Sistema de Reconocimiento de Gestos de la Mano basado en Acelerómetro para Interacción en TV Digital

J. Ducloux[1,2,3], P. Colla[2], P. Petrashin[1], W. Lancioni[1], L. Toledo[1]
[1]Facultad de Ingeniería, Universidad Católica de Córdoba
[2]Especialización en Sistemas Embebidos, Instituto Universitario Aeronáutico
[3]Consorcio Córdoba TDT
Córdoba, Argentina

***Resumen*—Este trabajo presenta el diseño y la implementación de un sistema de reconocimiento de gestos de la mano utilizando acelerómetro de 3 ejes. Este sistema será embebido dentro de un control remoto moderno destinado a mejorar la interacción hombre-máquina en el contexto de la TV digital de Argentina. Debido a que el reconocimiento de gestos de la mano es un problema de clasificación de patrones, dos técnicas basadas en redes neuronales artificiales (ANNs) son exploradas: perceptrón multicapa y máquina de soporte vectorial. Ambos tipos de ANNs son utilizadas con la finalidad de comparar resultados y seleccionar la herramienta que mejor se adapte al problema en cuestión. Conjuntamente, técnicas de procesamiento digital de señales son usadas para el preprocesamiento y adaptación de las señales de entrada a los modelos de reconocimiento de patrones. Un vocabulario gestual de 8 tipos de gestos es utilizado, el cual también fue usado por otros trabajos similares con el propósito de comparar resultados. Puesto que la solución será implementada dentro de un sistema embebido, una adecuada relación de compromiso entre precisión de reconocimiento del clasificador y utilización de recursos de la plataforma de hardware es requerida. Los resultados obtenidos de precisión y utilización de recursos son más que aceptables.**

Palabras clave—acelerómetro, redes neuronales artificiales (ANNs), TV digital, sistemas embebidos, reconocimiento de gestos de la mano, perceptrón multicapa (MLP), control remoto, máquina de soporte vectorial (SVM)

I. INTRODUCCIÓN

El reconocimiento de gestos se refiere al proceso de entendimiento y clasificación de movimientos significativos realizados por las manos, brazos, cara y, algunas veces, por la cabeza de las personas. Este se ha convertido en un campo muy atractivo de investigación para el diseño de interfaces hombre-máquina dotadas de inteligencia artificial para numerosas aplicaciones como domótica, lenguaje de señas, interfaces para personas discapacitadas, realidad virtual, videojuegos, etc. [1].

El reconocimiento de gestos es un área de investigación que está en pleno auge, tanto para reconocimiento basado en visión como reconocimiento basado en sensores inerciales de tecnología de sistemas micro-electromecánicos (MEMS, micro-electromechanical systems). La utilización de esta última tecnología resulta atractiva puesto que incluye a sensores de bajo costo que no sufren de grandes influencias como los sistemas de reconocimiento basados en visión, como ser los niveles de luz ambiente y el fondo. La mayor parte de la literatura disponible de reconocimiento de gestos de la mano basado tecnología MEMS sólo utiliza acelerómetros de 3 ejes, con buenos resultados en la precisión obtenida.

La mayoría de las técnicas y algoritmos utilizados en sistemas de reconocimiento de gestos de la mano basados en acelerómetros de las publicaciones actuales [2]-[6] han sido implementados en sistemas de amplios recursos, como notebooks o computadoras de escritorio, con procesadores veloces y suficiente memoria disponible. Los más ligeros algoritmos fueron implementados en smartphones, pero optimizaciones de código debieron ser realizadas, como así también modificaciones en los propios algoritmos para disminuir la carga computacional y la utilización de recursos [7].

En este trabajo, los movimientos de la mano en el espacio libre que describen alguna forma previamente definida manipulando un dispositivo de interacción como un control remoto se denominarán gestos. Los gestos son representados por vectores que contienen las variaciones de los niveles de aceleración en función del tiempo en el espacio tridimensional. Series temporales como señales de aceleración serán términos igualmente válidos para referirse a los gestos.

Dentro del contexto de la TV digital, los sistemas de televisión han pasado por enormes mejoras en este último tiempo. El surgimiento de la TV digital terrestre y televisores inteligentes permiten incorporar el concepto de interactividad que sumado a los servicios de Internet están produciendo un impacto innovador y un consecuente cambio radical en la experiencia de los usuarios. El control remoto utilizado para controlar estos dispositivos está, en la mayoría de los casos, aún resuelto por el tradicional control remoto infrarrojo, el cual se convirtió en un factor limitante en la interacción del usuario con la TV. En consecuencia, diferentes tipos de interfaces y nuevos métodos de control deben ser incluidos para mejorar la experiencia del usuario inmerso en esta evolución tecnológica. Algunos de los reconocidos fabricantes de dispositivos de TV y entretenimiento comenzaron a incorporar nuevas interfaces de usuario en sus productos de tope de gama, como control por voz y por imágenes, aunque la gran mayoría de los controles remotos utilizados en TV siguen siendo los que ofrecen una funcionalidad básica.

Este artículo se presenta como un informe preliminar correspondiente al Proyecto FSTICS N° 4, financiado por la Agencia Nacional de Promoción Científica y Tecnológica a través del FONARSEC, el cual tiene como Institución Beneficiaria a la Universidad Católica de Córdoba, y como Adoptante al Consorcio Córdoba TDT.

Los gestos, y particularmente los gestos de la mano, tienen dos aspectos en sus características de señal que los hacen dificultosos para su reconocimiento. En primer lugar, presentan ambigüedad en la segmentación, es decir, no se conocen los límites de la realización del gesto. En segundo lugar, presentan variabilidad espacio-temporal ya que el gesto varía dinámicamente en forma y duración, incluso para los mismos gestos y mismas personas.

Modelos como el perceptrón multicapa (MLP, Multilayer Perceptron) [8] y la máquina de soporte vectorial (SVM, support vector machine) [9] son tipos de redes neuronales artificiales (ANNs, artificial neural networks) que intentan reproducir el proceso de solución de problemas del cerebro [10]. Así como los humanos aplican el conocimiento obtenido de la experiencia a nuevos problemas o situaciones, una ANN toma como ejemplos problemas resueltos para construir un sistema que toma decisiones y realiza clasificaciones. Estas técnicas son ampliamente utilizadas en robótica, medicina, reconocimiento del habla y minería de datos, por citar algunos campos de aplicación. Los problemas adecuados para la solución neural son aquellos que no tienen una solución computacional precisa, o que requieren de algoritmos muy extensos para implementar la solución. Así, estas técnicas son adecuadas para ser aplicadas en reconocimiento de gestos de la mano.

En este trabajo, un sistema de reconocimiento de gestos de la mano utilizando ANNs es diseñado e implementado, para ser embebido dentro de un control remoto moderno y ser usado en el contexto de la TV digital de Argentina. El sistema propuesto es capaz de reconocer gestos aislados, ser independiente del usuario, trabajar con un vocabulario gestual definido de 8 clases y presentar una excelente tasa de reconocimiento. Los gestos reconocidos serán convertidos en comandos de control que ejecutarán diferentes acciones en los sistemas de TV digital del hogar. El sistema de reconocimiento de gestos será implementado dentro de un sistema embebido basado en microcontrolador. La plataforma de hardware es seleccionada para obtener reducidos tiempos de ejecución de los algoritmos del clasificador para una respuesta de tiempo real, y conseguir una adecuada tasa de reconocimiento para una mejor experiencia de usuario. Un análisis comparativo de dos tipos de ANNs es realizado con la finalidad de encontrar el modelo que provea la mejor relación entre precisión versus tiempo de ejecución. Para entrenar, validar y evaluar los modelos se utiliza una base de datos existente.

El documento está diagramado de la siguiente manera. En la Sección II, la composición y análisis de la base de datos utilizada, y la metodología para entrenar, validar y evaluar los modelos de ANNs son comentados. En la Sección III, la estrategia de diseño y los parámetros de comparación de los modelos de ANNs son establecidos. El diseño, implementación y test del sistema de clasificación son expuestos en la Sección IV. Los resultados obtenidos de precisión y tiempos de ejecución de los algoritmos se muestran y discuten en la Sección V. Por último, los logros, posibles mejoras y trabajo a futuro son comentados en las conclusiones.

II. Base de Datos para Entrenamiento, Validación y Evaluación de Modelos ANN

La base de datos empleada aquí es la base de datos del proyecto uWave [11], [12]. Este proyecto es pionero en reconocimiento de gestos de la mano basado en acelerómetros usando deformación temporal dinámica (DTW, dynamic time warping). El vocabulario gestual de la base de datos está compuesto por 8 tipos gestos de la mano, como puede verse en la Fig. 1.

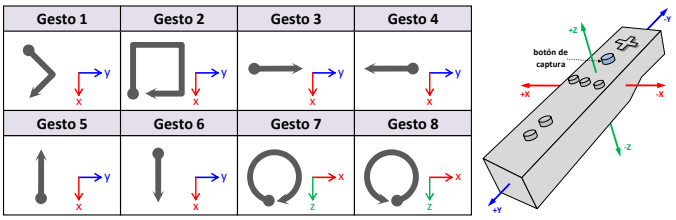

Fig. 1. Vocabulario gestual.

La base de datos consta de 4480 gestos, realizados por 8 usuarios, durante 7 días y 10 repeticiones por día. De esta manera, se tiene una base de datos que contiene los 8 tipos de gestos en cantidades proporcionadas.

Las técnicas para reconocimiento de patrones usadas en este trabajo son dos tipos de ANNs: MLP y SVM. Estos modelos requieren de datos de entrenamiento para aprender y almacenar el conocimiento que luego será utilizado para realizar la tarea de clasificación. Para ello, una base de datos para entrenar, validar y evaluar los modelos es necesaria, para seleccionar el mejor modelo con comportamiento óptimo para el problema. La base de datos fue analizada y procesada utilizando lenguaje R [13], mediante el entorno de desarrollo integrado RStudio [14].

A. *Análisis de la base de datos*

Información valiosa para el diseño fue extraída de la base de datos. Un análisis de datos faltantes y un tratamiento de outliers fueron realizados. La duración de los gestos en cantidades de muestras fue obtenida. Esta última información es utilizada para establecer la duración mínima y máxima de los gestos y determinar el tamaño de los buffers de entrada. Los valores mínimos y máximos de los niveles de aceleración de los gestos son de utilidad para seleccionar el sensor de aceleración en función de su rango dinámico de operación y para normalizar la entrada. La transformada de Fourier discreta fue aplicada para obtener el ancho de banda de las señales de aceleración. Un valor de 7 Hz fue obtenido. Este dato es de utilidad para seleccionar la frecuencia de muestreo del sistema y frecuencia de corte del filtro de entrada.

B. *Partición de la base de datos*

La base de datos es dividida aleatoriamente en dos grupos representativos de todas las observaciones, es decir, que el porcentaje de cada clase es conservado. El 80% de los datos es utilizado para entrenamiento y validación de los modelos. El 20% restante es utilizado para evaluación de los modelos. La técnica de validación cruzada k-fold leave one-out es implementada. La partición para entrenamiento y validación es dividida en k=20 grupos de datos, utilizando k-1 grupos para entrenar y 1 grupo para validar el modelo. Este proceso es repetido k=20 veces, utilizando así todos los datos para validar el modelo en cuestión. Así, 20 modelos validados son obtenidos. Los datos de evaluación son utilizados para evaluar el mejor de los modelos validados para obtener la precisión final de reconocimiento. Cada paso del proceso puede observarse en la Fig. 2.

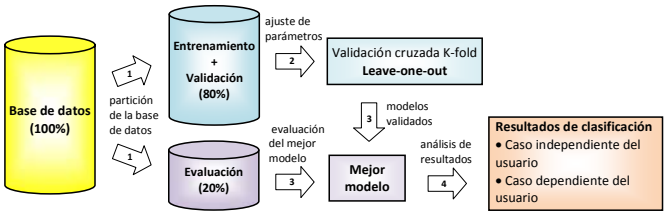

Fig. 2. Proceso de entrenamiento, validación y evaluación de modelos de ANNs.

Para el caso independiente del usuario, se utiliza la totalidad de los datos de las base de datos. Para el caso dependiente del usuario, se utilizan los datos del mismo usuario para entrenar, validar y evaluar los modelos siguiendo toda la metodología antes descripta.

III. Parámetros de Diseño y Comparación de Modelos de ANNs

La tasa de reconocimiento es utilizada como parámetro de comparación entre los modelos MLP y SVM con kernel lineal, fijando previamente un mismo tiempo de ejecución para los algoritmos de clasificación de ambos modelos. Los tiempos de ejecución dependen de la cantidad de operaciones aritméticas como multiplicaciones y sumas involucradas en los algoritmos. En ambos modelos, la cantidad de entradas y salidas son fijas. De esta manera, para el caso del MLP, el tiempo de ejecución puede ser variado según la cantidad de neuronas de la capa oculta. La cantidad de neuronas en la capa de salida es igual a 8, correspondiente a las clases que representan el vocabulario gestual. Para el caso de la SVM con kernel lineal, 28 clasificadores binarios deben implementarse para trabajar con clasificación multiclase, basada en la estrategia one-vs-one [15]. Una SVM con kernel polinómico de grado igual a 2 y estrategia multiclase one-vs-one es implementada sólo a modo de comparar resultados en la tasa de reconocimiento.

La mínima utilización de recursos para la implementación de una SVM con kernel lineal es considerada como punto de referencia para el diseño, que depende directamente de la cantidad de entradas y la cantidad de salidas del sistema. En consecuencia, la arquitectura del MLP resultará de igualar la utilización de recursos de la SVM con kernel lineal, ajustando la cantidad de neuronas de la capa oculta.

IV. Diseño, Implementación y Test del Sistema de Clasificación

Para llevar a cabo este trabajo, prácticas de ingeniería de software fueron utilizadas ya que el mismo involucra la creación de software como parte fundamental del sistema. Un análisis, definición y revisión de requerimientos fueron realizados. Los requerimientos validados son utilizados posteriormente para verificar el correcto funcionamiento del sistema. La etapa de análisis incluyó la extracción de información de la base de datos para conocer detalles del problema. Durante el diseño, la arquitectura y las interfaces del sistema fueron definidas, además de establecer un plan de pruebas de integración de los diferentes bloques que componen el clasificador. En la etapa de implementación se codificaron y probaron unitariamente los diferentes módulos de software. Por último, se realizan las pruebas de integración y de sistema que fueron planificadas en las etapas anteriores. La adopción de la metodología antes descripta permitió disminuir la inyección de errores dentro del software, reducir el re-trabajo, y obtener un sistema con cierto grado de calidad.

En primera instancia, el tipo de plataforma de hardware para implementar el sistema fue definido. Plataformas basadas en microcontrolador son seleccionadas, debido a la conjunción de características que se presentan como ventajas frente a otras plataformas para esta aplicación particular. Algunas de las características son bajo costo, bajo consumo energético, reúso y portabilidad de código, periféricos como I2C, SPI, UART, USB, alta frecuencia de operación, incorporación de módulos de hardware para procesamiento digital de señales, aritmética de punto flotante, además de proveer de diferentes maneras para actualizar el firmware.

A. Diseño

Diferentes configuraciones para el clasificador fueron analizadas. Los mejores resultados fueron obtenidos por el sistema cuya arquitectura se muestra en la Fig. 3.

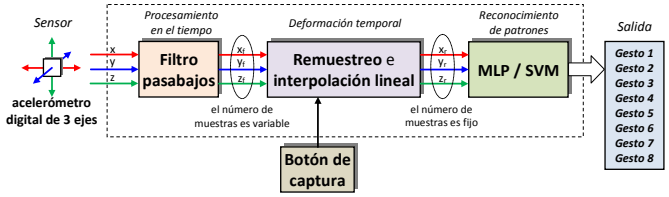

Fig. 3. Diagrama de bloques del sistema.

El sistema de reconocimiento de gestos de la mano tiene como entradas a las señales de aceleración y al botón de captura. La adquisición de los vectores de aceleración, correspondientes a un determinado gesto, inicia cuando el botón de captura del control remoto es presionado y finaliza cuando el botón es liberado. De esta manera, las muestras de los 3 ejes que componen un gesto son obtenidas. El sistema tiene una sola salida que corresponde al gesto reconocido a partir de las muestras capturadas, el cual será convertido en un comando de control para los sistemas de TV digital.

Antes de aplicar los algoritmos de clasificación del MLP y la SVM, los datos de entrada deben ser preprocesados. Los datos entregados por el acelerómetro son procesados por un filtro digital pasabajos de respuesta finita al impulso (FIR, finite impulse response) de fase lineal y rizado constante óptimo para atenuar cualquier componente de alta frecuencia como la asociada a vibraciones de la mano y ruido. Este filtro fue diseñado mediante el algoritmo de Parks-McClellan, para tener una frecuencia de corte de 7 Hz y una atenuación de -40 dB en la banda de rechazo. La longitud del filtro obtenido es igual a 9.

La etapa de remuestreo e interpolación lineal es la encargada de normalizar la entrada para los modelos de ANNs. Debido a que los gestos son variables en duración, es decir, ellos están compuestos por una cantidad variable de muestras de aceleración, una etapa de remuestreo e interpolación lineal es la encargada de mantener un número fijo de muestras de aceleración en los 3 ejes. Las muestras obtenidas de cantidad fija son las entradas para los modelos MLP y SVM. Durante la validación de los modelos se obtuvo empíricamente el valor óptimo de entradas para los modelos de ANNs, siendo igual a 10 entradas por eje.

Los modelos de MLP y SVM con kernel lineal llevan a cabo el reconocimiento de los patrones de los datos de entrada preprocesados. En las configuraciones propuestas para el MLP y la SVM dentro de este sistema, el tiempo queda representado de manera implícita, ya que la estructura temporal de la señal de entrada queda embebida en la estructura espacial de las ANNs [16].

B. Implementación

Los algoritmos de filtrado, de remuestreo e interpolación lineal, de clasificación del MLP y de la SVM con kernel lineal fueron implementados en lenguaje C y en formatos numéricos de punto flotante y de punto fijo. El formato numérico Q5.10 es una representación de punto fijo que utiliza números enteros para representar números reales de precisión finita. Esta representación utiliza una menor cantidad de recursos del hardware que la representación de punto flotante en dispositivos sin unidad de punto flotante (FPU, floating-point unit) para realizar operaciones aritméticas. El formato de punto flotante provee de mayor precisión en los cálculos y un mayor rango dinámico. Los algoritmos fueron implementados como módulos de software en lenguaje C. Una aplicación independiente y portable fue desarrollada e implementada en las diferentes plataformas de hardware bajo prueba disponibles. En un futuro, la funcionalidad del sistema de clasificación de gestos de la mano será ejecutada como una tarea dentro del contexto de un sistema operativo de tiempo real (RTOS, real time operating system).

Un gesto está representado por 3 vectores $\boldsymbol{x}$, $\boldsymbol{y}$, $\boldsymbol{z}$, de longitud N cada uno, que contienen las muestras de las variaciones de amplitud de aceleración en función del tiempo en el espacio tridimensional. Los datos crudos de entrada son limitados y normalizados para evitar posibles desbordamientos en las operaciones aritméticas. El filtro pasabajos es aplicado de manera continua a los datos de entrada. Por ejemplo, para el eje x, el filtro FIR de longitud $L=9$ con entrada $\boldsymbol{x}$ y salida $\boldsymbol{x_f}$ se describe mediante la ecuación en diferencias

$$x_f(n) = \sum_{k=0}^{L-1} b_k \, x(n-k) \qquad (1)$$

donde b_k es el conjunto de coeficientes del filtro. La ecuación (1) es utilizada para implementar el algoritmo del filtro.

Los gestos son variables en duración dependiendo de la velocidad del movimiento de la mano del usuario. Cuando el botón de captura es presionado, las muestras de entrada de cada eje son almacenadas en buffers hasta que el botón sea liberado. Antes de aplicar el algoritmo de remuestreo e interpolación lineal, se verifica que la duración del gesto sea válida. Para realizar la conversión desde frecuencia de muestreo original $F_s=1/T_s$ a la frecuencia de muestreo deseada $F_r=1/T_r$ simplemente se obtiene el nivel de la señal de entrada en los instantes de tiempo $t=m*T_r$, donde m es el índice de muestras de cantidad fija con valores de 0 a 9 por cada eje. Para el eje x, la fórmula para realizar la conversión de la frecuencia de muestreo y la interpolación lineal es

$$x_r(m) = \left(D * m - n_m + 1\right) x_f(n_m) + \left(D * m - n_m\right) x_f(n_m - 1) \qquad (2)$$

donde M es la cantidad de muestras fija de salida del remuestreo, $D=(N-1)/(M-1)$ es el factor de diezmado, y n_m es el índice de muestra de $\boldsymbol{x_f}$ que está situada en o justo por encima del valor de $D*m$. El algoritmo de remuestreo e interpolación lineal implica realizar comparaciones y multiplicaciones cuya cantidad depende de la duración de cada gesto. Así, el tiempo de ejecución de este algoritmo depende de la cantidad de muestras que componen cada gesto.

La Fig. 4 muestra cómo actúa el filtro pasabajos sobre las muestras crudas de aceleración del eje x, y cómo las muestras de salida del filtro pasabajos son procesadas por el algoritmo de remuestreo e interpolación lineal para mantener fija la cantidad de muestras.

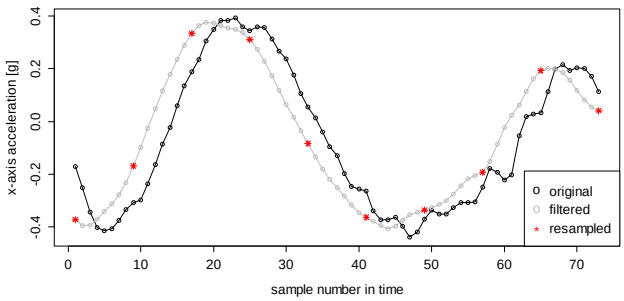

Fig. 4. Señal de aceleración correspondiente al eje x: original, filtrada y remuestreda.

Los vectores remuestreados $\boldsymbol{x_r}$, $\boldsymbol{y_r}$, $\boldsymbol{z_r}$ de longitud $M=10$ corresponden a las 30 entradas $\boldsymbol{e}$ de los algoritmos de clasificación del MLP y la SVM con kernel lineal.

Los MLP son ANNs de alimentación directa. Típicamente, la red consiste de un set de unidades sensoriales que constituyen la capa de entrada, una o más capas ocultas de nodos de cómputo, y la capa de salida de los nodos de cómputo. El MLP es entrenado mediante el algoritmo de propagación hacia atrás del error. Las fuerzas de las conexiones inter-neuronas, conocidas como pesos sinápticos, son usadas para almacenar el conocimiento adquirido. La salida de la j-ésima neurona de la capa oculta está dada por

$$V_j = g\left(\sum_{k=0}^{K} w_{jk} \, e_k\right) \qquad (3)$$

donde e_k representa a la k-ésima señal de entrada, K es la cantidad de neuronas de la capa de entrada, w_{jk} corresponde al peso sináptico que conecta la k-ésima neurona de la capa de entrada con la j-ésima neurona de la capa oculta, y $g()$ es la función de activación. Para $k=0$, la entrada corresponde al valor de bias que es una entrada fija en $e_0=-1$ y peso $w_{j0}=u_j$. La salida de la i-ésima neurona de la capa de salida está dada por

$$O_i = g\left(\sum_{j=0}^{J} W_{ij} \, V_j\right) \qquad (4)$$

donde J es la cantidad de neuronas de la capa oculta, W_{ij} corresponde al peso sináptico que conecta la j-ésima neurona de la capa oculta con la i-ésima neurona de la capa de salida, y $g()$ es la función de activación. Para $j=0$, la entrada corresponde al valor de bias que es una entrada fija en $V_0=-1$ y peso $W_{i0}=u_i$. En ambos casos, la función de activación de las neuronas que se utiliza en este trabajo es la función sigmoidea, implementada mediante una aproximación realizada con tramos lineales [17]. Utilizando (3) y (4) puede implementarse el algoritmo de clasificación del MLP.

La SVM es otra categoría de ANNs de alimentación directa que también puede ser utilizada para clasificación de patrones.

La SVM con kernel lineal es implementada utilizando la ecuación del hiperplano óptimo que representa una superficie de decisión lineal multidimensional en el espacio de entrada, definido por

$$\boldsymbol{w_0^T e} + b_0 = 0 \quad (5)$$

donde $\boldsymbol{e}$ es el vector de entrada, $\boldsymbol{w_0}$ es un vector de pesos y b_0 es el valor de bias. El objetivo del entrenamiento es encontrar el vector de parámetros $\boldsymbol{w_0}$ y el valor de bias b_0 que definan el hiperplano óptimo. Dado que una SVM usada para clasificar opera como clasificador binario, la estrategia multiclase one-vs-one es utilizada para resolver el problema de clasificación de *C=8* clases. De esta manera, se construyen *[C*(C-1)]/2=28* clasificadores binarios que corresponden a todas las combinaciones posibles de pares de clases. La clase de salida es obtenida por un esquema de votación. Utilizando (5) pueden implementarse los 28 clasificadores binarios de la SVM con kernel lineal.

Las ANNs que se implementan en este trabajo se entrenaron a través de un proceso de aprendizaje supervisado, utilizando las librerías provistas por R: nnet [18] y e1071 [19]. Para lograr la convergencia de los modelos de MLP y SVM, parámetros de ajuste fueron modificados durante el entrenamiento y la validación. El valor de ajuste final, número de iteraciones del ajuste de los pesos y valores iniciales de los pesos, fueron ajustados en el MLP. El entrenamiento fue realizado evitando mínimos locales y sobreentrenamiento. Los modelos de MLP han sido construidos para tener una correcta capacidad de generalización. Para el caso de la SVM con kernel lineal, el parámetro de regularización es el costo. El costo fue variado con la finalidad de obtener un comportamiento óptimo del modelo. Según el criterio de diseño de la Sección III, el MLP resultante tiene una arquitectura de 30 neuronas en la capa de entrada, 20 neuronas en la capa oculta y 8 neuronas en la capa de salida, con un total de 788 pesos sinápticos. Cada una de las 28 SVMs entrenadas tiene 30 entradas y, en consecuencia, 30 parámetros de entrada y 1 valor de bias, para un total de 840 parámetros de entrada y 28 valores de bias.

Además de la implementación en lenguaje C, los algoritmos fueron implementados en un aplicativo de software para el sistema operativo Android, mediante el paradigma de programación orientada a objetos. El aplicativo se ejecuta en un smartphone con la finalidad de realizar pruebas reales de los algoritmos del clasificador. El aplicativo de software utiliza el acelerómetro de 3 ejes del smartphone y ejecuta los algoritmos de clasificación cuando el gesto es capturado. El aplicativo también permitió crear una base de datos para entrenar los modelos de MLP y SVM.

C. Testing

Durante cada una de las etapas del ciclo de vida del proyecto, planes de pruebas fueron creados para guiar la etapa de validación correspondiente. Casos de prueba fueron definidos en cada uno de estos planes para obtener las pruebas a nivel unitario, integración y sistema.

Los test unitarios fueron realizados durante el proceso de codificación de cada módulo de software. El driver I2C para configuración y captura de datos desde el acelerómetro fue probado. Niveles de aceleración estática en los 3 ejes fueron medidos y la configuración de la frecuencia de muestreo fue realizada y verificada con osciloscopio. El filtro pasabajos fue testeado con un generador de funciones y osciloscopio, verificando la frecuencia de corte, ganancia en la banda de paso y atenuación en la banda de rechazo. El módulo de remuestreo e interpolación lineal fue testeado introduciendo muestras de cantidad variable de formas de onda conocidas para verificar la correcta interpolación y cantidad de muestras de salida. Los módulos del MLP y la SVM con kernel lineal se probaron con datos de la base de datos para verificar su correcta salida en ambos formatos numéricos.

Los tests de integración fueron realizados entre pares de módulos contiguos, verificando que las interfaces entre módulos operen adecuadamente. Principalmente, los tipos y formatos de los datos fueron verificados para asegurar una correcta integración de los módulos.

Los tests de sistema fueron realizados en base a los requerimientos definidos. Los test asociados a los requerimientos de performance se analizan en detalle en la Sección V. Como en toda implementación digital, el error debido a la longitud de palabra finita en la representación numérica está presente. Este error depende del formato numérico usado. Los resultados de clasificación de los modelos basados en MLP y SVM con kernel lineal e implementados en formatos numéricos de punto fijo y punto flotante en lenguaje C, fueron contrastados con los resultados de clasificación de los modelos en lenguaje R (el formato numérico de punto flotante IEEE-754 de doble precisión es usado). Los resultados de clasificación fueron exactamente los mismos. Como un test de sistema de una implementación cercana a la definitiva, los algoritmos fueron probados en un smartphone que ejecuta un aplicativo de software para Android, obteniendo muy buenos resultados de clasificación y tiempos de ejecución.

V. Resultados Experimentales

Luego de realizar el entrenamiento y validación de los modelos de MLP y SVM con kernel lineal y polinómico, el mejor modelo entrenado de cada tipo fue seleccionado para probar su desempeño con los datos de evaluación.

A. Tasa de reconocimiento

Matrices de confusión son utilizadas para mostrar el desempeño de los modelos. Las columnas representan a la salida estimada o predicha. Las filas representan a la salida real. La diagonal principal muestra la cantidad de clases correctamente clasificadas. En el resto de las celdas, los errores de clasificación son visualizados.

La Tabla I muestra la matriz de confusión normalizada del MLP para el caso independiente del usuario, con una tasa de reconocimiento promedio de 98.65%.

TABLA I. Matriz de confusión normalizada del MLP para el caso independiente delusuario.

Salidas		estimada							
		1	2	3	4	5	6	7	8
real	1	**0.982**	0	0	0.009	0	0.009	0	0
	2	0	**0.973**	0	0	0	0.009	0.009	0.009
	3	0	0	**0.982**	0	0	0.009	0.009	0
	4	0	0	0.009	**0.991**	0	0	0	0
	5	0	0	0	0.009	**0.991**	0	0	0
	6	0.009	0	0	0.009	0.009	**0.973**	0	0
	7	0	0	0	0	0	0	**1**	0
	8	0	0	0	0	0	0	0	**1**

La Tabla II muestra la matriz de confusión normalizada de la SVM con kernel lineal, con una tasa de reconocimiento promedio de 97.31%.

TABLA II. MATRIZ DE CONFUSIÓN NORMALIZADA DE LA SVM CON KERNEL LINEAL PARA EL CASO INDEPENDIENTE DEL USUARIO.

Salidas		estimada							
									
real		**0.982**	0	0	0.009	0	0.009	0	0
		0.009	**0.973**	0	0	0	0	0.018	0
		0	0	**0.991**	0	0	0	0.009	0
		0	0	0.018	**0.973**	0	0	0	0.009
		0	0	0.009	0	**0.991**	0	0	0
		0	0.009	0.009	0.027	0.018	**0.938**	0	0
		0	0.027	0.009	0	0	0	**0.964**	0
		0	0.027	0	0	0	0	0	**0.973**

La tasa de reconocimiento promedio alcanzada por la SVM con kernel polinómico de grado 2 fue del 99.21%.

Además, pruebas variando la cantidad de neuronas de la capa oculta del MLP fueron realizadas, obteniendo una tasa de reconocimiento promedio de 96.12% con 10 neuronas, con una disminución cercana al 50% sobre la utilización de recursos del hardware.

En el caso dependiente del usuario, es de esperar que los resultados en la tasa de reconocimiento mejoren respecto al caso independiente del usuario, debido a que se acota la variabilidad a un usuario en particular. Una tasa de reconocimiento promedio del 99.21% fue obtenida por el modelo MLP, 99.21% para la SVM con kernel lineal, y 99.43% para la SVM con kernel polinómico.

En la Tabla III se muestra una tabla comparativa con otros trabajos de investigación que utilizaron técnicas diferentes para reconocimiento de gestos de la mano utilizando acelerómetros.

TABLA III. COMPARATIVA CON OTROS TRABAJOS.

Técnica de reconocimiento de patrones	Precisión [%]		Número de gestos
	Dependiente del usuario	Independiente del usuario	
DTW con AP y CS [5]	99.79	96.00	18
FDSVM [6]	95.21	89.29	12
DTW [11]	93.50	75.40	8
DTW en 3 ejes [20]	99.20	96.40	8
PCA y árboles de decisión [21]	-	97.35	10
MLP propuesto	**99.21**	**98.65**	**8**
SVM con kernel lineal propuesto	**99.21**	**97.31**	**8**

B. Tiempos de ejecución de los algoritmos

La Tabla IV muestra los tiempos de ejecución de los algoritmos implementados en lenguaje C en diferentes plataformas de hardware basadas en microcontrolador y formatos numéricos, para los gestos de duración mínima y máxima encontrados en la base de datos, indicada según la cantidad de muestras. Como puede verse, ambos modelos MLP y SVM con kernel lineal presentan tiempos de ejecución similares.

TABLA IV. TIEMPOS DE EJECUCIÓN DE LOS ALGORITMOS.

Microcontrolador	Formato numérico	Remuestreo e interpolación lineal [msec]		MLP [msec]		SVM con kernel lineal [msec]		Reloj [MHz]
		Gesto de 17 muestras	Gesto de 315 muestras	Gesto de 17 muestras	Gesto de 315 muestras	Gesto de 17 muestras	Gesto de 315 muestras	
PIC24FJ256GB110 (núcleo de 16 bits)	IEEE-754 flotante 32-bits	1.274	8.893	16.307	16.638	16.484	16.430	32
	Q5.10	0.887	8.498	3.117	3.068	3.027	3.021	
dsPIC33FJ64GP204 (núcleo de 16 bits)	IEEE-754 flotante 32-bits	0.509	3.557	6.523	6.655	6.594	6.573	80
	Q5.10	0.355	3.399	1.247	1.227	1.211	1.208	
PIC32MX795F512L (núcleo de 32 bits)	IEEE-754 flotante 32-bits	0.089	0.633	1.546	1.576	1.567	1.569	80
	Q5.10	0.061	0.596	0.450	0.448	0.477	0.477	
STM32F407VGT6 with FPU disabled (núcleo de 32 bits)	IEEE-754 flotante 32-bits	0.032	0.174	0.440	0.450	0.440	0.440	168
	Q5.10	0.020	0.155	0.108	0.108	0.100	0.100	
STM32F407VGT6 with FPU enabled (núcleo de 32 bits)	IEEE-754 flotante 32-bits	0.006	0.028	0.142	0.146	0.122	0.122	168

Como puede apreciarse, una implementación en lenguaje C podría ser realizada sin inconvenientes en un microcontrolador para tiempo real. El formato de punto flotante IEEE-754 permite tener un mayor rango dinámico y precisión en los valores de los parámetros de los algoritmos, a costa de un mayor tiempo de procesamiento en las plataformas sin FPU. El formato de punto fijo Q5.10 permite una ejecución más veloz a costa de una menor precisión en los parámetros de los algoritmos debido al truncamiento. Las plataformas de hardware más atractivas para su implementación son las alternativas de 32 bits [22], [23], [24], puesto que el sistema también deberá ejecutar tareas adicionales asociadas a la funcionalidad de un control remoto moderno y dentro del contexto de un RTOS. La utilización de una FPU acelera la ejecución de los algoritmos utilizando el formato de punto flotante. Para comparar resultados con otras técnicas de reconocimiento de gestos de la mano, en [11], un tiempo de ejecución de 300 msec en un microcontrolador de 16 bits fue obtenido. En [20], un kit de desarrollo basado en tecnología FPGA es usado para implementar una unidad de aceleración por hardware con un tiempo de ejecución del algoritmo de clasificación igual a 58.3 msec.

Los algoritmos fueron implementados y probados dentro de un smartphone con buenos resultados en el reconocimiento y en los tiempos de ejecución. Un smartphone Samsung Galaxy S3 mini fue utilizado, corriendo el sistema operativo Android (plataforma 4.03 y nivel de API 15). La Tabla V resume los resultados.

TABLA V. TIEMPOS MÍNIMOS DE EJECUCIÓN DE LOS ALGORITMOS EN UN SMARTPHONE CORRIENDO ANDROID.

Formato numérico	Remuestreo e interpolación lineal [msec]		MLP [msec]		SVM con kernel lineal [msec]	
	Gesto de 17 muestras	Gesto de 315 muestras	Gesto de 17 muestras	Gesto de 315 muestras	Gesto de 17 muestras	Gesto de 315 muestras
flotante 32-bits	0.03	0.09	0.244	0.244	0.244	0.244

La precisión obtenida en la prueba con smartphone para el caso dependiente del usuario fue del 100%.

VI. CONCLUSIONES

El objetivo de diseñar y obtener un sistema de reconocimiento de gestos de la mano basado en acelerómetros de tecnología MEMS, con una buena tasa de reconocimiento y tiempos de ejecución reducidos, fue alcanzado. El clasificador propuesto involucra técnicas de procesamiento digital de señales y técnicas de reconocimiento de patrones, cuyos algoritmos serán embebidos dentro del hardware. Las pruebas realizadas en diferentes plataformas de hardware indican que es factible implementar el sistema en un microcontrolador de 32 bits. La familia de microcontroladores PIC32MX250 es la

plataforma seleccionada. Esta plataforma es elegida ya que ejecuta los algoritmos velozmente, con un reducido tiempo de uso de procesador, y cuenta con suficiente memoria de datos y de programa. Además, esta plataforma es la de menor costo entre las plataformas testeadas. Otros factores importantes de la elección de esta plataforma son la disponibilidad en el mercado local y herramientas de desarrollo de uso libre. Además, el sistema de clasificación con ambos modelos de ANNs fue probado en un smartphone con muy buenos resultados.

Dos tipos de ANNs fueron comparadas en busca de una solución óptima entre precisión y utilización de recursos del hardware, obteniéndose buenos resultados tanto en un MLP como en una SVM con kernel lineal. Para un mismo uso de procesador, el MLP mostró un desempeño levemente superior que la SVM con kernel lineal. El MLP permite controlar la utilización de recursos variando las neuronas de la capa oculta. El modelo MLP es el modelo seleccionado que finalmente será embebido dentro del control remoto. El diseño y la implementación de los modelos de inteligencia artificial fueron expuestos en detalle para ser implementados en sistemas embebidos. Diferentes plataformas de hardware basadas en microcontrolador fueron utilizadas para realizar y documentar las mediciones de los tiempos de ejecución de los algoritmos de clasificación.

Hemos demostrado que la implementación de un sistema de reconocimiento de gestos de la mano basado en acelerómetros y dispositivos de recursos limitados de bajo costo, es posible. Hemos logrado excelentes tasas de reconocimiento de patrones, alcanzando tiempos de ejecución que permiten una operación en tiempo real. Hemos verificado la adecuada utilización de modelos basados en ANNs para el desarrollo de una interfaz más natural y amigable para mejorar la experiencia de usuario en el contexto de la TV digital de Argentina.

VII. TRABAJO A FUTURO

Una base de datos será creada con el hardware definitivo para entrenar el modelo de MLP. Este modelo será embebido dentro del control remoto. Además, el número de clases del vocabulario gestual será incrementado. Un filtro bayesiano será incorporado para reforzar la salida correcta del MLP. Por último, la delimitación automática del inicio y finalización de los gestos será incluida. Esto es posible ya que el sistema podría funcionar en modo de reconocimiento continuo de gestos de la mano debido a los muy buenos tiempos de ejecución obtenidos.

REFERENCES

[1] S. Mitra and T. Acharya, “Gesture recognition: a survey,” in IEEE Transactions on Systems, Man, and Cybernetics, vol.37, pp. 311-324, May. 2007.

[2] S. Zhou, Q. Shan, F. Fei, J. Li, C. Kwong, P. Wu, B. Meng, C. Chan, J. Liou, “Gesture recognition for interactive controllers using MEMS motion sensors,” in 2009 4th IEEE International Conference on Nano/Micro Engineered and Molecular Systems (NEMS 2009), vol. 2, pp. 935-940, Shenzhen, China, Jan. 2009.

[3] S.Cho, E. Choi, W. Bang, J. Yang, J. Sohn, D. Kim, Y. Lee, S. Kim, “Two stage recognition of raw acceleration signals for 3D gesture understanding cell phones,” in 10th International Workshop on Frontiers in Handwriting Recognition, 2006.

[4] B. Lee Cosío, “ANN for gesture recognition,” M.S. thesis, School of Eng., Panamerican Univ., Mexico, 2012.

[5] A. Akl and S. Valaee, "Accelerometer-based gesture recognition via dynamic-time warping, affinity propagation, & compressive sensing," in IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP), vol. 4, pp. 2270-2273, Dallas, Texas, USA, Mar. 2010.

[6] J. Wu, G. Pan, D. Zhang, G. Qi, S. Li, “Gesture recognition with a 3-D accelerometer,” in 2009 Proceedings of the 6th International Conference on Ubiquitous Intelligence and Computing (UIC 2009), vol. 1, pp. 25-38, Brisbane, Australia, Jul. 2009.

[7] G. Niezen and G. Hancke, “Evaluating and optimising accelerometer-based gesture recognition techniques for mobile devices,” in AFRICON 2009, vol. 1, pp. 424-429, Nairobi, Kenya, Sep. 2009.

[8] S. Haykin, “Multilayer perceptrons,” in Neural Networks: A Comprehensive Foundation, 2th ed. New Jersey: Prentice Hall, 1999, ch. 4, sec. 1, pp. 178-180.

[9] V. Vapnik, “An overview of statistical learning theory,” in IEEE Transactions on Neural Networks, vol. 10, no. 5, pp. 988-999, Sep. 1999.

[10] S. Haykin, “Introduction,” in Neural Networks: A Comprehensive Foundation, 2th ed. New Jersey: Prentice Hall, 1999, ch. 1, sec. 1, pp. 23-27.

[11] L. Jiayang, W. Zhen, Z. Lin, “uWave: accelerometer-based personalized gesture recognition and its applications,” in 2009 IEEE International Conference on Pervasive Computing and Communications (PerCom 2009), vol. 1, pp. 113-121, Galveston, Texas, Mar. 2009.

[12] uWave project database [online]. Available: http://www.owlnet.rice.edu/~zw3/projects_uWave.html

[13] The R project for statistical computing: http://www.r-project.org

[14] RStudio IDE: http://www.rstudio.com

[15] S. Haykin, “Support Vector Machines,” in Neural Networks: A Comprehensive Foundation, 2th ed. New Jersey: Prentice Hall, 1999, ch. 6, sec. 4, pp. 351-356.

[16] S. Haykin, “Temporal Processing using feedforward networks,” in Neural Networks: A Comprehensive Foundation, 2th ed. New Jersey: Prentice Hall, 1999, ch. 13, sec. 1, pp. 657-658.

[17] M.T. Tommiska, “Efficient digital implementation of the sigmoid function for reprogrammable logic,” in IEE Proceedings Computers and Digital Techniques, vol. 150, no. 6, pp. 403-411, Nov. 2003.

[18] Package nnet [Online]. Available: http://cran.r-project.org/web/packages/nnet/index.html

[19] Package e1071 [Online]. Available: http://cran.r-project.org/web/packages/e1071/index.html

[20] S. Hussain and A. Rashid, “User independent hand gesture recognition by accelerated DTW,” in 2012 International Conference on Informatics, Electronics & Vision (ICIEV 2012), vol. 2, pp. 1033-1037, Dhaka, Bangladesh, May. 2012.

[21] X. Dang, W. Wang, K. Wang, M. Dong, L. Yin, “A user-independent sensor gesture interface for embedded device,” in 2011 IEEE SENSORS Proceedings, vol. 2, pp. 1465-1468, Limerick, Ireland, Oct. 2011.

[22] PIC32MX1xx/2xx datasheet [Online], Microchip, 2012. Available: http://www.microchip.com

[23] PIC32MX7xx datasheet [online], Microchip, 2010. Available: http://www.microchip.com

[24] STM32F407xx datasheet [online], STMicroelectronics, 2012. Available: http://www.st.com