Sistema de reconocimiento de palabras para la EDU-CIAA

Autor

Gonzalo Ávila Alterach - Padrón 94.950

Fecha:

2do cuatrimestre 2016

Este plan de trabajo ha sido realizado en el marco de la asignatura Seminario de Sistemas Embebidos entre agosto y octubre de 2016.

Tabla de contenido

Revisión	Cambios realizados	Fecha
1.0	Creación del documento	5/11/2016

Descripción técnica-conceptual del Proyecto a realizar

El propósito del proyecto es desarrollar una biblioteca que permita reconocer palabras aisladas, en español, esperando que el proyecto sea de utilidad para cualquier persona que desee integrar un sistema de estas características en un diseño existente.

La biblioteca usará las funciones de CMSIS-DSP, una biblioteca pensada para el procesamiento de señales en microcontroladores Cortex-M, a modo de utilizar de forma óptima las instrucciones del mismo que puedan mejorar la velocidad del procesamiento.

Además de la biblioteca principal, se están desarrollando diversos módulos de prueba, a modo de validación de cada una de las partes de la misma:

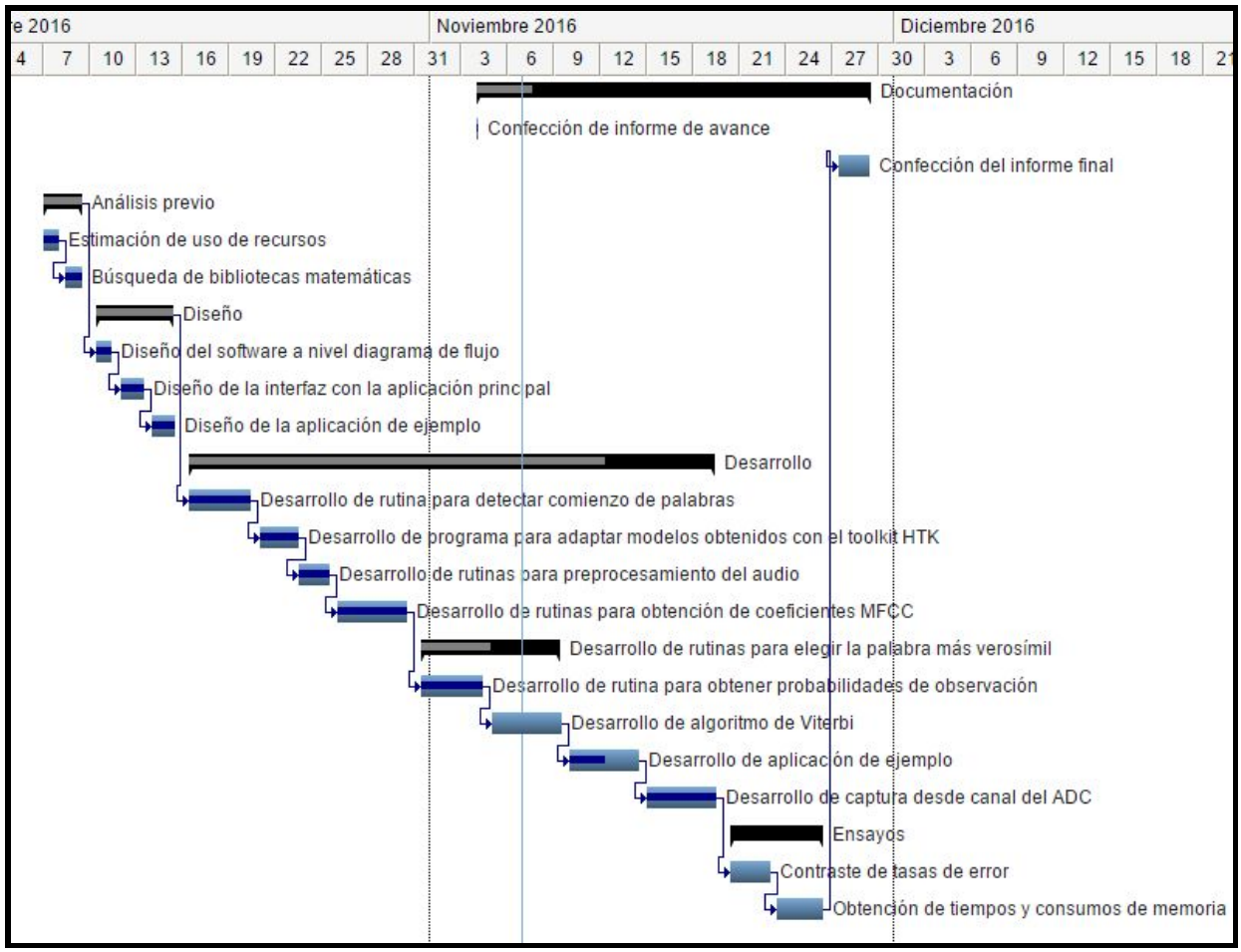
- Comparación de coeficientes MFCC obtenidos con la biblioteca propia con los obtenidos mediante el toolkit HTK.
- Comparación de tasas de error del sistema completo, comparando con los del toolkit HTK.
- Validación de las reglas utilizadas para convertir una palabra en los fonemas correspondientes, utilizando un diccionario de pronunciación provisto por el Grupo de Procesamiento del Habla de la FIUBA.

1. Avance en las tareas

1.1 Confección de informe de avance		1.2 Confección del informe final		2.1 Estimación de consumo de memoria y cantidad de procesamiento necesario		2.2 Búsqueda de bibliotecas matemáticas que implementen eficientemente las transformaciones requeridas	
$	--	$		$	-	$	-
3.1 Diseño del software a nivel diagrama de flujo		3.2 Diseño de la interfaz con la aplicación principal		3.3 Diseño de la aplicación de prueba			
$	=	$	-	$	-		
4.1 Desarrollo de rutina para detectar comienzo de palabras		4.2 Desarrollo de programa para adaptar modelos obtenidos con el toolkit HTK		4.3 Desarrollo de rutinas para preprocesamiento del audio		4.4 Desarrollo de rutinas para obtención de coeficientes MFCC	
$	--	$	+	$	-	$	=
4.5.1 Desarrollo de rutina para obtener probabilidades de observación		4.5.2 Desarrollo de algoritmo de Viterbi		4.3 Desarrollo de aplicación de ejemplo		4.4 Desarrollo de captura desde canal del ADC	
$	+	$	+	$	+	$	--
5.1 Contraste de tasas de error con las obtenidas usando el toolkit HTK		5.2 Obtención de tiempos y consumos de memoria de cada una de las rutinas					
$		$					

Diagrama de Gantt actualizado

	EDT		Nombre	Duración	% Completado	Inicio	Fin	Pred
1	1		**Documentación**	13d	14%	**03/11/2016**	**28/11/2016**	
2	1.1	✓	Confección de informe de avance	1h	100%	03/11/2016	03/11/2016	
3	1.2		Confección del informe final	6h	0%	26/11/2016	28/11/2016	23
4	2	✓	**Análisis previo**	1.5d	100%	**06/10/2016**	**08/10/2016**	
5	2.1	✓	Estimación de uso de recursos	3h	100%	06/10/2016	07/10/2016	
6	2.2	✓	Búsqueda de bibliotecas matemáticas	3h	100%	07/10/2016	08/10/2016	5
7	3	✓	**Diseño**	3d	100%	**09/10/2016**	**14/10/2016**	
8	3.1	✓	Diseño del software a nivel diagrama de flujo	4h	100%	09/10/2016	10/10/2016	4
9	3.2	✓	Diseño de la interfaz con la aplicación principal	4h	100%	11/10/2016	12/10/2016	8
10	3.3	✓	Diseño de la aplicación de ejemplo	4h	100%	13/10/2016	14/10/2016	9
11	4		**Desarrollo**	17.5d	79%	**15/10/2016**	**18/11/2016**	
12	4.1	✓	Desarrollo de rutina para detectar comienzo de palabras	10h	100%	15/10/2016	19/10/2016	7
13	4.2	✓	Desarrollo de programa para adaptar modelos obtenidos con	5h	100%	20/10/2016	22/10/2016	12
14	4.3	✓	Desarrollo de rutinas para preprocesamiento del audio	5h	100%	22/10/2016	24/10/2016	13
15	4.4	✓	Desarrollo de rutinas para obtención de coeficientes MFCC	10h	100%	25/10/2016	29/10/2016	14
16	4.5		**Desarrollo de rutinas para elegir la palabra más verosím**	5d	50%	**30/10/2016**	**08/11/2016**	
17	4.5.1	✓	Desarrollo de rutina para obtener probabilidades de obser	10h	100%	30/10/2016	03/11/2016	15
18	4.5.2		Desarrollo de algoritmo de Viterbi	10h	0%	04/11/2016	08/11/2016	17
19	4.6		Desarrollo de aplicación de ejemplo	10h	50%	09/11/2016	13/11/2016	18
20	4.7	✓	Desarrollo de captura desde canal del ADC	10h	100%	14/11/2016	18/11/2016	19
21	5		**Ensayos**	3.5d	0%	**19/11/2016**	**25/11/2016**	
22	5.1		Contraste de tasas de error con las obtenidas usando el toolki	7h	0%	19/11/2016	22/11/2016	20
23	5.2		Obtención de tiempos y consumos de memoria de cada una d	7h	0%	22/11/2016	25/11/2016	22



2. Cumplimiento de los requerimientos

1. Adquisición y almacenamiento	Req #1.1: Tasa de muestreo: 16 ksps
	Req #1.2: Bits de muestreo: 10 bits
	Req #1.3: Tipo de almacenamiento: continuo, en buffer circular
	Req #1.4: Duración de almacenamiento: 1 segundo **(originalmente eran 2 segundos, se redujo para asegurarse que el buffer de muestras entrase completo en un único banco de memoria)**

2. Detección de comienzo y fin de palabras	Req #2.1: Algoritmo a usar: basado en la energía de una ventana pequeña
	Req #2.2: Error menor al 10% con una entrada con relación señal-ruido de 25 dB
3. Reconoc. de palabras aisladas	Req #3.1: Reconocimiento estadístico, basado en cadenas ocultas de Markov y coeficientes MFCC
	Req #3.2: Soporte de modelos del tipo “mezcla de Gaussianas” con matriz de covarianza diagonal y formato propietario del toolkit HTK
	Req #3.3: Cantidad máxima de palabras a reconocer: al menos 10 palabras distintas
	Req #3.4: Cantidad máxima de fonemas por palabra: al menos 10 fonemas por palabra
	Req #3.5: Tasas de error dentro de un 5% de las obtenidas con el toolkit HTK
4. Interfaz biblioteca-programa	Req #4.1: Posibilidad de integración a una aplicación existente
	Req #4.2: Soporte para obtener las N palabras más probables y una cantidad que las relacione

3. Gestión de riesgos

Riesgo #1: Dificultad para conseguir la placa de desarrollo
Riesgo #2: Dificultad para cumplir con los tiempos de las tareas
Riesgo #3: Que el hardware de la EDU-CIAA no pueda ser usado para el proyecto
Riesgo #4: Implementar incorrectamente el algoritmo de reconocimiento