

# PROGRAMMING AND CONTROL OF A ROBOT WITH GESTURES



2/8/2009

A NOVEL APPROACH TO HUMAN-ROBOT  
INTERACTION

Mathieu GOEPFERT

Mälardalen Högskola

Supervisor: Baran Cürüklü, MDH



## Acknowledgement:

I sincerely want to thank my supervisor Baran Cürüklü who helped me every time I needed support and for being so kind to me.

And I want to give my sincere thanks to the ISPG and Christophe TOLLU who gave me the chance to do this wonderful experience that will help me all my life.

Sincerely,

Mathieu GOEPFERT

## **Abstract:**

Robots will be very important in the world of tomorrow. Even today we can see changes; we have robots to help the physically challenged and to entertain our children. Tomorrow we will have better robots to help in the medical field and possibly take us to the far reaches of space. From the deepest ocean to the farthest planet, robots will help us get there; robots are yesterday, today, and tomorrow. Robots will be the key to the next step of human evolution.

That's why ABB, in a partnership with Mälardalen University, tries to simplify the utilisation by trying to find different ways to instruct industrial robots. In this view, we can use some gesture, some speech and some language to control the robot.[15,16]

The project we are working is trying to regroup all of these methods and will show what the best method is and can be helpful for the company who want to control their robot in a new way.

**Keywords:** movement recognition, robot, HMM, Java, senseboard, GT2k, HTK, HMMPak

## Table of content:

I)	Introduction	4
II)	The project	5
	a. General Overview	5
	b. The different tasks on the project	6
	c. The Hand gesture task	6
III)	The resources	7
	a. The senseboard	7
	i. <i>The hardware</i>	7
	ii. <i>The data</i>	8
	b. The Hidden Markov Model	8
	c. HTK and GT <sup>2</sup> k toolkits	10
	i. <i>General Overview</i>	10
	ii. <i>Description of the toolkits</i>	10
	iii. <i>Using GT<sup>2</sup>k</i>	10
	1. Prepare the data	10
	2. Data Collection and Annotation	11
	3. Training	12
	4. Validation and system performance	12
	5. Recognition Application	13
	d. HMMPak	13
	i. <i>Overview of the package</i>	13
	ii. <i>Using the package</i>	13
	e. <i>The language</i>	14
	f. <i>The Interface</i>	15
IV)	Evaluation of the best solution	16
	a. Htk/gt <sup>2</sup> k	16
	b. Hmmpak	16
	c. Conclusion	17
V)	The solution	18
	a. General overview	18

	b. Evaluation of the solution	22
VI)	Future Work	25
VII)	Conclusion	26
VIII)	References	27
IX)	Appendices	29

## **I) Introduction:**

In numerous companies today, people use robots. The introduction of this kind of robots into the production process is a really important step in the company life. But unfortunately, in a couple of cases, the utilisation of this robot is really hard, takes a lot of time and could be used only by some people who has advance knowledge in robotics.

For a lot of robots, you have to program a sequence which will be executed as often as you need it. The major problem is that you have to redo this sequence anytime you change something in the process or production. The time that you use is really important and that means you use a lot of money. It is a really decisive domain for the health of the enterprise. [19]

The robotic field is really important in the company world because robots are present everywhere.

That is in this way that the project begins. The project has to use all the body's possibilities to control the robot. You have to try to combine all the different input (voice, gesture, video,...) you can have and after analyse all of them. [3]

With all these different input, the major problem is to create an application easy to use and powerful. Everybody has to use it without problems but the performance has to be maximum.

## **II) The project:**

### **a) General Overview:**

This thesis is a work about a part of a system called the  $\mu$ -Intelligent Human-Robot Interaction Architecture ( $\mu$ -iHRI)[1]. It consists of a novel high level language and cognitive robot architecture [1]. The  $\mu$ -iHRI system is designed to be highly intuitive to use for task experts as well as other user groups. This system will give the opportunity to the user to instruct the robot, a process which is fundamentally different from traditional robot programming. This process will be carried out using combination of following modalities:

- hand gestures,
- natural speech,
- Vision-based 3-dimensional object recognition,
- Vision-based hand and body posture recognition.

The system will give some feedback to the user like a real conversation with another person. We can present the system like this:

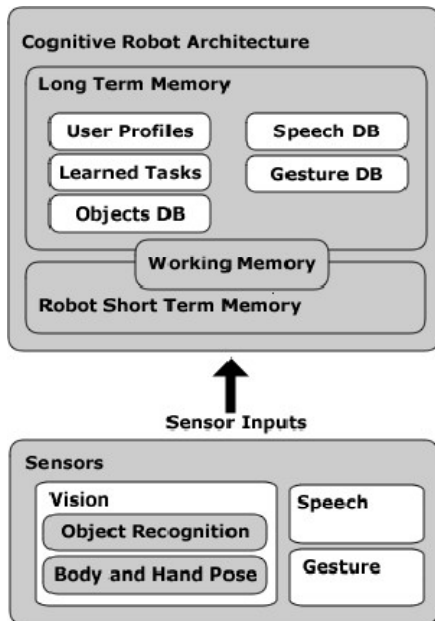


ABB is the partner of the university on this project. ABB has a big robotic department which is really active. The company want to find new way to control their robots for the future.

The real goal is to create an architecture which will be used in the future for the next robot generation and be easy to use. My thesis goal is to develop the gesture part of the project.

## b) The different tasks on the project:

In this project, there are four different tasks to do:

- **Hand gesture:**  
You control the robot by moving your hand. So you use a device to characterize the movement of your hand in space. You create a vocabulary which can be learned and understand by the system. This vocabulary has to be really clear because it will have a lot of influence on the accuracy of the recognition and on the result. For that you can use a device to record the movement, like a senseboard.
- **Speech:**  
This approach is really different than the first one because you have to speak with the system. The vocabulary, you will use, has to be really clear and the word different for two different tasks.
- **Vision-based 3-dimensional object recognition:**  
You have to recognize an object and model with its position in the space. You have a

camera which looks towards a special place and who take some picture of the scene. The system will analyse if something is in the field. You can interact with this object and the system will react in consequences.

- **Vision-based hand and body posture recognition:**

This last approach is really important and was used in a couple of other field like video-games with the “Eye Toys” by Sony. It just a camera who record your entire move and which has a database of movement in his system. When she see that there is a modification in front of the camera, the system analyse the scene and execute the action according to the modification.

Now that you know more about the project in general, we will develop more about the hand gesture part which is the subject of the thesis.

### **c) The Hand gesture task:**

Like you see previously, a lot of different tasks compose this project. My thesis is a work on the first part which is the hand gesture. The main idea is to control a robot with your hands. That means you have to move them and the robot will react.

In order to record the movement of your hands, we have to use a small device called “senseboard” and create a program that can analyse the data which are coming from this device.

The robot has to react to the following action:

- activate the system
- move forward
- move backward
- move up
- move down
- grab an object
- release an object
- cancel an order
- stop the system
- rotate an object.

For each of the movement, we have to create a language of movements who will be easy to do.

## **III) The resources:**

In order to analyse the data and get the data, we will have to use some tools. We will present it.



## a) The senseboard:

### i) The hardware:

The senseboard consist of one or two hand-worn units that each contains an embedded CPU, firmware, Bluetooth, a rechargeable Li-Polymer battery and the Senseboard GRT [14].

You can also integrate the senseboard into gloves.

We can see on the following picture the senseboard:



The senseboard is used to produce typed letters and numbers, provide mouse functions, enable movements, gestures and controls to be defined by industry, application, form factor. This technology is used in mobile computing, industrial controls, gaming and entertainment, and healthcare.

### ii) The data:

When you use the senseboard you have this kind of data:

% SQUARE\_LEFT ← the movement that the data describe

120 9 0 0 0 0 0

366.1 -199.6 -1030.3 -499.8 -56.6 25.3 -25.2

366.1 -199.6 -1030.3 -508.5 -49.3 25.2 -24.8

.....

348.9	-297.3	-1125.0	-477.2	-98.1	24.9	-24.3
374.9	-295.3	-1200.8	-604.4	-86.6	24.8	-24.0
MovX	MovY	MovZ	RotX	RotY	$\alpha$	$\beta$

We can see that you have seven columns which represent each one a special data:

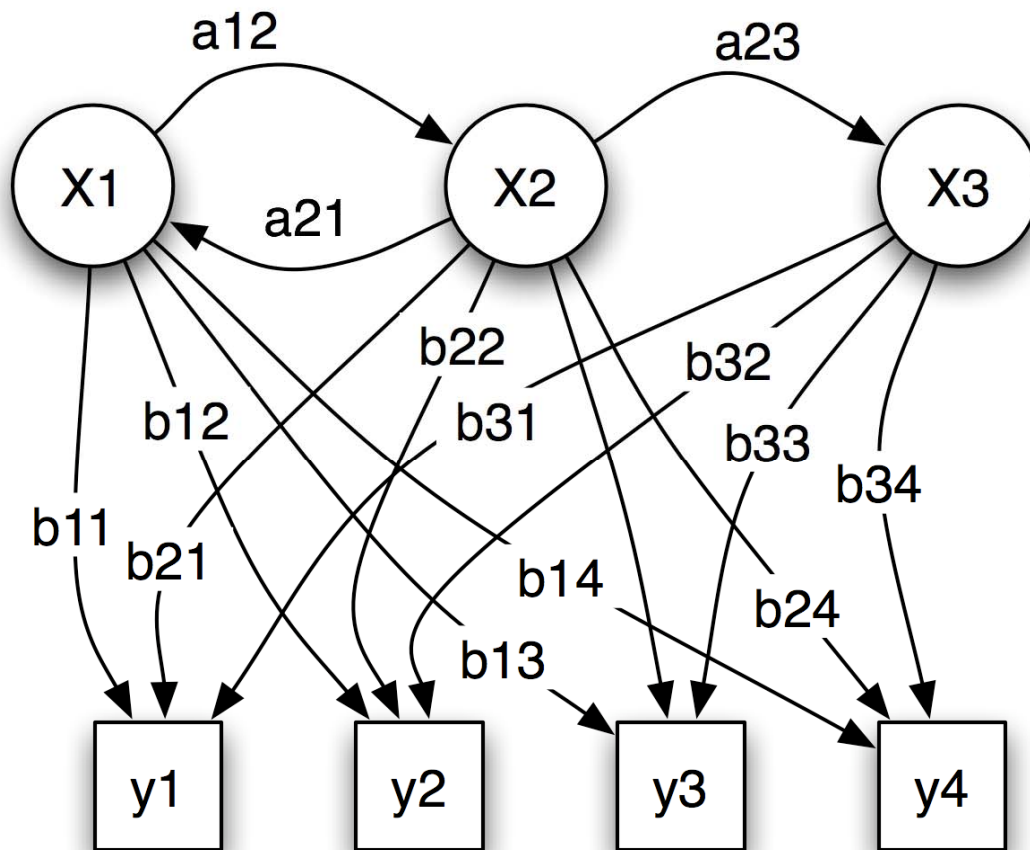
- MovX is the movement on the X axe
- MovY is the movements on the Y axe
- MovZ is the movement on the Z axe
- RotX is the rotation on the X axe
- RotY is the rotation on the Y axe
- $\alpha$  and  $\beta$  are two values that we don't use

## b) The Hidden Markov Model:

A hidden Markov Model, that you can call HMM, is a statistical model in which the system is assumed to be a Markov Model process with some parameter that you don't know the value. The challenge is to determine the hidden parameters using the observable parameters.

In a *hidden* Markov model, the state is not directly visible, but variables influenced by the state are visible. Each state has a probability distribution over the possible output tokens.

Hidden Markov models are especially known for their application in temporal pattern recognition such as speech, handwriting, gesture recognition [7, 9, 10].



Probabilistic parameters of a hidden Markov model

$x$  — states

$y$  — possible observations

$a$  — state transition probabilities

$b$  — output probabilities

## c) HTK and GT<sup>2</sup>k toolkits:

### *i) General Overview:*

GT<sup>2</sup>k is a toolkit who was design for gesture recognition and if you want to use this toolkit, you have to install another toolkit which was used at the beginning for speech recognition: HTK.

### *ii) Description of the toolkits:*

#### **HTK:**

The Hidden Markov Model Toolkit (HTK) is a portable toolkit for building and manipulating hidden Markov models. This toolkit is used for speech recognition research.[8]  
This toolkit consists of a set of library modules and tools available in C.  
These tools provide facilities for speech analysis, HMM training, testing and analyse of the results. The version contains also some example and documentation [12].

#### **GT<sup>2</sup>k:**

The gesture recognition is now a more common interaction tool.  
We can use the Hidden Markov models for the gesture recognition. Today it exists some HMM toolkits for the speech recognition and these can be adapted and be used for a gesture problem. But It requires some knowledge in speech recognition field.  
We have a toolkit which use a speech recognition tool HTK and provide some new function to support the gesture recognition. This is the Georgia Tech Gesture Toolkit: GT<sup>2</sup>k. You can have some trainings models for real-time and off-line recognition.

### *iii) Using GT<sup>2</sup>k:*

You have three different steps. The first one is to prepare the data, the second one is to train and the last one is to recognize [4, 6, 11].

#### 1) Prepare the data:

##### 1.1) Designing Gesture models:

For each gesture, a new HMM is create. This is a special topology, that is to say that a HMM has a set of states and the transitions between these states with some probabilities of appearance. For example if you have 3 states, you have may be 3 chances on 5 to go from the state 1 to the state 2 and 2 chances on 5 to go to the state 3.

If you want to design the gesture model, you have to involves some insight into the structure of the data and you have to try what is the best topology you have to use and make some try-out.

This toolkit has a special function which is that it can visualize the model which is easier for the user who is not into this field..

### 1.2) Specifying a Grammar:

The toolkit needs to be used with a special grammar to be able to perform the training and recognition part. For this operation you have two possibilities, you can generate it automatically by the toolkit or you can specify your own grammar in a text file. If you choose the first solution, the grammar will only allow gesture to be recognized one at a time. So if you want a continuous recognition you have to create your own grammar.

The grammar is easy to generate. It is a special language which uses variables and commands which are specified by using the symbol “\$”. The commands are really simple. Indeed it is a text strings which is associated with the variables and each of them can contain more than one commands by using the separator “|”. You can see an example in our case:

**\$gesture = put on | start | stop | cancel | forward | backward | up | down | grab | release | reset | rotate;**

**( \$gesture )**

Each name in the list corresponds to the name of a folder. This folder contains all the data for the movement. That is to say in the folder start, you will find some file which represents the movement start. So in the project, you will find twelve different folders which represent the twelve movements, and in each folder a couple of files which represents the different occurrence of the movement. If the folder start has ten occurrences, the system can recognize the movement in ten different ways.

### 2) Data Collection and Annotation:

In this toolkit, some sensors are used to take the data. You have some common sensors that you can use as cameras, accelerometer, laser, microphone or some other motion capture devices. All of these sensors return data measurements of the movements they observe. When you got the data you have two choices, you can use the data like this (raw data) and use it for the recognition or you can make some modification on it if the task needs it. The data is a numerical vector, feature vectors.

There is some information which is dependent to the application which is the length of the vector and the range of values. We can say that a typical gesture example is a sequence of these feature vectors.

The data can be also annotated by the user because the system has to understand the data that it receives by the sensor. So the user has to specify which gestures appear in each of the training examples.

### 3) Training:

When the preparation phase is complete, you have to do the training part. The training of the model requires that the user select a training validation method and configure some parameters. The process is automated and return results and models which can be used to analyse or for the recognition part. The toolbox uses a couple of training algorithms but you can also program some better and more efficient algorithm and use them.

You will receive some feedbacks from the training validation method concerning the training process. The system will separate the data in two sets, a training set and a validation set. The training set is used to train the models. The validation set is used to measure the performance of the trained models. That is to say, that for example you take 60% of the data which will be trained and the last 40% is for the validation and the recognition part.

You have two techniques for the training/validation part which are leave-one-out validation and cross validation:

- Leave-one-out validation selects one data as the validation set and uses the rest of the data as the training set. The training/validation process is repeated for every permutation of the data set with one element "left out" of the specified training set.
- Cross-validation selects randomly a percentage of the data (typically 66, 6%) as the training set. The remaining data (typically 33, 3%) acts as the validation set.

### 4) Validation and system performance:

The methods, concerning the training/validation, provide a measure of the system's performance based of the accuracy of the recognition. The toolkit uses for accuracy the data which are substitution, insertion and deletion errors. The substitution errors appear when the system incorrectly classifies a gesture, the insertion error appear when the system make a mistake about the occurrence of a gesture and the deletion errors appear when the system fails to recognize the gesture within a sequence of gestures.

We can calculate the accuracy with this operation:

$$\text{Accuracy} = \frac{N - S - D - I}{N}.$$

In this operation  $S$  represent substitutions errors,  $I$  represent insertion errors,  $D$  represent deletion errors, and  $N$  represent the total number of examples. When recognizing gestures in isolation, the values for  $D$  and  $I$  will always equal zero.

## 5) Recognition Application:

The models that you create during the training operation can be used now for the recognition of a new data. When the system receives a gesture which is not classified, the toolkit calculates the likelihood of each model. And the system compares the result and takes the most likely gesture. Once a model of each of the gesture has been trained, the system can use it independently of the training system. When the system recognized a movement, you have to apply the action which is relative to the data.

### **d) HMMPak:**

#### ***i) Overview of the package:***

We know that the HMM are really important and really good when you try to recognize some sequence. One of the best domains when you can use that is the speech recognition like you can see in a lot of project. In this case, you can see how HMM is powerful and can be useful for an application [5, 13].

But now people try to use this method for gesture recognition and a couple of areas. To be the best, you must have a package that is simple and can be really powerful and solve your problem. That's why Troy L. McDaniel, for the Center of Cognitive Ubiquitous Computing (CUbiC) from Arizona State University, creates a package which is written in two powerful programming languages: Java and C++.

You can find a zip file with the manual, a demo file and the package on the website: <http://www.public.asu.edu/~tmcdani/hmm.html>.

#### ***ii) Using the package:***

The first step when you want to use a package is to set up the Java package. For that you have to copy the file in the folder when you kept your packages. When this step is done, you call your package in your Java program like this:

```
Import HMMPak.*;
```

With this package you can do a lot of different operation. You can create your own HMM, in this case you have to put all the value for all the data necessary for the model that is to say the initial probabilities, the transition probabilities and the bias probabilities.

You can also generate some observation sequences and after use the different algorithm to know the probability of an observation.

But in our case, the most important function is that you can learn an HMM from a training data. That is to say, that you can have a file with your data and the system can train it for you and create the HMM according to the specific data file.

Before you use the K-Means Learner, you have to put the data in a certain format.

Indeed, the system accept only data like this:

X Y Z

12.3 14.4 16.7 -23.0

14.7 15.0 89.3 67.0

456.4 34.7 -345.7 345.0

We can in the first time comment the header. The first element X is the number of dimension, the second element is the length of observation sequences and the last one is the number of observation sequences in the training data.

In this example, we can put X = 1, Y = 4, Z = 3.

That is to say each column is a different data that we use to train.

But we can have another affectation. If X = 2, Y = 2, Z = 3.

That is to say that (12.3 14.4) is one observation with 2 dimensions and the same for the rest.

After that small modification of the file, we have to use the K-Means Learner algorithm which take the training data file as input and learn the HMM.

For that you have to give the name of the data file, the number of state for the HMM and the number of desired iteration. The more the value of iteration large is, the more accurate yours clusters will be.

When you have all your data trained, you can try to recognize one of them. For that you can use the method that compare two HMM and say if they have something in common or not. If they are nothing in common, the function result is -1. If they have something in common, the result is 0 or a number between 0 and 1 but really small.

For using this method the two outputs had to have the same output vocabulary and the same output vocabulary size.

### ***e) The language:***

Everybody knows that this project is about the gesture recognition. But one important point was to create a really simple and accessible language.

You can ask to one guy who controls a robot to do a lot of effort because you don't how his



condition is. May be he is too old to do a lot of movement or may be he has some handicap. So the most important was that the language has to be accessible by everybody. We speak with all the team about this part important because it was a really preoccupation. I propose one kind of language. You can find that on the CD with the report because it is only video.

We decide to develop only a couple of movement but using two approaches: one using only one senseboard and the second one two senseboard (one for each hand).

We describe with the vocabulary the following action:

- "Put on" which is the movement to put on the system;
- "Start" which is the gesture to say to the system that it has to recognize movement since now;
- "stop" => stop the recognition;
- "cancel" => cancel an order;
- "forward" => the robot has to go forward;
- "backward" => the robot has to go backward;
- "up" => the robot has to go up;
- "down" => the robot has to go down;
- "grab" => the robot has to grab the object;
- "release" => the robot has to release what he had grab;
- "reset" => the system reset;
- "rotate" => the robot has to rotate;

### ***f) The Interface:***

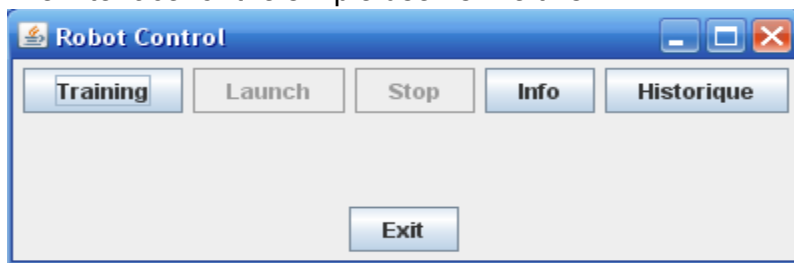
For the interface between the user and the robot to be better and more attractive, I decided to create a really simple Java interface in Java that I can use to begin the recognition part and the training part just to test if the algorithms are working [17].

So we have two kinds of people who can use the application. The first category is the administrators who have all the rights on the application and the second one is the normal user who can't create, modify and delete users.

The interface is not so different. We have here the interface for the administrator:



The interface for the simple user is like this:



## IV) Evaluation of the best solution:

### a) Htk/gt<sup>2</sup>k:

Positive Points:

All the tools you need to do some gesture recognition are design in these toolkits. So you don't have to create all the algorithm of data treatment. Furthermore, it looks really easy to use. A lot of example are present and help you to understand how the toolkits are working. Another important point is that the performance for the training part is really good, close to 100%. And indeed, it is really important for a company to have a really accurate program. The recognition part is good also. The performance is 80%.

Negative Points:

Our program has to work on Windows and Linux, the gt<sup>2</sup>k toolkit is working only on Linux. Even if the Georgia Tech University is working on a new version which can work on both platforms, this

version won't be available for a couple of months.

Another important point is that the documentation which is given with this toolkit is incomplete and hard to understand. Finally, when the examples are executed, the toolkit is not working well. Some movement are recognizing perfectly but a lot of them are not.

### **b) Hmmpak:**

#### Positive Points:

The package, which simulate an HMM, is really simple to use and all the commentary inside the source files are really precise and clear.

Java is the language used to create this package and this is an important point because you can try to change some properties or do a better and powerful package. And this package can be used on Linux and Windows. The training and recognition is close to 100%.

#### Negative Points:

The training of the data is a little bit slower. Furthermore, you have to know about the HMM model because you have to create the model by giving the good parameters to the method that will create the model. Another point is that all the data must have the same number of line so we will need to do a pre-treatment.

### **c) Conclusion:**

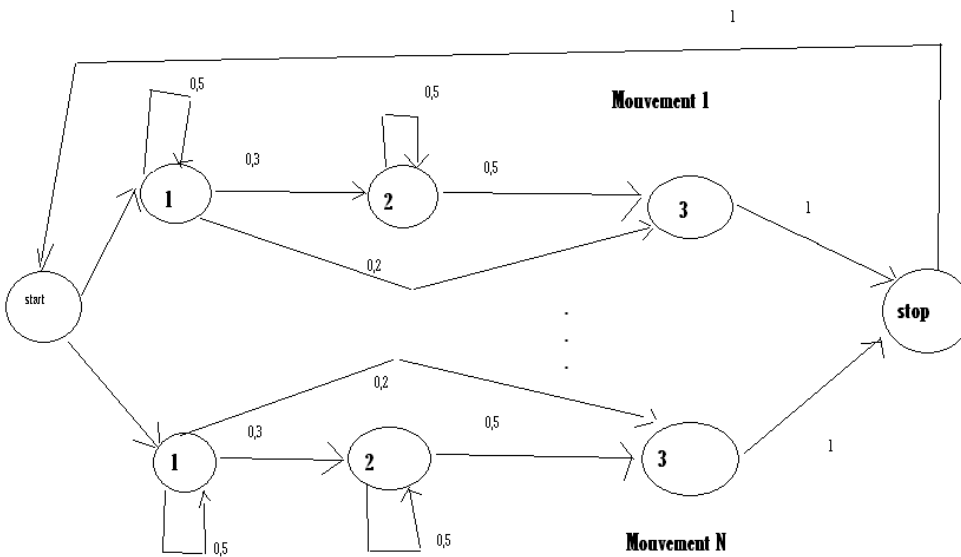
The two solutions have good and bad points but the one who need less work and who will be the best for our problem will be to use the Hmmpak. First of all, the package is written in Java so you can use it on both platform and you can modify some part if you want to make it powerful. Secondly the performance for the recognition part is better for the java package.

So we will use the Hmmpak for solving the problem.

## **V) The solution:**

### **a) General overview:**

Using the HMMPack, we have first of all to precise which HMM model we have to use. This model will be used to train the data and after do the recognition part as well. In our case, the HMM model could be draw like this. Each branch is for one movement and the number for the probabilities are chosen randomly:

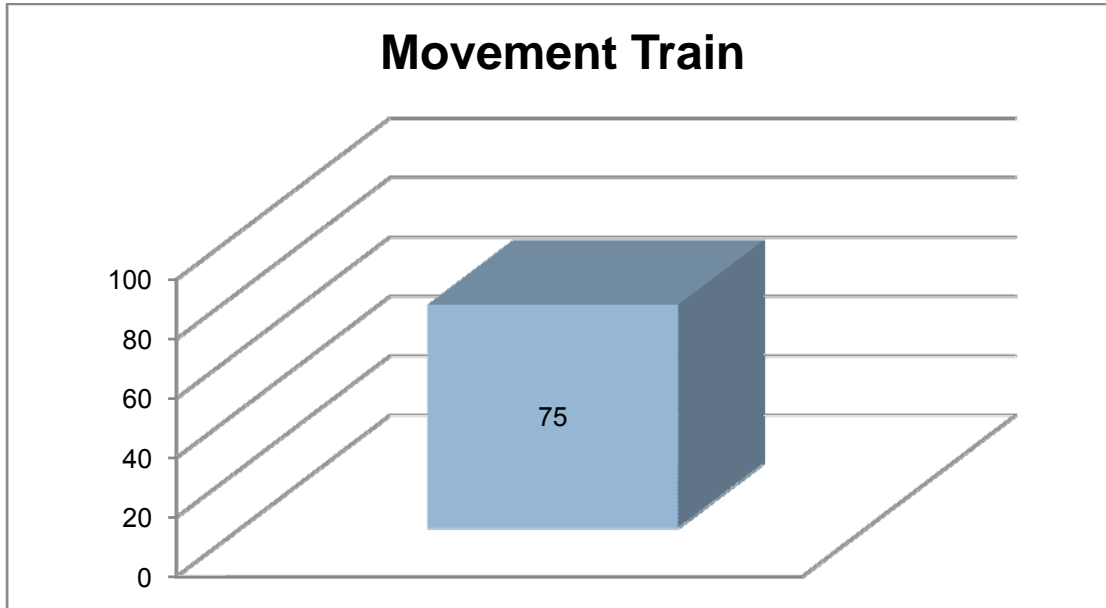


When you will use this model to train a data, you will have a result file which could be use to compare two data, so two movements.

The data are all in one file. Each data has a number of lines different that means we have to do a pre-treatment operation before you could try to train the data. The pre-treatment is simple; we have to find the smallest number of line and after take in all the different movement data just the number of lines we just decide. That means, if the number is 16 we have to take for all the data movement the 16 first lines. In this case, we could do the training and recognition part without problem.

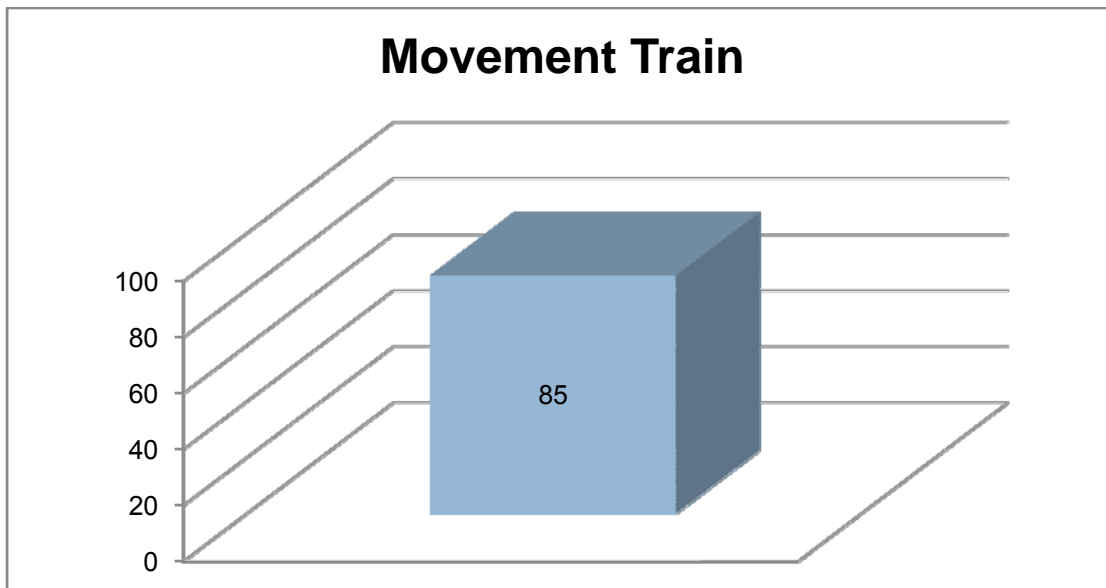
Furthermore, all the data will be used: one part for the training and the other one for the recognition. We try different percentages and we can see the result in the following diagrams:

If we take 40% of the data for the training and 60% for the recognition part, we obtain:



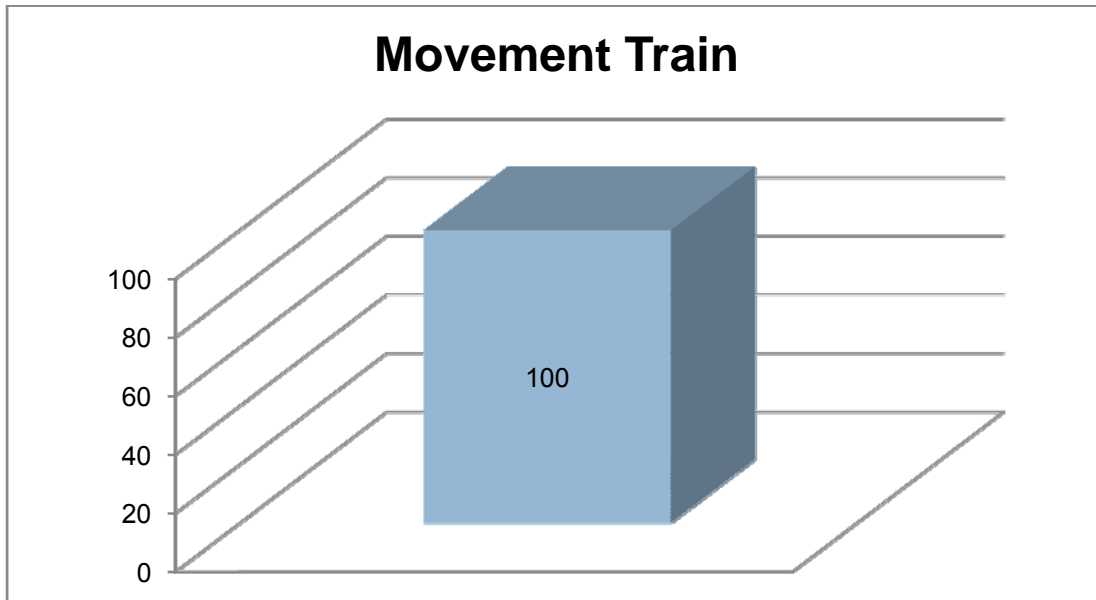
We can see on the diagram that 75 % of the movement which are contain in the file are trained. That means some of them can't be recognize because we used a too little amount of data for the recognition part. For example, if we had 100 movements in the file, only 75 will be recognized by the application because they were recognize.

If we take 50% of the data for the training and 50% for the recognition part, we obtain:



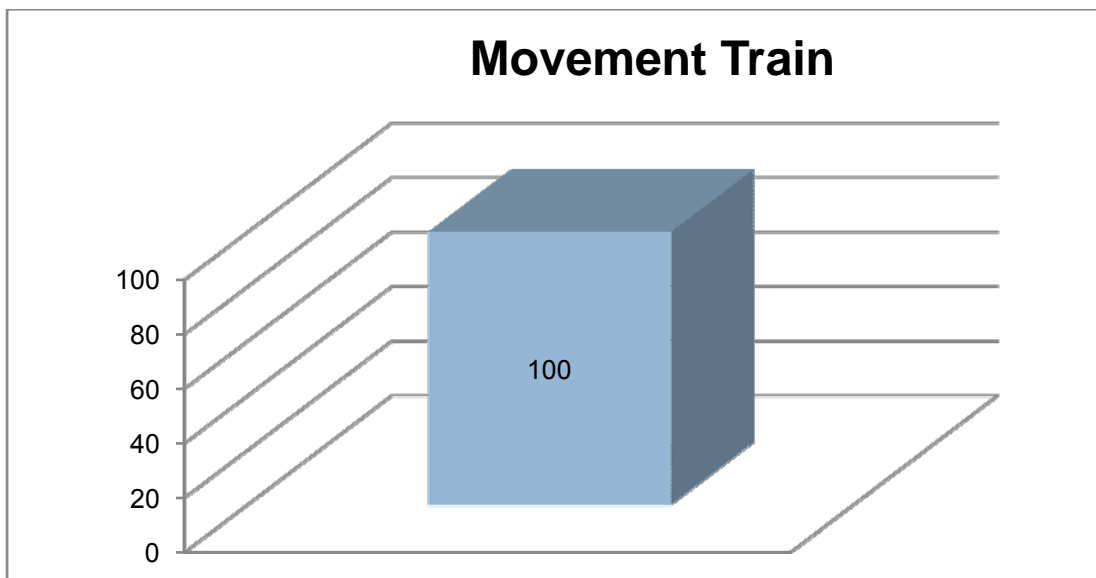
In this case, the result is better because we have 85 % of the movement which are recognize but the problem is that we have to be able to recognize all the movements of the file to have a result acceptable. So we have to take more data for the training.

If we take 60% of the data for the training and 40% for the recognition part, we obtain:



This is one important step because in this case we got 100 % of the movement will be trained and so will be recognized by the application. But we have to see if we take more of this data, what the result is. If the result is the same that will mean that this amount of training data is the best.

If we take 65% of the data for the training and 35% for the recognition part, we obtain:



We see that the result is the same so we can definitely decide that the best amount of training data is 60% of the file from the movement file and 40% for the recognition. We will work with this numbers for the rest of the result presentation.

So after some test, we see that the limit to have good results is to use 60% of the data for the training and 40% for the recognition.

In order to treat all the different movement in the training part, we create an algorithm who will take at least one iteration of each movement. So every movement, you had from the senseboard will be train and could be recognize. This algorithm is simple to explain. We first take a look of all the different movement from the senseboard and make a list of all of them. After that when we choose the 60% of the data for the training, we begin by taking one of each movement and after we continue to take randomly the rest of the data. So in this case, we could recognize every movement.

During the training part, we had an algorithm which assures that all the movement could be recognize.

The results we obtain will be the name of the movement which is recognized and it will appear in a special frame. After, we could do the action which is connected to this movement.

The algorithm for the recognition and training part is easy to understand. Indeed, you have the file which contains all the movement. The first part of the algorithm is to put all the data with the same number of data because it's one condition for the HMM Pack to work. So after we found the minimum number of line, we take in all the movement data this amount of data. The second task we have to do is to determine which movement will be taken for the training and which one for the recognition. For that we used a special function which will count the number of data in the file. It will select n randomly movement of the data file for the training using the following formula:

$$n = (60 * \text{number total of movement}) / 100$$

Every time a movement is chosen to be trained, the number of the movement is stored in a list. So you can't choose two times the same movement for the training or choose one movement for the recognition and for the training. You compare the movement you just choose to know if it's in the list. If it's not you can add it like a training movement. The algorithm stops when you got n movement.

When a movement is choose for the training, its number is store in the list and a HMM representation of the movement is generate. This file will be used to compare the different movement and used in the recognition part. At the end, in the HMM Train file, there is exactly n files.

When you got all your movements, you can use all the movement who are not trained. In the list, all the numbers of the movement trained are present. So you can deduce the movement left to recognize. A list of the movement left to train is generating by the algorithm. After this operation, you can begin the training operation and this part of the algorithm is really easy to use with the HMMPak.



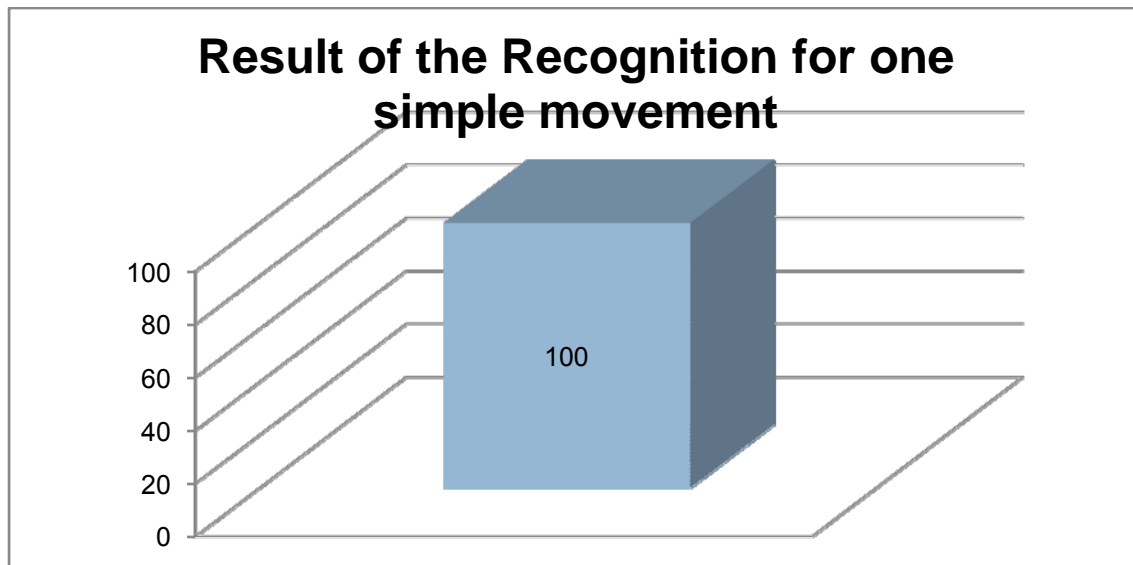
A movement try to be recognizing by the algorithm. A HMM file which represents the movement is generated. This file will be used in the function compare, which compare two HMM model. The system will try to compare all the HMM file from the training part with the one just generated. If a match is found, the system will return the nature of the movement. If not, a error message will be send. This message will inform the user that the movement wasn't trained.

### **b) Evaluation of the solution:**

For the evaluation of the solution, we will take 4 kinds of situations. The first one is taking only one movement and sees how accurate the system is. The second one is to take the 5 easiest movements to train and recognize. That means we have to choose the less complicated amount of data and see how the system reacts. The third case is that we will take 5 movements but randomly. The last one is to take all the data.

First case: The movement ARC\_RIGHT

This is one of the easiest movements. So we have to do the pre-treatment to the data. The training part is perfectly executed, the result is 100%. When we try to recognize the movement, we obtain the following results:

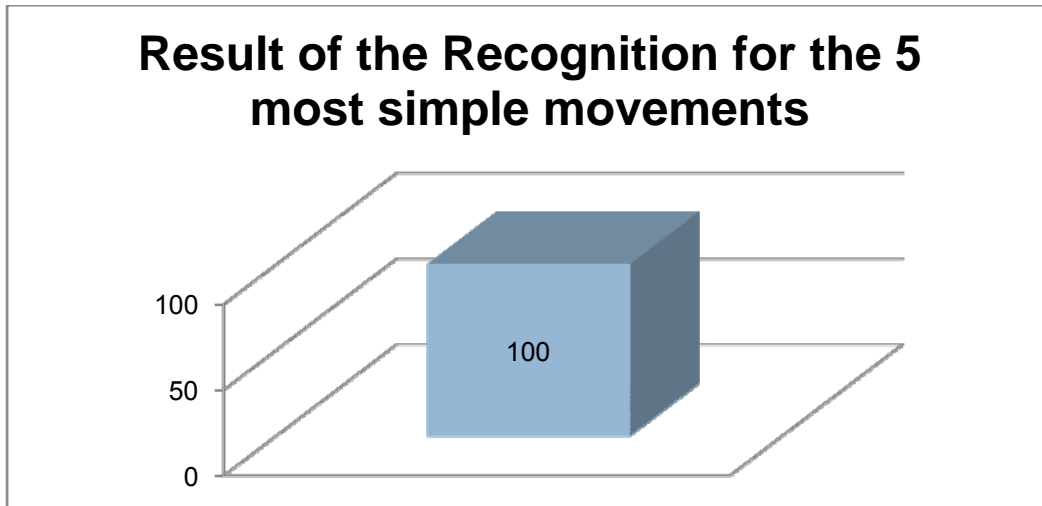


This case is the simple one, we can see that for only one movement which is store in a file. The file contains several iterations of the movement. The results are 100%. For a simple movement it's easy to use the application. That is to say if for a simple movement, the recognition is perfect.

### Second case: The 5 simple movements

In this case, we took the 5 simplest movements, that is to say the one who has the smallest amount of iteration and not a lot of line in their data. Because less you cut some line in the data, more the result will be accurate.

We can see the following result:

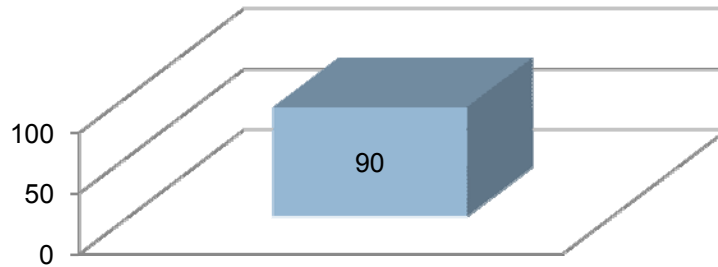


When you consider 5 simple movements, the result is the same as one simple movement. We have 100% of recognition with this kind of input data. The result is really good but it's for really simple movement, so we have to see how it is for some random movement.

### Third case: 5 movements that we take randomly

Now we took some movements randomly and see what the results are:

### Result of the Recognition for 5 movements



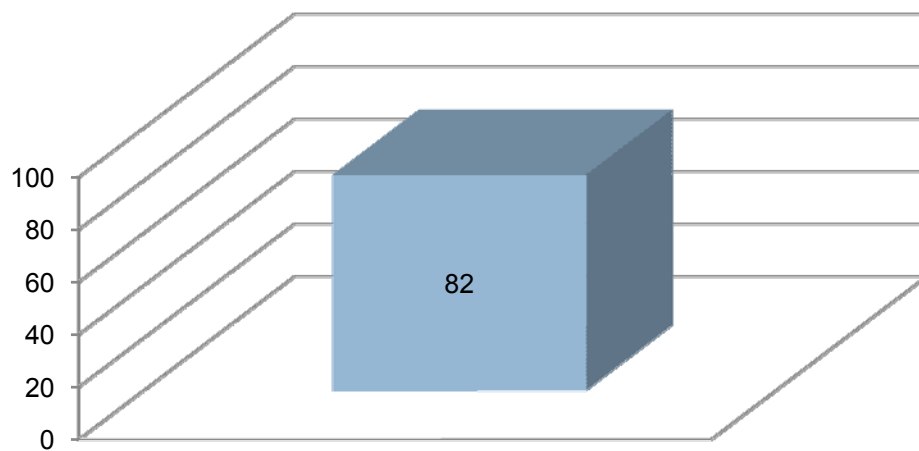
There is more problem when you use different movements because they have different number of lines and when you treat the movement you lose some information.

So in this situation, we have a really good result for the recognition even if we have some movements which are not recognized or it's not the good movement.

Fourth case: all the movements that we have in the data file

The last case is just taking the entire file and applies the algorithm of training and recognition on it:

### Result of the Recognition



So it's a quite good result but a couple of movement are not really recognize because they have a lot of data line and when you cut it during the pre-treatment you lose a lot of information. But this only happens for a small amount of data which is acceptable.

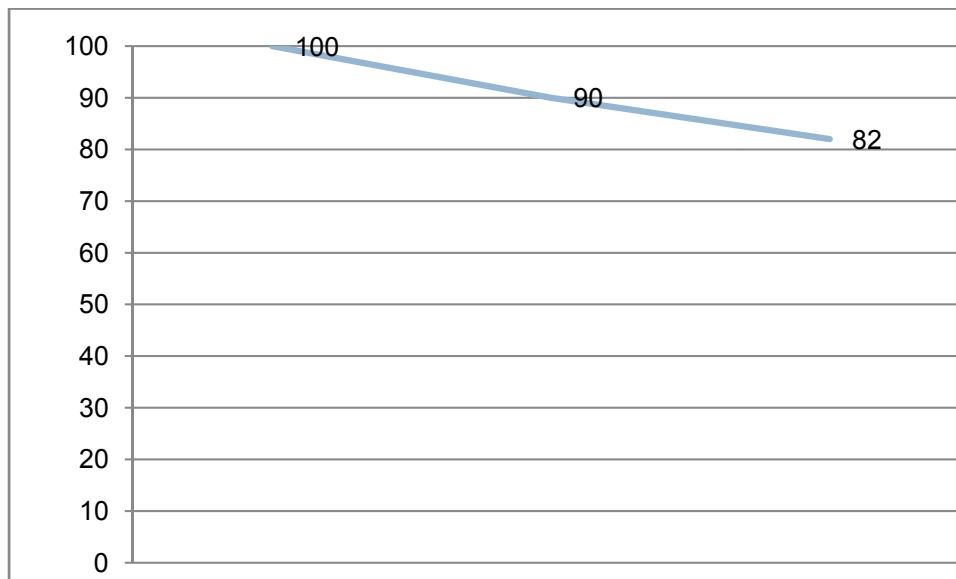
Now we study the rapidity of the recognition part because it's could be a powerful method but it has to be fast as well. For that we can take the five cases and try to see how fast the algorithm can work on all the data for the recognition part and the training part.

For the recognition part, all the process takes exactly 1,27s to be done what is not so fast but which is okay according that the time is not a really important key in this part of the problem. It is more important for the recognition part.

The recognition part is a little bit different. We have to check all the different cases. For the two first cases of recognition the time is 67ms and 78ms. For the third one the time is 95ms. But if we use all the data, we obtain a result time of 1.05s. We can see that the time is different according to the number of movement and the complexity of the data representing the movement. But this time is acceptable.

We need this time to recognize all the movement that means the operation of recognition is quite fast.

To conclude, we can see that this method is really useful and accurate and at least you will have 82% of good recognition if you use all the data. It's the result we found by doing a lot of experiment (at least 30) and we take the average result.



This result can show that the application is powerful but it can be improved by changing some data in the package or using some new tools. But this tool could be really useful.

## VI) Future Work:

One solution will be to use the new version of GT<sup>2</sup>k who will be released by Georgia Tech because

this version named GART could be use on both Linux and Windows. This version is creating in Java. Thus that will be easier to modify and understand. This version will be more stable than the actual version and will be really portable on every station.

Another solution could be to improve the accuracy of the HMMPack and try to have a better result with this powerful tool.

Finally you could try to find new tools more powerful and easier to use and try to have a better performance in both training and recognition.

## **VII) Conclusion:**

This study was about finding a good solution for a gesture recognition which a really important field in robotics today and will be really important in the next years. For that we focus on two different solutions. The first solution was using two toolkits HTK and GT<sup>2</sup>k which are complementary and really powerful. The second one was a java package which simulates, generate some HMM model and can compare them.

After a couple of test and comparison of the characteristics and our needs, we decide that the HMMPack was more powerful and useful. Indeed, we have a training percentage of 100% which means that the training part is done perfectly and the algorithm really powerful. The percentage of recognition is 82% which is a really interesting result because you have the majority of the data which are recognizing by the system.

Even if the result are satisfying, the system could be more powerful. One reason is the package is in java so you can modify a lot of parameters and function in a java code. That means, if a better algorithm could be found the results will better. A second reason is that we could try to reduce the constraint. In this system, the number of line of each data has to be exactly the same to be compare. So if we can change the algorithm and make it working for data which have not the same amount of information, that will be much powerful because we didn't have to cut some information.

Considering the evolution of the robotics and the new tools we could find in a couple of months or years, this system could evolve in a better way. For example, the new toolkit GART which is another version of GT<sup>2</sup>k could give us better results and helps us to make a much powerful system.

## VIII) References:

### a) Articles:

- [1] Johan Hägg, Batu Akan, Baran Çürüklü, Lars Asplund  
***Gesture Recognition Using Evolution Strategy Neural Network***, MDH, 2008
- [2] Jean-Christophe Lementec and Peter Bajcsy  
***Recognition of Arm Gestures Using Multiple Orientation Sensors: Gesture Classification***  
Intelligent Transportation Systems Conference, Washington, D.C., USA, 2004
- [3] Ronah Kabagambe  
***Minority Report System***  
Computer Engineering Program Master's Thesis, 20p, D-level, MDH, 2007
- [4] Tracy Westeyn, Helene Brashear, Amin Atrash, and Thad Starner  
***Georgia Tech Gesture Toolkit: GT<sup>2</sup>k*** December 13, 2003

[5] Troy L. McDaniel

***Java HMMPak v1.2 User Manual***

Center for Cognitive Ubiquitous Computing (CUBiC), Arizona State University, 2004

[6] Kent Lyons, Helene Brashear, Tracy Westeyn, Jung Soo Kim, Thad Starner

***Georgia Tech Gesture Toolkit: Supporting Experiment in Gesture Recognition***

Contextual Computing Group College of Computing Georgia Institute of Technology  
Atlanta, GA USA, 2003

[7] Troy L. McDaniel

***An Introduction to Hidden Markov Models and Gesture Recognition***

Center for Cognitive Ubiquitous Computing, Arizona State University, 2004

[8] Barbara Resch

***Automatic Speech Recognition with HTK: A Tutorial for the Course Computational Intelligence***

Signal Processing and Speech Communication Laboratory, 2003

[ 9] Hye Sun Park, Eun Yi Kim, Sang Su Jang, Se Hyun Park, Min Ho Park, Hang Joon Kim

***HMM-Based Gesture Recognition for Robot Control***

Department of Computer Engineering, Kyungpook National Univ., Korea, 2005

## **b) Websites:**

[10] Wikipedia website in French and English

[www.wikipedia.fr](http://www.wikipedia.fr) / [www.wikipedia.com](http://www.wikipedia.com)

[11] GT<sup>2</sup>k webpage:

<http://gt2k.cc.gatech.edu>

[12] HTK webpage:

<http://htk.eng.cam.ac.uk>

[13] HMMPak webpage:

<http://www.public.asu.edu/~tmcdani/hmm.htm>

[14] The senseboard webpage:

<http://www.senseboard.com>

[15] ABB webpage:

<http://www.abb.com>

[16] Mdh webpage:

<http://www.mdh.se>

[17] Java API:

<http://java.sun.com/j2se/1.5.0/docs/api>

[18] A Brief Overview of Gesture Recognition:

[http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL\\_COPIES/COHEN/gesture\\_overview.html](http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/COHEN/gesture_overview.html)

[19] Gesture recognition with a wiimote:

[http://fwiineur.blogspot.com/2008/05/gesture-recognition-with-wiimote\\_27.html](http://fwiineur.blogspot.com/2008/05/gesture-recognition-with-wiimote_27.html)

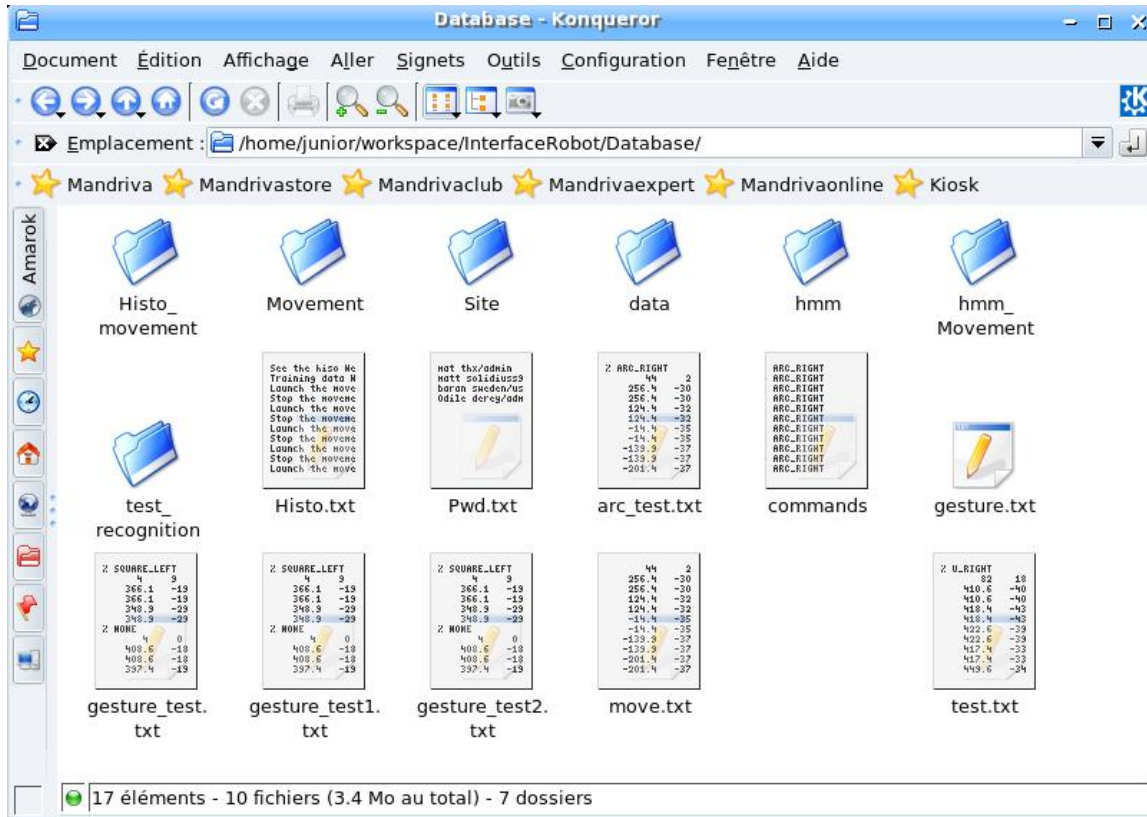
## **IX) Appendices:**

### ***User guide of the Java Application using the HMMPak:***

In this part, we have to show how you can use the application.

Before we begin to show how to use the application we can describe the folder when you have all the different files you have to use:

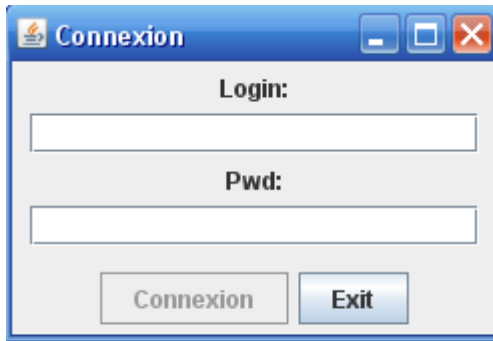




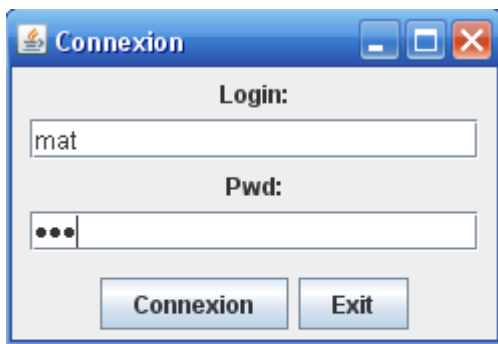
We can describe all the folder and files:

- Histo\_Movement is the folder where you can find all the precedent movement that the system recognize;
- Movement is the folder where you put the movement files;
- Site is the folder which has inside the two website I created (about the language and about me);
- Data is the folder where you have the data trained;
- Hmm is the folder where you find the hmm from the trained data;
- Hmm\_movement is the folder where you find the hmm from the movement that the system recognized;
- Histo.txt is the file where you can find all the action you have done with the application;
- Pwd.txt is the file where you find all the information about the users;
- All the others files are some test files.

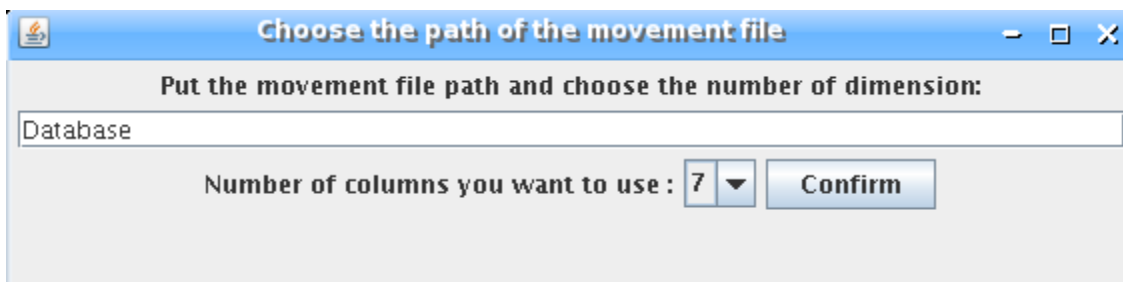
When you begin to launch the application you have the connection window:



The button “Connexion” is available only if you put a login and a password.  
If you put a wrong login or password a pop-up appears and you have to put a new login and a new password.

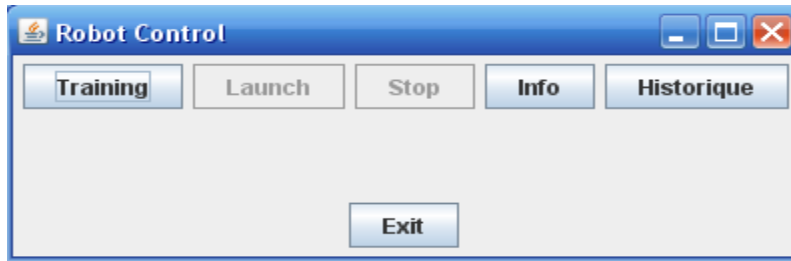


When you push the button “Connexion”, a new window appears and here you have to put the name of the folder when you have all your data. For me it is “Database”.  
You have to choose also how many dimension you want to use.  
In my case, I use 7.



And you have to push “Confirm” to go to the next window which is the principal one.

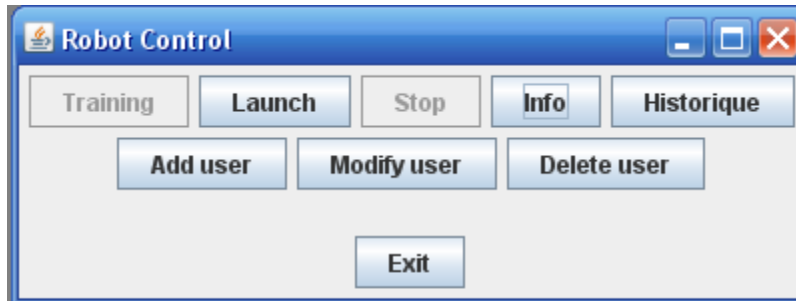
If you are a simple user, you will see this window:



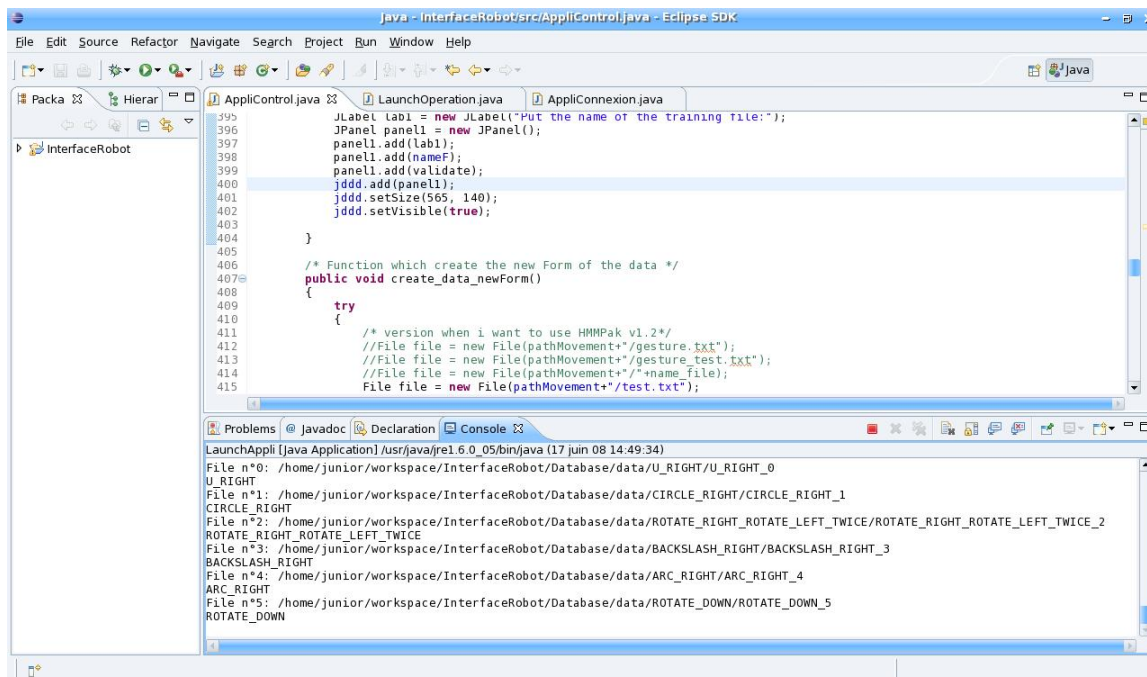
If you are an administrator, you will see this window ( after we will continue with this kind of window):



When you press the button Training, the application will train the data which are in your project directory and the button Launch will be available.



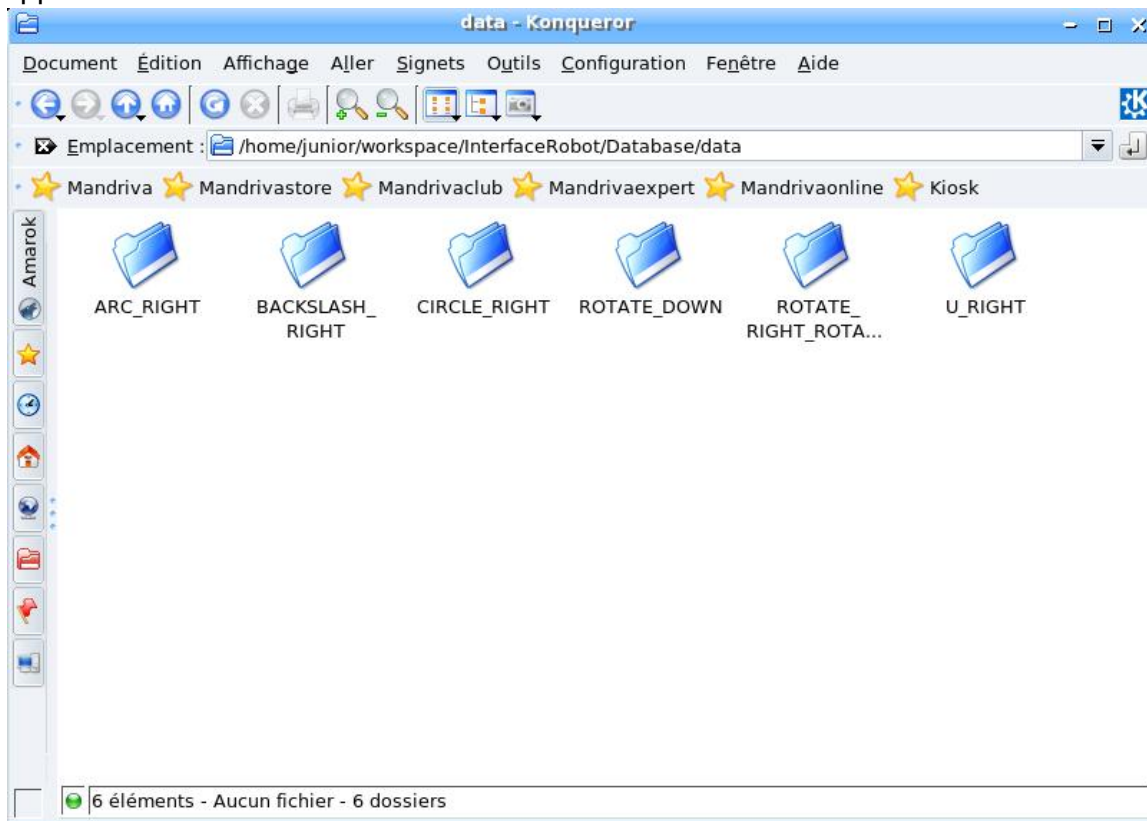
And you can see that in the console:



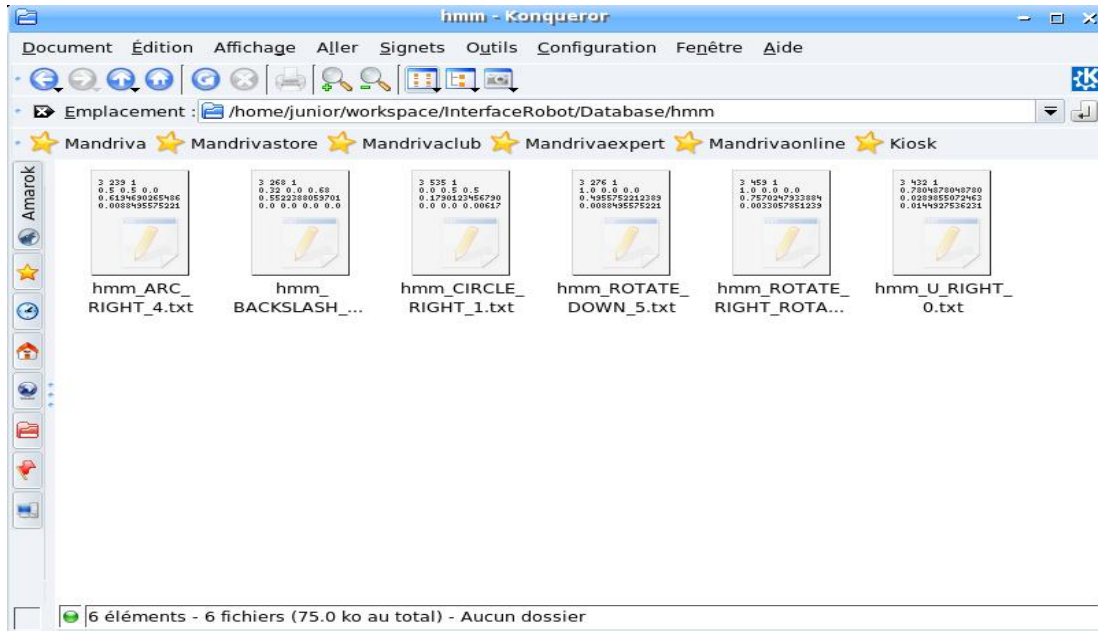
And on the screen, a new window appears:



That means that the data are trained now. And we can see in the folder “data” that one new folder appears with all the data inside.



You see the entire files, with the data. But you have to see that the program create the HMM model corresponding to each data.

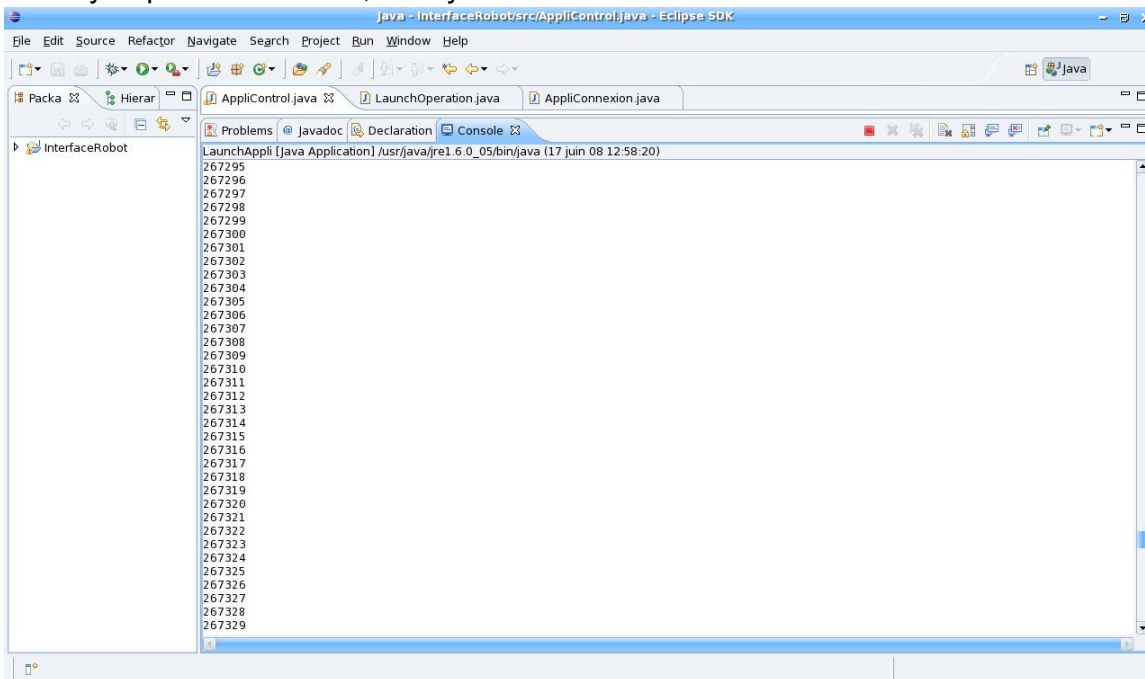


If you press the button Launch, that is to say you want to launch the recognize part and in this case the button stop could be available to stop this operation if you need.

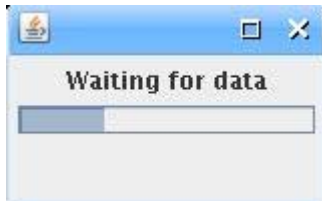
When you push the button Stop, you stop the recognize part and the button Launch is one more time available.



When you push this button, the system waits for a file with the name “move.txt”.

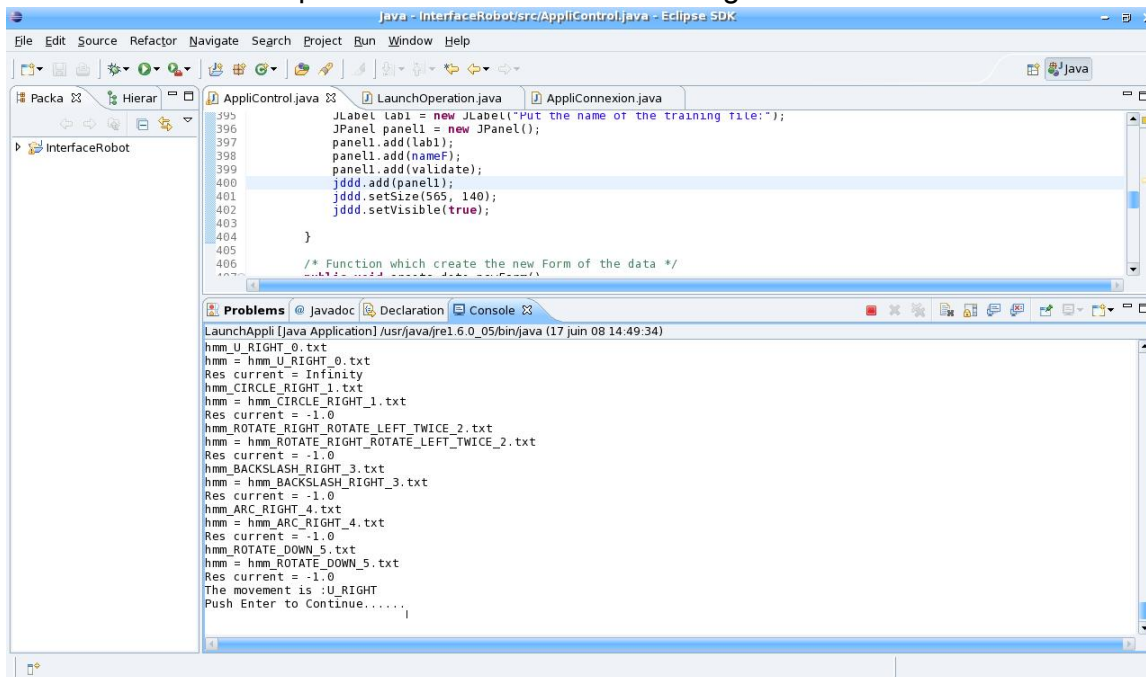


And a window appears for the waiting time:

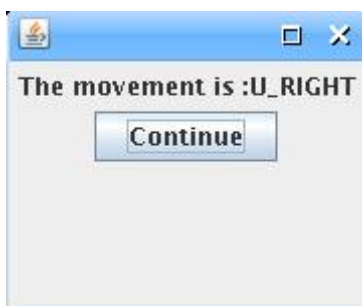


When a file is put in the “Movement” folder that is to say the system has to work on this file. If he recognizes the movement it is written in the console, if it is not recognize the system write that the movement is not trained.

We can see the example when the movement is recognized.

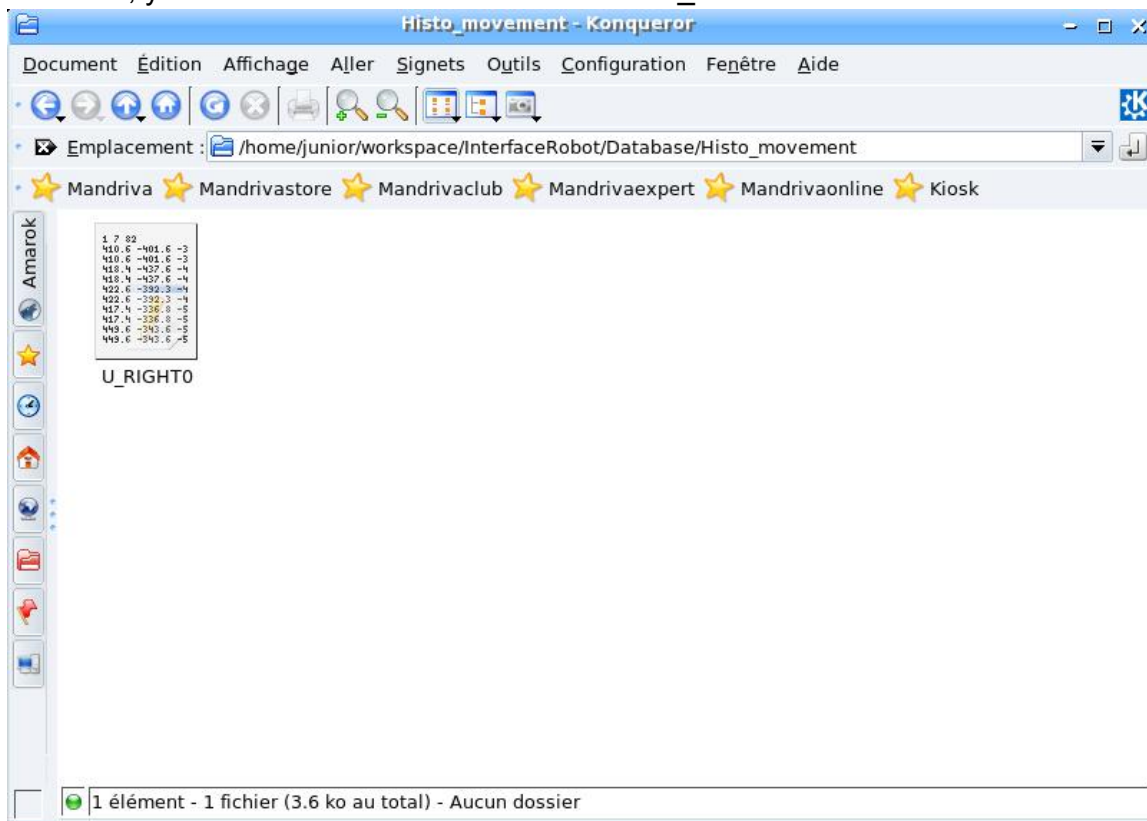


And this window appears at the same time on the screen:

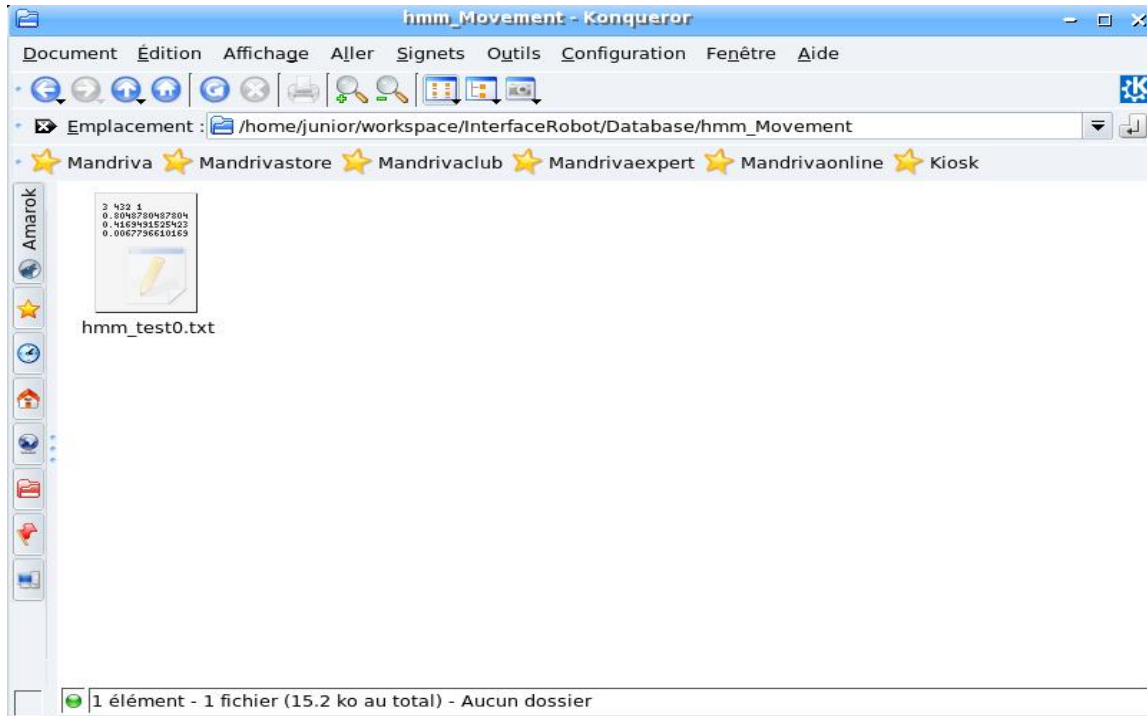




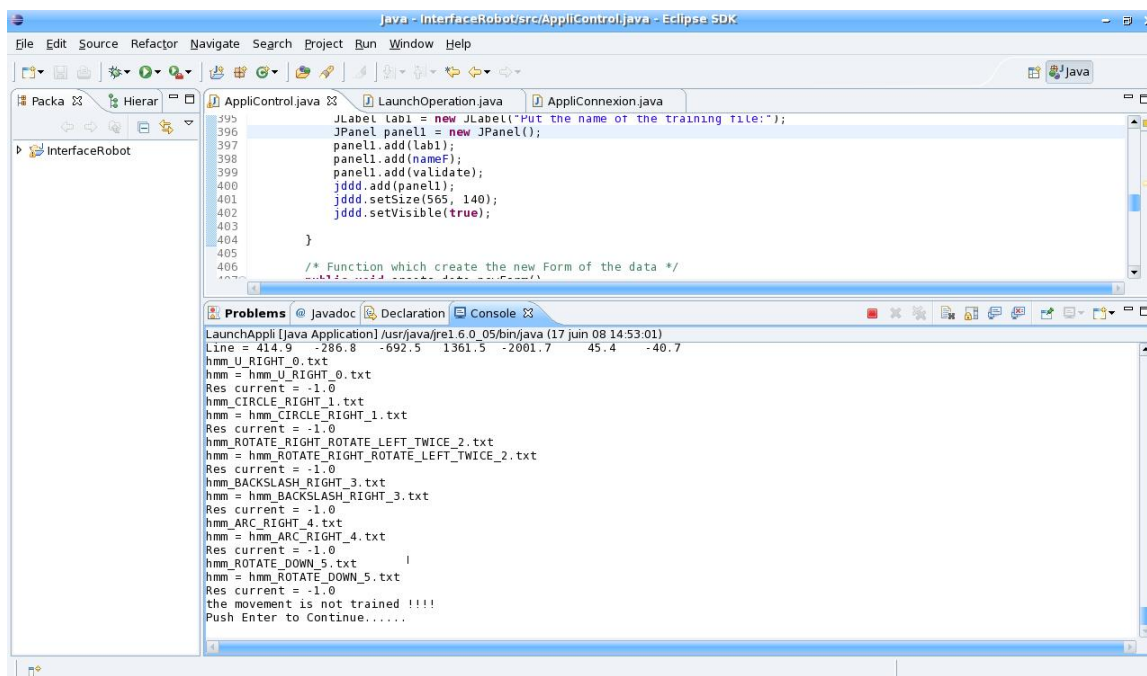
And after, you will find the movement in the “Histo\_movement” folder:



And the HMM in the “hmm\_Movement” folder:



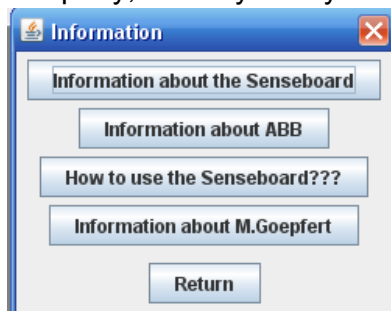
If the movement was not trained, the system put this message in the console:



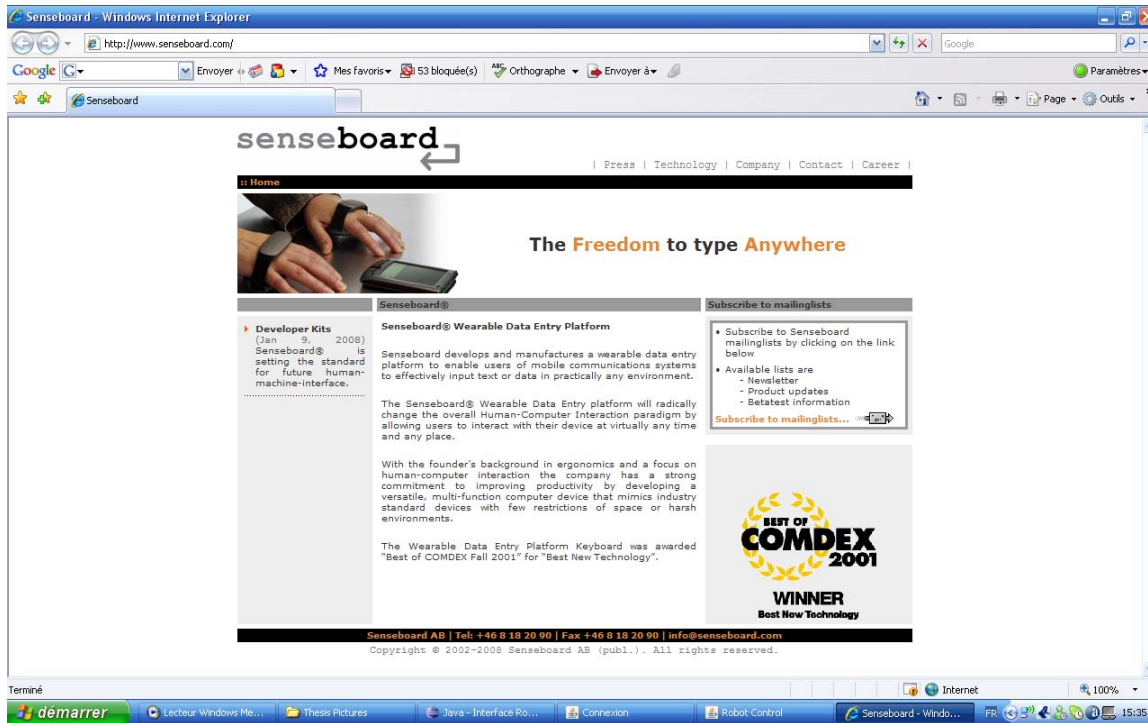
This window appears also on the screen:



If you push the button Info, you can have a lot of information about the Senseboard, the ABB company, the way how you can use the senseboard and about me.



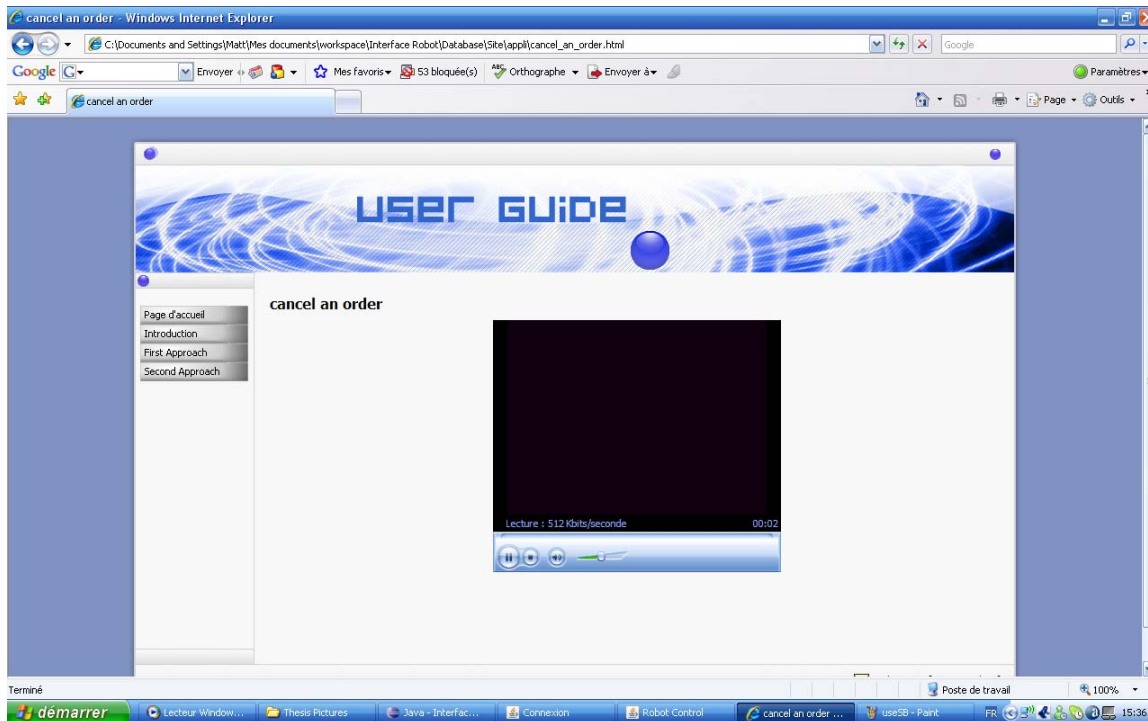
If you push the button "Information about the senseboard", you see the following window:



If you push the button “Information about ABB”, you will see this window:



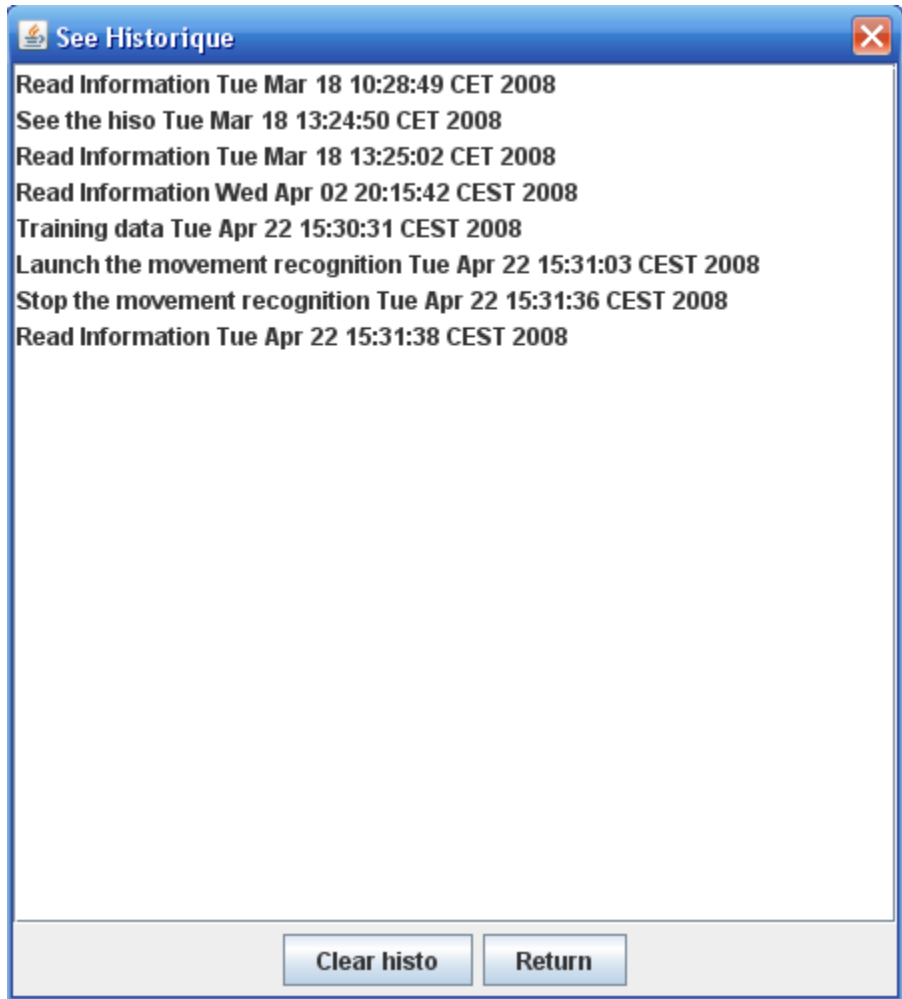
If you push the button “How to use the senseboard???” you can have all the information about the gesture language we design for this application.



If you push the button “Information about M.Goepfert”, you can see information about me:



When you push the button “Historique”, you have a list with all the operation you have done when you use the application.



So all this part is common for all the people, but now we are looking for the part which is only available for the administrator.

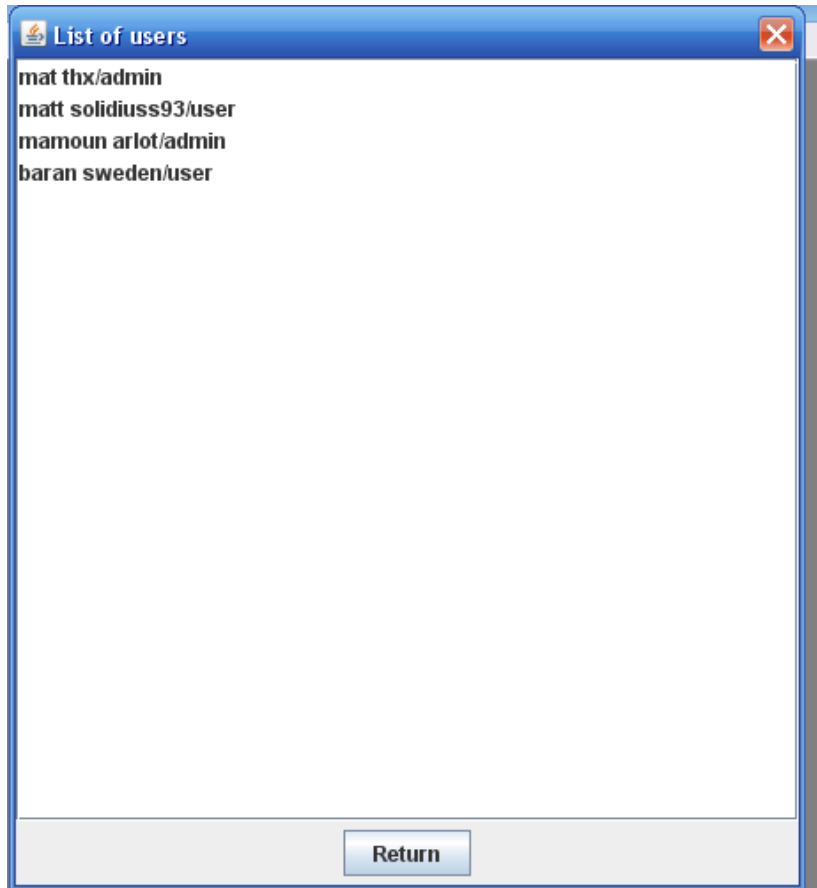
The first thing you can do is add a user for using the application.



The image shows a Windows-style dialog box titled "Add a person". It contains the following elements from top to bottom: a "Login:" label followed by a text input field; a "Choose Password:" label followed by a text input field; a "Confirm Password:" label followed by a text input field; a dropdown menu currently showing "admin"; and two buttons at the bottom labeled "Create User" and "Return".

You can also modify a user:





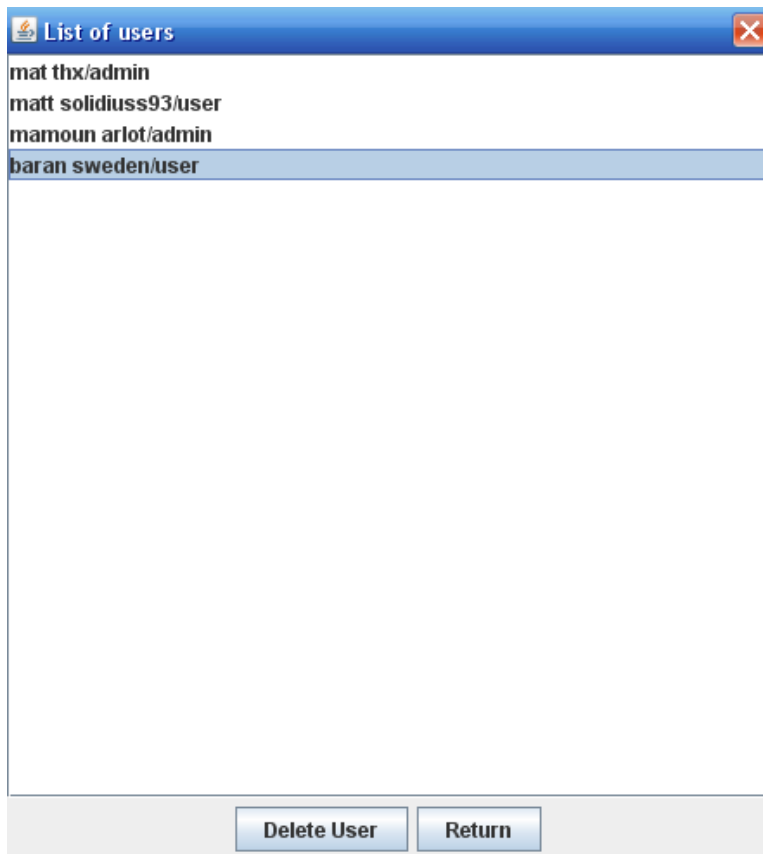
When you click on a name, all the information about the user appears and you can modify them.



The 'Modify a person' dialog box contains the following elements:

- Login:** A text input field containing the name 'baran'.
- Choose Password:** A password input field with seven dots.
- Confirm Password:** A password input field with seven dots.
- Role:** A dropdown menu currently showing 'admin'.
- Buttons:** 'Modify User' and 'Return' buttons at the bottom.

The last thing you can do is to delete a user.

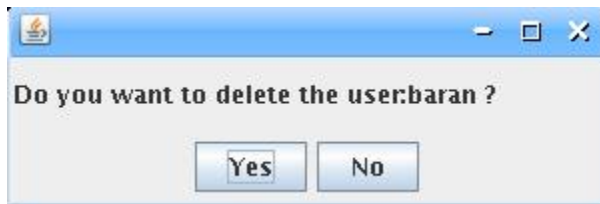


The 'List of users' dialog box displays a list of users with the following details:

Username	Role
mat thx	admin
matt solidiuss93	user
mamoun arlot	admin
baran sweden	user

The 'baran sweden/user' entry is highlighted. At the bottom, there are 'Delete User' and 'Return' buttons.

If you want to delete the user « baran », you have to select it and click on the button.  
So this window appears:



In the next window you can see that I delete the two last user.  
You have:



