باسمه تعالی

دانشگاه صنعتی شریف

دانشکده مهندسی برق

فاز دوم پروژه سیگنال‌ها و سیستم‌ها

نگارنده

امیرمهدی سلیمانی‌فر

شماره دانشجویی

۹۸۱۰۱۷۴۷

استاد درس

دکتر خلج

بهار ۱۴۰۰

۱- در قسمت اول برای بدست آوردن فایل بدون نویز از توابع آماده کتابخانه scipy استفاده می‌کنیم. تابع firwin برای ساختن فیلتر و تابع lfilter برای اعمال آن به کار رفته است. با توجه به اینکه در firwin می‌توانیم چند فیلتر مختلف را بصورت همزمان اعمال کنیم برای مثال قابلیت اعمال فیلتر bandpass را با هم داریم (این کار بوسیله آرایه‌ای از فرکانس‌های قطع انجام می‌شود) به همین دلیل می‌توانیم با آزمون و خطا فرکانس‌های قطع مورد نظر خود را پیدا کرده و آن‌ها را روی سیگنال اعمال کنیم. برای اطلاعات بیشتر در مورد نحوه به کار گرفته شدن این کتابخانه نیز می‌توانید از لینک زیر استفاده کنید:

https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.firwin.html

البته مقداری نویز سفید نیز در فایل‌های صوتی وجود داشت که با توجه به آنکه با آزمون و خطای بسیار مشخص شد که استفاده از این فیلترها به تنهایی امکان برطرف‌کردن آن‌ها را به ما نمی‌دهد چرا که نیازمند یک فرآیند غیرخطی هستیم برای رفع آن از برنامه‌های مخصوص ویرایش صدا استفاده کرده‌ام، اما نویز گیری مربوط به فرکانس‌های مختلف توسط خود برنامه انجام شده است و نتایج هر دو برای مشاهده موجود است (هر دو پوشته musics و test بررسی شوند). همچنین برای نتیجه مربوط به رمزگشایی فایل صوتی دوم هم ابتدا آرایه صوتی دریافت شده برعکس شده و سپس فیلترهای bandpass مختلفی بر روی آن اعمال شده است که نویزهای تک‌فرکانس در نواحی مختلف حذف شده و فایل نهایی باقی بماند.

هر دو این قسمت‌ها با آزمون و خطای بیشتر به نتایج بهتری منتهی خواهند شد که البته این کار بدلیل آشنایی ابتدایی با فیلترها است وگرنه با استفاده از فیلترهای پیشرفته‌تر می‌توان این کار را بسیار راحت‌تر انجام داد (نمونه‌های مختلف آن‌ها در وب موجود است که بدلیل عدم ارتباط به هدف در اینجا استفاده نشده‌اند). رمز فایل دوم نیز «درس سیگنال ۳ واحد است» بود که البته کلمات ابتدایی و انتهایی بدلیل کیفیت پایین فایل صوتی (و نه لزوما وجود نویز) واضح نبودند.

۲- در این قسمت ابتدا یک فایل صوتی نمونه را دریافت و با استفاده از دستوری که در اینجا کامنت شده است آن را mono-tone کرده‌ایم و خروجی آن در کانال‌های چپ و راست را بصورت برابر در آورده‌ایم. پس از آن ورودی n از کاربر دریافت می‌شود و اگر n بزرگتر از پنجاه بود به این معنی که خروجی بیشتر از گوش راست شنیده می‌شود صدای گوش چپ متناسب با آن تضعیف شده و اگر کمتر از پنجاه بود صدای گوش راست تضعیف خواهد شد. با توجه به mono-tone بودن صدا و شنیده شدن آن از هر دو گوش در ابتدا تقویتی انجام نمی‌دهیم تا کیفیت دچار تضعیف نشود. همچنین در حالت‌های صفر و صد قطعه کد جداگانه‌ای برای پرهیز از محاسبه بیهوده نوشته شده است. دلیل شنیده شدن صدا در طرف دیگر هندزفری نیز در این حالات ساختار داخلی آن است و اگر به آرایه خروجی صوتی دقت شود تمام خروجی‌های یک طرف در این حالت برابر با صفر هستند.

برای قسمت دوم همانگونه که گفته شده بود از توابع مثلثاتی استفاده شده است و برای آنکه مقادیر دچار تغییر نشود ضرایب همگی مثبت خواهند بود که این امر بوسیله تابع abs انجام شده است. در حالت مکالمه نیز با توجه به اینکه تابع سینوسی از صفر تا پی مثبت و از پی تا 2π نیز منفی است از سقف و کف مقدار آن برای مشخص کردن نحوه شنیده شدن صحبت از دو کانال چپ و راست استفاده کرده‌ایم. در این‌حالت با توجه به تناوب مقدار از منفی یک تا مثبت یک و اینکه مقدار قدرمطلق سقف و کف همواره یکی از دو مقدار صفر یا یک را دارند که این دو با هم اتفاق نمی‌افتند می‌توانیم یک حالت مکالمه طور را ایجاد کنیم. البته فایل استفاده شده در اینجا قسمتی از یک آهنگ بوده و ممکن است این اتفاق را آنطور که مدنظر است انجام ندهد، اما بعنوان یک نمونه پیاده‌سازی اتفاق رخ داده که صدا در هر بازه زمانی در یک کانال شنیده می‌شود مشهود است. فایل نهایی به نام 3d_moving_2.wav موجود است.

۳- در اینجا یک کپی از آرایه صوت‌های اصلی گرفته شده است و پس از آن با توجه به مقادیر n و t دریافتی خروجی‌های متوالی به آن اضافه شده است که البته بصورت نمایی تضعیف شده‌اند. تضعیف بصورت درصدی باعث ایجاد نویز فراوانی می‌شد و به همین دلیل از آن پرهیز کردیم. با توجه به کم بودن اثر نمایی در تعداد اکوهای پایین از یک تقویت نیز استفاده کرده‌ام که در کد با comment مشخص شده است. تضعیف اکو نیز به این صورت است که بعد از تعداد دفعات خواسته شده دامنه به دو درصد دامنه اصلی می‌رسد که با توجه به وجود موسیقی در نمونه استفاده شده عملا قابل تشخیص نیست. برای اعمال echo در فواصل مناسب نیز از یک حلقه استفاده شده است که شرط بیرون نرفتن از محدود آرایه را نیز بررسی می‌کند.

۴- این قسمت نیز دقیقا همانگونه که خواسته شده بود با سه بار اعمال یک فیلتر all-pass بر روی ورودی ساخته شده است که در هر بار فاز آن مقدار دچار تغییر می‌شود که باعث ایجاد صدای ربات گونه خواهد شد. وابستگی مورد استفاده در این قسمت wavio بوده است که از pip قابل دریافت است و برای نشان دادن خروجی حاصل از اعمال phase change به آن نیاز داشتیم چرا که قادر به ذخیره فایل بصورت مناسب با استفاده از کتابخانه scipy نشدم.