Universidad Carlos III de Madrid

Escuela Politécnica Superior

Grado en Ingeniería de Sistemas Audiovisuales

Trabajo Fin de Grado

Robot NAO Cantante

Autor: Adrián Caballero Pamos

Tutor: Nerea Luis Mingueza

Marzo de 2016

Resumen

En los últimos años la robótica ha experimentado un crecimiento exponencial incorporando todo tipo de funcionalidades. Introducir el mundo musical en los robots es una de ellas. En este Trabajo Fin de Grado se presenta el desarrollo de un sistema que permite al robot NAO leer una partitura, analizarla y reproducirla a modo de canto. La finalidad del trabajo es que el robot actúe como un intérprete frente a una partitura musical tal y como lo haría un humano. Debe ser capaz de interpretar cualquier melodía que se le presente debido a sus conocimientos generales musicales.

El proyecto ha sido desarrollado en lenguaje Python y se ha empleado el software Choregraphe como herramienta de soporte para el robot. El proyecto se ha llevado a cabo con un robot NAO de la Universidad Carlos III de Madrid y ha sido probado en el laboratorio del grupo de trabajo PLG (Planning and Learning Group)

Abstract

In recent years, robotics has grown exponentially incorporating different kinds of features. The entry of the robots in the musical world as singers and players are some examples of it.

In this Final Project will be developed a system that allows the robot NAO to read music, analyze and play different songs, even if the robot has not seen the music sheet before. The aim of this work is to get the robot NAO to sing a music sheet as a human singer, interpreting music with feelings. Robot NAO must be able to sing a melody due to its new musical knowledge, which it has learnt in this project.

The code has been developed in Python with the help of Aldebaran's Choregraphe simulator and NaoQi API to work with the robot and different audiovisual tools. The robot used has been a NAO robot property of Carlos III University of Madrid and all the experiments have been tested in the PLG's laboratory (Planning and Learning Group) in the Carlos III University.

Índice General

Índice de figuras

Índice de tablas

Chapter 1: Introduction

The first chapter of the work is divided into four sections. In the first one a description of the problem is presented. In the second one there are exposed the points of to develop this project. In the third part the aims are detailed in order to obtain the development steps and finally, the fourth and last paragraph where is explained the structure followed in the present document.

1.1 Description of the problem

In this project, the design and the development of a system is described. By using diverse algorithms, there will be given orders to NAO with the aim that he would be able to read and sing a musical score by itself. Instead of being an audio player, the challenge is to use artificial intelligence to solve the mentioned problem.

As when a human reads a musical score and establishes a few standards for the interpretation due to his musical knowledge, NAO should be capable of developing this logical and autonomous thoughts. Therefore, NAO will sing any musical melody that he sees on the music stand.

Along the development of the system, diverse levels of objectives have been set to reach the best possible solution thanks to a periodic and fluid communication between tutor and student.

At the end of this project there is completed a cycle that in a future can be continued whether working on the same project, or using the different functions developed for other technological challenges.

1.2 Motivation

The XXth and XXIst centuries have supposed an authentic technological revolution by reaching the technology to all the corners of the planet replacing our intellectual needs. Nowadays, the humanity is in the beginning of a new age, in which the mental work acquires more importance than the physical work needed during the industrial revolution.

The progress of the technology is unstoppable today. There exists a continuous development that generates the appearance of new electronic devices. We don't imagine our life without them. The motive of this is because they have been created to satisfy a need that facilitates our life.

Robotics has considerably progressed and programmers have developed prototypes with different forms and functions: humanoid robots as a person, industrial robots to work or perform household chores as floor cleaners, players in many games employed artificial intelligence or musician robots able to play the fastest melodies better than the virtuous human.

To have the possibility of developing a system of Artificial Intelligence in which a robot will be able to read any musical score was a great challenge for the author of this work.

The author, due to his wide musical education and his interest in the robotics, thinks that his project is a unique opportunity to connect music with technology. It has been a professional and a personal challenge.

1.3 Main objectives

The aim of this work is that the robot NAO is capable by itself, to read and to sing any musical score that can or cannot be seen in advance, by using the laws that govern the music.

The project consists of three parts:

- The first step is that, using the best resolution camera, NAO is able to take an image of the musical score placed on the lectern. The image will have to be sent to the computer for his later treatment.

- In the second part, from the computer and using code Python there will be executed the image analysis that was sent by NAO. In this processing, there will be used tools and algorithms dedicated to the digital treatment of images.

- In the last step, NAO will be able to play the audios corresponding to the notes written in the musical score, which have been processed by tools of audio processing.

For the implementation of this project, the author had to learn to use diverse applications and to programming in Python, a language new for him because it is not given in his degree.

1.4 Structure of the document

This document is structured in seven chapters which content is described below:

- In the first chapter is described an introduction of the work, the motivations and the aims of the same one. Also there are detailed the contents that will have the different chapters that form this document.
- In the second chapter there is explained the state of the art related to the work.
- In the third chapter is described the development of the system that has been implemented. It is divided in two sections: analysis and design.

- In the fourth chapter, is explained the results of the experimentation for the system job correctly.
- In the fifth chapter will be explained the management of the project through a description of the different phases, the planning to carry it out and the budget required.
- In the sixth chapter there will be descripted the conclusions: the foreground, the problems encountered, the problems of the goals and the future works.
- Finally, there are three annexes: the project summary in English, the installation manual and the user manual.

Capítulo 2: Estado del Arte

En este capítulo se describe el estado del arte relacionado con este trabajo. Se aporta información acerca de la robótica y la música.

En primer lugar, se hará referencia a qué es un robot, una breve historia del mismo, tipos y presentación del robot Nao acerca del cuál gira el proyecto. Seguidamente se presentarán las técnicas de programación utilizadas como el lenguaje y las técnicas de inteligencia artificial que se usan, como la visión artificial.

Posteriormente, se abordarán unas nociones básicas de música para entender el funcionamiento del proyecto. También se abordará cuál es el estado actual de la robótica y la tecnología en el mundo musical.

Por último, se hará una referencia a trabajos ya realizados que aborden temas de la misma disciplina.

2.1 Robótica

Robótica [1] es la ciencia que estudia el diseño y la programación de agentes físicos, conocidos como robots. Combina diversas disciplinas como son: la mecánica, la electrónica, la informática, la inteligencia artificial, la ingeniería de control y la física. Otras áreas importantes en robótica son el álgebra, los autómatas programables, la animatrónica y las máquinas de estados.

2.1.1 ¿Qué es un robot?

Un robot puede ser tanto un mecanismo electromecánico físico como un sistema virtual de software. Los dos coinciden en brindar la sensación de contar con capacidad de

pensamiento y/o resolución de actos humanos, pero en realidad se limitan a ejecutar órdenes dictadas por personas.

No existe una definición exacta de robot, pero sí se ha intentado argumentarlo de la mejor manera posible. Se muestran algunos ejemplos:

"Un mecanismo reprogramable con un mínimo de cuatro grados de libertad diseñado para manipular y transportar partes, herramientas o implementar manufactura especializada a través de movimientos programados para la ejecución de la una tarea específica de manufactura" British Automation & Robot Association (BARA) [2].

"Es un dispositivo reprogramable y multifuncional diseñado para mover materiales, piezas, herramientas o dispositivos especializados a través de movimientos programados" Robot Institute of America [3], 1979.

"Un sistema que existe en el mundo físico y que autónomamente sensa su medio ambiente y actúa sobre él" Maja Mataric/USC (University of Southern California) [4].

2.1.2 Historia de la robótica

La palabra robot [5] se empleó por primera vez en 1920 en la obra R.U.R. (Rossum's Universal Robots) por el escritor checo Karel Capek. La trama se presenta en una fábrica que crea personas artificiales llamadas robots con apariencia humana.

En 1942, el bioquímico, escritor y divulgador científico norteamericano Isaac Asimov predijo el aumento de una poderosa industria robótica [6], predicción que ya se ha hecho realidad. Además, empleó la palabra robótica en su relato corto "Runaround" [7], donde se describen las tres leyes de la Robótica:

1. Un robot no puede hacer daño a un ser humano o, por inacción, permitir que un ser humano sufra daño.
2. Un robot debe obedecer las órdenes dadas por los seres humanos, excepto si estas órdenes entrasen en conflicto con la Primera Ley.

3. Un robot debe proteger su propia existencia en la medida en que esta protección no entre en conflicto con la Primera o la Segunda Ley.

El concepto de robot apareció al plantearse sustituir a un operador por un programa de ordenador que controlase los movimientos del manipulador.

Se suelen distinguir cuatro generaciones [8] en el desarrollo de la industria robótica:

- **Primera generación**: los robots primitivos tenían capacidad para almacenar trayectorias de movimiento descritas punto a punto. Se trataba de robots programables y de tipo brazo manipulador. Sólo podían memorizar movimientos repetitivos programados anteriormente, asistidos por sensores internos que les ayudaban a moverse con precisión. El entorno en el que operan no les afecta. Su sistema de control se conoce como lazo abierto, en función de una entrada, se produce una salida y no influye en el resto.

- **Segunda generación**: entra en escena a finales de los años 70. Cuentan con sensores externos (tacto y visión por lo general) que dan al robot información (realimentación) limitada del mundo exterior. Pueden hacer elecciones limitadas o tomar decisiones y reaccionar ante el entorno de trabajo. Se les conoce por ello como robots adaptativos.

- **Tercera Generación**: robots capaces de memorizar y repetir una serie de pasos realizados por un operador humano. Se enfocan hacia trabajos de teleoperación. Al igual que los robots de primera generación, su sistema de control es de lazo abierto.

- **Cuarta Generación**: robots actuales que emplean inteligencia artificial (IA) y hacen uso de los ordenadores más avanzados. Estos ordenadores no sólo trabajan con datos, sino que también lo hacen con los propios programas, realizan razonamientos lógicos y aprenden. La IA permite a los ordenadores resolver

problemas inteligentemente e interpretar información compleja procedente de avanzados sensores.

Durante años los robots han sido considerados útiles sólo si se empleaban como manipuladores industriales. Recientemente han aparecido nuevos y variados papeles para los robots. A diferencia de los tradicionales robots fijos de manipulación y fabricación, estos nuevos robots móviles pueden realizar tareas en un gran número de entornos distintos y se les conoce como robots de servicio. Los robots de servicio se emplean en la investigación científica, en la educación, con fines de bienestar personal y social, etc. Son especialmente adecuados para el trabajo en áreas demasiado peligrosas para la vida humana o para la exploración de lugares anteriormente vetados al ser humano.

Este crecimiento revolucionario en el empleo de robots como dispositivos prácticos es un indicador de que los robots desempeñarán un importante papel en el futuro. Los robots del futuro podrán relevar al hombre en múltiples tipos de trabajo físico. Se piensa que los robots están en ese momento crítico antes de la explosión del mercado, como lo estuvieron los PCs (Personal Computers) en 1975. El campo de la robótica se desbordará cuando los robots sean de dominio público.

2.1.3 Tipos de robots

Los robots se pueden clasificar en cinco tipos [9]:

- **Zoomórficos**: sus sistemas de locomoción imitan a la de alguna criatura; por ejemplo, serpientes, cuadrúpedos o arácnidos. En la figura 2.1 se muestra un robot zoomórfico.
- **Humanoides**: (androides o ginoides, según su versión masculina o femenina) se caracterizan por tratar de reproducir la forma del ser humano y sus movimientos. Un ejemplo es el robot Asimo fabricado por Toyota que se puede visualizar en la figura 2.1.

- **Híbridos**: combinan las características de los demás tipos descritos anteriormente (por ejemplo, un brazo robótico montado sobre una plataforma móvil con ruedas). En la figura 2.2 se puede observar un ejemplo de robot híbrido.
- **Polimórficos**: pueden "reconfigurarse" a sí mismos, adaptando su estructura para realizar cada tarea específica: caminar, serpentear, nadar o volar. La figura 2.2 nos ilustra acerca de los robots polimórficos.

FIGURA 2.1: Robot zoomórfico y humanoide

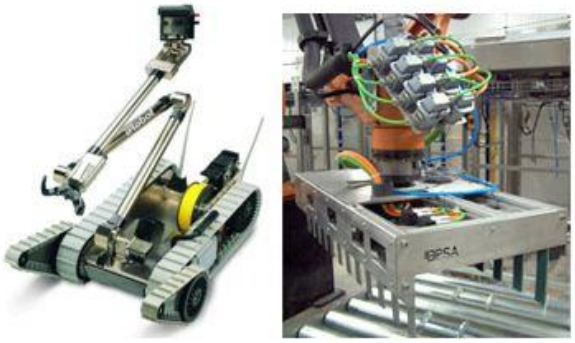


FIGURA 2.2: Robot híbrido y polimórfico

2.1.4 Robot NAO

NAO [10] es un robot humanoide de 57 cm de altura desarrollado por la empresa francesa Aldebaran robotics [11]. La figura 2.3 presenta a NAO H25. Posee reconocimiento de voz y

de órdenes, puede detectar las diferentes formas de los objetos y rostros, es sensible al tacto en muchas partes de su cuerpo, y tiene conectividad Wi-Fi que incluso le permite comunicarse con otros robots de su misma clase. Programable en varios lenguajes como C++, Python, Urbi, .NET, Java y Matlab. Equipado con un procesador Intel ATOM 1,6 GHz que funciona con un núcleo Linux y el software dedicado propio Aldebaran Naoqi [12]. Es el primer robot capaz de desarrollar emociones y formar lazos afectivos con los humanos que lo cuidan.

Mediante su tecnología puede detectar las emociones humanas estudiando el lenguaje corporal y las expresiones faciales, siendo capaz de entender el estado de ánimo de la persona. Es capaz de recordar sus encuentros e interacciones con las diferentes personas con las que se ha encontrado, así como de sus rostros. NAO se utiliza en multitud de Universidades y para los ámbitos de la educación, como es en el caso de niños autistas.

FIGURA 2.3: Robot NAO H25

2.2 Música

La música es el arte de combinar los sonidos en una secuencia temporal atendiendo a las leyes de la armonía, la melodía y el ritmo, o de producirlos con instrumentos musicales. Siguiendo estos patrones se consigue un conjunto de sonidos sucesivos y combinados, que por lo general producen un efecto estético o expresivo, creando diferentes sensaciones dependiendo del oyente.

En este apartado del trabajo se va a comenzar aportando unas nociones básicas de teoría musical, se continuará dando una visión actual de cómo se emplea la robótica en el mundo musical, y se finalizará explicando los distintos programas musicales empleados en el trabajo.

2.2.1 Teoría musical

En este apartado se explican los conceptos básicos musicales necesarios para ser capaz de entender una partitura y llegar a interpretarla.

- **El pentagrama**: soporte donde se disponen las diversas notas musicales y demás signos de notación. Formado por cinco líneas horizontales paralelas y equidistantes. En la figura 2.4 se ve un pentagrama musical.

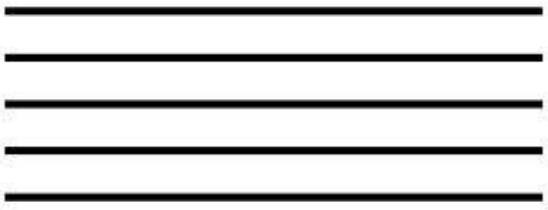


FIGURA 2.4: Pentagrama musical

- **Clave musical**: símbolo empleado en notación musical a comienzos del pentagrama, cuya función es asociar las notas musicales con las líneas o espacios del pentagrama. Existen siete claves que se pueden ver en la figura 2.5: sol en

segunda línea, fa en cuarta, fa en tercera, do en primera, do en segunda, do en tercera y do en cuarta.

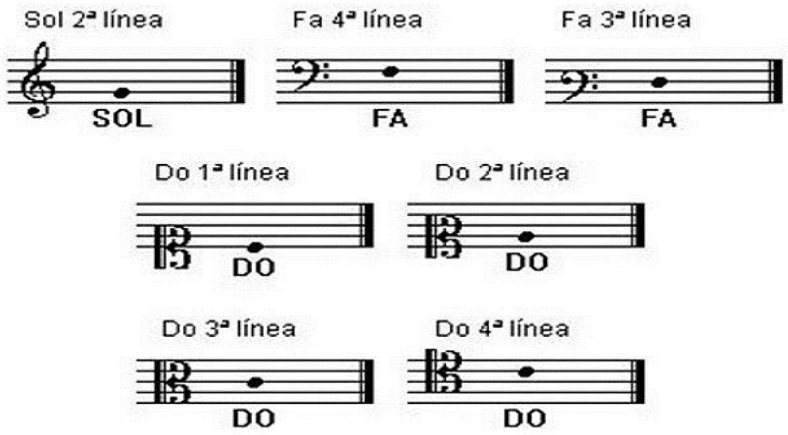


FIGURA 2.5: Claves musicales

- **Nota musical**: Elemento más básico y primordial del sonido y de la música a partir del cual se arman las diferentes melodías y armonías ya que cada una de ellas representa un sonido particular e indivisible que, puesta en conjunto con otras, arma un sonido más complejo y duradero. Las notas musicales son elementos abstractos, pero se representan de manera simbólica en los pentagramas para que puedan ser leídas e interpretadas por los músicos. Las notas musicales son siete: do – re – mi – fa – sol – la – si. La altura de las notas sobre el pentagrama representa el nombre de cada una de ellas. En la figura 2.6 se observan las notas de la escala musical para la clave de sol.

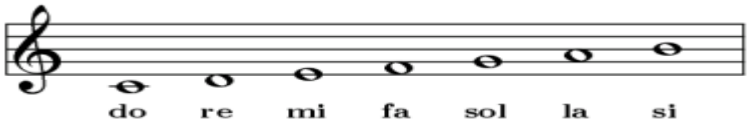


FIGURA 2.6: Notas musicales

- **Figura musical**: Representa gráficamente la duración musical de un determinado sonido en una pieza musical. La manera gráfica de indicar la duración relativa de una nota es mediante la utilización del color o la forma de la cabeza de la nota, la presencia o ausencia de la plica, segmento vertical que en determinadas figuras acompaña a la cabeza, así como la presencia o ausencia de corchetes con forma de ganchos. Típicamente se suelen emplear las siguientes siete figuras musicales presentadas en la tabla 2.1

Redonda		4 Tiempos
Blanca		2 Tiempos
Negra		1 Tiempo
Corchea		1/2 Tiempo
Semicorchea		1/4 Tiempo
Fusa		1/8 Tiempo
Semifusa		1/16 Tiempo

Tabla 2.1: Figuras musicales

- **Tiempo musical (tempo)**: se trata de la velocidad con la que debe ejecutarse una pieza musical. En las partituras de una obra, el tempo se representa al inicio de la pieza encima del pentagrama. Se mide en pulsos por minuto, de ahí que aparezca en notación numérica. En la figura 2.7 se muestra en la parte superior izquierda la indicación del tempo musical.

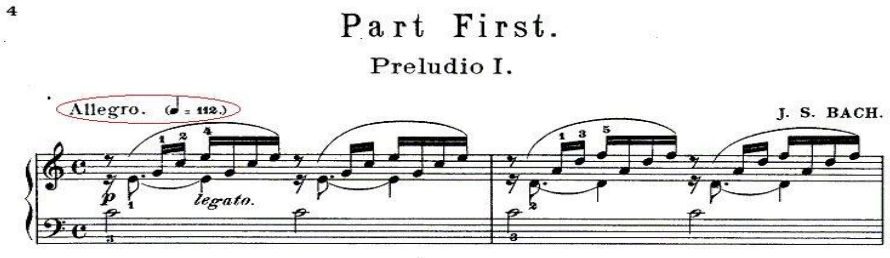


FIGURA 2.7: Tiempo musical

Además, frecuentemente suele indicarse el tempo según su significado. Los más comunes se plantean en la tabla 2.2:

Italian	English	Beats per minute
Presto	Very fast	168-208
Allegro	Fast	120-168
Moderato	Moderate speed	108-120
Andante	Moderate walking speed	76-108
Adagio	Slow (literally "at ease")	66-76
Largo	Slow and solemn	40-66

Tabla 2.2: Significado de tiempos musicales

- **Compás musical**: entidad métrica musical compuesta por varias unidades de tiempo (figuras musicales) que se organizan en grupos, en los que se da una contraposición entre partes acentuadas y átonas. En función del número de tiempos que los forman surgen los compases binarios, ternarios y cuaternarios.

  La división en compases se representa mediante unas líneas verticales, llamadas líneas divisorias o barras de compás, que se colocan perpendicularmente a las líneas del pentagrama, tal y como se ilustra en la figura 2.8.

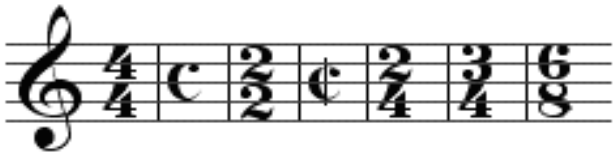


FIGURA 2.8: Compás musical

- **Cifrado americano**: es una de las formas más populares y sencillas de representar las notas musicales o incluso acordes. La técnica se basa en nombrar a las notas musicales con letras latinas tal y como se muestra en la tabla 2.3:

A	B	C	D	E	F	G
La	Si	Do	Re	Mi	Fa	Sol

Tabla 2.3: Cifrado americano

2.2.2 La robótica en el mundo musical

La primera vez que se fabricó un robot musical fue en el año 1737 cuando el inventor Jacques Vaucanson construyó su primer autómata llamado "El flautista" [13], una figura de tamaño natural de un pastor que tocaba el tambor y la flauta con un repertorio de doce canciones.

Actualmente son numerosos los fabricantes que trabajan sobre este sector con un gran público debido a su novedad, pues existen hasta grupos musicales únicamente integrados por robots. Se muestran algunos ejemplos:

- Vijay Kumar, de la Escuela de Ingeniería y Ciencias Aplicadas de la Penn University (USA) [14] desarrolló unos sistemas automatizados capaz de tocar diferentes instrumentos como una guitarra, un teclado, un platillo y un timbal.

  Los robots tenían acoplados los elementos necesarios para tocar su instrumento, tal y como se puede observar en la figura 2.9. El encargado del timbal contaba con una baqueta accionada por un pequeño motor, que hacía que golpease el instrumento en el momento preciso. El teclado lo tocaban entre cuatro robots, volando en vertical hacia abajo para pulsar las teclas.

FIGURA 2.9: Robots musicales de la Penn University

- HRP-4C [15] es la estrella pop del mundo robótico y fue desarrollada por el Instituto Nacional de Tecnología y Ciencia Industrial Avanzada de Tokio. Acompañada por cuatro bailarinas, su debut como cantante en el Digital Content Expo 2010 se puede observar en la figura 2.10 y llamó la atención de todos los aficionados a la tecnología.

FIGURA 2.10: Robot cantante HRP-4C

- StickBoy [16] es un batero de cuatro brazos y cresta metálica con un aspecto muy rockero tal y como se aprecia en la figura 2.11. Desarrollado en 2007, utiliza un

mecanismo de aire comprimido junto a un control MIDI para ejecutar sus obras y se presentó en la última Campus Party de Berlín, Alemania.

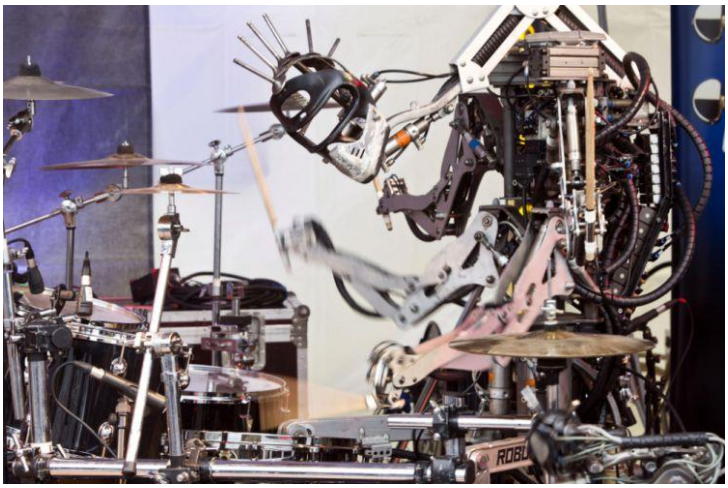


FIGURA 2.11: Robot musical StickBoy

- The Trons [17] es una banda neozelandesa formada por robots fabricados con desechos electrónicos. Lanzó su primer álbum con DVD, realizó giras en su país natal, en Europa y de forma reciente participó en el Festival de Melbourne, en Australia. En la siguiente figura 2.12 se observa esta singular banda musical.

FIGURA 2.12: Banda robótica musical The Trons

- Compressorhead [18] es una banda formada tres robots androides músicos que tocan hard rock y heavy metal. Puede visualizarse en la imagen 2.13. Creado por ingenieros alemanes a partir de piezas de chatarra reciclada, pero con componentes electrónicos sofisticados y dirigidos por ordenador, estos robots pueden tocar con una precisión inédita. Compressorhead hizo su primera aparición oficial en el Festival Metal de Australia con un éxito absoluto.

FIGURA 2.13: Banda robótica musical Compressorhead

- Z-Machines [19] es una banda de heavy metal y rock creada por un equipo de la Universidad de Tokio en el año 2013 junto con un compositor electrónico británico. Se crearon para averiguar si los robots pueden reproducir la música con carácter emocional. Los robots son capaces de tocar los instrumentos clásicos de una banda de rock de una forma más rápida que cualquier humano. En la figura 2.14 se puede ver a Z-Machines en un concierto en directo.

FIGURA 2.14: Banda robótica musical Z-Machines

2.2.3 Software musical

En esta sección se van a detallar aquellos programas musicales empleados en el trabajo para la parte de reproducción de audio.

2.2.3.1 Sibelius

Sibelius [20] es un software de composición y notación musical muy popular que ofrece herramientas sofisticadas pero fáciles de usar, preferidas tanto por los mejores compositores, arreglistas y editores como por educadores y estudiantes. Permite crear partituras precisas y reproducirlas mediante sintetizadores similares a los diferentes instrumentos musicales. Un sintetizador es un aparato que genera y manipula sonidos por medios electrónicos. La forma de la onda generada es alterada en su duración, altura y timbre mediante el uso de dispositivos tales como amplificadores, mezcladores, filtros, reverberadores, secuenciadores y moduladores de frecuencia. Gracias a este procesado se realizará el tratamiento de los archivos de audio obteniendo la sonoridad del NAO deseada.

2.2.3.2 Adobe Audition

Adobe Audition [21] es un editor digital de audio que forma parte de la gama de productos que ofrece la empresa de software Adobe Systems [22]. Posee una interfaz con forma de estudio de sonido que permite tanto un entorno de edición mezclado multipista: grabación, mezcla, edición y masterización. Herramienta básica para la producción musical.

2.2.3.3 Finale

Finale [23] es un programa completo para escribir, ejecutar, imprimir y publicar partituras de música. Fue creado por la empresa MakeMusic [24]. Está diseñado para toda clase de músicos, desde estudiantes y profesores hasta compositores profesionales.

Es el programa más importante de una serie de programas de edición de partituras creados por MakeMusic para Microsoft Windows y Mac OS X. Finale es uno de los programas de notación musical más popular del mercado internacional.

Permite escuchar lo que está escrito en la partitura del proyecto sobre el que se está trabajando, utilizando la tarjeta de sonido del ordenador. También permite grabar esa ejecución en un CD de audio.

2.2.3.4 Audacity

Audacity [25] es un editor de audio gratuito para grabar sonidos, reproducir sonidos, importar y exportar archivos WAV, AIFF, y MP3, y más. Edición de sonidos usando cortar, copiar y pegar, mezclar pistas, o aplicar efectos a tus grabaciones.

Posee un editor de envolvente de amplitud de la onda sonora único para Audacity, un modo espectrograma ajustable a medida y una ventana de análisis de frecuencia para aplicaciones de análisis de audio. Efectos propios como Bass Boost (realzador de graves), Wahwah, y cancelador de ruido, y también soporta efectos plug-in VST (Virtual Studio Technology) [26] una interfaz estándar desarrollada por Steinberg para conectar sintetizadores de audio y plugins de efectos a editores de audio y sistemas de grabación.

2.2.3.5 Musescore

MuseScore [27] es un programa de notación musical para sistema operativo Linux, Mac OS X y Microsoft Windows. Es un editor con soporte completo para reproducir partituras e importar o exportar MusicXML y archivos MIDI [28] estándar. Permite importar y exportar muchos formatos musicales, incluyendo MIDI y MusicXML [29], así como importar archivos de los programas comerciales de arreglos musicales, Band-in-a-Box. Puede generar documentos PDF, SVG o PNG.

2.2.3 Micrófono

Un micrófono es un transductor acústico-eléctrico que transforma las ondas sonoras en impulsos eléctrico. Se emplea para grabación y reproducción de audio. Está formado por un diafragma que vibra en función de la intensidad de las ondas sonoras que inciden en él. Está atraído por un electroimán que aprovecha el movimiento del diafragma para convertirlo en impulsos eléctricos.

2.3 Visión artificial

La visión artificial es un campo de la Inteligencia Artificial (IA) que, mediante la utilización de las técnicas adecuadas, permite la obtención, procesamiento y análisis de cualquier tipo de información especial obtenida a través de imágenes o vídeos digitales.

En la actualidad la visión artificial tiene muchas aplicaciones como: la detección de personas mediante reconocimiento facial o diferentes tipos de gestos, identificación de huellas dactilares o reconocimiento de patrones en imágenes médicas, entre otras.

Está compuesta por un conjunto de procesos con el fin de analizar las imágenes: captación de imágenes, memorización de la información, procesado e interpretación de los resultados.

En este trabajo se han utilizado varias técnicas de análisis de imágenes como son: binarizado, detección de rectas y curvas mediante la transformada de Hough, y librerías de Python como OpenCV y PIL.

2.3.1 Binarizado

El binarizado de imágenes es una técnica del procesamiento de imágenes que consiste en un proceso de reducción de la información de una imagen digital a dos valores: 0 (negro) y 255(blanco).

Este proceso consiste en comparar cada pixel de la imagen con un determinado umbral (valor límite que determina si un pixel será de color blanco o negro). Los valores de la imagen que sean mayores que el umbral toman un valor 255 (blanco), el resto de pixeles toman valor 0(negro).

En la figura 2.15 se observa el resultado de binarizar una imagen.

FIGURA 2.15: Imagen original e imagen binarizada

2.3.2 Transformada de Hough

La transformada de Hough es una herramienta que permite detectar curvas paramétricas en una imagen. Es una técnica muy robusta frente al ruido y a la existencia de huecos en la frontera del objeto. A la hora de aplicar la transformada de Hough a una imagen es necesario obtener primero una imagen binaria de los píxeles que forman parte de la frontera del objeto, de ahí que se haya binarizado la entrada a recibir por el algoritmo.

- Transformada de Hough para detectar segmentos rectos

Se buscan puntos alineados que puedan existir en la imagen, es decir, puntos en la imagen que satisfagan la ecuación de la recta, para distintos valores de los ejes en el espacio de Hough (ρ y θ), siendo la ecuación de la recta en forma polar: $\rho = x \cdot \cos\theta + y \cdot \text{sen}\,\theta$. Por

tanto, una recta en un plano con ejes cartesianos (x, y) será un punto en el Espacio de Hough (ρ, θ).

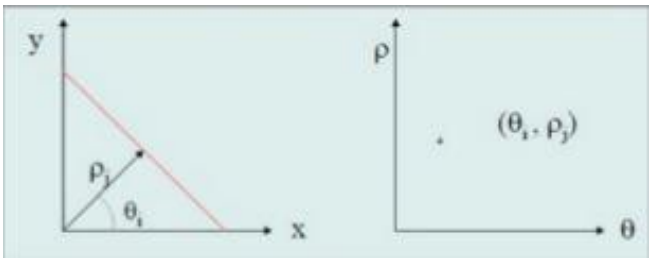


FIGURA 2.16: Espacio cartesiano y espacio de Hough

El siguiente paso es evaluar la ecuación de la recta para cada punto de la imagen (xk,yk). Si se cumple esta ecuación se incrementa en uno el número de votos de la celda. Un número de votos elevado indica que el punto pertenece a la recta.

- Transformada de Hough para detectar circunferencias

Una circunferencia está definida por 3 parámetros (h, k, r).

En esta implementación se busca eliminar un parámetro. Para esto se realiza una estimación previa del radio y luego se calcula el acumulador para radios en un entorno de dos pixeles sobre el radio estimado. Las coordenadas del máximo de este acumulador corresponden al trío (coordenadas del centro, y radio) que mejor se ajusta. Este proceso tiene mayor complejidad computacional que la detección de rectas.

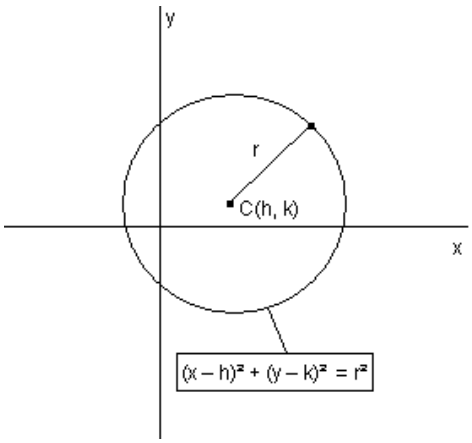


FIGURA 2.17: Características generales de una circunferencia

- Transformada de Hough para detectar elipses

Una elipse se define por cuatro parámetros (a,b,h,k) y su ecuación general es la siguiente:

$$\frac{(\times - h)^2}{a^2} + \frac{(y - k)^2}{b^2} = 1$$

El coste computacional de detección de los cuatro parámetros es excesivamente alto, por ello la transformada de Hough separar la ejecución en dos pasos:

1. Identificación del centro de la elipse, esto requiere dos parámetros de la transformada de Hough.

2. Evaluación de los otros dos parámetros (en referencia a los ejes) con una implementación sencilla centrada en la transformada de Hough.

El centro de la elipse puede definirse de la siguiente manera. Se consideran dos puntos, P y Q, en una elipse y calcular la tangente de estos puntos. Donde r es el punto donde se cruzan estas tangentes y M el punto central del segmento PQ. Para una elipse perfecta, el centro estará en la línea que se origina en r y pasa M. Las líneas formadas por diferentes pares de puntos de la elipse se cruzan en el centro de la misma. Se pueden observar en la figura 2.18. Estos parámetros de la transformada de Hough se utilizan para acumular estas líneas, y el máximo del histograma corresponden a la misma intersección. Como este algoritmo registra una entrada para cada par de píxel de la imagen, se requiere un amplio coste computacional.

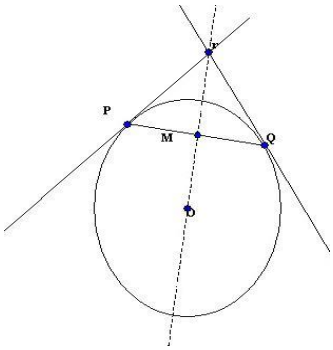


FIGURA 2.18: Elipse

- Ventajas del uso de la Transformada de Hough
  - Es útil para detectar formas complejas.
  - Robusta frente al ruido.
  - Cada píxel de la imagen se procesa de modo independiente, lo que facilita su implementación en paralelo.
  - Reconoce patrones ligeramente deformados, discontinuos u ocultos.
  - Permite buscar simultáneamente todas las ocurrencias de un patrón

- Inconvenientes del uso de la Transformada de Hough
  - Excesiva cantidad de tiempo y memoria empleados.
  - No ofrece una respuesta absoluta, sino un índice de probabilidad de que cada una de las formas posibles sea la buscada.

2.3.3 Librerías de visión

Dentro de las librerías de visión artificial, destacan dos: OpenCV y PIL. Ambas se detallarán a continuación.

2.3.3.1 OpenCV

OpenCV (Open Source Computer Vision) [30] es una librería de programación de visión artificial desarrollada para proporcionar algoritmos y funciones que permitan trabajar en tiempo real con visión artificial.

Inicialmente fue desarrollada por la empresa Intel [31] y más tarde, fueron varios de sus creadores los que siguieron desarrollando con gran éxito la librería acercándola al mundo de la robótica y de los videojuegos.

Tiene implementaciones en C++, Python, funciona en sistemas operativos Windows, Mac, Linux, iOS y Android y cuenta con más de 2500 algoritmos. Los algoritmos de esta librería permiten identificar objetos, caras, clasificar acciones humanas en vídeo, hacer tracking

de movimientos de objetos, extraer modelos 3D, encontrar imágenes similares, eliminar ojos rojos o seguir el movimiento de los ojos entre otros muchos.

2.3.3.2 PIL

PIL (Python Imaging Library) [32] es una librería de programación para el lenguaje Python empleada en visión artificial. Ofrece soporte para abrir, manipular y guardar una gran cantidad de diferentes formatos de archivo de imagen. Está disponible para Windows, Mac OS X y Linux. La última versión de PIL es 1.1.7, fue lanzado en septiembre de 2009 y es compatible con Python 1.5.2-2.7.

PIL ofrece varios procedimientos estándar para la manipulación de imágenes. Éstas incluyen: manipulaciones por píxel, enmascaramiento y manejo de la transparencia, filtrado de imágenes, tales como visión borrosa, contorno, suavizado, o búsqueda de aristas, mejora de la imagen, tales como la nitidez, ajuste de brillo, contraste o color y añadir texto a las imágenes entre otras muchas.

Algunos de los formatos de archivo soportados incluyen PPM, PNG, JPEG, GIF, TIFF y BMP. También es posible crear nuevos decodificadores de archivos para expandir la biblioteca de formatos de archivos accesibles.

2.4 Trabajos similares

En este punto del trabajo se van a presentar proyectos realizados anteriormente en el que se trabaja la visión artificial con el robot NAO.

Desarrollo de un sistema de juego al Tres en Raya para el robot NAO H25

Este trabajo [33] fue desarrollado por mi tutora Nerea Luis Mingueza en el año 2013. Se presenta el desarrollo de un módulo para el robot NAO que le permitirá jugar al “Tres en

Raya" con un humano de forma autónoma. El desarrollo del juego incluye una gran variedad de técnicas de inteligencia artificial como el aprendizaje por demostración, la visión artificial y la búsqueda heurística que le permitirán alcanzar al robot ese grado de autonomía.

Destacando el apartado de visión artificial, la detección de cambios en el entorno se realiza mediante histogramas, empleando la función histograma de la biblioteca de OpenCV. El robot captura una serie de imágenes detectando todo el tablero del Tres en Raya, estudia a través de sus histogramas para, posteriormente usando algoritmia, generar las acciones propias del juego contra el usuario.

Localización e interacción con objetos mediante visión artificial con el robot NAO

El siguiente trabajo [34] fue desarrollado por Ignacio Elizaga Navascués en el año 2012. Se presenta el desarrollo de una colección de funciones primitivas que permiten al robot NAO llevar a cabo comportamientos sencillos que luego puedan ser utilizados por un planificador automático.

Se destaca la técnica del uso de blobs para la detección de colores. Un blob es una región de una imagen que tiene propiedades de color o iluminación diferentes a todo lo que la rodea. Para realizar el procesamiento de una imagen, el sistema deberá encontrar los diferentes blobs que existan en la misma con el color calibrado por el usuario. El objetivo es que Nao sea capaz de procesar una imagen y distinguir en ella blobs o regiones de diferentes colores. Para ello, se emplea la librería CMVision (Color Machine Vision).

Revisión, modificación y validación experimental del sistema sensorial del robot HOAP3

El último trabajo [35] es una tesis de master de Igor de Miguel Andrés y se centra en un robot diferente como es el HOAP3, del cual realiza una revisión, modificación y validación experimental de los sensores del robot. Lleva consigo la realización de determinados

ensayos previos para comprobar el estado inicial de todos los sensores, seguido de una posterior modificación de determinados parámetros y vuelta a hacer nuevos ensayos, esta vez para comprobar la evolución de los sensores y todo ello bajo el objetivo real que se presenta y que consiste en la calibración de todos los sensores.

Un apartado que nos interesa especialmente es el estudio detallado del robot NAO. Se explican todos los sensores, las articulaciones para sus movimientos y la forma de visionado mediante sus diferentes cámaras, muy útil para el desarrollo de este trabajo.

Capítulo 3: Descripción del sistema

3.1 Introducción

En este capítulo se describe el proceso de desarrollo del sistema que se ha implementado para realizar el trabajo. Está divido en dos secciones: análisis y diseño.

En el apartado análisis del sistema, se describen varios apartados: descripción de las características funcionales, restricciones del sistema, entorno operacional, especificación de casos de uso y especificación de requisitos.

En la sección Diseño se presenta arquitectura del sistema, una descripción general del sistema, descripción de componentes y descripción del funcionamiento del sistema.

3.2 Análisis del sistema

En este apartado se realiza un análisis completo de todo el trabajo. En primer lugar, se describirán todas las características que forman el sistema y que hacen posible el funcionamiento del mismo. Seguidamente se detallarán las restricciones a nivel de software y de hardware del sistema. Se continuará describiendo el entorno sobre el que opera el sistema y sus consecuencias y se finalizará haciendo una especificación, primero, de los casos de uso, y segundo de requisitos.

3.2.1 Descripción de las características funcionales

El objetivo de esta sección es describir las características con las que funciona el sistema. Se presentarán todos los elementos que influyen en el proyecto, así como las tareas que realizan cada uno de ellos.

El objetivo del proyecto es que el robot NAO H25 sea capaz de leer, analizar y cantar una partitura musical. Para ello se dispone de un robot NAO H25, diferentes partituras

musicales, un atril, un ordenador, un router inalámbrico para realizar la conexión NAO-ordenador mediante WiFi y un micrófono de mano.

El conjunto de todos los elementos que forman el sistema se muestra en la figura 3.1.

FIGURA 3.1: Sistema completo del trabajo

Las funcionalidades del sistema se establecieron como objetivos al principio del proyecto, cuando se proponían los objetivos a alcanzar y la forma de llegar a ellos:

- El robot NAO H25 debe ser capaz de visualizar una partitura, sea cuales sean las notas escritas sobre la misma.
- Una vez visualizada, se deberá procesar la imagen mediante el desarrollo de código de programación específico que servirá para detectar varios objetivos:
  - Identificar las cinco líneas del pentagrama.
  - Conocer el número de notas escritas en la partitura.
  - Saber qué notas son.
  - Almacenar las notas en el orden correspondiente.
- Tras el análisis de la partitura, reproducir cada nota, y si son varias, cantarlas en el orden correcto.
- Realizar esta iteración completa cada vez que se toca uno de los sensores que posee en la cabeza.
- La partitura estará situada sobre un atril a una altura donde el robot sea capaz de visualizarla.

- La partitura estará formada por un pentagrama de cinco líneas, sin barras de compás.
- Las notas serán negras, es decir de duración un pulso y todas iguales.

3.2.2 Restricciones del sistema

En este apartado se van a describir las restricciones del sistema tanto a nivel software como a nivel hardware debido a los requisitos de los elementos del sistema.

3.2.2.1 Restricciones a nivel de software

- El sistema operativo sobre el que trabajará el proyecto es Linux, distribución Ubuntu 14.04 [36] de 64 bits. Dicho sistema operativo deberá, por tanto, estar instalado en el ordenador.
- El ordenador debe tener instalado la aplicación desarrollada por Aldebaran Robotics [11] para NAO: Choregraphe [37] y NaoQi [12] en las versiones 2.1.0.19 y 1.14.5 respectivamente.
- El lenguaje de programación empleado ha sido Python [38] por lo que se deberán instalar los paquetes necesarios para su versión 2.7.6.

3.2.2.2 Restricciones a nivel de hardware

- El ordenador debe cumplir unos requisitos mínimos: 1.5 GHz de CPU, 1 GB de RAM y tarjeta de red habilitada para Ethernet/Wi-Fi.
- La autonomía del robot utilizando su módulo de visualización es bastante limitada. Su batería de litio permite trabajar con él alrededor de 60 minutos, por lo que requiere de alimentación externa para un uso continuado.
- Si el robot está trabajando durante cierto tiempo de forma ininterrumpida, unos 50 minutos aproximadamente, requiere una pausa para enfriar la placa base del mismo.

- La conexión con el robot NAO se puede realizar inalámbricamente por Wi-Fi o mediante un cable Ethernet.
- El robot NAO tiene incorporadas dos cámaras frontales de distinta resolución, una en la boca y otra en la frente. Debido a su diseño interno no es posible el uso simultáneo de las mismas.
- La captura de la imagen se realiza con la cámara de mayor resolución. Ésta es la que posee en la frente con una calidad HD 1280x960 píxeles.
- Debido a los componentes que forman tanto el NAO como el ordenador, el procesado de la imagen puede demorarse un tiempo.

3.2.3 Entorno operacional

En esta sección se va a especificar detalladamente el entorno sobre el que operará el sistema y donde se ha ido desarrollando. Se tratarán dos apartados: software y hardware.

3.2.3.1 Entorno operacional software

- El sistema operativo utilizado ha sido Ubuntu 14.04. En primer lugar, se hizo uso de una máquina virtual de Linux sobre Windows 7. Para mejorar la rapidez y facilidad a la hora de programar el código y ejecutarlo se decidió crear una partición específica de Ubuntu en el disco duro del ordenador. Las herramientas empleadas para el procesado digital de la imagen requieren un sistema potente. La máquina virtual ralentizaba este proceso de manera muy notoria.
- El lenguaje de programación empleado ha sido Python en su versión 2.7.6. Se instalarán todos los paquetes necesarios para hacer uso de sus bibliotecas.
- Se hará uso de aplicaciones propias para el robot desarrollada por la empresa francesa Aldebaran Robotics:
  - Choregraphe versión 2.1.0.19: permite controlarán los módulos de visión y movimientos, ambos fundamentales para el proyecto.
  - NaoQi versión 1.14.5: recrea la simulación de un robot NAO.

- Para la reproducción de las notas ha sido necesario emplear programas musicales. A lo largo del proyecto se han utilizado herramientas como: Audacity, Musescore y Finale sin obtener los resultados deseados. Ha sido mediante los programas Sibelius y Adobe Audition donde se han alcanzado los objetivos planteados, ya que ambos poseían una edición de audio mucho más completa.

3.2.3.2 Entorno operacional hardware

- Robot humanoide NAO H25 fabricado por la empresa Aldebaran Robotics con las siguientes características:
  - Tiene unas dimensiones de 573x275x311mm y un peso de 5,2kg.
  - Permite conexiones Ethernet y WiFi para controlarlo desde un dispositivo externo.
  - Posee una batería de litio con una autonomía de 60 minutos de uso.
  - Tiene multitud de sensores de sonar para la localización de objetos o personas, dos cámaras de visión, cuatro micrófonos, girómetro, acelerómetro y bumpers de presión.
  - En las articulaciones posee motores de movimiento que le ofrecen multitud de grados de libertad.
  - Gracias a sus dos altavoces emite sonido al exterior.
- Un ordenador portátil con el entorno operacional software detallado en el apartado anterior y sobre el que ejecutará el programa.
- Un router para crear una LAN o red de área local que comunique al robot NAO con el ordenador portátil.
- Un micrófono con el que se grabarán los audios que reproducirá NAO.

3.2.4 Especificación de casos de uso

Un caso de uso define una interacción entre el usuario y el sistema. En este proyecto existen dos casos de uso:

- El primero representa el inicio del programa. El usuario ejecuta la orden de iniciar desde la terminal de Ubuntu.
- El segundo tiene lugar cuando NAO ha terminado de interpretar la partitura y el usuario pulsa el sensor de su cabeza. De este modo, se volverá a ejecutar el código volviendo al inicio del programa.

Para analizar de una forma más efectiva la interacción de todos los elementos del sistema se va a proceder a dibujar y explicar un diagrama de secuencia.

3.2.4.1 Diagrama de secuencia

El diagrama de secuencia es un gráfico que representa la secuencia de mensajes entre instancias de clases, componentes, subsistemas o actores que muestra una interacción. El tiempo avanza por el eje vertical y muestra el flujo de control de un participante a otro. Esta herramienta será ideal para representar las acciones que realizan los distintos actores que actúan en el proyecto.

3.2.4.1.1 Actores participantes

Los tres integrantes del proyecto que participan en él son:

- **Usuario:** Humano cuyas funciones, como hemos visto anteriormente, son iniciar el proyecto generando una orden al ordenador para que comience todo el proceso del trabajo, y hacer que NAO vuelva a reproducir una partitura pulsando el sensor de su cabeza.
- **Ordenador:** Se comunicará con el robot NAO y procesará la imagen que recibe de éste para luego indicar la orden de reproducción de audio.
- **Robot NAO:** Es el protagonista del trabajo, realizará la captura de la partitura y terminará la ejecución del programa cantando el conjunto de notas escritas en el atril.

3.2.4.1.2 Diagrama de secuencia

En la figura 3.2 se representa el diagrama de secuencia.

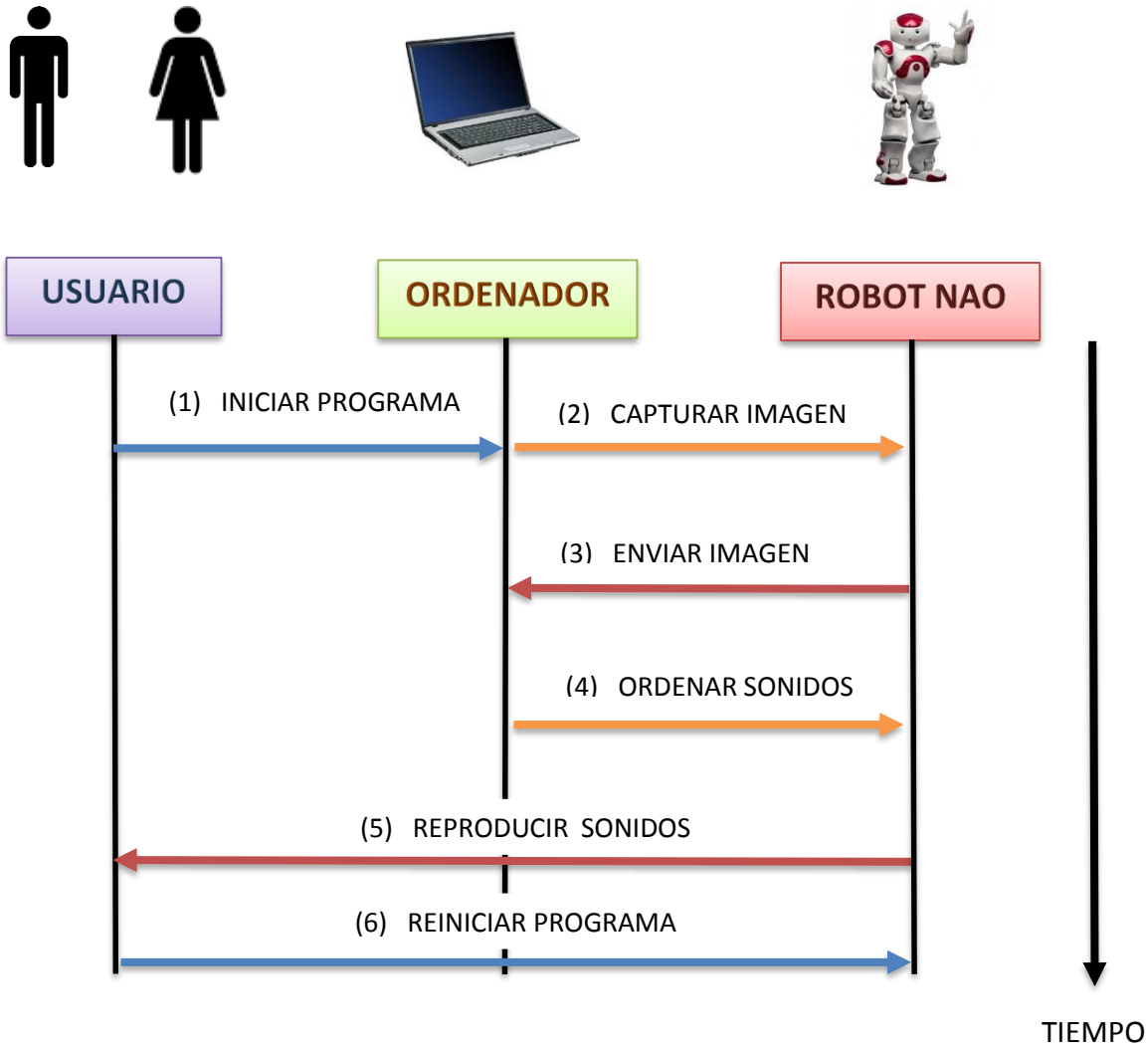


FIGURA 3.2: Diagrama de secuencia

Como se puede observar, existen 6 acciones definidas en el proyecto. Para conocer en detalle cada una de ellas, se van a realizar tablas independientes con la siguiente serie de atributos definidos previamente:

- **Código**: Identificación unívoca abreviada del requisito, se construye mediante el código de la acción, ACT, seguido de un - y de tres dígitos. Ejemplo, ACT-001.

- **Actor**: Protagonista de la acción.
- **Título**: Identificación extendida del requisito.
- **Descripción**: Se realiza una descripción básica del requisito que ha sido identificado.
- **Requisitos previos**: Necesidades que deben estar desarrolladas para que se realice la acción.

Las 6 acciones que describen el proyecto y que aparecen en el diagrama de secuencia son las siguientes:

Código	ACT-001
Actor	Usuario
Título	Iniciar el programa
Descripción	El usuario dará comienzo al programa lanzando una instrucción al ordenador, ejecutará el código Python en la consola de Ubuntu.
Requisitos previos	• El robot debe estar encendido y conectado a la red. • El ordenador debe estar conectado a la misma red que el robot.

Tabla 3.1: ACT-001. Iniciar

Código	ACT-002
Actor	Ordenador y NAO
Título	Capturar imagen
Descripción	El ordenador manda una orden al robot NAO para que éste realice una captura de la imagen que esté visualizando, es decir, de la partitura. Se realizará con la cámara que tiene éste en su frente ya que tiene mayor resolución.

Requisitos previos	• El robot debe estar encendido y conectado a la red. • El ordenador debe estar conectado a la misma red que el robot. • El usuario debe haber iniciado la ejecución del programa.

Tabla 3.2: ACT-002. Capturar imagen

Código	ACT-003
Actor	Robot NAO
Título	Enviar imagen
Descripción	NAO debe mandar al ordenador la imagen capturada para que éste se encargue de procesarla y analizarla.
Requisitos previos	• El robot debe estar encendido y conectado a la red. • El ordenador debe estar conectado a la misma red que el robot. • El robot NAO ha realizado una captura de la imagen que está visualizando.

Tabla 3.3: ACT-003. Enviar imagen

Código	ACT-004
Actor	Ordenador
Título	Ordenar sonidos
Descripción	El ordenador ya conoce cuáles son las notas que se encuentran en la partitura. NAO posee en su memoria los audios correspondientes a cada uno de los tonos. El ordenador envía al robot órdenes para que reproduzca las notas precisas y en el orden que aparece en la partitura.
Requisitos previos	• El robot debe estar encendido y conectado a la red.

	• El ordenador debe estar conectado a la misma red que el robot. • La imagen debe haber sido analizada por el ordenador.

Tabla 3.4: ACT-004. Ordenar sonidos

Código	ACT-005
Actor	Robot NAO
Título	Reproducir sonidos
Descripción	NAO finaliza el programa mediante la reproducción de los archivos de música, que se encuentran almacenados en su memoria interna, y que siguen las ordenes mandadas por el ordenador
Requisitos previos	• El robot debe estar encendido y conectado a la red. • El ordenador debe estar conectado a la misma red que el robot. • El ordenador debe haber mandado las directrices de las notas que forman la partitura.

Tabla 3.5: ACT-005. Reproducir sonidos

Código	ACT-006
Actor	Humano
Título	Reiniciar programa
Descripción	Una vez que NAO ha interpretado las notas de la partitura, el humano tocará el sensor que tiene el robot en su cabeza para que vuelva a realizar todo el proceso. De esta manera puede volver a leer una nueva partitura.
Requisitos previos	• El robot debe estar encendido y conectado a la red.

	• El ordenador debe estar conectado a la misma red que el robot. • El ordenador debe de haber finalizado su interpretación musical.

Tabla 3.6: ACT-006. Reiniciar programa

3.2.5 Especificación de requisitos

En este último apartado de análisis del sistema se van a describir con detalle los requisitos identificados. Al igual que se ha realizado en el apartado anterior, para cada uno de ellos se mostrará una tabla con los siguientes atributos:

- **Código:** Identificación unívoca abreviada del requisito, se construye mediante el código del requisito RS, seguido de un - y de tres dígitos. Por ejemplo, RS-001.
- **Descripción**: Se realiza una descripción básica del requisito que ha sido identificado.
- **Necesidad**: Determina el grado de implementación del requisito. Los valores que puede tomar este atributo son los siguientes:
  - Esencial: El requisito tiene que ser implementado.
  - Deseable: Es preferible implementar el requisito, pero no es obligatorio.
  - Opcional: El requisito se podrá implementar, pero no es importante ni obligatorio.
- **Prioridad**: Define la importancia del requisito, de forma que permita definir el orden en el cual serán incluido en el proceso de diseño y el orden de implementación. Los valores que puede tomar este atributo son los siguientes:
  - Alta: El requisito debe ser implementado en las fases iniciales del desarrollo.
  - Media: El requisito debe ser implementado una vez que hayan sido implementados los requisitos de prioridad alta.

Para que quede todo ordenado se dividirán en varias clases: entorno de trabajo, actividades y partitura.

3.2.5.1 Entorno de trabajo

Código	RS-001
Nombre	Sistema operativo
Descripción	El sistema operativo necesario para el desarrollo del trabajo es Ubuntu 14.04
Necesidad	Esencial
Prioridad	Alta

Tabla 3.7: RS-001. Sistema operativo

Código	RS-002
Nombre	NAO H25
Descripción	Robot humanoide para el cuál se va a implementar el código
Necesidad	Esencial
Prioridad	Alta

Tabla 3.8: RS-002. NAO H25

Código	RS-003
Nombre	Lenguaje de programación
Descripción	Código desarrollado en lenguaje Python
Necesidad	Esencial
Prioridad	Alta

Tabla 3.9: RS-003. Lenguaje de programación

Código	RS-004
Nombre	Choregraphe
Descripción	Software instalado en el ordenador. Versión 2.1.019
Necesidad	Esencial
Prioridad	Alta

Tabla 4.10: RS-004. Choregraphe

Código	RS-005
Nombre	NaoQi
Descripción	Framework que recrea la simulación de un robot NAO. Versión 1.14.5:
Necesidad	Esencial
Prioridad	Alta

Tabla 3.11: RS-005. NaoQi

Código	RS-006
Nombre	Comunicación ordenador-robot
Descripción	Router que crea la red de área local a la que se tendrán que conectar tanto el ordenador como el robot. La conexión puede ser inalámbrica por WiFi o por cable mediante ethernet.
Necesidad	Esencial
Prioridad	Alta

Tabla 3.12: RS-006. Comunicación ordenador-robot

3.2.5.2 Actividades

Código	RS-007
Nombre	Posición inicial
Descripción	Se definirá una posición de inicio enfrente del atril para que el robot visualice correctamente la partitura
Necesidad	Esencial
Prioridad	Alta

Tabla 3.135: RS-007. Posición inicial

Código	RS-008
Nombre	Capturar imagen
Descripción	El robot deberá visualizar la imagen con la cámara de la frente y realizar una captura en calidad HD.
Necesidad	Esencial
Prioridad	Alta

Tabla 3.14: RS-008. Capturar imagen

Código	RS-009
Nombre	Analizar imagen
Descripción	Desde el ordenador se realiza un procesamiento digital de la imagen que se compone por: binarizado, empleo de la transformada de Hough y algoritmos para calcular la nota deseada en función de su posición.
Necesidad	Esencial
Prioridad	Alta

Tabla 3.15: RS-009. Analizar imagen

Código	RS-010
Nombre	Interpretación
Descripción	El robot canta las notas que hay en la partitura.
Necesidad	Esencial
Prioridad	Alta

Tabla 3.16: RS-010. Interpretación

3.2.5.3 Partituras

Código	RS-011
Nombre	Pentagrama
Descripción	La partitura debe estar compuesta por un pentagrama de cinco líneas paralelas equispaciadas.
Necesidad	Esencial
Prioridad	Alta

Tabla 3.17: RS-011. Pentagrama

Código	RS-012
Nombre	Notas musicales
Descripción	Las notas musicales que aparezcan en el pentagrama deben ser negras, es decir formadas con un circulo negro y una plica.
Necesidad	Esencial
Prioridad	Alta

Tabla 3.18: RS-012. Notas musicales

Código	RS-013
Nombre	Atril
Descripción	La partitura deberá situarse sobre un atril regulable para que el robot sea capaz de visualizar toda la partitura.
Necesidad	Deseable
Prioridad	Alta

Tabla 3.19: RS-013. Atril

Código	RS-014
Nombre	Iluminación
Descripción	Para que el robot pueda obtener una imagen de buena calidad es necesario que la partitura esté bien iluminada. Es por ello que se requiere un entorno luminoso o el empleo de luz artificial
Necesidad	Esencial
Prioridad	Alta

Tabla 3.20: RS-014. Iluminación

3.3 Diseño del sistema

En este apartado se presenta el diseño arquitectónico de la aplicación. Se comenzará describiendo todos los componentes del mismo, entrando en profundidad en sus detalles. Se continuará estudiando la arquitectura del sistema y por último se describirán todos los procesos que se realizan en el sistema.

3.3.1 Descripción de componentes

En esta sección se presentan los diferentes componentes que forman parte del sistema describiendo sus características y sus funcionalidades. Es imprescindible la presencia de

todos ellos en la arquitectura, pues si no fuera así, sería imposible la ejecución del proyecto.

3.3.1.1 Robot NAO H25

El robot NAO H25 es el indiscutible actor principal de todo el proyecto. Todo el desarrollo del sistema está pensado para que NAO realice unas determinadas acciones. Como se puede ver en la figura 3.3, NAO H25 es un robot humanoide creado por la empresa francesa Aldebaran Robotics.

Tiene un gran éxito a nivel internacional donde es ampliamente conocido. Dispone de múltiples sensores (micrófonos, cámaras, giroscopio, acelerómetro, etc.) y actuadores (altavoces, motores, etc.).

Tiene 25 grados de libertad, por lo que posee una amplia movilidad.

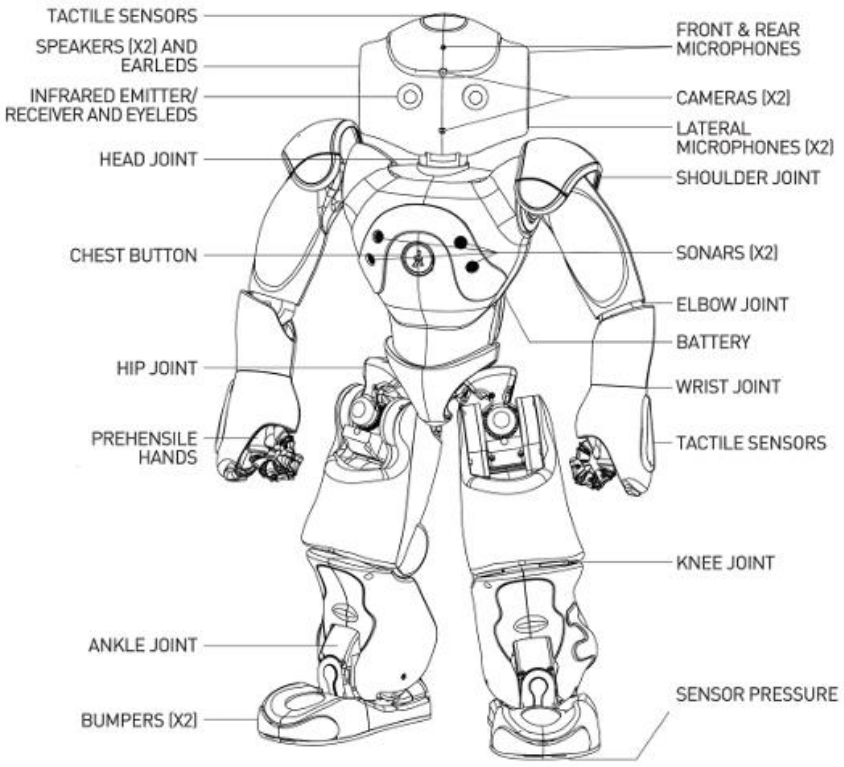


FIGURA 3.3: Diseño de NAO H25

Posee ocho partes principales en cuanto a articulaciones. Con ellas se controla su movimiento:

- Head: La cabeza del robot puede moverse de arriba a abajo (pitch) y de izquierda a derecha (yaw).
- Torso: En el torso se encuentran los cuatro sonar, el giroscopio y el acelerómetro del robot. Estos elementos le permiten prevenir el choque con obstáculos o equilibrarse entre otros. Los cuatro sonar son: dos emisores (transmitter) y dos receptores (receiver).
- RArm: El Right arm o brazo derecho permite mover el hombro (shoulder) sobre sí mismo (roll) o de arriba a abajo (pitch) y el codo (elbow) sobre sí mismo (roll) o de izquierda a derecha (yaw).
- RHand: La Right hand o mano derecha permite abrir y cerrar los dedos de la mano (RHand) y girar la muñeca sobre sí misma (RWristYaw).
- RLeg: La Right leg o pierna derecha permite girar y mover la pierna derecha (RHipRoll, RHip- Pitch), mover la rodilla de arriba a abajo (RKneePitch) y mover y girar el tobillo (ankle). El movimiento de RHipYawPitch es el mismo que el de LHipYawPitch, pues se mueven las dos piernas a la vez. Por lo tanto, comparten el mismo valor y podemos utilizar indistintamente uno u otro.
- LArm: El Left arm o brazo izquierdo permite mover el hombro (shoulder) sobre sí mismo (roll) o de arriba a abajo (pitch) y el codo (elbow) sobre sí mismo (roll) o de izquierda a derecha (yaw).
- LHand: La Left hand o mano izquierda permite abrir y cerrar los dedos de la mano (LHand) y girar la muñeca sobre sí misma (LWristYaw).
- LLeg: La Left leg o pierna izquierda permite subir y bajar ambas piernas (LHipYawPitch), girar y mover la pierna izquierda (LHipRoll, LHipPitch), mover la rodilla de arriba abajo (LKneePitch) y mover y girar el tobillo (Ankle).

3.3.1.2 NaoQi

NaoQi 1.14.5 es el framework desarrollado por Aldebaran Robotics que actúa de enlace entre el robot NAO y el ordenador, controlando la ejecución de cualquier comportamiento y capturando la información de todos los sensores con el único objetivo de que todo se produzca con éxito y sin incidentes. Es el soporte para que todo funcione.

Este framework se encuentra dentro del robot, a modo de cerebro, y también en el

ordenador desde el que se envían las órdenes. El framework se encuentra dividido en varias áreas. En este trabajo vamos a utilizar las siguientes:

- **NAOqi Core:** formado por un conjunto de módulos relacionados con los comportamientos genéricos del robot, la memoria del robot y los ficheros de configuración.
- **NAOqi Sensors:** contiene los módulos que proporcionan información sobre el estado del robot: batería, sensores, pose en la que se encuentra, etc.
- **NAOqi Motion:** dispone de todas las funciones relacionadas con el movimiento y permiten al robot desplazarse, controlando el equilibrio, el estado de las articulaciones y la prevención de caídas y choques.
- **NAOqi Vision:** formado por los módulos de visionado que permiten usar las cámaras del robot y realizar reconocimiento de caras y de objetos.
- **NAOqi Audio:** contiene los módulos que le permiten al robot interactuar con el usuario mediante la reproducción de sonidos, el habla y el reconocimiento de voz.

3.3.1.3 Choregraphe

Choregraphe versión 2.1.0.19 es un entorno de programación desarrollado por Aldebaran Robotics muy sencillo e intuitivo, ya que la programación se realiza mediante bloques. Posee una gran variedad de acciones, ya que permite programar acciones simples, complejas, mantener diálogos e incluso reconocimiento de objetos y de personas a través de sus cámaras. Es compatible con Windows, iOS y GNU/Linux Ubuntu 14.04.

Las acciones de NAO se programan a través de diagramas de flujo, encadenando unos bloques o cajas unos tras otros, pudiendo realizar varias tareas a la vez. Todas estas acciones están contenidas en librerías de cajas donde el usuario podrá crear nuevas, modificarlas y borrarlas. En la figura 3.4 se puede observar un ejemplo de programación por bloques.

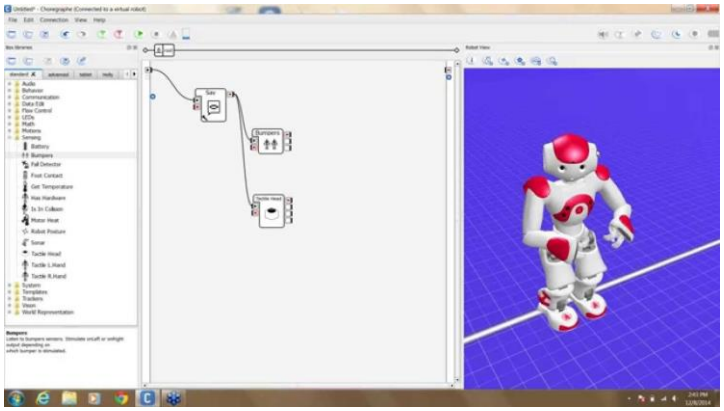


FIGURA 3.4: Programación por bloques en Choregraphe

Para los programadores más expertos existe la posibilidad de ver el código Python de cada caja simplemente haciendo doble clic, con lo que pueden realizar todos los cambios que necesiten e incluso pueden crear cajas de cero.

3.3.1.5 Router

El router será el encargado de crear la red local o LAN a la que deben estar conectados tanto el ordenador como el robot NAO. Cualquier router estándar comercial es válido. Es requisito fundamental que ambos estén en la red y haya conectividad entre ellos. La conexión será normalmente inalámbrica por Wi-Fi a la red NAOLAN creada en el laboratorio, aunque también es posible una conexión alámbrica mediante un cable UTP.

3.3.2 Arquitectura del sistema

El sistema está formado por tres subsistemas que se puede visualizar en la figura 3.5: capturar imagen, procesar imagen y reproducir sonidos. Se han barajado diferentes opciones para realizar cada una de ellas, las cuales se han probado, estudiado y decidido si eran la opción que cumplía los objetivos deseados.

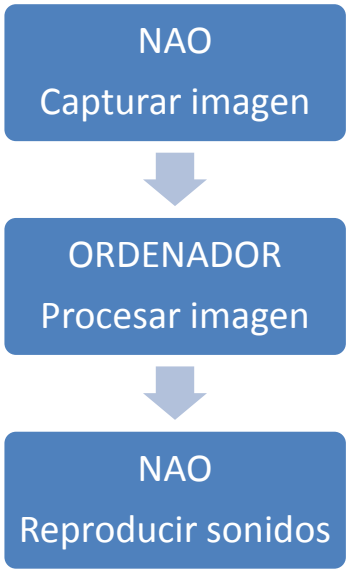


FIGURA 19.5: Sistema general

3.3.2.1 Capturar imagen

El ejecutor de este módulo es el robot NAO. Como su propio nombre indica, el objetivo es obtener una captura digital de la partitura que observa NAO. El principal requisito es que la captura sea lo más nítida posible para que su posterior procesado sea lo más efectivo posible.

Siguiendo las restricciones hardware que posee el NAO, se empleará la cámara con mayor resolución, es decir la cámara con calidad HD de 1280x960 píxeles, situada en su frente.

La cámara que posee en su boca es de calidad SD de 720×576 píxeles. Ambas se pueden ver en la figura 3.6.

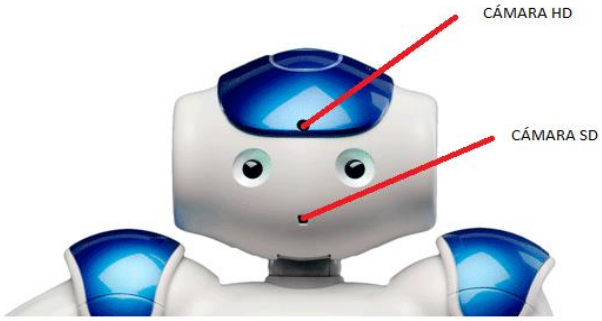


FIGURA 3.6: Cámaras HD y SD del robot NAO

En primer lugar, NAO se debe colocar de manera correcta frente a la partitura, tal y como se ve en la figura 3.7. Para ello, la primera instrucción que mandará el ordenador al robot será la de levantarse, de modo que éste quede en frente de la partitura.

Previamente, el atril estará regulado para que la altura de la partitura coincida con la visión que ofrece la cámara frontal del NAO.

Esto es fundamental, pues si la toma de imagen es incorrecta, todo el proceso siguiente será en vano.

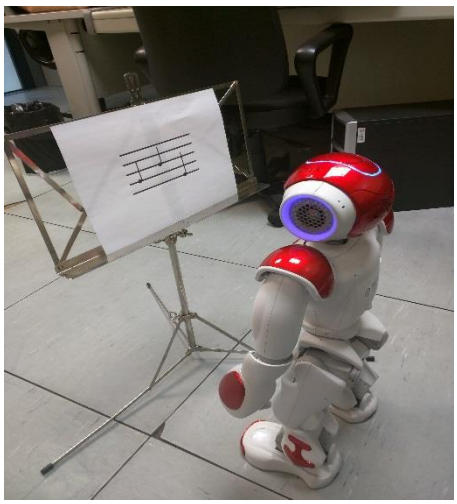


FIGURA 3.7: Posición inicial

Se controlará la visión de la cámara mediante la herramienta Choregraphe, de modo que quede centrada la partitura en su rango de visión. En la figura 3.8 se observa este control desde dicho software.

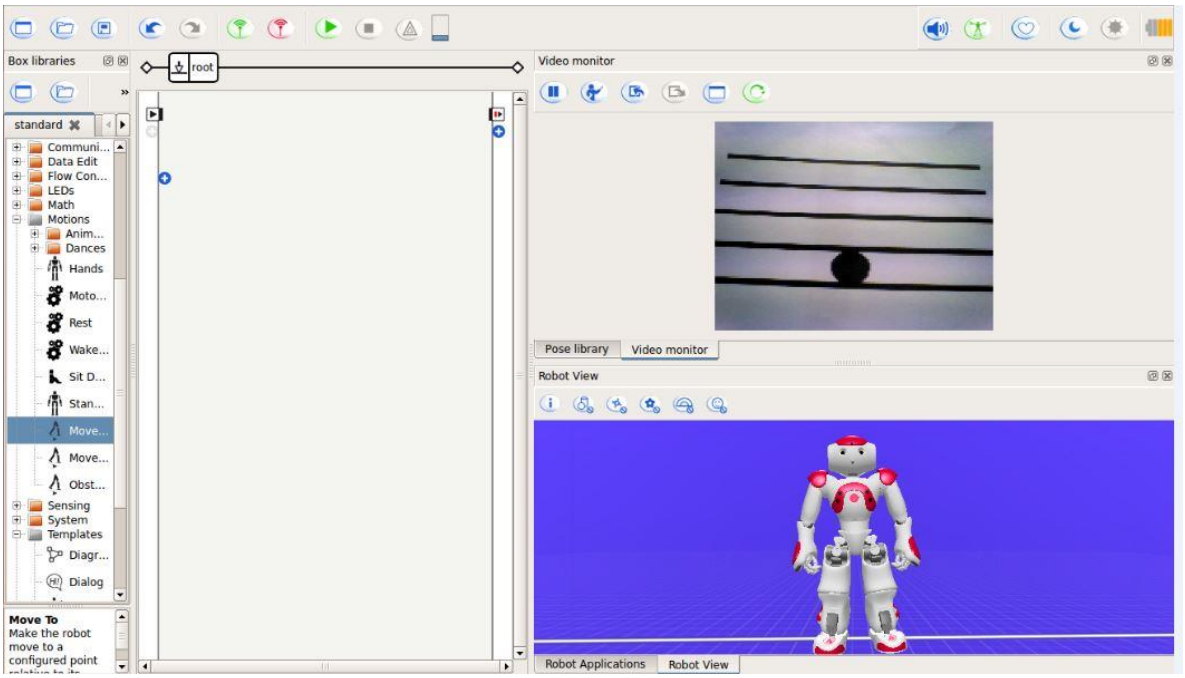


FIGURA 3.8: Visión con Choregraphe

Otro aspecto fundamental a la hora de tomar la imagen es la iluminación. Es necesario que la cantidad de luminancia en la sala sea la mayor posible para que la captura sea lo más clara que se pueda.

Si las condiciones luminosas no son suficientes para una buena toma de imagen, se ha de emplear luz externa como una lámpara o un fluorescente.

Por último, la imagen será enviada vía Wi-Fi al ordenador para comenzar el procesado. La figura 3.9 muestra una captura de una nota tomada por NAO.

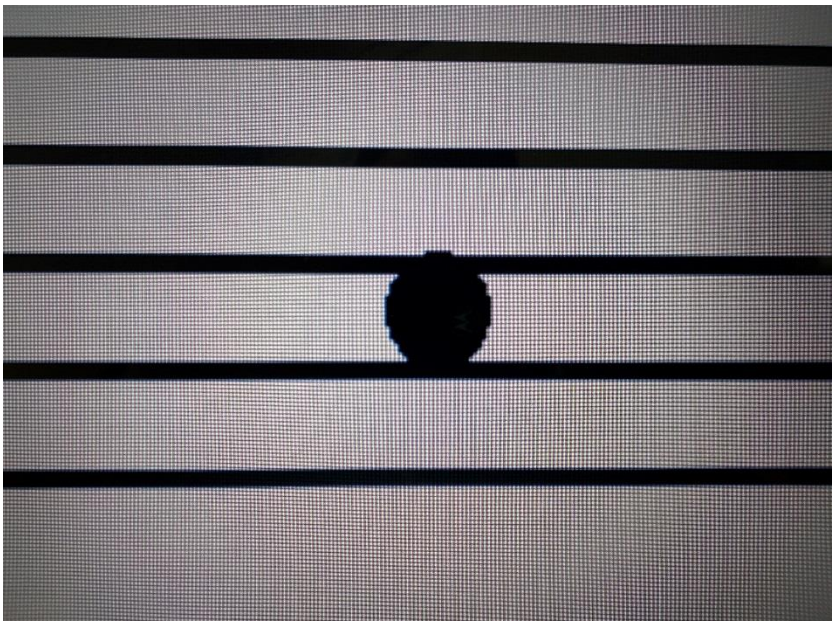


FIGURA 3.9: Nota capturada por NAO

3.3.2.2 Procesar imagen

La imagen capturada en el punto anterior será enviada por parte del robot NAO al ordenador mediante su conexión, normalmente Wi-Fi. El siguiente proceso será por parte del ordenador, el cual se encargará de procesar y analizar la imagen, a modo de cerebro del NAO.

En un principio se pensaron dos soluciones: primero utilizar una herramienta software existente que fuera capaz de analizar mediante su tecnología una imagen dada y obtener las notas, y segundo, desarrollar un analizador de imágenes en código Python, propietario para NAO y para este proyecto.

Como punto de inicio se realizó la búsqueda de la herramienta analizadora de partituras. El software con el que se hicieron las pruebas fue Musescore [27], del cual se ha hablado anteriormente.

Musescore emplea tecnología OCR (Optical Character Recogition) empleada para convertir imágenes en texto. Por tanto, una captura de una partitura mediante esta técnica puede

ser editada, cosa que, si es una imagen, no. Esto sería de gran ayuda en nuestro trabajo, pero la dificultad en obtener resultados fiables es muy alta.

Se comenzó importando una imagen capturada de una partitura y se procedió a su análisis. Los resultados dejaron bastante que desear pues no se asemejaba en ningún caso a la partitura original.

Se probaron otros descriptores gráficos, pero con resultados aún más deficientes que los obtenidos con Musescore. Es por ello que esta opción fue descartada rápidamente en favor de la segunda.

Como ya se ha dicho, la solución alternativa era programar un código en lenguaje Python que, a partir de la imagen capturada por las cámaras de NAO, fuera capaz de analizar todos sus parámetros y supiera reconocer las notas que se veían en la partitura, para una vez detectadas, mandarle a NAO la información para que éste las reproduzca.

Esta parte sería como el cerebro de todo el sistema. Al igual que hace un humano cuando visualiza una partitura y detecta el pentagrama, la clave, el tiempo, el tipo de nota y el nombre de la misma, estos procesos son los que debe de realizar nuestro programa de visión artificial a modo de cerebro para NAO.

Este procesamiento de la imagen está formado por varias etapas en función del desarrollo del sistema:

1. La primera etapa que se desarrolló fue detectar las notas siguiendo el cifrado americano. Se pueden observar en la figura 3.10. La idea inicial era detectar letras como si fueran notas musicales, de tal forma que cuando viese una letra A, NAO supiera que es una nota la.

   El objetivo además de reconocer la letra que es, era que lo hiciera independientemente del ángulo de rotación que tuviera dicha letra.

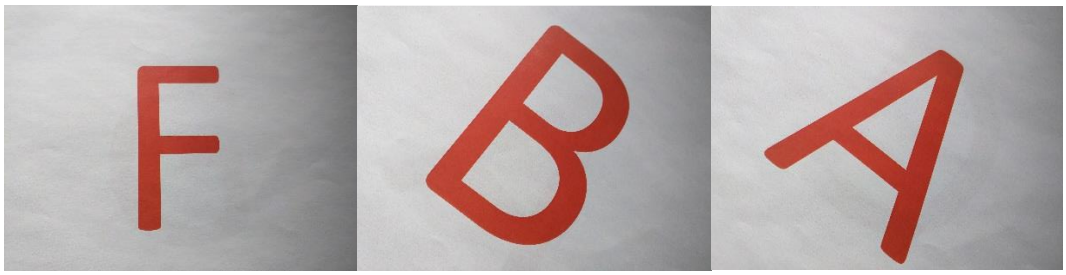


FIGURA 3.10: Notas con cifrado americano

En este apartado se trabajó en dos fases:

- Primero se empleó Choregraphe gracias a una herramienta de detección de figuras. Guardando la imagen en Choregraphe, NAO era capaz de encontrarla una vez que era vista por su cámara. Esta primera actividad sencilla sirvió para familiarizarse con Choregraphe e ir comenzado el análisis de las imágenes.

- En La segunda fase, se emplearon dos librerías de visión artificial como son OpenCv y PIL sobre código Python. Ambas poseen funciones desarrolladas para detectar líneas y formas determinadas.

Las pruebas fueron satisfactorias y una vez obtenido una tasa de acierto bastante alta, se procedió a trabajar en detectar notas sobre un pentagrama.

2. En la segunda etapa ya se trabaja sobre notas musicales y se programa un código capaz de detectar las cinco líneas de pentagramas y un círculo negro, tal y como se puede observar en la figura 3.7. La identificación de cada nota será a partir de su altura en la imagen. Se establecen varios procesos clave:

    1. Lo primero de todo es binarizar la imagen para eliminar el ruido producido por la captación de la imagen. De esta manera, quedará

reducida a blancos y negros puros donde se distinguirán claramente las líneas del pentagrama y las notas sobre el mismo. Por consiguiente, se aumentará el nivel de efectividad del programa.

Se ha desarrollado una función que establece un margen de división, un decisor binario. Establece que para cada pixel si es menor que un cierto valor adopte drásticamente el valor 0, o lo que equivale a un negro puro, y si es mayor, que valga 255, o blanco puro, siguiendo la escala de colores RGB. De esta forma el tratamiento de la imagen será mucho más precisa y cómoda. Este tratamiento se realizará antes de continuar con la Transformada de Hough.

2. El siguiente paso tras el binarizado es detectar las 5 líneas del pentagrama. Para ello emplearemos la técnica basada en la transformada de Hough. Esto permite detectar todas las líneas relativamente horizontales. Pueden tener una cierta desviación respecto a la horizontal, debido a que obtener una fotografía con el robot a 0º respecto la horizontal es muy difícil. Por otro lado, así el programa es más robusto. El número de líneas detectadas será excesivamente alto, tal y como se ve en la figura 3.11, ya que el objetivo es únicamente quedarnos con las cinco que hay en el pentagrama.

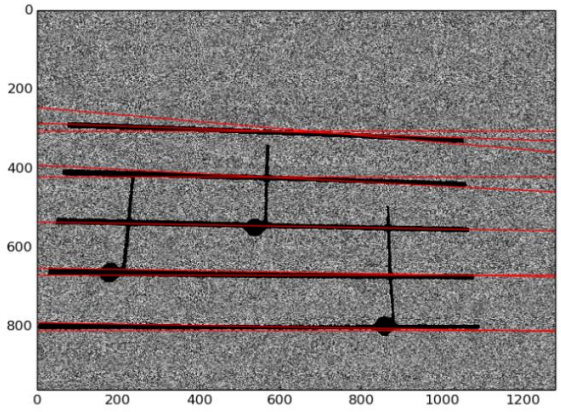


FIGURA 3.11: Multitud de líneas horizontales detectadas

De esta forma, la mayor concentración de líneas se situará en cada una de las propias del pentagrama por lo que las que están situadas en la parte más alta y más baja de la imagen se descartan. El resto, mediante un algoritmo que elimine las rectas que están a muy pocos pixeles de distancia, hará obtener las cinco líneas del pentagrama.

El dato que nos interesa es la altura a la que se sitúan cada una de las cinco líneas. Como las notas pueden estar situadas en los espacios, no únicamente sobre cada línea, debemos saber la altura a la que está ese punto medio. Esta operación es bastante sencilla, pues solo es calcular el punto medio entre la línea inferior y la superior. Por tanto, se tienen once alturas, cinco pertenecientes a las cinco líneas y seis de los espacios. La escala irá desde el re4, o espacio inferior de la primera línea del pentagrama, hasta sol5 o el espacio superior a la quinta línea. El número al lado del nombre de la nota indica la octava en la que se encuentra.

El siguiente paso es detectar la cabeza de la nota. Para ello vamos a volver a hacer uso de la transformada de Hough para detectar círculos.

Se realizará una búsqueda de todos los círculos presentes en la imagen y devolverá la posición del pixel central, que será denominado centroide. La altura del centroide en la imagen es un dato fundamental para el análisis.

El último paso es conocer la nota. Para ello se va a hacer uso de las alturas obtenidas anteriormente, la de cada una de las cinco líneas y la del centroide. El algoritmo será restar en valor absoluto, puesto que lo que queremos es saber la distancia, del centroide a cada una de las nueve alturas tanto de líneas como de espacios. La que tenga un valor menor será la más cercana a esa altura y ya sabremos qué nota es.

Estos valores se almacenarán en un vector de distancias de once posiciones, debido a los once lugares donde puede estar cada nota. El orden es de más grave re4 a más aguda sol5. Como la posición con menor valor será la nota detectada, extraemos el índice de dicha posición y se iguala a un vector construido con el nombre de notas de la siguiente forma: notas = ['re4', 'mi4', 'fa4', 'sol4', 'la4', 'si4', 'do5', 're5', 'mi5', 'fa5', 'sol5']. Como ya sabemos en qué posición se encuentra, sólo hace falta extraer el valor del campo en este segundo vector con el nombre real de la misma.

3. La tercera y última etapa ha sido incluir la plica en las cabezas de las notas y leer varias notas sobre el pentagrama. Para ello el código se ha aumentado respecto al anterior con nuevos algoritmos.

   La existencia de plicas en las notas no ha supuesto ningún problema debido a que la búsqueda se hace únicamente de círculos, obviando las líneas verticales.

Para detectar varias notas, se va a realizar la búsqueda de varios círculos de la misma forma que en el anterior apartado pero introducido en un bucle, de tal forma que al encontrar una no se pare la búsqueda, si no que recorra la imagen entera.

Los centroides encontrados se almacenarán en un vector de centroides donde se guardarán todas las características de cada uno en cada posición. La búsqueda finaliza con un alto número de círculos, muchos de ellos cercanos entre sí. Eso no quiere decir que en cada uno de ellos exista una nota, sino que, por el error de la calidad de la imagen, detecta varias notas. Para simplificar y quedarse únicamente con los círculos existentes, se debe filtrar y eliminar todos aquellos que se encuentren a una cercanía de unos 25 píxeles, debido a que las notas van a estar espaciadas a una distancia considerable, alrededor de 150 píxeles. De esta manera sólo se tendrán dos, tres, cuatro o las notas que compongan la partitura, en nuestro caso un único compás sin duración.

A continuación, se deben ordenar las notas y para ello se va a usar la coordenada horizontal de cada centroide, de manera que el de menor será el que pertenece a la primera nota (la que se encuentra más a la izquierda), el siguiente sería el segundo y así respectivamente.

De esta manera se obtienen las notas ordenadas que forman la partitura para su posterior proceso que será el de reproducirlas mediante NAO.

3.3.2.3 Reproducir sonido

El último de los pasos es reproducir sonido. Una vez que desde el ordenador ya se conoce la o las notas que forman la partitura, es tarea del robot NAO reproducirlas.

El módulo de audio del robot forma parte de NaoQi. Para acceder a él ha sido necesario establecer un proxy utilizando: el nombre de la librería (AlAudioPlayer), la dirección IP de NAO, 192.168.1.179, y el puerto, 9559.

Una vez que esto ha ocurrido, NAO tiene almacenados en su memoria, unos archivos .wav con el tono propio de cada nota. Estos archivos han sido tratados anteriormente, es decir, empleando diferentes programas de música como Audacity y Adobe Audition.

Para la creación y el tratamiento de los archivos de audio se han realizado dos etapas: grabación y edición.

- **Grabación de las notas:** Los archivos bases de audios serán grabaciones realizadas de notas cantadas. La cantante fue Clara Luis Mingueza y con el uso de un micrófono y el software Audacity se recogieron los audios.
  Las notas grabadas fueron desde re4 hasta sol5, todos naturales, es decir sin sostenido ni bemoles. Las frecuencias de las 11 notas se muestran en la siguiente tabla 3.21:

NOTA	FRECUENCIA (Hz)
RE 4	293,664768
MI 4	329,627557
FA 4	349,228231
SOL 4	391,995436
LA 4	440
SI 4	493,883301
DO 5	523,251131
RE 5	587,329536

MI 5	659,255114
FA 5	698,456463
SOL5	783,990872

Tabla 3.21: Frecuencia notas musicales

- **Edición de los archivos de audio:** Tras estudiar varios programas de tratamiento de audio como Sibelius, Finale o Adobe Audition, se ha decidio emplear este último debido a su amplia gama de efectos sonoros. Para cada uno de los 11 archivos se han realizado los siguientes procesos, poniendo ejemplos para uno de ellos:

1. Lo primero de todo es importar el archivo en el software Adobe Audition. En la figura 3.12 se observa cómo el archivo está listo para su edición.

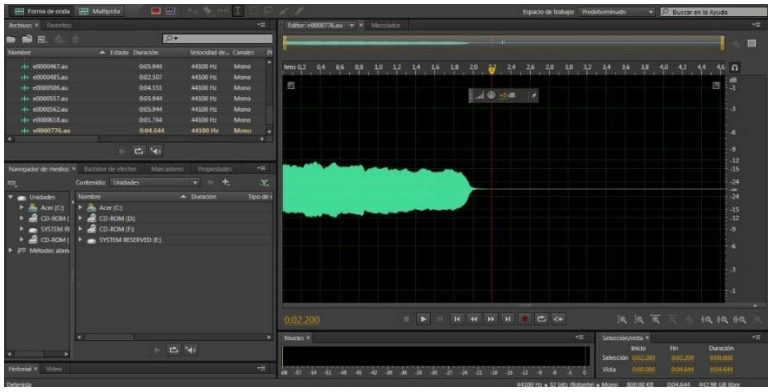


FIGURA 3.12: Adobe Audition

2. El primer procesado que se ha realizado es introducir reverberación al audio con el objetivo de evitar la planicie sonora. Existe un efecto personalizado denominado: Auditorio grande. Aplicando estos parámetros, la forma de la onda queda definida en la figura 3.13.

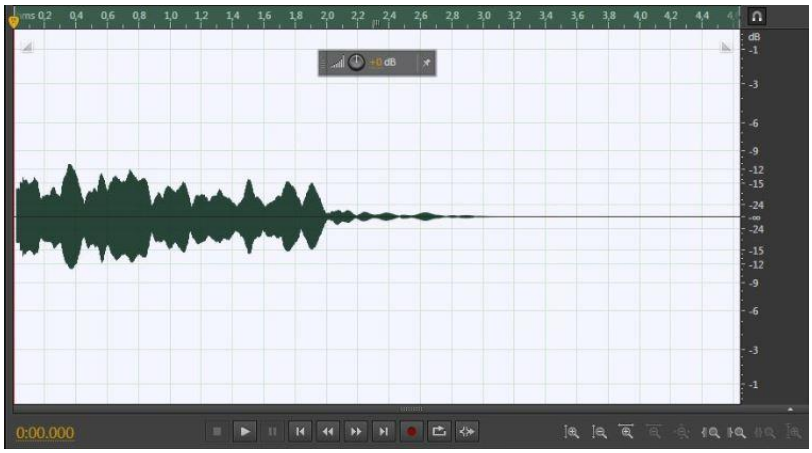


FIGURA 203: Reverberación

3. El siguiente efecto introducido es un bordeador. Crea un sonido sicodélico con desplazamiento de fase al mezclar un retardo variado corto con la señal original.

   Este efecto originalmente se conseguía enviando una señal de audio idéntica a dos grabadoras de cinta abierta y presionando periódicamente el borde de un carrete para hacerlo ir con más lentitud. Adobe Audition tiene desarrollado un bordeador para sonido robótico. La onda generada se observa en la figura 3.14.

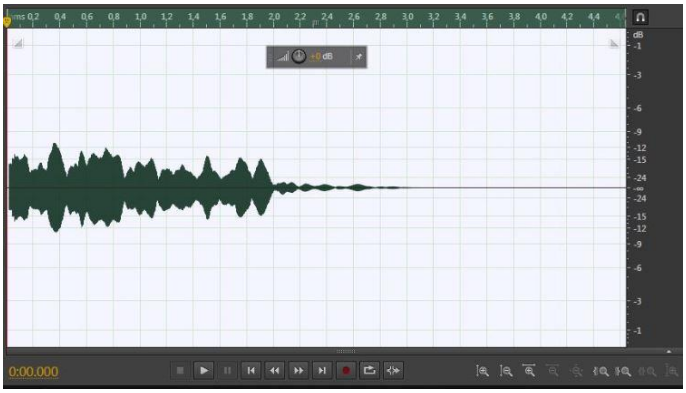


FIGURA 2114: Bordeador

4. El último proceso que se va a realizar es reducir la amplitud de la onda aplicando una atenuación de la señal modificando la amplitud en (-5) dB. De esta forma el audio no satura tanto tras aplicar el efecto de bordeado. La forma de onda del audio definitivo se puede observar en la figura 3.15.

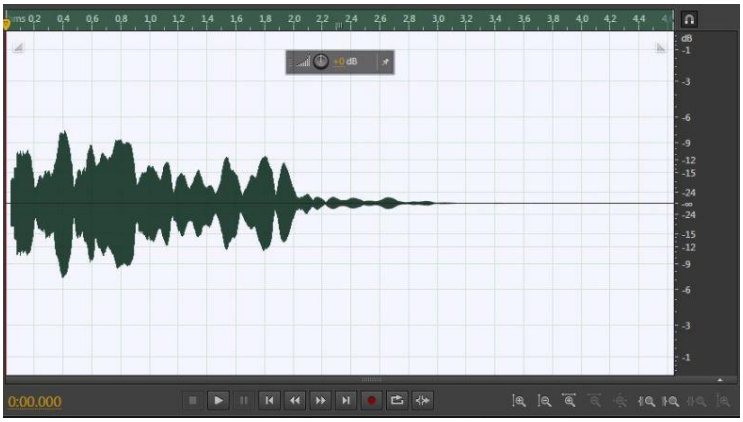


FIGURA 3.15: Audio final

Por último, el programa dará la orden de ejecutar los correspondientes archivos .wav para que NAO los reproduzca.

3.3.3 Descripción general del sistema

Una vez que se han explicado los tres subsistemas, se ofrece una visión completa de cómo funciona todo el sistema. Para entender mejor el funcionamiento, se va a mostrar en la figura 3.16 el diagrama de flujo general de la arquitectura.

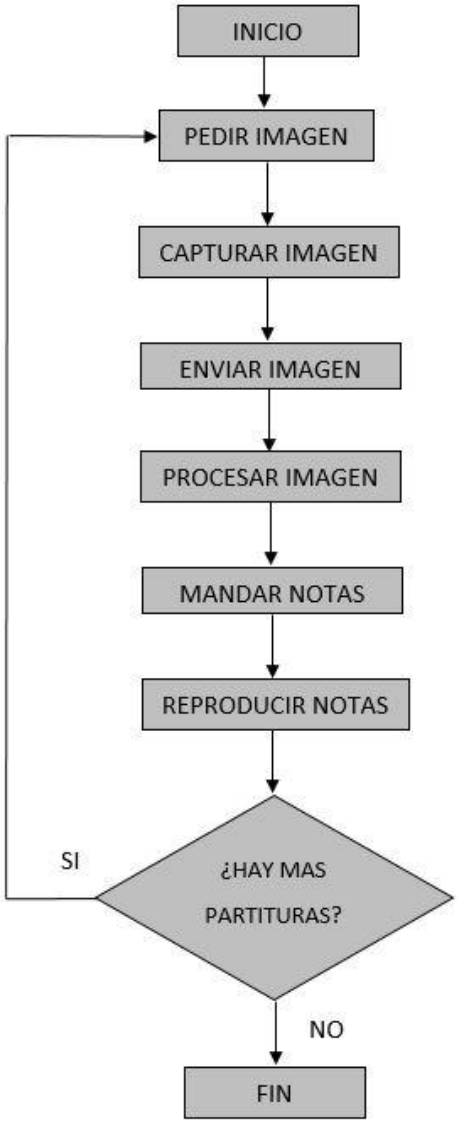


FIGURA 3.16: Diagrama de flujo del sistema

Se va a proceder a un análisis más detallado de cada una de las etapas:

- **Inicio:** es el punto de partida del programa y es llevado a cabo por el usuario que se encarga de ejecutar el código Python en la consola Linux del ordenador. Hay que destacar que anteriormente a todo esto es imprescindible que se cumplan dos requisitos:

- Tanto el ordenador como el robot NAO deben estar conectados a la red LAN creada por el router. Se puede comprobar según se describe en el apartado 3.3.1.5 Router.

- El robot NAO debe situarse en la posición inicial anteriormente descrita: de pie enfrente del atril con las partituras a la altura idónea. Todo esto se controla desde el software instalado en el ordenador Choregraphe.

- **Pedir imagen:** Una vez que se ha ejecutado el código, la primera instrucción es solicitar a NAO que realice una captura de su campo de visión. Como se ha aclarado con anterioridad, se pide que la foto la capture con la cámara de mayor resolución, la HD.

- **Enviar imagen:** Tras la obtención de la imagen, el siguiente paso es mandar dicha fotografía al ordenador para su posterior procesamiento. El medio de transporte será vía Wi-Fi, gracias a la conexión en red del ordenador y del robot.

- **Procesar imagen**: esta es la etapa más compleja y amplia de todo el proceso. Para una mayor comprensión se presenta un diagrama específico en la figura 3.17.

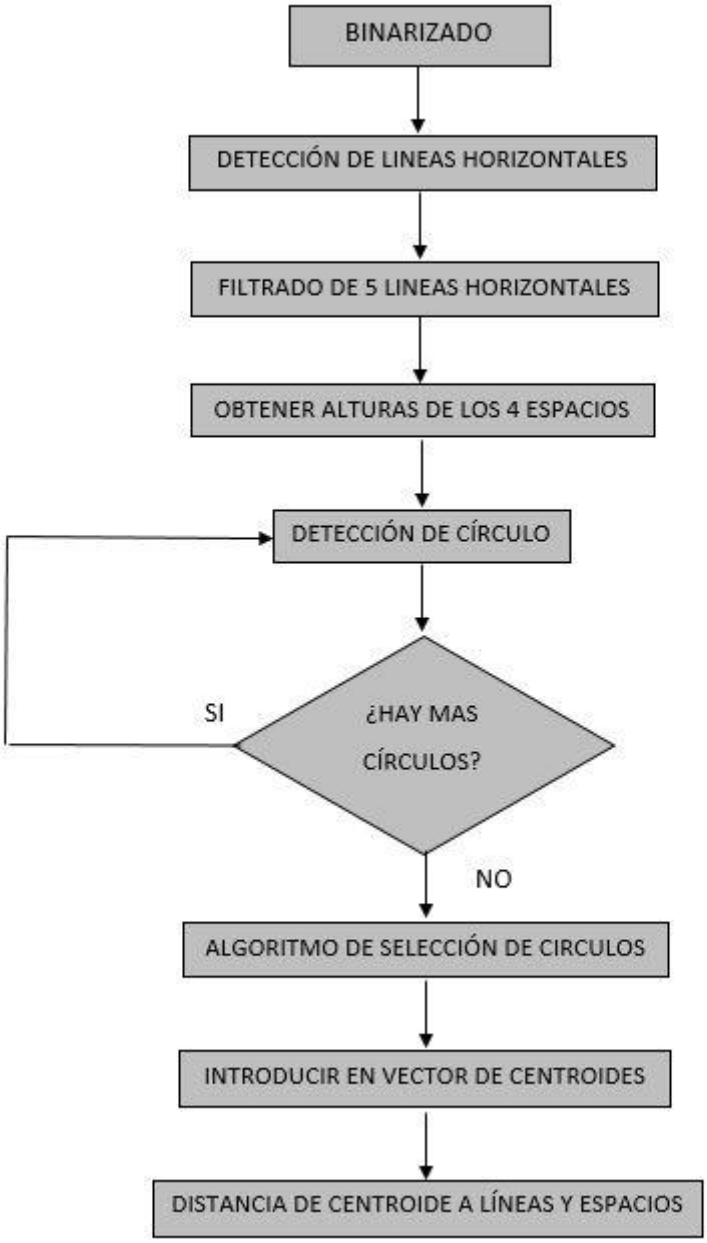


FIGURA .17: Diagrama de flujo de procesado de imagen

Destacar que la explicación de todos estos procesos se encuentra en el punto 3.3.2.2, pero se va a volver a insistir en estos pasos, pues son los definitivos para la finalización del proyecto.

A continuación, se va a entrar en detalle en los algoritmos desarrollados: basado en distancias, identificación de la nota y selección de círculos:

- **Algoritmo basado en distancias:** el objetivo es trabajar sólo con cinco líneas propias del pentagrama. Al hacer la búsqueda de líneas horizontales de la imagen mediante la transformada de Hough, el programa reconoce multitud de ellas debido al grosor que tienen las cinco líneas del pentagrama.

  Este algoritmo filtra en función de la distancia y realiza varias etapas:

  1. Lo primero de todo es crear un vector donde se van a almacenar las alturas de las 5 líneas seleccionadas.
  2. A continuación, se deben eliminar las líneas detectadas por el ruido en los extremos de la imagen. La altura de la captura es de 960 píxeles, y el pentagrama está centrado en ella, por lo que se realiza un filtrado para eliminar las líneas detectadas a una distancia de 50 píxeles desde los dos extremos, es decir, elimina las rectas encontradas entre 0 y 50 píxeles y entre 640 y 690 píxeles.
  3. Seguidamente, se tienen 5 grupos de varias líneas cada uno de ellos, por lo que se van a eliminar las sobrantes. Para ello, la primera y la quinta son sencillas de obtener, pues son la de mayor y menor altura. Se introducen ambas en el vector en la posición 0 y en la 4 respectivamente.
  4. Para identificar la tercera se realiza una media aritmética de las alturas de la primera y la quinta, ya que se encuentra en medio de ambas. Se introduce el valor de su altura en el vector en la posición 2.
  5. Por último, faltan obtener las líneas segundas y terceras. Al igual que en el apartado anterior, se calcula la media aritmética. La segunda mediante la primera y la tercera, y la cuarta mediante la tercera y la quinta. El algoritmo finaliza introduciendo la segunda altura en la posición 1 del vector y la cuarta en la 3.

- **Algoritmo de identificación de nota:** El fin de este algoritmo es saber qué nota es el círculo detectado. Para ello se hace uso del vector de alturas anterior y de los siguientes pasos:

1. El primer paso es la creación de un nuevo vector que contenga las alturas de todas las posibles notas, es decir las alturas de las líneas, ya calculadas, y los espacios. En total son 11 posibles notas, de re4 a sol5.
2. Para calcular los 6 espacios se hará uso de una variable denominada *paso*. Como las líneas del pentagrama son equidistantes, los espacios también tendrán siempre la misma distancia respecto a las líneas y a los demás espacios. La variable *paso* nos dará la distancia crítica entre un espacio y una línea. Para obtenerla basta con coger la altura de una línea y restarle el punto medio con la siguiente, es decir, donde estaría el espacio inferior.
3. A continuación, empleando la variable *paso* se va a rellenar el vector de alturas total. El orden será de más grave a más agudo, re4 a sol5. La primera nota, re4 se calcula restando paso a la primera línea y se almacena en la posición 0. La posición 1 tendrá el valor de la altura 1, la posición 3 será la suma de la altura 1 con la variable paso, la posición 4 almacenará el valor de la altura 2, y así sucesivamente hasta rellenar las 11 posiciones.
4. Por último, se va a crear una lista que almacenará en cada posición la distancia de la nota detectada con cada una de las 11 alturas. Esta operación es muy sencilla pues se obtiene la resta en valor absoluto de la altura de la nota con la de la posición en cuestión. Para saber que nota es la correcta, se extrae de este vector la posición que tenga menor distancia, pues será la más cercana a la nota. Este índice se buscará en un vector que contenga todos los nombres de notas es decir ‘re5’ en la posición 0, ‘mi5’ en la posición 1y así sucesivamente. De esta forma se obtendrá el nombre de la nota en cuestión.

- **Algoritmo de selección de círculos:** El objetivo de este algoritmo es detectar los círculos que forman las notas de la partitura. Al igual que ocurre en el anterior algoritmo, al realizar la búsqueda de círculos, se obtienen mucha cantidad de ellos concentrados cerca de un punto, que es donde está la nota real. Esto es debido a la dispersión de píxeles, también llamado ruido, tras la captura de la imagen.

  Este algoritmo tiene varios pasos:

  1. Lo primero de todo es crear una lista donde se almacenarán la posición horizontal del centro de cada de nota, a este concepto se le denomina centroide. Se tendrán tantos centroides como notas a interpretar.
  2. A continuación, se crea otro vector de posibles centroides, es decir, se almacenarán todos los círculos encontrados.
  3. El siguiente paso es filtrar el número de círculos. Para ello, se sabe que los círculos están separados una distancia suficiente, por lo que todos los círculos a menos de 25 píxeles de uno mismo, deberán eliminarse. Se coge el primer centroide y se calcula la distancia hacia todos los demás. Los que estén a menos de 25 píxeles quedan descartados y los que sí están se almacenan en un nuevo vector. Seguidamente, se hace el mismo proceso con el centroide de la segunda posición, ya que el de la primera es un centroide real porque no hay cercanos a él. Iterando estas operaciones, se acaba con un vector donde sólo existen los centroides reales.
  4. El último paso es ordenarlos de menor a mayor respecto a su coordenada horizontal, ya que la partitura se lee de izquierda a derecha.

- **Mandar notas:** Tras conocer el número y tipos de notas que forman la partitura, se manda la orden a NAO desde el ordenador para que éste sepa cuáles debe reproducir.

- **Reproducir notas:** Es el fin de todo el proceso. NAO se encarga de reproducir los archivos correspondientes, editados anteriormente y guardados en su memoria interna.

- **Pulsar sensor:** Por último, surge la posibilidad de que existan más partituras porque la canción no ha terminado, para ello se ejecuta de nuevo todo el proceso mediante el botón central que posee NAO en su cabeza y puede visualizarse en la figura 3.18. Cuando se pulse el botón central, el código comience de nuevo como si de un bucle se tratara.

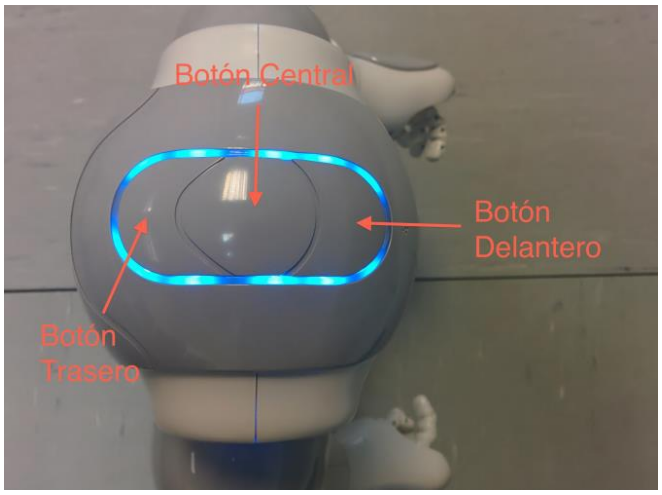


FIGURA 3.18: Sensores de NAO en la cabeza

Capítulo 4: Experimentación

En este apartado se recogen todas las pruebas realizadas al sistema anteriormente descrito los cuales han servido para el desarrollo del mismo y para probar cada una de sus funcionalidades por separado y de forma conjunta.

Se comienza estudiando el entorno en el que se realizaron las pruebas y después se explican los diferentes experimentos llevados a cabo con sus correspondientes soluciones.

Para realizar un seguimiento de la evolución del proyecto, se han realizado una serie de vídeos almacenados en el siguiente canal de YoutTube del equipo PLG *(Planning and Learning Group*) de la Universidad Carlos III de Madrid. La dirección web es la siguiente:

https://www.youtube.com/playlist?list=PLiD9gJlIfdWiSDty6NdC3KS-KBwyG6p0k

4.1 Entorno de pruebas

Todos las experimentos realizados se han llevado a cabo en el laboratorio 2.1.B16 del Edificio Sabatini perteneciente a la Universidad Carlos III de Madrid, en su campus de Leganés.

Un factor muy importante a tener en cuenta es que la luminosidad del laboratorio es siempre constante, debido a que su única fuente de luz es vía artificial mediante lámparas repartidas por todo el laboratorio. No posee ventanas ni puertas por los que pueda entrar luz solar. Esto proporciona una gran ventaja, pues al ser la cantidad de luz constante para todos los experimentos, ese parámetro no influía en el estudio del mismo.

Existen dos elementos indispensables que forman parte del proyecto. Estos son el atril y las partituras.

El atril es un soporte, en nuestro caso metálico, regulable en altura para la correcta visualización de las partituras. Dado que la altura de NAO es de unos 57 cm, la altura que tendrá el atril será de medio metro aproximadamente.

Las partituras están impresas sobre folios DIN A4 con las características mencionadas en el apartado 3.2.1. y se disponen como se observa en la figura 4.1.

FIGURA 4.1: Atril y partituras

A lo largo del desarrollo del proyecto, según se iban implementando nuevas funcionalidades para el sistema, se probaban de forma individual para comprobar que el funcionamiento era el esperado. En todos los casos, fue necesario rectificar parte de las mismas para conseguir dicho objetivo.

Estas pruebas de carácter sencillo comprobaban que las funciones implementadas cumplían los requisitos especificados en el capítulo 3.

A continuación, se detallan distintos experimentos que fueron realizados para conseguir las funcionalidades deseadas ya que los problemas que surgían se fueron solucionando con la experimentación.

4.2 Experimentos

En esta sección del capítulo se presentan los experimentos que se han llevado a cabo con las funcionalidades implementadas en el ordenador y cuáles eran sus efectos en el robot NAO. Para cada uno de ellos se describirán: los objetivos de dicho experimento y una descripción del mismo y del estado del entorno. Por último, se comentarán los resultados obtenidos y las dificultades encontradas.

4.2.1 Experimento 1

En este primer experimento se va a comprobar que NAO sea capaz de reconocer las letras que va visualizar en el atril. Es la fase inicial del módulo de visión y procesado digital de la imagen.

- Descripción del experimento:

Existen dos objetivos a alcanzar con este experimento:

1 Capturar una imagen con la cámara apropiada y enviarla al ordenador.
2 Analizar la imagen e identificar la letra que está en la partitura.

Para realizar las pruebas pertinentes, es necesario cumplir una serie de requisitos previos:

- Tanto NAO como el ordenador deben estar conectados a la red LAN con conectividad entre ellos.
- Se debe disponer de folios DIN A4 con las diferentes letras sobre un atril regulable a la altura apropiada para NAO.
- Poner en posición inicial a NAO, de pie delante del atril para comprobar mediante Choregraphe, que la cámara visualiza la letra del pentagrama.
- Correcto nivel de iluminación en el entorno de trabajo.

Para el objetivo número 1, se probará la función *captureImage()* implementada para mandar la orden a NAO de que capture la imagen que está visualizando con la cámara determinada, y recibirla en el ordenador.

Para el segundo objetivo, se implementará un código Python donde se hace uso de las herramientas OpenCV y PIL, descritas en el punto 2.3.2.1 y 2.3.2.2 respectivamente.

- Resultados y dificultades encontradas

Los resultados de estas pruebas fueron satisfactorios. El objetivo 1 se consiguió gracias a una serie de pruebas realizadas donde la mayor dificultad fue la conexión con NAO mediante el uso de varios proxys especificadas en la API de desarrollo del robot. En la figura 4.2 se observa la imagen de la letra capturada por NAO una vez que se encuentra en el ordenador.

FIGURA 4.2: Vista de una captura realizada por NAO desde el ordenador

El segundo objetivo fue más costoso de conseguir que el primero, pues aparte de la instalación de los paquetes necesarios, los algoritmos empleados en el código se fueron ajustando a los objetivos hasta que fueron alcanzados.

Se empleó el monitor de video que proporciona Choregraphe. Posee una función de reconocimiento de imágenes que permite detectar elementos. Para ello realiza una captura de un elemento y la guarda en su memoria. Cuando NAO ve este elemento,

Choregraphe le informa de que es conocido y por tanto se puede programar para que NAO haga una acción en ese momento.

4.2.2 Experimento 2

En el segundo experimento se va a trabajar ya con pentagrama y notas musicales. NAO debe ser capaz de analizar y extraer las características más importantes de una partitura compuesta por un pentagrama y una cabeza de nota.

- Descripción del experimento

El objetivo que se quiere alcanzar es que NAO detecte las cinco líneas del pentagrama, la cabeza de la nota y que averigüe qué nota es en función de la altura de la misma. En la figura 4.3 se observa una prueba realizada de este experimento en el que NAO lee una nota de un pentagrama sobre una caja a modo de atril.

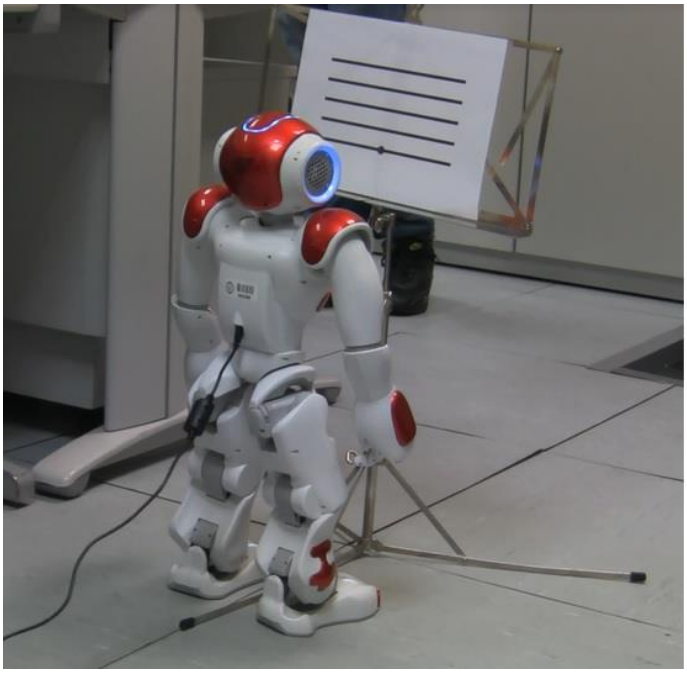


FIGURA 22: Experimento 2

Para realizar las pruebas pertinentes, es necesario una serie de requisitos previos:

- Tanto el NAO como el ordenador deben estar conectados a la red LAN con conectividad entre ellos.
- Tener folios DIN A4 con los pentagramas individuales para cada cabeza de nota sobre un atril regulable a la altura apropiada para NAO.
- Poner en posición inicial a NAO, de pie delante del atril para comprobar mediante Choregraphe, que la cámara visualiza la nota del pentagrama.
- Correcto nivel de iluminación en el entorno de trabajo.

En este experimento se realizan todas las pruebas necesarias para verificar que el programa empleado consigue su objetivo, utilizando partituras con todas las notas de la escala, para comprobar que es capaz de leer cualquiera.

- Resultados y dificultades encontradas.

Sin duda alguna, esta sección del trabajo ha sido la que más complejidad, cantidad de desarrollo y, por consiguiente, tiempo, ha tenido de todo el proyecto. El análisis de la imagen ha requerido multitud de pruebas y cambios en la algoritmia del programa. Primero consiguiendo objetivos individuales y luego mejorando la eficacia en los mismos.

Uno de los problemas más destacados es el tiempo de procesado. Al ser un análisis tan específico de la imagen se han necesitado multitud de bucles que aumentaban dicho tiempo. Uno de los trabajos que se ha realizado consiste en reducir este tiempo de procesado manteniendo la calidad del análisis. Para ello, se han mejorado los algoritmos para que las iteraciones de los bucles sean menores, manteniendo la misma funcionalidad.

Tras un intenso trabajo, los objetivos se consiguieron, obteniendo un primer descriptor de imagen para una única cabeza de nota sobre un pentagrama.

4.2.3 Experimento 3

En el tercer experimento se va a continuar con el trabajo obtenido en el anterior experimento. Tras adquirir esa base, el siguiente paso es añadir la plica a la cabeza de nota para que se forme una negra, y ser capaz de leer varias notas en la misma captura.

- Descripción del experimento

Existen dos objetivos a alcanzar con este experimento:

1 Detectar notas negras
2 Analizar varias notas en la misma imagen

Para realizar las pruebas pertinentes, es necesario una serie de requisitos previos:

- Tanto el NAO como el ordenador deben estar conectados a la red LAN con conectividad entre ellos.
- Tener folios DIN A4 con los pentagramas y varias notas sobre un atril regulable a la altura apropiada para NAO.
- Poner en posición inicial a NAO, de pie delante del atril para comprobar mediante Choregraphe, que la cámara visualiza la letra del pentagrama.
- Correcto nivel de iluminación en el entorno de trabajo.

En la figura 4.4 se observa la situación previa para realizar las pruebas necesarias.

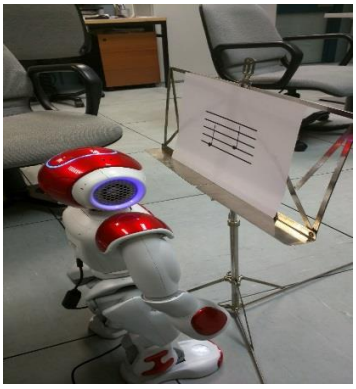


FIGURA 4.4: Experimento 3

Para conseguir estos objetivos ha sido necesario continuar con la implementación del código, teniendo como base el empleado en el experimento número 2. La solución ha sido emplear este algoritmo tantas veces como notas se identifiquen en la imagen, y mediante nuevo procesado programado, se obtiene los dos objetivos planteados.

- Resultados y dificultades encontradas.

Al igual que en el experimento 2, se ha necesitado emplear bastante tiempo debido a la complejidad y a la cantidad de funciones a implementar para conseguir los objetivos deseados. La mayor complejidad de esta evaluación fue realizar el bucle para detectar varias notas en la misma imagen, debido a que en un principio se detectaban muchísimas máscaras durante el procesado, es decir, muchas notas diferentes. Se implementaron funciones que actúan como filtros para quedarse únicamente con las notas especificadas en la partitura tal y como se explicó en el apartado 3.2.1.

Utilizando diferentes partituras en las que las notas variaban en cuanto al número existentes y al tipo, se fueron probando y ajustando los algoritmos de filtrado de círculos y de obtención de notas mediante el cálculo de las alturas. Mediante esfuerzo, trabajo y constancia se han obtenido los resultados deseados y se completa el analizador de la imagen del proyecto.

4.2.4 Experimento 4

En el cuarto experimento se van a realizar las pruebas necesarias relacionadas con el audio del robot NAO. Una vez que se han detectado el conjunto de notas, NAO debe ser capaz de reproducirlas.

- Descripción del experimento

El objetivo a alcanzar en este cuarto experimento es que NAO reproduzca los sonidos de las notas que anteriormente han sido detectadas y analizadas en la imagen.

Para realizar las pruebas pertinentes, es necesario una serie de requisitos previos:

- Tanto el NAO como el ordenador deben estar conectados a la red LAN con conectividad entre ellos.
- Tener folios DIN A4 con los pentagramas y varias notas sobre un atril regulable a la altura apropiada para NAO.
- Poner en posición inicial a NAO, de pie delante del atril para comprobar mediante Choregraphe, que la cámara visualiza la letra del pentagrama.
- Correcto nivel de iluminación en el entorno de trabajo.

Para conseguir este objetivo han sido necesario lograr dos etapas:

- Tratamiento de los ficheros de audio que tendrá NAO almacenados en su memoria interna.
- Realización de una función para conectar con el módulo de audio de NAO y mandar la orden de reproducir el conjunto de notas.

Estas pruebas se han realizado ejecutando todo el código anterior de manera que el módulo de audio entra en el proyecto en la última fase del mismo.

- Resultados y dificultades encontradas.

Los resultados obtenidos tras estas pruebas han sido satisfactorios. El audio tratado, como se ha especificado en los anteriores apartados tiene mucha relación con la estructura física del NAO, es decir, debido a su altura, se ha buscado una voz bastante aguda y con sonoridad metálica, propia de los robots.

Uno de los problemas estaba relacionado con la comunicación. Para que todo sea correcto, la conexión NAO – ordenador esté correctamente, pues si esto no es así no funcionará.

Por otro lado, se intentó que NAO reprodujera los archivos de audio estando éstos almacenados en la memoria interna del ordenador, cosa que se descartó rápidamente por la comodidad que ofrece que el almacenamiento se realice en la memoria interna de NAO, ya que, para su reproducción, únicamente hay que acceder a la ruta determinada de la memoria de NAO y mandar la orden desde el ordenador de reproducir los archivos correspondientes.

Capítulo 5: Gestión del proyecto

En este capítulo se va a describir cómo se ha llevado a cabo la gestión del proyecto. En primer lugar, se detallan las distintas fases de desarrollo del proyecto, indicando lo realizado en cada una de ellas. Por último, se muestra el presupuesto para el proyecto, especificando los costes materiales y de trabajadores que serían precisos para ejecutar el trabajo.

5.1 Planificación

En este apartado se explicará cómo se ha realizado la estimación de tiempo empleada en el desarrollo del proyecto. Seguidamente se describirá el modelo en espiral empleado, y por último se desarrollará la descripción de las tareas mediante el diseño de un diagrama de Gantt.

5.1.1 Metodología empleada

Como ya se ha descrito anteriormente, el proyecto tiene tres grandes fases: visión, procesamiento de imagen y reproducción de audio. Para llevar a cabo el desarrollo fue necesario emplear una metodología iterativa capaz de desarrollar cada etapa de forma independiente para que, una vez conseguidas, se pudieran unir y conseguir el objetivo final. Existían tres hipotéticas metodologías que fueron sometidas a estudio: incremental, prototipada y espiral.

- La metodología incremental está formada por dos etapas: inicialización e iterativa. Desde el principio era necesario elaborar un prototipo inicial, mejorándolo a medida que avanzaba el proyecto. Esto aumentaba notablemente el tiempo de ejecución.

- En la metodología prototipada, al igual que en la incremental, necesita un prototipo inicial avanzado y construido en poco tiempo.
- La metodología del modelo en espiral permite dividir el proyecto en fases casi independientes formada cada una de ellas por cuatro partes: especificación de objetivos, evaluación de alternativas y análisis de riesgo, desarrollo y pruebas, y, por último, planificación de la siguiente iteración.

A la hora de elegir uno u otro para el desarrollo del proyecto se ha seleccionado la metodología del modelo en espiral debido a la posibilidad de independizar cada una de las tres fases del trabajo.

5.1.2 El modelo en espiral

Este modelo de desarrollo de software fue diseñado por Barry Boehm en 1986 [39] y en él las diversas funciones de un proyecto forman una espiral, en la que cada bucle representa un conjunto de actividades que se realizan de forma iterativa.

El modelo en espiral permite realizar una aproximación al problema que se intenta resolver, en el que no se conocen desde el principio cuáles van a ser los requisitos finales del proyecto, debido al amplio número de campos que toca y a las limitaciones de tiempo que existen.

Para cada iteración de la espiral, es decir, una vuelta completa, se diferencian las siguientes actividades:

- Determinación de objetivos.
- Evaluación de alternativas y análisis de riesgos.
- Desarrollo y pruebas.
- Planificación de la siguiente iteración.

En la figura 5.1 se observa en qué cuadrante se encuentra cada una de las cuatro iteraciones:

FIGURA 5.1: Metodología en espiral

Dependiendo del sitio de la espiral donde se encuentre el trabajo, variará la realización de la tarea a realizar. Cada tarea forma parte de uno de los siguientes grupos: análisis, diseño, codificación, pruebas y documentación

- Análisis. Comprenden tareas como: estado del arte, estudio de tecnologías, identificación de requisitos, análisis de potenciales riesgos y posibles soluciones a los mismos, determinar el alcance de cada módulo de trabajo, entre otras.
- Diseño. Consisten en: elaboración de todo tipo de diagramas que relacionen el trabajo realizado en el análisis a la fase de codificación, tomar decisiones respecto al futuro funcionamiento del código programado.
- Codificación: incluyen tareas como: desarrollo del código del proyecto y el uso del hardware y software empleado para el trabajo.
- Pruebas: tareas como: sistema de evaluación de todo el proyecto mediante una batería de pruebas que compruebe el funcionamiento del mismo tal y como ha sido especificado en el capítulo 4 experimentación.

- Documentación: comprenden tareas como: redacción de la memoria del proyecto donde se pone por escrito todo lo relacionado con el mismo y los manuales desarrollados tanto como información como para su instalación.

5.1.3 Distribución de tareas

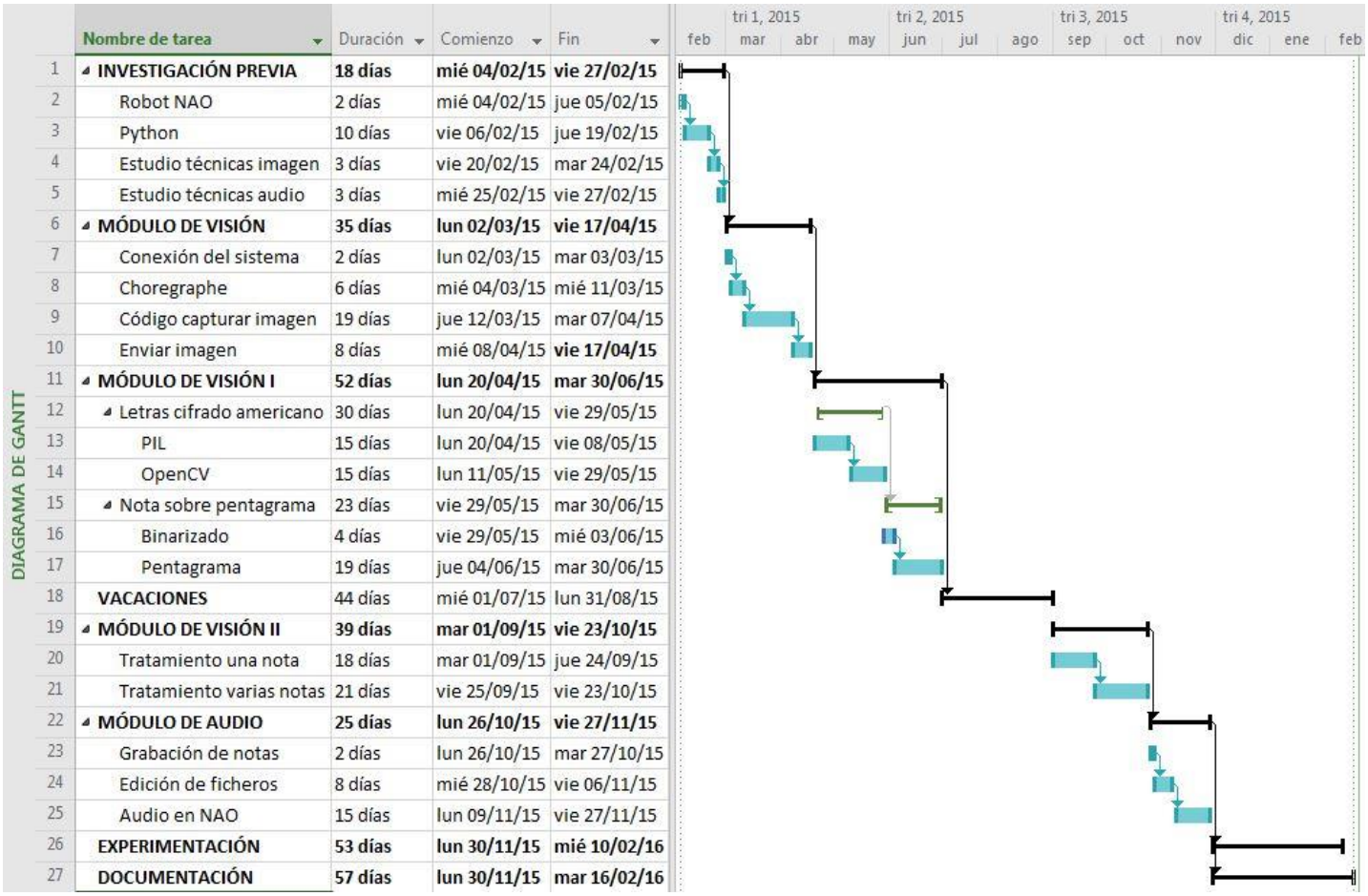
En este apartado se va a detallar la estimación del tiempo que se ha dedicado a cada tarea del proyecto. Se va a dar una visión de la distribución de tareas mediante un diagrama de Gantt.

El proyecto se inició el 4 de febrero de 2015 y se terminó el 17 de febrero de 2016. El trabajo duró aproximadamente un año.

En la planificación del mismo ha de tenerse en cuenta que sólo se han podido trabajar los días laborables ya que, al depender del robot que se encuentra en la universidad, no era posible avanzar en fin de semana, puentes o vacaciones.

Para iniciar el proyecto se necesitó un mes inicial para que el autor se informase en lenguaje Python, en el robot NAO y en todo lo relacionado con el entorno de trabajo, se ocuparon las fechas desde el 4 de febrero de 2015 hasta el 27 de febrero de 2015. Como se ha especificado anteriormente, el proyecto está basado en tres módulos independientes. El primero de ellos fue el de visión. Para este módulo se abarcó desde el 2 de marzo hasta el 17 de abril. Seguidamente se desarrolló el módulo del procesado digital de la imagen, que fue el que más tiempo consumió. Se comenzó el 18 de mayo y se terminó el 23 de octubre. Desde el 1 de julio hasta el 31 de agosto el proyecto queda interrumpido por vacaciones. El último módulo, el dedicado a la parte de audio, se realizó entre el 26 de octubre y el 27 de noviembre. A partir de estas fechas se realizaron las evaluaciones posibles, y en paralelo se realizó la correspondiente documentación. Estos dos procesos empezaron el 30 de noviembre, finalizando el apartado de la experimentación el 11 de febrero y la documentación el 17 de febrero de 2016.

Para tener una visión global más visual se va mostrar un diagrama de Gantt en la figura 5.2



	Nombre de tarea	Duración	Comienzo	Fin
1	**INVESTIGACIÓN PREVIA**	**18 días**	**mié 04/02/15**	**vie 27/02/15**
2	Robot NAO	2 días	mié 04/02/15	jue 05/02/15
3	Python	10 días	vie 06/02/15	jue 19/02/15
4	Estudio técnicas imagen	3 días	vie 20/02/15	mar 24/02/15
5	Estudio técnicas audio	3 días	mié 25/02/15	vie 27/02/15
6	**MÓDULO DE VISIÓN**	**35 días**	**lun 02/03/15**	**vie 17/04/15**
7	Conexión del sistema	2 días	lun 02/03/15	mar 03/03/15
8	Choregraphe	6 días	mié 04/03/15	mié 11/03/15
9	Código capturar imagen	19 días	jue 12/03/15	mar 07/04/15
10	Enviar imagen	8 días	mié 08/04/15	**vie 17/04/15**
11	**MÓDULO DE VISIÓN I**	**52 días**	**lun 20/04/15**	**mar 30/06/15**
12	Letras cifrado americano	30 días	lun 20/04/15	vie 29/05/15
13	PIL	15 días	lun 20/04/15	vie 08/05/15
14	OpenCV	15 días	lun 11/05/15	vie 29/05/15
15	Nota sobre pentagrama	23 días	vie 29/05/15	mar 30/06/15
16	Binarizado	4 días	vie 29/05/15	mié 03/06/15
17	Pentagrama	19 días	jue 04/06/15	mar 30/06/15
18	**VACACIONES**	**44 días**	mié 01/07/15	lun 31/08/15
19	**MÓDULO DE VISIÓN II**	**39 días**	**mar 01/09/15**	**vie 23/10/15**
20	Tratamiento una nota	18 días	mar 01/09/15	jue 24/09/15
21	Tratamiento varias notas	21 días	vie 25/09/15	vie 23/10/15
22	**MÓDULO DE AUDIO**	**25 días**	**lun 26/10/15**	**vie 27/11/15**
23	Grabación de notas	2 días	lun 26/10/15	mar 27/10/15
24	Edición de ficheros	8 días	mié 28/10/15	vie 06/11/15
25	Audio en NAO	15 días	lun 09/11/15	vie 27/11/15
26	**EXPERIMENTACIÓN**	**53 días**	**lun 30/11/15**	**mié 10/02/16**
27	**DOCUMENTACIÓN**	**57 días**	**lun 30/11/15**	**mar 16/02/16**

FIGURA 23: Diagrama de Gantt del proyecto

5.2 Presupuesto

En este apartado se va a realizar una estimación del presupuesto que será necesario para poder llevar a cabo el presente proyecto. Se van a diferenciar dos tipos de presupuestos o costes directos: personales y materiales. En el presupuesto general se añadirán los costes indirectos con un valor del 18% sobre el coste directo

5.2.1 Presupuesto para personal

Para realizar el cálculo del presupuesto dedicado para el personal se debe tener en cuenta que se van a dedicar una media de dos horas al trabajo del proyecto, descontando los fines de semana y festivos, que no serán incluidos en el número de hora trabajadas.

Durante el desarrollo del proyecto van a participar personal con diferentes roles:

- Jefe de proyecto: supervisor del trabajo realizado.
- Analista: encargado de la realización de las tareas de análisis presentadas en la distribución de tareas.
- Programador: encargado del diseño, implementación y evaluación del trabajo.
- Investigador: encargado de las investigaciones necesarias para cada tarea y de buscar información para las nuevas funcionalidades a desarrollar.

La labor de documentación ha sido repartida en igualdad de proporción entre el analista, el programador y el investigador.

El autor del proyecto ha asumido todos los roles mencionados excepto el de jefe de proyecto.

Para obtener el presupuesto total de personal que ha supuesto el proyecto se ha fijado el coste/hora en función del puesto de trabajo.

La tabla 5.1 muestra el presupuesto personal del proyecto.

Puesto de trabajo	Coste por hora (€)	Horas	Coste total (€)
Jefe de proyecto	35	50	1.750
Analista	30	75	2.250
Programador	25	250	6.250
Investigador	30	110	3.300
		TOTAL	**13.550 €**

Tabla5.1: Presupuesto de personal

5.2.2 Presupuesto para material

El presupuesto material del proyecto incluye todos los dispositivos empleados. Son los siguientes:

- Robot NAO H25. Coste unitario: 12.000€.
- Ordenador portátil. Coste unitario: 700€.
- Router comercial. Coste unitario: 30€.
- Atril regulable. Coste unitario: 15€
- Micrófono. Coste unitario: 200€

Hay que destacar que, para obtener el presupuesto del material, es necesario obtener la amortización de cada uno de ellos. Para ello, se ha de aplicar la siguiente fórmula:

$$\frac{\alpha}{\beta} \cdot \varepsilon$$ Donde:

- α: número de meses que se ha empleado el material en cuestión. En este caso, 12.
- β: número de meses de vida del material.
- ε: coste unitario del material.

Para cada dispositivo se han estimado los siguientes tiempos:

- Robot NAO H25. Se estima que el número de meses medio de vida del robot son unos 60. Y se ha empleado durante 6 meses.
- Ordenador portátil. El número de meses de vida útil de un ordenador suelen ser unos 36. Se ha utilizado durante todo el proyecto, 12 meses.
- Router comercial. La estimación media del tiempo de vida de un router comercial es de 36 meses. Se ha hecho uso del mismo durante 8 meses.

- Atril regulable. El tiempo de vida medio de un atril es de 96 meses. Durante el proyecto se ha empleado en 4 meses.
- Micrófono. El tiempo de vida medio de un micrófono son 18 meses. Durante el proyecto se ha empleado durante una semana. Se aproxima a 0.25 meses.

Adicionalmente se añaden la impresión de las partituras realizadas en color y en alta calidad con un coste tras amortización de 20€.

Según todos estos datos, en la tabla 5.2 está reflejado el presupuesto de cada uno de los recursos materiales empleados en el desarrollo del proyecto.

Ítem	Concepto	Coste Unitario (€)	Amortización	Coste amortización (€)
1	Robot NAO H25	12.000	0,1	1.200
2	Ordenador portátil	700	0,33	231
3	Router WiFi	30	0,22	6,6
4	Atril regulable	15	0,04	0,6
5	Partituras	20	1	20
6	Micrófono	200	0.014	2.8
			TOTAL	**1.461 €**

Tabla 5.2: Presupuesto material

5.2.3 Presupuesto total

El presupuesto total es la suma de los costes directos (presupuesto personal y material) y de los costes indirectos: facturas de luz, agua, internet, etc. En la tabla 5.3 se muestra el presupuesto total para este proyecto.

Tipo de coste	Coste (€)
Directo personal	13.550
Directo material	1.461
Indirecto (18%)	2.701,98
TOTAL	**17.712,98 €**

Tabla 5.3: Presupuesto total del proyecto

El presupuesto total necesario para el desarrollo del proyecto es de diecisiete mil setecientos doce euros con noventa y ocho céntimos.

Chapter 6: Conclusions and future works

In this chapter are described some conclusions obtained after finishing the project. First, the general conclusions of the same one are detailed. They explain the foreground and the most relevant problems found during the development. Later, the most important conclusions are described for each of the objectives established at the beginning of the project. Finally, future works are explained providing new tasks on developing related with this project.

6.1 General conclusions

This section is divided into two paragraphs. On the one hand, there will be described the new knowledge that I have needed for the development of the project. On the other hand, there will be detailed the most important problems which I have successfully overcome to achieve the objectives.

6.1.1 Foreground

Due to the curriculum of the Degree in Engineering of Audiovisual Systems, robotics is a field of research that does not offer itself direct teaching. Because of this issue, everything related to this science has been learned from scratch. Thanks to this work, I acquired formation on features and functioning of the robots, the development of applications for them, its environment of work, and the problems to obtain a correct functioning.

Particularly, to have the opportunity to work with NAO Robot, one of the most commercialized robots at present and immersed at multitude projects of research for diverse areas as education, occupational therapies, etc. it was a great opportunity to extend my formation in a sector so interesting as new. The foreground will be very useful for my future works in the robotics or other fields of study.

Along the qualifications, the projects have been developed in different programming languages. In this project a new language has been used, Python. It has been necessary to acquire notions and to look constantly for new information to learn to work on this language and to be able to develop the necessary functions in the project. This knowledge will be of great help for the future due to the fact that it is one of the most used languages for programming nowadays.

Other two issues at which I had worked during my degree but I have extended my foreground are: digital image and audio treatment. I have used tools and knowledge already used in some projects in the career and there have been used others that had knowledge of which they existed, but they had not been used in any previous project.

The use of dedicated software as Choregraphe, in robotics, or Adobe Audition, in audio, has been very satisfactory and productive.

Finally, the knowledge acquired as a result of working in an Artificial Intelligence (IA) project, has been very important for me and it will serve me for future technologies works.

6.1.2 Problems encountered

Along the development of the project a series of problems, of different difficulties have been presented. Solutions have been provided to these problems.

The first one is related to the use of Python. it has not been necessary to be a specialist in Python, but it have needed some knowledge in an ascending curve of learning. The precise tabulation of the code, the absence of the keys for opening and closing curls, and not including signs of punctuation at the end of every lines, are mandatory for programming in Python.

To get an environment of effective work also supposed some problems. At the beginning of the work it was used Ubuntu's virtual machine 14.04. When the code was being developed, the poor performance that offers the virtual machine when execution the code was too slow. I decided to make Ubuntu's partition on the hard disk of the computer to continue the work, obtaining a much better performance for programming.

Other problem has been the correct capture of the image, how to obtain the perfect image. It has had influence factors as the lighting, the NAO position and the camera resolution. The camera used is the one with the highest resolution. The quality is better but the time to process the image was longer.

6.2 Findings relating to objectives

The aims of the project were defined in the section 1.3. They are commented individually:

- NAO will be able to take a picture of the score and send it to the computer.
  I used choregraphe which allows to visualize the images to acquire and the NAO camera for reached this goal. Several factors taken some influence such as: brightness, viewing angle or the initial position of the NAO. The captured image is always different because of these factors. The image will be sent to the computer by WiFi. It depends on the size because more weight is equivalent to longest shipping time.

- Image analysis using digital techniques and algorithms.
  Using functions and algorithms in Python. This goal has been the most consumed time and effort because of the difficulty in the development and numerous tests to try it.

- NAO plays audio using edited audio files corresponding to the notes of the score.

On the one hand, the musical notes were recorded thanks to the singer, Clara Luis Mingueza using a microphone and the software Audacity. Later I edited these files using digital audios technologies with the musical software Adobe Audition. NAO has theses file in his memory.

On the other hand, from the computer the necessary orders were programmed in for NAO to play the corresponding audio files.

6.3 Future Works

There are many research lines that still open and many other techniques that could improve the performance of the system.

The ultimate goal of a hypothetical final would be NAO to be able to sing like a human. This requires that NAO learn more advanced musical techniques such as: to know some notes with different durations as whites or blacks, to express using shades musical phrasing like *piano*, *forte* or *mezzoforte*. That is, express itself as a true interpreter.

Another possible improvement would be NAO to know the treble clef and the double vertical bar indicating the music is over.

In addition, the robot might be placed by itself in front of the musical lectern independently of its initial position. NAO will look for the musical score and will stay in front of the lectern to begin the interpretation.

In technological terms, I could design a method that works in less time and more effective and make movements of NAO for its interpretation at the beginning and end of the music.

Finally, there exists the possibility of using part of the functionalities developed for another type of project of artificial vision for NAO or for another robot.

ANNEX A: NAO SINGER ROBOT

A.1 Introduction

The progress of the technology is unstoppable today. There exists a continuous development that generates the appearance of new electronic devices. We don't imagine our life without them. The motive of this is because they have been created to satisfy a need that facilitates our life.

Robotics has considerably progressed and programmers have developed prototypes with different forms and functions: humanoid robots as a person, industrial robots to work or perform household chores as floor cleaners, players in many games employed artificial intelligence or musician robots able to play the fastest melodies better than the virtuous human.

Unfortunately, in robotics, the cost and complexity of the development of a robot that includes all these features is very high. Hardware evolves faster than software, so that we can easily see hardy but barely intelligent robots

A.2 Objectives

This work includes the development of a system designed for the humanoid robot NAO H25. It is composed of various functions and algorithms to give orders to the robot NAO with the aim of being able to read and sing, by itself, a musical score. The challenge ahead is to use Artificial Intelligence to solve the problem.

The main objectives in this project are: first, NAO will be able to take a picture of the score and send it to the computer. Then, NAO will make an analysis of the capture using digital techniques and algorithms. Finally, NAO will play the corresponding notes of the score using edited audios.

A.3 System architecture

In this section, the architecture of the system is explained. The system elements can be seen in figure A.1 and they are: NAO robot H25, various musical scores, a musical lectern, a computer and a wireless router to connect computer with NAO by WiFi and a handheld microphone.

FIGURA A.1: System elements

In terms of software, it has been used:

- **Robot section:** Choregraphe 2.1.0.19, a multi-platform desktop application to create animations or behaviors, testing and have control of the NAO. On the other hand, it has been used NaoQi 1.14.5, the software that runs on the robot and controls it.
- **Audio section**: Audacity and Adobe Audition have been used to record and edit audio files.

Finally, it should be noted that all the deployed code has been developed in Python 2.7.6 in a computer with Ubuntu 14.04.

The robot used has been a NAO robot property of Carlos III University of Madrid and every work has been tested in the PLG's laboratory (Planning and Learning Group) in the Carlos III University.

Figure A.2 shows the sequence diagram to see the six interactions of the three actors in the system.

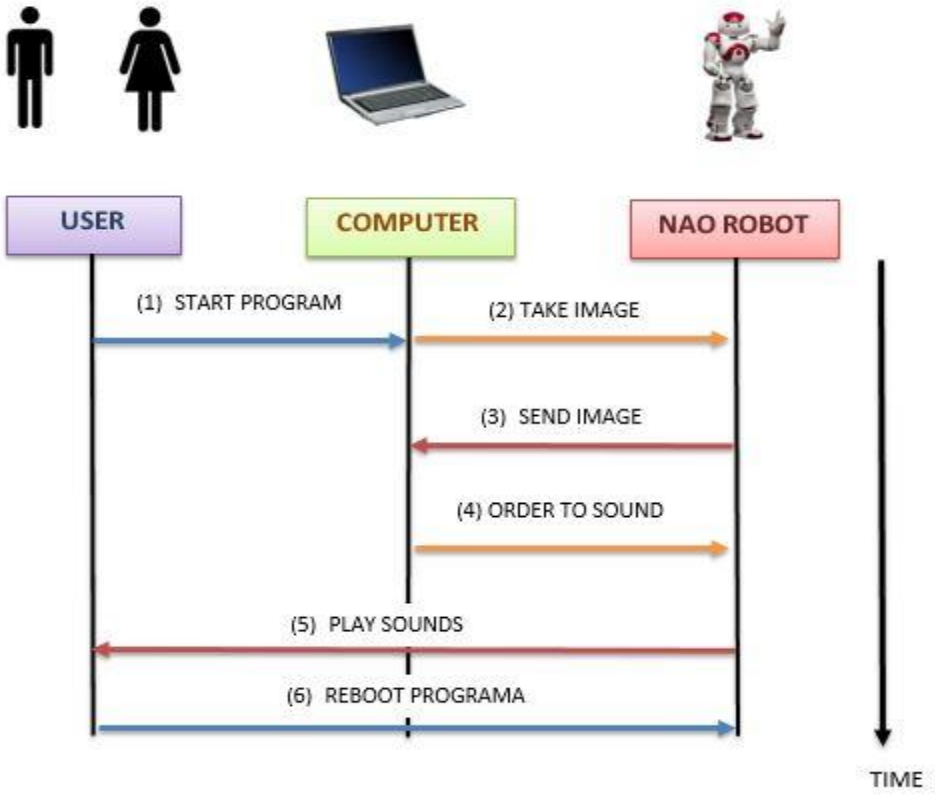


FIGURA A.224: Sequence diagram

1. **Start program:** user will start the program by launching an instruction to the computer, to execute the Python code in Ubuntu console.
2. **Take image**: computer sends an instruction to NAO robot in order to perform a capture of the image displayed, which ir the music score. This photo is taken with highest resolution camera.
3. **Send image**: NAO must send the captured image to the computer to be analyzed with digital techniques.
4. **Order to sound**: The computer already knows which are the notes found in the score. NAO has in its memory the audio corresponding to the notes. The computer sends commands to the robot to play the notes in the correct order.

5. **Play sounds**: NAO ends the program reproducing the music score with the audio files.
6. **Reboot program:** Once NAO has played the notes of the music score, the user can touch the NAO sensor located in his head to reboot the process. This is for NAO to read more scores.

The system architecture has been divided in three different modules: take image, image analysis and play audios. Each one of them has several functionalities that are essential for the correct performance of the system.

A.4 Take image

The aim of this process is to achieve that NAO is able to take an image of the music score. The main objective is that the picture may have the better resolution. NAO will use the camera located on his forehead because it is HD quality camera with 1280x960 pixels.

First, NAO should be placed correctly in front of the lectern. For this, the first instruction that the computer sends to the robot will be to get up. Previously, the musical stand will be regulated for a good vision for NAO.

The view of the camera is controlled in the computer using Choregraphe, so the musical score will be centered in the NAO range of vision.

A.5 Image analysis

It analyzes the image previously sent by NAO. This is the longest and complex process in the system. For this purpose, various techniques of digital image processing are employed.

Once computer has received the image, a binarized process for the pixels to convert them to black or white will be appled. The next step is to spot the five horizontal lines using the Hough transform. It will have many lines piled into 5 groups. It is necessary to filter using an algorithm to have only five final lines.

Then, it has five heights and after that it will calculate the six spaces. Thus, the system will have eleven heights corresponding to the eleven possible notes.

The next stop is to find the circles, or notes, in the image using the Hough transform. They should be filtered using an algorithm because lots of possibilities will appear in search results.

Finally, the system will be calculating the distances as a difference between notes and spaces or lines heights. Shorter distances will be the real notes.

A.6 Play sounds

The last stage of the system is to inform NAO of the notes that it has to sing. For this, it is necessary to use the previously edited audio files. The process is divided in two stages:

- **Recording notes.** The software used has been Audacity and with a microphone, the notes were recorded thanks to a singer.
- **Editing files**. Audio files have been edited using the software Adobe Audition. The process consists of reverb, an edger and an attenuator signal.

After that, NAO must play as a singer this edited audio corresponding with the right notes previously analyzed.

A.7 Conclusions and future works

A.7.1 Conclusions

Thanks to this work I have been acquired knowledge about robots, developing applications for them, their work environment and to know their operations. This knowledge will be useful for my future work, for others robotics works in another projects.

I have learned to program in a new language for me like is Python. Two other sections, that had worked, but I've expanded my knowledge are: digital image processing and audio. Furthermore, work in Artificial Intelligence (AI) is very gratifying.

Three goals were established at the start of this work:

- NAO will be able to take a picture of the score and send it to the computer.
  I used choregraphe and the NAO camera for reached this goal. Several factors influence like: brightness, viewing angle or the position initial of the NAO. The image captured is always different by this factors. The image will be sent to the computer by WiFi.

- Image analysis using digital techniques and algorithms
  Using functions and algorithms in Python code. This goal has been the most time and effort has needed by the difficulty in the development and numerous tests to try it.

- NAO plays audio using edited audio files corresponding to the notes of the score.
  Recording the notes with Audacity and editing the files using Choregraphe. They are in the internal memory of NAO to play the notes corresponding.

A.7.2 Future works

There are many research lines that still open and many other techniques which could improve the performance of the system.

The ultimate goal of a hypothetical final would be NAO to be able to sing like a human. This requires that NAO learn more advanced musical techniques such as: to know some notes with different durations as whites or blacks, to express using shades musical phrasing like *piano*, *forte* or *mezzoforte*. That is, express itself as a true interpreter.

Another possible improvement would be NAO to know the treble clef and the double vertical bar indicating the music is over.

In technological terms, I could design a method that works in less time and more effective and make movements of NAO for its interpretation at the beginning and end of the music.

Finally, there exists the possibility of using part of the functionalities developed for another type of project of artificial vision for NAO or for another robot.

Anexo B: Manual de instalación

En este anexo se van a dar las directrices para instalar todas las herramientas necesarias en el proyecto. Destacar que se ha de tener instalada la versión 14.04 de Ubuntu.

B.1 Python

La versión de Python que se debe instalar es Python 2.7.1. Una versión superior no es válida debido a que habría una incompatibilidad del código por estar desarrollado en una versión inferior.

Si se dispone de una versión de Python, pero no se conoce cuál es, desde la terminal de Ubuntu se escribirá el siguiente comando.

```
python – version
```

Para instalar los paquetes necesarios de la versión 2.7.1, se deberá introducir el siguiente comando:

```
sudo apt-get install python2.6
```

B.2 Choregraphe

El software Choregraphe puede descargarse desde la página web de Aldebaran Robotics. En concreto desde este enlace:

https://community.aldebaran.com/en/resources/software/language/en-gb

Para poder descargar el ejecutable es necesario darse de alta en el sistema mediante creándose una cuenta de usuario.

El uso del software es gratuito durante 90 días. A partir de estos días se requiere la obtención de la licencia.

Para instalarlo, es suficiente con descomprimir el archivo y guardarlo en un directorio. Para ejecutarlo basta con introducir el siguiente comando por la terminal:

```
./choregraphe
```

Si se obtiene algún error relacionado con la variable de entorno PYTHONPATH a la hora de ejecutarlo, habrá que modificar la ruta asociada a la misma por la de Python que se encuentra dentro de la carpeta Choregraphe. Para observar las variables de entorno existentes se debe llegar al directorio donde se encuentra el archivo *environment*. Para ello se debe introducir el siguiente comando en la terminal:

```
sudo gedit /etc/environment
```

Las variables de entorno deben quedar así:

```
export AL_DIR=~/Programas/choregraphe-suite-2.1.0.19-linux64
export LD_LIBRARY_PATH=$AL_DIR/lib:$LD_LIBRARY_PATH
export PYTHONPATH=$AL_DIR/lib:$PYTHONPATH
export PATH=$PYTHONPATH:$AL_DIR/bin:$PATH
```

B.3 NaoQi

La instalación de NaoQi se realiza de la misma forma que Choregraphe. Desde la web de Aldebaran Robotics se puede descargar el ejecutable. En concreto en esta dirección:

https://community.aldebaran.com/en/resources/software/language/en-gb

Para instalarlo, es suficiente con descomprimir el archivo y guardarlo en un directorio.

B.4 OpenCV

El primer paso es descargarse la versión 3.1.0 de OpenCv desde el siguiente enlace:

https://sourceforge.net/projects/opencvlibrary/files/opencv-unix/

A continuación, se descomprime el archivo descargado y se comprueba si se tienen instaladas en el sistema todas las dependencias. En una terminal se escriben las siguientes instrucciones:

```
sudo apt-get install build-essential libgtk2.0-dev libjpeg-dev libtiff4-dev
libjasper-dev libopenexr-dev cmake python-dev python-numpy python-tk
libtbb-dev libeigen2-dev yasm libfaac-dev libopencore-amrnb-dev
libopencore-amrwb-dev libtheora-dev libvorbis-dev libxvidcore-dev libx264-dev
libqt4-dev libqt4-opengl-dev sphinx-common texlive-latex-extra libv4l-dev
libdc1394-22-dev libavcodec-dev libavformat-dev libswscale-dev
```

Seguidamente desde la terminal hay que dirigirse a la ruta donde se ha descomprimido el archivo que contiene la librería completa de OpenCV. Además, emcomtrándose en el directorio de instalación, se deben escribir las siguientes instrucciones por la terminal para compilar la librería:

```
mkdir build
cd build
cmake -D WITH_TBB=ON -D BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON -D
INSTALL_C_EXAMPLES=ON -D INSTALL_PYTHON_EXAMPLES=ON -D BUILD_EXAMPLES=ON
-D WITH_QT=ON -D WITH_OPENGL=ON ..
```

Para finalizar, se ejecutan las instrucciones necesarias para instalar la librería:

```
make
sudo make install
```

B.5 PIL

Para instalar la librería PIL en Ubuntu será necesario instalar diversos paquetes mediante los siguientes comandos en la consola:

```
sudo apt-get install libjpeg-dev
sudo apt-get install libjpeg-dev
sudo apt-get install libtiff-dev
sudo apt-get install libfreetype6-dev
sudo apt-get install cmake
cmake .
sudo make install
pip install pil
```

Anexo C: Manual de usuario

Tras tener todas las aplicaciones instaladas y el entorno de trabajo correcto, el usuario se limitará a ejecutar dos acciones: conectar todos los dispositivos en la red e iniciar el programa.

C.1 Conexión de todo el sistema

Como ya se explicó, la conexión se realizará en red LAN, gracias al router, y cada uno tendrá una IP asignada:

- Router: 192.168.1.1
- Robot NAO: 192.168.1.179
- Ordenador: 192.168.1.85

El robot se conectará por sí mismo a la red. El usuario debe comprobar la conexión desde el ordenador al router y a NAO, de forma independiente.

Primero accederemos al router desde la terminal ejecutando el siguiente comando:

```
ping 192.168.1.1
```

Se observa que se tiene conectividad porque existe un envío y recepción de paquetes.

En segundo lugar, se comprueba que la conexión con NAO es correcta mediante el siguiente comando:

```
ping 192.168.1.179
```

Ya estarían todos los elementos conectados y se procede al siguiente punto.

Es posible que ocurran errores de conectividad entre los tres sistemas. En numerosas ocasiones realizando un reinicio del router, la red vuelve a estar activa. Si lo que falla es la conectividad con NAO, se debe esperar un tiempo prudencial para que se produzca la conexión. Si este tiempo excede en más de un minuto, se debe proceder a reiniciar el robot.

C.2 Iniciar el programa

En este último paso, el usuario debe dar comienzo al programa desde la terminal. Tiene que situarse sobre el directorio donde se encuentre el archivo Python con el programa. Solo ejecutando la siguiente línea el programa se iniciará:

```
python nombre_del_archivo.py
```

Bibliografía

[1] Robótica Ingeniería informática y de sistemas. URL: https://robotica.wordpress.com/about/

[2] British Automation & Robot Association (BARA). URL: http://www.bara.org.uk/

[3] "Robot"s – A historical perspective. URL: http://bowlesphysics.com/images/Robotics_-_A_historical_perspective.pdf

[4] Maja Matarić. The University of Southern California. URL: http://www-robotics.usc.edu/~maja/

[5] Michael Brady and Richard Paul, editors. Robotics Research: The First International Symposium. The MIT Press, Cambridge MA, 1984 URL: http://platea.pntic.mec.es/vgonzale/cyr_0708/archivos/_15/Tema_5.1.htm

[6] Robótica industrial. URL: http://platea.pntic.mec.es/vgonzale/cyr_0204/ctrl_rob/robotica/industrial.htm

[7] Runaround. Isaac Asimov, 1942. URL: http://www.singularitysymposium.com/laws-of-robotics.html

[8] Generaciones de robots. URL: http://roboticalp.blogspot.com.es/p/generaciones-de-robots.html

[9] Google Sites. Historia de la Robótica: ¿sueñan los androides con ovejas eléctricas? URL https://sites.google.com/site/historiadelosandroides/tipos-de-robots

[10] Aldebaran Robotics. NAO Robot. URL: https://www.aldebaran.com/en/cool-robots/nao

[11] Aldebaran Robotics. URL: https://www.aldebaran.com/en

[12] Aldebaran Robotics. NaoQi 1.14. URL: http://doc.aldebaran.com/1-14/index.html

[13] Cucoclock. Jacques Vaucanson (1709-1782). URL: http://www.cucoclock.com/P%C3%A1gina%2020.html

[14] Penn Engineering. Vijay Kumar. URL: http://www.kumarrobotics.org/

[15] Xatakaciencia. HRP-4C Miim, el robot con apariencia humana. URL: http://www.xatakaciencia.com/robotica/hrp-4c-miim-el-robot-con-apariencia-humana

[16] Robocross. StickBoy. URL: http://www.robocross.de/page7.html

[17] The Trons Robot band. URL: http://www.thetrons.co.nz/

[18] Compressorhead rock band. URL: https://compressorhead.rocks/

[19] DailyMail. Z-Machines heavy metal and rock band. URL: http://www.dailymail.co.uk/sciencetech/article-2486775/Robot-rock-band-Z-Machines-features-guitarist-78-fingers-drummer-21-sticks.html

[20] AVID. Sibelius. URL http://www.avid.com/ES/products/sibelius#overview

[21] EcuRed. Adobe Audition. URL http://www.ecured.cu/Adobe_Audition

[22] Adobe System. URL: http://adobe-systems-incorporated.software.informer.com/

[23] Manual Finale. Finale. URL http://manualfinale.blogspot.com.es/2010/04/introduccion.html

[24] Makemusic. URL: http://www.makemusic.com/

[25] Aulavirtual. Audacity. URL http://aulaintercultural.org/2006/03/30/audacity-programa-para-editar-sonido-digital-software-libre/

[26] Steingberg. VST. URL: https://www.steinberg.net/en/products/vst.html

[27] Software Musescore. URL http://musescoretutoriales.blogspot.com.es/

[28] MIDI. URL: http://www.css-audiovisual.com/areas/guias/midi.htm

[29] Formato MUSICXML. URL: http://www.musicxml.com/

[30] Opencv, 2015. URL http://opencv.org/.

[31] Intel, website official. URL: http://www.intel.es/content/www/es/es/homepage.html

[32] PIL 1.1.7 URL http://www.pythonware.com/products/pil/

[33] Luis Mingueza, Nerea. Desarrollo de un sistema de juego al Tres en Raya para el robot NAO H25, 2013. URL: http://e-archivo.uc3m.es/handle/10016/19841

[34] Miguel Andrés, Igor de. Revisión, modificación y validación experimental del sistema sensorial del robot HOAP 3 (masterThesis). URL: http://e-archivo.uc3m.es/handle/10016/13572

[35] Elizaga Navascués, Ignacio. Localización e interacción con objetos mediante visión artificial con el robot NAO. URL: http://e-archivo.uc3m.es/handle/10016/16780

[36] Website official Ubuntu. URL: http://www.ubuntu.com/

[37] Aldebaran Robotics. Documentation Choregraphe 2.1.0.19 URL: http://doc.aldebaran.com/1-14/software/choregraphe/index.html

[38] The official home of the Python Programming Language. URL: https://www.python.org/

[39] Barry Boehm. A spiral model of software development and enhancement. ACM SIGSOFT Software Engineering Notes, 1986.