grasp the perch structure. This last step is also simplified through the knowledge of the drone's and the perch's positions in the fixed frame at all times and it reduces the whole perching control problem to waypoint control.

Prior to the perching manoeuvre, it is also necessary to implement a control law for the drone to follow before it has detected any perching points. In this stage, the drone should be autonomously patrolling until it locates a suitable perch structure. There are many ways that one could choose to implement this patrol phase, such as following obstacle free trails [49] or trajectory optimisation elements to allow the drone to fit through geometrically constrained regions [50] as it looks for perches. The inclusion of such features would be invaluable in specific use-cases, but both are outside the scope of this paper where an obstacle-free environment has already been assumed. This assumption again greatly simplifies the problem as the patrol phase can now also be approached by flying directly to certain waypoints within the flight space until a perch is located. This could potentially miss perching structures outside the field of view of the quadrotor as it flies between waypoints. Another approach would be to have the quadrotor remain at a fixed location and rotate about its z-axis until it spots a suitable perching point. This has the disadvantage of potentially missing perches that are located too far away from the quadrotor for the perching point detection system to pick up. Combining the two approaches and having the quadrotor fly to fixed waypoints and then rotate helps to mitigate both of these shortcomings.

Both the patrol and perching phases of the mission can be accomplished by controlling the drone to certain waypoints with a given velocity and orientation. This gives the main requirement of the guidance system to be that it should generate a trajectory for the quadrotor between given waypoints with a desired final velocity and orientation. The control system should then control the drone's actual position using the continuous desired position and attitude of the quadrotor as an input. A secondary requirement for all parts of the GNC system is that they should be as computationally inexpensive as possible. This will leave as much processing power as possible available for the perching point detection system. These requirements are given in numbered form below for each :

1. The guidance system should generate a continuous trajectory between waypoints with given final quadrotor velocities and poses.
2. The navigation system should provide the actual position and attitude of the quadrotor continuously in a fixed reference frame.
3. The control system should use the continuous drone position and attitude inputs to allow control of the quadrotor to input waypoints with a given final quadrotor velocity and pose.
4. All GNC systems should be as computationally lightweight as possible.

5.2. Guidance strategy

The objective of the guidance system is trajectory planning: the generation of a path for the drone to follow between waypoints. This path is used to determine the desired position and pose at each timestep which can then be passed onto the controller. Following the requirements presented in section 5.1, a trajectory planner that can generate a trajectory between waypoints that have a desired attitude and velocity associated with them is required. Since the assumption has already been made that operation will occur in an obstacle-free environment and to best meet the computational cost requirement, the trajectory generation should be kept simple and computationally lightweight. Trajectory planning is often solved as an optimisation problem. Feasible trajectories that meet mission requirements can be found by, for example, minimising the position and attitude error at waypoints given the quadrotor dynamics. Due to the differentially flat dynamics of quadrotors, this optimisation problem can be reduced into a quadratic program that can be solved in real time [51]. Nonetheless, this optimisation step adds an extra layer of complexity to the process. Looking at nature, one can find a more direct approach to path planning, general tau theory [52]. Tau theory allows for the direct generation of trajectories in real-time based on a defined motion gap between the current and desired final state. This is perfect for application in perching where the aim is to reach the perch location with the velocity and orientation needed for the perching mechanism to grasp the perch structure. The good fit of general tau theory to UAV perching is also reflected in literature.

In his 1976 paper, David Lee presents his tau theory in the context of how drivers visually control their braking using information on the time-to-collision [53]. This has since been expanded to a bio-inspired general theory to describe animal movement. It states that control actions are taken with the goal of closing a motion-gap, the difference between the current state and a desired state. Tau, τ, is then defined as the time to close this motion gap. This theory has been used to describe all manner of animal behaviour, from the way drivers control their braking in Lee's own paper to how honey bees control their landing speed in Baird et al.'s paper [54]. More relevant to the area of drone perching, tau theory has also been used for trajectory generation. Zhang et al. present a framework for how tau theory can be used for UAV perching where they show how they can generate trajectories from a hovering or flight state to a perching point with a desired final position, pose and velocity [55]. For this they close multiple motion-gaps, for example the gaps in the UAVs horizontal and vertical position, at the same time using tau coupling. For this, the time to close both motion-gaps, tau, is kept constant so both gaps are closed at a constant rate relative to one another. This same tau-coupling principle can be used to couple the distance and angle motion-gaps. The freedom in the definition of the motion-gap already make tau theory trajectory generation a great fit for perching, but this does not consider the computational load. Fortunately, this algorithm can be very lightweight, as demonstrated by Wang et al., who were able to run a trajectory planning based on the same principles as Zhen et al.'s with addition optical-flow calculations at 30fps on a Raspberry-Pi 3B+ microcomputer [56]. This is extremely good performance and serves to further validate the use of general tau theory for autonomous perching trajectory planning.

5.3. Navigation system

The navigation system is required to continuously provide the quadrotor's attitude and position in a fixed reference frame. This information is then used as an input for the drone's controller. This allows the drone to determine the position and orientation of suitable perching points in fixed space using the perching point detection system's output of the position and orientation of these points relative to the drone. Many methods exist that allow for this objective to be achieved. Common on-board approaches consist of combining information from multiple sensors. An example of this can be found in the work done by Deilamsalehy and Havens where the movement of points of interest inside a camera frame is measured and combined with information from an IMU and a LiDAR sensor using an extended Kalman filter [57]. Doing this, they are able to estimate the pose of a platform in three-dimensional space. Another common example is the combination of IMU and global positioning system (GPS) data. This allows for a navigation system that overcomes the shortcomings of each of these individual sensors [58].

To meet both the navigation functionality and the computational load requirements, an off-board navigation system was deemed to be the best fit. This completely removes the computational load of estimating pose from the drone itself and means that this can instead be performed using an external computer. An off-board system also eliminates the need for extra sensors, such as a camera or GPS system, that would be used solely for navigation purposes. This results in a lower total drone weight and, consequently, better endurance and performance. Off-board position and orientation estimation can be performed using a motion capture system. This is an accurate and extremely common approach for indoor flight tests and has been used in much of the previously cited work: Nguyen et al. used motion capture in the testing of their soft-bodied perching drone to name but a single example [9]. There are a few disadvantages to using motion capture. Markers may need to be placed on the drone to allow the system to accurately track the drone and these could have some impact on performance, depending on the form they take. The much larger drawback is that this system only works inside the range of the motion capture system. This makes this setup unsuitable for flying outdoors in real-world environments where it will likely be impossible to accommodate a motion capture system. In these situations, a separate on-board navigation system will have to be implemented. This is definitely an area of interest for future work, but, for the current set of requirements, a motion capture system is the best fit.

5.4. Control strategy

At its most basic level, quadrotor control consists of controlling a quadrotor to follow a pre-determined trajectory as closely as possible. This is far from a trivial problem and must be broken down into three smaller steps: position control, attitude control and motor speed control. These controllers can be either linear or non-linear, as will be discussed later. A well-know general approach to this problem was presented by Mahony et al. [51] and the working principle presented there will be summarised again shortly here. Quadrotor control takes the form of several hierarchical nested feedback loops, with the position controller being the outermost loop and the motor controller being the innermost. Before the control process can even start a trajectory must be planned. This can becan be accomplished using general tau theory, as described in section 5.2. This provides a continuous desired trajectory position and attitude which are then input into the position and attitude controllers, respectively. The position controller then converts the error between the actual position and attitude, from the navigation system, and the desired trajectory position to inputs for both the motor controller and the attitude controller. The attitude controller takes both the inputs from the trajectory planner and the position controller to calculate the rotation needed to achieve the desired trajectory attitude and then converts this to a second control input for the motor controller. This innermost loop, the motor controller, takes both the control inputs from the position and the attitude controller and uses these to control the motor voltage of each of the rotors. This controls the amount of thrust generated by each rotor and, through the quadrotor's dynamics, the quadrotor's actual trajectory. To complete the loop, this is then input back into the position controller for the next timestep.

UAV controllers can be either linear or non-linear and both approaches have their own strengths and weaknesses. In their survey, Mo and Farid present an overview of various non-linear and adaptive techniques for UAV control [59]. Quadrotor dynamics are themselves non-linear and the linearisation of these dynamics necessary for linear controllers can lead to constricting the operating space of the quadrotor. Especially in the presence of unknown perturbations such as gusts, non-linear controllers can lead to better stability due to their more complete description of the dynamics. Non-linear controllers are also necessary for performing acrobatic aerial manoeuvres outside the bounds of a linear controller, as shown in Antal et al.'s paper where they developed a non-linear controller for backflipping small quadrotors [60]. One method to develop a non-linear controller is through the use of neural networks. This approach has the benefit of not requiring an exact model of the UAV dynamics, especially useful when these dynamics may be complicated due to the presence of an added perching mechanism. Network-based controllers have also been sshown to be extremely capable, for example the stability shown in Hwangbo et al.'s controller trained through reinforcement learning [61] or the acrobatics performed in Kaufmann et al.'s paper [62]. The simulation-to-reality gap can cause errors for these kinds of controllers, particularly when encountering scenarios not present in the training process. Another issue with network-based controllers is the computational cost of both training and running these networks. The training process of neural networks is infamous for its high cost, luckily this can mostly be performed off-board, but even running inference on board a drone can be expensive compared to other methods. This is the main drawback to non-linear controllers in general when compared to linear controllers. The linearisation of the dynamics reduces the computational cost of the controller. Additionally, more control tools are available for linear controllers which can aid in further optimising these controllers compared to their non-linear counterparts. For this reason, and taking the computational cost into account, a linear controller should be used if it can sufficiently meet the control requirement. Only if the dynamics of the quadrotor and perching mechanism prevent this requirement from being met, should a simple non-linear controller be implemented instead.

There is one additional requirement that one may want to impose on the GNC subsystem to expand the possible perching space of the drone that would require a non-linear controller. This is the capability to invert the drone so it can perch both above and below a perch with only a single perching mechanism. This would expand the perching space by allowing the drone to perch under certain structures where perching above is not possible due to obstructions in the environment. Yu and Wong present a framework for vision-based inverted perching using bi-directional thrust where this functionality can be seen [63]. Due to the modularity of this framework, the inversion manoeuvre presented here could be implemented in a separate design with minimal issues. It would, however, require both a non-linear controller and the hardware capable of generating thrust in two directions. It should also be noted that little to no adjustment would have to be made to the perching mechanism described in chapter 2 to allow it to work for perching both above and below perches. This is, of course, given that it has already been designed to grasp with a force greater than the drone weight. Since this functionality is not part of the scope of this research objective, it will not be included but this is again an interesting area for future work.

To conclude this chapter, a visual representation of the GNC system of the quadrotor is given below, in Figure 5.1

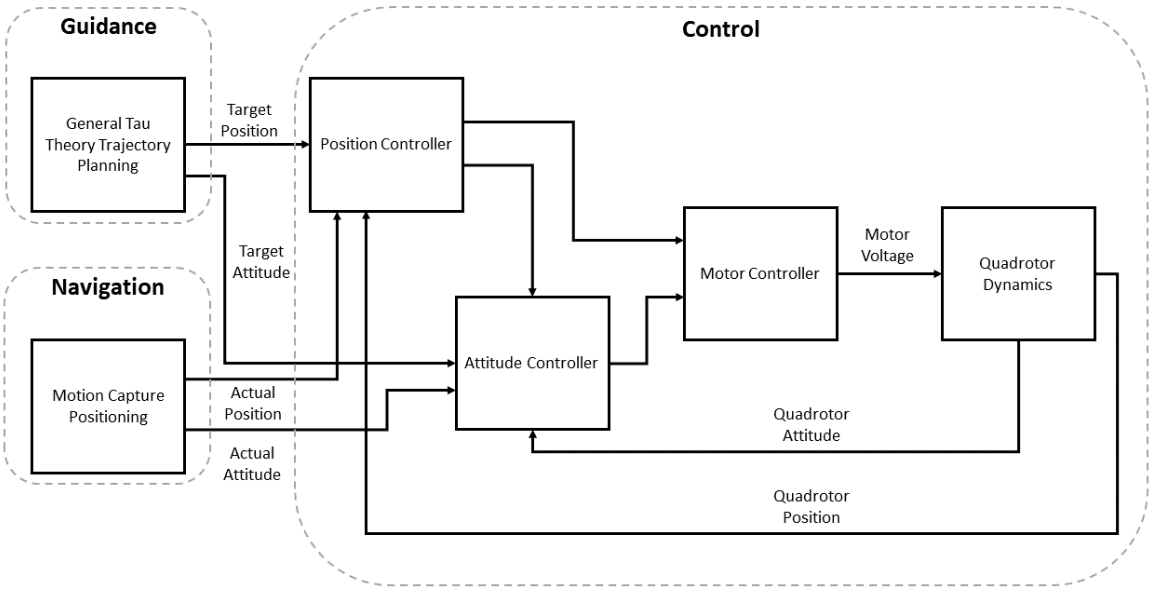


Figure 5.1: Block diagram of GNC system

6 Integration

This final chapter in the main body of the report discusses the integration of the different subsystems into a final and complete drone design. This is shown to be relatively straight forward as most of the subsystems are only connected to each other physically and do not directly communicate during operation. The final two sub-questions from the introduction are answered here, each in their own section.

6.1. Required hardware

The perching mechanism discussed in chapter 2 can almost be considered as a stand alone part of the drone design. Through its design it indirectly affects the design of other subsystems, such as the perching point detection system, but during operation it is not required to communicate with any other subsystems. Due to the passive design it does not require control inputs or access to computational power; it will automatically close upon contact with a perch. This means that the only hardware necessary for this part of the drone is the perching mechanism itself and it's connection to the quadrotor frame. The design principle for the mechanism consists solely of slapper bracelets covered with a non-slip material. How exactly this should be attached to the drone is difficult to say this early in the design process, but one could easily envision several possibilities. For example, a connector that allowed the slapper bracelets to be screwed to the frame could be 3D printed. This would allow for precise customisation of the orientation and position of the mechanism relative to the frame. An even simpler possible connector would be to simply zip-tie the mechanism to the frame. This approach would sacrifice some of the stability and flexibility of a custom 3D printed connector, but would be an extremely lightweight and quick to assemble option. Again, the exact design of this connector should be chosen later in the design process once both the mechanism and quadrotor frame have already been built but, suffice it to say there are many possible and easy to construct solutions.

The origami bumpers are a similarly isolated part of the drone, only connected to the rest of the drone via a physical attachment. The bumpers themselves can be cut from a single sheet of a lightweight material, for example cellulose acetate, and then folded to the correct shape. These will then need to be attached to the ends of each of the quadrotor arms so that they can protect the rotors. There will again be several possible ways of achieving this and the final decision should only be made after the bumpers and quadrotor have been assembled, but it is hard to imagine a more elegant approach to that taken by Dilaveroglu and Ozcan in their original design [14]. They designed the bumper in such a way that it could be mounted directly under the rotor and then fastened with a single screw. This approach should be used as inspiration for any final connection designs.

The perching point detection system's hardware is the first that cannot be considered in isolation. This subsystem requires a depth sensor and a companion computer to handle the extra processing. These will need to be connected to each other and to the GNS subsystem so that they can be used to provide control inputs for the drone. As mentioned in chapter 4, there are many options for both the depth camera, such as the Intel RealSense series, and the companion computer, such as the Raspberry Pi

4, but a definitive choice for these can be made once the perching point detection software has been finished. The chosen hardware should allow the software to run optimally. The sensor and companion computer will also need to be mounted to the quadrotor and the best method for this will again depend on the quadrotor frame, the sensor and computer hardware that is used. One possible solution is 3D printing mounting points, although printing a custom printed circuit board (PCB) is perhaps the more interesting approach. A PCB would allow for some of the wiring to be housed inside the board, cutting down on the external wiring that would otherwise be needed. The board could also be designed to include mounting points so it can be used as a connector. Printing custom boards through commercial companies can often only be done in bulk which may result in a slower construction process due to the delivery times. This could make testing several prototypes to make iterative changes a difficult and expensive process.

The hardware for the guidance, navigation and control system has two parts, the on-board hardware and the off-board hardware. The on-board component consists of the flight controller. An off-the-shelf flight controller that is compatible with the companion computer should be chosen for this. For example, if the Raspberry Pi 4 is used as the companion computer, the PX4 flight controller would be a good choice. This is a well known and commonly used flight controller leading to widespread availability and easy to find support. The PX4 also supports position and orientation estimation through motion capture meaning that it will be able to receive and process inputs from the off-board hardware [64]. This off-board hardware is, of course, the motion capture component of the GNC system. The choice of the motion capture system is not too important as long as it is compatible with the flight controller and able to estimate the quadrotor's position and orientation at a sufficiently high rate and resolution. Two well known options are VICON [65] and OptiTrack [66] and both of these have been widely used in drone applications.

The final hardware component is the quadrotor itself. The exact form that this quadrotor takes is not overly important as long as it can support all the other on-board hardware components. This means that it needs propellers and motors that are capable of generating enough thrust to fly with the combined weight of all the drone subsystems, that the battery is capable of supplying the electronics of the drone with enough power throughout the mission's duration and that the structural strength of the frame is sufficient to support the weight of all hardware that is to be mounted to it. The main focus of this research was the perching capabilities and the mechanical collision resilience measures, the exact form that the quadrotor takes was not a priority. This means that although it would be possible to design and build a custom quadrotor, in this case it would be more efficient to incorporate the other subsystems onto a preexisting off-the-shelf lightweight quadrotor body. This will minimise losses if a quadrotor is destroyed during testing as a new one could simply be purchased and will also allow more time to be spent on the other aspects of the design.

6.2. Subsystem integration

There are two ways that the different subsystems of the drone interact with one another. The first is in the design phase. Here the way in which each subsystem is designed may have an indirect impact on the design of another subsystem. An example of this would be that if the perching mechanism is designed to be able to grasp features of certain dimensions, this means that the perching point detection system needs to be able to detect features of those dimensions. Another example would be that if the origami bumpers are designed to protect the quadrotor for collisions up to a certain maximum velocity, the control system should ensure that the velocity of the drone is kept under that maximum value. These kinds of interactions can be found for each of the subsystems and are very specific to the exact implementation of each of the design principles.

The second and more general interactions between subsystems are those that occur during the drone's operation. This is, for example, the communication between the motion capture system and the flight controller or between the depth camera and the companion computer. An overview of these interactions is presented below in Figure 6.1. In this figure, it can be seen that the perching mechanism and origami bumpers do not directly interact with any of the other subsystems. While this is true if considering only communication between subsystems, these systems, along with each of the others, do all mechanically interface with the quadrotor body through their attachment to the frame. Since this is the same for each of the systems it was considered redundant to show it in the image.

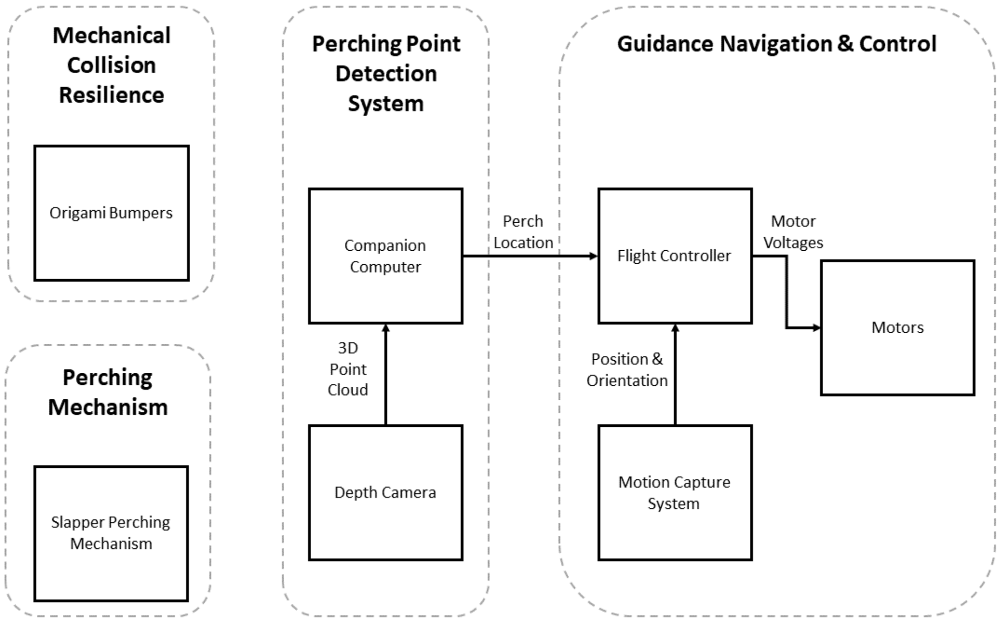


Figure 6.1: Interaction between subsystems during operation

7 Conclusions and Future Work

This literature review is concluded by revisiting the research objective and questions presented in the introduction. The aim of the research behind this review was to analyse the current perching drone literature in preparation for the design, construction and testing of a new drone concept that will be the focus of a master's thesis project. In this chapter, the plan for the continuation of this project is presented along with the conclusions of the literature review.

7.1. Conclusion

The main objective of this research project was to answer the following question: *How can a passive perching mechanism be combined with mechanical collision resilience to achieve perching on long narrow features with a lightweight quadrotor without prior knowledge of the location of suitable perching points?* To make the research process more manageable, this question was split into five sets of sub-questions. The first four of these each related to a separate subsystem and the final one related to how each of these systems can be combined in a complete design. The first three subsystems, the perching mechanism, the mechanical collision resilience measures and the perching point detection system, were the priorities for this research project. By combining state-of-the-art individual designs for each of these subsystems into a single design, this project makes a unique contributes to the current body of research. The GNC subsystem and quadrotor body were viewed as secondary interests as these are both well-developed research fields and to spend excessive time designing new approaches for either runs the risk of reinventing the wheel while taking time away from the design of other aspects. The goal of both these systems was to keep them simple but still capable enough to facilitate the other subsystems.

It was decided that a passive perching mechanism design is more suitable for this application than an active design. A passive mechanism results in a much simpler and lighter weight system as it does not require the use of servos for actuation or a control system capable of actively closing the mechanism at the correct time. To help with the design of this passive gripper, the following list of six requirements was made: the mechanism should firmly attach the UAV directly above or below the perch, the bistable mechanism should close on contact with the perching point, the mechanism should be robust to perch structure, the mechanism should be lightweight, the perching mechanism should be easy to construct and a mathematical model should exist for the perching mechanism. Three different perching mechanism designs concepts, claw grippers, helical grippers and slapper grippers, were then evaluated based on how well they fulfilled each of the requirements. The slapper grippers came out on top of this analysis and thus it was decided to adopt these design principles into the final perching mechanism design. For how this design might look, inspiration was taken from Nguyen et al.'s work where a similar perching mechanism was used [9]. This design should, however, be modified to use slapper bracelets instead of tape measure segments due to the ease of construction. The basis for the design should be one or two of these segments aligned in parallel to each other and covered in a non-slip surface. This design is best suited to meet all of the perching mechanism requirements.

To justify the inclusion of mechanical collision resilience an analysis was first performed to see where this would be of use and to compare it to alternatives such as an obstacle avoidance system. Collision resilience was found to be extremely beneficial to a perching drone in the real-world where the drone may not be able to reliably navigate through the obstacle-dense environments where perches tend to be located without collisions. Mechanical measures are more suitable than software-based obstacle avoidance due to a higher reliability, no computational load and not limiting flight speeds due to an algorithm only being capable of operating at a certain rate. Exoskeletons, soft drone bodies and rotor bumpers were each evaluated to determine the most suitable form of collision resilience for a lightweight perching quadrotor. Soft rotor bumpers, despite their somewhat smaller coverage of impact approach angles, were deemed the most appropriate due to their small form factor and low impact on the aerodynamics of the drone. To investigate how to implement soft rotor bumpers, inspiration was taken from the principles of origami structures. These structure can offer great collision resilience while being lightweight and easy to construct. For the exact implementation of these techniques into drone bumper design, Dilaveroglu and Ozcan's work was chosen as the best approach due to the simplicity and extremely low profile [14]. Finally, a discussion on how to test the collision resilience measures was presented. Bench tests such as drop tests or pendulum tests should be combined with in-flight tests to properly evaluate the performance of the collision resilience measures and determine for what range of impacts they allow recovery.

The goal of the perching point detection system is to identify structures in the environment on which the quadrotor could perch. To help with the design of this system, it was decided that it must meet the following three requirements: the system should detect the relative position of perching points of suitable dimensions, the system should run fast enough on board the drone to be used as a control input and the additional weight required by the system should be kept as low as possible. Much like for the other subsystems, several possible approaches from the literature were analysed and evaluated based on how well they fulfilled these requirements. These approaches were RANSAC-based methods, Hough-transform-based methods and neural-network-based methods. Of these three, a RANSAC-based approach was found to be the best fit due to it being the only method that has already been shown to work on board a drone for perching point detection. It also allows for detection of a wider range of possible perches, not just limited to cylinders. A final point that the RANSAC approach has in its favour is the fact that libraries already exist that will greatly simplify the system's development. Finally, the implementation of a RANSAC-based perching point detection system was discussed both from a software and hardware perspective. The algorithm presented by von Frankenberg and Nokleby [15] was chosen as this allows for detection of the widest range of perching points. To run this algorithm a depth camera and companion computer would be required.

As previously stated, innovation was not considered a priority in the design of the GNC system. The GNC subsystem consists of three parts: guidance, determining the path that the drone should follow; navigation, determining where the drone is in space; and control, the issuing of commands to the drone to ensure that its position and the desired position along the trajectory are as closely aligned as possible. The following set of requirements were imposed upon the GNC system: The guidance system should generate a continuous trajectory between waypoints with given final quadrotor velocities and poses, the navigation system should provide the actual position and attitude of the quadrotor continuously in a fixed reference frame, the control system should use the continuous drone position and attitude inputs to allow control of the quadrotor to input waypoints with a given final quadrotor velocity and pose and all GNC systems should be as computationally lightweight as possible. It was decided to base the guidance portion of the GNC system on general tau theory. The light computational load and the fact that it has been proven to work for perching makes it an ideal fit for a lightweight perching quadrotor. The navigation will be handled mostly off board the drone using a motion capture system. These systems can provide extremely accurate continuous position and orientation information and will reduce the on-board computational load. Despite this limiting the possible mission space, this is considered a worthwhile trade-off, especially when viewed through the lens of the GNC requirements. Finally, a linear flight controller will be used for the control segment. These controllers do not perform as well as non-linear controllers in the presence of disturbances or while performing aerial acrobatics, but the lower computational load make them the best option for the purposes of an initial perching drone concept where neither of these things will be an issue. A non-linear controller could be an area of interest for further work, however, as it could allow for both inverted and conventional perching using just a single perching mechanism.

To conclude this paper, an overview of the hardware needed to combine all these subsystems into a complete design along with the interactions between each of the separate subsystems is presented. The hardware includes all the sensors and processors necessary for each subsystem, the attachment methods of each subsystem to the drone frame and the quadrotor itself. The theme for each hardware component is that as long as compatibility is not a problem, everything should be kept as easily available and simple as possible. This will simplify construction of the drone which is especially relevant during design validation, since quadrotor testing is infamously destructive. The components interact with each other in both the design phase, where the limitations of one system might inform the requirements of another, and in the operation phase, where different subsystems will physically and electronically interact with one another. Each subsystem interacts with at least the drone body through its hardware connection, but only certain systems communicate with other systems during operation. In fact, the only communication between subsystems is between the perching point detection system and the GNC system. Both the perching mechanism and collision resilience features are completely passive and thus do not require any active control. With this final overview the final set of sub-questions is answered and the research objective complete. The next step is to validate the findings presented in this paper through the construction of a real-world lightweight perching quadrotor. How this will be approached is discussed below.

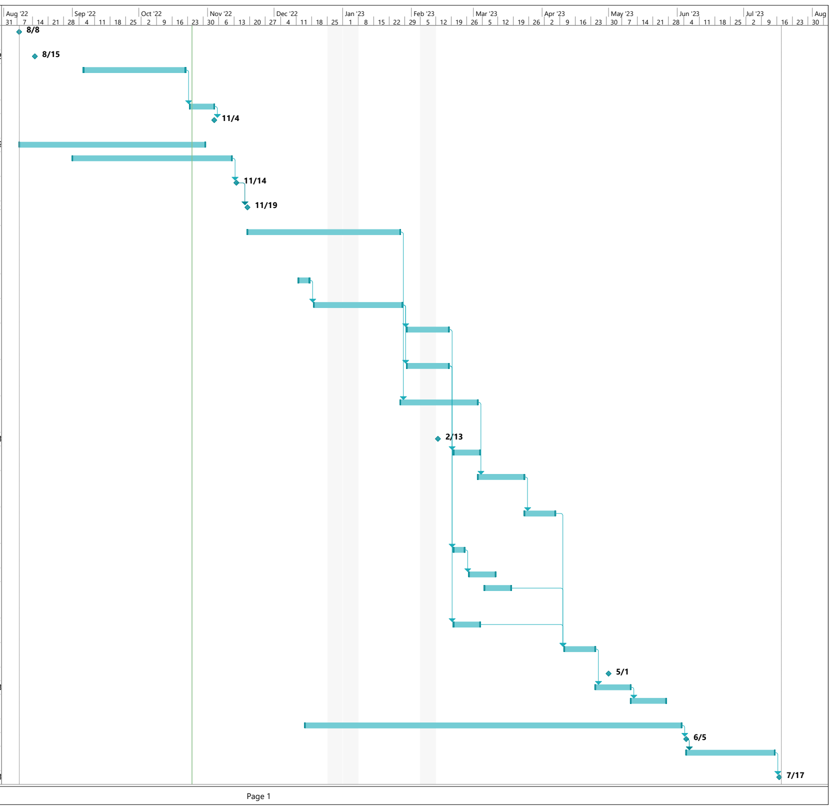
7.2. Project plan

This literature review was conducted as part of a master's thesis in Control and Simulation at the aerospace faculty of the Delft University of Technology. The ultimate objective of this thesis is to design, construct and test a mechanically collision-resilient quadrotor capable of autonomous perching. The first phase of this project consisted of a thorough analysis of the current literature in this area and the findings of this research were presented in this document. These findings will all be verified and validated in the later phases of this thesis project when the proposed design principles will be implemented to create, construct and test a final drone design. A plan for these next steps has already been created to ensure that the project can be completed within the nominal period of study.

An overview of this planning is available in the Gantt chart included below. Here all the tasks that will need to be completed over the duration of the project and the deadlines that must be met during the completion of a Control and Simulation master's thesis are shown. These deadlines were decided upon ahead of time with the supervisor of this project, Salua Hamaza. Holidays have been included in this planning since these will inevitably cause certain tasks to be delayed. These holidays align with those in the academic calendar of the university for the academic year 2022/2023 [67].

There are several similar types of tasks that take place multiple times throughout the project. The first of these is the familiarisation stage. This occurs at the earliest stage of the design process for each subsystem and refers to building a base level of experience with the equipment and techniques necessary for later stages. It was decided to allocate a longer period, either one or two months, for these tasks since more time spent refining these base skills will ensure that future tasks can progress more smoothly. These tasks are planned in parallel with other tasks, which is also why it is useful to reserve additional time for them. The second type of recurring task is the initial design and construction stage. This refers to the design and construction of the first version of a component or subsystem to the point where it can be tested. These tasks can only begin after the initial familiarisation stage is complete and differing amounts of time are allocated based on perceived complexity. Once the familiarisation and design and construction tasks have been completed, each subsystem must be tested. These tests are very briefly summarised in the Gantt chart, with a detailed description available in the full project plan document, also completed for this thesis [68]. The tests are dependent on the initial designs and often also on other tests. These tests sometimes depend on an integration stage where different subsystems are combined so that their combined functionality can be tested. Two weeks has been allocated for each of the tests. The tests themselves will likely not take a full two weeks, but time is also reserved for result analysis and possible adjustments to the test setup. Finally, five months is allocated over the duration of the project for the writing of the thesis with an additional twenty days for presentation preparation. These are both required by the university and will be the main deliverables of the thesis project.

ID	ⓘ	Task Mode	Task Name	Duration	Start	Finish
1		📌	Start Graduation Project	0 days	Mon 8/8/22	Mon 8/8/22
2		📌	Kick-off Session	0 days	Mon 8/15/22	Mon 8/15/22
3		📌	Watch Research Methodologies Videos	6.8 wks	Tue 9/6/22	Fri 10/21/22
4		📌	Write Project Plan	9 days	Mon 10/24/2	Thu 11/3/22
5		📌	Project Plan Deadline	0 days	Fri 11/4/22	Fri 11/4/22
6		📌	Read Literature	61 days	Mon 8/8/22	Sun 10/30/22
7		📌	Write Literature Review	52 days	Thu 9/1/22	Fri 11/11/22
8		📌	Literature Study Deadline	0 days	Mon 11/14/22	Mon 11/14/22
9		📌	Literature Study Presentation	0 days	Sat 11/19/22	Sat 11/19/22
10		📌	Familiarisation ROS and C++ for Quadrotor Control and Simulation	2 mons	Sat 11/19/22	Thu 1/26/23
11		📌	MAVLab Safety Instruction	1 wk	Mon 12/12/22	Fri 12/16/22
12		📌	Familiarisation Basic Quadrotor Hardware	1 mon	Mon 12/19/22	Fri 1/27/23
13		📌	Perching Mechanism Initial Design and Construction	2 wks	Mon 1/30/23	Fri 2/17/23
14		📌	Bumper Initial Design and Construction	2 wks	Mon 1/30/23	Fri 2/17/23
15		📌	Perching Point Detection Algorithm Coding	1 mon	Fri 1/27/23	Thu 3/2/23
16		📌	Mid Term Review	0 days	Mon 2/13/23	Mon 2/13/23
17		📌	Bench Test Perching Mechanism	2 wks	Mon 2/20/23	Fri 3/3/23
18		📌	Integration Perching Point Detection into Quadrotor	3 wks	Fri 3/3/23	Thu 3/23/23
19		📌	Stationary Tests Perching Point Algorithm	2 wks	Fri 3/24/23	Thu 4/6/23
20		📌	Integration Bumpers into Quadrotor	1 wk	Mon 2/20/23	Fri 2/24/23
21		📌	Initial Test Bumpers	2 wks	Mon 2/27/23	Fri 3/10/23
22		📌	Integration Perching Mechanism into Quadrotor	2 wks	Mon 3/6/23	Fri 3/17/23
23		📌	Flight Tests Collision Resilience	2 wks	Mon 2/20/23	Fri 3/3/23
24		📌	Flight Test Perching Point Detection	2 wks	Tue 4/11/23	Mon 4/24/23
25		📌	Green Light Review	0 days	Mon 5/1/23	Mon 5/1/23
26		📌	Flight Test Perching	2 wks	Tue 4/25/23	Wed 5/10/23
27		📌	Flight Test Complete System	2 wks	Thu 5/11/23	Fri 5/26/23
28		📌	Thesis Writing	5 mons	Thu 12/15/22	Fri 6/2/23
29		📌	Hand-in Thesis	0 days	Mon 6/5/23	Mon 6/5/23
30		📌	Thesis Defence Presentation	30 days	Mon 6/5/23	Fri 7/14/23
31		📌	Thesis Defence	0 days	Mon 7/17/23	Mon 7/17/23



Page 1

Bibliography

[1] William R. T. Roderick, Mark R. Cutkosky, and David Lentink. “Touchdown to take-off: at the interface of flight and surface locomotion”. In: *Interface Focus* 7.1 (Feb. 6, 2017), p. 20160094. ISSN: 2042-8898, 2042-8901. DOI: 10.1098/rsfs.2016.0094. URL: https://royalsocietypublishing.org/doi/10.1098/rsfs.2016.0094 (visited on 09/06/2022).

[2] Alexis Lussier Desbiens, Alan T Asbeck, and Mark R Cutkosky. “Landing, perching and taking off from vertical surfaces”. In: *The International Journal of Robotics Research* 30.3 (Mar. 2011), pp. 355–370. ISSN: 0278-3649, 1741-3176. DOI: 10.1177/0278364910393286. URL: http://journals.sagepub.com/doi/10.1177/0278364910393286 (visited on 09/16/2022).

[3] Morgan Pope. *Microspines make it easy for drones to perch on walls and ceilings*. May 2016. URL: https://spectrum.ieee.org/microspines-make-it-easy-for-drones-to-perch-on-walls-and-ceilings.

[4] Hideyuki Tsukagoshi et al. “Aerial manipulator with perching and door-opening capability”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015 IEEE International Conference on Robotics and Automation (ICRA). Seattle, WA, USA: IEEE, May 2015, pp. 4663–4668. ISBN: 978-1-4799-6923-4. DOI: 10.1109/ICRA.2015.7139845. URL: http://ieeexplore.ieee.org/document/7139845/ (visited on 09/16/2022).

[5] Wanchao Chi et al. “An optimized perching mechanism for autonomous perching with a quadrotor”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014 IEEE International Conference on Robotics and Automation (ICRA). Hong Kong, China: IEEE, May 2014, pp. 3109–3115. ISBN: 978-1-4799-3685-4. DOI: 10.1109/ICRA.2014.6907306. URL: http://ieeexplore.ieee.org/document/6907306/ (visited on 09/17/2022).

[6] Ryan Kitchen et al. “Design and Evaluation of a Perching Hexacopter Drone for Energy Harvesting from Power Lines”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Las Vegas, NV, USA: IEEE, Oct. 24, 2020, pp. 1192–1198. ISBN: 978-1-72816-212-6. DOI: 10.1109/IROS45743.2020.9341100. URL: https://ieeexplore.ieee.org/document/9341100/ (visited on 09/17/2022).

[7] Pu Xie et al. “A Bio-inspired UAV Leg-Foot Mechanism for Landing, Grasping and Perching Tasks”. In: *AIAA Atmospheric Flight Mechanics Conference*. AIAA Atmospheric Flight Mechanics Conference. Kissimmee, Florida: American Institute of Aeronautics and Astronautics, Jan. 5, 2015. ISBN: 978-1-62410-340-7. DOI: 10.2514/6.2015-1689. URL: https://arc.aiaa.org/doi/10.2514/6.2015-1689 (visited on 09/17/2022).

[8] Courtney E. Doyle et al. “An Avian-Inspired Passive Mechanism for Quadrotor Perching”. In: *IEEE/ASME Transactions on Mechatronics* 18.2 (Apr. 2013), pp. 506–517. ISSN: 1083-4435, 1941-014X. DOI: 10.1109/TMECH.2012.2211081. URL: http://ieeexplore.ieee.org/document/6291791/ (visited on 09/06/2022).

[9] Pham H. Nguyen et al. *A Soft-Bodied Aerial Robot for Collision Resilience and Contact-Reactive Perching*. May 2, 2022. arXiv: 2204.13155[cs, eess]. URL: http://arxiv.org/abs/2204.13155 (visited on 07/31/2022).

[10] Christian Geckeler and Stefano Mintchev. “Bistable Helical Origami Gripper for Sensor Placement on Branches”. In: *Advanced Intelligent Systems* (Aug. 15, 2022), p. 2200087. ISSN: 2640-4567, 2640-4567. DOI: 10.1002/aisy.202200087. URL: https://onlinelibrary.wiley.com/doi/10.1002/aisy.202200087 (visited on 09/16/2022).

[11] Ricardo de Azambuja et al. “When Being Soft Makes You Tough: A Collision-Resilient Quadcopter Inspired by Arthropods’ Exoskeletons”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022 IEEE International Conference on Robotics and Automation (ICRA). Philadelphia, PA, USA: IEEE, May 23, 2022, pp. 7854–7860. ISBN: 978-1-72819-681-7. DOI: `10.1109/ICRA46639.2022.9811841`. URL: `https://ieeexplore.ieee.org/document/9811841/` (visited on 10/08/2022).

[12] Stefano Mintchev, Sebastien de Rivaz, and Dario Floreano. “Insect-Inspired Mechanical Resilience for Multicopters”. In: *IEEE Robotics and Automation Letters* 2.3 (July 2017), pp. 1248–1255. ISSN: 2377-3766, 2377-3774. DOI: `10.1109/LRA.2017.2658946`. URL: `http://ieeexplore.ieee.org/document/7833174/` (visited on 07/31/2022).

[13] Pooya Sareh et al. “Rotorigami: A rotary origami protective system for robotic rotorcraft”. In: *Science Robotics* 3.22 (Sept. 19, 2018), eaah5228. ISSN: 2470-9476. DOI: `10.1126/scirobotics.aah5228`. URL: `https://www.science.org/doi/10.1126/scirobotics.aah5228` (visited on 10/08/2022).

[14] Levent Dilaveroglu and Onur Ozcan. “MiniCoRe: A Miniature, Foldable, Collision Resilient Quadcopter”. In: *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. 2020 3rd IEEE International Conference on Soft Robotics (RoboSoft). New Haven, CT, USA: IEEE, May 2020, pp. 176–181. ISBN: 978-1-72816-570-7. DOI: `10.1109/RoboSoft48309.2020.9115993`. URL: `https://ieeexplore.ieee.org/document/9115993/` (visited on 08/12/2022).

[15] Florentin von Frankenberg and Scott Nokleby. “Detection of Long Narrow Landing Features for Autonomous UAV Perching”. In: *2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*. 2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON). Vancouver, BC, Canada: IEEE, Nov. 4, 2020, pp. 0565–0570. ISBN: 978-1-72818-416-6. DOI: `10.1109/IEMCON51383.2020.9284856`. URL: `https://ieeexplore.ieee.org/document/9284856/` (visited on 08/06/2022).

[16] Tahir Rabbani. “EFFICIENT HOUGH TRANSFORM FOR AUTOMATIC DETECTION OF CYLINDERS IN POINT CLOUDS”. In: (2005), p. 6.

[17] Rui Figueiredo et al. “A robust and efficient framework for fast cylinder detection”. In: *Robotics and Autonomous Systems* 117 (July 2019), pp. 17–28. ISSN: 09218890. DOI: `10.1016/j.robot.2019.04.002`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0921889017308710` (visited on 08/15/2022).

[18] Riccardo Spica, Paolo Robuffo Giordano, and Francois Chaumette. “Active structure from motion for spherical and cylindrical targets”. In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. 2014 IEEE International Conference on Robotics and Automation (ICRA). Hong Kong, China: IEEE, May 2014, pp. 5434–5440. ISBN: 978-1-4799-3685-4. DOI: `10.1109/ICRA.2014.6907658`. URL: `http://ieeexplore.ieee.org/document/6907658/` (visited on 09/03/2022).

[19] Justin Thomas et al. “Visual Servoing of Quadrotors for Perching by Hanging From Cylindrical Objects”. In: *IEEE Robotics and Automation Letters* 1.1 (Jan. 2016), pp. 57–64. ISSN: 2377-3766, 2377-3774. DOI: `10.1109/LRA.2015.2506001`. URL: `http://ieeexplore.ieee.org/document/7347389/` (visited on 09/01/2022).

[20] Cai Luo, Leijian Yu, and Peng Ren. “A Vision-Aided Approach to Perching a Bioinspired Unmanned Aerial Vehicle”. In: *IEEE Transactions on Industrial Electronics* 65.5 (May 2018), pp. 3976–3984. ISSN: 0278-0046, 1557-9948. DOI: `10.1109/TIE.2017.2764849`. URL: `https://ieeexplore.ieee.org/document/8074761/` (visited on 08/08/2022).

[21] Kaiyu Hang et al. “Perching and resting—A paradigm for UAV maneuvering with modularized landing gears”. In: *Science Robotics* 4.28 (Mar. 13, 2019), eaau6637. ISSN: 2470-9476. DOI: `10.1126/scirobotics.aau6637`. URL: `https://www.science.org/doi/10.1126/scirobotics.aau6637` (visited on 08/06/2022).

[22] Raphael Zufferey et al. “HOW ORNITHOPTERS CAN PERCH AUTONOMOUSLY ON A BRANCH”. In: (), p. 22.

[23] Haijie Zhang, Jiefeng Sun, and Jianguo Zhao. “Compliant Bistable Gripper for Aerial Perching and Grasping”. In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019 International Conference on Robotics and Automation (ICRA). Montreal, QC, Canada: IEEE, May 2019, pp. 1248–1253. ISBN: 978-1-5386-6027-0. DOI: 10.1109/ICRA.2019.8793936. URL: https://ieeexplore.ieee.org/document/8793936/ (visited on 09/06/2022).

[24] E Kebadze, S.D Guest, and S Pellegrino. “Bistable prestressed shell structures”. In: *International Journal of Solids and Structures* 41.11 (June 2004), pp. 2801–2820. ISSN: 00207683. DOI: 10.1016/j.ijsolstr.2004.01.028. URL: https://linkinghub.elsevier.com/retrieve/pii/S0020768304000460 (visited on 09/11/2022).

[25] S.D Guest and S Pellegrino. “Analytical models for bistable cylindrical shells”. In: *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 462.2067 (Mar. 8, 2006), pp. 839–854. ISSN: 1364-5021, 1471-2946. DOI: 10.1098/rspa.2005.1598. URL: https://royalsocietypublishing.org/doi/10.1098/rspa.2005.1598 (visited on 09/11/2022).

[26] Zi Chen et al. “Nonlinear Geometric Effects in Mechanical Bistable Morphing Structures”. In: *PHYSICAL REVIEW LETTERS* (2012), p. 5.

[27] Seung-Won Kim et al. “Towards a bio-mimetic flytrap robot based on a snap-through mechanism”. In: *2010 3rd IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*. EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob 2010). Tokyo, Japan: IEEE, Sept. 2010, pp. 534–539. ISBN: 978-1-4244-7708-1. DOI: 10.1109/BIOROB.2010.5627994. URL: https://ieeexplore.ieee.org/document/5627994 (visited on 09/11/2022).

[28] Abdulla Al-Kaff et al. “Survey of computer vision algorithms and applications for unmanned aerial vehicles”. In: *Expert Systems with Applications* 92 (Feb. 2018), pp. 447–463. ISSN: 09574174. DOI: 10.1016/j.eswa.2017.09.033. URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417417306395 (visited on 10/08/2022).

[29] Young K. Kwag and Chul H. Chung. “UAV based collision avoidance radar sensor”. In: *2007 IEEE International Geoscience and Remote Sensing Symposium*. 2007 IEEE International Geoscience and Remote Sensing Symposium. ISSN: 2153-7003. July 2007, pp. 639–642. DOI: 10.1109/IGARSS.2007.4422877.

[30] Adam Klaptocz et al. “Euler spring collision protection for flying robots”. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2013). Tokyo: IEEE, Nov. 2013, pp. 1886–1892. ISBN: 978-1-4673-6358-7 978-1-4673-6357-0. DOI: 10.1109/IROS.2013.6696606. URL: http://ieeexplore.ieee.org/document/6696606/ (visited on 10/08/2022).

[31] Adrien Briod et al. “A Collision-resilient Flying Robot: A Collision-resilient Flying Robot”. In: *Journal of Field Robotics* 31.4 (July 2014), pp. 496–509. ISSN: 15564959. DOI: 10.1002/rob.21495. URL: https://onlinelibrary.wiley.com/doi/10.1002/rob.21495 (visited on 10/08/2022).

[32] Droneshop.nl. *DJI Mavic Mini - 360° Propeller Guard*. URL: https://droneshop.nl/dji-mavic-mini-360-graden-propeller-guard.

[33] Nicholas Turner, Bill Goodwine, and Mihir Sen. “A review of origami applications in mechanical engineering”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 230.14 (Aug. 1, 2016). Publisher: IMECHE, pp. 2345–2362. ISSN: 0954-4062. DOI: 10.1177/0954406215597713. URL: https://doi.org/10.1177/0954406215597713 (visited on 10/14/2022).

[34] Cagdas D. Onal, Robert J. Wood, and Daniela Rus. “Towards printable robotics: Origami-inspired planar fabrication of three-dimensional mechanisms”. In: *2011 IEEE International Conference on Robotics and Automation*. 2011 IEEE International Conference on Robotics and Automation. ISSN: 1050-4729. May 2011, pp. 4608–4613. DOI: 10.1109/ICRA.2011.5980139.

[35] Dadong Fan et al. “Quadrotor UAV: Collision Resilience Behaviors”. In: *IEEE Transactions on Aerospace and Electronic Systems* (2022). Conference Name: IEEE Transactions on Aerospace and Electronic Systems, pp. 1–12. ISSN: 1557-9603. DOI: 10.1109/TAES.2022.3211479.

[36] Katie M. Popek et al. “Autonomous Grasping Robotic Aerial System for Perching (AGRASP)”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Madrid: IEEE, Oct. 2018, pp. 1–9. ISBN: 978-1-5386-8094-0. DOI: 10.1109/IROS.2018.8593669. URL: https://ieeexplore.ieee.org/document/8593669/ (visited on 07/31/2022).

[37] Lingxiao Li et al. “Supervised Fitting of Geometric Primitives to 3D Point Clouds”. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. 2019, pp. 2652–2660. URL: https://openaccess.thecvf.com/content_CVPR_2019/html/Li_Supervised_Fitting_of_Geometric_Primitives_to_3D_Point_Clouds_CVPR_2019_paper.html (visited on 10/31/2022).

[38] Hoseong Seo, Suseong Kim, and H. Jin Kim. “Aerial grasping of cylindrical object using visual servoing based on stochastic model predictive control”. In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017 IEEE International Conference on Robotics and Automation (ICRA). Singapore, Singapore: IEEE, May 2017, pp. 6362–6368. ISBN: 978-1-5090-4633-1. DOI: 10.1109/ICRA.2017.7989751. URL: http://ieeexplore.ieee.org/document/7989751/ (visited on 09/01/2022).

[39] *Using RANSAC for estimating geometric transforms in computer vision*. 2022. URL: https://nl.mathworks.com/discovery/ransac.html.

[40] Peter Rousseeuw and Annick Leroy. *Robust Regression and Outlier Detection.* Vol. 152. Canada: John Wiley & Sons, Nov. 2, 1987. 341 pp. URL: https://www.researchgate.net/profile/Peter-Rousseeuw/publication/284537526_Robust_Regression_and_Outlier_Detection/links/60fbc1521e95fe241a84161a/Robust-Regression-and-Outlier-Detection.pdf.

[41] *Nvidia embedded systems for next-Gen Autonomous Machines*. 2022. URL: https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/.

[42] *Coral Products*. 2022. URL: https://coral.ai/products/.

[43] *Point Cloud Library*. 2022. URL: https://pointclouds.org/.

[44] *Tara - USB 3.0 Stereo Camera*. 2022. URL: https://www.e-consystems.com/3D-USB-stereo-camera.asp.

[45] *Stereo depth solutions from Intel RealSense*. Oct. 2022. URL: https://www.intelrealsense.com/stereo-depth/.

[46] *Companion computer for Pixhawk Series*. Sept. 2022. URL: https://docs.px4.io/main/en/companion_computer/pixhawk_companion.html.

[47] *Raspberry Pi 4 Model B Specifications*. 2022. URL: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/.

[48] Julio L. Paneque et al. “Perception-Aware Perching on Powerlines With Multirotors”. In: *IEEE Robotics and Automation Letters* 7.2 (Apr. 2022), pp. 3077–3084. ISSN: 2377-3766, 2377-3774. DOI: 10.1109/LRA.2022.3145514. URL: https://ieeexplore.ieee.org/document/9691874/ (visited on 07/31/2022).

[49] Nikolai Smolyanskiy et al. “Toward low-flying autonomous MAV trail navigation using deep neural networks for environmental awareness”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Vancouver, BC: IEEE, Sept. 2017, pp. 4241–4247. ISBN: 978-1-5386-2682-5. DOI: 10.1109/IROS.2017.8206285. URL: http://ieeexplore.ieee.org/document/8206285/ (visited on 08/13/2022).

[50] Zhepei Wang et al. “Geometrically Constrained Trajectory Optimization for Multicopters”. In: *IEEE Transactions on Robotics* (2022), pp. 1–10. ISSN: 1552-3098, 1941-0468. DOI: 10.1109/TRO.2022.3160022. URL: https://ieeexplore.ieee.org/document/9765821/ (visited on 07/31/2022).

[51] Robert Mahony, Vijay Kumar, and Peter Corke. “Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor”. In: *IEEE Robotics & Automation Magazine* 19.3 (Sept. 2012), pp. 20–32. ISSN: 1070-9932, 1558-223X. DOI: `10.1109/MRA.2012.2206474`. URL: `https://ieeexplore.ieee.org/document/6289431/` (visited on 09/04/2022).

[52] David N Lee et al. “Lee’s 1976 Paper”. In: *Perception* 38.6 (Jan. 1, 2009). Publisher: SAGE Publications Ltd STM, pp. 837–858. ISSN: 0301-0066. DOI: `10.1068/pmklee`. URL: `https://doi.org/10.1068/pmklee` (visited on 11/08/2022).

[53] David N Lee. “A Theory of Visual Control of Braking Based on Information about Time-to-Collision”. In: *Perception* 5.4 (Dec. 1976), pp. 437–459. ISSN: 0301-0066, 1468-4233. DOI: `10.1068/p050437`. URL: `http://journals.sagepub.com/doi/10.1068/p050437` (visited on 11/08/2022).

[54] Emily Baird et al. “A universal strategy for visually guided landing”. In: *Proceedings of the National Academy of Sciences* 110.46 (Nov. 12, 2013), pp. 18686–18691. ISSN: 0027-8424, 1091-6490. DOI: `10.1073/pnas.1314311110`. URL: `https://pnas.org/doi/full/10.1073/pnas.1314311110` (visited on 08/10/2022).

[55] Zhen Zhang, Pu Xie, and Ou Ma. “Bio-inspired trajectory generation for UAV perching”. In: *2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. 2013 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM). Wollongong, NSW: IEEE, July 2013, pp. 997–1002. ISBN: 978-1-4673-5320-5 978-1-4673-5319-9. DOI: `10.1109/AIM.2013.6584224`. URL: `http://ieeexplore.ieee.org/document/6584224/` (visited on 08/10/2022).

[56] Biao Wang et al. “Autonomous deck landing of a vertical take-off and landing unmanned aerial vehicle based on the tau theory”. In: *Transactions of the Institute of Measurement and Control* (July 13, 2022). Publisher: SAGE Publications Ltd STM, p. 01423312221104424. ISSN: 0142-3312. DOI: `10.1177/01423312221104424`. URL: `https://doi.org/10.1177/01423312221104424` (visited on 11/08/2022).

[57] Hanieh Deilamsalehy and Timothy C. Havens. “Sensor fused three-dimensional localization using IMU, camera and LiDAR”. In: *2016 IEEE SENSORS*. 2016 IEEE SENSORS. Orlando, FL, USA: IEEE, Oct. 2016, pp. 1–3. ISBN: 978-1-4799-8287-5. DOI: `10.1109/ICSENS.2016.7808523`. URL: `http://ieeexplore.ieee.org/document/7808523/` (visited on 09/04/2022).

[58] Francois Caron et al. “GPS/IMU data fusion using multisensor Kalman filtering: introduction of contextual aspects”. In: *Information Fusion* 7.2 (June 2006), pp. 221–230. ISSN: 15662535. DOI: `10.1016/j.inffus.2004.07.002`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S156625350400065X` (visited on 11/09/2022).

[59] Hongwei Mo and Ghulam Farid. “Nonlinear and Adaptive Intelligent Control Techniques for Quadrotor UAV – A Survey”. In: *Asian Journal of Control* 21.2 (2019). _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.100 pp. 989–1008. ISSN: 1934-6093. DOI: `10.1002/asjc.1758`. URL: `http://onlinelibrary.wiley.com/doi/abs/10.1002/asjc.1758` (visited on 11/07/2022).

[60] P. Antal, T. Péni, and R. Tóth. “Nonlinear Control Method for Backflipping with Miniature Quadcopters”. In: *IFAC-PapersOnLine* 55.14 (2022), pp. 133–138. ISSN: 24058963. DOI: `10.1016/j.ifacol.2022.07.595`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S2405896322010060` (visited on 09/04/2022).

[61] Jemin Hwangbo et al. “Control of a Quadrotor With Reinforcement Learning”. In: *IEEE Robotics and Automation Letters* 2.4 (Oct. 2017), pp. 2096–2103. ISSN: 2377-3766, 2377-3774. DOI: `10.1109/LRA.2017.2720851`. URL: `http://ieeexplore.ieee.org/document/7961277/` (visited on 09/04/2022).

[62] Elia Kaufmann et al. *Deep Drone Acrobatics*. June 11, 2020. arXiv: `2006.05768[cs]`. URL: `http://arxiv.org/abs/2006.05768` (visited on 09/04/2022).

[63] Pengfei Yu and Kc Wong. “An implementation framework for vision-based bat-like inverted perching with bi-directionalthrust quadrotor”. In: *International Journal of Micro Air Vehicles* 14 (Jan. 2022), p. 175682932110736. ISSN: 1756-8293, 1756-8307. DOI: `10.1177/17568293211073672`. URL: `http://journals.sagepub.com/doi/10.1177/17568293211073672` (visited on 08/19/2022).

[64] *Using vision or motion capture systems for position estimation*. Aug. 2022. URL: `https://docs.px4.io/main/en/ros/external_position_estimation.html`.

[65] *VICON Motion Capture Systems*. Nov. 2022. URL: `https://www.vicon.com/`.

[66] *OptiTrack Motion Capture Systems*. 2022. URL: `https://optitrack.com/`.

[67] *Academische Jaarindeling*. 2022. URL: `https://www.tudelft.nl/studenten/onderwijs/academische-jaarindeling`.

[68] Seamus McGinley. *Autonomous Perching Drone Project Plan*. Nov. 2022.