Designing an Emotional Basis for Aibo

Miquel Perelló Nieto

January 18, 2013

Contents

Chapter 1

Introduction

In this section we will see the principal objectives of this project. And then a brief environment installation for working with Tekkotsu API.

1.1 Motivation

This project tries to solve the problem of abandoning pets. It wants to create a robotic pet that mimics the real dog behaviours. The ultimate goal is to explain to children how to interpret dogs behaviours. Furthermore give some responsibilities to children taking care of the robotic pet.

One of the typical situations around the abandoned pets is the next. One child wants a pet. Then parents goes to buy this animal for him. After some months the child is bored with the dog and the parents needs to take the responsibility. Furthermore the dog may feel wrong and lonely.

Now, imagine the next situation. When the parent goes to buy the pet, the shop assistant let you this robotic pet. Then the child has to take care of them every day. This is a training for the child, to understand the different behaviours of a dog. The child will know that it needs food, water, to play, and also to rest.

After some time the parents and the child are able to hive some feedback about the experience. Moreover the salesman is able to see a log file indicating the "happiness" or "hungry" on average days.

1.2 Objectives

If the motivation above gives a real and big work, in this project I am going to focus in the cognitive and interactive part. This project aims to start a basis for all explained in motivation section. Therefore I created some internal states to simulate

the excitement level of the robot. Additionally some internal and external actions to interact with this states.

With this two basic concepts it is easy to expand and create all necessary to achieve the entire problem.

As I said, the project is a small part of a big one. The finished project with all the states must be like the figure 1.1. This diagram represents all the perceptions, states, actions and peripherals of the robot (taken from Kismet cognitive architecture [7]).

It have the sensors that are the physical tools to perceive the external world. Then in the perceptual system the robot translate this information and makes an interpretation. This information can be an achieved goal, fear situation or absence of stimulus that can become in a lonely emotion.

With this perceptions the robot will be able to modify his emotions with internal variables. The emotion system at the same time can change the behaviour and the expressions of the robot. It can be in a Toy behaviour, but with a fatigue emotion the expression system will make the velocity of the motors slower.

All this systems are interconnected to create a huge variety of states, and is able to change dynamically.

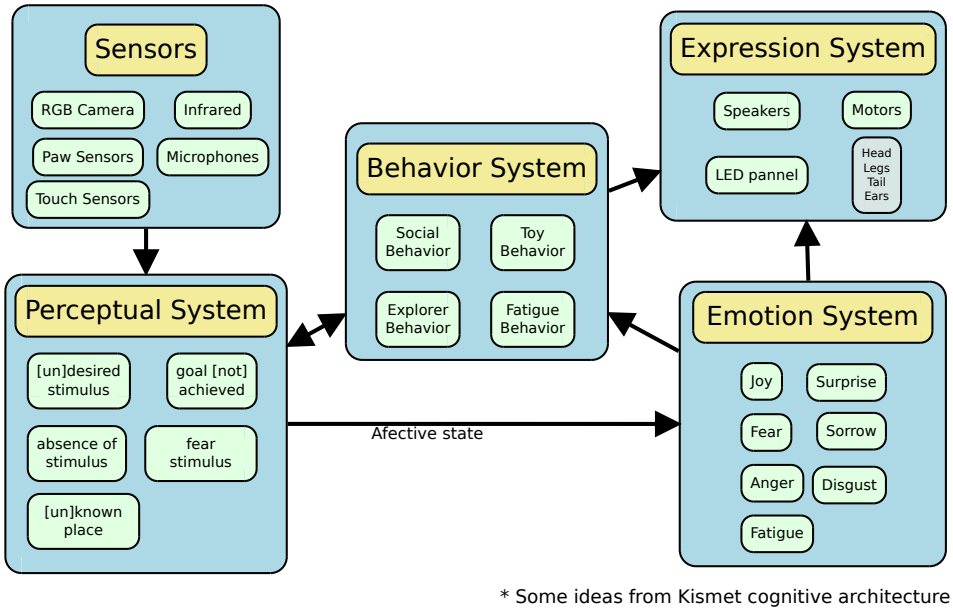

Figure 1.1: Cognitive architecture

1.3 Planning

In this section is possible to see the initial and the final planning. The difference is that at the begining I thought to prepare the environment and start program-

ming was easy. When I started with the environment I had some problems with Tekkotsu and I decided to create two appendix, the Environment installation B and the Freqüent Asked Qüestions C for future projects. For that reason the time for testing the project with children has gone.

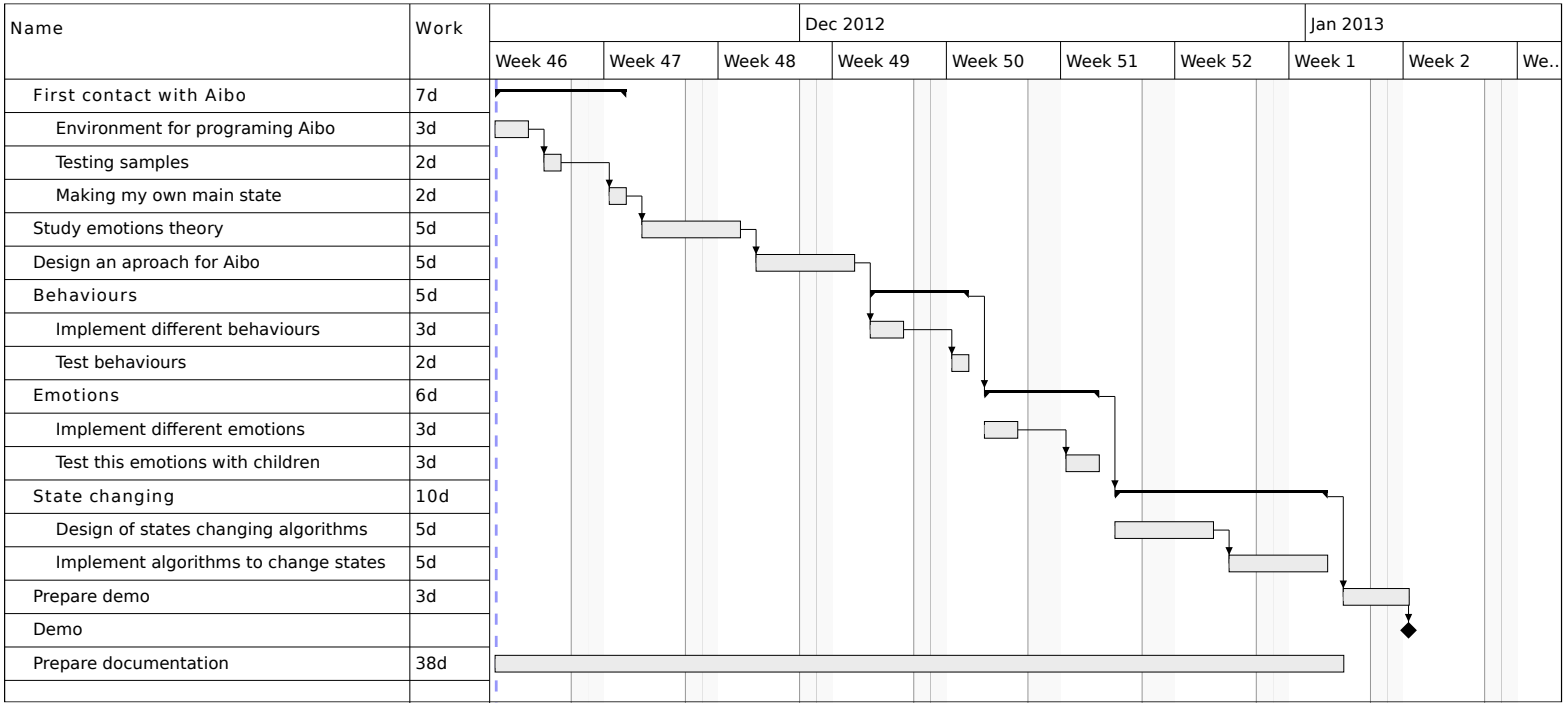

Figure 1.2: InitialGantt

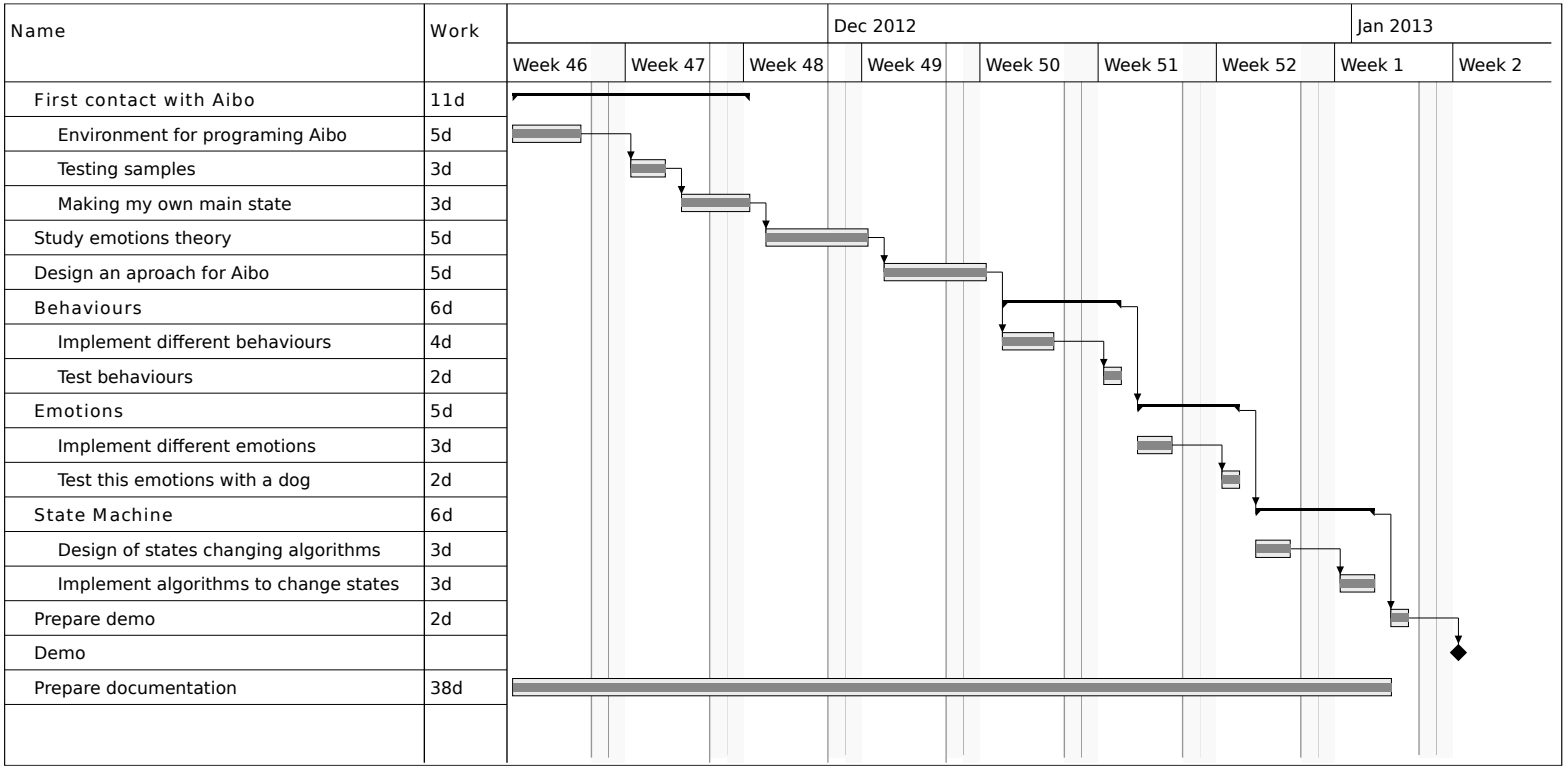

Figure 1.3: FinalGantt

1.4 Requirement Analysis

1.4.1 Platform

The Aibo robot is a mechanoid pet that looks like a baby dog. In the section 1.5 is possible to see a deep description of this platform.

1.4.2 Social Scenario

The scenario must to be the family in a home environment.

1.4.3 User profile target

Child that wants to buy a dog.

1.4.4 Roles

The role of Aibo in this project is a pet.

1.4.5 Modality of Interaction

- Motion
- Touching sensors
- Speaking
- Gestures

1.4.6 Robot Ressources for Interaction

- Camera
- Infrared
- Paw sensors
- microphones
- touch sensors
- head, legs, tall, ears
- LED pannel
- speakers

1.4.7 Metrics to Assess Interaction

- Time in different emotions
- User feedback
- Parents feedback

1.4.8 Quality

- Engagement of the child
- Attitude
- User satisfaction

1.5 Social robot features description

1.5.1 Appearance

The Aibo ERS-7 is a robot with a dog appearance but in a mechanoid way. This is because it does not wants to seem like a real dog, but it can be similar in his actions. It is also baby like.

Figure 1.4: Aibo robot.

1.5.2 Verbal natural cues

- Is able to spoke to a human.
- Potentially can spoke with text to a human.
- Can potentially receive text from human.

1.5.3 Non verbal natural cues

- It have legs, mouth, ears, tail.
- It can look at.
- It can do joint attention.
- The head, legs, tail and ears are mobile.
- It is able to move in a plane.
- It can change his velocity.
- It is able to follow little objects slowly.
- It detects the proximity.

1.5.4 Non-natural cues

- It have lights in the face.
- Buttons in the head, back and paws.
- It can make sounds with a speaker.
- It can show messages to a computer screen.

Chapter 2

Behaviours design

The objective of this project is to create identifiable moods for Aibo robot. Transitions between them are also important for this task. The different behaviours that have been designed and implemented was the sleepy, tired, recently awakened, active and extremely active. It is really a gradient, and I selected ten steps from the quietest to the most agitated position.

Then the behaviours needs different postures to be recognisable. This postures can be seen in the next section.

Apart from this there are some acts and gestures to help. Typical gestures for dogs are waging the tail, yapping and walking. The next section will show which are this gestures and the situations for each.

Finally the last section merges all explained to create a Dynamic State Machine. The nodes are all the postures and gestures and the dynamic transitions between them.

2.1 Emotional Basis

The internal state of the robot controls the behaviour and the movements of it. This internal state is composed by different variable levels. All of them guides the behaviour system to select the appropriated one. These different variables are cited in [2] saying that Ekmann proposed 6 basic emotional states: happiness, anger, sadness, fear, surprise and disgust.

In the mentioned paper they used the Takanishi's model, which is 3-dimensional: pleasant, arousal and confidence. In robot Kismet they use the same approach creating a 3 dimensional space with arousal, valence and

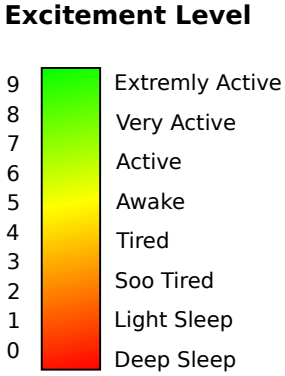

Figure 2.1: Excitment level

closed levels (see figure 2.2 from "Robot Emotion: A Functional Perspective" [8]). But in this project I only use one variable. It is easy scalable to three variables and for the purpose of this project this is enough.

Aibo starts with a zero level of excitement when it is turned on. Every time Aibo receives a touch in the head it increments his excitement level modifying his conduct and creating the new behaviours. The robot internally decreases his excitement level with the time, achieving the deep sleep level in five minutes. You can maintain this level touching his head spontaneously.

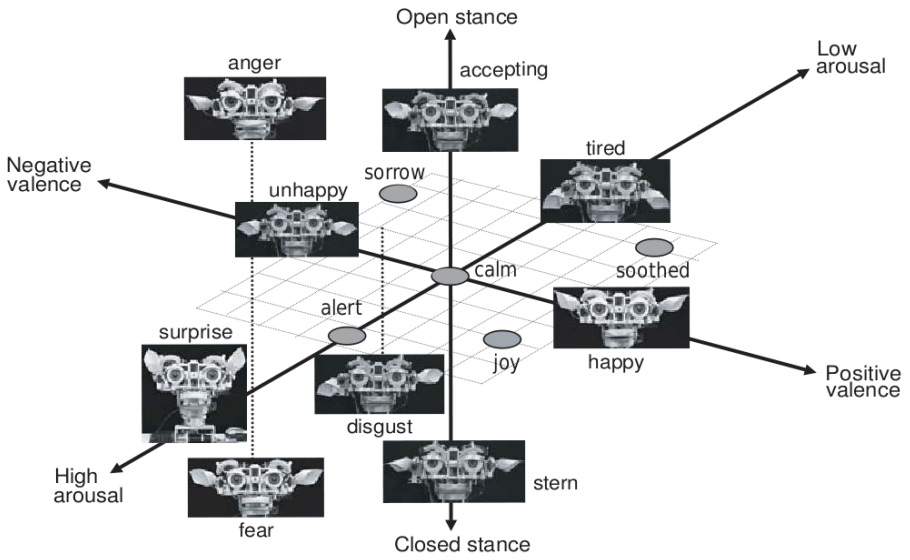

Figure 2.2: Kismet's 3D affect space from C. Breazeal and R. Brooks [8]

2.2 Poses

In this section we will see the different postures that Aibo is able to adopt. This postures will denote different intentions and consequently different behaviours.

2.2.1 Neutral

Neutral pose are typical postures that does not denote any behaviour. For that reason it is used in all the behaviours (except in sleep mode, in which case the robot is always lying). The posture *pounce* is also a very good position between different other postures and helps to intermediate between them (see figures 2.3a,2.3b).

(a) Pounce lateral (b) Pounce front

Figure 2.3: Neutral

2.2.2 Playful

These postures denotes that a dog wants to play. They denote sometimes submission, to enforce the other dog start playing. And other times imposing submission to others. Next figures represents this behaviours.

Lying down is an example of submission pose. It tries to look short provoking the other dog feel stronger and start playing (figures 2.4a, 2.4b).

In the second posture the dog is seated, but it is with the nose pointing up. It is an intermediate posture between submission and strength. It is usually with some barks (figures 2.4c, 2.4d).

The last pose is standing with a strong posture. This tries to show the strength of the dog. It is also a pose to react quickly against any sudden action (figures 2.4e, 2.4f).

(a) Liedown lateral

(b) Liedown front

(c) Playful latteral

(d) Playful front

(e) Stand latteral

(f) Stand front

Figure 2.4: Playful

2.2.3 Sleepy and tired

These postures are used when the dog is sleeping or it is very tired. Both of them are duplicated in left and right side, creating four postures. In this figures we can see only two of theme.

The firsts are when the dog is tired. They maintains the two legs in a position that allows arise faster (figures 2.5a, 2.5b).

The sleepy posture is totally lying down (figures 2.5c, 2.5d).. This pose does not allow to stand up directly. It needs an intermediate posture like tired, or pounce.

(a) Tired lateral

(b) Tired front

(c) Sleep latteral

(d) Sleep front

Figure 2.5: Sleepy

2.3 Gestures

Gestures are very important expressing mood. Apart of the posture the dog maintain. The robot is able to stay at lie down position and denote different moods. These extra gestures helps to interpret the behaviour.

Yap The kind of yaps used in this project are only of playful dogs. And for that reason they denotes playful. On the other side is possible to add angry barks.

Yawn It is denoting that the dog is tired.

Tail wagging If the tail is waging it denotes energy to start playing. On the contrary if the tail keeps quite this position denotes a neutral mood.

2.4 Transitions

The transitions and times between different states are very important. In an excitable behaviour the times between transitions is very short. In the other hand when someone is tired the times and the motions are very slow. For that reason when the robot changes from one state to another all the transitions changes.

In the project have been created three types of transitions.

Fixed Transition These transitions are fixed.

Probability Transition This transitions are probabilities to select one of the nodes. For example when the dog is tired the probabilities to be in a tired or sleep positions are half for each and zero for all the rest.

	move	stand	turn	sit	lie	tired1	tired2	sleep1	sleep2
Deep Sleep	0	0	0	0	0	0	0	0.5	0.5
Light Sleep	0	0	0	0	0	0.25	0.25	0.25	0.25
Soo tired	0	0	0	0	0	0.4	0.4	0.1	0.1
Tired	0	0	0	0.1	0.5	0.2	0.2	0	0
Awake	0	0	0.1	0.4	0.5	0	0	0	0
Active	0.2	0	0.2	0.4	0.2	0	0	0	0
Very Active	0.4	0.2	0.2	0.2	0	0	0	0	0
Extremely active	0.6	0.2	0.2	0	0	0	0	0	0

Table 2.1: Probability transitions table depending on the state.

Random Interval Time This transitions remain waiting in the specified node certain time. The time is in a range and is selected randomly from this. For example, the time between yapping when the robot is very active can be between for to seven seconds. Another example, in sleepy moments the times in one posture are larger.

	pounce	move	stand	turn	sit	lie	tired1	tired2	sleep1	sleep2
DS	0	0	0	0	0	0	0	0	$10 \approx 15$	$10 \approx 15$
LS	1	0	0	0	0	0	$5 \approx 10$	$5 \approx 10$	$5 \approx 10$	$5 \approx 10$
ST	1	0	0	0	0	0	$11 \approx 17$	$11 \approx 17$	$8 \approx 15$	$8 \approx 15$
T	$1 \approx 3$	0	0	0	$2 \approx 7$	$4 \approx 9$	$5 \approx 10$	$5 \approx 10$	0	0
A	$1 \approx 3$	0	0	$1 \approx 2$	$4 \approx 7$	$3 \approx 6$	0	0	0	0
A	$1 \approx 3$	$2 \approx 5$	0	$1 \approx 4$	$4 \approx 7$	$3 \approx 6$	0	0	0	0
VA	$1 \approx 3$	$4 \approx 8$	$2 \approx 6$	$2 \approx 5$	$4 \approx 7$	0	0	0	0	0
EA	$1 \approx 2$	$3 \approx 6$	$3 \approx 5$	$2 \approx 6$	0	0	0	0	0	0

Table 2.2: Random Interval Time transitions : time in seconds remaining in each node depending on the state (the first row is the first letter of the state, it is because the size of the table).

	tail	no tail	yap	yawn
Deep Sleep	$1 \approx 5$	$30 \approx 35$	$120 \approx 240$	$15 \approx 25$
Light Sleep	$3 \approx 5$	$10 \approx 15$	$60 \approx 240$	$10 \approx 30$
Soo tired	$1 \approx 5$	$30 \approx 35$	$120 \approx 240$	$15 \approx 30$
Tired	$1 \approx 5$	$30 \approx 35$	$30 \approx 60$	$30 \approx 60$
Awake	$1 \approx 9$	$4 \approx 9$	$9 \approx 15$	$65 \approx 120$
Active	$1 \approx 9$	$4 \approx 9$	$9 \approx 15$	$65 \approx 120$
Very Active	$1 \approx 9$	$4 \approx 9$	$4 \approx 15$	$120 \approx 600$
Extremely active	$1 \approx 9$	$1 \approx 9$	$2 \approx 8$	$120 \approx 600$

Table 2.3: Random Interval Time transitions : time in seconds remaining in each node depending on the state

Chapter 3

State Machine

Finally with all above exposed. I created a dynamic State Machine. This is dynamic because the probabilities and random interval times are changed dynamically depending on internal and external perceptions.

The State Machine starts creating four parallel instances. These are for yawning, tail wagging, yapping and deciding the posture to adopt. Instead selecting a posture, it can select turning or moving around. If it is moving ahead and there is an obstacle it will turn to avoid them.

The yawn and yap nodes restarts each time they are played. But the time between repetitions is randomly selected in an interval time depending on the aibo's state. For example, in a "soo tired" state, the yawn will repeat randomly every 15 to 30 seconds. The yap will repeat randomly every 2 to 4 minutes. On the opposite side in an "extremely active" state the yawn will repeat randomly every 2 to 10 seconds. The yawn in this state will be repeated every 2 to 5 minutes. It is important to notice that the time in this experiments is accelerated to see these different gestures in a small piece of time.

The same exposed about yawn and yap is made with tail waging, but in this case I needed to define the time waging and not wagging his tail. In active behaviours the time wagging the tail is larger than in tired moments.

About the postures of the robot, it always start with the pounce position. This position allows to reach all positions without collision of their limbs. Once in this position depending on the emotional state it will wait much time or less jumping to the "motion decision" node. This node select automatically the next movement. All the possible movements are weighted depending on the internal sate and sums to one. And one of them is selected randomly.

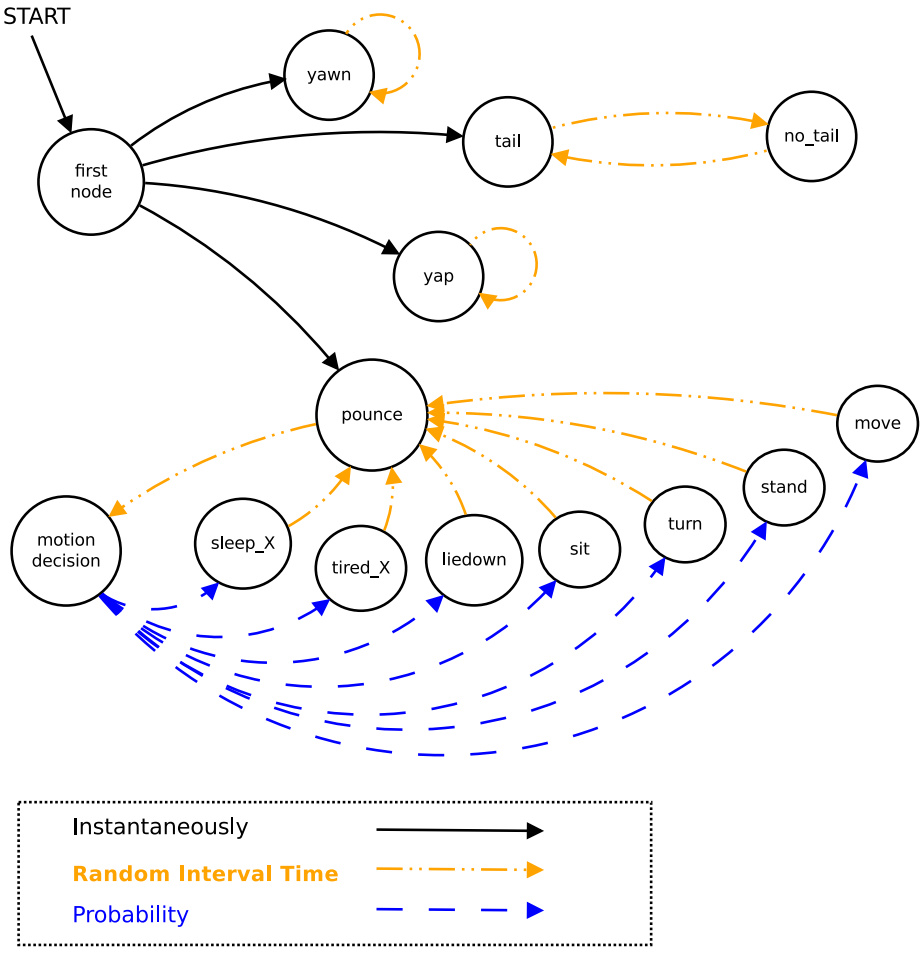

Figure 3.1: State machine with nodes and transitions

Chapter 4

Test

4.1 Research Question

- Is it possible to determine the mod of an Aibo?
- How many time is needed to achieve this task?
- Is there a difference between children and adults?

4.2 Variables

Engagement, attitude, HRI.

4.3 Hypothesis

- It is possible to know which are the robot mood.
- The required time must be about one minute.
- This task must be easy for children and adults.
- Participants
- 20 children, 20 adults, one supervisor.

4.4 Procedure

1. First of all the participant needs to stay some time with Aibo robots. This is to avoid natural behaviors of people when meet with something new and unknown.

2. The robot waits in a certain mood in a room.
3. This moods are:
    - Aibo is playful and wants to play.
    - Aibo is tired and wants to sleep.
    - The Aibo is hungry and thirsty.
4. In the same place there are some tools like food, water, a ball, a bed, and unnecessary objects.
5. The participant have to select one tool for the actual mood.
6. The experiment is iterated three to five times per person.

4.5 Metrics and Instruments

- Time spent thinking, or looking the robot.
- How the participant tries to interact with the robot.
- Correctness of the solution.
- Answers to questionnaire.
- Three cameras in the room and microphones.
- Correlated relations between what Aibo do and the participant responses.
- Distances between participant and different areas.
- Distance between participant and the robot.

4.6 Results

It was not possible to do this test because there was some problems programming the Aibo with Tekkotsu API. But instead of that I was able to do a little experiment to test if the playful behaviour seems like a real little dog (see the appendix A).

Chapter 5

Conclusions

In this project I designed the basis for a bigger project. This bigger project is to make an Aibo dog to look like a real dog. Thereby a human must understand what wants the robot. Because humans and dogs have been together almost 10.000 years ago. Therefore creating a robotic dog able to communicate non-verbally to a human is very useful in a lot of situations.

Then in this project this designed model is not implemented, but it is implemented the basis. One internal variable instead of three mentioned in [2]. One external perception with a button in the head. That modifies the internal states. One internal perception of the time that also changes this internal states. And dynamic transitions between different postures, velocities and gestures that performs a realistic behaviour.

All this transitions and weights are in this moment hard-coded in the program. But the randomness of the decision makes the system so real. But it is possible to define a Reinforcement Learning Algorithm to train this weights and design this transitions automatically. It would be interesting to make this system, but it is for a new project.

It was not possible to carry out the test with humans, but the test is described. However was possible to see that a real dog responded to the Aibo actions like it was a real dog. Then in my opinion I think that the playful behaviour is understandable for people familiarised with dogs. And then it achieves the principal objective of creating a robot like a real dog.

Appendix A

Test two

A.1 Research Question

Can an Aibo behave like a real dog?

A.2 Variables

Engagement, attitude, Animal Robot Interaction.

A.3 Hypothesis

- The dog will react in the same manner than interacting with an other dog.
- Participants
- one dog, one supervisor.

A.4 Procedure

1. The Aibo robot will stay in some place waiting for a dog arrival.
2. The stopped robot will be presented to the dog.
3. Then the Aibo will be turned on.
4. The Aibo will act like a playful dog.

A.5 Metrics and Instruments

- How the dog tries to interact with the robot.
- One camera with microphone.
- Correlated relations between what Aibo do and the dog responses.
- Distances between the dog and different areas.
- Distance between the dog and the robot.

A.6 Results

In this experiment, the Aibo was presented to a dog. The Aibo was running in the Playful behavior. It was possible to see the dog responding correctly to each reaction. In the video the first reaction of the dog was curiosity. It starts looking the robot like another dog (figure A.1a). Then after robot stops the dog approaches to smell it. Then the robot do the stand posture. This posture is typical in the dogs when they wants a submission of the others. Then the dog responds in the same way, adopting the same pose (figure A.1b).

The other position is lying down. This posture means that accepts the submission. This is to provoke the game start (figure A.1c). Both of them makes this position. After that the dog search a toy and let it in the sofa (in the video is possible to think that I put the toy, but I was only positioning the robot in the correct side). At the end of the video the dog and the robot adopts the same lie down position, but the dog with the front legs in the sofa (figure **??**). After that, the robot starts yapping (figure A.1e), and the dog responds with the same action (figure A.1f).

Is possible to see this video of 2:27 minutes in `http://www.youtube.com/watch?v=ERrQ43jVA4s&feature=share&list=LLUb7f6YuMm2dwaMWnNTVfNg`.

(a) Curiosity (b) Stand pose (c) Liedown pose

(d) Liedown in sofa (e) Aibo yaps (f) Dog yaps

Figure A.1: Testing playful behaviour with a dog

Appendix B

Environment Installation

B.1 Prerequisits

sudo apt-get install flex gettext patch bison texinfo make g++

B.2 Open-R

- `http://www.dogsbodynet.com/openr/install_openr_linux.html`
- `http://www.tekkotsu.org/openr-install.html`

You must have gcc-3.4 installed on your computer (OPEN-R is incompatible with gcc-4.1), and will possibly need several other packages.

For debian add this lines into /etc/apt/sources.list:

```
deb      http://snapshot.debian.org/archive/debian/20070730T000000Z/
     lenny main
deb-src http://snapshot.debian.org/archive/debian/20070730T000000Z/
     lenny main
deb      http://snapshot.debian.org/archive/debian-security/20070730
    T000000Z/ lenny/updates main
deb-src http://snapshot.debian.org/archive/debian-security/20070730
    T000000Z/ lenny/updates main
```

Once this is done, do:

```
apt-get update
sudo apt-get install gcc-3.4
```

- build-aibo-toolchain-3.3.6-r3.sh
- gcc-3.3.6.tar.bz2 (23MB, do not unzip, mirrors)

- binutils-2.15.tar.bz2 (11MB, do not unzip, mirrors)
- newlib-1.15.0.tar.gz (10MB, do not unzip, mirror)
- OPEN_R_SDK-1.1.5-r5.tar.gz (12MB, do not unzip)

Installing Sony's Aibo development environment (OPEN-R SDK) Note: we've slightly updated the compiler toolchain as originally used by Sony (originally gcc 3.3.2, newlib 1.10.0, binutils 2.14). We updated the toolchain because the 3.3.2-based one no longer builds under newer distributions due to stricter language adherence in current versions of gcc. If you run into any strange errors and want to try the original toolchain configuration, contact us for a 3.3.2 based build script.

Check prerequisites listed above Download all of the files listed above and place them together into a temporary directory. The build script will install the toolchain into */usr/local/OPEN_R_SDK*. If you prefer a different location, edit build-aibo-toolchain-3.3.6-r3.sh now. If you change the installation directory, you will need to assign the path to the OPENRSDK_ROOT environment variable so Tekkotsu's build scripts will be able to find it later. Run build-aibo-toolchain-3.3.6-r3.sh (and go find some articles to read...)

```
chmod a+x build-aibo-toolchain-3.3.6-r1.sh
export CC=gcc-3.4
sudo ./build-aibo-toolchain-3.3.6-r1.sh
```

You may also need to run the script as root, or otherwise create the OPEN_R_SDK directory and ensure your user account write access. Your target directory (i.e. /usr/local/OPEN_R_SDK) should now contain: OPEN_R/ include/ lib/ mipsel-linux/ bin/ info/ man/ share/

B.2.1 Build a Sample

To test the tools installed properly, try compiling a sample:

```
cd ~/aibo/sample/ers7/BallTrackingHead
make
make install
ls -al MS/OPEN-R/MW/OBJS
```

You should see the following binaries:

- BALLTRCK.BIN
- LFSOUND.BIN
- MVHEAD2.BIN
- MVLEGS2.BIN

- POWERMON.BIN

To compile for the ERS-7 instead, use path " /aibo/sample/ers7/BallTrackingHead7".

B.2.2 Test Sample

To try the sample on AIBO, you'll need a programmable memory stick and reader. Click here for details. Open with a file browser.

```
cp -r /usr/local/OPEN_R_SDK/OPEN_R/MS_ERS7/BASIC/memprot/* /media/
    memstick/
cp -r ~/aibo/sample/ers7/BallTrackingHead7/MS/* /media/memstick/
```

Be sure that the second copy command runs fine, it is possible that the copy command do not rewrite some files and consequently Aibo will turn on, but it won't go correctly.

Eject the memstick from the reader, and insert into AIBO. Boot AIBO and give it a try! AIBO will stretch, then start scanning for the ball. Once found AIBO bleeps and tracks it.

B.3 Tekkotsu

`http://www.tekkotsu.org/downloads.html`

- Recommended: Check out of CVS, see the CVS Usage page for more information.
- Release: Tekkotsu 4.0.1 Release (7.2MB)

B.3.1 CVS version 5.03-CVS

Initial Checkout:

1. Log into the server:

   ```
   cvs -d :pserver:anonymous@cvs.tekkotsu.org:/cvs login
   ```

   This step is not strictly necessary depending on your version of the CVS client -d tells CVS to connect to the server specified in the argument that follows it Use your email address as your password

2. Change to the directory which you would like to contain the Tekkotsu framework

   ```
   cd /usr/local; # or other location of your choosing
   ```

If you choose a location other than /usr/local, you will need to change the TEKKOTSU_ROOT value in Environment.conf after the initial checkout is complete. See the Tekkotsu configuration instructions for assistance.

3. Check out the source:

```
cvs -d :pserver:anonymous@cvs.tekkotsu.org:/cvs checkout -P
    Tekkotsu
```

This will get the latest version of the source, and will be stored in a directory named Tekkotsu. -P will remove ("prune") empty directories which once held files, but are no longer used.

4. Configure your installation (modify file TEKOTSU_ROOT/project/Environment.conf)

```
TEKKOTSU_TARGET_MODEL ?= TGT_ERS7
FILENAME_CASE ?= upper
```

5. Compile the code

```
cd /usr/local/Tekkotsu
sudo make all
```

Appendix C

F.A.Q.

C.1 Can not program

Be sure that the second copy command runs correctly, it is possible that the copy command do not rewrite some files and consequently Aibo will turn on, but it won't run.

C.2 make ftpupdate 553 error

Reported and solved in : `http://tech.groups.yahoo.com/group/tekkotsu_dev/message/1758`

CVS C.0:

```
--- /usr/local/Tekkotsu/project/aperios/Makefile.aperios
2007-11-07 18:27:01.000000000 -0500
+++ ./project/aperios/Makefile.aperios 2007-12-14
    15:08:50.000000000
-0500
@@ -238,6 +238,7 @@

compile-ftpupdate: compile
@echo "Updating the memorystick via ftp using ftpupdate with the
IP adress $(IPADDRESS)"
+ @rm $(MSIMGDIR)/bld_info.txt;
@$(TEKKOTSU_ROOT)/tools/ftpupdate $(IPADDRESS) $(MSIMGDIR);
$
```

C.3 Small movements

Legs moving (small, occasional twitches) is a symptom of noise in the position sensor. The e-stop in 2.3 has been tweaked to reduce smooth its own response to this during e-stop, but there is still some at a hardware level that can't be resolved (which is true for humans too for that matter)

C.4 Ears flapping

The ears flapping is a low battery warning, controlled by BatteryMonitorBehavior, found under "Background Behaviors" : "System Daemons". The flapping will become more frantic (higher frequency) as the battery level drops.

With older batteries (i.e. early 210-era), the battery dies rather quickly at the end, so it doesn't get a chance to ramp up. ers-7 owners should still see a more leisurely progression.

C.5 PostureNode is not defined as a type

In Tekkotsu 4.1 there is an error in the PostureNode.h file.

In the header it says:

Codi C C.1: Header PostureNode.h Tekkotsu 4.1

```
1 #ifndef INCLUDED_TailWagNode_h_
2 #define INCLUDED_TailWagNode_h_
```

When it must be:

Codi C C.2: Header PostureNode.h Tekkotsu 5.1

```
1 #ifndef INCLUDED_PostureNode_h_
2 #define INCLUDED_PostureNode_h_
```

C.6 Tekkotsu files are in lower or upper case

When compiling the Tekkotsu project it will create a directory "ms" or "MS" depending on the configuration. And the Open-R files have the same issue. In the Tekkotsu root exist two scripts that allows to change to uppercase or lowercase all the files and directories you indicate recursively with sub-directories.

Codi Bash C.3: Tekkotsu lower/upper-case

```
#this will convert all files on MS to lower names
TEKKOTSU_ROOT/tools/makelowercase ./MS
#this will convert all files on MS to upper names
TEKKOTSU_ROOT/tools/makeuppercase ./ms
```

And then there is a configuration variable in the Environment.conf file to specify this preference. Concretely this line

Codi Bash C.4: Environment.conf filename_case

```
# in case your memory stick drivers use uppercase, you'll need to
# set 'FILENAME_CASE' to 'upper'
FILENAME_CASE ?= upper
```

Bibliography

[1] Baum, L. E.; Petrie, T. (1966). "Statistical Inference for Probabilistic Functions of Finite State Markov Chains". The Annals of Mathematical Statistics 37 (6): 1554–1563. doi:10.1214/aoms/1177699147. Retrieved 28 November 2011.

[2] Arkin, R., Fujita, M., Takagi, T., Hasegawa, R. (2003). An ethological and emotional basis for human–robot interaction. Robotics and Autonomous ..., 42(3-4), 191–201. doi:10.1016/S0921-8890(02)00375-5

[3] Duffy, B. R. (2003). Anthropomorphism and the social robot. Robotics and Autonomous Systems, 42(3-4), 177–190. doi:10.1016/S0921-8890(02)00374-3

[4] Arbib, M. a, Fellous, J.-M. (2004). Emotions: from brain to robot. Trends in cognitive sciences, 8(12), 554–61. doi:10.1016/j.tics.2004.10.004

[5] Breazeal, C. (2004). Function Meets Style: Insights From Emotion Theory Applied to HRI. IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews), 34(2), 187–194. doi:10.1109/TSMCC.2004.826270

[6] Réant, G, Huggard, M. (2006). "W/BO project - Wi-Fi over AIBO". Final Year project. Mobile Communications. Department of Computer Science. Faculty of Engineering and Systems Sciences (Trinity College, University of Dublin) `http://www.reant.net/tcd/WIBO_Project/pdf/WIBO%20project%20-%20report.pdf`

[7] Vernon, D., Metta, G., Sandini, G. (2007). A Survey of Artificial Cognitive Systems: Implications for the Autonomous Development of Mental Capabilities in Computational Agents.

[8] Breazeal, C., Brooks, R. (2005). Robot emotion: A functional perspective.