Technical Paper for the International Aerial Robotics Competition

Matias Christensen Håkon Flatval Magnus Reiersen
Martin Sollie Christian Wilhelmsen

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

As this was our second year attending the IARC competition, we attempted to learn from our mistakes from last year and further improve our system. We also made much more of an effort to build a solid and extendable drone platform that can serve our needs in the coming years, without the need to redesign major parts of the system each year. Our goal was also to refine our positioning algorithm, implement a initial ground robot detection system and rewrite our high level planning system.

Conceptual solution

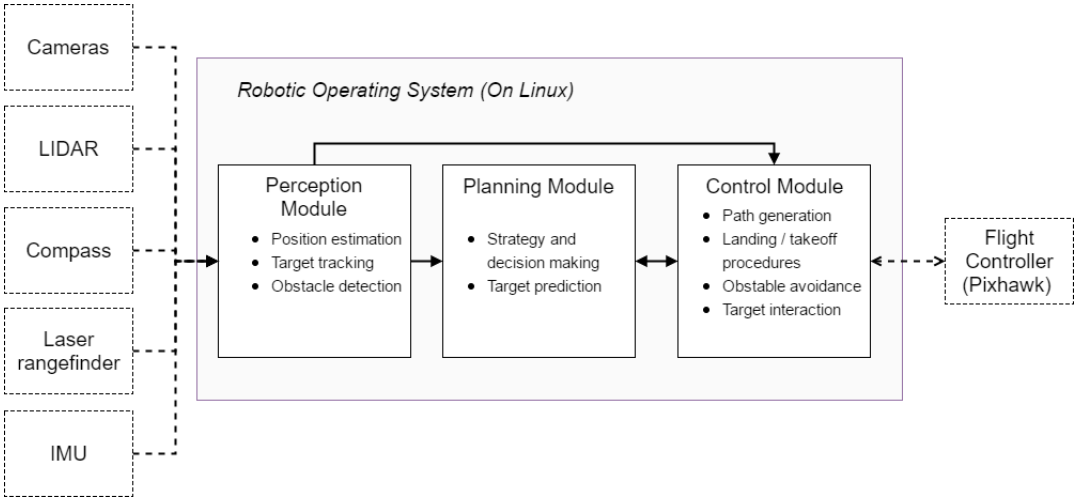
Our organizational breakdown of the problem was largely based on last year's, with the addition of a hardware group to facilitate a more ambitious focus on creating custom mechanical and electronic parts. The technical members of this year's team were broken into four groups:

- 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 four 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 four 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 2-4 Nvidia Jetson TX2. 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 deliverd 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 last year's drone.

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 Since the Grid Detector only measures the position of the drone relative to the closest grid cell, another algorithm is required to determine the actual position of the drone inside the arena. The Global Positioner takes as input images from the four side-mounted cameras, as well as the full orientation and altitude of the aerial vehicle. The algorithm determines the position of the air vehicle in the arena in the following way:

Firstly, the image is blurred with Gaussian blur, using an exponentially decreasing standard deviation from top to bottom. After this, the lines can be obtained by thresholding the gradients of the pixels and using a region-growing method to eliminate noise and artifacts outside the arena. When having this image, mostly consisting of pixels of the grid, and by utilizing the known orientation and elevation of the drone, we deproject lines known to be parallel to the grid lines into the world and iterate over them across the image to gain a characteristic that can be used to decide whether or not the line crossed the arena. Using this as a fundament, the border of the arena can be found by binary searching with these lines across the image. This in turn helps decide where inside the arena our drone is located. Combining this with the Grid Detector, we can achieve a great accuracy regarding the air vehicle's whereabouts.

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 multiple GPU's provided by several 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.

Vertical Robot Detector For precision landing and planning, the ground robots are also tracked with a downward facing wide-angle camera. This algorithm detects visible targets in each frame by identifying connected pixel regions of red or green color. For each region

we can predict the center of the target as the mean of pixels contained within the region and obtain its 3D grid position by back projecting the pixel onto the ground plane. These observations are used to track targets over time, so as to estimate their velocity and associate a consistent unique identifier to the currently visible targets. The identifier is updated by associating each new observation with the nearest existing track, or creating a new track otherwise. Tracks are considered valid once they have sufficiently many observations, and are discarded once they have not been observed for some time. Positions are updated with new detections by low-pass filtering, and velocity is computed by subtracting positions in subsequent frames.

Obstacle Detector The obstacle detector utilizes a 2D-LIDAR for finding the moving obstacle robots. The output from the sensor is a point cloud in a plane. We start by searching for relatively thin objects obstructing the LIDAR's line of sight, and use the size to determine whether it is an obstacle robot. Candidates found over multiple frames are eventually assigned a state of "tracked". While an obstacle robot is tracked, its estimated movement is handled by a Kalman Filter. The LIDAR can only detect obstacles that are in the same flat plane relative to the devices orientation, making it difficult to rely on alone. Therefore, this algorithm is supplemented by the findings of the Horizontal Robot Detector.

Planning & 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
- TRACK(id) - follow a ground robot

The planning module takes in observations generated from the perception algorithms and output commands to our control module. In our implemented solution, we have modelled Mission 7 as a Markov Decision Process (MDP) with an estimated optimal policy calculated from the current state. The planning module consists of two algorithms attempting to solve the MDP in real time, called the *Plank Algorithm* and a *Reinforcement trained network*, and one helping-subsystem to keep track of state entropy. Providing the optimal conditions for each subsystem, the control over the drone is differentiated between the three by a simple switch case.

Plank Algorithm

This MDP solver is treating each ground robots cloud of proximity as an abstract structure which we name a *plank* (given the cloud resemblance to a long stick). Each plank is tupled with its coherent, expected Mission 7 points for staying in this state (no interaction). We call these points *expected reward*. These rewards are calculated based on a distance to red or green lines. Implementing an action on a ground robot is considered to change its plank, thus a new reward is coupled. Using these set of rules, one can model actions and rewards as a graph, where each action is a edge, and each node a coherent reward. Using a depth first search, one can find the optimal set of actions to achieve the greatest gain of rewards in the least amount of steps required.

Reinforcement trained network

Under some conditions, the Plank Algorithm is not optimal. During these times, the decision system will switch over to a artificial neural network trained to make an estimated guess. This artificial neural network was trained using reinforcement q-learning on our Mission 7 simulator. It takes in the ground robots positions and outputs the learned best action.

State entropy subsystem

The MDP solvers above rely on better state estimation than just real time observations (which is limited in visible range). This subsystem was invented to keep track of what we cannot see. It keeps memory of past-observed ground robot positions out of range, model their likely behaviour over time, output continuous maps of position probabilities, and provides information on entropy of these models. If the entropy is beyond the max threshold, this subsystem takes control over the drone and calculates the optimal way points for harvesting as much information as possible in shortest time possible. As soon as entropy is reduced to a low state, control is given back to the MDP solving algorithms.

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

We made the decision this year to move the computing power from a ground based desktop connected to the drone by a wireless network, to four Nvidia Jetson TX2 carried on-board the drone. 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. For this we are creating a custom

Jetson carrier board that will provide power and IO for up to 4 Jetsons as well connecting them all on a Ethernet switch embedded in the carrier board.

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.

Communications

The four 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 through our custom made safety shutdown switch. 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.

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 onboard 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.

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 prop guards and large slow-rotating propellers. 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

With this years drone, our aim was to build a solid, flexible and extendable platform that can be built upon for several years to come. We where able to build on what we learned from last year and create a much more solid design. We made the shift from using a separate desktop

computer to moving all the computing on-board, solving multiple issues with latency and bandwidth faced by our previous drone.

We where able to make major advancement in our ground robot detection system. We are now able to accurately and reliably detect and classify ground robots, even at some distance. We also made improvements on our positioning algorithms, although this will need further work to be able to solve Mission 7A.

At the time of writing the drone still not completely finished and changed may be made as issues are discovered in the testing done up to the competition date.

REFERENCES

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