TecnoCampus

Centres universitaris adscrits a la

Universitat Pompeu Fabra Barcelona

Doble Grado en Informática de Gestión y Sistemas de Información /

Grado en Diseño y Producción de Videojuegos

Watermarking en firma manuscrita online

Sergio Sánchez González

Tutor: Marcos Faúndez-Zanuy

Abstract

In this project, the process of watermarking handwritten signatures has been investigated. Using a database with numerous signers, the necessary force to optimally apply the watermark has been analyzed and tested. Additionally, code has been developed for another researcher in their work on signature writing using a robotic arm.

Resum

En aquest projecte, s'ha investigat el procés de watermarking en signatures manuscrites. Utilitzant una base de dades amb nombrosos signants, s'ha analitzat i provat la força necessària per aplicar la marca d'aigua de manera òptima. A més, s'ha desenvolupat codi per a un altre investigador en el seu treball sobre l'escriptura de signatures utilitzant un braç robòtic.

Resumen

En este proyecto, se ha investigado el proceso de watermarking en firmas manuscritas. Utilizando una base de datos con numerosos firmantes, se ha analizado y probado la fuerza necesaria para aplicar dicha marca de agua de manera óptima.
Adicionalmente, se ha desarrollado código para otro investigador en su trabajo sobre la escritura de firmas utilizando un brazo robótico.

Índice

Índice de figuras

Listado de acrónimos

BMP - Mapa de píxeles

JPEG - Joint Photographic Experts Group

WAV - Formato de archivo de audio Waveform

MP3 - Capa de audio MPEG-1/2, nivel 3

LSB - Bit menos significativo

MSB - Bit más significativo

ISB - Bit significativo intermedio

DCT - Transformada de coseno discreta

DFT - Transformada discreta de Fourier

DTFT - Transformada de Fourier de tiempo discreto

DWT - Transformada Wavelet Discreta

SVD - Descomposición en valores singulares

IDE - Entorno de desarrollo integrado

MCYT - Ministerio de Ciencia y Tecnología

DTW - Dynamic Time Warping

DCF - Detection Cost Function

MAE - Error absoluto medio (Mean Absolute Error)

MSE - Error cuadrático medio (Mean Squared Error)

1. Objeto del proyecto

En la era digital, la autenticación y protección de documentos electrónicos, como las firmas digitales, se ha vuelto de suma importancia. En este contexto, el uso de marcas de agua emerge como una técnica prometedora para garantizar la autenticidad y la integridad de dichos documentos.

Este trabajo de final de grado investiga el proceso de añadir marcas de agua a las firmas digitales con el objetivo de poder lograr un reconocimiento robusto de las mismas, manteniendo al mismo tiempo la capacidad de extraer información específica de la marca de agua incrustada.

Se intenta encontrar el equilibrio entre la capacidad de extraer los metadatos incrustados mediante la marca de agua y la posibilidad de realizar un reconocimiento biométrico satisfactorio.

2. Estudio previo: Contexto, antecedentes y necesidades de información

2.1. Firma

Según el diccionario de Cambridge, una firma es “tu nombre escrito por ti mismo, siempre de la misma forma, probablemente para demostrar que algo ha sido escrito por ti o se está de acuerdo con algo” [1].

Algunos de los ejemplos más antiguos conocidos de la firma utilizada para validar una identidad datan de hasta el 3000 a.C. Culturas como los sumerios y egipcios comenzaron a utilizar una serie de imágenes y símbolos, los pictogramas, para transmitir la identidad del creador. Una tablilla de arcilla sumeria, que data de alrededor del 3100 a.C., tiene las marcas de su creador, llamado Kushim, lo que la convierte en uno de los primeros ejemplos definitivos de uso de palabras y símbolos para denotar identidad [2].

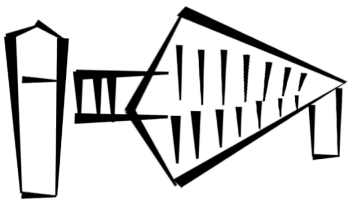


Fig. 2.1.1. Pictograma de Kushim. [3]

La firma ha evolucionado mucho desde entonces, en la antigua Roma los emperadores utilizaban sellos de cera para certificar documentos. El primer documento del cual se tiene constancia que fue firmado en el alfabeto Latín está en España, “el Cid” hizo una donación a la catedral de Valencia, y firmó al final del documento con su nombre.

La Ley del Estatuto de Fraudes aprobada en el Parlamento inglés en 1677 convirtió la firma en el marcador cotidiano que es hoy en día. La nueva ley establecía que los contratos, testamentos y concesiones debían ser firmados, una medida que en su tiempo era una garantía efectiva contra el fraude [4].

2.1.1 Firma digital

Una firma digital es un algoritmo matemático utilizado para verificar la autenticidad de un documento o mensaje electrónico. Proporciona una manera de asegurar que el documento o mensaje no ha sido manipulado y que de hecho fue enviado por la persona o entidad que afirma haberlo enviado.

Las firmas digitales funcionan utilizando un par de claves criptográficas, una pública y una privada. La clave privada es conocida únicamente por el propietario y se utiliza para crear la firma digital. La clave pública se comparte con cualquier persona que necesite verificar la

firma. Cuando se crea una firma digital, se adjunta al documento o mensaje y se puede utilizar para verificar su autenticidad.
Las firmas digitales tienen sus ventajas y desventajas, tales como indica Shiv Kumar [5].

Pros de las firmas digitales:

- Seguridad: Las firmas digitales proporcionan un alto nivel de seguridad para documentos y mensajes electrónicos. Son extremadamente difíciles de falsificar o manipular, lo que las convierte en una forma ideal de verificar la autenticidad de documentos importantes y transacciones.
- Ahorro de tiempo: Las firmas digitales pueden ahorrar una cantidad significativa de tiempo y dinero al eliminar la necesidad de imprimir, firmar y enviar documentos en papel. Esto puede ser particularmente útil en situaciones donde los documentos necesitan ser firmados rápidamente o cuando las partes están ubicadas en diferentes partes del mundo.
- Legalmente vinculantes: Las firmas digitales son legalmente vinculantes y tienen el mismo nivel de autenticidad que las firmas manuales. Esto significa que pueden ser utilizadas en la corte como evidencia de un acuerdo o contrato.
- Beneficios ambientales: La emisión de firmas digitales es un proceso en línea. Al reducir la necesidad de documentos en papel, las firmas digitales pueden ayudar a reducir el impacto ambiental de las transacciones comerciales. Esto puede ser especialmente importante para empresas comprometidas con la sostenibilidad y la reducción de su huella de carbono.

Contras de las firmas digitales:

- Dependencia de la tecnología: Las firmas digitales dependen de la tecnología, que puede ser vulnerable a hackeos y otras formas de ciberdelincuencia. Esto significa que las empresas que utilizan firmas digitales deben asegurarse de que sus sistemas sean seguros y estén actualizados con los últimos parches de seguridad y actualizaciones.
- Complejidad: Las firmas digitales pueden ser complejas de configurar y usar, especialmente para personas que no están familiarizadas con la tecnología. Esto puede provocar errores y confusiones, lo que puede socavar la efectividad del sistema. En el caso de los ciudadanos mayores, a veces la emisión de una firma digital puede ser una tarea difícil.
- Aceptación limitada: Por ejemplo, India es un país en desarrollo, donde la tecnología no está disponible en todos los lugares, por lo que lleva tiempo reemplazar las firmas manuales con firmas digitales.

2.1.2. Firma manuscrita

La biometría de escritura a mano, también conocida como "handwriting biometrics", es un campo de estudio que se centra en el análisis y la autenticación de la escritura a mano de un individuo. Se basa en el hecho de que cada persona tiene un estilo de escritura único y distintivo, que puede ser utilizado como una forma de identificación biométrica.

Este enfoque se ha vuelto cada vez más importante en áreas como la seguridad de la información, la autenticación de documentos y la verificación de identidad. Utiliza técnicas computacionales avanzadas para analizar diversos aspectos de la escritura a mano, como la forma de las letras, la presión ejercida sobre el papel, la velocidad de escritura y otros atributos.

La biometría de escritura a mano está estrechamente relacionada con la firma manuscrita, ya que ambas se centran en características únicas y distintivas del patrón de escritura de un individuo. La firma manuscrita es un tipo específico de escritura a mano que se utiliza para autenticar la identidad de una persona en documentos legales, financieros y otros. La biometría de escritura a mano aprovecha esta singularidad para identificar y autenticar a un individuo mediante el análisis de su estilo de escritura.

Un sistema biométrico de escritura a mano debe aprovechar la gran cantidad de información que se puede extraer de una tarea de escritura simple. Esto se debe a que la escritura a mano es una tarea cognitiva en la que se emiten órdenes neuromotoras sincronizadas desde la corteza cerebral para llevar a cabo la acción planeada. Las técnicas de procesamiento de señales y reconocimiento de patrones aplicadas para analizar este procedimiento pueden revelar información muy útil sobre el escritor, incluida su identidad y estado de salud [6].

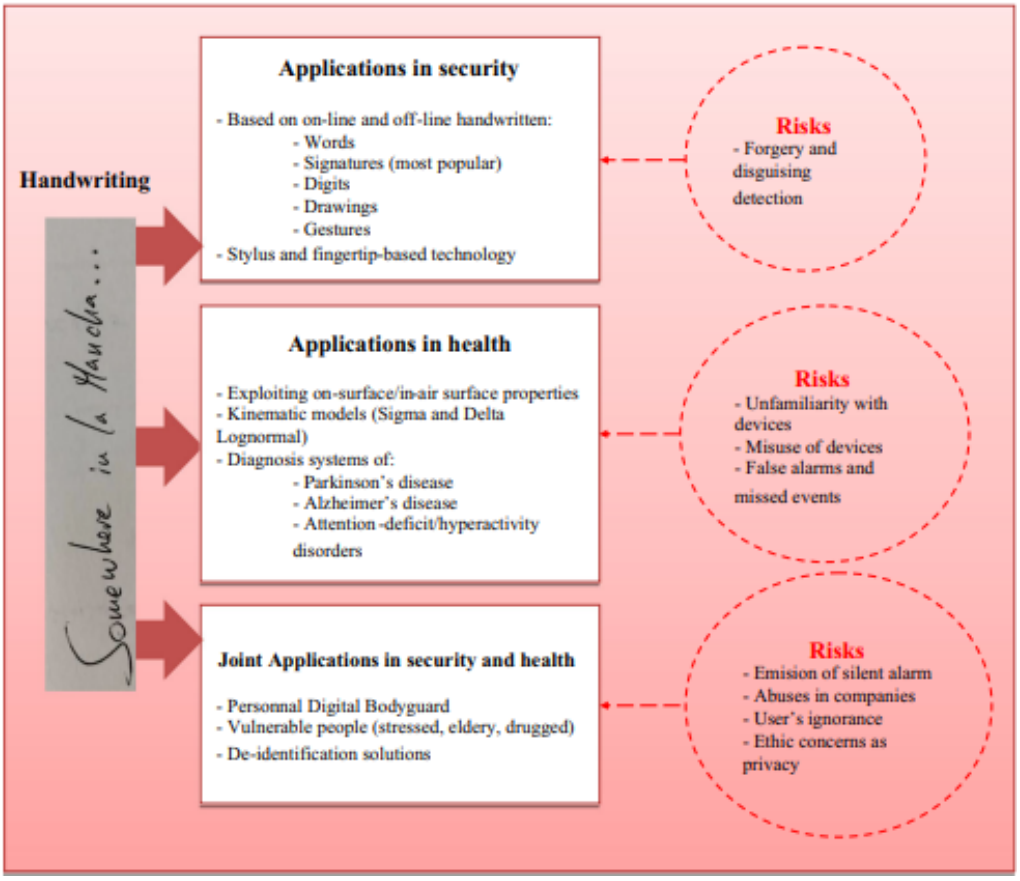


Fig. 2.1.2.1. Aplicaciones del sistema biométrico de escritura a mano [6]

En el caso de la seguridad se han hecho estudios en los cuales varias personas escribían distintas palabras y luego se podía ver en qué porcentaje se identificaba correctamente la palabra escrita, y este resultado suele ser mayor al 90% [7-9], se ha intentado también ver en qué caso se puede identificar cuando alguien está intentando imitar las escritura de otra persona [9], aunque aún hay mucho trabajo en ese aspecto.

En el campo médico, se ha demostrado que mediante el análisis de la escritura aumenta la precisión de la identificación de la disgrafía parkinsoniana, el Alzheimer y el Autismo, así como detectar situaciones en las que el usuario está bajo alto estrés o bajo el consumo de drogas [10].

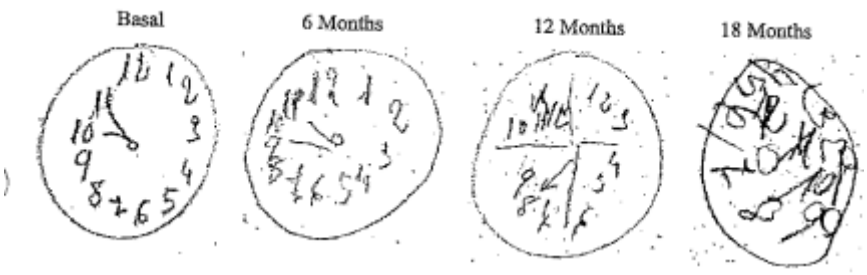


Fig. 2.1.2.2. Diferencia en el dibujo de un reloj hecho por un paciente de Alzheimer, al descubrir su enfermedad y tras 6, 12 y 18 meses [11].

2.2. Watermarking

La necesidad de protección de los derechos de la propiedad intelectual no es fruto de la evolución tecnológica, sino que ya existe desde la antigüedad. La inclusión de firmas manuscritas en documentos, obras pictóricas, etc. responde a este propósito [12].

Hartung y Kutter [13] localizan los orígenes de las marcas de agua en el arte de la fabricación de papel hecho a mano hace casi 700 años. El papel con marca de agua más antiguo encontrado en archivos data de 1292 y tiene su origen en Fabriano, Italia, que es considerado el lugar de nacimiento de las marcas de agua.
A finales del siglo XIII, alrededor de 40 molinos de papel compartían el papel marcado en Fabriano y producían papel con diferentes formatos, calidades y precios. Producían papel crudo y áspero que era alisado y postprocesado por artesanos, y vendido por comerciantes. La competencia no solo era entre los molinos de papel, sino también entre los artesanos y comerciantes, y era difícil llevar un registro del origen del papel y, por lo tanto, de la identificación del formato y la calidad. La introducción de marcas de agua ayudó a evitar cualquier posibilidad de confusión.

Hartung y Kutter [13] denotan la diferencia entre Esteganografía y Watermarking:
La esteganografía se refiere a técnicas en general que permiten la comunicación secreta, generalmente mediante la inserción u ocultamiento de la información secreta en otros datos no sospechados. Los métodos esteganográficos generalmente se basan en la suposición de que la existencia de la comunicación encubierta es desconocida para terceros y se utilizan principalmente en comunicaciones secretas punto a punto entre partes de confianza. Como resultado, los métodos esteganográficos en general no son robustos, es decir, la información oculta no se puede recuperar después de la manipulación de datos.

Por otro lado, el marcado de agua, a diferencia de la esteganografía, tiene la noción adicional de robustez contra ataques. Incluso si se conoce la existencia de la información oculta, es difícil, idealmente imposible, para un atacante destruir la marca de agua incrustada, incluso si el principio algorítmico del método de marcado de agua es público.

Dependiendo del objetivo buscado, la cantidad de datos a ocultar será grande o pequeña. Cuanto menor sea la cantidad de información a ocultar, más fácil será conseguir una buena protección frente a manipulaciones de la información original.

Faundez [12] destaca que una marca de agua debe poseer las siguientes características:

- Invisible a nivel perceptivo. Aunque resulta inevitable que la marca de agua altere el mensaje, la calidad de la imagen o señal sonora marcada, no deben presentar mermas de calidad. Lógicamente en el caso de textos debe mantenerse la coherencia de significado, corrección ortográfica y gramatical, etc. Existirá un compromiso entre la degradación introducida y la robustez de la marca a alteraciones. Para conseguir que una marca sea invisible, hay que tener en cuenta las características del sistema visual y auditivo humano.
- Invisible a nivel estadístico. Si todos los productos marcados con nuestra clave presentan una misma característica común, resultará sencillo detectar la protección si se dispone de un número considerable de productos marcados. Para evitarlo, puede usarse una marca de agua que dependa del contenido de la información.

- Complejidad. De forma análoga a los billetes, deberá tener una cierta complejidad para evitar que alguien pueda falsificar nuestra marca.
- Clave de acceso. En ocasiones interesa que la posibilidad de incrustar y/o detectar la marca de agua vaya vinculada a un número de código.
- Robustez. Es necesario que la información marcada sea robusta a manipulaciones. Por ejemplo, si se marca una imagen BMP que la marca permanezca al convertirla a JPEG, o en un fichero wav al pasar a MP3. Por otra parte, la información debe estar contenida en los datos, no en las informaciones de cabecera de formato de fichero. De esta forma, resulta más difícil su eliminación.
- Coste computacional de la inclusión y extracción de la marca. Interesa que el tiempo necesario para incluir la marca, y especialmente para comprobar su presencia, sea razonable.

2.2.1. Usos de Watermarking

Begum [14] habla por diversos usos del watermarking:

- Monitoreo de Transmisiones. Esta aplicación permite a un propietario de contenido verificar cuándo y dónde se transmitió el contenido. También verifica la hora exacta de transmisión del contenido a través de la televisión por satélite y los medios de transmisión. Antes de la transmisión, se puede insertar una marca de agua única en cada clip de sonido o video. Es útil para varias organizaciones e individuos cuando los anunciantes quieren asegurarse de que el contenido se transmita en el horario exacto acordado por el cliente y la empresa publicitaria. Esta aplicación se puede utilizar para garantizar la transmisión legal de productos de televisión.

- Protección de Derechos de Autor, Declaración de Propiedad o Identificación del Propietario. En aplicaciones de protección de derechos de autor, una marca de agua visible identifica al propietario de los derechos de autor. La marca de agua requiere una robustez sólida, de manera que la imagen con marca de agua no se pueda eliminar sin distorsión de datos.

(a) Host image (512 × 512 pixels) *(b) Watermarked image PSNR = 42.2096* *(c) Watermark (128 × 128 pixels)*

Fig. 2.2.1.1. Ejemplo de uso de una marca de agua para proteger derechos de autor [15]

- Control de copias: El control de copia evita que las personas hagan copias ilícitas de contenido. En este sentido, una marca de agua puede ser utilizada para restringir la copia informando a los dispositivos de hardware o software. En la protección contra la copia, un pirata conoce el estado de los mensajes ocultos, que es la verdadera amenaza. El mensaje contiene "No Copiar Más", "Copiar Una Vez" o "Nunca Copiar". Por otro lado, en los esquemas de huellas dactilares o seguimiento de transacciones, un usuario inocente no puede ser incriminado por la colusión de piratas, y al menos un pirata puede ser rastreado por el detector. Similar a las huellas dactilares, que identifican a un individuo, el seguimiento de transacciones identifica de manera única cada copia de la obra.

- Autenticación de Contenido y Verificación de Integridad: Las imágenes digitales pueden ser modificadas con la ayuda de herramientas de procesamiento de imágenes sofisticadas ampliamente disponibles. Para una comunicación segura, la información debe estar protegida contra el acceso no autorizado; esta propiedad se conoce como integridad. Una marca de agua verifica la autenticidad de una imagen. Cualquier modificación significativa de la imagen también puede cambiar la marca de agua. Estos cambios pueden ser detectados, lo que indica que los datos han sido manipulados.

- Indexado: Esta técnica utiliza comentarios y marcadores o información clave, la cual se incrusta como una marca de agua en videos, películas y noticias. Luego, la técnica recupera los datos requeridos utilizados por el motor de búsqueda.

- Aplicaciones médicas: El marcado de agua en imágenes se puede aplicar para proteger los derechos de autor de las imágenes médicas. La información del paciente puede ser protegida contra el acceso ilegal mediante técnicas de marca de agua. La telemedicina conecta especialistas y pacientes separados por una distancia física. Por lo tanto, para garantizar la confidencialidad, autenticidad, integridad y disponibilidad asociadas con el intercambio de datos del expediente electrónico del paciente, se pueden utilizar técnicas de marcado de agua adecuadas. En estas aplicaciones, la calidad de la imagen no debe verse afectada por los datos de la marca de agua.

2.2.2. Técnicas de Watermarking

Begum y Shorif [13], y Faundez [12] hablan sobre distintas técnicas con las que aplicar estas marcas de agua a imágenes:

- Modificación del bit menos significativo (LSB): el bit menos significativo (LSB) de píxeles seleccionados al azar puede ser alterado para ocultar el bit más significativo (MSB) de otro. Se genera una señal aleatoria utilizando una clave específica. La marca de agua se inserta en los bits menos significativos de la imagen hospedada y puede ser extraída de la misma manera. Algunos programas comerciales modifican previamente la paleta de color, para hacer menos visibles las alteraciones introducidas.

- Modificación del bit intermedio significativo (ISB): Las técnicas de LSB son las más comunes y simples técnicas avanzadas de marcado de agua en el dominio espacial, pero no garantizan la robustez contra ataques. Por esta razón, se han desarrollado métodos alternativos, como los métodos de bits significativos intermedios (ISB), para mejorar la robustez y preservar la calidad del sistema de marcado de agua. Varios estudios han desarrollado métodos de ISB utilizando diferentes algoritmos. Uno de estos métodos reemplaza la técnica clásica de LSB con ISB encontrando el mejor valor de píxel entre un rango de píxeles. En este método, la imagen de marca de agua está protegida contra varios ataques y se minimiza la alteración de la imagen con marca de agua.

- Patchwork: es un proceso estadístico pseudoaleatorio que se incrusta en una imagen original de forma invisible utilizando codificación de patrones redundantes mediante una distribución gaussiana. Se eligen pseudoaleatoriamente dos parches, A y B, y los datos de imagen del primer parche (A) se desvanecen, mientras que los de B se oscurecen. Los métodos de Patchwork muestran una mejor robustez contra modificaciones máximas de imagen no geométricas y el proceso es independiente del contenido de la imagen original.

- La transformada discreta del coseno (DCT): separa una imagen en sus coeficientes de frecuencia equivalentes modificando los componentes de frecuencia, los cuales pueden expresarse como una suma de funciones coseno. La DCT es una transformada relacionada con la transformada de Fourier y contiene una secuencia finita de puntos de datos. Aquí solo se pueden utilizar números reales. Su varianza determina la utilidad de los coeficientes de la DCT [16].

- La transformada discreta de Fourier (DFT): utiliza muestras que están uniformemente espaciadas. En este caso, una secuencia de números de longitud fija de muestras uniformemente espaciadas de una función se convierte en una secuencia de la misma longitud de muestras uniformemente espaciadas en la Transformada de Fourier en Tiempo Discreto (DTFT).
  La DTFT utiliza un conjunto de funciones exponenciales complejas relacionadas armónicamente (magnitud y fase). La DFT representa la secuencia de entrada original en el dominio de la frecuencia y produce una señal que es discreta y periódica. Muchas aplicaciones prácticas, incluyendo el procesamiento de señales,

procesamiento de imágenes, filtros, operaciones de convolución, análisis de espectro de sinusoides y análisis de Fourier, se realizan mediante DFT [17].

- La transformada discreta de wavelet (DWT): es cualquier transformada de wavelet que descompone una señal en wavelets, en lugar de frecuencias. En una DWT, los wavelets están muestreados de forma discreta. La resolución temporal es una de las ventajas de la DWT sobre las transformadas de Fourier (es decir, DCT y DFT). Esto hace que la DWT sea un área de investigación más atractiva, capturando múltiples aspectos de la información, como la ubicación en el tiempo y la frecuencia. Se utiliza un conjunto de wavelets, que son funciones matemáticas, para descomponer la señal. La transformada de wavelet es útil en el procesamiento digital de señales, compresión de imágenes y eliminación de ruido de la señal. La idea clave en una transformada de wavelet es el uso de un conjunto de funciones de base (llamadas wavelets) que ofrecen localización en el dominio de la frecuencia. Se puede obtener resolución de alta frecuencia en bajas frecuencias, y resolución de alta temporal en altas frecuencias cuando se utiliza una transformada de wavelet [18].

- Descomposición en valores singulares (SVD): es el producto de una matriz real o compleja. Este método es la generalización de la descomposición en valores propios de una matriz simétrica con valores propios no negativos a cualquier matriz m × n a través de una extensión de la descomposición polar. La transformación SVD ha sido ampliamente utilizada en estadísticas y procesamiento digital de señales [19].

- Aproximación estadística: Se trata de modificar una determinada estadística dentro de la imagen que alberga la información. Un ejemplo sencillo sería: se aumenta y disminuye el brillo de unos determinados píxeles de la imagen (a,b), de forma que $a'=a+\Delta$ y $b'=b-\Delta$. La selección de los píxeles se realiza mediante un generador de números pseudoaleatorios. De esta forma se altera la estadística del valor de la diferencia entre dos píxeles de la imagen tomados aleatoriamente. Este ejemplo únicamente permite introducir un bit de información, pero presenta la ventaja de que no es posible eliminar la marca (sin afectar a la calidad de la imagen) si no se conoce la clave utilizada para general la secuencia de números pseudoaleatorios.

- Codificación de bloques de texturas: Consiste en seleccionar y copiar una porción de imagen de una determinada textura (hierba, asfalto, etc.) en otro área de la imagen de características similares. De esta forma se obtienen dos zonas de imagen con idénticas texturas. Para detectar estas regiones en una imagen marcada bastará con calcular la autocorrelación de la imagen para detectar la posición, y restar la imagen con ella misma pero desplazada a la posición indicada por la autocorrelación. Tras este proceso se aprecian zonas de imagen en las que la diferencia vale cero, que serán las correspondientes a las zonas copiadas. La forma geométrica descrita por el perfil de la zona copiada puede ser el dibujo de la marca de agua (el nombre de la empresa, figura geométrica, etc.). Si la totalidad de la imagen sufre una transformación uniforme, ambas regiones se verán afectadas de idéntica manera y seguirá siendo posible detectar la presencia de esas dos partes iguales. Sin embargo, este método requiere una inspección visual para detectar las posibles zonas a copiar, y el impacto visual que produce el proceso.

2.3. Matlab

Matlab es un entorno de programación y un lenguaje utilizado predominantemente en el ámbito científico y de ingeniería. Su nombre, una abreviatura de "Matrix Laboratory", refleja su enfoque en el manejo eficiente de operaciones relacionadas con matrices. Su popularidad radica en su capacidad para abordar una amplia gama de problemas numéricos y analíticos de manera eficiente y efectiva.

Una de las características distintivas de Matlab es su enfoque en el manejo de matrices. Esto significa que muchas de las operaciones en Matlab se pueden realizar de manera vectorizada, lo que resulta en un código más limpio y eficiente en comparación con otros lenguajes de programación. Además, la capacidad de realizar operaciones matriciales de manera rápida y sencilla hace que Matlab sea especialmente adecuado para el procesamiento de señales, el análisis de datos y la simulación de sistemas dinámicos.

Otra ventaja importante de Matlab es su entorno de desarrollo integrado (IDE), que proporciona una interfaz interactiva para escribir, depurar y ejecutar código. Este entorno facilita la experimentación y el desarrollo iterativo, lo que es fundamental en el proceso de investigación y desarrollo. Además, Matlab ofrece una amplia variedad de herramientas de visualización, lo que permite a los usuarios representar gráficamente los datos y los resultados de manera clara y comprensible.

Además de su amplio conjunto de funciones incorporadas, Matlab también cuenta con una activa comunidad de usuarios que contribuyen con bibliotecas y herramientas adicionales a través del File Exchange de Matlab. Esto significa que los usuarios tienen acceso a una amplia gama de recursos y soluciones para una variedad de problemas.

3. Objetivo y abasto

- Investigar sobre el uso de marcas de agua y como puede ser aplicado a las firmas manuscritas.

- Elaborar un código de Matlab que, utilizando una base de datos de firmantes, sea capaz de identificar cuando dos firmas pertenecen a una misma persona.

- Modificar el código anterior para comparar en su lugar firmas con marca de agua y ver si se sigue distinguiendo las que pertenecen a una misma persona.

- Generar un script de Matlab que añada una marca de agua a la firma y sea capaz de recuperar tanto la firma como la marca de agua posteriormente.

- Escribir un programa de Matlab que puede comparar una misma firma con su versión con marca de agua.

- Obtener pruebas sobre cuál es la fuerza adecuada para aplicar la marca de agua sobre la firma, dañando lo mínimo ambos elementos.

4. Metodología

Se ha visto que para este proyecto, hay que crear distintos códigos independientes entre sí, y obtener conclusiones de cada uno de ellos, por lo tanto, lo ideal sería hacer cada uno por separado y cuando se haya acabado de explorar una tarea por completo, pasar a la siguiente.

Probar cada función de manera secuencial, podría considerarse el uso de una metodología Waterfall.

El enfoque en cascada enfatiza una progresión estructurada entre fases definidas. Cada fase consiste en un conjunto definido de actividades y entregables que deben completarse antes de que pueda comenzar la siguiente fase. Aunque las fases suelen tener nombres diferentes, la idea básica es que la primera fase intenta capturar qué hará el sistema, sus requisitos de sistema y software; la segunda fase determina cómo se diseñará. La tercera etapa es donde los desarrolladores comienzan a escribir el código, la cuarta fase se dedica a la prueba del sistema y la fase final se centra en tareas de implementación, como capacitación y documentación extensa. Sin embargo, en la práctica de la ingeniería, el término "en cascada" se utiliza como un nombre genérico para todas las metodologías secuenciales de ingeniería de software [20].

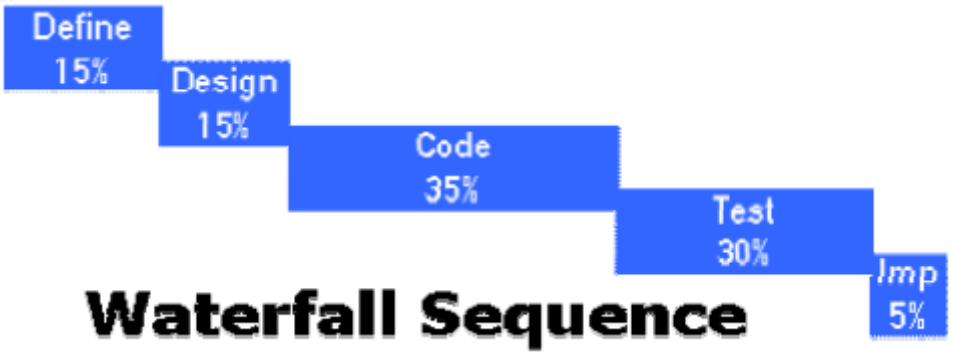


Fig. 4.1. La secuencia de Waterfall. [20]

5. Definición de requerimientos funcionales y tecnológicos

5.1. Funcionales

- Entender el uso de las firmas manuscritas.
- Investigar sobre la utilidad de las marcas de agua.
- Instalar el entorno de programación de Matlab en el ordenador.
- Obtener una base de datos de firmas.
- Ejecutar el código de Matlab en el ordenador.
- Obtener unas conclusiones y ver cómo se relacionan con la teoría obtenida.

5.2. Tecnológicos

El código se hará en la herramienta Matlab, este es un lenguaje de programación e IDE. Matlab es muy útil para trabajar con grandes cantidades de datos y permite mostrar representaciones visuales de los datos, lo cual será muy útil para documentar las conclusiones obtenidas.
Sin embargo, no se está muy familiarizado con Matlab, lo cual requerirá más tiempo para aprender el lenguaje y practicarlo, aun así es una buena oportunidad para salir de la zona de confort y aprender algo nuevo, por ese motivo se ha escogido este lenguaje de programación.

6. Desarrollo del proyecto

Se ha obtenido una base de datos de 330 firmantes, estas fueron recogidas para el MCYT baseline corpus [21].
Esta base de datos fue recogida en 2003 en colaboración entre cuatro universidades, esta se centra en huellas dactilares y firmas, pero para este proyecto solo se necesitan las firmas.

La MCYT database está compuesta por 330 firmantes, usando una tableta gráfica, cada participante firmó usando su propia firma 5 veces distintas, además de realizar imitaciones de las firmas de otros firmantes.

6.1. Código común en varios scripts

Los códigos a realizar incluyen leer la base de datos y procesarlos para poder ser analizados, estas funcionalidades se repiten para varios scripts. Lo mejor es comentar estos métodos y así cuando se repitan no hace falta entrar en detalle.

El primer paso es leer las firmas. Estas están en archivos .fpg, que son el resultado de recoger las firmas con la tableta gráfica. De cada firma se obtienen distintos parámetros:

- Posición en el eje x
- Posición en el eje y
- Presión aplicada por el lápiz
- Ángulo de azimut del lápiz
- Ángulo de inclinación del lápiz
- Si el lápiz está en el aire o tocando la tableta
- Momento en el que se ha recogido la información

Estos datos se leen en un loop, en el cual se recoge esta información y se llenan estas variables en el workspace.

```
for firmante=[0:144,200:274,300:374,400:434]
    for firma=firma_ini:firma_final
                disp(['Leyendo firmante nº ',num2str(firmante),' firma= ',num2str(firma)])
        %[vectores,nvectores,mvector,Fs]=leer_firma(nombre)
        if firmante < 10
            if firma < 10
                [vect,n,m,fs]=leer_firma([unidad,'000',num2str(firmante),'\000',num2str(firmante),'v0',num2str(firma),'.fpg']);
            else
                [vect,n,m,fs]=leer_firma([unidad,'000',num2str(firmante),'\000',num2str(firmante),'v',num2str(firma),'.fpg']);
            end
        elseif firmante < 100
            if firma < 10
                [vect,n,m,fs]=leer_firma([unidad,'00',num2str(firmante),'\00',num2str(firmante),'v0',num2str(firma),'.fpg']);
            else
                [vect,n,m,fs]=leer_firma([unidad,'00',num2str(firmante),'\00',num2str(firmante),'v',num2str(firma),'.fpg']);
            end
        else
            if firma < 10
                [vect,n,m,fs]=leer_firma([unidad,'0',num2str(firmante),'\0',num2str(firmante),'v0',num2str(firma),'.fpg']);
            else
                [vect,n,m,fs]=leer_firma([unidad,'0',num2str(firmante),'\0',num2str(firmante),'v',num2str(firma),'.fpg']);
            end
        end


        vect=trasladaFirma(vect,PCX,PCY);
        [vectn]=featuresdd(vect(:,1),vect(:,2),vect(:,3),point);
                disp(['calculando distancia firmante=',num2str(firmante),' firma nº= ',num2str(firma+firmas_train)])
        modelo{firmante+1,firma}=vectn;
    end %de firma
end %de firmante
```

Fig. 6.1.1. Loop principal de lectura de las firmas.

Los firmantes están separados por las universidades donde se recogieron las firmas, por eso el loop va saltando números. Del 0 al 144 son de la Universidad Politécnica de Madrid, de 200 a 274 son de Universidad de Valladolid, de 300 a 374 son de Universidad del País Vasco y de 400 a 434 son de Escuela Universitaria Politécnica de Mataró.
Los formatos de las firmas no son iguales, por lo que ha tenido que utilizar un if/else para detectar el tipo de firma y aplicar la transformación necesaria con el método *leer_firma*.

El método *leer_firma* recibe el archivo .fpg para leerlo. Devuelve varios valores:

- *vectores*: Una matriz que contiene los vectores de parámetros de las firmas.
- *nvectores*: El número de muestras (vectores) presentes en el archivo.
- *mvector*: El número de parámetros por vector (coordenadas X, Y, Z, Acimut e Inclinación).
- *fs*: La frecuencia de muestreo original de los datos de la firma.

Primero abre el archivo, verificando si está en el formato correcto y si se puede abrir. Lee la cabecera del archivo para determinar la versión del formato y extrae la información necesaria, este acaba recabando la información necesaria, aunque para algún formato tenga que hacerlo de manera diferente. Almacena en *vectores* la información obtenida y cierra el archivo.

Cuando se han leído las firmas, y sus datos están en el workspace, cada firma se traslada al centro de coordenadas para poder compararse con el resto, usando *trasladaFirma*.

```
function vect= trasladaFirma (vect,PCX,PCY)
% vect= trasladaFirma (vect)
%
% Traslada la firma contenida en el vector vect
% hasta que su centro de masas coincida con el
% centro de la Tableta Gráfica

% global PCX;
% global PCY;

[X,Y]=centroMasas (vect); %Calcula el centro de masas

distX=-X;
distY=-Y;
%distX=PCX-X;
%distY=PCY-Y;

disp(['Trasladando firma.']);
disp(['Moviendo coord X: ',int2str(distX),' posiciones']);
disp(['Moviendo coord Y: ',int2str(distY),' posiciones']);

   for i=1:size(vect,1)

      vect(i,1)=vect(i,1)+distX;
      vect(i,2)=vect(i,2)+distY;
   end
end
```

Fig. 6.1.2. Método *transladaFirma*.

```
function [X,Y]=centroMasas (firma,dibuja)
if nargin <2
    dibuja=0;
else
    dibuja=1;
end

X=centroMasasCol(firma(:,1));
Y=centroMasasCol(firma(:,2));


if dibuja

    plot(X,Y,'b*')
end
end
```

```
function    Media = centroMasasCol(vector)
N=size(vector,1);
Suma=0;
for i=1:N

   Suma=Suma+vector(i);
end
Media=Suma/N;
Media=round(Media);
end
```

Fig. 6.1.3. Método *centroMasas* y *centroMasasCol*.

El traslado se realiza comprobando dónde está el centro de masas de cada firma en ambos ejes. La variable *dibuja* indica si se debe dibujar el centro de masas, si se proporciona un valor, lo dibuja. Se llama a otro método para calcular el centro de masas de los ejes X e Y.

En *trasladaFirma* se mueve la firma las unidades necesarias para estar en el centro.

Otra función utilizada es *featuresdd* (figura 6.1.4), la cual devolverá una matriz normalizando las coordenadas, esta es más precisa entonces para hacer la comparativa entre otras. Esta función coge X, Y y presión. para devolver otra versión de la matriz con X, Y, presión, derivada de X, derivada de Y, derivada de la presión y la segunda derivada de X e Y. Para el cálculo de las derivadas se usan 11 puntos [22].

```
function [vectn]=featuresdd(x,y,p,point)
%normalization
avg1=mean(x);std1=std(x);xn=(x-avg1)./std1;
avg2=mean(y);std2=std(y);yn=(y-avg2)./std2;
avg3=mean(p);std3=std(p);pn=(p-avg3)./std3;
dx=audioDelta(x,point);
dy=audioDelta(y,point);
dp=audioDelta(p,point);
ddx=audioDelta(dx,point);
ddy=audioDelta(dy,point);
avg4=mean(dx);std4=std(dx);dxn=(dx-avg4)./std4;
avg5=mean(dy,point);std5=std(dy);dyn=(dy-avg5)./std5;
avg6=mean(dp);std6=std(dp);dpn=(dp-avg6)./std6;
avg7=mean(ddx);std7=std(ddx);ddxn=(ddx-avg7)./std7;
avg8=mean(ddy);std8=std(ddy);ddyn=(ddy-avg8)./std8;
N=length(x);
vectn=zeros(N,8);
vectn(:,1)=xn;
vectn(:,2)=yn;
vectn(:,3)=pn;
vectn(:,4)=dxn;
vectn(:,5)=dyn;
vectn(:,6)=dpn;
vectn(:,7)=ddxn;
vectn(:,8)=ddyn;
end
```

Fig. 6.1.4. Función *featuresdd*.

6.2. Comparación de firmas por DTW

El propósito del primer script, mainDTWfirmas.m, es utilizar la base de datos para analizar múltiples firmas de distintas personas, compararlas entre sí, y determinar mediante DTW en qué medida el programa puede identificar si una firma pertenece a la misma persona, incluso cuando en la base de datos hay firmas que son réplicas de las de otras personas.

Una vez se han leído y almacenado las firmas en el workspace, y estas han sido movidas al centro de coordenadas, ya se puede operar el cómo encontrar la similitud entre ellas.

```
vect=trasladaFirma(vect,PCX,PCY);
[vectn]=featuresdd(vect(:,1),vect(:,2),vect(:,3),point);
        disp(['calculando distancia firmante=',num2str(firmante),' firma nº= ',num2str(firma+firmas_train)])
modelo{firmante+1,firma}=vectn;
```

Fig. 6.2.1. Tratamiento y almacenaje de las firmas en el workspace.

Para ello, se itera entre todas las firmas de los firmantes y se calcula la distancia entre la firma actual y todas las demás, utilizando DTW (Dynamic Time Warping) [23].

El algoritmo de DTW encuentra el mejor alineamiento entre dos secuencias, minimizando la distancia entre los puntos correspondientes a lo largo de las dos secuencias. Esto se realiza mediante la creación de una matriz de costos, donde cada celda representa el costo de alinear dos puntos en las secuencias. Luego, se encuentra el camino de menor costo a través de esta matriz, lo que indica cómo deben alinearse las dos secuencias. Este camino define la correspondencia óptima entre los puntos de las secuencias, permitiendo así la comparación basada en esta alineación.

A diferencia de las técnicas de comparación de series de tiempo tradicionales, como la distancia euclídea, DTW permite alinear las secuencias temporalmente antes de calcular la similitud. Esto es especialmente útil cuando las secuencias tienen diferentes longitudes o cuando las relaciones temporales entre los puntos son más importantes que sus ubicaciones absolutas. [24].

Tras generar una matriz con cada valor de la distancia entre una firma y todas las demás, se escoge la que tiene menor distancia como la firma más similar.

```
disp(['calculando distancia firmante=',num2str(firmante),' firma nº= ',num2str(firma+firmas_train)])
for firmantes=[0:144,200:274,300:374,400:434]
    for firmas=1:firmas_train
        distan(firmas)=dtw(modelo{firmantes+1,firmas}',vectn')/length(modelo{firmantes+1,firmas});
    end
    mat_d(firmante+1,firmantes+1,firma)=min(distan);
end
```

Fig. 6.2.1. Parte del loop principal donde comparan las firmas entre sí mediante DTW.

Por último se calcula la tasa de aciertos, esto se consigue inicializando un contador de aciertos y errores e iterando de nuevo las firmas, viendo si ha sido capaz de reconocer si es de la misma persona o no, y sumando un punto a acierto o error en función de si lo identifica correctamente.

```
%calcula tasa de aciertos
temp=mat_d(1+[0:144,200:274,300:374,400:434],1+[0:144,200:274,300:374,400:434],:);
acierto=0;
error=0;
for firma=firma_ini:firma_final
    for locu=1:330
        [emin,reconocido]=min(temp(locu,:,firma));
        if (reconocido==locu), acierto=acierto+1;
        else, error=error+1;
        end
    end
end
tasa(point)=100*acierto/(acierto+error);
```

Fig. 6.2.2. Cálculo de la tasa de aciertos.

El resultado final del código es su tasa de acierto, el DCF óptimo para firmas genuinas y de impostores.

El DCF se define como una combinación lineal de la tasa de falsa aceptación y la tasa de falsa omisión, donde los pesos de estas tasas reflejan los costos relativos de cada tipo de error en el contexto de una aplicación específica. El objetivo es minimizar el DCF, lo que implica encontrar un equilibrio óptimo entre la tasa de falsa aceptación y la tasa de falsa omisión. [25]

En este programa hay dos tipos de DCF, el de las falsificaciones aleatorias y el de las habilidosas. Primero se ha intentado comprobar comparando todas las falsificaciones entre sí, para ver si alguna podía hacerse pasar por la firma en cuestión, eso da lugar a las falsificaciones aleatorias. Posteriormente se ha intentado calcular la DCF de las falsificaciones que han sido realizadas viendo la firma de otra persona e intentando recrearla.

```
DTW Ident.(%)=99.3333 random forgeries DCF_opt=0.0059805 skilled DCF_opt=0.030242
```

Este es el resultado final, se puede apreciar una tasa de aciertos del 99.33% (1639 acierto y 11 errores) y DCF óptimo para firmas genuinas de 0.005 y de 0.03 para las de impostores.

El DTW compara dos firmas, estas han de ser dos matrices de mismas dimensiones de los vectores, aunque las longitudes de las mismas pueden ser variables. La comparación realizada es entre los resultados de la aplicación de featuresdd, lo cual es: X, Y, presión, derivada de X, derivada de Y, derivada de la presión y la segunda derivada de X e Y. Para el cálculo de las derivadas se usan 11 puntos.
Sin embargo se podría hacer la comparación por DTW directamente sobre el vector recogido en la base de datos, sobre sus variables X, Y y presión. En este caso el resultado es menos preciso. Esto se ha probado creando un script variando el script DTWmain.m para aplicar esto sobre el propio *vect* obtenido de leer el documento, sin ninguna normalización, el nuevo script se ha llamado DTWtoVect.m.

```
vect=trasladaFirma(vect,PCX,PCY);
disp(['calculando distancia firmante=',num2str(firmante),' firma nº= ',num2str(firma+firmas_train)])

for firmantes=[0:144,200:274,300:374,400:434]
    for firmas=1:firmas_train
        distan(firmas)=dtw(modelo{firmantes+1,firmas}',vect')/length(modelo{firmantes+1,firmas});
    end
    mat_d(firmante+1,firmantes+1,firma)=min(distan);
end
```

```
DTW Ident.(%)=97.7576 random forgeries DCF_opt=0.012935 skilled DCF_opt=0.043636
```

Fig. 6.2.3. Captura del código y resultado de la ejecución aplicando DTW sobre la firma directamente.

```
vect=trasladaFirma(vect,PCX,PCY);
watermarked_signal=addWatermark(vect,alpha);
[vectn]=featuresdd(watermarked_signal(:,1),watermarked_signal(:,2),vect(:,3),point);
disp(['calculando distancia firmante=',num2str(firmante),' firma nº= ',num2str(firma+firmas_train)])


for firmantes=[0:144,200:274,300:374,400:434]
    for firmas=1:firmas_train
        distan(firmas)=dtw(modelo{firmantes+1,firmas}',vectn')/length(modelo{firmantes+1,firmas});
    end
    mat_d(firmante+1,firmantes+1,firma)=min(distan);
end
```

```
DTW Ident.(%)=99.3333 random forgeries DCF_opt=0.0059805 skilled DCF_opt=0.030242
```

Fig. 6.2.4. Captura del código y resultado de la ejecución aplicando DTW sobre la firma normalizada.

Viendo que el resultado obtenido mejora aplicando dtw sobre la versión normalizada de la firma, se va a utilizar este procedimiento a menos que se indique lo contrario.

6.3. Adición de la marca de agua mediante DCT y DWT

La adición de la marca de agua sobre la firma, tal y como ha sido explicado anteriormente, se puede realizar con diversos métodos, para este apartado práctico se emplea el uso de DCT y de DWT, y se pretende mostrar el proceso de aplicación de la misma. Para estas pruebas se han desarrollado los scripts de watermark414DCT.m y watermark414DWT.m.
Estos trabajan sobre solamente una firma, la 414, de ahí el nombre, y aplican los principios de DCT y DWT para la aplicación de la marca de agua.

Esta demostración de estas técnicas se hace con una única firma, posteriormente se aplica al resto de firmas de la base de datos.
El código empieza leyendo la firma, trasladándose al centro de masas y normalizandola.

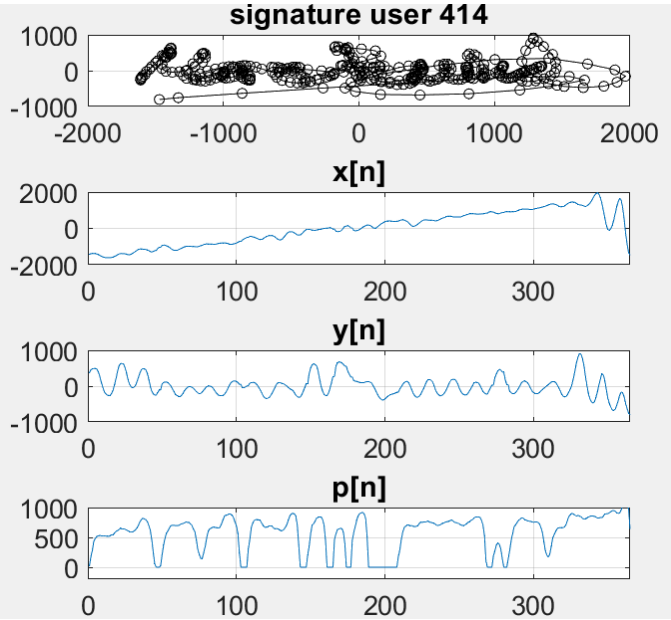


Fig. 6.3.1. Representación de la firma mediante sus puntos registrados y del movimiento de ejes X e Y

Esta representación muestra los puntos obtenidos de la lectura de los datos para una determinada firma, y los relaciona, generando una representación visual de la firma tal y como la escribió el firmante, esta representación es obtenida por los valores de X e Y.

La marca de agua a añadir consiste en una matriz de valores binarios aleatorios sobre los valores de X e Y de una misma firma. Los valores utilizados para este caso se generan aleatoriamente, pero podría añadirse mediante código binario cualquier mensaje.

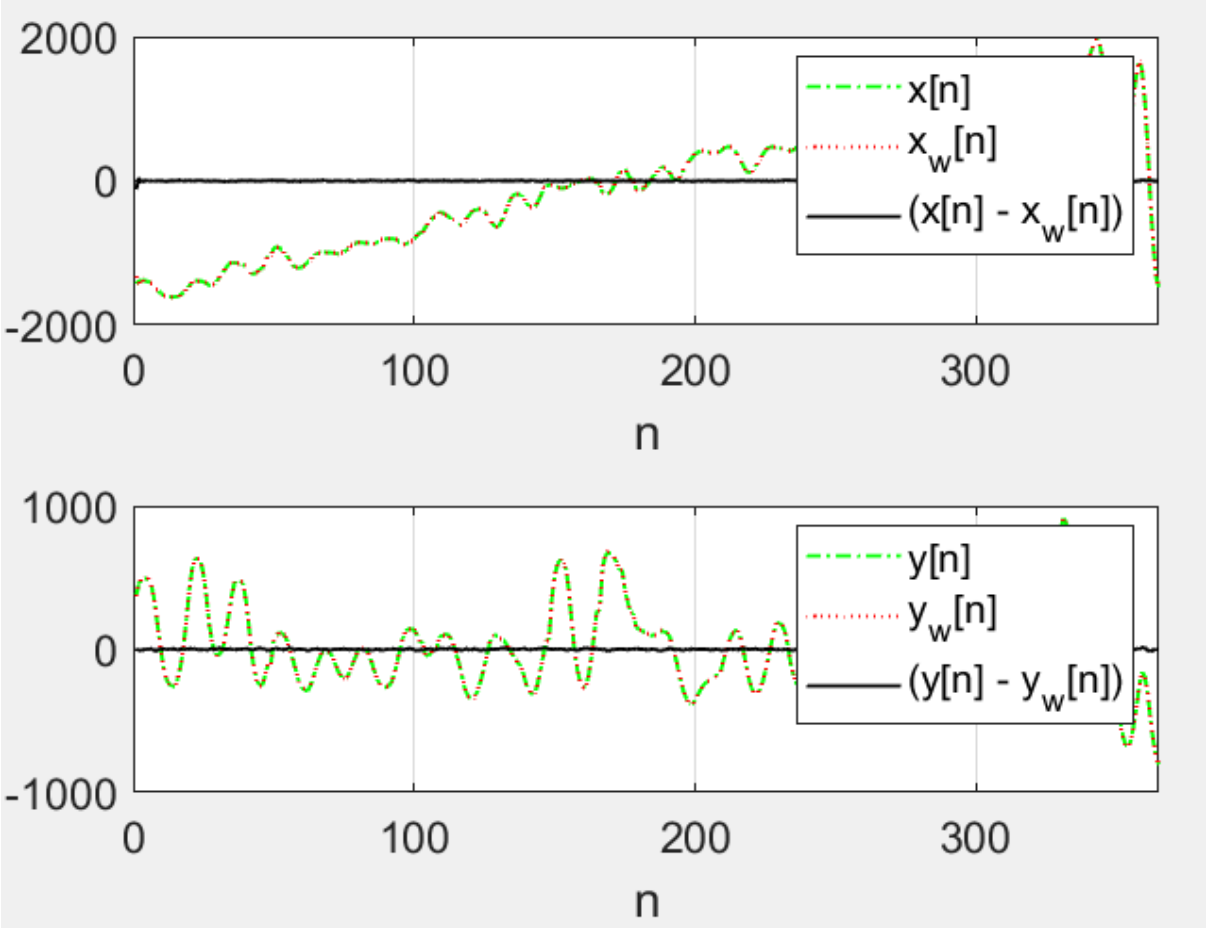


Fig. 6.3.2. Diferencia en los ejes X e Y de la firma original respecto a su versión con marca de agua de *alpha* 10

En este gráfico se observa que hay una sutil pero apreciable diferencia entre ambas versiones. La línea negra en el centro representa lo que se modifica respecto a la firma original, y se aprecia que esta no es totalmente recta.

Poder visualizar cómo es realmente la firma permite ver las diferencias con su versión con marca de agua, como se mostraría en la siguiente figura.

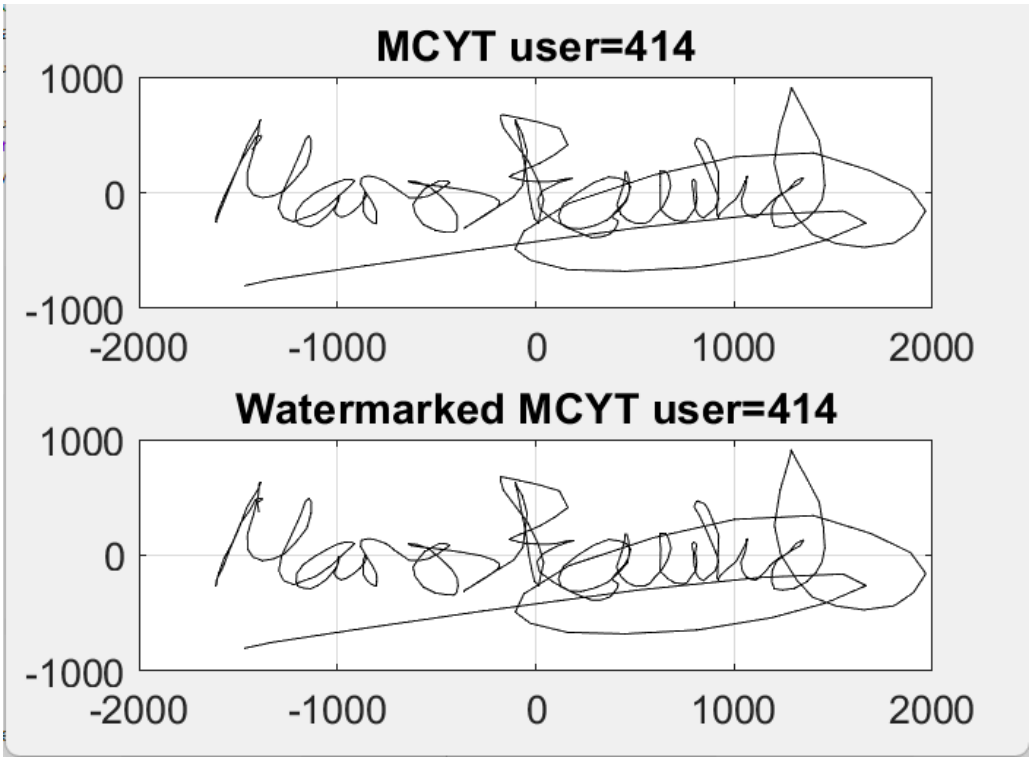


Fig. 6.3.3. Diferencias entre la firma sin marca y su versión marcada con un *alpha* de 2.

Las firmas parecen similares, y perfectamente a primera vista puede engañar a la gente, sin embargo, al prestar la suficiente atención se aprecia que esta firma tiene cambios respecto a su versión original.

Las marcas de agua son aplicadas sobre la firma con una fuerza determinada, esta será llamada *alpha*, cuanto menor sea esta *alpha*, la marca de agua será más sutil y cuanto mayor sea, más impacto tendrá la marca de agua sobre la firma original.

La adición de la marca de agua a la firma es realizada en este proyecto de las siguientes manera:

- DCT (Transformada discreta del coseno):

    1. Generación de las marcas de agua: Se generan dos marcas de agua utilizando números aleatorios para cada firma, una para cada coordenadas (X e Y).
    2. Transformada DCT de la firma digital: Se aplica la Transformada Coseno Discreta (DCT) a las coordenadas X e Y de la firma digital utilizando la función dct2. La DCT transforma la firma digital del dominio del tiempo al dominio de la frecuencia, produciendo coeficientes DCT que representan la contribución de diferentes frecuencias a la firma digital.
    3. Incrustación de las marcas de agua en los coeficientes DCT: Se incrustan las marcas de agua en los coeficientes DCT de la firma digital modificando los coeficientes DCT originales. Esto se hace multiplicando las marcas de agua por el valor de alpha y sumándolas a los coeficientes DCT originales.
    4. Transformada DCT inversa para obtener la firma digital marcada: Se aplica la Transformada Coseno Discreta Inversa (IDCT) a los coeficientes DCT

modificados utilizando la función idct2. Esto invierte la transformación DCT y produce la firma digital marcada reconstruida en el dominio del tiempo, que contiene las marcas de agua incrustadas.

```
function watermarked_signal=addWatermark(vect,alpha)
watermark1=round(rand(length(vect(:,1)),1));
watermark2=round(rand(length(vect(:,1)),1));

% Perform DCT on the original signal
dct_original = dct2(vect(:,1:2));

% Embed the watermark in the DCT coefficients
dct_watermarked = dct_original + alpha * [watermark1,watermark2];

% Inverse DCT to obtain watermarked signal
watermarked_signal = round(idct2(dct_watermarked));

disp(['Waterkmark aplicada con alpha ',num2str(alpha)])
end
```

Fig. 6.3.4. Inserción de marcas de agua a una firma usando DCT

- DWT (Transformada discreta de wavelet):

1. Generación de las marcas de agua: Se generan dos marcas de agua utilizando números aleatorios.
2. Aplicación de la DWT a la firma digital: Se aplica la Transformada Discreta Wavelet (DWT) a las coordenadas X e Y de la firma digital. Esto se hace utilizando la función dwt, que descompone la firma digital en dos componentes: una aproximación (LL) y una detalle (HL) en cada dimensión (X e Y). Esto proporciona una representación en el dominio de la frecuencia y la escala de la firma digital.
3. Incrustación de las marcas de agua: Se añaden las marcas de agua a las componentes de aproximación (LL) de las coordenadas X e Y de la firma descompuesta. Esto se hace multiplicando las marcas de agua por el valor de alpha y sumándolas a las componentes de aproximación. Esto incrusta las marcas de agua en la firma digital sin alterar significativamente su contenido.
4. Reconstrucción de la firma marcada: Se reconstruye la firma digital marcada a partir de las componentes de aproximación modificadas y las componentes de detalle originales en cada dimensión (X e Y). Esto se hace utilizando la función idwt, que es la inversa de la DWT. La firma digital reconstruida ahora contiene las marcas de agua incrustadas en ambas dimensiones.
5. Manejo de la longitud impar: Si la longitud de la firma digital es impar, se añade el último valor de la coordenada X e Y de la firma original a las coordenadas X e Y de la firma marcada. Esto es para mantener la consistencia en la longitud de las firmas digitales en ambas dimensiones.

```
watermark1=round(rand(watermark_length,1));
watermark2=round(rand(watermark_length,1));

% Apply DWT to the original signal
wname='haar';

[LLx, HLx] = dwt(vect(1:signal_length,1), wname);
[LLy, HLy] = dwt(vect(1:signal_length,2), wname);

LL_wmx = LLx + alpha * watermark1;
LL_wmy = LLy + alpha * watermark2;

% Reconstruct the watermarked signal
watermarked_signal1 = round(idwt(LL_wmx, HLx, wname));
watermarked_signal2 = round(idwt(LL_wmy, HLy, wname));
if(is_odd_length)
    watermarked_signal1=[watermarked_signal1;vect(end,1)];
    watermarked_signal2=[watermarked_signal2;vect(end,2)];
end
```

Fig. 6.3.5. Inserción de marcas de agua usando DWT

Ambas formas de insertar la marca de agua son funcionales, pero durante la mayoría de scripts se utiliza el método de DCT.

6.4. Alpha mínima en la marca de agua

Una de las intenciones de insertar una marca de agua es la posterior recuperación y lectura de la misma, por lo que la recuperación de estos datos ha de ser una prioridad. Esta información está almacenada en cada bit de la firma, así que se puede extraer la marca de agua y compararse con la añadida al inicio para ver que % de bits son los que conservaría iguales.

```
% Extract the watermark from the DCT coefficients
extracted_watermark = round((dct_watermarked - dct_original) / alpha);
%number of erroneous bits
errors=sum(xor(extracted_watermark,[watermark1,watermark2]),'all');
% % of erroneous bits
percentage_error(index)=100*errors/numel(extracted_watermark);
```

Fig. 6.4.1. Código de la extracción de la marca de agua y contabilización de bits erróneos

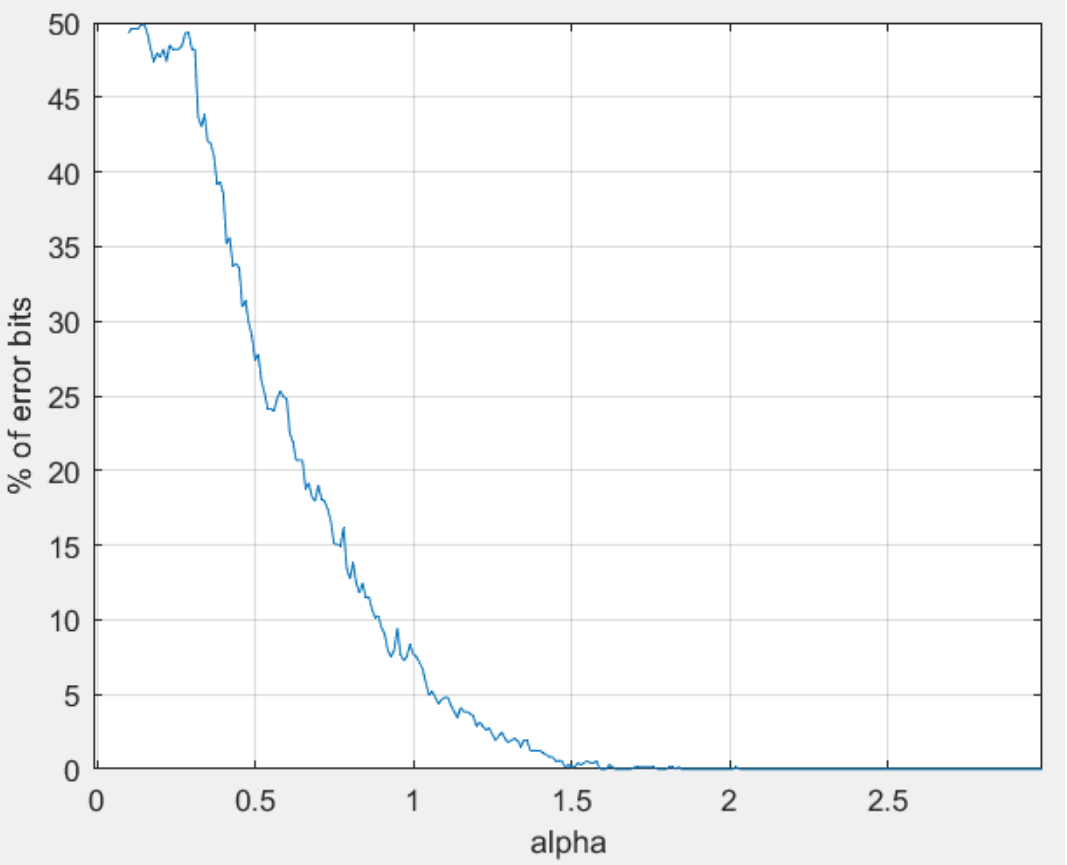


Fig. 6.4.2. Gráfico que muestra el % de bits erróneos a lo largo de distintos valores de alpha para una única firma.

Este gráfico compara la marca creada originalmente con la extraída de la firma, se aprecia en el gráfico que con un menor *alpha*, esta firma tiene mayor cantidad de bits erróneos respecto a su original que con un *alpha* mayor. A partir de aproximadamente un valor de *alpha* de 2, esta marca de agua extraída no presentaría ninguna diferencia con la original.

En el script de MinAlphaChecked.m se ha probado cual es el valor de *alpha* mínimo por el cual se obtiene un 0% de bits erróneos en distintas firmas, para eso se ha utilizado las

firmas de MCYT, y se ha comprobado en qué punto al aplicarles la marca de agua, esta se puede extraer sin obtener fallos.

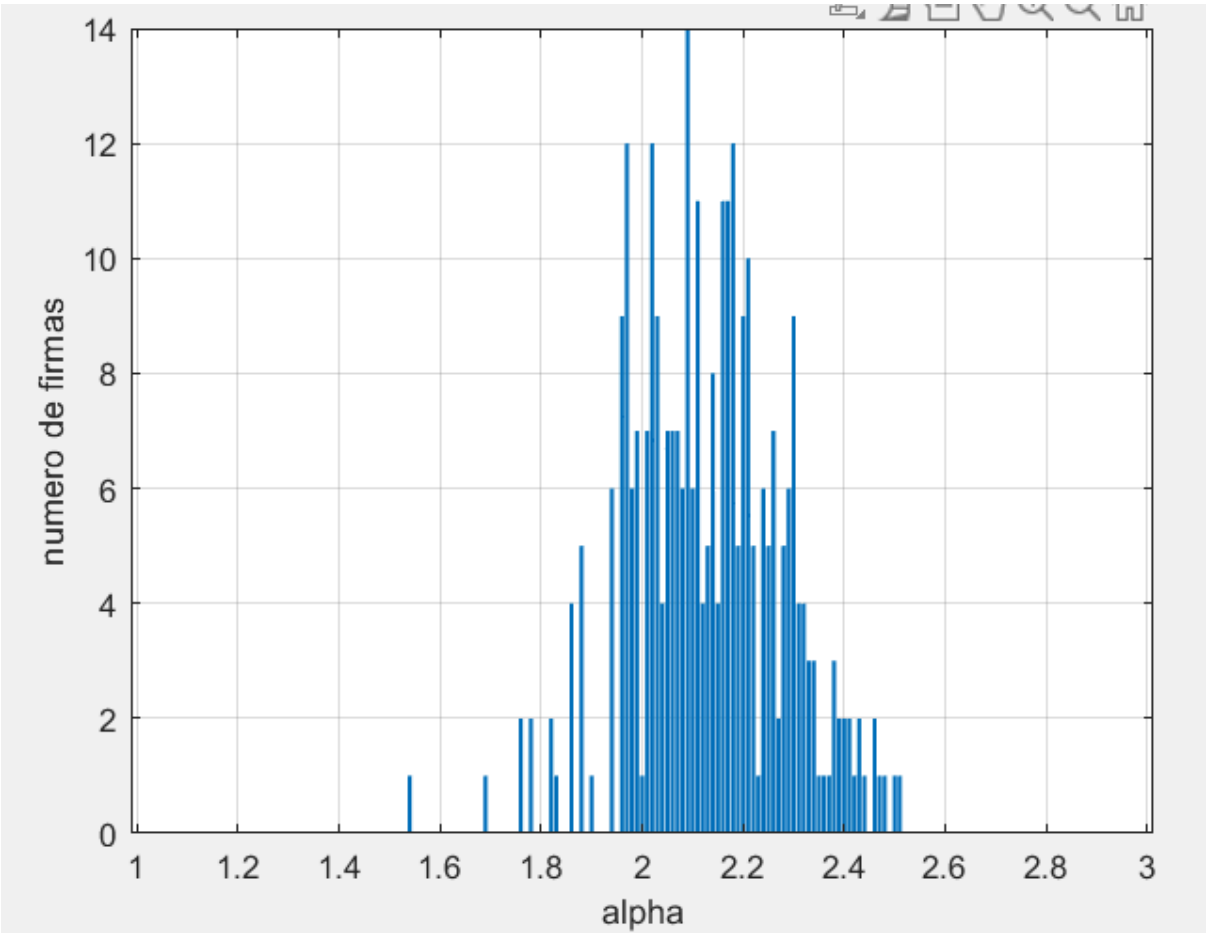


Fig. 6.4.3. Gráfico que muestra la cantidad de firmas de MCYT que tienen un 0% de bits erróneos en cada valor de *alpha*.

Esta información ha sido obtenida contabilizando a qué valor de *alpha* una firma de cada uno de los firmantes llega a no tener bits erróneos y es capaz de mantenerse así durante unos cuantos valores más (para asegurarse que es un valor estable y no aumente de nuevo posteriormente). Se utiliza una firma de cada firmante en lugar de todas las firmas porque cada firmante repite la suya varias veces, por lo cual si se usasen todas, es probable que algunas firmas tuviesen resultado similares entre sí.

```
            %number of erroneous bits
            errors=sum(xor(extracted_watermark,[watermark1,watermark2]),'all');
            disp(['Alpha=',num2str(alpha),' erroneous bits=',num2str(errors),', % erroneous bits=
            percentage_error(index)=100*errors/numel(extracted_watermark);
            if errors>0
                confirmation=0;
            end
            if errors == 0 && lastErrorFound(index) == 4 && confirmation<confirmNeeded
                confirmNum=alpha;
                confirmation=confirmation+1;
            end
            if errors == 0 && lastErrorFound(index) == 4 && confirmation==confirmNeeded
                lastErrorFound(index) = confirmNum;
                confirmation=0;
            end
        end
        index=index+1;
end %de firmante
```

Fig. 6.4.3. Código que muestra el proceso de almacenar el último valor de *alpha* en el cual el error es 0 y puede mantenerse estable.

Media	Moda	Mediana	Desviacion_Estandar	Minimo	Maximo
2.1132	2.09	2.11	0.16281	1.54	2.51

Fig. 6.4.3. Estadísticas descriptivas del valor de alpha en que cada firma de MCYT tiene un 0% de bits erróneos.

Con este gráfico obtenido del análisis de 330 firmas se ve que el punto más bajo de *alpha* es 1,54 y el más alto 2,51. De esto se puede asumir que con un valor mínimo de *alpha* 3 no debería haber problema para extraer la marca de agua y no encontrar errores respecto a la original.

6.5. Alpha máxima en la marca de agua

Como se ha visto en el apartado de la adición de las marcas de agua por DCT y DWT, el impacto de la marca de agua sobre su firma es dependiente de la fuerza con la que se aplique, o como se la llama anteriormente, su *alpha*.
Anteriormente se ha visto que si el *alpha* es más pequeño de 2.51 entonces la firma no puede ser extraída correctamente, porque se obtendría con bits erróneos, diferentes de la marca de agua original, cuanto menor *alpha*, mayor número de errores en la marca de agua extraída.
Se puede comprobar que aunque el alpha aumente enormemente, se podría extraer sin problema la marca de agua, con 0% de bits erróneos.

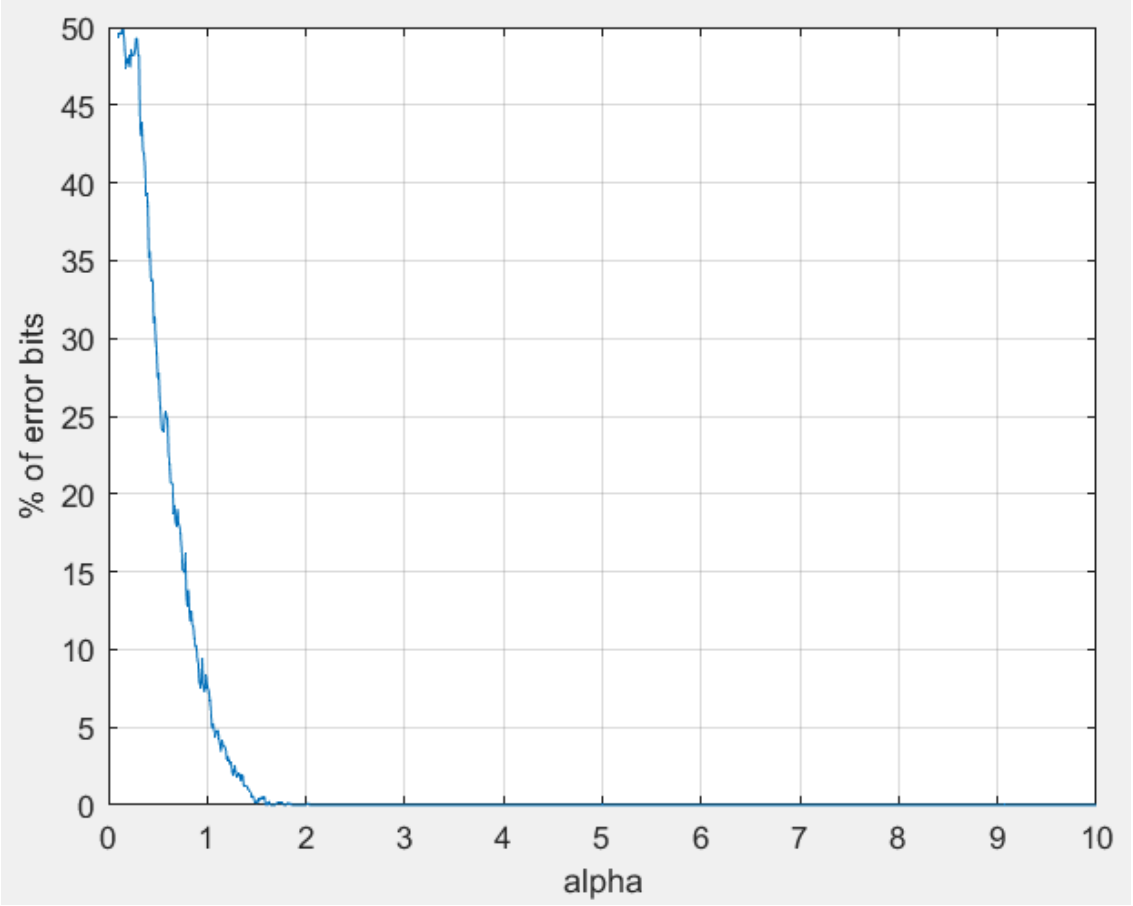


Fig. 6.5.1. Gráfico que muestra el % de bits erróneos de la marca de agua extraída a lo largo de distintos valores de alpha.

Ver que la marca de agua se puede extraer sin problema siempre y cuando supere el mínimo establecido previamente puede hacer creer que entonces no hay nada que impida hacer marcas de agua con un alpha enorme. Se podría llegar a asumir que la extracción siempre será satisfactoria siempre y cuando no sea más bajo del alpha mínimo para la firma.
Esta asunción es incorrecta, puesto que la firma se modifica enormemente en el momento de la inserción.

Se aprecia esta modificación en los cálculos de MSE y MAE de una firma marcada.
El Mean Squared Error (MSE):

- El MSE calcula la media cuadrática de las diferencias entre los valores originales y los valores con marca de agua.
- Cuanto más bajo sea el MSE, mejor será la calidad de la señal marcada, ya que indica que las diferencias entre la señal original y la señal marcada son pequeñas.

El Mean Absolute Error (MAE):

- El MAE calcula la media de las diferencias absolutas entre los valores originales y los valores marcados.
- Al igual que el MSE, un MAE más bajo indica una mejor calidad de la señal marcada.
- El MAE es menos sensible a los valores atípicos que el MSE, ya que no eleva al cuadrado las diferencias.

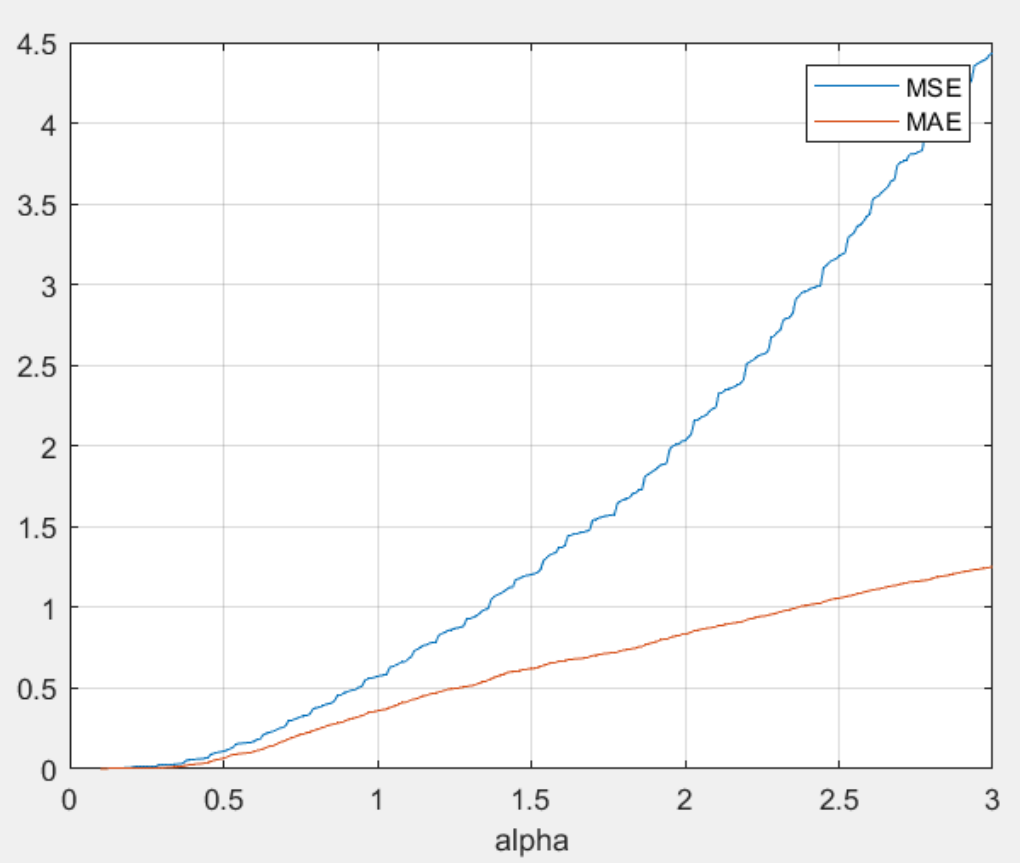


Fig. 6.5.2. Gráfico que muestra en una sola única firma el MAE y MSE a lo largo de distintos valores de alpha.

El ver este gráfico puede no dar tanto la impresión de cómo se deforma la firma respecto a la original con alphas altas, sin embargo se puede apreciar mejor al representar esta misma.

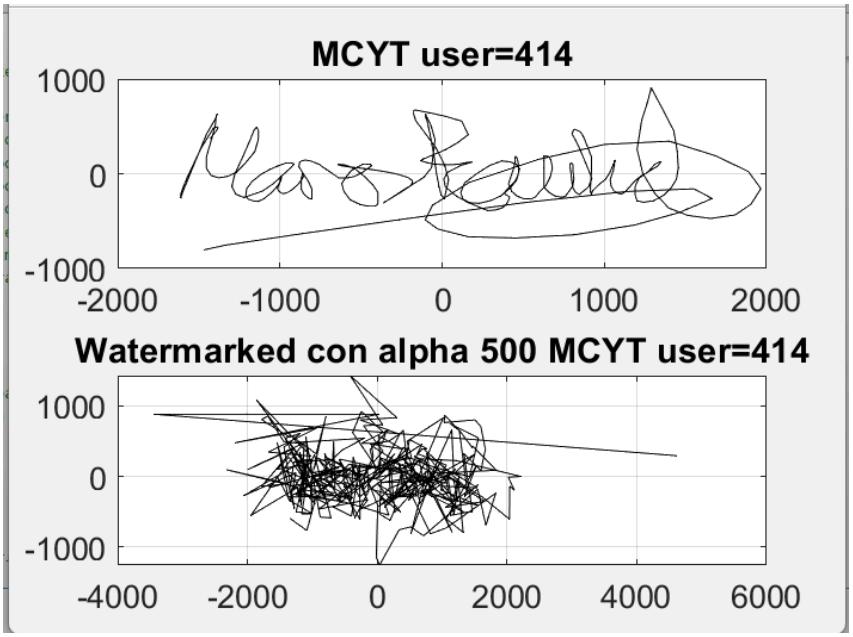


Fig. 6.5.3. Muestra de la deformación de la firma con watermark aplicada con un alpha de 500.

Se ha visto que con un alpha de 2.5 ya se podrían extraer los bits de marca de agua correctamente, pero se puede teorizar sobre cuál sería el otro extremo, en caso de tener una marca de agua más fuerte de 2.5, cómo afectaría a la comparación de firmas por DTW.

Como se ha visto ya, al aplicar el DTW sobre cada una de las firmas, esta produce un porcentaje de identificación de las firmas, este se obtiene de comparar con DTW la distancia entre dos firmas y luego ver si el programa lo ha descubierto satisfactoriamente. Este resultado es los aciertos entre la suma de los aciertos y los fallos. Este valor según se obtuvo del programa original es 99.3333%. También se investiga con cada firma su comparación con la demás firmas (falsificaciones aleatorias) o firmas que intencionalmente les tratan de copiar (falsificaciones habilidosas), estas en la primera ejecución se ve que el DCF óptimo es de 0.0059 para las falsificaciones aleatorias y 0.030 para las habilidosas.

Este proceso se ha comprobado usando el script DTWMaxAlphaChecker.m, este código ejecuta un código similar al mainDTWfirmas.m, sin embargo lo ejecuta aplicando una marca de agua previamente y lo largo de diversos valores de alpha separados, y luego compara estos resultados entre sí.

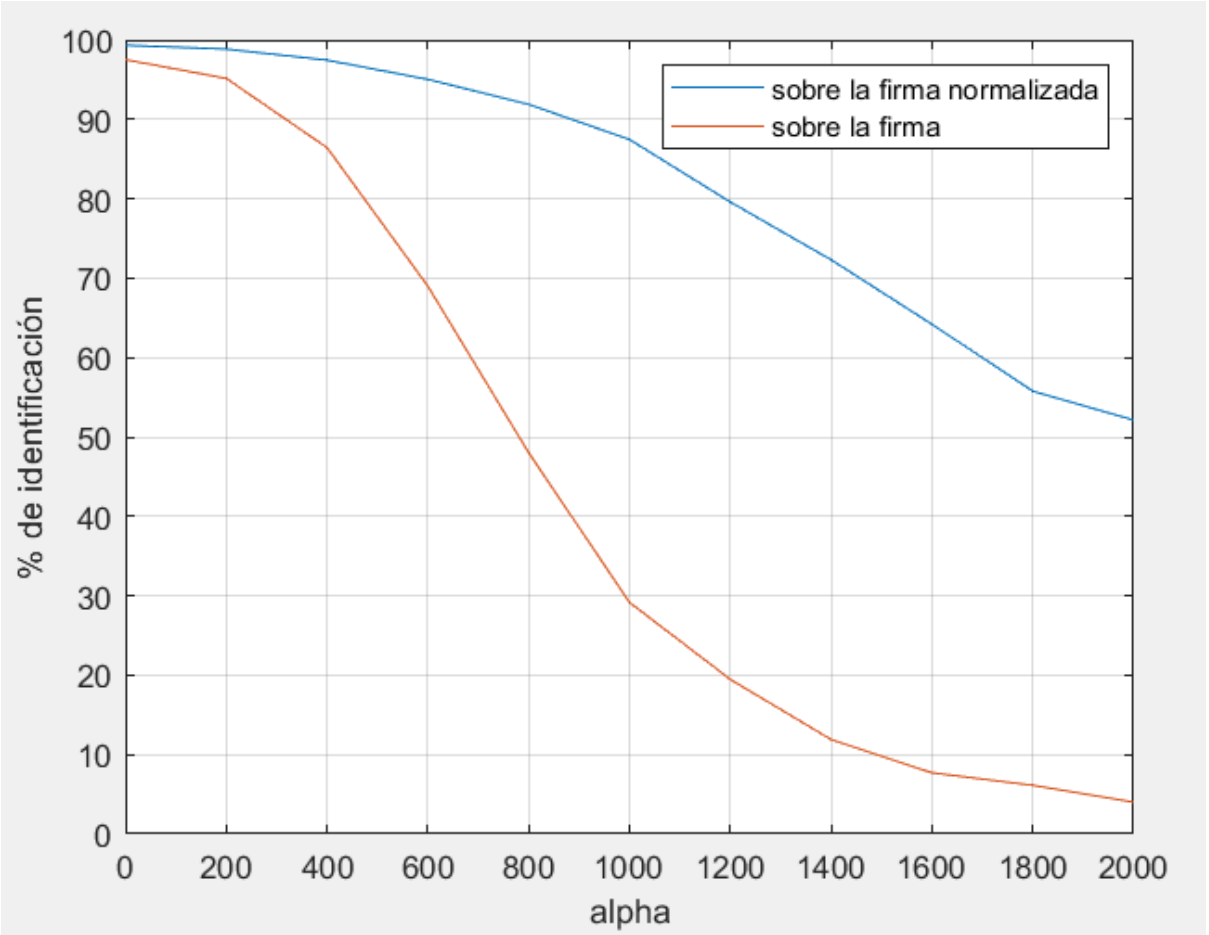


Fig. 6.5.4. % de identificación a lo largo de distintos valores de *alpha* usando dtw sobre la firma normalizada y sin normalizar

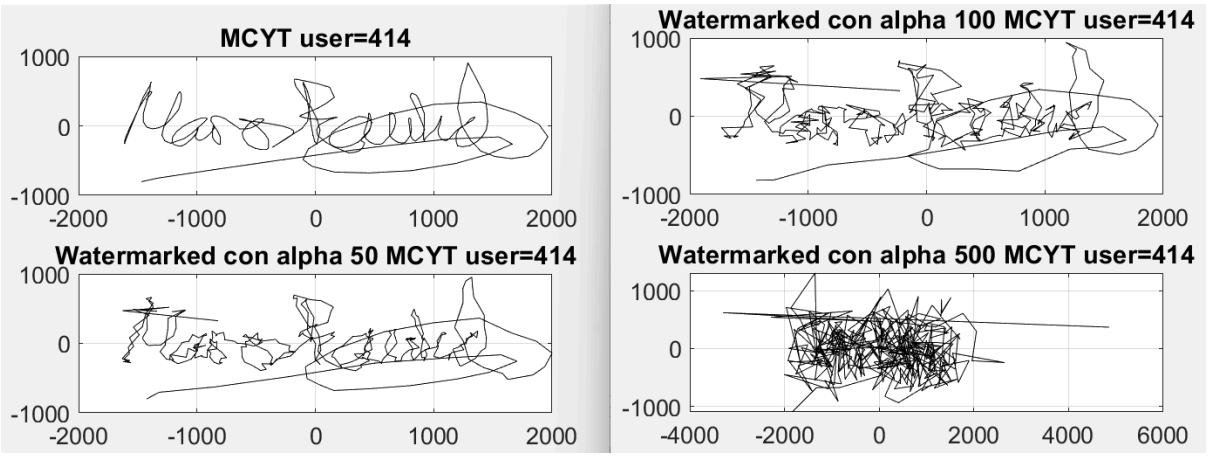
En este gráfico se puede apreciar como según aumenta el alpha, disminuye también el reconocimiento de las firmas, pues el aplicar la marca de agua con mayor fuerza deforma las firmas, al principio la deformación es tan pequeña que varía muy levemente el resultado, pero según aumenta el alpha, esta deformación se ve más clara, y esto se agudiza más si la transformación proviene de la firma sin aplicarle la normalización.

```
Waterkmark aplicada con alpha 80
calculando distancia firmante=434 firma n°= 29
point=11 DTW Ident.(%)=99.1515 random forgeries DCF_opt=0.0059805 skilled DCF_opt=0.030848

Waterkmark aplicada con alpha 150
calculando distancia firmante=434 firma n°= 29
point=11 DTW Ident.(%)=98.9697 random forgeries DCF_opt=0.0071944 skilled DCF_opt=0.033455

Waterkmark aplicada con alpha 250
calculando distancia firmante=434 firma n°= 29
point=11 DTW Ident.(%)=98.8485 random forgeries DCF_opt=0.0092014 skilled DCF_opt=0.037152
```

Fig. 6.5.5. Resultado de diversas ejecuciones usando alphas de 80, 150 y 250



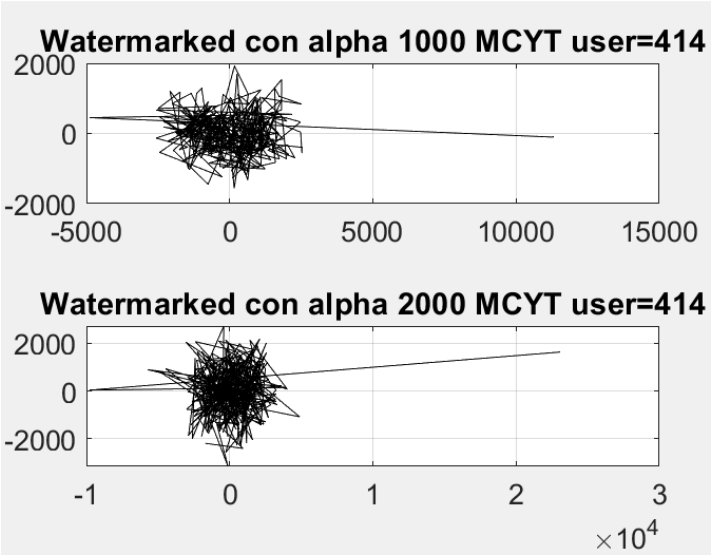


Fig. 6.5.6. Modificación de la firma con *alpha* de 0, 50,100, 500, 1000 y 2000

A lo largo de diversas ejecuciones no solo cambia el porcentaje de identificación de la firma, sino que también los DCT se modifican al aumentar el alpha.

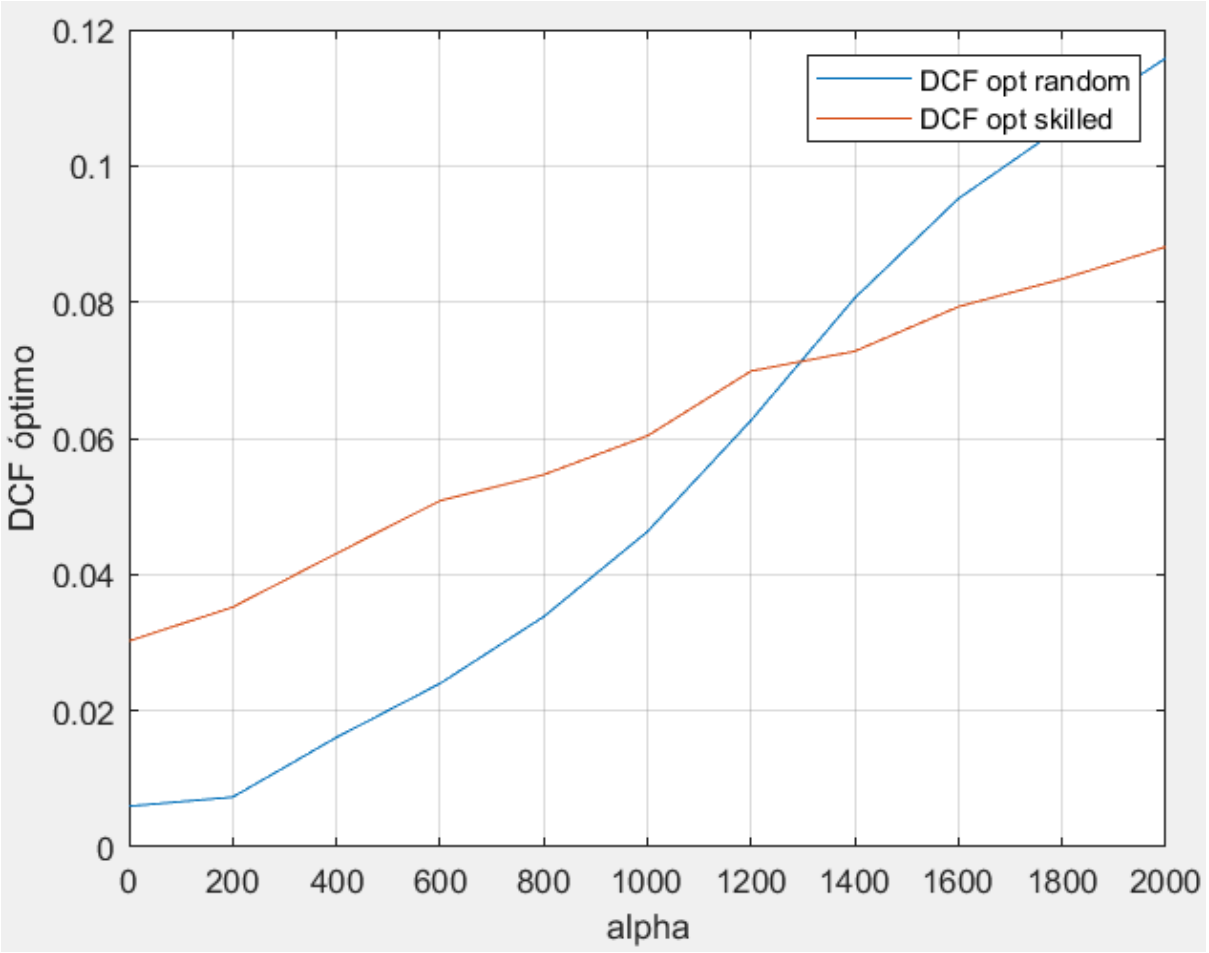


Fig. 6.5.7. DCF óptimo a lo largo de distintos valores de *alpha*

Se observa que a medida que el valor de alpha aumenta y, en consecuencia, la firma se deforma, los DCF también aumentan. Este fenómeno parece deberse a que a medida que las firmas se asemejan más, debido a una mayor modificación, puesto que son todos garabatos, resulta más fácil confundir al programa, lo que hace que las falsificaciones sean más convincentes.

Una observación interesante en el gráfico es que a partir de un valor de alpha de 1300, los resultados de los dos tipos de falsificaciones se cruzan. Esto se debe a que las falsificaciones habilidosas se mantienen más diferenciadas entre sí que las falsificaciones aleatorias. Aunque ambas llegado este nivel de alpha están siendo representadas por garabatos, los garabatos de las falsificaciones habilidosas están más próximos a los de las firmas originales que los de las falsificaciones aleatorias. Por lo tanto, el aumento en la tasa de error de las falsificaciones habilidosas es más gradual.

De este apartado se extrae que no solo tiene importancia que el alpha sea mayor que un mínimo para así poder extraerla correctamente, sino que cuando más aumenta peor se distinguirá la firma, por lo cual el valor de alpha no ha de aumentar demasiado.

6.6. Escritura mediante brazo robótico

Este apartado ha sido elaborado en colaboración con Rubén Noya Soriano, estudiante del grado de ingeniería electrónica industrial y automática, en su proyecto “Escritura con robot colaborativo (cobot)”. Su TFG se centra en la tarea de lograr que un brazo robótico sea capaz de firmar con precisión. Se plantea que este brazo robótico pueda replicar firmas “humanas” con marcas de agua, las cuales serán leídas por una tableta gráfica para su posterior comparación con las originales.

Es crucial que las marcas de agua sean resistentes a posibles ataques, como la repetición de una firma genuina en un intento de hacerla pasar por auténtica. Se ha documentado la detección de intentos de repetición de firmas, por ejemplo, en falsificaciones habilidosas mencionadas en el MCYT [21]. Sin embargo, además de la posibilidad de que una persona replique una firma que está visualizando para hacerla pasar por propia, también existe el riesgo de replicación por parte de un brazo robótico.

Fig. 6.6.1. Brazo robótico utilizado para esta apartado

La réplica de una firma por parte de una persona en comparación de la réplica de un brazo robótico se diferenciaría en:

- Precisión y habilidad: Un brazo robótico generalmente está diseñado para movimientos precisos y repetitivos. Cuando se programa correctamente, puede replicar un trazo con una precisión muy alta, lo que resulta en una copia muy cercana a la firma original. En cambio, una persona que intenta falsificar una firma puede no tener la misma habilidad o precisión en su trazo, lo que podría resultar en una firma falsificada que sea menos precisa o similar a la original.

- Consistencia: Los brazos robóticos pueden replicar el mismo trazo una y otra vez con una consistencia casi perfecta, mientras que una persona puede tener dificultades para mantener la misma calidad y estilo de firma en cada intento de falsificación.

- Velocidad: Dependiendo de la programación y la configuración, un brazo robótico puede realizar la tarea de copiar una firma a una velocidad constante y rápida. Por otro lado, una persona puede llevar más tiempo para falsificar una firma, especialmente si está tratando de imitar cuidadosamente cada detalle.

6.6.1. Transformación de la base de datos

El primer objetivo es lograr que el brazo robótico pueda recrear una firma. Este hito ha sido alcanzado por Rubén, quien ha desarrollado un software mediante el cual el brazo robot recibe las variables X e Y de una firma y utiliza esta información para replicar la firma recibida. Este proceso ha sido validado mediante pruebas en las cuales el brazo robótico sujeta un rotulador y firma sobre papel, logrando recrear la firma de manera aparentemente precisa.

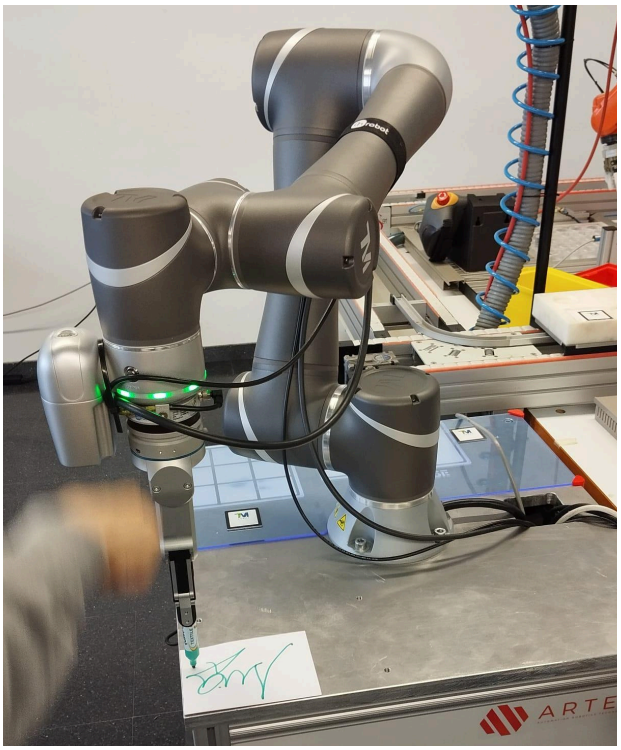


Fig. 6.6.1.1. Brazo robótico sujetando un rotulador para escribir sobre papel

El primer problema que se encuentra es la comunicación con el brazo robótico, la cual se realiza a través de código escrito en Python. Hasta el momento, no se ha encontrado una solución para leer los archivos .fpg que contienen las firmas del MCYT desde Python. Como alternativa, se propone convertir cada firma a un formato de archivo más simple, como .csv, que pueda ser procesado desde Python.

Para llevar a cabo esta conversión de archivos, se recurre a crear un script de Matlab llamado firmasFpgToCsv.m, se utiliza Matlab debido a su capacidad para manejar grandes cantidades de datos de manera eficiente. Además, esta etapa se aprovecha para realizar modificaciones en las firmas, agregándoles la posibilidad de, mediante la simple modificación de una variable, poder añadir marcas de agua a los valores de X e Y.

```
%vect=trasladaFirma(vect,PCX,PCY);              %Centrar la firma en el eje de coordenadas
watermarked_signal=addWatermark(vect,alpha);    %Añadir la marca de agua con el valor de alpha adecuado
watermarked_signalZ=addZvalue( horzcat(watermarked_signal(:,1), watermarked_signal(:,2), vect(:,3)));
%Modificacion a la presion para el uso del brazo robótico

savingname=strrep(savingname, unidad, "");                    %Indicar la carpeta destino
savingname=strrep(savingname, originalformat, savingformat);    %Indicar el formato destino

% Crear una nueva fila con las etiquetas "x", "y" y "z"
nueva_fila = {'x', 'y', 'z'};

% Asegurarse de que tenga el mismo número de columnas que nueva_fila
num_columnas = size(nueva_fila, 2); % Obtener el número de columnas de nueva_fila
watermarked_signalZ = watermarked_signalZ(:, 1:num_columnas);

% Insertar la nueva fila al principio de los datos
datos_con_etiquetas = [nueva_fila; num2cell(watermarked_signalZ)];

% Convertir los datos de celdas en un objeto de tabla
tabla = cell2table(datos_con_etiquetas(2:end, :), 'VariableNames', datos_con_etiquetas(1, :));

% Obtener la carpeta del directorio
[carpetafirmante, ~, ~] = fileparts(savingname);%Crear la carpeta del firmante si no existe ya
if ~isfolder(fullfile(destino, carpetafirmante))
     mkdir(fullfile(destino, carpetafirmante));
end

writetable(tabla, fullfile(destino, savingname)); %Guardar cada uno en su respectiva carpeta
```

Fig. 6.6.1.2. Código para transformar toda la base de datos de ftg a csv

El proceso a seguir implica la lectura de las firmas para posteriormente aplicarles la marca de agua. Esta marca de agua se aplica con un coeficiente alfa de 3, aunque se ha observado que para las firmas del MCYT bastaría con un valor de 2.52. Sin embargo, se ha tomado la decisión de redondear este valor al entero más cercano.

Un aspecto crucial a considerar sobre el funcionamiento del brazo robótico es su incapacidad para ajustar adecuadamente la presión durante la firma. Dado que el robot opera en alturas, la presión ejercida sobre la tableta al recolectar las firmas sería traducida directamente a una altura.

Esta disparidad entre la presión ejercida en la tableta y la altura del brazo robótico genera la necesidad de modificar los datos. En lugar de utilizar la columna de presión, se empleará un valor z. Inicialmente se contempló que este valor z tendría tres componentes, los cuales representan tres situaciones distintas: cuando el lápiz está en contacto con la tableta, cuando no está en contacto pero sigue siendo detectado y cuando está lo suficientemente alejado de la superficie de la tableta como para no ser detectado.

```
function watermarked_signalZ=addZvalue(vect)
    % Obtener la columna P del vector
    columnaP = vect(:, 3);

    % Aplicar las modificaciones
    for i = 1:length(columnaP)
        if columnaP(i) >= 500       %Si hay mucha presión
            columnaP(i) = 0;
        elseif columnaP(i) <= 0     %No esta tocando la tableta
            columnaP(i) = 100;
        else                        %No toca la tableta pero lo detecta
            columnaP(i) = 10;
        end
    end

    % Actualizar la última columna del vector
    watermarked_signalZ = vect;
    watermarked_signalZ(:, 3) = columnaP;

end
```

Fig. 6.6.1.3. Código para modificar la presión por el eje Z

Se ha pensado que la forma más fácil es modificar los valores anteriores por unos nuevos que sí que tengan sentido para el brazo robótico.

Al final se ha visto que el estado en el cual está lejos de de la tableta, es obviado por el registro, simplemente haciendo que no aparezca en la lectura de la firma. Esto hace que el valor Z se convierta en un valor binario, que indica si se toca la tableta o no. Cada uno de estos valores se traduce a una altura distinta para el robot cuando firma.

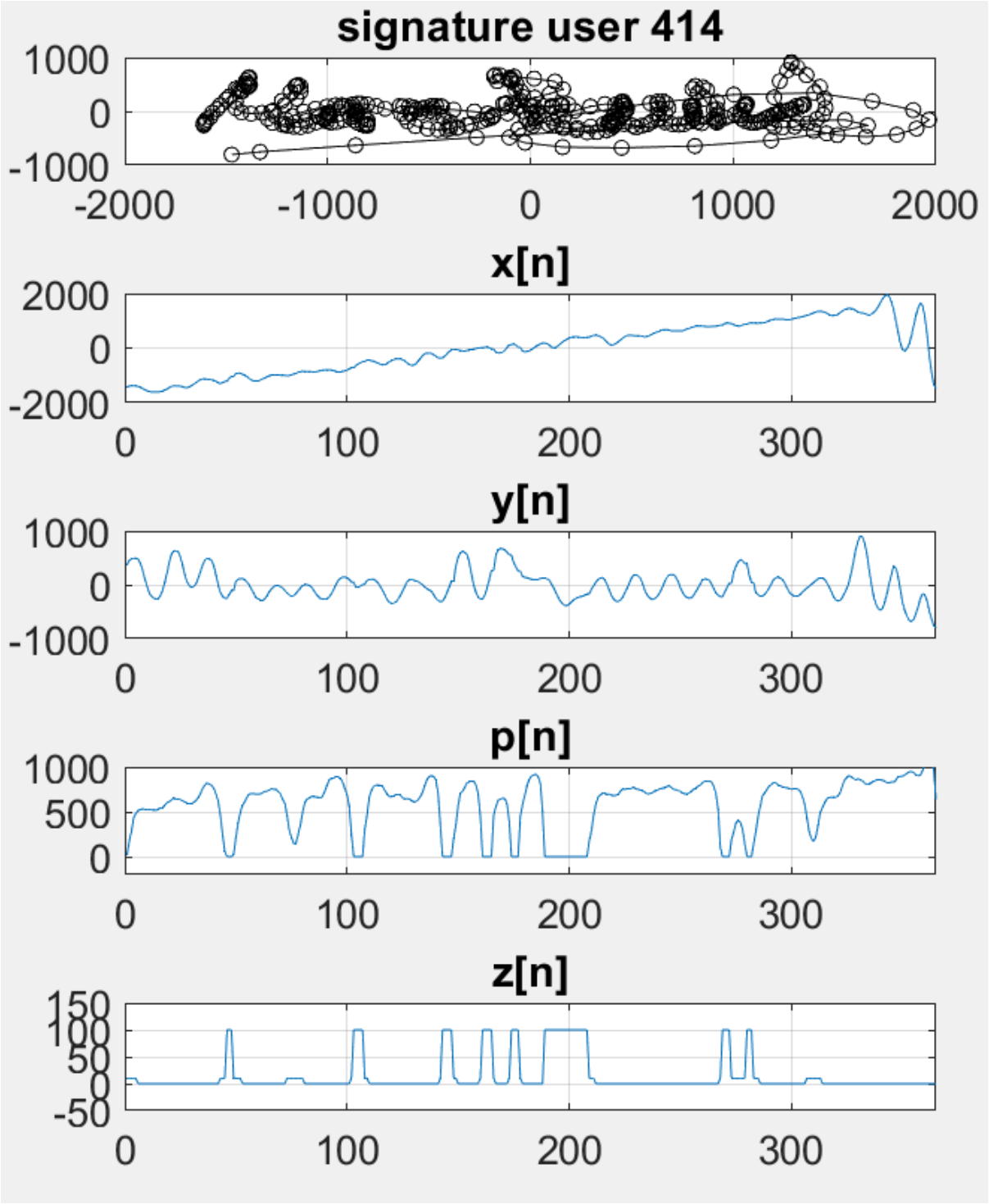


Fig. 6.6.1.4. Representación del cambio de la presión a eje Z

Los siguientes pasos son crear una tabla con esta matriz resultante, y guardarla en una carpeta destino con su mismo nombre pero en formato .csv para que se pueda leer desde Python. Este proceso se puede aplicar a toda la base de datos de firmas.

6.6.2. Lectura de los datos del brazo robótico

La tableta gráfica intuos Wacom tiene la función de registrar las firmas realizadas por el brazo robot, el software utilizado es una tableta que funciona igual que la utilizada para recoger las firmas en la base de datos de MCYT originalmente.
Rubén encontró la problemática que cuando el robot recrea estas firmas, no hay nadie para indicar cuando acaba una y empieza la siguiente. Se ha tenido que hacer un script de matlab el cual tiene como función separar el output de la tableta gráfica en las firmas representadas, para así que estas puedan ser comparadas con las originales.

Fig. 6.6.2.1 Tableta gráfica utilizada por el brazo robot para la escritura de firmas

x	y	t	p	azi	alt	p
6496	9601	17838051	1	1130	390	46
6496	9600	17838058	1	1130	390	98
6496	9600	17838066	1	1130	390	146
6496	9600	17838073	1	1130	390	196
6496	9599	17838081	1	1130	390	250
6497	9599	17838088	1	1130	390	302
6497	9598	17838096	1	1130	390	354
6497	9598	17838103	1	1130	390	398
6497	9598	17838111	1	1130	390	446
6497	9598	17838118	1	1130	390	500

Fig. 6.6.2.2 Muestra de los datos recogidos por la tableta

El archivo tiene valores de las posiciones X e Y del lápiz sobre la tableta, el tiempo en el cual se registra, calculado en milisegundos, un valor booleano que indica si el lápiz toca la tableta o no, las inclinaciones del lápiz y la presión sobre la tableta.

El script creado para realizar esta tarea se llama SignatureSplitter.m. Su primer paso consiste en la lectura del archivo, donde cada línea generalmente está compuesta por las siete variables tal como se presentan en cada instancia. Esta estructura facilita la carga de información en una matriz, lo que permite construir fácilmente una matriz de grandes

dimensiones que luego puede ser separada. En el caso de que alguna línea no cumpla con este requisito, será ignorada durante la lectura de la información.

```
% Leer el archivo recibido línea por línea
fid = fopen(archivo, 'r');
lineas = textscan(fid, '%s', 'Delimiter', '\n');
fclose(fid);

% Convertir las líneas a una matriz de celdas
lineas = lineas{1};
datos = [];

% Extraer los valores del texto
for i = 1:numel(lineas)
    % Dividir la línea en tokens usando espacios en blanco como delimitador
    tokens = strsplit(lineas{i});

    % Verificar si la línea tiene el número esperado de valores
    if numel(tokens) == 8
        valores_numericos = str2double(tokens(1:7)); % Tomar solo los primeros 7 valores
        datos = [datos; valores_numericos];
    else
        disp(['La línea ', num2str(i), ' no cumple con los requisitos y será ignorada']);
    end
end
```

Fig. 6.6.2.3. Código que lee las líneas del documento y las almacena en una matriz

Después de cargar los datos en una única matriz, el proceso a seguir es relativamente simple. Se lee cada momento registrado y se compara con el momento anterior. Si ha transcurrido suficiente tiempo (por defecto son 3 segundos) desde el último registro, se infiere que ha ocurrido un cambio de firma. Interesa que este tiempo de espera entre firmas sea el más bajo posible para minimizar el tiempo necesario en mandar la BBDD al robot.

Cada línea se lee y almacena como parte de una firma. Cuando se detecta un cambio, todo lo almacenado hasta ese punto se considera como una firma completa.
Además, se ha solicitado eliminar los datos redundantes, ya que al finalizar la firma, en muchas ocasiones el brazo robótico permanece en una posición fija hasta que el lápiz es levantado, lo que genera varios fragmentos donde no hay movimiento.

```
% Iterar sobre cada fila de la matriz
for i = 1:size(datos, 1)
    % Registrar el tiempo
    newTime=datos(i, time);

    if i==1                                     % Guardar al principio el tiempo con el que empieza
        lastTime=datos(i, time);
    end;

    if abs(newTime-lastTime) > timeToSkip                   % Si se supera el tiempo se guarda el nuevo firmante
        firmantes{firmante} = segmentoActual;               % Agregar el fragmento guardado como un nuevo firmante
        segmentoActual={};                                  % Vaciar el segmento actual

        while ~ismember(firmante+1, listafirmantes)         % Pasar al siguiente firmante
            firmante = firmante + 1;
        end
        firmante = firmante+1;
    end;

    %Añadir cada firma al nuevo segmento, siempre y cuando no sean iguales que la linea anterior
    if i~=1 && datos(i-1, XValue)~=datos(i, XValue) && datos(i-1, YValue)~=datos(i, YValue)
        segmentoActual=[segmentoActual; datos(i, :)];
    end;

    lastTime=newTime;
end
% Agregar el último fragmento después del último punto de división
firmantes{firmante} = segmentoActual;
```

Fig. 6.6.2.4. Código que separa la matriz en distintas firmas

El siguiente paso implica almacenar la información para su posterior comparación. Se ha adoptado un formato similar al del script anterior para facilitar la comparación directa.

En primer lugar, se crean las carpetas necesarias para cada usuario. El primer usuario de cada documento se especifica al inicio de la ejecución, lo que permite evitar la necesidad de un solo archivo que contenga todas las firmas, permitiendo su separación en varias lecturas.

Cada carpeta se completa con la firma correspondiente a su respectivo usuario, almacenando únicamente los valores de X, Y y Z, donde Z representa la adaptación de la presión utilizada por el brazo robótico.

Aunque el proceso podría volverse más complejo si se requiere más de una firma por usuario, se ha optado por leer solo una firma por usuario, considerando la velocidad del robot al escribir.

```
%Guardar la firma en un archivo nuevo
for i = 1:numel(firmantes)
    if ~isempty(firmantes{i})
        % Obtener el número de la carpeta correspondiente
        num_carpeta = i;
        carpeta_numero = sprintf('%04d', num_carpeta - 1);

        % Construir la ruta completa del archivo
        nombre_archivo = [num2str(carpeta_numero),'v06', '.csv'];
        ruta_archivo = fullfile(destino, carpeta_numero, nombre_archivo);

        % Obtener la firma actual
        firma_actual = firmantes{i};

        % Inicializar una matriz para almacenar las columnas seleccionadas
        firma_actual_seleccionada = [];

        % Iterar sobre las filas de la firma actual
        for j = 1:size(firma_actual, 1)
            % Seleccionar las columnas 1, 2 y 7 de cada fila y concatenarlas
            fila_seleccionada = firma_actual{j}(:, [1, 2, 7]);
            firma_actual_seleccionada = [firma_actual_seleccionada; fila_seleccionada];
        end
        firma_actual_seleccionadaZ=addZvalue( horzcat(firma_actual_seleccionada(:,1), ...
            firma_actual_seleccionada(:,2), firma_actual_seleccionada(:,3)));
```

Fig. 6.6.2.5. Código que guarda las distintas firmas en documentos .csv

MCYT se adquirió con un software que proporcionaba formato .fpg y actualmente la tableta gráfica usa otro software, diseñado por Josep Roure que proporciona un formato distinto, llamado svc. El archivo que se recibe desde la tableta, que contiene las firmas, es .svc, por lo cual se ha decidido que se guarde también en este formato cada firma individual.

6.6.3. Comparativa de los resultados obtenidos

El brazo robot utilizado es uno comunitario, por lo cual está sujeto a ciertas medidas de seguridad, como por ejemplo una limitación a su velocidad. Debido a esta limitación, no se ha podido hacer una copia exacta de todas las firmas de la base de datos de MCYT, en su lugar se ha recreado solo una firma de cada firmante, para un total de 330 firmas, estas serán comparadas con las originales utilizando DTW, con este método se podrá comprobar si estas son capaces de engañar al sistema usado previamente.

Se pretende comparar las firmas creadas por personas en contraposición con las copias realizadas por el brazo robot. La forma de comprobar su similitud es mediante DTW generar un gráfico que muestra sus diferencias. Este gráfico muestra una de las firmas a comparar como líneas de referencia en un eje diagonal y luego posiciona la otra como otro eje sobre la anterior, mostrando cómo se alinean.

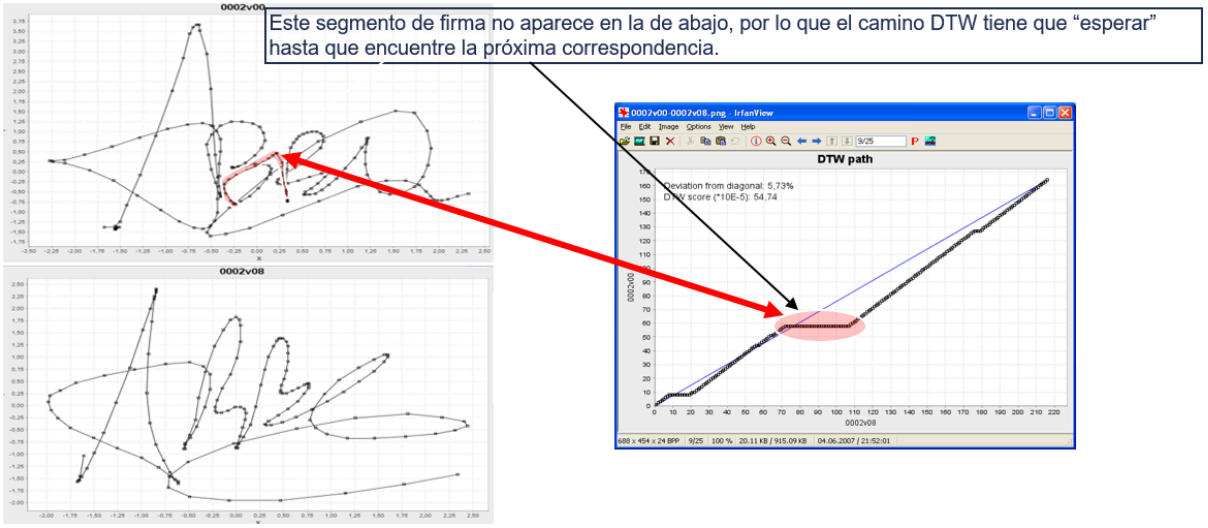


Fig. 6.6.3.1. Muestra del uso de los gráficos con DTW para la comparación de firmas [26]

El primer paso sería comprobar cada firma cómo se compara con la firma original a la cual está copiando, esto se realiza en el script DTWRobotComparator.m.

Las firmas que se han pasado por el brazo robótico son el número seis de cada firmante. Lo primero que se quiere comprobar es cómo se compara cada firma con la versión copiada de sí misma. Se ha hecho la comparativa por DTW a 31 firmas de firmantes distintos y tras ello se ha obtenido un porcentaje de identificación, tal y como sucede en el primer script. En la siguiente imagen se muestra el resultado de esta prueba.

```
Número de aciertos: 15
Número de fallos: 16
DTW Ident.(%)=48.3871
```

Fig. 6.6.3.2. Resultado de la comparativa por DTW entre firmas humanas v6 con robóticas v6

El resultado obtenido tiene cerca de la mitad de aciertos, lo cual es curioso, pues se está comparando una firma con una copia de sí misma.

Lo sucedido es que el robot no ha tenido una pieza que facilite la sujeción del lápiz de la tableta, con lo cual este se iba moviendo ligeramente cuando no tocaba, las copias son muy similares visualmente, sin embargo al usar DTW para ver la comparativa se notan estas ligeras diferencias agravadas, también se puede observar como varían X o Y a lo largo de la firma.

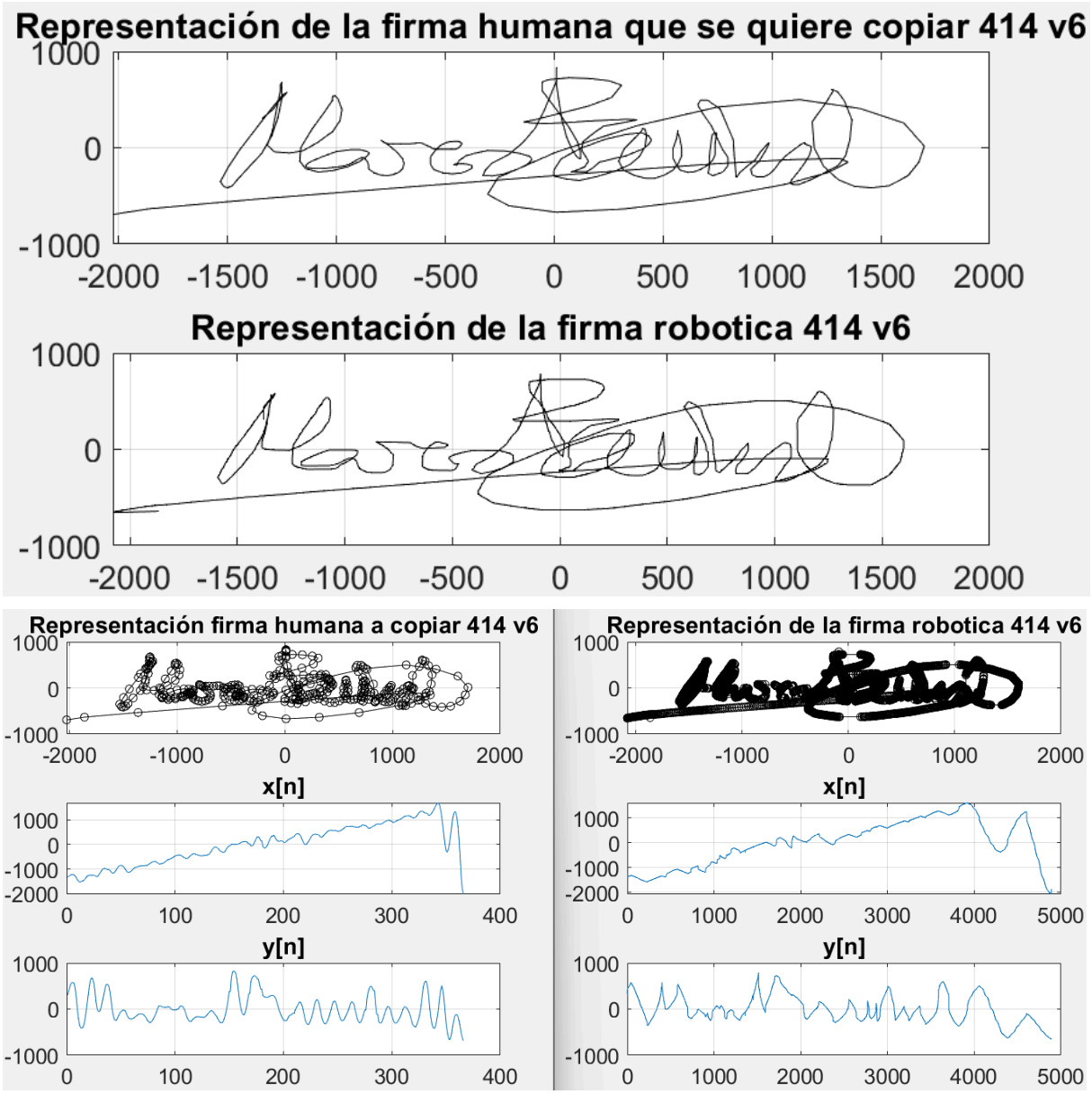


Fig. 6.6.3.3. Diferencias entre firmas humanas v6 con robóticas v6 y su representación gráfica

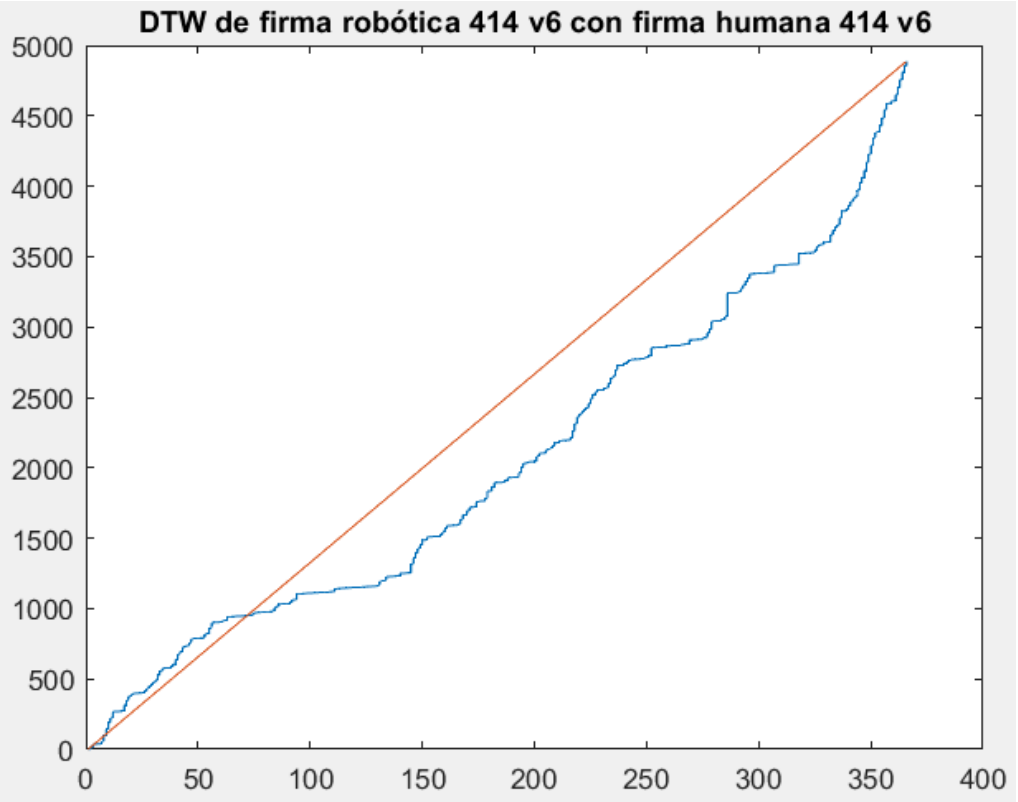


Fig. 6.6.3.4. Comparativa por DTW entre firmas humanas v6 con robóticas v6 y su representación gráfica

Cabe señalar la diferencia entre las velocidades de ambas firmas, al colocar los ejes X e Y de cada firma, uno sobre el otro, se aprecia que la firma hecha mediante el robot tarda casi 8 veces más que la hecha por el humano.

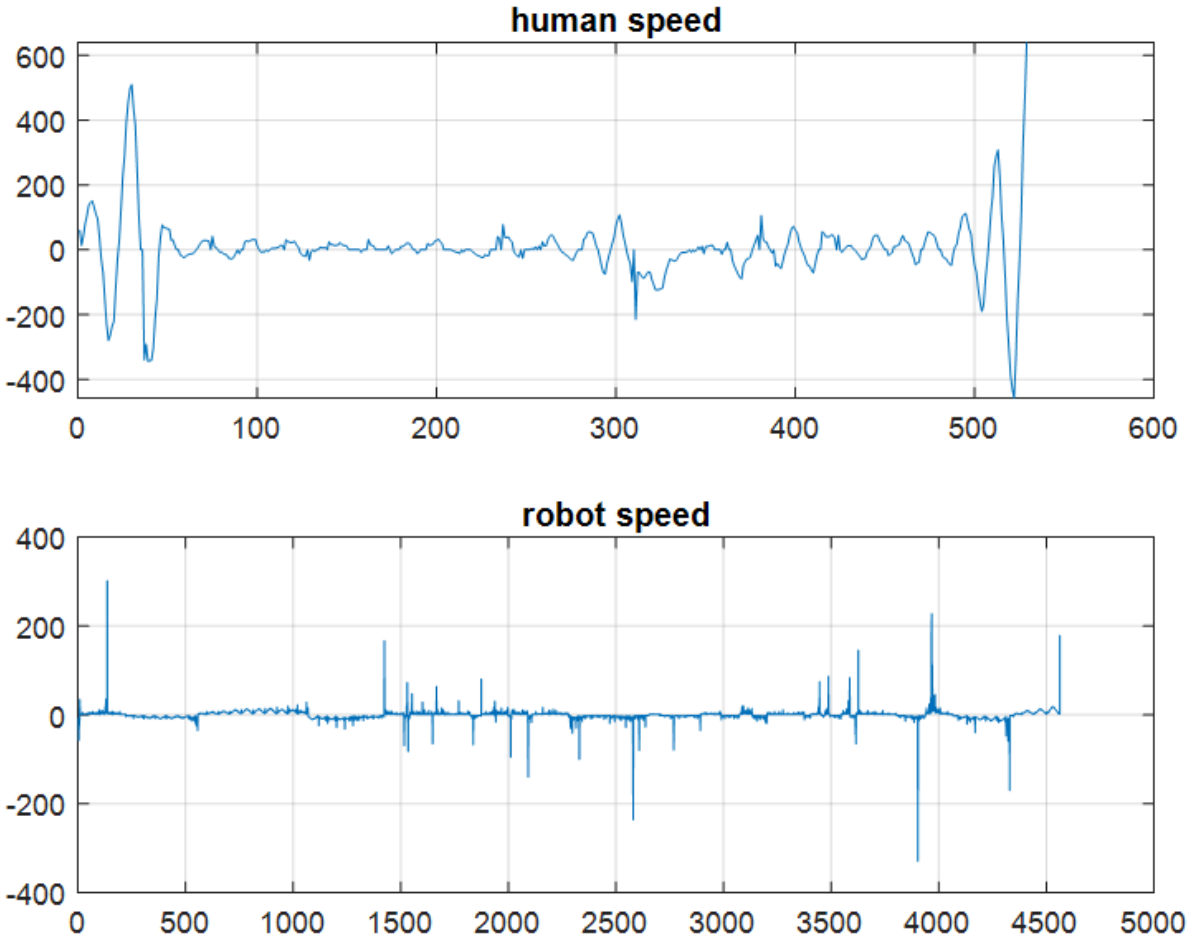


Fig. 6.6.3.5. Comparativa de tiempo empleado por el humano en comparación por el empleado por el robot

También al colocar estos ejes juntos se aprecia la diferencia en el tiempo, y si se alinean ambas señales para alargar la humana, al tener cada eje junto a su copia, se observa como la copia no es tan buena.

Esto da lugar a la conclusión que este robot no es demasiado bueno imitando las firmas de una persona.

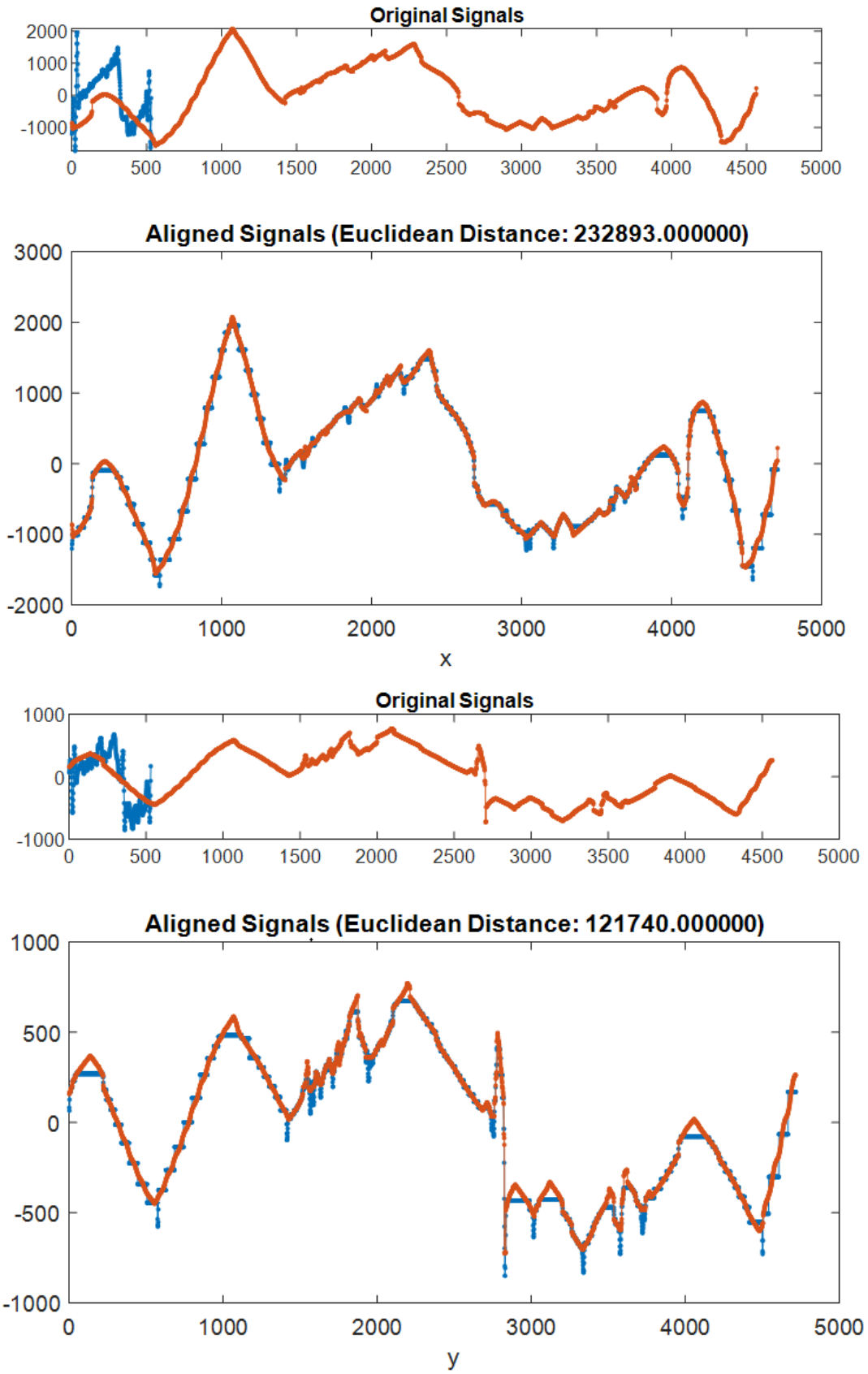


Fig. 6.6.3.6. Comparativa de X e Y de la firma humana con la robot

La idea del DTW en el primer script es localizar cuando una firma pertenece a una misma persona, aún cuando está analizando firmas distintas.
En el caso anterior se está comparando una firma consigo misma, pero también se puede comparar las firmas realizadas por el robot (número 6), con otras firmas distintas del mismo firmante que no han pasado por el robot (las firmas 1).

```
Número de aciertos: 10
Número de fallos: 21
DTW Ident.(%)=32.2581
```

Fig. 6.6.3.7. Resultado de la comparativa por DTW entre firmas humanas v1 con robóticas v6

Es de esperar que si el anterior resultado ha mostrado ya cerca de un 50% de identificación, el porcentaje decaiga en este análisis, pues se está comprobando sobre firmas distintas, ciertamente tienen similitudes, pero menos que una comparación entre dos firmas que originalmente eran la misma.

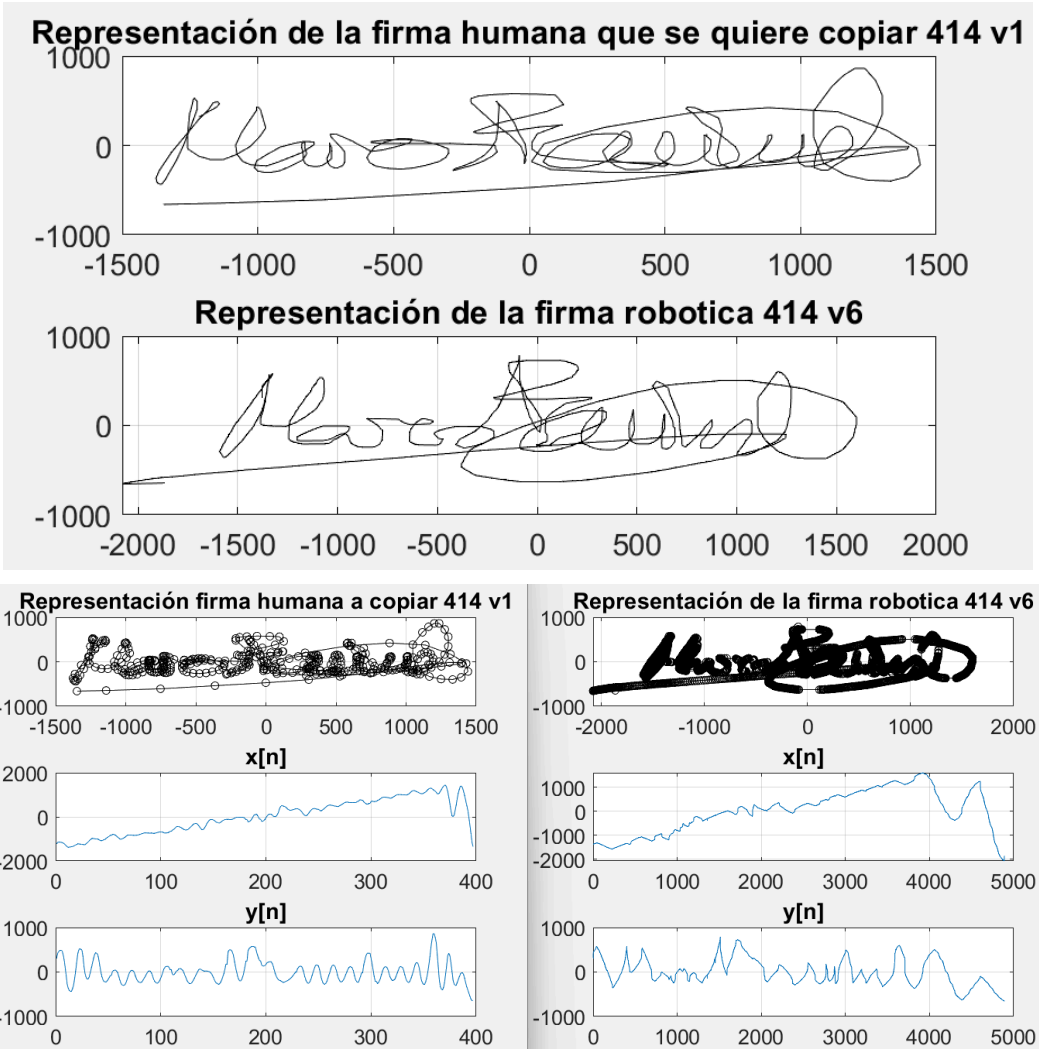


Fig. 6.6.3.8. Diferencias entre firmas humanas v1 con robóticas v6 y su representación gráfica

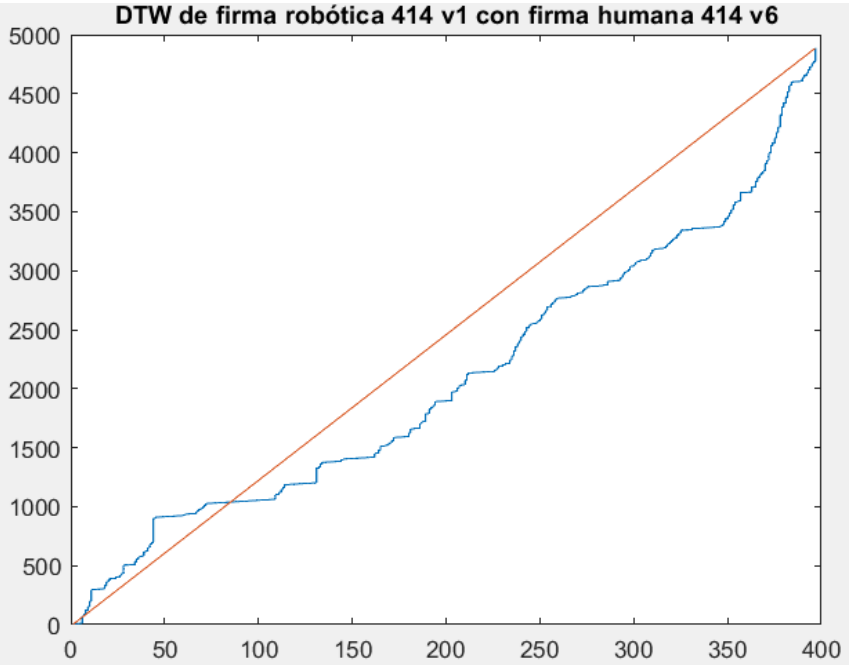


Fig. 6.6.3.9. Comparativa por DTW entre firmas humanas v1 con robóticas v6 y su representación gráfica

Adicionalmente, se le ha dado al robot ciertas versiones de una misma firma, con marcas de agua aplicadas con distintas alphas, para poder comprobar cómo de buena es la extracción de la marca de agua aplicada sobre esta copia. Se han generado estas firmas usando el script ConclusionGenerator.m, ahí se ha generado 5 archivos csv con la firma 414 pero con alphas de 3, 30,60, 100 y 300 de alpha, y se ha guardado los datos de la edición de estas firmas en el workspace para poder comparar posteriormente los resultados.

```
for alpha=listaAlpha
    carpeta_numero = sprintf('%04d', firma);
    vect=leer_firmaFPG([directorioOriginales,carpeta_numero,'\',carpeta_numero,'v',firmafile,'.fpg']);
    vect=trasladaFirma(vect,PCX,PCY);
    [watermarked_signal,watermark1,watermark2]=addWatermark(vect,alpha);
    data{indice,1}=vect;
    data{indice,2}=watermark1;
    data{indice,3}=watermark2;
    data{indice,4}=[watermarked_signal(:, 1:2), vect(:, 3)];
    data{indice,5}=alpha;

    %Guardado en csv
    carpeta_numero = sprintf('%04d', firma);
    nombre_archivo = [num2str(carpeta_numero),'a',num2str(sprintf('%03d', alpha)), '.csv'];
    ruta_archivo = fullfile(output, nombre_archivo);

    writematrix([watermarked_signal(:, 1:2), vect(:, 3)], ruta_archivo);

    disp(['Firma guardada en ', ruta_archivo]);
```

Fig. 6.6.3.10. Registro de los datos de las firmas enviadas al robot

Al obtener las versiones del robot se ha intentado extraer la marca de agua y compararla con la que se injertó al inicio, el resultado obtenido muestra que no se ha sido capaz de extraer la marca de agua perfectamente.

```
Alpha=3 erroneous bits=7159, % erroneous bits=73.0958%
Alpha=30 erroneous bits=1218, % erroneous bits=11.282%
Alpha=60 erroneous bits=692, % erroneous bits=5.9512%
Alpha=100 erroneous bits=484, % erroneous bits=3.5567%
Alpha=300 erroneous bits=482, % erroneous bits=2.0467%
```

Fig. 6.6.3.11. El % de identificación a lo largo de distintas alphas

En las alphas más pequeñas se ha obtenido una recuperación muy baja de la marca de agua, sin embargo en cuanto aumenta el alpha si que se puede obtener mejor esta misma. Pero como ya se ha mostrado anteriormente, el aumentar el alpha deforma la firma, generando cada vez menos similitud con la firma original.

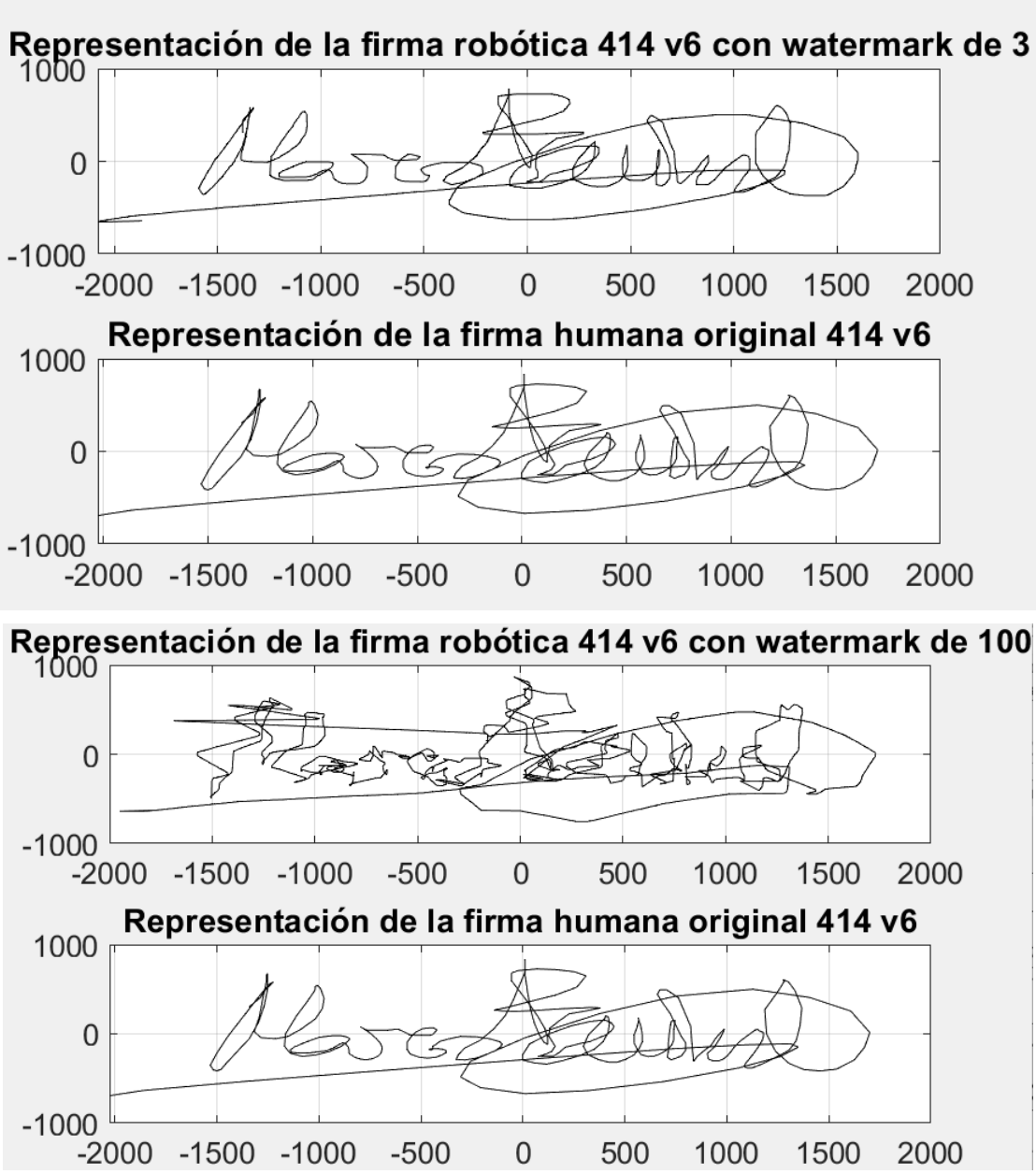


Fig. 6.6.3.12. Muestra de la deformación de la firma en las alphas 3 y 100

Este resultado muestra que hay que utilizar una marca de agua con suficiente fuerza como para que en posteriores intentos de copia se pueda recuperar correctamente.
No hay que utilizar el valor de alpha que está inmediatamente por encima del valor mínimo, pues la marca de agua puede perderse en las copias, tal y como ha ocurrido en este caso con la firma con marca de agua de alpha 3.

En el caso de una marca de agua demasiado fuerte, aunque permite mejor recuperación de la marca de agua, esta deforma la firma, como queda visto en la firma con marca de agua de alpha 100.

7. Conclusiones

Las marcas de agua son una excelente forma de almacenar información en cualquier documento escrito y las firmas son un método eficaz para demostrar la veracidad de un documento; por ello, es importante protegerlas, y el uso de marcas de agua es un método idóneo para lograrlo.
Esta investigación ha sido posible gracias a la base de datos de firmas de MCYT [21]. En estas firmas se han probado diversas cualidades de la marca de agua, utilizando una secuencia binaria como marca de agua e incrustándola en las firmas. El objetivo es determinar con qué fuerza, representada por el valor alpha, se obtiene el resultado óptimo.

- El valor mínimo de alpha depende de cada firma; de las 330 firmas revisadas, la media se encontraba en un valor aproximado de X, con el valor más alto siendo 2,51. Este valor mínimo se obtiene al incrustar la marca de agua y determinar con qué valor de alpha es posible extraerla y recuperar el mismo resultado. En caso de tratar de realizar una copia de la firma, el valor mínimo obtenido puede no ser capaz de traspasarse a la copia, tal y como se ha mostrado mediante las copias realizadas por brazo robótico.

- El valor máximo, sin embargo, no es tan simple de calcular, pues al aumentar la fuerza de la marca de agua, la firma experimenta una deformación. Se ha probado añadir distintas marcas de agua y comparar cuándo disminuye el reconocimiento de la firma mediante DTW. El resultado muestra que cuanto más se incrementa el valor de alpha, peor es el reconocimiento, por lo que no debería aumentar en grandes cantidades.

- Conviene encontrar el valor de alpha que permita mantener el equilibrio entre no deformar la forma de la firma y ser capaz de conservar la información de la marca de agua para su posterior recuperación.

8. Bibliografía

[1] Cambridge Dictionary, "Signature," Cambridge Dictionary, [Online]. Available: https://dictionary.cambridge.org/dictionary/english/signature . Accessed: Feb. 29, 2024.

[2] Legalesign, "History of Signatures," Legalesign Blog, [Online]. Available: https://legalesign.com/blog/history-of-signatures/ . Accessed: Feb. 29, 2024.

[3] Wikipedia, "Statute of Frauds" Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Statute_of_Frauds. Accessed: Mar. 6, 2024.

[4] Wikipedia, "Kushim (Uruk period)," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Kushim_%28Uruk_period%29 . Accessed: Feb. 29, 2024.

[5] S. K. Sharma, "Digital Signature: Pros & Cons of Digital Signature," TaxGuru, [Online]. Available: https://taxguru.in/corporate-law/digital-signature-pros-cons-digital-signature.html#What_are_the_Cons_of_Using_Digital_Signatures . Accessed: Feb. 29, 2024.

[6] M. Faundez-Zanuy, J. Fierrez, M. A. Ferrer, M. Diaz, R. Tolosana, y R. Plamondon, "Handwriting biometrics: Applications and future trends in e-security and e-health," Cognitive Computation, vol. 12, pp. 940-953, 2020.

[7] J. Chapran, "Biometric writer identification: feature analysis and classification," Int J Pattern Recognit Artif Intell, vol. 20, pp. 483–503, 2006.

[8] E. Sesa-Nogueras and M. Faundez-Zanuy, "Biometric recognition using online uppercase handwritten text," Pattern Recogn, vol. 45, pp. 128–144, 2012.

[9] M. Parizeau and R. Plamondon, "What types of scripts can be used for personal identity verification?," in Computer Recognition and Human Production of Handwriting, R. Plamondon, C. Y. Suen, and M. Simner, Eds. 1989, pp. 77–90.

[10] M. A. Ferrer, A. Morales, J. F. Vargas, I. Lemos, y M. Quintero, "Is it possible to automatically identify who has forged my signature? Approaching to the identification of a static signature forger," in Proc, 10th IAPR International Workshop on Document Analysis Systems, 2012, pp. 175–179.

[11] M. Faundez-Zanuy y J. Mekyska, "Privacy of online handwriting biometrics related to biomedical analysis," en C. Vielhauer, Ed., User-Centric Privacy and Security in Biometrics, pp. 17-39, IET, 2017.

[12] M. Faundez-Zanuy, "Nuevas tecnologías y protección de los derechos de la propiedad intelectual," Mundo electrónico, pp. 38-47, 2000. Disponible online en https://www.researchgate.net/publication/255947780_Nuevas_tecnologias_y_proteccion_de_los_derechos_de_la_propiedad_intelectual

[13] F. Hartung y M. Kutter, "Multimedia watermarking techniques," Proceedings of the IEEE, vol. 87, no. 7, pp. 1079-1107, 1999.

[14] M. Begum y M. S. Uddin, "Digital image watermarking techniques: a review," Information, vol. 11, no. 2, p. 110, 2020.

[15] C.S. Hsu y S.F. Tu, "Digital Watermarking Scheme for Copyright Protection and Tampering Detection," Int. J. Inf. Technol. Secur., vol. 11, pp. 107–119, 2019.

[16] MathWorks, "Discrete Cosine Transform," MathWorks, [Online]. Available: https://es.mathworks.com/help/images/discrete-cosine-transform.html . Accessed: Mar. 7, 2024.

[17] MathWorks, "Discrete Fourier Transform," MathWorks, [Online]. Available: https://es.mathworks.com/help/signal/ug/discrete-fourier-transform.html . Accessed: Mar. 7, 2024.

[18] Wikipedia, "Discrete Wavelet Transform," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/Discrete_wavelet_transform . Accessed: Mar. 7, 2024.

[19] MathWorks, "Singular Values," MathWorks, [Online]. Available: https://es.mathworks.com/help/matlab/math/singular-values.html . Accessed: Mar. 7, 2024.

[20] M. A. Awad, "A comparison between agile and traditional software development methodologies," University of Western Australia, 30, pp. 1-69, 2005.

[21] J. Ortega-Garcia, J. Fierrez-Aguilar, D. Simon, J. Gonzalez, M. Faundez-Zanuy, V. Espinosa, et al., "MCYT baseline corpus: a bimodal biometric database," IEE Proceedings-Vision, Image and Signal Processing, vol. 150, no. 6, pp. 395-401, 2003.

[22] M. Faundez-Zanuy, M. Diaz," On the Use of First and Second Derivative Approximations for Biometric Online Signature Recognition". In: Rojas, I., Joya, G., Catala, A. (eds) Advances in Computational Intelligence. IWANN 2023. Lecture Notes in Computer Science, vol 14134. Springer, Cham. https://doi.org/10.1007/978-3-031-43085-5_36

[23] Marcos Faundez-Zanuy, "On-line signature recognition based on VQ-DTW", Pattern Recognition, Volume 40, Issue 3, 2007, Pages 981-992, ISSN 0031-3203, https://doi.org/10.1016/j.patcog.2006.06.007.

[24] MathWorks, "DTW," MathWorks, [Online]. Available: https://es.mathworks.com/help/signal/ref/dtw.html . Accessed: Mar. 7, 2024.

[25] Bayometric, "Tasa de falsa aceptación (TFA) y tasa de falso reconocimiento (TFR) en la biometría.", Bayometric, [Online]. Available: https://www.bayometric.com.mx/tasa-de-falsa-aceptacion-tfa-y-tasa-de-falso-de-reconocimiento-tfr-en-la-biometria/ . Accessed: Mar. 13, 2024.

[26] M. Faundez-Zanuy, "Reconocimiento de firmas manuscritas", Apuntes de la asignatura de biometría. Escuela Universitaria Politécnica de Mataró (Barcelona UPC), 2024.