دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

پایان‌نامه کارشناسی

گرایش فناوری اطلاعات

عنوان

طراحی و پیاده سازی سیستم تشخیص چهره در تصاویر با رزولوشن کم جهت نظارت بر عبور و مرور افراد

نگارش

محمد حسین دانش

استاد راهنما

جناب آقای دکتر محمد رحمتی

فروردین ۱۳۹۶

به نام خدا

تعهدنامه اصالت اثر

تاریخ: ۱۳۹۶/۱/۲۱

دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)

اینجانب محمد حسین دانش متعهد می‌شوم که مطالب مندرج در این پایان نامه حاصل کار پژوهشی اینجانب تحت نظارت و راهنمایی اساتید دانشگاه صنعتی امیرکبیر بوده و به دستاوردهای دیگران که در این پژوهش از آنها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مآخذ ذکر گردیده است. این پایان نامه قبلاً برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نگردیده است.

در صورت اثبات تخلف در هر زمان، مدرک تحصیلی صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پیگیری قانونی خواهد داشت.

کلیه نتایج و حقوق حاصل از این پایان نامه متعلق به دانشگاه صنعتی امیرکبیر می‌باشد. هرگونه استفاده از نتایج علمی و عملی، واگذاری اطلاعات به دیگران یا چاپ و تکثیر، نسخه‌برداری، ترجمه و اقتباس از این پایان نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مآخذ بلامانع است.

نام و نام خانوادگی دانشجو: محمد حسین دانش

امضا

نخستین سپاس و ستایش از آن خداوندی که بنده کوچکش را در دریای بیکران اندیشه، قطره‌ای ساخت تا وسعت آن را از دریچه‌های ناب آموزگارانی بزرگ به تماشا بنشیند.

سپاس از استاد گران‌قدرم، جناب آقای دکتر رحمتی که مرا یاری کردند.

تقدیم به

پدر و مادر عزیز و مهربانم،

که در سختی‌ها و دشواری‌های زندگی همواره یاوری دلسوز و فداکار و پشتیبانی محکم و مطمئن برایم بوده‌اند.

چکیده

هدف این پروژه، طراحی و پیاده سازی نرم افزاری برای انجام عملیات تشخیص چهره در تصاویر کم رزولوشن می‌باشد. در این پروژه با استفاده از ابزارهای پردازش تصویر، کتابخانه‌ها و توابع پیاده‌شده پیشین سعی بر آن است تا تصویری به عنوان ورودی دریافت شود، و با استفاده از الگوریتم‌های مورد نظر رزولوشن تصویر افرایش یافته و برای انجام عمل تشخیص چهره مورد پردازش قرار بگیرد.

پردازش‌هایی که بر روی یک تصویر پیاده می‌شوند به شرح زیرند:

- دریافت تصویر
- بهبود و افزایش رزولوشن تصویر
- یافتن چهره در تصویر
- انجام عمل تشخیص چهره
- ذخیره و نمایش نتیجه

در این پروژه، کتابخانه‌ی OpenCV برای پردازش تصاویر به کاررفته است و زبان برنامه‌ی نوشته شده پایتون می‌باشد. باید توجه داشت نسخه‌های مختلف زبان پایتون، ویژگی‌ها و توابع متفاوتی دارند. همچنین نسخه‌های مختلف کتابخانه مورد استفاده، توابع و ورودی و خروجی‌های متفاوتی دارند. پس از مطالعه‌ی چندین نسخه از آنها، نسخه‌ی ۲,۷ زبان پایتون و نسخه ۳,۲ کتابخانه OpenCV برای این پروژه برگزیده شد. با اجرای برنامه، کاربر تصویر مورد نظر خود را به برنامه می‌دهد. پس از آن تصویر خوانده شده به صورت MAT ذخیره سازی شده و پردازش‌های لازم روی آن شروع می‌شود.

واژه‌های کلیدی:

پردازش تصویر، رزولوشن کم، تشخیص چهره، سوپر رزولوشن

فهرست عناوین

صفحه

فهرست اشکال

صفحه

فهرست جداول

صفحه

۱ فصل اول

مقدمه

با پیشرفت روزافزون هوش مصنوعی و شاخه‌های مختلف آن، کاربردهـای آن بـیش از پـیش در زندگی انسان‌ها خود را نشان می‌دهد. به عنوان یکی از موفق‌ترین کاربردهـای تجزیـه و تحلیـل تصـویر، اخیرا تشخیص چهره توجه بسیاری را به خود جلب کرده است. حداقل دو دلیل برای این گرایش می‌توان پیدا کرد: اولا طیف گسترده کاربردهای تجاری و قانونی، دوما دسترسی به تکنولوژی‌های قابل قبول بعـد از حدود ۳۰ سال پژوهش. با این‌که سیستم‌های تشخیص ماشینی به سطح مشخصی از بلوغ رسیده انـد، موفقیتشان به شرایط اعمال شده توسط بسیاری از برنامه‌های واقعی محدود شده است. به عنـوان مثـال، تشخیص چهره در تصاویری که در فضای آزاد[۱] و با تغییرات در نورپردازی[۲] و حالت چهره[۳] گرفتـه شـده اند، به عنوان یک مساله باز[۴] مطرح هستند. به عبارت دیگر، سیستم‌های کنـونی بـا توانـایی درک انسـان بسیار فاصله دارند.[۱]

از طرفی، روز به روز دوربین‌های مداربسته و کنترلی[۵] در سطح شهر بیشتر به کار گرفته می‌شوند، و همچنین گوشی‌های هوشمند دارای دوربین نیز مقبولیت بیشتری نزد مردم پیدا می‌کنند. با توجه به کاربردهای بسیاری که سیستم‌های تشخیص چهره دارند، می‌توان از آن‌ها جهت کنترل عبور و مرور افراد و یا شناسایی افراد استفاده کرد، ولی مشکلی که وجود دارد این است که وضوح این دوربین‌ها برای انجام عملیات تشخیص چهره مطلوب نیستند و بالابردن وضوح آن‌ها هزینه‌های بسیاری در پی خواهد داشت.[۲]

---

۱ Outdoor Environment

۲ Illumination

۳ Pose

۴ Open Problem

۵ Surveillance Cameras

یکی از اصلی ترین مساله‌هایی که دقت سیستم‌های تشخیص چهره را تحت تاثیر قرار می‌دهد، وضوح مختلف تصاویر می‌باشد. به این صورت که هرچه وضوح تصاویر مورد نظر برای انجام عمل تشخیص چهره پایین‌تر باشد، دقت سیستم با شدت بیشتری کاهش می‌یابد. برای مقابله با این مشکل، الگوریتم‌ها و روش‌های بسیاری پیشنهاد شده است که به وسیله آن‌ها می‌توان وضوح تصاویر را تا حد مطلوبی بهبود داد و سپس عملیات تشخیص چهره را انجام داد.

از مهم ترین چالش‌های بهبود وضوح تصاویر[۶]، کم کردن بار پردازش و تولید تصویر با وضوح مناسب[۷] جهت تشخیص چهره است. هرچه هزینه افزایش رزولوشن تصویر بیشتر باشد، کاربرد الگوریتم محدودتر خواهد بود، زیرا در سیستم‌های کمتری می‌توان از آن‌ها استفاده کرد. همچنین تصاویر تولید شده باید از سطح کیفی مناسبی برخوردار باشند تا عملیات تشخیص چهره نتایج مطلوب تری داشته باشد.

از کاربردهای نرم افزار تشخیص چهره در تصاویر با وضوح پایین[۸] می‌توان به موارد مختلفی اشاره کرد که تعدادی از آن‌ها در ادامه آمده است.[۳]

- شناسایی افراد توسط دوربین‌های مداربسته
- برچسب زدن افراد در تصاویر گرفته شده توسط گوشی‌های هوشمند
- شناسایی پلاک خودروها توسط دوربین‌های مداربسته

در این پروژه، هدف شناسایی افراد در تصاویری است که توسط دوربین‌های مداربسته گرفته شده اند.

سیستم تشخیص چهره‌ای که پیاده‌سازی شده، شامل مراحلی می‌باشد که در ادامه بیان می‌گردد. در مرحله اول، ورودی به صورت ویدئو از دوربین[۹] یا به صورت یک فایل تصویر دریافت می‌شود. اگر ورودی نرم افزار یک ویدئو باشد، این ویدئو توسط ابزارهای مناسب، به فریم‌ها تبدیل شده و برای پردازش آماده می‌شود. اگر ورودی به صورت تصویر باشد، به مرحله دوم که پردازش‌هایی را بر روی

---

۶ Super-Resolution

۷ High Resolution

۸ Low Resolution Face Recognition Software

۹ Webcam

تصاویر اعمال می‌کند فرستاده می‌شود. در مرحله‌ی دوم، پردازش‌هایی برای انجام عملیات بهبود کیفیت و وضوح تصویر انجام می‌شود. در مرحله سوم، کاربر تصویر مورد نظر برای انجام عملیات تشخیص چهره را به عنوان ورودی به برنامه می‌دهد، و برنامه با استفاده از الگوریتم‌ها و ابزارهای موجود در کتابخانه OpenCV، عملیات تشخیص چهره را انجام داده و نتیجه را به عنوان خروجی نمایش می‌دهد. در مرحله سوم کاربر می‌تواند الگوریتم‌ها و روش‌های مورد نظر خود را جهت استخراج ویژگی‌ها و شناسایی چهره[۱۰] انتخاب کند.

پس می‌توان گفت ورودی ، خروجی و پردازش این سیستم به صورت زیر خواهد بود:

- ورودی: ویدئو دریافتی از محیط با
- پردازش: تبدیل ویدئو به فریمها و پیاده سازی الگوریتم‌های انتخابی بر روی آن‌ها جهت بهبود وضوح تصویر و سپس انجام عملیات تشخیص چهره[۱۱]
- خروجی: تصویر پردازش شده

۱.۱ ساختار پایان نامه

در ادامه‌ی این پایان‌نامه و در فصل دوم، به بررسی ابزارها و کاربردهای مشابه سیستم تشخیص چهره و جایگاهشان در زندگی روزمره و یا سیستم‌های پیچیده می‌پردازیم. فصل سوم، به نحوه‌ی پیاده-سازی، ساختار برنامه و کد و توابع استفاده شده می‌پردازد. در فصل چهارم، نمونه‌ای از کار با نرم افزار نمایش داده شده و قابلیت‌های مختلف آن بررسی و ارزیابی می‌شود. در نهایت در فصل پنجم نتیجه-گیری و جمع بندی از فعالیت‌های صورت گرفته ارائه‌شده و پیشنهاداتی برای کارها و محصولات آینده مطرح خواهد شد.

[۱۰] Face Detection

[۱۱] Face Recognition

۲ فصل دوم
پیشینه

همانطور که پیش از این بیان شد، امروزه تکنولوژی تشخیص چهره در ابزارهای مختلف بسیار مورد توجه اند. در این فصل، پس از تعریف تشخیص چهره و تصاویر با وضوح پایین، به بررسی تلاش‌های پیشین در زمینه‌ی تولید تکنولوژی‌های مبتنی بر تشخیص چهره و تصاویر با وضوح پایین می‌پردازیم.

۱.۲ شناسایی چهره

۱.۱.۲ تعریف

تکنولوژی تشخیص چهره، یکی از انواع سیستم زیست‌سنجی[۱] محسوب می‌شود و از مهمترین تکنولوژی‌های تشخیص و شناسایی افراد است که در کنترل دسترسی[۲] نیز مورد استفاده قرار می‌گیرد و پس از موفقیت سیستم شناسایی از طریق اثر انگشت در چند سال اخیر جزء مهمترین تکنولوژی‌های تشخیص زیست‌سنجی به شمار می‌آید.

همانطور که مشاهده شد، پردازش تصویر کاربردهای بسیاری دارد و تشخیص چهره یکی از آن‌هاست. اما در این پروژه، مقصود و منظور ما از تشخیص چهره، به طور خاص، ویدئو ایست که از با استفاده از دوربین‌های معمولی ضبط شده است و به دلیل کیفیت بعضا پایینی که دارند، چالش‌های

[۱] Biometrics

[۲] Access Control

مخصوصی برای انجام عملیات تشخیص چهره دارند. بطورکلی یک سیستم زیست‌سنجی تشخیص چهره، از چهار بخش تشکیل یافته است که در ادامه شرح داده می‌شوند.

شکل ۱ نمونه‌ای از یک برنامه تشخیص چهره

۲.۱.۲ بخش‌های مختلف سیستم تشخیص چهره

۱.۲.۱.۲ بخش حسگر

در حالی که تشخیص چهره دو بعدی با دوربین معمولی امکان پذیر می‌باشد در روش سه بعدی نیاز به یک سنسور پیچیده و سطح بالایی از لحاظ فنی می‌باشد چرا که بایستی قابلیت کسب اطلاعات عمیق تر را داشته باشد. بخش سنسور وظیفه گرفتن تصویر اشخاص را بر عهده دارد و بسته به نیاز و کاربرد دستگاه گیرنده می‌تواند یک دوربین سیاه و سفید و یا رنگی و یا یک بخش مخصوص با قابلیت استخراج اطلاعات عمیق تر و یا یک دوربین مادون قرمز با تصاویر مادون قرمز باشد.

۲.۲.۱.۲ بخش مخصوص تشخیص و استخراج اطلاعات

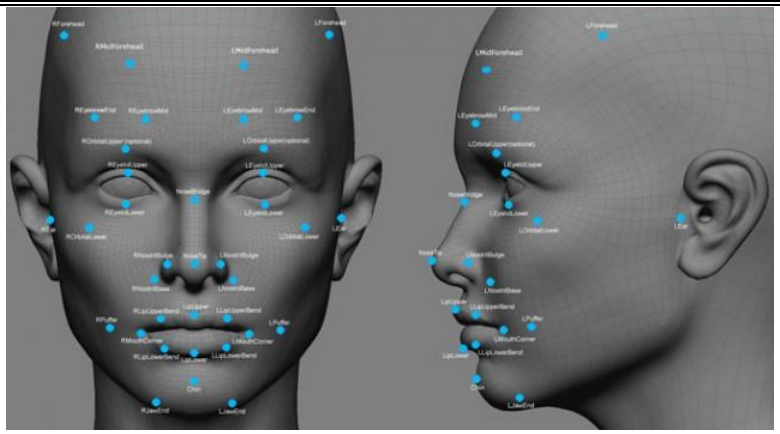
تصاویر بدست آمده توسط این بخش در ابتدا ارزیابی محتوایی شده و داده‌های نامربوط ازقبیل پس زمینه، موها، گردن و شانه حذف و تنها محتوای ناحیه چهره را شناسایی می‌کند. سپس تصویر بدست آمده تحت فرایندهای محاسباتی و عملیاتی پیچیده برای استخراج اطلاعات مربوط به ویژگی‌های سطحی چهره و تجزیه اطلاعات کلی تصویر قرار می‌گیرد. در حقیقت در این مرحله تصویر خروجی که بایستی توسط بخش طبقه بندی کننده برای تعیین هویت و تشخیص چهره مورد استفاده قرار گیرد در این مرحله با استفاده از روش‌های پیچیده PCA, LDA و غیره آماده می‌گردد.

۳.۲.۱.۲ بخش طبقه بندی

در این بخش قالب تصویر استخراج شده از مرحله قبلی با قالب‌های موجود در گالری تصاویر مقایسه می‌گردد و در نتیجه معلوم می‌شود که آیا چهره گرفته شده جزء قالب‌های موجود می‌باشد و قابل شناسایی است یا خیر و در صورت مثبت بودن جواب بخش تصمیم گیری هویت شخص را که بر اساس نتیجه مقایسه بخش طبقه بندی بوده است را تایید می‌کند. بر اساس امتیاز بدست آمده از مقایسه که همان درصد تطابق قالب گرفته شده با قالب‌های موجود می‌باشد کاربر مورد نظر مورد تایید قرار گرفته و یا پذیرفته نمی‌شود.

۴.۲.۱.۲ بخش پایگاه داده‌ها

این بخش برای ثبت نام، نگهداری واکشی قالب چهره کاربران را بر عهده دارد. در طول ثبت نام بخش سنسور تصاویر را ثبت کرده و مجموعه این تصاویر همان گالری تصاویر را ایجاد می‌کند که در مرحله طبقه بندی مورد استفاده قرار می‌گیرد. در بیشتر روش‌های تشخیص چهره چندین نمای متفاوت از یک شخص در حالت‌های مختلف مثل خنده، اخم، عصبانیت، عادی و یا با عینک از کاربر گرفته می‌شود و این تعدد در بالابردن ضریب شناسایی اهمیت ویژه‌ایی دارد.



شکل ۲ نقاط طلایی پردازش تشخیص چهره

۳.۱.۲ کارهای انجام شده

۱.۳.۱.۲ روش سنتی

تعدادی از الگوریتم‌های اولیه تشخیص چهره، ویژگی‌های چهره را با استفاده از استخراج نشانه‌ها[۱]، و یا ویژگی‌ها[۲] از تصویر یک چهره شناسایی می‌کنند. برای مثال، یک الگوریتم می‌تواند موقعیت مکانی، سایز، و یا حالت چشم‌ها، بینی، گونه‌ها و فک را آنالیز کند. این ویژگی‌ها در نهایت برای پیدا کردن تصاویر با ویژگی‌های مشابه استفاده می‌شوند. تعدادی از الگوریتم‌ها یک گالری از تصاویر چهره را نرمال سازی می‌کنند و سپس آن‌ها را فشرده می‌کنند، و تنها اطلاعاتی که برای پردازش‌های تشخیص چهره قابل استفاده هستند را نگه می‌دارند. سپس یک تصویر کاوشگر[۳] با اطلاعات چهره ذخیره شده مقایسه می‌شود. یکی از موفق ترین سیستم‌های اولیه، بر پایه ی تکنیک‌های تطبیق الگوها می‌باشد که بر روی یک دسته از ویژگی‌های چشمگیر تصاویر چهره اعمال می‌شود.[۴]

[۱] Extracting Landmarks

[۲] Features

[۳] Probe Image

الگوریتم‌های تشخیص چهره به دو دسته اصلی تقسیم می‌شوند، هندسی، که بر روی ویژگی‌های متمایز کننده توجه می‌کند، و فتومتریک، یک روش آماری که تصویر را به مقادیر مختلف تقسیم می‌کند و این مقادیر را با الگو مقایسه می‌کند و تفاوت‌ها را از بین می‌برد.

الگوریتم‌های شناخته شده‌ای در این دسته هستند؛ از جمله الگوریتم PCA[۱] که از eigenfaces استفاده می‌کند، الگوریتم LDA[۲] که از fisherface استفاده می‌کند، و الگوریتم مدل پنهان مارکوف[۳] که از ارائه تنسور[۴] استفاده می‌کند.

۲.۳.۱.۲ تشخیص ۳بعدی

تشخیص چهره ۳بعدی، روند تازه‌ای است که ادعا می‌کند دقت بیشتری را به دست آورده است. این روش از حسگرهای ۳بعدی برای به دست آوردن اطلاعات درباره حالت یک چهره استفاده می‌کند. سپس این اطلاعات برای شناسایی ویژگی‌های متمایز کننده در سطح یک چهره، مثل کانتور کاسه چشم، بینی، و چانه استفاده می‌شود.

یکی از مزایای تشخیص چهره ۳بعدی این است که با تغییرات نور، بر خلاف روش‌های پیشین، نتایجش دچار تغییر نمی‌شود. همچنین می‌تواند یک چهره را از زوایای مختلف شناسایی کند. نقاط داده ۳بعدی یک چهره دقت تشخیص چهره را به صورت چشمگیری افزایش می‌دهد. با توسعه حسگرهای پیچیده تحقیقات ۳بعدی پیشرفت کرده است. این حسگرها با طرح ریزی نور ساختاریافته بر روی چهره کار می‌کنند. تعداد بسیار زیادی از این حسگرها را می‌توان در تراشه‌های CMOS قرار داد.[۵]

حتی تکنیک‌های بی نقص تطبیق ۳بعدی نیز نسبت به حالات چهره حساس هستند. برای این بررسی این مساله، گروهی با استفاده از هندسه متریک، راه حل قطعی‌ای برای تطبیق ۳بعدی تصاویر چهره ارائه داد.[۶]

---

[۱] Principal Component Analysis

[۲] Linear Discriminant Analysis

[۳] Hidden Markov Model

[۴] Tensor Representation

۳.۳.۱.۲ آنالیز بافت پوست[۱]

یک روند رو به رشد دیگر، از جزئیات بصری پوست، همانطور که در یک تصویر دیجیتال استاندارد و یا یک تصویر اسکن شده وجود دارد، استفاده می‌کند. این تکنیک، که آنالیز بافت پوست نامیده شده است، خطوط منحصر به فرد، الگوها، و لکه‌های آشکار در پوست یک فرد را به یک فضای ریاضی تبدیل می‌کند. آزمایش‌ها نشان داده اند که به همراه آنالیز بافت پوست، کارایی تشخیص چهره ۲۰ تا ۲۵ درصد افزایش می‌یابد.[۵]

۴.۳.۱.۲ دوربین‌های حرارتی[۲]

دوربین‌های حرارتی، شکل دیگری از دریافت داده ورودی برای تشخیص چهره است. با این روش، دوربین تنها شکل سر را تشخیص داده و وسایلی مثل عینک، کلاه، و یا آرایش چهره را نادیده می‌گیرد. مشکلی که در استفاده از تصاویر حرارتی برای تشخیص چهره وجود دارد این است که پایگاه داده‌ها برای تشخیص چهره محدود هستند.

۲.۲ فراتفکیک پذیری[۳]

۱.۲.۲ تعریف

[۱] Skin Texture Analysis

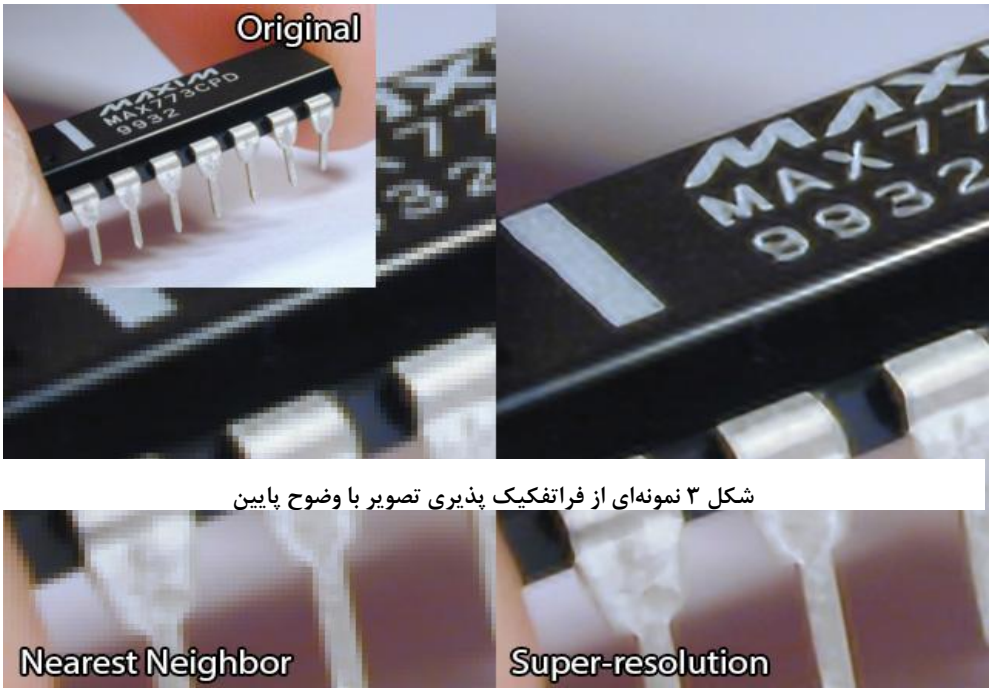
[۲] Thermal Cameras

[۳] Super Resolution

فراتفکیک پذیری دسته‌ای از تکنیک‌هاست که به کمک آن‌ها می‌توان وضوح یک تصویر را افزایش و بهبود داد. در بعضی از تکنیک‌های فراتفکیک پذیری، که به عنوان فراتفکیک پذیری نوری[۱] مطرح می‌شوند، سیستم‌های انکسار محدود[۲] برتری یافتند، و در سایر تکنیک‌ها، که به عنوان فراتفکیک پذیری هندسی[۳] مطرح می‌شوند، وضوح حسگرهای تصویربرداری برتری یافتند.

۲.۲.۲ تکنیک‌های موجود

۱.۲.۲.۲ فراتفکیک پذیری نوری



شکل ۳ نمونه‌ای از فراتفکیک پذیری تصویر با وضوح پایین

این تکنیک به چند زیرشاخه اصلی تقسیم می‌شود که در ادامه هر کدام توضیح داده می‌شود.

[۱] Optical SR

[۲] Diffraction Limit Systems

[۳] Geometrical SR

جایگزینی گروه‌های فرکانس-مکانی[۱]: اگرچه پهنای باند مجاز در انکسار[۲] مقدار مشخصی می‌باشد، با این حال می‌تواند در هرجایی از طیف فرکانس-مکانی جای گیرد.

تسهیم گروه‌های فرکانس-مکانی[۳]: یک تصویر با استفاده از فیلتر میان گذر عادی دستگاه نوری تشکیل شده است. سپس برخی از ساختارهای شناخته شده نور، به عنوان مثال مجموعه‌ای از حاشیه نور که داخل فیلتر میان گذر است، بر روی هدف اضافه می‌شوند. حال تصویر شامل اجزایی می‌باشد که نتیجه ترکیب هدف و ساختار نوری اضافه شده می‌باشد، و اطلاعات مربوط به جزییات هدف را منتقل می‌کند.

کاربرد پارامترهای متعدد درون محدوده انکسار سنتی[۴]: اگر هدفی قطبش یا ویژگی‌های طول موج مشخصی نداشته باشد، دو حالت قطبش یا محدوده طول موج غیرهم‌تداخل می‌توانند برای رمزگذاری اطلاعات و جزییات هدف استفاده شوند. هر دو از انتقال فیلتر میان‌گذر طبیعی استفاده می‌کنند اما در نهایت به صورت جداگانه رمزگشایی می‌شوند تا ساختار هدف را با وضوح بیشتری بازسازی کنند.

کاوش اختلال الکترومغناطیس نزدیک-میدان[۵]: بحث معمول پیرامون فراتفکیک پذیری شامل تصویر برداری معمولی از یک شی به وسیله یک سیستم نوری می‌باشد. اما تکنولوژی مدرن اجازه کاوش اختلال‌های الکترومغناطیس درون فواصل مولکولی منبع را که دارای ویژگی‌های وضوح فوق العاده می‌باشد را می‌دهد.

۲.۲.۲.۲ فراتفکیک پذیری هندسی

این تکنیک به چند زیرشاخه اصلی تقسیم می‌شود که در ادامه هر کدام توضیح داده می‌شود.

---

[۱] Substituting spatial-frequency bands

[۲] diffraction

[۳] Multiplexing spatial-frequency bands

[۴] Multiple parameter use within traditional diffraction limit

[۵] Probing near-field electromagnetic disturbance

کاهش نویز تصویر با ارائه متعدد[۱]: وقتی که یک تصویر با نویز همراه شود، حتی در محدوده انکسار، جزییات بیشتری در تعداد متوسطی از ارائه‌ها را شامل می‌شود.

رفع تاری یک تصویری[۲]: نقص‌های شناخته شده در یک وضعیت تصویر برداری، مثل عدم فوکس یا انحرافات، می‌توانند به صورت کلی یا جزئی در یک تصویر توسط فیلتر فرکانس-مکانی مناسب کاهش یابند. این رویه‌ها همه در درون فیلتر میان گذر وجود دارند.

محلی سازی زیر-پیکسل‌های تصویر[۳]: محل یک منبع را می‌توان با محاسبه مرکز جاذبه

شکل ۴ نمونه‌ای روش کاهش نویز تصویر با ارائه متعدد

توزیع نور که بر روی چند پیکسل مجاور گسترده شده است مشخص کرد(شکل ۳). در شرایطی که نور کافی وجود داشته باشد، این را می‌توان با دقت دلخواه، بسیار بهتر از پهنای پیکسل دستگاه تشخیص و محدودیت وضوح برای تصمیم گیری اینکه منبع واحد یا دوگانه است، بدست آورد. این روش با این پیش

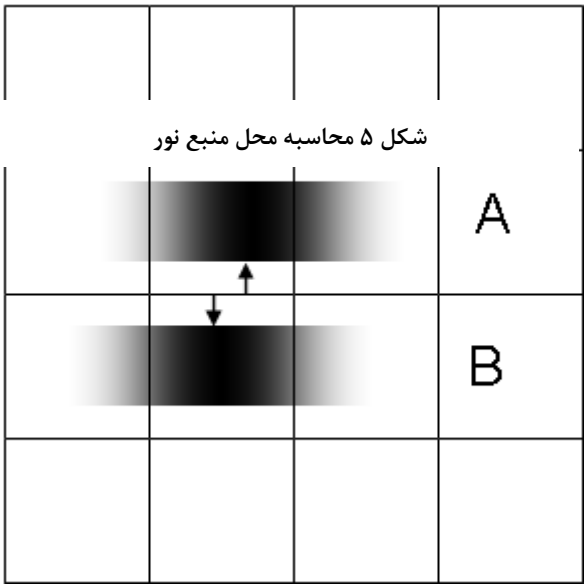
[۱] Multi-exposure image noise reduction

[۲] Single-frame deblurring

[۳] Subpixel image localization

فرض که تمام نور از یک منبع واحد ساطع می‌شود، پایه‌ی فراتفکیک پذیری میکروسکوپی را تشکیل می‌دهد.

القای بیزی فراتر از حد انکسار سنتی[1]: برخی از ویژگی‌های اشیا، هرچند فراتر از حد انکسار، ممکن است با سایر ویژگی‌های اشیا که در همان محدوده هستند در ارتباط باشند. با استفاده از روش‌های آماری، می‌توان از داده‌های در دسترس تصویر درباره کل اشیا نتیجه گیری کرد.[۷]



شکل ۵ محاسبه محل منبع نور

[1] Bayesian induction beyond traditional diffraction limit

۳ فصل سوم
پیاده سازی

در این بخش،پس از معرفی ابزارها و کتابخانه‌های مورد نیاز برنامه، به چگونگی نصب آنها و بررسی کد برنامه می‌پردازیم.

۱.۳ ابزارها

۱.۱.۳ برنامه PyCharm

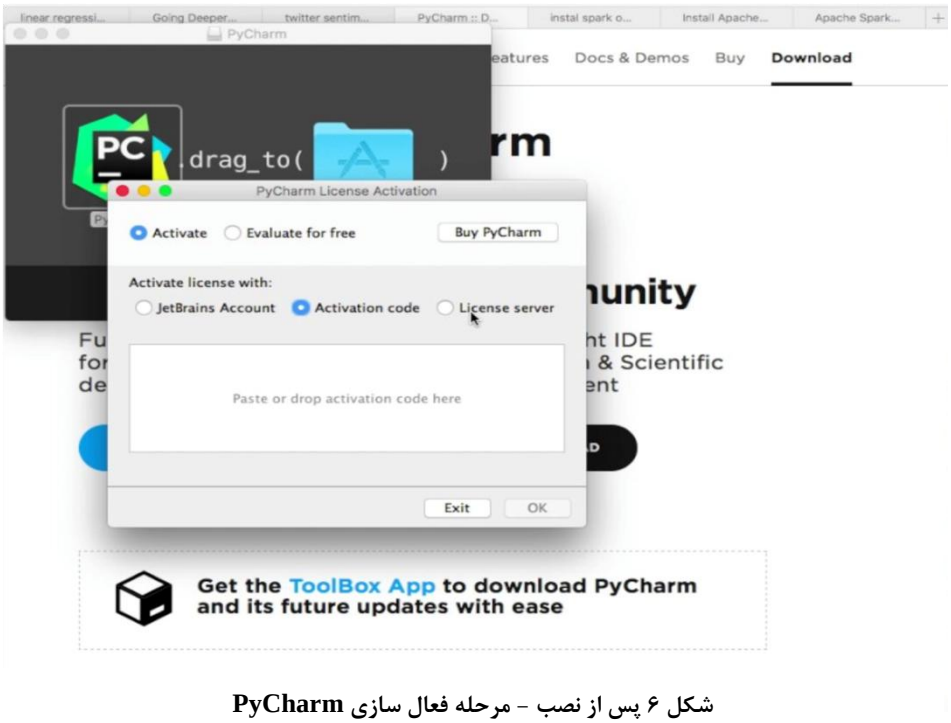
برنامه PyCharm یک محیط برنامه‌نویسی است که به طور مشخص برای زبان پایتون طراحی شده است. این برنامه توسط شرکت JetBrains طراحی شده است. از جمله قابلیت‌های این برنامه آنالیز و تحلیل کد، دیباگر گرافیکی و توسعه وب به وسیله Django می‌باشند. برنامه PyCharm چندسکویی[۱] می‌باشد، یعنی می‌توان از آن در ویندوز، مکینتاش و توزیع‌های مختلف لینوکس استفاده کرد. دو نسخه از برنامه PyCharm موجود می‌باشد، نسخه رایگان که تحت لیسانس آپاچی می‌باشد، و نسخه حرفه‌ای که باید خریداری شود و قابلیت‌های بیشتری دارد.[۸]

در این پروژه از نسخه ۲۰۱۷,۱ برنامه PyCharm استفاده شده است.

[۱] Cross-Platform

۱.۱.۱.۳ نصب

برای دانلود و نصب برنامه PyCharm کافی است از این آدرس[۱] نسخه مورد نظر دانلود شود. پس از دریافت فایل نصبی، آن را در محل مناسب نصب می‌کنیم. تمامی‌این مراحل به سادگی قابل انجام هستند.



شکل ۶ پس از نصب – مرحله فعال سازی PyCharm

۲.۱.۳ کتابخانه OpenCV

OpenCV یک کتابخانه ی متن باز برای بینایی کامپیوتر است که از آدرس http://SourceForge.net/projects/OpenCVlibrary، قابل دسترسی است. این کتابخانه به زبان C

[۱] https://www.jetbrains.com/pycharm/download/

و C++ نوشته شده و تحت لینوکس، ویندوز و مکینتاش قابل اجراست. همچنین، برای واسط‌هایی چون Matlab، Ruby، Python، جاوا و غیره، توسعه‌های فعالی دارد.

هدف از طراحی OpenCV، پردازش کارا به خصوص برای کاربردهای بی‌درنگ است. OpenCV می‌تواند با پردازنده‌های چند هسته‌ای نیز کار کند. درصورتی که تمایل دارید از بهینه‌سازی خودکار بیشتری روی معماری‌های اینتل بهره ببرید، می‌توانید کتابخانه‌های IPP۲ اینتل که شامل روتین‌های بهینه شده سطح پایین در بسیاری از زمینه‌های الگوریتمی‌هستند را خریداری کید. OpenCV به صورت خودکار IPP مناسب را در زمان اجرا در صورتی که کتابخانه نصب باشد، به کار می‌گیرد.

یکی از اهداف OpenCV فراهم کردن یک زیربنای بینایی کامپیوتر با کاربرد ساده است؛ به طوری که افراد بتوانند برنامه‌های بینایی نسبتاً پیچیده خود را به سرعت بسازند. کتابخانه OpenCV شامل بیش از ۵۰۰ تابع پیرامون موضوعات مختلف بینایی، از بررسی محصول کارخانه گرفته تا تصویربرداری پزشکی، امنیت، واسط کاربر، تنظیم دوربین، رباتیک و بینایی دوچشمی(استریو) است. از آنجا که همواره قرابت زیادی بین بینایی کامپیوتر و یادگیری ماشین وجود اشته است، OpenCV شامل یک کتابخانه یادگیری ماشین همه‌منظوره (MLL۳) نیز هست. این زیر کتابخانه، روی مباحث تشخیص الگوی آماری و دسته‌بندی تمرکز دارد.

از زمان انتشار نسخه آلفا در ژانویه ۱۹۹۹، OpenCV در بسیاری از کاربردها، محصولات و تلاش‌های تحقیقاتی مورد استفاده قرار گرفته است. این کاربردها شامل چسباندن تصاویر ماهواره‌ای و نقشه‌های وب به یکدیگر، تنظیم تصویر اسکن شده، کاهش نویز تصاویر پزشکی، تحلیل شئ در سامانه‌های تشخیص اختلال و امنیت، نظارت خودکار و سامانه‌های امنیت، سامانه‌های بازرسی صنعتی، تنظیم دوربین، کاربردهای نظامی‌و وسایل نقلیه هوایی، زمینی و زیرآبی بدون سرنشین است. حتی از آن می‌توان در تشخیص موزیک و صوت نیز استفاده کرد به این روش که از تکنیک‌های تشخیص بینایی برای تصاویر طیف‌نگار صدا استفاده شود. OpenCV یک جزء کلیدی سامانه بینایی ربات دانشگاه

استنفورد، به نام استنلی بود که در مسابقات بزرگ ربات صحرایی برنده دو میلیون دلار جایزه از طرف دارپا[1] شد.

در این پروژه از نسخه ۳,۲ تحت سیستم عامل مک و زبان برنامه‌نویسی پایتون استفاده شده است.

۱.۲.۱.۳ نصب

برای نصب کتابخانه OpenCV به ترتیب باید دستورات زیر را در محیط ترمینال اجرا کنیم. در نسخه‌های جدید کتابخانه OpenCV، باید مخزن اضافه‌ای جهت انجام کارهایی نظیر شناسایی ویژگی‌ها و شناسایی متن در تصویر دانلود و نصب کنیم. به این منظور، ابتدا کتابخانه OpenCV را به همراه مخزن مورد از وبسایت گیت‌هاب دانلود می‌کنیم، سپس نسخه آن را چک می‌کنیم.

```
Mohamads-MacBook-Pro:~ Mohamad$ git clone https://github.com/Itseez/opencv_contrib
Mohamads-MacBook-Pro:~ Mohamad$ cd opencv_contrib/
Mohamads-MacBook-Pro:~ Mohamad$ git chekout 3.2.0
```

شکل ۷ مراحل دانلود کتابخانه OpenCV

سپس به فولدر کتابخانه OpenCV می‌رویم و مراحل نصب را آغاز می‌کنیم.

```
Mohamads-MacBook-Pro:~ Mohamad$ cd ~/opencv_contrib/
Mohamads-MacBook-Pro:~ Mohamad$ mkdir build
Mohamads-MacBook-Pro:~ Mohamad$ cd build
Mohamads-MacBook-Pro:~ Mohamad$ cmake
Mohamads-MacBook-Pro:~ Mohamad$ make -j8
Mohamads-MacBook-Pro:~ Mohamad$ make install
```

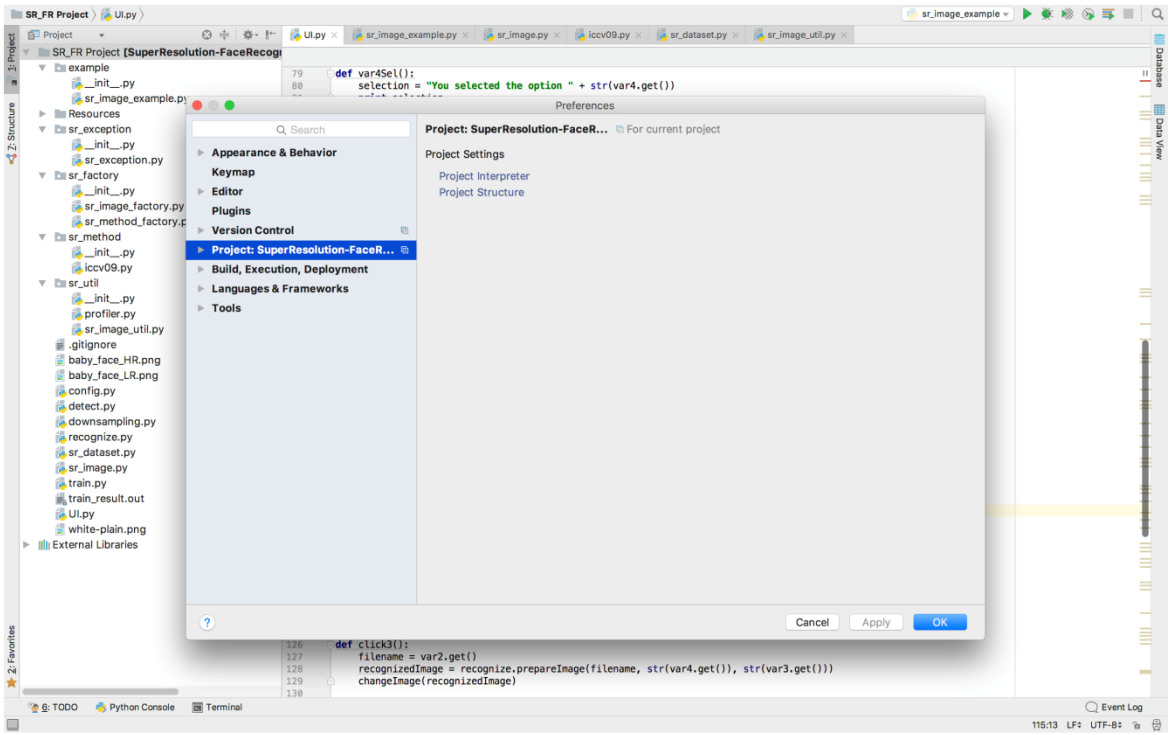
شکل ۸ مراحل نصب کتابخانه OpenCV

[1] DARPA

پس از انجام این مراحل، در صورتی که به خطایی برخورد نکنیم، کتابخانه OpenCV برای پایتون موجود تحت سیستم عامل به درستی نصب شده است. در این پروژه از نرم افزار PyCharm برای پیاده سازی استفاده شده است. در ادامه نحوه اضافه کردن OpenCV به Pycharm مورد بررسی قرار می‌گیرد.

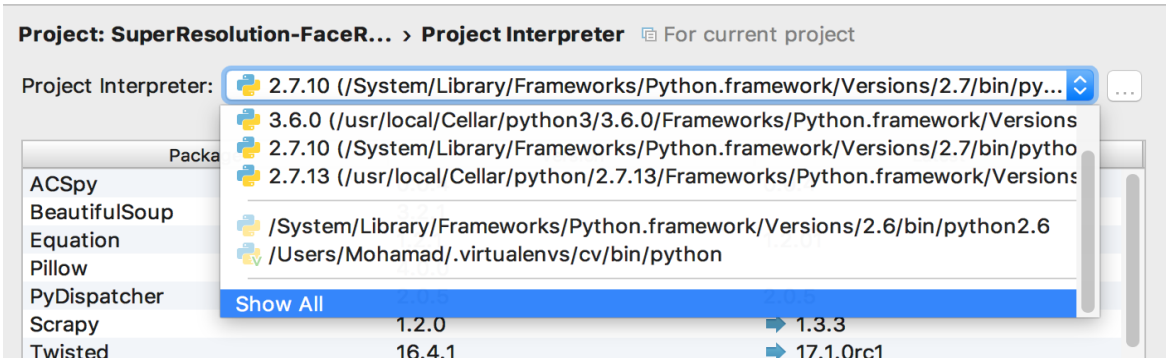
مراحل راه‌اندازی OpenCV در PyCharm به شرح زیر است:

۱. در پروژه مورد نظر، به قسمت تنظیمات می‌رویم.



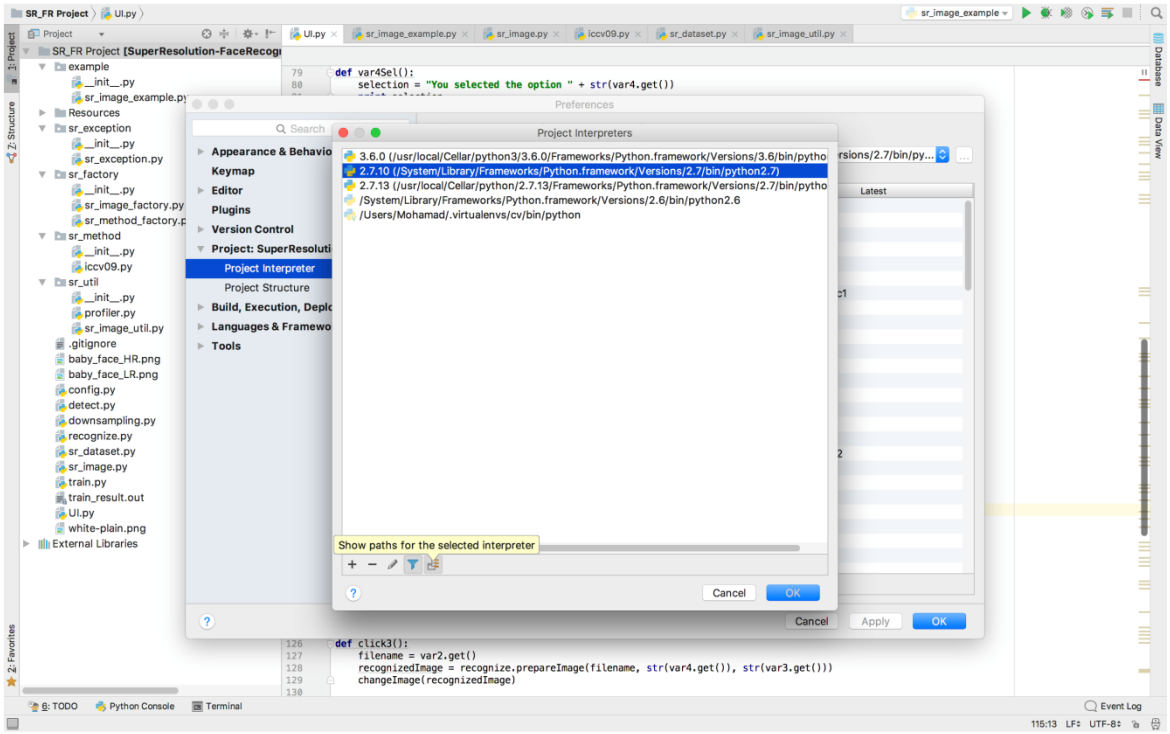
شکل ۹ قسمت تنظیمات پروژه

۲. با کلیک بر روی Project Interpreter، به پنجره جدیدی می‌رویم که در آن طبق شکل ۱۰، بر روی Show All باید کلیک کنیم.



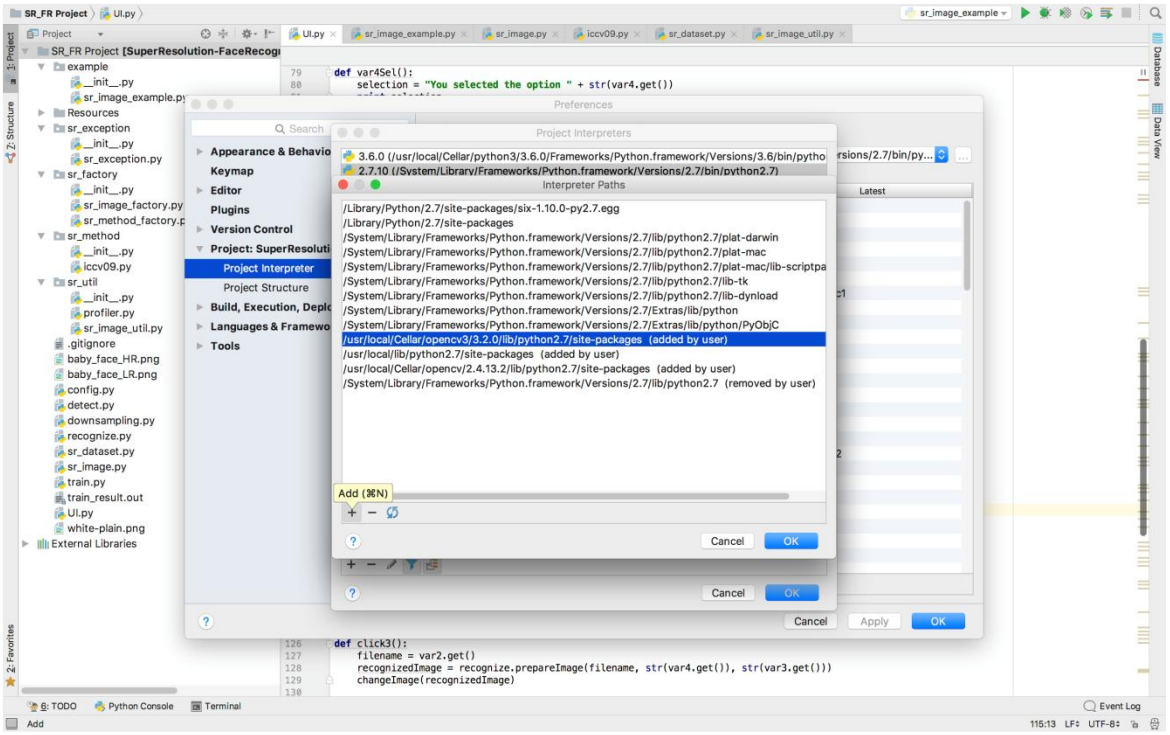
شکل ۱۰ نمایش همه‌ی نسخه‌های نصب شده پایتون در سیستم

۳. در پنجره باز شده، با انتخاب مفسر پایتون مورد نظر، بر روی Show paths for the selected interpreter کلیک می‌کنیم.



شکل ۱۱ انتخاب مفسر پایتون مورد نظر و بررسی پکیج‌های نصب شده آن

۴. حال، آدرسی که در آن کتابخانه OpenCV نصب شده است را به عنوان مسیری برای پکیج‌های نصب شده نسخه پایتون مورد نظر مشخص می‌کنیم. این آدرس با توجه به هر سیستم عامل و هر نسخه OpenCV یا پایتون می‌تواند متفاوت باشد.



شکل ۱۲ مشخص کردن مسیر کتابخانه OpenCV نصب شده

۲.۳ کد برنامه

ساختار کلی پیاده‌سازی نرم‌افزار را می‌توان به صورت زیر در نظر گرفت:

- دریافت تصویر ورودی با وضوح پایین
- انجام عملیات فراتفکیک پذیری
  - ساخت تصاویر با مقیاس پایین‌تر
  - یافتن نواحی مشابه

- یادگیری تبدیل
- ایجاد رابطه خطی

- دریافت تصویر مورد نظر جهت تشخیص چهره
- انجام عملیات تشخیص چهره
  - آموزش
  - شناسایی چهره
  - استخراج ویژگی
  - تشخیص چهره
- نمایش خروجی‌ها

۱.۲.۳ دریافت تصویر ورودی با وضوح پایین

کاربر در این برنامه می‌تواند ورودی مورد نظر خود را تعیین نماید. برنامه قابلیت دریافت ویدئو از دوربین[۱] و یا بارگذاری تصویری از پیش ذخیره شده را با توجه به آدرس فایل آن دارد. برای این منظور از تابع videoCapture() از توابع OpenCV استفاده می‌شود. این تابع، درصورت دریافت ورودی ۰، به دوربین متصل می‌شود.

```
60
61  if __name__ == '__main__':
62
63      cv2.namedWindow("camera", 1)
64      capture = cv2.VideoCapture(0)
65
66      # To improve performance, can specify a lower resolution. e.g. 320x240
67      width = 3
68      height = 4
69
```

شکل ۱۳ دریافت ورودی توسط دوربین

پس از مشخص شدن مبدا ورودی، در صورتی که کاربر جریان مورد نظرش ویدئو باشد، جریان دیگری فعال می‌شود که پردازش ویدئو را بر عهده دارد و تا زمانی که کاربر فرمان توقف نداده باشد و یا

[۱] webcam

ویدئو به پایان نرسیده باشد ادامه دارد. در این جریان تمام فعالیت‌ها در حلقه‌ای فراگیر رخ می‌دهد که به ازای هر فریم جدید از مبدا، تکرار می‌شود. هر فریم به صورت یک MAT دریافت می‌شود که به عبارتی مجموعه‌ای از اطلاعات در مورد تصویر دریافت شده از جمله نوع کانال رنگی، اندازه، پیکسل‌ها و رنگ‌هایشان و ... می‌باشد. فریم، در این برنامه MAT حاوی فریم اولیه‌ی دریافت شده است. در صورتی هم که کاربر جریان ورودی را تصویر انتخاب کند، آدرس تصویر مورد خود را به عنوان ورودی به برنامه داده و برنامه تصویر را همانطور که پیشتر مطرح شد، به صورت یک MAT دریافت می‌کند.

در صورتی که سیستم قادر به دریافت فریم جدید از مبدا نباشد، برنامه متوقف می‌شود و تا دریافت دوباره فریم صبر می‌کند. همچنین در صورتی که آدرس فایل ورودی به تصویر اشاره نکند، پیغام خطا برگردانده می‌شود.

۲.۲.۳ انجام عملیات فراتفکیک پذیری

پس از آنکه تصویر ورودی به درستی دریافت شد، با استفاده از تابع create_sr_image()، مولفه‌های y، C_b، C_r تصویر را بدست می‌آوریم. این کار را برای سادگی پردازش تصویر مورد نظر انجام می‌دهیم. مولفه y همان میزان روشنایی پیکسل‌ها و مولفه‌های C_b و C_r میزان رنگ‌های آبی و قرمز پیکسل‌ها می‌باشند.

```
19      @classmethod
20      def create_sr_image(cls, image):
21          """Create a SR image from a PIL image.
22
23          @param image: an instance of PIL image.
24          @type image:
25          @return: an instance of SRImage
26          @rtype: L{sr_image.SRImage}
27          """
28          y_image, cb_image, cr_image = sr_util.sr_image_util.decompose(image)
29          return SRImage(y_image, cb_image, cr_image)
```

شکل ۱۴ تابع create_sr_image()

خروجی این تابع، به تابع reconstruct() جهت اجرای عملیات فراتفکیک پذیری فرستاده می‌شود. عملیات فراتفکیک پذیری به صورت کلی به دو شاخه تقسیم می‌شود. شاخه اول که در آن سعی دارد با استفاده از یادگیری یک زوج از نواحی با کیفیت و بی کیفیت، تصویر با وضوح بالا را از روی تصویر با وضوح پایین ورودی بسازد، که این روش فراتفکیک پذیری برپایه نمونه[۱] نامیده می‌شود. شاخه دوم که سعی دارد با استفاده از تعدادی از تصاویر با وضوح پایین و اعمال روابط خطی، تصویر با وضوح بالا را بسازد. این شاخه به عنوان فراتفکیک پذیری کلاسیک[۲] مطرح شده است. روشی که در این پروژه برای بهبود وضوح تصاویر استفاده کرده ایم، یک چارچوب یکپارچه از این شاخه‌ها می‌باشد.[۹] این فرایند شامل تعدادی بخش می‌باشد که در ادامه این بخش‌ها توضیح داده می‌شوند.

۱.۲.۲.۳ ساخت تصاویر با مقیاس پایین‌تر[۳]

در این مرحله، تصویر ورودی را گرفته و تصاویری با مقیاس پایین‌تر از آن می‌سازیم. این کار را می‌توان با تابع pyrDown() در کتابخانه OpenCV انجام داد. این تابع یک مرحله مقیاس تصویر را پایین آورده و سپس آن را تار[۴] می‌کند. ورودی این تابع تصویر مورد نظر برای ساخت مقیاس پایین‌تر می‌باشد.[۱۰]

۲.۲.۲.۳ یافتن نواحی مشابه

نواحی موجود در تصویر با وضوح کم ورودی، در تصاویر با مقیاس پایین‌تر ساخته شده جستجو می‌شوند. در هر تصویر به همراه تصاویر مقیاس پایین‌ترش، نواحی(به عنوان مثال، مربع ۵*۵) بسیار زیاد مشابهی وجود دارد که می‌توان از آن‌ها استفاده کرد. یافتن نواحی مشابه با استفاده از الگوریتم Approximate Nearest Neighbor انجام می‌شود.[۱۱]

---

[۱] Example-based SR

[۲] Classical SR

[۳] Downscaled

[۴] Blur

۳.۲.۲.۳ یادگیری تبدیل

پس از اینکه برای یک ناحیه، ناحیه مشابهی در تصویر مقیاس پایین‌تر پیدا شد، ناحیه والد در تصویر مقیاس پایین‌تر به دست می‌آید و ناحیه والد را در موقعیت مناسب در تصویر با وضوح بالای مجهول[۱] با توجه به فاصله موجود بین ناحیه فرزند و ناحیه والد کپی می‌شود.

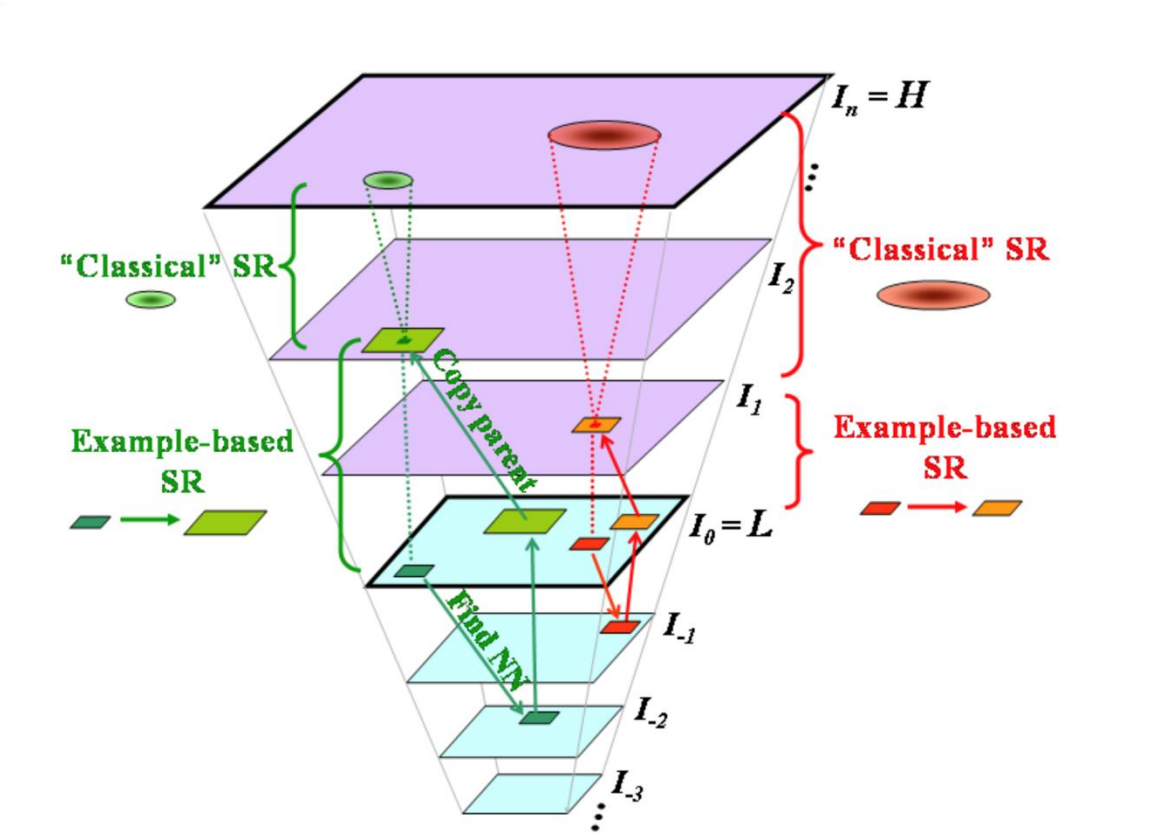
۴.۲.۲.۳ ایجاد رابطه خطی[۲]

ناحیه کپی شده در تصویر با وضوح بالای مجهول، یک رابطه خطی برای به دست آوردن مقادیر پیکسل‌ها در تصویر وضوح‌بالای مطلوب به دست می‌دهد. در صورتی که این روابط خطی به تعداد مناسبی برسند، می‌توان تصویر با وضوح بالای مطلوب را به دست آورد. با تکرار مراحل قبل به ازای هر پیکسل موجود در تصویر ورودی، تعداد روابط خطی موجود به اندازه قابل قبول برای به دست آوردن تصویر با وضوح بالای مجهول می‌رسد. این روابط خطی به صورت زیر هستند:

که در آن، H بیانگر تصویر با وضوح بالا، L تصویر با وضوح پایین، و B بیانگر تابع تار کننده هستند. پس از این مرحله، تصویر با وضوح بالای مجهول ساخته شده و خروجی به صورت فایل ذخیره می‌شود. طول و عرض تصویر ورودی، هر دو حدود ۴ برابر می‌شود و در نتیجه تصویر نهایی حدود ۱۶ برابر تصویر ورودی خواهد شد.

[۱] Unknown High-Resolution Image

[۲] Linear Constraints



شکل ۱۵ شمای کلی قسمت فراتفکیک پذیری پروژه

۳.۲.۳ دریافت تصویر مورد نظر جهت تشخیص چهره

در این قسمت از پروژه، کاربر مسیر تصویر ورودی جهت انجام پردازش‌های لازم جهت تشخیص چهره را در برنامه وارد می‌کند. این تصویر ورودی می‌تواند تصویری باشد که در مرحله قبل عملیات فراتفکیک پذیری بر رویش صورت گرفت و یا هر تصویری که کاربر مد نظر قرار دارد. مسیر تصویر ورودی به تابع مورد نظر جهت اجرا پردازش‌های تشخیص چهره فرستاده می‌شود.

۴.۲.۳ انجام عملیات تشخیص چهره

در این مرحله، عملیات تشخیص چهره و پارامترهای مورد نظر آن مشخص می‌شوند. عملیات تشخیص چهره از چند مرحله تشکیل شده است که این مراحل در ادامه مطرح می‌گردند.[۱۲]

به منظور انجام عملیات تشخیص چهره، در ابتدا باید پایگاه داده‌ای مناسب جهت آموزش و تست پروژه داشته باشیم. پایگاه داده مورد استفاده در این پروژه، پایگاه داده‌ای شامل ۴۵۰ تصویر چهره از ۲۷ فرد متفاوت در شرایط نوری و حالت‌های چهره متفاوت است.[۱۳] در این پروژه، از هر فرد یک تصویر به منظور تست جدا شده و سایر تصاویر برای آموزش مورد استفاده قرار گرفته‌اند.

۱.۴.۲.۳ مرحله اول: آموزش

در مرحله اول، باید به نرم افزار آموزش دهیم که هر چهره مربوط به کدام یک از افراد می‌باشد. اگر از هر فرد n تصویر موجود باشد، $n-1$ تصویر جهت آموزش استفاده شده‌اند و یک تصویر باقیمانده جهت تست استفاده شده است. برای آموزش باید مشخص کنیم که از کدام یک از روش‌های استخراج ویژگی‌ها استفاده می‌کنیم. این روش‌ها در ادامه توضیح داده می‌شود. در صورتی که از هر یک از این روش‌ها استفاده کنیم، هنگام تست نیز باید از همان روش جهت استخراج ویژگی‌ها استفاده نماییم. توابع مورد نظر جهت آموزش در کلاس train پیاده سازی شده‌اند که در نهایت خروجی این مرحله یک فایل شامل ویژگی‌ها و فاصله‌های هر تصویر و هر فرد می‌باشد. در این کلاس، در ابتدا یک شی به نام recognizer با توجه به الگوریتم مورد نظر جهت استخراج ویژگی‌ها می‌سازیم. این کار به صورت زیر انجام می‌شود.

```
59      # FEATURE EXTRACTION ALGORITHM
60
61      if featureExtractionAlg == "1":
62          recognizer = cv2.face.createLBPHFaceRecognizer()
63      elif featureExtractionAlg == "2":
64          recognizer = cv2.face.createFisherFaceRecognizer()
65      else:
66          recognizer = cv2.face.createEigenFaceRecognizer()
67          # recognizer = cv2.face.createEigenFaceRecognizer(10)
68
69
```

شکل ۱۶ تکه کد مربوط به ساخت شی recognizer

تابع saveRecognizer() وظیفه ذخیره سازی شی recognizer را بر عهده دارد. پس از فراخوانی این تابع، فایل خروجی نتیجه آموزش در سیستم ذخیره می‌شود. از این فایل جهت شناسایی چهره در آینده استفاده می‌شود.

```
109
110     def saveRecognizer(recognizer, filename=config.RECOGNIZER_OUTPUT_FILE):
111         recognizer.save(filename)
112
```

شکل ۱۷ تابع saveRecognizer()

۲.۴.۲.۳ مرحله دوم: شناسایی چهره

در این مرحله، ابتدا باید چهره موجود در هر تصویر شناسایی و پیدا شود. این کار در کلاس detect و تابع detectFaces() انجام می‌شود. شناسایی چهره با استفاده از کتابخانه OpenCV به سادگی قابل انجام است. تنها کافی است تصویر را به فضای رنگی سیاه سفید برده، و با استفاده از تکنیک‌های مختلف شناسایی چهره، چهره‌های گوناگون را در تصاویر بیابیم. برای این منظور، کتابخانه OpenCV هنگام نصب در سیستم، تعدادی فایل با فرمت XML در سیستم قرار می‌دهد. این فایل‌ها، با استفاده از الگوریتم template matching و ویژگی‌های مشخصی آموزش داده شده اند و با فراخوانی تابع detectFaces() از کلاس detect و سپس تابع detectMultiScale()، که ورودی‌اش تصویر مورد نظر است، عملیات شناسایی چهره انجام می‌شود.

```
32      def detectFaces(image, faceCascade, eyeCascade=None, returnGray=True):
33          cas_rejectLevel = 1.3
34          cas_levelWeight = 5
35
36          # Convert color input image to grayscale
37          gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
38
39          #
40          # Detect faces using grayscale images and XML Files
41          #
42          faces = faceCascade.detectMultiScale(gray, cas_rejectLevel, cas_levelWeight)
43
```

شکل ۱۸ تابع detectFaces()

۳.۴.۲.۳ مرحله سوم: استخراج ویژگی[۱]

تصاویر به دلیل بعد بسیار بالایی که دارند، بار پردازشی سنگینی بر روی سیستم‌ها به وجود می‌آورند که در صورتی که از این بار پردازشی کاسته نشود، تنها ابرکامپیوترها قادر به انجام عملیات پردازش تصویر خواهند بود. به همین منظور، باید ویژگی‌های غیرضروری تصاویر را حذف کرده و تا جایی که می‌شود ویژگی‌های لازم را نگه داشت. هدف استخراج ویژگی این است که داده‌های خام به شکل قابل استفاده‌تری برای پردازش‌های آماری بعدی درآیند. روش‌های مختلف استخراج ویژگی بنا به فلسفه پشت سرشان ممکن است یک یا چند کار زیر را انجام دهند:

- حذف نویز داده‌ها
- جداسازی اجزای مستقل داده‌ها
- فروکاهی ابعاد برای تولید بازنمایی مختصرتر
- افزایش بعد برای تولید بازنمایی جدایی‌پذیرتر

انجام استخراج ویژگی فرایند بسیار متداولی در انواع مختلف پردازش داده‌ها چون پردازش تصویر، پردازش صوت و غیره است. فضای صورت معمولا با تحلیل مولفه‌های اصلی[۲] و یا تجزیه و تحلیل تفکیک خطی[۳] از پایگاه داده صورت محاسبه می‌شود. تحلیل مولفه‌های اصلی در الگوریتم استخراج ویژگی eigenfaces و تحلیل تفکیک خطی در الگوریتم استخراج ویژگی fisherfaces مورد استفاده قرار می‌گیرد. این الگوریتم‌ها شرح ریاضی از ویژگی‌های غالب مجموعه آموزشی به صورت کلی ارائه می‌کنند. الگوریتم استخراج ویژگی دیگری که در این پروژه پیاده سازی شده است، الگوریتم استخراج ویژگی lbph می‌باشد. این الگوریتم با آنالیز هر تصویر مجموعه آموزشی به صورت تکی، توصیفی از الگوهای محلی می‌سازد.

---

[۱] Feature Extraction

[۲] Principal Component Analysis

[۳] Linear Discriminant Analysis

الگوریتم‌های fisherfaces و eigenfaces برای شرایطی که تغییرات نور داریم مناسب‌تر هستند زیرا میزان تاثیر تغییرات نور را کم می‌کنند. در این پروژه، این سه الگوریتم استخراج ویژگی‌ها پیاده سازی شده‌اند. مورد قابل توجه این که با هر الگوریتم استخراج ویژگی‌هایی که سیستم آموزش دیده است، با همان باید عملیات تست را انجام داد.

۴.۴.۲.۳ مرحله چهارم: تشخیص چهره

در این مرحله، عملیات تشخیص چهره با استفاده از پردازش‌هایی که تا الان انجام داده ایم صورت می‌گیرد. عملیات تشخیص چهره شامل الگوریتم‌های متخلفی با کارایی‌های متفاوت می‌باشد، از جمله آن‌ها:

- تکنیک‌های دسته بندی[۱]
  - ماشین بردار پشتیبان[۲]
  - شبکه عصبی[۳]
- مدل‌های آماری[۴]
  - دسته بندی کننده بیز[۵]
- مقیاس فاصله/شباهت[۶]
  - فاصله اقلیدسی
  - فاصله منهتن

[۱] Classification Techniques

[۲] Support Vector Machines

[۳] Neural Network

[۴] Statistical Modeling

[۵] Bayes classifier

[۶] Distance/Similarity Measures

در این پروژه، الگوریتم تشخیص چهره پیاده سازی شده فاصله اقلیدسی می‌باشد که می‌توان گفت به نوعی از دسته الگوریتم Approximate Nearest Neighbor می‌باشد. محاسبه فاصله تصویر ورودی با تصاویر آموزش داده شده با استفاده از تابع predict() که به صورت پیش فرض توسط کتابخانه OpenCV پیاده سازی شده و آماده می‌باشد، انجام می‌شود. این تابع در تابع recognizeFace() در کلاس recognize فراخوانی می‌شود.

```
10
11      def recognizeFace(image, faceCascade, eyeCascade, faceSize, threshold, recognizer):
12          found_faces = []
13
14          gray, faces = detect.detectFaces(image, faceCascade, eyeCascade, returnGray=1)
```

شکل ۱۹ فراخوانی تابع predict()

پس از فراخوانی تابع predict()، ویژگی‌های تصاویر آموزشی با تصویر ورودی مقایسه می‌شوند و تصویر آموزشی که کمترین فاصله را با تصویر ورودی داشته باشد، به عنوان صاحب تصویر ورودی در نظر گرفته می‌شود و برچسب یکسانی متناسب با نام فرد دریافت می‌کنند.

۵.۲.۳ نمایش خروجی‌ها

برای پیاده سازی این بخش از پروژه، دو قابلیت مختلف باید پیاده شوند؛ اول آنکه تصویر باید به صورت یک فایل در سیستم ذخیره شود تا در زمان مورد نیاز، کاربر بتواند آن را مشاهده کند و دوم آنکه در پنلی خروجی پردازش تشخیص چهره نمایش داده شود.

۱.۵.۲.۳ ذخیره خروجی

برای این منظور با استفاده از تابع imwrite() که به صورت پیش فرض در کتابخانه OpenCV موجود می‌باشد، تصویر خروجی را به صورت فایل jpg بر روی سیستم ذخیره می‌کنیم.

۲.۵.۲.۳ نمایش خروجی

در این قسمت، باید در پنلی که به عنوان رابط گرافیکی[۱] کاربر ساخته ایم، خروجی تصویری که جهت تشخیص چهره به برنامه داده ایم نمایش داده شود. پس از اینکه فایل خروجی در سیستم ذخیره شد، آن را به رابط گرافیکی فرستاده و در آن جا نمایش داده می‌شود.

۳.۳ چالشها

در روند پیاده سازی این پروژه، مسائلی مورد توجه هستند:

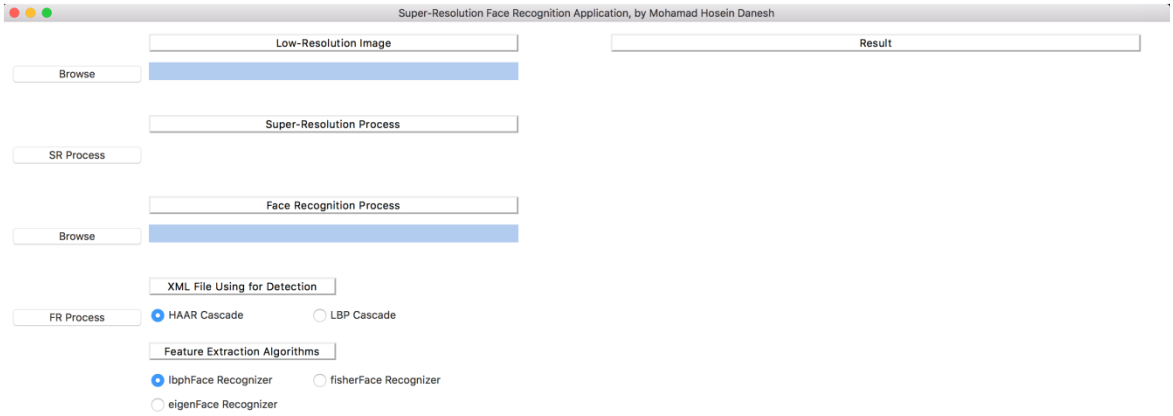
- با توجه به این که کتابخانه‌های OpenCV در اصل به زبان C و C++ هستند، نسخه‌های موجود برای زبان‌هایی مانند پایتون، توابع بسیار کمتری از بسته‌ی اصلی را دارند و بسیاری از توابع منتقل شده، عملکرد درست و مشخصی ندارند. علاوه بر آن حمایت و استناد از این نسخ بسیار کم هستند و عملکرد برخی بخش‌ها مشخص نیست. به طوری که در پیاده سازی آنها مجبور به تعریف دوباره توابع و عملکردشان هستیم.
- چالش دیگر پروژه آن است که پایگاه داده مناسبی شامل عبور و مرور افراد که بتوان بر روی آن عملیات تشخیص چهره و فراتفکیک پذیری را پیاده کرد به صورت رایگان وجود ندارد. به همین دلیل، عملیات فراتفکیک پذیری و تشخیص چهره بر روی پایگاه داده‌ای شامل تصاویر چهره افراد پیاده سازی شده است.
- مشکل دیگر مسئله از ابتدای کار، کیفیت بالای دوربین وب‌کم سیستم بود. کیفیت و وضوح بالای این دوربین، عملیات فراتفکیک پذیری را به شدت کند می‌کند، به صورتی که به ازای هر تصویر گرفته شده توسط این دوربین مدت زمان زیادی را باید صرف عملیات فراتفکیک پذیری نمود که عملا کاربردی نخواهد بود.

[۱] GUI

۴ فصل چهارم

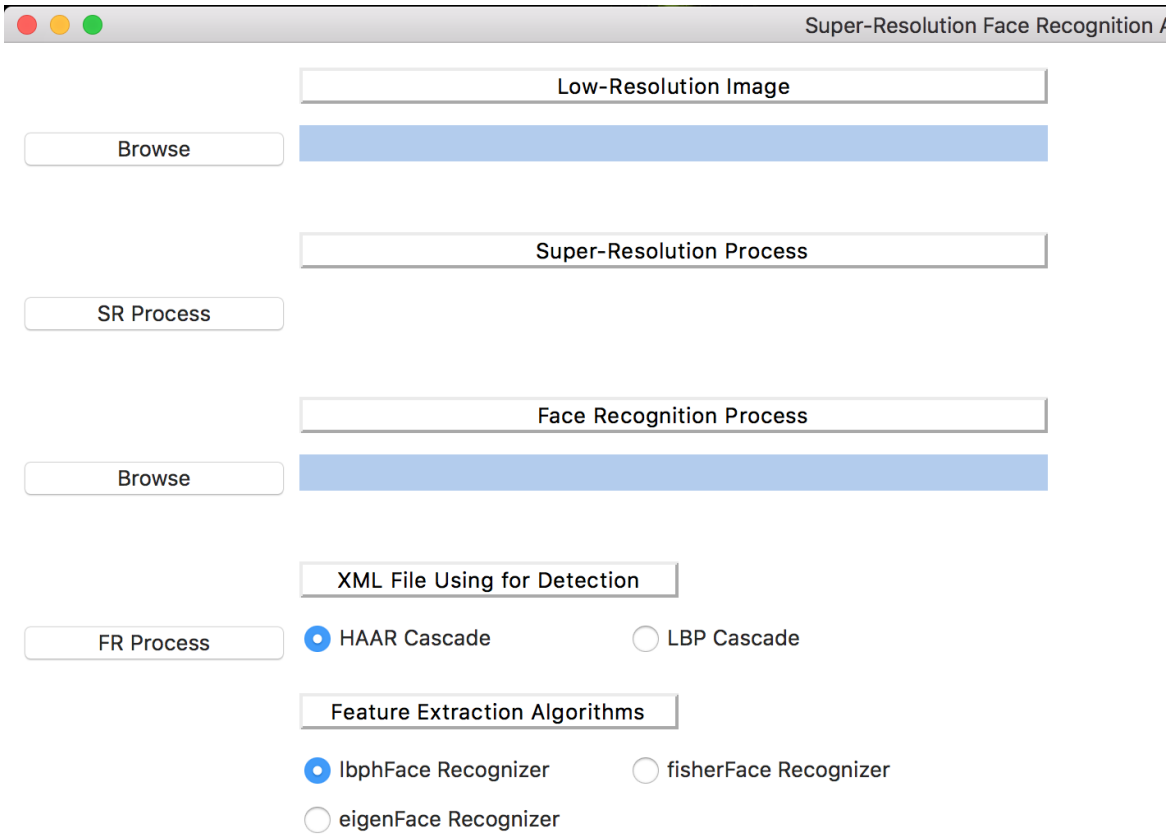
نرم افزار تشخیص چهره تصاویر فراتفکیک پذیر

در این بخش نحوه‌ی اجرا و تنظیم ورودیهای مورد نظر و دریافت خروجی در نرم افزار پیاده شده مطرح خواهد شد. تصویر زیر، شمای کلی برنامه را نمایش می‌دهد. در ادامه‌ی بحث به بررسی کارکرد هر کدام از بخش‌های نرم افزار می‌پردازیم.



شکل ۲۰ نمای صفحه اصلی برنامه

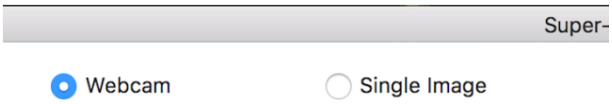
کاربر می‌تواند با استفاده از پنل سمت چپ ورودی سیستم را مشخص کند و تنظیمات مورد نظر را پیاده سازی نماید.



شکل ۲۱ پنل ورودی و تنظیمات مورد نظر

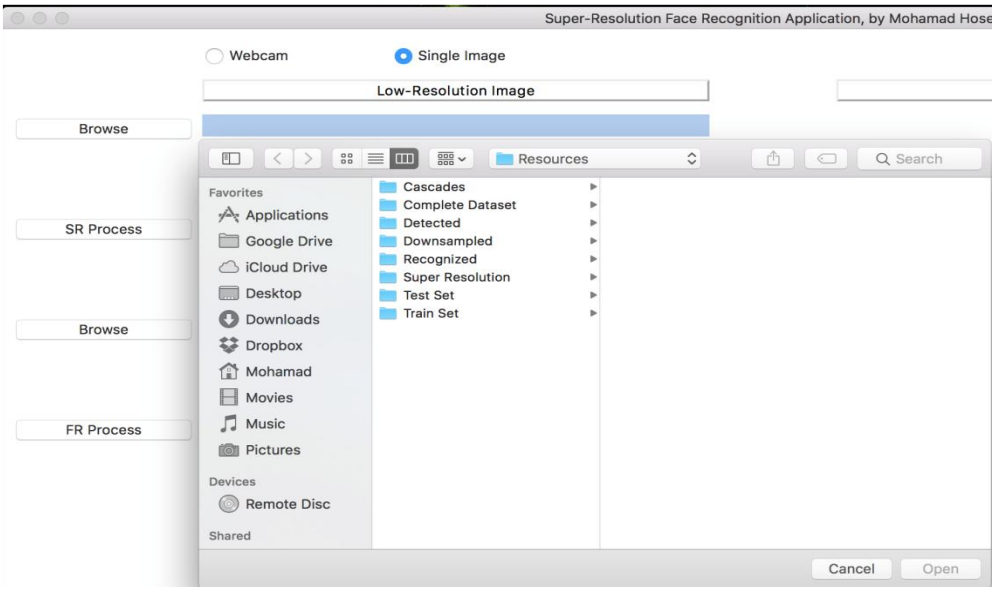
۱.۱.۴ فایل ورودی فراتفکیک پذیری

قبل از راه اندازی سیستم، ابتدا باید مبدا ورودی مورد پردازش معین شود. برای این منظور کاربر می‌تواند با استفاده از دکمه‌های رادیویی منبع ورودی را مشخص کند. در صورتی که کاربر وب‌کم را انتخاب کند، ویدئو ورودی از وب‌کم دریافت شده و آخرین فریم آن به عنوان تصویر مورد نظر جهت انجام پردازش فراتفکیک پذیری جدا می‌شود. در صورتی که کاربر ورودی تصویر را انتخاب کند، باید آدرس تصویر ورودی را به برنامه داده و برنامه آن را جهت پردازش فراتفکیک پذیری اجرا می‌کند.



شکل ۲۲ دکمه‌های رادیویی انتخاب ورودی

اگر کاربر از این منو، گزینه‌ی وبکم را انتخاب نماید، می‌تواند مستقیما پردازش را شروع کند. اما اگر فایل را انتخاب نماید، باید فایل ورودی را با کلید بر روی دکمه "Browse" مشخص نماید.

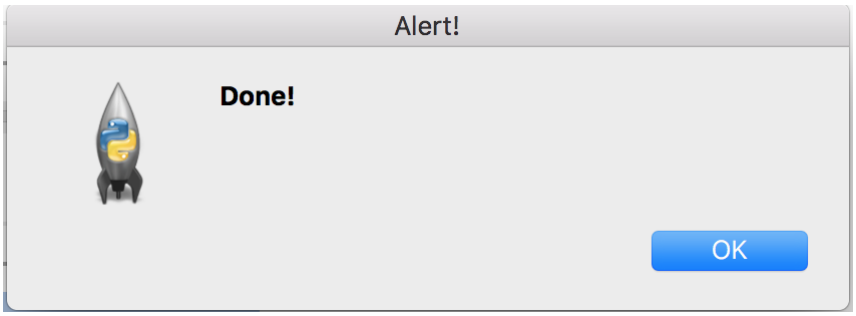


شکل ۲۳ انتخاب فایل تصویر ورودی

۲.۱.۴ پردازش فراتفکیک پذیری

پس از انتخاب ورودی، کاربر می‌تواند با استفاده از کلید "SR Process" پردازش را با استفاده از تنظیمات اولی و پیش فرض شروع نماید. تنظیمات برنامه در هر لحظه از اجرای برنامه قابل تغییر هستند.

پس از آن که پردازش فراتفکیک پذیری بر روی تصویر پایان یافت، پیغامی‌مبنی بر پایان یافتن پردازش به درستی نمایش داده می‌شود و فایل خروجی این مرحله به نام "high-res.png" در سیستم ذخیره می‌شود.



شکل ۲۵ پیغام تمام شدن پردازش فراتفکیک پذیری

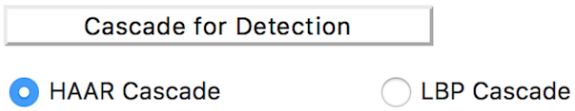
۳.۱.۴ ورودی تشخیص چهره

در این بخش، کاربر باید آدرس تصویر مورد نظرش را جهت پردازش‌های مربوط به تشخیص چهره به برنامه بدهد. این تصویر ورودی هم می‌تواند تصویری باشد که در مرحله قبل وضوحش بهبود یافت، هم تصویر دیگری که مدنظر کاربر می‌باشد.

۴.۱.۴ انتخاب الگوریتم تشخیص چهره

در این مرحله کاربر باید الگوریتم مناسب و موردنظرش را جهت پردازش تشخیص چهره انتخاب کند. این الگوریتم‌ها با استفاده از دکمه‌های رادیویی تعبیه شده اند. haar cascade یک روش بسیار

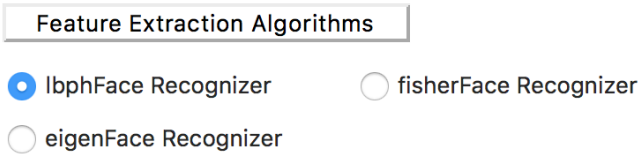
موثر است که اولین بار توسط Paul Viola و Michael Jones در سال ۲۰۰۱ مطرح شد.[۱۴] lbph cascade در مقایسه با haar cascade دقت کمتر(۱۰ الی ۲۰ درصد) و سرعت بیشتری دارد. این الگوریتم‌ها بر پایه یادگیری ماشین بوده و بدین صورت عمل می‌کند که با استفاده از تصاویری که چهره در آنها وجود دارد (تصاویر مثبت) و تصاویری که چهره در آنها وجود ندارد (تصاویر منفی) تابعی را آموزش می‌دهد تا بتواند مولفه‌هایی مانند وجود فاصله بین چشم‌ها را پیدا کند. این الگوریتم منحصر به تشخیص چهره نیست؛ و بیشتر یک آموزش دهنده است و می‌توان با آموزش دادن تابع دلخواه خودتان هر شی مانند ماشین، میز، مداد و ... را شناسایی کنید. OpenCV مجموعه‌ای از توابع از پیش آموزش دیده را برای تشخیص چهره، چشمان، لبخند و ... در خود دارد که می‌توان به سادگی از آنها استفاده کرد.



شکل ۲۶ دکمه‌های رادیویی انتخاب فایل cascade

۵.۱.۴ انتخاب الگوریتم استخراج ویژگی

در این قسمت کاربر باید الگوریتم مورد نظر خود را جهت استخراج ویژگی‌های تصویر ورودی انتخاب کند. باید توجه داشت که کاربر باید همان الگوریتمی‌را انتخاب کند که هنگام آموزش از آن استفاده کرده است، زیرا ویژگی‌های تصویر ورودی باید با فایل خروجی از تصاویر آموزشی مقایسه شود و در صورتی که از الگوریتم‌های مختلف استفاده شود، نتیجه‌ای بدست نخواهد آمد.



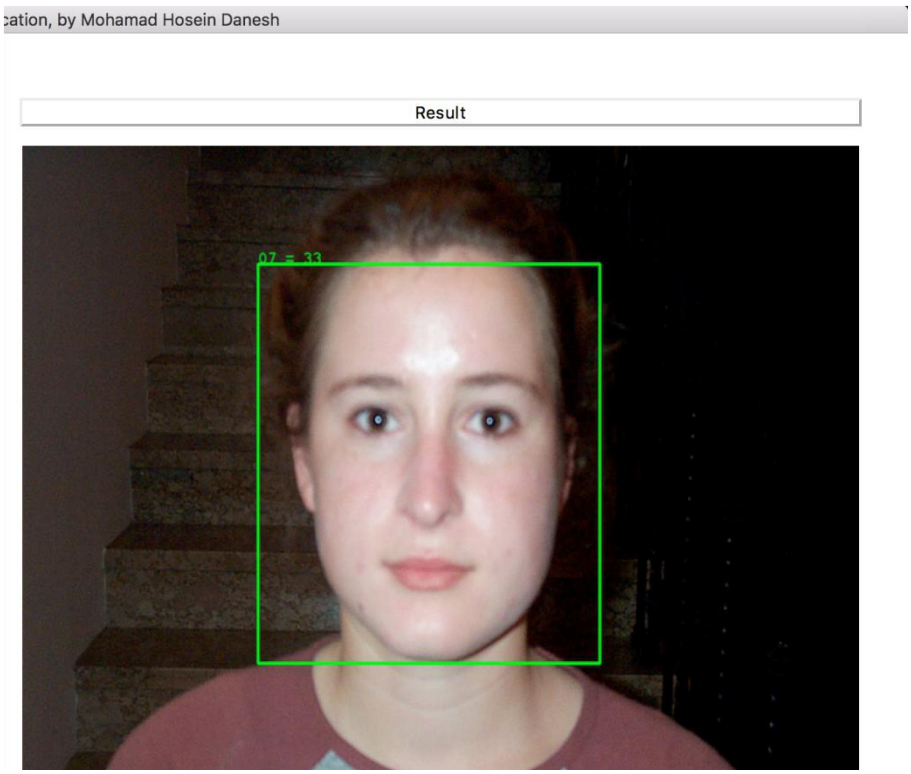
شکل ۲۷ دکمه‌های رادیویی انتخاب الگوریتم استخراج ویژگی

۶.۱.۴ پردازش تشخیص چهره

پس از انتخاب الگوریتم‌های مورد نظر، کاربر باید پردازش تشخیص چهره را آغاز کند. این کار با کلیک بر روی دکمه "FR Process" انجام می‌شود.

۲.۴ مشاهده ی خروجی

پس از آن که کاربر بر روی دکمه "FR Process" کلیک کرد، پردازش تشخیص چهره آغاز می‌شود و نتیجه آن در قسمت سمت راست رابط کاربری نمایش داده می‌شود. بر روی چهره شناسایی شده موجود در تصویر، یک مربع سبز کشیده می‌شود، و برچسب و میزان شباهت تصویر ورودی با برچسب نمایش داده می‌شود. به عنوان مثال، در شکل ۲۸، تصویر شناسایی شده برچسب ۷ و میزان شباهت ۳۳ دارد.



شکل ۲۸ نمونه خروجی پردازش تشخیص چهره

۳.۴ تست نرم افزار

در این بخش به بررسی مشخصات سیستم کامپیوتری مورد استفاده و ورودی‌های فراهم شده می‌پردازیم.

۱.۳.۴ سیستم

مشخصات اولیه‌ی سیستم مورد استفاده در تصویر زیر نمایش داده شده است.

شکل ۲۹ مشخصات سیستم مورد استفاده

۲.۳.۴ آزمایش‌ها

برای آزمایش، پایگاه داده‌ای شامل ۴۴۹ تصویر از ۲۸ فرد داشتیم که از میان آن‌ها ۴۲۱ تصویر جهت آموزش و ۲۸ تصویر جهت تست جدا کردیم.

در ابتدا جهت عملیات فراتفکیک پذیری، تصاویر با وضوح ۱۴۸*۲۲۴ به عملیات فراتفکیک پذیری داده شدند و تصاویر با وضوح ۵۹۱*۸۹۳ ساخته شدند. سپس آن‌ها به عنوان ورودی به عملیات تشخیص چهره داده شدند و دقت سیستم به دست آمد.

در آزمایش دیگری، تصاویر با وضوح بالا(بدون کاهش مقیاس و سپس فراتفکیک پذیری) به سیستم داده شدند که دقت بسیار بیشتری داشتند. طبیعی است که تصاویری که اطلاعات بیشتری دارند، دقت بیشتری نیز داشته باشند.

در جدول ۱ نتایج این آزمایش‌ها مشخص و مقایسه شده اند.

جدول ۱ نتایج بدست آمده در آزمایش‌ها

Cascade	**HAAR Cascade**			**LBP Cascade**		
Feature Extraction Algorithm	**lbphFace Recognizer**	**fisherFace Recognizer**	**eigenFace Recognizer**	**lbphFace Recognizer**	**fisherFace Recognizer**	**eigenFace Recognizer**
Preprocessed Face Images	۹۲,۸۶٪	۸۲,۱۴٪	۸۵,۷۱٪	۸۹,۲۸٪	۶۷,۸۶٪	۶۷,۸۶٪
Down-Sampled Face Images	۱۷,۸۶٪	۶۷,۸۶٪	۴۲,۸۶٪	۲۸,۵۷٪	۶۲,۲۸٪	۶۴,۲۸٪
Super-Resolution Face Images	۴۲,۸۶٪	۷۸,۵۷٪	۷۸,۵۷٪	۳۲,۱۴٪	۷۵٪	۶۷,۸۶٪

۵ فصل پنجم

جمع بندی و کارهای آینده

همانطور که بررسی شد هدف این پروژه، طراحی و پیاده‌سازی نرم افزاری برای تشخیص چهره در تصاویر فراتفکیک پذیر بود. در این پروژه با استفاده از ابزارهای پردازش تصویر، کتابخانه‌ها و توابع پیاده‌شده ورودی دریافت می‌شود، و تصویر مورد نظر مورد پردازش و تغییر قرار می‌گیرد.

پردازش‌هایی که بر روی یک فریم تصویر پیاده شدند به شرح زیرند:

- دریافت تصویر با وضوح پایین
- انجام عملیات فراتفکیک پذیری
- ذخیره تصویر پردازش شده
- آموزش سیستم تشخیص چهره
- دریافت تصویر جهت عملیات تشخیص چهره و تبدیل ابعاد آن به بعد پیش فرض برنامه
- مقایسه تصویر ورودی با داده‌های خروجی آموزش
- ذخیره و نمایش تصویر نتیجه

در این پروژه، کتابخانه‌ی OpenCV برای پردازش تصاویر به کار رفت و زبان برنامه‌ی نوشته شده پایتون می‌باشد. با اجرای برنامه، نوع ورودی توسط کاربر تعیین شده و اندازه‌ی فریم‌های ورودی هماهنگ می‌شوند. پس از آن فریم خوانده شده از مبدا به صورت MAT ذخیره سازی شده و پردازش‌های لازم روی آن انجام می‌شود. سپس فایل تصویر با وضوح بالاتر ذخیره شده و کاربر تصویر مورد نظرش را جهت اجرای عملیات تشخیص چهره به برنامه می‌دهد. پس از اینکه عملیات تشخیص

چهره انجام شد، تصویر نهایی در پنل رابط گرافیکی برنامه نمایش داده شده و در سیستم به صورت فایل تصویر ذخیره می‌شود.

در آینده می‌توان با در نظر داشتن مسائل زیر پروژه را بهبود داد:

- تهیه پایگاه داده‌ای که شامل تصاویر دوربین‌های مداربسته باشد و چهره افراد در آن به صورت روبرو نمایش داده شود. در این صورت، می‌توان آموزش بخش تشخیص چهره را نیز با استفاده از فریم‌های این پایگاه داده انجام داد.
- بالا بردن وضوح تصاویر بار پردازشی زیادی بر روی سیستم دارد. می توان با پیاده‌سازی الگوریتم‌های بهینه‌تر زمان پردازش و بار پردازش کمتری داشته باشیم.
- الگوریتم فراتفکیک پذیری پیاده‌سازی شده در این پروژه به صورت عمومی برای همه تصاویر کاربرد دارد. می‌توان الگوریتم‌هایی مخصوص برنامه‌های تشخیص چهره ابداع و پیاده‌سازی کرد.

منابع و مراجع

[۱] W. Zhao, R. Chellappa, P.J. Phillips and A. Rosenfeld, “Face Recognition: A Literature Survey” in ACM Computing Surveys, Vol. ۳۵, No. ٤, ۲۰۰۳, pp. ۳۹۹-٤٥٨.

[۲] S. Biswas, K. W. Bowyer, and P. J. Flynn, “Multidimensional scaling for matching low-resolution face images” in IEEE Transactions on Pattern Analysis and Machine Intelligence, ۳٤(۱۰):۲۰۱۹–۲۰۳۰, ۲۰۱۲.

[۳] “۱۰ great uses of image and face recognition”, by Martin Bryant, Available: https://thenextweb.com/apps/۲۰۱۱/۰۸/۱۹/۱۰-great-uses-of-image-and-face-recognition. [Accessed: ۰۸-Apr-۲۰۱۷]

[٤] R. Brunelli, T. Poggio, “Face Recognition: Features versus Templates”, in IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. ۱٥, pp. ۱۰٤۲-۱۰٥۲.

[٥] “Better Face-Recognition Software”, by Mark Williams Pontin, Available: https://www.technologyreview.com/s/٤۰۷۹۷٦/better-face-recognition-software. [Accessed: ۰۷-Apr-۲۰۱۷]

[٦] A. M. Bronstein, M. M. Bronstein, R. Kimmel, “Three-Dimensional Face Recognition” in Int J Comput Vision (۲۰۰٥), ٦٤: ٥. doi:۱۰,۱۰۰۷/s۱۱۲٦۳-۰۰٥-۱۰۸٥-y

[۷] L. J. Harris, “Resolving power and decision making”, J. opt. soc. Am. ٥٤, ٦۰٦-٦۱۱.

[۸] “Installing and Launching”, Available: https://www.jetbrains.com/help/pycharm/۲۰۱٦,۳/installing-and-launching.html. [Accessed: ۰۸-Apr-۲۰۱۷]

[۹] D. Glasner, S. Bagon, M. Irani, “Super-Resolution from a Single Image” in Proc. of ICCV. (۲۰۰۹)

[۱۰] “Image Filtering”, Available: http://docs.opencv.org/۲,٤/modules/imgproc/doc/filtering.html. [Accessed: ۰۸-Apr-۲۰۱۷]

[۱۱] S. Arya, D. M. Mount, ۱۹۹۳. “Approximate nearest neighbor searching” in Proc. ۴th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. ۲۷۱–۲۸۰.

[۱۲] X. Qi, “Face Recognition”, Utah State University.

[۱۳] Computational Vision at Caltech, “Faces ۱۹۹۹ (Front)”.

[۱۴] P. Viola, M. Jones, “Rapid object detection using a boosted cascade of simple features” in: CVPR, issue ۱, ۲۰۰۱, pp. ۵۱۱–۵۱۸.

Amirkabir University of Technology
(Tehran Polytechnic)

Computer Engineering and IT department

B.Sc. Thesis

Title

Design and Implementation of a Super-Resolution Face Recognition Software

By

Mohamad Hosein Danesh

Supervisor

Dr. Mohammad Rahmati