Blimpcast-Autonomous Blimp Livestream
Dane Schoelen
Norman Advanced Robotics
DaneSchoelen@gmail.com

Blimpcast- Autonomous Blimp Livestream

Introduction

Last year two members of the Norman Advanced Robotics team brought a robust live streaming system to the Global Conference of Educational Robotics. NAR intends to continue to improve its capabilities to bring the Botball conference to anyone with a internet connection. This year we will be adding an autonomous blimp bearing a streaming webcam to the fleet of live feeds. The blimp, dubbed Blimpcast, will provide an eye in the sky view of the conference that will give viewers the ability to experience the conference as a whole instead of just one table at a time. Used in conjunction with the table streams, Blimpcast will provide unprecedented coverage of the conference that can be accessed from around the world, in real time.

1 Purpose

Last year at the Global Conference for Educational robotics Norman Advanced took third place. Through the streaming service our team built last year, my parents who were still in Norman were able to watch as we competed day by day. This was wonderful and provided my parents with a way to support our team without actually being there. So we began to contemplate how to make the service better. What if my parents could tune into a stream that is traveling around the conference, getting a look at the practice areas, the team pits, and the competition boards, all from a birds eye view. They would feel much more connected to the conference because they could get a sense of all the activity, not just what is taking place on the boards. Blimpcast can provide such an service, and so NAR set out to build it.

2 A Streaming Head Start

The Blimpcast project stands on the shoulders of the Norman Advanced members that created the table streaming project. Half of the challenge of creating Blimpcast is having the capabilities to stream wirelessly. Fortunately all of the leg work for that portion of the project has been covered by the streaming project, leaving just the challenge of building and programing an autonomous blimp. It is still no small feat, but we are happy to be able to build upon the experience gathered by others on our team.

3 Inspiration

Blimpcast will not be the first autonomous blimp, in fact in our research we found two autonomous blimp kits that were fairly cheap and easy to assemble[1][2]. We decided

to build our own build from scratch for two reasons. First, neither of the blimps we found had enough payload capabilities for a camera system, and second the kits did not come with any room for expansion. We intend to give our blimp as much sensing and movement capabilities as possible, which would not be supported in either of the kits. Bear in mind that we did use the projects found online as inspiration, but all of our design is original.

4 Design

The first question when starting an autonomous project is what kind of vehicle to build. In our case, we chose a blimp because of its stability. It can support itself in the air for hours without power and poses no risks to GCER participants as it flys above them. A blimp also has the capability to stay in the air for longer than most aerial vehicles thanks to the lift mechanism being very energy efficient. It can make minor adjustments and lightly propel itself to get around a room, so it does not consume much battery and can stay up for longer. Lastly, a blimp can carry a relatively high payload, which is ideal when dealing with a wireless video system.

4.1 Payload

Blimpcast will be supported by two smaller blimps side by side. This is because larger blimps are handmade and therefore very expensive. So we decided to use two hobby blimps that are far cheaper and could be replaced easily in case of failure. The gondola hangs just beneath the blimps and carries all the payload. This includes the Arduino microcontroller, battery, sensors, motor divers, wireless serial communication transmitter(XBee), camera radio transmitter, and servos. Excess lift will be countered with weights and the blimp will be balanced before flight to ensure that it can hover without any effort.

4.2 Propulsion

The propulsion system consists of two propellers mounted on an axle that extends out 5 inches from each side of the gondola. The propellers are small and fairly weak causing the blimp to make very slow and gradual movements, ideal for conserving battery and saving weight. The propellers can be throttled up and down independently, allowing the blimp to turn. The supports for the motors is actuated by a servo inside the gondola allowing it to rotate back and forth. This motion results in the propellers turning up or down enabling the blimp to control its altitude. An optional tail fin mounted propeller, would allow the blimp to turn faster if the main propellers prove to be too slow.

4.3 Sensor array

There are three ultrasonic range finders, one on top for altitude measurements, and two in front for navigation. The two front mounted sensors will be facing out front and to the side of the blimp, giving the blimp two readings to decide which direction to turn when encountering an obstacle. Four IR light sensors face out each side of the blimp in

order for the blimp to navigate using the IR transmitting ground beacon. The streaming camera will be mounted on the underbelly of the gondola and be capable of pan/tilt using two mini-servos.

5 Programing and Logic

Blimpcast will be piloted by a Arduino microcontroller inside its gondola that runs C-based programing environment. The microcontroller will be capable of serial communication to a ground based Arduino for inputs from an operator such as flight mode, camera position, and even navigation override. The steaming system will be separate from the microcontroller, but the arduino will still control the pan/tilt functions of the camera. The ground operator will be able to set flight modes from the ground so the blimp does not have to land for new instructions.

5.1 Flight modes

Blimpcast will operate under one of four sets of behavioral parameters, or flight patterns. The Arduino will use specific parts of its code based on the flight mode the operator has set. The first flight mode will be totally autonomous flight where the blimp flies in a straight line, avoiding obstacles when necessary. The second will be a random flight mode where the blimp will randomly change course periodically, but will still avoid obstacles. Third is ground beacon following where the blimp attempts to stay as close to the beacon as possible. The blimp will monitor which of its IR sensors is receiving the most light from the ground beacon and turn into that side. This behavior results in a circular flight pattern around the beacon. And last is RC control where the blimp disables its autonomous functions and allows the ground operator to pilot through the serial link. In every flight mode.

Refrences

[1] "BlimpDuino Home Page - DIY Drones." *BlimpDuino Home Page - DIY Drones*. Web. 04 June 2012. <http://diydrones.com/profiles/blog/show?id=705844:BlogPost:44817>.

[2] "Ollie â a DIY Autonomous Robotic Blimp." *Me and Ollie*. Web. 04 June 2012. <http://meandollie.com/>.