Autonomous airship

Introduction

The objective of the project is to build a small airship which will be able to control altitude, move and avoid obstacles. The airship will consist of a helium balloon and motors which control speed, direction and altitude. To measure quantities we will also need appropriate sensor. Additionally it should also be possible to control the ship with a bluetooth device using a bluetooth connection.

Starting point

There were several initial challenges to be considered:

- The weight of the machine has to be carefully optimized, as the balloon can carry only so much weight
- The airship will require sensors to detect obstacles, selecting and configuring these may prove problematic
- Bluetooth connections and communications might prove difficult

Targets - Primary/secondary:

Primary:

- The airship can fly on its own (detect and avoid obstacles)
- Airship can maintain and correct altitude

Secondary:

- The airship can be controlled wirelessly
- The airship can follow people
- The airship can be guided by hand

Our preliminary idea is to have a balloon which has 3 motors: one motor to control the altitude and two motors to control the speed and direction. These motors will be on the sides, and you can control direction by turning one motor off or switching its direction. In other words, the airship would turn in a way similar to how a tank turns.

Realization
Research & Component selection

Sensors

To be able to sense surroundings properly the balloon needed at least three distance sensors and an additional sensor that measures the altitude. Several options for these were available but we decided to settle on ultrasonic sensors due to their lightness, wide measuring angle, easy application and affordability. Sensors are aligned so that two sensors point sideways (to detect obstacles on the sides), one points forward and one points down.

Motors

We considered many options when selecting motors. We eventually settled on x-dart quad copter brushed DC motors. These motors are quite lightweight, only 20g, while still providing adequate thrust for our airship. In fact, despite their small size, the motors turned out to be more than enough for our airship, so much so, that we only ran them at 10-20 % of maximum power. To avoid any possible compatibility issues, we also bought the propellers which were meant for these particular motors. Two motors were placed on the sides of the balloon to allow turning, one one was placed inside the frame. The last motor controls balloon altitude.

Balloon

For the balloon there were two popular choices for these kind of devices: mylar film and latex balloons. The main advantage in mylar film balloons is their durability. However premade mylar balloons were costly and they had too low carrying capacity. Also making a mylar balloon on our own would have proved quite challenging: We read from several sources that making airtight balloons out of mylar film sheets would be very hard. Due to these reasons we decided to settle on latex balloons which only required filling with helium. We calculated that 120 cm long ellipsoidal balloon with 80 cm diameter would be sufficient.

Gondola

Our device frame (the gondola) was made of Design factory proto foam. It's a styrox-like material, only more durable while still being lightweight. We made the frame by carving out a space inside the foam. The frame has a lid, which is attached to the frame by magnets, for easy removal. The lid is then attached to the balloon by rubber bands. We also have a crossbeam for housing motors. It's a simple piece of plywood with motor holes drilled into it. The plywood crossbeam also has space in it for attaching weights. Weights were used to balance the balloon in such a way that it's just heavier than air, so that in improbable case of a system failure, the balloon will slowly descend by itself.

Electronics

We chose Arduino Nano to be the main microcontroller of the device since we needed a small and lightweight controller due to the limited space inside gondola and the weight requirements. Another advantage with Arduino Nano is that it fits easily to the breadboard which makes prototyping and testing the electronics easier.

Coding

The airship is controlled by Arduino Nano, which is coded using c-code. For our ultrasound sensors, we used stock functions to control the sensors and output the distance information. Based on this information, the automated program can detect obstacles on its path and adjust course accordingly. The program also has an automated system for managing the altitude. An ultrasound sensor reads the ship's altitude and adjusts height motor accordingly, so as to maintain level altitude. Ship also has an option for manual control via bluetooth. By default, the altitude control stays active even when the ship is controlled manually. The source code is attached below.

Building of electronics

In hindsight, our frame could have had more space. But since space was limited, we couldn't want to use circuit boards or prebought breadboards. Instead we soldered some wires directly and used connectors for others. Since our frame didn't have enough height, we had to bend some ultrasound sensor pins and then directly solder the wires into those pins. The schematic of the electronics is found in the attachments with filename AIRSHIP_CIRCUIT.pdf.

Testing

We hope to have everything ready several weeks before mechatronics circus. We will be thoroughly testing core functions, and once we get those working we will spend the remaining time testing, improving and developing our secondary targets.

User manual

Balloon assembly

1. Fill the balloon.
2. Wrap the gondola lid rubber bands around the balloon.
3. Attach the gondola to the lid (both the lid and the gondola have magnets for this)

Uploading the program
1. Make sure that the battery and the bluetooth module are disconnected from the Arduino.
2. Connect the Arduino to the computer via USB-cable.
3. Open Arduino IDE and upload the airship_bt.ino to the Arduino. (Note: that airship_bt.ino requires airship.h.)
4. Once uploading is complete detach the USB-cable and connect the bluetooth module and the battery to the Arduino.

**Flying the ship**

**Connection between Bluetooth module, controlling platform and smartphone**

In this case we used smartphone to create connection to the Bluetooth module and control Arduino with Bluetooth terminal. There are several suitable apps for transferring data between smartphone and module. In this project we used app called Bluetooth SPP and with that send simple commands to the Arduino. It is possible to use "command line"-type IO terminal or use programmable keys.

When uploading code or sketches to Arduino via USB, Arduino RX/Bluetooth TX pin should be disconnected because this may prevent connection between PC and Arduino.

**Step-by-step:**

1. Disconnect Arduino RX/Bluetooth TX pin
2. Upload sketch/code from PC
3. Connect Arduino RX/Bluetooth TX pin
4. Set Bluetooth ON from your platform device (in this case smartphone)
5. Search available bluetooth devices (default name HC-06)
6. Pairing with bluetooth devices (at this point prompt the device password, usually default 1234 or 0000 in bluetooth devices)
7. If pairing was successful, open the Bluetooth SPP
8. Connect to device "HC-06"
9. Choose IO mode: "Keyboard mode"
10. Set customizable keys and start controlling your Arduino

The ship controls take a little getting used to. The ship has some inertia, so control movements occur at a small delay. The control scheme has directional controls: forward, backwards, left and right. Even the up and down movements can be controlled manually, even though these are best left to the automatic altitude control system. The program also has a specific command for controlling the throttle (motor speeds). This is used in combination with the movement commands (the ship will not execute any commands if throttle is set to 0). The ship can fly surprisingly fast, so when applying throttle, lowest possible speeds are recommended.

In the current control scheme, you have to specifically ask for the remaining battery life, and stop the airship if the battery is about to run out. If the battery is about to run out, it has to be disconnected, as letting it run too low might damage the battery. Instructions about how to charge the batteries can be found in the charger's manual. There aren't suitable plugs for the connection between the battery and the charger so the user has to improvise those or buy some proper wires and plugs.

**Conclusions and suggestions**

Overall, aside from a few faulty sensors, we met both our primary and secondary goals. The airship is able to fly and is relatively easy to control manually with a bluetooth connection. It can also maintain and control a set altitude. Moreover the problem of the weight of the airship and the lift created by the helium balloon can be easily overcome with washers and nuts attached to the airship. The power created by the motors selected proved to be more than enough and the airship moves fast with just 20-30% of the motor power.

The next step should be fixing (or replacing) two of the direction sensors and the direction control. For further development, the bluetooth control could be extended to internet control. Another improvement could be making a bigger, more spacious body as there is enough lift to support more weight. This would make servicing and fixing of all the parts easier.

**Schedule**

Estimated milestones:

- Research & component selection: Friday 15.2.
- Parts acquired by 1.3.
- Coding: Preliminary coding done before parts arrive, fine-tuning after that
- Assembly: After parts arrive
- Testing: 15.3.

**Budget**

4 Motors & propellers: 11,71 €
4 Sensors: 8,72 €
Helium balloons: 11 €
Arduino nano: 9,99 €
2 Motor controllers: 4,50 €
1 Battery charger: 18,59 €
3 Batteries: 11,97 €
Total cost: 76,48 €

We built the frame ourselves from leftover materials found at Aalto design factory. We got our helium from the mechtronics circus organizers.

**Team Members**

Pyry Kärki
Jyri Lehto
Juho Halinen
Mauno Kemppainen
Aleksi Turunen