Technical Paper for the International Aerial Robotics Competition

Matias Christensen Håkon Flatval Magnus Reiersen
Martin Sollie Christian Wilhelmsen Filip Lolland
Kevin Kaldvansvik Håvard Mellbye Rasmus Munter
Ulrich Isachsen

Ascend NTNU
Norwegian University of Science and Technology
Trondheim, Norway

ABSTRACT

The aim of this paper is to describe a system for a fully autonomous MAV capable of solving the seventh IARC mission. The system is designed to perform on-line strategic planning, collision avoidance, robot-to-robot interaction and navigation, relying on INS sensors and camera vision in a GPS-denied environment.

INTRODUCTION

Statement of the problem

In the seventh mission of IARC, the goal is to develop a fully autonomous drone, whose objective is to guide at least 4 out of 10 wheeled robots across a green line in a 20x20 meter flat arena, through physical interaction. The drone must accomplish its objective within a given time constraint, whilst detecting and avoiding moving obstacles. Furthermore, the drone cannot rely on external measurements, such as GPS or camera tracking systems. This description constitutes part A of the mission, while in part B the drone will additionally compete against another drone simultaneously.

Yearly Milestone

Since this is the last year of mission 7 we have been working on finishing and improving our previously developed design and algorithms. There have been two main goals this year. Firstly we've focused on continuous testing throughout the year to increase the reliability of our system compared to last year. Secondly we have worked on integrating our planning module to ensure that we make the most efficient choices during the competition.

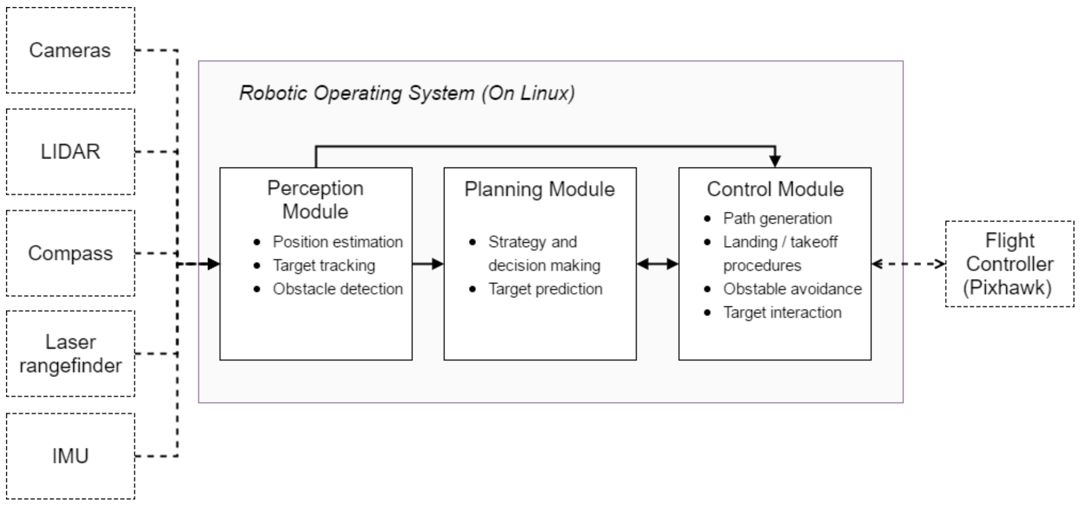
Conceptual solution

Our organizational breakdown of the problem was largely based on last year's, with the addition of a hardware group to facilitate a more ambitious focus on creating custom mechanical and electronic parts. The technical members of this year's team were broken into four groups:

- Perception: Estimate the absolute position of the drone in relation to the grid, as well as the position of moving targets and obstacles.
- Planning: Compute a plan of waypoints and actions so as to efficiently interact with the targets to solve the objective in time.
- Control: Compute and follow collision-free paths between waypoints, and perform necessary maneuvers to interact with targets whilst avoiding nearby obstacles.
- Hardware: Design, construct and assemble the mechanical and electronic components of the drone.

System architecture

Our system is running on Linux with mainly three different modules communicating to each other using a framework called The Robotic Operating System (ROS). This framework provides conventions, libraries and easy communication between modules across multiple devices. The three modules used is a *Perception module*, a *Planning Module* and a *Control module*. The Perception module streams real time observations gathered from onboard sensors, continuously on the ROS network such that other modules can listen to it. The Planning Module use this information to create commands which is sent to the Control module. The latter is a navigation system, breaking commands into subtasks and converting these subtasks to a language the flight controller (Pixhawk) can understand.



Overview System Architecture

AIR VEHICLE

Our vehicle have a quadrotor layout and was designed to accommodate the housing of four horizontal and one vertical camera, two LIDARs, and 2 Nvidia Jetson TX2s. The drone has a size of 96 cm from propeller tip to propeller tip, and a weight of 2900g. This year a new suspension system is used that has a built in functionality for detecting if the drone has landed on the ground or on a ground robot. The drone also has sensors to detect ground robots that collide with it while landed.

Propulsion

For propulsion highly efficient Tiger NM-35 brush-less motors with 13" inch low pitch carbon fiber propellers are used. The power is delivered to the motors trough 30A LittleBee OPTO ESC's. The propulsion system has a thrust to weight ratio of 3 and a 15% higher thrust per watt used, compared to the drone of 2016.

Guidance, Navigation, and Control

Our guidance and navigation needs are motivated by our planning module that computes an efficient plan of waypoints and actions to solve the mission. The planner relies on the knowledge of the drone's 3D location in the arena to compute the best plan, and as such necessitates a method of estimating our position. Furthermore, the drone must be able to follow waypoint paths while avoiding obstacles.

Stability System

We use the Pixhawk flight controller, with a modified version of the PX4 flight stack, to perform low-level stabilization and flying. Commands to the flight controller can be sent over a serial connection from one of the onboard Jetson TX2s.

World State Estimation

In order to make decisions and solve the mission, we must have information about the position of the drone and the ground robots relative to the arena, with some degree of accuracy. In addition to intrinsic measurements from the IMU in the flight controller, multiple computer vision algorithms have been deployed to acquire this information. The computer vision algorithms include two algorithms determining the pose of the drone within the arena, and two algorithms supplying information about the ground robot herd. In addition, an algorithm utilizing a 2D-scanning LIDAR is used for obstacle detection.

Air Vehicle Localization

In localizing the air vehicle, two computer vision algorithms are put to use. We call them the *Grid Detector* and the *Global Positioner.*

Grid Detector The Grid Detector utilizes the grid in the arena to determine the position and potential orientations of the air vehicle within a grid cell. The algorithm runs on the image stream from our vertically pointing fisheye camera and takes roll, pitch and vertical position of the drone as inputs. To extract the grid from these images, the pixels are thresholded by gradient, then undistorted, and undergo a Hough Transform, localizing the most probable grid lines in the image. After a number of lines are found, a search for suitable squares among the lines is executed, and upon finding matches good enough, determines the position and orientation of the drone relative to the grid cell right underneath the drone.

Global Positioner While the grid detector only measure the pose of the drone within a grid cell, the global positioner estimates the position of the drone on the ground plane of the arena. There are two systems working together to estimate this position.

By using the IMU and estimated position from the grid detector, the global position can be integrated based on these values. This is a simple integrator that use a low-pass filter on the grid-detector position and updates the global position estimate based on this. An additional edge detection algorithm is running in parallel using images from the downwards pointing camera with a low update rate, which gives a rough estimate on where the drone is on the playing field relative to the sides. This is passed as an offset to the position integrator. The global position is the position of the integrator plus the latest offset coming from the edge detection algorithm. Most of our systems are based on the integrated position without the global offset for a more stable flight.

The edge detection algorithm is using the following observation. All lines inside the grid are shorter than one meter, meaning that all lines longer than one meter is outside. The algorithm will first perform a segmentation of the image, which either detects the inside/outside of the field or the white lines. A binary edge detection is run on this, and the Douglas Peucker algorithm is run on these edge pixels after they are connected by their neighbourhood [2]. The end points of the simplified lines are ray-casted into the arena ground plane such that the true line position is obtained. The two lines closest to the drone (if any) that are in separated quadrants of the unit circle, are considered edges of the arena if they are longer than a certain threshold greater than 1 meter. The position of the drone relative to the edge is obtained from this.

Ground Robot Localization

We have two algorithms whose purpose is to estimate the position of ground robots. In this paper, we will call them the *Horizontal Robot Detector* and the *Vertical Robot Detector*, named after which camera outputs they operate on. We also use a 2D-LIDAR for detecting obstacles with an algorithm called *Obstacle Detector*.

Horizontal Robot Detector The estimated positions of ground robots from side mounted camera feeds is provided by a trained artificial neural network which is based on the SSD architecture [1]. The heavy operations required is feasible because of the GPU provided by one of the Nvidia Jetsons aboard our drone. After positions of ground robots are found in the image, the position and orientation of the drone is utilized to compute the positions of the robots in the arena frame. The artificial neural network distinguishes between robots of different colors as well as obstacle robots. It detects robots with great accuracy, although the frame rate is limited and the positioning is sensitive for inconsistencies in the measured drone pose. The training data is self labeled material from previous IARC competitions and from test flights.

Vertical Robot Detector For precision landing and planning, the ground robots are also tracked with the downward facing camera. The ground-robot detection algorithm starts by color enhancment, extracting red and green colors with the following formula

$$v = \frac{|r - g|}{r + g + b}$$

where the r, g, b values are the color values of the inspected pixel and v is the result gray level value of that pixel. This is performed at every pixel. The next step is to threshold this data, which is followed by a clustering algorithm on the thresholded pixels where pixels are grouped into regions. If the regions are larger than a certain threshold (based on flight height) they are considered ground robots, and a bounding box is drawn around it. The robot direction is found by tracing a line from the center of the bounding box, parallel to the shortest sides of the bounding box rectangle away from the center of mass of the segmented region of the colored plate.

Obstacle Detector The 2D-LIDAR gives a planar point cloud which is used for obstacle detection. The algorithm starts with a median filter, and then looks for hops which follow the following condition $|d_i - d_{i+1}| > t_h$ where d_i is a radial distance from the lidar at index i and t_h is some threshold. It measures the distance between such hops (doing a deprojection), and if the distance is within some threshold, the region is considered a ground robot. This method will also detect many objects that are not obstacle robots, but it is ok as it's not desirable to crash into those things either.

Path planning and Obstacle avoidance To achieve efficient movement from A to B without engaging the obstacle avoidance system, an A* (A star) path planning algorithm built into the Control module plans the shortest safe path. The algorithm predicts obstacle movement ahead of time, and prefers planning routes behind a moving obstacles. For unforeseen situations, such as obstacles moving towards the drone while hovering, a low level obstacle avoidance system avoids obstacles not handled by other high level systems by "pushing" the drone away from obstacles. Obstacle location is provided by the obstacle detector.

Ground robot landing Ground robot landing is implemented using a modified Bearing Guidance algorithm. The algorithm calculates XYZ velocity setpoints based on distance to target. The algorithm attempts to maintain a high altitude for as long as possible, to maintain good position estimation.

Navigation

Controlling the drone consists of the interaction between three systems, the decision making system a.k.a Planning module, a navigation system a.k.a Control module and the flight controller. The decision making system is loosely coupled with the navigation system, which takes in commands as input, break it down to sub tasks, and convert the tasks to a language which the flight controller can understand. The commands are classified in the following types.

- GO-TO(x,y,z) - travel to a point
- LAND-AT(x,y) - land at a position on ground
- LAND-ON(id) - touch the top of a ground robot
- TAKEOFF - takeoff at current point

Planning

The planning module takes in observations generated from the perception algorithms and output commands to our control module. The implemented solution builds on last years algorithm which we call the *plank* algorithm.

The algorithm estimates the optimal solution for a simpler mission where there is only one ground robot. This was modeled as a Markov Decision Process (MDP) with a discrete state and action space. The state consisted of the ground robots position rounded to the nearest meter and the robot angle. The action space was split into three possible instantaneous actions: landing in front, landing on top or doing nothing to the single ground robot. This simplified mission was then simulated with the properties above and with rewards defined by rules of Mission 7.

We ran value iteration on the setup above sampling the state at 5 second intervals. Value iteration estimates the value of all states based on the rewards in the mission. The state values were then used to calculate the estimated optimal action by calculating the action that leads to the highest value state. This was saved in a 3D matrix.

The module used for Mission 7 consists of choosing to interact with the robot that has the highest value state and performing the action stored in the 3D matrix that corresponds to that robots state.

The simplifying assumptions in the algorithm above does result in some sub-optimal actions, such as when the target ground robot collides with another ground robot. These types of situations are not considered by the algorithm due to the challenge predicting them compared to how often the situation occurs. Ignoring these situations has had the benefit of simplifying the code, increasing the robustness of the algorithm by simplifying debugging and reducing the number of edge cases.

We have not been able to perform full scale testing yet due to space constraints but in simulations with partial observability this algorithm gets 4-7 ground robots over the green line in 10 minutes.

Flight Termination System

We are using a custom designed killswitch with a PCB designed to handle 30V 100A continuous current, with bursts up to 350A. This is far beyond our needs and allow the killswitch to run very cool, with the limiting factor being the XT60 connectors mounted on the board egdes, which are designed for 60A continuous. The killswitch is controlled by its own FrSky RC receiver, and defaults to the kill-state. It must be continuously given given the correct signal from the RC receiver to allow current to pass to the motors.

PAYLOAD

Computing

This year we are using two Nvidia Jetson TX2 mounted on the drone, instead of a ground based desktop connected to the drone by a wireless network. This was done as we faced multiple issues with using a wireless connection on our previous drone. We had issues with latency, bandwidth and connection stability, all of witch where improved by moving the computing on-board the drone and only using the ground computer as a command and debugging tool.

Sensors

The Pixhawk flight controller includes an affordable IMU sensor, consisting of an accelerometer, magnetometer and gyro. Additional sensors are supported as plug-ins. We also include the laser rangefinder Lightware SF10/A for height measurement. The rangefinder has a range of 25 m, well above the designated operational limits for the mission.

Four side facing cameras and a downward facing fisheye camera provide additional motion measurements. The four side facing cameras are LI-M021C, 720p 60 FPS cameras, with Lensagon BM3516ND lenses giving a 81 degree field of view. The downward facing camera is ELP-USBFHD01M-L180 fisheye camera. It has a field of view of 180 degrees, but is cropped to 144 degrees. Pixels outside this range suffer from heavy geometric distortion, and are unsuited for image processing.

Physical contact with ground and other solid objects are detected by four strain gauges glued onto aluminum tubing which is mounted on each landing leg. The signal from the gauges is measured by our instrumentation PCB, developed in-house. The same PCB is used to detect Rumbas that are colliding with the drone, using micro-switches and suspended strings surrounding the landing gear.

Communications

The two onboard Nvidia Jetson TX2's are connected using a wired gigabit ethernet network. One of the Jetsons, chosen as the "master", is connected to the ground station computer using Wifi.

The Pixhawk is connected to one of the on-board Nvidia Jetson TX2s using UART. This gives the computer vision algorithms access to pose estimates, and allows us to send position measurements and commands to the Pixhawk.

Communication between the Pixhawk, the onboard Nvidia Jetson TX2's and the ground station is handled primarily by the Robot Operating System (ROS). MAVROS, a ROS package that wraps around the MAVLINK protocol, provides communication between the Pixhawk and the onboard computers. Heartbeat messages must be sent regularly to establish link connectivity, and is handled automatically by ROS.

Power Management

The drone is powered using two 4000 mAh, 4S1P batteries connected in parallel. Electric current generated from these flow through the Pixhawk power module which measures voltage and current, and powers the Pixhawk. The current then passes a safety shutdown switch rated at 100A continuous. The voltage and current measured by the Pixhawk power module is sent to the ground station computer and shown in our mission control software. In addition we use a FrSky FLVSS battery monitor to have the battery voltages shown on the RC transmitter used by the safety pilot. This year we also have our own custom designed power distribution board(PDB). The PDB features an on board killswitch as well as a battery management system(BMS). The BMS provided far bigger challenges than predicted, hence it is not certain that the PDB will be used in the competition.

OPERATIONS

Flight Preparations

To ensure the safe operation of our vehicle, the following checklists are used for every flight:

Preflight checklist

1. Verify that the vehicle is in good physical condition, with no loose parts and with no objects in danger of being hit by any propeller
2. Turn on killswitch transmitter, ensure it is set to the kill position
3. Turn on RC controller
4. Connect batteries to the killswitch input connector and to the FrSky voltage monitor
5. Verify that the battery voltages are visible on the RC controller screen, and that the batteries are in a charged state
6. *If the on-board computers are to be used:* Power them on and connect to them using SSH over WiFi from the ground station computer, start any necessary software

Takeoff checklist

1. Verify that all persons are at a safe distance, and behind a protective net if available
2. Verify that all switches on the RC controller are in the correct position, and throttle set to minimum
3. Prepare the drone for autonomous takeoff using the ground station computer (idle command)
4. Enable killswitch to power motors, wait for the correct audible response from the motor controllers

5. Arm flight controller using the RC controller, transition to offboard control to allow the onboard computers to control the drone. The drone will then be idling under offboard control.
6. Issue the takeoff command from the ground station computer

Landing and postflight checklist

1. Gently land the vehicle
2. Disarm the flight controller using the RC controller
3. Set the killswitch transmitter to the kill position
4. *If the on-board computers are running:* Gracefully shut them down
5. Remove the batteries

Man/Machine Interface

We have developed our own ground control graphical user interface for monitoring and controlling the autonomous aspects of the drone while it is flying. It enables us to monitor the state of the batteries, the temperatures and resource utilization of the onboard computers, the drone and ground robot positions found by the computer vision algorithms and 2D LIDAR, the next planned actions and more. Is also allows us to override the autonomous control (i.e. for testing the command interface to the flight controller) for debugging and testing purposes.

For safety purposes, we still only allow the flight controller to be armed from the RC controller used by the safety pilot. The safety pilot chooses when the on board computers gets control over the vehicle, and can regain manual control at any time in the case that an anomaly should occur.

RISK REDUCTION

Vehicle Status

Shock & Vibration Isolation Shock and vibrations can easily disturb the sensors on the control board, in addition to making the video from the cameras blurred. Vibrations should therefore be reduced as much as possible. Several methods were used to reduce the vibrations. The vibration reduction starts in the propellers and motors, they have been carefully measured and balanced to reduce the production of vibrations, from unbalanced weight distribution. The arms of the drones are made of carbon fiber, which helps to keep the frame lightweight and strong, but it is also a good material to absorb the vibrations from the motors. The frame have been built with material use and a geometry that minimizes deflections, vibrations and asynchronous vibrations. The control board is mounted with shock absorbing tape to reduce high frequency vibrations, but still allows the control board to get quick feedback from the movement of the drone. The horizontal cameras are mounted on 3D-printed arms designed using an iterative process in order to reduce blurring of the video from the cameras.

EMI & RFI Solutions The electronic speed controllers (ESC) have potential to generate a lot of noise to nearby circuits. To avoid this sensitive electronics and the ESC are placed far away from each other. Especially the magnetometer. There is also made space for EM shielding if needed. To ensure good connection between kill switch and the multirotor, a Frsky frequency hopping radio signal is used.

Safety

To ensure the safety of the drone, we use large slow-rotating propellers and a killswitch. Combined with routines for quality control and procedures for flight. In cases of doubt the drone will stay grounded until the issue in question is found and resolved.

Modeling & Simulation

Validation of the frame Finite element analysis in Abaqus and SolidWorks was used to optimize and validate the structural integrity of the drone. The drone were designed far stronger than the yielding point for added rigidity and reduction of vibrations and deflection between the IMU, sensors, and the motors.

Testing

Our drone pose estimation algorithms have been tested and compared to ground truth measurements from an Optitrack camera motion tracking system. Thus we have been able to tune and improve our algorithms with continuous testing. The physical parts of our vehicle have been developed with several design iterations and rapid prototyping, allowing us to test the strength of our parts and improve the design when needed.

CONCLUSION

This year our goal was to complete and test all the systems needed to win mission 7A. The rigorous testing schedule we have followed has resulted in 3 crashes and delayed the completion of the project. However, it has also unveiled many issues and errors that would have appeared at the competition had it not been for these test runs. It has also allowed us to ensure that many of our key systems are robust and reliable. At the time of writing this paper what remains to be done is integrating and testing our planning module and our land on top of ground robot algorithm. The team are planning to use the weekends during the summer to finish what we have been working so hard to accomplish, winning the International Aerial Robotics Competition 2018. We are confident that our system will be one of the top competitors at the competition this summer.

At the time of writing this paper we are still testing and integrating our AI. The team are planning to use the weekends during the summer to finish what we have been working so hard to accomplish, winning the International Aerial Robotics Competition 2018.

REFERENCES

[1] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.Y., Berg, A. (2015). SSD: Single Shot MultiBox Detector.

[2] Douglas, H., Peucker, T. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature.*Cartographica: The International Journal for Geographic Information and Geovisualization*