Hexacopter design for carrying payload for warehouse applications

M. Aswath¹*, S. Jeevak Raj¹
School of Electronics Engineering, Vellore Institute of Technology, Chennai

E-mail: m.aswath08@gmail.com

Abstract. Airborne vehicles are gaining traction in research fields and in bringing it for commercial use. Multi-rotors for payload transport is one of the highly researched topics. In this paper, a hexacopter was prototyped based on commercially available components in the market and dynamic analysis was performed with constraints derived from ideal conditions, such as neglecting the effect of air resistance. The experiment shows the flight of the hexacopter to be stable with and without payload attached to it. Also, various computer vision methods were explored to detect and identify waypoints, in the form of gates, for the hexacopter to follow during its flight mission. The detection mechanism employed is HSV mask based filtering on input image and performing moment calculation on the extracted contour to determine the center of the gates. Localization of the drone with respect to the environment is performed with a monocular visual SLAM method known as ORBSLAM2.

Keywords: Hexacopter, Multi-rotor design, Computer vision, ROS, Modelling and Simulation

1. Introduction

Unmanned Ariel Vehicles(UAVs) are aiding us in a plethora of fields such as agriculture, swarm-based collaborative robots for construction sites, warehouses management, package delivery, and many more. A particular type of UAV called VTOLs (Vertical Take-Off and Landing) has made it possible to get airborne without a runway. It is high time that progress and improvement in these fields are facilitated. For agriculture[1], drones are used to monitor crop health and estimate harvest size. It is also used to spray pesticides over the entire crop region. For this, the drone must carry the heavyweight of the pesticide tank with the accompanying mechanism. At the same time, it must also be robust to turbulence caused by the wind. In construction and warehouse management [2], the drones must be agile and have a fast control response because it might have to avoid any debris, falling objects in the air. By having good agility, it makes a safe working environment. An active area of research is in the domain of swarming behaviors. Robot swarming refers to agents that forms a network and are able to reorganize dynamically as the agents leave or enter the network. They provide greater flexibility in terms of cooperation among multi-agents. Swarming may open the path to the fusion of data, such as mapping of the environment with multiple bots simultaneously. Some bio-inspired flight swarms utilize steering and flocking behaviors. For the control of the UAVs, first, an overview of existing controllers [3] and their effectiveness should be studied. This would help us decide which type of controllers would be most helpful for this implementation. The most common type of control mechanism used is the PID controller which can be used to achieve basic hovering and attitude control. The UAVs has autonomy features such as the ability to maintain a flat position, maintaining altitude, position hold, headless mode, care-free, automatic takeoff and landing, return to home, follow-me mode, GPS way-point navigation, Orbit

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd
around target object and pre-programmed aerobatics such as flips and sharp maneuvers. ROS(Robot Operating System) is acts as a middle-ware between heterogeneous systems and processes. ROS processes are in the form of nodes, connected by edges called topics. ROS nodes can pass messages between each other through topics. The messages do not pass through the master, rather the master enables peer-to-peer communication for topics. This decentralized architecture and flexibility of ROS makes it well suitable for robotics applications. The main advantage introduced is that the robot may also easily use off-board computers for heavy computations. Visualization tools are also provided in ROS such as rviz and rqtgraph which are used extensively by robotics developers. A simulation software called Gazebo is also a main package in ROS. The novelty of this study lies in the ability to integrate open-sourced research algorithm such as computer vision, SLAM and other image processing algorithms(CLAHE, HSV mask tuning), which can be both user-friendly and helps in accelerating research in commercial products as they are ubiquitously available in the market and easily procurable by individuals.

2. Literature review

The study on "Hexacopter for heavy payload transport" by Bhakti Yudho Suprapto et al. [4] provided us with useful insights and information to approach this problem and tailor a variant of the drone for our objective.

Yu Miao et al. [5] introduced the faster and efficient algorithm of CLAHE by introducing the parameter T in bilinear interpolation, which speeds up the interpolation and improves the speed of the entire algorithm.

S. Li et al. [6] developed a faster pose estimation algorithm and has tolerance to noise using the n-point method. This algorithm detects the gates’ four corner points even in blurred vision and uses a predictive feed-forward strategy to find the gate that is not in the visibility.

Adrian Rosebrock [7] at pyimagesearch proposed an algorithm for estimating the distance between two objects, which uses the triangular similarity to find the distance.

Zahid Sadreddini et al. [8] proposed a method to estimate the distance to an object from the camera using feature lines extracted from the environment, the tiled floor, in their case. The feature line is then used to calculate the distance to the object that lies on the horizontal horizon line of the image.

Raul Mur-Artal, Juan D. Tardos came up with SLAM based on ORB features [9]. ORB-SLAM2 supports monocular, stereo, and RGB-D cameras. Because stereo and RGB-D cameras are expensive, bulky, and more power-consuming, a more economical approach with a ubiquitously available monocular camera was chosen for this purpose.

Ebrahim Karami, Siva Prasad and Mohamed Shehata [10] performed a comprehensive study on various feature detectors like SIFT, SURF, BRIEF and ORB. The study illustrates the performances and computation time involved executing each of these algorithms. Further, feature matching capability of the algorithms was also tested by identifying the keypoints from the given two images at when subjected to scaling, rotation, distortions and noise. This helps in deciding the type of feature detector that can work best in the visual SLAM pipeline.

Pawel Burdziakowski [11] presented his implementation of ORB-SLAM2 running on a Jetson TX2 onboard computer mounted on the UAV. He uses a stereo rig setup and uses the stereo cam package offered by the ORB-SLAM2 API. His study evaluates errors that are caused by inaccurate camera calibration. He obtains the intrinsic parameters from each calibration session and then compares the path mapped by the UAV under these different camera parameters. The path comparison was also done with GPS guided mode and with the ground truth path. From this study, the performance of ORB-SLAM2 can be understood and the correct method of camera calibration is verified and supported by the obtained result.

Victor M. Becerra compiled the study on uses and applications of UAVs [12], which suggests that they are increasing rapidly, and the market is about to hit $14 billion by 2027. Growth in the production of UAVs is expected to reach $13.1 billion by 2027. Seeing the demand and the growth of autonomous features in UAVs alongside the UAV market, the applications are widespread from military to civilian, including surveillance, remote sensing, delivery services, agricultural sector, target acquisition, aerial imaging, and infrastructure monitoring, industrial inspection, and emergency medical aid.

Ram Prasad Padhy et al. proposed a solution [13] for using the Convolutional Neural Network model, which is trained over dataset images and responsible for controlling the movement in the left and right direction. Since the dataset wouldn’t be available every time, our method uses image processing algorithms to control its movement.
3. Design of Hexacopter

3.1. Hexacopter configuration

Hexacopter has six motors attached to the frame with arms, these motors rotate the propeller at very high speeds to generate thrust. To achieve stability, three motors are made to rotate in clockwise(CW) and counterclockwise(CCW) directions. With this setup, it is able to cancel the net moment about the drone’s yaw axis. The type of motors used in multi-rotors is BLDC motors, the rotations speed is controlled by using 3 different phases of current. The set speed of rotation of the motors is maintained with the help of ESC(Electronic Speed Controllers). ESCs constantly monitor the feedback current from the motor and make minute changes to the input current for the motors. Inertial Measurement Unit(IMU) which comprises of accelerometer and gyroscope, is used to measure short displacements and determine the orientation of the drone. This is done based on accelerations experienced by the body. Pixhawk is open-sourced flight controller, with on-board sensors that has many inbuilt algorithms to filter the raw sensor data and provides a precise data estimate. It also supports high level protocols such as MavLink which is used to communicate with a computer that may send commands to move the drone to a certain location in world’s GPS coordinates. Batteries are used to power the on-board computer, sensors and motors. Optical flow sensor is used to measure very short displacements at very high frequency, this can be thought of as an odometry present in the wheeled robots. The model of optical flow sensor used is the PX4Flow, this sensor is chosen because it is widely used in UAV applications and has robust filtering algorithms like the EKF(Extended Kalman Filter), which has gained traction in recent years and finding many applications in real world implementations. Along providing the displacement along the ground plane, it also measures the altitude of the drone with the help on inbuilt sonar sensor. The ultrasonic sensor measures the distance from ground to the UAV’s frame to determine the altitude of the flight. Moreover, the sensor is also capable of providing correct readings in indoor as well as outdoor environments, even without the need of illumination of LEDs. The optical flow methods attempt to calculate the motion at each voxel position between two image frames, which are taken at \( t \) and \( t+\Delta t \). These methods are referred to as differential because they are based on local image signal approximations of the Taylor series; that is, partial derivatives are used with regard to spatial and temporal coordinates. Jetson nano is a computer that has powerful computation capabilities given its small footprint and weight profile. It is capable of running Neural Networks for image classification, object detection, semantic segmentation. The power consumption of this computer is as low as mere 5 watts. Lithium Polymer batteries are most commonly used due to their ability to provide high discharge rates and relatively procurable battery type. The specification of the battery chosen is 11.4 volts provided by three cells placed in series, with a discharge rate of 30C. The main frame chosen for the hexacopter design is the DJI F550. This frame is intended towards hobbyist which enables to attach custom components such as GPS, telemetry, LIDARs(Light Detection and Ranging), on-board computers and attach custom motors. Based on these mentioned components, a physical model of the hexacopter was designed in Solidworks and the final 3D CAD(Computer Aided Design) model is exported as URDF(Unified Robotic Description Format) format to load it into the ROS-Gazebo simulation world. ROS, Robot Operating System, allows heterogeneous computers to communicate with each other and provide low level device controls and message passes between processes.

| S.no | Item                                | Weight(gm) |
|------|-------------------------------------|------------|
| 1    | F550 Hexa-Copter Frame              | 620        |
| 2    | LiPo 8000mAh 3S 30C/60C             | 615        |
| 3    | Jetson Nano                         | 249        |
| 4    | BLDC Motors EMAX RS2306             | 204        |
| 5    | ESC Emax BLHeli 30A                 | 168        |
| 6    | Pixhawk                             | 40         |
| 7    | Optical Flow Sensor                 | 30         |
| 8    | Camera 8MP IMX219-170               | 22         |
| 9    | Tri Blade Propellers                | 15         |
|      | Total                               | 1900       |
3.2. Hardware composition
It is necessary to evaluate the total weight distribution of the hexacopter; this allows for analysis of the components and highlights the importance for each part and track its performance with previous and future iterations of the Multi-rotor designs. The heaviest component is all multirotor designs is the battery. For achieving a longer flight time, a battery with larger capacity may be chosen. However, under the additional weight of the larger battery pack, it in-turn directly affects the flight time. However, since the implementation is done without LIDAR, significant amount of weight that would otherwise occur is eliminated. LIDARs also consumes an expensive amount of power from the battery. The weight distribution of all the components of the hexacopter is as shown in Table 1.

3.3. Structural and Stability Analysis
The physical dynamics of the system must be studied to understand the expected motion when different forces are exerted to the hexacopter during its flight. Two significant studies can be conducted to understand the structural integrity and the stability of the hexacopter during its flight. The structural analysis is used to determine the integrity of the frame, arms, and landing gear when subjected to forces that may be caused due to gravity, thrust and collisions that the hexacopter would be able to withstand. For the structural analysis in Figure 1, the arm of the hexacopter is isolated and the base plate was fix and force of 10 Newtons is applied at the end of the arm where the motor exerts force to simulate a situation in flight when it generates thrust to carry its weight along with the payload. The displacement of the plastic material after applying the force is found to be 8mm.

![Figure 1. Stress test](image)

The stability of the physical body is done to check the placement of COM(Center Of Mass) of the structure. By identifying the location of COM, the direction of motion of the hexacopter can be predicted when the frame is subjected to forces around it’s COM. By having the COM in the middle, the thrust by the propellers acts evenly around the COM, thus cancelling any net torque that may cause undesired rotation and drift. Thus, for stability analysis, the center of mass of the hexacopter with and without the payload is evaluated. The payload is attached well below the arms and the body frame. The mass of the payload is given as 2 Kgs. The resulting location of the COM with payload is shown in Figure 2 and COM location without the payload is shown in Figure 3. Adding the payload towards one side of the hexacopter will cause the hexacopter to drift or put more strain on the motors present directly above the payload. This is because, the motors above the payload need to provide more thrust to balance the frame and prevent it from toppling over during the flight.
3.4. Control system modelling
The CAD model is loaded into a simulation environment called Gazebo where the dynamics of bodies can be simulated. Since the CAD file contains the mass properties like the center of mass, the moment of inertia it is possible to estimate the real-world responses safely in a simulation. The initial objective must be to test and take-off and landing of the hexacopter; this means that it must reach a set-point altitude. To achieve this, a PID (Positional, Integral and Derivate) feedback control was used. The PID controller receives the data from the output and feeds it back into the input to generate actuator command for the next time step. The controller has three main parameters $K_p$, $K_i$, $K_d$ through which the error of the output data is multiplied and added together. The parameter $K_p$ denotes the proportional error constant, the higher the error, a larger thrust command is sent to the motors. The integral gain $K_i$ is used to nullify any errors that are accumulated overtime such as, a constant wind that may force the hexacopter away from the target position or attitude. The derivative gain $K_d$ is the amount by which sensitivity to changes in physical quantities such as changes in wind velocity can be accounted for changing the thrust commands appropriately. The concerned output parameter in this case is the current altitude of flight in the world, and the error is calculated with respect to the desired set-point height of the hexacopter to reach as its final state.

![Figure 2. COM without payload](image1)
![Figure 3. COM with payload](image2)

![Figure 4. Initial response](image3)
The Gazebo simulation publishes the current altitude of the hexacopter at small and discrete timesteps of around 1 milliseconds. The simulation was performed to obtain the initial response, as shown in Figure 4 is obtained. The hexacopter model was made to reach a desired setpoint height of 1 meter with random $K_p$, $K_i$, $K_d$ values as the PID constants. Then, the system was made to linearize around the desired state by using the error response. The MATLAB’s PID tuner, proportional, integral, and derivative gains are adjusted until a smooth transient response, as shown in Figure 5 is obtained.

### 4. Navigation with Computer Vision

Computer vision plays an important role in detection and alignment to the gate. The block diagram as shown in Figure 6, summarizes and provides an overview of the process described below. So, the flow of computer vision is as follows. First, the image is captured from the camera. Then, CLAHE (Contrast Limited Adaptive histogram equalization) is applied to the image to improve the visibility and ensure that there is no glare or disturbances in the image. CLAHE is applied to tiles (smaller regions of the image) and combined using Bilinear Interpolation. Since CLAHE needs to be applied to the color image, the image was separated into three channels and for each channel in HSV, CLAHE was applied and then the channels were recombined. Next, the HSV mask is obtained by the following method. First, to detect the gate by it’s yellow color frame, the HSV values were tuned with trackbars with a python script to create a mask. After obtaining the values, the HSV mask was applied such that now, only the yellow color gets detected. Then, the contours are detected from this image with canny edge detection, each of the contours is looped by checking the solidity, dimension, and aspect ratio. If the value matches the requirement then the distance to the gate from the camera is estimated using triangular similarity. The rectangle with the maximum area and the minimum distance is chosen and the trajectory is generated.

To estimate the distance to the gate, the triangle similarity method has been used. This method works as follows, first the perceived focal length of the camera is calibrated and the width of the gate is known. So, by applying the variable the length in pixels of the gate, obtained from the processed image, to the Formula 1, the actual distance $d$ to the gate is obtained. Where $W$ is known width of the gate, $f$ is the focal length of the camera lens, and $p$ is the width in pixel, by using these values, the distance can effectively predict.

$$d = \frac{W f}{p} \quad (1)$$
The hexacopter aligns itself using the PID control, it also uses the previous error values to find both pan error and tilt error. As it is carrying a payload, the center of the frame for the drone should be a little high so this is done by setting appropriate values to the dead zone if the drone finally gets aligned to the center then it passes the gate.

4.1. Localization of the camera frame

As the implementation requires high-level understanding from digital images, videos for ensuring that the robot traverses in the right direction. So, practically, the robot has to capture the entire essence of its surroundings in order to choose a path that seems less likely to have any obstacles and there are a lot more factors that have to be analyzed. This puts forward a necessity to implement novel computer vision algorithms. So, in order to have these tasks performed, ORB feature detector and descriptor is one of the tools that marks the basis for the rest of the perception stack.

For localization of the drone with respect to the environment, ORBSLAM2 was used. This method is able to localize the camera by identifying ORB features in the image, as shown in Figure 7. Oriented FAST and Rotated BRIEF(ORB) was developed by Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R. Bradski at OpenCV labs, as an initiative to use open-source algorithm for feature extraction unlike SIFT and SURF which were patented. Incorporating FAST keypoint detector and BRIEF descriptor makes the ORB feature detector attractive, both of these together represent an object.

4.1.1 FAST Keypoint Detector

The implementation of the drone requires real-time video processing for executing corner detectors. FAST stands for Features from Accelerated Segment Test, and it is basically a method for corner detection. Moreover, features aren’t something that can be given a definition as it is something that varies with applications. They are typically an interesting part of an image and to put it simply, they are necessary fragments of information (image) that are relevant for a particular application. Features in images can assume any form, be it edges, corners or regions of interests. FAST features, however, do not have an orientation component or multiscale characteristics. Given a pixel \( P \) in an image, the brightness of \( P \) is quickly compared to the neighboring 16 pixels that are around \( P \) in a small circle. The circle’s pixels are then sorted into three classes (lighter than \( p \), darker than \( P \), or \( P \)-like). If more than 8 pixels are darker or brighter than \( p \), the keypoint will be selected. So keypoints found quickly give us information on the location of an image’s edges. A multiscale image pyramid is used in the ORB algorithm. An image pyramid is a multi-scale representation of a single image consisting of arrangements of images at different resolutions, all of which are versions of the image. The downsampled version of the image is included in each pyramid level as compared to the previous level. Once a pyramid has been created by the orb, the FAST algorithm is used to identify keypoints in the picture. By detecting keypoints at different resolutions of the image, ORB is partly scale invariant in this manner. After locating keypoints, ORB assigns orientation to each keypoint like left or right, depending on the direction the pixel intensity level changes around that keypoint. Intensity centroid is used by ORB which calculates the changes in intensity levels. The intensity centroid adopts that a corner’s intensity as offset from its center, so orientation is assigned with this vector. So, for our application, features would be something that would help in object recognition, motion detection, video
tracking, and other purposes. So, FAST is used this to extract such feature points which would be later used to track and map objects.

4.1.2. BRIEF Descriptor BRIEF, which stands for Binary Robust Independent Elementary Features, is a descriptor that gives out feature descriptions or feature vectors. Ideally, it encodes the feature points in such a way that they are invariant under any image transformation, such as shear, rotation and scaling. It is also robust to different lighting condition to an extent. As the drone moves in its surrounding with varying lighting conditions, the camera may have a slightly different exposure levels at each frame, it has to be ensured that the feature points are kept tracked. This adds the necessity of including image descriptors that suits well for such situations. BRIEF takes all the keypoints found by the FAST algorithm and transforms them into a vector of binary features so that they can represent an object together. Binary vector features are also known as a binary feature descriptor, a feature vector containing only 1 and 0. In short, a function vector that is a string of 128 to 512 bits describes each keypoint. To make the descriptor from being invariant to high frequency noise, BRIEF starts by smoothing the image using a Gaussian kernel. In a defined neighborhood around that keypoint, then BRIEF selects a random pair of pixels. A patch is known as the defined neighborhood around pixels, which is a square with a certain width and height of pixels. The first pixel from the random pair is drawn with a standard deviation or spread of sigma from a Gaussian distribution centered around the keypoint. The second pixel from the random pair is drawn with a standard deviation or spread of sigma by from a Gaussian distribution centered around the first pixel. Now, if the first pixel is brighter than the second pixel, the value 1 is assigned, else 0 is assigned to the corresponding bit. For a 128-bit vector, this method is repeated for a keypoint 128 times. For each keypoint in an image, BRIEF develops a vector like this. However, BRIEF is not invariant to rotation, so rBRIEF(Rotation-aware BRIEF) is used by the ORB algorithm. Without losing out on the speed aspect of BRIEF, ORB attempts to add this feature. Similarly, BRIEF requires far less storage capacity with a speed that is well above what the conventional feature descriptors can offer. ORB does is a sort of fusing FAST detector and BRIEF descriptor. Though both of them have several advantages, there are few drawbacks, and ORB fuses them in a way that would overcome the major drawbacks. For example, BRIEF has been observed to perform poorly with rotation, and ORB overcomes the drawback by steering BRIEF accordingly with the orientation of feature points. Hence the name rotated BRIEF. This is how ORB is able to provide faster and more efficient performance than other algorithms such as SIFT (Scale-Invariant Feature Transform), SURF (Speeded-Up Robust Features). Hence, in effect, ORB provides: a faster and accurate oriented component to FAST and a computation that is more efficient by orienting BRIEF features. Thus the ORB feature extraction has proven to be advantageous for our implementation.

4.1.3. SLAM SLAM, also known as Simultaneous Localization and Mapping, is a computational problem of constructing a map of an environment that usually is not known while simultaneously keeping track of the location of an agent within that environment. Now a visual SLAM based on the feature points is implemented. There is a need of a constructing a map of the environment that is initially unknown to the robot, wherever the robot is supposed to navigate, and that’s how the robot will be guided along a path without having to run into any obstructions. So, SLAM is something that totally fits into the requirements.

Even though a camera may contain many number of lenses placed before the CMOS sensor, these cameras can be reduced to a simple pin hole model. However, before the output image from the camera can be used, the distortions introduced by the lens must first be corrected. This can be done by calibrating the camera with a checkered board pattern. The result of the calibration with checkered board pattern gives the intrinsic camera parameters. The intrinsic parameters comprises of the focal length in x and y directions of the image frame and the principal point(x,y), which is the center of the image denoted in pixel space. There is also a shear coefficient s, which denotes the amount of shear present in the image pushed out by the camera. The extrinsic matrix of the camera is used to denote the transformation of the camera with respect to the world coordinate frame. The extrinsic matrix, comprising of rotation and translation, is computed by matching ORB features that is extracted in each frame at subsequent time instances.

The features of bundle adjust algorithm can briefly explained as, refining both the camera poses and the point cloud with the help of each other. Two types of Bundle Adjust is performed in ORBSLAM2: full bundle adjust and local bundle adjust. When a set of images is rendered, describing
a set of 3D points from several viewpoints, bundle adjust is used for simultaneous refinement of 3D coordinates depicting the scene geometry, the thresholds of relative motion, including the optical parameters, intrinsic and extrinsic, of the cameras installed to procure the images. Bundle adjust has a very essential role to be played in feature-based monocular SLAM. In several new era SLAM pipelines, bundle adjustment is known to perform to have a proper estimate of the 6-DOF camera trajectory as well as 3D map (sparse 3D point cloud) directing from the input feature tracks. In the ORBSLAM2 pipeline, The local bundle adjust is performed after it receives a new keyframe. After loop closing is detected, full bundle adjust is performed to correct drift in translation of the camera’s trajectory and correct the

location of world points in 3D which were reconstructed from the acquired images during traveling in the loop. Considering all the aspects mentioned, two basic weaknesses circumvents SLAM pipeline based on bundle adjustment. Firstly, the requirement of carefully initializing bundle adjustment depicts that all variables, basically the map, must have an estimation as precise as possible, as well as maintained over time, that makes the overall algorithm very complex. Secondly, as estimating the 3D structure (which needs sufficient baseline) is considered to be inherent in bundle adjustment. The visual SLAM algorithm will face challenges during the time of slow motion or pure rotational motion.

Bundle adjustment gets down to the fact of minimizing the projection error between the image positions of analyzed and predicted image points. The error is computed as the sum of squares of a greater number of nonlinear, real-valued functions. Hence, the minimization is attained involving nonlinear least-squares algorithms and approaches. For error calculation and correction in these three types of bundle adjust, Levenberg-Marquadt has shown to be one of the most efficient due to its ease of incorporation and its utilization of an effective damping strategy that renders the capability to converge to the solution in very short period of time, from a wide range of initial prompts. Through iterative linearization of the function to be reduced in the neighborhood of the current estimate, the Levenberg–Marquardt algorithm uses the concepts of linear systems, known as the normal equations. These all can be exploited to produce high computational benefits by installing a sparse variant of the Levenberg–Marquardt algorithm which takes an edge of the normal equations’ zeros arrangement, while avoiding storing and operating on the zero-elements.

Hence, with minor translations of the camera, the features are compared with the successive frames to estimate the rotation and translations of the camera in pixel space. The localization of the camera frame with respect to the world is shown in Figure 8. To convert from pixel space to real-world coordinates, a scale factor is multiplied with the translation vector. This scale factor is obtained during the initialization of the ORBSLAM2 algorithm.

![Figure 7. ORBSLAM2 feature detection](image1)

![Figure 8. Camera frame localization](image2)
5. Results
The hexacopter is tested for takeoff and landing with the payload attached to it, in the simulation world as shown in Figure 9. Initially, the hexacopter is placed at origin and touching the ground. After the simulation starts, the thrust commands are sent from Simulink to the hexacopter’s motors in the Gazebo simulation. The response plot of the hexacopter’s altitude at each time step attained by the is obtained. The response is plotted along with the desired setpoint altitude is shown in Figure 10. The yellow coloured lines represent the setpoint given by us, and the blue coloured plot is the response of the hexacopter. The hexacopter was able to rise from ground to an altitude of 1 meter in about 1.2 seconds in a smooth transient behavior. The hovering was smooth without any oscillatory behavior. The landing also exhibited the same characteristics as the takeoff.

The image processing part was tested for a single gate detection as shown in Figure 11, the model was tested using single gate, adjusting it’s position by moving left and right, which worked fine. And once it is aligned, it moved forward after calculating the distance. Since the single gate detection approach was working fine with single gate, testing was done with 15 gates to obtain the desired outcome. Now, the constraint is to detect multiple gates in a single camera frame. First, the camera calculates the area of each of the gates in its field of view, it then chooses the gate closest to it, the gate with the maximum area, and aligns itself to the center of that gate and passes through it. The image in the first quadrant is the original input image, the second quadrant depicts the HSV mask for detecting the yellow color of the gate, the third quadrant shows the edge detection and the fourth quadrant shows the alignment direction towards the center. Thus as shown in Figure 12, it is evident that the multi-rotor tries to align itself towards the center(by moving left as shown in the arrow mark) of the gate and passes through it. The multi-rotor passes through 15 gates covering a total distance of 15 meters set on a straight aisle. The average speed was found to be 0.7 m/s.
Figure 12. i) Top left: raw input image. ii) Top right: HSV mask. iii) Bottom left: border extraction. iv) Bottom right: arrow indicates the direction of motion to align the multi-rotor to the center of the gate.

6. Conclusion
The physical design process of a hexacopter with a CAD software was explored and structural and stability analysis was performed. The 3D model was then used to perform simulation to validate the design. Moreover, the implementation and verification of computer vision algorithms to locate the waypoints in the form of gates was done and the monocular visual SLAM to localize drone with respect to the environment was also performed.

References
[1] M. Reinecke and T. Prinsloo. “The influence of drone monitoring on crop health and harvest size”. In: 2017 1st International Conference on Next Generation Computing Applications (NextComp). 2017, pp. 5–10.
[2] J. H. Ong, A. Sanchez, and J. Williams. “Multi-UAV System for Inventory Automation”. In: 2007 1st Annual RFID Eurasia. 2007, pp. 1–6.
[3] B. Han et al. “A Review of Control Algorithms for Quadrotor”. In: 2018 IEEE International Conference on Information and Automation (ICIA). 2018, pp. 951–956.
[4] B. Y. Suprapto et al. “Design and development of heavy-lift hexacopter for heavy payload”. In: 2017 International Seminar on Application for Technology of Information and Communication (iSemantic). 2017, pp. 242–247.
[5] Yu Miao et al. “Application of the CLAHE Algorithm Based on Optimized Bilinear Interpolation in Near Infrared Vein Image Enhancement”. In: Proceedings of the 2nd International Conference on Computer Science and Application Engineering. CSAE ’18. Hohhot, China: Association for Computing Machinery, 2018. ISBN: 9781450365123. DOI:10.1145/3207677.3277957. URL:https://doi.org/10.1145/3207677.3277957.
[6] Shuo Li et al. “Autonomous drone race: A computationally efficient vision-based navigation and control strategy”. In: CoRR abs/1809.05958 (2018). arXiv:1809.05958. URL:http://arxiv.org/abs/1809.05958.
[7] Adrian Rosebrock. Find distance from camera to object using Python and OpenCV. URL:https://www.pyimagesearch.com/2015/01/19/find-distance-camera-objectmarker-using-python-opencv/(visited on 09/12/2020).
[8] Z. Sadreddini, T. Çavdar, and H. B. Jond. “A distance measurement method using single camera for indoor environments”. In: 2016 39th International Conference on Telecommunications and
Signal Processing (TSP). 2016, pp. 462–465.

[9] Raul Mur-Artal and Juan D Tardós. “ORB-SLAM2: An open-source slam system for monocular, stereo, and RGB-D cameras”. In: IEEE Transactions on Robotics 33.5 (2017), pp. 1255–1262.

[10] Ebrahim Karami, Siva Prasad, and Mohamed Shehata. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images. 2017. arXiv:1710.02726 [cs.CV].

[11] P. Burdziakowski. “Towards Precise Visual Navigation and Direct Georeferencing for MAV Using ORB- SLAM2”. In: 2017 Baltic Geodetic Congress (BGC Geomatics). 2017, pp. 394–398.

[12] Victor Becerra. “Autonomous Control of Unmanned Aerial Vehicles”. In: Electronics 8 (Apr. 2019), p. 452. DOI:10.3390/electronics8040452.

[13] Ram Padhy et al. “Deep Neural Network for Autonomous UAV Navigation in Indoor Corridor Environments”. In: Procedia Computer Science 133 (Jan. 2018), pp. 643–650. DOI:10.1016/j.procs.2018.07.099.

Acknowledgment
The authors would like to express their gratitude to their faculties for guidance on this paper and parents, family, and friends for their moral support throughout this project.