Drone Control

Future User Interfaces Course
University of Fribourg

Adrian Kurt
Mansour Hamidi
Urs Zysset

May 2015

Contents

1 Introduction

One of the most important parts of any program is the user interface since it is the only way a user can interact with a program. Therefore, the quality of the user interface strongly affects the value of a program.

Human-computer interaction has not changed fundamentally for nearly two decades. Most users interact with computers by typing, pointing, and clicking. The majority of work in human-computer interfaces in recent decades has been aimed at creating graphical user interfaces that give users direct control and predictability [1]. However, computing devices are becoming smaller and ubiquitous, and interaction with them is becoming more and more pervasive in our daily lives. A profound shift is now occurring toward users' natural behavior as the center of the human-computer interface. Multimodal interfaces are being developed, which permit our highly skilled and communicative behavior to control system interactions like never before. Our voice, hands, and entire body are becoming the ultimate mobile multimodal input devices [2].

1.1 Motivation

Due to their small size, precision control, agility, power and stability, drones for personal use get more and more popular. Drones, also called quadcopters, are mostly used by hobby pilots and cameramen to fly for fun, or to record from the air, respectively. The control of these drones is different depending on the manufacturer. Many manufacturers sell their drones with a conventional remote controller, other manufacturers provide a smartphone application that enables the customer to easily control the drone with the own smartphone. Due to the fact that drones were becoming popular recently, and the fact that conventional remote controllers really fulfill their purpose, only a few other user interfaces have been researched to fly a drone. For this reason, we devote ourselves to the task of finding a new way to control a quadcopter drone.

1.2 Task Formulation

In order to meet the requirements of the course, we develop and evaluate two multimodal interfaces in order to determine which of these interfaces is more immersive to control a drone.

The control of a drone can be accomplished along three dimensions. Like an aircraft, particularly a helicopter, the drone can tilt forward, backward, right and left to move into the given direction, or spin right and left to turn in a certain direction. Of course the drone must also have the ability to go up and down. So, in total we have four movements: tilt onwards, tilt sideways, spinning, and height changes.

The technologies used to enable these four drone movements are the Wii balance board and the Wii remote controller. As you can see on the following listing of our modalities, we implement two different modalities *(A and B)* in two different multimodal ways *(AB and BA)*.

1. multimodal interface:
   - **Wii board:** tilt forward / tilt backward / tilt right / tilt left
   - **Wii mote:** move up / move down / spin left / spin right
2. multimodal interface:
   - **Wii mote:** move up / move down / spin left / spin right
   - **Wii board:** tilt forward / tilt backward / tilt right / tilt left

1.3 Initial Plan

Initially we planned to develop a multimodal interface that can control a drone with speech commands for some of the movements. Additionally, we wanted the user to wear an Oculus Rift, which enables to fly from the perspective of a pilot in order to have a more immersive feeling of flying. However, our plans needed to be changed due to several reasons:

Before we implemented our speech recognition application directly to our drone controller program, we tested it regarding its performance. The results of that test lead us to drop our plan of flying a drone with speech commands since a delay of $3s$ is inappropriate for a control modality. Besides, our speech recognition application was server-based, which was not reachable anymore when we started the connection to the drone WiFi.

Although we did not drop the idea to fly with an Oculus Rift, we tested our modalities without this nice technology because flying a drone with a Wii board and Wii mote is anyway not an easy task. Wearing an Oculus Rift while flying has the disadvantage that the pilot only has a partial view and therefore, does not see the obstacles behind and to the side of the drone. Additionally, it is very hard to estimate the current movement of the drone since slight drone movements, such as a small tilt, are not perceptible through the Oculus Rift, which yield false feedbacks to the pilot. Apart from this, the stream often freezes for one, two seconds. Anyway, the functionality is implemented and can be used by advanced pilots.

2 Architecture

The architecture of our application is as follows:
The user operates on the Wii board and Wii mote to trigger commands for the drone control. The movements then are sent to the Osculator that assigns these movements to events, which then are listened by a Node server. The Node server finally triggers the corresponding commands on the drone.

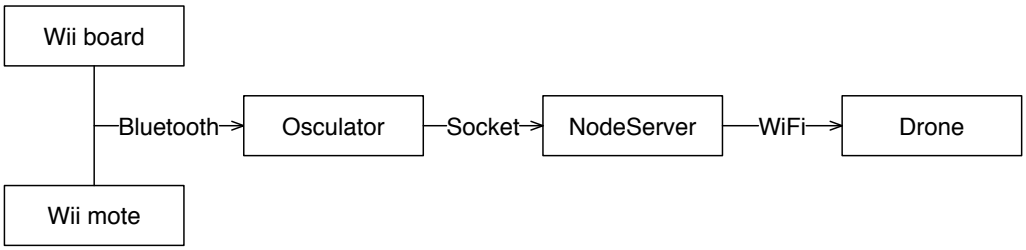


Figure 1: Application architecture

2.1 Hardware

2.1.1 AR.Drone2

The French company Parrot, founded in 1994 by Henri Seydoux, creates, develops and markets advanced consumer technology products for smartphones and tablets [3].

Parrot AR.Drone, first revealed at the International CES 2010 in Las Vegas, is a WiFi controlled flying quadcopter designed to be controlled by mobile or tablet operating systems. The hull, made from foam, encases the circumference of the blades for protection, and measures

57 cm across. The drone itself is constructed of nylon and carbon fiber parts, and contains a miniaturized inertial measurement unit for stabilisation purposes.

The successor drone, the AR.Drone 2.0, was revealed two years later in 2012. Improvements were maid on hardware and control functionalities. On the hardware side they supplied an increased camera quality (720p sensor with 93° lens, recording up to 30fps), more sensitive sensors (accelerometer, gyroscope, magnetometer, air pressure sensor), updated WiFi hardware standard (802.11n), and a more powerful battery. The upgraded functionalities, such as recording, tracking, and extended flying features, facilitate the control of the drone [4].

2.1.2 Wii Balance Board

The Wii Balance Board, first introduced on July 11, 2007 at the Electronic Entertainment Expo, is an wireless input device using standard Bluetooth technology to communicate with the Wii console. The board contains four pressure sensors, which are used to measure the user's center of balance and weight. Every pressure displacement, caused by shifting weight on the board, triggers corresponding information signals that can be sent to a receiver. In our case the signals are sent to the Osculator [5].

For our application we used the balance board to fly the drone in accordance with body movements / weight displacements on the board. The board has also been used to start and land by stepping on/off the board.

2.1.3 Wii Remote Controller

The Wii Remote, also known as Wiimote, is a wireless device to communicate with the Wii console, first revealed at the Tokyo Game Show on September 14, 2005. It implements the same Bluetooth HID protocol as the Wii board for host communications, and exposes most of its functionality via an extension controller.

The Wii remote can be expanded through the use of attachments, such as the Nunchuk. Due to the Joystick, the Nunchuck complements the Wii remote by providing functions similar to those in gamepad controllers [6].

In our application we exclusively used the Nunchuck Joystick, without the main Remote, to control drone movements.

2.2 Software

2.2.1 Osculator

Osculator is a software available for MacOSX that creates a link between different hardware devices and the Computer. It supports the OSC protocol that makes it able to be used with a wide variety of software and devices. When launching, Osculator listens for incoming messages, such as from a gamepad, on a defined input port. As soon it receives data, it can use that to perform useful actions by defining the type and value of events. In order to be able to use these events, the OSC messages can be sent to specific targets, such as to a host [7].

In our case we used the Osculator to receive messages from the Wii board and Wii mote over a Bluetooth connection. These messages we then sent with a UDP connection to the localhost, where our Node server was listening for commands.

2.2.2 Node.JS

Node.JS is an open-source, cross-platform environment built on Chrome's JavaScript runtime for easily building fast, scalable, server-side network applications. It uses an event-driven, non-

blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices [8].

For our implementation we used the Node server on one hand to listen to incoming events from the Osculator, and on the other hand to create a drone client that asks and reacts to these events. The communication between the Node server and the AR.Drone client was performed through a WiFi connection.

3 Modalities

In order to conceptualize possible relationships between input and output modalities, there are two design spaces to formalize human/machine multimodal interactions: the CASE and CARE model.

The CASE model includes four types of communications in multimodal systems representing the machine-side aspect, whereas the CARE model includes four usability properties covering the human-side aspect.

3.1 CASE

In our application we use the Wii board and Wii mote as input modalities. Each input modality has two functions: either moving the drone in x and z-axis plane, or move and spin it at y-axis plane. All these functions can be executed in parallel and independent from each other with no temporal constraint, whether between the modalities or between functions in the same modality, for instance, the drone can be moved simultaneously along the x and y-axis (between modalities), or it can spin and fly up at the same time (between functions of same modality). Thus, on the machine-side our application has the *Concurrent* property.

3.2 CARE

Concerning the human-side of fusion in our application we can state that it cannot be assigned with a single property. Rather it depends on the action that is needed to reach a given state.

If the given state is to reach a certain height in a certain horizontal distance, then no modality taken individually is sufficient to reach that given state. Both modalities are needed that can occur sequentially or in parallel. Due to this reason, the application owns the property of *Complementarity*.

However, if the given state is to fly the drone around a pole without any vertical movements, then a single modality is sufficient to reach that given state. Therefore, the application also has the property of *Assignment*.

4 Evaluation

To evaluate the two multimodal interfaces we decided to use a series of controlled experiments with several subjects. Therefore, we evaluate quantitative with controlled experiments.

4.1 Experiment

Hypothesis When controlling the drone with a joystick for directional movement (modality 2) a subject can do more rounds on a course than using the Wii balance board (modality 1) for directional movement.

Independent Variables In our experiment we change what the modalities control (either the directional movement or the altitude and orientation).

Dependent Variables The dependant variable in our experiment is the number of rounds a subject can fly on our given course.

Subject Selection The subjects, who participated in our experiment, were mainly family members and friends. We decided to conduct a within group experiment, in which the participants have been assigned to one of two groups. One group began with modality 1 and then modality 2, and the other group the other way around.

4.2 Measurement Setup

Initially, our idea was to let the subjects fly a complex course involving flying over, under and around obstacles. But due to various difficulties we decided to simplify the course.

The subjects were introduced to both modalities in advance of the test. They began with one modality given one minute of free flight. The same they did with the second modality, i.e., again one minute of free flight. After this introduction the subjects had the task to fly rounds around two pillars for two minutes, which were placed 5 meters from each other. First, with the first introduced modality then with the second introduced modality. We measured the number of rounds each subject managed to fly in the given two minutes.

4.3 Results

modality 1	modality 2
3	2
2	1
3	4
3	2
4	2
3	2
$\overline{x}_1 = 3$	$\overline{x}_2 = 2.2$

Table 1: Number of Rounds during 2 Minutes of Flight Time

In Table 1 the amount of rounds the subjects made during the two minutes are stated. This gives us an average of 3 rounds for modality 1 and an average of 2.2 rounds for modality 2. Note that the users in Table 1 were alternately introduced to modality 1 or 2 first.

T-Test To figure out whether the difference between $\overline{x}_1$ and $\overline{x}_2$ is significantly different from each other, we conduct a t-Test:

H_0: $\quad \overline{x}_1 = \overline{x}_2$
H_1: $\quad \overline{x}_1 > \overline{x}_2$

The t-Test provides $t = 1.7461$ with $p = 0.1114$, which implies that H_0 cannot be rejected at confidence interval of 95%. So, $\overline{x}_1$ and $\overline{x}_2$ can be considered as equal.

First Round vs. Second Round To see if the subjects improve when flying for the second round we compare the amount of rounds the users fly in the first measurement and the second measurement. For this we provide a t-Test between $modality_x group_1$ and $modality_x group_2$.

modality 1	modality 2
3	2
3	4
4	2

Table 2: Number of Rounds of Group 1

modality 1	modality 2
2	1
3	2
3	2

Table 3: Number of Rounds of Group 2

In Table 2 the amount of rounds the subjects of group 1 flew are shown. Group 1 was first introduced to modality 1 and was second introduced to the modality 2.

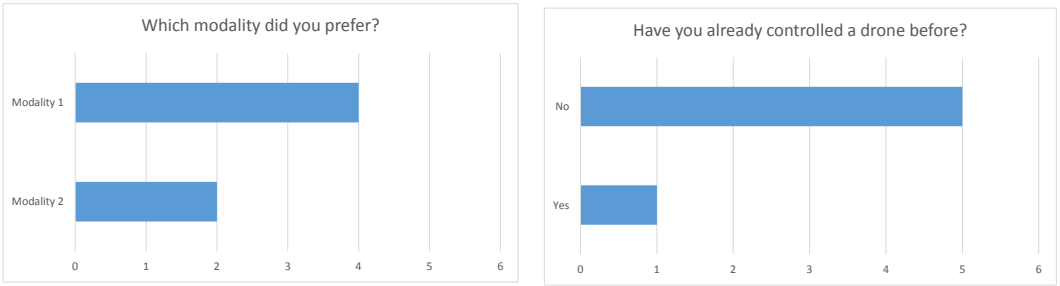
In Table 3 the results for group 2 are shown. Group 2 was introduced to modality 2 first and was second introduced to modality 1.

For *modality1* we get the result $t = 1.4142$ with $p = 0.2302$ that implies that the null hypothesis cannot be rejected.

For *modality2* we get the result $t = 1.3416$ with $p = 0.2508$ that also implies that the null hypothesis cannot be rejected.

We can conclude from the t-Tests that there is no improvement in flying when participants fly for the second round.

Survey Results In addition to our measurements we asked our subjects a few questions. We had 1 female and 5 male participants. As we see, most of our subjects preferred modality 1 over modality 2.



(a) Modality preference

(b) Previous experiences

Figure 2: Qualitative User Evaluations

5 Conclusions

5.1 Conclusions

In our experiments we have seen that there is no significant difference between modality 1 and modality 2, despite the fact that the number of flown rounds when using modality 1 has a higher average. Thus, the hypothesis that modality 2 is easier to fly with is wrong.

Also the presumption that participants improved their flying skills in the second round can be rebutted. There is no effect of learning when participants changed the modality.

The qualitative tests in form of subjective user ratings showed us that the majority of the participants have felt more comfortable with modality 1. This might be reasoned with the fact that all participants had difficulties operating on both input devices simultaneously. To counteract this difficulty, many participants figured out in modality 1 that it is much easier to control the drone when they leaned forward to move the drone forward, and used the joystick for direction changes. This style of flying facilitates the simultaneous use of both modalities. On the other hand, to facilitate the simultaneous use of two input devices on modality 2, many participants only used the joystick to control the drone that resulted in a flying style without any spin movements. Operations on the Wii board only were made when the drone lost on height. However, this flying style became difficult when the angle of the drone changed due to unconscious weight displacements on the Wii board.

Our user observations lead us to suspect that our participants mainly tried to control the drone with the Wii mote since joysticks require less body movements and represent a device that is conventionally used to control any movable objects. Anyway, finger movements are much more sensitive than movements by feet.

5.2 Difficulties & Best Practise

In our project we ran into a few difficulties due to working with a drone. First of all the drone was not at stock and therefore, we had to wait quite a while for it to arrive.

Working with a drone when using experimental code can easily result in a lot of damage. Therefore, we ensured that the control of the drone could easily be overtaken by another application to safely land it. We further secured the drone by attaching a leash to it. We either fixated the leash to objects (for indoor code tests) or held it in a hand during the measurements to prevent the drone from flying off in one direction or colliding into obstacles.

Due to the low flight time of the drone with a fully charged battery (15 to 20 minutes) we had to shrink down our test course and the time the subjects had to test the controls. Having a second battery helped to give us additional time to fly but due to long charge times (about one hour) we had to introduce quite a lot of long breaks, or continue with the work on an other day. This reduced flight time required us to reduce the course of the experiment.

Using the Oculus Rift in combination with the video feed from the drone's camera is not a recommended way to pilot the drone since the video feed only contains information about the front of the drone and not about its surroundings.

As a final note to this part we wanted to remark that the drone is very weather dependent as you can not fly outdoors when there is wind or rain.

References

[1] Matthew Turk and George Robertson. Perceptual user interfaces (introduction). *Commun. ACM*, 43(3):32–34, March 2000.

[2] Sharon Oviatt and Philip Cohen. Perceptual user interfaces: Multimodal interfaces that process what comes naturally. *Commun. ACM*, 43(3):45–53, March 2000.

[3] About parrot. `http://www.parrot.com/usa/aboutparrot/corporate-overview/`. [Online; accessed 21-May-2015].

[4] Ar.drone specs. `http://diydrones.com/profiles/blogs/parrot-ardrones-specs-arm9`. [Online; accessed 21-May-2015].

[5] Wii balance board. `http://wiibrew.org/wiki/Wii_Balance_Board`. [Online; accessed 21-May-2015].

[6] Wii remote controller. `http://wiibrew.org/wiki/Wiimote`. [Online; accessed 21-May-2015].

[7] Osculator documentation. `http://www.osculator.net/doc/`. [Online; accessed 21-May-2015].

[8] Node.js documentation. `https://nodejs.org/documentation/`. [Online; accessed 21-May-2015].