RCHeli: Infrastructure For PC-Controlled Micro Helicopter

Sanghoon Cha

Brown University
scha@cs.brown.edu

1 Abstract

Our goal is to have a stable, extensible system that allows us to control the micro helicopter and further become a basis of autonomous flights. In this project, we use several commercially available components to set up a basic system that is capable of integrating Wiimote[1] control and camera vision. The architecture consists of an RC helicopter, PCTx device[2], DX6 transmitter[3], a wireless camera, and a Wiimote. This system is currently rather incomplete; however, it has great potential to be extended into many different capable directions, such as Unmanned Aerial Vehicle(UAV) project. Our goal is to have this system being low level support for future projects.

2 Introduction

Robots have been increasingly becoming an integral part of people's lives. Some of them used in research are custom made, tightly integrated with engineering, but the ones closest to people are commercially available robots, such as radio controlled devices. AIBO[4] and iRobots[5] are front-runners of user-friendly robots that are actively used in research areas as well. Radio controlled devices such as RC cars and helicopters are are also used due to its simplicity of control, and inexpensive price. In this project, we focus on RC helicopters and state how this system allows a person to control the helicopter from a personal computer.

We use a widely known controller, Wiimote, rather than a normal radio transmitter in this project, because it allows us to gather user control data; for example, the Nunchuk joystick movement is digitized into a number between -1 and 1. With this, we can further use this system in machine learning, mainly in Learning From Demonstration[6].

3 Architecture

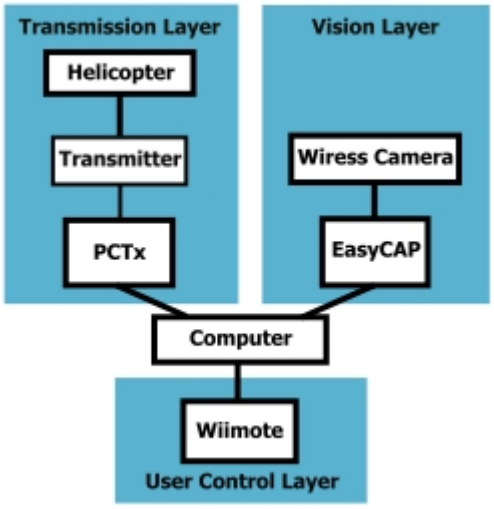

Figure 1: Architecture of RCHeli system

There are mainly 5 components in this system. First, the very essential component of this project is the helicopter. We used Esky 6-channel Belt-CP helicopter, which is capable of 4 directional moves; throttle, roll, pitch, and yaw, using Cyclic/Collective Pitch Mixing (CCPM).

We will describe this in detail in section 3.1. Furthermore, we introduce a new transmitter, Spektrum DX6, as well as Endurance R/C PCTx device; two components combined allows us to send radio signal from computer. A small and light wireless camera from LYD provides a fairly reliable video feed over 2.4GHz, and it is used in inside-out vision from the helicopter. Finally, a Wiimote is used by a person to control the helicopter using the devices described above. All of the devices that we used in this project is fairly inexpensive and easily obtainable from Internet, with total $< \$500$.

3.1 Helicopter

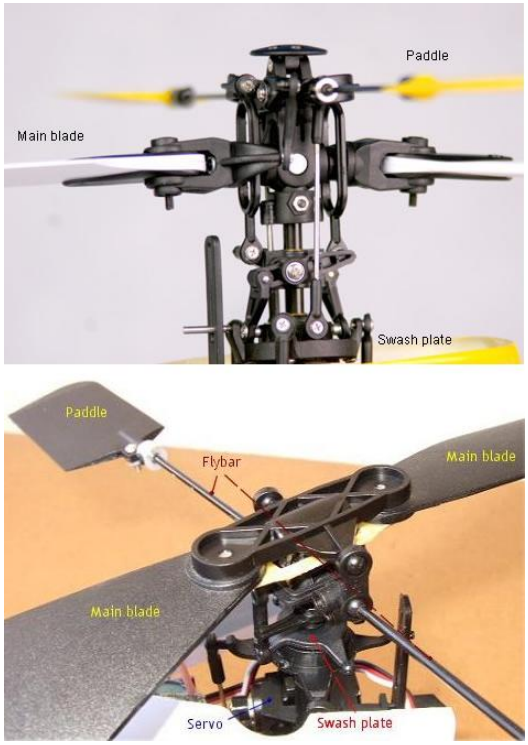

Figure 2: Comparison of Head Assembly (Top, 6-channel; Bottom, 4-channel)

In general, RC Helicopters can be classified by its number of channels: 2-channel, 3-channel, 4-channel, and 6-channel. Ones with less than 4 channels have limited mobility. For example, a 2-channel helicopter may only be able to move up, down, left, and right. Generally, 2 to 4-channel helicopters are considered to be a beginner level helicopter, and tend to be smaller and less complicated. In contrast, 4-channel and 6-channel helicopters are for intermediate or advanced flyers.

The main difference between 4-channel and 6-channel ones is that 4-channel ones come with fixed pitch, and 6-channel ones come with collective pitch(See Figure 2. Pitch in a helicopter refers to the angle of the blade with respect to the horizontal plane. Thus, a helicopter with a fixed pitch will have main blades remaining in a fixed position, whereas one with a collective pitch will allow the pitch of each main blade to change, independently if wanted. To put it simply, fixed pitch (4-channel) helicopters are easier to fly, since there is one less thing to consider. The helicopter we used is primarily intended for intermediate/advanced flyers, and therefore capable of omnidirectional movements as well as 3D aerobatics. Basically, the movement of the 6-channel helicopter is dependent on 4 different servos: aileron, elevator, pitch, and rudder.

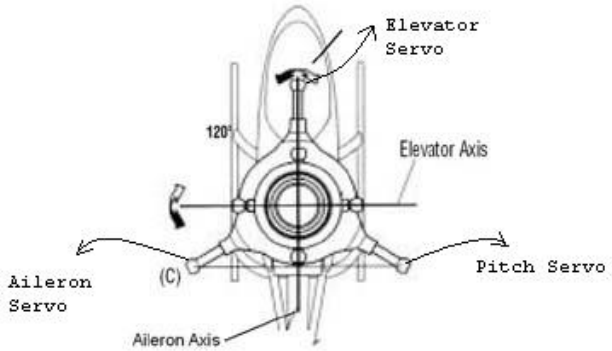

Figure 3: Cyclic-Collective Pitch Mixing (CCPM)

A 6-channel helicopter moves according to the position and angle of its swash plate. A swash plate is a core element which controls the pitch of the main blades both in a *cyclic* way, which results in forward,backward,left, and right movement, and in a *collective* way, which controls height and vertical movement. Figure 3 illustrates the relationship between the swashplate and the three servos; aileron,

elevator, and pitch. Normally, when controlled from a transmitter, pitch servo is not directly mapped from the position of a control stick; in fact, it is calculated with regard to aileron and elevator servos according to the CCPM mixing formula, which will be explained in detail in section 4. Rudder servo, which is independent from other three, rotates the helicopter left and right, controlling yaw of the helicopter.

Esky Belt-CP is one of well-known professional micro helicopter of length 27 inch and weight 670g. We chose to work with this helicopter despite its large size and heavy weight because we may want to attach additional devices onto the helicopter and we wanted it to fly without any problems; namely, we wanted to attach a Wiimote as an accelerometer, which weighs about 200g. Through some experiments, it was able to take off the ground without any problem, despite additional weight.

3.2 Transmitter

Esky Belt-CP helicopter comes with default 6-channel transmitter EK2-0406, which works with a default receiver EK2-0420A, transmitting the radio signal at frequency 72MHz. It has a *trainer* port which allows a connection between the transmitter and another transmitter or a computer, allowing signal forwarding from the transmitter. One can use the USB cable to maneuver the controller as a joystick to run the simulator/practice flight program. However, it was unable for us to send signals into the transmitter, which is what PCTx device does, and therefore there came a need to buy a different transmitter/receiver package. We have acquired Spektrum DX6 transmitter with AR6000 receiver, which works in 2.4GHz band. This transmitter was known by PCTx manufacturers to be functioning with PCTx. In our project, transmitter works as a signal forwarding device, from PCTx to the receiver, and therefore we do not utilize the full functionality. The *trainer* button on the transmitter must be pressed at all times to allow the signal to be forwarded.

Channel #	Function
1	Throttle
2	Aileron[1]
3	Elevator
4	Rudder
5	Gear(unused)
6	Pitch

Figure 4: Mappings of channels to functions

3.3 PCTx

Normally, a computer cannot generate a radio signal on its own, and that is why similar projects use their own circuit board or even a small single-board computer with bluetooth technology;i.e. Gumstix[7]. However, this requires an extra installation inside the helicopter and additional complications in terms of coding since the Gumstix runs Linux. Meanwhile, Endurance R/C's PCTx device was a perfect fit for this system, because of its simplicity and inexpensive nature[2]. PCTx provides a USB connection from any type of transmitter to a computer, and using the manufacturer's provided API, we can easily send data through up to 9 channels. While the configuration may differ with a different transmitter, the current setup is described in Figure 4.

3.4 Wireless Camera

While controlling the helicopter is important, in order for us to move further onto autonomous flights, the inevitable portion of a system is tracking. It can be done, for example, by having a GPS device on the helicopter, as done in Stanford's Autonomous Helicopter project[8]. However, this is only possible when the helicopter is big and heavy enough that an additional component attached would not affect the helicopter's functionality. However, we use micro helicopters which are of smaller size, and

[1] Currently this is mapped onto channel 5 due to the broken receiver which doesn't receive signal in this channel; of course, the aileron servo is connected to Gear port in the receiver.

adding a wireless GPS device may complicate the system due to its weight as well as size.

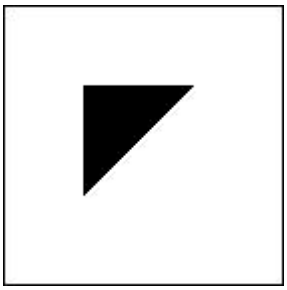

Figure 5: Helipad used in tracking

Therefore, one viable option is using computer vision. Using cameras, we can track the helicopter quite efficiently and frequently. There are two types of computer vision: *outside-in* and *inside-out* vision. Outside-in vision is having an external cameras at ground, wall, ceiling, etc. to distinguish the helicopter from background and calculate the location and orientation from those images. However, inside-out vision has a camera attached to the helicopter, resulting in an location-independent system, which can function inside or outside. Using a "helipad" of shape from Figure 5, we can calculate height, orientation, and estimate of location of the helicopter, assuming that it is somewhat close to the helipad. The simplicity of the helipad was a result of an effort to minimize computer vision overhead; however, it can be replaced with a bigger, complicated pattern that can be then analyzed by augmented reality libraries, such as ARToolkit[9].

Wireless camera that we use here is a pinhole camera, attached at the bottom of the helicopter, looking down. It constantly sends signal through 2.4GHz band to receiver using 9V battery as a main source of power. The receiver then feeds the signal through Composite Video (RCA), and we use EasyCAP[10] device that converts the signal into USB signal that could be treated like a webcam. Since the device converts the video feed into usb feed, we can use any computer vision library to analyze the images.

3.5 Wiimote

Wiimote is a primarily a controller for popular game console Nintendo Wii[1]. Due to its innovative technologies, such as IR camera in front of a controller that recognizes infrared lights from a sensor bar and more importantly, accelerometer. The built-in accelerometer provides acceleration data in x,y, and gravity direction. Using the data we can apply more user-friendly controls such as gesture recognition. In this system, we use Wiimote as our primary controller, where the joystick in Nunchuk controller controls throttle and rudder of the helicopter, and a tilt of the Wiimote is directly represented as a tilt of the helicopter; tilt can be either pitch, roll, or a combination of two.

4 Implementation

In this section, we will describe the specific implementation that we used to combine the system together. We used Microsoft Visual C++ 2005 with Windows XP, primarily because the API that the manufacturer of PCTx supplied only supported Windows. For computer vision, we use OpenCV library [11], which provides very extensive support for various needs, including object tracking and even machine learning. Finally, we make use of WiiYourself! libary[12], the third-party open source library known to be fully functional in Windows.

Currently, the application is comprised of three threads. The main thread first connects to the PCTx device and a Wiimote, possibly two if another Wiimote is used as an accelerometer in the helicopter. Then it creates the other threads, and continuously poll on Wiimote data. Another thread writes to PCTx every 50ms, only if the Z button in the Nunchuk controller has been pressed. Finally, the third thread keeps polling images from the EasyCAP device with a frame

```
double throttle, aileron, elevator,
       pitch, rudder;
int t,a,e,p,r;
void setRudder(double val);
void setPitch(double val);
void setRoll(double val);
void setThrottle(double power);
void calc();
void send(controller *c);
```

Figure 6: Structure of HeliData class

Servo	Position	Value
Throttle	Low	90
Throttle	High	200
Aileron	Low	127
Aileron	High	184
Elevator	Low	112
Elevator	High	160
Pitch	Low	180
Pitch	High	122
Rudder	Center	137

Figure 7: Values mapped to PCTx

rate of 50 frames per second, and each image being 720×480 pixels; then it analyzes to find the helipad and calculate orientation, height, and position of the helicopter. These threads take advantage of HeliData class structure.

4.1 HeliData

HeliData class, as we can see from Figure 6, supplies methods to simplify user's code which will consist of a series of method calls and keeps track of internal state of each servo, ranged from 0 to 1 (double variables). As explained above in Section 3.1, each servos are controlled by collective and cyclic pitch mixing. The throttle value being 0 means that the motor is not running, and 1 means full throttle; thus, this value only controls the power of the motor. Aileron, elevator, and pitch servos have 0 values when they are at the bottom, and 1 when at the top. Finally rudder is at half when the helicopter does not rotate, and 0 when it rotates left most rapidly, and 1 when rotating right.

These values are calculated by both collective and cyclic mixing; for example, when throttle increases, the swashplate must move up accordingly because otherwise it would not take off from the ground; this will result in an increase in values for aileron, elevator, and pitch values. This is called collective pitch mixing. When we set positive pitch, meaning that the helicopter will move forward, we decrease the elevator value and increase both aileron and pitch values by the same amount to tilt the swashplate forward. Similarly, negative pitch will result in an opposite movement. When we set positive roll, meaning that the helicopter moves to the right, we decrease pitch value and increase aileron to tilt the swashplate to the right. Again, the negative roll will result in an opposite movement. These are called cyclic pitch mixing.

These values, however, does not map to the actual data that needs to be sent to PCTx. The range of numbers that we can send to PCTx ranges from 0 to 255, and working range differs for each servo. We need to experimentally find the working range, and map those values onto our range of 0 to 1. For example, aileron servo at the lowest position maps to 127, and highest to 184, meanwhile pitch servo at the lowest position maps to 180 and highest at 122. The actual values that we used in the program is shown in Figure 7.

4.2 Vision

Here we briefly about the vision algorithm that we used to track the helipad. The helipad consists of a black right-angled equilateral triangle in a while background. Therefore, it is natural to increase the contrast of an image to distinguish black and white better. After adjusting the contrast, we then use OpenCV library to obtain a list of contours from the image. For each contour, we approximate to a polygon again using a function in the library. We find a big enough

triangle that inside is filled black. This will give us the coordinates of the vertices and then we can apply some calculations to find out orientation, position, and height. Currently, this is implemented as simple, but not correct manner, which uses the fact that the area is inversely proportional to height squared, and the orientation is the direction from the midpoint of triangle's hypotenuse to the vertex with right angle.

5 Issues/Future Work

This system is unfortunately incomplete by itself; although it is functional, it is not stable enough for us to try flying high and smoothly. This may primarily be due to incomplete adjustments of the numbers; with wrong numbers, the helicopter rotates or skews in a certain direction, and adjusting the numbers in Figure 7 independently will help more stable flight. Also, the helicopter's fragile nature became a huge obstacle. Once it crashes into something or into ground, it is costly and very time-consuming because we would need to order the parts that need to be replaced online. In fact, this delayed this project by weeks; creating video out of this instable system proved to be difficult.

Besides these issues, as we mentioned in the introduction, this system has a great potential to grow and be extended. First, with the computer vision code that correctly calculates the position and orientation of helicopter, this system can be used to have the helicopter stay in one position. Once that becomes possible, we can move on to give tasks to the helicopter; for example, to hover following a given path, or to go to a certain point and pick an object up and come back to original position. One possible approach that we can try in the future is that rather than using 6-channel helicopter, we can use 4-channel fixed pitch helicopter, which will save us time taken in calibrating the numbers, since there are fewer numbers to adjust.

More importantly, this can be used in a learning system, where we control the helicopter with Wiimote, record the controls as well as the position, orientation, height, and possibly accelerometer data in order to map our controls to helicopter's state. This way, we can apply learning from demonstration approach, thereby achieve a 3-dimensional robot learning.

6 Conclusion

This RCHeli system utilizes commercially available inexpensive devices to provide a full control over a micro RC Helicopter. As we progress on this project, we have found some similar projects on the website, such as CopterControl[13], ChRoMicro[7], etc. One main difference between this project and theirs is that we did not build anything and therefore it is easier to understand and use the system without prior knowledge. Moreover, the top-down vision incorporated in the helicopter allows tracking the helicopter virtually anywhere. In conclusion, this system provides an important building block that lies at the low level, and any programmer can start using this system and extend to achieve more functionality.

References

[1] http://en.wikipedia.org/wiki/Wii_Remote

[2] http://endurance-rc.com/pctx.html

[3] Spektrum DX6, http://www.spektrumrc .com/Articles/Article.aspx?ArticleID=1535

[4] AIBO, http://en.wikipedia.org/wiki/AIBO

[5] iRobot, http://en.wikipedia.org/wiki/Irobot

[6] Learning multi-objective control policies from demonstration, http://cs.brown.edu/people/dang/ publications/GrollmanJenkins_IROS2008 _WKSHP.pdf

[7] ChRoMicro(Cheap Robotic Microhelicopter) http://www.pabr.org/chromicro/ doc/chromicro.en.html

[8] Learning for Control from Multiple Demonstrations, http://heli.stanford.edu/papers/ coatesabbeelng_icml2008.pdf

[9] ARToolkit, http://www.hitl.washington .edu/artoolkit/

[10] EasyCAP USB video capture adapter, http://easycap.co.uk/

[11] OpenCV, http://opencv.willowgarage.com/ wiki/

[12] WiiYourself!, http://wiiyourself.gl.tter.org/

[13] CopterControl, http://coptercontrol.sourceforge.net/