Oregon State University
Autonomous Aerial Robotics Team

Daniel Miller, Team Lead, millerd4@engr.oregonstate.edu, 541-728-8090
Kyle Dillon, Tim Niedermeyer, Ryan Skeele, Michael Williams, Soo-Hyun Yoo

1148 Kelley Enginnering Oregon State University, Corvallis, OR 97330

ABSTRACT

The Oregon State University Autonomous Aerial Robotics Team has developed an indoor autonomous quadrotor with custom hardware and software to compete in the International Aerial Robotics Competition (IARC). Onboard, an ATXmega128a3 microcontroller runs a 200 Hz PD orientation controller. The quadrotor is capable of sending live video, LIDAR scans, and altitude measurements to the base station which passes navigational commands back to the quadrotor. The quadrotor possesses a passively compliant robotic hand that will be used to pick up the USB flash drive in competition.

INTRODUCTION

Our strategy for indoor automous flight involves specific functional goals for each part of the system. Flight stability is handled by the flight platform, however, all navigational data gathered from distance sensors and cameras mounted on the flight platform is transmitted to the base station for processing. The base station then transmits navigational commands to the flight platform.

Stabilization sensory is handled using a three axis gyroscope, accellerometer and magnetometer. Navigational data comes from a number of distance sensors mounted on servos that act as an improvised LIDAR. This solution was chosen because of the low electrical and computing power required to process distance sensor data. A seperate wireless camera is implemented to provide object recognition using the OpenCV libraries.

HARDWARE

Chassis

Ultimately, we aim to construct a chassis that is lightweight, durable, modular, and safe.

For initial tests, however, a simple "X" configuration quadrotor was assembled from

laser-cut plywood. The symmetric construction of the "X" allowed us to make convenient assumptions about the vehicle's flight characteristics.

The final frame is constructed in an "I" configuration, which simplifies the task of mounting cameras and distance sensors on the periphery of the quadrotor. The carbon fiber booms are strong yet lightweight. The booms are also separable from each other, facilitating repairs and modification. The motors are mounted to the frame with aluminum clamps. Similar clamps are used to mount the sensors elsewhere on the booms. Finally, carbon fiber shrouds prevent any of the four propellers from contacting objects or people.

End Effector

One of the primary objectives of the competition is to grasp a USB flash drive. Our grasping mechanism will consist of an under-actuated, passively compliant four-fingered hand. Each of the four fingers will contain two flexure joints. A single cable runs through the center of each finger to its tip. These four cables are actuated by a single linear actuator. The use of only a single actuator in this under-actuated system helps keep down weight and cost and allows the hand to automatically adapt to the shape of the object being grasped without the need for a separate control system.

Electronics

A single 3-cell lithium polymer battery powers the motors and motor controllers on the quadrotor. The battery is also regulated down to 5 volts to power an Atmel Xmega microcontroller, a 2.4 GHz XBee radio, and various distance and inertial sensors. The wireless camera is powered by a seperate 9 volt source.

The organization of the electrical system is illustrated in Figure 1.

SOFTWARE

Aerial Stabilization and Flight Control

Orientation Kinematics

An accurate measurement of the quadrotor's orientation is key to autonomous stabilization. The orientation can be represented with a direction cosine matrix (DCM), which is a 3x3 matrix containing the cosines between each of the 9 possible pairs of axes of two separate Cartesian coordinate systems.

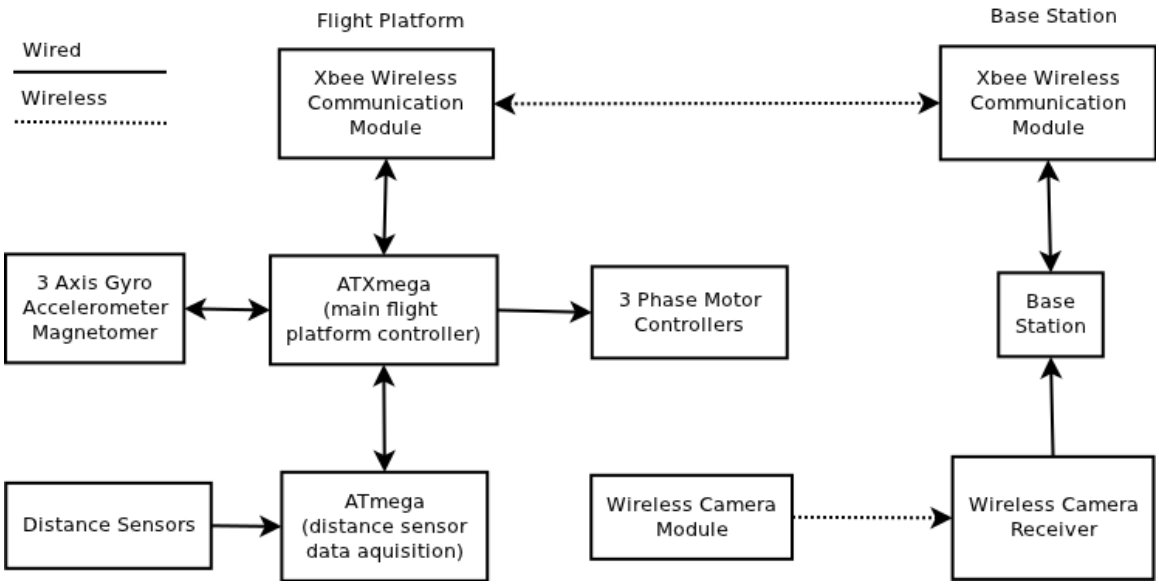

Figure 1: Control Hardware Block Diagram

In the context of inertial measurement in robotics, a 3D vector could be represented in either the global (earth) or the local (body) frames of reference. For example, the location of an end effector may be represented as $\langle 1, 0, 0 \rangle$ in the local frame but have different (and changing) X, Y, and Z components in the global frame, and vice versa.

If the controller loop frequency is high enough (on the order of 100 Hz), an axis of the DCM, its rotational vector, and its linear velocity are approximately orthogonal to each other. Thus, the magnitude of the angular velocity of a unit vector approximately equals its linear velocity, which means that the DCM can be calculated by integrating the gyroscope readings.

Unfortunately, since the gyroscope only measures the change in the orientation, the DCM will drift over time. A 3-axis accelerometer can be used to correct the roll and pitch drift. Similarly, a 3-axis magnetometer can be used to correct for yaw drift. With these corrections, an accurate DCM can be maintained.

Cascading PID Controllers

The D controller in a simple angular position PD controller can hinder the responsiveness of the quadrotor, which is crucial to stability. A better alternative is an angular position P controller that feeds a desired velocity to an angular velocity PD controller.

At first thought, it might seem that this combination of a position P controller and a velocity PD controller is no different than a position PD controller, since both the position D and the velocity P are based on velocity measurements. However, the two controllers do different things with the velocity measurements.

The position D controller has a damping effect on motion in that it always resists a change in position. This means that if the quadrotor experiences a perturbation away from some desired position, the D controller will help resist the motion, as desired. However, as the quadrotor tries to recover from this error state, the D controller blindly hinders the movement back towards the desired position, which is not at all optimal.

The velocity P controller, on the other hand, pushes the velocity to whatever it should be, whether that means slowing it down or speeding it up. The velocity D controller ensures that the velocity change does not happen too abruptly, helping reduce the chance of overshoot. The controller is able to do this by keeping track of a desired velocity in addition to the current velocity, which provides a context upon which the controller can "decide" whether it should help a positional movement or hinder it, instead of blindly hindering all movement as is the case with a position D controller.

This means that the P/PD controller as a whole can take a desired position input and accelerate the body towards and maintain a target angular velocity until the current position nears the target position. Only then will the controller actively slow down the movement. This makes for a controller that can respond much more quickly while maintaining stability, making flight possible (Figure 2).

Figure 2: Prototype Flight Platform Operating Outdoors

Navigation

Navigation Hierarchy

Each component of our system has a specific responsibility with little overlap with those of others. The onboard microcontroller is solely responsible for maintaining angular orientation. The microcontroller streams data from distance sensors through the XBee link to the base station. A video stream is sent to the base station from a wireless camera. The base station is responsible for processing this data and send navigational commands back to the quadrotor in order to complete the mission.

Navigation Strategy

In order to navigate indoor environments we implement an array of simple distance sensors, with some mounted on pan or tilt mechanisms, to serve as an improvised LIDAR system. This approach was chosen because of its accessibility and low power and bandwidth requirements.

Distance sensor data gathered on the flight platform is sent to the base station. A navigation system at the base station processes both distance sensor data and images sent from the flight platform to make navigational decisions. These decisions are then relayed back to the quadrotor as movement commands.

Robot Operating System

Robot Operating System (ROS) is a collection of libraries and tools that allow developers from around the world to contribute software packages such as device drivers, messaging libraries, and visualizers in a consistent format for others to use. We have developed our navigation system to operate within this software architecture. This allows us to use third-party joystick drivers, point cloud libraries, and SLAM algorithms, which frees us from having to reimplement solved problems.