Escola Tècnica Superior d'Enginyeria Informàtica
Universitat Politècnica de València

Desarrollo de un caso de estudio para la interacción entre humanos y robots móviles influida por las emociones

Trabajo Fin de Grado

Grado en Ingeniería Informática

Autor: Ros Gil, Jorge

Tutor: Carrascosa Casamayor, Carlos

Co-tutor: Julián Inglada, Vicente

Director Experimental: Rincón Arango, Jaime

2016-2017

Resumen

Este trabajo tiene por objetivo, dotar al robot humanoide NAO de una funcionalidad que permita decodificar estados emocionales en seres humanos, y a partir de ahí realizar una retroalimentación sencilla mediante posturas corporales que se conceptualizan como estados emocionales. Para poder conseguir este objetivo principal, ha sido necesario establecer un marco teórico dentro de la disciplina de psicología social que argumentara y complementara el proceso de implementación del comportamiento del robot. Por otra parte, la pretensión de establecer emociones al robot, aunque sencillas, han tenido como último cometido que estas llegaran a derivar en una conducta de ayuda propiciada a un ser humano. En esta línea, ha sido de obligado cumplimiento recoger el marco teórico de conducta prosocial o de ayuda existente hoy día en la psicología social y más cercano a nuestro cometido.

Palabras clave: robot, humanoide, emociones, psicologia, humanos, interacción.

Abstract

This paper aims to develop a functionality on the humanoid robot NAO to make it able to decode emotional states on human beings, answering to them with corporal postures that can be understood as emotional states. For being able to achieve this goal, it has been required to establish a theoretical guide in social psychology which could complement the implementation process of the behaviors of the robot. On the other hand, to provide NAO with emotions, has had the goal of making the robot acquire a help behavior towards humans. To achieve this, it has been required to gather theoretical aspects on social psychology and prosocial behavior.

Keywords : robot, humanoid, emotions, psychology, humans, interaction.

Agradecimientos

A los tutores de este trabajo por tener el interés de adentrarse en la investigación en este campo, tan interesante y complejo, ya que siempre he tenido un gran acercamiento a él. Y por haber mantenido una actitud abierta a las propuestas que he tenido durante el desarrollo de este trabajo. La tolerancia y apertura de miras de los tutores, ha hecho que profundice con seguridad en el campo de la disciplina de psicología social.

A mi madre, Dra. en Psicología, que me ha transmitido desde pequeño la belleza del comportamiento humano y la ciencia que detrás esconde. Su afán por la psicología social aplicada a la salud ha hecho que me sintiera seguro al acometer este trabajo con la disciplina de psicología.

Y a mi padre, que junto a mi madre, han soportado los momentos de frustración pero también las alegrías que he pasado durante el grado.

Tabla de contenidos

Contenido

1. Introducción

Las nuevas tecnologías, pretenden introducir en la vida cotidiana, formas de hacer esta mucho más sencilla y agradable. En la actualidad podemos encontrar un asistente personal en nuestro móvil, nuestro vehículo y otros lugares; que nos ayuda a recordar eventos importantes, o a encontrar la heladería más cercana al lugar en que nos encontramos. Al llegar a casa, esta también puede ser inteligente siendo capaz de encender la calefacción una hora antes de nuestra llegada, o puede incluso habernos enviado a nuestro móvil la lista de la compra, gracias también a la existencia de neveras inteligentes.

Todos estos avances tecnológicos están al alcance de nuestra mano, pero tienen un inconveniente. Tratan de hacer nuestra vida más sencilla, intentando en algunos casos como los asistentes personales, imitar el comportamiento humano mediante la interacción con la voz. Sin embargo, no transmiten la sensación de estar interactuando con un ser humano, de ser comprendido. Todo lo contrario, escuchar una voz robótica o unas respuestas predefinidas a ciertas preguntas, alejan la sensación de humanidad.

Es por ello que este trabajo de fin de grado intenta realizar un caso de estudio en el que un robot es capaz de interactuar con el usuario utilizando emociones. No solo es capaz de detectar la emoción del humano si no que puede adoptar una postura corporal que transmite esta emoción, dando la sensación a la persona de que es comprendida y que el robot tiene un aspecto humano. Uno de los objetivos de este trabajo por lo tanto es dotar de capacidad de detección e imitación de emociones a un robot humanoide.

Otro de los objetivos de este proyecto es realizar un acercamiento claro de la ingeniería informática a la psicología mediante la introducción de teorías puramente psicológicas sobre emociones, permitiendo así una comprensión más clara de lo que se intenta conseguir con este trabajo y creando una referencia para futuros trabajos en esta materia.

1.1 Objetivos

1. Desarrollar un caso de estudio en el que un robot humanoide sea capaz de interaccionar con un ser humano, detectando emociones a través del canal no verbal.

2. Dotar de reacciones al robot relacionadas con las emociones detectadas en la persona con la que está interactuando.
3. Obtener el perfil psicológico del usuario mediante un test de personalidad, cuya formulación de ítems sea emitida por el robot.
4. Introducir de forma pormenorizada la disciplina de psicología social en la ingeniería informática y más concretamente, en el campo de la interacción persona-computador.
5. Analizar los marcos teóricos de psicología social que podrían ser implementados para crear un proceso de interacción-ayuda entre personas y robots.
6. Crear una base teórica y experimental a partir del punto anterior para trabajos futuros que logren desarrollar la interacción persona-robot utilizando emociones.

2. Trabajo relacionado

Para introducirnos en los conceptos ya establecidos relacionados con el trabajo, como son la robótica, la inteligencia artificial, la psicología, etc., vamos a hacer un repaso describiendo estas disciplinas, y aspectos que se han considerado importantes y necesarios incluir para poder comprender la importancia de las mismas para este proyecto.

Primero, hablaremos sobre la robótica, campo estrechamente relacionado con este trabajo, ya que la plataforma sobre la que se ha desarrollado este, es un robot humanoide que cuenta con inteligencia artificial, de la cual también hablaremos.

Posteriormente, introduciremos conceptos relativos a la emoción, y lo que es más importante; como puede ser detectada por un robot o programa informático y a la vez, como podrían ser simuladas por un robot humanoide.

2.1 Robótica e Inteligencia Artificial

Como se ha comentado en los párrafos anteriores, en este apartado se trata la situación actual de la robótica y la inteligencia artificial, viendo también antecedentes históricos de estos dos campos y sus perspectivas de futuro. Es importante situarse en estos dos temas, ya que durante todo este trabajo se van a tratar diversos aspectos de IA y robótica, centrándonos sobre todo en inteligencia artificial relativa a robots humanoides y cómo estos pueden ser útiles en la vida de las personas al introducir emociones en su comportamiento.

2.1.1 Robótica: Breve historia

Desde la antigüedad, el ser humano ha perseguido el objetivo de construir máquinas que fueran capaces de imitar movimientos humanos o realizar tareas demasiado duras, complejas o repetitivas; pudiendo así evitar realizarlas. Del griego antiguo, proviene la palabra autómata que según el diccionario de la RAE significa:

autómata. Del pl. lat. *automăta,* y este del pl. gr. αὐτόματα *autómata*

1.*«Instrumento o aparato que encierra dentro de sí el mecanismo que le imprime determinados movimientos.»*

2.*«Máquina que imita la figura y los movimientos de un ser animado.»*

En un principio, el interés por los autómatas se redujo a la segunda definición del término, ─máquinas que intentaban imitar la forma y movimientos de seres vivos─. También en el renacimiento, continúa este interés hacia los autómatas, que eran utilizados como atracciones en ferias.

Fuera del interés lúdico, los árabes crearon dispositivos que introdujeron en la vida cotidiana, relacionados mayoritariamente con el agua. A finales del siglo XVIII y principios del XIX, empiezan a crearse máquinas mecánicas útiles para la industria; como telares mecánicos, hiladores, etc. Este momento, da paso a la automatización industrial, con el uso de mecanismos capaces de acelerar el trabajo repetitivo o peligroso que antes se realizaban a mano.

El término **robot** aparece por primera vez en 1921, en una obra del escritor checo Karel Capek. Proveniente de la palabra eslava *robota*, referente al trabajo que se realiza de manera forzada (Barrientos, 2007).

Quienes realmente popularizan el término, son los escritores de ciencia ficción del siglo XX que lo utilizan con el mismo mensaje que utilizó Capek en su obra, la sumisión de la especie humana por seres mecánicos hechos a su semejanza.

Sin lugar a dudas, fue Isaac Asimov quien realmente popularizó la palabra **robot** con sus conocidas como las "*tres leyes de la robótica"*:

1. Un robot no puede perjudicar a un ser humano, ni con su inacción permitir que un ser humano sufra daño.
2. Un robot ha de obedecer las órdenes recibidas de un ser humano, excepto si tales órdenes entran en conflicto con la primera ley.
3. Un robot debe proteger su existencia mientras tal protección no entre en conflicto con la primera o segunda ley.

La ciencia ficción, como en muchos otros aspectos de la ciencia en general, además de la robótica, ha contribuido al establecimiento de las ideas que han permitido avances significativos para la humanidad.

Si bien es cierto que el término robot suele asociarse a una máquina con forma humanoide que imita movimientos y comportamientos del ser humano, el desarrollo de la robótica se enfocó más al ámbito industrial durante el siglo XX, en el que la eficiencia en la producción fue un objetivo crucial para el desarrollo económico.

Los primeros desarrollos, eran mecanismos puramente mecánicos que permitían la manipulación de elementos peligrosos desde una distancia segura. Mediante una

técnica *maestro-esclavo* en la que el operador manipulaba el elemento maestro, se conseguía que el elemento esclavo reprodujese los movimientos que el operador del **telemanipulador** realizaba. Pocos años más tarde, estos telemanipuladores incorporaron elementos eléctricos que sustituyeron a los mecánicos.

Estos mecanismos quedaron relegados a un mercado más limitado (como la industria nuclear), y fueron sustituidos por lo que hoy conocemos como robot. Un robot, elimina la necesidad de que sea el operario quien manipule el mando del mecanismo, añadiendo para ello un computador que se encarga del manejo.

Desde la introducción del primer robot industrial controlado por computador hasta nuestros días, se han desarrollado diferentes tecnologías y usos para los robots. En el siguiente punto, se discutirán los tipos de robots que podemos encontrar en la actualidad y los usos que tienen.

2.1.2 Robots según sus características

2.1.2.1 Robots manipuladores industriales

Un robot manipulador industrial por su definición en la ISO 8373:

robot manipulador industrial.

> *«un manipulador programable en tres o más ejes, controlado automáticamente, reprogramable y multifuncional, que puede estar fijo en un lugar o móvil para uso en aplicaciones automáticas de la industria»*

Robots ABB en una cadena de producción de automóviles

De esta definición extraemos que un robot industrial, permite manipular objetos o realizar diversas tareas tras su programación, y que este puede encontrarse sujeto sobre una superficie o por el contrario moverse sobre ruedas, railes, etc. Uno de los aspectos a destacar en esta definición es la necesidad de que el robot, para que sea considerado como tal, debe tener tres o más ejes; es decir, grados de libertad sobre los que puede realizar movimientos.

Un robot manipulador industrial, está basado en cierta forma en la anatomía del brazo humano. De hecho, se dice que un robot de este tipo está formado por articulaciones. Un número de articulaciones forman el brazo y el resto la muñeca.

Existen diversas configuraciones de robot industrial, dependiendo de la forma en que está diseñado su brazo y su muñeca. Cada una de estas configuraciones es útil para determinado tipo de aplicaciones, dependiendo de la precisión que se requiera, la velocidad de realización de la tarea, etc. También, dependiendo de la tarea que debe realizar el robot, se puede cambiar la herramienta con la que cuenta la muñeca. Por ejemplo, un soldador o unas pinzas.

Un robot industrial, puede haber sido programado con anterioridad y realizar la tarea en bucle o, por el contrario, ser manejado por un operario mediante un mando.

2.1.2.2 Robots móviles

Un robot móvil es un vehículo autónomo que no se mueve por raíles, en su lugar utiliza ruedas, piernas, cadenas, etc. Suelen estar alimentados por baterías, ya que no pueden estar conectados mediante ningún cable.

iRobot Roomba es un ejemplo de robot móvil

Al ser vehículos autónomos necesitan sensores que les permitan guiarse, como GPS. Pueden contar también con una ruta predefinida. Sean completamente autónomos o con una ruta a seguir, necesitan sensores que les permitan evitar colisiones o caídas. Estos son los sensores extereoceptivos, que aportan a un robot móvil información sobre su entorno.

Además de estos, un robot móvil requiere sensores propioceptivos que le aporten información sobre su estado. Estos pueden ser, el nivel de batería o las posiciones de los *encoders* de las ruedas.

Los robots humanoides pertenecen a un subconjunto de los robots móviles y se hablará sobre ellos en el siguiente punto.

2.1.2.3 Robots humanoides

Un robot humanoide es un caso particular de robot móvil. Imita la anatomía humana y por consiguiente sus movimientos. Cuenta con dos brazos, una cabeza y un torso. En muchos casos cuenta también con dos piernas, que pueden ser sustituidas por una base con ruedas (Russell y Norvig, 2004).

El gran reto ingenieril en este tipo de robots, es su estabilidad al andar sobre dos piernas. Proveer a un robot de la capacidad de andar imitando la forma en que lo hacen los humanos, es una tarea compleja, debido a la sincronización necesaria de numerosos motores y articulaciones que permitan mover el peso del robot pudiendo así equilibrarlo.

Asimo, robot humanoide de Honda

La empresa japonesa Honda ha hecho grandes avances en este aspecto con su robot **Asimo**. Disminuir su altura y peso en las diferentes versiones, les ha permitido hacer que el robot sea más estable y por lo tanto más autónomo y ágil. Con 130cm de altura y 50kg de peso en su última versión, Asimo es capaz de andar a 2.7km/h y correr a 7km/h, además de subir y bajar escaleras con cierta facilidad.

Aunque un modelo mucho más simple que el de Honda, NAO, el robot utilizado para este proyecto, es un robot humanoide. En el capítulo de **herramientas utilizadas** de esta memoria se describirán con detalle sus características software y hardware, concretamente en el apartado 3.1.

2.1.2 Inteligencia Artificial

Desde que el ser humano comprende que cuenta con unas capacidades mentales asombrosas y que es capaz de percibir y entender su mundo, persigue conocer cómo llevar esto a cabo.

La inteligencia artificial, intenta simular en cierto modo la forma en la que el ser humano comprende su entorno y actúa sobre él. Existen diversas definiciones dependiendo de a qué se refieren. A continuación, se muestran definiciones recogidas en libro Inteligencia artificial, un enfoque moderno (Russell y Norvig, 2004) por distintos autores. Entre las que se refieren a *procesos mentales y razonamiento,* están:

2.1.2.a Sistemas que piensan como humanos:

«[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje...» (Bellman, 1978)

2.1.2.b Sistemas que piensan racionalmente:

≪El estudio de las facultades mentales mediante el uso de modelos computacionales≫. (Charniak y McDermott, 1985)

Otras definiciones recogidas por Russell y Norvig se refieren a la inteligencia artificial teniendo en cuenta la conducta:

2.1.2.c Sistemas que actúan como humanos:

≪El arte de desarrollar maquinas con capacidad para realizar funciones que cuando son realizadas por personas requieren de inteligencia≫. (Kurzwell, 1990)

Diversas disciplinas de la ciencia aportan ideas y contribuyen al desarrollo de la inteligencia artificial. Puede pensarse que la IA solo depende de teorías computacionales que permiten su desarrollo, pero disciplinas como la filosofía con su estudio del conocimiento son importantes en el desarrollo de toda la teoría en este campo.

La **filosofía,** permite delinear las ideas clave de la IA, pero es necesario formalizar todas estas ideas mediante una formulación matemática. Las tres áreas de las **matemáticas** más utilizadas para este propósito son: lógica, computación y probabilidad.

Aunque no se va a entrar a discutir la importancia de estos tres campos de las matemáticas en la IA de manera exhaustiva, sí debe destacarse la importancia de teorías como por ejemplo la de la **NP-completitud**, que permite reconocer si un problema es o no tratable. También la **teoría de la probabilidad** es inmensamente utilizada en aplicaciones de aprendizaje automático y percepción, por ejemplo. La **lógica proposicional**, es utilizada en aspectos de ingeniería del conocimiento, razonamiento, sistemas expertos, etc.

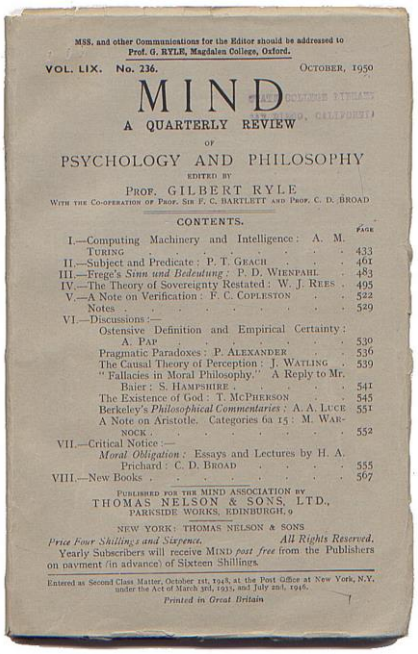
El campo de la **economía,** aporta a la IA teorías como la de la decisión, que utiliza la teoría de la **probabilidad** y la de la **utilidad**; esta última conocida como la del «beneficio deseado». Otro amplio campo utilizado en IA es la **teoría de juegos**, que se basa en un problema de toma de decisiones con incertidumbre. La incertidumbre viene dada por las decisiones de otros jugadores o el azar.

La **neurociencia,** estudia el sistema neurológico. Existe un gran interés en este campo, ya que puede explicar cómo se consigue el razonamiento y de esta manera intentar imitar el comportamiento de nuestro cerebro, trasladándolo a un computador que intente simularlo.

Otra ciencia que interviene de manera importante en la IA es la **psicología,** gracias a su estudio del comportamiento humano y del razonamiento. De esta ciencia, se hablará en capítulos posteriores más extensamente por su implicación en estudios sobre emociones y comportamiento humano.

Un campo de la ciencia muy importante para la IA es la **lingüística**. Es la encargada del estudio del aprendizaje del lenguaje. Todas estas teorías son utilizadas para el reconocimiento del lenguaje natural en inteligencia artificial (Russell y Norvig, 2004).

2.1.3 Test de Turing


MSS. and other Communications for the Editor should be addressed to
Prof. G. RYLE, Magdalen College, Oxford.

VOL. LIX. No. 236. OCTOBER, 1950

MIND

A QUARTERLY REVIEW

OF

PSYCHOLOGY AND PHILOSOPHY

EDITED BY

PROF. GILBERT RYLE

WITH THE CO-OPERATION OF PROF. SIR F. C. BARTLETT AND PROF. C. D. BROAD

CONTENTS.

PUBLISHED FOR THE MIND ASSOCIATION BY
THOMAS NELSON & SONS, LTD.,
PARKSIDE WORKS, EDINBURGH, 9

NEW YORK: THOMAS NELSON & SONS

Price Four Shillings and Sixpence.
Yearly Subscribers will receive MIND *post free* from the Publishers on payment (in advance) of Sixteen Shillings.

Entered as Second Class Matter, October 1st, 1928, at the Post Office at New York, N.Y. under the Act of March 3rd, 1933, and July 2nd, 1946.

Printed in Great Britain

Nº 236 de la revista Mind en el que Turing publica su artículo

En enero de 1950, el científico de la computación Alan Turing, publica un artículo científico en la revista *Mind* con el título "***Computing Machinery and Intelligence***". En él Turing, propone el que posteriormente se conoció como el Test de Turing.

Ya en la primera línea de su artículo, el autor propone la pregunta: "¿Pueden pensar las máquinas?". Debido a la ambigüedad que puede suponer el significado de las palabras **pensar** y **máquina**, Turing decide plantear un juego al que denomina "**juego de la imitación**", que permita responder a su pregunta. La prueba fue diseñada para conseguir un método que permitiera definir la inteligencia de una manera operacional (Warwick y Shah, 2015).

Mediante este juego, Turing evita proporcionar una serie de requisitos y cualidades que una máquina debe cumplir para ser considerada inteligente artificialmente (Russell y Norvig, 2004).

En este juego existen dos actores, la máquina (A) y el interrogador (B). El objetivo de B, tras un conjunto de preguntas realizadas a A, consiste en determinar si este es un ser humano, o por el contrario, se trata de una máquina. El objetivo de A será por lo tanto evitar que B descubra que es una máquina. El computador habrá superado la prueba si consigue que el interrogador no sea capaz de distinguir si las respuestas proporcionadas, son de una persona o no. Más concretamente, el test se pasa si el interrogador confunde al computador con un humano el 30% de las veces.

En la actualidad, existen diversos proyectos cuyo objetivo es construir una IA capaz de pasar el Test de Turing. Algunos de estos proyectos, han sido capaces de pasar el test. Es el caso del programa llamado Eugene Goostman, que simula un chico ucraniano de 13 años. El 7 de junio de 2014, fue capaz de convencer en un 33% de las ocasiones al interrogador de que era un humano, superando así el test (Warwick y Shah, 2015).

Si bien es cierto que parece que el test está empezando a poder ser superado con la tecnología actual, muchos científicos y expertos en la materia se cuestionan si

realmente superar el Test de Turing, significa contar con inteligencia artificial. Esto es debido a que un programa o computador diseñado específicamente para la tarea de superar la prueba, puede contar con las habilidades necesarias para ello, pero no con otras que pueden ser necesarias para ser considerado inteligente. Warwick y Huma (2015), consideran en su artículo **"Passing the Turing test does not mean the end of humanity"** que "el Test de Turing no es más que un simple test de la habilidad de comunicación de una máquina".

Diversos autores consideran que el test ha quedado obsoleto para la tecnología y objetivos que hoy se tienen. Por este motivo, existe un gran debate sobre qué podemos considerar inteligente y qué no.

2.2 Psicología de las emociones

En este trabajo, en el que se realiza un caso de estudio donde un robot cuenta con capacidad para detectar emociones; lo que le permite interactuar con el usuario de una manera más cercana, se ha creído necesario hacer una introducción al concepto de emoción desde el punto de vista de la psicología.

Si bien es cierto que este trabajo de fin de grado corresponde al grado en Ingeniería Informática, existen pocos ejemplos de mayor confluencia de disciplinas científicas que la que se produce en el ámbito de la robótica. Es por ello, que una aproximación completamente psicológica al concepto de la emoción se ha considerado muy útil para situarnos en contexto y llegar a determinar las posibilidades que tenemos hoy en día de incluir emociones en un robot humanoide.

En primer lugar, definiremos el concepto de emoción, viendo que es uno de los campos de la psicología más caóticos debido al gran número de teorías que existen al respecto, llegando al extremo de contradecirse entre sí. Después, pasaremos a exponer distintos estudios que han contribuido a la clasificación y detección de emociones a partir de la expresión facial.

2.2.1 Definición de emoción

Si nos encontráramos en una conversación cotidiana, con nuestra familia o amigos y en ella surgiera la palabra “emoción”, todos los participantes en ella entenderían de qué se está hablando. ¿Pero qué ocurriría si pidiéramos que alguno de ellos nos diera una definición? Muy probablemente, esta persona sería incapaz de hacerlo. Wenger y cols. (1962), ya expresaron: «”Emoción” es una extraña palabra. Casi todo el mundo piensa que entiende lo que significa hasta que intenta definirla. Es entonces cuando prácticamente nadie afirma ya entenderla.”»

Esto es lo que ocurre en psicología. El término “emoción”, cuenta con numerosas definiciones que se basan en diferentes conceptos. Esto puede ser debido a la complejidad del propio término de estudio y la gran variedad de metodologías que son utilizadas para la investigación en este campo.

En lo que sí coinciden muchos de los investigadores que se han dedicado a esta materia es que la emoción, podría caracterizarse por la aparición de tres reacciones o sistemas de respuesta. En este sentido, en una reacción emocional aparecen tres reacciones simultáneamente: vivencia, comportamiento y reacción fisiológica. Por otro lado, están involucrados los sistemas: cognitivo/subjetivo, conductual/expresivo y fisiológico/adaptativo (Schmidt-Atzert, 1985).

Para comprender esto podemos utilizar el ejemplo que pone Schmidt-Atzert (1985) en su libro **Psicología de las emociones**:

> Imaginemos que alguien es amenazado por una banda de maleantes armados. Su reacción es: huye (comportamiento), su corazón late más deprisa (reacción fisiológica) y siente miedo (vivencia subjetiva). Sería posible medir esta reacción mediante observación del comportamiento, mediciones fisiológicas y describiendo verbalmente los correspondientes sentimientos.

Estas tres reacciones que se han comentado, y que son producidas como consecuencia de un estímulo o situación dada que desencadena una emoción, es lo que en psicología se conoce como la **tríada reactiva**. No obstante, aunque el término "tríada", puede llevarnos a pensar que todos sus componentes están relacionados, algunos estudiosos del tema como Izard (1977), afirman que los tres modos de reacción, son considerados como indicadores de emociones, pero no cabe esperar que, en todas las situaciones semejantes, aparezcan las tres reacciones.

Como es evidente, al menos por el momento, un robot no puede contar con emociones, no puede tener una reacción fisiológica ya que no se trata de un organismo vivo, no puede sentir miedo, alegría, etc. Lo que sí es posible es simular estas tres reacciones. No hay problema en simular un comportamiento en un robot siempre y cuando su arquitectura y morfología nos lo permita. El ejemplo anterior puede ser perfectamente simulado. Frente a un estímulo, el robot podría correr intentando huir. Para simular la reacción fisiológica, si este cuenta con un sistema de audio como lo hace NAO, podría reproducirse un sonido de respiración acelerada o incluso de latidos de corazón. Y, por último, podríamos simular un comportamiento no verbal que pudiese ser descodificado por un receptor como estado de miedo.

Esta pequeña introducción al concepto de emoción, permite comprender que es necesaria la colaboración entre robótica y psicología para poder crear robots que en el futuro puedan contar con emociones y detectarlas en los sujetos con los que interactúan.

2.2.2 Funciones de las emociones

Tras haber definido el concepto de emoción y destacado su arbitrariedad conceptual, podemos pasar a estudiar la utilidad de esta, y el cometido que creen los psicólogos que tiene la emoción en los seres humanos.

Ninguna emoción se produce sin motivo. Todas las emociones cuentan con alguna función que permite, por ejemplo, tener reacciones conductuales apropiadas con respecto al estímulo o situación que las ha provocado (Chóliz, 2005). Este autor argumenta que no solo tienen una función las emociones agradables, sino también las consideradas desagradables. Reeve (1994), realizó una clasificación de las funciones principales de la emoción:

- Funciones adaptativas
- Funciones sociales
- Funciones motivacionales

2.2.2.1 Funciones adaptativas

Darwin (1872-1984) fue el primero en argumentar que las emociones permiten tomar una conducta apropiada hacia el estímulo que la ha provocado, preparando así al organismo para permitir adaptarse de manera adecuada a las condiciones ambientales.

Plutchik (1980), establece una correspondencia entre ocho emociones muy comunes y su función, reflejada en el siguiente cuadro:

Emoción	Función
Miedo	Protección
Ira	Destrucción
Alegría	Reproducción
Tristeza	Reintegración
Confianza	Afiliación
Asco	Rechazo
Anticipación	Exploración
Sorpresa	Exploración

Funciones de las emociones (Plutchik, 1980)

De esta tabla obtenemos, por ejemplo, que la principal función de sentir miedo es protegernos del estímulo que lo ha creado. Nuestro cerebro, interpreta que algo puede ser perjudicial para nosotros y crea una reacción negativa que permita que nos alejemos de ello.

2.2.2.2 Funciones sociales

Que una de las funciones de las emociones sea facilitar que aparezca una conducta apropiada a un estímulo recibido, permite que los demás, a partir de nuestra emoción puedan predecir cuál va a ser nuestro comportamiento tras haber detectado la aparición de la misma. Esto facilita las relaciones interpersonales. Izard (1989), plantea que «las emociones pueden tener funciones sociales como permitir la comunicación de estados afectivos, controlar la conducta de los demás, promover la conducta prosocial o hacer más fácil la interacción social». Una emoción como la felicidad, puede favorecer la socialización y las relaciones con otras personas, mientras que la ira o el miedo pueden traer evitación o confrontación.

También puede tener función social reprimir una emoción, ya que la reacción que esta puede tener, podría afectar a las relaciones. Sin embargo, se ha demostrado que, en ciertas ocasiones, expresar una emoción puede llevar a otros al altruismo y una conducta prosocial, mientras que reprimirlas puede traer malentendidos y situaciones indeseables (Pennebaker, 1993).

Es probablemente esta función de las emociones, la que más nos interesa en este trabajo. Que NAO o cualquier otro robot con el que interactúe un humano, cuente con emociones y sea capaz de detectarlas en los demás, puede permitir facilitar sus relaciones con humanos, y despertar en él una conducta prosocial, permitiendo que ayude a aquel que vea con necesidades.

2.2.2.3 Funciones motivacionales

La relación entre emoción y motivación, es intima. Tanto es así, que en cualquier tipo de actividad nos encontramos con la “emoción”, que tiene el cometido de facilitar o dificultar una ejecución; y de una “motivación”, que se apoyará en la emoción del sujeto en un determinado momento, y que facilitará u obstaculizará la actividad a llevar a cabo, influida siempre por la emoción del momento (Chóliz, 2005).

2.2.3 Comunicación no verbal

La conducta, tanto verbal como no verbal, es el medio que utiliza la gente para comunicarse con los demás.

El lenguaje verbal se realiza a través de las palabras, pero éstas no poseen significados en sí mismas, son las personas las que les confieren el significado.

La comunicación no verbal se refiere a todos los aspectos comunicativos que no son estrictamente verbales. Entre ellos se encuentra el lenguaje corporal (gestos faciales, posturas, movimientos, distancia, contacto…) (Acinas, 2004). El lenguaje no verbal tiene un carácter analógico, está constituido por signos visuales, gestuales, auditivos, etc., que guardan relación con el objeto al que representan. Las expresiones del rostro, gestos o la posición del cuerpo pueden transmitir mensajes con mayor fiabilidad que las palabras (Marroquín y Villa, 1995).

Un individuo puede decidir no hablar, o puede no ser capaz de comunicarse verbalmente, pero todavía sigue transmitiendo mensajes sobre las emociones que está sintiendo en ese momento a los demás por medio de su cara y su cuerpo (Caballo, 1988).

Ekman y Friesen (1969) señalan incluso, que la mayor parte de los estudiosos de la comunicación no verbal, han mostrado que la clase de información que puede recogerse de las palabras que emite el individuo -información sobre afectos, actitudes, estilos interpersonales- puede extraerse también de su concomitante conducta no verbal.

Las funciones de la comunicación no verbal en relación con la comunicación verbal son (Knapp, 1972):

- **Repetición.** El mensaje verbal y el no verbal transmiten la misma información. Ej.: decir que sí y mover la cabeza de arriba hacia abajo.
- **Contradicción.** El mensaje verbal se opone al no verbal. Se toma como válido el no verbal.
- **Sustitución.** Una conducta no verbal ocupa el lugar de un mensaje oral. Ej.: sonreír, para dar a entender conformidad.
- **Complementación.** La conducta no verbal complementa una verbal, la modifica, la termina o la elabora de algún modo.
- **Acentuación.** Se enfatiza el mensaje verbal con el uso de registros no verbales.

- **Regulación.** La conducta no verbal contribuye a regular el flujo de la conversación. Ej.: el turno de conversación suele expresarse mediante un movimiento que implica cejas y barbilla.

2.2.3.1 La expresión facial

Existe una gran evidencia de que la cara es el principal sistema para mostrar las emociones, además de ser el área más importante y compleja de la comunicación no verbal y la parte del cuerpo que más de cerca se observa durante la interacción.

La expresión facial juega varios papeles en la interacción social humana (Argyle, 1969):

1. Muestra el estado emocional de un interactor, aunque este puede tratar de ocultarlo.
2. Proporciona una retroalimentación continua sobre si se comprende, se está sorprendido, se está de acuerdo, etc., con lo que se está diciendo.
3. Indica actitudes hacia los demás.
4. Puede actuar de meta-comunicación, modificando o comentando lo que se está diciendo o haciendo al mismo tiempo.

Si tenemos en cuenta la gran cantidad de músculos que hay en la cara, no es sorprendente que la gama de expresiones faciales sea muy amplia. En la comunicación, las expresiones faciales se usan fundamentalmente para expresar un cierto grado de emoción, pero existe un número limitado de emociones que la mayoría de las personas puede reconocer con cierta fiabilidad (Wainwright, 1998).

Hay un total de seis principales emociones, caracterizadas por su expresión y tres áreas de la cara responsables de su manifestación. Las seis emociones son: alegría, sorpresa, tristeza, miedo, ira y asco/desprecio, y las tres regiones faciales, la frente/cejas, ojos/párpados y la parte inferior de la cara. Esas expresiones faciales son universales e innatas. (Ekman y Friesen, 1975).

Seis principales expresiones de las emociones de izquierda a derecha. 1. Asco, 2. Ira, 3. Sorpresa, 4. Miedo, 5. Tristeza, 6. Felicidad

Argyle (1978), propone que la expresión facial, actúa como una forma de procurar retroalimentación sobre lo que está diciendo el otro. Las cejas proporcionan una interpretación continua, que es la siguiente (Argyle, 1978):

Posición de las cejas	Interpretación
Completamente elevadas	Incredulidad
Medio elevadas	Sorpresa
Normales	Sin comentarios
Medio fruncidas	Confusión
Completamente fruncidas	Enfado

El área en torno a la boca contribuye a la interpretación, variando según esté vuelta hacia arriba (“agrado”) o hacia abajo (“desagrado”).

Debido a que la expresión facial es el lugar del cuerpo en el que más evidentemente es expresada una emoción, sistemas de reconocimiento de emociones como el Emotion API de Microsoft, que se ha utilizado en este proyecto, basan su funcionalidad en muchas de estas teorías. Extraen los rasgos faciales, posición de las

cejas, forma de la boca, etc., para interpretar la emoción que la persona o personas en una imagen está sintiendo.

2.2.3.2 La postura corporal

A pesar de que la expresión facial proporciona más información sobre las emociones, Ekman y Friesen (1975) comprobaron que la postura corporal indica su grado de intensidad.

La posición del cuerpo y de los miembros, la forma en que está de pie una persona, etc., refleja sus actitudes y sentimientos sobre sí misma y su relación con los otros (Mehrabian, 1972). Este mismo autor, Mehrabian (1968), señala que hay cuatro categorías posturales:

a) **Acercamiento.** Una postura atenta comunicada por una inclinación hacia delante del cuerpo.
b) **Retirada.** Una postura negativa, de rechazo o de repulsa, comunicada retrocediendo o volviéndose hacia otro lado.
c) **Expansión.** Una postura orgullosa, engreída, arrogante o despreciativa, comunicada por la expansión del pecho, un tronco erecto o inclinado hacia atrás, cabeza erecta y hombros elevados.
d) **Contracción.** Una postura depresiva, cabizbaja o abatida, comunicada por un tronco inclinado hacia delante, una cabeza hundida, hombros que cuelgan y un pecho hundido.

Una inclinación hacia delante comunica una actitud relativamente positiva mientras que una inclinación hacia atrás o el volverse hacia otro lado, comunica una actitud más negativa (Mehrabian, 1968). También mediante la postura corporal general, una persona puede indicar su estado emocional. Por ejemplo, si está tensa o relajada, así como su estatus (o más bien la forma en que una persona percibe su estatus en relación con los demás) (Argyle, 1969; 1978).

Sarbin y Hardyck (1953), prepararon una serie de figuras hechas con varillas, y pidieron a los sujetos que describieran los estados emocionales representados por las diversas posturas.

Comparados con la expresión facial, los gestos y posturas corporales constituyen un canal de comunicación relativamente limitado.

Ekman y Friesen (1975) llegan a la conclusión de que el cuerpo es capaz de comunicar el carácter positivo o negativo de un estado emocional, así como el grado de intensidad de la emoción, pero en general no puede transmitir la emoción específica.

Estados emocionales transmitidos por la postura (Sarbin y Hardyck, 1953)

2.2.4 Conducta prosocial

En el subapartado 2.2.2.2, que titulamos como “Funciones sociales”, hemos hecho mención de la **Conducta prosocial;** destacando, aunque de forma breve, tanto el beneficio de expresar emociones, como el perjuicio de reprimirlas.

En este trabajo de fin de grado, se ha considerado imprescindible desarrollar de forma resumida el marco teórico relativo a la conducta prosocial, pretendiendo destacar el posible beneficio futuro que puede suponer añadir en NAO o cualquier otro robot humanoide, una funcionalidad que implemente diferentes conductas de ayuda dirigidas hacia los seres humanos.

Este apartado tiene como objetivo, presentar el estado actual de la investigación en conducta prosocial. Se hace necesario mencionar de forma breve, la problemática suscitada por distintos investigadores a la hora de diferenciarla de la conducta altruista.

Se considera imprescindible en este trabajo, abordar -aunque sea en un pequeño abanico de definiciones y conceptos- la complejidad que entraña tanto el constructo de conducta prosocial como el de conducta altruista. El motivo se debe a que, si distintos autores especialistas en el tema que nos ocupa, han considerado oportuno ofrecer definiciones precisas en función del campo de estudio dentro de las relaciones interpersonales; tanto más difícil puede ser tratar de reducir el *“concepto de*

ayuda al otro", si en este momento lo que pretendemos es abrir la posibilidad de que en un futuro; un robot humanoide, pueda interactuar con un ser humano en situaciones que posiblemente pudieran no ser muy atractivas para un sujeto, como es el caso de las situaciones de petición y/o necesidad de ayuda.

El objetivo de recopilar algunas de las muchas definiciones de conducta prosocial y altruista recogidas en las investigaciones desarrollados con seres humanos y en situaciones reales; es que en un futuro puedan ser valoradas para la aplicación en el campo de la interacción entre un ser humano y un robot humanoide.

2.2.4.1 Definición de conducta prosocial

Antes de adentrarnos en las definiciones que da la psicología sobre conducta prosocial y altruista, deberíamos preguntarnos si el ser humano es capaz de sacrificarse por los demás sin esperar una recompensa a cambio (Munné, 1986).

Munné (1986), destaca dos tipos de altruismo:

1. El denominado *"por reciprocidad"*, que consiste básicamente en ayudar a los que nos han ayudado. Este altruismo no puede considerarse puro, sino de agradecimiento y más bien fundado en la esperanza de recibir otras ayudas en el futuro.
2. El altruismo de *"responsabilidad social"*, al que considera un altruismo más depurado, con una conducta simplemente de ayuda destinada al que la necesita.

Esta concepción de la conducta altruista y otras similares, conducen a Batson y Powell (2003), a destacar la dificultad para poder distinguir una conducta prosocial de una altruista.

Para Howard y Piliavin (2000), el "altruismo" se considere como el resultado de anteponer la necesidad de los otros a la propia, persiguiendo el beneficio exclusivo del receptor y no del actor.

López (1994), define el "altruismo" como una disposición y orientación hacia el bien de los otros. Por esto, "la conducta altruista" es toda acción voluntaria realizada con la intención de ayudar a los demás, provocando o manteniendo efectos positivos.

Holmgren, Eisenberg y Fabes (1998), proponen definir la conducta altruista como aquellos actos sociales beneficiosos para los demás, llevados a cabo por motivos o valores internos sin buscar ningún tipo de recompensa externa.

Para González Portal (2000), la concepción de "conducta altruista" va ligada a la existencia de un móvil de tipo altruista por parte del que la ofrece (benefactor), es decir, **motivo que conduzca** a llevar a cabo la propia conducta, como se desprende de la definición por los autores anteriores. Argumenta que la discusión sobre si la conducta realizada, tiene una base motivada o no para proyectar la ayuda al otro, ha sido uno de los argumentos realizados por distintos estudiosos del tema; que ha conducido a un callejón sin salida para poder diferenciar estos dos tipos de conducta: la prosocial y la altruista.

Para acabar la exposición relacionada con "conducta altruista", añadiremos que gracias a los estudios desarrollados sobre el denominado *"efecto del espectador"*, cuyos pioneros fueron Latané y Darley (1968); irrumpen con fuerza las investigaciones dirigidas a clarificar lo que en su día se llamó "altruismo" (cit., González Portal, 1992, pp.20). Hoy día, este término ha sido reformulado en el campo de estudio de la psicología social, en términos de "conducta prosocial de ayuda". Es más, el hecho de dejar de lado el "altruismo" con un cariz más filosófico que psicológico, ha ayudado a crear un campo nuevo de conocimiento en psicología.

En este campo de la psicología social, al término "conducta prosocial" se le ha añadido el de "tipo asistencial". Este gran constructo *"conducta prosocial de tipo asistencial"*, presenta hoy día; un amplio marco teórico y práctico, destinado al desarrollo de habilidades sociales de interacción para profesionales de la salud (Pades, 2003).

Pasamos ahora a abordar algunas definiciones de conducta prosocial. Para Benson, Scales, et al. (2006), la conducta prosocial se da para poder satisfacer la necesidad de apoyo físico y emocional de otra persona.

A esta idea de comportamiento se suman Eisenberg y Fabes (1998), argumentando además que la conducta prosocial, está basada en conductas voluntarias que se adoptan para cuidar, asistir, confortar y ayudar a otros.

En un intento de justificar el motivo por el que la "conducta prosocial", pudiera ser estudiada empíricamente y también poder desarrollar diferentes tipologías para hacer más factible su aplicación; González Portal (1992), propuso una definición que cerrará las controversias existentes relacionadas con los constructos de altruismo y prosocial, y que han sido marcadas entre distintos autores.

> *Se entiende por conducta prosocial toda conducta social positiva con/sin motivación altruista (pp.36).*

De esta forma, el término es tomado en su acepción más abarcativa y de este modo, puede representar un área considerable de estudio, pudiendo considerar como referente a “toda conducta de orientación interpersonal que beneficie a otro”.

González Portal, sigue argumentando que con la definición que aporta:

> *Podemos admitir una definición de conducta prosocial como “toda conducta social positiva con/sin motivación altruista.”, o simplemente, que no simplonamente, “conducta social positiva”, sobreentendiendo la innecesaria especificación motivacional por no añadir nada determinante al respecto (pp. 37).*

2.2.4.2 Clasificación de la conducta prosocial

La misma diversidad que se encuentra en el momento de definir el concepto de conducta prosocial, se traslada al intento de establecer una clasificación de los tipos de comportamientos que esta podría incluir (González Portal, 1992).

Entre las diversas taxonomías propuestas, se ha adoptado la elaborada por Roche (1997) en su trabajo sobre educación prosocial, debido a que se ha considerado la más completa de las consultadas. El autor establece las siguientes clases de comportamientos:

- **Ayuda y servicio físico**: conducta no verbal de asistencia a otros para cumplir objetivos determinados.

- **Dar**: entregar objetos y bienes a otros.

- **Ayuda verbal**: explicar y compartir ideas y experiencias que son útiles para otros en la consecución de objetivos.

- **Consuelo verbal**: expresiones verbales que tienden a reducir un estado angustioso o problemático elevando la autoestima.

- **Confirmación y valorización positiva del otro**: refuerzos verbales positivos para confirmar el valor de los otros o aumentar su autoestima.

- **Escucha profunda**: conductas y actitudes de atención que expresan interés sostenido hacia los contenidos expresados por los otros.

- **Conducta empática**: verbalización que expresa comprensión cognitiva de los pensamientos y emociones de otros.

- **Solidaridad:** conductas físicas y verbales que asumen la responsabilidad de ayudar a otros, particularmente a aquellos que se encuentran en una situación de vulnerabilidad.

- **Presencia positiva y unidad**: presencia personal que expresa actitudes de proximidad psicológica, atención, escucha profunda, empatía, disponibilidad para el servicio, la ayuda y la solidaridad para con otros y que contribuye al clima psicológico de bienestar.

Una vez hemos definido y mostrado la problemática actual en el campo de la psicología relativa a la *conducta de ayuda a los demás*, podríamos considerar que, desde el campo de la ingeniería informática, esta puede ser útil para la interacción entre humanos y robots humanoides.

Introducir en un futuro a un robot, una actitud de ayuda a los demás sin pedir nunca nada a cambio, ni sentir la necesidad de obtener una recompensa, puede permitir crear nuevas formas de interacción que beneficien al ser humano.

3. Herramientas utilizadas

En este capítulo se exponen las diferentes herramientas que han sido necesarias para el desarrollo del proyecto, desde el robot, elemento clave hasta las API, software o lenguajes de programación que han sido útiles.

3.1 Robot humanoide NAO

NAO es un robot humanoide diseñado y construido por la empresa francesa Aldebaran-Robotics y que fue elegido como el robot a utilizar en las ligas RoboCup a partir del año 2008 para sustituir al cuadrúpedo AIBO (Gouaillier et al., 2009).

Un robot humanoide es aquel cuya anatomía y apariencia imitan a la del ser humano. Es decir, cuentan con una cabeza, dos brazos y salvo en ciertos casos, dos piernas sobre las que caminan.

Robot humanoide NAO de Aldebaran Robotics

Los propios diseñadores destacan que una de las características que diferencian a NAO de otros robots humanoides es su asequibilidad. En su lanzamiento, su precio rondaba los diez mil euros, encontrándose en la actualidad en unos seis mil euros. Su precio permite que este sea más accesible a todo aquel que quiera realizar investigaciones con robots bípedos.

En cuanto a las características físicas del robot, este tiene una altura de 0,57 m y un peso de 4,5 kg. Cuenta con 25 grados de libertad. Que le permiten imitar el movimiento de un ser humano y moverse con cierta soltura por su entorno. Además, cuenta con un módulo de inercia que es el que le permite mantener el equilibrio y saber si se encuentra de pie y sentado.

Cuenta con cuatro **micrófonos**, dispuestos en su cabeza de tal manera que le permiten escuchar todos los sonidos que ocurren a su alrededor. Dos **cámaras** dispuestas en su cara simulando ojos, le permiten reconocer caras y objetos, además de medir su proximidad mediante sensores de infrarrojos. Junto a los micrófonos le permiten recibir información del entorno.

Además de los sensores infrarrojos que le ayudan a medir la distancia que le separa de los objetos a su alrededor, también cuenta con sensores táctiles en la cabeza y las manos, que permiten interaccionar con él. También tiene sensores de tipo *bumper* en los pies, que permiten saber si se ha chocado con un objeto.

En el pecho, dispone de un botón. Es el utilizado para apagar y encender el robot. Además, cuando este está encendido, aporta datos como su dirección IP. Este botón aporta información sobre el estado del robot cambiando de color.

Botón	**Significado**
	Una luz blanca significa que el robot está listo para usarse
	La luz verde indica que la batería se encuentra a 3/5 de su capacidad
	Una luz amarilla indica que la batería se encuentra entre 3/5 y 1/5 de su capacidad. Si la luz amarilla parpadea, NAOqi no está funcionando y el robot debe reiniciarse
	Una luz roja parpadeante indica que la batería debe ser cargada inmediatamente ya que se encuentra en un 1/5 de su capacidad

Posibles estados del botón del pecho cuando NAO está encendido

Puede informar sobre su estado también mediante la voz, con mensajes como “batería baja” o “motor caliente”. También cuenta con LEDs en sus orejas (realmente altavoces a través de los que emite los sonidos), que informan sobre el progreso de encendido del robot.

NAO ha sido elegido para la realización de este proyecto ya que, entre otros, su sistema de reconocimiento facial permite mantener una base de datos de las personas a las que el robot conoce, manteniendo así su perfil emocional. Además, su cámara también nos permite tomar imágenes para después reconocer la emoción que la persona en cuestión está teniendo en ese instante.

3.2 Python

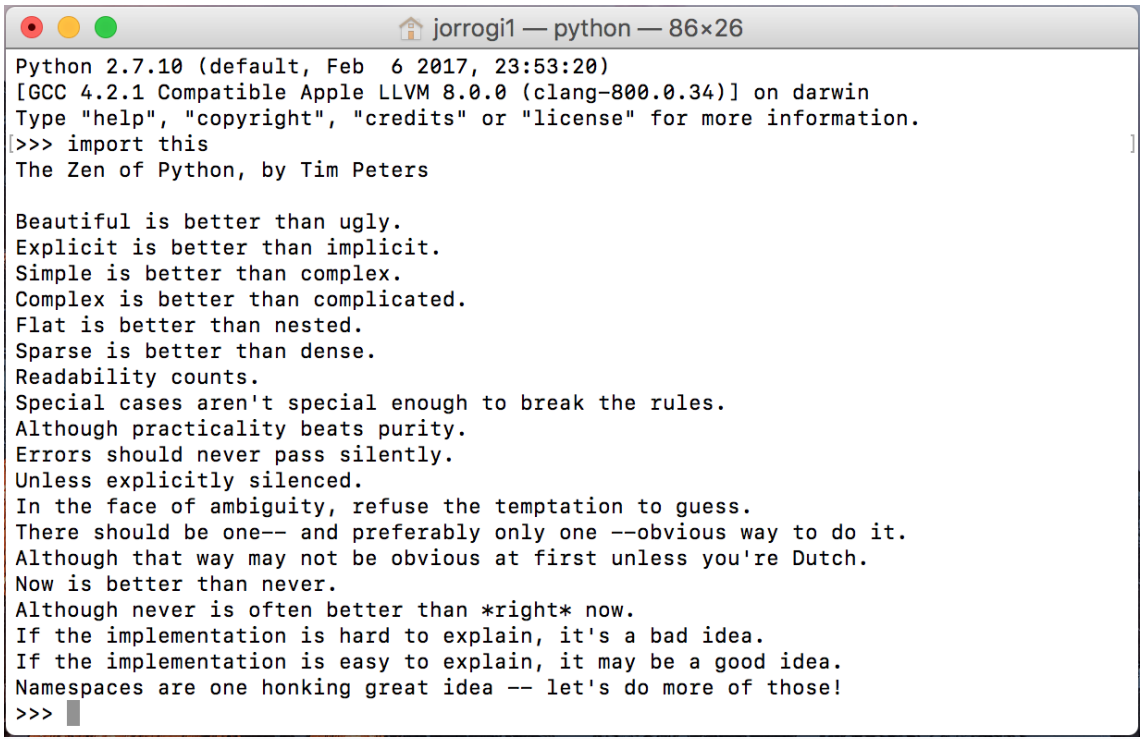
Python es un lenguaje de programación de propósito general definido comúnmente como *lenguaje de scripting orientado a objetos*. Esto es debido a que Python es un lenguaje interpretado que soporta orientación a objetos (Lutz, 2013).

Guido van Rossum, creador de Python en 2006

Fue creado a finales de los años ochenta por Guido van Rossum con el fin de crear un lenguaje más accesible a la gente que tuviera un nivel muy básico de programación. Estas ideas con las que fue creado Python se mantienen muy vivas, hasta tal punto que sus usuarios se refieren a menudo al término **Filosofía Python.** Entre los principios de esta filosofía se encuentran la legibilidad y transparencia del código.

Tan importante es esta filosofía, no solamente para sus desarrolladores si no también para los usuarios del lenguaje, que abriendo un terminal interactivo Python a partir de la versión 2.1.2 y ejecutando `import this`, se nos muestra un *Easter egg* con los principios de diseño y limpieza de código de Python, haciéndolo así fácilmente accesible a cualquiera que quiera consultarlo.

Se ha elegido utilizar Python en lugar de C++ en este proyecto ya que, además de la facilidad de escribir código en este lenguaje, el robot NAO es capaz de ejecutar módulos Python directamente en él. Como los propios creadores de NAO dicen, “Utilizar Python es la forma más sencilla de programar con NAO”.



Resultado de escribir "import this" en un terminal Python

3.3 NAOqi Python API

El robot NAO es controlado por un sistema operativo especializado basado en Linux, denominado NAOqi. Cuenta con una **interfaz de programación de aplicaciones**, **API** por sus siglas en inglés, que proporciona instrucciones a ejecutar en el robot de una manera más sencilla.

Esta se encuentra disponible para los lenguajes Python y C++. Como se ha comentado antes, para este proyecto se ha escogido utilizar Python.

NAOqi Python API se divide en diversos módulos, cada uno especializado en diferentes funcionalidades que puede realizar el robot:

- **Core:** Se encarga de la parte más crítica del robot, acceder a su memoria, crear módulos, administrar recursos, etc.
- **Motion:** Provee métodos que facilitan hacer que NAO se mueva.
- **Audio:** Es el encargado de utilizar los componentes software de audio del robot. En este módulo se encuentran módulos tan interesantes como "ALSpeechRecognition" que permite el reconocimiento del habla y "ALTextToSpeech" que permite que el robot hable.

- **Vision:** En este módulo se encuentran los métodos que permiten utilizar las funcionalidades de la cámara del robot. En él se encuentra por ejemplo el módulo “ALFaceDetection” útil para reconocer personas en su entorno.
- **Sensors:** Permite leer la información de los sensores que incorpora el robot. Como nivel de batería, su sonar, etc.
- **Trackers:** Es un módulo sencillo que permite a NAO seguir objetos como caras o una pelota roja con la mirada.

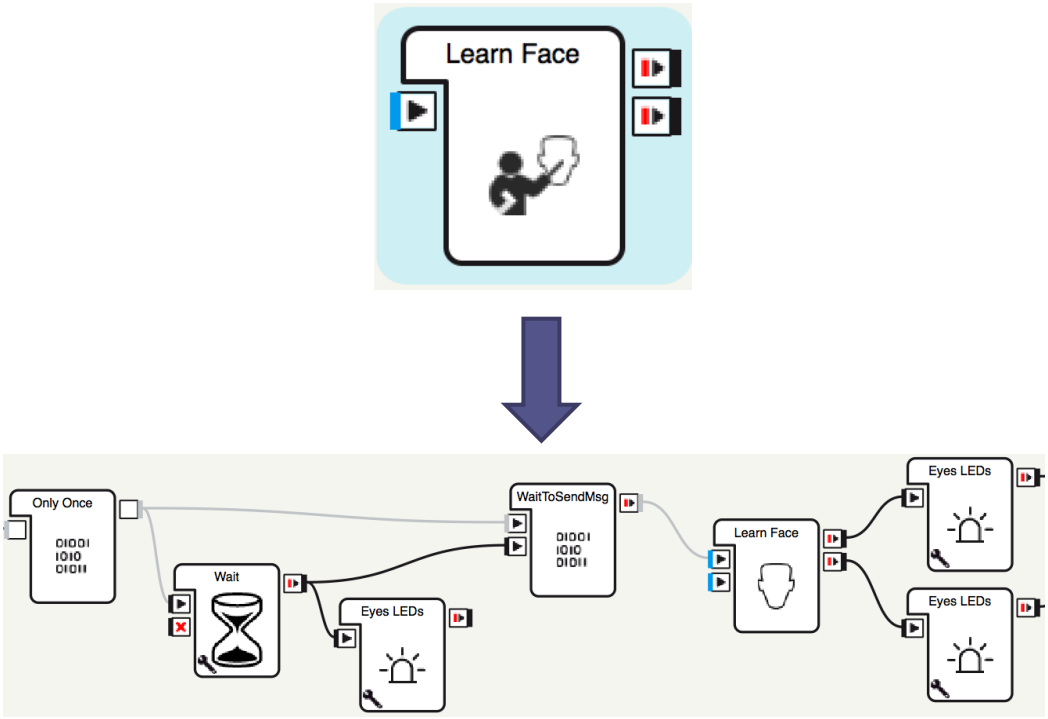
3.4 Entorno gráfico Choregraphe

En el punto anterior se ha hablado del API Python que permite programar el comportamiento del robot. Si bien es cierto que este API y el lenguaje Python permiten programar comportamientos complejos con cierta simplicidad en el código desarrollado, no todo el mundo cuenta con los conocimientos necesarios para hacerlo, y dado que uno de los objetivos desde el diseño de NAO fueron la accesibilidad a él de cualquier persona, su programación presentaba una barrera.

Por ello, los creadores de NAO vieron necesario proveer de una forma más sencilla de programar el robot. Con esta pretensión nació Choregraphe, que permite la programación de aplicaciones complejas de manera gráfica, haciendo así esta tarea más rápida y sencilla.

Choregraphe muestra en su ventana principal un diagrama de flujo en el que se pueden añadir cajas que implementan comportamientos del robot. Cada uno de estos comportamientos predefinidos cuenta en su interior con más comportamientos (Monceaux, 2009). Por ejemplo, la caja *Learn Face* que permite que el robot almacene en su memoria la cara de una persona para poder reconocerla después, cuenta en su interior con cajas de espera, de activación de los LEDs de su cara, etc.

Si bien es cierto que Choregraphe puede ser muy útil en aplicaciones sencillas, para este proyecto su funcionalidad no ha sido suficiente. El simulador no es capaz, por ejemplo, de ejecutar las funciones de reconocimiento de voz del robot haciéndolo por lo tanto poco útil para el cometido de este trabajo. Por lo tanto, este programa se ha utilizado para probar ciertos comportamientos de manera más rápida y sencilla que haciéndolo mediante código Python.



Cajas que forman la caja de jerarquía superior *Learn Face*

3.5 Microsoft Emotion API

El Emotion API de Microsoft es un servicio de Microsoft Azure que tomando como entrada una imagen, devuelve las caras que hay en ella y el porcentaje de confianza en una serie de emociones. Estos porcentajes permiten saber con qué probabilidad la persona o personas que salen en la foto tienen una emoción.

Los datos son recibidos en formato JSON, acrónimo de JavaScript Object Notation, un formato de texto ligero que permite el intercambio sencillo de datos. Entre la información recibida, encontramos un apartado para cada cara que ha sido detectada. En cada apartado encontramos dos subapartados, referentes a la posición de la cara dentro de la imagen y a las emociones asociadas a esa cara.

Las emociones que el API puede detectar son: ira, desprecio, aversión, miedo, felicidad, neutralidad, tristeza y sorpresa. Microsoft Emotion API supone que la forma en que se expresa una emoción facialmente es compartida por todas las culturas y por lo tanto universal.

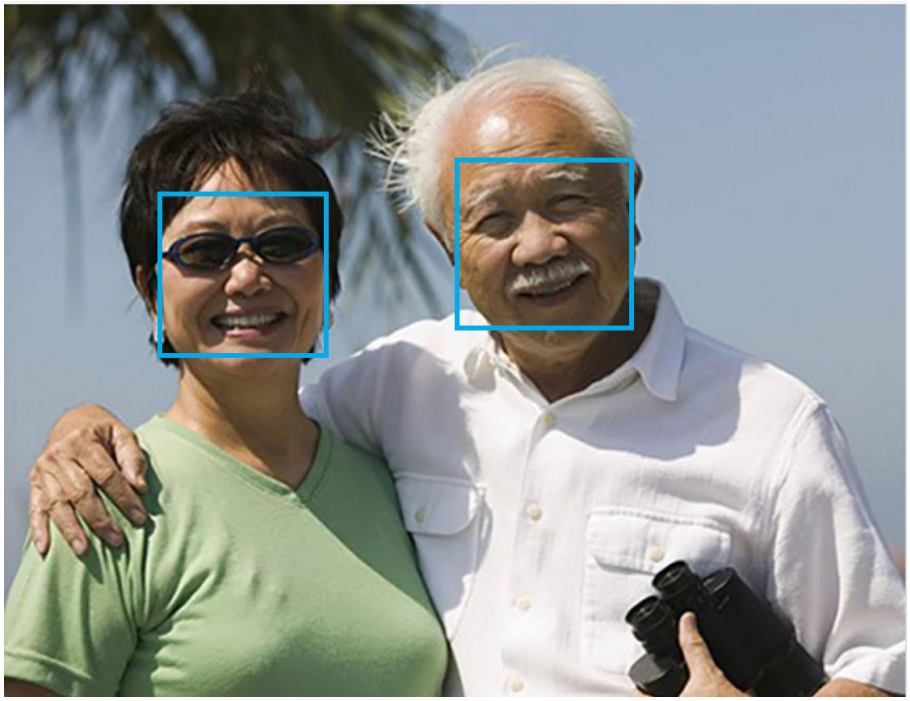


Imagen utilizada para detectar emociones con Emotion API. Se muestran los rectángulos de detección de cara.

```
Resultado de la detección:
2 caras detectadas

 JSON:
 [
  {
    "FaceRectangle": {
      "Top": 103,
      "Left": 297,
      "Width": 117,
      "Height": 117
    },
    "Scores": {
      "Anger": 3.03948218E-05,
      "Contempt": 1.30746948E-05,
      "Disgust": 0.00325552234,
      "Fear": 8.074536E-08,
      "Happiness": 0.9926525,
      "Neutral": 0.00363545,
      "Sadness": 0.000333539385,
      "Surprise": 7.943756E-05
    }
  },
  {
    "FaceRectangle": {
      "Top": 126,
      "Left": 104,
      "Width": 113,
      "Height": 113
    },
    "Scores": {
```

```
      "Anger": 0.000137224473,
      "Contempt": 6.922984E-07,
      "Disgust": 7.38530143E-05,
      "Fear": 1.87493924E-07,
      "Happiness": 0.999765158,
      "Neutral": 1.25752331E-05,
      "Sadness": 1.67030578E-06,
      "Surprise": 8.613004E-06
    }
  }
]
```


Resultado devuelto por Emotion API para la imagen mostrada arriba

4. Funcionalidad desarrollada

Uno de los objetivos de este trabajo, como se ha comentado anteriormente, es desarrollar un sistema en el que un robot es capaz de percibir las emociones de los usuarios que le rodean e interactuar de manera acorde a esas emociones. En primer lugar, se expondrá el diseño de la aplicación, para comprender como funciona. Posteriormente se explica el código desarrollado para que sea posible ejecutar esta funcionalidad.

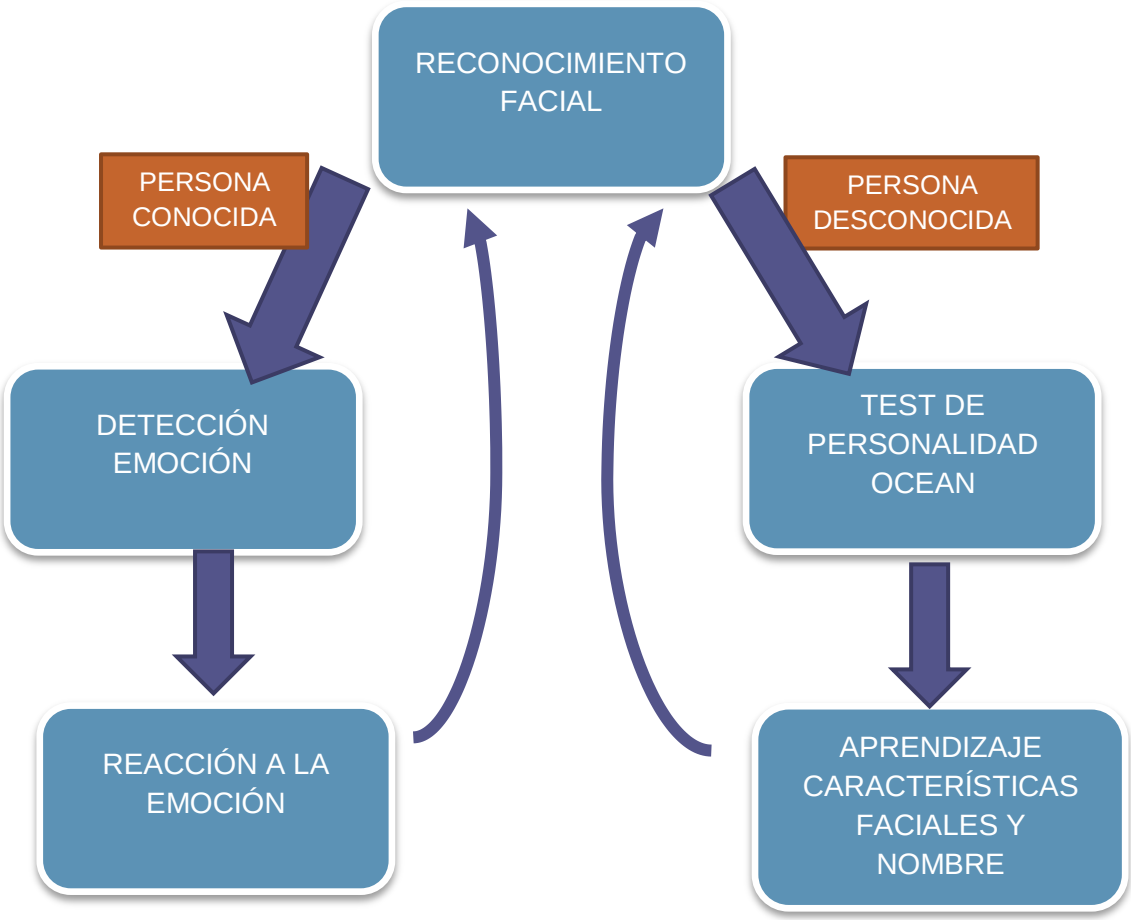
4.1 Diseño de la aplicación

Esta aplicación podría verse como un autómata de estados finitos con cinco posibles estados que se ejecutan una y otra vez si la aplicación sigue en marcha.

En primer lugar, se ejecuta el reconocimiento facial que implementa el robot. Esta fase permite decidir cuál será el siguiente paso a ejecutar. En caso de que la persona con la que el robot se encuentra interactuando sea **conocida**, podrá pasarse a reconocer la emoción que tiene en ese momento, para poder así actuar consecuentemente. Si, por el contrario, la persona **no es conocida**, se pasará a un proceso de aprendizaje de sus características faciales tras un test de personalidad de tipo OCEAN, que permitirá en futuros trabajos conocer el perfil psicológico de la persona con la que se interactúa.

Tanto si la persona es conocida, en cuyo caso el robot actuará reaccionando a la emoción que ha detectado, o si es desconocida, en cuyo caso aprenderá las características necesarias del usuario, el bucle volverá al principio, es decir, se volverá al estado de reconocimiento facial ejecutado en el primer paso.

Descrito el funcionamiento de la aplicación, pasamos a explicar los detalles de implementación de cada uno de los pasos de ejecución de la aplicación, teniendo en cuenta el código desarrollado y las tecnologías utilizadas para conseguir los objetivos planteados.



4.2 Script principal FaceDet.py

Todo el código de este proyecto ha sido realizado mediante scripts Python, como se comentó en puntos anteriores. El script principal que controla todo el flujo de ejecución es **FaceDet.py**, que corresponde con el estado "RECONOCIMIENTO FACIAL".

4.2.1 Imports

Las primeras líneas con las que nos encontramos en el script ejecutan los *import* de librerías y otros ficheros necesarios para la implementación del código posterior.

```
1  import time
2  import sys
3  import oceanreduced as ocean
4  import emotionAPI as emotion
5  import takephoto
6  from random import randint
7
8
9  from naoqi import ALProxy
10 from naoqi import ALModule
11 from naoqi import ALBroker
12
13 from optparse import OptionParser
14
```

***Imports* script FaceDet.py**

Se importan librerías propias del lenguaje Python, como *time*, que permite medir tiempos de ejecución de código pudiendo medir así en cierta manera la eficiencia de la funcionalidad desarrollada. En este caso, ha sido utilizado para medir el tiempo medio necesario para realizar una llamada a la Emotion API de Microsoft y obtener una respuesta.

La librería *sys* ha sido utilizada para permitir parar la ejecución del programa ya que es un bucle infinito. Mediante un atajo de teclado (**Control+C),** la instrucción **sys.exit(0)** permite parar el programa terminando así el bucle. También se importa la instrucción ***randint*** de la librería *random*, que permite obtener un número entero aleatorio en un rango especificado mediante dos argumentos. ***Optparse*** de la librería *OptionParser* permite la configuración del bróker de comunicación con el robot NAO.

En las líneas 3,4 y 5 se pueden ver *imports* relativos a otros scripts desarrollados para este proyecto, como **oceanreduced.py** que permite que el usuario

realice un test de personalidad del **modelo de los cinco grandes**, que se comentará en puntos siguientes.

El script **emotionAPI.py**, permite realizar llamadas a la Emotion API de Microsoft y leer su respuesta, devuelta en formato JSON.

Takephoto.py utiliza la cámara del robot NAO para tomar una instantánea de la cara de la persona y guardarla en el ordenador desde el que se llama al script **FaceDet.py**. Esta imagen será la que se enviará a la Emotion API y permitirá detectar la emoción que está teniendo el usuario en ese instante.

Las líneas 9, 10 y 11 permiten importar las librerías necesarias para que nuestro código pueda comunicarse con el robot. ALProxy del API NAOqi es la más importante, ya que permite crear proxies con las distintas funcionalidades del robot, como pueden ser el reconocimiento facial, del habla, etc.

4.2.2 Creación de los proxies de comunicación

Como ya se ha comentado en el punto anterior, para poder utilizar módulos del API NAOqi que nos permitan utilizar la funcionalidad de NAO, es necesario crear proxies. Un proxy es un objeto que se comportará como el módulo al cual representa, y, por lo tanto, en este objeto se encontrarán todos los métodos del módulo en cuestión.

```
class FaceDetModule(ALModule):

    def __init__(self,name):
        self.IP = "192.168.1.9"
        self.PORT = 9559
        ALModule.__init__(self, name)
        try:
            global memory
            memory = ALProxy("ALMemory",self.IP,self.PORT)
        except Exception, e:
            print "Error: ",e
        try:
            global faceProxy
            faceProxy = ALProxy("ALFaceDetection",self.IP,self.PORT)
        except Exception, e:
            print "Error: ",e
        try:
            global asr
            asr = ALProxy("ALSpeechRecognition",self.IP,self.PORT)
        except Exception, e:
            print "Error: ",e
        try:
            global tts
            tts = ALProxy("ALTextToSpeech",self.IP,self.PORT)
        except Exception, e:
            print "Error: ",e
        try:
            global managerProxy
            managerProxy = ALProxy("ALBehaviorManager", self.IP, self.PORT)
        except Exception, e:
            print "Error: ",e
```

Declaración de los proxies utilizados en FaceDet.py

Lo primero que se hace en el código es declarar las variables **self.IP** y **self.PORT**, que corresponden a la dirección IP del robot y el puerto por el que se realiza el proxy.

Posteriormente, se pasa a crear el proxy con el módulo **ALMemory**, el cual entre otras cosas permite acceder a la memoria del robot, pudiendo así extraer por ejemplo la última palabra que ha detectado este mediante su sistema de reconocimiento del habla.

El siguiente proxy creado es con el módulo **ALFaceDetection** que es el empleado para detectar caras, reconocer personas, seguirlas con la mirada, y en definitiva, gestionar toda la funcionalidad de reconocimiento facial que tiene implementado el robot.

El proxy con el módulo **ALSpeechRecognition**, permite utilizar el sistema de reconocimiento del habla del robot.

El siguiente proxy es el creado con el módulo **ALTextToSpeech** es utilizado para que el robot sea capaz de comunicarse. El método *say* de este módulo, permite

que el robot diga en voz alta las palabras que se le pasan como entrada en una cadena de caracteres.

Y, por último, se crea el proxy con el módulo **ALBehaviorManager**, que permite instalar, cargar, eliminar y en definitiva administrar comportamientos del robot (ficheros .crg), que contienen acciones que el robot debe realizar, como movimientos, un diálogo, etc. y que pueden ser ejecutados desde los scripts que estamos utilizando. En nuestro caso, este módulo será útil para llamar a comportamientos que debe realizar el robot en función de la emoción que ha sido detectada mediante el Emotion API de Microsoft.

Es importante destacar que, para poder utilizar estos módulos en todo el código, las variables deben ser declaradas como globales, de esta manera serán accesibles desde cualquier punto. Para crear una variable global en Python, es necesaria la palabra reservada “global”.

4.2.3 Fase de reconocimiento facial

Para este caso de estudio, la detección facial y el reconocimiento de personas es un factor muy importante, ya que para cada persona que el robot ya conoce, se guarda un perfil de personalidad obtenido mediante el test de los cinco grandes, también denominado OCEAN del que se hablará en puntos posteriores.

El primer paso que debemos realizar en nuestro código para ello es suscribirnos al servicio de reconocimiento facial del robot NAO (línea 48). Esto permite que el robot encienda la cámara y busque caras.

```
48          memory.subscribeToEvent("FaceDetected","FaceDet","onFaceDetected")
49          tts.setVolume(0.3)
50
51      # This will be called each time a face is detected.
52      def onFaceDetected(self, eventName, value, subscriberIdentifier):
53          # Unsubscribe to the event when talking, to avoid repetitions
54          try:
55              memory.unsubscribeToEvent("FaceDetected","FaceDet")
56          except IndexError, e:
57              print e
58
```

Suscripción al sistema de reconocimiento facial y evento lanzado al reconocer una cara

El método **onFaceDetected** cuya cabecera se encuentra en la línea 52, es invocado cuando se lanza un evento “FaceDetected”. Este método por lo tanto permite realizar las acciones deseadas cuando una cara ha sido detectada por el sistema de reconocimiento facial del robot. Es importante destacar que este evento se lanza

cuando se **detecta** una cara, no cuando esta es reconocida, es decir, no es necesario que el robot ya conozca a la persona a la que está viendo para que este evento sea lanzado.

Como se puede observar en la línea 55 del código, al entrar en el evento, se desuscribe al robot del evento, para evitar así que este sea lanzado repetidas veces mientras se están realizando las acciones deseadas. Esto debe realizarse dentro de un **try,** ya que la operación de desuscripción puede lanzar un **IndexError** y producir fallos en el programa, abortando su ejecución inesperadamente.

Tras haber desuscrito al robot del servicio de reconocimiento facial, podemos pasar a tratar la información que hemos recibido mediante el evento. Toda esta información relativa a la cara que ha visto NAO se encuentra en la variable **value**. En value, podemos encontrar todas las características que el robot detecta en la cara del usuario, y que posteriormente le permiten comprobar si lo conoce o no. Se recogen medidas como distancia entre los ojos, anchura de la mandíbula, y demás rasgos.

```
58
59          try:
60              persona = value[1][0][1][2]
61              if persona == "":
62                  valOcean = ocean.main()
63                  #Preguntar nombre persona
64                  nombre = crearNombre(valOcean)
65                  print(nombre)
66
67                  OCEANProfiles = open("oceanProfiles.txt","a")
68                  #Escribimos valores de la persona en el fichero de
69                  #perfiles OCEAN
70                  OCEANProfiles.write(nombre+"\n")
71                  OCEANProfiles.write(str(valOcean)+"\n")
72                  OCEANProfiles.close()
73
74                  #Nao aprende la cara de la persona
75                  faceProxy.learnFace(nombre)
```

Extracción del nombre de la persona detectada y código ejecutado en caso de no conocerla

En la línea 60, se puede ver cómo se extrae el nombre de la persona detectada si esta es conocida. Si este usuario es nuevo para el robot, este valor será un *string* vacío. Dado que la información se devuelve mediante un array multidimensional, extraer el nombre con el que se ha guardado a esa persona no consiste más que en navegar entre las dimensiones hasta llegar al nivel en que este se encuentra.

En caso de que el robot no conozca a la persona todavía, se lanza el script que contiene el test de personalidad, pudiendo así crear un pequeño perfil psicológico del usuario.

La próxima acción consiste en crear un nombre aleatorio para la persona. Tras diversos intentos con el sistema de reconocimiento de voz de NAO, se vió imposible utilizarlo para que reconociera el nombre de la persona cuando ésta se lo dijera en voz alta, ya que este sistema, aunque muy útil, funciona a partir de un vocabulario limitado que el programador tiene que pasar al robot.

En el apartado en el que se hable del método **crearNombre**, se desarrollará más la problemática surgida y cómo se pudo solucionar.

Tras crear el nombre aleatorio para el usuario que está siendo reconocido, el siguiente paso es escribir en un fichero su valor OCEAN y nombre, pudiendo así mantener una base de datos con los perfiles psicológicos de todas las personas a las que el robot ya conoce.

La última acción a realizar en caso de no conocer al usuario con el que se está tratando es aprender las características de su cara. Para ello existe un método del módulo ALFaceDetection que lo realiza, guardando en la base de datos del robot las características faciales de la persona con el nombre que se le indica mediante la entrada del método.

En el caso contrario, en el que el robot si conoce a la persona, la variable “persona” no estará vacía, si no que estará compuesta por el nombre con el que se guardó la cara de esta en el robot. Es por esto que en caso de que la persona ya sea conocida para el robot, el código entra en el bloque ***else*** que se muestra a continuación.

```
            else:
                #Abrimos fichero perfiles OCEAN
                archnombres = open("OCEANProfiles.txt").read().split()
                #Obtenemos posicion del nombre de la persona en el archivo
                #de perfiles ocean
                nom = archnombres.index(persona)
                valPersona = archnombres[nom+1]
                #Tomamos foto de la persona
                takephoto.main()
                #Pasamos foto al detector de emociones de la API de Microsoft
                emotions,values = emotion.detEmotion("./emocion.png")
                valmax = max(values)
                pos = values.index(valmax)
                emocionmax = emotions[pos]
```

Código ejecutado en caso de que el robot si conozca al usuario

El primer paso es extraer del fichero en que se encuentran todos los perfiles psicológicos de las personas que el robot conoce, el perfil de la persona en cuestión. Dado que sabemos el nombre de la persona gracias al reconocimiento facial, es sencillo encontrar su perfil en el fichero.

Cuando ya se conoce su valor OCEAN, se procede a tomar una instantánea de la cara de la persona mediante el script takephoto.py, del que se hablará próximamente con más detalle. Este script permite utilizar la cámara del robot NAO, tomar con ella una instantánea y guardarla en el ordenador y directorio desde el que se ha ejecutado FaceDet.py.

El siguiente paso será por lo tanto utilizar el script emotionAPI.py que permite hacer una llamada al Emotion API de Microsoft y obtener la respuesta.

Finalmente, se ejecutarán los comportamientos que debe realizar el robot dependiendo de cuál es la emoción que ha detectado mediante el API. Este fragmento de código se desarrollará en punto separado debido a su mayor complejidad.

4.2.4 Nombre aleatorio mediante crearNombre

Como ya se ha comentado en el punto anterior, el sistema de reconocimiento del habla del robot NAO, no es todo lo abierto que debería. NAO solo puede reconocer las palabras o frases que mediante la instrucción setVocabulary del módulo ALSpeechRecognition se le pasan, creando así un vocabulario de las palabras que es necesario reconocer.

```
89      vocabulary = ["uno","dos","tres","cuatro","cinco"]
90      #Pausa del sistema de reconocimiento de voz para configuracion
91      asr.pause(True)
92      asr.setLanguage("Spanish")
93      asr.setVocabulary(vocabulary,False)
94      asr.pause(False)
```

Creación de vocabulario en el que se podrán reconocer los números del 1 al 5

Es por ello **imposible** que el robot pueda preguntar a la persona de la que está aprendiendo la cara su nombre, ya que no sería capaz de comprender la respuesta. Si hubiera sido necesario realizarlo, tendría que haberse creado un vocabulario casi infinito de nombres de persona posibles, método poco viable.

También se consideró poco útil para este problema pedir que el usuario escribiera su nombre mediante teclado, ya que la interacción con el robot sería mucho más compleja.

Por todo esto, se decidió crear un nombre aleatorio a cada usuario, que sería utilizado para la funcionalidad del robot, pero nunca revelado a la persona. Así, cada persona tendría un identificador con el que el robot sería capaz de distinguirlo. Con este fin, se creó el método **crearNombre**.

```
98
99  def crearNombre(valOcean):
100     nombre = "persona"+str(randint(10,99))
101     for i in valOcean:
102         nombre+=str(i)
103     return nombre
104
```

Método crear nombre que permite asignar un nombre semi aleatorio a cada usuario

En el código se puede observar que se emplea el valor OCEAN obtenido del test para identificar a cada usuario. Por lo tanto, cada identificador se compondrá de la palabra persona, seguido de un número entero elegido de forma aleatoria entre 10 y 99, útil para evitar colisiones entre personas con exactamente el mismo perfil de personalidad y por último, de su valor OCEAN.

Un ejemplo de nombre sería: **persona4122919359.** Siendo 41 el número aleatorio entre 10 y 99 y 22919359 el valor obtenido mediante el test de personalidad de los cinco grandes.

4.3 Test de los cinco grandes

Como ya se ha comentado en párrafos anteriores, el script FaceDet.py sirve de control de ejecución de la funcionalidad desarrollada. Realizará una llamada a un script o a otro dependiendo de si la persona con la que el robot está interactuando, es ya conocida o no.

En caso de que las características faciales del usuario no se encuentren en la memoria del robot, se lanza un test de personalidad denominado **test de los cinco grandes**.

El test de los cinco grandes, también denominado de los cinco factores u OCEAN, es un modelo de personalidad que permite extraer el perfil psicológico de una persona con relativa sencillez. Creado a finales de los años 80 por John, P., Naumann, L. y Soto, C., fue revolucionario tras su publicación. Otros modelos siempre se han basado en modelos anteriores creados por teóricos de renombre en esta materia.

El desarrollo del BFM (Big Five Model), fue más empírico. Sus autores se basaron en la lengua, consideraron que aquellos aspectos más importantes y de más calado para las personas en cuanto a personalidad, tendrían un mayor número de grados y formas de expresarlos en el lenguaje.

Los primeros estudios se llevaron a cabo en inglés, se sacaron cinco factores importantes, que fueron: *extraversion* (extroversión), *agreeableness* (agradabilidad), *conscientiousness* (control), *neuroticism* (neuroticismo o inestabilidad emocional) y *openness* o *unconvencionality* (originalidad). De estos factores se extrae el término OCEAN:

Openness

Conscientiousness

Extraversion

Agreeableness

Neuroticism

Para este proyecto se ha utilizado un test de los cinco grandes reducido de cincuenta preguntas. Otros test de estas características pueden llegar a contar con cientos de preguntas, cantidad inviable para nuestra aplicación, que pretende ser ágil y poco pesada en la interacción.

Con el fin de que el robot pase el test al usuario, se ha desarrollado el script oceanreduced.py, que lee de un fichero de texto plano todas las preguntas del test, las lee en voz alta, recoge la respuesta del usuario y finalmente realiza un cálculo que muestra el valor OCEAN de la persona. A continuación, se detallan los aspectos de implementación de este script.

Como en el caso de FaceDet.py, en este script también necesitamos crear proxies con módulos del API NAOqi del robot. En este caso, necesitamos ALTextToSpeech, módulo que permite que el robot lea la pregunta en voz alta al usuario, ALSpeechRecognition, que permite recoger el número del uno al cinco con el que el usuario ha respondido, y por último, ALMemory, útil para extraer de la memoria del robot la palabra que ha reconocido mediante el sistema de reconocimiento del habla.

```
61  def main():
62      #Init variables globales
63      preg = 0
64      answers = []
65      IP = "192.168.1.9"
66
67      #Creacion proxies con Modulos NaoQi necesarios
68      #ALTextToSpeech, ALSpeechRecognition, #ALMemory
69      try:
70          global tts
71          tts = ALProxy("ALTextToSpeech",IP , 9559)
72      except Exception, e:
73          print "Error: x",e
74      try:
75          global asr
76          asr = ALProxy("ALSpeechRecognition",IP,9559)
77      except Exception, e:
78          print "Error: ",e
79      try:
80          global memory
81          memory = ALProxy("ALMemory",IP, 9559)
82      except Exception, e:
83          print "Error: ",e
84
```

Creación de proxies necesarios para oceanreduced.py

Tras crear los proxies, se pasa a configurar los sistemas de reconocimiento del habla y de transformación de texto en voz. En el caso de la transformación de texto a

voz, simplemente se indica que el idioma en el que va a hablar el robot es español y que lo haga con un volumen de 0.5, en el que 0 es el mínimo y 1 el máximo.

En cuanto al sistema de reconocimiento del habla, se siguen pasos parecidos, ya que se fija el idioma el cual NAO tiene que reconocer como español. Además, como se comentó en puntos anteriores, este sistema requiere saber qué palabras debe detectar. Para ello, debe usarse la instrucción setVocabulary del módulo ALSpeechRecognition pasando como argumento una lista de cadenas de caracteres, que serán las palabras que el robot será capaz de comprender. En este caso, su vocabulario son los números del 1 al 5, las posibles respuestas a las preguntas que se realizan en el test OCEAN, significando 1 totalmente en desacuerdo y 5 totalmente de acuerdo.

En la línea 97 se abre el archivo de texto que contiene las cincuenta preguntas del test, asignándolo a la variable archtest.

```
84
85      #Set idioma en que habla NAO
86      tts.setLanguage("Spanish")
87      tts.setVolume(0.5)
88      #Vocabulario a reconocer
89      vocabulary = ["uno","dos","tres","cuatro","cinco"]
90      #Pausa del sistema de reconocimiento de voz para configuracion
91      asr.pause(True)
92      asr.setLanguage("Spanish")
93      asr.setVocabulary(vocabulary,False)
94      asr.pause(False)
95      #Lectura fichero preguntas Test Ocean
96      global archtest
97      archtest = open("OceanEspanol.txt").read().split(".")
98      intro(preg,answers)
99
100     o = calcOCEAN(answers)
101     return o
```

Ajuste de parámetros y lanzamiento del test

En la línea 98 se lanza el método intro, que permite empezar el test, y se verá con detalle a continuación. Finalmente, cuando el test ha finalizado, se calcula el valor OCEAN del usuario que se devuelve mediante un *return* al script FaceDet.py, que se encargará de guardarlo en el archivo de texto de perfiles de personalidad.

4.3.1 Control de flujo

Para poder leer una pregunta detrás de otra, evitando colisiones y permitiendo que el robot esperara una respuesta del usuario, se realizó un control de flujo que se explica a continuación. Como ya se ha comentado, el método que permite comenzar el

test es intro. Este método recibe dos argumentos, el número de pregunta en el que nos encontramos y *answers*, un array que contendrá las respuestas del usuario. Este método solamente es invocado una vez, siendo el valor de preg igual a 0, que permitirá que el robot lea la introducción al test mediante la instrucción "tts.say(archtest[preg])", añadir un -1 al principio del array de respuestas e invocar el método nextQuestion que permite invocar nuevas preguntas.

El método nextQuestion es invocado tras terminar el método intro y el método askQuestion, ya que permite aumentar el índice del array de preguntas hasta que se llegue a la última. En caso de que este índice alcance un valor mayor a cincuenta, significará que ya se han hecho todas las preguntas, por lo que el robot dice en voz alta un mensaje de agradecimiento por pasar el test y termina la ejecución del mismo.

En caso de que el índice de la pregunta sea menor o igual que cincuenta, esto significará que todavía quedan preguntas por realizar. Si es así, se lanza el método askQuestion, que ordena al robot decir la pregunta correspondiente en voz alta, quedar a la espera de reconocer alguna de las respuestas que se incluyeron en el vocabulario, añadir esta respuesta al array *answers* y llamar de nuevo al método nextQuestion.

```python
import time
from naoqi import ALProxy

def intro(preg,answers):
    tts.say(archtest[preg])
    answers.append("-1")
    time.sleep(3)
    nextQuestion(preg,answers)

def askQuestion(preg,answers):
    tts.say(archtest[preg])
    asr.subscribe("ocean")
    time.sleep(5)
    asr.unsubscribe("ocean")
    data = memory.getData("WordRecognized")
    print(data)
    answers.append(data[0])
    nextQuestion(preg,answers)

def nextQuestion(preg,answers):
    preg+=1
    if preg<=50:
        askQuestion(preg,answers)
    else:
        tts.say("Gracias por responder a estas preguntas")
        calcOCEAN(answers)

```

Control de flujo oceanreduced.py

4.3.2 Cálculo valor OCEAN

Tras haber emitido todas las preguntas y recibido las respuestas a ellas, nextQuestion llama al método calcOCEAN, que toma las respuestas del usuario, almacenadas en el array *answers* y las transforma en números enteros.

Recordemos, que el vocabulario del robot consiste en una lista de cadenas de caracteres. Por lo tanto, en el array *answers* se habrán almacenado cadenas de caracteres. Para poder realizar cálculos aritméticos, necesitamos traducir esto a enteros. La solución es simple, ya que solo contamos seis opciones, los cinco números posibles y el -1, por lo que pueden utilizar estructuras condicionales *if-else* que comprueben qué número se encuentra en esa posición del *array* para después añadirlo al *array Answers*.

```
def calcOCEAN(answers):
    Answers = []
    for i in answers:
        if i==-1:
            Answers.append(-1)
        elif i=="uno":
            Answers.append(1)
        elif i=="dos":
            Answers.append(2)
        elif i=="tres":
            Answers.append(3)
        elif i=="cuatro":
            Answers.append(4)
        else:
            Answers.append(5)
    oceanValue = calcOCEANValue(Answers)
    return oceanValue

def calcOCEANValue(Answer):
    E = 20+Answer[1]-Answer[6]+Answer[11]-Answer[16]+Answer[21]-Answer[26]+Answer[31]-Answer[36]+Answer[41]-Answer[46]
    A = 14-Answer[2]+Answer[7]-Answer[12]+Answer[17]-Answer[22]+Answer[27]-Answer[32]+Answer[37]+Answer[42]+Answer[47]
    C = 14+Answer[3]-Answer[8]+Answer[13]-Answer[18]+Answer[23]-Answer[28]+Answer[33]-Answer[38]+Answer[43]+Answer[48]
    N = 38-Answer[4]+Answer[9]-Answer[14]+Answer[19]-Answer[24]-Answer[29]-Answer[34]-Answer[39]-Answer[44]-Answer[49]
    O = 8+Answer[5]-Answer[10]+Answer[15]-Answer[20]+Answer[25]-Answer[30]+Answer[35]+Answer[40]+Answer[45]+Answer[50]

    return E,A,C,N,O
```

Cálculo valor OCEAN

Tras esta conversión, puede llamarse al método calcOCEANValue que calcula el valor OCEAN mediante la fórmula que puede verse en la imagen.

4.4 Persona conocida por NAO

En los puntos previos se ha desarrollado el comportamiento que tiene el robot NAO en caso de no conocer a la persona con la que está tratando, lanzaría un test de personalidad que el usuario debería completar para obtener de él un pequeño perfil psicológico.

En el caso de que la persona ya se encuentre en la memoria del robot, es decir, ya haya realizado previamente el test y NAO haya aprendido las características de su cara y el nombre que aleatoriamente se le ha asignado, se puede pasar al siguiente paso, reconocer la emoción que tiene el usuario en ese instante.

Para conseguir esto, se hace uso de la cámara que integra NAO en su cabeza para tomar instantáneas y de la Emotion API de Microsoft, que permite detectar emociones en personas a partir de fotografías. Para tomar una fotografía con la cámara del robot y poder guardarla se ha desarrollado el script **takephoto.py**. Las llamadas al API de Microsoft, se ha realizan mediante el script **emotionAPI.py**.

4.4.1 Toma de instantáneas

El Emotion API de Microsoft permite detectar emociones en una imagen en la que hay personas, devolviendo mediante porcentajes en qué grado puede contar con una u otra emoción cada uno de los sujetos. Para más detalle, ver el capítulo de **Herramientas utilizadas,** en el que se desarrolla la forma en que este API funciona con mayor detalle.

Por lo tanto, existe la necesidad de que el robot tome una fotografía que posteriormente pueda ser enviada al API y permita reconocer las emociones del usuario o usuarios que están utilizándolo en ese momento.

En el propio API del robot, existe un módulo que permite tomar fotografías mediante la cámara que este integra en su cabeza, **ALPhotoCapture.** El inconveniente es que las instrucciones que ofrece este módulo, solo permiten guardar las fotografías tomadas en la memoria del robot. Este acceso a memoria es difícil de realizar, además podría ralentizar los tiempos de ejecución de la aplicación.

Por ello se tomó la decisión de utilizar otro módulo del API NAOqi, **ALVideoDevice**. Este módulo es el encargado de proveer, de una manera eficiente, de imágenes provenientes de la cámara a los módulos que pueden requerirlas, como **ALFaceDetection** o **ALVisionRecognition**. Es por lo tanto útil para nuestro fin, ya que en todo momento se encarga de manejar la cámara, y permite que, tomando un *frame* en un determinado instante del vídeo, podamos obtener una imagen y guardarla en el ordenador desde el que se ejecuta la aplicación.

Para realizar esto se desarrolló el script Python takephoto.py. En el método *main* del código se realiza toda la funcionalidad. En primer lugar, como en todos los scripts que ya se han desarrollado en este proyecto, se crea un proxy con el módulo ALVideoDevice, que va a ser el utilizado para tomar la imagen.

Posteriormente, en las líneas 24 y 25 se crean las variables *resolution* y *colorSpace*, que servirán como parámetros de ajuste para posteriormente suscribirnos al servicio de video de NAO. Estar suscritos a este servicio, permite que posteriormente podamos realizar la llamada a getImageRemote, que devuelve la última imagen tomada por la cámara de video y realiza transformaciones que hagan que esta sea devuelta en el formato requerido. Finalmente, nos desuscribimos del servicio.

```
14  def main():
15      IP = "192.168.1.9"
16      PORT=9559
17      try:
18          #Suscripcion ALVideoDevice
19          global video
20          video = ALProxy("ALVideoDevice",IP,9559)
21      except Exception, e:
22          print "Error: ",e
23
24      resolution = 2    # VGA
25      colorSpace = 11   # RGB
26
27      videoClient = video.subscribe("python_client", resolution, colorSpace, 5)
28
29      t0 = time.time()
30
31      naoImage = video.getImageRemote(videoClient)
32
33      t1 = time.time()
34
35      # Time the image transfer.
36      print "acquisition delay ", t1 - t0
37
38      video.unsubscribe(videoClient)
39
```

Toma de la imagen mediante ALVideoDevice

El siguiente paso necesario tras haber realizado la toma de la imagen es guardar esta en el ordenador desde el que se ha invocado al script principal FaceDet.py. Para ello se utiliza el módulo ImageDraw de Python, que permite crear imágenes o modificarlas, con el fin sobre todo de que estas sean utilizadas para web.

```
40
41      # Utilizando ImageDraw convertimos la imagen (un array) a PNG
42
43      imageWidth = naoImage[0]
44      imageHeight = naoImage[1]
45      array = naoImage[6]
46      image_string = str(bytearray(array))
47
48      # Crear imagen PIL a partir del array de datos
49      im = Image.frombytes("RGB", (imageWidth, imageHeight), image_string)
50
51      # Guardar imagen
52
53      im.save("emocion.png", "PNG")
54
55      im.show()
```

Transformación de la imagen y guardado en formato PNG

La instrucción getImageRemote de ALVideoDevice, nos devuelve la imagen como un array de bytes, que contienen la información relativa a la imagen. Para poder transformarla en un formato comprensible para el API de Microsoft, la transformamos primero a una imagen de tipo Image mediante la instrucción frombytes, que el módulo ImageDraw es capaz de comprender.

Hecho esto, no queda más que guardarla en el ordenador en el formato deseado, en este caso PNG y con el nombre de archivo elegido.

Tras haber sido ejecutado este script, contaremos con una imagen de la cara de la persona o personas con las que el robot está tratando guardada en nuestro ordenador. Ahora sí, podemos enviar una llamada al API de Microsoft que devuelva la emoción o emociones que están sintiendo.

4.4.2 Interacción con Emotion API

El último paso para detectar la emoción es enviar la imagen que hemos obtenido al API. Para poder utilizarlo es necesario obtener un "subscription key" que permita que hagamos llamadas al API. Es gratuito realizar hasta cinco llamadas por minuto con un *key.*

Para poder realizar una llamada con el protocolo http al Emotion API, necesitaremos primero especificar las cabeceras 'Content-type', que se refiere al tipo de imagen que vamos a enviar, en nuestro caso, es de tipo application/octet-stream, lo que permite enviar imágenes guardadas como archivo en nuestro ordenador, si por el contrario queremos pasar una URL de una imagen de internet, utilizaremos application/json.

```
1  import httplib, urllib, base64, ast, time, json
2
3  # Establecimiento parametros llamada a Emotion API
4  headers = {
5      #octect-stream permite que la imagen provenga de un archivo y no URL
6      'Content-type': 'application/octet-stream',
7      'Ocp-Apim-Subscription-Key': '721d2ad21d4a4106bd80185d091f982a ',
8  }
9
10 params = urllib.urlencode({
11 })
12
```

Establecimiento parámetros llamada al API de Microsoft

El otro parámetro de la cabecera es 'Ocp-Apim-Subscription-Key', que permite enviar al API nuestro "subscription key".

Establecidos los parámetros, se ejecuta el método detEmotion, que ha sido invocado desde el script principal FaceDet.py. Este toma como argumento la ruta de la imagen en nuestro ordenador. La primera acción que realiza es abrirla y asignar a la variable *body* la lectura de la misma.

Una vez está la imagen contenida en la variable *body*, se realiza una llamada http al API de Microsoft, mediante un mensaje de tipo POST.

En *response* se obtiene la respuesta devuelta por el API, que posteriormente es leída en la variable *data*.

Debido a que la respuesta es obtenida en formato JSON, es necesario transformarla a un array Python que permita tratarla más fácilmente. Para ello se utiliza la instrucción *loads* de la librería json.

Para retornar de una manera más ordenada los valores de cada una de las emociones que se han obtenido tras la llamada, estas se dividen entre el array "emotion_names", que contiene los nombres de cada una de las emociones que el API puede detectar (*anger, contempt, disgust, fear, happiness, neutral, sadness, surprise*) y "emotion_scores", que contiene el grado en el que la persona cuenta con cada una de las emociones.

Estos *arrays* son devueltos a FaceDet.py, que los trata de la forma en la que se comentó en puntos anteriores.

```
14  def detEmotion(image):
15      f = open(image, "rb")
16      body = f.read()
17
18      try:
19          i = time.time()
20          conn = httplib.HTTPSConnection('westus.api.cognitive.microsoft.com')
21          conn.request("POST", "/emotion/v1.0/recognize?%s" % params, body, headers)
22          response = conn.getresponse("")
23          data = response.read()
24          conn.close()
25
26          response = json.loads(data)
27          scores = response[0]['scores']
28          anger = scores['anger']
29          contempt = scores['contempt']
30          disgust = scores['disgust']
31          fear = scores['fear']
32          happiness = scores['happiness']
33          neutral = scores['neutral']
34          sadness = scores['sadness']
35          surprise = scores['surprise']
36
37          emotion_names = ["Anger", "Contempt", "Disgust", "Fear", "Happiness", "Neutral", "Sadness", "Surprise"]
38          emotion_scores = [anger, contempt, disgust, fear, happiness, neutral, sadness, surprise]
39
40          f = time.time()
41          return emotion_names, emotion_scores
```

Obtención emociones mediante llamada al API de Microsoft

4.5 Reacciones del robot a las distintas emociones detectadas

Dependiendo de la emoción que el Emotion API de Microsoft devuelva como más probable que el usuario esté sintiendo en ese momento, el robot realizará unas acciones u otras intentando imitar la emoción detectada.

Como se ha comentado anteriormente, esto puede realizarse mediante *behaviors* de NAO, que pueden ser administrados mediante el módulo ALBehaviorManager del API NAOqi. Debido a que el robot no cuenta con una cara o pantalla que permita reproducir la emoción que el robot puede sentir tras detectar la que siente el usuario, ésta se simula mediante **posturas corporales**.

En primer lugar, deberán instalarse todos los *behaviors* en el robot, para que en caso de ser necesario puedan ser ejecutados. Esto puede realizarse mediante la instrucción installBehavior() del módulo **ALBehaviorManager**.

```
56          managerProxy.installBehavior("./behaviors/sorpresa.crg")
57          managerProxy.installBehavior("./behaviors/tristeza.crg")
58          managerProxy.installBehavior("./behaviors/ira.crg")
59          managerProxy.installBehavior("./behaviors/miedo.crg")
60          managerProxy.installBehavior("./behaviors/alegria.crg")
61          managerProxy.installBehavior("./behaviors/desprecio.crg")
62          managerProxy.installBehavior("./behaviors/aversion.crg")
```

Instalacion de los siete archivos de comportamiento

Tras haber recibido respuesta del Emotion API de Microsoft y haber extraído aquella con mayor probabilidad de la manera que se comentó en el punto anterior, se lanza el *behavior* asociado a esta emoción mediante la instrucción runBehavior(). En la imagen solamente se muestran dos casos debido a que el resto son idénticos. La variable “corriendo” nos permitirá saber qué *behavior* está ejecutándose para poder parar su ejecución después.

```
104                 corriendo = ""
105                 if emocionmax=="Surprise":
106                     managerProxy.runBehaviour("sorpresa")
107                     corriendo = "sorpresa"
108                 elif emocionmax=="Sadness":
109                     managerProxy.runBehaviour("tristeza")
110                     corriendo = "tristeza"
```

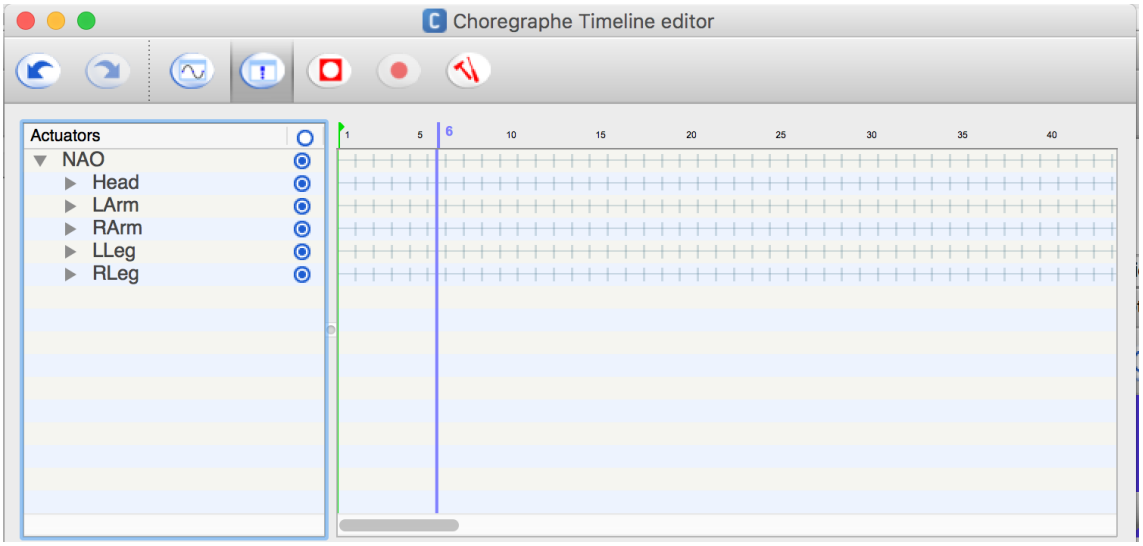
Lanzamiento del *behavior* asociado a la emoción detectada

Una vez el *behavior* en cuestión haya terminado, este se para mediante la instrucción stopBehavior(), comprobando antes si realmente este se está ejecutando para evitar excepciones.

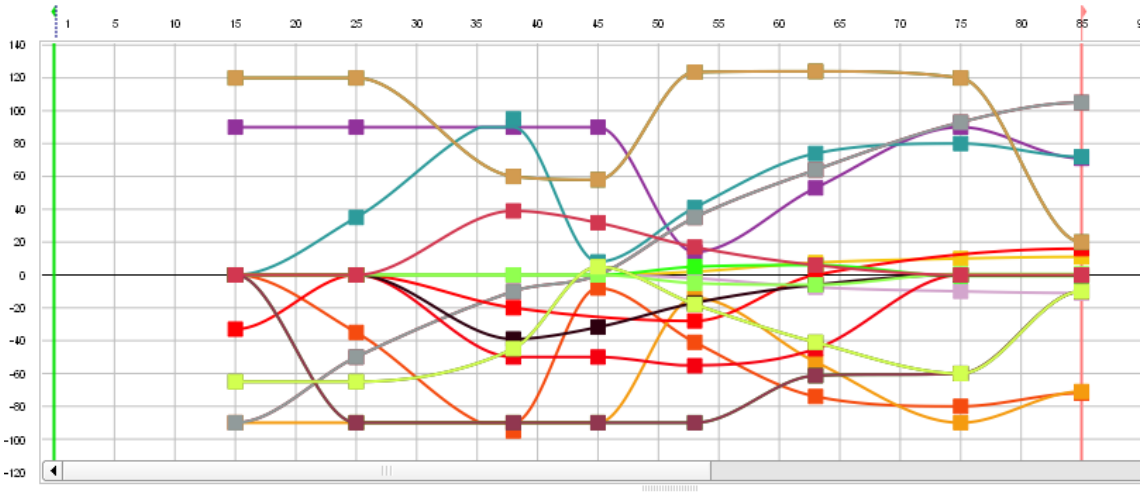
```
129
130                 if (managerProxy.isBehaviorRunning(corriendo)):
131                     managerProxy.stopBehavior(corriendo)
132                     time.sleep(1.0)
133
```

Detención de la ejecución previa comprobación

Para poder crear estas poses en las que NAO mueve los brazos, el tronco, la cabeza, etc., se ha utilizado el **modo grabación** que se encuentra en el software de simulación **Choregraphe** desarrollado por Aldebaran Robotics. Este modo, permite realizar movimientos de las articulaciones del robot y que estos queden grabados en una **línea de tiempo** para después poder ser reproducidos tantas veces como se desee. Es una manera sencilla de hacer que el robot se mueva sin tener que utilizar código ni realizar cálculos sobre la estabilidad que tendrá NAO realizando ciertos movimientos.



Vista del modo grabación de Choregraphe



Vista de la línea de tiempo tras una grabación

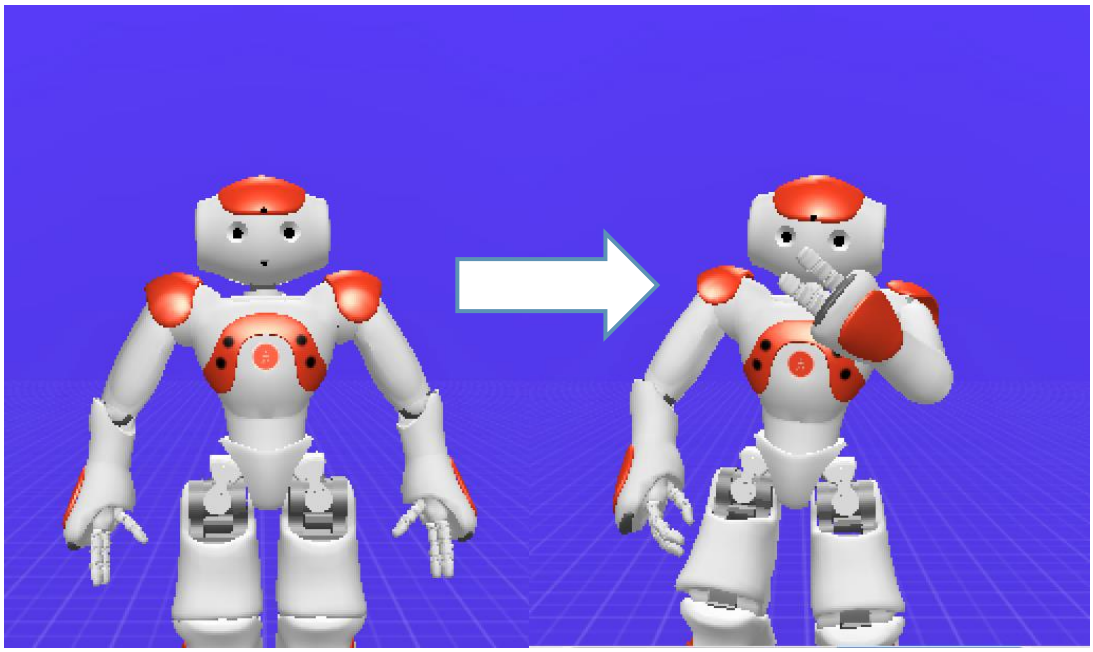
De las ocho emociones que el API de Microsoft es capaz de detectar, se ha implementado una respuesta específica de NAO para siete de ellas, que son:

- Sorpresa (“Surprise”)
- Tristeza (“Sadness”)
- Rabia (“Anger”)
- Miedo (“Fear”)
- Alegría (“Happiness”)
- Repulsión/Aversión (“Disgust”)
- Desprecio (“Contempt”)

4.5.1 Reacción a la emoción sorpresa

Cuando NAO detecta mediante el API de Microsoft que la emoción más probable que puede estar teniendo el usuario sea la sorpresa, este adoptará una postura corporal que puede interpretarse como de sorpresa.

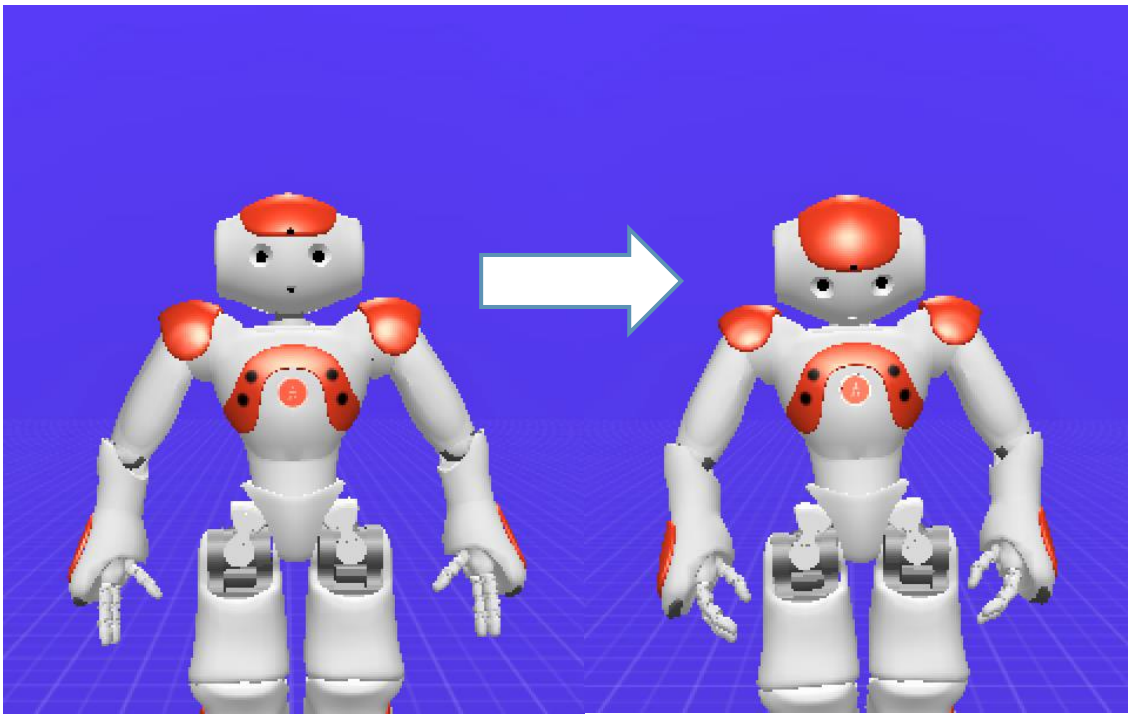
Partiendo de una postura inicial, de pie en su posición predeterminada, se pasa a una postura en la que NAO se inclina un poco hacia un lado tapándose la boca con su mano izquierda, transmitiendo así la sensación de incredulidad o sorpresa.



En la izquierda, NAO en la postura inicial. Derecha, NAO con la postura corporal de sorpresa

4.5.2 Reacción a la emoción tristeza

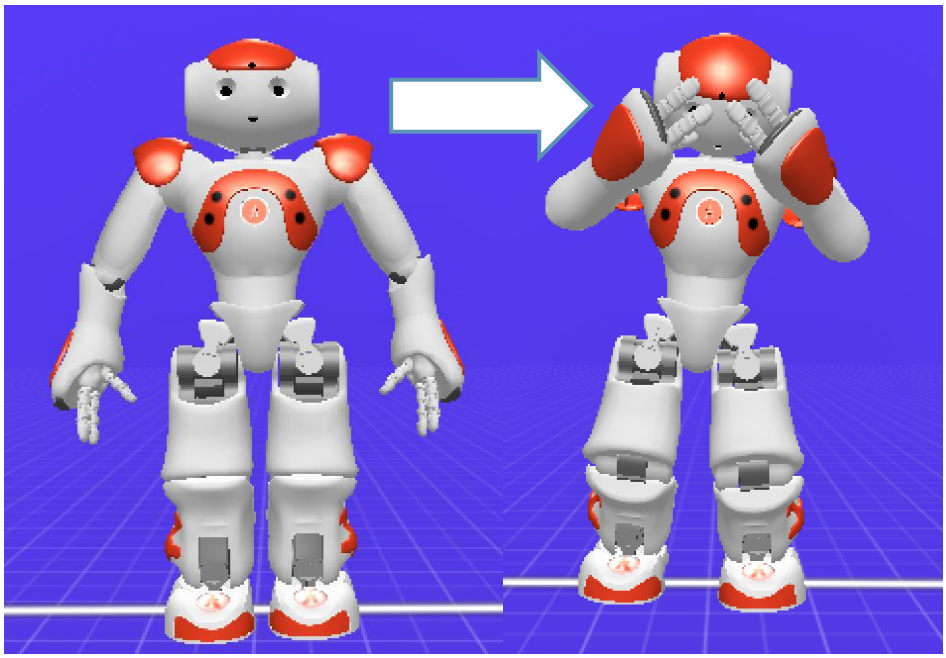
Otra de las emociones para las que se ha implementado una reacción es la de tristeza. Cuando NAO interprete que el usuario con el que está interactuando se siente triste, es decir, cuando el Emotion API la devuelva como la emoción que se está produciendo en la imagen con mayor probabilidad, este adoptara una postura que sugiera tristeza. Esto puede simularse con NAO haciendo que agache la cabeza, signo interpretado en psicología como de tristeza o aflicción.



En la izquierda, NAO en la postura inicial. Derecha, NAO con la postura corporal de tristeza

4.5.3 Reacción a la emoción miedo

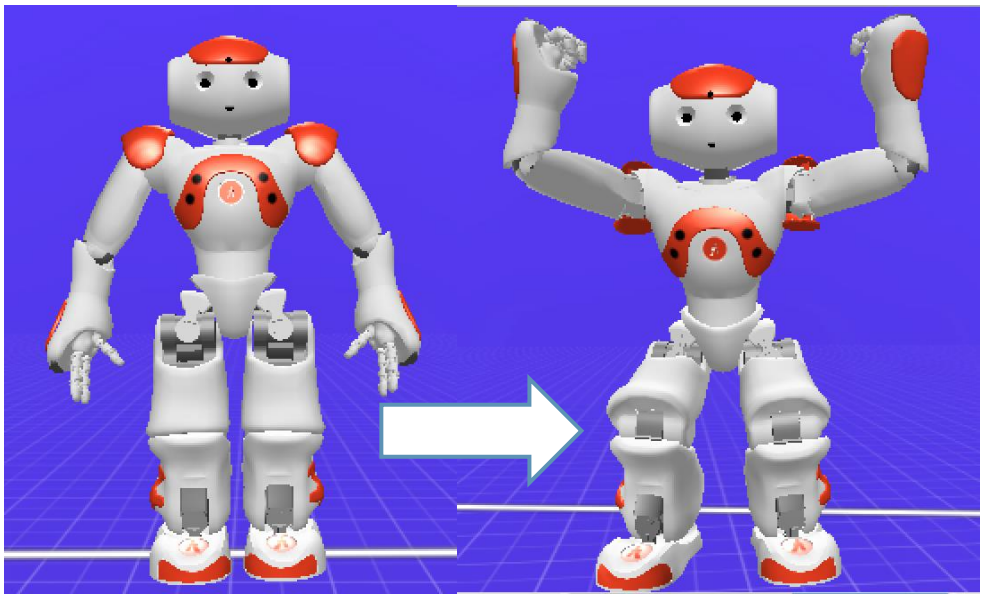
Al detectar NAO que la emoción que está sintiendo el usuario es miedo, este adoptara también una postura que transmita que tiene miedo. No olvidemos que el hecho de que NAO imite en la medida de lo posible (ya que solo puede hacerlo con el cuerpo al no contar con cara), las emociones que tiene el humano con el que interactúa, puede hacer que el robot sea visto como un agente más humano y cercano. Así pues, una persona que sienta miedo por una situación en la que también se encuentra el robot, puede sentirse acompañada y comprendida.



En la izquierda, NAO en la postura inicial. Derecha, NAO con la postura corporal de miedo

4.5.4 Reacción a la emoción alegría

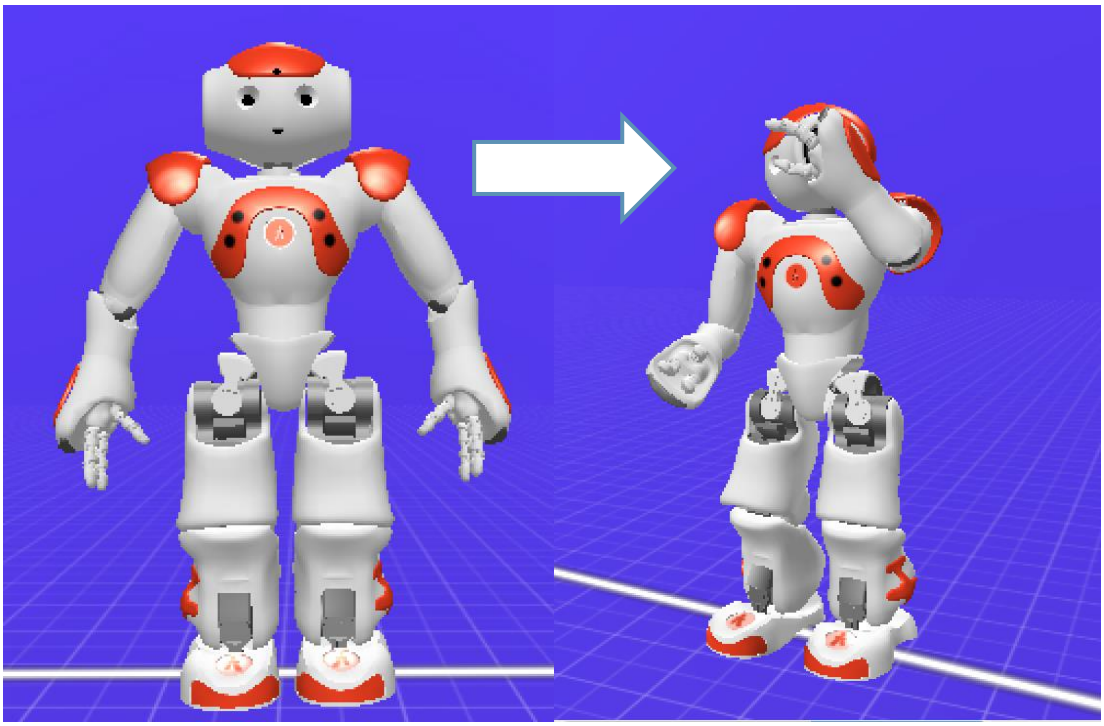
Los movimientos que indican alegría en robots humanoides son ampliamente utilizados, así como los de baile, que permiten investigar en estabilidad y movimientos de su cuerpo. Consisten en pequeñas coreografías en las que se mueven sobre todo las articulaciones de los brazos en un intento de vitoreo que expresa alegría. Así pues, se ha optado por hacer que NAO alce los brazos con los puños cerrados en señal de victoria, agachándose y levantándose repetidamente.



En la izquierda, NAO en la postura inicial. Derecha, NAO con la postura corporal de alegría

4.5.5 Reacción a la emoción repulsión

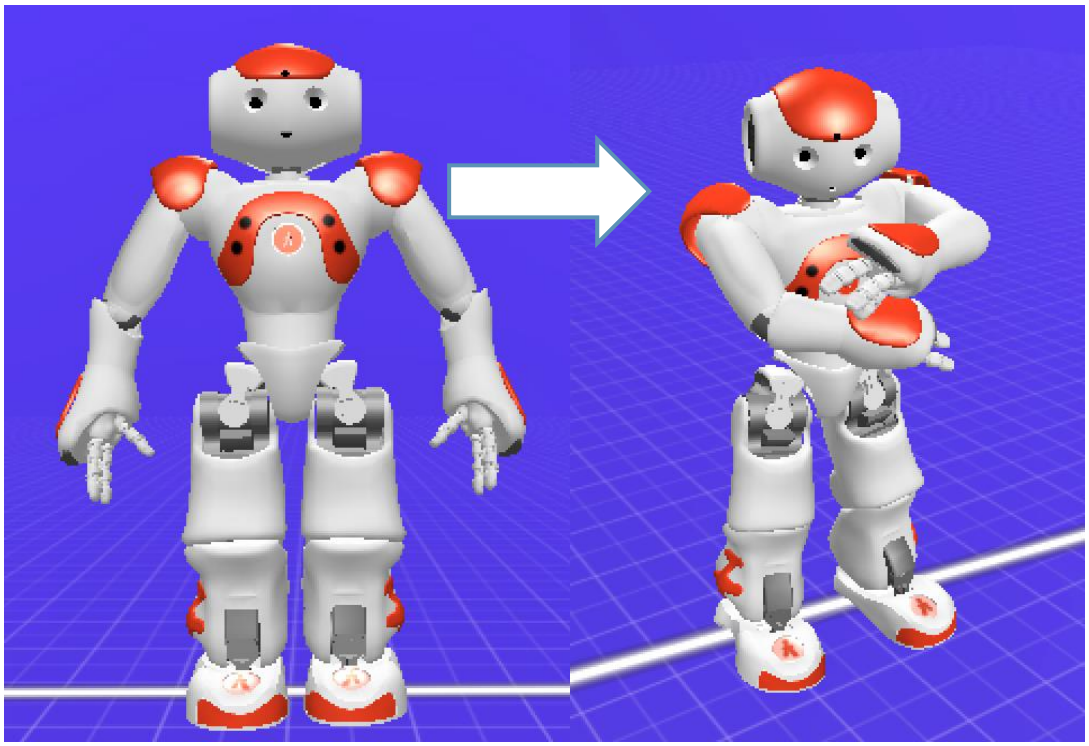
Cuando NAO detecta que la persona con la que interactúa está sintiendo repulsión o aversión hacia algo, adopta una postura corporal que sugiere lo mismo, ladea la cara, intentando no mirar y llevándose la mano a la cara intentando taparse los ojos. Así se consigue que de la sensación de que NAO está intentando evitar mirar aquello hacia lo que siente repulsión.



En la izquierda, NAO en la postura inicial. Derecha, NAO con la postura corporal de repulsión

4.5.6 Reacción a la emoción desprecio

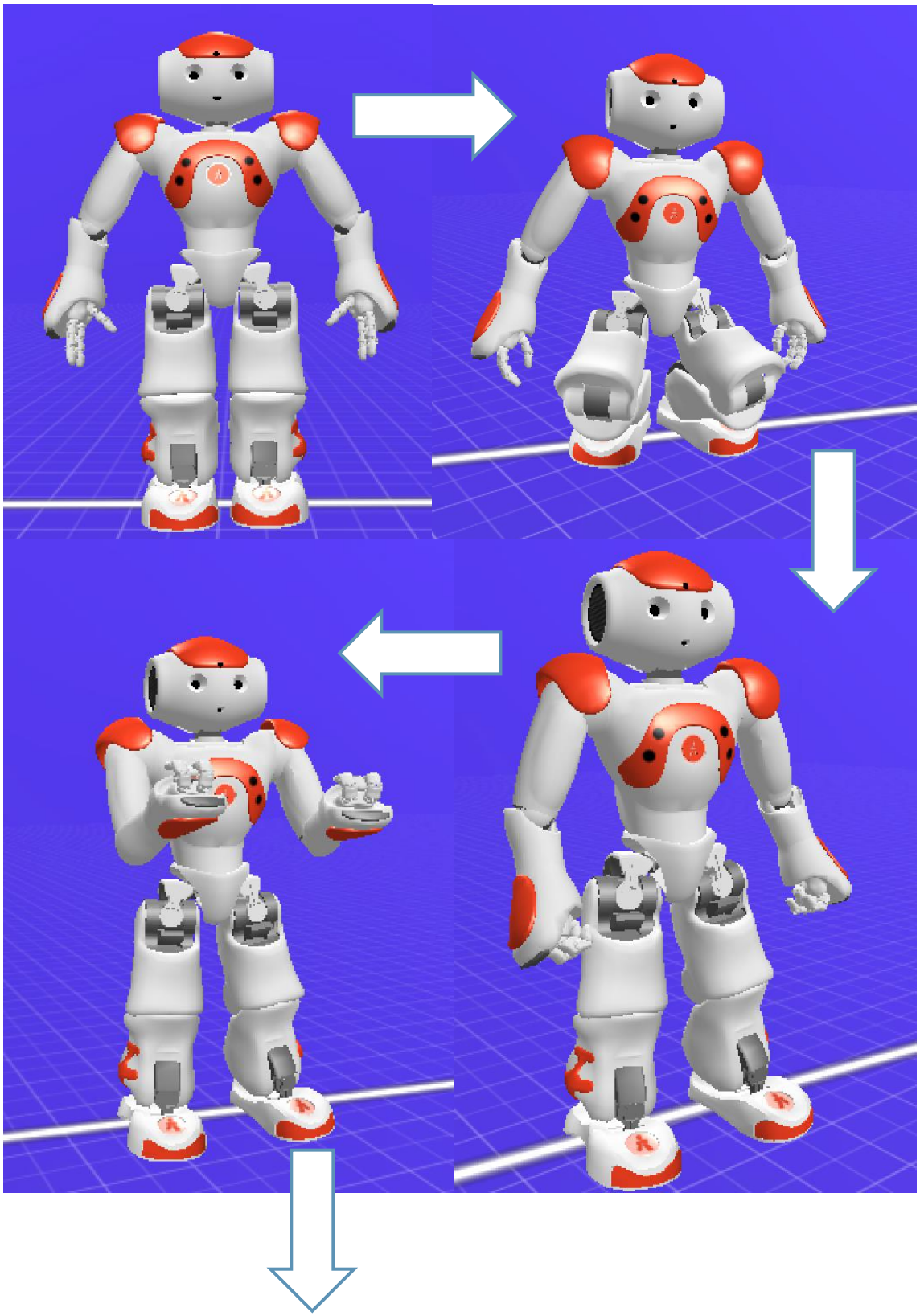
Otra de las emociones que nos permite detectar el Emotion API es el desprecio. En caso de que esta se la que está sintiendo la persona, NAO también adoptará una actitud de desprecio, cerrando los brazos en señal de no aceptación y distanciamiento.

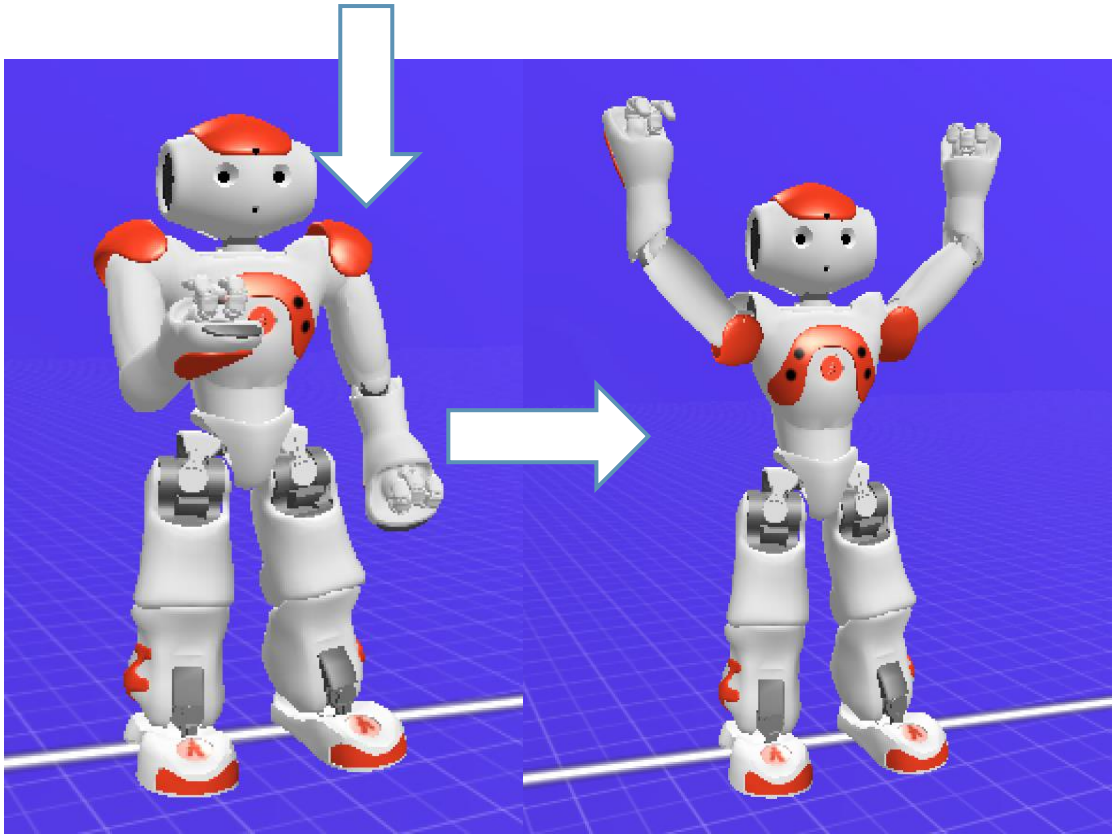


En la izquierda, NAO en la postura inicial. Derecha, NAO con la postura corporal de desprecio

4.5.7 Reacción a la emoción ira

En último lugar se encuentra la emoción ira. No es casualidad que esta se encuentre en esta posición en la explicación de reacciones, ya que es con diferencia la más difícil de imitar contando solo con la postura corporal. La ira se expresa mayoritariamente mediante las cejas y los labios. Cuando alguien se encuentra enfadado normalmente los frunce. Al no contar NAO con una cara con la cual pudiéramos expresar esta emoción, se ha optado por añadir un **huevo de pascua** (*Easter Egg*) en el trabajo. Si el robot detecta que la persona con la que está interactuando se encuentra enfadada, este dirá: “Cuando estoy enfadado me gusta hacer deporte, me ayuda calmarme”. Posteriormente el robot se agacha a recoger unas mancuernas imaginarias del suelo que pasa a levantar posteriormente, creando el efecto de que está ejercitando sus músculos. Para finalizar, alza los brazos en gesto de victoria. A continuación, se detallan los pasos de la animación mediante imágenes.





Reacción de NAO al detectar la emoción ira

4.6 Pruebas

Tras implementar todo el código mediante scripts Python y comportamientos del robot en Choregraphe, se pasa a las pruebas de funcionalidad.

Como es lógico, en cada parte de la funcionalidad desarrollada, se realizaron pruebas que permitieran determinar si realmente el código realizaba la función desarrollada. Por ejemplo, el script que implementa la realización del test de personalidad OCEAN, se hicieron tests del reconocimiento del habla implementado y del correcto cálculo del valor OCEAN.

Para hacer las pruebas de un modo más ágil, se implementó un script que permitiera pasar un test OCEAN reducido de solamente diez preguntas, pudiendo así lanzar el test en repetidas ocasiones.

En cuanto al reconocimiento facial, en el que NAO debe determinar si conoce o no a la persona con la que está tratando, se realizó otro script de prueba cuya funcionalidad era únicamente decir el nombre de la persona en alto cuando la conociese y decir “No te conozco” en caso contrario. Se encontraron problemas en el reconocimiento debido a la sensibilidad que tiene el sistema a la luz ambiental. Si la

imagen que NAO conseguía era oscura o estaba muy iluminada, el robot no era capaz de reconocer al usuario a pesar de que este se encontrara en la base de datos.

El reconocimiento de emociones pudo probarse con una batería de imágenes encontradas en internet que habían sido previamente etiquetadas con la emoción que la persona sentía en ellas. Tras enviar todas ellas al API de Microsoft y extraer la emoción con mayor probabilidad, se comparó y se obtuvo que efectivamente, el código de reconocimiento de emociones funcionaba correctamente.

Las posturas corporales que NAO adopta como reacción a la emoción que detecta en el usuario, fueron probadas en el entorno de simulación Choregraphe, que permite una mayor agilidad.

Tras probar toda la funcionalidad por separado y comprobar que cada componente de esta funcionaba, se procedió a realizar las pruebas uniendo todos los módulos implementados, obteniendo resultados satisfactorios.

En ocasiones, la posición en la que el usuario se sienta frente a NAO dio problemas. Es necesario para un reconocimiento facial correcto que la persona con la que NAO está interactuando, se encuentre de frente a él y mirándolo, en caso contrario el usuario no será reconocido. También se encontraron problemas con la luz ambiental como se ha comentado anteriormente, todos ellos fácilmente solucionables.

Es por esto que puede decirse que la funcionalidad que en un principio se planteó desarrollar para este trabajo se ha realizado satisfactoriamente.

5. Conclusiones

Finalizada esta memoria podemos concluir que los objetivos que se plantearon para este trabajo de fin de grado se han cumplido satisfactoriamente. El primero, desarrollar un caso de estudio que permitiera que un robot humanoide fuera capaz de interactuar con un ser humano haciendo uso de las emociones, ha sido cumplido como ha podido verse en el capítulo de **funcionalidad desarrollada.** Tras diseñar la aplicación con los diferentes pasos a llevar a cabo, se pasó al desarrollo de código. Este supuso un gran esfuerzo debido primero, al periodo de aprendizaje que se necesitó para programar el robot y segundo, a que algunos de los problemas no tuvieron una solución trivial, como por ejemplo, tomar una instantánea con las cámaras que contiene NAO en su cabeza.

El segundo objetivo planteado fue dotar al robot de reacciones a las emociones detectadas. Esto se ha conseguido haciendo que NAO adopte diferentes posturas corporales como reacción a la emoción del usuario.

El siguiente objetivo fue que NAO, fuera capaz de emitir una batería de preguntas que permitieran obtener el perfil psicológico del usuario. Esta funcionalidad ha sido implementada en el script oceanreduced.py, como se comentó en el punto 4.3.

Los tres últimos objetivos que se plantearon, fueron los de hacer de este trabajo una base no solo práctica, sino también teórica de la que fuera posible partir para futuros trabajos sobre esta materia. Es por ello, que se ha incluido una gran cantidad de información sobre teoría de psicología social, útil para aquellos que pretendan realizar una aplicación que tenga en cuenta las emociones del usuario. Es importante en cualquier trabajo de investigación en ingeniería y cualquier otra ciencia, partir con los conocimientos asentados sobre lo que se quiere desarrollar, teniendo en cuenta las disciplinas relacionadas. Podemos decir que estos objetivos, también se han cumplido. Tomando como referencia artículos y libros de psicólogos que han dedicado gran parte de su investigación a lo social, hemos podido reunir en pocas páginas parte de la teoría necesaria sobre emociones y conducta de ayuda para el desarrollo de este trabajo.

5.1 Trabajo futuro

Sin lugar a dudas, tras haber realizado este trabajo, se hace más evidente la necesidad de que en un futuro, muchas aplicaciones informáticas incorporen emociones en su funcionalidad.

Puede hablarse de que las personas van a tender a ser solitarias, y como animal social, el ser humano necesita interaccionar. Si un robot o cualquier aplicación cuenta con emociones, una persona forzada a encontrarse sola, podrá contar con apoyo y compañía. Estas afirmaciones pueden parecer extraídas de una película o libro de ciencia ficción, en el que los humanos dejan de comunicarse con otros humanos para pasar a hacerlo solo con las máquinas. Y puede que algún día lleguemos a ese punto. No obstante, en estos momentos muchas personas mayores se encuentran solas, y podrían hacer uso de tecnologías como esta.

No solo puede utilizarse la emoción como forma de que un robot pueda dar compañía a una persona. Estudios recientes, han desarrollado aplicaciones que permiten a los psicólogos tratar a niños con ciertos tipos de autismo utilizando robots y simulación de emociones.

Es por ello que este trabajo puede ser continuado, desarrollando en el robot humanoide conductas de ayuda proporcionadas a un ser humano que satisficiesen su necesidad de apoyo. En principio, podría conseguirse investigando de qué modo un robot debe dirigirse a una persona que está sintiendo una determinada emoción.

6. Bibliografía

ACINAS ACINAS, Mª.P. *Habilidades de comunicación y estrategias asistenciales en el ámbito sanitario.* Primera edición. Formación Alcalá, 2004. ISBN: 8496224015

ANNÉ, S.; BLUM, D.; ABAL, F.; LOZZIA, G.S.; ATTORRESI, H.F. La conducta prosocial: estado actual de la investigación. En: *Perspectivas en psicología, Vol. 11, Nº2, pp. 21-33*. Noviembre 2014.

ARGYLE, M. *Social interaction.* Primera edición. Londres: Methuen, 1969. ISBN: 9780202368993

ARGYLE, M. *Psicología del comportamiento interpersonal.* Primera edición. Madrid: Alianza, 1978. ISBN: 9788420622026

ASIMO Specifications | ASIMO Innovations by Honda. [consulta: 12 mayo 2017]. Disponible en: http://asimo.honda.com/asimo-specs/

BARRIENTOS, A. *Fundamentos de robótica.* Segunda Edición. S.A. Mcgraw-Hill / Interamericana de España, 2007. ISBN: 9788448156367

BENSON, P.L.; SCALES, P.C.; HAMILTON, S.F. y SESMA, A. Positive youth development: Theory, research and applications. En: *R.M. Lerner (Ed.), Theoretical models of human development. Handbook of child psychology (pp. 894-941).* Hoboken, NJ: Wiley. 2006.

BATSON, C.D. y POWELL, A. *Altruism and prosocial behavior.* En: M. Theodore (Ed.) y L. Melvin (Ed.) *Handbook of psychology: Personality and Social Psychology, (5).* Nueva York: Wiley y Sons, Inc. XIX.

BOTTENBERG, E. H. Phenomenological and Operational Characterization of Factor-Analytically Derived Dimensions of Emotion. En: *SAGE Journals*. Diciembre 1975 (Traducción del texto: *Bottenberg E. H. Emotionspsychologie. Muenchen: Goldmann, 1972*)

CABALLO, V.E. *Teoría, evaluación y entrenamiento de las habilidades sociales.* Primera edición. Valencia: Editorial Promolibro, 1988. ISBN: 8486201842

DAVIS, F. *La comunicación no verbal.* Octava edición. Alianza Editorial, 1983. ISBN: 8420616168

Diccionario Real Academia Española. *DEL: Lista de entradas – Diccionario de la lengua española – Edición del Tricentenario.* [Varias fechas de consulta]. Disponible en: http://dle.rae.es/

EISENBERG, N. y FABES, R. Prosocial development. En: *W. Damon y N. Eisenberg (Ed.), Handbook of child psychology: vol. 3. Social, emotional and personality development (pp. 701-778)* Nueva York: Wiley, 1998.

EKMAN, P. *Como detectar mentiras.* Primera edición. Ediciones Paidós Ibérica, 1992. ISBN: 8475097251

EKMAN, P. y FRIESEN, W. V. *Unmasking the face.* Primera edición. Prentice-Hall, 1975. ISBN: 978-1883536367

GONZÁLEZ PORTAL, M.D. *Conducta prosocial: evaluación e intervención*. Primera edición. Ediciones Morata, 1992. ISBN: 8471123665

GONZÁLEZ PORTAL, M.D. *Conducta prosocial: evaluación e intervención*. Ediciones Morata, 2000. ISBN: 8471123665

GOUAILLIER, D.; HUGEL, V.; MONCEAUX, J. y MAISONNIER, B. Mechatronic design of NAO humanoid. En: *Proceedings – IEEE International Conference on Robotics and Automation*. Junio 2009.

HOLMGREN, R.; EISENBERG, N. y FABES, R. The relations of children's situational empathy-related emotions to dispositional prosocial behavior. En: *International Journal of Behavioral Development, 22 (1), pp. 171-173.*

HOWARD, J. A., & PILIAVIN, J. A. *Altruism*. Encyclopedia of Sociology. New York: Macmillian, 2000.

IZARD, C.E. *Human emotions.* Nueva York: Plenum, 1977. ISBN: 978-1-4899-2211-3

KNAPP, M.L. *La comunicación no verbal. El cuerpo y el entorno.* Barcelona: Paidós, 1988. ISBN: 9788475091853

LATANÉ, B. y DARLEY, J.M. Group inhibition of bystander intervention. En: *Journal of personality and Social Psychology, 10, 215-221.* 1968.

LÓPEZ, F. *Para Comprender la Conducta Altruista*. Navarra: Verbo Divino, 1994.

LUTZ, M. *Learning Python*. Quinta Edición. O'Reilly, 2013. ISBN: 978-1-449-35573-9

MALLO, Mª J.; FERNÁNDEZ, DOLS J.M. y WALLBOTT, H. Reconocimiento de emociones a partir de la expresión y el contexto: Una réplica. En: *Revista de psicología social*. 1989

MARROQUÍN PÉREZ, M. y VILLA SÁNCHEZ, A. *La comunicación interpersonal.* Primera edición. Ediciones Mensajero, 1995. ISBN: 8427119526

MEHRABIAN, A. Inference of attitudes from the posture, orientation and distance of a comunicator. En: *Journal of Consulting and Clinical Psychology,* 32, 296-308. 1968.

MEHRABIAN. A. *Nonverbal communication.* Aldine-Atherton, 1972.

Microsoft Emotion API – *Emotion API: Detección de expresiones emocionales* | *Microsoft Azure.* [consulta: 11 mayo 2017]. Disponible en: https://azure.microsoft.com/es-es/services/cognitive-services/emotion/

MONCEAUX, J. y MAISONNIER, B. Choregraphe: a Graphical Tool for Humanoid Robot Programming. En: *IEEE Xplore*. Noviembre 2009.

MUNNÉ, F. *Psicología social.* Tercera edición. Ediciones CEAC, 1986. ISBN: 8432988049

NAOqi APIs – Aldebaran 2.1.4.13 documentation. [consulta: 9 mayo 2017]. Disponible en: http://doc.aldebaran.com/2-1/naoqi/

PADES, A. *Habilidades sociales en enfermería: Propuesta de un programa de intervención.* Universitat de les Illes Balears: Departament de Psicología. 2003.

PENNEBAKER, J. Putting stress into word. Health, linguistic, and therapeutic implications. En: Behaviour Research and Therapy, 31, 539-548, 1993.

ROCHE, R. *Educación prosocial de las emociones, actitudes y valores en la adolescencia.* Barcelona: UAB-LIPA, 1997.

RUSSELL, L. y NORVIG, P. *Inteligencia Artificial. Un Enfoque Moderno.* Segunda Edición. Pearson Prentice Hall, 2004. ISBN: 978-84-205-4003-0

SARBIN, T.R Y HARDYCK, C.D. Contributions to role-taking theory: role perception on the basis of postural cues. Unpublished. 1953. Citado por Argyle, M. en 1979.

SCHMIDT-ATZERT, L. *Psicología de las emociones.* Primera Edición. Herder, 1985. ISBN: 9788425414534

TRULL, T. y WIDIGER, T. Dimensional models of personality: the five-factor model and the DSM-5. En: *Pubmed*. Junio 2013.

TURING, A. Computing machinery and intelligence. En: *Mind 59(236).* Enero 1950.

WAINWRIGHT, G.R. *El lenguaje del cuerpo.* Primera edición. Ediciones Pirámide, 2000. ISBN: 8436811968

WARWICK, K. y SHAH, H. Passing the Turing test does not mean the end of humanity. En: *Cognitive Computation*. Diciembre 2015.