
بسم الله الرحمن الرحيم

عنوان پایان نامه :

سیستم های هوشمند

(تشخیص هویت توسط صدا در شبکه های عصبی)

استاد راهنما:

سرکار خانم دکتر حیدری

تدوین گر:

سیده شیما موسویان

سال ۱۳۸۹

تقدیم به

این اثر ناچیز را در نهایت عشق به پدر و مادر عزیزم که همواره در طول حیات پربارشان از حمایت های بی دریغ آنها بهره مند بوده ام تقدیم می کنم و تقدیم به همه آنان که مرا علم آموختند .

سپاسگزاری

سپاس بیکران پروردگار توانایی را که به انسان قدرت اندیشیدن بخشید تا به یاری این موهبت، راه ترقی و تعالی را بپیماید.

در اینجا لازم می دانم که از سرکار خانم دکتر حیدری و تمامی اساتیدم که در طی این مدت از آنها کسب علم و ادب نمودم صمیمانه قدردانی نمایم.

فهرست مطالب

فهرست اشکال

عنوان **صفحه**

فصل اول

۱–۱ مقدمه

سیستمهای تشخیص گوینده در حالت کلی به دو نوع سیستمهای تأیید هویت گوینده و سیستمهای بازشناسی گوینده تقسیم می‌شوند. تفاوت این دو سیستم در نحوهٔ پذیرش ورودی است: در سیستمهای نوع اول گوینده با ارائه یک شناسه ادعای هویت یک کاربر خاص را می‌نماید حال آن که در سیستمهای نوع دوم گوینده فقط عبارت عبور خود را بیان می‌کند و سیستم او را از بین تمامی کاربران خود تشخیص می‌دهد.

ساختارهایی که برای هر دو نوع سیستم ارائه شد هر دو دارای یک مرحله برای تشخیص میزان شباهت الگوهای متعلق به گویندهٔ حاضر با گویندهٔ مورد ادعا (نوع اول) یا همهٔ گویندگان است که با استفاده از آن معیاری برای تصمیم گیری در اختیار ما قرار داده می‌شود.

همچنان که برای تشخیص الگو الگوریتمهای متعدد و روشهای گوناگون وجود دارد الگوریتمهای گوناگونی نیز برای یافتن میزان شباهت میان الگوها وجود دارد که انتخاب هر کدام از آنها بستگی به ساختار سیستم مقصد دارد.

حال سعی داریم در این پروژه به بررسی سیستمهای تشخیص هویت توسط صدا بپردازیم:

۱–۲ موضوع پژوهش

موضوع پژوهش عبارت است از

" سیستم های هوشمند (تشخیص هویت توسط صدا در شبکه های عصبی)"

۱–۳ بیان مسأله

کاربردهای فن‌آوریهای زیستی عبارت است از:

- تجارت الکترونیک،

- نظارت امنیتی،
- دسترسی به پایگاه داده‌ها،
- کنترل مرزها و مهاجرت،
- تحقیقات قضایی و پزشکی از راه دور.

برتری سیستمهای مبتنی بر زیست‌سنجی آن است که به شدت به ویژگیهای فردی اشخاص وابسته‌اند و به راحتی نمی‌توانند مورد سوء استفاده قرار گیرند.

تشخیص گوینده :

عبارت است از فرایند تشخیص خودکار هویت شخص صحبت‌کننده بر اساس اطلاعات یکتای موجود در موج صوتی صحبت او.

انواع سیستمهای تشخیص گوینده

۱. سیستمهای تأیید هویت گوینده

شخص عموماً با انتخاب یا وارد کردن نام یکی از کاربران خاص سیستم ادعا می‌کند که او همان کاربر ثبت‌شدهٔ سیستم است. در این حالت سیستم وظیفه دارد ویژگیهای صوتی شخص مدعی را با ویژگیهای صوتی ذخیره شدهٔ کاربر ثبت شدهٔ مورد ادعا مقایسه نموده و با استفاده از نتیجهٔ به دست آمده ادعای شخص را بپذیرد یا رد کند.

۲. سیستمهای بازشناسی هویت گوینده(بهتر)

شخص صحبت کننده ادعای هویت یک کاربر خاص ثبت شده را نمی‌نماید و این سیستم است که وظیفه دارد که او را در میان کاربران ثبت شدهٔ سیستم بازشناسی نماید و یا تشخیص دهد که ویژگیهای صوتی او با هیچ یک از کاربران ثبت شده همخوانی ندارد.

تفاوت این دو سیستم در نحوهٔ پذیرش ورودی است: در سیستمهای نوع اول گوینده با ارائهٔ یک شناسه ادعای هویت یک کاربر خاص را می‌نماید حال آن که در سیستمهای نوع دوم گوینده فقط عبارت عبور خود را بیان می‌کند و سیستم او را از بین تمامی کاربران خود تشخیص می‌دهد.

سه روش عمدهٔ یافتن میزان شباهت الگوها

۱. روشهای مبتنی بر چشمپوشی زمانی پویا

۳. روشهای مبتنی بر مدلهای نهان مارکف

۴. روشهای مبتنی بر مقدارگزینی برداری

روشهای پیاده‌سازی

ویژگیهای فیزیکی افراد نظیر ساختار اندامهای صوتی، اندازهٔ چالهٔ بینی و ویژگیهای تارهای صوتی منحصر به فرد بوده و از طریق الگوریتمهای پردازش سیگنال به صورت پارامترهای خصیصه‌ای یا مجموعهٔ خصایص قابل استخراج می‌باشند. این حقیقت پایهٔ روشهای پیاده‌سازی سیستمهای تشخیص صحبت می‌باشند.

برخی ویژگیهای خاص صدای دیجیتالی از قبیل نرخ نمونه‌برداری، تعداد بیت هر نمونه و یک‌کاناله یا دوکاناله بودن صدا را مشخص کنیم.

آیندهٔ فن‌آوریهای پردازش صحبت:

صحبت سریع‌ترین و کاراترین روش ارتباط انسانهاست.

تشخیص صحبت پتانسیل جایگزینی نوشتن، تایپ، ورود صفحه‌کلید و کنترل الکترونیکی را که توسط کلیدها و دکمه‌ها اعمال می‌شود را داراست •استفاده از کامپیوتر را برای کلیهٔ افراد ناتوان بدنی که دارای تواناییهای شنوایی و گفتاری مناسب هستند آسان‌تر سازد.

۱-۴ ضرورت انجام پژوهش

با توجه به رشد و پیشرفت علم فناوری اطلاعات و رشد بازار نسبت به سیستم های تشخیص هویت (اثر انگشت، الگوی شبکیه، عنبیه و...) می توان از سیستم های تشخیش هویت

توسط صدا که صدای فرد مورد نظر را ابتدا دریافت نموده و سپس پردازش کرده و برای ورود فرد به سیستم تعریف نمود.

۱–۵ هدف از انجام تحقیق:

تشخیص هویت توسط صدا برای شناسایی افراد و اجازه ورود به سیستم.

۱–۶ سؤالات تحقیق:

۱. تشخیص هویت توسط صدا برای شناسایی افراد چه کاربردی برای جامعه دارد؟

۲. چقدر لازم است صوت خوب به نظر برسد؟

۳. چه نرخ داده‌ای قابل تحمل است؟

۱–۷ روش انجام تحقیق:

این تحقیق از لحاظ هدف، کاربردی و از نظر روش، اجرایی و به صورت پیمایشی انجام می شود.

داده های این تحقیق از روش کتابخانه ای و اینترنتی جمع آوری شده است. روش جمع آوری داده های اینترنتی استفاده مطالب و ترجمه متون لاتین می باشد که با کلیک بر روی سایت مورد نظر در قسمت منابع این پروژه میتوان به آن دسترسی پیدا کرد.

۱–۸ تعریف واژگان کلیدی

زیست‌سنجی:

دانش و فن‌آوری اندازه‌گیری و تحلیل آماری داده‌های زیستی

تشخیص گوینده :

عبارت است از فرایند تشخیص خودکار هویت شخص صحبت‌کننده بر اساس اطلاعات یکتای موجود در موج صوتی صحبت او.

همانند شماره‌گیری صوتی، بانکداری تلفنی، خرید تلفنی، خدمات دسترسی به پایگاه داده‌ها، خدمات اطلاعاتی، پست الکترونیکی صوتی، کنترل امنیتی برای ورود به قلمروهای اطلاعاتی محرمانه و دسترسی از راه دور به کامپیوترها را فراهم می‌آورد.

انواع سیستمهای تشخیص گوینده :

۱. سیستمهای تأیید هویت گوینده ۲. سیستمهای بازشناسی گوینده

فصل ۲

بخش اول

زیست‌سنجی و کاربردهای آن در سیستمهای امنیتی

۱– ۱ تعریف، ضرورت و کاربردها

زیست‌سنجی[۱] عبارت است از دانش و فن‌آوری اندازه‌گیری و تحلیل آماری داده‌های زیستی[۲]. در فن‌آوری اطلاعات واژهٔ زیست‌سنجی به مجموعه فن‌آوریهایی اطلاق می‌گردد که در آنها از اندازه‌گیری و تحلیل ویژگیهایی از بدن انسان همچون اثر انگشت، اثر کف دست، شبکیه و عنبیهٔ چشم، الگوهای صوتی، الگوهای مربوط به رخسار ، دمانگاری صورت[۳]، شکل دست یا گوش، داده‌های به دست آمده از گام، الگوهای وریدی[۴]، دی.ان.ای[۵] و یا ویژگیهایی همچون دستخط(امضا) و دینامیک ضربه زدن به صفحه‌کلید[۶] برای تأیید هویت[۷] اشخاص استفاده می‌شود.[۸] این فن‌آوریها در تلاشند تا اندازه گیری و مقایسهٔ ویژگیهای برشمرده شده را به منظور بازشناسی افراد به صورت خودکار درآورند.

فن‌آوریهای زیستی در ابتدا برای کاربردهای تخصصی نیازمند امنیت بالا پیشنهاد شدند اما اینک به عنوان عناصر کلیدی در توسعهٔ تجارت الکترونیک و سیستمهای برخط[۹] و به همان صورت برای سیستمهای امنیتی نابرخط[۱۰] و سیستمهای امنیتی منفرد[۱۱] مطرح می‌باشند.

این فن‌آوریها اجزاء مهمی را برای تنظیم و نظارت بر نحوهٔ دسترسی و حضور در سیستم فراهم می‌آورند. محدوده‌های عمدهٔ کاربرد این فن‌آوریها عبارتند از : تجارت الکترونیک، نظارت

---

[1] biometrics

[۳] تعریف از http://www.whatis.com/ برداشته شده است.

[3] facial thermography
[4] vein patterns
[5] DNA
[6] keystroke dynamics
[7] authentication

[۴] تعریف ذیل در منبع شمارهٔ ۳ فصل نیز جالب توجه است:

‹زیست‌سنجی عبارت است از اندازه‌گیری ویژگیهای فیزیکی یا زیستی معین یک فرد برای ایجاد یک شناسهٔ یکتا که بتواند به صورت الکترونیکی ذخیره، بازیابی و به منظور اهداف بازشناسی مشخص مقایسه گردد.›

[9] online
[10] off-line
[11] standalone

امنیتی، دسترسی به پایگاه داده‌ها، کنترل مرزها و مهاجرت، تحقیقات قضایی و پزشکی از راه دور[۱۲].

توسعهٔ فن‌آوریهای زیست‌سنجی فراتر از کاربردهای سنتی نیازمند امنیت بالا، یک اجبار نشأت گرفته از انگیزه‌های مالی است. امنیت معاملات برای آیندهٔ توسعهٔ تجارت الکترونیک یک مسألهٔ حیاتی است و نگرانیهای فراوانی دربارهٔ راه حلهای فعلی وجود دارد. مشکل شماره‌های شناسایی شخصی[۱۳] و شناسه‌های هویتی[۱۴] - مانند کارتها- این است که آنها صحت هویت شخصی را که از آنها استفاده می‌کند تأیید نمی‌کنند. آمارها میزان زیان ناشی از تقلب را به طور سالیانه برای کارتهای اعتباری بالغ بر چهارصد و پنجاه میلیون دلار و برای خودپرداز[۱۵]ها حدود سه‌میلیارد دلار برآورد می‌کنند. برتری سیستمهای مبتنی بر زیست‌سنجی آن است که به شدت به ویژگیهای فردی اشخاص وابسته‌اند و به راحتی نمی‌توانند مورد سوء استفاده قرار گیرند.

۲– ۱ بررسی عملکرد سیستمهای موجود

فعالیتهای انجام شده تا به حال منجر به ظهور ماشینهای گران قیمت زیست- سنجی شده است که علاوه بر قیمت زیاد معمولاً از لحاظ سرعت و عملکرد مناسب نیستند یا حداقل برای دستیابی به عملکرد مناسب باید محیط استفادهٔ آنها شرایط خاصی را داشته باشد و یا کاربران آنها آموزشهای گسترده‌ای را گذرانده باشند.

در حالی که بعضی از فن‌آوریهای زیست‌سنجی در قالب تولیدات تجاری به بازار عرضه شده‌اند بسیاری از این دسته فن‌آوریها در مرحلهٔ تحقیق و آزمایش قرار دارند. فن‌آوریهای مزبور نیازمند کارهای مطالعاتی بیشتر برای افزایش پایداری و بهبود عملکردشان برای استفاده در کاربردهای ویژه هستند.

---

[12] telemedicine
[13] personal identification numbers (PINs)
[14] identity tokens
[15] ATM (Automated Teller Machine)

پایداری در برابر تقلب ،دقت عملکرد، سرعت و تجهیزات مورد نیاز، همخوانی با سخت‌افزار و نرم‌افزار موجود، هزینه ،سادگی استفاده و پذیرش از سوی کاربر از جمله عوامل تعیین‌کننـده در موفقیت هر یک از فن‌آوریهای به کار گرفته شده می‌باشند.

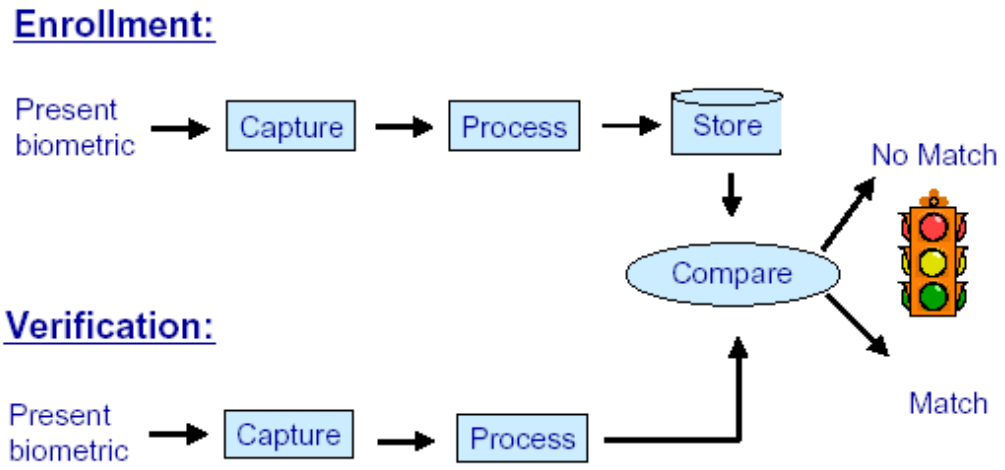
جدول شمارهٔ ۱-۱ مقایسه‌ای از معمول‌ترین سیستمهای زیست‌سنجی موجود را ارائه می‌دهد.

نوع سیستم	دقت عملکرد	سادگی استفاده	میزان پذیرش کاربر
اثر انگشت	بالا	متوسط	پایین
هندسهٔ دست	متوسط	بالا	متوسط
صوت	متوسط	بالا	بالا
شبکیهٔ چشم	بالا	پایین	پایین
عنبیهٔ چشم	متوسط	متوسط	متوسط
امضا	متوسط	متوسط	بالا
رخسار	پایین	بالا	بالا

جدول شمارهٔ ۱- ۱ مقایسهٔ سیستمهای زیست‌سنجی معمول (منبع شمارهٔ ۱)

۳- ۱ اجزای سیستمهای زیست‌سنجی

عملیات سیستمهای زیست‌سنجی در بر دارندهٔ دو مرحلهٔ مجـزا مـی‌باشـد: ثبـت کـاربر و بازشناسی کاربر. در مرحلهٔ اول اطلاعات مربوط به کاربر به سیستم وارد می‌شوند و در مرحلـهٔ دوم اطلاعات ورودی حاضر با اطلاعات ذخیره شده مقایسه می‌گردند.



شکل شمارهٔ ۱-۱ مراحل لازم عملیاتی در یک سیستم امنیتی مبتنی بر زیست‌سنجی (منبع شمارهٔ ۳)

مرحلهٔ تأیید هویت[۱۶] عبارت است از تطبیق ویژگیهای مورد ادعای یک شخص بر ویژگیهای موجود او در پایگاه داده‌ها که یک فرایند یک به یک است.

سیستمهای امنیتی مبتنی بر زیست‌سنجی بنا به انتخاب به وجود آورنده، به جای مرحلهٔ تأیید هویت می‌توانند مرحلهٔ دیگری را که بازشناسی[۱۷] نامیده می‌شود جایگزین کنند. در این روش نیاز نیست که درخواست کننده ادعای هویت شخص خاصی را بنماید بلکه سیستم ویژگیهای او را با تمامی رکوردهای موجود مقایسه می‌کند و در صورت تطابق با یکی از آنها او را به عنوان شخص دارای ویژگیهای موجود در رکورد یافت شده بازشناسی می‌کند که این فرایند یک پردازش یک به چند را شکل می‌دهد.

سیستمهای تشخیص هویت زیستی معمول غالباً شامل اجزای زیر می‌باشند:

الف)گیرندهٔ اطلاعات[۱۸]: زیرسیستمی است که گرفتن نمونه‌های زیست‌سنجی (صوتی، تصویری و...) را بر عهده دارد. ویژگیهای خاص استخراج شده از نمونه‌ها قالبهایی را برای مقایسهٔ بعدی تشکیل می‌دهند. این فرایند باید سریع و ساده بوده در عین حال قالبهایی با کیفیت خوب را تولید کند.

---

[16] verification
[17] identification
[18] capture

ب) ذخیره کننده[19]: قالبهای به دست آمده باید برای مقایسهٔ بعدی ذخیره شوند. این زیر سیستم می‌تواند جزئی از وسیلهٔ گیرندهٔ اطلاعات سیستم باشد و یا در یک سرور مرکزی قابل دستیابی توسط یک شبکه جای گیرد. جایگزین دیگر، یک شناسهٔ قابل حمل نظیر یک کارت هوشمند[20] است. هر کدام از انتخابهای فوق مزایا و مشکلات خاص خود را دارد.

ج) مقایسه گر[21]: اگر سیستم زیست‌سنجی در مقام بازشناسی افراد به کار گرفته شود باید هویت شخص با قالب ذخیره شدهٔ مورد ادعای او مقایسه شود. در بعضی سیستمها ممکن است امکان بروزآوری خودکار قالب مورد مراجعه پس از هر تطبیق درست وجود داشته باشد. این امر به سیستم توانایی سازگاری با تغییرات تدریجی کوچک در ویژگیهای کاربر را می‌دهد.

د) اتصالات[22]: غالباً برای ایجاد ارتباط بین گیرندهٔ اطلاعات، ذخیره کننده و مقایسه‌گر نیاز به اتصالات لازم وجود دارد. غالباً سیستمهای زیست‌سنجی نیازمند شبکه و رابطهای برنامه‌نویسی مورد نیاز برای ایجاد اتصال بین اجزاء می‌باشند. امنیت و کارایی، عناصر کلیدی برای این جزء می‌باشند.

۴– ۱ ارزیابی کارایی سیستمهای امنیتی مبتنی بر زیست‌سنجی

موضوع مهمی که در پذیرش سیستمهای زیست‌سنجی از اهمیت شایان توجهی برخوردار است تعیین کارایی هر یک از اجزاء و کل سیستم زیست‌سنجی به روشی قابل اعتماد و هدفمند است.

برای تعیین کارایی سیستمهای امنیتی مبتنی بر زیست‌سنجی معیارهای ویژه‌ای به کار گرفته می‌شوند. در این کاربردها تعدادی کاربر (سرویس‌گیرنده)[23] به سیستم وارد می‌شوند و

[19] storage
[20] smart card
[21] comparison
[22] interconnections
[23] client

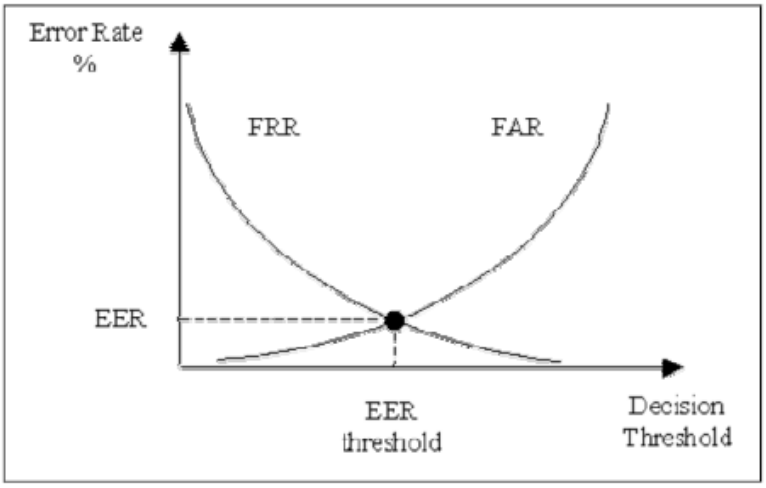
متقلب[۲۴] به عنوان شخصی تعریف می‌شود که مدعی هویت شخص دیگری است. متقلب ممکن است به عنوان کاربر در سیستم وجود داشته باشد و عمل وی ممکن است عمدی یا غیرعمدی باشد. عمل تأیید هویت باید کاربران را بپذیرد و متقلبان را رد کند.

نرخ پذیرش نادرست[۲۵] (اف. ای. آر) به عنوان نسبت تعداد متقلبانی که به اشتباه توسط سیستم پذیرفته شده‌اند به تعداد کل متقلبان آزمایش شده تعریف گردیده، به صورت درصد بیان می‌شود. این نرخ، احتمال پذیرش متقلبان را توسط سیستم بیان می‌کند و باید در سیستمهای نیازمند امنیت بالا کمینه شود.

نرخ عدم پذیرش نادرست[۲۶] (اف. آر. آر) به عنوان نسبت تعداد کاربران سیستم که به اشتباه توسط سیستم پذیرفته نشده‌اند به تعداد کل کاربران مورد آزمایش قرار گرفته تعریف گردیده، به صورت درصد بیان می‌شود. این نرخ، احتمال عدم پذیرش کاربران مجاز را توسط سیستم بیان می‌کند و باید به صورت ایده‌آل مخصوصاً در سیستمهایی که در آنها کاربر در صورت عدم پذیرش از دسترسی به سیستم محروم می‌شود کمینه گردد.

روند تشخیص هویت مبتنی بر زیست‌سنجی دربردارندهٔ محاسبهٔ فاصلهٔ قالب ذخیره شده و نمونهٔ حاضر است. تصمیم برای پذیرش یا رد نمونهٔ حاضر بر اساس یک آستانهٔ[۲۷] از پیش تعریف شده اتخاذ می‌گردد. بنابراین واضح است که کارایی سیستم به شدت وابسته به انتخاب این آستانه است و این امر موجب ایجاد یک بده‌بستان بین نرخ پذیرش نادرست و نرخ عدم پذیرش نادرست می‌گردد. نرخ خطای برابر[۲۸] (ای.ای.آر) به صورت آستانهٔ برابری این دو نرخ تعریف می‌شود و غالباً به عنوان یک ویژگی نشان دهندهٔ کارایی سیستم مطرح می‌گردد. شکل شمارهٔ ۲ -۱نشان دهندهٔ رابطهٔ سه پارامتر تعریف شده برای یک سیستم نمونه است.

[24] imposter
[25] False Acceptance Rate (FAR)
[26] False Reject Rate (FRR)
[27] threshold
[28] Equal Error Rate (EER)



شکل شمارهٔ ۲-۱ FAR، FRR و ERR برای یک سیستم نمونه (منبع شمارهٔ ۱)

پارامتر مهم دیگر کارایی، زمان تشخیص هویت[۲۹] است که به صورت زمان متوسط صرف شده برای فرایند تشخیص هویت تعریف می‌شود. این زمان شامل زمان لازم برای گرفتن نمونهٔ حاضر نیز می‌باشد.

در حالی که بعضی از عرضه‌کنندگان سیستمهای امنیتی مبتنی بر زیست‌سنجی برای محصولاتشان پارامترهای کارایی فوق را در شرایط آزمایشگاهی بیان می‌کنند پارامترهای کارایی قابل طرح در جهان واقعی برای سنجش کارایی واقعی این گونه سیستمها به ندرت وجود دارند. علت این امر این واقعیت است که به حساب آوردن همهٔ پیچیدگیهای ممکن جهان واقعی تأثیر گذار بر سیستمهای زیست‌سنجی تقریباً غیر ممکن است. به عنوان نمونه زمان واقعی تشخیص هویت به شدت وابسته به میزان آموزش کاربر، محیط عملیاتی و شرایط روانی کاربر همچون میزان فشار روحی اوست. مشخصات ارائه شده توسط عرضه‌کننده را باید به دید راهنماهای نه چندان متناسب با دنیای واقعی نگریست.

[29] verification time

بخش دوم

سیستمهای امنیتی مبتنی بر تشخیص گوینده

۲–۱ تعریف و کاربردها

تشخیص گوینده[۳۰] عبارت است از فرایند تشخیص خودکار هویت شخص صحبت‌کننده بر اساس اطلاعات یکتای موجود در موج صوتی صحبت او.

این فن‌آوری امکان تشخیص هویت شخص گوینده و در نتیجه امکان کنترل دسترسی او در هنگام استفاده از خدماتی همانند شماره‌گیری صوتی، بانکداری تلفنی، خرید تلفنی، خدمات دسترسی به پایگاه داده‌ها، خدمات اطلاعاتی، پست الکترونیکی صوتی، کنترل امنیتی برای ورود به قلمروهای اطلاعاتی محرمانه و دسترسی از راه دور به کامپیوترها را فراهم می‌آورد. علاوه بر موارد فوق که عموماً با کامپیوتر و کاربران آن سروکار دارند این فن‌آوری در مسائل قضایی نیز کاربردهای خاص خود را دارد.

۲– ۲ انواع سیستمهای تشخیص گوینده

سیستمهای تشخیص گوینده از لحاظ روش استفاده، همانند آنچه برای کلیهٔ سیستمهای امنیتی مبتنی بر زیست‌سنجی در فصل پیش بیان شد[۳۱]، عموماً در دودستهٔ سیستمهای تأیید هویت گوینده[۳۲] و سیستمهای بازشناسی هویت گوینده[۳۳] قرار می‌گیرند.

در یک سیستم تأیید هویت گوینده، شخص عموماً با انتخاب یا وارد کردن نام یکی از کاربران خاص سیستم ادعا می‌کند که او همان کاربر ثبت‌شدهٔ سیستم است. در این حالت سیستم وظیفه دارد ویژگیهای صوتی شخص مدعی را با ویژگیهای صوتی ذخیره شدهٔ کاربر ثبت شدهٔ مورد ادعا مقایسه نموده و با استفاده از نتیجهٔ به دست آمده ادعای شخص را بپذیرد یا رد کند.

---

[30] speaker recognition

[۲] به صفحهٔ ۴ رجوع شود.

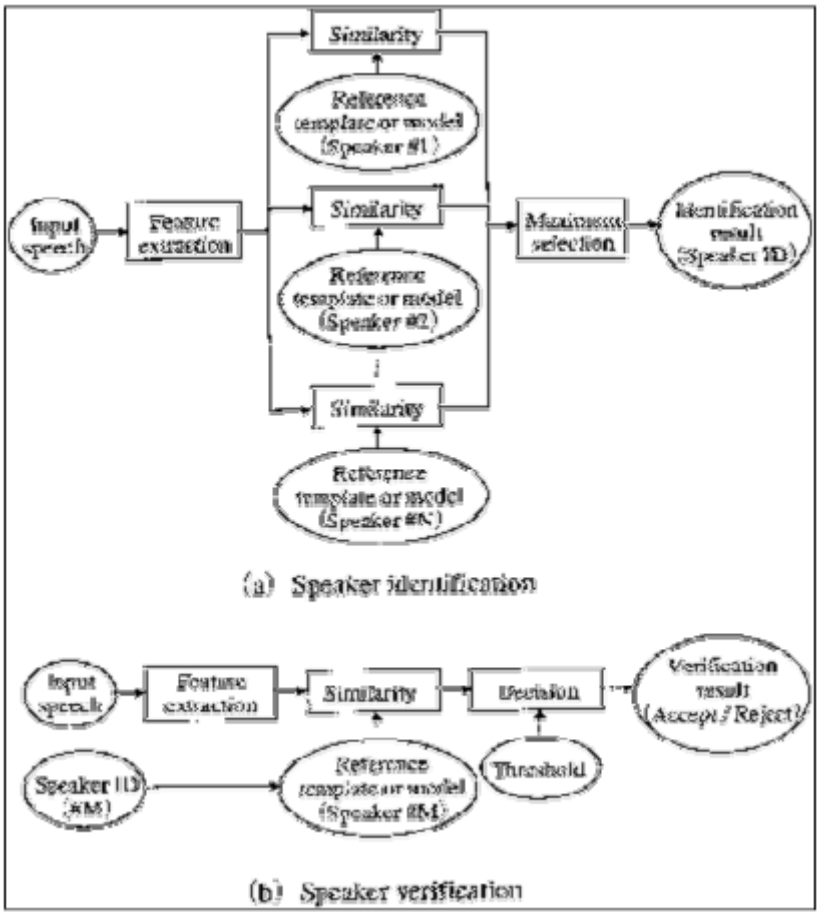
[32] speaker verification systems

[33] speaker identification systems

در یک سیستم بازشناسی هویت گوینده، شخص صحبت کننده ادعای هویت یک کاربر خاص ثبت شده را نمی‌نماید و این سیستم است که وظیفه دارد که او را در میان کاربران ثبت شدهٔ سیستم بازشناسی نماید و یا تشخیص دهد که ویژگیهای صوتی او با هیچ یک از کاربران ثبت شده همخوانی ندارد.

به نظر می‌رسد در آینده کاربردهای سیستمهای نوع دوم در سیستمهای بزرگ چند کاربره چشمگیرتر از کاربردهای سیستم نوع اول باشد[۳۴]، هر چند که در اساس این دو سیستم تفاوتهای چشمگیری مشاهده نمی‌شود.

شکل شمارهٔ ۱-۲ ساختار اساسی این دو نوع سیستم تشخیص گوینده را به تصویر می‌کشد.



شکل شماره ۱-۲ ساختار اساسی سیستمهای بازشناسی هویت و تأیید هویت گوینده (منبع شمارهٔ ۱)

[۱] این عقیده، نظر منبع شمارهٔ ۲ است (ر.ک. صفحهٔ ۱۹ آن منبع)

سیستمهای تشخیص گوینده از دیدگاه دیگری به دو دستهٔ سیستمهای تشخیص گویندهٔ وابسته به متن[۳۵] و سیستمهای تشخیص گویندهٔ مستقل از متن[۳۶] تقسیم می‌شوند. روش اول نیازمند آن است که گوینده کلمات کلیدی یا جمله‌های ثابتی را چه در مرحلهٔ یادگیری و چه در آزمونهای تشخیصی بیان کند، در حالی که دومی وابسته به جمله یا کلمهٔ خاصی نیست.

هر دو روش دارای یک مشکل هستند و آن این است که می‌توان از صدای ضبط شدهٔ کاربران ثبت‌شده برای ورود به سیستم استفاده نمود و به آسانی سیستم را فریب داد. برای غلبه بر این مشکل روشهایی وجود دارند مثلاً می‌توان از یک مجموعهٔ کوچک از کلمات مانند ارقام به عنوان کلمات کلیدی استفاده نمود و در هر زمان به صورت تصادفی از کاربر خواست که یک دنباله از آنها را بیان کند. حتی این روش هم کاملاً قابل اطمینان نیست چرا که می‌تواند با استفاده از تجهیزات پیشرفتهٔ الکترونیکی که توانایی تولید دنباله‌های عبارات را دارند فریب داده شود. سیستمهای دارای ساختار اخیر به سیستمهای تشخیص گویندهٔ اعلان متن[۳۷] (متن تولید شده توسط ماشین) معروفند.

۳– ۲ روشهای پیاده‌سازی

تقریباً در تمامی سیستمهای تشخیص هویت با استفاده از فرایندی که به تشخیص الگو[۳۸] شهرت دارد شباهت هر زوج نمونه نمره‌گذاری می‌شود. استفاده از این روش نیازمند وجود دسته‌ای از خصایص منحصر به فرد و قابل مقایسه که از ویژگی انتخاب شده به عنوان ورودی سیستم استخراج شده می‌باشد.

[35] text-dependent speaker recognition systems
[36] text-independent speaker recognition systems
[37] text-prompted speaker recognition systems
[38] pattern recognition

ویژگیهای فیزیکی افراد نظیر ساختار انـدامهای صـوتی، انـدازهٔ چالـهٔ بینـی و ویژگیهـای تارهای صوتی منحصر به فرد بوده و از طریق الگوریتمهای پردازش سیگنال به صورت پارامترهـای خصیصه‌ای[۳۹] یا مجموعهٔ خصایص[۴۰] قابل استخراج می‌باشند. این حقیقت پایهٔ روشهای پیاده‌سازی سیستمهای تشخیص صحبت می‌باشند.

مهمترین گلوگاه سیستمهای تشخیص گوینده (و بـه تبـع هـم خـانواده بـودن مهمتـرین گلوگاه سیستمهای تشخیص صحبت) نحوهٔ عملکـرد آنهـا در مکانهـای دارای شـرایط متفـاوت بـا شرایط آزمایشگاهی که از ویژگیهای عمدهٔ آنها می‌تـوان بـه حضـور نـویز در سیسـتم اشـاره کـرد می‌باشد. برای غلبه بر این مشکل از روشهای هنجارسازی[۴۱] استفاده می‌گردد که ایـن روشـها نیـز انواع مختلفی دارند و در سیستمهای تجاری موجود، اغلب نمود پیدا می‌کنند.

[39] feature parameters
[40] feature set
[41] normalization

بخش سوم

پردازش صوت : پیش‌زمینه‌های تئوری

۱-۳ دستگاه شنوایی انسان

پردازش صوت محدوده‌های گوناگونی را در بر می‌گیرد که همه به منظور ارائهٔ صدا به شنوندگان انسانی ابداع شده‌اند. سه محدودهٔ تکثیر موسیقی با کیفیتی به خوبی اصل همانند آنچه در سی‌دی‌های صوتی وجود دارد ،ارتباط صوتی از راه دور که نام دیگر شبکهٔ تلفنی است و ،ترکیب صحبت[۱] که در آن کامپیوترها الگوهای صوتی انسان را تولید کرده یا تشخیص می‌دهند از دیگر قلمروهای دانش پردازش صوت مهم‌ترند. با وجود این که اهداف و مسائل این کاربردها متفاوتند همگی در یک نقطهٔ مشترک به هم می‌رسند و آن گوش انسان است.

گوش انسان یک عضو به گونه‌ای فزاینده پیچیده است. قضیه وقتی پیچیده‌تر می‌شود که اطلاعات ارسالی از دو گوش در یک شبکهٔ پیچیدهٔ گیج کننده که همانا مغز انسان باشد با هم ترکیب می‌شوند. به یاد داشته باشیم که بیان فوق یک گذر کلی بر قضیه است و تعداد زیادی از پدیده‌ها و آثار دقیق مرتبط با گوش انسان هنوز به درستی درک نشده‌اند.

شکل ۱-۳ قسمت اعظم ساختارها و پردازشهایی را که گوش انسان را در بر دارند به تصویر می‌کشد. گوش خارجی از دو بخش تشکیل شده است: نرمی پوست قابل مشاهده و غضروف متصل به کنار سر و کانال گوش که لوله‌ایست به قطر تقریبی ۰.۵ سانتیمتر و تا حدود ۳ سانتیمتر در داخل سر فرو می‌رود. این ساختارها صداهای محیط را به بخشهای حساس گوش میانی و گوش داخلی که در درون استخوانهای جمجمه محافظت می‌شود راهبری می‌کنند. در انتهای کانال گوش یک ورقهٔ نازک از نسوج که پردهٔ صماخ[۴۲] یا طبل گوش نامیده می‌شود کشیده شده است. امواج صدا با برخورد به پردهٔ صماخ باعث لرزش آن می‌شوند. گوش میانی مجموعه‌ای از استخوانهای کوچک است که لرزش مزبور را به حلزون گوش[۴۳] (گوش داخلی) انتقال می‌دهند و در آنجا این لرزشها تبدیل به ضربه‌های عصبی می‌گردند. حلزون گوش یک لولهٔ پر از مایع است که به

[42] tympanic membrane
[43] cochlea

زحمت قطر آن به ۲ میلیمتر و طول آن به ۳ سانتیمتر می‌رسد. اگر چه حلـزون گـوش در شـکل شمارهٔ ۱-۳ به صورت یک لولهٔ مستقیم نشان داده شده اما در واقع به دور خودش هماننـد صـدف حلزون پیچ خورده است و وجه تسمیهٔ آن که ریشه در کلمه‌ای یونانی به معنای حلـزون دارد نیـز این واقعیت است.

وقتی یک موج صوتی سعی دارد از هوا وارد مایع شود تنها کسر کوچکی از آن از بـین دو محیط عبور می‌کند و باقیماندهٔ انرژی آن بازتابیده می‌شود. دلیل این امر مقاومت مکـانیکی پـایین هوا (ناشی از پایین بودن میزان فشار صوتی و سرعت بالای ذرات هوا که بـه نوبـهٔ خـود از چگـالی پایین و تراکم‌پذیری بالای آنها نشأت می‌گیرد) در برابر مقاومت مکـانیکی بـالای مـایع اسـت. بـه عبارت ساده‌تر دلیل این امر مشابه دلیل این موضوع است که برای ایجاد مـوج بـا دسـت در درون آب به تلاش بیشتری به نسبت انجام این کار در هـوا نیازمنـدیم. تفـاوت موجـود باعـث بازتـابش قسمت اعظم صوت در مرز هوا/مایع می‌گردد.

گوش میانی یک شبکهٔ تطبیق مقاومت[۴۴] است که کسر انرژی صوتی وارد شـده بـه مـایع گوش داخلی را زیاد می‌کند. برای نمونه ماهی پردهٔ صماخ یا گوش میانی ندارد چرا که نیـازی بـه شنیدن در هوا ندارد. تغییر شدت، بیشتر ناشی از تفاوت مساحت پردهٔ صماخ (کـه صـدا را از هـوا دریافت می‌کند) و دریچه بیضوی[۴۵] (که مطابق شکل ۱صدا را بـه داخـل مـایع انتقـال مـی‌دهـد) می‌باشد. مساحت پردهٔ صماخ حدوداً ۶۰ میلیمتر مربع است حال آن که دریچهٔ بیضـوی حـدوداً ۴ میلیمتر مربع مساحت دارد. از آنجا که فشار برابر است با نسـبت نیـرو بـه مسـاحت، ایـن تفـاوت مساحت فشار موج صدا را حدوداً ۱۵ برابر افزایش می‌دهد.

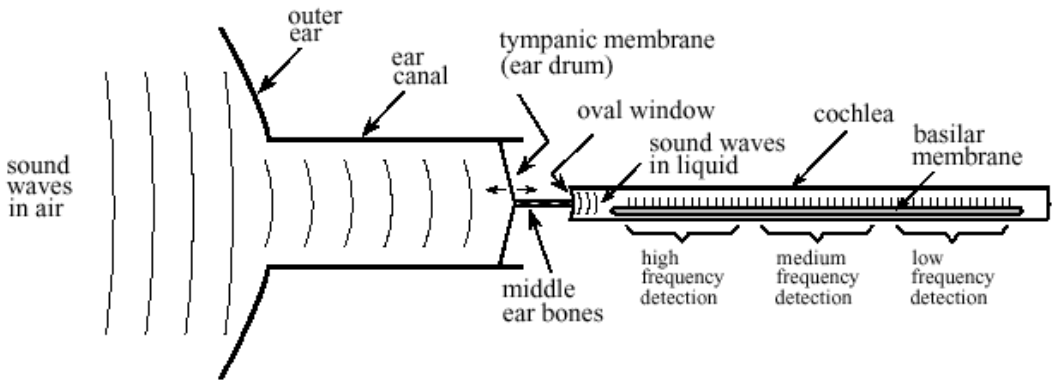
در داخل حلزون گوش پردهٔ اصلی[۴۶] قرار دارد که ساختاری را برای ۱۲۰۰۰ سلول حسـی که شکل‌دهندهٔ عصب حلزونی است ایجاد می‌کند. پردهٔ اصلی در نزدیکـی دریچـهٔ بیضـوی بسـیار سفت است و در انتهای دیگر انعطاف‌پذیرتر است که این امر به این عضو کمک می‌کند تا به عنوان

[44] impedance matching
[45] oval windows
[46] basilar membrane

تحلیلگر طیف فرکانسی عمل کند. وقتی پردهٔ اصلی در معرض یک سیگنال با فرکانس بالا قرار می‌گیرد در قسمت سفت‌تر طنین می‌اندازد که سبب تحریک سلولهای عصبی نزدیک به دریچهٔ بیضوی می‌گردد. به همین ترتیب فرکانسهای پایین موجب تحریک انتهای دورتر پردهٔ اصلی می‌شوند. این امر موجب پاسخگویی رشته‌های خاص عصب حلزونی در برابر فرکانسهای خاص می‌گردد. این سازوکار **اصل مکان**[۴۷] نامیده می‌شود و در سراسر مسیر به سمت مغز حفظ می‌شود.

طرح کدگذاری اطلاعات دیگری نیز در شنوایی انسان به کار می‌رود که **اصل رگبار**[۴۸] نامیده می‌شود. سلولهای عصبی اطلاعات را با تولید پالسهای الکتریکی کوچکی که پتانسیل کنش[۴۹] نامیده می‌شوند انتقال می‌دهد. یک سلول عصبی واقع بر پردهٔ پایینی می‌تواند اطلاعات صوتی را با تولید یک پتانسیل کنش در پاسخ هر سیکل لرزش کدگذاری کند. برای نمونه یک موج صدای ۲۰۰ هرتزی می‌تواند توسط یک نورون ایجاد کنندهٔ ۲۰۰ پتانسیل کنش در ثانیه نشان داده شود. در هر صورت این روش تنها در فرکانسهای زیر حدوداً ۵۰۰ هرتز – بالاترین سرعت ممکن تولید پتانسیل کنش در نورونها – به کار می‌آید. گوش انسان برای غلبه بر این مشکل به نورونها اجازه می‌دهد که برای انجام این کار دسته‌جمعی عمل کنند. برای نمونه یک صدای ۳۰۰۰ هرتزی می‌تواند توسط ده سلول عصبی که هر کدام ۳۰۰ ضربه در ثانیه علامت می‌دهند نشان داده شود. این پدیده بازهٔ کارایی اصل رگبار را تا ۴ کیلوهرتز گسترش می‌دهد که بالاتر از بازهٔ عملیاتی اصل مکان می‌باشد.

[47] place principle
[48] volley principle
[49] action potential

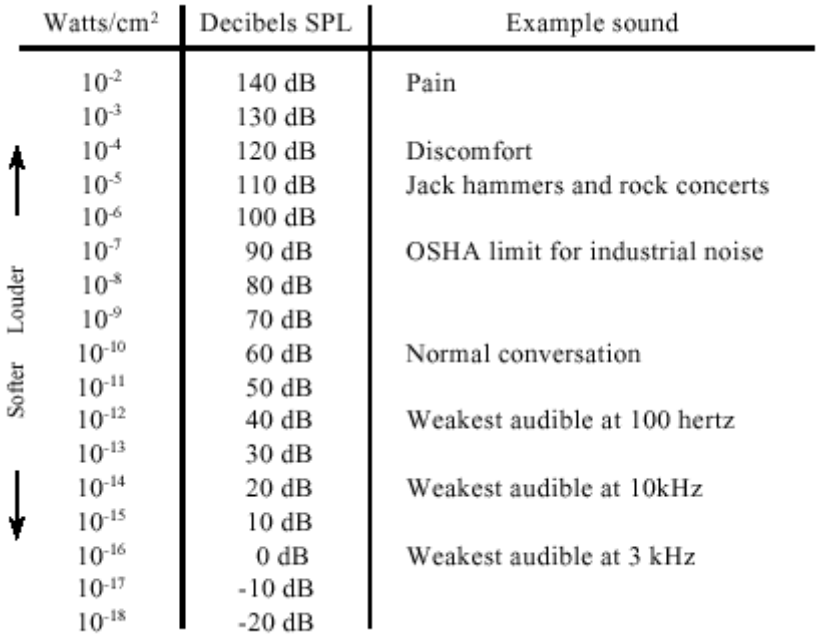


شکل شمارهٔ ۱-۳ توضیحات مربوط به شکل: نمودار کارکردی گوش انسان.

گوش خارجی امواج صوتی را از محیط می‌گیرد و آنها را به سوی پردهٔ صماخ (طبل گوش) که ورقهٔ نازکی از بافت است و هماهنگ با شکل موج هوا می‌لرزد راهبری می‌کند. استخوانهای گوش میانی (استخوانهای چکشی، سندانی و رکابی) این لرزشها را به دریچهٔ بیضوی که پرده‌ای منعطف واقع در حلزون گوش پر از مایع است انتقال می‌دهند. در داخل حلزون گوش پردهٔ اصلی قرار دارد که ایجاد کنندهٔ ساختاری برای ۱۲۰۰۰ سلول عصبی شکل‌دهندهٔ عصب حلزون گوش است. بسته به سفتی متغیر پردهٔ پایینی، هر سلول فقط به بازهٔ کوچکی از فرکانسهای صدا پاسخ می‌دهد که این پدیده گوش را تبدیل به یک تحلیلگر طیف فرکانسی می‌نماید.

شکل شمارهٔ ۲-۳ رابطهٔ بین شدت صدا و بلندی مشاهده شده را نشان می‌دهد. غالباً شدت صدا را با یک اندازهٔ لگاریتمی که دسی‌بل اس.پی.ال.[۵۰] (سطح توان صدا) نامیده می‌شود نشان می‌دهند. در این معیار ۰ دسی‌بل اس.پی.ال موج صدایی با قدرت ده به توان منفی شانزده وات بر سانتیمتر مربع است که حدوداً ضعیف‌ترین صدای قابل تشخیص توسط گوش انسان است. صحبت معمولی حدوداً ۶۰ دسی‌بل اس.پی.ال است و صدایی با شدت ۱۴۰ دسی‌بل اس.پی.ای برای گوش دردناک و زیان‌آور است.

[50] decibel SPL



شکل شمارهٔ ۲ - ۳ واحدهای شدت صدا.

شدت صدا به صورت توان بر واحد مساحت تعریف می‌شود (مثلاً وات بر سانتیمتر مربع) یا به صورت معمول‌تر با استفاده از یک اندازهٔ لگاریتمی که دسی‌بل اس.پی.ال خوانده می‌شود. همچنان که این جدول نشان می‌دهد قوهٔ شنوایی انسان بیشتر به صداهای بین ۱کیلوهرتز تا ۴ کیلوهرتز حساس است.

اختلاف بلندترین و ضعیف‌ترین صداهایی که انسان می‌تواند بشنود ۱۲۰ دسی‌بل است که از لحاظ دامنه معادل بازه‌ای حدود یک میلیون است. شنونده تغییر بلندی صدا را وقتی صدا حدود ۱ دسی‌بل (۱۲٪ در دامنه) تغییر کند تشخیص می‌دهد به عبارت دیگر تنها ۱۲۰ سطح بلندی صدا از ملایم‌ترین نجوا تا بلندترین تندر قابل تشخیص است. حساسیت گوش آنقدر جالب توجه است که هنگام شنیدن به ضعیف‌ترین صداها پردهٔ صماخ به اندازه‌ای کمتر از قطر یک ملکول به لرزش درمی‌آید!

احساس بلندی صدا با توان صدا رابطهٔ توانی با نمای 1/3 دارد. به عنوان نمونه اگر شما توان صدا را ده برابر کنید شنوندگان آن صدا دو برابر شدن بلندی صدا را احساس و گزارش می‌کنند.

این مسأله یک مشکل بزرگ برای حذف صداهای محیطی ناخواسته به وجود می‌آورد. برای نمونه فرض کنید که شما ۹۹٪ دیوار را با عایق صوتی پوشانده‌اید و تنها ۱٪ که مربوط به درها، گوشه‌ها، منافذ و... هستند باقی مانده‌اند. با وجود آن که توان صدا تا اندازهٔ ۱٪ مقدار اولیهٔ آن کاسته شده بلندی صدا تنها به اندازهٔ ۲۰٪ کاهش پیدا کرده‌است.

بازهٔ شنیداری انسان بین ۲۰ هرتز تا ۲۰ کیلوهرتز در نظر گرفته می‌شود، حال آن که بیشتر صداهای قابل حس در بازهٔ ۱ کیلوهرتز تا ۴ کیلوهرتز قرار دارند. برای نمونه شنوندگان می‌توانند صدایی به میزان صفر دسی‌بل را در فرکانس ۳ کیلوهرتز بشنوند حال آن که برای شنیدن یک صدای ۱۰۰ هرتزی حداقل مقدار آن باید ۴۰ دسی‌بل باشد. شنوندگان می‌توانند بگویند که دو صدا متفاوتند اگر فرکانس آنها بیش از حدود ۰.۳٪ در ۳ کیلوهرتز متفاوت باشد. به عنوان نمونه کلیدهای کنار هم در پیانو به اندازهٔ حدود ۶٪ تفاوت فرکانس دارند.

مهم‌ترین مزیت داشتن دو گوش تشخیص جهت صداست. شنوندگان انسانی می‌توانند تفاوت بین دو منبع صدا را که فاصله‌ای به کمی ۳ درجه دارند (حدوداً برابر با عرض یک انسان در فاصلهٔ ده متری) تشخیص دهند. این اطلاعات جهتی به دو روش جداگانه به دست می‌آیند. اولاً فرکانسهای حدوداً بالای ۱ کیلوهرتز به شدت زیر سایهٔ سر قرار می‌گیرند. به بیان دیگر گوشی که به منبع نزدیک‌تر است سیگنال قوی تری را به نسبت گوشی که در جهت مخالف دارد دریافت می‌کند. روش دیگر تشخیص جهت آن است که گوش دورتر به خاطر فاصلهٔ بیشترش از منبع صدا را کمی دیرتر از گوش نزدیک‌تر دریافت می‌کند. به واسطهٔ اندازهٔ معمول سر (حدوداً ۲۲ سانتیمتر) و سرعت صوت (حدود ۳۴۰ متر در ثانیه) تفاوت‌گذاری زاویه‌ای سه درجه دقت زمانی حدود ۳۰

میکروثانیه نیاز دارد. چون این فاصلهٔ زمانی نیازمند اصل رگبار است این روش جهت‌یابی برای صداهای دارای فرکانس کم‌تر از حدود ۱ کیلوهرتز به کار می‌رود.

در حالی که قوهٔ شنوایی انسان می‌تواند جهت صدا را تشخیص دهد در نشخیص فاصلهٔ منبع صدا مشکل دارد. این امر بدان علت است که چیزهای کمی در موج صدا وجود دارد که اطلاعات این گونه را در اختیار بگذارد. شنوایی انسان به صورت ضعیفی در می‌یابد که منابع صداهای با فرکانس بالا نزدیکند و صداهای با فرکانس پایین از فاصلهٔ دورتری پخش می‌شوند. این به آن دلیل است که صداها در فاصله‌های دور از میزان فرکانسشان کاسته می‌شود. پژواک روش ضعیف دیگری برای تشخیص فاصله است و با استفاده از آن مثلاً می‌توان ابعاد یک اتاق را حدس زد. برای نمونه صداهای موجود در یک تالار بزرگ پژواکهایی با وقفهٔ ۱۰۰ میلی ثانیه دارند، حال آن که برای یک دفتر کار کوچک این مقدار ۱۰ میلی ثانیه است. بعضی از موجودات با استفاده از دستگاه طبیعی تشخیص فاصلهٔ صوتی[۵۱] مسألهٔ فاصله‌یابی را حل کرده‌اند. مثلاً خفاشها و دلفینها صداهایی مثل تیک و جیغ تولید می‌کنند که از سوی اشیاء نزدیک بازتابیده می‌شوند. با اندازه‌گیری میزان وقفهٔ بازتاب این صداها این جانوران می‌توانند با دقت ۱سانتیمتر اشیاء را مکانیابی کنند. تجربیات نشان داده‌اند که بعضی انسانها به خصوص نابینایان تا حد کمی از روش مکانیابی با استفاده از پژواک استفاده می‌کنند.

۲–۳ ویژگیهای امواج صوتی

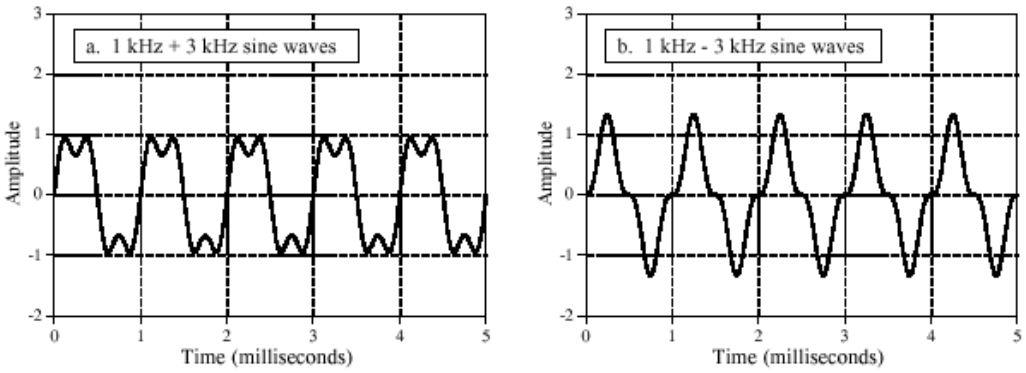
غالباً برای درک یک صوت پیوسته مثل نت یک ابزار موسیقیایی سه بخش مجزا را باید تشخیص داد: بلندی صدا، زیری یا بمی صدا (پیچ)[۵۲] و طنین[۵۳] صدا. بلندی همانگونه که قبلاً توضیح داده شد معیاری برای شدت موج صوتی است. پیچ، فرکانس جزء اصلی صدا – فرکانسی تکرار موج صوتی توسط خودش – می‌باشد.

[51] sonar
[52] pitch
[53] timbre

طنین صدا از دو جزء قبلی پیچیده‌تر است و با تعیین محتوای همساز[۵۴] صدا تعیین می‌گردد. شکل شمارهٔ ۳ -۳ دو موج را که هر دو از جمع یک موج سینوسی یک کیلوهرتزی با دامنهٔ یک و یک موج سینوسی سه کیلوهرتزی با دامنهٔ یک دوم به وجود آمده‌اند نشان می‌دهد. تفاوت آنها در آن است که در شکل b جزء با فرکانس بالاتر ابتدا معکوس شده و سپس با موج دوم جمع شده است. علی‌رغم موجهای در دامنهٔ زمان بسیار متفاوت این دو صوت یکسان به نظر می‌رسند. این به خاطر آن است که شنوایی انسان بر اساس دامنهٔ فرکانسهاست و نسبت به فاز آنها بسیار غیر حساس است. شکل موج صوتی در دامنهٔ زمان فقط به صورت غیر مستقیم با شنوایی رابطه دارد و معمولاً در سیستمهای صوتی در نظر گرفته نمی‌شود.

عدم حساسیت گوش به فاز صدا با توجه به روش پخش شدن آن در محیط قابل درک است. فرض کنید که شما در یک اتاق به صحبتهای فردی گوش می‌دهید. بیشتر صداهایی که گوش شما دریافت می‌کند حاصل بازتاب صدای اصلی از دیوارها، سقف و کف اتاق است. از آنجا که انتشار صدا بستگی به فرکانس آن دارد و میرایی ،بازتاب و مقاومت در برابر صدا بر روی آن تأثیرگذار است فرکانسهای متفاوتی از مسیرهای متفاوت به گوش می‌رسد. این به این معنی است که وقتی شما جای خود را در اتاق عوض می‌کنید فاز هر یک از فرکانسها تغییر می‌کند. چون گوش این تغییر فازها را نادیده می‌انگارد با وجود تغییر مکان شما تغییری در صدای شخص صحبت کننده احساس نمی‌کنید. از دیدگاه فیزیکی فاز یک سیگنال صدا در هنگام پخش در یک محیط پیچیده به صورت تصادفی تغییر می‌کند. از طرف دیگر گوش به فاز صدا غیر حساس است زیرا این جزء دارای اطلاعات قابل استفادهٔ بسیار کمی می‌باشد.

[54] harmonic content

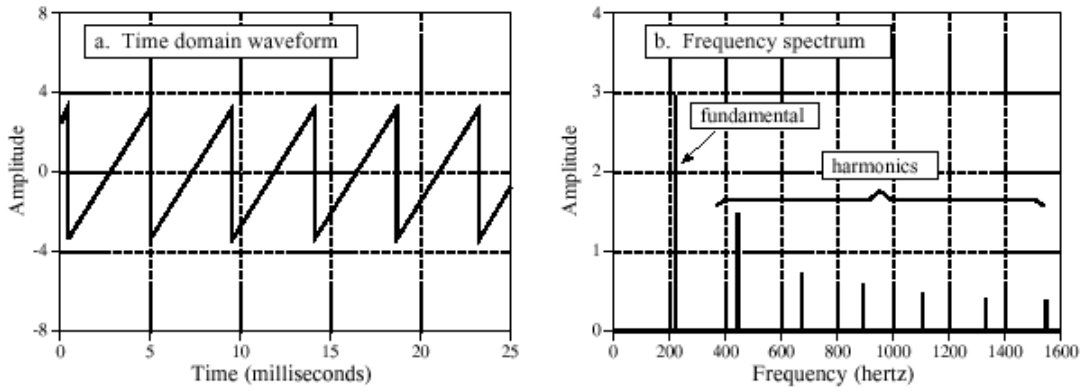


شکل شمارهٔ ۳ –۳ تشخیص فاز توسط گوش انسان.

گوش انسان نسبت به فاز نسبی سینوسیهای مرکب بسیار غیر حساس است. برای نمونه این دو موج یکسان به نظر خواهند رسید، زیر دامنهٔ اجزاء آنها یکسان است اگر چه فاز نسبی آنها متفاوت است.

در حالت کلی نمی‌توان گفت که گوش نسبت به فاز کاملاً ناشنواست. چرا که تغییر فاز می‌تواند باعث تغییر آرایش زمانی یک سیگنال صوتی شود. اما چنین امری یک پدیدهٔ نادر است که در محیطهای شنیداری طبیعی اتفاق نمی‌افتد.

فرض کنید از یک نوازندهٔ ویولون خواسته‌ایم نتی را بنوازد. وقتی که موج صوتی ایجاد شده بر روی اسیلوسکوپ نشان داده شود یک موج دندانه‌اره‌ای مانند شکل شمارهٔ ۴-۳ (a) مشاهده می‌شود. شکل شمارهٔ ۴-۳ (b) نشان می‌دهد که این صوت چگونه توسط گوش دریافت می‌شود. گوش یک فرکانس اساسی (در مثال شکل ۲۲۰ هرتز) را و همسازهایی را در ۴۴۰، ۶۶۰، ۸۸۰ و... هرتز دریافت می‌کند. اگر این نت بر روی ابزار دیگری نواخته شود گوش هنوز هم همان ۲۲۰ هرتز (همان فرکانس اساسی) را دریافت می‌کند. و از این لحاظ دو صوت مشابهند که گفته می‌شود این دو صوت پیچ یکسانی دارند ولی چون دامنهٔ همسازها متفاوت است دو صوت یکسان نیستند و گفته می‌شود که طنین دو صوت متفاوت است.



شکل شمارهٔ ۴ – ۳ موج صوتی ویولن.

ویولن موج دندانه‌اره‌ای ایجاد می‌کند (شکل a)، صدای دریافت شده شامل فرکانس اساسی و همسازهای آن است (شکل b)

اغلب گفته می‌شود که طنین صدا از روی شکل موج صوتی تعیین می‌گردد. این مسأله درست است ولی کمی گمراه کننده است. احساس طنین صدا از روی میزان هارمونیکهای تشخیص داده شده توسط گوش تعیین می‌گردد. در حالی که هارمونیکها از روی شکل موج صوتی تعیین می‌گردد عدم حساسیت گوش به فاز رابطه را بسیار یک طرفه می‌کند. به همین دلیل هر موج صوتی فقط یک طنین دارد حال آن که یک زنگ خاص متعلق به تعداد بی‌نهایتی از موجهای صوتی است.

گوش بیشتر برای شنیدن هارمونیکهای اساسی تنظیم شده است. اگر یک شنونده به صدایی که حاصل ترکیب دو موج صوتی سینوسی ۱ کیلوهرتز و ۳ کیلوهرتز است گوش دهد آن را مطلوب و طبیعی توصیف خواهد کرد حال آن که اگر از موجهای ۱ کیلوهرتزی و ۳.۱ کیلوهرتزی استفاده شود برای شنونده شکایت برانگیز خواهد بود. این مسأله اساسی برای اندازه‌ها و اختلافهای استاندارد ابزارهای موسیقیایی فراهم می‌آورد.

۳-۳ روشهای دیجیتالی ذخیرهٔ صدا

در طراحی یک سیستم صوتی دیجیتال دو پرسش وجود دارند که باید پاسخ داده شوند: ۱- چقدر لازم است صوت خوب به نظر برسد؟ ۲- چه نرخ داده‌ای قابل تحمل است؟ جواب به این پرسشها غالباً به یکی از این سه انتخاب منجر می‌شود: اول **موسیقی با وفاداری بالا**[55] که در آن کیفیت صدا مهم‌ترین چیز است و تقریباً هر نرخ داده‌ای قابل قبول است. دوم **ارتباط تلفنی**[56] که نیازمند طبیعی به نظر رسیدن صحبت و یک نرخ دادهٔ پایین برای کاهش هزینهٔ سیستم است. سوم **صحبت فشرده شده**[57] که در آن کاهش نرخ داده بسیار مهم است و مقداری غیر طبیعی به نظر رسیدن کیفیت صدا قابل تحمل است. این مورد در بر دارندهٔ ارتباطات نظامی، تلفنهای سلولی و صحبت ذخیره شده به صورت دیجیتال برای پست الکترونیکی صوتی یا کاربردهای چند رسانه‌ای است.

شکل شمارهٔ ۵-۳ بده بستانهای موجود در انتخاب هر یک از این سه روش را نشان می‌دهد.

در حالی که موسیقی نیازمند پهنای باند ۲۰ کیلوهرتز است صحبتی که طبیعی به نظر برسد فقط به پهنای باندی در حدود ۳.۲ کیلوهرتز نیازمند است. در این حال هر چند پهنای باند به اندازهٔ ۱۶٪ مقدار اولیه محدود می‌شود ولی فقط ۲۰٪ اطلاعات اولیه از دست می‌رود.

سیستمهای ارتباط راه‌دور اغلب از نرخ نمونه‌برداری در حدود ۸ کیلوهرتز استفاده می‌کنند که اجازهٔ انتقال صحبت را با کیفیتی در حد طبیعی می‌دهد ولی اگر از آن برای انتقال موسیقی استفاده شود تا میزان بالایی از کیفیت آن از دست می‌رود. شما احتمالاً با تفاوت این دو میزان آشنایی دارید: ایستگاههای رادیویی اف.ام با پهنای باندی در حدود ۲۰ کیلوهرتز اقدام به

[55] high fidelity music
[56] telephone communication
[57] compressed speech

پخش می‌کنند حال آن که ایستگاههای ای.ام محدود به ۳.۲ کیلوهرتز هستند. صحبت و صداهای معمول روی ایستگاههای نوع دوم طبیعی به نظر می‌رسد حال آن که موسیقی این گونه نیست.

Sound Quality Required	Bandwidth	Sampling rate	Number of bits	Data rate (bits/sec)	Comments
High fidelity music (compact disc)	5 Hz to 20 kHz	44.1 kHz	16 bit	706k	Satisfies even the most picky audiophile. Better than human hearing.
Telephone quality speech	200 Hz to 3.2 kHz	8 kHz	12 bit	96k	Good speech quality, but very poor for music.
(with companding)	200 Hz to 3.2 kHz	8 kHz	8 bit	64k	Nonlinear ADC reduces the data rate by 50%. A very common technique.
Speech encoded by Linear Predictive Coding	200 Hz to 3.2 kHz	8 kHz	12 bit	4k	DSP speech compression technique. Very low data rates, poor voice quality.

شکل شمارهٔ ۵ – ۳ نرخ دادهٔ صوتی در برابر کیفیت صدا.

کیفیت صدای یک سیگنال صوتی دیجیتال به نرخ دادهٔ آن که برابر با حاصل‌ضرب نرخ نمونه‌برداری آن در تعداد بیتهای آن در هر نمونه بستگی دارد که به سه بخش تقسیم می‌شود: موسیقی باوفاداری بالا (۷۰۶کیلوبیت بر ثانیه)، صحبت با کیفیت تلفن (۶۴کیلوبیت بر ثانیه) وصحبت فشرده شده (۴ کیلوبیت بر ثانیه)

سیستمهایی که فقط با صدا (و نه موسیقی) سر و کار دارند می‌توانند مقدار دقت را از ۱۶ بیت به ۱۲ بیت بدون از دست رفتن دقتی قابل توجه کاهش دهند. این میزان می‌تواند با انتخاب اندازهٔ نامتساوی برای گام مقدارگزینی[۵۸] می‌تواند به ۸ بیت در هر نمونه نیز کاهش یابد. یک نرخ نمونه‌برداری ۸ کیلوهرتز با دقت ای.دی.سی ۸ بیت در هر نمونه به نرخ دادهٔ ۶۴کیلوبیت بر ثانیه می‌انجامد. این یک حد نهایی برای طبیعی به نظر رسیدن صحبت است. دقت کنید که صحبت نیازمند نرخ داده‌ای معادل ۱۰٪ نرخ دادهٔ موسیقی با وفاداری بالاست.

[58] quantization step

نرخ دادهٔ ۶۴ کیلو بیت بر ثانیه نمایانگر کاربرد نهایی نظریهٔ نمونه‌برداری و مقـدارگزینی بـرای سیگنالهای صوتی است. روشهای کاهش نرخ داده به اندازه‌ای بیشتر از این مبتنی بر فشرده‌سـازی جریان داده با حذف تکرارهای ذاتی سیگنال صـحبت اسـت. یکـی از کـاراترین روشـهای موجـود ال.پی.سی[۵۹] است که انواع و زیرگروههای متعدد دارد. بر اساس کیفیت سیگنال صحبت مورد نیاز این روش می‌تواند نرخ داده را تا اندازه‌ای بین ۲ تا ۶ کیلو بیت بر ثانیه کاهش دهد.

[59] LPC (Linear Predictive Coding)

بخش چهارم

پردازش صوت : برنامه‌نویسی و پیاده‌سازی

۴-۱ ساختار مورد نیاز برای نگهداری ویژگیهای صدا

همچنان که در فصل پیش اشاره شد برای ذخیره یا بازخوانی یک نمونه صدا به صورت دیجیتال نیازمند آنیم که برخی ویژگیهای خاص صدای دیجیتالی از قبیل نرخ نمونه‌برداری، تعداد بیت هر نمونه و یک‌کاناله یا دوکاناله بودن صدا را مشخص کنیم.

برای این منظور در محیط برنامه‌نویسی مورد نظر ما (ویندوز) از ساختاری به نام WAVEFORMATEX استفاده می‌گردد که به صورت زیر تعریف می‌گردد:

```
typedef struct {
WORD  wFormatTag;
WORD  nChannels;
DWORD nSamplesPerSec;
DWORD nAvgBytesPerSec;
WORD  nBlockAlign;
WORD  wBitsPerSample;
WORD  cbSize;
```

در این ساختار فیلد wFormatTag فرمت فایل را که نشان دهندهٔ نوع الگوریتمهای به کار گرفته شده برای فشرده‌سازی صدا و... است را مشخص می‌کند. برای استفادهٔ مورد نظر ما فرمت خاصی که با ثابت WAVE_FORMAT_PCM مشخص می‌گردد و فرمت پی.سی.ام[۶۰] نامیده می‌شود مناسب است. علاوه بر آن فیلد cbSize برای فرمتهای غیر پی.سی.ام استفاده می‌شود و ما همواره مقدار آن را صفر در نظر خواهیم گرفت.

از آنجا که پردازش این ساختار در برنامه‌نویسی صدا برای پروژهٔ مورد نظر بارها صورت می‌گیرد و از آنجا که یک شیوهٔ طراحی شیءگرا (شیوهٔ ام.اف.سی[۶۱])برای پیاده‌سازی پروژه در نظر گرفته شده بود و از آنجا که پردازش این ساختار نیاز به برخی محاسبات تکراری (تعیین nBlockAlign و nAvgBytesPerSec) دارد و به چند دلیل دیگر تصمیم گرفته شد که این

[60] PCM
[61] MFC

ساختار و پردازش آن به صورت یک کلاس با نام HSound پیاده سازی گردد که ضمن خودکـار نمودن پردازش این ساختار کلاسهایی که به اعمال پخش و ضبط را بر عهده دارند از ایـن کـلاس ارثبری نموده برنامه نویسی را آسان‌تر و کد به دست آمده را خواناتر نمایند.

تعریف این کلاس به صورت زیر است:

```
class Hsound
{
public:
//constructor and destructor:
HSound();
virtual ~HSound();
//setting wave data:
void SetBitsPerSample(int bps);
void SetSamplesPerSecond(int sps);
void SetNumberOfChannels(int nchan);
//retrieving wave data:
WAVEFORMATEX* GetFormat();
int GetSamplesPerSecond();
int GetBitsPerSample();
int GetNumberOfChannels();
protected:
WAVEFORMATEX m_wfData;
private:
void Update();
};
```

قبل از هر چیز باید به این نکتـه اشـاره شـود کـه ایـن کـلاس بـرای پـردازش فرمـت پی.سی.ام در نظر گرفته شده، لذا ضمن تعریف مقادیر پیش‌فرض لازم برای این فرمت و استفاده از

روشهای خاص این فرمت برای محاسبهٔ فیلدهای مختلف امکان تغییر فرمت و استفاده از سایر فرمتها را به برنامه‌نویس نمی‌دهد و به منظور پردازش سایر فرمتها ساختار کلاس می‌بایست تغییر کند.

فیلد wBitsPerSample تعداد بیتِ هر نمونه را مشخص میکند که برای فرمت پی.سی.ام فقط می‌تواند یکی از دو مقدار ۸ و ۱۶ را داشته باشد و برای سایر فرمتها مقادیر ممکن بستگی به مشخصات منتشر شده توسط شرکتهای به وجود آورنده و پشتیبانی‌کنندهٔ آنها دارد.

متدی که در پی می‌آیند آن را مقدارگذاری می‌کند (در مورد متد Update و علت فراخوانی آن در ادامه توضیح داده خواهد شد) :

```
 void HSound::SetBitsPerSample(int bps)
{
m_wfData.wBitsPerSample = bps;
Update();
}
```

و متد زیر مقدار انتخاب شده را برمی‌گرداند:

```
 int HSound::GetBitsPerSample()
{
return m_wfData.wBitsPerSample;
}
```

فیلد nSamplesPerSec تعداد نمونه‌ها در هر ثانیه (نرخ نمونه‌برداری) را مشخص می‌کند. برای فرمت پی.سی.ام مقادیر معمول ۸کیلوهرتز (۸۰۰۰)، ۱۱.۰۲۵کیلوهرتز (۱۱۰۲۵)، ۲۲.۰۵کیلوهرتز (۲۲۰۵۰) و ۴۴.۱کیلوهرتز (۴۴۱۰۰) می‌باشد و برای سایر فرمتها مقادیر ممکن بستگی به مشخصات منتشر شده توسط شرکتهای به وجود آورنده و پشتیبانی‌کنندهٔ آنها دارد.

متد مقدارگذاری این فیلد:

```
void HSound::SetSamplesPerSecond(int sps)
{
```

```
m_wfData.nSamplesPerSec = sps;
Update();
}
```

و متد دریافت مقدار آن:

```
int HSound::GetSamplesPerSecond()
{
return m_wfData.nSamplesPerSec;
}
```

فیلد nChannels تعداد کانالهای موج صوتی را مشخص می‌کنند. صداهای تک کانال (مقدار فیلد برابر با ۱) مونو و صداهای دوکاناله (مقدار فیلد برابر با ۲) استریو خواهند بود.

متد مقدارگذاری این فیلد:

```
void HSound::SetNumberOfChannels(int nchan)
{
        m_wfData.nChannels = nchan;
        Update();
}
```

و متد دریافت مقدار آن:

```
int HSound::GetNumberOfChannels()
{
return m_wfData.nChannels;
}
```

هر چند تعداد کانالهای صدا در این کلاس قابل تغییراست اما در کلاسهای مشتق شده همواره الگوریتمها برای موج صوتی تک‌کاناله نوشته شده‌اند و استفاده از آنها برای پردازش موج

صوتی دو کاناله نیازمند دستکاری کد این کلاسهاست که به لحاظ استفاده‌ای که ما از این کلاسها نموده‌ایم یک کار اضافی و غیرضروری به نظر می‌رسد.

فیلد nBlockAlign کمینهٔ تعداد واحد داده را برای فرمت انتخاب شده تأیید می‌کند که اگر فرمت انتخاب شده پی.سی.ام باشد برابر با حاصل ضرب تعداد کانالها (nChannels) در تعداد بیتِ هر نمونه (nBitsPerSample) تقسیم بر تعداد بیتهای موجود در هر بایت (۸) خواهد بود و برای سایر فرمتها بستگی به مشخصات منتشر شده توسط شرکتهای به وجود آورنده و پشتیبانی‌کنندهٔ آنها دارد. فیلد nAvgBytesPerSec نیز تعداد متوسط بایتهای موجود در هر ثانیهٔ صدا را مشخص می‌کند و برای فرمت پی.سی.ام برابر با تعداد نمونه‌های موجود در هر ثانیه (nSamplesPerSec) در کمینهٔ تعداد واحد داده (nBlockAlign) خواهد بود و برای سایر فرمتها بستگی به مشخصات منتشر شده توسط شرکتهای به وجود آورنده و پشتیبانی‌کنندهٔ آنها دارد.

متد Update که در کد مقدارگذاری سایر فیلدها محاسبات توضیح داده شدهٔ بالا را انجام می‌دهد:

```
void HSound::Update()
{
      m_wfData.nBlockAlign=
  m_wfData.nChannels
  *
  (m_wfData.wBitsPerSample/8);
      m_wfData.nAvgBytesPerSec=
  m_wfData.nSamplesPerSec
  *
  m_wfData.nBlockAlign;
}
```

در صورتی که نیاز باشد با ساختار اصلی WAVEFORMATEX کار شود متد زیر مقدار عضوی از کلاس را که از این نوع است باز می‌گرداند:

```
WAVEFORMATEX* HSound::GetFormat()
{
	return &m_wfData;
}
```

در متد سازندهٔ این کلاس به طور پیش‌فرض برای نمونهٔ صوتی مورد نظر نرخ نمونه‌برداری ۴۴.۱کیلوهرتز با ۱۶ بیت در هر نمونه در نظر گرفته شده و فرض بر آن است که نمونهٔ صوتی یک کاناله است:

```
HSound::HSound()
{
	m_wfData.wFormatTag  = WAVE_FORMAT_PCM;
	m_wfData.cbSize      = 0;
	SetBitsPerSample(16);
	SetSamplesPerSecond(44100);
	SetNumberOfChannels(1);
}
```

همچنانکه از روی تعریف کلاس قابل فهم است این کلاس در واقع تمامی اعمال را روی عضو دادهٔ محافظت شدهٔ m_wfData اِعمال می‌نماید و با غیر مستقیم نمودن دسترسی به این عضو داده برای برنامهٔ استفاده کننده ضمن رعایت اصل پنهانسازی اطلاعات به فراخوانی رویهٔ Update در متدهای تغییر دهندهٔ اعضای مرتبط با nBlockAlign و nAvgBytesPerSec تغییرات لازم را به آنها اعمال می‌کند.

۲–۴ انجام پردازش صدا به صورت یک رشتهٔ[۶۲] مستقل

می‌توان با استفاده از توابع کار با صدای ویندوز به گونه‌ای برنامه‌نویسی نمود که نیازی به ایجاد رشته‌های مستقل برای پردازنده‌های صدا نباشد، اما وجود دلایلی از قبیل عدم انعطاف‌پذیری این روش و تک‌وظیفه‌ای شدن برنامه در حین انجام عملیات پردازش صدا باعث می‌شود که روش استفاده از رشته‌های مستقل مورد توجه ما قرار گیرد.

از آغاز در نظر داشتیم که رابط برنامه به گونه طراحی شود که کاربر در هنگام کار با برنامه و انجام عملیاتی نظیر ضبط صدا از عملکرد برنامه مطمئن باشد. به این معنی که مثلاً در حین هنگام صدا با استفاده از یک رابط گرافیکی مانند یک نمایشگر اسیلوسکوپی از این که برنامه واقعاً و به درستی در حال ضبط صدای اوست و یا به لحاظ فاصلهٔ نامتناسب با میکروفن یا عدم اتصال درست آن به کارت صوتی یا خرابی آن بیشتر آنچه ضبط می‌شود سکوت و یا نویز است مطلع گردد. یک روش مناسب برای ایجاد چنین رابطی استفاده از پیام فرستاده شده برای پردازش صدا توسط یک رشته برای فعال شدن یک تابع رسم‌کنندهٔ نمودار اسیلوسکوپی است که نیاز به آن دارد که بدون قطع شدن جریان ضبط پردازش دیگری صورت گیرد. به این منظور و با استفاده از کد اولیه‌ای که در منبع شمارهٔ ۲ به آن اشاره شده کلاسی به نام HSoundRunner را از کلاس HSound اعضای داده و متدهای مرتبط با پردازش صوت و از کلاس ام.اف.سی CwinThread اعضای داده و متدهای لازم برای یک رشته را ارث‌بری می‌کند به صورت زیر تعریف نمودیم:

```
class HSoundRunner:
        public CWinThread,
        public HSound
{
public:
        DECLARE_DYNCREATE(HSoundRunner)
```

[62] thread

```
        HSoundRunner();
        ~HSoundRunner();

        void   SetBufferSize(int nSamples);
        int      GetBufferSize();

        //this metheods should be overriden:
        void AddBuffer();
        BOOL Start(WAVEFORMATEX* pwfex=NULL);
        BOOL Stop();

        //for graphical display:
        void SetOwner(CWnd* pWnd);
        void ClearOwner(COLORREF crBkColor=0x000000);

public:

        //{{AFX_VIRTUAL(HSoundRecorder)
        public:
        virtual BOOL InitInstance();
        //}}AFX_VIRTUAL

protected:
        DWORD m_dwThreadID;

        int m_iBufferSize; // number of samples per each period
        int m_nBuffers;             //number of buffers remained to be run

        int m_nSamples;             //number of samples stored
```

```
        short* m_pSamples;        //samples stored

        BOOL m_bRunning;        //indicated running or not

//if graphical display is intended set this value
        CWnd* m_pOwner;

        void DrawBuffer(
            int nSamples,
            short* pSamples,
            COLORREF crBkColor=0x000000,
            COLORREF crLineColor=0x00FF00
           );
};
```

اعضای دادهٔ این کلاس در روند انجام عملیات توسط کلاسهای مشتق شده از آنها نقش خود را نشان خواهند داد و به عنوان نمونه عضو دادهٔ m_iBufferSize که نشان دهندهٔ آن است که بعد از ضبط با پخش چند نمونه تابع پزدازندهٔ پیام در کلاس پنجرهٔ کنترل کننده باید فراخوانی شود در این هیچکدام از متدهای این کلاس نقش عملی پیدا نمی‌کند و فقط مقدارگذاری آن از طریق متد SetBufferSize و دیافت مقدار فعلی آن از طریق GetBufferSize صورت می‌گیرد:

```
void HSoundRunner::SetBufferSize(int nSamples)
{
        m_iBufferSize=nSamples;
```

```
}

int HSoundRunner::GetBufferSize()
{
	return m_iBufferSize;
}
```

عضو دادهٔ m_dwThreadID مقدار شناسهٔ رشتهٔ ایجاد شده را که در کلاس سازنده با فراخوانی CreateThead ایجاد می‌شود در بر می‌گیرد که در کلاسهای مشتق شده برای کار با فراخوانیهای ای.پی.آی پردازش صدا کاربرد پیدا می‌کند. مقدار این عضو داده در متد بازنویسی[۶۳] شدهٔ InitInstance و به صورت زیر تعیین می‌گردد:

```
BOOL HSoundRunner::InitInstance()
{
	m_dwThreadID = ::GetCurrentThreadId();
	return TRUE;
}
```

اعضای دادهٔ m_nSamples و m_pSamples به ترتیب تعداد نمونه‌های ضبط شده یا آماده برای پخش و آرایهٔ حاوی آنها -که به لحاظ آسان‌تر شدن کار با کتابخانه‌ای که برای پردازش سیگنال صحبت استفاده شده و همسان با آن از نوع short در نظر گرفته شده- را نشان می‌دهند. این دو عضو داده به صورت پویا پس از انجام عملیات ضبط مقدارگذاری می‌شوند و برای عملیات پخش باید قبلاً مقدارگذاری شده باشند.

آنچه این کلاس انجام می‌دهد غیر از ایجاد یک رشته برای انجام پردازش صدا فراهم آوردن روشی برای نمایش اسیلوسکوپی صداست که از طریق متد DrawBuffer انجام می‌شود.

[63] overriden

این متد در توابع پیامهای مربوط به پردازش صوت خود به خود فراخوانی می‌گردد و در صورتی که مقدار m_pOwner اشاره‌گر به یک پنجره یا کنترل انتخاب شود (توسط متد SetOwner) آرایهٔ ورودی که معمولاً یک تکهٔ تازه ضبط با پخش شده از کل صداست متناسب با طول و عرض پنجرهٔ مورد نظر بر روی آن کشیده می‌شود. این کار با ایجاد یک ابزار متن[۶۴] و یک بیتمپ متناسب با ابزار متن پنجرهٔ مورد نظر، کشیدن طرح لازم با استفاده از این دو و در نهایت نمایش تصویر ایجاد شده بر روی پنجرهٔ مقصد و مطابق با کد زیر انجام می‌شود:

```
void HSoundRunner::DrawBuffer(
                    int nSamples,
                    short* pSamples,
                    COLORREF crBkColor,
                    COLORREF crLineColor
                   )
{
        if(m_pOwner==NULL)
                return;

        CRect rc;
        m_pOwner->GetClientRect(&rc);
        int iWidth=rc.Width();
        int iHeight=rc.Height();

        CDC* pDC=m_pOwner->GetDC();

        CBitmap Bitmap;
        Bitmap.CreateCompatibleBitmap(pDC, iWidth, iHeight);
```

[64] Device Context (DC)

```
CDC dc;
dc.CreateCompatibleDC(pDC);

dc.SelectObject(&Bitmap);
CBrush Brush(crBkColor);
dc.FillRect(&rc,&Brush);
CPen Pen(PS_SOLID,1,crLineColor);
dc.SelectObject(&Pen);
dc.SetBkColor(crBkColor);
if(GetBitsPerSample()==16)
{
        float fx=iWidth/float(nSamples);
        float fy=float(iHeight/32767.0);
        dc.MoveTo(0, iHeight/2);
        int i=0;
        for(
float f=0; f<iWidth&&i<nSamples; f+=fx, i++
 )
                dc.LineTo(
   int(f), int(iHeight/2+fy*pSamples[i])
         );
        pDC->BitBlt(
   0, 0,
   iWidth, iHeight,
   &dc,
   0, 0,
   SRCCOPY
      );
}
```

```
}
```

متد ClearBuffer یک روش قابل دسترسی توسط برنامه برای پاک کردن پنجرهٔ مورد استفاده به وجود می‌آورد و شامل یک فراخوانی متد محافظت شدهٔ DrawBuffer با یک آرایهٔ به طول صفر است:

```
void HSoundRunner::ClearOwner(COLORREF crBkColor)
{
	DrawBuffer(0,NULL,crBkColor);
}
```

عضو دادهٔ m_nBuffers تعداد بافرهای اختصاص داده شده و استفاده نشده را نشان می‌دهد که در متد AddBuffer این کلاس و بازنویسی شدهٔ آن برای کلاسهاس مشتق شده مقدار آن به ازای هر بار فراخوانی یک واحد افزوده می‌شود و در پیامهای پردازش صدا که از طرف سیستم عامل فعال می‌شوند و نشانگر استفاده شدن بافر مورد نظر است (در کلاسهای مشتق شده) یک واحد کاهش می‌یابد. در نهایت صفر نبودن این عضو داده نشانگر استفادهٔ ناکامل از بافرهای اختصاص داده شده (معادل با کامل انجام نشدن فرایند ضبط یا پخش) است که می‌تواند پردازش مناسب برای آن صورت گیرد:

```
void HSoundRunner::AddBuffer()
{
	m_nBuffers++;
}
```

عضو دادهٔ m_bRunning به منظور تشخیص این که برنامه در حال اجـرای عملیـات پردازش صداست و یا نه به منظور جلوگیری از ایجاد اشکال با فراخوانیهای تکراری در نظر گرفتـه شده که در هنگام آغاز عملیات مقدار آن برابر با TRUE و در پایان آن برابر با FALSE انتخاب می‌گردد. همچنان که در ادامه توضیح داده خواهد شد کلاسهای مشتق شده متدهایی از لحاظ نام متناسب با عملی که برای آن در نظر گرفته شده‌اند (ضبط یا پخش) برای بازگرداندن این مقدار به برنامه‌نویس کاربر این کلاسها دارند:

```
BOOL HSoundRunner::Start(WAVEFORMATEX* pwfex)
{
	if(m_bRunning)
		return FALSE;
	if(pwfex != NULL)
		m_wfData = *pwfex;
	return TRUE;
}

BOOL HSoundRunner::Stop()
{
	if(m_bRunning)
	{
		m_bRunning=FALSE;
		Sleep(500);
		return TRUE;
	}
	return FALSE;
}
```

فراخوانی استاندارد Sleep در متد Stop برای مصرف کامل بافر ایجاد شده انجام می‌گردد. در ضمن متد Start روشی برای جایگزینی مقدار پیش‌فرض m_wfData (عضو کلاس Hsound) با مقدار جدید در اختیار می‌گذارد.

در متد سازنده اعضای داده با مقادیر پیش‌فرض مقدارگذاری شده و رشتهٔ مورد نظر با فراخوانی CreateThead ایجاد می‌گردد:

```
HSoundRunner::HSoundRunner()
{
	m_iBufferSize= 2048;
	m_nBuffers = 0;
	m_bRunning = FALSE;
	m_nSamples=0;
	m_pSamples=NULL;

	m_pOwner=NULL;

	CreateThread();
}
```

در متد ویرانگر[۶۵] نیز در صورتی که شیء از نوع این کلاس در حال انجام عملیات پردازش صوت باشد متوقف خواهد شد:

```
HSoundRunner::~HSoundRunner()
{
	if(m_bRunning)
		Stop();
}
```

[65] destructor

به لحاظ آن که آرایهٔ داده‌ها (m_pSamples) در این کلاس ایجاد نمی‌گردد در متـد ویرانگر آزاد شدن آن پیشبینی نشده است.

۳-۴ ضبط صدا

برای ضبط صدا و انجام پردازشهای مرتبط با آن کلاسی به نـام HSoundRecorder به صورت زیر از کلاس HSoundRunner مشتق گردید:

```
class HSoundRecorder :
        public HSoundRunner
{
        DECLARE_DYNCREATE(HSoundRecorder)

public:
        HSoundRecorder();
        virtual ~HSoundRecorder();

protected:
        void AddBuffer();

        //Message Map For WM_WIM_DATA:
        afx_msg void OnDataReady(UINT uParm, LONG lWaveHdr);

private:
        HWAVEIN m_hWaveIn;
        HShortQueue* m_pQueue;

public:
```

```
    BOOL Start(WAVEFORMATEX* pwfex=NULL);
    BOOL Stop();
    short* GetSamples(int& nSamples);
    BOOL IsRecording();

    DECLARE_MESSAGE_MAP()
};
```

برای استفاده از این کلاس کافی است متدهای Start و Stop آن فراخوانی گردند ولی درک کامل نحوهٔ عملکرد آن نیاز به برخی مقدمات دارد.

برای شروع کار ضبط از فراخوانی ای.پی.آی زیر با پارامترهای مناسب برای در اختیار گـرفتن یـک ابزار ورودی صدا استفاده می‌کنیم:

```
MMRESULT waveInOpen(
  LPHWAVEIN phwi,
  UINT uDeviceID,
  LPWAVEFORMATEX pwfx,
  DWORD dwCallback,
  DWORD dwCallbackInstance,
  DWORD fdwOpen
);
```

که در آن phwi اشاره‌گر به بافری است که یک handle به ابزار باز شده برای ورودی صدا را در اختیار می‌گذارد. این پارامتر ابزاری را برای دسترسی به وسـیلهٔ ورودی صـدا در اختیـار می‌گذارد که ما در سایر فراخوانیهای مرتبط به آن نیاز داریم لـذا در کـلاس تعریـف شـده متغیـر m_hWaveIn را برای ذخیرهٔ این پارامتر پس از این فراخوانی و دسترسی به آن در سایر متـدها در نظر گرفته‌ایم.

پارامتر uDeviceID شناسهٔ ابـزار ورودی را بـه تـابع مـی‌دهـد. مـی‌تـوان از شناسـهٔ WAVE_MAPPER استفاده کرد که با استفاده از آن برنامه سخت‌افزار پیش‌فرض موجـود را که دارای قابلیت پردازش فرمت انتخاب شده که توسط پارمتر pwfx به تابع داده می‌شود و مـا از عضو دادهٔ m_wfData از کلاس HSound برای انتخاب مقدار آن استفاده مـی‌کنـیم بـه طـور خودکار انتخاب می‌کند.

پارامتر dwCallback شناسهٔ پنجره، پردازه یا رشته‌ای را که پیامهای چندرسانه‌ای به آن ارسال خواهد شد به تابع می‌دهد که ما از شناسهٔ رشته (عضو دادهٔ m_dwThreadID) برای تعیین این پارامتر استفاده خـواهیم نمـود. پـارامتر بعـدی dwCallbackInstance دادهٔ سـطح کاربری را که به ساز و کار فرتخوانی callback ارسال می‌شود تعیین می‌نماید و ما از این پارامتر استفاده نخواهیم نمود.

پارامتر آخر یعنی fdwOpen پرچمی بـرای ابـزار ورودی اسـت کـه سـازوکار تفسـیر پارامترها را مشخص می‌کند و چون ما از سازوکار فراخوانی رشته‌ای استفاده می‌کنیم مقـدار آن را برابر با CALLBACK_THREAD انتخاب می‌نماییم.

مقدار بازگشتی در صورت بروز اشکال غیرصفر خواهد بـود مسـائلی از قبیـل اختصـاص یافتن وسیلهٔ ورودی به یک پردازهٔ دیگر به صورتی که سیستم عامل به صـورت اشـتراکی آن را در اختیار نگذارد، کمبود حافظه و ... ممکن است باعث بروز اشکال شوند که نوع اشکال بـا توجـه بـه مقدار بازگشتی مشخص می‌گردد و میتواند به کاربر اعلام گردد.

بعد از در اختیار گرفتن یک وسیلهٔ ورودی لازم است که برای عملیات حافظه اختصاص یابد و این عمل می‌تواند با فراخوانی AddBuffer به تعداد کافی صورت گیرد.

بعد از تخصیص حافظه توسط فراخوانی ای.پی.آی زیر عمل ضبط شروع می‌گردد:

```
MMRESULT waveInStart(
```

```
HWAVEIN hwi
);
```

پارامتر وروی همان پارامتر بازگشت با مقدار فراخـوانی waveInOpen یعنـی phwi است که همچنانکه اشاره شد مـا آن را در عضـو دادهٔ m_hWaveIn نگهـداری مـی‌کنـیم. ایـن فراخوانی نیز مانند قبلی در صورت موفقیت‌آمیز بودن مقدار صفر باز می‌گرداند.

توضیحات بالا مقدمات کافی را برای درک کد متد Start کـه در زیـر مـی‌آیـد فـراهم می‌آورد:

```
BOOL HSoundRecorder::Start(WAVEFORMATEX* pwfex)
{
        if(!HSoundRunner::Start(pwfex))
                return FALSE;
        m_pQueue=new HShortQueue;

        //Open the wave device:
        if(
    ::waveInOpen(
                &m_hWaveIn,
                WAVE_MAPPER,
                &m_wfData,
                m_dwThreadID,
                0L,
                CALLBACK_THREAD
               )
      )
                return FALSE;
```

```
        //Add several buffers to queue:
        for(int i=0;i<3;i++)
                AddBuffer();
        if(::waveInStart(m_hWaveIn))
                return FALSE;

        m_bRunning=TRUE;
        return TRUE;
}
```

اما توابع چندرسانه‌ای ویندوز سازوکار خاصی برای اضافه کردن بافر دارند که می‌بایست آنها را در نسخهٔ بازنویسی شدهٔ AddBuffer این کلاس لحاظ کنیم. برای اضافه کردن یک بافر ابتدا باید آن را توسط فراخوانی زیر برای استفاده آماده کنیم:

```
MMRESULT waveInPrepareHeader(
  HWAVEIN hwi,
  LPWAVEHDR pwh,
  UINT cbwh
);
```

به جای پارامتر hwi عضو دادهٔ m_hWaveIn را که قبلاً در فراخوانی waveInOpen مقدارگذاری شده قرار می‌دهیم. پارامتر دوم یک اشاره‌گر به متغیری با ساختار زیر است:

```
typedef struct {
   LPSTR  lpData;
   DWORD  dwBufferLength;
   DWORD  dwBytesRecorded;
```

```
    DWORD  dwUser;
    DWORD  dwFlags;
    DWORD  dwLoops;
    struct wavehdr_tag * lpNext;
    DWORD  reserved;
} WAVEHDR;
```

که لازم است اشاره‌گر lpData به یک حافظه حاوی تعداد مورد نیاز عضو اشاره کند. از آنجا که ما هر بار بافری با اندازهٔ m_iBufferSize در نظر می‌گیریم، تعداد خانه‌های این آرایه بر حسب بایت برابر با اندازهٔ بافر ضرب در حداقل تعداد بلوک برای فرمت انتخاب شده (فیلد nBlockAlign ساختار WAVEFORMATEX) می‌باشد و لازم است که به این تعداد حافظه اختصاص داده اشاره‌گر lpData را برابر با آدرس آن انتخاب کنیم، در ضمن اندازهٔ بافر اختصاص داده شده را از طریق فیلد dwBufferLength به اطلاع تابع استفاده کننده می‌رسانیم. پارامتر آخر فراخوانی مورد بحث باید برابر با اندازهٔ پارامتر دوم بر حسب بایت قرار داده شود.

بعد از آماده شدن بافر آن را توسط فراخوانی زیر به بافرهای آماده برای اعمال چندرسانه‌ای اضافه می‌کنیم:

```
MMRESULT waveInAddBuffer(
  HWAVEIN hwi,
  LPWAVEHDR pwh,
  UINT cbwh
);
```

مانند فراخوانیهای قبل مقدار خروجی دو فراخوانی اخیر در صورت عدم بروز خطا صفر خواهد بود. کد زیر پیاده‌سازی کامل متد بازنویسی شدهٔ AddBuffer را برای کلاس HSoundRecorder به نمایش می‌گذارد:

```
void HSoundRecorder::AddBuffer()
{
        //new a buffer:
        char    *sBuf=
       new char[m_wfData.nBlockAlign*m_iBufferSize];

        //new a header:
        LPWAVEHDR      pHdr=new WAVEHDR;
        if(!pHdr)      return;
        ZeroMemory(pHdr,sizeof(WAVEHDR));

        pHdr->lpData=sBuf;
        pHdr->dwBufferLength=
              m_wfData.nBlockAlign*m_iBufferSize;

        //prepare it:
        ::waveInPrepareHeader(
                 m_hWaveIn,
                 pHdr,
                 sizeof(WAVEHDR)
                );
        //add it:
        ::waveInAddBuffer(m_hWaveIn, pHdr, sizeof(WAVEHDR));
```

```
        HSoundRunner::AddBuffer();
}
```

بعد از آن که عمل ضبط آغاز شد و پس از پر شدن هر بافر پیغامی به پنجره یا رشته‌ای که شناسهٔ آن به فراخوانی waveInOpen داده شده ارسال می‌گردد که با شناسهٔ MM_WIM_DATA می‌توان به آن مراجعه نمود. در هنگام فعال شدن این پیغام لازم است بافر استفاده شده برای استفادهٔ مجدد آماده گردد و در ضمن مکان مناسب برای ذخیرهٔ داده‌های ضبط شده و همچنین نمایش آنها پس از فعال شدن این پیغام است.

از آنجا که طول زمان ضبط صدا مشخص نیست طول بافری که نهایتاً داده‌ها باید در آن قرار گیرند قابل پیشبینی نمی‌باشد. از این رو ما از یک ساختار که نوع آن را HShortPocket نامگذاری کرده‌ایم برای ذخیرهٔ داده‌های ارسال شده استفاده می‌نماییم و حاصل را در صفی که در قالب کلاس HShortQueue پیاده‌سازی شده درج می‌نماییم.(این دو ساختار ربطی به برنامه‌نویسی پردازش صدا ندارند و درک عملکرد آنها نیاز به توضیح اضافی ندارد لذا در اینجا توضیح داده نمی‌شوند.) عضو دادهٔ m_pQueue از کلاس HSoundRecorder صفی است که در سطور قبل در مورد آن بحث شد.

ما به شیوهٔ ام.اف.سی برای پیغام MM_WIM_DATA تابعی به نام OnDataReady می‌سازیم که پارامترهای آن پارامترهای ارسالی از طرف پیغام هستند که دومین آنها که حاوی ساختار بافر استفاده شده است برای ما اهمیت دارد. با توجه به توضیحات داده شده درک کد این تابع میسر است:

```
void
HSoundRecorder::OnDataReady(UINT uParm, LONG lWaveHdr)
{
        LPWAVEHDR      pHdr=(LPWAVEHDR)lWaveHdr;
```

```
::waveInUnprepareHeader(
            m_hWaveIn,
            pHdr,
            sizeof(WAVEHDR)
           );
if(m_bRunning)
{
        //Save Input Data:
        m_pQueue->InsertItem(
                pHdr->dwBytesRecorded/2,
                (short*)pHdr->lpData
               );

        //Draw Buffer:
        DrawBuffer(
    pHdr->dwBytesRecorded/2,
    (short*)pHdr->lpData
      );

        //reuse the header:
        ::waveInPrepareHeader(
              m_hWaveIn,
              pHdr,
              sizeof(WAVEHDR)
             );
        ::waveInAddBuffer(
            m_hWaveIn,
            pHdr,
            sizeof(WAVEHDR)
```

```
                );
            return;
        }
        //we are stopping:
        delete pHdr->lpData;
        delete pHdr;
        m_nBuffers--;
}
```

در صورتی که متد Stop احضار شده باشد مقدار m_bRunning برابر با FALSE است در این هنگام نه تنها نیازی به اضافه کردن بافر نداریم بلکه می‌توانیم بافرهای اختصاص داده شده را آزاد کنیم. قسمت آخر کد چنین عملی را انجام می‌دهد.

پس از انجام عمل و احضار متد Stop توسط برنامه لازم است با فراخوانی waveInStop عمل ضبط را متوقف کنیم و با فراخوانی waveInClose ابزار ضبط آن را برای استفادهٔ سایر برنامه‌ها آزاد کنیم. علاوه بر این در این هنگام تمامی بافرها ارسال شده‌اند و می‌توانیم آن را به صورت یک آرایهٔ معمولی ذخیره کنیم و متغیر m_pQueue را آزاد نماییم:

```
BOOL HSoundRecorder::Stop()
{
        if(!HSoundRunner::Stop())
                return FALSE;

        ::waveInStop(m_hWaveIn);
        ::waveInClose(m_hWaveIn);

        m_pSamples=m_pQueue->ConvertToArray(m_nSamples);
        delete m_pQueue;
```

```
        return TRUE;
}
```

پس از انجام این عمل برنامه می‌تواند با فراخوانی GetSamples به آرایهٔ حاوی صدای ضبط شده دسترسی پیدا کند:

```
short* HSoundRecorder::GetSamples(int& nSamples)
{
        nSamples=m_nSamples;
        return m_pSamples;
}
```

این آرایه در هنگام آزاد شدن متغیر از نوع HSoundRecorder در متد ویرانگر آزاد می‌شود:

```
HSoundRecorder::~HSoundRecorder()
{
        if(m_pSamples)
                delete []m_pSamples;
}
```

در متد سازندهٔ کلاس پدر مقدا m_pOwner برابر با NULL در نظر گرفته می‌شود. این به این معنی است که در حالت پیش‌فرض عملیات به صورت گرافیکی نشان داده نمی‌شود و برای انجام این عمل لازم است که ابتدا مقدار m_pOwner به اشاره‌گر به یک پنجره یا کنترل

توسط فراخوانی SetOwner مقدارگذاری شود. در متد سازندهٔ این کلاس به منظور جلـوگیری از مقدارگزینیهای ناخواسته مقدار m_hWaveIn برابر با NULL انتخاب می‌گردد:

```
HSoundRecorder::HSoundRecorder()
{
	m_hWaveIn =NULL;
}
```

۴-۴ پخش صدا

پردازشهای مربوط به پخش صدا مشابهت زیادی با پردازشهای مربوط به ضـبط دارد در اینجا نیز باید ابتدا یک ابزار صدا را برای خروجی باز کرد، تعدادی بافر اضافه نمـود، در تـابع پیـام بافرهای جدید آماده نمود و سرانجام در متد Stop ابزار خروجی را بست:

```
class HSoundPlayer:
	public HSoundRunner
{
public:
	HSoundPlayer();
	~HSoundPlayer();

	void SetData(int nSamples, short* pSamples);
	BOOL Start(WAVEFORMATEX* format=NULL);
	BOOL Start(
	  int iSize,
```

```
        short* pData,
        WAVEFORMATEX* pwfex=NULL
       );
    BOOL Stop();
    BOOL IsPlaying();

    void AddBuffer();

    DECLARE_MESSAGE_MAP()
    afx_msg void OnMM_WOM_DONE(UINT parm1, LONG
parm2);

private:

    HWAVEOUT m_hWaveOut;
    int m_nSamplesPlayed;
};
```

تفاوت تنها در این مورد است که در اینجا ما باید داده‌های آماده را به برنامه بدهیم. به دلیل آن که در برنامه‌های ما خروجی همیشه پس از ورودی مطرح می‌گردد و ما پس از پایان ورودی به داده‌های آن دسترسی داریم ساده‌ترین راه دادن داده‌ها مقداردهی اشاره‌گر m_pSamples با بافر حاوی داده‌های ورودی است که معمولاً ما آن را از شیء ضبط کننده می‌گیریم:

```
void HSoundPlayer::SetData(int iSize, short* pData)
{
```

```
        m_nSamples=iSize;
        m_pSamples=pData;
}
```

به منظور افزایش انعطاف‌پذیری می‌توان این داده‌ها را در متد Start نیز دریافت نمود:

```
BOOL HSoundPlayer::Start (
                    int iSize,
                short* pData,
                WAVEFORMATEX* format
                     )
{
        SetData(iSize, pData);
        return Start(format);
}
```

فراخوانی waveOutOpen عملی مشابه waveInOpen را برای خروجی صدا انجام می‌دهد و پارامترهای آن مشابه با آن است:

```
BOOL HSoundPlayer::Start(WAVEFORMATEX* format)
{
        if(m_pSamples==NULL)
                return FALSE;
        if(!HSoundRunner::Start(format))
                return FALSE;
```

```
        else
        {
                // open wavein device
                MMRESULT mmReturn = 0;
                mmReturn = ::waveOutOpen(
                        &m_hWaveOut,
                        WAVE_MAPPER,
                        &m_wfData,
                        m_dwThreadID,
                        NULL,
                        CALLBACK_THREAD
                       );
                if(mmReturn)
                        return FALSE;
                else
                {
                        m_bRunning = TRUE;

                        m_nSamplesPlayed=0;
                        for(int i=0; i<3; i++)
                                AddBuffer();
                }
        }
        return TRUE;
}
```

در اینجا تفاوتی نیز وجود دارد. در فرایند ضبط این برنامهٔ کاربر بود که پیغام Stop را ارسال می‌کرد حال آن که در اینجا علاوه بر کاربر، تمام شدن بافر حاوی داده‌ها نیز باید باعث فعال شدن آن پیغام شود. برای این منظور از متغیر شمارشگری به نام m_pSamplesPlayed استفاده کرده‌ایم که تعداد نمونه‌های پخش شده بر حسب تعداد حافظهٔ short (که نصف تعداد نمونه در حافظهٔ معادل بر حسب بایت است) را ذخیره می‌کند.

تفاوت دیگری نیز وجود دارد و آن این است که ما باید توسط فراخوانی waveOutWrite داده‌ها را روی بافر ارسالی بنویسیم:

```
void HSoundPlayer::AddBuffer()
{
	MMRESULT mmReturn = 0;

	// create the header
	LPWAVEHDR pHdr = new WAVEHDR;
	if(pHdr == NULL) return;

	// new a buffer
	pHdr->lpData=
	    (char*)(m_pSamples+
	        m_nSamplesPlayed);//buffer;
	pHdr->dwBufferLength = m_iBufferSize;
	pHdr->dwFlags = 0;

	// prepare it
	mmReturn=
	::waveOutPrepareHeader(
	    m_hWaveOut,
```

```
            pHdr,
            sizeof(WAVEHDR)
                        );
        // write the buffer to output queue
        mmReturn =
        ::waveOutWrite(
            m_hWaveOut,
            pHdr,
            sizeof(WAVEHDR)
                  );
        if(mmReturn) return;
        // increment the number of waiting buffers
        m_nSamplesPlayed+=m_iBufferSize/2;

        HSoundRunner::AddBuffer();
}
```

در انجـام عمـل پخـش نیـز پیغـامی پـس از پایـان پخـش هـر بـافر بـا شناسـهٔ MM_WON_DONE به پنجره یا رشتهٔ کنترل کننده ارسال می‌شود که در آن می‌توانیم دادهٔ پخش شده را به صورت گرافیکی نشان دهیم، بافر بعدی را بفرستیم و در صورت رسیدن به پایـان داده‌ها پیغام Stop را ارسال کنیم:

```
void
HSoundPlayer::OnMM_WOM_DONE(UINT parm1, LONG parm2)
{
```

```
    LPWAVEHDR pHdr = (LPWAVEHDR) parm2;
    //Draw Buffer:
    DrawBuffer(
       pHdr->dwBufferLength/2,
       (short*)pHdr->lpData
      );

    if(
 ::waveOutUnprepareHeader(
                    m_hWaveOut,
                    pHdr,
                    sizeof(WAVEHDR)
                   )
)
            return;

    m_nBuffers--;

    if(m_bRunning)
    {

            if(!(m_nSamplesPlayed+m_iBufferSize/2>=m_nSamples))
            {
                    AddBuffer();
                    // delete old header
                    delete pHdr;
                    return;
            }
            else
```

```
            {
                Stop();
            }
        }

        // we are closing the waveOut handle,
        // all data must be deleted
        // this buffer was allocated in Start()
        delete pHdr;

        if(m_nBuffers == 0 && m_bRunning == false)
        {
            if (::waveOutClose(m_hWaveOut))
                return;
        }
}
```

اگر دقت کرده باشید در هنگام رسیدن به پایان داده‌ها در همین تابع اخیر ما ابزار خروجی را می‌بندیم. اگر زودتر پیغام Stop توسط برنامه ارسال شود نیز به لحاظ FALSE شدن مقدار m_bRunning این عمل انجام می‌پذیرد لذا کافی است که در متد Stop ابزار برای استفادهٔ بعدی به طور کامل آزاد گردد:

```
BOOL HSoundPlayer::Stop()
{
    if(HSoundRunner::Stop())
        return (::waveOutReset(m_hWaveOut)!=0);
    return FALSE;
```

```
}
```

از آنجا که در این کلاس حافظه‌ای اختصاص داده نمی‌شود در متد ویرانگری نیز آزاد شدن حافظه انجام نمی‌گیرد. مشابه متد سازندهٔ کلاس HSoundPlayer مقدار m_hWaveOut در ابتدا برابر با NULL در نظر گرفته می‌شود:

```
HSoundPlayer::HSoundPlayer()
{
	m_hWaveOut = NULL;
}
```

۵-۴ کتابخانهٔ پردازش صوت

از آنجا که کلاسهای پیاده‌سازی شده که در این فصل توضیح داده شدند مورد استفادهٔ بیش از یک برنامه قرار گرفته‌اند آنها را به صورت یک کتابخانهٔ ایستا و با نام HSoundLib گردآوری نمودیم تا تغییر کد این کتابخانه راحت‌تر در تمامی برنامه‌ها اعمال گردد و نیاز به تغییر کد تک تک آنها نباشد.

بخش پنجم

پردازش صحبت

۵–۱ ترکیب و تشخیص صحبت

کاربردهای نیازمند پردازش صحبت اغلب در دو دستهٔ ترکیب صحبت[۶۶] و تشخیص صحبت[۶۷] مورد بررسی قرار می‌گیرند.

ترکیب صحبت عبارت است از فن‌آوری تولید مصنوعی صحبت به وسیلهٔ ماشین و به طور عمده از پرونده‌های متنی به عنوان ورودی آن استفاده می‌گردد. در اینجا باید به یک نکتهٔ مهم اشاره شود که بسیاری از تولیدات تجاری که صدای شبیه به صحبت انسان ایجاد می‌کنند در واقع ترکیب صحبت انجام نمی‌دهند بلکه تنها یک تکهٔ ضبط شده به صورت دیجیتال از صدای انسان را پخش می‌کنند. این روش کیفیت صدای بالایی ایجاد می‌کند اما به واژه‌ها و عبارات از پیش ضبط شده محدود است. از کاربردهای عمدهٔ ترکیب صحبت می‌توان به ایجاد ابزارهایی برای افراد دارای ناتوانی بینایی برای مطلع شدن از آنچه بر روی صفحهٔ کامپیوتر می‌گذرد اشاره کرد.

تشخیص صحبت عبارت است از تشخیص کامپیوتری صحبت تولید شده توسط انسان و تبدیل آن به یک سری فرامین یا پرونده‌های متنی. کاربردهای عمدهٔ موجود برای این گونه سیستمها دربرگیرندهٔ بازهٔ گسترده‌ای از سیستمها وکاربردها از سیستمهای دیکتهٔ کامپیوتری که در سیستمهای آموزشی و همچنین سیستمهای پردازش واژه کاربرد دارد گرفته تا سیستمهای کنترل کامپیوترها به وسیلهٔ صحبت و به طور خاص سیستمهای فراهم آورندهٔ امکان کنترل کامپیوترها برای افراد ناتوان از لحاظ بینایی یا حرکتی می‌باشد.

کاربرد مورد نظر ما یعنی تشخیص گوینده از لحاظ نحوهٔ پیاده‌سازی و استفاده تناسب فراوانی با خانوادهٔ دوم یعنی تشخیص کامپیوتری صحبت دارد، ولی از لحاظ اهداف و کاربردها می‌تواند در خانواده‌ای جداگانه از کاربردهای نیازمند پردازش صحبت قرار گیرد.

---

[66] speech sysnthesis
[67] speech recognition

ترکیب و تشخیص کامپیوتری صحبت مسائل دشواری هستند. روشهای مختلفی مورد آزمایش قرار گرفته‌اند که موفقیت کمی داشته‌اند. این زمینه از زمینه‌های فعال در تحقیقات پردازش سیگنال دیجیتال (دی.اس.پی) بوده و بدون شک سالها این گونه خواهد ماند. در حال حاضر از ابزارهای برنامه‌نویسی جاافتاده در زمینه‌های برشمرده شده می‌توان به ای.پی.آی صحبت شرکت مایکروسافت[۶۸] اشاره نمود که دارای تواناییهای عمده‌ای در زمینه‌های تشخیص و ترکیب صحبت است و توانایی آن تا حدی گسترده است که در محصول بزرگ و توانمند MS Office XP از آن استفادهٔ عملی شده است. ابزار عمدهٔ دیگر تولید شرکت آی.بی.ام است و ViaVoice نام دارد که به لحاظ پشتیبانی آن برای سیستم‌عاملهای متعدد و زبانهای گوناگون از اهمیت و کاربرد خاصی برخوردار است.

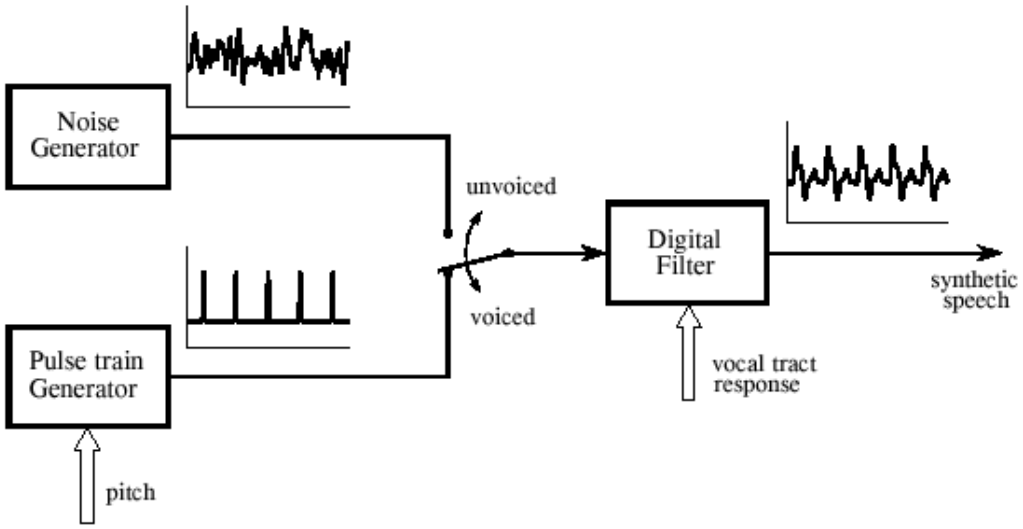
۲–۵ مدلی برای توصیف روش تولید صحبت

تقریباً تمام تکنیکهای ترکیب و تشخیص صحبت بر اساس مدل تولید صحبت انسان که در شکل شمارهٔ ۳ نشان داده شده است ایجاد شده‌اند. بیشتر صداهای مربوط به صحبت انسان به دو دستهٔ **صدادار**[۶۹] و **سایشی**[۷۰] تقسیم می‌شوند. اصوات صدادار وقتی که هوا از ریه‌ها و از مسیر تارهای صوتی به بیرون دهان یا بینی رانده می‌شوند ایجاد میگردند. تارهای صوتی دو رشتهٔ آویخته از بافت هستند که در مسیر جریان هوا کشیده شده‌اند. در پاسخ به کشش ماهیچه‌ای متفاوت تارهای صوتی با فرکانسی بین ۵۰ تا ۱۰۰۰هرتز ارتعاش می‌کنند که باعث انتقال حرکتهای متناوب هوا به نای می‌شود. در شکل شمارهٔ ۱- ۵ اصوات صدادار با یک مولد پالس ترِین[۷۱] با پارامتر قابل تنظیم پیچ (فرکانس پایهٔ موج صوتی) نشان داده شده است.

---

[68] Microsoft SAPI
[69] voiced
[70] fricative
[71] pulse train generator

در مقایسه، اصوات سایشی به صورت نویز تصادفی و نه حاصل از ارتعاش تارهای صوتی به وجود می‌آیند. این حادثه زمانی رخ می‌دهد که تقریباً جریان هوا به وسیلهٔ زبان و لبها یـا دنـدانها حبس می‌شود که این امر باعث ایجاد اغتشاش هوا در نزدیکی محل فشردگی می‌گردد.



شکل شمارهٔ ۱- ۵ مدل صحبت انسان.

در یک تکه زمان کوتاه، حدود ۲ تا ۴۰ میلی‌ثانیه صحبت می‌تواند با استفاده از سه پارامتر مدلسازی شود: ۱- انتخاب یک آشفتگی متناوب یا نویزوار. ۲- پیچ آشفتگی متنـاوب ۳- ضـرایب یک فیلتر خطی بازگشتی که پاسخ اثر صوتی را تقلید می‌کند.

اصوات سایشی زبان انگلیسی عبارتند از s، f، sh، z، v و th. در مدل شکل شـمارهٔ ۱-۵ اصوات سایشی با استفاده از یک مولد نویز نشان داده شده‌اند.

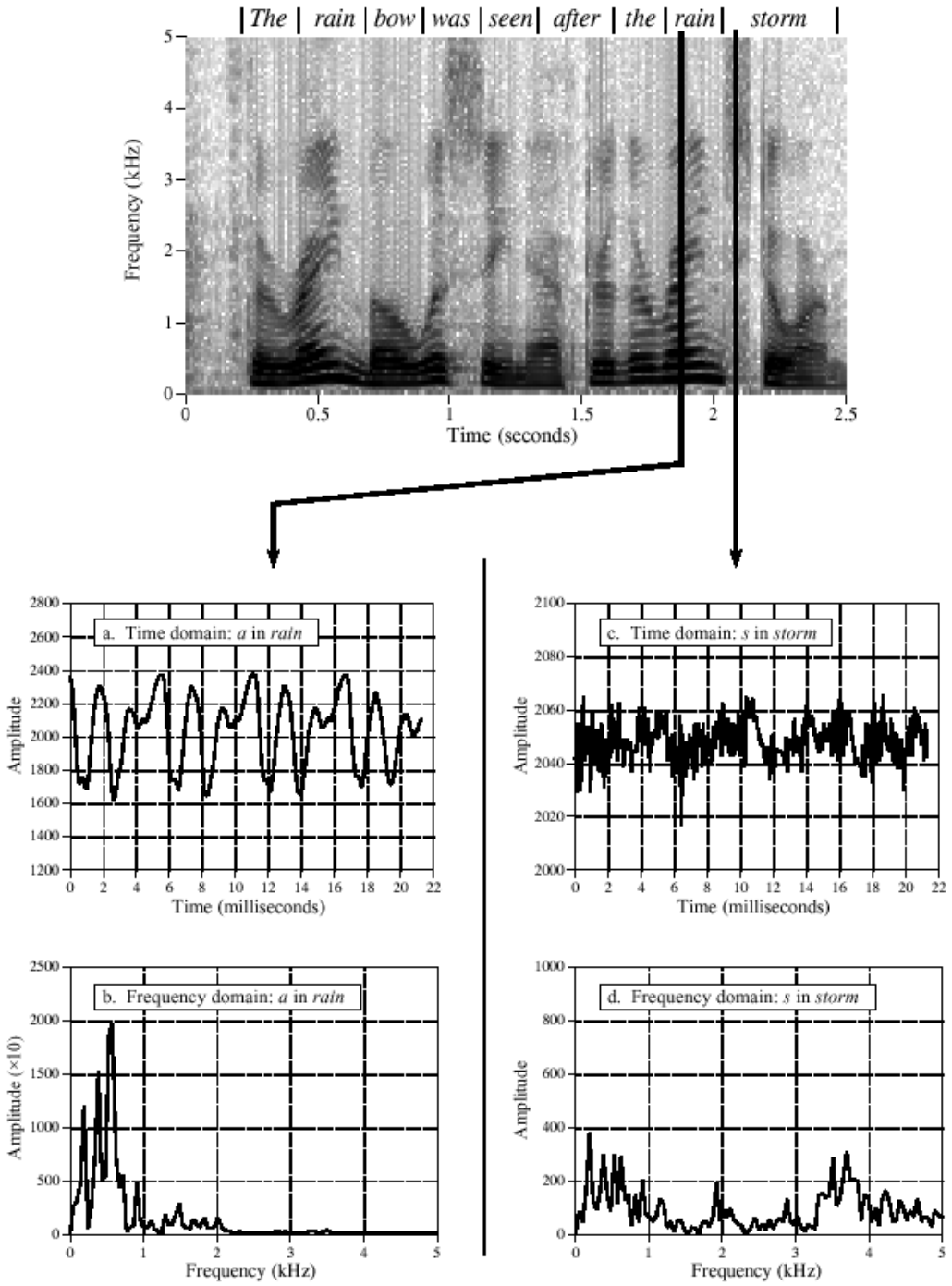
هر دو نوع این اصوات، توسط چاله‌های صوتی که از زبان، لبها، دهان، گلـو و گـذرگاههای بینی تشکیل شده‌اند دچار تغییر می‌شوند. چون انتشار صدا در این سـاختارها یـک فراینـد خطـی است می‌تواند با استفاده از یک فیلتر خطی با یک پاسخ ضربهٔ مناسب نمایش داده شود. در بیشـتر موارد از یک فیلتر بازگشتی که ضرایب بازگشتی آن ویژگیهای فیلتر را مشخص می‌کنـد اسـتفاده

می‌شود. به خاطر این که چاله‌های صوتی ابعادی به اندازهٔ چند سانتیمتر دارند پاسخ فرکانسی یک دنباله از تشدیدها با اندازه‌های کیلوهرتزی است. در اصطلاح پردازش صوت این قله‌های تشدید **فرکانسهای فرمانت**[۷۲] خوانده می‌شوند. با تغییر جایگاه نسبی زبان و لبها فرکانسهای فرمانت هم از لحاظ دامنه و هم از لحاظ فرکانس ممکن است تغییر کنند.

شکل شمارهٔ ۴ (صفحهٔ بعد) یک روش معمول برای نمایش سیگنالهای صحبت را نشان می‌دهد که طیف‌نگاره[۷۳] یا اثر صوت[۷۴] خوانده می‌شود. سیگنال صوتی به تکه‌های کوچک به اندازهٔ ۲ تا ۴۰ میلی‌ثانیه تقسیم می‌شوند و از الگوریتم اف.اف.تی برای یافتن طیف فرکانسی هر تکه استفاده می‌شود. این طیفها در کنار هم قرار داده شده تبدیل به یک تصویر سیاه و سفید[۷۵] می‌شود (دامنه‌های پایین روشن و دامنه‌های بالا تیره می‌شوند). این کار یک روش گرافیکی برای مشاهدهٔ این که چگونه محتویات فرکانسی صحبت با زمان تغییر می‌کند به وجود می‌آورد. اندازهٔ هر تکه بر اساس اعمال یک بده‌بستان بین دقت فرکانسی (که با تکه‌های بزرگ‌تر بهتر می‌شود) و دقت زمانی (که با تکه‌های کوچک‌تر بهتر می‌شود) انتخاب می‌گردد.

همچنانکه در شکل ۲- ۵ دیده می‌شود اصوات صدا دار مثل a در rain دارای موج صوتی متناوبی مانند آنچه در شکل a نشان داده شده و طیف فرکانسی آنها که

[72] formant frequencies
[73] specrogram
[74] voice print
[75] grayscale



شکل شمارهٔ ۲- ۵ طیف صوت.

شکلهای a و b ویژگیهای عمومی اصوات صدادار و شکلهای c و d ویژگیهای عمـومی اصـوات سایشی را نمایش می‌دهند.

عبارت است از یک دنباله از همسازهای با اندازهٔ منظم مانند شکل b می‌باشد در مقابل، اصوات سایشی مانند s در storm دارای یک سیگنال نویزی در دامنهٔ زمان مانند شکل c و یک طیف نویزی مانند شکل d هستند.این طیفها همچنین شکل فرکانسهای فرمانت برای هر دو نوع صوت نشان می‌دهند. همچنین به این نکته توجه کنید که نمایش زمان-فرکانس کلمهٔ rain در هر دو باری که ادا شده شبیه به هم است.

در یک دورهٔ کوتاه برای نمونه ۲۵ میلی‌ثانیه یک سیگنال صحبت می‌تواند با مشخص کردن سه پارامتر تقریب زده شود:

۱) انتخاب یک اغتشاش متناوب یا نویزوار

۲) فرکانس موج متناوب (اگر مورد استفاده قرار گرفته باشد)

۳) ضرایب فیلتر دیجیتالی که برای تقلید پاسخ تارهای صوتی استفاده شده است.

صحبت پیوسته با بروزآوری این سه پارامتر به صورت پیوسته به اندازهٔ ۴۰ بار در ثانیه ترکیب شود. این راهکار برای یکی از کاربردهای تجاری دی.اس.پی که «صحبت و املا» نامیده می‌شود و یک وسیلهٔ الکترونیکی پرفروش برای بچه‌هاست مناسب است. کیفیت صدای این نوع ترکیب کنندهٔ صحبت پایین است و بسیار مکانیکی و متفاوت با صدای انسان به نظر می‌رسد. ولی در هر صورت نرخ دادهٔ خیلی پایینی در حدود چند کیلوبیت بر ثانیه نیاز دارد.

همچنین این راهکار پایه‌ای برای روش **کدگذاری پیشگویانهٔ خطی**[۷۶] (ال.پی.سی) در فشرده‌سازی صحبت فراهم می‌آورد. صحبت ضبط شدهٔ دیجیتالی انسان به تکه‌های کوچک تقسیم می‌شود و هر کدام با توجه به سه پارامتر مدل توصیف می‌شود. این عمل به طور معمول نیاز به یک دوجین بایت برای هر تکه دارد که نرخ داده‌ای برابر با ۲ تا ۶ کیلوبایت بر ثانیه را طلب می‌کند. این تکهٔ اطلاعاتی ارسال می‌شود و در صورت لزوم ذخیره می‌گردد و سپس توسط ترکیب کنندهٔ صحبت بازسازی می‌شود.

[76] Linear Predictive Coding

الگوریتمهای تشخیص صحبت با تلاش برای شناسایی الگوهای پارامترهای استخراج شده از این روش نیز پیش‌تر می‌روند. این روشها معمولاً شامل مقایسهٔ تکه‌های اطلاعاتی با قالبهای صدای از پیش ذخیره شده در تلاش برای تشخیص کلمات گفته شده می‌باشند. مشکلی که در اینجا وجود دارد این است که این روش همیشه به درستی کار نمی‌کند. این روش برای بعضی کاربردها قابل استفاده است اما با تواناییهای شنوندگان انسانی خیلی فاصله دارد.

۳– ۵ آیندهٔ فن‌آوریهای پردازش صحبت

ارزش ایجاد فن‌آوریهای ترکیب و تشخیص صحبت بسیار زیاد است. صحبت سریع‌ترین و کاراترین روش ارتباط انسانهاست. تشخیص صحبت پتانسیل جایگزینی نوشتن، تایپ، ورود صفحه‌کلید و کنترل الکترونیکی را که توسط کلیدها و دکمه‌ها اعمال می‌شود را داراست و فقط نیاز به آن دارد که کمی برای پذیرش توسط بازار تجاری بهتر کار کند.

ترکیب صحبت علاوه بر آن که همانند تشخیص صحبت می‌تواند استفاده از کامپیوتر را برای کلیهٔ افراد ناتوان بدنی که دارای تواناییهای شنوایی و گفتاری مناسب هستند آسان‌تر سازد به عنوان یک وسیلهٔ خروجی کاربرپسند در محیطهای مختلف می‌تواند با جایگزین کردن بسیاری از علائم دیداری (انواع چراغها و...) و شنوایی (انواع زنگهای اخطار و ...) با گفتارهای بیان کنندهٔ کامل پیامها استفاده از و رسیدگی به سیستمهای نیازمند این گونه پیامها را بهینه کند.

در اینجا لازم است به این نکته اشاره شود که پیشرفت در فن‌آوری تشخیص صحبت (و همچنین تشخیص گوینده) همان قدر که محدودهٔ دی.اس.پی را در بر می‌گیرد نیازمند دانش به دست آمده از محدوده‌های هوش مصنوعی و شبکه‌های عصبی است. شاید این تنوع دانشهای مورد نیاز به عنوان عامل دشواری مطالعهٔ مبحث پردازش صحبت در نظر گرفته شود حال آن که این گونه نیست و این تنوع راهکارها بخت رسیدن به سیستم با کارایی مطلوب را افزایش می‌دهد.

تواناییهای ابزارهایی که در بخش اول این فصل به آنها اشاره شد امیدواریهای فراوانی را در زمینهٔ موفقیت ابزارهای موجود فراهم می‌آورد و دامنهٔ وسیع شرکتها و مراکز دانشگاهی که در این زمینه فعالیت می‌کنند بر تنوع در قابلیتها و کاربردهای پیاده‌سازی شدهٔ این ابزارها می‌افزاید.

بخش ششم

مدلسازی سیگنال

۶–۱ اهمیت مدلسازی سیگنال

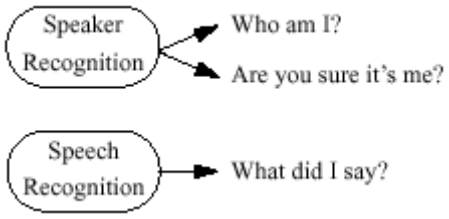
تشخیص کامپیوتری صحبت در واقع بر دارندهٔ دو نوع عمل اصلی تشخیص است: تشخیص صحبت و تشخیص گوینده. با تحلیل یک موج صوتی می‌توان خصیصه[۷۷]های اندامهای گفتاری گوینده را تخمین زد که این خصیصه‌ها راهکاری برای تشخیص هویت و تصدیق آن به روش زیست‌سنجی فراهم می‌آورند. در مقابل، سیستمهای تشخیص صحبت برای درک مفهوم موج صوتی گفته شده تلاش می‌کنند. جهت بیشتر تحقیقات فعلی در فن‌آوری تشخیص صحبت به سمت ایجاد سیستمهای مستقل از گوینده است که توانایی تبدیل صحبت همهٔ گویندگان را داشته باشد. در حالی که اهداف این دو نوع سیستم کاملاً متفاوت به نظر می‌رسند هر دو عمیقاً از آبشخوری به نام الگوریتمهای پردازش سیگنال برای استخراج خصیصه‌ها تغذیه می‌شوند. در هر دو زمینه تلاش برای پیدا کردن دسته‌ای از خصیصه‌ها که در مقابل تغییرات محیطی پایدار باشند ادامه دارد. این قسمت مروری خواهد داشت بر الگوریتمهای استخراج خصیصه‌ها[۷۸] که در هر دو زمینه استفاده شده‌اند و شامل ارزیابی کوتاهی از الگوریتمهای گوناگون مدلسازی سیگنال با آزمایشهای تشخیصی کوچک می‌باشد.

۶–۲ آشنایی با مدلسازی سیگنال

هدف سیستمهای تشخیص گوینده بازشناسی خصیصه‌های اندامهای گفتاری و حالت صحبت کردن با استفاده از صدای گوینده به منظور اهداف تشخیص هویتی می‌باشد. ساختار اندامهای صوتی، اندازهٔ چالهٔ بینی و ویژگیهای تارهای صوتی همگی با استفاده از تحلیل سیگنال قابل تخمین هستند. تشخیص گوینده اصطلاحی کلی است که به اعمال تشخیص هویت گوینده و

[77] feature
[78] feature extraction

تأیید هویت گوینده اطلاق می‌گردد. برای تشخیص، خصیصه‌های تخمینی گوینده با خصیصه‌های موجود در یک پایگاه داده‌ها از کاربران ثبت شده برای یافتن نزدیک‌ترین خصیصه‌های قابل تطبیق مقایسه می‌شوند. برای تأیید هویت، ادعای هویتی گوینده بر اساس امضای زیست‌سنجی وی پذیرفته می‌شود و یا رد می‌گردد.



شکل شمارهٔ ۱ – ۶ وظایف مختلف

تشخیص صحبت تلاش دارد تا یک سیگنال صوتی صحبت را به واژه‌ها تبدیل کند. انسانها واژه‌ها را با حرکت دادن اندامهای صوتی به یک سری از مکانهای قابل پیشبینی ادا می‌کنند. اگر این دنباله‌ها از سیگنال استخراج گردند واژه‌های گفته شده می‌توانند تشخیص داده شوند. بسیاری از کاربردهای تشخیص صحبت نیازمند سیستمهای مستقل از گوینده می‌باشند این تولیدات می‌توانند صحبت هر گوینده‌ای را تشخیص دهند.

اگر چه این دو هدف کاملاً متفاوت به نظر می‌رسند هر دوی آنها بر روی داده‌های صحبت تشخیص الگو را اعمال می‌کنند. بعضی از سیستمهای موجود مانند Nuance 6 server هم تشخیص صحبت و هم تأیید هویت گوینده را به صورت همزمان اعمال می‌کنند. به خاطر همین شباهت رویه هر دوی این فن‌آوریها از یک نقطه ضربه می‌خورند: یک تنزل کارایی شدید در اثر تفاوتهای محیطهای آموزشی و آزمایشی به وجود می‌آید. به طور خلاصه کارایی این فن‌آوریها شدیداً به محیطی که در آن توسعه می‌یابند وابسته است و بنابراین حالات پر از نویز جهان واقعی آنها را به کارایی زیر کارایی بهینه راهبری می‌کند.

الگوریتمهایی مورد استفادهٔ محصولات پردازش کنندهٔ صحبت بر اساس مدل صوتی ناحیهٔ صوتی و کانال گوش استوارند. بخش بعدی اهمیت استخراج خصیصه‌ها را با یک مرور کلی از تشخیص الگو روشن می‌کند و سپس با توصیف الگوریتمهای رایج در محصولات پراستفاده ادامه پیدا می‌کند.

۳-۶ تشخیص الگو

یک سیستم تشخیص الگو شامل دو جزء است: یک استخراج کنندهٔ خصیصه‌ها و یک طبقه‌بندی کننده. ایده‌آل آن است که وقتی داده‌ها به فضای داده‌های خصیصه‌ها انتقال پیدا کرد به سمت طبقه‌ای کشیده شود که از همه به آن نزدیک‌تر است و از طرف طبقه[۷۹]های متفاوت دیگر بازپس زده شود. وقتی که به طبقه‌بندی کننده[۸۰] آموزش داده شد که بین طبقه‌ها در این فضای انتقال داده شده از خصیصه‌ها تمایز قائل شود یک سیستم تشخیص نیازمند آن است که تنها داده‌های ورودی را از طریق همان سیستم استخراج خصیصه‌ها انتقال دهد و مشخص کند که در کدام طبقه یک مشاهدهٔ جدید رخ می‌دهد.

دو مشکل مهم در اعمال این راهکار به پردازش صحبت وجود دارد. اولی آن است که هیچ التزامی وجود ندارد که محیط آموزش و محیط آزمایش قابل مقایسه باشند. استفاده از یک میکروفون متفاوت، نویز پس‌زمینه و کانالهای انتقال می‌تواند باعث کاهش کارایی جدی شود (یک معیار اساسی برای قضاوت در مورد یک مجموعه از خصیصه‌ها پایداری آن در مقابل چنین تغییرات کانالی می‌باشد) . دومین مشکل آن است که که برهم‌نهی زیادی بین طبقه‌های موجود در فضای خصیصه‌ها وجود دارد. ژائو[۸۱] نمودارهایی برای نشان دادن این برهم‌نهی در دودسته داده‌های صحبت جمع‌آوری شده از طریق شبکهٔ تلفن ارائه می‌کند. موتورهای تشخیص صحبت برای غلبه

[79] class
[80] classifier
[81] Zhao

بر این مشکل برهم‌نهی از پردازشهای آماری توانمند برای یکسان‌سازی مدل زبان استفاده می‌کنند که فراتر از حد این نوشتار است.

۶-۴ الگوریتمهای مدلسازی سیگنال

هدف مدلسازی سیگنال (که اغلب از آن با عنوان استخراج خصیصه‌ها یاد می‌شود) انتقال داده‌های صوتی به فضایی است که مشاهدات مربوط به یک طبقه با هم در یک گروه قرار گیرند و مشاهدات مربوط به طبقات متفاوت از هم جدا شوند. این انتقالها بر اساس مطالعات زیست‌شناختی سیستمهای صوتی و اندامهای گفتاری انسان انتخاب می‌شوند. برای مثال اندامهای گفتاری نمی‌توانند از یک مکان به مکان دیگر در کمتر از حدود پنج میلی‌ثانیه جابه‌جا شوند لذا سیستمهای عملی می‌توانند از طیف ۱۰۰ بار در ثانیه نمونه‌برداری کنند در حالی که از دقت عملیات فقط مقدار بسیار کمی کاسته شود.

صحبت یک سیگنال پویاست لذا ما علاقمند به آزمون طیف بازهٔ کوچک هستیم. زمان استمرار یک قاب به صورت طول زمانی که یک مجموعه از پارامترها معتبر هستند تعریف می‌شوند. با وجود این که قابها همپوشانی ندارند ما معمولاً از پنجرهٔ تحلیل دارای همپوشانی برای در نظر داشتن تعداد بیشتری از نمونه‌های سیگنال برای هر اندازه‌گیری طیف استفاده می‌کنیم. اعمال مستقیم تحلیل طیفی بر روی چنین مقدار کمی از داده‌ها معادل با اعمال یک پنجرهٔ مستطیلی تیز به سیگنال است که باعث ایجاد اعوجاج طیفی می‌شود. پاسخ فرکانسی پالس مستطیلی یک تابع sinc می‌باشد($\text{sinc } x=\sin x/x$) که دارای یک باند عبور منحنی شکل و مقدار زیادی ناهمواری در باند توقف می‌باشد. شکلهای مختلف برای پنجره‌ها از طریق اعمال یک تابع وزن به دست می‌آیند. پنجرهٔ همینگ[۸۲] با رابطهٔ

$$w(n)= (\alpha-(1-\alpha)\cos(2\pi/(N-1))/ \beta$$

[82] Hamming window

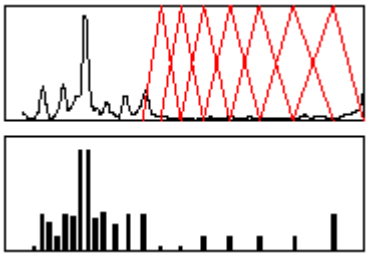
یک نمونهٔ ویژه از پنجرهٔ هنینگ[۸۳] با $\alpha = 0.54$ می‌باشد. پارامتر β برای هنجارسازی به گونه‌ای انتخاب می‌شود که انرژی سیگنال در خلال آزمایش بدون تغییر باقی بماند. شکل پنجرهٔ همینگ یک تحلیل طیفی با باند عبور هموارتر و باند توقف به طور قابل ملاحظه‌ای بدون اعوجاج به دست می‌دهد که هر دوی این خصوصیات برای به دست آوردن تخمینهای پارامتری متغیر مهم هستند. بیشتر سیستمهای امروزی از یک از یک فریم با اندازهٔ زمانی ۱۰ میلی‌ثانیه و یک پنجره با اندازهٔ زمانی ۲۵ میلی‌ثانیه استفاده میکنند.

یک خصیصهٔ استخراج شده از سیگنال انرژی مطلق سیگنال است. دستهٔ دیگر، اندازه‌گیری طیفی انرژی فرکانسهای خاص است. این اندازه‌ها مشابه حالات اولیهٔ حرکات دستگاه صوتی انسان هستند (سلولهای مو در حلزون گوش برای دستیابی به هدف مشابهی استفاده می‌شوند). سه راه برای دستیابی به این اندازه‌های صوتی وجود دارد: اعمال مستقیم یک بانک فیلتر دیجیتال در دامنهٔ زمان، استفاده از تبدیل فوریه و تحلیل پیشگویانهٔ خطی. دو روش اخیر به لحاظ کارایی محاسباتی در سیستمهای امروزی رایج‌ترند.

از آنجا که شنوایی انسان در طول یک اندازهٔ خطی به صورت مساوی حساس نیست، ما طیف را به یک اندازهٔ فرکانسی قابل درک[۸۴] نقش می‌کنیم. تجربیات در مورد ادراک انسان نشان داده‌اند که فرکانسهایی با یک پهنای باند معینِ یک فرکانس اسمی که به پهنای باند بحرانی معروف است نمی‌توانند به صورت جداگانه از هم تشخیص داده شوند. اندازهٔ مل[۸۵] یک تقریب ساده‌تر است که پیچ قابل مشاهدهٔ یک صدا را به اندازهٔ خطی نقش می‌کند. استیونز[۸۶] و فولکمن[۸۷] در سال ۱۹۴۰ به صورت تجربی نگاشتی بین اندازهٔ مل و فرکانسهای واقعی تعیین کردند. تفاوت اندازه به سختی به صورت خطی زیر ۱۰۰۰هرتز و به صورت لگاریتمی بالای ۱۰۰۰هرتز می‌باشد.

---

[83] Hanning window
[84] perceptual
[85] mel scale
[86] Stevens
[87] Volkman



شکل شمارهٔ ۲- ۶ بانکهای فیلتر با فضای مثلثی مل

بانکهای فیلتر مبتنی بر تبدیل فوریهٔ ساده که برای خصیصه‌های نهایی طراحی شده‌اند دقت فرکانسی دلخواه را بر اساس مقیاس مل[۸۸] به دست می‌دهند. برای پیاده‌سازی این بانک فیلتر پنجرهٔ داده‌های صحبت با استفاده از تبدیل فوریه به دامنهٔ فرکانس انتقال می‌یابد. در دامنهٔ فرکانس ضرایب دامنهٔ هر بانک فیلتر با اعمال یک ترکیب خطی از طیف و پاسخ فرکانسی فیلتر دلخواه پیدا می‌شوند. در عمل بانکهای فیلتر مثلثی دارای برهم‌نهی استفاده می‌شوند که در آن از فرکانس مرکزی یک فیلتر به عنوان نقاط انتهایی دو فیلتر مجاور استفاده می‌شود. بنابراین ضرایب دامنهٔ هر بانک فیلتر مقدار متوسط طیف در کانال فیلتر را نشان می‌دهند:

$$S_{avg}(f) = \frac{1}{N_s} \sum_{n=0}^{N_s} w_{FB}(n)|S(f)|$$

که در آن N(s) تعداد نمونه‌های استفاده شده برای دستیابی به مقدار متوسط و W(n) تابع وزنیابی (مشابه تابع مثلثی که قبلاً توضیح داده شد) می‌باشد و S(f) مقدار پاسخ فرکانسی است که با تبدیل فوریه محاسبه می‌شود.

تحلیل پیشگویانه خطی[۸۹] وسیله‌ای برای به دست آوردن پوشش طیفی هموار P(w) از یک مدل تمام- قطب طیف توان است. ضرایب خطی پیشگو همبستگی مستقیمی با نسبتهای ناحیهٔ لگاریتمی که پارامترهای هندسی مدل لوله‌ای نقصان برای تولید صحبت هستند دارد.

[88] Mel frequency
[89] Linear Predictive (LP) anlaysis

دامنه‌های بانک فیلتر با نمونه‌برداری از مدل طیفی پیشگویانهٔ خطی در فرکانسهای بانک فیلتر مناسب به دست می‌آیند. این کار می‌تواند با ارزیابی مستقیم مدل ال.پی.سی انجام شود ولی در عمل تبدیل فوریه بر روی ضرایب پیشگو اعمال می‌شود. چون تعداد ضرایب ال.پی.سی کمتر از نمونه‌های صوت است این روش از لحاظ محاسباتی کاراست. ضرایب دامنهٔ بانک فیلتر همان گونه که از طیف حاصل از تبدیل فوریه[۹۰] به دست می‌آمدند از طیف حاصل از پیشگویانهٔ خطی[۹۱] به دست می‌آیند.

یک سیستم همریخت[۹۲] برای پردازش صحبت قابل استفاده است زیرا روشی برای جدا کردن سیگنال آشفتگی از شکل ناحیهٔ صوتی فراهم می‌آورد. یک فضای دارای این ویژگی سپستروم[۹۳] است که با محاسبهٔ عکس تبدیل فوریهٔ گسستهٔ لگاریتم انرژی به دست می‌آید.ضرایب سپسترال[۹۴] با محاسبهٔ دامنه‌های بانک فیلتر با استفاده از معادلهٔ زیر به دست میآیند:

$$c(n) = \frac{1}{N_s} \sum_{k=0}^{N_s} \log\left|S_{avg}(k)\right| e^{j\frac{2\pi}{N_s}kn}, \quad 0 \le n \le N_s - 1$$

که $S(avg)$ مقدار متوسط سیگنال در کانال kام فیلتر است. در عمل تبدیل کسینوسی گسسته به خاطر کارایی محاسباتی استفاده می‌شود. ضرایب سپسترال اغلب برای کمینه کردن تغییراتی که منجر به ایجاد اطلاعات نمی‌شوند وزنیابی می‌گردند که این پردازه لیفترینگ[۹۵] نامیده می‌شود. جالب است بدانیم که در ادبیات تشخیص صحبت خصیصه‌های مربوط به گوینده به عنوان تغییرات غیر داده‌زا حذف می‌گردند ولی سیستمهای تشخیص گوینده نیز از لیفترینگ استفاده می‌کنند.

[90] FT-deriven spectrum
[91] LP-deriven spectrum
[92] homomorphic
[93] cepstrum
[94] cepstral
[95] liftering

هر دو نوع سیستم تشخیص صحبت و تشخیص گوینده اطلاعات موضعی زمان کوتاه را با گرفتن مشتق خصوصیات اولیه نسبت به زمان به دست می‌دهند. به عنوان مثال یک صوت صدادار می‌تواند با پیدا شدن فرمانتهای[۹۶] آن در طیف تشخیص داده شود، حال آن که یک صوت بی‌صدا (سایشی) با استفاده از انتقال طیف مدل می‌شود. مقادیر مشتق مرتبهٔ اول خصائص ضرایب دلتا[۹۷] و مقادیر مشتق مرتبهٔ دوم آن شتاب[۹۸] یا ضرایب دلتا- دلتا[۹۹] نامیده می‌شوند.مشتق زمانی با استفاده از یک رابطهٔ رگرسیون که یک مجموعه فریم را پیش و پس از فریم کنونی می‌کشد تقریب زده می‌شود.

سیستمهای تشخیص گوینده از یک پیمانهٔ انتخاب خصیصه نیز در چارچوب تشخیص الگو استفاده می‌کنند. برای تشخیص صحبت تمامی سیگنال باید به یک نمایش متنی نگاشته شود حال آن که سیستم تشخیص گوینده نیازی به کار تحت این اجبار ندارد. بنابراین پیمانهٔ انتخاب خصیصه فقط خصیصه‌ها مربوط به اصوات صدادار را ذخیره می‌کند. اصوات صدادار مستقیماً فرضیات مدلسازی پیشگویانهٔ خطی را برآورده

می‌سازند و کمتر تحت تأثیر نویز صوتی قرار می‌گیرند.

[96] formants
[97] delta coefficients
[98] acceleration
[99] delta-delta coefficients

بخش هفتم

روشهای طراحی سیستمهای تشخیص گوینده

۱-۷ مقدمه

همچنان که پیش از این گفته شد سیستمهای تشخیص گوینده در حالت کلی به دو نوع سیستمهای تأیید هویت گوینده[۱۰۰] و سیستمهای بازشناسی گوینده[۱۰۱] تقسیم می‌شوند. تفاوت این دو سیستم در نحوهٔ پذیرش ورودی است: در سیستمهای نوع اول گوینده با ارائهٔ یک شناسه ادعای هویت یک کاربر خاص را می‌نماید حال آن که در سیستمهای نوع دوم گوینده فقط عبارت عبور خود را بیان می‌کند و سیستم او را از بین تمامی کاربران خود تشخیص می‌دهد.

در فصل قبل در مورد ساختار الگوهای مورد بحث صحبت کردیم و متوجه شدیم که عمل مدلسازی سیگنال یا استخراج خصیصه‌ها[۱۰۲] با حذف ویژگیهای بدون استفادهٔ سیگنال صحبت و حفظ ویژگیهای قابل استفاده برای بازشناسی عبارات خاص الگوهایی را با ویژگیهای انتخاب شده در اختیار ما قرار می‌دهد.

ساختارهایی که برای هر دو نوع سیستم ارائه شد هر دو دارای یک مرحله برای تشخیص میزان شباهت الگوهای متعلق به گویندهٔ حاضر با گویندهٔ مورد ادعا (نوع اول) یا همهٔ گویندگان است که با استفاده از آن معیاری برای تصمیم گیری در اختیار ما قرار داده می‌شود.

همچنان که برای تشخیص الگو الگوریتمهای متعدد و روشهای گوناگون وجود دارد الگوریتمهای گوناگونی نیز برای یافتن میزان شباهت میان الگوها وجود دارد که انتخاب هر کدام از آنها بستگی به ساختار سیستم مقصد دارد.

انتخاب یک روش به ویژگیهای سیستم هدف بستگی دارد. بعضی از روشهای موجود تنها می‌توانند فقط برای سیستمهای وابسته به متن[۱۰۳] یا فقط برای سیستمهای مستقل از متن[۱۰۴] مورد استفاده قرار گیرند و بعضی می‌توانند برای هر دو نوع مورد استفاده قرار گیرند.

---

[100] Speaker Verification
[101] Speaker Identification
[102] feature extraction
[103] text-dependent
[104] text-independent

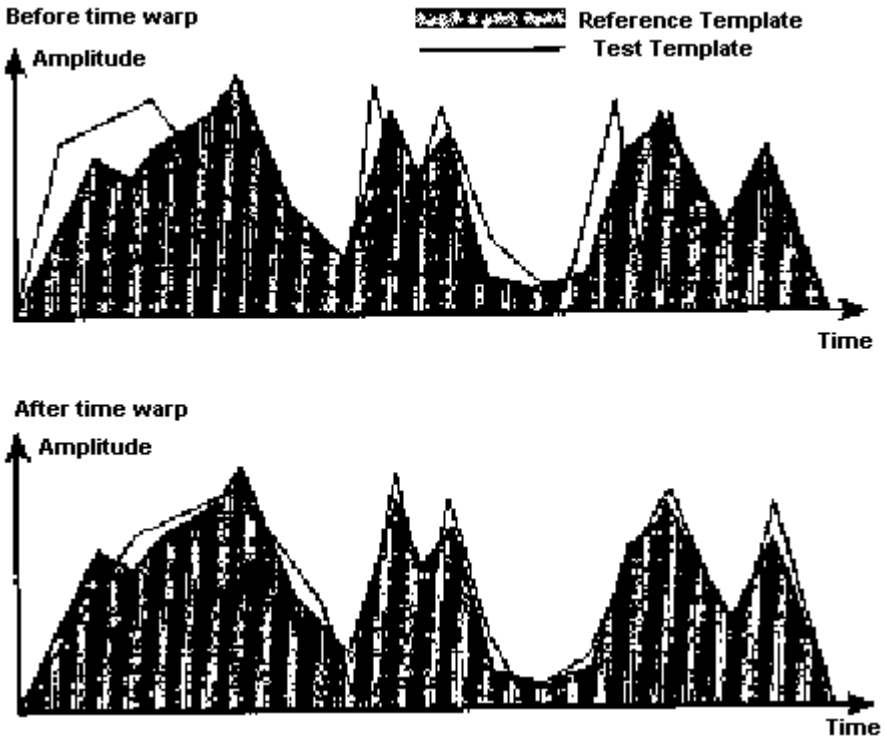
بحث این فصل که سه روش عمدهٔ یافتن میزان شباهت الگوها را بـه صـورت کلـی مـورد بحث قرار خواهد داد عملاً پیش‌زمینه‌های نظری لازم برای طراحی سیستم هدف را کامل می‌کند.

۲-۷ روشهای مبتنی بر چشمپوشی زمانی پویا[۱۰۵]

این روش کلاسیک برای تشخیص خودکار گوینده در حالت وابسـته بـه مـتن بـر اسـاس یکسان‌سازی الگوها با استفاده از الگوهای طیفی[۱۰۶] یا روش طیف‌نگاره[۱۰۷] اسـتوار اسـت. در حالـت کلی سیگنال صحبت به صورت یک دنباله از بردارهای خصیصه[۱۰۸] کـه رفتـار سـیگنال صـحبت را برای یک گویندهٔ خاص مشخص می‌کند نمایش داده می‌شود. یک الگـو مـی‌توانـد نمایشـگر یـک عبارت چند کلمه‌ای، یک کلمهٔ منفرد، یک هجا یا یک صدای ساده باشد.

در روشهای یکسان‌سازی الگوها مقایسه‌ای بین الگـوی عبـارت ورودی و الگـوی مرجـع بـرای تشخیص هویت گوینده انجام می‌گیرد. یک جزء مهم در این روشها بهنجارسازی تغییرات زمانی هر آزمون تا آزمون بعدی می‌باشد. بهنجارسازی می‌تواند با روش چشمپوشی زمانی پویا صورت گیـرد. این روش یک تابع بهینهٔ توسیع/ فشرده‌سازی زمانی را برای ایجاد صف‌بندی زمـانی غیرخطـی بـه کار می‌گیرد. شکل ۱ الگوها را پیش و پس از اعمال این روش نشان می‌دهد. به ایـن نکتـه توجـه شود که چگونه چشمپوشی الگوهای نمونهٔ آزمون میزان نزدیکی دو الگو را افزایش داده است:

[105] Dynamic Time Wrapping (DTW)
[106] spectral templates
[107] spectogram
[108] feature vector



شکل شمارهٔ ۱ – ۷ نمونهٔ یک الگو پیش و پس از اعمال روش چشمپوشی زمانی پویا

در شکل شمارهٔ ۱- ۷ فریمهای صحبت که الگوهای آزمون و مرجع را به وجود می‌آورند به صورت مقادیر دامنه‌ای اسکالر بر روی نموداری که محور افقی آن نشانگر زمان است نشان داده شده‌اند. بنابراین یک تابع تصمیم‌گیری با جمع‌آوری اندازه‌گیریها بر حسب زمان می‌تواند محاسبه شود. در عمل الگوها بردارهای چند بعدی هستند و فاصله بین آنها به صورت فاصلهٔ اقلیدسی[۱۰۹] مورد محاسبه قرار می‌گیرد. نوع دیگر فاصله که برای مقایسهٔ دو مجموعه از ضرایب پیشگویانهٔ خطی مورد استفاده قرار می‌گیرد فاصلهٔ ایتاکورا[۱۱۰] می‌باشد.

۷–۳ روشهای مبتنی بر مدلهای نهان مارکف[۱۱۱]

روشهای مبتنی برمدل نهان مارکف جایگزینهایی برای روش یکسان‌سازی الگوها که توسط روشهای چشمپوشی زمانی پویا ارائه شد می‌باشند که مدلهای احتمالی از سیگنال صحبت

[109] Euclidean distance
[110] Itakura distance
[111] Hidden Markov Model (HMM)

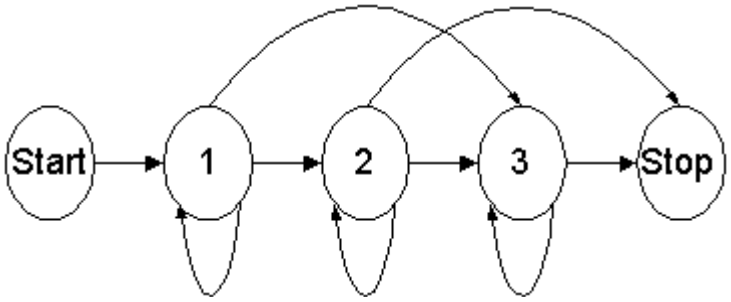
به وجود می‌آورند که ویژگیهای متغیر با زمان آن را توصیف می‌کند. یک مدل نهـان مـارکف یـک فرایند اتفاقی[۱۱۲] دوگانه برای ایجاد یک دنباله از نشانه‌های مشاهده شده است. معنای دوگانه بودن این فرایند اتفاقی آن است که این فرایند دارای یک زیرفرایند اتفاقی دیگر است که قابل مشـاهده نمی‌باشد (از اینجا مفهوم عبارت نهان مشخص می‌گردد) ولی می‌تواند توسط فرایند اتفاقی دیگری که یک دنباله از مشاهدات را ایجاد می‌کند مشاهده گـردد. در سیسـتمهای نشـخیص صـحبت یـا تشخیص گوینده دنبالهٔ موقتی طیف صوتی می‌تواند به صورت یک زنجیـرهٔ مـارکف[۱۱۳] مدلسـازی شود تا روشی را که یک صدا به صدای دیگری تبدیل می‌شود توصیف کند. این عمل سیستم را تـا اندازهٔ یک مدل که قادر است فقط در یکی از یک تعداد متناهی از حالات متفاوت باشد (به عنـوان نمونه یک ماشین حالت متناهی[۱۱۴]) کوچک می‌کند. روشهای مبتنی بر مدل نهان مارکف می‌توانند هم در سیستمهای وابسته به متن و هم در سیستمهای مستقل از متن مورد استفاده قرار گیرند.

وقتی که بعد از یک انتقال حالت وارد یک حالت دیگر در ماشین حالت متناهی می‌شـویم یک نشانه از مجموعه نشانه‌های آن حالت به عنوان خروجی برگزیده می‌شود. خروجی می‌تواند یک تعداد متناهی (روش مدل نهان مارکف گسسته) و یا یک مقدار پیوسته از خروجیهـا (روش توزیـع پیوسته) باشد. هر دو مدل به صورت مـؤثر اطلاعـات مـوقتی را مدلسـازی مـی‌کننـد. سیسـتم در بازه‌های منظم زمانی تغییر حالت می‌دهد. حالتی که مدل در هر آغاز هر بازهٔ زمانی به آن مـی‌رود به احتمالات بستگی دارد.

تعدادی توپولوژی مدل که برای نمایش ماشین حالت متناهی استفاده می‌شوند وجـود دارنـد. یک ساختار معمول ساختار چپ به راست است که به آن مدل بکیس[۱۱۵] هم گفته می‌شود و مثال آن نمونه‌ای است که در شکل ۲ -۷ نشان داده شده است. هر حالـت یـک انتقـال توقـف[۱۱۶]، یـک

[112] doubly stochastic process
[113] Markov chain
[114] Finite State Machine (FSM)
[115] Bakis model
[116] stay transition

انتقال پیش‌رونده[۱۱۷] و یک انتقال جهشی[۱۱۸] دارد. با وجود آن که در شکل نشان داده نشده است احتمالهای مختلفی به انتقالهای حالت متناهی وابسته‌اند و همچنین خروجی هر حالت را کنترل می‌کنند. نوع دیگر توپولوژی مدل نهان مارکف که در اینجا نشان داده نشده ساختار ارگودیک[۱۱۹] می‌باشد که در آن همانند یک شبکهٔ کاملاً متصل به هم هر حالت به همهٔ دیگر حالات دارای انتقال است.



شکل شمارهٔ ۲ – ۷ مثالی از ساختار مدل نهان مارکف چپ به راست

۷–۴ روشهای مبتنی بر مقدارگزینی برداری[۱۲۰]

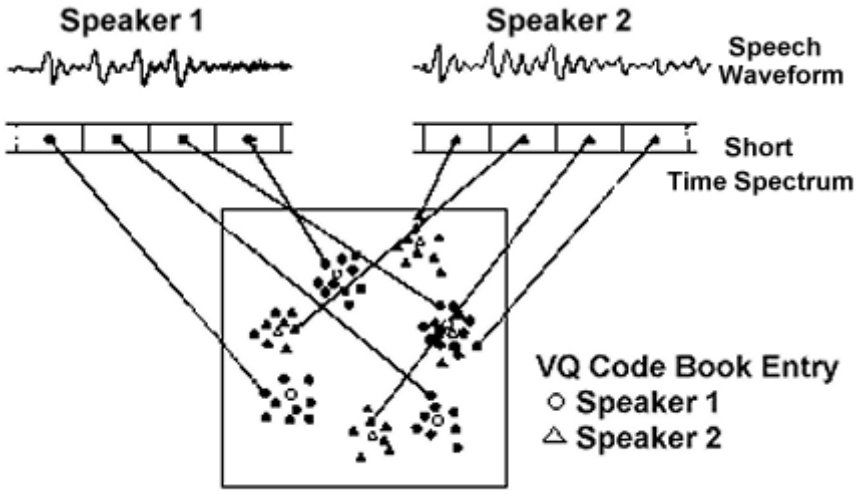
یک مجموعه از بردارهای خصیصهٔ بازهٔ کوتاه زمانی یک گوینده که برای آموزش سیستم به سیستم داده می‌شوند می‌توانند مستقیماً برای نمایش ویژگیهای مهم عبارت ایراد شده توسط وی به کار گرفته شوند. در هر صورت نتیجهٔ کار آن است که نیازمندیهای حافظه برای ذخیرهٔ داده‌ها و پیچیدگی محاسباتی به سرعت با افزایش تعداد بردارهای آموزش دهندهٔ سیستم افزایش می‌یابد. بنابراین یک نمایش مستقیم عملی نخواهد بود.

مقدارگزینی برداری اساساً روشی برای فشرده‌سازی داده‌های آموزش دهندهٔ سیستم تا اندازه‌ای قابل مدیریت و کارا می‌باشد. با استفاده از یک دفتر کد[۱۲۱] مقدارگزینی برداری که شامل

---

[117] progressive transition
[118] skip transition
[119] ergodic
[120] vector quantization (VQ)

تعداد کمی بردارهای خصیصه با نمایانگری بالاست می‌توان داده‌های اولیه را به مجموعهٔ کـوچکی از نقاط نمایانگر کاهش داد. مقدارگزینی بـرداری هـم در سیسـتمهای وابسـته بـه مـتن و هـم در سیستمهای مستقل از متن قابل استفاده است.



شکل شمارهٔ۳– ۷ نمودار مفهومی که شکل گیری یک دفتر کد مقدارگزینی برداری را به تصویر می‌کشد

شکل ۳-۷ یک نمودار مفهومی را که مثالی از شکلگیری یک دفتر کد مقدارگزینی برداری را به تصویر می‌کشد نشان می‌دهد. یک گوینده می‌تواند بر اساس مکان مرکـز ثقـل بردارهـا از دیگـری تشخیص داده شود. در شکل ۳-۷ خصیصه‌های طیفی زمان کوتاه با یک فضای اقلیدسی دوبعـدی نشان داده شده‌اند. برای ایجاد یک مجموعه از نقاط گامهای زیر اجرا شده‌اند:

- از دو گوینده خواسته شده تا چند دنباله عبارت برای آموزش سیستم بیان کنند.
- دنباله‌های آموزش دهندهٔ سیستم تحلیل می‌شوند و برای آموزش دفتر کد مقدارگزینی برداری استفاده می‌گردند.

[121] codebook

- سپس نقاط به بخشهای جداگانه افراز می‌گردند و دو دفتر کد تولید می‌گردد که هر کدام چهار عنصر دارند. عناصر دفتر کد مقدارگزینی برداری به صورت دایره و مثلث نمایش داده می‌شوند و مرکز ثقل بخشهای مرتبط با فضای خصیصهٔ هر گوینده را نشان می دهند.

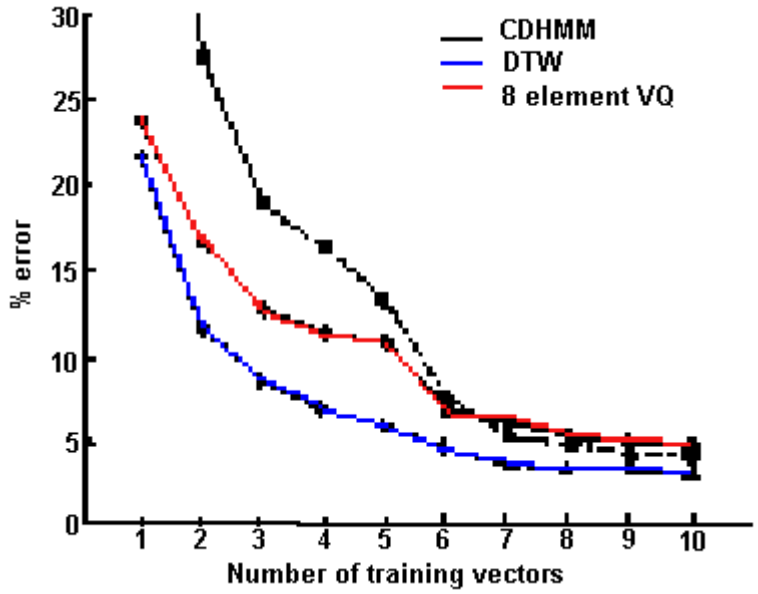
همچنان که در شکل ۳ - ۷ قابل مشاهده است با وجود کمی روی‌هم‌افتادگی دو دفتر کد هنوز کاملاً مجزا هستند و بنابراین هر گوینده می‌تواند از دیگری تشخیص داده شود. هدف آموزش یک دفتر کد مقدارگزینی برداری یافتن افرازهای مناسب از یک فضای برداری به صورت تعدادی ناحیهٔ بدون روی‌هم‌افتادگی می‌باشد. هر افراز با یک بردار مرکز ثقل مرتبط نشان داده می‌شود. روشی معمول برای یافتن یک افرازبندی مناسب استفاده از یک رویهٔ بهینه‌سازی مانند الگوریتم تعمیم‌یافتهٔ لوید[۱۲۲] که آشفتگی متوسط در بین بردارهای آموزش سیستم و مرکز ثقلها را کمینه می‌کند می‌باشد. سایر روشها عبارتند از معیار کمترین بیشینه[۱۲۳] (کمینه کردن بیشترین آشفتگی) که الگوریتم پوشش[۱۲۴] نیز نامیده می‌شود و استفاده از قانون Kامین همسایهٔ نزدیک[۱۲۵] به جای قانون نزدیک‌ترین همسایه در محاسبهٔ آشفتگی.

۵-۷ مقایسهٔ کارایی

آزمایشهای گوناگونی برای تعیین این که کدام روش برای تشخیص گوینده بهترین روش است صورت گرفته است و مهم است که به این نکته توجه شود که چگونه محققان مختلف در وضعیتهای گوناگون به نتایج متفاوتی دست پیدا نموده‌اند. به عنوان نمونه اروین[۱۲۶] در نوشتار خود در ارتباط با آزمایشهایی که وی در زمینهٔ سیستمهای وابسته به متن برای مقایسهٔ سه روش برشمرده شده انجام داده است به این نتیجه رسیده است که روش مقدارگزینی برداری بهترین

[122] Loyd
[123] minimax criterion
[124] covering algorithm
[125] K-nearest neighbour
[126] Irvine

کارایی را ارائه می‌کند. حال آن که یو[۱۲۷]، میسن[۱۲۸] و اگلبی[۱۲۹] در مقالهٔ خود اشاره به اجرای آزمایشهایی مشابه نموده‌اند که نتایج متفاوتی را احراز نموده‌اند. نتیجهٔ تجربهٔ آنان که در بردارندهٔ آزمایشهایی برای سه روش توضیح داده شده برای سیستمهای وابسته به متن و دو روش متأخر برای سیستمهای مستقل از متن است نمودار شکل ۴ -۷ برای سیستمهای مستقل از متن و شکل ۵- ۷ برای سیستمهای مستقل از متن است. همچنان که در شکل ۴ - ۷ مشاهده می‌شود بر اساس تجربیات این گروه روش چشمپوشی زمانی پویا دارای بهترین کارایی است و همچنین روشهای مدل نهان مارکف با چگالی پیوسته[۱۳۰] و مقدارگزینی برداری هشت‌عنصری استفاده شده به ازای تعداد بردارهای آموزش سیستم متفاوت کاراییهای متفاوت دارند:

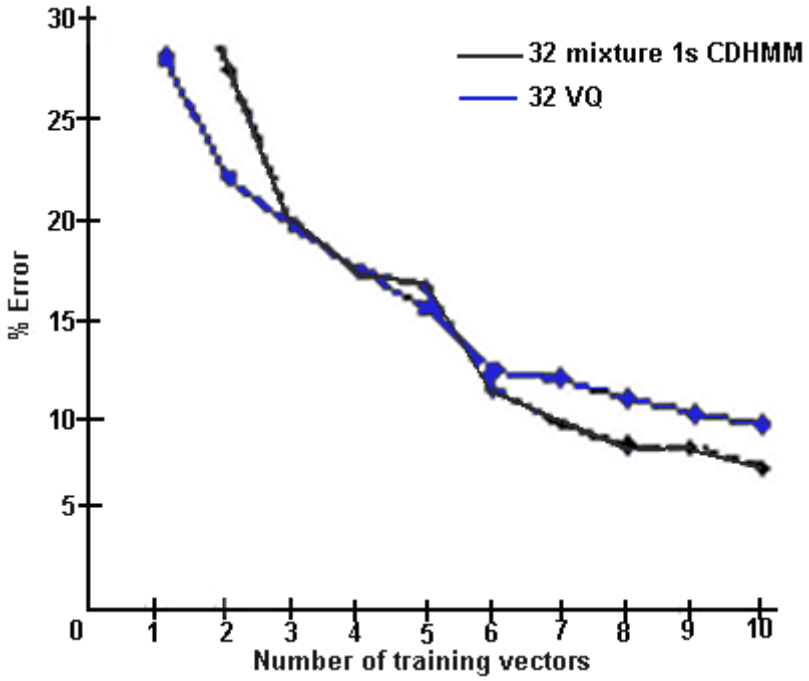


شکل شمارهٔ ۴ – ۷ درصد خطا بر اساس تعداد بردارهای آموزش سیستم برای روشهای وابسته به متن چشمپوشی زمانی پویا، مقدارگزینی برداری ۸ عنصری و مدل نهان مارکف با چگالی پیوستهٔ ۸ حالتهٔ ۱ ترکیبه

[127] Yu
[128] Mason
[129] Ogleby
[130] Continuous Density Hidden Markov Model (CDHMM)

همچنین از روی نمودار می‌توان نتیجه گرفت که با وجود آن که برای تعداد بردارهای آموزش کم روش چشمپوشی زمانی پویا عملکرد بهتری دارد با افزایش تعداد بردارها این تفاوت عملکرد دیگر به صورت واضح مشاهده نمی‌شود.

شکل شمارهٔ ۵-۷ نتیجهٔ تجربیات این گروه را برای سیستمهای مستقل از متن نشان می‌دهد: از این شکل این گونه بر می‌آید که روش مدل نهان مارکف با چگالی پیوسته نیازمند تعداد بردارهای آموزش سیستم بیشتری می‌باشد.

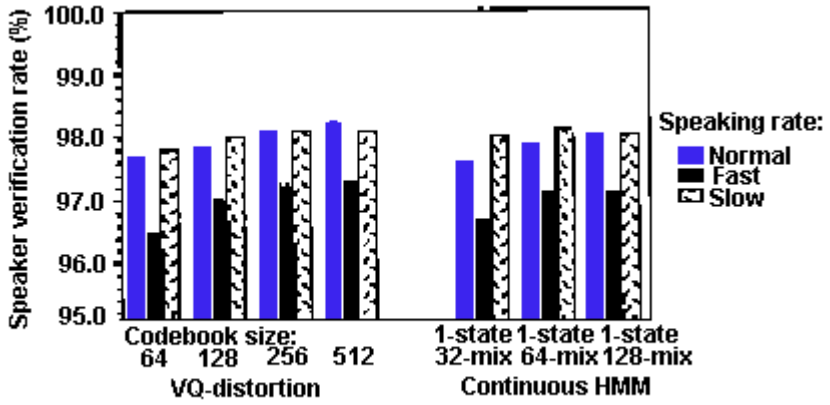


شکل شمارهٔ ۵ – ۷ درصد خطا بر اساس تعداد بردارهای آموزش سیستم برای روشهای مستقل از متن مقدارگزینی برداری ۳۲ عنصری و مدل نهان مارکف با چگالی پیوستهٔ تک حالتهٔ ۳۲ ترکیبه

ماتسوی[۱۳۱] و فروی[۱۳۲] نیز سیستمهای مستقل از متن پیاده‌سازی شده با دو روش متأخر را مقایسه نمودند و اشاره نموده‌اند که روش مدل نهان مارکف ارگودیک پیوسته در مقابل تغییرات عبارت پایداری همسانی با روش مقدارگزینی برداری دارد و عملکرد بسیار بهتری نسبت به روش

[131] Matsui
[132] Furui

مدل نهان مارکف ارگودیک گسسته دارد. آنها همچنین به نتیجه‌ای مشابه بـا گـروه قبلـی دسـت یافته‌اند و آن این است که سیستمهای مبتنی بر روش مقدارگزینی برداری برای مقـادیر کـم داده پایدارتر از سیستمهای مبتنی بر روش مدل نهان مارکف پیوسـته مـی‌باشـند. شـکل ۶-۷ نتیجـهٔ تجربیات آنان را به تصویر می‌کشد:



شکل شمارهٔ ۶ – ۷ مقایسهٔ سیستمهای مستقل از متن (ماتسوی و فوروی ۱۹۹۲)

نتیجه گیری:

با توجه به رشد و پیشرفت علم فناوری اطلاعات و رشد بازار نسبت به سیستم های تشخیص هویت (اثر انگشت، الگوی شبکیه، عنبیه و...) می توان از سیستم های تشخیش هویت توسط صدا که صدای فرد مورد نظر را ابتدا دریافت نموده و سپس پردازش کرده و برای ورود فرد به سیستم تعریف نمود.

منابع

1) Farzin Deravi, University of Kent at Canterbury, Audio-Visual Person Recognition for Security and Access Control, from http://www.jtap.uk.ic/

2) The BioAPI Consortium, BioAPI Specification Version 1.1 – March 16,2001, from http://www.bioapi.org/

3) Catherine Tilton, SAFLINK – chair of BioAPI Consortium, BioAPI-An Open Systems Interface Standard for Biometric Integration, from http://www.saflink.com/

4) Sadaoki Furui, NTT Human Interface Laboratories, Tokyo, Japan, Speaker Recognition, from clsu.cs.ogj.edu

5) Martin Cultenbruner, Audiotry User Interfaces for Desktop, Mobile and Embeded Applications

6) Richard Duncan, Mississipi State University, A Description And Comparison Of The Feature Sets Used In Speech Processing

7) Microsoft ®, MSDN Library (January 2000 Edition)

8) Thomas Holme, How to play and record sound, from http://www.codeproject.com/

9) Steven W. Smith,The Scientist and Engineer's Guide to Digital Signal Processing, Chapter 22: Audio Processing, from http://www.dspguide.com/

10) Woon Wei Kian and Yap Wei Wum, Approaches to Speaker Verification Methods (Part of an article titled as Surprise 98 … reporting on Speaker Verification), from http://www.iis.ee.ic.ac.uk/~frank/surp98/report/wwy2/approaches.htm