
بسم الله الرحمن الرحيم

دانشکده مهندسی برق و رباتیک

گروه الکترونیک

گرایش سیستم

پایان نامه کارشناسی ارشد

الگوریتم ابرپیکسل بهینه شده در چارچوب بینایی فرد منظر

سیّد امیر حسین فرزانه

استاد راهنما: دکتر حسین خسروی

تیر ماه ۱۳۹۴

شماره : ۱۲۹۵/أ.ت.ب
تاریخ : ۹۴/۰۴/۲۰
ویرایش : ------

بسمه تعالی

مدیریت تحصیلات تکمیلی
فرم شماره (۶)

فرم صورتجلسه دفاع پایان نامه تحصیلی دوره کارشناسی ارشد

با تأییدات خداوند متعال و با استعانت از حضرت ولی عصر (عج) جلسه دفاع از پایان نامه کارشناسی ارشد خانم / آقای :

سید امیر حسین فرزانه رشته : **برق - الکترونیک** گرایش : **سیستم**

تحت عنوان : **الگوریتم ابر پیکسل بهینه شده در چارچوب بینایی فرد منظر**

که در تاریخ **۹۴/۰۴/۲۰** با حضور هیأت محترم داوران در دانشگاه صنعتی شاهرود برگزار گردید به شرح زیر است :

قبول (با درجه : بسیار خوب امتیاز ۱۸٫۷۵) ☑ دفاع مجدد ☐ مردود ☐

۱ـ عالی (۲۰ ـ ۱۹)
۲ـ بسیار خوب (۱۸/۹۹ ـ ۱۸)
۳ـ خوب (۱۷/۹۹ ـ ۱۶)
۴ـ قابل قبول (۱۵/۹۹ ـ ۱۴)
۵- نمره کمتر از ۱۴ غیر قابل قبول

عضو هیأت داوران	نام ونام خانوادگی	مرتبهٔ علمی	امضـــاء
۱ـ استاد راهنما	حسین حسنی	استاد	
۲ـ استاد مشاور	------	------	------
۳ـ نماینده شورای تحصیلات تکمیلی	سامان ناصح	استادیار	
۴ـ استاد ممتحن	علیرضا احمدی فرد	دانشیار	
۵ـ استاد ممتحن	حمید حسن پور	استاد	

رئیس دانشکده : علیرضا احمدی فرد

تقدیم به پدر و مادر عزیزم،

که در این مسیر من را باور داشتند و ...

تقدیم به پدر بزرگ هنرمند و با احساسم (سیّد حسین فرزانه) به پاس همه حمایتهایش.

با سپاس ...

از زحمات و راهنمایی های استاد راهنمای بزرگوارم جناب آقای دکتر حسین خسروی که در مسیر این تحقیق همواره مرا یاری نمودند.

سیّد امیر حسین فرزانه ۱۳۹۴

چکیده

پیشرفتهای نوین تکنولوژی زمینه‌ی مناسبی برای تولید دوربینهای سبک با کاربرد وسیع و قابل اتصال بر روی سر، فراهم کرده است. محصولاتی چون عینک گوگل نمونه ای از ابزارکهای پوشیدنی است که همگان می‌توانند از آن بهره‌مند شوند. این ابزارکها ایده ویدئوهای فرد منظر (اول شخص) را به عنوان مسیر و چهار چوبی جدید، پیش روی جامعه پژوهشگران بینایی ماشین قرار داده اند. در چارچوب ویدئوهای فرد منظر می توان به بازشناسی اشیاء و فعالیتها از دید فرد پرداخت. در پزشکی، صنعت و هنر قابلیت های بینایی فرد با پردازش این تصاویر افزایش می یابد و راه حل های نوینی به منظور غلبه بر چالش ها معرفی می گردد.از سویی تصاویر فرد منظر، پژوهشگران را با چالش هایی چون حرکت غیر خطی، تاری ناشی از حرکت، گذار سریع و سرعت الگوریتم های پردازشی روبرو می کند. در این پایان نامه، به منظور بهینه‌سازی یک الگوریتم ابرپیکسل در چارچوب تصاویر فرد منظر، روند جدیدی پیشنهاد شده است. در کاربرد ویدئوهای فرد منظرِ دریافتی از ابزارکهای پوشیدنی، به یک الگوریتم پردازشی سریع و قابل پیاده سازی بر روی ابزارکهای یاد شده، نیاز است. در روش پیشنهادی، با برقراری سازش میان کیفیت جداسازی و زمان اجرای الگوریتم، به همگرایی سریعتر و پردازش بلادرنگ نزدیکتر شده ایم. برای ویدئویی با ابعاد ۷۲۰×۱۲۸۰ پیکسل در راهبردهای الگوریتمی متفاوت، از ۲۳ تا ۶۴ درصد بهبود عملکرد حاصل شده است. همچنین الگوریتم جدید می‌تواند شبکه منظم تر و پایدارتری از ابرپیکسل‌ها ارائه دهد. بهبود سرعت الگوریتم، سبب کاهش ناچیزِ کیفیت جداسازی در نواحی هموار یک قاب از ویدئو شده است که البته این کاهش، تاثیری در کیفیت جداسازی مرزهای اشیاءِ درون تصویر، ندارد.

کلید واژه‌ها

ابر پیکسل، ویدئو، فرد منظر، پردازش ویدئو.

فهرست

فهرست شکل ها

فهرست جدول‌ها

۱. فصل اول: مقدمه

مقدمه

در طول تاریخ همواره استفاده از سامانه‌های کامپیوتری پوشیدنی در صنعت فیلم و سرگرمی، هنرهای نمایشی و ادبیات علمی‌تخیلی، رویاپردازی شده است؛ آرمان شهرهایی که در آن ابزارهای الکترونیکی، به اندازه ای کوچک و سبک هستند که جزئی از لباس یا بدن انسان‌ها هستند. امروزه شاهد گسترش روز به روز و پر شتاب ابزارک‌هایی در دنیای تکنولوژی چون تلفن‌های هوشمند، تبلت‌ها و ... هستیم. قابلیت حمل به عنوان مهم ترین ویژگی، گستره کاری این ابزارکها را وسعت بخشیده است. اما محدودیت‌هایی نیز وجود دارد؛ در کاربری این ابزارک‌ها به توجه کامل و حداقل استفاده از یک دست انسان نیاز است. ظهور ابزارک‌های پوشیدنی پاسخی برای این محدودیت‌ها است. ابزارک‌های پوشیدنی، تعامل انسان با تکنولوژی را متحول کرده اند. علاوه بر فراهم آوردن تجربه کاربری بهتر در شرایط خاص، این نوع از ابزارک‌ها شامل حسگرهایی هستند که هوشیاری زمینه ای[1] کاربر را افزایش می‌دهند. در علم کامپیوتر، هوشیاری زمینه ای به مفهوم حس کردن و واکنش به محیط اطراف توسط کامپیوترها اشاره دارد.

ابزارک‌های پوشیدنی قادرند تا بهتر در مورد موقعیت و فعالیت‌های انسان‌ها آگاهی یابند. برای مثال، یک تلفن هوشمند در مورد جهت قرارگیری خودش اطلاع دارد، اما یک عینک پوشیدنی هوشمند، در مورد جهت دید انسانی که آن را بر روی صورت دارد آگاهی دارد. ابزارک‌های پوشیدنی هم چنین بستر بسیار مناسبی برای پیشرفت و توسعه مفهوم دید افزوده[2] -مرز مشترک میان علم کامپیوتر و واقعیت افزوده- به شمار می‌آیند.

پژوهشهای رایج در علم بینایی ماشین غالبا به تصاویر ایستا یا ویدئوهای دریافتی از دوربینهای ایستا، اختصاص دارد. در سالهای اخیر با پیشرفتهای تکنولوژی، دوربینهای پوشیدنی و سبک با زاویه‌ی دید فرد منظر در اختیار عموم قرار گرفته است که حوزه جدیدی از پژوهشهای بینایی ماشین را در برابر پژوهشگران قرار داده است. عینک گوگل[1] سکوی رایانه ای قابل حمل و پوشیدنی شرکت Google، GoPro دوربینی با قابلیت اتصال بر روی کلاه ورزشکاران، Microsoft HoloLense عینک واقعیت مجازی شرکت Microsft، Recon Jet عینک هوشمند شرکت Recon و Smart EyeGlass عینک هوشمند شرکت Sony از جمله ابزارکهایی هستند که توانایی ذخیره ویدئوهای فرد منظر[3] را دارند (شکل ۱-۱). این ابزارک‌ها در آینده ای نه چندان دور نقش پر اهمیتی در سامانه‌های تعاملی

[1] Contextual Awareness
[2] Visual Augmentation
[3] Google Glass
[3] Egocentric

ایفا خواهند کرد. بینایی فرد منظر، انگاره‌ای است که میان انسان و ابزارک‌های پوشیدنی پیوند برقرار کرده است. با پردازش ویدئوهای فرد منظر، قادر به افزایش توانایی‌های دید انسان هستیم (شکل ۱-۲).

شکل ۱-۱: عینک ها و دوربینهای هوشمند. بالا: Google Glass، وسط راست: GoPro، وسط چپ: Sony SmartEyeGlass پایین راست: Microsoft HoloLense، پایین چپ: Recon Jet

به عنوان نخستین گام در حوزه‌ی فرد منظر، پژوهشگران کوشیده‌اند سامانه‌هایی را توسعه دهند که به شکل خودکار فعالیت انسان‌ها، روابط آنان با یکدیگر و نیز شناخت اشیاء و تعامل انسان با آنها را درک کنند. سامانه‌هایی که همانند انسان مشاهده و درک می‌کنند کاربردهای بسیاری در افزایش توانایی انسان‌ها دارند. به عنوان نمونه می‌توان از سامانه‌هایی که اشخاص نزدیک به فرد را تشخیص می‌دهند، موقعیت‌های خطرناک را درک می‌کنند و به فعالیت‌هایی همچون جراحی‌های پزشکی، ورزش، سرگرمی‌و غیره کمک می‌کنند، نام برد. در این میان

روانشناسان علاقه مند به پژوهش در حوزه توجه دیداری[1] نیز از این چارچوب نوین بهره برده اند؛ از این روست که مهندسان بینایی کامپیوتر برای کمک بیشتر و وسعت بخشیدن به این مهم، پژوهش‌های قابل توجهی را انجام داده اند.

پژوهش‌ها در قلمرو ویدئوهای فرد منظر را می‌توان به سه دسته تقسیم کرد: ۱. بازشناسی اشیاء ۲. تشخیص یا بازشناسی فعالیت و ۳. خلاصه سازی ویدئو.

شکل ۱-۲: نمونه‌هایی از قاب‌های یک ویدئو فرد منظر

[1] Visual Attention

از دیدگاه بینایی کامپیوتر، تصاویر دریافتی از دوربین‌های فرد منظر با چالش‌هایی نیز مواجه است. به عنوان نمونه، به سبب حرکت پیوسته دوربین، تصاویر غیرخطی و غیر قابل پیش‌بینی هستند. بنابراین اشیاء ممکن است به سرعت از دید دوربین محو شوند یا در معرض دید آن قرار گیرند. آشوب پس زمینه[1] دیگر چالشی است که دوربین‌های با حرکت زیاد و غیر قابل پیش بینی، ایجاد می‌کنند [۱]. در مواردی هم ممکن است با تاری ناشی از حرکت[2]، گذار سریع و انعکاس نور[3] مواجه شویم. از سوی دیگر، برخی از ویژگی‌های ویدئوهای فرد منظر می‌توانند از نظر پردازشی مفید باشند؛ مثلا از دید فرد، اشیاء، اشخاص یا چهره‌هایی که با آنها تعامل دارد، در مرکز تصویر قرار دارند (الگوواره[4] فرد منظر) و نسبت به تصاویر دریافتی از دوربینهای ایستا (سوم شخص) کمتر در معرض انسداد[5] قرار می‌گیرند. اما مهم ترین چالشی که یک پژوهشگر در حوزه بینایی فرد منظر با آن رو به رو است، سرعت اجرای الگوریتم‌های پردازش تصویر و ویدئو است و به همین سبب تلاش‌های بسیاری به منظور کاهش حجم محاسبات صورت گرفته است که یکی از آنها مفهوم ابرپیکسل[6] است.

ابرپیکسل

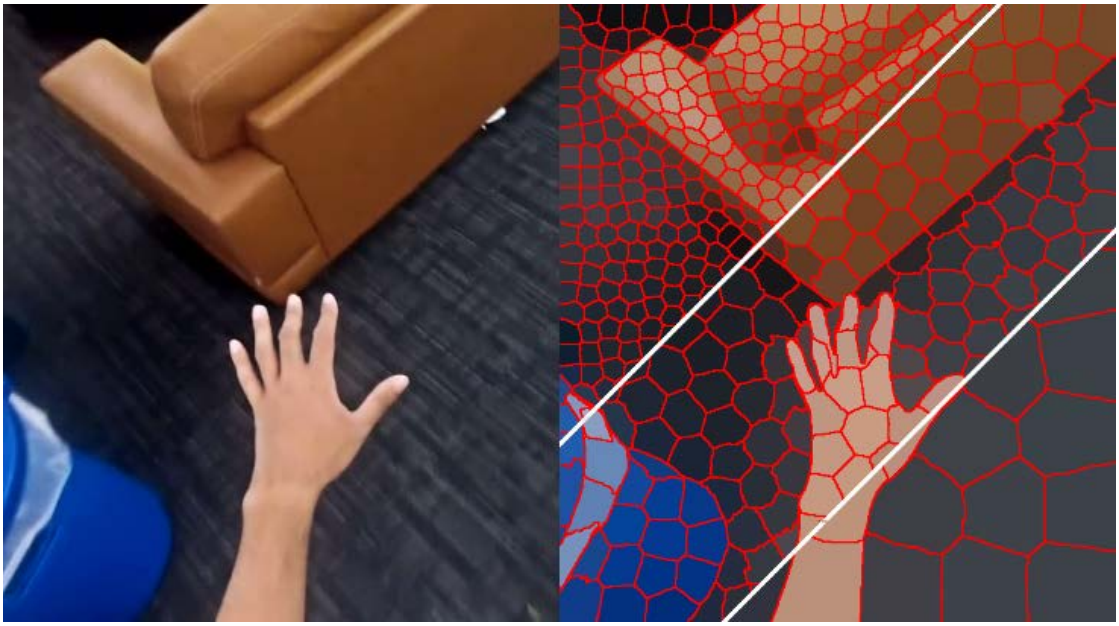
اصطلاح ابرپیکسل برای اولین بار توسط رِن و مالک [۲] در سال ۲۰۰۳ معرفی شد. مفهوم ابرپیکسل در پاسخ به دو چالش اساسی ایجاد شده است: ۱. پیکسل‌ها به تنهایی هویتی ندارند، بلکه تنها نتیجه نوعی گسسته سازی هستند ۲. تعداد پیکسل‌ها به عنوان عاملی محدود کننده در عملکرد الگوریتم‌ها به شمار می‌آید. در دهه‌ی اخیر، مفهوم ابرپیکسل تقریبا به تکامل رسیده است و اکنون در جامعه‌ی بینایی کامپیوتر به یک ابزار پردازشی تبدیل شده است. الگوریتم‌های ابرپیکسل به منظور ایجاد ساختاری با سطح بالاتر از یک تصویر یا یک قاب ویدیو، پیکسل‌های مشابه را به نواحی یا خوشه‌هایی معنادار تقسیم بندی می‌کنند. نواحی جدید، به جای ساختار متداول شبکه پیکسلی یک تصویر نشسته و افزونگی[7] تصویر را از بین می‌برند. این الگوریتم‌ها در تلاش برای فراهم آوردن خوشه‌بندی معنایی یک تصویر، وجه شباهت را با رنگ و فاصله فضایی پیکسل‌ها می‌سنجند. با توجه به نحوه‌ی عملکرد الگوریتم‌های ابرپیکسل، نمی‌توان آنها را از الگوریتم‌های بیش‌ناحیه‌بند[8] به روشنی متمایز کرد. با این همه در یک توافق کلی می‌توان ناحیه‌بندی با ابرپیکسل‌ها را نوعی بیش‌ناحیه‌بندی با نواحی هم‌شکل و هم اندازه که به

---

[1] Background Clutter
[2] Motion Blur
[3] Glare
[4] Paradigm
[5] Occlusion
[6] Super Pixel
[7] Redundancy
[8] Over-segmentation

صورت همگون بر روی صفحه تصویر پراکنده شده اند، تعریف کرد [۳]. بنابر این هر ناحیه‌بندی با ابرپیکسل‌ها، نوعی بیش-ناحیه‌بندی است؛ اما عکس این مسأله صادق نیست.

بسیاری از الگوریتم‌های پردازش تصویر در سطح پیکسل عمل می‌کنند، با این همه پردازش در سطوح بالاتر می‌تواند بهینه‌تر و با سرعت بیشتری انجام گیرد. به عنوان مثال در شکل ۱-۳ با استفاده از یک الگوریتم ابرپیکسل، تصویری با سطوح بالاتر در ۶۰، ۳۵۰ و ۹۰۰ ابرپیکسل مشخص شده است. بدین ترتیب توانسته ایم صدها هزار پیکسل را به چند صد ابرپیکسل کاهش دهیم؛ در عین حال مرزها و دیگر ویژگیهای اشیاء موجود در تصویر (مانند اطلاعات رنگ) را نیز حفظ می‌شود.



شکل ۱-۳: ایجاد ساختاری با سطوح بالاتر از یک تصویر به کمک الگوریتم ابرپیکسل

ابرپیکسل‌ها، با کاهش پیچیدگی محاسباتی برای پردازشهای تصویری پسین، ابزاری پیش پردازشی به شمار می‌آیند. از این رو به عنوان بلوکی کلیدی در ساختار بسیاری از الگوریتم‌های بینایی کامپیوتر مورد استفاده قرار می‌گیرند. ناحیه‌بندی [۴]، جداسازی پیش زمینه از پس زمینه [۵]، بازشناسی و محلی‌سازی اشیاء [۶] و [۷]، ردگیری[1] [۸]، [۹] و [۱۰]، اعلام تغییر صحنه [۱۱] جداسازی اشیاء چند طبقه ای [۱۲]، تخمین عمق [۱۳]، تشخیص سقوط [۱۴] و مدل‌سازی بدن انسان [۱۵] تنها برخی از کاربردهای ابرپیکسل در حوزه بینایی کامپیوتر هستند. هم چنین با ارائه یک سطح میانی از تصویر، الگوریتم‌های ابرپیکسل به منظور استخراج اطلاعات از تصاویر،

[1] Tracking

در تشخیص برجستگی دیداری[1] [۱۶] به کار برده می‌شوند. بر اساس تعریف [۱۷]، ناحیه‌بندی با ابرپیکسل‌ها نوعی بیش‌ناحیه‌بندی است که ویژگیهای پیکسل‌های سطح پایین‌تر (تصویر در سطح پیکسل) خود را به اندازه کافی حفظ می‌نماید. از این رو است که ابرپیکسل‌ها برای اهدافی چون محاسبه ویژگی نیز مورد استفاده قرار می‌گیرند.

شیوه‌های گوناگونی برای ایجاد ابرپیکسل‌ها ارائه شده است که هرکدام متناسب با کاربردی خاص بوده و عملکردی متفاوت دارد. به عنوان نمونه روشهای مبتنی بر گراف همچون [۱۸] که در آن گرافی از شبکه ابرپیکسل‌ها ساخته می‌شود، تبعیت بیشتری از مرزهای موجود در تصویر دارند. با این حال، این شیوه با مشکل الزام پس پردازش به منظور اطمینان از پیوستگی[2] خوشه‌ها مواجه است [۱۹]. برخی روشها، کانتورهای منظم تری از خوشه‌ها را ارائه می‌دهند [۲۰] و [۲۱]، در حالیکه شیوه‌های قدیمی‌تر خوشه‌هایی را با اشکال و اندازه‌های ناهمگون ایجاد می‌کنند [۲۲].

به طور کلی الگوریتم‌های پیشرفته و تکامل یافته ابرپیکسل‌ها را می‌توان به دو دسته تقسیم کرد: ۱. ابرپیکسل‌های مبتنی بر گراف ۲. ابرپیکسل‌های مبتنی بر گرادیان صعودی. در روشهای مبتنی بر گراف، هر پیکسل به عنوان یک گره در یک شبکه (یا گراف) توسط یالی به همسایه‌های خود متصل است. وزن این یالها متناسب با شباهت پیکسل‌ها در فضای رنگ تعریف شده، انتخاب می‌شود. آنگاه ابرپیکسل‌ها با آمیختن یالهای مشابه در یک همسایگی و در نهایت با کمینه کردن یک تابع هزینه از پیش تعریف شده [۲۱] و یا پیدا کردن کمینه درختهای پوشا[3] [۱۸] ایجاد می‌شوند. از سوی دیگر روشهای مبتنی بر گرادیان صعودی با یک خوشه‌بندی تقریبی اولیه آغاز می‌شوند. سپس خوشه‌ها در یک فرآیند تکراری به روز می‌شوند تا به یک همگرایی سراسری دست یابند [۲۳] و یا فرایند به روز رسانی به تعداد از پیش تعیین شده ای، بر روی خوشه‌ها تکرار شود [۲۴]. با آنکه انتخاب رهیافت ایده آل کاری دشوار به نظر می‌آید، اما می‌توانیم برای یک الگوریتم ابرپیکسل معیارهای زیر را مشخص کنیم:

۱. ابرپیکسل‌ها باید به خوبی از مرزهای موجود در تصویر تبعیت کنند.

۲. به منظور کاهش پیچیدگی محاسباتی در پردازشهای پسین، ابرپیکسل‌ها در عین استفاده آسان، بایستی سرعت مناسب و بهره وری موثری را از حافظه ارائه دهند.

۳. در کاربرد ناحیه‌بندی، ابرپیکسل‌ها باید قادر باشند تا سرعت و کیفیت مطلوبی ارائه بدهند.

---

[1] Visual Saliency
[2] Enforce Connectivity
[3] Spanning Trees

۴. ویژگیهایی چون اندازه، شکل و میزان پراکندگی ابرپیکسل‌ها بر روی صفحه تصویر مطلوب باشد.

با بررسی الگوریتم‌های ابرپیکسل، می‌توان ادعا کرد که هر الگوریتم توازنی میان این معیارها است.

با توجه به آنچه بیان شد، به منظور پردازش سریعتر ویدئوهای فرد منظر، نیاز به یک الگوریتم ابرپیکسل با عملکردی بهینه احساس می‌شود. این الگوریتم می‌بایست پایداری و دقت کافی را داشته باشد. برآنیم تا در این پایان نامه الگوریتم ابرپیکسلی را با روندی متفاوت و بهبود یافته با این ویژگیها و در چارچوب ویدئوهای فرد منظر ارائه و پیاده سازی کنیم. بنابراین تغییراتی را پیشنهاد خواهیم داد تا با ایجاد توازنی میان مهمترین معیارهای معرفی شده (سرعت اجرا و کیفیت) به هدف اصلی یعنی نزدیکتر شدن به عملکرد پردازش بلادرنگ، دست یابیم. در نهایت اثرات مثبت و منفی روش پیشنهادی را از نظر کیفیت، سرعت و پایداری با سایر پژوهش‌ها مقایسه خواهیم کرد.

۲. فصل دوم: پیشینه پژوهش

بازشناسی اشیاء در حوزه فرد منظر

یکی از اولین تحلیل‌های بازشناسی اشیاء در قلمرو فرد منظر توسط رِن و فیلیپوز [۲۵] انجام گرفته است. نویسندگان این مقاله بر این اساس که اشیائی که در دست کاربر هستند اطلاعات مهمی‌در مورد فعالیت کاربر می‌دهند به بازشناسی اشیاء پرداختند و البته در این راه با چالش‌ها و ویژگیهای ویدئوهای فرد منظر مواجه شدند. آنها دادگانی متشکل از ۴۲ ویدئو شامل اشیاء مختلفی که یک کاربر به طور روزانه با آنها تعامل دارد جمع آوری کردند. الگوریتم پیشنهادی آنها از نوعی سامانه بازشناسی SIFT که در [۲۶] معرفی شده است بهره گرفته است. در نهایت اشیاء با یک SVM طبقه بندی شده اند. در ابتدا نتیجه پژوهش، بازشناسی با نرخ ۱۲٪ بر روی تصاویری دارای چالشهای آشوب پس زمینه و انسداد دست گزارش شده است. سپس با آزمایش بر روی دادگانی جدید و بدون چالشهای یاد شده، به بازشناسی با نرخ ۶۳,۷٪ دست یافته است. پیرو این پژوهش ژایافنگ و همکارش [۱] به منظور بهبود دقت بازشناسی، رهیافتی مبتنی بر حرکت را توسعه دادند. آنها نظم حرکتی موجود در ویدئوهای فرد منظر را بررسی کرده و ادعا کردند: در زمان تعامل با اشیاء موجود در صحنه، اشیاء و دست در مرکز صفحه تصویر قرار دارند. حرکت آنها کم و غالبا افقی است. در الگوریتم پیشنهادی، ابتدا با میانگین گیری ماسکهای ناحیه بندی شده مرجع، توزیعی از موقعیت پیشین پیکسل‌ها را به دست آمده است. سپس با محاسبه جریان نوری[۱] در نواحی از تصویر که صرفا شامل پس زمینه هستند (بدون هیچ شئ یا دستی)، تخمینی تقریبی از جریان پس زمینه را محاسبه کرده و در مرحله بعد، از نشانه‌های زمانی که اطلاعات ماسک ناحیه‌بندی از قاب‌های قبل را در بر دارند، بهره گرفته است. در نهایت با استفاده از یک الگوریتم جریان نوری [۲۷] میان دو فریم و الگوریتم RANSAC[۲] بردارهای حرکتی، بر لایه‌هایی همگر[۳] تطبیق داده شده است. با استفاده از این بردارهای ویژگی حرکت و توزیع‌های پیشین، طبقه بندی آموزش یافته و خروجی‌های طبقه‌بند با الگوریتم برش گراف[۴] مجددا تقسیم بندی شده اند. نویسندگان به منظور آزمایش از همان دادگان [۲۵] استفاده کرده اند. نتایج آزمایش‌ها افزایش دقت بازشناسی را از ۱۲٪ به ۲۰٪ گزارش داده اند. نویسندگان، همچنین با استفاده از سامانه بازشناسی مبتنی بر HOG نتایج را به ۳۸٪ تا ۴۶٪ بهبود داده اند.

همانطور که پیش از این گفته شد، طبق الگوواره فرد منظر، اشیاء مورد توجه در این دید غالبا در مرکز صفحه تصویر و با مقیاس بزرگی حضور دارند. [۲۸] با تبعیت از این الگو، به بازشناسی اشیاء پرداخته است. فتحی و همکارانش در این پژوهش، ادعا کرده اند که تنها نام اشیاء در صحنه مشخص است و هیچ اطلاعاتی در مورد مکان

---

[1] Optical Flow
[2] Random sample consensus
[3] Affine
[4] Graph Cut

یا شکل اشیاء موجود نیست. بنابراین آنها از نوعی نظارت ضعیف در یادگیری سامانه بهره گرفته اند. در این پژوهش، هر قابی از ویدئو به ناحیه دست، شئ و پس زمینه تقسیم بندی و با بهره گیری از آموزش چند نمونه ای[1] به منظور تطبیق اشیاء بین قابهای ویدئو، اشیاء محلی سازی شده اند. سپس اشیاء به وسیله ورارسانی[2] و SVM طبقه بندی شده اند. در این مقاله، نویسندگان به نرخ بازشناسی ۱۰٪ (برای پاکت شکر) تا ۹۵٪ (قهوه) دست یافته اند.

با مهاجرت به فضای سه بعدی، [۲۹] با بررسی ارتباط میان شئ مورد نظر و مدل سه بعدی دستی که آن را گرفته است، به بازشناسی شئ پرداخته است. نویسندگان این مقاله بیان کردند که مدل سه بعدی دست از دید فرد منظر در بهبود دقت بازشناسی شئ تاثیر دارد. آنها نخست، با ایجاد خط لوله سه بعدی بینایی فرد منظر، به ناحیه بندی دست و تشکیل ابری از نقاط سه بعدی برای آن پرداختند. سپس ویژگیهای بصری (SIFT) از این ابر نقاط استخراج شده و در نهایت اشیاء با SVM طبقه بندی گردیده است. نتیجه این پژوهش بهبودی به اندازه ۲۲,۵٪ در نرخ بازشناسی گزارش کرده است.

بازشناسی فعالیت در حوزه فرد منظر

بسیاری از فعالیت‌هایی که از دید فرد قصد بازشناسی آنها را داریم، در تعامل ناظر با اشیاء مقابلش شناخته می‌شود. در حالیکه از دید سوم شخص و دوربین‌های ایستا، اشیاء به دلیل انسداد، دشوارتر بازشناسی شده و فعالیت انسان با مدل سازی حرکات بدن او بازشناسی می‌شود. در حوزه فرد منظر قادر به تشخیص فعالیتهایی چون گرفتن چاقو، باز کردن در یخچال، درست کردن چای و ... هستیم. به عنوان اولین قدم، اسپریگز و همکارانش [۳۰] به صورت با نظارت و بی نظارت به بازشناسی فعالیت در ویدئوهای فرد منظر پرداخته اند. آنان رهیافتهای خود را بر روی دادگانی متشکل از ویدئوهایی که در آنها اشخاص مشغول پخت انواع غذا هستند، آزمایش کرده اند. هر قاب از ویدئو برچسب فعالیت (مثلا هم زدن) خورده است. این پژوهش با تکیه بر الگوواره فرد منظر، به منظور بازشناسی فعالیت از ویژگی GIST [۳۱] بهره گرفته است. هم چنین برای کاهش ابعاد بردار ویژگی‌ها از PCA استفاده شده است. نتیجه پژوهش برای بازشناسی فعالیت «هم زدن» نرخ ۷۰٪ گزارش شده است. نویسندگان با آموزش شبکه HMM با خروجی مخلوط گوسی بر روی ویژگیهای GIST، نرخ بازشناسی ۹,۳۸٪ (نرخ تصادفی ۳٪) را در آموزش بدون نظارت گزارش کرده اند. در نهایت با اعمال مدل KNN بر روی هر قاب مورد آزمایش، برچسب قابی از مجموعه قابهای موجود در دادگان با کوچکترین فاصله اقلیدسی را به قاب مورد آزمایش تخصیص داده اند. نرخ بازشناسی در این راهبرد، ۴۸,۶۴٪ گزارش شده است.

---

[1] Multiple Instance Learning

[2] Transduction

فتحی و همکارانش [۳۲] در تلاش برای گسترش پژوهش خود، مدلی مبتنی بر گراف را توسعه داده‌اند تا از رابطه معنایی میان حرکات و اشیاء، فعالیت را بازشناسی کنند. آنها از همان دادگان [۲۸] شامل ویدئوهای آماده کردن انواع ساندویچ برای ارزیابی الگوریتم خود استفاده کرده‌اند. در این پژوهش از راهبردی مشابه بیشینه سازی امید ریاضی شرطی [۱] برای تشخیص حرکات و در نهایت تشخیص فعالیت بهره گرفته شده است. این پژوهش در نهایت به نرخ بازشناسی ۳۲,۴٪ (نرخ تصادفی ۱,۶٪) رسیده است. نویسندگان این اثر، پژوهش خود را با بهره گیری از جهت چشم [۲] در [۳۳] توسعه داده اند. آنان، با ترکیب ابزار تشخیص جهت چشم و دوربین‌های فرد منظر، سامانه ای جدید برای بازشناسی طراحی کرده اند. این راهبرد، الهامی از مطالعات فیزیولوژی [۳۴] است. در این مطالعات، ادعا می‌شود که مقدار قابل توجهی از دقت چشم در هنگام فعالیتی خاص، معطوف به مرکز همان فعالیت است. نویسندگان، بردار ویژگی استخراج شده از قاب‌های مورد آزمایش را مبتنی بر این الگوواره و ویژگیهای موجود در آثار قبل خود [۳۲] و همچنین ویژگیهای ظاهری تشکیل داده اند. نتیجه این پژوهش بهبود ۲۰٪ ای را نسبت به پژوهش پیشین [۳۲] گزارش کرده است.

فتحی و همکارانش باری دیگر در پژوهشی نوین [۳۵] با راهبردی متفاوت فعالیتهای تصویر برداری شده از دید فرد را بازشناسی کرده اند. اساس این بازشناسی بر تغییر وضعیت اشیاء، زمانی که فرد با آنها تعامل دارد، بنا نهاده شده است. به همین منظور برای آزمایش، برچسب فعالیتها با وضعیت قاب آغازین، قاب نهایی، برچسب فعالیت و مجموعه ای از صفات برای توصیف شئ مورد تعامل مشخص شده است. در این اثر، با تمرکز بر اشیاء موجود در پس زمینه، نواحی از قاب که قبل و بعد از فعالیت تغییر می‌کنند، خوشه‌بندی شده اند. سپس این نواحی با ویژگیهایی چون رنگ، بافت و شکل معرفی شده و SVMای خطی با آنها آموزش داده شده است. نتیجه این پژوهش با نرخ بازشناسی ۳۹,۷٪ (بر پایه ۶۱ طبقه از فعالیت) گزارش شده است.

ریو و ماتئیس [۳۶] جزو اولین پژوهشگرانی هستند که در حوزه بینایی فرد منظر به بازشناسی تعاملات دیگر انسانها با فرد پرداخته اند. تعاملاتی چون رفتارهای دوستانه (مانند دست دادن) یا غیر دوستانه (ضربه زدن یا پرتاب شئ) در این پژوهش مورد بازشناسی قرار گرفته است. بر اساس اینکه این نوع تعاملات باعث حرکات شدیدی در دید فرد شده اند، نویسندگان شناسه‌های حرکتی محلی و سراسری را در سامانه تعریف کرده اند. این شناسه‌های حرکت، توسط الگوریتم k-means خوشه بندی شده تا مجموعه‌ای از کلمات بصری ایجاد شود. هر فعالیت در ویدئوهای فرد منظر به صورت هیستوگرامی از این کلمات تعریف شده و در نهایت SVMای بر پایه آنها آموزش یافته است. نتیجه پژوهش، نرخ بازشناسی ۸۹,۶٪ (بر اساس ۷ نوع فعالیت مختلف) گزارش کرده است.

---

[1] Expectation-Conditional Maximization

[2] Gaze

[۳۷] به بررسی کاربرد بینایی فرد منظر در ورزش پرداخته است. کیتانی و همکارانش در این اثر، راهبردی بی‌نظارت و سریع برای برچسب‌گذاری ویدئوها توسعه داده اند تا ورزشکاران بخشهای خاصی از آنها را بدون جستجوی دستی مرور و موقعیت یابی کنند. در این پژوهش نیز همچون [۳۶] بر پایه اصل تحرک زیاد دید فرد در ویدئوهای ورزشی، هیستوگرامی‌از قسمتهای متحرک تصویر، به منظور توصیف حرکات در یک ویدئوی ورزشی خاص ایجاد شده است. در مرحله بعد تبدیل DFT بر روی دامنه این هیستوگرام اعمال شده تا هیستوگرام فرکانس به دست آید. در نهایت مدل مخلوط دیریکله [۳۸] اعمال شده تا دفتر کدی[۱] از حرکات و سپس طبقه‌بندی حرکات فرد منظر شناسایی شود. نتایج آزمایش این الگوریتم از نظر عملکرد با مقیاس F (مقیاس دقت و بازشناسی) گزارش شده است. نتایج این پژوهش بر روی ویدئوهای ورزشی موجود در تارنمای YouTube معادل F = 0.6 گزارش شده است.

خلاصه سازی ویدئو در حوزه فرد منظر

ایده واقعه برداری از زندگی[۲] یا خلاصه سازی ویدئو دیگر کاربرد جذاب و مورد توجه در حوزه فرد منظر به شمار می‌آید. منظور از واقع برداری این است که فرد، با یک دوربین اول شخص پوشیدنی رویدادهای یک روز خود را ثبت می‌کند. ایده خلاصه برداری از این ویدئو نیز با آرمان ایجاد سامانه‌هایی برای یاری به افراد کم حافظه طراحی شده است [۳۹]. دوهرتی و همکارانش [۴۰] از اولین پژوهشگرانی بودند که با استفاده از Microsoft SenseCam راهکار انتخاب قاب کلیدی را در حوزه خلاصه سازی ویدئو فرد منظر بررسی کردند. در این پژوهش نویسندگان ابتدا مجموعه تصویرهای گرفته شده را به رویدادهای مختلف تقسیم بندی کرده اند. این دسته بندی بر اساس شباهت رنگ و توصیفگرهای لبه صورت گرفته است. آنان، شیوه‌های مختلفی را برای استخراج قاب کلیدی از هر رویداد آزمایش کرده اند. ۱۳۰۰۰ قاب کلیدی استخراج شده از این روش‌ها توسط کاربران امتیاز دهی شده است. نویسندگان انتخاب قاب کلیدی در حوزه فرد منظر را با چالشی روبه‌رو دیده اند. بدین ترتیب که انتخاب قاب کلیدی در رویدادی که حرکت بسیار زیادی متوجه آن است (مانند راه رفتن) بسیار دشوار است.

[۴۱] با هدف خلاصه سازی ویدئو شیوه ای مبتنی بر علائم برجستگی[۳] مانند اشیاء و افرادی که ناظر با آنها تعامل دارد، ابداع کرده است. به عبارتی در این پژوهش، هر قاب توسط روش کمینه برش‌های پارامتریک محدود [۴۲] به نواحی مختلفی نقسیم‌بندی شده است. سپس وایازشی[۴] آموزش داده شده تا به هر ناحیه در قاب امتیاز دهی

[1] Codebook
[2] Life-Logging
[3] Importance Cues
[4] Regressor

شود. این امتیاز بر پایه ویژگیهایی چون تعامل (فاصله اقلیدسی مرکز ناحیه تا مرکز دست)، جهت چشم (فاصله اقلیدسی تا مرکز)، فرکانس (میزان تکرار ناحیه در مجموعه قابها)، ظاهر شئ گونه (مبتنی بر تابع امتیاز دهی [۴۲])، حرکت شئ گونه و احتمال قرار گیری چهره در ناحیه (به شیوه ویولا جونز [۴۳]) است. در نهایت ویدئو بر پایه تفاوت هیستوگرام‌ها به رویدادهای مختلف خوشه‌بندی می‌شود. سپس هر رویداد با قابی که بالاترین امتیاز وایازش را دارد نمایش است. نتایج پژوهش بر روی دادگان ویدئو که با تُرک مکانیکی آمازون[1] برچسب گذاری شده است، جلب نظر ۶۸,۷۵٪ از کاربران را نشان داده است. لو و گرامن [۴۴] با توسعه این پژوهش به نتایج بهتری دست یافتند. آنان، با پیشنهاد روشی مبتنی بر واقعه[2] بیان کردند که می‌توان مقیاسی تعریف کرد که پیوستگی وقایع در یک ویدئو را ثبت کند. آنها هم چنین شیوه ناحیه‌بندی مخصوصی برای ویدئوهای فرد منظر طراحی کردند تا رویدادها خوشه بندی شوند. در این پژوهش کوشیده شده تا نوع حرکت ناظر شناسایی شود. به عبارتی میان حرکت از نقطه‌ای به نقطه دیگر و حرکت سر ناظر تمایز ایجاد شده است. نویسندگان SVMای را آموزش داده تا بر مبنای ویژگیهای اجزاء متحرک تصویر و تاری ناشی از حرکت [۴۵]، نوع حرکت ناظر شناسایی شود. نویسندگان با این روش رویدادهایی به اندازه ۱۵ فریم را استخراج کردند. هر رویداد با اشیاء تشخیص داده شده بیان شده است. سپس هر رویداد به عنوان گره‌ای در یک زنجیره در نظر گرفته شده و زنجیره‌ای متشکل از k قاب به صورت داستان وار ایجاد شده است. نویسندگان نتایج الگوریتم را بر روی دادگان خود و دادگان [۴۶] آزمایش کرده و نظر ۸۷٪ از کاربران را جذب کردند.

ابرپیکسل

الگوریتم‌های ایجاد ابرپیکسل‌ها عموما به دو دسته ۱. ابرپیکسل‌های مبتنی بر گراف و ۲. ابرپیکسل‌های مبتنی بر گرادیان صعودی تقسیم بندی می‌شوند. در هر دو دسته، ورودی تصویری به صورت I با عرض W و ارتفاع H تعریف می‌شود. بنابراین تعداد پیکسل‌ها در تصویر $N = W \times H$ محاسبه می‌شود. در این بخش مروری بر روشهای موجود در هر دو دسته می‌پردازیم.

ابرپیکسل‌های مبتنی بر گراف

الگوریتم‌های مبتنی بر گراف، هر پیکسل در صفحه تصویر را به عنوان گره‌ای در یک گراف در نظر می‌گیرند. هر پیکسل توسط یال‌هایی به پیکسل‌های موجود در همسایگی ۴ تایی (یا ۸ تایی) خود متصل است. وزن یال‌های

[1] Amazon's Mechanical Turk
[2] Story-driven

متصل میان هر دو گره، متناسب با شباهت میان پیکسل‌های همسایه در نظر گرفته می‌شود. در نهایت ابرپیکسل‌ها با کمینه کردن تابع هزینه‌ای تعریف شده بر روی گراف، ایجاد می‌شوند.

NC00[1] الگوریتم پیشنهادی موسوم به برشهای هنجار یافته [۲۱] در روندی تکراری، گراف تشکیل شده از پیکسل‌های یک تصویر را بر اساس کانتور و بافت، ناحیه‌بندی کرده است. سپس تابع هزینه تعریف شده بر روی لبه‌های موجود در هر ناحیه را کمینه ساخته است. این شیوه، ابرپیکسل‌هایی منظم و از نظر بصری قابل قبول ایجاد کرده است. ابرپیکسل‌هایی خروجی از این الگوریتم تبعیت مناسبی از مرزها نشان نداده و زمان اجرای بسیار کندی دارد؛ گرچه تلاشهایی در سرعت بخشیدن به این الگوریتم وجود دارد [۴۷]. تابع پیچیدگی این روش $O\left(N^{\frac{3}{2}}\right)$ تعریف شده است.

GS04[2] نویسندگان [۱۸] راهکاری متفاوت از تشکیل ابرپیکسل‌های مبتنی بر گراف ارائه داده اند. آنها نوعی الگوریتم خوشه‌یابی سلسله مراتبی را برای پیکسل‌های یک گراف پیشنهاد داده اند. در این خوشه‌بندی، هر ابرپیکسل توسط الگوریتم درختهای پوشای کمینه به وجود آمده است. به عبارت دیگر یک ابرپیکسل در گراف پیکسل‌ها، درختی است که میان درختهای پوشای آن گراف مجموع وزن یالهای آن، کمترین مقدار ممکن باشد. در عمل، این شیوه تبعیت مناسبی از مرزها دارد اما ابرپیکسل‌ها شکل و اندازه نامنسجمی دارند. تابع پیچیدگی این روش $O(N \log N)$ تعریف شده است. زمان اجرای این الگوریتم سریع است، اما نظارتی بر تعداد و تراکم ابرپیکسل‌ها موجود نیست.

SL08[3] در سال ۲۰۰۸ مور و همکارانش شیوه ای را پیشنهاد دادند که در آن با یافتن مسیرها یا شکافهای[4] بهینه و تقسیم صفحه تصویر به نواحی عمودی یا افقی کوچکتر، ابرپیکسل‌هایی با نظم شبکه‌ای ایجاد شده است [۸]. ، در این روش موسوم به ابرپیکسل‌های شبکه‌ای، مسیر بهینه با الهام از روش شکاف کاوی [۴۸] یافت شده است. بنا به ادعای نویسندگان تابع پیچیدگی این روش به صورت $O(N^{\frac{3}{2}} \log N)$ گزارش شده اما در این تابع، زمان اجرای یافتن شکاف‌ها، که تاثیر زیادی بر سرعت و کیفیت خروجی دارد، محاسبه نشده است.

---

[1] Normalized Cuts [2000]

[2] Graph-Based Segmentation [2004]

[3] Superpixel Lattices [2008]

[4] شکاف (seam) در تصویر به دو صورت شکاف عمودی و شکاف افقی تعریف می‌شود. منظور از شکاف عمودی، مسیری از پیکسل‌های پیوسته از بالا تا پایین تصویر است به نحوی که در هر ردیف، یک پیکسل انتخاب شده باشد. شکاف افقی نیز مسیری پیوسته از پیکسل‌های یک تصویر از چپ به راست است که در هر ستون فقط یک پیکسل انتخاب شده باشد.

شکل ۲-۱: نمونه ای از اجرای الگوریتم SL08

CIS10[1] [۴۹] مشابه روش تولید بافت در [۵۰] از نوعی بهینه سازی سراسری بهره گرفته است. در این روش ابتدا سطح تصویر با مربع‌هایی که با یکدیگر همپوشانی دارند، پوشانده شده است. هر مربع نماینده یک ابرپیکسل است به نحوی که هر پیکسل می‌تواند به یکی از این مربع‌ها تخصیص یابد. در نهایت ابرپیکسل‌ها با کمینه کردن تابع هزینه ای ایجاد شده اند. این الگوریتم موسوم به ابرپیکسل‌هایی با شدت روشنایی ثابت، تنها بر رو تصاویر با سطوح خاکستری قابل پیاده سازی است.

ERS11[2] الگوریتم [۵۱] تابع هدفی بر نرخ آنتروپی مدل ولگشت[3] گراف ایجاد شده تعریف کرده است. این تابع به نحوی بهینه شده که تعداد مؤلفه‌های همبند[4] در گراف پیکسل‌ها کمتر یا مساوی تعداد ابرپیکسل‌های مطلوب (k) باشد. با آغاز از یک خوشه‌بندی اولیه به نحوی که هر پیکسل به تنهایی یک خوشه (ابرپیکسل) باشد، الگوریتم با اضافه کردن یال ابرپیکسل‌ها را ادغام کرده و ابرپیکسل‌هایی بزرگتر تولید کرده است. سرعت اجرای این الگوریتم اما، مطلوب نیست.

PB11[5] در الگوریتم مبتنی بر بهینه‌سازی تابع شبه بولین [۵۲] ابتدا صفحه تصویر با نوارهای افقی و عمودی پوشانده شده است به گونه‌ای که هر پیکسل با دو نوار عمودی و دو نوار افقی پوشانده شود. بدین ترتیب اگر تنها نوارهای افقی در نظر گرفته شود، هر پیکسل برچسب ۰ یا ۱ دارد. تخصیص برچسب با استفاده از الگوریتم بیشینه

---

[1] Constant Intensity Superpixels [2010]
[2] Entropy Rate Superpixels [2011]
[3] Random Walk
[4] Connected Component
[5] Superpixels via Pseudo Boolean Optimization [2011]

جریان[1] صورت گرفته است. برچسبهای نوارهای افقی و نوارهای عمودی در ترکیب با یکدیگر ابرپیکسل‌ها را تشکیل داده اند. تبعیت از مرزها در این شیوه کیفیت مطلوبی ندارد.

CRS13[2] در شیوه تخمین کانتور که نوعی راهبرد آماری است، فرض شده که تصویر نتیجه چندین فرآیند تصادفی است [۵۳]. در واقع مقدار پیکسل x_n در کانال c نتیجه فرآیندی تصادفی متناسب با ابرپیکسل متناظرش به شمار آمده است. در نهایت تابع هزینه‌ای به روش الگوریتم تپه نوردی[3] بهینه سازی شده تا ابرپیکسل‌ها تولید شوند. سرعت اجرای ان الگوریتم مطلوب نیست و نظارتی بر پارامترهای ابرپیکسل وجود ندارد.

TPS12[4] [۵۴] روشی موسوم به ابرپیکسل‌هایی با توپولوژی حفظ شده ارائه داده است. در این روش نویسندگان بیان می‌کنند که ابرپیکسل‌ها را می‌توان با موقعیتی مشخص شده بر روی شبکه‌ای با توپولوژی تعیین شده ایجاد کرد. بنابراین پس از تعیین توپولوژی شبکه، موقعیت ابرپیکسل‌ها به ناحیه‌ای از تصویر با بیشترین لبه انتقال یافته است. این موقعیتها گرافی بدون جهت بر اساس موقعیتهای نسبی تشکیل داده اند. سپس موقعیتهای همسایه در شبکه با کوتاهترین مسیر در گرافی وزن دار (وزن بین هر دو پیکسل با احتمال یال آن پیکسل‌ها نسبت عکس دارد) به یکدیگر متصل شده اند. کوتاهترین مسیر توسط الگوریتم دایکسترا[5] [۵۵] جستجو شده است.

ابرپیکسل‌های مبتنی بر گرادیان صعودی

در این رویکرد، ابتدا خوشه‌بندی اولیه ای از پیکسل‌ها ایجاد می‌شود. سپس روش‌های مبتنی بر گرادیان صعودی، در فرایندی تکراری خوشه‌ها را تا زمان رسیدن به معیاری تعریف شده برای همگرایی به روز رسانی می‌کنند.

MS02[6] در الگوریتم انتقال میانگین [۵۶] با هدف یافتن بیشینه محلی یک تابع چگالی، روندی تکراری طراحی شده است تا در فضای رنگ یا فضای شدت نور از یک تصویر مُدها استخراج شوند. پیکسل‌هایی که به مُد یکسان همگرا هستند، یک ابرپیکسل را تشکیل داده اند. این شیوه ابرپیکسل‌هایی نامنظم با اندازه‌های ناهمگون تولید کرده است. تابع پیچیدگی، $O(N^2)$ تعریف شده است و زمان اجرای الگوریتم بسیار کند است. هم چنین نظارتی بر تعداد، اندازه یا تراکم پیکسل‌ها به کاربر داده نمی‌شود.

---

[1] Max-flow Algorithm
[2] Contour Relaxed Superpixels [2013]
[3] Hill-climbing Algorithm
[4] Topology Preserved Superpixels [2012]
[5] Dijkstra's Algorithm
[6] Mean Shift [2002]

QS08[1] الگوریتمی‌موسوم به انتقال سریع، همانند روش قبل از نوعی روند تکراری برای جستجوی مُد در صفحه تصویر بهره گرفته است [۲۳]. در این روش، ناحیه‌بندی با نوعی روند انتقال medoid آغاز شده است. سپس هر نقطه در فضای ویژگی‌ها به منظور افزایش تخمین چگالی پارزن[2] به نزدیکترین همسایه انتقال یافته است. اگر چه این شیوه تبعیت خوبی از مرزهای موجود در تصویر دارد اما با تابع پیچیدگی $O(dN^2)$ (d متغیر ثابتی کوچک [۲۳]) الگوریتمی‌کند به شمار می‌آید. این روش نیز نظارتی بر تعداد و اندازه ابرپیکسل‌ها به کاربر ارائه نداده است.

TP09[3] راهبرد توربوپیکسل به صورت تصاعدی خوشه‌های مجموعه‌ای از بذرها را گسترش داده است [۲۰]. این شیوه مبتنی بر گرادیان‌های تصویر، ابرپیکسل‌ها را به صورت منظم بر صفحه تصویر گسترانده است. بر خلاف سایر روشها، توربوپیکسل ابرپیکسل‌هایی با اندازه و تراکم همگونی را ارائه داده است. گرچه نویسندگان این پژوهش ادعا کرده اند که تابع پیچیدگی این الگوریتم $O(N)$ است، اما در عمل، الگوریتم زمان اجرای بسیار کندی دارد و تبعیت مناسبی از مرزها به دست نمی‌آید.

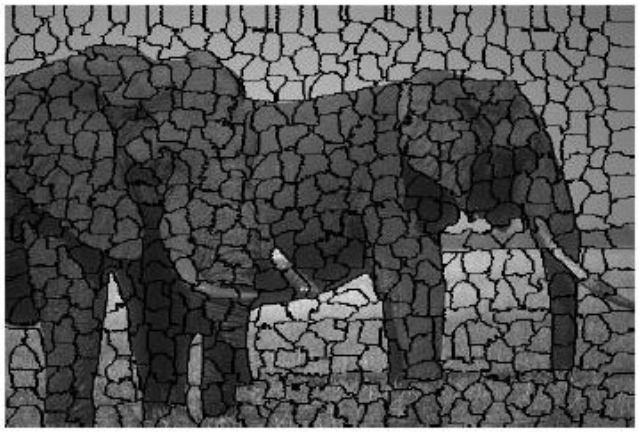
SLIC10[4] الگوریتم خوشه یابی خطی ساده [۲۴] به عنوان اساس و الهامی برای بسیاری از پژوهش‌های پس از خود چون [۵۷]، [۵۸]، [۵۹] و [۵۳] نقش آفرینی کرده است. این الگوریتم از جهت سادگی استفاده، قابل توجه بسیاری از پژوهش‌های مبتنی بر ابرپیکسل است. این الگوریتم با پیاده‌سازی نوعی خوشه‌یابی k-means به صورت محلی، به تعداد k ابرپیکسل ایجاد کرده است. این الگوریتم در فصل سوم به طور کامل بررسی می‌شود.

SEEDS12[5] الگوریتم ارائه شده در [۵۹] مبتنی بر نمونه برداری انرژی-ران، ابرپیکسل‌ها را ایجاد کرده است. با یک خوشه‌بندی اولیه، الگوریتم با جابه‌جایی بلوک‌های (مربع شکل) پیکسلی با بلوک‌های همسایه و به صورت مرحله ای، ابرپیکسل‌ها را تولید کرده است. هدف از جابه‌جایی، ترکیب بلوکهایی با هیستوگرام رنگ یکسان در نظر گرفته شده است. با آزمایش بر روی تصاویر با سطح تفکیک 720×1280 عملکرد این الگوریتم، کند گزارش شده است.

SPPS07[6] روهکول و انگل به طور مشابه در الگوریتم [۶۰] با آغاز از یک خوشه‌بندی اولیه، پیکسل‌ها را به طور مستقل در ابرپیکسل‌های همسایه، جابه‌جا کرده اند. با این تفاوت که شبکه اولیه بر خلاف [۵۹] متشکل از بلوک‌هایی شش ضلعی است. معیار جابه‌جایی، شباهت در رنگ پیکسل‌ها در نظر گرفته شده است.

---

[1] Quick Shift [2008]
[2] Parzen
[3] Turbo-Pixels [2009]
[4] Simple Linear Iterative Clustering [2010]
[5] Superpixels Extracted via Energy-Driven Sampling [2012]
[6] Superpixels using Pairwise Pixel Similarities [2007]



شکل ۲-۲: نمونه ای از اجرای الگوریتم SPPS07

SSP09[1] با هدف ایجاد روشی بهینه، دراکر و مک کورمیک کوشیده اند تا با استفاده از برنامه نویسی پویا، مسیرِ با کم‌ترین هزینه را بر نقشه لبه‌ها یا نقشه ویژگیها بیابند [۶۱]. این روش را در همین فصل در بخش «ابرپیکسل در چارچوب ویدئوهای فرد منظر» با جزئیات بیشتر بررسی خواهیم کرد.

SSS11[2] نویسندگان [۶۲] در روشی موسوم به ابرپیکسل‌های حساس به ساختار، سعی داشته‌اند تا الگوریتمی مبتنی بر الگوریتم لوید[3] و فاصله ژئودزیک[4] طراحی کنند. بدین ترتیب ابرپیکسل‌ها تبعیت بیشتری از تصویر دارند؛ به نحوی که در بخشهایی از تصویر با بافت زیاد، ابرپیکسل‌ها کوچکتر هستند، در حالیکه در نواحی همگون تصویر شاهد ابرپیکسل‌های بزرگتری هستیم.

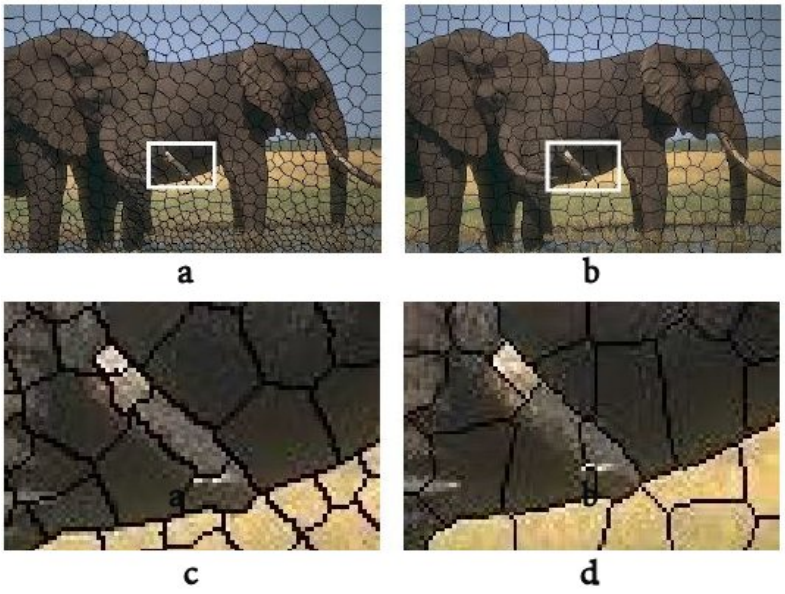
HS11[5] [۶۳] در الگوریتمی مبتنی بر ولگشت مارکوف، سعی در تولید ابرپیکسل‌هایی با اندازه و شکل همگون دارد. در این پژوهش با الهام از خوشه‌یابی مارکوف و هرس گراف نوعی راهبرد، موسوم به هرس متراکم ابداع شده است. بدین ترتیب ساختار محلی ذاتی تصویر استخراج شده و ابرپیکسل‌های خروجی با اندازه و تراکم همگون ایجاد شده اند.

IER15[6] در نوین ترین الگوریتم ابرپیکسل مبتنی بر گرادیان صعودی، نویسندگان [۶۴] شیوه‌ای موسوم به اصلاح تکراری لبه‌ها را ابداع کرده اند. در این رویکرد، ابتدا صفحه تصویر با شبکه‌ای منظم از ابر پیکسل‌های اولیه

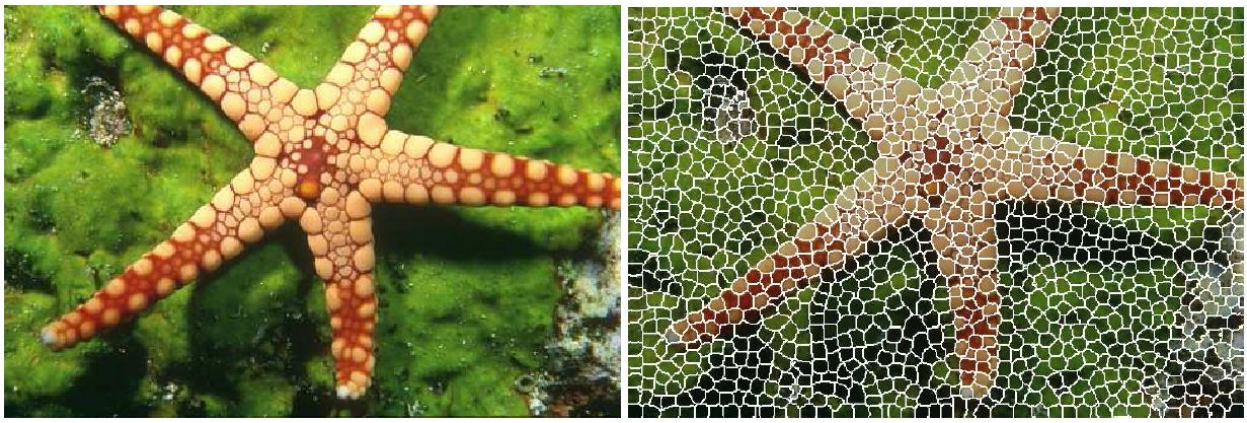
---

[1] Superpixels from Strong Paths [2009]
[2] Structure Sensitive Superpixels [2011]
[3] Lloyd Algorithm
[4] Geodesic
[5] Homogeneous Superpixels [2011]
[6] Iterative Edge Refinement [2015]

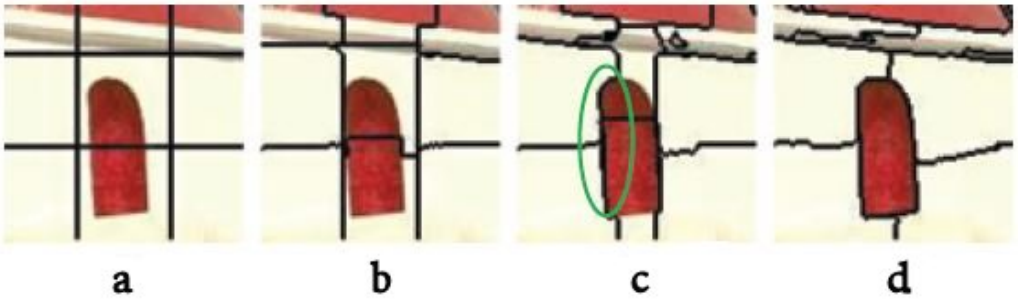
پوشانده شده و سپس در روندی تکراری، پیکسل‌های حاشیه هر بلوک به روزرسانی شده اند. با تکرار این اصلاح، مرزهای بلوکهای اولیه جابه جا شده تا به تبعیت از مرزهای واقعی موجود در تصویر دست یافته شود. نتیجه این پژوهش با آزمایش بر روی دادگانی متشکل از تصاویر با سطح تفکیک 321 × 481 پیکسل، زمان اجرای ۴۸ میلی‌ثانیه را گزارش داده است. نویسندگان این اثر ادعا کرده اند که از نظر عملکرد، نتیجه بهتری از SLIC10 ارائه داده اند. روند اصلاح خوشه این الگوریتم در شکل ۲-۵ مشخص است.



شکل ۲-۳: نمونه ای از اجرای الگوریتم SSS11 در مقایسه با الگوریتم TP09. a و c: الگوریتم SSS11، b و d: الگوریتم TP09



شکل ۲-۴: نمونه ای از اجرای الگوریتم HS11



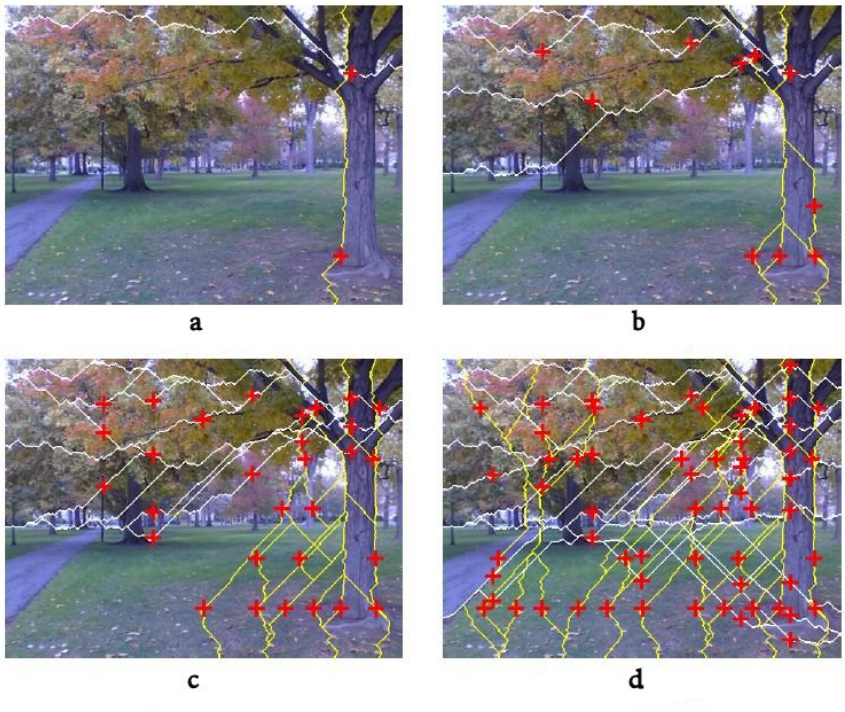
شکل ۲-۵: نمونه ای از روند اصلاح خوشه الگوریتم IER15. a: خوشه‌بندی اولیه، b: بعد از ۴ دوره، c: بعد از ۱۲ دوره و d: خروجی نهایی

در فصل سوم به بررسی بیشتر و مقایسه کیفی و کمی‌الگوریتم‌های معرفی شده خواهیم پرداخت.

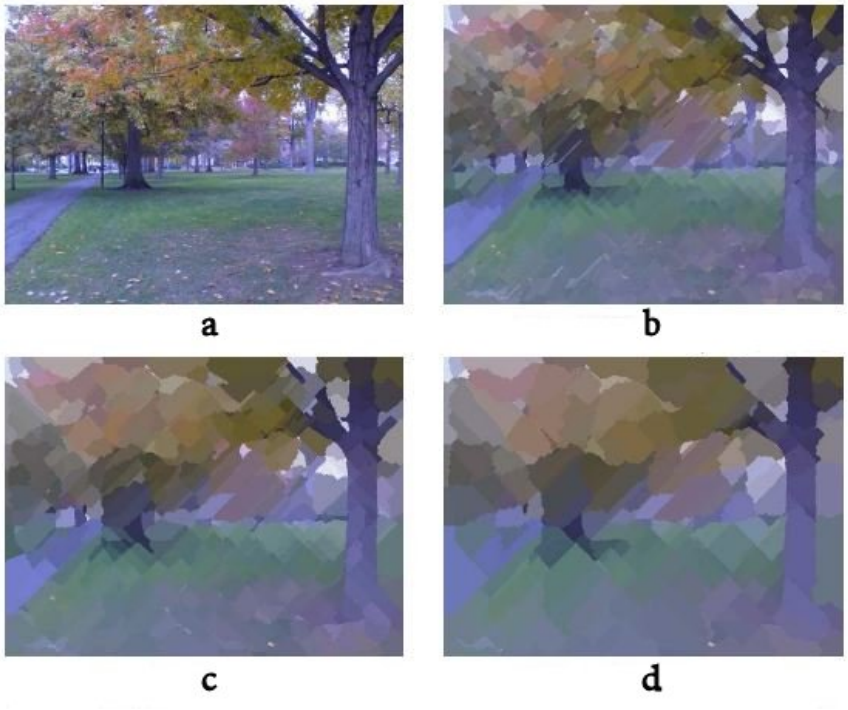
ابرپیکسل و ویدئو

گفتنی است به رغم پژوهشهای فراوان در این موضوع، کاربرد الگوریتم‌های ابرپیکسل در تحلیل ویدئو همچنان در آغاز راه خود قرار دارد. برای پیاده سازی یک ابرپیکسل سریعتر توسط [۶۱] کوششهایی انجام گرفته است؛ اما در این کوششها باز هم دقت، قربانی سرعت شده است. در این الگوریتم که به اختصار آن را SSP09 نامیدیم، قابهای ویدئو، به شکافهای عمودی و افقی تقسیم بندی شده است. این شکافها (مسیرها) از مرزهای موجود در تصویر تبعیت می‌کنند. با بهره گیری از نوعی برنامه نویسی پویا، بهینه ترین (قویترین) مسیرها یافته می‌شوند. البته در این الگوریتم از موازی ماندن شکافهای موازی، اطمینان حاصل نشده است. از این رو الگوریتم، عملکردی سریع دارد اما نظارتی مستقیم بر تعداد و یا اندازه ابرپیکسل‌ها وجود ندارد. هم چنین عدم نظارت بر روی جهت مسیرها، باعث ایجاد الگوهای ناخواسته در تصویر شده است. الگوریتم SSP09 پارامتری موسوم به g یا شکاف بذر[1] دارد. این پارامتر معرف کم ترین فاصله مجاز افقی میان بذر جدید و هر مسیر عمودی از قبل انتخاب شده است. شکل ۲-۸ روند اجرای الگوریتم را در یک قاب ویدئو و شکل ۲-۹ تاثیر مقیاس g را بر ابرپیکسل‌ها نمایش می‌دهند.

[1] Seed Gap



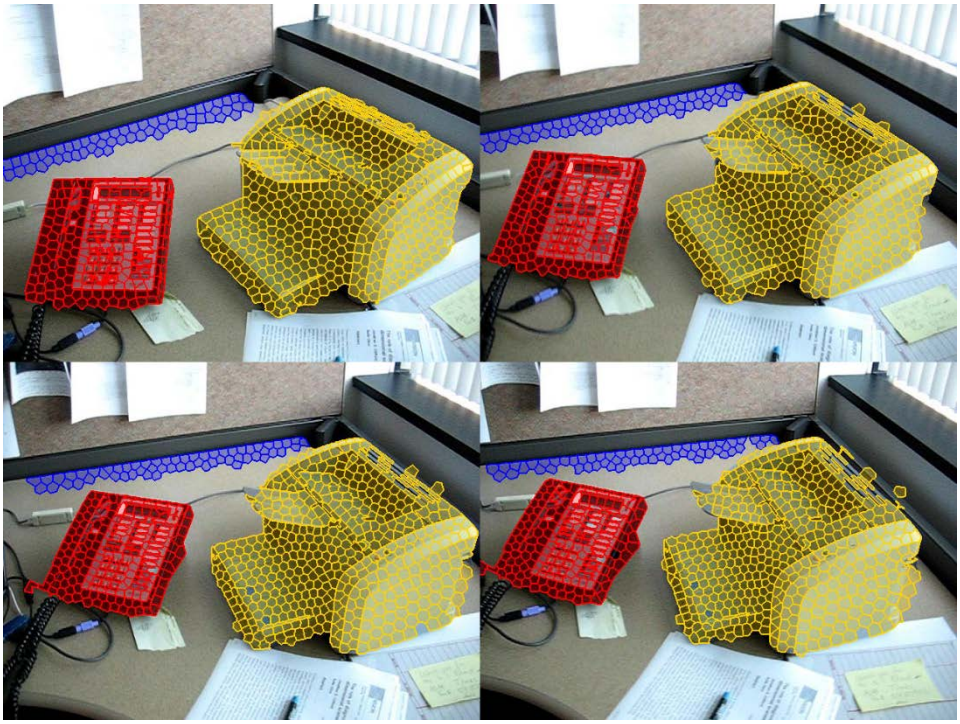
شکل ۲-۶: نمونه ای از اجرای الگوریتم SSP09. a: ۱ مسیر، b: ۵ مسیر، c: ۱۵ مسیر و d: ۳۰ مسیر



شکل ۲-۷: نمونه ای از اجرای الگوریتم بر روی قابی از یک ویدئو با تنظیم پارامتر g. a: قاب اصلی، b: g=10 و تشکیل ۴۹۴۳ ابرپیکسل، c: g=20 و تشکیل ۱۲۵۸ ابرپیکسل و d: g=30 و تشکیل ۵۱۰ ابرپیکسل

نتایج الگوریتم SSP09 بر روی یک ویدئو با ابعاد 144 × 176 پیکسل، زمان اجرای ۴۰ میلی ثانیه به ازای هر قاب را گزارش داده است. این زمان اجرا برای چارچوب ویدئوهای فرد منظر (معمولا به ابعاد 720 × 1280) مناسب به نظر نمی‌آید.

[۱۷] نیز در تلاش برای ایجاد پلی ارتباطی میان ویدئو و ابرپیکسل از ویژگی ابرپیکسل‌های زمانی[1] بهره گرفته است. نویسنگان این پژوهش ادعا کرده اند که اجرای الگوریتم ابرپیکسلی به طور مستقل بر روی هر فریم در طول زمان ابرپیکسل‌هایی نامتناسب با آنچه واقعا در صحنه وجود دارد، تولید می‌کند. هر ابرپیکسل زمانی، مجموعه ای محلی از پیکسل‌های یک ویدئو در فضای تصویر است که قسمت مشخصی از از یک شئ را در طول ویدئو دنبال می‌کند. شکل ۲-۸ اجرای این الگوریتم را در چهار قاب متوالی نمایش می‌دهد. همانطور که دیده می‌شود سه شئ در تصویر، در طول ویدئو با سه مجموعه از ابرپیکسل‌های زمانی دنبال شده‌اند. البته این الگوریتم به سبب نیاز به دهها ثانیه در پردازش یک فریم و فاصله زیاد با عملکرد بلادرنگ، در عین موثر بودن، برای چارچوب پردازشی ویدئوهای فرد منظر مناسب نیست.



شکل ۲-۸: نمونه ای از اجرای الگوریتم ابرپیکسل‌های زمانی [۱۷]

[1] Temporal Superpixels (TSPs)

در این میان [۶۵] با الهام از رویکرد شکاف کاوی، سعی در پیاده‌سازی الگوریتم ابرپیکسل برای کاربردهای بلادرنگ دارد. با این حال نتایج این پژوهش بر روی دادگان تصاویر ایستا BSDS500 آزمایش شده است. تصاویر موجود در این دادگان سطح تفکیکی به مقدار 321×481 پیکسل دارند. نویسندگان این پژوهش ادعا کرده اند عملکرد الگوریتمشان در ویدئوهایی با همین سطح تفکیک زمان اجرایی معادل ۱۵ قاب بر ثانیه دارد. بنابراین در چارچوب ویدئوهای فرد منظر این عملکرد نیز مناسب به نظر نمی‌رسد.

ابرپیکسل در چارچوب ویدئوهای فرد منظر

ابرپیکسل‌ها در چهار چوب فرد منظر تنها در [۶۶] به منظور ارزیابی عملکرد و در [۶۷] به منظور جداسازی دست در صحنه و ردگیری آن به کار گرفته شده اند. در این دو پژوهش از الگوریتم ابرپیکسل خوشه یابی خطی ساده [۲۴] که در بخشهای قبل معرفی گردید، بهره گرفته شده است.

[۶۶] در تلاش برای بهینه کردن الگوریتم خوشه‌یابی خطی ساده بر روی ویدئوهای فرد منظر، با الهام از فیلتر بیزین[1] و روشهای کدگذاری ویدئو، به جای مقدار دهی اولیه به صورت قاب به قاب، اطلاعات یک قاب را به قاب بعد انتقال داده است تا همگرایی، سریعتر اتفاق بیافتد. با این حال عملکرد در چهار چوب فرد منظر همچنان با پردازش بلادرنگ فاصله دارد. رویکرد این پژوهش را در فصل سوم به طور کامل بررسی خواهیم کرد.

[۶۸] اگر چه الگوریتمی بلادرنگ را از SLIC ارائه داده، اما این الگوریتم برای برنامه نویسی مبتنی بر واحد پردازنده گرافیکی[2] به NVIDIA CUDA یا چارچوب اختصاصی شرکت NVIDIA نیاز دارد. این چهار چوب برنامه نویسی می‌بایست بر روی واحدهای پردازنده گرافیکی شرکت NVIDA پیاده سازی شود؛ این نوع پیاده سازی در حال حاضر به ابزارک‌های پوشیدنی غیر قابل انتقال است.

[1] Bayesian

[2] Graphical Processing Unit (GPU) Programming

۳. فصل سوم: روش پیشنهادی

در این بخش ابتدا به مقایسه و بررسی الگوریتم‌های ابرپیکسل معرفی شده در فصل دوم خواهیم پرداخت. سپس الگوریتم ابرپیکسلی متناسب با آنچه در چارچوب پردازشی بینایی فرد منظر نیاز است، برگزیده می‌شود. معیار انتخاب و جزئیات الگوریتم بیان شده و نحوه پیاده سازی آن بر روی ویدئوهای فرد منظر بررسی خواهد شد. نکات مثبت و کمبودهای این پیاده‌سازی نیز شرح داده می‌شود. در مرحله بعد به منظور توسعه الگوریتمی با معیارها و ویژگی‌های معرفی شده در بخش مقدمه، رویکردهای متفاوتی از الگوریتم برگزیده شده پیشنهاد داده می‌شود.

همانطور که پیش از این در مقدمه بیان شد، برای یک الگوریتم ابرپیکسل، معیارهای زیر مطلوب است:

۱. ابرپیکسل‌ها باید به خوبی از مرزهای موجود در تصویر تبعیت کنند.

۲. به منظور کاهش پیچیدگی محاسباتی در پردازشهای پسین، ابرپیکسل‌ها در عین استفاده آسان، بایستی سرعت مناسب و بهره‌وری موثری را از حافظه ارائه دهند. ابرپیکسل‌هایی که پارامترهای تنظیم شونده زیادی دارند منجر به عملکرد نامناسب می‌شوند.

۳. در کاربرد ناحیه‌بندی، ابرپیکسل‌ها باید قادر باشند تا سرعت و کیفیت مطلوبی ارائه بدهند.

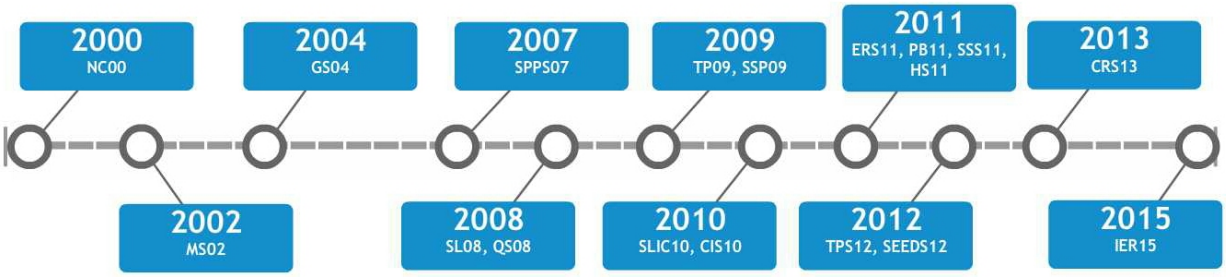
۴. ویژگیهایی چون اندازه، شکل و میزان پراکندگی ابرپیکسل‌ها بر روی صفحه تصویر مطلوب باشد. ابرپیکسل‌های متراکم و منسجم به دلیل اندازه محدود و تعداد همسایه‌های کم‌تر، گرافی تفسیر بَردارتر ارائه می‌دهند. از این رو، ویژگیهای محلی مرتبط‌تری از تصاویر تجزیه شده استخراج می‌شود. باید توجه داشت که تراکم در توازن با تبعیت از مرز قرار دارد. بنابراین قابلیت نظارت بر چنین سازشی می‌تواند مفید ظاهر شود.

۵. نظارت بر تعداد ابرپیکسل‌ها موجود باشد. کم تر الگوریتمی یافت می‌شود که چنین نظارتی را در اختیار کاربر قرار دهد.

با توجه به آنچه گفته شد، بایستی در جستجوی ابرپیکسلی با توازن مناسب از این معیارها بود. در ادامه، ابتدا تصاویر تجزیه شده توسط الگوریتم‌های معرفی شده در فصل دوم را از نظر کیفی با یکدیگر مقایسه کرده و سپس با معرفی چند مقیاس، به ارزیابی کمی‌آنها خواهیم پرداخت.

مقایسه الگوریتم‌های ابرپیکسلی

در این بخش الگوریتم‌های ابرپیکسلی را از نظر کیفیت بصری و عملکرد کمی‌مقایسه خواهیم کرد. نخست در شکل ۳-۱ گاه‌شماری از الگوریتم‌های توسعه داده شده مشخص شده است.

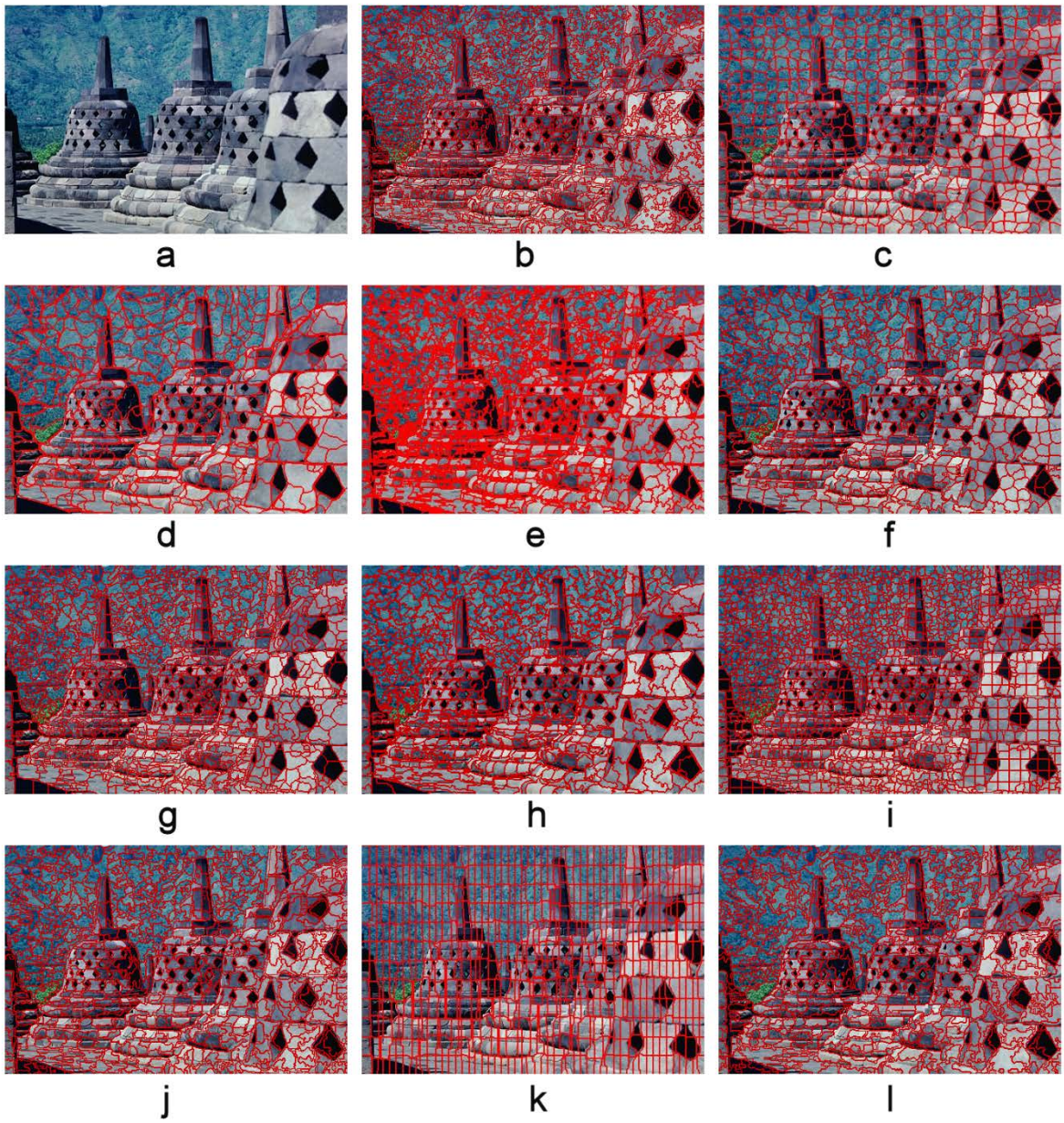


شکل ۳-۱: گاه‌شمار الگوریتم‌های ابرپیکسل توسعه یافته

شماری از الگوریتم‌های ابرپیکسل معرفی شده، بر روی دادگان ناحیه‌بندی و کانتوریابی دانشگاه برکلی (BSDS500 [۶۹]) توسط [۷۰] و به منظور ارزیابی کیفی و کمّی، پیاده‌سازی شده است. تصاویر موجود در این دادگان ، سطح تفکیکی معادل با 321 × 481 پیکسل دارند. [۲۴] نیز با پیاده‌سازی برخی از ابرپیکسل‌ها بر روی چند تصویر نمونه، تصاویر تجزیه شده را به شکل کیفی و کمی‌ارزیابی کرده است. الگوریتم‌های SL08، SPPS07، SSP09، SSS11، HS11 و IER15 به دلیل موجود نبودن کد الگوریتم، در این مقایسه لحاظ نشده اند. نتایج این الگوریتم‌ها در فصل دوم به طور مستقل نمایش داده شده است.

مقایسه کیفی

تصاویر تجزیه شده توسط ابرپیکسل‌های معرفی شده در فصل دوم به منظور ارزیابی کیفی در شکل ۳-۲ [۷۰] و شکل ۳-۳ [۲۴] نمایش داده شده‌اند. با بررسی معیارهای کلی معرفی شده در ابتدای این فصل، مشاهده می‌شود که الگوریتم‌های SLIC10 و TP09 نتیجه مطلوبی را دارا هستند. ابرپیکسل‌های تولید شده توسط این دو روش شبکه همگون تری را ارائه داده اند. اندازه و شکل ابرپیکسل‌ها نیز همگونتر است.



شکل ۳-۲: نتایج بصری برخی الگوریتم‌های ابرپیکسل(سری اول [۷۰]) . a: تصویر اصلی، b: الگوریتم GS04 [۱۸]، c: الگوریتم TP09 [۲۰]، d: الگوریتم NC00 [۲۱]، e: الگوریتم QS08 [۲۳]، f: الگوریتم SLIC10 [۲۴]، g: الگوریتم CIS10 [۴۹]، h: الگوریتم ERS11 [۵۱]، i: الگوریتم PB11 [۵۲]، j: الگوریتم CRS11 [۵۳]، k: الگوریتم TPS12 [۵۴]، l: الگوریتم SEEDS12 [۵۹]

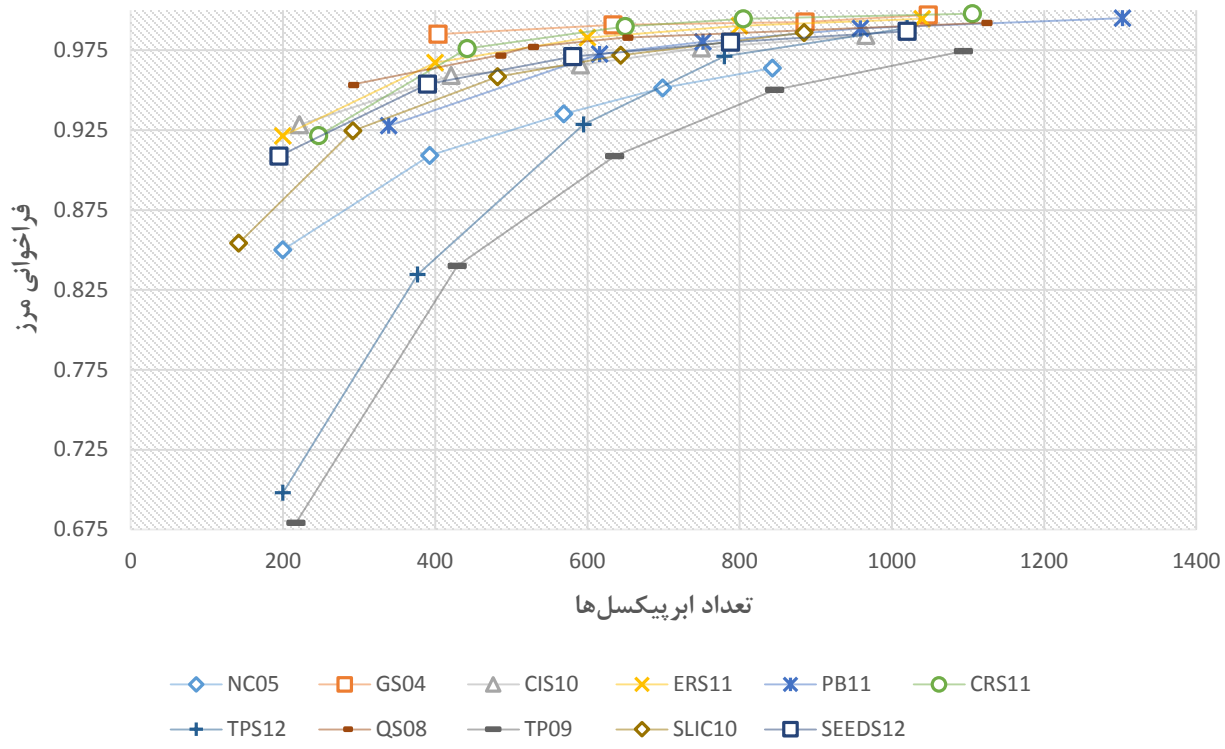
شکل ۳-۳: نتایج بصری برخی الگوریتم‌های ابرپیکسل(سری دوم [۲۴]). ستون a: الگوریتم GS04، ستون b: الگوریتم NC00، ستون c: الگوریتم TP09، ستون d: الگوریتم QS08، ستون e: الگوریتم SLIC10

مقایسه کمّی

فراخوانی مرز[1] این مقیاس، بیان می‌کند که چه کسری از صفر حقیقی لبه‌های تصویر، در محدوده‌ای به اندازه دو پیکسل از مرزهای ابرپیکسل‌ها قرار می‌گیرد. بنابراین، فراخوانی مرز، مقیاسی به منظور ارزیابی تبعیت از مرزهای تصویر به شمار می‌آید. فراخوانی مرز بالا، به این معنی است که در تولید ابرپیکسل‌ها میزان کمی از مرزهای واقعی تصویر از بین رفته است. این مقیاس عددی است در بازه [۱ ۰] که برای الگوریتم‌های متفاوت به ازای تعداد ابرپیکسل‌های مختلف در نمودار ۳-۱ نمایان است.

[1] Boundary Recall

نمودار ۳-۱: مقایسه معیار فراخوانی مرز در الگوریتم‌های ابرپیکسل



خطای فروناحیه‌بندی[1] مقیاسی دیگر برای ارزیابی تبعیت از مرزهای تصویر، خطای فروناحیه‌بندی نامیده می‌شود. فرض می‌شود که ناحیه‌ای از صفر حقیقی ناحیه‌بندی شده تصویر به صورت g_i باشد و مجموعه‌ای از ابرپیکسل‌های مورد نیاز برای پوشاندن این ناحیه به صورت $s_j|s_j \cap g_i$ تعریف شود. آنگاه مقیاس خطای فروناحیه‌بندی، تعداد پیکسل‌هایی از s_j که از مرز g_i خارج شده‌اند را محاسبه می‌کند. اگر $|.|$ اندازه یک ناحیه در مقیاس پیکسل، M تعداد نواحی صفر حقیقی تصویر و B کمینه پیکسل‌های s_j که g_i را می‌پوشانند باشد، خطای فرو ناحیه‌بندی به صورت رابطه ۳-۱ تعریف می‌شود.

$$U = \frac{1}{N}\left[\sum_{i=1}^{M}\left(\sum_{(s_j|s_j \cap g_i > B)} |s_j|\right) - N\right]$$

رابطه ۳-۱

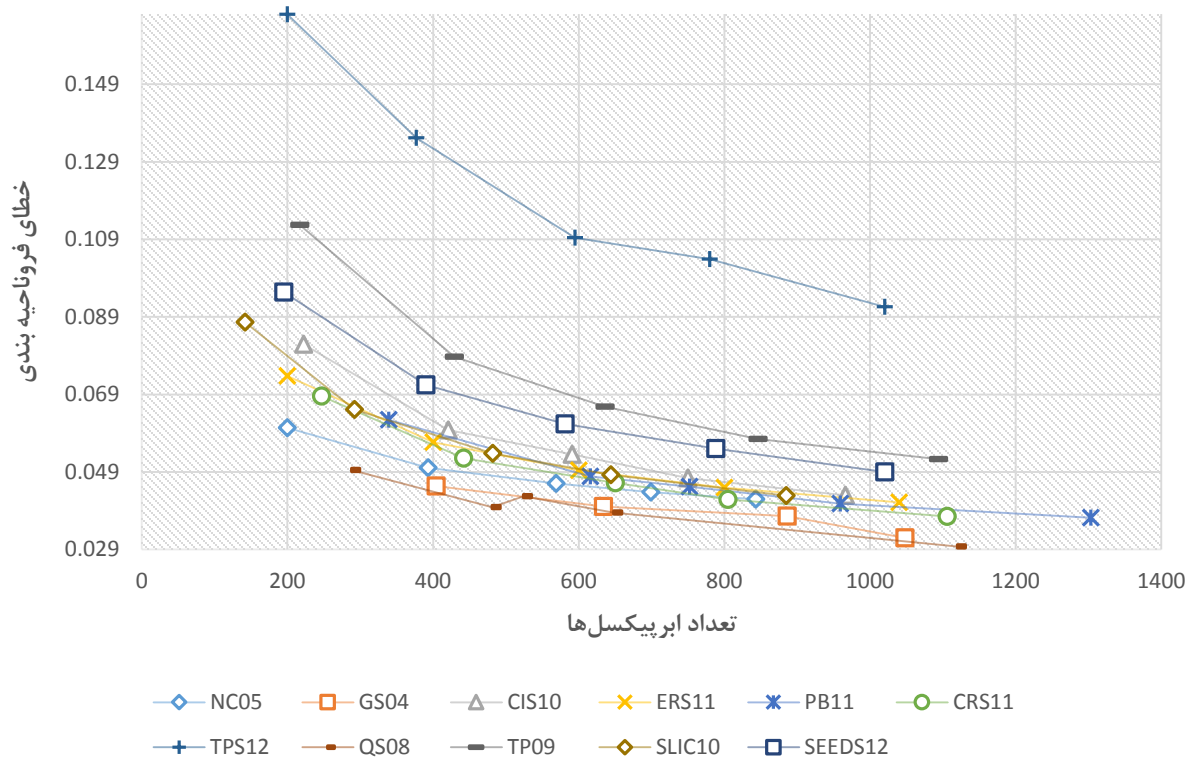
[1] Under-Segmentation Error

[۷۰] به منظور حذف پارامترهای قابل تنظیم در محاسبه مقیاس خطای فروناحیه‌بندی، از رابطه معرفی شده در [۷۱] بهره می‌گیرد (رابطه ۳-۲). مقایسه خطای فروناحیه‌بندی برای الگوریتم‌های مختلف در نمودار ۳-۲ مشخص است.

$$U = \frac{1}{N}\sum_{i=1}^{M}\sum_{(s_j|s_j \cap g_i \neq \emptyset)} \min\left\{\left|s_j \bigcap g_i\right|, |s_j - g_i|\right\}$$

رابطه ۳-۲

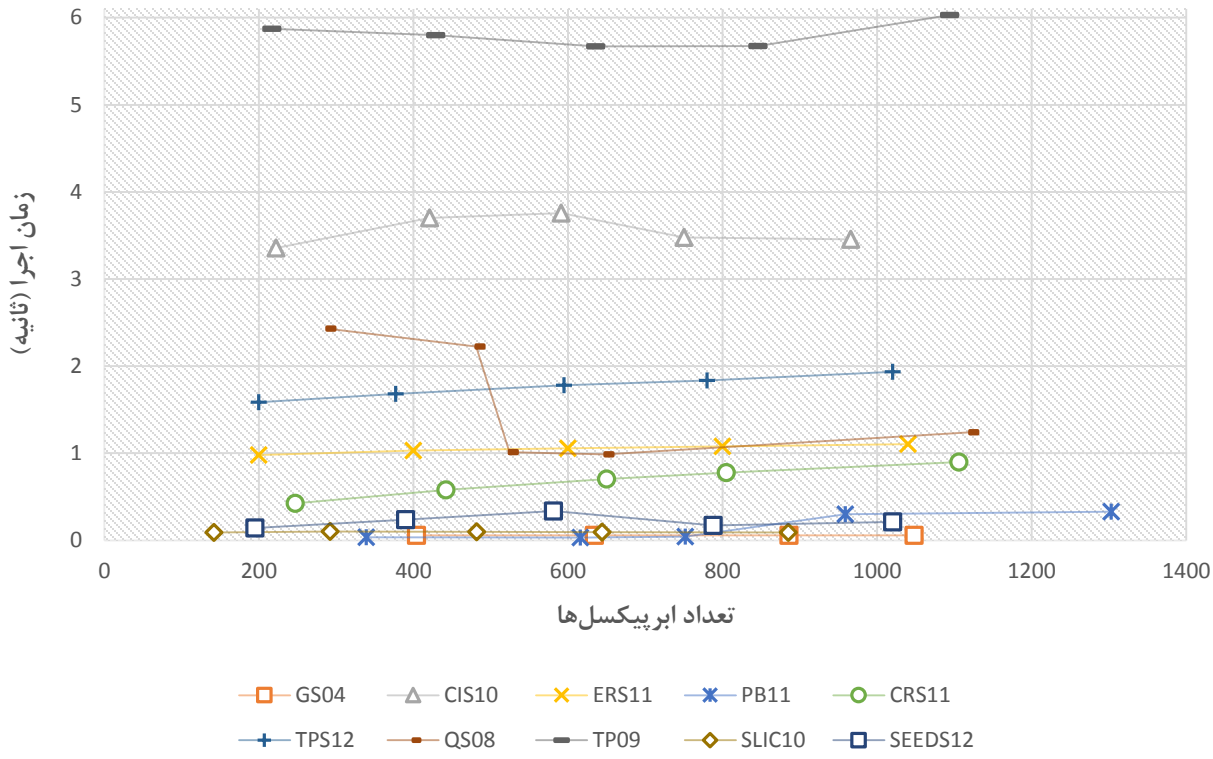
نمودار ۳-۲: مقایسه معیار خطای فروناحیه‌بندی در الگوریتم‌های ابرپیکسل



زمان اجرا به منظور سرعت بخشیدن به سایر الگوریتم‌ها، ابرپیکسل‌ها به جای ساختار متداول شبکه پیکسلی، اطلاعات موجود در یک تصویر را توصیف می‌کنند. از این رو در وهله اول تولید بهینه ابرپیکسل‌ها حائز اهمیت است. زمان اجرای لازم برای تصاویر دادگان BSDS500 به ازای تعداد ابرپیکسل‌های مختلف محاسبه و در نمودار ۳-۳ نمایان است. الگوریتم‌های SLIC10 و PB11 با تابع پیچیدگی $O(N)$ و الگوریتم GS04 با تابع پیچیدگی

$O(N \log N)$ در زمره الگوریتم‌های پرسرعت قرار دارند. جالب توجه است که با افزایش تعداد ابرپیکسل‌ها تفاوت سرعت در این الگوریتم‌ها بسیار ناچیر است.

نمودار ۳-۳: مقایسه زمان اجرای الگوریتم‌های ابرپیکسل



بهره وری از حافظه بهره وری موثر از حافظه نیز از مقیاس‌های مهم ابرپیکسل‌ها به شمار می‌آید. یک ابرپیکسل بایستی قادر به پردازش موثر تصاویر با سطح تفکیک بالا نیز باشد. بنا به ادعای [۲۴] SLIC10 با نیاز به ذخیره‌سازی تنها N ممیز شناور(ذخیره فاصله پیکسل‌ها تا نزدیکترین خوشه)، موثرترین شیوه در بهره وری از حافظه است. سایر رویکردها نیازمند حافظه بالاتری هستند. به عنوان مثال GS04 به 5N ممیز شناور به منظور ذخیره وزن یالها و آستانه‌ها در یک اتصال ۴ همسایگی (9N ممیز شناور در اتصال ۸ همسایگی) نیاز دارد. هم چنین الگوریتم NC00 با ایجاد خطای «حافظه ناموجود» در ناحیه‌بندی تصاویری با سطح تفکیک 1536 × 2048 ناموفق عمل کرده است.

نتیجه گیری

با مقایسه بصری مشاهده شد که الگوریتم TP09 و الگوریتم SLIC10 ابرپیکسل‌های همگونی را ارائه دادند. از سوی دیگر هر دو شیوه موفق به ایجاد ساختاری منسجم از ابرپیکسل‌ها شده‌اند (معیار ۴). الگوریتم TP09 تنها یک

پارامتر تنظیم شونده دارد و ابرپیکسل‌های متراکم و همگونی ایجاد کرده است؛ اما با مقایسه کمّی نتیجه گرفته می‌شود که از نظر زمان اجرا و تبعیت از مرزهای موجود (مقیاسهای فراخوانی مرز و خطای فروناحیه‌بندی) در تصویر، توازن از دست رفته است. بنابراین SLIC10 پیروز این میدان است.

از دیدگاه تبعیت از مرزهای موجود در تصویر (مقیاس‌های فراخوانی مرز و خطای فرو ناحیه بندی)، الگوریتم‌های ERS11، CRS11، GS04 و QS08 با تفاوتی ناچیز بهتر از SLIC10 ظاهر شده اند؛ اما از دیدگاه معیار همگونی شبکه ابرپیکسل‌ها بازنده اند. QS08 و GS04 برای اجرا به تنظیم چندین پارامتر نیاز دارند. از نظر زمان اجرا نیز SLIC10 رقابت تنگاتنگی با GS04 و PB11 دارد. PB11 تنها در این معیار برنده است و توازن را در سایر معیارها به هم می‌زند. GS04 نیز از نظر بصری نتیجه مطلوبی ندارد. از این رو، بار دیگر SLIC10 پیروز است. با توجه به ارزیابی کیفی و کمّی الگوریتم‌های معرفی شده در فصل دوم، الگوریتم SLIC10 پاسخی است برای آنچه در میان الگوریتم‌های مختلف جستجو کردیم. از سوی دیگر با استناد به آنچه در فصل دوم بیان شد، تنها الگوریتمی‌که در چارچوب ویدئوهای فرد منظر از آن بهره گرفته شده است، الگوریتم SLIC10 است. در نهایت نیز این الگوریتم برای این پایان نامه برگزیده شده است.

SLIC10 شیوه ای نوین در ایجاد ابرپیکسل‌ها به شمار می‌آید. این شیوه نسبت به سایر روشها از نظر سرعت، بهره وری از حافظه، تبعیت کامل از مرزهای موجود در تصویر و عملکرد ناحیه‌بندی، برتری دارد. الگوریتم SLIC10 اقتباسی از خوشه‌بندی k-means است. در ادامه این الگوریتم را به تفصیل شرح خواهیم داد.

الگوریتم SLIC

الگوریتم خوشه‌یابی تکراری خطی ساده یا SLIC[1] شیوه ای مبتنی بر گرادیان صعودی است. به رغم سادگی، SLIC همانند روشهای پیشین و در مواردی بهتر ازآنها، از مرزهای موجود در تصویر تبعیت می‌کند. بهره‌وری خوب از حافظه و سرعت بیشتر آن نسبت به سایر روشها، SLIC را در زمره الگوریتم‌های پیشرفته با عملکرد مناسب قرار می‌دهد. علاوه بر اینها، SLIC به آسانی قابل استفاده است و در تراکم و تعداد ابرپیکسل‌ها انعطاف بسیار بالایی دارد. این الگوریتم با الهام از روش خوشه‌بندی k-means مشابه اسلوب [۱۳] ابرپیکسل‌ها را تولید می‌کند، اما با دو تفاوت زیر:

۱. با محدود کردن فضای جستجو، متناسب با اندازه ابرپیکسل دلخواه، تعداد محاسبات فاصله در الگوریتم بسیار کاهش یافته است. از این رو پیچیدگی محاسباتی به صورت خطی و به اندازه تعداد پیکسل‌های موجود در تصویر (N) در آمده است. این پیچیدگی مستقل از تعداد ابرپیکسل‌ها (k) است.

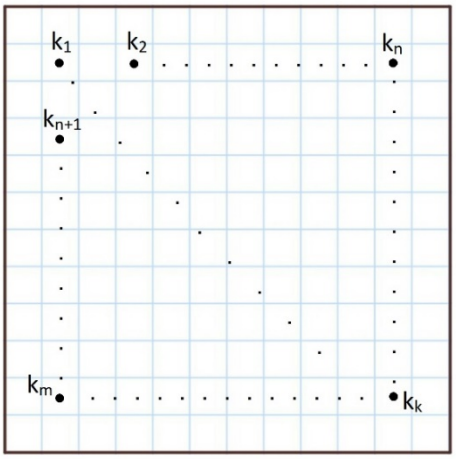
۲. اعمال ضریب وزن دار از فاصله (شباهت) پیکسل‌ها از نظر رنگ و فاصله فضایی سبب شده است تا نظارت بیشتری بر اندازه و تراکم پیکسل‌ها وجود داشته باشد.

SLIC، مشابه رهیافتی که در [۱۳] به عنوان مرحله پیش‌پردازش در تخمین عمق پیاده‌سازی شده است، عمل می‌کند.

الگوریتم

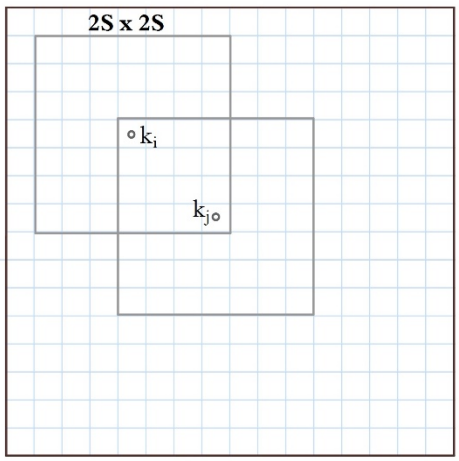
SLIC به آسانی قابل فهم و استفاده است. به طور پیش فرض تنها پارامتر الگوریتم، تعداد ابرپیکسل‌های تقریبا هم اندازه یا k است. پردازش بر روی تصاویر، در محیط رنگی CIELAB انجام می‌شود. روند خوشه‌بندی با مرحله مقداردهی اولیه آغاز می‌شود. در این مرحله، k مرکزِ خوشه‌های اولیه به صورت یک بُردار ویژگی ۵ بعدی $C_k = [l_i \ a_i \ b_i \ x_i \ y_i]^T$ با فواصلی مساوی و منظم به اندازه S نمونه برداری می‌شوند (شکل ۳-۴). به منظور تولید ابرپیکسل‌هایی با اندازه یکسان، گام نمونه‌برداری به صورت $S = \sqrt{N/k}$ انتخاب می‌شود. سپس این مراکز اولیه به نقطه ای با کم ترین گرادیان در یک همسایگی ۳ در ۳، انتقال می‌یابند. این اصلاح مکانی برای جلوگیری از قرار گرفتن مرکز یک خوشه بر روی مرزهای تصویر است. بدین ترتیب از بذرپاشی ابرپیکسلی به مرکز یک پیکسل نویزی نیز جلوگیری شده است.

[1] Simple Linear Iterative Clustering



شکل ۳-۴: شبکه مقداردهی اولیه برای مراکز خوشه‌ها

مرحله بعد، مرحله تخصیص نام دارد. در این مرحله، هر پیکسل i به نزدیکترین مرکزی که در محدوده جستجوی آن قرار گرفته است، تخصیص می‌یابد. این روندِ تخصیص، کلید سرعت الگوریتم SLIC است. به عبارتی، با محدود کردن اندازه ناحیه جستجو، تعداد محاسبه فاصله‌ها، در هر دوره به مقدار قابل توجهی کاهش می‌یابد. در حالیکه در الگوریتم رایج k-means فاصله هر پیکسل با تمام مراکز خوشه‌ها محاسبه و مقایسه می‌شود. از آنجا که انتظار می‌رود وسعت رشد یک ابرپیکسل به اندازه $S \times S$ باشد، SLIC به جستجوی پیکسل مشابهی در همسایگی $2S \times 2S$ حول هر مرکز خوشه می‌پردازد (شکل ۳-۵).



شکل ۳-۵: نحوه جستجوی پیکسل مشابه بر روی صفحه‌ای از یک تصویر در الگوریتم SLIC

زمانی که تمام پیکسل‌ها به نزدیکترین مرکز خوشه تخصیص یافتند، در مرحله‌ی به روزرسانی، مراکز خوشه‌ها به میانگین بُرداری پیکسل‌های متعلق به آن خوشه، به روزرسانی می‌شوند. با استفاده از نُرم L_2 خطای پس ماند E

(متناسب با تفاوت بردار ویژگی یک خوشه در یک دوره با دوره قبل) میان مراکز خوشه جدید و مراکز خوشه قبلی محاسبه می‌شود. مراحل تخصیص و به روزرسانی تا زمانی که این خطا همگرا شود ($0 \leq E \ll 1$)، تکرار می‌شوند. نویسندگان [۲۴] ادعا می‌کنند که تعداد ۱۰ دوره برای اغلب تصاویر کافی است. در پایان، در یک مرحله پس پردازش، پیکسل‌های ناهمبند به پیکسل‌های مجاور پیوست می‌شوند. خلاصه این شیوه در الگوریتم ۳-۱ نمایان است.

الگوریتم ۳-۱: ناحیه‌بندی با ابرپیکسل SLIC

- /* ***مقدار دهی اولیه*** */
- با گام نمونه برداری $\boldsymbol{S}$ مراکز خوشه‌های اولیه $\boldsymbol{C_k} = [\boldsymbol{l_i}\ \boldsymbol{a_i}\ \boldsymbol{b_i}\ \boldsymbol{x_i}\ \boldsymbol{y_i}]^{\boldsymbol{T}}$ مقداردهی اولیه شوند.
- مراکز خوشه‌ها به پایین ترین گرادیان در یک همسایگی ۳ در ۳ انتقال یابند.
- برای هر پیکسل $\boldsymbol{i}$ برچسب $\boldsymbol{l(i) = -1}$ تنظیم شود.
- برای هر پیکسل $\boldsymbol{i}$ فاصله $\boldsymbol{d(i) = \infty}$ تنظیم شود.

- **تا زمانیکه** (خطای پس ماند تمام خوشه‌ها < آستانه **یا** شماره دوره > بیشینه تعداد دوره‌ها)
  - /* ***تخصیص*** */
  - **برای هر** مرکز خوشه $\boldsymbol{C_k}$:
    - **برای هر** پیکسل $\boldsymbol{i}$ در ناحیه $\boldsymbol{2S \times 2S}$ حول $\boldsymbol{C_k}$:
      - فاصله $\boldsymbol{D}$، میان $\boldsymbol{C_k}$ و پیکسل $\boldsymbol{i}$ محاسبه شود.
      - **اگر** $\boldsymbol{D < d(i)}$ **آنگاه**:
        - $\boldsymbol{d(i) = D}$ تنظیم شود.
        - $\boldsymbol{l(i) = k}$ تنظیم شود.
  - /* ***به روزرسانی*** */
  - مراکز خوشه‌های جدید محاسبه شود.
  - خطای پس ماند $\boldsymbol{E}$ محاسبه شود.

معیار فاصله

همانطور که بیان شد، ابرپیکسل‌های SLIC متناظر با فضای رنگ $labxy$ تولید می‌شوند. در الگوریتم SLIC، فاصله میان پیکسل i ام و مرکز خوشه C_k است. از دیگر سوی، مؤلفه رنگی یک پیکسل در فضای CIELAB با بردار $[l\ a\ b]^T$ نمایش داده می‌شود. محدوده تغییرات این بردار برای هر پیکسل میزان مشخصی دارد. اما، موقعیت مکانی هر پیکسل، یعنی بردار $[x\ \ y]^T$ ، محدوده ای متناسب با اندازه تصویر دارد. از این رو تعریف D به عنوان فاصله اقلیدسی ۵ بعدی در فضای $labxy$ سبب ناسازگاری در رفتار ابرپیکسل‌ها با اندازه‌های مختلف می‌شود. بدین ترتیب که در ابرپیکسل‌های بزرگتر، فاصله مکانی اهمیت بیشتری نسبت به فاصله رنگ‌ها پیدا

می‌کند. از این رو، ابرپیکسل‌های متراکمی‌که تبعیت مناسبی از مرزهای موجود در تصویر ندارند، تولید می‌شوند. عکس این مسأله برای ابرپیکسل‌های کوچکتر صادق است.

به منظور تلفیق دو فاصله رنگی و مکانی و ایجاد یک معیار فاصله هنجار یافته، لازم است تا فواصل مکانی و رنگی، متناسب با بیشینه مقادیرشان (به ترتیب N_s و N_c) در هر خوشه، نرمال‌سازی شوند. بنابراین فاصله به صورت رابطه ۳-۳ بیان می‌شود.

$$d_c = \sqrt{(l_j - l_i)^2 + (a_j - a_j)^2 + (b_j - b_i)^2}$$

$$d_s = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

$$D = \sqrt{\left(\frac{d_s}{N_s}\right)^2 + \left(\frac{d_c}{N_c}\right)^2}$$

رابطه ۳-۳

بیشینه فاصله مکانی در یک خوشه، متناسب با گام نمونه برداری است؛ بنابر این $N_s = S = \sqrt{N/k}$. با توجه به اینکه خوشه به خوشه و تصویر به تصویر فاصله‌های رنگی تغییرات گوناگونی دارند، بیشینه فاصله رنگی N_c به صورت ثابتی مانند m در نظر گرفته می‌شود. بنابراین رابطه ۳-۳ به صورت رابطه ۳-۴ بازنویسی گردیده و در نهایت به شکل رابطه ۳-۵ ساده می‌شود.

$$D = \sqrt{\left(\frac{d_s}{S}\right)^2 + \left(\frac{d_c}{m}\right)^2}$$

رابطه ۳-۴

$$D = \sqrt{\left(\frac{d_s}{S}\right)^2 m^2 + d_c^2}$$

رابطه ۳-۵

با تعریف D به صورت رابطه ۳-۳، فاصله مکانی و رنگی قابل تنظیم و توازن هستند. پارامتر m (پارامتر تراکم) در انسجام ابرپیکسل‌ها تاثیر مستقیم دارد. به ازای mهای بزرگ اهمیت فاصله مکانی بالاتر و ابرپیکسل‌ها تراکم بیشتر دارند. از سوی دیگر، به ازای mهای کوچک، تبعیت بیشتری از مرزها حاصل می‌شود؛ اما اندازه و شکل ابرپیکسل‌ها ناهمگون تر می‌گردند. [۲۴] محدوده تغییرات m را در بازه [40 1] در نظر می‌گیرد.

پس پردازش

همانند دیگر الگوریتم‌های ابرپیکسلی معرفی شده در فصل دوم، SLIC نیز به طور مستقیم، اجباری بر پیوستگی خوشه‌ها ندارد. در پایان مرحله خوشه‌بندی، برخی پیکسل‌ها به صورت منفرد باقی می‌مانند و در خلال پردازش به مؤلفه همبند خود تخصیص نمی‌یابند. برای حل این مشکل، با استفاده از نوعی الگوریتم مؤلفه همبند، پیکسل‌های منفرد، به نزدیکترین خوشه تخصیص می‌یابند.

پیچیدگی محاسباتی

با محلی سازی جستجو، SLIC از محاسبه هزاران فاصله اضافی، جلوگیری می‌کند. در عمل، هر پیکسل در ناحیه جستجوی ۸ (یا کمتر) مرکز خوشه قرار می‌گیرد. بنابراین پیچیدگی الگوریتم SLIC به صورت $O(N)$ و مستقل از تعداد ابرپیکسل‌ها تعریف می‌شود؛ در حالیکه پیچیدگی بهینه ترین k-means، پیاده سازی شده توسط [۷۲]، به صورت $O(Nkw)$ -که در آن w تعداد دوره‌ها است- تعریف می‌شود. حال آنکه روشهای گوناگونی برای سرعت بخشیدن به الگوریتم k-means، مانند نمونه برداری با گام اعداد اول [۷۳]، نمونه برداری تصادفی [۷۴]، مبادله محلی خوشه [۷۵]، نامساوی مثلثی و تعیین حد بالا و پایین [۷۶]، پیشنهاد شده است؛ اما این روشها طبیعت عملکردی کاملا سراسری دارند.

معیارهای فاصله پیچیده‌تر

ممکن است به نظر برسد که معیارهای فاصله با سطح و پیچیدگی بالا باعث بهبود عملکرد SLIC شوند. [۲۴] با مقایسه ضریب نرمال‌سازی وفقی در الگوریتم SLIC وفقی یا ASLIC[1] و معیار فاصله ژئودزیک در الگوریتم GSLIC[2] به این نتیجه رسیده است که همان معیار ساده فاصله بررسی شده در بخش قبل، از جهت سرعت، حافظه و تبعیت از مرز نسبت به معیارهای فاصله پیچیده‌تر برتری دارد.

همانطور که در بخش قبل توضیح داده شد، S و m به ترتیب بیشینه فاصله مکانی و بیشنیه فاصله رنگی هستند. این متغیرهای ثابت وظیفه نرمال‌سازی فواصل رنگی و مکانی را بر عهده دارند. از طرفی این دو متغیر می‌توانند تنها در یک پارامتر تنظیم شونده برای خوشه‌بندی ترکیب شوند. الگوریتم ASLIC به جای استفاده از این متغیرهای ثابت، فواصل هر خوشه را توسط بیشینه فواصل رنگی و مکانی مشاهده شده (m_s و m_c) از دوره قبل، به شکلی وفقی نرمال‌سازی می‌کند. بنابراین مقیاس فاصله به صورت رابطه ۳-۶ تعریف می‌شود.

---

[1] Adaptive-SLIC
[2] Geodesic-SLIC

$$D = \sqrt{\left(\frac{d_s}{m_s}\right)^2 + \left(\frac{d_c}{m_c}\right)^2}$$

رابطه ۳-۶

در این رویکرد، در دوره اول از همان متغیرهای ثابت بهره گرفته می‌شود؛ اما در دوره‌های بعد، الگوریتم به شکل پیوسته، بیشینه فواصل را در هر خوشه اندازه می‌گیرد. مزیت این روش این است که ابرپیکسل‌ها متراکم‌تر تولید می‌شوند و نیازی به تنظیم پارامتر m نیست. هرچند، این مزیت در ازای تبعیت نامناسب از مرزها حاصل می‌شود [۲۴].

الگوریتم GSLIC معیار فاصله در رابطه ۳-۵ را با فاصله ژئودزیک تعویض می‌کند. فاصله بی‌علامت ژئودزیک از یک پیکسل $I(p_i)$ تا پیکسل $I(p_j)$ به صورت رابطه ۳-۷ تعریف می‌شود.

$$G\left(I(p_i), I\left(p_j\right)\right) = \min_{P \in \Gamma} d(P)$$

رابطه ۳-۷

در این رابطه، Γ مجموعه‌ای از مسیرهای میان $I(p_i)$ و $I(p_j)$ و $d(P)$ هزینه تعریف شده بر روی مسیر P است. $d(P)$ به شکل رابطه ۳-۸ تعریف می‌شود.

$$d(P) = \sum_{i=2}^{n} \|I(p_i) - I(p_{i-1})\|$$

رابطه ۳-۸

در این رابطه، $\|I(p_i) - I(p_{i-1})\|$ فاصله اقلیدسی میان بردار فاصله پیکسل p_i و p_{i-1} در فضای تصویر CIELAB تعریف می‌شود. این رویکرد، همبستگی در فضای مکانی را ضمانت می‌کند؛ از این رو به مرحله پس‌پردازش نیازی نیست. در حالیکه تبعیت مناسبی از مرزها حاصل نمی‌شود [۲۴].

پیاده‌سازی SLIC بر روی ویدئوهای فرد منظر

زمانی که سخن از پیاده‌سازی یک الگوریتم پردازش تصویر بر روی سیگنالهای ویدئویی به میان می‌آید، این سوال مطرح می‌شود که آیا می‌توان با بهره گیری از اطلاعات یک قاب، قاب پس از آن را پردازش کرد؟ آچانتا و همکارانش [۶۶] در پاسخ به این سوال، چارچوب فیلتر بیزین (تخمین بازگشتی بیزین) و فیلتر Kalman [۷۷] را پیشنهاد می‌دهند. فرض کنیم مقادیر ورودی (x_i) به یک سامانه، از مدل ریاضی مشخصی مانند $p(x|\theta)$ پیروی می‌کند؛

اما پارامتر θ از سامانه، نامشخص است. هم چنین دانش اولیه‌ای از این مقادیر ورودی به شکل $p(\theta|D^0) = p(\theta)$ در دسترس است. سپس فیلتر بیزین به ازای ورود داده جدید، به صورت بازگشتی و با استفاده از شرایط اولیه، تخمینی از وضعیت جدید سامانه به دست می‌آورد (رابطه ۳-۹).

$$p(\theta|D^1) \propto p(x|\theta)p(\theta|D^0)$$

رابطه ۳-۹

به همین ترتیب به ازای دنباله ای از ورودیها، وضعیت جدید سیستم به صورت بازگشتی، پیش‌بینی می‌شود (رابطه ۳-۱۰). در نهایت با استفاده از رهیافت «برآورد بیشینه احتمال»[1]، تخمینی از θ به دست می‌آید.

$$p(\theta|D^n) = \frac{p(x_n|\theta)p(\theta|D^{n-1})}{\int p(x_n|\theta)p(\theta|D^{n-1})d\theta}$$

رابطه ۳-۱۰

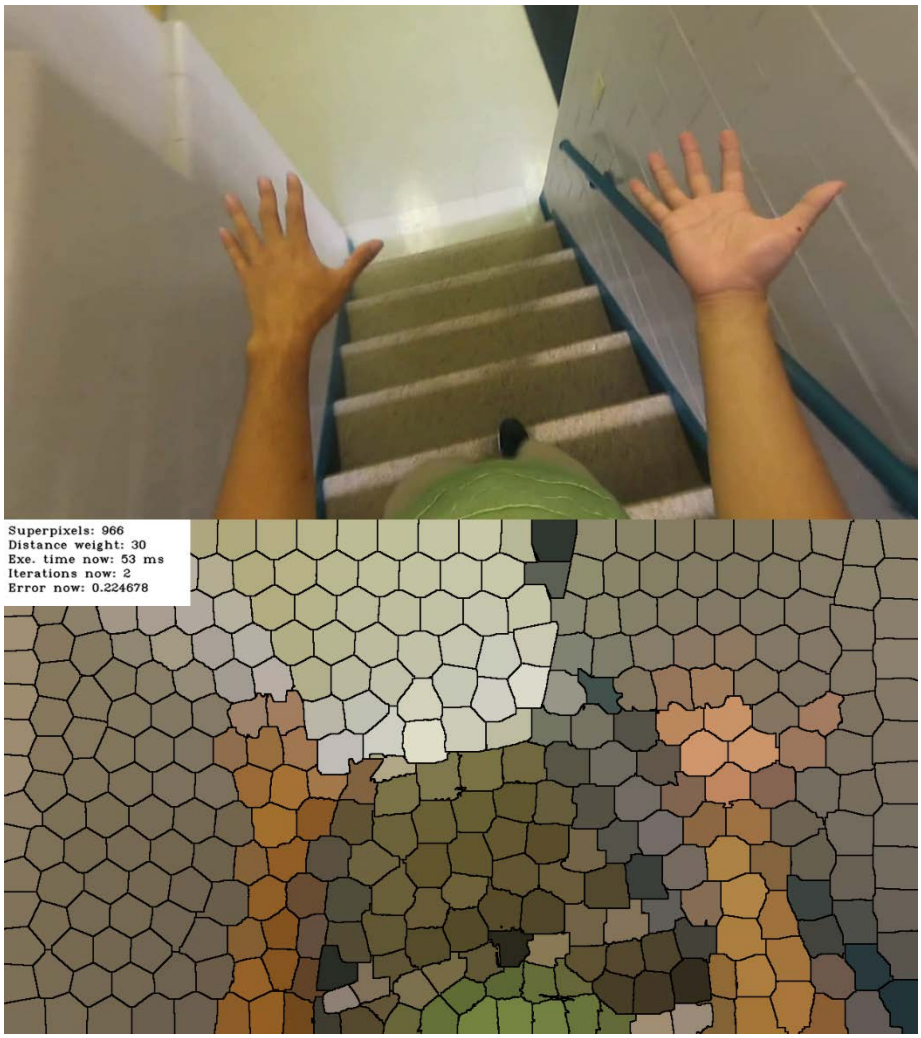
از سویی دیگر، فیلتر Kalman با کم کردن مربعات خطا، به طور مشابه، تخمینی از وضعیت یک فرآیند ارائه می‌دهد. این فیلتر دارای دو مرحله «پیش بینی» و «به روزرسانی» است.

پیاده سازی SLIC بر روی قابهای یک ویدئو به صورت مستقل، بسان پرش از روی مرحله پیش بینی در فیلتر Kalman است. بنابراین، با الهام از این دو فیلتر، در اجرای الگوریتم SLIC فرض می‌شود، مرکز خوشه‌های فریم جدید، در همسایگی کوچکی از مرکز خوشه‌های فریم قبل قرار دارد. آنچه در [۶۶] پیشنهاد می‌شود، استفاده مستقیم از دو فیلتر بیزین و Kalman نیست؛ بلکه الهامی‌از این دو رهیافت است. از این رو، به جای مقدار دهی اولیه خوشه‌ها در هر فریم به صورت شبکه ای از مراکز با فاصله، (شکل ۳-۴) موقعیت این مراکز در فریم‌های متوالی پیش بینی می‌شود. فیلتر Kalman بازگو می‌کند که اگر دانش دقیقی از سامانه موجود نبود، بهترین گزینه، فرض یکسان بودن وضعیت جدید سامانه با وضعیت قبلی است. به عبارتی دیگر مقدار اولیه مراکز خوشه‌های هر فریم برابر با مراکز خوشه‌های فریم پردازش شده قبل در نظر گرفته می‌شود. با اعمال چنین الگویی در اجرای الگوریتم SLIC بر روی ویدئوهای فرد منظر، به دوره‌های کمتری نیاز می‌شود و همگرایی الگوریتم سریعتر اتفاق می‌افتد. به منظور مقایسه در بخش نتایج، راهبرد پیاده‌سازی SLIC به روش بیزین را الگوریتم «خام» می‌نامیم.

اما پیاده سازی چنین راهبردی مشکلاتی در بر دارد. همانطور که در شکل ۳-۶ مشاهده می‌شود، با گذشت مدتی از اجرای الگوریتم، ابرپیکسل‌ها رفتار ناپایداری را از خود نشان می‌دهند. به نحوی که در نواحی پر مرز تصویر،

[1] Maximum Likelihood Estimation

شاهد ایجاد ابرپیکسل‌های ناهمگون و عدم تبعیت مناسب از مرزهای موجود هستیم. از این رو مصنوعات ناخواسته ای در تصویر پدیدار می‌شود.دلیل ایجاد چنین مصنوعاتی، پیاده سازی محلی k-means است.



شکل ۳-۶: ایجاد ابرپیکسل‌های ناهمگون و مصنوعات ناخواسته پس از گذشت مدتی از اجرای الگوریتم SLIC بر روی ویدئوی فردمنظر

بنابراین در قابهایی از یک ویدئو که در آنها سرعت واگرایی اطلاعات پیکسلی زیاد است، احتمال دارد مراکز خوشه‌ها در یک همسایگی، از یکدیگر دور شده و شکافی میان پنجره‌های جستجوی مجاور هم ایجاد کنند. در این صورت، پیکسل‌های موجود در شکاف، در روند جستجو قرار نگرفته و در قاب کنونی پردازش نمی‌شوند؛ از این رو برچسب دوره قبل را بر خود دارند. [۶۶] به منظور حل این مشکل نیز دو راه حل ارائه می‌دهد:

۱. **تکرار مقدار دهی اولیه به صورت شبکه ای منظم از مراکز خوشه با فاصله S بعد از پردازش تعدادی قاب(تزریق میان قابی):** پس از پردازش تعداد مشخصی از قاب‌ها، مراکز خوشه‌ها بار دیگر

مقدار دهی اولیه می‌شوند و این فرآیند در طول اجرای الگوریتم با نرخ مشخصی تکرار می‌گردد. در صورت اعمال این راهبرد، بار محاسباتی به مقدار اندکی افزایش یافته و از سرعت پردازش، به دلیل افزایش تعداد دوره‌های مورد نیاز برای همگرایی، کاسته می‌شود. هرچه این نرخ بیشتر باشد، همگرایی کندتر صورت می‌گیرد. این راهبرد، اقتباسی است از آنچه در مباحث رمزگشایی ویدئوها مطرح می‌شود.

۲. **اضافه کردن مقداری نویز گوسی:** نویز به گونه‌ای از ایجاد مصنوعات ناخواسته می‌کاهد (شکل ۳-۷). البته تزریق نویز، بار محاسباتی و تعداد دوره‌های مورد نیاز برای همگرایی را افزایش می‌دهد. هرچه نویز بیشتر باشد، مصنوعات کم‌تری ملاحظه می‌شود.

شکل ۳-۷: کاهش ابرپیکسل‌های ناهمگون و مصنوعات ناخواسته با اضافه کردن نویز گوسی به قاب

در مسیر بهینه سازی الگوریتم SLIC رویکردهای متفاوتی آزموده شد که در ادامه به بررسی آنها می‌پردازیم.

روش پیشنهادی ۱

همانطور که بیان شد. الگوریتم SLIC نوعی خوشه‌یابی محلی با اقتباس از k-means است. الگوریتم خوشه یابی k-means ابزاری رایج در تحلیل داده و آموزش به شمار می‌آید. در قلب این الگوریتم، مسئله بهینه‌سازی بسیار ساده‌ای نهفته است: با شرط داشتن مجموعه‌ای از داده‌ها (در فضایی برداری)، k نقطه دیگر را (k مرکز را) به نحوی موقعیت‌یابی می‌کند که فاصله (مربعات فاصله) بین هر داده و نزدیکترین مرکزش کمینه باشد. اگرچه الگوریتم k-means رایج و به سادگی قابل پیاده‌سازی است، اما رویکرد خام این الگوریتم، از نظر پردازشی هزینه بر و دارای محاسبات افزونه است.

در این بخش رویکردی مبتنی بر نامساوی مثلثی را بررسی خواهیم کرد که از محاسبات افزونه جلوگیری کرده و در عین بهینه بودن، خروجی برابری با رویکرد خام الگوریتم k-means ارائه می‌دهد. با بررسی پژوهش‌های مرتبط انتظار داریم که بتوانیم الگوریتم SLIC را نیز بهینه کنیم. الگوریتم k-means در [۷۸] به عنوان یکی از ۱۰ الگوریتم برتر داده کاوی انتخاب شده است. این الگوریتم در بسیاری از بسته‌های نرم افزاری تجاری و متن باز تحلیل آماری داده چون MATLAB، SAS، Stata، SPSS، R، Weka و OpenCV گنجانده شده است. شهرت k-means به علت شفاف بودن، سادگی و شهودی بودن تابع بهینه‌سازی‌اش با رویکردهای متفاوتی پیاده‌سازی شده است؛ اما اغلب این رویکردها عملکردی بهینه ندارند. با بررسی کد منبع این الگوریتم در بسته‌های نرم‌افزاری ELKI، graphlab، Mahout، MATLAB، MLPACK، Octave، OpenCV، R، SciPy، Weka و Yael مشخص می‌شود که هیچکدام رویکردی مبتنی بر نامساوی مثلثی ندارند.

رویکردهای استاندارد برای حل مسئله بهینه‌سازی الگوریتم k-means ،الگوریتم لوید [۷۲] (الگوریتمی‌دسته ای، هم چنین معروف به الگوریتم لوید-فورجی [۷۹]) و الگوریتم مک کوئین [۸۰] است. هر کدام از این الگوریتم‌ها زمان زیادی را صرف محاسبه فاصله بین داده‌های خوشه بندی شده و مرکز خوشه در جریان می‌کنند. در حالیکه در اغلب مواقع این محاسبه‌ها غیر ضروری و هزینه بَر هستند.

دلیل اصلی بر بهینه نبودن الگوریتم‌های دسته ای مسئله بهینه‌سازی k-means این است که در هر دوره بایستی برای تمام نقاط خوشه بندی شده، تزدیکترین مرکز خوشه پیدا شود. از این رو در رویکردهای خام متداول به تعداد nk فاصله میان n نقطه خوشه‌بندی شده و k مرکز محاسبه می‌شود. بعد از هر دوره، مراکز جابه‌جا می‌شوند و ممکن است تمامی این فاصله‌های محاسبه شده تغییر کرده و نیاز به محاسبه مجدد باشد. اما در عمل مراکز جابه‌جایی زیادی ندارند؛ غالبا بعد از چندین دوره اول، نزدیکترین مرکز خوشه به یک نقطه در دوره قبل، در دوره جدید نیز نزدیک‌ترین خوشه به آن نقطه می‌ماند. بنابراین ردگیری نزدیکترین خوشه برای هر داده خوشه‌بندی شده، می‌تواند در کم کردن بار محاسباتی در دوره‌های بعد مفید باشد. در حالت ایده‌آل زمانیکه موقعیت نزدیکترین

مرکز تغییر نکند، نیازی به محاسبه فاصله میان آن نقطه و دیگر مراکز خوشه نیست. حتی در صورتی که نزدیکترین مرکز خوشه برای نقطه‌ای تغییر کند، با در نظر گرفتن شرایطی می‌توان از محاسبه فاصله آن نقطه تا سایر مراکز جلوگیری کرد. سپس، تنها چند مرکزی که مطمئن هستیم نسبت به سایر مراکز به آن نقطه نزدیک‌ترند را بررسی می‌کنیم.

مسیری موثر در سرعت بخشیدن به الگوریتم‌های یادگیری در پژوهش‌های علوم کامپیوتر پیموده شده است. اصلی‌ترین رویکردها در این مسیر، اقتباسی از بهینه سازیهای الگوریتمی، موازی سازی (ریسه بندی [1]، چندپردازشی و رایانش توزیع شده [2]) و تقریب سازی هستند. بهره گیری از نامساوی مثلثی نوعی بهینه سازی الگوریتمی است. در ادامه به بررسی الگوریتم خوشه یابی لوید و سپس به بهینه سازی آن توسط نامساوی مثلثی می‌پردازیم.

الگوریتم خوشه یابی Lloyd's Algorithm

زمانیکه الگوریتم k-means خوشه‌یابی می‌کند، در واقع نوعی اغتشاش و یا مجموع مربعات خطا میان داده‌ها و مراکزی که به آنها تخصیص یافته‌اند کمینه می‌شود. اغتشاش با دو رویکرد کمینه می‌شود: تغییر خوشه تخصیص یافته به نقاط و جابه‌جایی مراکز خوشه. خلاصه ای از علائم و اختصاراتی که به منظور شرح الگوریتم به آنها اشاره خواهیم کرد جدول ۳-۱ مشاهده می‌شود.

نام و نوع اختصار	توضیح
$\boldsymbol{d} \in \mathbb{N}$	ابعاد مراکز خوشه و داده‌های ورودی.
$\boldsymbol{n} \in \mathbb{N}$	تعدا نقاطی که باید خوشه بندی شوند.
$\boldsymbol{k} \in \mathbb{N}$	تعداد مراکز خوشه‌ها.
$\boldsymbol{X} \subset \boldsymbol{\mathcal{R}^{n \times d}}$	مجموعه داده‌هایی که بایستی خوشه بندی شوند، اندیس شده به صورت $x(i)$ به ازای $1 \leq i \leq n$.
$\boldsymbol{C} \subset \boldsymbol{\mathcal{R}^{n \times d}}$	مجموعه خوشه‌ها، اندیس شده به صورت $c(j)$ به ازای $1 \leq j \leq k$.
$\boldsymbol{n(j)} \in \boldsymbol{\mathcal{R}}$	تعداد نقاطی که در حال حاضر به خوشه j تخصیص یافته اند.
$\boldsymbol{N} = \{\, \boldsymbol{i} \in \mathbb{N} \mid \boldsymbol{1} \leq \boldsymbol{i} \leq \boldsymbol{n}\}$	اندیس نقاط موجود در مجموعه X
$\boldsymbol{K} = \{\, \boldsymbol{j} \in \mathbb{N} \mid \boldsymbol{1} \leq \boldsymbol{j} \leq \boldsymbol{k}\}$	اندیس نقاط موجود در مجموعه C
$\boldsymbol{a} : \boldsymbol{N} \rightarrow \boldsymbol{K}$	اندیس نزدیکترین مرکز اختصاص یافته به هر نقطه

جدول ۳-۱: خلاصه ای از علائم و اختصارات الگوریتم خوشه یابی لوید

[1] Threading
[2] Distributed Computing

در حالتی خاص با داشتن مجموعه ای ثابت از نقاط X مایل به کمینه کردن تابع اغتشاش در رابطه ۳-۹ با انتخاب بهترین مجموعه خوشه‌ها (C) هستیم.

$$J(X,C)=\sum_{i\in N}\left\|x(i)-c\big(a(i)\big)\right\|^{2}$$

رابطه ۳-۱۱

الگوریتم دسته ای لوید در سه مرحله به کمینه کردن تابع اغتشاش می‌پردازد:

۱. مراکز مقدار دهی اولیه شوند.
۲. تا زمانی که الگوریتم همگرا شود:
   ۲-۱. هر نقطه را به نزدیکترین مرکز اختصاص بده.
   ۲-۲. هر مرکز را به میانگین مراکزی که در حال حاضر تخصیص یافته انتقال بده.

مراحل بالا در الگوریتم ۳-۳ نیز نمایان است.

الگوریتم ۳-۲: پیاده سازی k-means توسط الگوریتم لوید

- **تا زمانیکه** (الگوریتم همگرا نشده است)
  - **برای هر** $i \in N$: نزدیکترین مرکز تا هر $x(i)$ را بیاب:
    - $1 \rightarrow a(i)$
    - **برای هر** $j \in K$
      - **اگر** $\|x(i)-c(j)\| < \|x(i)-c(a(i))\|$ **آنگاه:**
        - $j \rightarrow a(i)$ ♦
  - **برای هر** $j \in K$: مراکز را جابه جا کن:
    - $c(j)$ را به میانگین $\{x(i)|a(i)=j\}$ **انتقال بده.**

مرحله ۱ تنها یک بار اتفاق می‌افتد، در حالیکه مراحل ۲-۱ و ۲-۲ تا زمانی که الگوریتم همگرا شود، تکرار می‌شوند. به علت اینکه دو مرحله ۲-۱ و ۲-۲ تابع $J(X,C)$ را کاهش می‌دهند و مسیرهای متناهی برای تقسیم بندی n نقطه میان k خوشه وجود دارد [۸۱]، همگرایی تضمین شده است.

پژوهش‌های متفاوتی در باب بهینه‌سازی مرحله ۱ یعنی مرحله مقداردهی اولیه برای الگوریتم k-means انجام گرفته است [۸۲]. پژوهشگران از رویکردهای چندین مثال اولیه [۸۱]، انتخاب k نقطه از X به صورت تصادفی

[۸۳] و رویکرد دورترین-اول [۸۴] به منظور بهینه سازی مرحله مقدار دهی بهره گرفته اند. موثرترین و بهینه ترین شیوه، مربوط به رویکرد مقدار دهی اولیه یا k-means++ است [۸۵]. در این رویکرد با الهام از شیوه دورترین-اول، مقداردهی اولیه مناسبی به صورت تصادفی و با احتمال بالا صورت می‌گیرد.

مرحله ۲-۲ به سادگی با ذخیره اطلاعات آماری هر خوشه به صورت نهان، بهینه‌سازی می‌شود: برداری از مجموع نقاط تخصیص یافته به خوشه و تعداد نقاطی که به آن خوشه اختصاص یافته است. با ذخیره چنین اطلاعات آماری، از محاسبه مجموع تمام نقاط در هر دوره جلوگیری می‌شود. هر زمان که خوشه تخصیص یافته به نقطه‌ای تغییر کند، این اطلاعات آماری به روز می‌شود. بعد از چندین دوره اول، اغلب نقاط تغییر خوشه نمی‌دهند. بنابراین ذخیره چنین اطلاعات آماری در هر مرحله کم هزینه‌تر می‌شود. الگوریتم SLIC و روش ارائه شده از این بهینه سازی بهره می‌گیرند.

زمان اجرای الگوریتم لوید متناسب با تابع پیچیدگی $O(wnkd)$ به ازای w دوره، k مرکز و n نقطه با ابعاد d ارزیابی می‌شود. برای مجموعه دادگان ثابت و متغیر ثابت k، تعداد دوره‌ها (w) بسته به مقدار دهی اولیه است. به عبارت دیگر، ممکن است متناسب با n، فراخطی و یا در بدترین حالت نمایی باشد [۸۶]. در حالیکه با درنظر گرفتن شرایط متعادل، الگوریتم لوید پیچیدگی‌ای به صورت چند جمله ای نرم دارد[۸۷]. به این معنی که w در n و $1/\sigma$ چندجمله ای است (σ میزان اختلال مجاز بر روی دادگان).

زمانیکه از منظر الگوریتم گرادیان نزولی نگریسته می‌شود، بوتو و بنجیو [۸۱] نشان داده‌اند که همگرایی در الگوریتم دسته ای لوید با سرعت فراخطی حاصل می‌شود. این سرعت، نتیجه محدود بودن مشتق دوم تابع هزینه است. از این رو، الگوریتم لوید مشابه اسلوب نیوتن اغتشاش را کمینه می‌کند.

رویکرد نامساوی مثلثی

نامساوی مثلثی ابزاری ساده اما پر قدرت از هندسه به شمار می‌آید. اگر $a, b, c \in \mathbb{R}^d$ باشند، نامساوی مثلثی بیان می‌کند:

$$\|a - c\| \leq \|a - b\| + \|b - c\|$$

رابطه ۳-۱۲

به نحوی که بردار نُرم اقلیدسی به صورت $\|a\| = \sqrt{a^T a}$ تعریف می‌شود. به صورت شهودی، نامساوی به این معنی است که طول پاره خط (a, c) حداکثر برابر مجموعه دو پاره خط (a, b) و (b, c) است. این نامساوی به اشکال گوناگون قابل اعمال به الگوریتم k-means است.

به طور کلی، قصد بر این است که اثبات شود که یک مرکز، نسبت به سایر مراکز، به نقطه‌ای که قصد خوشه‌بندی آن را داریم، نزدیکتر است. در حالت ایده آل نیز بایستی این رویکرد با کم‌ترین بار محاسباتی پیاده‌سازی شود. برای نقطه ای مانند x و دو مرکز c و c' روشهای متفاوتی برای اعمال نامساوی مثلثی بر الگوریتم k-means وجود دارد که در این پایان‌نامه یکی از رویکردها آزمایش شده است:

با در اختیار داشتن فواصل $\|c' - x\|$ و $\|c' - c\|$ اثبات شود که c' در مقایسه با c، به x نزدیکتر است [۸۸].

فیلیپس، دو راهبرد به منظور استفاده از نامساوی مثلثی برای سرعت بخشیدن به الگوریتم k-means ارائه می‌دهد [۸۹]. هر دوی این راهبردها از نامساوی مثلثی بهره می‌گیرند تا اثبات کنند که مراکزی که از خوشه کنونی دور هستند از نقاط این خوشه نیز دور هستند. بنابراین نیازی به محاسبه فاصله نقاط تا آن مراکز نیست. فیلیپس نام الگوریتم‌های خود را «مقایسه میانگین‌ها» و «چینش میانگین‌ها» قرار داده است.

و از طرفی اگر مرکز دیگری مانند c از مرکز c' دور باشد، پس مرکز c' باید از c به x نزدیکتر باشد. در الگوریتم مقایسه میانگین‌ها با استفاده از نامساوی مثلثی اثبات می‌شود که اگر مرکز c' به نقطه x نزدیک باشد با داشتن فواصل $\|c' - x\|$ و $\|c' - c\|$ از قبل و اعمال نامساوی مثلثی می‌توان نشان داد:

$$\|c - c'\| \leq \|x - c\| + \|x - c'\|$$

یا

$$\|c - c'\| - \|x - c'\| \leq \|x - c\|$$

رابطه ۳-۱۳

بنابراین اگر بدانیم که شرط لازم و کافی $2\|x - c'\| \leq \|c - c'\|$ (به دلیل دانستن این فواصل از قبل، بار محاسباتی نداریم) برقرار است، می‌توان نشان داد:

$$2\|x - c'\| - \|x - c'\| \leq \|x - c\|$$

یا

$$\|x - c'\| \leq \|x - c\|$$

رابطه ۳-۱۴

رابطه ۳-۱۴ بدون محاسبه فاصله $\|x - c\|$ اثبات می‌کند که c از c' به x نزدیکتر نیست. فیلیپس الگوریتم مقایسه میانگین‌ها را در داخلی ترین حلقه الگوریتم لوید به کار می‌برد تا نشان دهد که برخی فواصل میان نقطه و مراکز

نیازی به محاسبه ندارند. این الگوریتم فواصل بین مراکز را قبل از شروع هر دوره محاسبه و به صورت پنهان ذخیره می‌کند.

الگوریتم چینش میانگین‌ها، هر بار که مراکز جابه جا می‌شوند، ابتدا ماتریسی با ابعاد $k \times k$ را برای ذخیره فواصل میان مراکز را تشکیل داده و سپس هر ردیف از این ماتریس را بر حسب فاصله مرتب‌سازی می‌کند. بنابراین درجه‌ای از فاصله برای هر مرکز نسبت به سایر مراکز به دست می‌آید. هر زمان که الگوریتم بخواهد تا نزدیکترین مرکز را برای نقطه x بیابد، به دنبال مراکز دیگر به ترتیب افزایش فاصله در این ماتریس می‌گردد. پس اگر الگوریتم بتواند شرط لازم و کافی را برقرار کند، دست از جستجو بر می‌دارد. به عبارت دیگر الگوریتم چینش میانگین مشابه با الگوریتم مقایسه میانگین‌ها از نامساوی مثلثی بهره می‌برد اما با ترتیب متفاوتی به جستجوی میان مراکز می‌پردازد. بدین ترتیب از بررسی مراکز بسیار دور ممکن است جلوگیری شود. البته توجه داریم که الگوریتم چینش میانگین به دلیل مرتب سازی نسبت به الگوریتم مقایسه میانگین‌ها بار محاسباتی اضافه دارد. در این پایان نامه به منظور آزمایش نامساوی مثلثی در بهینه سازی الگوریتم SLIC از الگوریتم مقایسه میانگین‌ها استفاده شده است.

SLIC با رویکرد نامساوی مثلثی (مقایسه میانگین‌ها)

نخست ماتریس M را برای ذخیره سازی فاصله بین مراکز به صورت رابطه ۳-۱۵ تعریف می‌کنیم.

$$M_{k\times k} = \begin{bmatrix} m_{11} & \cdots & m_{1k} \\ \vdots & \ddots & \vdots \\ m_{k1} & \cdots & m_{kk} \end{bmatrix} \mid m_{i,j} = \left\| C_i - C_j \right\|$$

رابطه ۳-۱۵

این ماتریس در مرحله تخصیص و قبل از شروع هر دوره محاسبه می‌شود. سپس با بررسی شرط لازم و کافی $2\left\|x - c'\right\| \le \left\|c - c'\right\|$ تصمیم گرفته می‌شود که آیا فاصله نقطه مورد بررسی تا مرکز جدید محاسبه شود یا خیر. بدین ترتیب انتظار داریم اگر نقطه‌ای این شرط را ارضا کند محاسبه افزونه صورت نگرفته و بار محاسباتی کم شود. خلاصه این رویکرد پیشنهادی در الگوریتم ۳-۳ مشاهده می‌شود.

الگوریتم ۳-۳: ناحیه‌بندی با ابرپیکسل SLIC بهینه (روش پیشنهادی ۱)

- /* ***مقداردهی اولیه*** */
- با گام نمونه برداری S مراکز خوشه‌های اولیه $C_k = [l_i \ a_i \ b_i \ x_i \ y_i]^T$ مقداردهی اولیه شوند.
- مراکز خوشه‌ها به پایین ترین گرادیان در یک همسایگی ۳ در ۳ انتقال یابند.
- برای هر پیکسل i برچسب $l(i) = -1$ تنظیم شود.
- برای هر پیکسل i فاصله $d(i) = \infty$ تنظیم شود.

- **تا زمانیکه** (خطای پس ماند تمام خوشه‌ها < آستانه **یا** شماره دوره < بیشینه تعداد دوره‌ها)
  - /* ***تخصیص*** */
  - **محاسبه ماتریس** M
  - **برای هر** مرکز خوشه C_k:
    - **برای هر** پیکسل i در ناحیه $2S \times 2S$ حول C_k :
      - **اگر** $2d(i) \geq m_{k,l(i)}$ **آنگاه:**
        - فاصله D، میان C_k و پیکسل i محاسبه شود.
        - **اگر** $D < d(i)$ **آنگاه**:
          - $d(i) = D$ تنظیم شود.
          - $l(i) = k$ تنظیم شود.
  - /* ***به روزرسانی*** */
  - مراکز خوشه‌های جدید محاسبه شود.
  - خطای پس ماند E محاسبه شود.

البته شایان ذکر است که در برنامه نویسی C++ به منظور کاهش بار محاسباتی در مقایسه فاصله‌ها، مجذور فاصله‌ها با یکدیگر مقایسه می‌شوند. تابع sqrt() معمولا در زبان‌های برنامه نویسی بار محاسباتی زیادی دارد. از سویی شرط لازم و کافی $\|c - c'\| \leq \|x - c'\| 2$، مقایسه را با ریشه دوم خروجی تابع $\|.\|^2$ انجام می‌دهد. بنابراین شرط $\|c - c'\| \leq \|x - c'\| 2$ بایستی به صورت $\|c - c'\| \leq \|x - c'\| 4$ یا در الگوریتم ۳-۳ به صورت $4d(i) \geq m_{k,l(i)}$ بررسی شود.

برای آزمایش، از ویدئویی با قابهایی به اندازه 720×1280 پیکسل استفاده کردیم. تعداد ابرپیکسل‌ها را نیز ۱۰۰۰ عدد تنظیم کردیم. برخلاف انتظار، روش پیشنهادی ۱ عملکردی کندتر نسبت به نتایج [۶۶] ارائه داد. خلاصه ای از این نتایج در جدول ۳-۲ مشخص است.

الگوریتم SLIC	میانگین زمان اجرا بر روی هر قاب (میلی ثانیه)	قاب بر ثانیه
الگوریتم خام [۶۶]	۶۴,۲	۱۵,۶
الگوریتم خام پیشنهادی ۱	۱۴۴,۸۴	۶,۹

جدول ۳-۲: نتیجه پیاده سازی الگوریتم پیشنهادی ۱

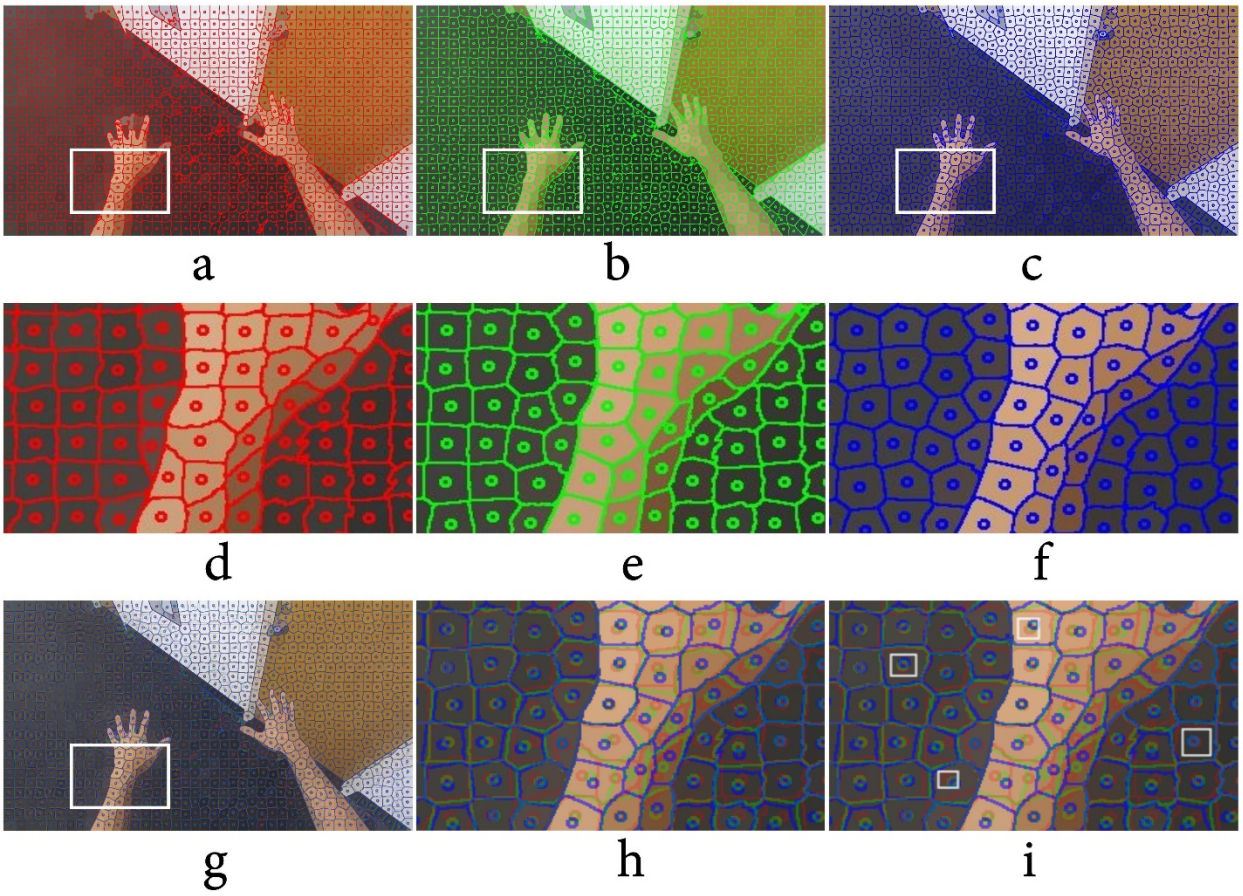
با تفسیر اجرای خط به خط الگوریتم و جستار متغیرهای آن، متوجه شدیم که الگوریتم به طور میانگین از تعداد ۳۶۰۰ پیکسل موجود در یک محدوده جستجو، تنها از محاسبه فاصله ۵۳ پیکسل جلوگیری می‌کند. هم چنین بررسی شرط نامساوی مثلثی و تشکیل ماتریس M در هر دوره، بار محاسباتی الگوریتم را افزایش می‌دهد. بنابراین رویکردی دیگر را توسعه و پیشنهاد دادیم (روش پیشنهادی ۲).

روش پیشنهادی ۲

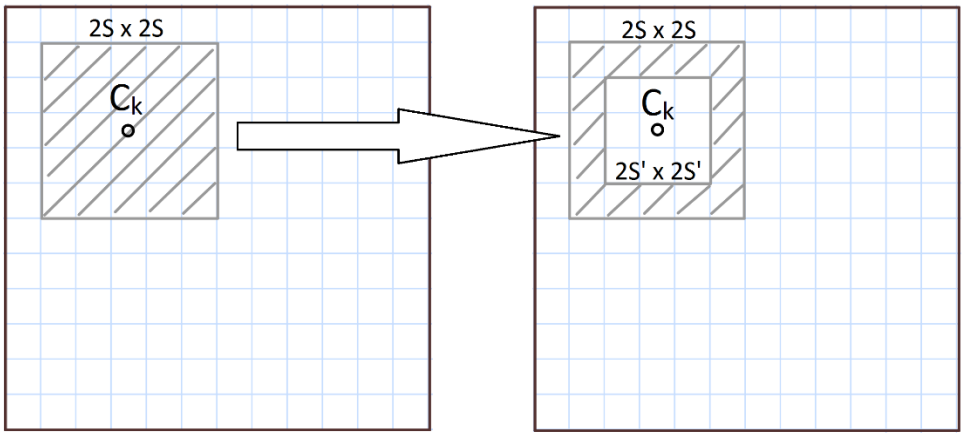
با بررسی قاب‌های تجزیه شده با ابرپیکسل SLIC در [۶۶] متوجه شدیم که در هر دوره از اجرای الگوریتم بر روی یک قاب، تنها پیکسل‌هایی که دور از مرکز خوشه هستند تغییر خوشه زیادی دارند و پیوسته به‌روز می‌شوند (شکل ۳-۸). بنابراین پیش‌بینی کردیم که با تبدیل محدوده مربع شکل جستجو به یک محدوده جستجوی حلقه‌ای شکل (شکل ۳-۹) می‌توان از بار محاسباتی الگوریتم کاست. همانطور که در تصویر i از شکل ۳-۹ مشاهده می‌شود در اغلب خوشه‌ها، با ۱۸ دوره تکرار الگوریتم SLIC، پیکسل‌های بسیاری در همان خوشه قبل خود مانده و در فرآیند الگوریتم، به روز رسانی نشده‌اند. بدین ترتیب بعد از دوره اول تنها پیکسل‌های دور از مرکز را در محدوده جستجو قرار داده و فواصل را محاسبه کردیم.

مشابه روش پیشنهادی ۱، برای آزمایش، از ویدئویی با قابهایی به اندازه 720×1280 پیکسل استفاده کردیم. تعداد ابرپیکسل‌ها ۱۰۰۰ و اندازه $S' = 4$ قرار دادیم. بدین ترتیب محدوده جستجوی حلقه‌ای شکل را در بدبینانه ترین حالت، طوری تنظیم کردیم که حفره جستجو تنها ۶۴ پیکسل از ۳۶۰۰ پیکسل را در بر بگیرد.

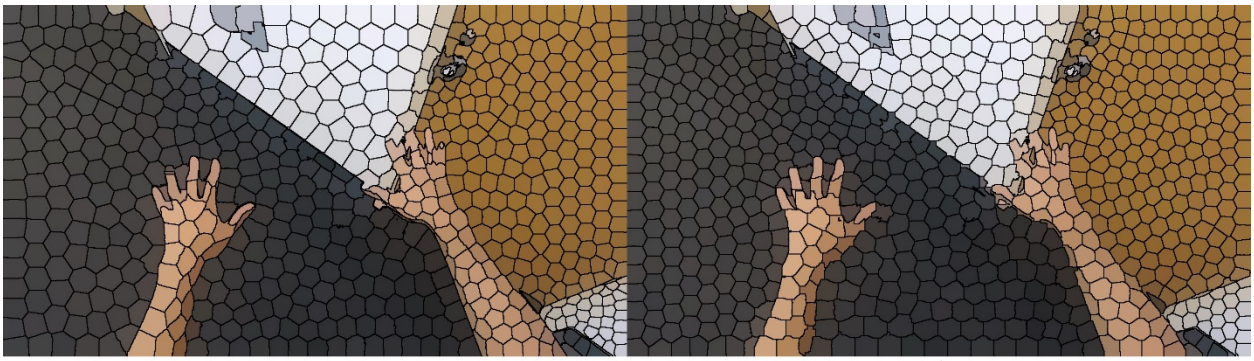
پیاده سازی این روش اما، تبعیت از مرزهای موجود در تصویر را دچار اغتشاش کرده و توازنی را که به دنبال آن بودیم بر هم می‌زند. همانطور که در شکل ۳-۱۰ مشخص است، مرزهای موجود در تصویر در مقایسه با آنچه [۶۶] ارائه می‌دهد، تبعیت قابل قبولی از مرزها نشان نمی‌دهد. بنابراین به توسعه الگوریتم و روند متفاوتی اندیشیدیم (روش پیشنهادی ۳).



شکل ۳-۸: بررسی خروجی SLIC در دوره های متفاوت. a: قاب تجزیه شده بعد از دوره اول، b: قاب تجزیه شده بعد از ۸ دوره، c: قاب تجزیه شده بعد از ۱۸ دوره، d، e و f: کادرهای بزرگنمایی شده در a، b و c، g: همپوشانی قابهای تجزیه شده a، b و c، h: کادر بزرگنمایی شده قاب h، i: نواحی به روز نشده در طی تجزیه قاب



شکل ۳-۹: محدوده جستجوی حلقه‌ای به جای محدوده جستجو مربع شکل



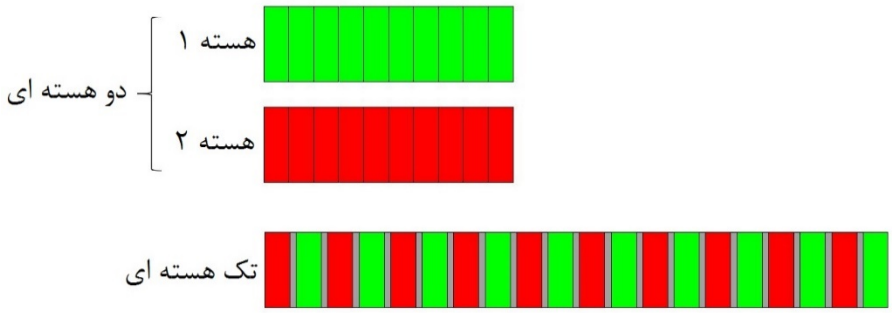
شکل ۳-۱۰: مقایسه کیفی روش پیشنهادی ۲ و الگوریتم [۶۶]. a: نتیجه الگوریتم [۶۶]، b: نتیجه الگوریتم پیشنهادی ۲

روش پیشنهادی ۳

اگر چه ایده استفاده از رایانش موازی با واحد پردازنده چندهسته‌ای توسط [۶۶] بر روی الگوریتم SLIC اعمال شده است؛ اما این نوع از رایانش به ساده‌ترین شکل ممکن و با استفاده از کتابخانه Intel Thread Building Blocks (در مورد این کتابخانه در فصل ۴ کامل بحث خواهیم کرد) تنها بر روی حلقه‌های for موجود در کد الگوریتم پیاده سازی شده است. از این رو تلاش کردیم تا الگوریتم SLIC را از پایه به صورت همروند[۱] بازنویسی کنیم. همروندی به معنی دو یا چند کار همزمان و مستقل است. زمانی که از زبان علم کامپیوتر همروندی تعریف می‌شود، منظور سامانه ای است که پردازش‌های محاسباتی را به شکل موازی و نه به ترتیب و پشت سر هم، مدیریت می‌کند. این پردازش‌ها در روند برنامه ممکن است با یکدیگر اندرکنش[۲] داشته یا به صورت مستقل محاسباتی را دنبال کنند. امروزه با فراگیر شدن واحدهای پردازنده چند هسته‌ای در رایانه‌های شخصی، همروندی الگوواره‌ای است که به صورت پیش فرض، در اجرای نرم افزارها با آن سر و کار داریم. شکل ۳-۱۱ طرحی از دو رایانه را به تصویر می‌کشد که در حال انجام دو وظیفه هستند. هر وظیفه به ۱۰ پاره وظیفه[۳] تقسیم شده است. در رایانه دو هسته ای، هر وظیفه بر روی یک هسته انجام می‌شود. از سویی دیگر رایانه تک هسته ای با مدیریت تعویض وظیفه[۴] پاره وظیفه‌ها را بازه بندی می‌کند.

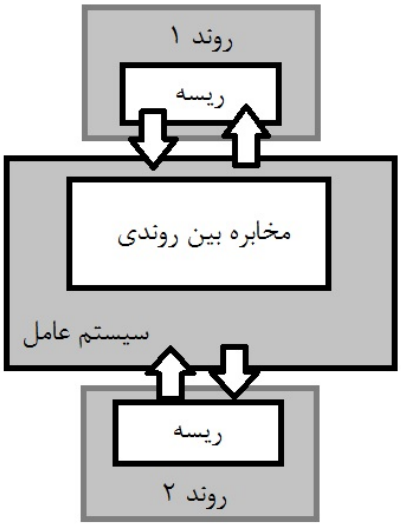
---

[1] Concurrent
[2] Interaction
[3] Task Chunk
[4] Task Switching



شکل ۳-۱۱: رایانش موازی در یک سامانه چند هسته ای و تعویض وظیفه در یک سامانه تک هسته ای

تقسیم یک الگوریتم به چندین روند تک ریسه‌ای[1] مستقل که به صورت همزمان اجرا می‌شوند، اولین رویکرد همروندسازی است (شکل ۳-۱۲). این روندهای مستقل، در نهایت می‌توانند پیغام یا داده‌ای را به یکدیگر مخابره کنند. طراحی چنین سامانه مخابره‌ای ممکن است دشوار یا پیچیده باشد. از این رو بایستی میان روندها نوعی محافظت داده پیاده سازی شود تا از دسترسی ناخواسته و یا تصادفی یک روند به داده‌های دیگر روندها جلوگیری شود.

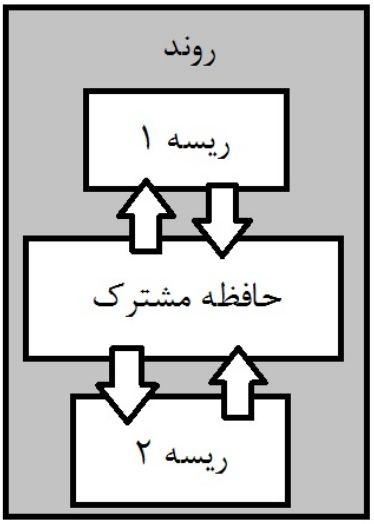


شکل ۳-۱۲: مخابره میان دو روند همزمان در دو روند تک ریسه ای

در رویکردی دیگر، همروندی به شکل اجرای چندین ریسه در یک ریسه واحد تعریف می‌شود (شکل ۳-۱۳). هر ریسه مستقل از ریسه‌های دیگر اجرا می‌شود و ممکن است خط دستوری متفاوتی را دنبال کند. تمام ریسه‌ها از یک فضای آدرسی مشترک استفاده می‌کنند. اغلب داده‌ها از تمام ریسه‌ها قابل دسترسی هستند. هم چنین،

[1] Single-Threaded

ارجاعات و اشارهگرها به متغیرها می‌توانند میان ریسه‌ها تبادل شوند. در این رویکرد به اشتراک گذاری حافظه میان ریسه‌ها طراحی پیچیده و مدیریت دشواری نیاز دارد؛ از این رو به منظور همروندسازی الگوریتم SLIC، رویکرد اول را بر می‌گزینیم.



شکل ۳-۱۳: مخابره میان دو ریسه که به شکل همزمان در یک روند واحد، رایانش می‌شوند

در شیوه پیشنهادی به منظور تقسیم وابستگی پردازشی و بهبود عملکرد الگوریتم SLIC، مسئله را از پایه همروند خواهیم ساخت. برخی از الگوریتم‌ها مستعد همروند سازی هستند. به این نوع الگوریتم‌ها عنوان «طبیعتا همروند» را نسبت می‌دهیم. غالب الگوریتم‌های ابرپیکسل مبتنی بر گرادیان نزولی، طبیعتا همروند هستند. SLIC نیز از جمله این الگوریتم‌ها است.

به منظور همروند سازی الگوریتم SLIC تصمیم گرفتیم تا هر قاب ورودی از یک ویدئو را به چهار پاره قاب با تعداد مساوی پیکسل تقسیم کنیم و سپس در ۴ روند تک ریسه‌ای، وظیفه تجزیه هر پاره قاب را به صورت موازی و مستقل به یک ریسه بسپاریم. در نهایت با بررسی مجدد خوشه‌هایی که در حاشیه‌های هر قاب قرار دارند، ۴ پاره قاب را با یکدیگر ادغام می‌کنیم. به منظور محافظت از داده‌ها، آرایه‌هایی که از آنها داده‌های قاب را می‌خوانیم و نتایج پردازش را بر آنها می‌نویسیم، به ۴ قسمت مساوی تقسیم می‌کنیم. بدین ترتیب هر ریسه، حافظه مخصوص به خود را برای خواندن و نوشتن در اختیار دارد. در مرحله نهایی، این آرایه‌ها نیز ادغام می‌شوند. خلاصه رایانش موازی در ریسه‌ها به روش پیشنهادی در الگوریتم ۳-۴ نمایان است.

الگوریتم ۳-۴: ناحیه‌بندی با ابرپیکسل SLIC بهینه (روش پیشنهادی ۳: مرحله رایانش موازی)

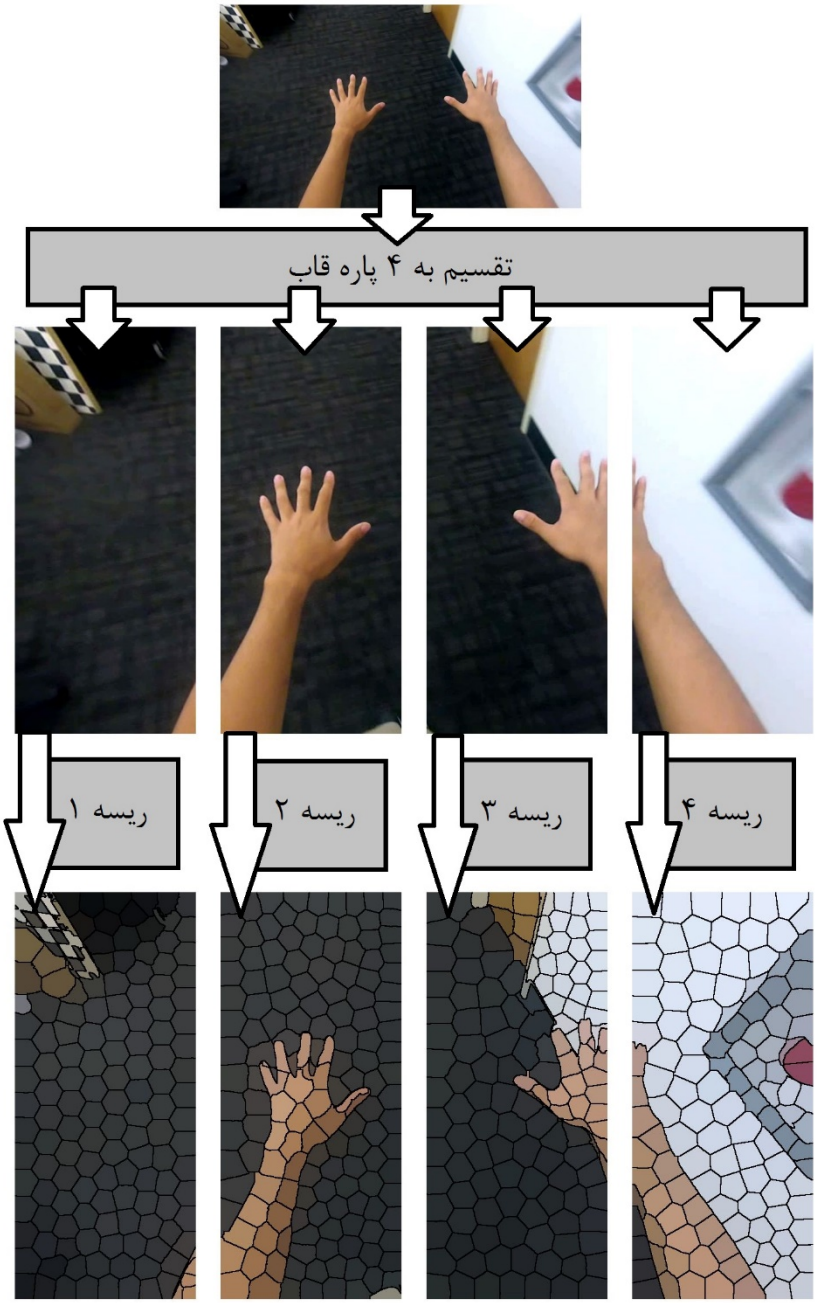
- /* ریسه j | $4 \leq j \leq 1$، ورودی: پاره قاب j | $4 \leq j \leq 1$ */

}

- /* ***مقدار دهی اولیه*** */
- با گام نمونه برداری S مراکز خوشه‌های اولیه $C_{j,k} = [l_i \ a_i \ b_i \ x_i \ y_i]^T$ مقداردهی اولیه شوند.
- مراکز خوشه‌ها به پایین ترین گرادیان در یک همسایگی ۳ در ۳ انتقال یابند.
- برای هر پیکسل i برچسب $l_j(i) = -1$ تنظیم شود.
- برای هر پیکسل i فاصله $d_j(i) = \infty$ تنظیم شود.

- **تا زمانیکه** (خطای پس ماند تمام خوشه‌ها < آستانه **یا** شماره دوره > بیشینه تعداد دوره‌ها)
  - /* ***تخصیص*** */
  - **برای هر** مرکز خوشه $C_{j,k}$:
    - **برای هر** پیکسل i در ناحیه $2S \times 2S$ حول $C_{j,k}$:
      - فاصله D، میان $C_{j,k}$ و پیکسل i محاسبه شود.
      - **اگر** $D < d_j(i)$ **آنگاه**:
        - $d_j(i) = D$ تنظیم شود.
        - $l_j(i) = k$ تنظیم شود.
  - /* ***به روزرسانی*** */
  - مراکز خوشه‌های جدید محاسبه شود.
  - خطای پس ماند E_j محاسبه شود. }

در مرحله نخست آزمایش، عملکرد پردازش موازی را ارزیابی کردیم. همانند روشهای پیشنهادی قبل، از ویدئویی با قابهای 720 × 1280 پیکسلی استفاده کردیم. هر قاب ورودی مانند آنچه در شکل ۳-۱۴ مشاهده می‌شود به ۴ پاره قاب تقسیم شده است. برای تجزیه قاب ورودی به ۱۰۰۰ ابرپیکسل، متغیر تعداد ابرپیکسل‌ها را برای هر ریسه به مقدار ۲۵۰ تنظیم کردیم. هر ریسه، الگوریتم ابرپیکسل SLIC را بر روی یکی از این ۴ پاره قاب اجرا می‌کند. نتیجه تجزیه هر پاره قاب در شکل ۳-۱۴ مشخص شده است. بر خلاف آنچه انتظار داشتیم، با توجه به جدول ۳-۳، الگوریتم پیشنهادی از نظر عملکرد نتیجه مناسبی نداشت. از این رو به توسعه مرحله ادغام نتایج نپرداختیم.



شکل ۳-۱۴: روند پیاده سازی الگوریتم SLIC به صورت همروند

الگوریتم SLIC	میانگین زمان اجرا بر روی هر قاب (میلی ثانیه)	قاب بر ثانیه
الگوریتم خام [۶۶]	۶۴,۲	۱۵,۶
الگوریتم خام پیشنهادی ۳	۸۴,۶	۱۱,۸

جدول ۳-۳: نتیجه پیاده سازی الگوریتم پیشنهادی ۳

روش پیشنهادی ۴

بهبود سرعت SLIC همانند دیگر پژوهش‌های بینایی ماشین، در پی توازنی میان سرعت و کیفیتِ ناحیه‌بندی، محقق خواهد شد. بنابراین برای سرعت بخشیدن به الگوریتم، ناگزیریم از کیفیت ناحیه‌بندی تا حدودی بکاهیم. با این همه ملاحظه خواهیم کرد، روش پیشنهادی به خوبی کیفیت ناحیه‌بندی را در مرزهای درون تصویر، مشابه کیفیت الگوریتم [۶۶] حفظ می‌کند.

به پیشنهاد [۳] در اجرای الگوریتم SLIC بر روی تصاویر ثابت، می‌توان کمی‌از خطای ناحیه‌بندی در نواحی هموار یک تصویر چشم پوشی کرد. از نظر بصری مراد از نواحی هموار، بخش‌هایی از یک قاب هستند که لبه‌های کمتری در آنها دیده می‌شود و از دید الگوریتم SLIC، نواحی هموار خوشه‌هایی از یک قاب هستند که در آنها، پیکسل‌ها از یک دوره به دوره بعد تغییر خوشه‌ای ناچیزی دارند. این خوشه‌ها را خوشه‌های غیر فعال نامیده‌ایم. با بررسی خوشه‌های غیر فعال در هنگام اجرای الگوریتم، می‌توان از پردازش‌های اضافی جلوگیری کرد.

آستانه گذاری محلی

در این بخش، الگوریتم [۶۶] را با پیاده‌سازی ایده آستانه‌گذاری محلی، از نظر سرعت بهبود می‌بخشیم و نتایج آن را از نظر سرعت و کیفیت با یکدیگر مقایسه می‌کنیم و در بخش آزمایش و نتایج، الگوریتم ارائه شده را بر روی دادگان ویدئوهای فرد منظر [۹۰] آزمایش خواهیم کرد.

همانطور که پیشتر گفته شد، ایده SLIC بهره‌گیری از نوعی k-means محلی است تا در یک همسایگی، پیکسل‌ها کوتاهترین فاصله را از مرکز خوشه مورد بررسی داشته باشند. اما باید توجه داشت که SLIC از نوعی معیار سراسری برای توقف الگوریتم استفاده می‌کند. در الگوریتم پیشنهادی معیاری محلی را برای هر خوشه تعریف کرده و از بازبینی خوشه‌ها و ناحیه‌هایی از قاب که نسبت به دوره قبل تغییر چندانی ندارند، جلوگیری خواهیم کرد. بدین ترتیب مانع از تحول خوشه‌ها در نواحی هموار تصویر می‌شویم.

معیار توقف سراسری الگوریتم SLIC در [۶۶] بدین شکل تعریف شده است:

«هنگامی‌که سطح خطای مطلوب در یک قاب برای تمام خوشه‌ها برآورده شد یا حداکثر مجاز دوره‌ها اجرا شد، الگوریتم را متوقف کن.»

اما با این معیار توقف، ناحیه‌های بزرگی از تصویر که در دوره قبل تغییر چندانی نداشته اند، در دوره جدید بازبینی شده و پیکسل‌ها به روز می‌شوند. بنابراین معیار توقف محلی جدیدی را برای هر خوشه، بدین قرار تعریف خواهیم کرد:

اگر در یک خوشه تغییر چندانی مشاهده نشد، در دوره بعد، آن خوشه نباید به روزرسانی شود.

معیار توقف سراسری نیز به صورت زیر اصلاح می‌شود:

هنگامی‌که هیچ خوشه‌ای به روز نشد یا حداکثر مجاز دوره‌ها اجرا شد یا تغییر چندانی در قاب مشاهده نشد، الگوریتم را متوقف کن.

همانطور که ملاحظه شد در معیارهای جدید، خطا را حذف کرده و شرط رسیدن به کمینه تغییرات در هر خوشه را جایگزین کرده ایم. به عبارت دیگر با چشم‌پوشی از به روزرسانی خوشه‌های غیر فعال، خطای بیشتری را تحمل خواهیم کرد . بدین ترتیب تقریبی از SLIC را پیاده‌سازی کرده و دفعات دسترسی به پیکسل‌ها را در نواحی هموار کاهش خواهیم داد. حتی اگر به تعداد دوره یکسان الگوریتم اجرا شود، تمام خوشه‌ها به روزرسانی نخواهند شد. خلاصه این شیوه پیشنهادی در الگوریتم ۳-۲ مشاهده می‌شود.

الگوریتم ۳-۲: ناحیه‌بندی با ابرپیکسل SLIC بهینه (الگوریتم پیشنهادی ۴)

- /* ***مقداردهی اولیه*** */
- با گام نمونه برداری S مراکز خوشه‌های اولیه $C_k = [l_i \ a_i \ b_i \ x_i \ y_i]^T$ مقداردهی اولیه شوند.
- مراکز خوشه‌ها به پایین ترین گرادیان در یک همسایگی ۳ در ۳ انتقال یابند.
- برای هر پیکسل i برچسب $l(i) = -1$ تنظیم شود.
- برای هر پیکسل i فاصله $d(i) = \infty$ تنظیم شود.
- برای هر خوشه C_k میزان تغییرات $c(k) = \infty$ تنظیم شود.

- **تا زمانیکه** (خوشه‌ها در حال به روز شدن هستند **یا** شماره دوره < بیشینه تعداد دوره‌ها **یا** تغییر در این دوره > کمینه تغییرات قاب)
  - /* ***تخصیص*** */
  - **برای هر** مرکز خوشه C_k :
    - **اگر** (میزان تغییرات $c(k)$ > کمینه تغییرات هر خوشه)
      - **برای هر** پیکسل i در ناحیه $2S \times 2S$ حول C_k:
        - فاصله D، میان C_k و پیکسل i محاسبه شود.
        - **اگر** $D < d(i)$ **آنگاه**:
          - $d(i) = D$ تنظیم شود.

$l(i) = k$ ◊ تنظیم شود.

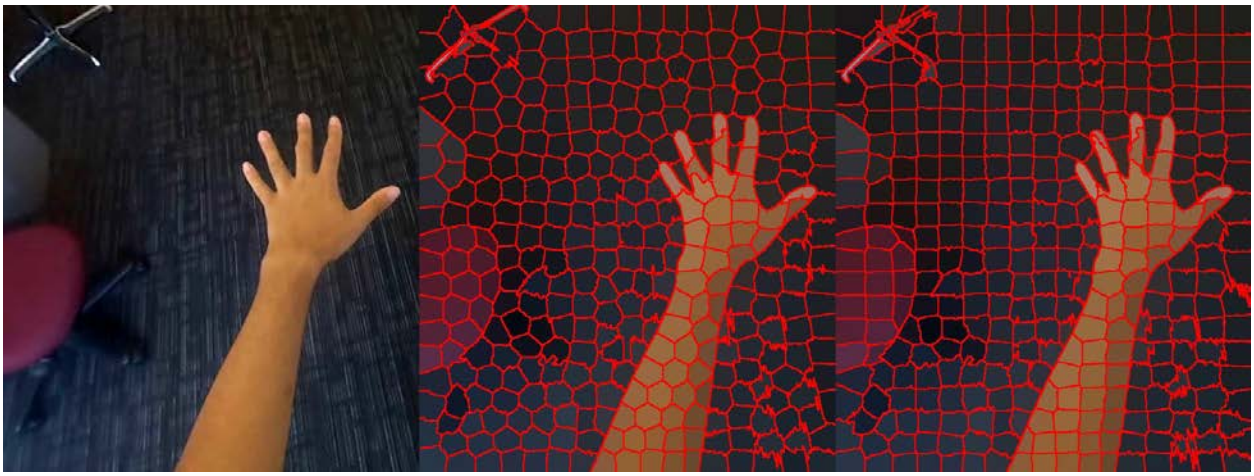
o /* ***به روزرسانی*** */

o میزان تغییرات برای هر خوشه محاسبه شود.

o مراکز خوشه‌های جدید محاسبه شود.

o خطای پس ماند E محاسبه شود.

تفاوت کیفیت ناحیه بندی برای یک تصویر نمونه در شکل ۳-۵ مشاهده می‌شود. همانطور که مشخص است، در نواحی هموار تصویر به پایداری بهتر رسیده‌ایم و تفاوت چندانی در تبعیت از مرزها دیده نمی‌شود. تفاوت کیفی دو الگوریتم را در فصل ۴ با جزئیات بررسی خواهیم کرد. از نظر عملکرد نیز طبق جدول ۳-۱ در رویکرد الگوریتم خام به ۲۳٪ بهبود سرعت دست یافته ایم. نتایج کمی‌الگوریتم پیشنهادی در فصل ۴ در رویکردهای مختلف با جزئیات بررسی خواهد شد.



شکل ۳-۱۵: مقایسه کیفی الگوریتم پیشنهادی و الگوریتم SLIC. چپ: تصویر اصلی، وسط: الگوریتم SLIC و راست: الگوریتم پیشنهادی

الگوریتم SLIC	میانگین زمان اجرا بر روی هر قاب (میلی ثانیه)	قاب بر ثانیه
الگوریتم خام [۶۶]	۶۴,۲	۱۵,۶
الگوریتم خام پیشنهادی ۴	۴۹,۴	۲۰,۲

جدول ۳-۴: بخشی از نتایج کمی‌الگوریتم پیشنهادی

بهینه سازی کد

در روند دست‌یابی به راه حل مسئله، به منظور دست یابی به عملکرد بهینه الگوریتم پیشنهادی، کد برنامه نویسی شده الگوریتم نیز دستخوش تغییراتی شد. در ادامه به بررسی بهینه‌سازی‌های اعمال شده و بررسی نتایج آن می‌پردازیم.

مقدار دهی اولیه بردارها در زبان برنامه نویسی C++:

در زبان برنامه نویسی C++ به طور معمول یک بردار پویا (vector) با دستور زیر مقداردهی (desiredValue) می‌شود:

```
vector.push_back(desiredValue);
```

قالب push_back در هر بار فراخوانی اندازه حامل (بردار) را یک واحد افزایش داده و به طور خودکار حافظه را بازنویسی می‌کند. بنابراین برای مقداردهی یک بردار در یک روند تکراری استفاده از قالب push_back بهینه به نظر نمی‌رسد. از این رو از قالب std::fill بهره می‌گیریم:

```
std::fill(vector.begin(), vector.end(), desiredValue);
```

قالب std::fill تمام المانهای موجود در حامل (vector) را با مقدار مورد نظر (desiredValue) پر می‌کند. بنابراین قبل از استفاده از این دستور بایستی اندازه حامل تعریف شود:

```
vector.resize(desiredSize);
```

از این رو نیازی به بازنویسی حافظه به صورت تکراری نیست. در آزمایشی، برداری به اندازه 10^6 را با مقدار -1 پر کردیم. نتایج آزمایش در جدول ۳-۲ مشخص است. همانطور که مشاهده می‌شود نتیجه آزمایش، مقداردهی با قالب std::fill را پرسرعت گزارش می‌دهد. بنابراین بهینه است در الگوریتمی مانند SLIC که به مقداردهی اولیه تعداد زیادی بردار در روندی تکراری نیاز داریم، از قالب std::fill استفاده کنیم. اعمال std::fill در الگوریتم SLIC بهبود سرعتی به اندازه ۳-۲ میلی ثانیه (به طور میانگین) در اجرای الگوریتم به ارمغان می‌آورد.

قالب مقدار دهی	زمان اجرا (میلی ثانیه)
.push_back()	۱۸
std::fill()	۵

جدول ۳-۵: مقایسه سرعت عملکرد قالبهای مقداردهی به یک حامل در برنامه نویسی C++

دسترسی به پیکسل‌ها در کتابخانه OpenCV:

به طور معمول دسترسی به یک پیکسل از تصویر با استفاده از کتابخانه OpenCV توسط دستور زیر انجام می‌گیرد:

```
pixel = image.at<Vec3b>(y, x);
```

قالب at<>(y, x). نخست مختصات x و y را آزمایش کرده و در صورت قرار گرفتن در محدوده تعریف شده، به آن مختصات در بردار دسترسی پیدا میکند. بنابراین زمانیکه در برنامه نویسی، محدوده بردار را بدانیم و متغیرها را در همان محدوده تعریف کنیم، مرحله آزمایش قالب at<>. هزینه‌بَر به نظر می‌رسد. کتابخانه پردازش تصویر OpenCV نوعی دیگر از دسترسی به پیکسل‌ها توسط اشاره‌گر را در اختیار کاربر قرار می‌دهد:

```
Vec3b* pixelData = image.ptr<Vec3b>(j);
```

بدین ترتیب با استفاده از اشاره‌گر pixelData به آدرس ردیف j اُم از پیکسل‌های تصویر دسترسی داریم. برای دسترسی به پیکسل i اُم از این ردیف کافی است از دستور زیر استفاده کنیم:

```
pixel_i = pixelData[i];
```

در این رویکرد برای دسترسی به پیکسل‌های یک ردیف، نیازی به فراخوانی ردیف مورد نظر در هر بار اعلام دسترسی نیست. از این رو، با کم کردن تعداد فراخوانی، عملکرد دسترسی به مقادیر پیکسل‌ها بهینه می‌شود. بر اساس ادعای [۹۱] رویکرد دسترسی به پیکسل‌ها توسط اشاره‌گر، بهبودی به اندازه ۴۳ میلی‌ثانیه در سرعت دارد. آزمایش این رویکرد در الگوریتم SLIC به منظور بهینه‌سازی کد، نتیجه‌ای در برنداشت و بهبودی در سرعت عملکرد SLIC مشاهده نشد.

۴. فصل چهارم: آزمایش‌ها و نتایج

در این بخش سکویی که از آن برای ارزیابی روشهای پیشنهادی استفاده کردیم را از نظر سخت افزاری و نرم افزاری بررسی خواهیم کرد. کتابخانه‌هایی که از آنها بهره گرفته‌ایم را معرفی کرده و در نهایت نتایج ارزیابی را با جزئیات به شکل کیفی و کمی‌تفسیر خواهیم کرد.

سکو و چارچوب ارزیابی

سکوی اجرای الگوریتم ابر پیکسل SLIC بهینه، رایانه ای با پیکربندی سخت افزاری مشخص شده در جدول ۴-۱ است.

پردازنده	**AMD FX-8350 8x Core x64 based @ 4.00GHz**
حافظه	8GB RAM
معماری	Windows 8.1 x64 Operating System

جدول ۴-۱: سکوی ارزیابی الگوریتم پیشنهادی

محیط یکپارچه توسعه نرم افزار[1] نیز Visual Studio 2013 Ultimate و زبان برنامه نویسی C++ است.

کتابخانه‌ها

OpenCV 2.4.10 به منظور پردازش بر روی تصاویر از کتابخانه متن باز OpenCV بهره گرفتیم. نظارت بر ورود و خروج سندهای تصویری و ویدئویی، تغییر فضای رنگ، دسترسی به پیکسل‌های تصویر و اعمال تغییرات بر روی آنها، توسط توابع موجود در این کتابخانه صورت گرفته است.

Intel Thread Building Blocks (TBB) این کتابخانه چارچوبی غنی در رایانش موازی به یک برنامه نوشته شده توسط C++ معرفی می‌کند. TBB با بهره گیری از پردازنده‌های چند هسته‌ای و بدون نیاز به تخصص در رایانش موازی، عملکرد بهینه‌ای را به ارمغان می‌آورد. توابع و قالب‌های موجود در این کتابخانه مبتنی بر موازی سازی وظیفه‌ها نوشته شده اند. برخلاف عنوان این کتابخانه، TBB قابل پیاده‌سازی بر روی تمامی پردازنده‌ها از جمله AMD است. برای استفاده از این کتابخانه کافی است وظیفه‌ای تعریف شود. سپس کتابخانه به صورت خودکار وظیفه تعربف شده توسط کاربر را بر روی ریسه‌هایی که خود ایجاد می‌کند، نگاشت می‌کند.

مزیت استفاده از موازی‌سازی مبتنی بر وظیفه نسبت به ریسه‌بندی این است که، وظیفه‌ها بار محاسباتی کمتری دارند. بر روی یک سیستم عامل Linux شروع و خاتمه یک وظیفه ۱۸ برابر سریعتر از شروع و خاتمه یک ریسه

[1] Integrated Development Environment (IDE)

است. بر روی سیستم عامل ویندوز، نرخ به ۱۰۰ برابر می‌رسد [۹۲]. TBB با ارائه قالب parallel_for تمام حلقه‌های تکراری for را به صورت موازی اجرا می‌کند. به عنوان مثال به جای حلقه تکراری زیر

```
for (int centreIndex = 0; centreIndex < clustersNumber; centreIndex++)
```

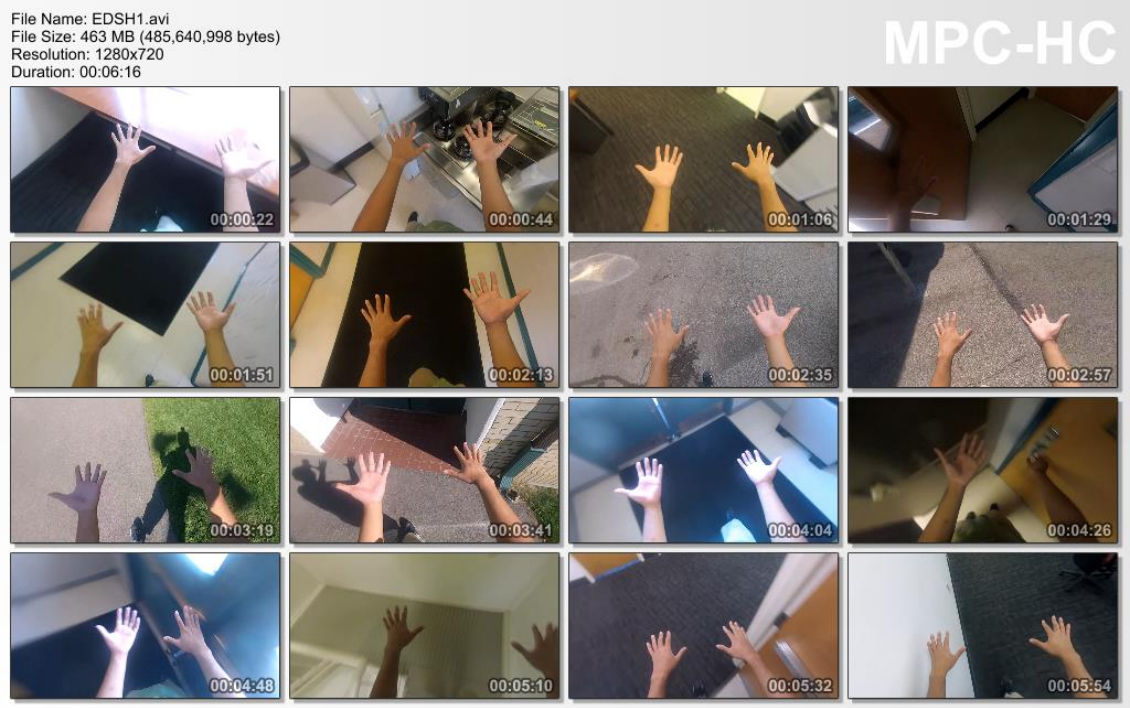
از قالب TBB زیر استفاده می‌کنیم:

```
tbb::parallel_for<unsigned>(0, clustersNumber, 1, [=](unsigned centreIndex)
```

BOOST این کتابخانه مجموعه‌ای از توابع بهینه شده برای استفاده در برنامه‌های C++ است. از این کتابخانه تنها از تابع زمان‌گیری آن استفاده کردیم تا با عملکردی سریعتر، اطمینان حاصل کنیم که عمل زمان‌گیری و ارزیابی سرعت الگوریتم با دقت انجام می‌گیرد.

دادگان

دادگان فرد منظر دانشگاه Carnegie Mellon University موسوم به EDSH [۹۰]، متشکل از سه ویدئو که اساسا به منظور جداسازی دست ایجاد شده است، مبنای آزمایش و ارزیابی‌های این پایان‌نامه است. این دادگان در شرایط نوری گوناگون تصویربرداری شده است. قاب‌های این ویدئو سطح تفکیکی معادل 720 × 1280 پیکسل دارند. نمونه ای از قابهای ویدئو در شکل ۴-۱ مشاهده می‌شود.



شکل ۴-۱: نمونه از قابهای دادگان مورد آزمایش

آزمایش‌ها و تفسیر نتایج

تمامی‌آزمایشها به منظور مقایسه بر روی ویدئوها یا قاب‌های استخراج شده از دادگان [۹۰] انجام شده اند. به دلیل اینکه [۶۶] به روز ترین و تنها پژوهش در زمینه بهبود الگوریتم SLIC در چارچوب ویدئوهای فردمنظر است، نتایج آزمایشها را با نتایج [۶۶] مقایسه خواهیم کرد. اندازه قاب‌ها 1280×720 پیکسل است. هم چنین پارامترهای الگوریتم SLIC طبق جدول ۴-۲ ترتیب یافته اند.

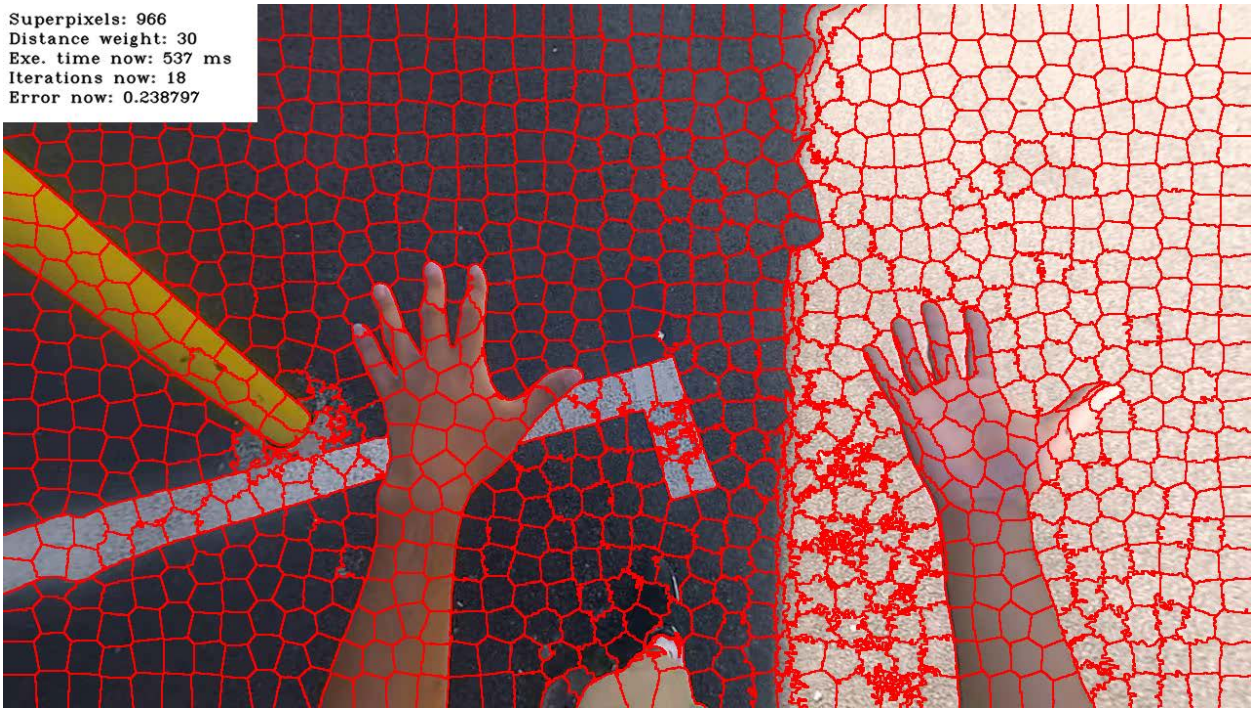
عنوان پارامتر	الگوریتم [۶۶]	الگوریتم پیشنهادی ۴
تعداد پیکسل‌های موجود در هر قاب (N)	$1280 \times 720 = 921600$	$1280 \times 720 = 921600$
تعداد ابرپیکسل‌ها	$k = 1000$	$k = 1000$
اندازه ابرپیکسل‌ها ($S = \sqrt{N/k}$)	$S = 30$	$S = 30$
تراکم ابرپیکسل‌ها	$m = 30$	$m = 30$
خطای مطلوب	$E = 0.25$	-
کمینه تغییرات محلی در خوشه	-	10
کمینه تغییرات سراسری در قاب	-	$10 \times k$

جدول ۴-۲: پارامترهای SLIC

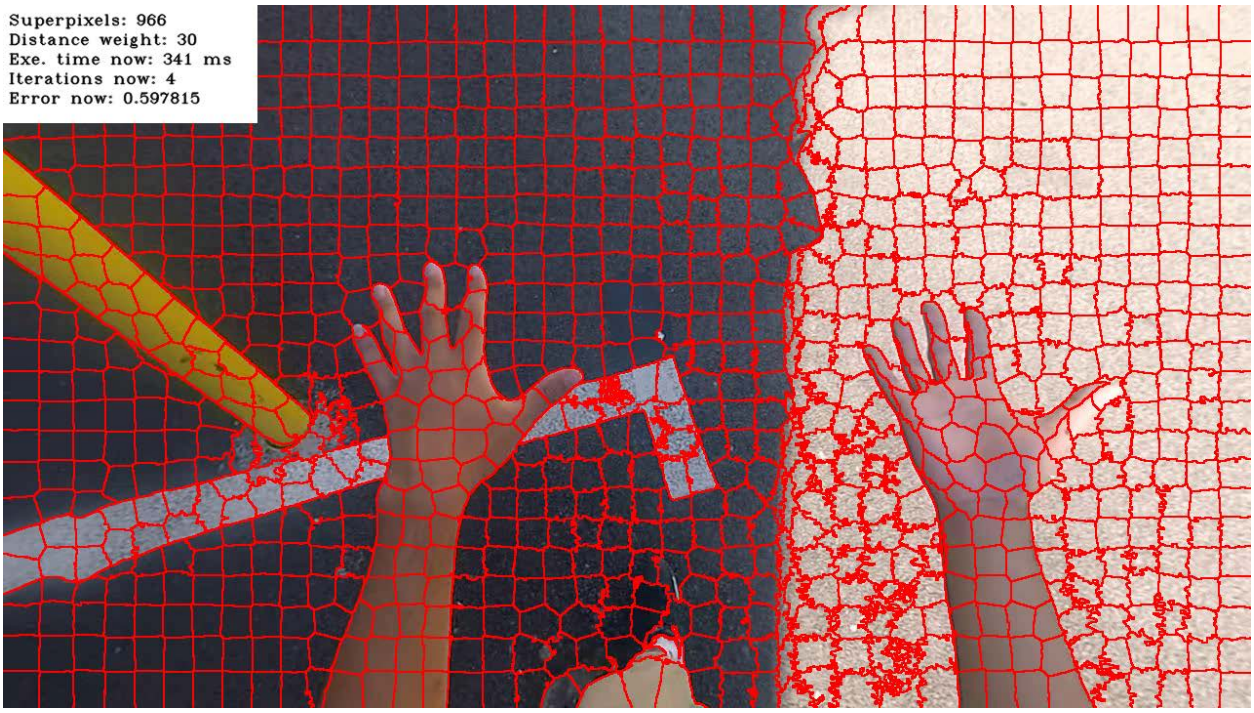
آزمایش بر روی یک قاب

ابتدا با آزمایش بر روی یک قاب، نتایج روش پیشنهادی را با نتایج [۶۶] مقایسه خواهیم کرد. این آزمایش به ما امکان می‌دهد تا با دقت و جزئی نگری افزون تر، کیفیت و عملکرد اجرای الگوریتم را با شیوه‌های پیشین مقایسه کنیم. قاب‌ها به طور مستقیم از دادگان [۹۰] استخراج شده اند. می‌توان تفاوت کیفیت ناحیه‌بندی را برای یک قاب در شکل ۴-۲ و شکل ۴-۳ مشاهده کرد. با مقایسه نتایج ملاحظه می‌شود که سرعت بهبود یافته و کیفیت ناحیه‌بندی نیز تغییر چندانی نداشته است.

معیار توقف محلی و جلوگیری از تحول مرزهای ابرپیکسل‌ها در نواحی هموار سبب شده است تا خوشه‌ها شبکه نظام یافته تری داشته باشند. این پدیده در شکل ۴-۳ به صورت مرزهای تقریبا افقی و یا عمودی در نواحی هموار ظاهر شده است. از آنجا که تعداد کمتری از ابرپیکسل‌ها در نواحی هموار تصویر، نظم مربع گونه خود را از دست داده اند، می‌توان مدعی شد که به جداسازی پایدار تری نیز دست یافته ایم.



شکل ۴-۲: SLIC پیاده سازی شده توسط [۶۶] همراه با پس پردازش اجبار همبستگی



شکل ۴-۳: SLIC به روش پیشنهادی ۴ همراه با پس پردازش اجبار همبستگی

با مقایسه شکل ۴-۲ و شکل ۴-۳ در می‌یابیم که خطای خوشه‌بندی به دلیل تحمل خطای بیشتر در نواحی هموار، افزایش یافته است. با این حال مرزهای میان اشیاء موجود در صحنه، با همان کیفیت الگوریتم [۶۶] جداسازی شده اند. الگوریتم پیشنهادی با تکیه بر شرط سراسری، تعداد دوره‌ها را به اندازه ۱۴ دوره (۷۸ درصد کاهش) کاهش داده است. از سویی موفق شده ایم تا زمان اجرای الگوریتم را حدود ۲۰۰ میلی ثانیه کاهش دهیم که بیانگر بهبود ۳۶ درصدی در سرعت اجرای برنامه است.

آزمایش بر روی ویدئو

در این بخش از دادگان [۹۰] مشتمل بر ویدئوهای فردمنظر، ویدئوهایی را استخراج کرده، آزمایش را بر روی آنها انجام می‌دهیم. ویدئوی EDSH1.avi تقریبا ۶ دقیقه و دارای ۱۱۲۹۰ قاب به اندازه 720×1280 پیکسل است. جدول ۴-۳ نتایج عددی پردازش بر روی ۱۱۲۹۰ قاب را نمایش می‌دهد. اجرای الگوریتم SLIC با راهبرد بیزین به شیوه پیشنهاد شده، سرعت عملکرد الگوریتم را به اندازه ۱۴٫۸ میلی ثانیه (۲۳ درصد بهبود) افزایش داده است. این بهبود ما را به تعداد ۴٫۶ قاب، به پردازش بلادرنگ نزدیکتر کرده است.

الگوریتم SLIC	میانگین زمان اجرا بر روی هر قاب (میلی ثانیه)	قاب بر ثانیه
الگوریتم خام [۶۶]	۶۴٫۲	۱۵٫۶
الگوریتم خام پیشنهادی ۴	۴۹٫۴	۲۰٫۲
الگوریتم [۶۶] همراه با تزریق نویز	۱۷۱٫۷	۵٫۸
الگوریتم پیشنهادی ۴ همراه با تزریق نویز	۸۳٫۷	۱۱٫۹
الگوریتم [۶۶] همراه با تزریق میان قابی	۱۹۱٫۲	۵٫۲
الگوریتم پیشنهادی ۴ همراه با تزریق میان قابی	۶۷٫۲	۱۴٫۹

جدول ۴-۳: نتایج عددی

سپس الگوریتم را با اضافه کردن نویز به قاب‌ها اجرا می‌کنیم. نویز، با توزیع گوسی $N(\mu = 0, \sigma = \frac{S}{5})$ تزریق می‌کنیم. همانطور که ملاحظه می‌شود، به دلیل افزایش بار محاسباتی، سرعت عملکرد در هر دو شیوه، کاهش قابل توجهی داشت. با این حال، موفق شدیم سرعت عملکرد در این چارچوب را، به اندازه ۸۸ میلی ثانیه (۵۱ درصد بهبود) افزایش داده و به تعداد ۶٫۱ قاب به پردازش بلادرنگ نزدیکتر شویم.

سرانجام آزمایشی با تزریق میان قابی با نرخ ۱:۳۰ (هر ۳۰ قاب، یک قاب با شبکه مراکز خوشه‌های منظم) انجام دادیم. در این آزمایش نیز بار محاسباتی افزایش یافت و با کاهش سرعت عملکرد نسبت به الگوی خام الگوریتم

مواجه شدیم. اما اجرای الگوریتم به شیوه پیشنهاد شده، افزایش سرعت عملکردی، به اندازه ۱۲۴ میلی ثانیه (۶۴ درصد بهبود) را به ارمغان آورد. از دیگر سوی، به تعداد ۹,۷ قاب به پردازش بلادرنگ نزدیک شدیم. با بررسی نتایج ملاحظه می‌گردد بیشترین میزان بهبود، در ارتباط با شیوه پیشنهادی ۴ همراه با تزریق میان قابی است. اما الگوریتم خام پیشنهادی از نظر سرعت عملکرد، نزدیکترین شیوه به پردازش بلادرنگ است.

۵. فصل پنجم: جمع بندی

در این پایان نامه، شیوه‌های متفاوت از اجرای الگوریتم ابرپیکسل را بر روی ویدئوهای فرد منظر ارائه داده و شیوه ای با عملکرد بهینه را برگزیدیم. با ارزیابی معیارهای مطلوب یک ابرپیکسل و هم چنین به دلیل برتری کیفی و کمّی نسبت به سایر الگوریتمهای ابرپیکسلی، الگوریتم SLIC را که اقتباسی از الگوریتم خوشه‌یابی k-means است برای این هدف انتخاب کردیم. SLIC روشی مبتنی بر گرادیان صعودی، با خوشه‌یابی پیکسل‌های موجود در تصویر در همسایگی های محلی و تجزیه تصویر به ابرپیکسل‌ها، سطح بالاتری از تصویر را در اختیار قرار می دهد تا سرعت الگوریتمهای پردازشی پسین خود را افزایش دهد.

به منظور بهینه سازی، رویکردهای الگوریتمی و نرم افزاری را ارزیابی کردیم. در شیوه اول (رویکردی الگوریتمی)، با الهام از نامساوی مثلثی و کاهش تعداد دسترسی به پیکسلها و محاسبه فاصله میان آنها در روند اجرای SLIC، برای سرعت بخشیدن به SLIC کوشیدیم؛ اما بر خلاف انتظار به دلیل بررسی مؤلفه‌های جدید معرفی شده در الگوریتم، بار محاسباتی افزایش یافت.

در رویکرد الگوریتمی دیگر، با هدف کاهش تعداد پیکسل‌های مورد بررسی در یک همسایگی، محدوده جستجوی حلقه‌ای شکل را ارزیابی کردیم. بدین ترتیب از بررسی پیکسل‌هایی که در طول تکرار اجرای SLIC بر روی یک قاب ویدئو خوشه‌هایشان به روزرسانی نمی‌شود، جلوگیری کردیم. با پیاده سازی این رویکرد اما، کیفیت ناحیه‌بندی کاهش یافته و تبعیت از مرزهای موجود در تصویر از بین رفت.

در رویکردی نرم افزاری، حل مسئله SLIC را به صورت همروند پیشنهاد دادیم. قاب ورودی را به چهار پاره قاب تقسیم کرده و در ریسه های مستقل، الگوریتم SLIC را بر روی هر پاره قاب پیاده سازی کردیم. اما در این رویکرد نیز با افزایش بار محاسباتی روبرو شدیم. شروع و پایان هر ریسه و انتظار برای پایان اجرای SLIC بر روی هر پاره قاب در پردازش یک قاب ورودی، سرعت عملکرد الگوریتم SLIC را کاهش داد.

در نهایت، در شیوه چهارم با پیشنهاد تغییراتی در الگوریتم ابرپیکسل SLIC و ایجاد توازنی میان معیارهای سرعت و کیفیت جداسازی، عملکرد الگوریتم SLIC را بر روی ویدئوهای فردمنظر بهبود بخشیدیم و موفق شدیم کیفیت جداسازی مرزهای موجود در تصویر را به خوبی حفظ نماییم. الگوریتم پیشنهادی با سرعتی نزدیکتر به پردازش بلادرنگ عمل می‌نماید.

بهبود عملکرد الگوریتم با تعریف معیار محلی جدید، برای توقف به روزرسانی هر خوشه و اصلاح معیار سراسری برای توقف الگوریتم حاصل شد. در الگوریتم SLIC تنها یک معیار سراسری برای توقف الگوریتم در نظر گرفته شده است. با تعریف معیار توقف محلی جدیدی، از به روزرسانی خوشه‌هایی که در آنها، پیکسل‌ها تغییر خوشه اندکی دارند جلوگیری کردیم. بنابراین از خطای ناحیه بندی در نواحی هموار تصویر چشم پوشی کردیم تا به سرعت عملکرد بیشتری دست یابیم. معیار توقف سراسری را همگرایی به بیشینه تعداد دوره ها و یا به روز رسانی

نشدن هیچکدام از خوشه‌ها و یا مشاهده تغییرات خوشه اندک در کل یک قاب تعریف کردیم. با آزمایش بر روی دادگان تصاویر فرد منظری که در اختیار داشتیم متوجه شدیم که الگوریتم پیشنهادی با معیارهای توقف جدید، ناحیه‌بندی پایدارتری را به ارمغان می آورد. این پایداری در نواحی هموار تصویر به صورت مرزهایی تقریبا افقی و عمودی ظاهر شدند. در نهایت عملکرد الگوریتم پیشنهادی را با راهبردهای گوناگون بر روی دادگان معرفی شده ارزیابی کردیم و دیدیم که سرعت عملکرد الگوریتم SLIC بهینه شده به ۲۰,۲ قاب در ثانیه رسید که تقریبا بلادرنگ محسوب می‌شود. این بهبود عملکرد، با هدف قابلیت انتقال بر روی ابزارک‌های پوشیدنی به صورت نرم افزاری ارائه شد.

سعی بر آن داشتیم تا با مهاجرت به چارچوب برنامه نویسی مبتنی بر واحد پردازنده گرافیکی OpenCL با قابلیت انتقال بر روی ابزارکهای پوشیدنی، عملکرد الگوریتم SLIC را سریعتر از آنچه به دست آوردیم، بهبود دهیم؛ اما محدودیت زمانی مانع از توسعه در این چارچوب شد. بنابراین توسعه SLIC در چارچوب OpenCL را به عنوان پژوهشی در آینده پیشنهاد می‌دهیم.

بهبود کیفیت ناحیه بندی الگوریتم نیز به عنوان رویکردی دیگر در بهینه سازی SLIC در چارچوب فرد منظر، پیشنهاد می‌شود. از این رو می‌توان با پیاده‌سازی رویه ای، مانع از تشکیل مصنوعات ناخواسته در رویکرد خام الگوریتم شد.

به عنوان پژوهشی در آینده، دیگر الگوریتم‌های ابرپیکسل معرفی شده در این پایان نامه را بر روی ویدئوهای فرد منظر می‌توان ارزیابی کرد و حتی رویکردی متفاوت از آنها ارائه داد. در بررسی‌های انجام شده تبدیل فضای رنگ RGB به LAB نیز می‌تواند دستخوش تغییرات شود تا عملکردی بهینه تر از ابرپیکسل SLIC حاصل شود.

مراجع

[1] R. Xiaofeng and G. Chunhui, "Figure-ground segmentation improves handled object recognition in egocentric video," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, 2010, pp. 3137-3144.

[2] X. Ren and J. Malik, "Learning a classification model for segmentation," *Proceedings Ninth IEEE International Conference on Computer Vision,* 2003.

[3] P. Neubert and P. Protzel, "Compact Watershed and Preemptive SLIC: On Improving Trade-offs of Superpixel Segmentation Algorithms," in *2014 22nd International Conference on Pattern Recognition*, ed: IEEE, 2014, pp. 996-1001.

[4] P. Mehrani and O. Veksler, "Saliency segmentation based on learning and graph cut refinement," in *British Machine Vision Conference (BMVC)*, ed, 2010, pp. 1-12.

[5] J. Bescós, F. Navarro, and M. Escudero-Viñolo, "SP-SIFT: enhancing SIFT discrimination via super-pixel-based foreground–background segregation," *Electronics Letters,* vol. 50, pp. 272-274, 2014.

[6] C. Pantofaru, C. Schmid, and M. Hebert, "Object recognition by integrating multiple image segmentations," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 5304 LNCS, ed, 2008, pp. 481-494.

[7] B. Fulkerson, A. Vedaldi, and S. Soatto, "Class segmentation and object localization with superpixel neighborhoods," in *Computer Vision, 2009 IEEE 12th International Conference on*, ed: IEEE, 2009, pp. 670-677.

[8] A. P. Moore, S. J. D. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, ed: IEEE, 2008, pp. 1-8.

[9] S. Wang, H. Lu, F. Yang, and M.-H. Yang, "Superpixel tracking," in *2011 International Conference on Computer Vision*, ed: IEEE, 2011, pp. 1323-1330.

[10] P. Z. P. Zhang, E. E. Milios, and J. Gu, "Graph-based Automatic Consistent Image Mosaicking," in *2004 IEEE International Conference on Robotics and Biomimetics*, ed: IEEE, 2004, pp. 558-563.

[11] P. Neubert, N. Sunderhauf, and P. Protzel, "Appearance change prediction for long-term navigation across seasons," in *2013 European Conference on Mobile Robots, ECMR 2013 - Conference Proceedings*, ed: IEEE, 2013, pp. 198-203.

[12] Y. Yang, S. Hallman, D. Ramanan, and C. Fowlkes, "Layered object detection for multi-class segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2010, pp. 3113-3120.

[13] C. L. Zitnick and S. B. Kang, "Stereo for image-based rendering using image over-segmentation," *International Journal of Computer Vision,* vol. 75, pp. 49-65, 2007.

[14] C.-J. Chong, W.-H. Tan, Y. C. Chang, M. F. N. Batcha, and E. Karuppiah, "Visual based fall detection with reduced complexity horprasert segmentation using superpixel," ed, 2015, pp. 462-467.

[15] G. Mori, "Guiding model search using segmentation," in *Proceedings of the IEEE International Conference on Computer Vision* vol. II, ed, 2005, pp. 1417-1423.

[16] Y. Xie, H. Lu, and M.-H. Yang, "Bayesian saliency via low and mid level cues.," *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society,* vol. 22, pp. 1689-98, 2013.

[17] J. Chang, D. Wei, and J. W. Fisher, "A video representation using temporal superpixels," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed: IEEE, 2013, pp. 2051-2058.

[18] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based image segmentation," *International Journal of Computer Vision,* vol. 59, pp. 167-181, 2004.

[19] P. Morerio, L. Marcenaro, and C. S. Regazzoni, "A generative superpixel method," in *Information Fusion (FUSION), 2014 17th International Conference on*, 2014, pp. 1-7.

[20] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, and K. Siddiqi, "TurboPixels: Fast superpixels using geometric flows," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 31, pp. 2290-2297, 2009.

[21] J. Shi and J. Malik, "Normalized Cuts and Image Segmentation," *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE,* vol. 22, pp. 888-905, 2000.

[22] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 13, pp. 583-598, 1991.

[23] A. Vedaldi and S. Soatto, "Quick shift and kernel methods for mode seeking," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 5305 LNCS, ed, 2008, pp. 705-718.

[24] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods.," *IEEE transactions on pattern analysis and machine intelligence,* vol. 34, pp. 2274-82, 2012.

[25] X. Ren, M. Philipose, and Xiaofengren, "Egocentric recognition of handled objects: Benchmark and analysis," in *2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009*, ed, 2009, pp. 49-56.

[26] D. G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the Seventh IEEE International Conference on Computer Vision,* vol. 2, 1999.

[27] T. Brox, N. Papenberg, and J. Weickert, "High Accuracy Optical Flow Estimation Based on a Theory for Warping," *Computer Vision - ECCV 2004,* vol. 4, pp. 25-36, 2004.

[28] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2011, pp. 3281-3288.

[29] Y. Lin, G. Hua, and P. Mordohai, "Egocentric Object Recognition Leveraging the 3D Shape of the Grasping Hand," in *Computer Vision - ECCV 2014 Workshops*. vol. 8927, L. Agapito, M. M. Bronstein, and C. Rother, Eds., ed: Springer International Publishing, 2015, pp. 746-762.

[30] E. H. Spriggs, F. De La Torre, and M. Hebert, "Temporal segmentation and activity classification from first-person sensing," in *2009 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2009*, ed, 2009, pp. 17-24.

[31] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International Journal of Computer Vision,* vol. 42, pp. 145-175, 2001.

[32] A. Fathi, A. Farhadi, and J. M. Rehg, "Understanding egocentric activities," in *Proceedings of the IEEE International Conference on Computer Vision*, ed, 2011, pp. 407-414.

[33] A. Fathi, Y. Li, and J. M. Rehg, "Learning to recognize daily actions using gaze," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 7572 LNCS, ed, 2012, pp. 314-327.

[34] M. F. Land and M. Hayhoe, "In what ways do eye movements contribute to everyday activities?," in *Vision Research* vol. 41, ed, 2001, pp. 3559-3565.

[35] A. Fathi and J. M. Rehg, "Modeling actions through state changes," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2013, pp. 2579-2586.

[36] M. S. Ryoo and L. Matthies, "First-person activity recognition: What are they doing to me?," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2013, pp. 2730-2737.

[37] K. M. Kitani, T. Okabe, Y. Sato, and A. Sugimoto, "Fast unsupervised ego-action learning for first-person sports videos," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2011, pp. 3241-3248.

[38] L. Wang and D. B. Dunson, "Fast Bayesian Inference in Dirichlet Process Mixture Models," in *Journal of Computational and Graphical Statistics* vol. 20, ed, 2011, pp. 196-216.

[39] S. Hodges, L. Williams, E. Berry, S. Izadi, J. Srinivasan, A. Butler*, et al.*, "SenseCam: A Retrospective Memory Aid," in *UbiComp 2006: Ubiquitous Computing*. vol. 4206, P. Dourish and A. Friday, Eds., ed: Springer Berlin Heidelberg, 2006, pp. 177-193.

[40] A. R. Doherty, D. Byrne, A. F. Smeaton, G. J. F. Jones, and M. Hughes, "Investigating keyframe selection methods in the novel domain of passively captured visual lifelogs," in *Proceedings of the 2008 international conference on Contentbased image and video retrieval CIVR 08*, ed: ACM Press, 2008, pp. 259-268.

[41] Y. J. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2012, pp. 1346-1353.

[42] J. Carreira and C. Sminchisescu, "CPMC: Automatic object segmentation using constrained parametric min-cuts," in *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 34, ed: IEEE, 2012, pp. 1312-1328.

[43] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001,* vol. 1, 2001.

[44] Z. Lu and K. Grauman, "Story-driven summarization for egocentric video," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2013, pp. 2714-2721.

[45] F. Crete, T. Dolmiere, P. Ladret, and M. Nicolas, "The Blur Effect: Perception and Estimation with a New No-Reference Perceptual Blur Metric," in *Vol. 6492, Human Vision and Electronic Imaging XII* vol. 6492, B. E. Rogowitz, T. N. Pappas, and S. J. Daly, Eds., ed: SPIE, 2007, pp. 1-11.

[46] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2012, pp. 2847-2854.

[47] T. Cour, F. Bénézit, and J. Shi, "Spectral segmentation with multiscale graph decomposition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition* vol. 2, ed, 2005, pp. 1124-1131.

[48] S. Avidan and A. Shamir, "Seam carving for content-aware image resizing," in *ACM Transactions on Graphics* vol. 26, ed, 2007, p. 10.

[49] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 6315 LNCS, ed, 2010, pp. 211-224.

[50] V. Kwatra, A. Schodl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," *ACM Transactions on Graphics,* vol. 22, pp. 277-286, 2003.

[51] M. Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, ed, 2011, pp. 2097-2104.

[52] Y. Zhang, R. Hartley, J. Mashford, and S. Burn, "Superpixels via pseudo-Boolean optimization," in *Proceedings of the IEEE International Conference on Computer Vision*, ed, 2011, pp. 1387-1394.

[53] C. Conrad, M. Mertz, and R. Mester, "Contour-Relaxed Superpixels," in *Energy Minimization Methods in Computer Vision and Pattern Recognition*. vol. 8081, A. Heyden, F. Kahl, C. Olsson, M. Oskarsson, and X.-C. Tai, Eds., ed: Springer Berlin Heidelberg, 2013, pp. 280-293.

[54] D. Tang, H. Fu, and X. Cao, "Topology Preserved Regular Superpixel," in *2012 IEEE International Conference on Multimedia and Expo*, ed: IEEE, 2012, pp. 765-768.

[55] C. C. E. Leiserson, R. R. L. Rivest, C. Stein, and T. H. Cormen, "Introduction to Algorithms, Third Edition," *BOOK,* p. 1312, 2009.

[56] D. Comaniciu and P. Meer, "Mean shift: a robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 24, 2002.

[57] H. Fu, X. Cao, D. Tang, Y. Han, and D. Xu, "Regularity Preserved Superpixels and Supervoxels," *IEEE Transactions on Multimedia,* vol. 16, pp. 1165-1175, 2014.

[58] J. Papon, A. Abramov, M. Schoeler, and F. Worgotter, "Voxel cloud connectivity segmentation - Supervoxels for point clouds," in *Proceedings of the IEEE Computer*

Society Conference on Computer Vision and Pattern Recognition, ed, 2013, pp. 2027-2034.

[59] M. Van Den Bergh, X. Boix, G. Roig, B. De Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* vol. 7578 LNCS, ed, 2012, pp. 13-26.

[60] C. Rohkohl and K. Engel, "Efficient image segmentation using pairwise pixel similarities," *DAGM Symposium on Pattern Recognition,* vol. 29, pp. 254-263, 2007.

[61] F. Drucker and J. MacCormick, "Fast superpixels for video analysis," in *2009 Workshop on Motion and Video Computing, WMVC '09*, ed: IEEE, 2009, pp. 1-8.

[62] P. Wang, G. Zeng, R. Gan, J. Wang, and H. Zha, "Structure-sensitive superpixels via geodesic distance," *International Journal of Computer Vision,* vol. 103, pp. 1-21, 2013.

[63] F. Perbet, B. Stenger, and A. Maki, "Homogeneous superpixels from Markov random walks," *IEICE Transactions on Information and Systems,* vol. E95-D, pp. 1740-1748, 2012.

[64] S. Jiang, D. Cao, P. Hu, S. Zhu, and Y. Wu, "Fast superpixel segmentation by iterative edge refinement," *Electronics Letters,* vol. 51, pp. 230-232, 2015.

[65] P. Siva and A. Wong, "Grid Seams: A Fast Superpixel Algorithm for Real-Time Applications," in *2014 Canadian Conference on Computer and Robot Vision*, ed: IEEE, 2014, pp. 127-134.

[66] P. Morerio, G. C. Georgiu, L. Marcenaro, and C. Regazzoni, "Optimizing Superpixel Clustering for Real-Time Egocentric-Vision Applications," *IEEE Signal Processing Letters,* vol. 22, pp. 469-473, 2015.

[67] G. Serra, M. Camurri, and L. Baraldi, "Hand Segmentation for Gesture Recognition in Ego-vision," in *Proceedings of the 3rd ACM international workshop on Interactive multimedia on mobile & portable devices*, ed. New York, New York, USA: ACM Press, 2013, pp. 31-36.

[68] C. Y. Ren and I. Reid, "gSLIC: a real-time implementation of SLIC superpixel segmentation," *University of Oxford, Department of Engineering Science,* pp. 1-6, 2011.

[69] P. Arbeláez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 33, pp. 898-916, 2011.

[70] D. Stutz, "Superpixel Segmentation Using Depth Information," B.Sc., Fakultät für Mathematik, RWTH Aachen University, Germany, 2014.

[71] P. Neubert and P. Protzel, "Superpixel Benchmark and Comparison," in *tu-chemnitz.de*, 2012, pp. 1-12.

[72] S. Lloyd, "Least squares quantization in PCM," in *IEEE Transactions on Information Theory* vol. 28, ed, 1982, pp. 129-137.

[73] O. Verevka and J. W. Buchanan, "Local K-Means Algorithm for Colour Image Quantization," *Proc of GI'95,* pp. 128-135, 1995.

[74] a. Kumar, Y. Sabharwal, and S. Sen, "A simple linear time (1 + e)-approximation algorithm for k-means clustering in any dimensions," in *45th Annual IEEE Symposium on Foundations of Computer Science*, ed: IEEE, 2004, pp. 454-462.

[75] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman, and A. Y. Wu, "A local search approximation algorithm for k-means clustering," in *Comput Geom* vol. 28, ed: ACM Press, 2004, pp. 89-112.

[76] C. Elkan, "Using the Triangle Inequality to Accelerate K-Means," in *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, ed: AAAI Press, 2003, pp. 147-153.

[77] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME-Journal of Basic Engineering,* vol. 82, pp. 35-45, 1960.

[78] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda*, et al.*, "Top 10 algorithms in data mining," in *Knowledge and Information Systems* vol. 14, ed, 2008, pp. 1-37.

[79] E. Forgy, "Cluster analysis of multivariate data: efficiency versus interpretability of classifications," *Biometrics,* vol. 21, pp. 768-769, 1965.

[80] J. MacQueen, "Some methods for classification and analysis of multivariate observations," *Proceedings of the fifth Berkeley symposium on …,* vol. 233, pp. 281-297, 1967.

[81] L. Bottou and Y. Bengio, "Convergence properties of the K-means algorithms," *Advances in Neural Information Processing Systems 7,* pp. 585 - 592, 1995.

[82] M. E. Celebi, H. A. Kingravi, and P. A. Vela, "A comparative study of efficient initialization methods for the k-means clustering algorithm," *Expert Systems with Applications,* vol. 40, pp. 200-210, 2013.

[83] G. Hamerly and C. Elkan, "Alternatives to the k-means algorithm that find better clusterings," in *Proceedings of the eleventh international conference on Information and knowledge management* vol. 4, ed: Acm, 2002, pp. 600-607.

[84] D. S. Hochbaum and D. B. Shmoys, "A Best Possible Heuristic for the k-Center Problem," in *Mathematics of Operations Research* vol. 10, ed, 1985, pp. 180-184.

[85] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, ed: Society for Industrial and Applied Mathematics Philadelphia, PA, USA, 2007, pp. 1027-1035.

[86] A. Vattani, "k-means Requires Exponentially Many Iterations Even in the Plane," *Discrete and Computational Geometry,* vol. 45, pp. 596-616, 2011.

[87] D. Arthur, B. Manthey, H. R, #246, and glin, "Smoothed Analysis of the k-Means Method," *J. ACM,* vol. 58, pp. 1-31, 2011.

[88] M. E. Celebi, *Partitional Clustering Algorithms*: Springer, 2015.

[89] S. Phillips, "Acceleration of K-Means and Related Clustering Algorithms," in *Algorithm Engineering and Experiments*. vol. 2409, D. Mount and C. Stein, Eds., ed: Springer Berlin Heidelberg, 2002, pp. 166-177.

[90] C. Li and K. M. Kitani, "Model Recommendation with Virtual Probes for Egocentric Hand Detection," in *2013 IEEE International Conference on Computer Vision*, ed: IEEE, 2013, pp. 2624-2631.

[91] R. Laganière, *OpenCV 2 Computer Vision Application Programming Cookbook*: Packt Publishing, 2011.

[92] J. Reinders, *Intel threading building blocks: outfitting C++ for multi-core processor parallelism* vol. 23: O'Reilly Media, Inc., 2007.

Shahrood University

Faculty of Electrical and Robotic Engineering

An Optimized Superpixel Algorithm in the Framework of Egocentric Vision

Seyed Amir Hossein Farzaneh

Supervisor:

Dr. Hossein Khosravi

July 2015

Abstract

Recent technological advances have made manufacturing lightweight and head-mountable cameras with many applications, possible. Google Glass, as a wearable gadget, is a publicly affordable example. These gadgets have introduced the idea of egocentric (first person) vision as a new framework for the computer vision community. In this framework, one is capable to recognize objects and activities in his own field of view. Vision capabilities are also augmented in medical surgeries, industry and art, thus new solutions to overcome challenges in these fields are introduced. Nonlinear motion, blur, fast transition and runtime of processing algorithms are such challenges. In this thesis, to optimize a superpixel algorithm in the framework of egocentric vision, a new approach is proposed. In the framework of egocentric videos captured from the wearable gadgets, a fast and portable implementation of a processing algorithm is needed. In order to get closer to the real-time performance, we have established a trade-off between segmentation quality and runtime of a state-of-the-art algorithm in the proposed method. For a video of 1280×720 resolution, we have achieved a 23% up to 64% increase in performance for various approaches. The algorithm optimization, has caused a slight decrease in segmentation quality in homogenous regions of a video frame. This decrease however, has not influenced the edge segmentation quality.

Keywords: Superpixel, Video, Egocentric Vision, Video Processing