

Toward pen-based interaction with ATC tools

Stéphane Chatty and Patrick Lecoanet and Christophe Mertz

Modernizing workstations for air traffic controllers is a challenge: designers must increase efficiency without affecting safety in any way. Air traffic control is a time-intensive and safety-critical activity, and thus interaction efficiency and low error rates are crucial. Classical interaction techniques have been used in prototype workstations, but the resulting efficiency is not always satisfactory. This leads designers to consider more advanced interaction techniques. This paper reports on the design and a preliminary evaluation of the first prototype of project IMAGINE, which represents the second generation of graphical interfaces for air traffic control. This prototype, GRIGRI, uses a high resolution touch screen and provides mark based input through the screen. The use of gestures, as well as the use of multi-modal techniques, make interaction faster, and closer to the controllers' habits.

INTRODUCTION

During the 80s, a number of modern countries including France have decided to modernize their air traffic control working positions. Their goal was usually twofold: replacing obsolete equipment was a good opportunity for improving the productivity of controllers, through the use of state-of-the-art hardware and software. The hopes of many specialists were that artificial intelligence or other software techniques would help alleviate the controller's workload, through conflict resolution tools for instance. User interfaces were seen as a more technical issue, mainly related to the number and the complexity of the pieces of information to be displayed. Screen size and the use of colour to code information were the most obvious issues, and the design of electronic flight strips was the next challenge. With the advent of the Macintosh in 1983, graphical interaction looked very promising, and only needed to be adapted to air traffic control, with research on which pointing device was best suited.

In France, these hopes led to the PHIDIAS project, launched in 1986. PHIDIAS is aimed at replacing the current en-route air traffic control workstations within the next few years, and it is now in its final development stage. It makes heavy use of classical interaction techniques, and most of its original features are to be found in its hardware or low-level software components. PHIDIAS provides a standard WIMP (windows, icon, mouse, pointer) interface on two screens, one of which is a square 20" wide Sony screen. A lot of work has been spent on identifying and solving the software issues raised by such a large screen, especially the management of many graphical objects moving independently. This work has led to a workstation which offers good performances, and whose hardware and software architecture are undisputed. A consensus has been reached about the look of the radar image: a colour coding scheme has been adopted, and even the palette of colours is now being approved, after a study with ergonomists and vision specialists.

If those technical issues have now been solved, a number of design choices made in the interface of PHIDIAS can still be improved, if only because hardware, interaction techniques, and our knowledge of ATC have improved over time. This article describes the design of GRIGRI¹, the first prototype in project IMAGINE, which prepares the future generation of interfaces for air traffic control. GRIGRI implements design choices very remote from current designs, and finds its inspiration in pen computing. Contrasting with the notepads that made pen computing popular a few years ago, GRIGRI does not use handwriting recognition, but only the recognition of simple gestures, drawn with a pen or a fingertip. Those gestures (marks) are performed directly on the surface of the screen, thanks to a high resolution touch-screen. GRIGRI also uses direct manipulation for some functions and makes use of sampled sounds to improve the confidence of users and to make some interaction modes more explicit. Finally, the layout of GRIGRI was designed to encourage two-handed interaction.

This article is organized as follows. The next section identifies several drawbacks in the use of classical user interface techniques. The third section

¹ *grigri* is a colloquial French word for *scribble*

D	L	T	2	2	5	<3105>	<150>	150	150	CDN	LACAN	AMB	BALON	IMG	IE
E120	250	LFPB	(LFBL)	150	150	150	150	150	55	W	59	03	15	27	PT
1855				1700	1700	1700	1700	1700	16		16	17	17	17	19

Figure 1. An annotated flight strip

describes related work on gesture recognition and pen based computing. The following sections describe the design choices implemented in GRIGRI: first with an overview of a first prototype and the lessons learned from it, then with a detailed description of the current prototype. The last sections report on evaluations of the system, and on possible evolutions of GRIGRI and related technologies in the field of ATC.

LIMITS OF TRADITIONAL INTERACTION

When the plan for modernizing the French ATC workstations was launched in 1986, one of the goals was to take advantage of new technologies in order to alleviate the workload of air traffic controllers. This goal had immediate consequences, such as the use of color screens to code information useful to executive controllers. It also had indirect but major consequences. A number of ATC experts placed hopes on intelligent tools that would help radar controllers solve problems. But such tools would need data that are not available to the ATC system in its current state. For instance, the system does not know which controller is in charge of a given aircraft, because transfers are performed through procedures based on voice. Consequently, the idea of helping controllers quickly turned into the need for making them provide more data to the system. This is why designers were interested in turning paper strips into a graphical interface: the information previously contained in annotations would then be known to the system.

These goals, as well as the set of available techniques, have had a strong influence on the design that was chosen for the new workstations. The screen gained even more importance and its size became critical. In addition to the radar image which covers a 50 cm wide round screen in the current system, the screen also has to display the new “electronic strips”, as well as the potential intelligent tools. Air traffic controllers, afraid of missing important information that would be hidden, are generally opposed to overlapping windows. It was thus decided to use screens as large as possible, namely 50 cm wide square screens, featuring 2048x2048 pixels, developed by Sony for the US FAA. But those screens are so large that users must be seated far from them if they want to see the whole contents. This excluded any possibility of direct interaction on the screen, and led designers to using a mouse and classical interaction techniques: menus, buttons, double clicks, etc. Studies were then focused on information coding (colors, icons, etc) and on the software techniques needed to efficiently display and move many objects on a large screen.

On this screen, the main interactive areas are the fields of flight strips and aircraft labels on the radar image. Users may manipulate them to store a

AFR1499	EDDK LFPG PG	-	HDG	+	30	ADORA	REM	BSN	BSN	
AT42	23 220		FIN			08:00	11	21	21	
			▲▲							
AFR1445	EDDF LFPG PG		▲		80	MMD	REM	BSN	BSN	
EA32	27 240		280			08:00	06	12	12	
			270							
			260							
LGL1203	ELLX LFPG PG		250		60	REM	BSN		BSN	
FK50	22 180		240			08:02	12		12	
			230							
			220							
AFR2401	EHAM LFPG PG		210		80	NEBUL	BSN		BSN	
B737	27 230		▼			>>>	11		11	
			▼▼							

Figure 2. Manipulating a flight strip through menus.

new heading or altitude or select an aircraft and display its route. All these manipulations require the designation of an aircraft or a strip, and the input of a command and its other parameters. Several commands are possible for nearly every active field (label, field of a label, part of a strip). Among the classical interaction techniques, menus seemed to be the most appropriate. But in order to minimize the number of items without having to use cascading menus, it was decided to associate a different menu with every field. For instance, figure 2 shows the menu associated to the “heading” field of a strip.

The system that we just described is still being developed. However, several design choices have been criticized by future users. For instance, the manipulations on electronic strips are considered slow and painful. It has been observed that with today’s system, a controller writes an annotation on a strip every 17 seconds [7]. With such usage frequencies, manipulation times and the degree of attention needed are very important. But manipulating menus is relatively slow, and users have to pay attention to what they are doing. Even worse, before selecting an item in a menu, a controller has to click in a zone which is sometimes very small. This increases manipulation times according to Fitt’s law [4]. In the case of flight strips, the decrease in performance is noticeable, compared to manual manipulations that are often performed in parallel with other tasks, without paying much attention. With no simple solution to such problems, a number of users ask for the status quo, and want to keep paper strips until the time when alternative tools will make them really obsolete.

Even if electronic strips are discarded, manipulation times will still be an issue for radar images, and for every tool provided by the ATC system. This leads us to exploring possible solutions to that issue, by exploring new interaction techniques. Gesture recognition and mark-based input through a digitizing tablet or a touch screen look like a promising direction. Other techniques are also explored, among which are speech recognition and computer

vision. But gesture recognition seemed mature enough to be proposed to air traffic controllers for a medium-term implementation.

RELATED WORK

Gesture recognition deals with the movements performed with one's hand, or any part of the body, with or without an instrument (pen, glove, etc). The input parameters are the successive positions of the tip of the pen, those of fingertips, or the successive angles of articulations, for instance. The devices used are pointing devices (mouse, digitizing tablet, touch-screen), digital gloves and position locators, or video cameras. Some of the techniques developed for 2D gestures performed with a pen can be transposed to more complex situations like 3D gestures performed with a glove [1]. Among those techniques, the most popular is Rubine's algorithm [13], which incrementally computes geometrical features of gestures, and uses statistical methods.

2D gestures, or marks, have been used in several ways in graphical interfaces. In addition to obvious applications to text input, gestures can be used to issue commands to the system. Kurtenbach has studied the issues raised by mark-based input, on the user's as well as on the system's side [10, 9]. Pie menus [3], although apparently close to traditional menus, are close to mark-based input, in that they use the orientation of gestures performed by the user. T-Cube even chains pie menus, thus associating each command to a broken line [14]. In T-Cube, menus are only displayed if the user hesitates while issuing a command. Unistrokes [8] fit between handwriting recognition and mark-based input. It models each letter with a simplified letter composed of only one stroke, in the same way as shorthand.

Pen computers, or notepads, which use the techniques mentioned above, have been popular at the beginning of the decade [11]. One of the first widely available environments was GO Corp's PenPoint. Apple's Newton uses handwriting recognition and mark-based input, combined with high quality feedback. But handwriting recognition accuracy often determines how such systems are accepted by users [6], and it is still difficult to achieve a very high accuracy, especially in countries where cursive writing is dominant, like France. Graffiti, a commercial variation of Unistrokes by Palm Computing, improves the accuracy and thus the usability of such systems at the cost of learning a simplified alphabet.

Finally, it is interesting to note that French air traffic controllers have been familiar with touch-screens for decades. The Digitatron, introduced in the 60s, is a touch sensitive alphanumeric screen with a low resolution. It gives access to a videotext system, used for infrequent operations such as checking which military zones are activated or modifying a flight plan.

A FIRST PROTOTYPE

As mentioned earlier, the management of flight strips is a good candidate for mark-based input. The annotations written on paper strips are made of numbers, and of simple and codified marks. Moreover, annotations are frequent, and the expected gain in time provides a good motivation. This

led us to build a demonstrator in order to check whether gesture recognition was useful in this context. The proposed interface was based on the proposed format for electronic flight strips in PHIDIAS, which is very close to real flight strips. The recognized marks, made with a pen on a digitizing tablet, were of two types. First, it was possible to sort, group and shift strips, just as it is done with paper strips. Then, the usual annotations performed on paper strips were mimicked: highlighting of values by underlining them, modification of values by striking them off and writing the new ones, input of direct routes by drawing arrows between beacon names.

This prototype of flight strip management was soon enriched with a simplified radar image manipulated with gestures, so as to check how well the technique could be applied to interfaces where the use of gestures was not as obvious as for flight strips. This radar image was built with no button or menu, and every command was performed through gestures. There were gestures for zooming and panning, and for several operations on aircraft and beacons.

The results of this pilot study were very encouraging, as well as full of lessons. The lessons were related to the handling of errors, the acceptance of the system by users, and the type of applications that could easily take advantage of gesture recognition.

Errors

Error rates were a major concern, because high error rates would have meant that the technology could not be applied to air traffic control in its current state: errors make interaction slow and painful when they are noticed, and are dangerous when unnoticed. Although serious evaluations were not carried out at that stage of the project, the results were comforting. Our worst concern was interpretation errors: one gesture recognized as another gesture. But such errors were rare. More frequent were interpretation failures: gestures that could not be related to any gesture class. Those failures were especially numerous during the first contacts with the system. The absence of adequate feedback could induce users into errors and decrease their confidence in the system.

Acceptance

During this pilot study, the way the prototype was accepted by users was important: previous experience showed that full studies are useless if air traffic controllers are not confident about the system's potential usefulness. Comfortingly, the acceptance of the prototype by the first air traffic controllers who tried it was good. They found most gestures natural and fast enough, even when they had no equivalent in the current ATC system. The gestures proposed to interact with the radar image were especially well accepted, mainly because of the low precision that is required to perform operations. However, the physical setup, composed of a digitizing tablet and a normal workstation screen, was frustrating to users. They would have preferred a more direct interaction with the contents of the screen.

Potential applications

The use of gestures to interact with the radar image was appreciated and considered as potentially useful by the users, even though it was not the primary goal of prototype, and was presented to them as such. Gestures for interacting with the contents of flight strips were appreciated too. However, the interface proposed for manipulating the strips themselves was the cause of problems. Users, who took pleasure in being able to use their pen again to annotate strips, were disappointed to be unable to move them with their bare fingers. Moving strips with a pen was far less natural. Moreover, the gestures for moving strips often caused confusion with gestures for annotating them. For instance, downward moves were interpreted as downward arrows when made close to a number representing a flight level.

The lessons learned with this first prototype were used when developing the prototype described in the rest of this article. First, we investigated the set of available hardware allowing direct interaction on the screen, either by projecting images on a digitizing tablet (like Rank Xerox EuroPARC's Digital Desk [15]), or by using touch-screens. Second, because of the problems with the manipulation of strips, it was decided to split separate concerns, and to concentrate on radar images, leaving flight strip management for future research. Finally, more attention was paid to feedback during interaction.

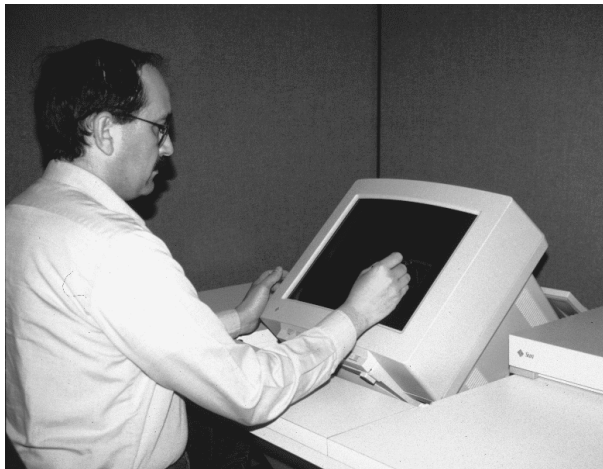


Figure 3. The physical setup of GRIGRI

THE PROTOTYPE GRIGRI

Using the first results described above, a second version of the system was built, devoted to more systematic evaluations with air traffic controllers. This version, described below, offers a more complete interface and set of functions than the first prototype, so as to allow more realistic tests.

The physical setup of our prototype uses a touch-screen. It includes a color

screen built in the surface of a desk with an angle of approximately 30 degrees (see figure 3). The screen is covered with a high resolution touch sensitive layer, that can be used with a pen or a fingertip. It is used to display a radar image and a bar of controls, as shown on figure 4. Depending on their type, commands are issued through the bar of controls, through gestures on the radar image, or through longer direct manipulation interactions.

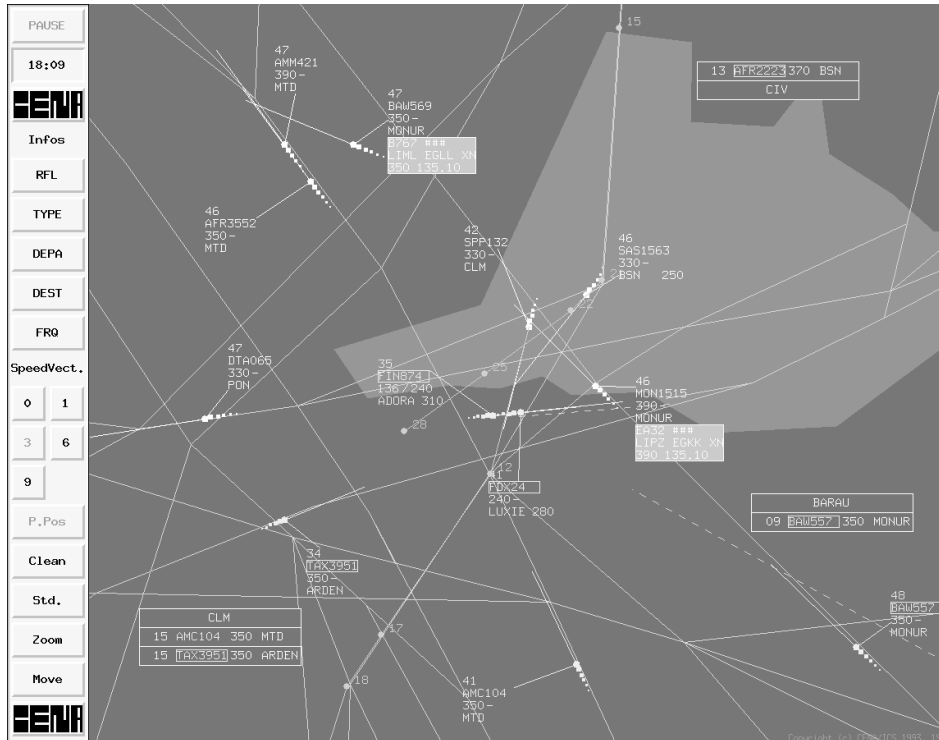


Figure 4. The screen of GRIGRI

Contents of the screen

For the reasons explained earlier, GRIGRI only features a radar image, and no flight strips. Our purpose is that evaluations yield results about mark-based input only, without introducing perturbations due to design choices in strip management. For similar reasons, the proposed radar image is not much different from radar images offered by PHIDIAS and its earlier prototypes. Controllers are familiar with their design, and we did not want to add novelties that would have changed their perception of the system. The radar image is thus composed of a plain background on which appear airways and restricted zones, represented by lines and polygons of a different color. Aircraft nearly have the same representation as on today's radar screen: icons for the current and past positions, a segment for speed. Labels are similar to

those of PHIDIAS: they contain several text fields displaying the call sign of the aircraft, its speed, its flight level, etc. The label can be further enriched on demand, by displaying data from the flight plan of the aircraft: destination airport, route, etc. Users can also obtain graphical representations of routes on the display. Finally, the bar of controls on the side of the screen provides a number of global commands.

Global commands

Most commands offered by air traffic control interfaces are related to a given aircraft. Only a few commands are global to the radar image, and make it possible to change parameters of the display: filters to display only certain categories of flights, zoom factor, panning, length of speed vectors (expressed in minutes of flight), etc. Though we plan to return to real hardware buttons and knobs for a number of those parameters, the corresponding controls were offered in the form of software buttons, grouped in a bar on the side of the screen. The size of those buttons is large enough to allow fast manipulations with fingertips, without having to use a pen. This provides maximum flexibility to users, so that future observations may determine whether they prefer manipulating them with the pen, with fingertips, or even with the non-dominant hand. For that purpose, the buttons were laid on the side of the screen in order to study whether users would be tempted to use their non-dominant hand for pressing them. With this design, we hoped to observe a specialization of hands, the non-dominant one dealing with secondary tasks, and the dominant one being dedicated to fine manipulations on aircraft representations.

Mark-based commands

Most commands concerning a given aircraft are simple operations, for which one needs to specify the aircraft, the operation to be performed, and one or two additional parameters (field to be highlighted or modified, type of warning, etc.). When those parameters can have many values, special interactions will be necessary. But in other cases, the number of possibilities is low enough to code each combination of operation and parameters by a mark. For instance, controllers highlight the flight level or the heading of an aircraft to register the fact that the valued was confirmed to the pilot. This highlighting operation can be designed in two ways. It is possible to go for a unique, generic command, by assigning a specific behaviour to every field representing a value. It is the choice made in PHIDIAS: users click on the field they want to highlight, thus making a menu appear, and choose the appropriate item in the menu.

With mark-based input, another choice can be made: do not specialize fields, but have several marks for the same operation, and have each mark code for a field. For instance, instead of having a mark for the command “modify a field”, that can be applied to any field, let us have a mark for “modify speed”, a mark for “modify flight level”, and so on, with the meaning of a mark being independent from the location where it is issued. This design choice yields a growth of the number of commands that must be learnt (about 15 different

marks). But it dramatically lowers the constraints imposed on users, because gestures can be started anywhere on the label, and their sizes may vary much more. This choice allows us to expect a better efficiency in interaction, at the expense of some training, which is acceptable for professional users such as air traffic controllers.

However, even if we were confident that specific gestures were a good solution, we also implemented generic commands as an alternative way of performing operations. Those generic commands (modify, highlight) are interpreted according to the field on which they are performed. Doing this, users were allowed to choose the interaction style they preferred, thus getting some hints about whether our reasoning was right. The set of all possible gestures is shown in figure 5.
















	assume control		set warning		select
	change heading		change speed		enter route
	change T flight level		change C flight level		change
	alarm		highlight		erase
	zoom		pan		undo

Figure 5. The set of gestures recognized.

To recognize gestures, GRIGRI uses Rubine's classifying algorithm. This algorithm can classify gestures composed of a single stroke, and involves an individual training period. For each class of gestures, each user provides about fifteen examples, using a tool described later in this article. During this training, the geometrical features of each gesture are extracted, and are used to produce a dozen features that are significant of the class. Later, when using the system, the same features are extracted from every gesture drawn, and compared to those of the defined classes. The classifying algorithm identifies the class whose features are closest, or yields a recognition failure if the confidence ratio is too low. Classification is very fast, and response times are not perceptible by users. Having noticed that recognition failures had to be quickly and clearly signaled to users, we had to devise appropriate feedback. Considering that the visual channel was already very busy, sound capacities were added to GRIGRI for that purpose. Two sampled sounds are used: one signals recognition failures, and the other signals gestures that are correct but meaningless at the place where they are performed. However, sound was not used for successful operations yielding an immediate visual feedback, so as to restrict the use of sound to situations where it is necessary.

Long interactions

Some commands of GRIGRI involve the input of numerical or geometrical data, and cannot be implemented as a simple gesture or a button click. In the current version of GRIGRI, there are several of those “long interactions”, used for zooming, or to input a heading, a flight level, a speed, a warning, or an alarm. A warning is a visual sign used by a controller as a reminder. An alarm is a problem that has to be notified to other controllers when they assume control of the flight. Heading, warning and alarm input are triggered by a gesture on the label of an aircraft. Then begins an input sequence using classical techniques of direct manipulation: rubber band for headings, and palette for alarms. Zoom and input of flight levels and speeds are more complex.

To enter a flight level, one needs to specify an aircraft, the operation itself, and the new value. These data are entered as follows. Like other operations, the command begins with a gesture on the label of the appropriate aircraft. Then GRIGRI opens a window that allows the user to enter the new flight level. In many regions of airspace, only a few values are acceptable (330, 350, 370, 390, for instance), and systems such as PHIDIAS propose to enter the new value through a menu, allowing themselves to propose a default value. However, a number of air traffic controllers have to deal with many more possible flight levels, which makes this solution less acceptable. What was implemented in GRIGRI is the recognition of hand written numbers, in order to compare the efficiency of the two techniques. The input window has its own gesture classifier, independent from the one which is used in the radar image. It recognizes the ten digits (drawn with a single stroke) as well as gestures for correction and validation. The input window is fairly large, so as to let users write at their preferred size. In order to avoid masking a large portion of the radar image for too long, this window is semi-transparent, after Xerox's see-through tools [2] (see figure 6).

Changing the zoom factor involves a more classical type of interaction. The user first needs to go into zoom mode; this is done with a gesture, or by pressing a button in the bar of controls on the side of the screen. Then, the image is stretched or condensed by moving the pen outward or inward; the system goes back to normal mode when the user lifts the pen again. The two ways of starting the operation have been implemented because we hoped to identify whether users preferred to use a button or a gesture. We also hoped to induce users into using forms of two-handed interaction: the bar of controls is located on the side of the non-dominant hand, while the pen is usually held in the dominant hand. This is a situation where using two hands may be useful: the non-dominant hand rests on the border of the screen and presses buttons on demand, while the other hand stays where the attention of the user is focused, and keeps holding the pen. Even though it is impossible to perform parallel actions with current touch-screens (which might be a problem, as described in [5]), we wanted to assess whether a purely sequential operation was possible.

Finally, conscious that long interactions place the system in modes that

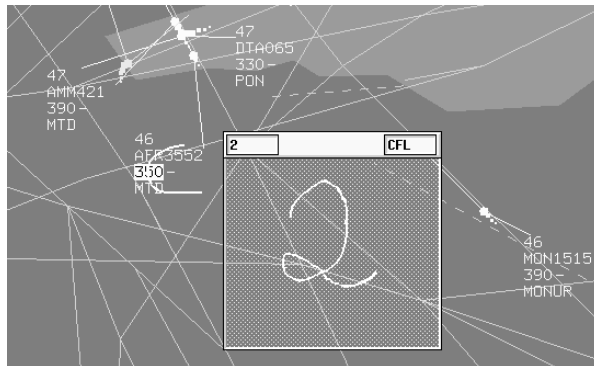


Figure 6. Entering a flight level

should be made explicit to the user, it was decided to use sounds. When the user starts a long interaction, or when the action is terminated, the system produces a fairly long sound of the kind which is usually associated to transitions, teleportations in science fiction movies, for instance. Though it has not yet been planned to test the usefulness of such sounds, we believe they have a significant role in the perception users have of the results of their actions. However, air traffic controllers are still very cautious about sound, and this will need to be studied more closely.

Training

The algorithm used in GRIGRI for gesture recognition has a learning phase. Training has to be done for each user, even when the gesture set is fixed, which is the case for GRIGRI. This is partly because the dynamics of a gesture may vary, even if the result looks the same, and partly because complex gestures, such as digits, vary a lot. Although already available gesture editors could have been used, it was decided to develop a new one, specific to our system, for several reasons. First, we wanted to make sure that the size and shape of gestures would not change between the learning phase and the evaluation phase, and we thought that the context, and especially the size of aircraft and labels, had a role. We also knew that the learning phase would be the first exposure of our users to GRIGRI, and we wanted to use it as a training period for them as well. Finally, it had been noticed during the earlier experiments that gestures would vary when drawn near the edges of screen. Taking this into account, the training system of GRIGRI makes users draw gestures in different locations of the screen, thus feeding the algorithm with all variations of gestures.

IMPLEMENTATION

GRIGRI was implemented on a Sun SPARCstation, with a software and hardware environment adapted to our needs. Only the screen of the workstation

is accessible to users: the mouse was removed, and the keyboard is used by observers during tests. The screen is the standard screen of Sun workstations, attached to a custom made table. The input device is a resistive layer held between the screen and its frame, with a resolution of about 200 dots per inch.

Our system uses the X Input extension to the X Window System distributed by MIT, which had to be modified to suit our needs. With these extensions installed, the touch-screen replaces the mouse for every purpose. Gesture recognition is based on Rubine's algorithm, in its C version. The software layers that feed positions to it had to be modified to account for the slight unevenness of the screen that caused the pen to bounce. Rebounds were interpreted as the end of a mark and the start of a new one, which led to many interpretation failures. We introduced a time-based filter that removes most of those parasitic events.

GRIGRI was built using a hybrid prototyping environment developed at CENA. This environment is based on a public domain Scheme interpreter: time-intensive functions are programmed in C and exported as primitives of our environment, while most of the system is programmed in Scheme. This approach, similar to that of Tcl [12], combines the efficiency of C with the flexibility of an interpreted language, and proved very useful for prototyping. In GRIGRI, the most important primitives used are the gesture recognition algorithm and a graphical widget dedicated to animated interactive images.

EVALUATION METHOD

One of the main motivations in developing GRIGRI was to perform systematic evaluations with professional air traffic controllers in a semi-realistic environment, so as to assess the possibilities of using pen computing in future workstations. The first phase of those evaluations was carried out in the first half of 1995, and focused on two main topics. The first topic was how well the proposed interaction technique is accepted by controllers. The second topic was how the system was used, and how well the gesture classification algorithm performed. Once those results have been analysed and the system has been updated to take them into account, other evaluations will be carried out, as explained later in this article.

Preliminary tests

Before carrying out the evaluations with air traffic controllers, a first step was a series of preliminary tests, which relied on CENA staff as users. Those persons had no background as professional controllers, but had a good knowledge of air traffic control. While not using the time of air traffic controllers, which is an expensive resource, those tests allowed to identify and eliminate several design mistakes. They also helped to design a training scheme for new users. During the first tests, users were given a short demonstration, then were rapidly asked to provide the system with sample gestures, before further manipulating the system. Those sample gestures were usually drawn very carefully, as one would do to teach a child. But the familiarization with

the system was very fast, and after 10 to 20 minutes of training, users became enthusiastic and thought they could write normally as with real pen and paper. This had two drawbacks. First, several users increased their manipulation speed too much, thus producing many interpretation failures and getting disappointed. This problem was solved by repeatedly reminding the users of the purpose and the limits of the system during their first contact with it. The other problem was that the samples provided during the first minutes of manipulation were very different from the gestures drawn after a longer period of time. This is why the learning phase was split in two phases for the real experiment: after the first period of training, users were asked to provide the system with new sample gestures.

The experiment

After the first series of tests, more systematic evaluations were carried out with volunteer air traffic controllers from different en-route control centers in France. The experiment was made in a semi-realistic environment, and yielded subjective as well as objective data.

The experiment took place in an experimentation room at CENA. The system described earlier in this paper was fed with simulated traffic, displayed on its radar image. No paper nor electronic strips were proposed. The evaluation involved 12 air traffic controllers, each of whom spent one full day with an ergonomist, who first assisted then observed them using the system. The morning was dedicated to training the user and the classification algorithm. One hour was spent to explain the project and the techniques involved, and to give a demonstration. Then one hour and a half was spent to train the classification algorithm through the training application. During this training phase, the controller was asked to draw about 20 examples for each of the 29 gesture classes of the system. This training was also useful for the users to learn the system, even though it sometimes appeared later as insufficient. The morning session ended with one hour of exercises, so that the controller was familiarized with the display, the device, and the gestures used in the system. The exercises were also useful to detect and fix problems resulting from a bad training: ambiguous gestures, lack of examples, etc. During the afternoon, one hour was spent on the evaluation itself, then the controller answered a questionnaire, and finally had the opportunity to analyse his or her performance by commenting the logs of the experiment.

One of the main goals of the experiment was to gather objective data on the usage of the system and its performance, so as to assess its usability and to identify possible improvements. In order to reach this goal, GRIGRI was designed to store as much data as possible, at different levels of abstraction. At the lowest level, every action of the user was timestamped and stored in a file. This makes it possible to replay the gestures drawn, even though this capability has not been used yet. It also allowed us to build a library of all drawn gestures, so that we were later able to test improvements of the classifying algorithm. At a higher level, the system stored for each gesture the result of its classification. We considered four types of results:

- good recognition: the gesture was associated to a meaningful gesture class
- no recognition: the system was not able to classify the gesture
- out of context: the gesture was associated to a class that corresponds to an invalid command. An example of out of context gesture is a 'C' drawn on the background of the screen.
- false recognition: the gesture was associated to the wrong class

The system could only detect the first three types of situations. This result was then refined at two stages. First, the observer was able to provide information about gestures in real time: keys of the keyboard were used to signal out of context gestures that were not detected by the system (a line drawn on the wrong label, for instance), and false recognitions (a '0' understood as a '6', for instance). Then the logs of the experiment were carefully studied to refine this classification, as will be described in the next section.

The data obtained by logging is a wealth of information, and can be used for various sorts of analyses and comparisons. However, we also felt the need to get subjective feedback from the users. This was obtained through a questionnaire composed of 15 questions, and through free-form comments from the users. Some of the questions were general ("what you think of pen-based computing applied to ATC?", "Do you perceive this interaction technique as natural?", "How do you perceive the recognition rate?"), while others focused on the physical setup, the usability of each command, and the choice of gestures.

EVALUATION RESULTS

The experimental method we just described was used with 12 volunteer air traffic controllers from French en-route control centres, mostly motivated by curiosity and good will. This provided us with 12 filled questionnaires, and 5800 gestures drawn during the 12 hours of the experiment. We used this data to produce results and analyses about the performance of the algorithm, the speed of interaction, the limitations of the hardware setup, the choice of gestures, the acceptability by controllers, and their ability to learn the system.

General appreciation

From the general questions and the free-form comments emerged the general opinion that the technique proposed is promising, but still needs work. Out of the 12 users, 3 considered it as "usable as such", 8 as "interesting, but to be improved", and 1 had no opinion. On the ease of use of the system, the answers were similar: 1 user considered it as "natural", 8 as "natural but with too many constraints", and 3 as "not natural at all".

In addition to those answers, most of the 40 free-form comments obtained were general comments on the usefulness of the technique. The most prominent theme was about the speed and ease of use of the system: approximately

half of those comments were positive, and the other half was negative: “Dialogue is direct and fast”, “The system is fast (gain on speed) and efficient”, “This system imposes less constraints than a mouse”, “It is slower than a mouse”, “Data input is sometimes slow and painful”, “The system requires too much precision”, etc. Other comments aim at analysing the system and its potential strengths and drawbacks: “actions are more direct and manual, thus physical, this improves memorisation”, “this restores a controller/strip relation”, “this way of working is very similar to a strip board”, “even with its imperfections, this system allows me to think about something else while writing”, “will this work well with real traffic and under stress?”, “you’ll have to check whether this still works with heavy traffic when flight labels are packed”.

All those comments clearly show that this technique is not yet mature enough, but they also highlight its potential usefulness, provided that its main drawbacks are addressed. The next sections are focused on some of those issues.

Recognition rates

As explained earlier in this article, we evaluated the performance of the classifying algorithm by carefully studying each of the gestures performed by the users and comparing them to the results produced by the system. From the four types of results produced by the system and the annotations added in real time, the analysis produced four new categories that are more appropriate for evaluating the real performance of the system. This was done by dispatching “out of context” gestures in other categories, considering for instance that a ‘C’ purposely drawn on the background on the screen and recognized as such was a good recognition, even if this is an invalid command. Out of context gestures will be further discussed in the “hardware limitation” section below. In addition to suppressing this category, a distinction was introduced among the non recognitions, between those that could be recognized by a human looking at them and those which obviously were bad manipulations: gestures composed of one point, for instance. Figure 7 shows the outcome of that classification. For each category, the first figure is the percentage over all 5800 gestures, and the figure in brackets is the percentage over the 5200 well-formed gestures (obtained by eliminating the 10.3% that were badly formed).

	rate	explanation
good recognition	80.6% (89.9%)	system in agreement with humans
no recognition	7.9% (8.8%)	system unable to classify a well formed gesture
false recognition	1.2% (1.3%)	system disagrees with humans
bad gestures	10.3%	badly formed gestures: false touches, etc.

Figure 7. Recognition rates during the experiment

The data analysis that yielded those results helped to identify a number of possible causes for non recognitions. Among those causes were artefacts at the beginning and the end of gestures: segments that were not conform to the overall shape of the gesture. Those artefacts are caused by parasitic

movements of the hand when the pen reaches or leaves the sensitive surface, and are not well handled by the classifying algorithm. The results were greatly improved by detecting and removing those artefacts. Other artefacts occurred in the middle of gestures because of the slight roughness of the screen. The introduction of a time-based filter further improved the results. The revised algorithm was then applied to the gestures that had been stored during the experiment, and Figure 8 shows the results obtained:

	rate	explanation
good recognition	84.7% (94.4%)	system in agreement with humans
no recognition	3.9% (4.4%)	system unable to classify a well formed gesture
false recognition	1.1% (1.2%)	system disagrees with humans
bad gestures	10.3%	badly formed gestures: false touches, etc.

Figure 8. Recognition rates with the revised algorithm

Hardware limitations

Most of the problems encountered during the experiment were caused by the hardware used in GRIGRI. Some of them had consequences on the accuracy of recognition: parallax problems, and badly formed gestures. Others were observed in the questionnaires. Finally, other limitations were not observed with the prototype, but will make the design of a full size system more difficult.

Many of the problems observed were caused by an insufficient integration of the display and the sensitive layer. When working with a digitizing tablet dissociated from the screen, it had been noticed that the coordination between the hand and the eyes was not satisfactory, and that the comfort of manipulation had to be improved. This is why a touch screen was used in the later version of the system. However, this touch screen uses a classical TV monitor, i.e. a screen with a thick glass layer. The sensitive layer adds a few millimeters to the thickness of materials separating the tip of the pen from the image itself. This distance induces users into two kinds of errors. First, they do not correctly evaluate the distance between the pen and the surface of the screen, and contact occurs earlier than thought, yielding badly formed gestures. The technology used did not provide proximity detection, that would have allowed for a visual proximity feedback helping to prevent such errors. Then, because of the angle at which the screen is observed, a parallax error occurred: when the eye, the tip of the pen, and a point on the screen are aligned, the pen actually points at a zone located a few millimeters below that point. This led to a number of wrong designations and out of context gestures. This parallax error was partly corrected by a geometrical transformation in the software, but it still made users uncomfortable. An informal experiment carried out on the same system using a digitizing tablet showed that badly positioned gestures were reduced to 0.5%, thus confirming the relation between such errors and the parallax problem.

Another problem caused by the technology used is the sensitiveness of the touch layer to any kind of contact such as the tip of the pen, a fingertip, but

also a wrist. Furthermore, this device does not appropriately detect multiple touches: only the mean position is emitted. This means that it is impossible to rest one's hand on the screen when writing. This led to a majority of users (9) feeling strain in their arms after some time of manipulation, and to a number of badly formed gestures because of multiple contacts. Furthermore, because of the size of its tube, the parallax errors and lighting problems, the use of a TV monitor made it impossible to tilt the screen as much as desired, thus increasing the arm strain. However, users would consider a horizontal working surface as appropriate or very appropriate (8 answers, against 3 answers for little appropriate or inappropriate). In the light of these findings, it appears that the ideal technology would probably be a flat sensitive screen, or an image projected onto or from under a digitizing surface, as demonstrated in Xerox's Digital Desk or in other commercial products. However, such devices either do not exist yet or do not offer a sufficient image quality. Furthermore, even if such devices exist in the near future, their surface will be limited by the span of users' arms. Using a 20" wide touch screen would probably be uncomfortable. This will lead to rethink the organization of the control workstation if pen-based computing is to be used.

Gesture set

The gestures used in GRIGRI were chosen with simple mnemonic rules: in most cases, the gesture for a command was close to the first letter of the name of that command (in English). This was inappropriate for several reasons. The first of those reasons is memorization: 7 users had problems for memorizing the gestures, and informal comments confirmed that. However, this would have to be confirmed after a longer learning phase. Another problem is that a few gestures exhibited significantly lower mean recognition rates. For instance, the 'h' used to input headings caused serious problems to 4 users, yielding a total 18% error rate. Other gestures (such as '4' and '7' or '0' and '6') were sometimes confused by the system. Finally, a few gestures were incompatible with the policy used to determine the target of a gesture. As most graphical interfaces, GRIGRI determines which graphical object is concerned by a command by determining the first object located under the point at which the command occurred. In GRIGRI, this command "hot spot" was obtained by taking the centre of the gesture. However, in a few cases such as the gesture for changing a TFL, the centre of the gesture was often out the surface of the target object. In the case of the TFL command, it was because the gesture had to be drawn from right to left, and because users begin gestures at the left of objects. This was corrected by changing the policy for choosing the hot spot (by using the first point instead of the centre), but this remains a potential problem. It appears that, in the same way as a set of icons has to be designed in order to be consistent and easy to understand, a set of gestures has to be designed so as to be easily recognized, easily distinguished, and to use the same hot spot selection policy. This will have to be taken into account when designing a command set for the next versions of GRIGRI.

PERSPECTIVES

This first experiment with gesture recognition applied to air traffic control showed that such technology definitely has to be considered for future control workstations. It also evidenced a number of issues that had to be studied before its application in the real world. Among the perspectives opened by this study are future evolutions and evaluations of GRIGRI, and also variations of that technology that could be applied at different time scales.

After this first evaluation mostly focused on the accuracy of the gesture recognition algorithm, other evaluations will have to be performed on GRIGRI. First, the speed of interaction it allows will have to be compared with that of more traditional interaction techniques. Such an evaluation will involve a more sophisticated experimental setup: in most cases, it is impossible to determine the time at which an action began without a video recording of the scene. If pen computing appears to be faster than mouse and menus as expected, other evaluations will have to assess the usability of this technique in a more realistic environment: longer training periods will probably improve the accuracy of the recognition, while denser traffic and longer work periods could evidence new problems.

The system will also have to be improved. A consistent gesture set will have to be designed and evaluated. More importantly, the hardware will have to be modified to account for real work situations. This could take different directions. First, a better technology will have to be found to reduce arm strain and improve the integration between the display and the sensitive layer. This will probably take some time, even though other application domains have the same needs. Furthermore, the integration of that interaction technique in a real workstation will have to be studied carefully. It is improbable that pen-based computing will be usable on 20" square screens. Solutions might involve using such a large screen as a display only, while a smaller flat horizontal screen would be used to interact with the ATC system, thus replacing the flight strips. One could even think of locating that flat screen between the radar controller and the planning controller, providing them with a support for collaboration. However, this would require touch screens to allow multiple touches at the same time, a need that is not fulfilled yet.

While those evolutions will take some time, other uses of pen computing might become useful within a shorter period of time. A number of reactions provoked by GRIGRI show that pen computing or touch screens might be useful even without gesture recognition. For instance, tower controllers express the need to take notes on the screen and be able to send those notes to other users. This could be easily done with a stripped down version of GRIGRI. In a similar way, approach controllers need to take notes at a very fast rate. This could be done by using pen computing without recognition, or even by using a touch screen to implement fast menus. Another application might be the interface for pseudo-pilots involved in the training of controllers. Such an interface, used to 'fly' multiple simulated aircraft, shows strong similarities with what could be the control interface in presence of an air-ground data-link.

Finally, flight strip management will have to be studied. The fate of flight

strips does not appear to be definitively sealed. While it appears that current electronic strips are not a panacea, different solutions can be studied. First, the interface to electronic strips could be improved by extending GRIGRI to flight strips. This would ideally involve touch screens that would be able to discriminate between a pen and a fingertip, so as to be able to move strips and write on them. Another solution would be to 'augment' paper strips with computing capabilities, along the lines of augmented reality. This option will be studied in the near future at CENA. Finally, flight strips could be replaced by a new way of taking notes, as suggested by comments of users on GRIGRI: even though no flight strips were proposed, some of them had the same feeling as with flight strips.

CONCLUSION

We have described in this article the reasons why we considered mark-based input as an interaction technique for air traffic control. We have explained the choices made while designing an experimental workstation based on that technology, and we have described the most significant technical issues encountered. Finally, we have given first evaluation results that make us confident regarding the possible applications of pen computing to air traffic control. Nevertheless, there are many hurdles to overcome when designing interactive systems for a domain such as air traffic control. The techniques that we are studying have only been over the first of those hurdles. Then, if they prove worthwhile, they will have to be integrated in a full size workstation. They will also have to be tested with air traffic controllers under stress; the fatigue induced by their use will have to be measured; their impact on the collaboration between a radar controller and a planning controller will have to be studied. If all those hurdles are overcome, then pen computing might become part of future air traffic control workstations.

ACKNOWLEDGEMENTS

The first version of GRIGRI was developed by Gwenael Bothorel, using previous work by Thomas Baudel (University of Paris-Sud). The current version was built with Jean-Luc Dubocq and Frédéric Lepied. Thomas Baudel and Wendy Mackay provided helpful comments and helped with English grammar.

Bibliography

- [1] T. Baudel and M. Beaudouin-Lafon. Charade: remote control of objects using free-hand gestures. *Communications of the ACM*, pages 28–35, July 1993.
- [2] E. Bier, M. Stone, K. Fishkin, W. Buxton, and T. Baudel. A taxonomy of see-through tools. In *Proceedings of the ACM CHI*, pages 358–364. Addison-Wesley, 1994.
- [3] J. Callahan, D. Hopkins, M. Weiser, and B. Shneiderman. An empirical comparison of pie vs linear menus. In *Proceedings of the ACM CHI*, pages 95–100, 1988.

- [4] S. K. Card, T. P. Moran, and A. Newell. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates, 1983.
- [5] S. Chatty. Extending a graphical toolkit for two-handed interaction. In *Proceedings of the ACM UIST*, pages 195–204. Addison-Wesley, Nov. 1994.
- [6] C. Frankish. Recognition accuracy and user acceptance of pen interfaces. In *Proceedings of the ACM CHI*, pages 503–510. Addison-Wesley, 1995.
- [7] C. Garoff-Mercier and M. Chollet. Analyse de l'activité des contrôleurs du trafic aérien : utilisation des vecteurs d'information et des communications. Technical Report R90-07, Centre d'Études de la Navigation Aérienne, 1990.
- [8] D. Goldberg and C. Richardson. Touch-typing with a stylus. In *Proceedings of the ACM CHI*, pages 80–87, 1993.
- [9] G. Kurtenbach and T. Baudel. Hypermarks: issuing commands by drawing marks in Hypercard. In *CHI'92 Posters and Short Talks*, page 64, 1992.
- [10] G. Kurtenbach and W. Buxton. Issues in combining marking and direct manipulation techniques. In *Proceedings of the ACM UIST*, pages 137–144, 1991.
- [11] A. Meyer. Pen computing. A technology overview and a vision. *SIGCHI Bulletin*, 27(3):46–90, 1995.
- [12] J. K. Ousterhout. *Tcl and the Tk toolkit*. Addison-Wesley, 1994.
- [13] D. H. Rubine. *The automatic recognition of gestures*. PhD thesis, Carnegie Mellon University, 1991.
- [14] D. Venolia and F. Neiberg. T-Cube: a fast, self-disclosing pen-based alphabet. In *Proceedings of the ACM CHI*, pages 265–270, 1994.
- [15] P. Wellner. Interacting with paper on the Digital Desk. *Communications of the ACM*, 36(7), July 1993.

BIOGRAPHIES

Stéphane Chatty is a researcher at Centre d'Études de la Navigation Aérienne, where he heads a division. He graduated at École Polytechnique (Paris, France) and École Nationale de l'Aviation Civile (Toulouse, France), and holds a PhD in computer science from the University of Paris-Sud. His research interests are the software aspects of human-computer interaction, multimodal interaction and computer supported collaborative work.

Patrick Lecoanet is a research staff at CENA, where he leads a development group working on interactive software components for air traffic control prototype workstations. He has authored several user interface tools widely used within CENA and the Internet community. His technical interests are programming languages and user interface toolkits.

Christophe Mertz works at Centre d'Études de la Navigation Aérienne on hire from the French software house CR2A-DI. He graduated at École Nationale Supérieure de Télécommunication de Bretagne (Brest, France). He manages several projects centered on new Human Computer Interactions and their evaluation with Air Traffic Controllers. These projects use rapid prototyping software tools to develop mock-ups for simulations with controllers.

List of Figures

1	An annotated flight strip	3
2	Manipulating a flight strip through menus.	4
3	The physical setup of GRIGRI	7
4	The screen of GRIGRI	8
5	The set of gestures recognized.	10
6	Entering a flight level	12
7	Recognition rates during the experiment	16
8	Recognition rates with the revised algorithm	17