

بیتنام خدا



دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گروه هوش مصنوعی

پایان نامه کارشناسی ارشد

فشرده سازی تصاویر ویدیویی مبتنی بر مدل سازی پس زمینه

مهدی صدیقی دستجرد

استاد راهنما: دکتر حمید حسن پور

استاد مشاور: دکتر علی اکبر پویان

جهت اخذ درجه کارشناسی ارشد

دی ۱۳۸۹

دانشگاه صنعتی شاهرود

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گروه هوش مصنوعی

پایان نامه کارشناسی ارشد آقای مهدی صدیقی تحت عنوان فشرده سازی تصاویر ویدیویی مبتنی بر مدل سازی پس زمینه در تاریخ ۸۹/۱۱/۱۶ توسط کمیته تخصصی زیر جهت اخذ مدرک کارشناسی ارشد مورد ارزیابی و با درجه مورد پذیرش قرار گرفت.

امضاء	اساتید مشاور	امضاء	اساتید راهنما
	نام و نام خانوادگی :		نام و نام خانوادگی :
	نام و نام خانوادگی :		نام و نام خانوادگی :

امضاء	نماینده تحصیلات تکمیلی	امضاء	اساتید داور
	نام و نام خانوادگی :		نام و نام خانوادگی :
			نام و نام خانوادگی :
			نام و نام خانوادگی :
			نام و نام خانوادگی :

تقدیم به عزیزان از دست رفته ام

مادرم و پدرم

سالی که بر من و خانواده ام گذشت سالی سراسر مصیبت و سختی بود. این سال به جرات بدترین سال زندگی من تاکنون بود سالی که در آن عزیزترین کسانم را از دست دادم و اگر امید و همراهی دوستان و اساتیدم نبود این راه به پایانش نمی رسید.

در این جا بر خودم واجب می دانم از زحمات استادگرامیم دکتر حسن پور که صادقانه شرایط مرا درک کردند و اگر حمایت ها و باور های ایشان نبود هرگز این راه که امید من و عزیزان از دست رفته ام بود به پایان نمی رسید و با سپاس فراوان از استادان گرامیم دکتر پویان و دکتر زاهدی به خاطر تغییر نگرش من به زندگی و همراهی بزرگوارانه ایشان در سختی ها و مشکلاتی که یک سال همزاد من بود.

و تشکر از دوست نازنین و مهربانم مهندس رحمان یوسف زاده که بیشتر از یک همکلاسی و دوست بر گردن من حق داشتند و برادرانه در کنارم بودند.

و در پایان از دوستانم مهندس صادق آقازاده، مهندس احمد صادقی، مهندس حمزه هدهدکیان و خانم اسدی امیری که به نوعی مرا در رسیدن به این هدف همراهی کردند تشکر می کنم.

دانشجو تأیید می نماید که مطالب مندرج در این پایان نامه (رساله) نتیجه تحقیقات خودش می باشد و در صورت استفاده از نتایج دیگران مرجع آن را ذکر نموده است.

کلیه حقوق مادی مترتب از نتایج مطالعات ، آزمایشات و نو آوری ناشی از تحقیق موضوع این پایان نامه (رساله) متعلق به دانشگاه صنعتی شاهرود می باشد .

بهمن ۸۹

چکیده

رسانه‌های ویدیویی معمولا حجم زیادی از داده‌ها را تولید می‌کنند. این حجم عظیم اطلاعات مشکلاتی از قبیل عدم وجود فضای لازم برای ذخیره کردن اطلاعات و پهنای باند کافی برای انتقال آنها به همراه دارد. برای مقابله با این مشکلات از سیستم‌های فشرده سازی برای ذخیره کردن تصاویر ویدیویی استفاده می‌شود. این سیستم‌ها با بهره گرفتن از افزونگی بین فریم‌ها، سعی در بالا بردن درصد فشرده‌سازی دارند.

برای یافتن افزونگی، فریم تصاویر به بلوک‌هایی تقسیم می‌شود سپس این بلوک‌ها با هم مقایسه می‌گردند تا از ذخیره سازی بلوک‌های تکراری جلوگیری شود. مولفه انطباق بلوک علاوه برآنکه یکی از زمانبرترین مولفه‌ها در سیستم‌های فشرده سازی است اگر از نظر دقت دارای عملکرد مناسبی نباشد، درصد فشرده‌سازی کاهش می‌یابد. در روش‌های موجود نظیر MPEG2 در فرایند انطباق بلوک، تمام بلوک‌ها مورد بررسی قرار می‌گیرند. به همین دلیل این گونه روش‌ها بار محاسباتی بالایی دارند.

در این پایان‌نامه با استفاده از روش‌های مدل سازی پس‌زمینه، نواحی که در آن تغییرات قابل توجه اتفاق می‌افتد را تعیین، و عملیات انطباق بلوک را به این ناحیه محدود می‌کنیم. برای تعیین نواحی که در آن تغییر قابل توجه به وجود آمده، ابتدا پس زمینه را مدل کرده و بعد از حذف نویز، خروجی را به مولفه دیگر ارسال می‌کنیم. در این مولفه، پیکسل‌ها به یکی از سه کلاس پس زمینه، شی یا سایه تقسیم می‌شوند با بهره‌گرفتن از این اطلاعات بلوک‌هایی که در آن تغییر قابل توجهی اتفاق افتاده است تعیین می‌شود و عملیات انطباق بلوک به این نواحی محدود می‌گردد. مقایسه نتایج این الگوریتم جدید با چهار الگوریتم جستجو دیگر (2-D logarithm search، UCBD، 4SS و CSD) نشان می‌دهد زمان الگوریتم پیشنهادی تقریبا یک دهم زمان محاسبات الگوریتم‌های یاد شده می‌باشد. نتایج

همچنین نشان می‌دهد درصد فشرده سازی نیز به نسبت الگوریتم های یاد شده بهبود قابل توجهی پیدا می‌کند. در نمونه های مورد بررسی حجم خروجی سیستم ارائه شده، به نسبت سیستم غیر بهینه تقریبا ۵۰٪ کاهش می‌یابد.

کلمات کلیدی: فشرده سازی ویدیو، مدل سازی پس زمینه، فشرده سازی تصویر، معیارهای شباهت

و MPEG

لیست مقالات استخراج شده:

پذیرفته شده:

Video Frame's Background Modeling: Reviewing the Techniques, Hamid Hassanpour, Mehdi Sedighi and Ali Reza Manashty , Journal of Signal and Information Processing

ارائه شده:

بهبود فرایند انطباق بلوک در فشرده سازی ویدیو با استفاده از مدل سازی پس زمینه، حمید

حسن پور، مهدی صدیقی، مجله مهندسی برق و کامپیوتر ایران

بررسی اثر معیار های شباهت بر کارایی تکنیک های فشرده سازی ویدیو حمید حسن پور،

مهدی صدیقی، مجله مهندسی برق دانشگاه تبریز

فهرست مطالب

- ۱- فصل اول مقدمه ۱
- ۱-۱- ضرورت فشرده سازی ۲
- ۲-۱- تعریف فشرده سازی و انواع آن ۳
- ۳-۱- رهیافت پایان نامه ۵
- ۴-۱- ساختار پایان نامه ۵
- ۴-۱- فصل اول (در این فصل اجزا و مولفه های استاندارد JPEG تشریح می شود) ۶
- ۴-۲- فصل دوم (اجزای و مولفه های استاندارد MPEG استفاده شده در این سیستم ارائه شده تشریح می شود) ۷
- ۴-۳- فصل سوم (در این بخش روش های مدل سازی پس زمینه، حذف نویز و تشخیص سایه مورد استفاده در سیستم ارائه شده مورد بررسی قرار می گیرند) ۸
- ۴-۴- فصل چهارم (جدا کردن اشیا از پس زمینه با استفاده از رهیافت جداکننده) ۹
- ۴-۵- فصل پنجم (ارزیابی و نتایج سیستم جدید در مقایسه با استاندارد MPEG) ۱۰
- ۲- فصل دوم اجزا و استانداردهای فشرده سازی تصاویر ۱۲
- ۲-۱- مقدمه ۱۳
- ۲-۱-۱- نمونه برداری کاهشی (Chrominance Sub sampling) ۱۴
- ۲-۱-۲- اصول فشرده سازی تصویر ۱۵

۱۷کاهش همبستگی بین پیکسل‌ها
۱۷Transform Coding کدگذاری تبدیلی
۲۱کاهش همبستگی در حوزه فرکانس
۲۳کد گذاری پیشگو (Predictive coding)
۲۶چندی سازی (Quantization)
۲۷رمزگذاری آنتروپی (Entropy Coding)
۲۷رمزگذاری هافمن (Huffman Coding)
۳۱رمزگذاری ریاضی (Arithmetic Coding)
۳۲کدگذاری LZW
۳۳روش های کلاسیک برای فشرده سازی تصاویر
۳۴کدگذار (JPEG Encoder)
۴۱کدگشای JPEG Decoder
۴۲استاندارد JPEG۲۰۰۰
۴۲تبدیل رنگی معکوس پذیر
۴۳چندی سازی در JPEG۲۰۰۰
۴۳جمع بندی (از نتایج پایان نامه)
۴۸فصل سوم اجزا و استانداردهای سیستم‌های فشرده سازی ویدئو
۴۹مقدمه

۵۲	۱-۱-۳ نحوه تولید ویدیو
۵۵	۲-۱-۳ معیار ها ارزیابی روش های فشرده سازی ویدیو
۵۵	۳-۱-۳ نمونه برداری رنگی
۵۷	۲-۲-۳ گام های فشرده سازی در استاندارد H.۲۶۱
۵۶	۱-۲-۳ کد کردن فریم نوع ۱
۵۹	۲-۲-۳ کد کردن فریم P
۶۰	۳-۲-۳ ساختار داده ها در استاندارد H.۲۶۱
۶۲	۳-۳ استاندارد MPEG
۶۳	۱-۳-۳ تفاوت های MPEG و H.۲۶۱
۶۳	۲-۳-۳ مراحل الگوریتم MPEG ۱
۷۶	۴-۳ تخمین بردار های حرکت
۷۹	۱-۴-۳ مروری بر روش های تخمین بلوک
۹۴	۲-۴-۳ مقایسه الگوریتم های مختلف تخمین بلوک (از نتایج پایان نامه)
۹۸	۴- فصل چهارم مدل سازی پس زمینه
۹۸	۱-۴- مقدمه
۹۹	۲-۴- مدل سازی پس زمینه
۹۹	۱-۲-۴ فیلتر میانه تقریبی
۱۰۹	۲-۲-۴ الگوریتم میانگین برای مدل سازی پس زمینه

۱۱۶	۳-۲-۴- الگوریتم PCA برای مدل سازی پس زمینه
۱۲۴	۴-۲-۴- مدل سازی پس زمینه با استفاده از روش های آماری
۱۴۴	۳-۳-۴- بررسی و مقایسه عملکرد روش های موجود مدلسازی پس زمینه (از نتایج پایان نامه)
۱۵۱	۴-۴- حذف سایه
۱۵۵	۵-۴- حذف نویز
۱۵۶	۴-۵-۱- حذف نویز با استفاده از الگوریتم های ریخت شناسی
۱۶۵	۴-۵-۲- Connected Component Labeling (CCL) and Area Filter
۱۶۷	۴-۵-۳- حذف نویز با استفاده از الگوریتم میانه و میانگین
۱۶۹	۵- فصل پنجم جداسازی اشیا از پس زمینه
۱۷۰	۵-۱- مقدمه
۱۷۵	۵-۲- توابع جداکننده
۱۷۶	۵-۳- حداقل مربعات خطا برای کلاسه بندی
۱۷۸	۵-۴- الگوریتم پرسپترون
۱۸۱	۵-۵- مدل آماری
۱۹۱	۵-۶- Iterative reweighted least squares
۱۸۹	۵-۷- نتایج (از نتایج پایان نامه)
۱۹۵	۶- فصل ششم بهبود سیستم فشرده سازی استاندارد MPEG۲
۱۹۶	۶-۱- مقدمه

۱۹۷	۲-۶-معیار های شباهت و کارایی آن بر فشرده سازی ویدیو.....
۱۹۸	۱-۲-۶- تعاریف توابع مشابهت.....
۲۰۰	۲-۲-۶- آزمایش.....
۲۰۲	۳-۶- نتیجه.....
۲۰۳	۴-۶- بهبود انطباق بلوک با استفاده از مدل سازی پس زمینه.....
۲۰۴	۱-۴-۶- پیدا کردن ناحیه فعال.....
۲۰۴	۲-۴-۶- مدل سازی پس زمینه.....
۲۰۵	۳-۴-۶- تعیین ناحیه فعال.....
۲۰۶	۵-۶- آزمایش.....
۲۰۶	۱-۵-۶- زمان محاسبات.....
۲۰۸	۲-۵-۶- حجم فشرده سازی.....
۲۰۹	۳-۵-۶- کیفیت تصویر.....
۲۱۰	۶-۶- نتیجه.....
۲۱۲	۷- فصل هفتم نتایج و پیشنهادات.....
۲۱۳	۱-۷- جمع بندی.....
۲۱۹	۲-۷- کارهای آینده.....

شکل ۱-۱ اجزا و مولفه های سیستم پیشنهادی در این پایان نامه..... ۱۱

شکل ۱-۲ طرحی کلی سیستم فشرده سازی تصاویر دیجیتالی..... ۱۳

شکل ۲-۲ روند طی شده توسط سیستم فشرده سازی تصویر، تصویر وارد کدگذار می شود و رشته ای از بیت ها تولید می گردد. اگر حجم این جریان بیتی کمتر از حجم تصویر اصلی باشد به هدف فشرده سازی دست یافته ایم. برای این که بتوانیم تصویر را بازیابی کنیم باید جریان بیتی به کدگشا وارد گردد تا تصویر اصلی بازیابی شود..... ۱۵

شکل ۳-۲ روند نمای کلی سیستم فشرده سازی تصویر با از دست دادن اطلاعات، ابتدا افزونگی بین داده ها کاهش می یابد در گام دوم خروجی چندی سازی شده و در نهایت خروجی به کدگذار انتروپی ارسال میشود..... ۱۷

شکل ۴-۲ نمایش همبستگی بین داده های جدول ۲-۸..... ۲۱

شکل ۵-۲ طرح یک سیستم پیشگو ساده برای فشرده سازی تصویر، در این روش به جای این که مقدار پیکسل را انتقال دهیم مقدار تفاوت آن را با پیکسل های بعدی انتقال می دهیم از آنجایی که پیکسل های همسایه به هم نزدیک می باشند تفاضل مقادیر می تواند کوچک و نزدیک به صفر باشند..... ۲۴

شکل ۶-۲ تصویر سمت چپ هیستوگرام تصویر پیش از پردازش و هیستوگرام سمت راست، هیستوگرام تصویر پردازش شده را نشان می دهد. هیستوگرام یک تصویر به طور یکنواخت پراکنده شده است. به همین دلیل اگر از کد با طول متغییر نیز استفاده کنیم، تاثیر چندانی را در فشرده سازی ندارد. اما همانطوری که در تصویر تفاضلات می بینید، مقادیر تفاضل ها در اطراف یک عدد خاص تمرکز یافته است. به همین دلیل در فشرده سازی به درصد بالاتری از فشردگی دست خواهیم یافت..... ۲۵

شکل ۲-۷ اثر گدگذار پیشگو بر روی تصویر هیستوگرام های این تصاویر را به ترتیب در الف و ب در تصویر ۲-۶ نشان داده شده است. همانطور که مشاهده می شود هیستوگرام تصویر خروجی سیستم پیشگو حول ۰ متمرکز می باشد. به همین دلیل با استفاده از روش های انتروپی حجم فشرده سازی آن بیشتر خواهد شد. ۲۵

شکل ۲-۸ نمودار تصویر چندی سازی یکنواخت با طول گام ۱ ۲۶

شکل ۲-۹ گام های اول تا سوم برای الگوریتم هافمن ۲۹

شکل ۲-۱۰ گام های چهارم تا پنجم برای الگوریتم هافمن ۲۹

شکل ۲-۱۱ گام نهایی در الگوریتم هافمن ۳۰

شکل ۲-۱۲ مراحل الگوریتم JPEG انتقال تصویر به فضای YCbCr جز استاندارد jpeg نمی باشد اما اکثر نرم افزار هایی که این استانداردها را پیاده سازی می کنند ابتدا این تبدیل را انجام می دهند. با انجام این تبدیل، تا حد زیادی حجم تصاویر کاهش می یابد. بعد از آن که داده ها به این فضا انتقال یافت و نمونه برداری شد، به Encoder ارسال و در آنجا به دنباله ای از بیت ها تبدیل می شوند. ۳۴

شکل ۲-۱۳ اجزای کدگذار JPEG، ابتدا داده ها شیفت داده می شود، سپس با استفاده از الگوریتم DCT کاهش افزونگی در حوزه فرکانس انجام می گیرد و در نهایت چندی سازی و انتروپی بر روی آن اعمال می گردد. ۳۵

شکل ۲-۱۴ نمونه ای از اعمال شیفت سطح بر روی داده ها ۳۵

شکل ۲-۱۵ تصویر مولفه های تبدیل گسسته کسینوسی ۳۶

شکل ۲-۱۶ در این تصویر نحوه اعمال تبدیل DCT را بر روی بلوک های تصویر نشان می دهد ۳۷

شکل ۲-۱۷ مثال عددی از تبدیل DCT بر روی تصویر ۳۷

شکل ۲-۱۸ نحوه چندی سازی درایه ها در این تصویر مثالی عددی را نشان می دهد که چگونه داده ها در استاندارد JPEG کوانتیزه می شوند ۳۸

شکل ۲-۱۹ نمودار فوق نحوه مرتب کردن ضرایب فرکانس، جهت خطی کردن مولفه ها را نشان می دهد. ۳۹

شکل ۲-۲۰ مراحل کدگشایی در الگوریتم JPEG ۴۱

شکل ۲-۲۱ اجزای کدگشا در استاندارد JPEG ۴۱

شکل ۲-۲۲ برای بررسی اثر DCT و چندی سازی، تصویر را ابتدا با اعمال DCT به حوزه فرکانس ارسال می کنیم بعد با ضریب های مختلف کوانتیزه می شود و دو باره تصاویر را از حوزه فرکانس به حالت اولیه بر می گردانیم. منظور از ضریب عددی ثابت است که در ماتریس Q ضرب می شود. ۴۴

شکل ۲-۲۳ رابطه بین سطوح فشرده سازی و میزان فشرده سازی در استاندارد JPEG۲۰۰۰ محور y معکوس درصد فشرده سازی را نشان می دهد. ۴۵

شکل ۲-۲۴ تصاویر بازیابی شده از استاندارد JPEG۲۰۰۰، تعداد سطح تعداد مراحل اعمال فیلتر را نشان می دهد همانطور که مشاهده می شود اگر تعداد سطح فشرده سازی بیشتر از ۸ سطح باشد کیفیت تصویر بازیابی کاهش می یابد. (ادامه تصویر) ۴۷

شکل ۳-۱ تصویر ب را از حوزه RGB به حوزه YCBCR انتقال می دهیم و در این فضا از مولفه ها CB و CR نمونه برداری می کنیم حجم داده های جدید یک دوم حجم تصویر ب می شود. برای بدست آوردن تصویر الف دوباره بر روی داده های نمونه برداری شده تبدیل معکوس اعمال می شود تا داده ها را از فضای YCBCR به فضای RGB برگردد اما همانطور که مشاهده می شود کیفیت دو تصویر مشابه می باشند به این نوع افزونگی درون فریم می گویند. ۵۰

شکل ۳-۲ تصویر الف و ب دو فریم ویدیویی را در زمان های مختلف نشان می دهد. همانطور که مشاهده می شود قسمت عمده تصویر (پس زمینه) در دو فریم یکسان می باشد و صرفاً تغییرات محدود به ناحیه کوچکی از تصویر است که اشیا در آن حضور دارد..... ۵۱

شکل ۳-۳ نحوه تولید تصاویر ویدیویی برای نمونه برداری از دنیای واقعی دوربین در هر لحظه یک خط را پیمایش می کند زمانی که یک خط پیمایش می شود یک سیگنال که آن خط را نمایش می دهد تولید می شود مقدار این ولتاژ متناسب با سطح خاکستری بالا و پایین می رود هر چه سطح خاکستری روشنتر باشد ولتاژ بالاتر می باشد و هر چه تیره تر باشد میزان ولتاژ کمتر می باشد..... ۵۳

شکل ۳-۴ تعداد خطوط لازم برای به تصویر کشیدن صحنه در استاندارد های PAL و NTSC برای ایجاد هماهنگی بین سیستم های پخش و نمایش تلویزیونی دو نوع استاندارد NTSC و PAL (برای تعیین تعداد خطوط مورد نیاز برای نمونه برداری) وضع شده است. در استاندارد NTSC هر صحنه شامل ۵۲۵ خط و در PAL ۶۲۵ خط مورد استفاده قرار می گیرد..... ۵۴

شکل ۳-۵ نمونه برداری به صورت progressive. در این تصویر هر خط ستونی نمایش یک فریم را در یک لحظه نشان می دهد هر نقطه نمایش دهنده یک خط نمونه برداری می باشد..... ۵۴

شکل ۳-۶ نمونه برداری به صورت interlaced در این روش برای به تصویر کشیدن صحنه خطوط به صورت یکی در میان نمونه برداری می شود..... ۵۵

شکل ۳-۷ مراحل لازم برای فشرده سازی فریم نوع ۵۸

شکل ۳-۸ Encoder برای فریم نوع ۱، در این الگوریتم فریم به حوزه فرکانس منتقل می شود و در حوزه فرکانس کوانتیزه می شود تا حجم آن کاهش یابد. بعد از کوانتیزه شدن فریم برای انتقال به الگوریتم هافمن فرستاده می شود. تا کد گذاری مناسبی برای آن انتخاب شود. به این وسیله حجم فشرده سازی بالا می رود. روی فریم فشرده سازی شده عمل معکوس کوانتیزه و معکوس تبدیل کسینوسی انجام می شود. این فریم حفظ می شود تا با استفاده از آن فریم های نوع P را تقریب بزینیم..... ۵۹

شکل ۳-۹ Encoder برای فریم نوع P، بعد از کوانتیزه شدن فریم برای انتقال به الگوریتم هافمن فرستاده می شود تا کد گذاری مناسبی برای آن انتخاب شود به این وسیله حجم فشرده سازی بالا می رود. روی فریم فشرده سازی شده عمل معکوس کوانتیزه و معکوس تبدیل کسینوسی اعمال می شود این فریم در حافظه ذخیره می گردد..... ۶۰

شکل ۳-۱۰ ساختار لازم برای ذخیره ماکرو بلاک..... ۶۱

شکل ۳-۱۱ ساختار فیلد برای جریان های H.۲۶۱..... ۶۲

شکل ۳-۱۲ در تصویر فوق نقطه ضعف استاندارد H.۲۶۱ را برای پیش بینی بلوک مناسب با استفاده از یک فریم مرجع را نشان می دهد. در این شکل نمی توان بلوک سیاه رنگ را با استفاده از فریم های قبلی نمایش داد. به همین دلیل در استاندارد MPEG۲ فریم نوع B را معرفی کردند. در این نوع فریم به جای استفاده از یک فریم از دو فریم مرجع استفاده می شود..... ۶۲

شکل ۳-۱۳ یک نمونه از گروه ها فریم در استاندارد MPEG..... ۶۳

شکل ۳-۱۴ مراحل فشرده سازی ویدیو در استاندارد MPEG..... ۶۵

شکل ۳-۱۵ ماکرو بلاک های مورد استفاده در استاندارد MPEG..... ۶۶

شکل ۳-۱۶ مراحل مورد نیاز برای فشرده سازی فریم نوع I..... ۶۷

شکل ۳-۱۷ مراحل لازم برای تبدیل DCT بر روی تصویر، ابعاد بلوک ها $8*8$ می باشد. همانطور که در این تصویر مشاهده می شود که هر کدام از تصویر به صورت بلوک تقسیم می شود و بر روی هر کدام از بلوک های تبدیل DCT اعمال می شود تا ماتریس D را نمایش می دهد. هر کدام از بلوک های A_{ij} ماتریسی با ابعاد $8*8$ می باشد..... ۶۸

شکل ۳-۱۸ مراحل لازم برای کوانتیزه کردن ماتریس فرکانس برای سادگی ابعاد این ماتریس را $4*4$ در نظر گرفتیم در حالی که در سیستم های فشرده سازی ابعاد ماتریس $8*8$ می باشد. برای چندی سازی هر عنصر در حوزه فرکانس را بر درایه متناظر آن در ماتریس کوانتیزه تقسیم می کنیم..... ۶۹

شکل ۳-۱۹ مراحل مورد نیاز برای کد کردن فریم نوع P..... ۷۰

شکل ۳-۲۰ استخراج بردار های حرکت..... ۷۱

شکل ۳-۲۱ بازسازی فریم با استفاده از بردار حرکت و فریم مرجع..... ۷۱

شکل ۳-۲۲ بردار حرکت برای چند فریم دلخواه..... ۷۲

شکل ۳-۲۳ روش پیش بینی ماکرو با استفاده از روش پیش رو..... ۷۳

شکل ۳-۲۴ روش پس رو برای پیش بینی ماکرو بلاک..... ۷۳

شکل ۳-۲۵ روش درونیابی برای پیش بینی ماکرو بلاک ها..... ۷۴

شکل ۳-۲۶ مراحل کد کردن فریم برای فریم های نوع B..... ۷۴

شکل ۳-۲۷ بررسی اثر اندازه بلوک بر میدان های بردار های حرکت در فریم..... ۷۶

شکل ۳-۲۸ جستجو یک بلوک درون یک ناحیه..... ۷۸

شکل ۳-۲۹ الگوی مورد استفاده برای جستجو در الگوریتم UCBD..... ۸۰

شکل ۳-۳۰ جستجو بر روی راس در الگوریتم UCBD..... ۸۱

شکل ۳-۳۱ جستجو بر روی لبه در الگوریتم..... ۸۱

شکل ۳-۳۲ جستجو در مرحله پایانی الگوریتم UCBD..... ۸۲

- شکل ۳-۳۳ نقاط مورد جستجو برای الگوریتم ARPS-ZMP ۸۳
- شکل ۳-۳۴ الگوی گام اول ۸۴
- شکل ۳-۳۵ الگوی گام دوم و سوم نقاط آبی نقاطی را نشان می دهد که قبلا مورد ارزیابی قرار گرفته اند و نقاط بنفش نقاط جدید را نشان می دهد. ۸۵
- شکل ۳-۳۶ الگو در گام نهایی ۸۵
- شکل ۳-۳۷ یک نمونه از جستجو در الگوریتم ۴SS ۸۶
- شکل ۳-۳۸ الگوهای مورد استفاده در الگوریتم Diamond Search ۸۷
- شکل ۳-۳۹ انواع تغییر وضعیت ها در الگوریتم Diamond Search ۸۷
- شکل ۳-۴۰ یک نمونه از جستجو با استفاده از الگوریتم Diamond Search ۸۸
- شکل ۳-۴۱ الگوهای مورد استفاده در الگوریتم جستجو کراس ۸۹
- شکل ۳-۴۲ یک نمونه مورد استفاده برای الگوریتم کراس ۹۰
- شکل ۳-۴۳ الگو مورد استفاده در الگوریتم HEXBS ۹۱
- شکل ۳-۴۴ مرحله اول الگوریتم ۲D-Logarithm ۹۲
- شکل ۳-۴۵ مرحله دوم جستجو ۲D-Logarithm ۹۳
- شکل ۳-۴۶ مرحله دوم جستجو ۹۴
- شکل ۴-۱ نمونه ای از بکارگیری روش میانه برای مدل سازی پس زمینه جهت تشخیص اشیا $T=15$ ، $\alpha=0,1$ و تعداد فریم های مورد استفاده ۱۵۰ می باشد زمان مصرفی برابر ۱۲,۰۴۷۲۹۶ ثانیه می باشد. ۱۰۳

شکل ۴-۲- خطای false negative این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها تعداد فریم های استفاده شده برای مدل سازی پس زمینه را نشان می دهد و محور عمودی میزان خطا را نشان می دهد. ۱۰۴.....

شکل ۴-۳ خطای false positive این خطا درصد پیکسل هایی را نشان می دهد که جز به شی نبوده اند اما محاسبه شده اند. محور x ها تعداد فریم های استفاده شده برای مدل سازی پس زمینه را نشان می دهد و محور عمودی میزان خطا را نشان می دهد. ۱۰۵.....

شکل ۴-۴ خطای false negative این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها میزان α و محور عمودی میزان خطا FN را نشان می دهد. ۱۰۵.....

شکل ۴-۵ خطای false positive این خطا درصد پیکسل هایی را نشان می دهد که جز به شی نبوده اند اما محاسبه شده اند. محور x میزان α و محور عمودی میزان خطا FPI را نشان می دهد. ۱۰۶.....

شکل ۴-۶ خطای false negative این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها میزان حد آستانه و محور عمودی میزان خطا FN را نشان می دهد. ۱۰۶.....

شکل ۴-۷ خطای false positive این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها میزان حد آستانه و محور عمودی میزان خطا را نشان می دهد. ۱۰۷.....

شکل ۴-۸ نتایج الگوریتم در محیط درونی با $\tau=15$, $\alpha=0,1$ و تعداد فریم های مورد استفاده ۱۵۰ می باشد. سطر اول مدل استخراج شده برای پس زمینه را در چند فریم دلخواه، نشان می دهد. در این مدل، شی نیز به عنوان پس زمینه محاسبه شده است، زیرا سرعت شی کم می باشد و این متد نمی تواند به درستی آن را از درون مدل جدا کند. سطر دوم اشیا استخراج شده را به صورت خودکار نشان می دهد. ۱۰۸.....

شکل ۴-۹ اثر حد آستانه در محیط خارجی بر خطای FP ۱۰۹.....

شکل ۴-۱۰ اثر حد آستانه در محیط خارجی بر خطای FN ۱۰۹.....

- شکل ۴-۱۱ اثر alpha در محیط خارجی بر خطای FP..... ۱۰۹
- شکل ۴-۱۲ اثر alpha در محیط خارجی بر خطای FN..... ۱۰۹
- شکل ۴-۱۳ اشیا استخراج شده از محیط درون ساختمان با استفاده از الگوریتم های رهیافت میانگین (ادامه تصویر)..... ۱۱۵
- شکل ۴-۱۴ داده های اولیه مورد استفاده شده..... ۱۱۷
- شکل ۴-۱۵ داده های اولیه به همراه ترسیم بردار های ویژه..... ۱۱۸
- شکل ۴-۱۶ داده های بعد از نگاشت با استفاده از بردار های ویژه..... ۱۲۰
- شکل ۴-۱۷ داده ها بعد از بازیابی..... ۱۲۲
- شکل ۴-۱۸ نتیجه مدل سازی پس زمینه با استفاده روش PCA..... ۱۲۳
- شکل ۴-۱۹ در شکل ابتدا ۱۰۰۰ نقطه را که به صورت تصادفی توسط تابع تصادفی نرمال تولید شدند به کمک روش هیستوگرام سعی در تخمین تابع توزیع چگالی را آنها را هستیم همانطوری که مشاهده می کنید زمانی که تعداد بین ها کم باشد میزان خطاها بالاست..... ۱۲۷
- شکل ۴-۲۰ این تصویر بلوک های را که برای بررسی نحوه تغییرات سطح خاکستری مورد استفاده قرار می گیرد نشان داده شده است..... ۱۲۸
- شکل ۴-۲۱ این تصویر نمودار هستوگرام ۴ بلوک دلخواه را نشان می دهد همانطور که مشاهده می شود سطح خاکستری در تمام ۱۷ فریم تقریباً ثابت می باشد..... ۱۲۹
- شکل ۴-۲۲ اندازه بلوک ها برابر ۱۶*۱۶ می باشد شماره بر روی عنوان تصاویر نشان دهنده شماره فریم در ویدیو می باشد. اشیا استخراج شده توسط الگوریتم مدل سازی پس زمینه با استفاده از هستوگرام..... ۱۲۹

شکل ۴-۲۳ اندازه بلوک ها برابر $8*8$ می باشد شماره بر روی عنوان تصاویر نشان دهنده شماره فریم در ویدیو می باشد. ۱۳۰

شکل ۴-۲۴ اندازه بلوک ها برابر $4*4$ می باشد شماره بر روی عنوان تصاویر نشان دهنده شماره فریم در ویدیو می باشد. اشیا استخراج شده توسط الگوریتم مدل سازی پس زمینه با استفاده از هستوگرام ۱۳۰

شکل ۴-۲۵ نتیجه خروجی روش مدل سازی پس زمینه با استفاده از هستوگرام برای استخراج اشیا از صحنه در محیط برونمی. ۱۳۱

شکل ۴-۲۶ مدل سازی توزیع احتمال یا استفاده از چند کرنل استاندارد. ۱۳۲

شکل ۴-۲۷ تصاویر استخراج شده از مدل سازی با کرنل با استفاده از کرنل نرمال. ۱۳۴

شکل ۴-۲۸ تصاویر استخراج شده از مدل سازی با کرنل با استفاده از کرنل Biweight. ۱۳۵

شکل ۴-۲۹ تصاویر استخراج شده از مدل سازی با کرنل با استفاده از کرنل Triweight. ۱۳۵

شکل ۴-۳۰ استخراج شده از مدل سازی با کرنل با استفاده از کرنل Epanechnikov. ۱۳۶

شکل ۴-۳۱ استخراج شده از مدل سازی با کرنل با استفاده از کرنل Triangle. ۱۳۶

شکل ۴-۳۲ بررسی اثر عرض کرنل بر کارایی مدل سازی با استفاده از کرنل نرمال. ۱۳۷

شکل ۴-۳۳ بررسی استخراج اشیا در محیط بیرونی با استفاده از رهیافت مدل سازی با استفاده از کرنل. ۱۳۷

شکل ۴-۳۴ در این تصویر زیر چند نمونه از اشیایی را می بینید که توسط مدل گوسی از پس زمینه جدا شده اند اندازه بلوک ها برابر $8*8$ می باشد. ۱۴۰

شکل ۴-۳۵ در این تصویر چند نمونه از اشیایی را می بینید که توسط مدل گوسی از پس زمینه جدا شده اند اندازه بلوک ها برابر $4*4$ می باشد. ۱۴۰

- شکل ۴-۳۶ چند نمونه از اشیایی که توسط مدل گوسی از پس زمینه جدا شده اند. اندازه بلوک ها برابر $4*4$ می باشد محیط مورد بررسی خارجی می باشد. ۱۴۱.....
- شکل ۴-۳۷ اشیا استخراج شده با استفاده از مدل گوسی با تغییر حد آستانه. ۱۴۱.....
- شکل ۴-۳۸ اشیا استخراج شده توسط الگوریتم Gaussian Mixture Model در محیط indoor. ۱۴۳.....
- شکل ۴-۳۹ اشیا استخراج شده توسط الگوریتم Gaussian Mixture Model در محیط outdoor. ۱۴۴.....
- شکل ۴-۴۰ تصویر اشیا استخراج شده از صحنه به همراه سایه آن یا استفاده از رابطه (۴-۳۴). ۱۵۳.....
- شکل ۴-۴۱ سایه استخراج شده با استفاده از تصویر الگوریتم NCC. ۱۵۵.....
- شکل ۴-۴۲ کرنل مورد استفاده برای سایش. ۱۶۱.....
- شکل ۴-۴۳ تصویر اصلی. ۱۶۱.....
- شکل ۴-۴۴ قسمت های قرمز نواحی سایه شده تصویر اصلی را نشان می دهد. ۱۶۱.....
- شکل ۴-۴۵ تصویر ۳۹ بعد از یک بار سایش با کرنل ۳۸. ۱۶۱.....
- شکل ۴-۴۶ قسمت سبز در بار دوم سایش شده. ۱۶۱.....
- شکل ۴-۴۷ بعد از دو بار سایش. ۱۶۱.....
- شکل ۴-۴۸ قسمت قرمز افزوده شده برای بار دوم می باشد. ۱۶۲.....
- شکل ۴-۴۹ گسترش تصویر ۴۳ با کرنل ۳۹ قسمت سبز قسمت افزوده شده را نشان. ۱۶۲.....
- شکل ۴-۵۰ تصویر کرنل مورد استفاده. ۱۶۲.....
- شکل ۴-۵۱ تصویر اصلی. ۱۶۲.....
- شکل ۴-۵۲ قسمت قرمز قسمت سایه شده را نشان می دهد. ۱۶۲.....
- شکل ۴-۵۳ تصویر سایه شده با استفاده از کرنل ۴۶. ۱۶۲.....
- شکل ۴-۵۴ قسمت سبز در مرحله دوم سایه شده. ۱۶۳.....
- شکل ۴-۵۵ تصویر بعد از دو بار سایه شده. ۱۶۳.....

- شکل ۴-۵۶ قسمت قرمز بعد گسترش برای بار دوم..... ۱۶۳
- شکل ۴-۵۷ بعد از گسترش با استفاده از کرنل ۴۶..... ۱۶۳
- شکل ۴-۵۸ استفاده از روش های ریخت شناسی برای حذف نویز..... ۱۶۴
- شکل ۴-۵۹ بررسی عملکرد الگوریتم CCL بر روی تصویر شامل چند شی ۱۶۶
- شکل ۴-۶۰ حذف نویز با استفاده از الگوریتم CCL..... ۱۶۷
- شکل ۴-۶۱ نتایج عملکرد فیلتر میانگین بر روی تصاویر به منظور حذف نویز..... ۱۶۸
- شکل ۴-۶۲ نتایج عملکرد فیلتر میانه بر روی تصاویر به منظور حذف نویز..... ۱۶۸
- شکل ۵-۱ داده های که با علامت (+) نشان داده شده اند، متعلق به کلاس اول. و داده های که با علامت (0) نشان داده شده است، متعلق به کلاس دوم می باشند. حال یک نقطه جدید داریم که با علامت (*) نشان داده شده است، هدف کلاسه بندی این است که تعیین کند این داده متعلق به کدام کلاس می باشد..... ۱۷۲
- شکل ۵-۲ خطی که بین داده های دو کلاس کشیده شده است و می توان با استفاده از تابع $F(x)$ آن را نشان داد، مرزی را تعیین می کند که داده های متعلق به هر کدام از کلاس ها را از هم جدا می کند..... ۱۷۴
- شکل ۵-۳ فضای جدید که داده ها در آن به صورت خطی جدا می باشند..... ۱۷۴
- شکل ۵-۴ فضای اولیه که داده ها در آن به صورت خطی جدا نمی باشند..... ۱۷۴
- شکل ۵-۵ خط قرمز سطح جداکننده را نشان می دهد روابط فوق برای فضای دو بعدی به تصویر کشیده شده است..... ۱۷۶
- شکل ۵-۶ کلاسه بندی داده ها با استفاده از الگوریتم حداقل مربعات..... ۱۷۸
- شکل ۵-۷ داده ها متعلق به دو کلاس می باشند داده های متعلق به هر کلاس با استفاده از علائم مورد استفاده جدا شده اند..... ۱۷۸

شکل ۵-۸ اثر داده های انحرافی بر روی نتیجه حداقل مربعات خطا..... ۱۷۸

شکل ۵-۹ گام های طی شده توسط الگوریتم پرسپترون برای پیدا کردن جداکننده بهینه (در داده های ایده آل داده ها تقریباً بطور کامل جداپذیر خطی می باشند. این الگوریتم در ابتدا مقداری که به بردار جداکننده انتساب می کند مقداری تصادفی می باشد، به همین دلیل تعدادی از داده ها به صورت نادرستی کلاسه بندی شده اند. در گام های دیگر رفته رفته این خطا کمتر می شود به طوری که در گام ۸ کاملاً داده ها به درستی کلاسه بندی می شود.)..... ۱۸۱

شکل ۵-۱۰ احتمال posterior..... ۱۸۴

شکل ۵-۱۱ توزیع چگال احتمال prior..... ۱۸۴

شکل ۵-۱۲ کانتور های ۳ کلاس زمانی که دارای کواریانس مشترک می باشند..... ۱۸۴

شکل ۵-۱۳ مرز جدا کننده بین داده های دو کلاس..... ۱۸۴

شکل ۵-۱۴ کانتور ها ۳ کلاس زمانی که کلاس سوم متمرکز تر می باشد..... ۱۸۴

شکل ۵-۱۵ مرز جدا کننده کلاس ها..... ۱۸۴

شکل ۵-۱۶ کانتور کلاس ها زمانی که یک کلاس نسبت به دو کلاس دیگر پراکنده تر می باشد... ۱۸۴

شکل ۵-۱۷ مرز جداکننده بین کلاس..... ۱۸۴

شکل ۵-۱۸ پیدا کردن مرز جداکننده با الگوریتم regression بر روی داده های ایده آل..... ۱۸۵

شکل ۵-۱۹ گام های طی شده توسط الگوریتم Iterative reweighted least squares برای پداکردن مرز جداکننده..... ۱۸۹

شکل ۵-۲۰ نتایج بدست آمده با استفاده از الگوریتم حداقل مربعات خطا، سطر ۴ نتایج الگوریتم حداقل مربعات خطا را نشان می دهد سایر سطر ها نتایج الگوریتم ها استاندارد می باشد..... ۱۹۱

شکل ۵-۲۱ نتایج استخراج اشیا توسط الگوریتم پرسپترون..... ۱۹۲

شکل ۵-۲۲	نتایج عمل بر روی داده های واقعی با الگوریتم Logistic regression	۱۹۳
شکل ۵-۲۳	نتایج بر روی داده ها ویدیو با استفاده از الگوریتم Iterative reweighted least squares	۱۹۴
شکل ۶-۱	ابتدا اختلاف بین پس زمینه و فریم جاری بدست می آوریم اختلاف بالاتر حد آستانه را نگه می داریم.	۲۰۵
شکل ۶-۲	نتیجه اعمال فیلتر میانه بر روی تصویر ۶-۱	۲۰۵
شکل ۶-۳	مقایسه الگوریتم جدید با الگوریتم رایج	۲۰۸
شکل ۶-۴	مقایسه حجم فشرده الگوریتم جدید با الگوریتم های رایج	۲۰۹
شکل ۶-۵	شکل الگوریتم ۲D logarithm	۲۱۱
شکل ۶-۶	الگوریتم جدید	۲۱۱
شکل ۶-۷	شکل بازیابی شده از الگوریتم CSD	۲۱۱
شکل ۶-۸	شکل بازیابی شده از الگوریتم UCBD	۲۱۱
شکل ۶-۹	شکل بازیابی شده از الگوریتم ۴SS	۲۱۱

فهرست جداول

جدول ۲-۱	رابطه بین PSNR و کیفیت تصویر	۱۶
جدول ۲-۲	نمونه ای از داده های همبسته	۱۸
جدول ۲-۳	داده های جدول ۲-۲ بعد از انتقال حول محور با زاویه چرخش ۶۸	۱۹
جدول ۲-۴	داده های بازیابی شده	۱۹
جدول ۲-۵	کد های پیشنهاد شده توسط الگوریتم هافمن	۳۰

- جدول ۲-۶ قسمتی از جدول کد هافمن استفاده شده در استاندارد JPEG..... ۴۱
- جدول ۲-۷ نسبت فشرده سازی با ضرایب کوانتیزه مختلف (این نسبت از حجم تصویر فشرده شده بر حجم تصویر اصلی بدست می آید)..... ۴۴
- جدول ۳-۱ بررسی اثر اندازه بلوک ها بر زمان کد کردن ۷۵
- جدول ۴-۱ در این جدول میزان حافظه مورد نیاز برای هر کدام از روش ها نشان می دهد..... ۱۴۷
- جدول ۴-۲ زمان مورد نیاز برای هر الگوریتم ها ۱۴۹
- جدول ۴-۴- میزان خطای چندی از الگوریتم را برای شرایط بیرونی نشان می دهد..... ۱۵۱
- جدول ۶-۱ توابع مختلف برای اندازه گیری میزان مشابهت..... ۲۰۰
- جدول ۶-۲ زمان مورد نیاز برای کد و دیکد بر روی ۴۰ فریم..... ۲۰۲
- جدول ۶-۳ حجم فشرده سازی بر روی ۴ فریم..... ۲۰۲
- جدول ۶-۴ مقایسه زمان محاسبات الگوریتم جدید با الگوریتم های رایج..... ۲۰۶
- جدول ۶-۵ مقایسه حجم فشرده الگوریتم جدید با الگوریتم های رایج..... ۲۰۹

فصل اول

مقدمه

۱-۱- ضرورت فشرده سازی

رسانه های ویدیویی معمولاً حجم زیادی از داده ها را تولید می کنند. مثلاً در تلوزیون های HDTV، حجم داده‌ی تولید شده در یک ثانیه از مرز یک گیگا بیت می‌گذرد. اگر ویدیویی با کیفیت تلوزیون های آنالوگ را در نظر بگیریم، آنگاه نرخ تولید ویدیو برابر ۱۶ مگابیت بر ثانیه خواهد بود که برای ۱ ساعت (۳۶۰۰*۱۶=۵۷۶۰۰) مگابیت خواهد شد. برای ویدئوهایی با کیفیت معادل VHS یا VCD نیز، حجم داده‌ی تولید شده تقریباً یک چهارم می‌شود که همچنان حجم داده بالایی می‌باشد.

انتقال این حجم عظیم اطلاعات عملاً بر روی شبکه های کم سرعت موجود غیر ممکن است. داده تولید شده در یک ثانیه توسط سیستم های ویدیویی از مرز یک گیگابیت می‌گذرد اما پهنای باندی که در دسترس عموم قرار دارد توان انتقال این حجم داده را ندارد و شبکه هایی که توان انتقال چند گیگا بیت بر ثانیه را دارند گران و استفاده آن برای بسیاری عملی نمی‌باشد.

این مسئله نه تنها گیربان گیر شبکه های کم سرعت امروزی هست، ذخیره کردن این حجم داده نیز غیر ممکن می‌باشد زیرا داده تولید شده در یک ساعت از مرز ترا بایت می‌گذرد و حافظه های جانبی که امروزه در بازار موجود و برای عموم قابل استفاده می‌باشد گنجایش حداکثر چند ترابایت را دارند. حتی اگر حافظه های جانبی توان ذخیره کردن این حجم داده را داشته باشند تمام پهنای باند گذرگاه داده، درگیر این انتقال خواهد شد. این اتلاف منابع، محدود به حافظه ی جانبی و پهنای باند نمی‌شود بلکه منابع دیگر مانند حافظه اصلی و واحد پردازش نیز به هدر می‌رود.

به همین دلیل نیازمند روش هایی هستیم که با کمترین حجم داده، بیشترین حجم اطلاعات را منتقل و یا ذخیره کنیم. فشرده سازی ویدئو یکی از مسائل اساسی در ذخیره سازی و انتقال اطلاعات می‌باشد. یکی از دلایل فراگیر شدن ویدئو دیجیتال در بین اقشار جامعه، استفاده از تکنیک های فشرده سازی ویدیویست.

۱-۲- تعریف فشرده سازی و انواع آن

یک اطلاع را می توان به صور مختلفی نمایش داد از بین روش های مختلف نمایش، هدف سیستم فشرده سازی این است که نمایشی را برگزیند که با استفاده از آن بتوان اطلاع یکسان را با حداقل داده ممکن، انتقال داد. نمایش بهینه (از جهت فشرده سازی) باید دارای دو ویژگی اساسی باشد. اول در نمایش بهینه باید به نسبت روش های نمایش دیگر، انتخاب کد برای پیام ها، هوشمندانه تر است. (به عنوان مثال فرض کنید ۵ پیام داریم یکی از این پیام های احتمال رخدادش برابر ۹۰٪ باشد و دو روش نمایش برای این پیام ها داشته باشیم در نمایش اول برای هر کدام از پیام ها سه بین اختصاص بدهیم در نمایش دوم برای پیام پر تکرار یک بیت و برای سایر پیام ها ۳ بیت اختصاص بدهیم. اگر فرض کنیم که اطلاعی داشته باشیم که دارای ۱۰۰۰ پیام باشد پس انتظار داریم ۹۰۰ پیام متعلق به پیام پر رخداد و ۱۰۰ پیام دیگر متعلق به گروه دیگر باشد. نمایش دوم ۱۸۰۰ بیت کمتر برای نمایش این اطلاع نیاز دارد. پس این روش دوم در انتخاب کد، هوشمندانه تر از جهت فشرده سازی عمل کرده است.) دومین ویژگی اینست که روش بهینه باید حداقل افزونگی ها (اطلاعات و پیام های تکرار) ممکن را داشته باشد.

روش های فشرده سازی را می توان به دو دسته بدون از دست دادن اطلاعات و با از دست دادن اطلاعات تقسیم کرد. در روش بدون از دست دادن اطلاعات باید اطلاعات منتقل شده در داده های اولیه و داده های فشرده شده عینا برابر باشد. در کاربرد هایی مانند یک فایل اجرایی حتی یک بیت از دست رفته می تواند کل برنامه را از بین ببرد بنابراین این روش برای کاربرد هایی که نباید هیچ تغییری در اطلاع اولیه به وجود آید، مناسب می باشد. اما در بسیاری از کاربرد ها داده تولید شده توسط یک سیستم الکترونیکی (مانند سیستم صوتی یا یک سیستم اخذ تصویر)، برای استفاده کاربر انسان، هست. یک تفاوت اساسی انسان و ماشین، برتری محسوس ماشین در اخذ جزئیاتی است که در محدوده ادراکی انسان نمی باشد. گاهی این جزئیات برای حواس انسان قابل لمس و در بعضی از مواقع

اختلاف این جزئیات برای حواس انسان قابل تمایز نمی باشد به عنوان مثال در یک تصویر می تواند ۱۰۰ نوع رنگ آبی وجود داشته باشد در حالی که دقیق ترین انسان شاید بیشتر از ۲۰ نوع آبی را نتوانند تمییز دهد. در این کاربرد ها می توان در عوض اطلاع اولیه از اطلاعی معادل، استفاده کرد که علاوه بر تفاوت با هم از نظر ماشین، برای انسان این دو اطلاع هم ارز می باشند. به این روش فشرده سازی که نه تنها نوع نمایش اطلاع بلکه خود اطلاع را نیز تغییر می دهد فشرده سازی با از دست دادن اطلاعات می گویند.

فشرده سازی ویدیو در دسته فشرده سازی با از دست دادن اطلاعات قرار می گیرد زیرا بسیاری از جزئیات (مانند فرکانس های بسیار بالا) درون تصویر برای انسان قابل لمس نمی باشد و با حذف آنها می توان داده فشرده تری بدست آورد. افزونگی مهمی که در ویدیو وجود دارد افزونگی بین فریم می باشد این افزونگی ناشی از اطلاعات تکراری در فریم های متوالی می باشد. در سیستم های فشرده سازی با بهره گیری از این نوع افزونگی، تلاش می کنند تا درصد فشرده سازی را بالا ببرند. در روش های رایج فشرده سازی برای استفاده از این نوع افزونگی، فریم های ویدیویی را به بلوک های تقسیم می کند. این بلوک ها با هم مقایسه می شوند تا از ذخیره کردن بلوک های تکراری صرف نظر شود.

جستجو بلوک ها از زمانبرترین مراحل در سیستم های فشرده سازی ویدیو می باشد علاوه بر آن، عملکرد نادرست این مرحله باعث اختلال در کارایی سیستم فشرده سازی می شود. ایراد وارده بر سیستم های فشرده سازی رایج در این است که همه بلوک ها را (چه در آن تغییر قابل توجه اتفاق افتاده و چه اتفاق نیفتاده باشد) مورد بررسی قرار می دهد این جستجوهای اضافه، علاوه بر این که زمان پردازش را بالا می برد گاهی بلوک هایی که بر اثر نویز تغییر کرده اند را با یک بلوک دیگر نادرست همتا تشخیص می دهد و در این صورت درصد فشرده سازی کاهش می یابد.

۱-۳- رهیافت پایان نامه

در این پایان نامه با استفاده از مدل سازی پس زمینه سعی کردیم نواحی که اشیا، پس زمینه یا نویز در آن واقع شده است را تعیین کنیم با استفاده از این اطلاعات تصمیم می گیریم در کدام یک از نواحی تغییر قابل لمس برای انسان اتفاق افتاده است و در نهایت جستجو را به این ناحیه محدود می کنیم مقایسه نتایج این الگوریتم جدید با چهار الگوریتم جستجو دیگر (2-D logarithmic search، 4SS,UCBD و CSD) نشان می دهد که به طور متوسط برای هر فریم الگوریتم های 2-D logarithmic search و 4SS, CBD, search به ترتیب به 0,1062, 0,0780, 0,1003 و 0,1156 ثانیه نیاز دارد در حالی که الگوریتم جدید ما در 0,0124 ثانیه اجرا می شود. که تقریباً یک دهم زمان محاسبات الگوریتم های یاد شده می باشد. نتایج همچنین نشان می دهد درصد فشرده سازی نیز به نسبت الگوریتم های یاد شده بهبود قابل توجهی پیدا می کند. تقریباً حجم داده فشرده شده سیستم جدید نصف سیستم های غیر بهینه می باشد.

۱-۴- ساختار پایان نامه

در شکل ۱-۱ مسیری که یک فریم (در سیستم ارائه شده در این پایان نامه) طی می کند، را نشان می دهد. این سیستم دارای سه مولفه اصلی می باشد مولفه اول تعیین می کند کدام نواحی در فریم باید جستجو شود برای این منظور در این مولفه ابتدا پس زمینه مدل می شود و اشیا از پس زمینه جدا می گردد خروجی این مرحله به مولفه حذف نویز ارسال و در این مولفه نویز ها حذف شده و خروجی آن به مولفه تشخیص سایه ارسال می شود و در گام نهایی پیکسل ها به یکی از سه کلاس شی، سایه و پس زمینه تقسیم می گردد و این داده ها به مولفه تعیین نواحی قابل جستجو ارسال و در این مولفه تصمیم گرفته می شود کدام نواحی مورد ارزیابی قرار گیرد.

مولفه دوم استاندارد MPEG2 را نشان می دهد اطلاعاتی که از مولفه اول بدست آمده به این مولفه فرستاده می شود تا در جستجو بلوک های فریم های نوع P و B فقط بلوک هایی که دارای تغییرات

قابل لمس می باشند جستجو انجام گیرد. بعد از جستجو و پیش بینی فریم با استفاده از فریم مرجع و بردار حرکت، تفاضل فریم جاری با فریم پیش بینی شده، محاسبه و برای ادامه فرایند فشرده سازی به مولفه دیگر که فشرده سازی JPEG می باشد ارسال می گردد.

خروجی مولفه دوم به مولفه نهایی ارسال می شود در این مرحله سرآینده هایی که برای همگام سازی و مدیریت خطا لازم است، به داده فشرده شده اضافه و در نهایت در فایل ذخیره می شود. (این مولفه پیاده سازی نشده) در فصول مختلف این تز هر کدام از این مولفه ها را با جزئیات بیشتری تشریح می کنیم. در ادامه اهداف هر کدام از فصل ها و ارتباط آنها را با سیستم ارائه شده بیان می کنیم.

۱-۴-۱- فصل اول (در این فصل اجزا و مولفه های استاندارد JPEG تشریح می شود)

در این فصل مختصرا اشاره می شود که چگونه می توان با نگاشت تصویر به فضای رنگی YCbCr، اطلاعات رنگی از اطلاعات درخشندگی جدا می گردد و از اطلاعات آن نمونه برداری کرد بدون آن که کیفیت تصویر تغییر عمده ای بکند. در ادامه ساختار کدگذار های تصاویر را تشریح می کنیم. کدگذار دارای سه قسمت کاهش افزونگی بین پیکسل ها، کوانتیزه کردن و کدگذار آنتروپی می باشد. هر کدگذار در گام اول لازم است تا داده ها را به فضایی نگاشت دهد که بین پیکسل ها کمترین همبستگی وجود داشته باشد. در ادامه سه روش برای کاهش همبستگی بین پیکسل ها (۱- کاهش همبستگی با استفاده از روش های کدینگ ۲- کاهش افزونگی بین پیکسل با استفاده انتقال به حوزه فرکانس و ۳- کاهش همبستگی با استفاده از روش پیشگو) را معرفی می کنیم. دومین گام هر کدگذار، کوانتیزه می شود، در این گام به جای این که برای هر کدام از اعداد (که نمایش دهنده یک سطح در تصویر می باشند)، سطوحی که به هم نزدیک می باشند را گروه بندی می کند و به این طریق بدون از دست رفتن کیفیت قابل توجه، زمان پردازش کاهش و درصد فشرده سازی افزایش می یابد. گام نهایی در یک کدگذار، کدگذار آنتروپی می باشد. در کدگذار آنتروپی، لازم است تا به هر کدام از گروه ها که از مرحله کوانتیزه بدست آمده است، یک عدد منسوب می شود بطوری که حجم داده فشرده شده

کاهش یابد. در ادامه این بخش به معرفی ۳ روش کدگذار آنتروپی، ۱- هافمن ۲- کد گذار ریاضی و ۳- LZW می پردازیم. بعد از معرفی اجزای یک کد گذار کلی به معرفی اجزای استانداردهای JPEG و JPEG۲۰۰۰ خواهیم پرداخت. در پایان به مقایسه و ارزیابی هر کدام از دو روش می پردازیم.

۱-۴-۲- فصل دوم (اجزای و مولفه های استاندارد MPEG استفاده شده در این سیستم

ارائه شده تشریح می شود)

در فصل سوم به بیان نحوه تولید و ذخیره سازی ویدیو می پردازیم. در ادامه به بیان معیار هایی که برای ارزیابی سیستم های فشرده سازی استفاده می شود، اشاره خواهد شد. دو استاندارد معروف فشرده سازی استاندارد های خانواده H و MPEG می باشند، که به معرفی آنها و تاریخچه آن می پردازیم. اجزای استاندارد H.۲۶۱ را مورد ارزیابی قرار می دهیم. این استاندارد دو نوع فریم I و P دارد مراحل کد کردن این نوع فریم ها تشریح خواهد شد. در استاندارد های خانواده MPEG به تشریح استاندارد MPEG۲ پرداخته و تفاوت های این استاندارد با استاندارد H.۲۶۱ بیان می شود. یکی از زمانبرترین قسمت ها در الگوریتم های فشرده سازی، انطباق بلوک می باشد. برای پیدا کردن بهترین انطباق الگوریتم های فراوانی وجود دارد که در این تز، الگوریتم های: (۱- الگوریتم جستجو Unrestricted Center-Biased Diamond-۲ الگوریتم جستجو ARPS-ZMP-۳ الگوریتم جستجو چهار گام (Four-Step Search Algorithm)-۴ جستجو لوزی (DIAMOND SEARCH ALGORITHM)-۵ الگوریتم جستجو کراس (CROSS-SEARCH ALGORITHM(CSA)-۶ الگوریتم جستجو شش وجهی (Hexagon-Based Search Pattern)) پیاده سازی و آنها را از جهت زمان و تعداد محاسبات برای پیدا کردن بلوک مناسب مورد ارزیابی قرار داده ایم.

۱-۴-۳- فصل سوم (در این بخش روش های مدل سازی پس زمینه، حذف نویز و

تشخیص سایه مورد استفاده در سیستم ارائه شده مورد بررسی قرار می

گیرند)

در فصل دوم بیان شد که افزونگی را می توان به دو دسته افزونگی درون فریم و افزونگی بین فریم تقسیم نمود. در افزونگی بین فریم ها دیدیم که حجم عمده یک فریم را پس زمینه تشکیل می دهد که در فریم های متوالی مشترک می باشد، آنچه تغییر می کند اشیا و تغییرات جزئی در پس زمینه (که قابل درک برای چشم انسان نمی باشند مانند امواج دریا یا لرزش های آرام شاخ و برگ درختان) می باشند، پس اگر بتوان پس زمینه را مدل کرد، می توان از ذخیره کردن حجم عمده فریم که مشترک می باشد، صرف نظر کرد و به این طریق به حجم بالایی از فشرده سازی در ویدیو دست یافت. در این فصل ابتدا به معرفی روش های مدل کردن پس زمینه می پردازیم. الگوریتم های مدل سازی پس زمینه ارائه شده در این فصل الگوریتم های: ۱- الگوریتم میانه ۲- الگوریتم PCA ۳- الگوریتم های مبتنی بر میانگین (۵ روش) ۴- الگوریتم مبتنی بر روش های آماری (روش های پارامتریک (۲ روش) و غیر پارامتریک (۲ روش)) می باشند. برای هر کدام از الگوریتم ها نحوه پیاده سازی و بررسی آن در شرایط داخلی و خارجی مورد تحلیل قرار می گیرد. در قسمت نهایی فصل به مقایسه این ۸ روش از جهت دقت، زمان محاسبات و حافظه مصرفی، از دو مرتبه و نیاز عملی به حافظه و زمان، بیان می کنیم. از آنجایی که اختلاف بین سایه اشیا و پس زمینه قابل توجه می باشد، بنابراین معمولا سایه اشیا نیز به عنوان شی در نظر گرفته می شود و در نتیجه اشیا استخراج شده دارای مشکلاتی می باشند (معمولا شی و سایه آن به عنوان یک شی در نظر گرفته می شوند). از آنجایی که نتایج این مرحله معمولا برای پردازش لایه های بالاتر در سیستم های نظارتی (مثلا سیستمی که از شی استخراج شده برای شناسایی افراد استفاده می کند) مورد استفاده قرار می گیرد، خطا این مرحله در کل سیستم منتشر می شو، و صحت عملکرد آن را تحت تاثیر قرار می دهد. روش های متفاوتی به

منظور تشخیص سایه از اشیا وجود دارد، در ادامه فصل به معرفی دو روش رایج برای حذف سایه می پردازیم. اشیا استخراج شده دارای نویز هایی می باشند، بنابراین لازم است، با استفاده از تکنیک های مرسوم اثر آنها را کاهش دهیم، در قسمت انتهایی فصل به معرفی این تکنیک ها (۱-میان ۳- میانگین ۳-CCCL- الگوریتم های ریخت شناسی (الف) باز کردن ب) بستن پ) گسترش ت) سایش)) می پردازیم.

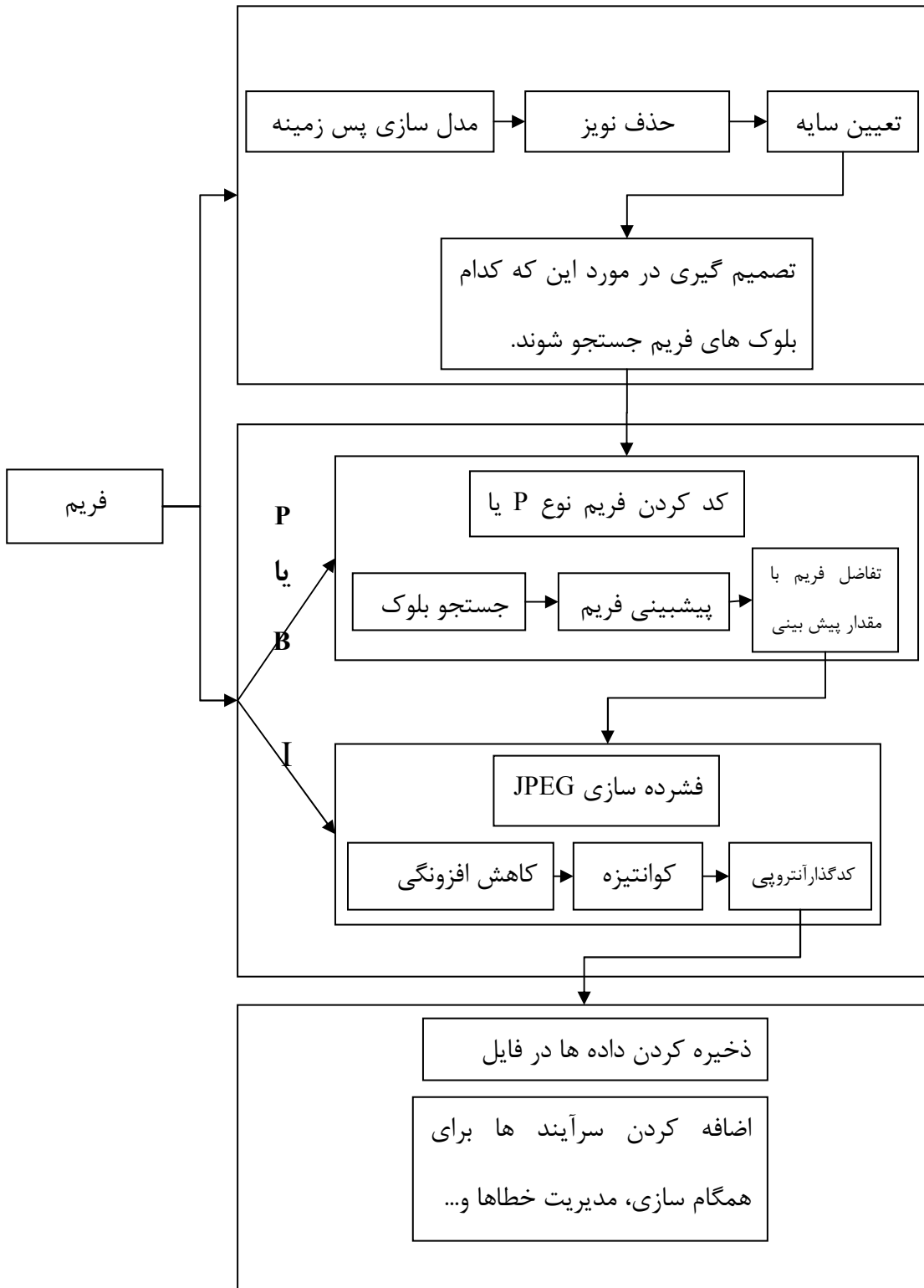
۱-۴-۴- فصل چهارم (جدا کردن اشیا از پس زمینه با استفاده از رهیافت جداکننده)

در فصل چهارم به معرفی راهکار رایج برای مدل کردن پس زمینه و استخراج اشیا از پس زمینه پرداختیم. در راهکار های ارائه شده در فصل چهارم دیدیم که روش های آماری علاوه بر کارایی در تقسیم کردن فریم ها به دو گروه اشیا و پس زمینه، از سه مشکل عمده (۱- بالا بودن حجم بالای محاسبات برای بدست آوردن مدل ۲- فرض مدلی خاص برای رفتار پس زمینه، در حالی که این فرض ممکن است با واقعیت فاصله معنا داری داشته باشد. ۳- و در نهایت در بسیاری از مواقع تعداد داده ای که برای آموزش مدل استفاده می شود ناکافی می باشد) رنج می برد. به همین دلیل لازم است تا از رهیافت دیگر (رهیافت جداکننده) که این مشکلات را ندارد، استفاده شود. برای مقابله با مشکلات فوق، در این فصل، با استفاده از رهیافت جداکننده، اشیا را از پس زمینه جدا می کنیم. در حالت ایده آل می توان داده ها را به دو کلاس پس زمینه و اشیا، که به صورت خطی جداپذیر می باشند، در نظر بگیریم. اگر مرز جداکننده تعیین شود، با استفاده از این تابع جداکننده، می توان اشیا را از پس زمینه جدا کرد. در این رهیافت، فرضی در مورد مدلی که داده ها را تولید می کند در نظر نمی گیرد. در ضمن زمانی که تابع جداکننده بدست آمد، دیگر لازم نیست مدام آموزش ببیند (اینکه هر چند فریم تابع باید بروز شود وابسته به محیط و تغییرات آن دارد) پس حجم محاسبات کمتری نیز دارد. از طرفی برای آموزش آن لازم نیست تعداد زیادی داده در دست داشته باشیم. در این فصل، چهار روش (۱- حداقل مربعات خطا ۲- پرسپترون ۳- مدل آماری ۴- Iterative reweighted least squares)

برای تعیین مرز جدا کننده اشیا معرفی می کنیم و در قسمت نهایی فصل هر کدام از الگوریتم ها ارائه شده را ارزیابی می کنیم.

۱-۴-۵- فصل پنجم (ارزیابی و نتایج سیستم جدید در مقایسه با استاندارد MPEG)

در این فصل بحث بر روی دو محور انجام می گیرد. در اولین ارزیابی، معیار مشابهت مورد استفاده در سیستم های فشرده سازی، را با معیار های مشابهت دیگر از جهت زمان و اثر آن بر روی زمان کد و دیکد مورد ارزیابی قرار دادیم. و در بخش انتهایی با استفاده از اطلاعاتی که از مدل کردن پس زمینه بدست آوردیم، کارایی سیستم های فشرده سازی ویدیو را بهینه کردیم این بهینه سازی نشان داد روش جدید از جهت زمان مصرفی و درصد فشرده سازی بر روی روش های موجود (۱- $2D$ - $UCBD$ - 3 - CSD - 4 - SS) برتری قابل توجه دارد و با حفظ این برتری ها از جهت کیفیت همتراز با الگوریتم های رایج می باشد.



شکل ۱-۱ اجزا و مولفه های سیستم پیشنهادی در این پایان نامه

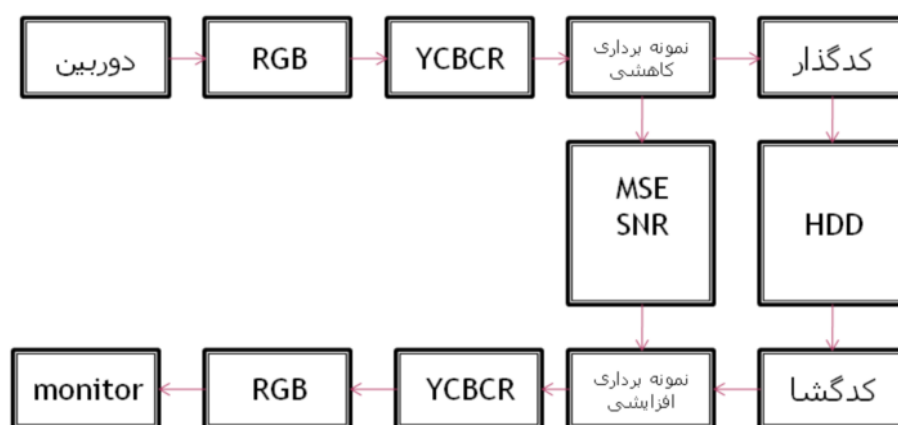
فصل دوم

اجزا و استانداردهای فشرده‌سازی تصاویر

۲-۱- مقدمه

فشرده سازی تصویر یکی از اقسام فشرده سازی داده است که در آن حجم تصاویر را برای اهداف مختلف کاهش می دهند. فشرده سازی به دو دسته فشرده سازی با از دست دادن اطلاعات و فشرده سازی بدون از دست دادن اطلاعات، تقسیم می گردد. در دسته اول، اطلاعات تصویر بازیابی شده با تصویر اصلی، اندکی در نمایش جزئیات، متفاوت می باشد. در این روش امکان دستیابی به درصد بالایی از فشرده سازی وجود دارد [۱۵]. این گونه روش برای کاربردهایی که داده فشرده شده باید بعد از بازیابی دقیقاً مانند داده اولیه باشد، (مانند فایل های متنی و فایل برنامه های اجرایی) مناسب نیستند. در روش فشرده سازی بدون از بین رفتن اطلاعات، اطلاعات بازیابی شده دقیقاً با اطلاعات اولیه برابر است، اما در این روش معمولاً میزان فشرده سازی داده محدود می باشد.

در شکل ۲-۱ طرح کلی سیستم فشرده سازی تصاویر نشان داده شده است. در این تصویر اجسام توسط دوربین به صورت سه مولفه RGB دریافت می شود. سپس سیستم این سه مولفه را مورد پردازش قرار داده و حجم آن را به کمک الگوریتم فشرده سازی کاهش می دهد. قسمت های مختلف این سیستم در بخش های بعدی توضیح داده می شوند.



شکل ۲-۱ طرح کلی سیستم فشرده سازی تصاویر دیجیتالی

۲-۱-۱- نمونه برداری کاهشی (Chrominance Sub sampling)

وقتی در یک سیستم دیجیتال از یک صحنه، تصویری عکاسی یا فریم ویدیویی برداشته می شود به صورت سه ماتریس ذخیره می شود که هر کدام از این ماتریس ها، داده های مرتبط با یکی از سه رنگ اصلی (RGB) را ذخیره می کند. فضای رنگ RGB برای فشرده سازی تصویر مناسب نمی باشد زیرا در این فضا، اطلاعات افزونه کمتری یافت می شود [۱۵]؛ به همین دلیل لازم است تا پیش از ادامه پردازش تصویر آن را به فضای رنگ مناسب که در آن افزونگی بین داده ها بیشتر نمایش داده می شود نگاشت دهیم.

فضایی که معمولا برای فشرده سازی تصاویر مورد استفاده قرار می گیرد فضای رنگ YCbCr می باشد. مولفه Y، میزان درخشندگی، Cr و Cb، اطلاعات رنگی تصویر را نشان می دهند. تفاوت سطح خاکستری با مولفه آبی و به طریق مشابه Cr، تفاوت سطح خاکستری با مولفه قرمز را نشان می دهد. تبدیلی که از آن برای انتقال فضای رنگ RGB به YCbCr استفاده می شود در رابطه (۱-۲) آمده است [۱۵]:

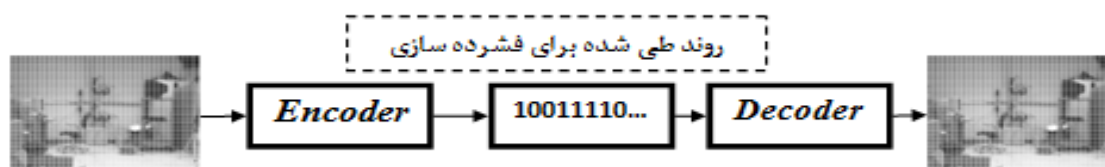
$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.334 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} * \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 0 \\ 128 \\ 128 \end{pmatrix} \quad (1-2)$$

بر اساس ویژگی های روانشناختی سیستم بینایی انسان (HSV)^۱، چشم به درخشندگی (مولفه Y) حساسیت بیشتری نسبت به Cb و Cr دارد [۳۷]. به همین دلیل می توان با نمونه برداری از مولفه های رنگی، حجم تصویر را کاهش داد بدون آن که اطلاعات از بین رفته، برای سیستم بینایی انسان قابل تمیز باشد.

^۱ Human System Vision

۲-۱-۲- اصول فشرده سازی تصویر

بعد از نمونه برداری از فضای YCbCr سه مولفه به وجود آمده به رمز گذار^۲ فرستاده می شود تا با استفاده از الگوریتم های فشرده سازی، حجم داده ها کاهش یابد.



شکل ۲-۲ روند طی شده توسط سیستم فشرده سازی تصویر، تصویر وارد کدگذار می شود و رشته ای از بیت ها تولید می گردد. اگر حجم این جریان بیتی کمتر از حجم تصویر اصلی باشد به هدف فشرده سازی دست یافته ایم. برای این که بتوانیم تصویر را بازیابی کنیم باید جریان بیتی به کدگشا^۳ وارد گردد تا تصویر اصلی بازیابی شود.

همانطوری که در شکل ۲-۲ نشان داده شد، تصویر اصلی وارد رمزگذار می شود و به رشته ای از بیت-ها تبدیل می گردد. اگر حجم این جریان بیتی کمتر از حجم تصویر اصلی باشد به هدف فشرده سازی دست یافته ایم. برای این که بتوانیم تصویر را بازیابی کنیم باید جریان بیتی به رمزگشا^۴ وارد گردد تا تصویر اصلی بازیابی شود.

اگر از فشرده سازی با از دست دادن اطلاعات استفاده کنیم، تصویر بازیابی شده با تصویر اصلی دارای تفاوت هایی خواهد بود. اما برای سیستم بینایی انسان دشوار است که بتواند این تفاوت را (در صورتی که از حدی کمتر باشد) تشخیص دهد. به منظور ارزیابی الگوریتم های فشرده سازی لازم است تا

^۲ Encoder

^۳ Decoder

معیارهایی را تعریف کنیم. یکی از معیارها نسبت فشرده سازی می باشد که می توان آن را به صورت (۲-۲) بیان نمود:

$$Crate = \frac{n_1}{n_2} \quad (2-2)$$

که در رابطه فوق n_1 حجم تصویر اصلی و n_2 حجم تصویر فشرده شده، را نشان می دهد. دو معیار استاندارد دیگر، یکی ریشه کمترین مربعات خطا (RSME) و دیگری نسبت سیگنال به نویز PSNR است و به ترتیب دارای روابط (۲-۳) و (۲-۴) می باشد:

$$RMSE = \sqrt{\frac{\sum_{x=1}^{W-1} \sum_{y=1}^{H-1} [f(x,y) - f'(x,y)]^2}{WH}} \quad (3-2)$$

$$PSNR = 20 \log_{10} \frac{255}{MSE} \quad (4-2)$$

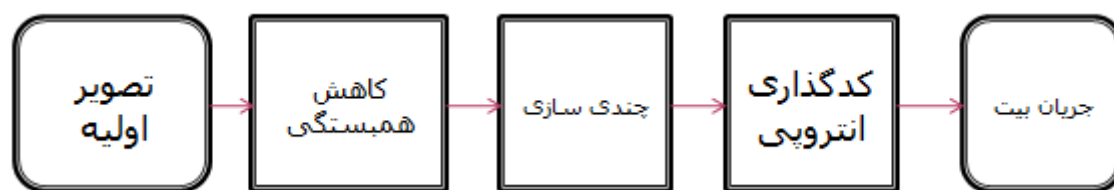
$$MSE = \sum_{x=1}^{W-1} \sum_{y=1}^{H-1} [f(x,y) - f'(x,y)]^2 \quad (5-2)$$

در عبارت فوق $f(x,y)$ مقدار پیکسل را در مکان x,y از تصویر اصلی و $f'(x,y)$ مقدار پیکسل را در موقعیت x,y از تصویر فشرده شده، نشان می دهد. هر چه مقدار RSME کوچکتر و مقدار PSNR بزرگتر باشد فشرده سازی مناسب تر می باشد. به کمک PSNR می توان نتایج فشرده سازی تصاویر را به کلاس هایی تقسیم کرد که در جدول ۲-۱ آمده است [۳۷].

جدول ۲-۱ رابطه بین PSNR و کیفیت تصویر

PSNR	زیر ۲۰	۲۰-۳۰	۳۰-۴۰	بالتر از ۴۰
کیفیت	غیر قابل قبول	ضعیف	خوب	عالی

تاکنون درک ساده ای از فشرده سازی تصویر ارائه شد. در ادامه روش های فشرده سازی با جزئیات بیشتر تشریح می شود. از آنجایی که در اکثر روش های فشرده سازی تصویر، از روش با از دست دادن اطلاعات استفاده می گردد، در این نوشتار بر روی این نوع فشرده سازی تمرکز خواهد شد. روند کلی فعالیت هایی که در این سیستم لازم است در شکل ۲-۳ نشان داده شده است.



شکل ۲-۳ روند نمای کلی سیستم فشرده سازی تصویر با از دست دادن اطلاعات، ابتدا افزونگی بین داده ها کاهش می یابد در گام دوم خروجی چندی سازی شده و در نهایت خروجی به کدگذار انتروپی ارسال می شود.

۲-۲- کاهش همبستگی بین پیکسل ها

معمولا پیکسل های مجاور در یک تصویر مشابه می باشند. به عبارتی بین پیکسل های مجاور همبستگی وجود دارد. با اعمال تبدیل بر روی داده های تصویر و کاهش همبستگی بین پیکسل های مجاور حداکثر نرخ فشرده سازی قابل استحصال خواهد بود. برای کاهش همبستگی بین پیکسل ها از روش هایی چون تبدیل کد، کاهش افزونگی در حوزه فرکانس و روش پیشگو استفاده می شود.

۲-۲-۱- کدگذاری تبدیلی Transform Coding

ایده این روش این است که همبستگی بین پیکسل ها، با چرخش محور های مختصات قابل تغییر می باشد. فرض کنید مجموعه نقاط جدول 2×2 X را در فضای دو بعدی داریم و می خواهیم آنها را بر روی فضای دو بعدی دیگر نگاشت دهیم. همانطوری که در شکل ۲-۴ مشاهده می شود داده ها حول یک خط که با محور X زاویه ۶۸ درجه می سازد انباشته شده اند [۱۷]. از آنجایی که داده ها حول این

خط انباشته شده، می توان به کمک یک تبدیل، داده ها را ۶۸ درجه دوران داد. دوران در جبر ماتریسی به صورت (۶-۲) نمایش داده می شود:

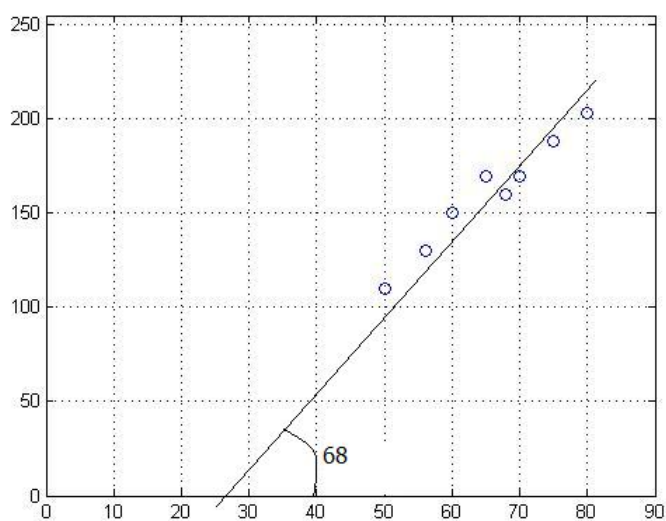
$$Y = AX \quad (۶-۲)$$

ماتریس A که ماتریس دوران و دارای رابطه (۷-۲) می باشد.

$$A = \begin{bmatrix} \cos \theta & \sin \theta \\ -\sin \theta & \cos \theta \end{bmatrix} \quad (۷-۲)$$

جدول ۲-۲ نمونه ای از داده های همبسته

مختصات X	مختصات Y
۶۵	۱۷۰
۷۵	۱۸۸
۶۰	۱۵۰
۷۰	۱۷۰
۵۶	۱۳۰
۸۰	۲۰۳
۶۸	۱۶۰
۵۰	۱۱۰



شکل ۴-۲ نمایش همبستگی بین داده های جدول ۲-۲

که در آن θ زاویه بین محور x ها و خطی که می خواهیم دوران دهیم در مثال فوق زاویه چرخش برابر 68° می باشد. بعد از نگاشت، همانطوری که در جدول ۲-۳ مشاهده می شود غالب داده ها حول محور X ها انباشته می شوند. در فضای جدید مقادیر حول محور x به صفر نزدیک می باشند. به همین دلیل می توان با صرف نظر از آن، بدون آنکه اطلاعات زیادی از دست داد، به فشرده سازی مناسبی دست یافت.

جدول ۲-۳ داده های جدول ۲-۲ بعد از انتقال حول محور با زاویه چرخش 68°

مختصات X	مختصات Y
۱۸۱,۹۷۰۷	۳,۴۱۶۲
۲۰۲,۴۰۶۱	۰,۸۸۷۳
۱۶۱,۵۵۴۰	۰,۵۶۰۰
۱۸۳,۸۴۳۷	-۱,۲۱۹۷
۱۴۱,۵۱۱۹	-۳,۲۲۳۴
۲۱۸,۱۸۶۸	۱,۸۷۰۴
۱۷۳,۸۲۲۷	-۳,۱۱۱۴
۱۲۰,۷۲۰۶	-۵,۱۵۲۵

با صفر قرار دادن مقدار y ها می توان به فشرده سازی مناسب رسید. برای بازیابی می توان x را به کمک تبدیل معکوس بازیابی کرد. داده بازیابی شده، بسیار به داده اولیه نزدیک است. نتیجه این اعمال را می توان در جدول ۴-۲ مشاهده نمود.

جدول ۴-۲ داده های بازیابی شده

مختصات X	مختصات Y	$\hat{X} = A^{-1}Y$	مختصات X	مختصات Y
۱۸۱,۹۷۰۷	۰		→	۶۸,۱۶۷۴
۲۰۲,۴۰۶۱	۰		۷۵,۸۲۲۶	۱۸۷,۶۶۷۶
۱۶۱,۵۵۴۰	۰		۶۰,۵۱۹۱	۱۴۹,۷۹۰۲
۱۸۳,۸۴۳۷	۰		۶۸,۸۶۹۰	۱۷۰,۴۵۶۹
۱۴۱,۵۱۱۹	۰		۵۳,۰۱۱۲	۱۳۱,۲۰۷۵
۲۱۸,۱۸۶۸	۰		۷۷,۲۱۸۹	۱۹۱,۱۲۳
۱۷۳,۸۲۲۷	۰		۶۵,۱۱۵۱	۶۱,۱۶۵۵
۱۲۰,۷۲۰۶	۰		۴۵,۲۲۲۷	۱۱۱,۹۳۰۱

تبدیل کارهانون

برای کاهش همبستگی بین داده ها، بخصوص برای داده های چند بعدی نظیر تصویر کار آسانی نیست کارهانون روشی برای پیدا کردن این تبدیل بهینه ارائه کرده است (این روش معمولاً در منابع به صورت مخفف KLT بیان می شود [۳]). فرض کنید داده های ورودی را به صورت X و ماتریس کواریانس آن به صورت C_{xx} نمایش داده می شود که از رابطه (۸-۲) بدست می آید.

(۸-۲)

$$C_{xx} = E \left[(X - E(X))(X - E(X))^T \right] = \begin{bmatrix} E[x_0^2] & E[x_0x_1] & \dots & E[x_0x_{N-1}] \\ E[x_1x_0] & E[x_1^2] & \dots & E[x_1x_{N-1}] \\ \vdots & \vdots & \ddots & \vdots \\ E[x_{N-1}x_0] & E[x_{N-1}x_1] & \dots & E[x_{N-1}^2] \end{bmatrix}$$

برای رسیدن به کمترین همبستگی باید ماتریس کواریانس قطری و عناصر قطر، دارای حداکثر مقدار ممکن شود. زیرا هر چه عناصر غیر قطری کوچکتر شود نشان می دهد که وابستگی بین متغیرهای متفاوت کم و به عبارتی از هم مستقل هستند. اگر Y برداری باشد که از تبدیل بردار ورودی X به کمک تبدیل A باشد، آنگاه ماتریس کواریانس به صورت C_{yy} نمایش داده می شود و از روابط (۹-۲) و (۱۰-۲) بدست می آید.

$$C_{yy} = E \left[(Y - E(Y))(Y - E(Y))^T \right] \quad (۹-۲)$$

$$C_{yy} = E \left[(AX - E(AX))(AX - E(AX))^T \right] \quad (۱۰-۲)$$

و اگر همه مقادیر بردار X را از مقدار میانگین کم کنیم آنگاه داریم:

$$C_{yy} = E \left[(AX)(AX)^T \right] \quad (۱۱-۲)$$

که با انجام محاسبات داریم:

$$C_{yy} = AC_{xx}A^T \quad (12-2)$$

اگر A ماتریسی باشد که ستون های آن، بردار های ویژه^۵ باشد آنگاه C_{yy} ماتریس قطری می شود که عناصر قطر آن مقادیر ویژه می باشند که به صورت زیر نمایش داده می شود:

$$C_{yy} = \begin{bmatrix} \gamma_0 & 0 & \dots & 0 \\ 0 & \gamma_1 & \dots & 0 \\ \cdot & \cdot & \dots & \cdot \\ 0 & 0 & \dots & \gamma_{N-1} \end{bmatrix} \quad (13-2)$$

مهم ترین مزیت این تبدیل آن است که تصویر بدست آمده، دارای کمترین وابستگی بین پیکسل ها می باشد. برای فشرده کردن داده، تعدادی از بردار های ویژه را که دارای مقادیر ویژه بالایی هستند را نگه می داریم و بقیه بردار ها را حذف می کنیم با نداشتن داده ها با استفاده از این بردار ها داده هایی استحصال خواهد شد که ابعاد کمتری دارند و به این طریق حجم داده ها کاهش می یابد. اما عیب آن این است که محاسبه بردار های ویژه برای داده های بزرگ دشوار است. علاوه بر آن برای بازیابی تصویر لازم است تا مقادیر ویژه و بردارهای ویژه را ذخیره کرد تا در هنگام بازیابی از آنها استفاده شود به همین دلیل از این تبدیل معمولاً برای فشرده سازی تصاویر استفاده نمی شود.

۲-۲-۲- کاهش همبستگی در حوزه فرکانس

مجموعه ای از نقاط یک تصویر را بر روی صفحه در نظر بگیرید. برای درج آن تصویر از روش های مختلف می توان استفاده کرد. مثلاً یک روش برای درج نقاط می توان از فاصله بین آنها استفاده کرد اما این روش ثبت یک تصویر پر در دسر می باشد. زیرا برای نمایش N نقطه، لازم است از $O(N^2)$ داده

^۵ Eigen Vector

استفاده شود. اما روش رایج برای به تصویر کشیدن نقاط، استفاده از محورهای مختصات می باشد. برای نمایش یک نقطه در یک صفحه، می توان از دو مولفه استفاده کنیم.

بنابراین سوال این است که آیا ممکن است تا ارتباط بین توابع را بیابیم. در فضای توابع نیز می توان از مفهوم معادل به نام توابع متعامد استفاده کنیم. محور ها در این حالت، توابع متعامد می باشند. تعامد را می توان به صورت رابطه (۲-۱۴) نشان داد:

$$\{\varphi_i(x) | i = 1, 2, \dots, n\} \quad (2-14)$$

$$\int_{-\infty}^{\infty} \varphi_i(x) \varphi_j(x) dx \begin{cases} 0 & \text{if } i \neq j \\ 1 & \text{if } i = j \end{cases} \quad (2-15)$$

به عبارت دیگر نگاشت هر کدام از توابع متعامد بر روی هم برابر صفر می شود و نگاشت این تابع بر روی خودش برابر یک می شود. بنابراین برای نمایش یک تابع دلخواه با استفاده از توابع متعامد، می توان از حاصل جمع تصویر تابع روی هر کدام از توابع متعامد استفاده کرد. براین اساس تابع را می توان به صورت (۲-۱۶) نمایش داد.

$$f(x) = \sum_{i=1}^{\infty} c_i \varphi_i(x) \quad (2-16)$$

حال لازم است تا ضرایب (c_i) را بدست آوریم. برای این کار طرفین معادله فوق را در تابع متعامد φ_i ضرب و انتگرال معین از منهای بی نهایت تا مثبت بی نهایت را محاسبه می کنیم. این انتگرال برای تمام مقادیر به جز c_i برابر صفر می شود. پس داریم:

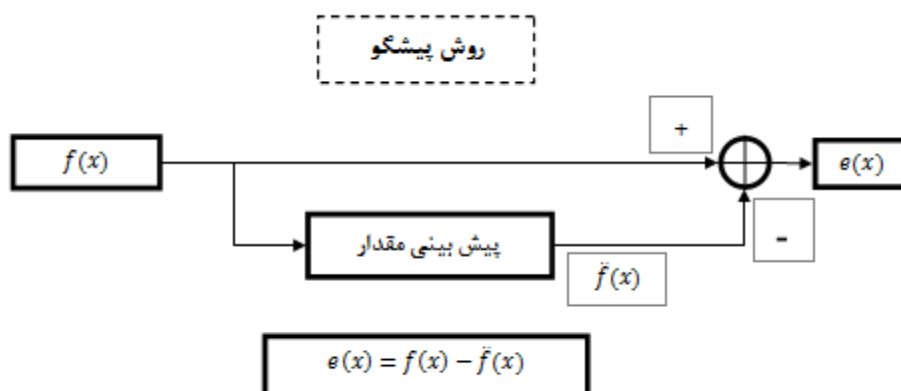
$$c_i = \int_{-\infty}^{\infty} f(x) \varphi_i(x) dx \quad (2-17)$$

توابع متعامد فراوانی را می توان نام برد که می توان برای نمایش توابع از آنها استفاده کرد. یک دسته از این توابع، توابع سینوسی می باشند زمانی که از توابع سینوسی برای نمایش استفاده می کنیم به

این تبدیل، تبدیل فوریه می گویند [۱۷]. تبدیل فوریه کاربرد های فراوانی دارد. یکی از این کاربرد ها در حوزه فشرده سازی تصویر می باشد. می دانیم که چشم انسان قادر به تشخیص فرکانس های بسیار بالا نمی باشد [۱۵]، به همین دلیل در نمایش تصویر می توانیم از فرکانس های بالاتر از حد آستانه بینایی انسان، چشم پوشی کرد. بنابراین برای فشرده سازی تصویر، ابتدا تصویر را با استفاده از تبدیل فوریه به حوزه فرکانس می بریم و در حوزه فرکانس، فرکانس های بالا را حذف می کنیم. در استاندارد JPEG از تبدیل فوریه گسسته سینوسی برای انتقال به حوزه فرکانس استفاده می شود [۳۶] و در استاندارد JPEG۲۰۰۰ از موجک ها برای انتقال تصویر به حوزه های که در آن داده ها کمترین همبستگی را داشته باشند، استفاده می کند [۳۶].

۲-۲-۳- کد گذاری پیشگو (Predictive coding)

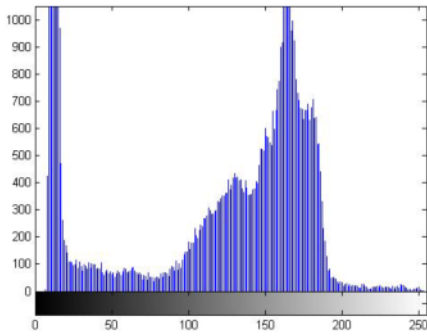
در این روش به جای مقدار پیکسل، تفاوت آن با پیکسل همجوار انتقال داده می شود. از آنجایی که پیکسل های همسایه به هم نزدیک می باشند تفاضل مقادیر می تواند کوچک و نزدیک به صفر باشند. شکل ۲-۵ سیستم ساده از *predictive coding* را نشان می دهد. در این سیستم، f پیکسل تصویر اصلی، \hat{f} مقدار پیشگویی آن پیکسل توسط سیستم، و e مقدار خطایی است که بین پیکسل موجود و مقدار پیش بینی شده وجود دارد.



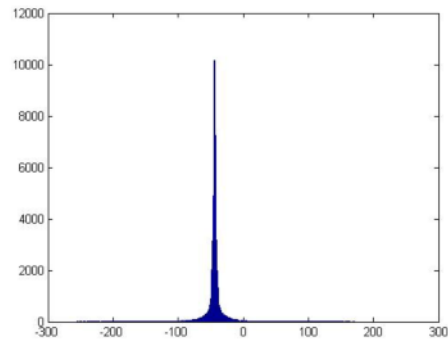
شکل ۲-۵ طرح یک سیستم پیشگو ساده برای فشرده سازی تصویر، در این روش به جای این که مقدار پیکسل را انتقال دهیم مقدار تفاوت آن را با پیکسل های بعدی انتقال می دهیم از آنجایی که پیکسل های همسایه به هم نزدیک می باشند تفاضل مقادیر می تواند کوچک و نزدیک به صفر باشند.

همانطوری که در شکل ۲-۶ به تصویر کشیده شده است هیستوگرام یک تصویر به طور نسبتاً یکنواختی پراکنده است. به همین دلیل اگر از کدهای با طول متغیر نیز استفاده کنیم [۳۵]، تاثیر چندانی در فشرده سازی ندارد. اما در تصویر تفاضلی، مقادیر تفاضل ها در اطراف یک عدد خاص متمرکز یافته است. بنابراین می توان داده ها را بیشتر فشرده کرد [۳۷].

از روش پیشگو برای کاهش افزونگی بین فریم ها در تصاویر ویدیویی استفاده می شود. به کمک آن می توان تغییرات لحظه ای در فریم ها را کنترل و ویدیو ها را فشرده کرد. در شکل ۲-۷ تصویر اصلی (الف) و تصویر تفاضلی (ب) حاصل از آن نشان داده شده است. هیستوگرام های نشان داده شده در شکل ۲-۶ مربوط به این تصاویر می باشند. همانطور که مشاهده می شود هیستوگرام تصویر خروجی سیستم پیشگو حول یک عدد خاص متمرکز می باشد. به همین دلیل با استفاده از روش های انتروپی درصد فشرده سازی آن بیشتر خواهد شد [۳۵].



ب) هیستوگرام تصویر اصلی



الف) هیستوگرام تصویر خروجی از سیستم پیشگو

شکل ۲-۶ تصویر سمت چپ هیستوگرام تصویر پیش از پردازش و هیستوگرام سمت راست، هیستوگرام تصویر پردازش شده را نشان می دهد. هیستوگرام یک تصویر به طور یکنواخت پراکنده شده است. به همین دلیل اگر از کد با طول متغییر نیز استفاده کنیم، تاثیر چندانی را در فشرده سازی ندارد. اما همانطوری که در تصویر تفاضلات می بینید، مقادیر تفاضل ها در اطراف یک عدد خاص تمرکز یافته است. به همین دلیل در فشرده سازی به درصد بالاتری از فشردگی دست خواهیم یافت.



ب) تصویر اصلی



الف) تصویر اختلاف مقادیر متوالی خروجی سیستم پیشگو

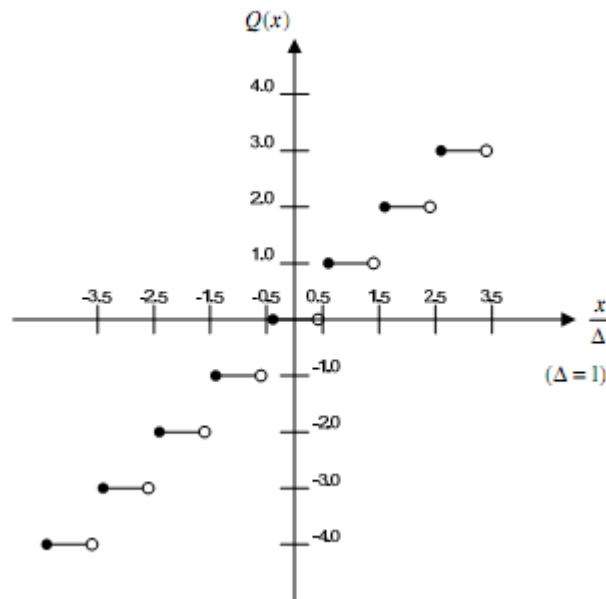
شکل ۲-۷ اثر گذار پیشگو بر روی تصویر هیستوگرام های این تصاویر را به ترتیب در الف و ب در تصویر ۲-۶ نشان داده شده است. همانطور که مشاهده می شود هیستوگرام تصویر خروجی سیستم پیشگو حول ۰ متمرکز می باشد. به همین دلیل با استفاده از روش های انتروپی حجم فشرده سازی آن بیشتر خواهد شد.

۳-۲- چندی سازی (Quantization)

در این قسمت روش‌هایی مورد بحث قرار می‌گیرد که با تعریف بازه‌هایی جدید، روشی به نام کوانتیزیشن ارائه می‌کند که در آن تنوع داده‌های خروجی تقلیل می‌یابد و به این طریق به فشرده کردن داده‌ها کمک می‌کند. *Quantization* به دو دسته یکنواخت و غیر یکنواخت تقسیم می‌شود. در روش یکنواخت بازه داده ورودی، به قسمت‌هایی به اندازه یکسان تقسیم و برای هر کدام از اعداد درون بازه یک عدد تخصیص داده می‌شود. در صورتی که به ازای هر بازه، از مقدار میانی بازه استفاده کنیم به رابطه *quantization* (۱۸-۲) می‌رسیم.

$$Q(x) = \frac{x}{\Delta} + 0.5 \quad \Delta = 1 \text{ step size} \quad (18-2)$$

شکل ۸-۲ مقدار کوانتیزیشن را با طول بازه نشان می‌دهد.



شکل ۸-۲ نمودار تصویر چندی سازی یکنواخت با طول گام ۱

آنچه می توان از این نمودار به آن اشاره کرد این است که هرچه اندازه بازه ها بزرگتر باشد حجم داده بیشتر کاهش خواهد یافت. از طرفی با انجام اینکار دقت کاهش می یابد و اطلاعات بیشتری از بین می رود. در چندی سازی غیر یکنواخت به جای تقسیم بازه ها به اندازه های مساوی، بر اساس احتمال وقوع هر بازه، بازه ها سگمنت بندی می شود. در این روش بازه های پر تکرارتر را کوچکتر و بازه های کم تکرار را به بازه های بزرگتر تقسیم می کنند [۳۶].

۲-۴- رمزگذاری آنتروپی (Entropy Coding)

بعد از چندی سازی ضرایب، به کمک کدگشا ضرایب به جریان بیت تبدیل می شوند. نکته ای که در رمزگذار آنتروپی مورد استفاده قرار می گیرد این است که به ضرایبی که احتمال وقوع بالاتری دارد کد کوچکتر و برای ضرایب کم رخداد کد های بزرگتری اختصاص داده می شود. روش هایی برای کدگذاری آنتروپی وجود دارد که در ادامه به بعضی از آنها اشاره می شود.

۲-۴-۱- رمزگذاری هافمن (Huffman Coding)

این روش جز روش های کد گذاری با طول متغیر می باشد و در متد های فشرده سازی مانند jpeg و mpeg مورد استفاده قرار می گیرد. کد هافمن به طور میانگین، کوتاهترین کد را برای منبع اطلاعات فراهم می کند. هافمن الگوریتم حریصانه ای را طراحی کرد که از طریق ساخت یک درخت دودویی متناظر با یک کد بهینه، یک کد بهینه دودویی را تولید می کند [۱۸].

در شبه کدی که در ادامه می آید فرض می شود که C یک مجموعه از n کاراکتر است و هر کاراکتر $c \in C$ یک شی با تعداد تکرار تعریف شده $f[c]$ است. الگوریتم هافمن درخت T متناظر با کد بهینه را به روش پایین به بالا می سازد. این الگوریتم با یک مجموعه از $|C|$ برگ شروع می کند و یک توالی از $1 - |C|$ عمل ادغام را اجرا می کند تا درخت مورد نظر بدست آید. از صف اولویت مینیمم، برای پیدا

کردن کلیدهای با کمترین تکرار استفاده می شود. نتیجه ادغام دو شی، شی جدیدی است که تعداد تکرار آن، برابر مجموع تعداد تکرار دو شی است که از ادغام آنها شی جدید به وجود آمده است.

HUFFMAN(c)

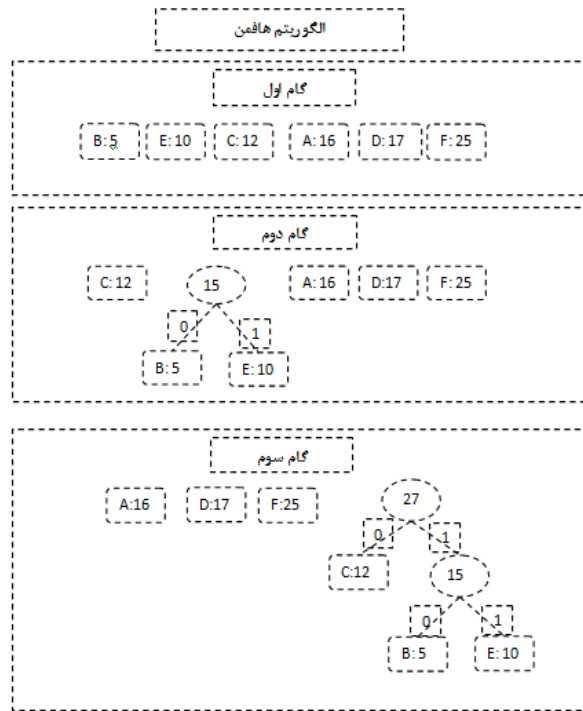
۱. $n=|C|$
۲. $Q=C$
۳. For $i=1$ to $n-1$
۴. .Do allocate a new node Z
۵. .Left[z]= x =extract_min(Q)
۶. .Right= y =extract_min(Q)
۷. .F[z]=F[x]+F[y]
۸. .Insert(Z,Q)
۹. Return extract_min(Q)

الگوریتم ۲-۱ شبه کد فوق الگوریتم هافمن را نشان می دهد.

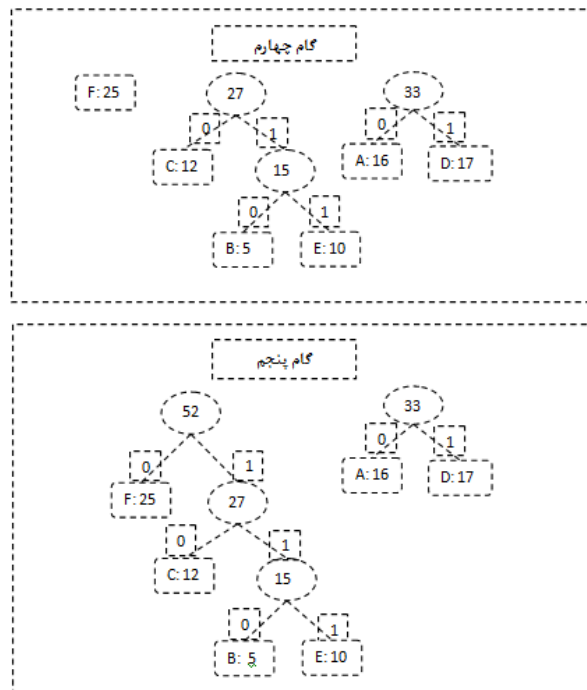
در خط دوم، صف مینیمم اولویت Q با استفاده از کارکترهای مجموعه C مقدار دهی می شود. حلقه *for* به صورت پی در پی دو گره x,y با کمترین مقدار از درون صف خارج می کند آنگاه این دو گره را با هم ادغام می کند و سپس درون گره Z قرار داده، و گره Z را در صف Q قرار می دهد.

برای توجیه بهتر فرض کنید مجموعه کارکترهای A, \dots, F با تعداد تکرار زیر داریم برای ذخیره این داده ها در روش معمول به ۳ بیت نیاز است پس کلا به ۱۸ بیت نیاز است در صورتی که از کد هافمن استفاده شود همانطور که در جدول ۲-۵ نشان داده شده، تعداد کل بیت ها برابر ۱۴ خواهد شد. مراحل الگوریتم در شکل های ۲-۹ تا ۲-۱۱ آمده است.

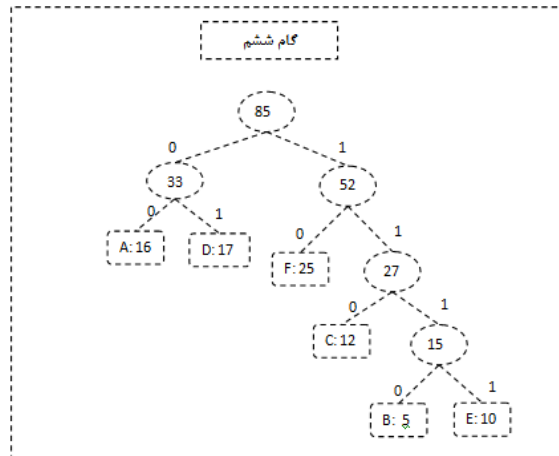
A	B	C	D	E	F
۱۶	۵	۱۲	۱۷	۱۰	۲۵



شکل ۲-۹ گام های اول تا سوم برای الگوریتم هافمن



شکل ۲-۱۰ گام های چهارم تا پنجم برای الگوریتم هافمن



شکل ۱۱-۲ گام نهایی در الگوریتم هافمن

پس کد ها پیشنهادی برای سمبول ها با فراوانی داده شده در جدول ۲-۵ آمده است.

کد هافمن دارای سه خاصیت مهم می باشد که در زیر لیست شده است

۱. پیشوند یکتا: در این خاصیت پیشوند ها هیچ دو کدی یکسان نمی باشند.
۲. رمز گشایی سریع: با توجه به آن که پیشوند ها یکتا می باشد بازیابی آن سریع می باشد. به عنوان مثال اگر کد ضربی، یک باشد به محض آن که این کد را در ورودی دیده شود، دیگر لازم نیست حروف دیگر را مورد ارزیابی قرار گیرد.
۳. بهینه: کد هایی که توسط الگوریتم هافمن تولید می شود بهینه می باشد.

جدول ۲-۵ کد های پیشنهاد شده توسط الگوریتم هافمن

a	b	C	D	e	f
۰۰	۱۱۱۰	۱۱۰	۰۱	۱۱۱۱	۱۰

اما این الگوریتم دارای دو مشکل می باشد:

- استاتیک می باشد در صورتی که توزیع احتمال ها تغییر کند لازم است تا دوباره درخت بازنویسی شود.
- فضای زیادی برای انجام الگوریتم، نیاز می باشد.

۲-۴-۲- رمزگذاری ریاضی (Arithmetic Coding)

این روش کد گذاری یکی از مهم ترین روش های کد گذاری است که در پردازش تصویر و ویدیو مورد استفاده قرار می گیرد. کد هافمن به سمبولی که دارای احتمال رخداد نزدیک به یک می باشد کد یک اختصاص می دهد. حالی که طبق قانون آنتروپی اطلاعاتی که از آن قابل اخذ می باشد نزدیک به ۰ می باشد. به همین دلیل اختصاص یک بیت به آن هزینه بسیاری را در پی دارد. کد این الگوریتم در زیر آمده است. برای جزئیات بیشتر این روش به [۳۷] مراجع شود.

۱. Input symbol is l
۲. Previous_s_low is lower bound for the old interval
۳. Previous_s_high is lower bound for the old interval
۴. Range is Previous_s_high- Previous_s_low
۵. Previous_s_low= l , Previous_s_high= h ,
۶. Rang= Previous_s_high- Previous_s_low
۷. While(input symbole $l \neq$ EOF)
۸. Get input symbol l
۹. Range= Previous_s_high- Previous_s_low
۱۰. New_previous_s_low= Previous_s_low+Range*interval_low of l
۱۱. New_previous_s_low=Previous_s_high+Range*interval_high of l
۱۲. End

الگوریتم ۲-۲ این شبه کد، الگوریتم کد گذار ریاضی را نشان می دهد.

۲-۴-۳- کدگذاری LZW

این الگوریتم برای اولین بار در سال ۱۹۷۷ ارائه شد مبنای کار آن استفاده از یک کد برای دنباله ای از کاراکترها (رشته) است. در این روش فشرده سازی هیچ گونه آنالیزی بر روی متن ورودی انجام نمی گیرد. اگر الگوی رشته ورودی، قبلاً مشاهده نشده باشد کد جدیدی تولید و در یک جدول ذخیره می شود. در صورتی که الگو قبلاً مشاهده شده باشد می توان به جای آن رشته، از کد آن استفاده نمود [۳۸]. در استفاده از تکنیک کدگذاری LZW از دو الگوریتم مجزا برای فشرده سازی و کدگشا استفاده می شد که در زیر مختصر توضیح داده شده است. برای مشاهده مثال های بیشتر به مرجع [۳۸] مراجعه شود.

الف) فشرده سازی (Compression)

در استفاده از الگوریتم lzw برای فشرده سازی، برای رشته هایی که کدهای آن تاکنون مشخص شده است، از کد آن استفاده می شود. رشته هایی که کد آن نامعلوم باشد کد جدید به آن اختصاص داده می شود و در جدول ذخیره می گردد. شبه کد زیر مراحل طی شده توسط الگوریتم فشرده سازی را نشان می دهد.

۱. STRING = get input character
۲. WHILE there are still input characters DO
۳. CHARACTER = get input character
۴. IF STRING+CHARACTER is in the string table then
۵. STRING = STRING+character
۶. ELSE
۷. output the code for STRING
۸. add STRING+CHARACTER to the string table
۹. STRING = CHARACTER
۱۰. END of IF
۱۱. END of WHILE
۱۲. output the code for STRING

الگوریتم ۲-۳ این شبه کد الگوریتم رمزگذار LZW را نشان می دهد.

ب) کدگشایی (Decompression)

الگوریتم همراه الگوریتم کد گذار در روش LZW، الگوریتم کدگشا می باشد. این الگوریتم لازم است تا جریان کاراکترهای خروجی الگوریتم compression را، به عنوان ورودی دریافت و رشته های ورودی به الگوریتم compression را تولید کند. یکی از دلایل کارایی الگوریتم lzw این است که، لازم نیست جدول الگوریتم فشرده سازی را به عنوان ورودی به الگوریتم Decompression ارسال شود. این الگوریتم دقیقاً همان جدولی را تولید می کند که کدگذار تولید می کند این کار عملی است زیرا زمانی که به کد نیاز است قبلاً محاسبه می شود.

۱. Read OLD_CODE
۲. output OLD_CODE
۳. WHILE there are still input characters DO
۴. Read NEW_CODE
۵. STRING = get translation of NEW_CODE
۶. output STRING
۷. CHARACTER = first character in STRING
۸. add OLD_CODE + CHARACTER to the translation table
۹. OLD_CODE = NEW_CODE
۱۰. END of WHILE

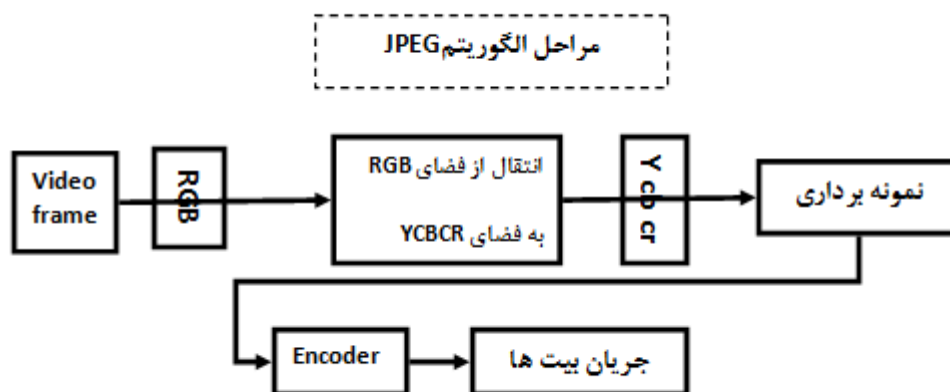
الگوریتم ۲-۴ این شبه کد الگوریتم رمزگشا LZW را نشان می دهد.

این الگوریتم همانند الگوریتم قبلی می باشد ابتدا کد جدید در خروجی چاپ می شود اولین کاراکتر رشته دریافت می شود و به همراه کد قبلی ترکیب می شود و در یک جدول ذخیره می شود.

۲-۵- روش های کلاسیک برای فشرده سازی تصاویر

روش های JPEG و JPEG۲۰۰۰ از استانداردهای بین المللی برای فشرده سازی تصاویر خاکستری و رنگی می باشد. بسیاری از روش های مورد استفاده در فشرده سازی ویدیویی نیز از این روش و

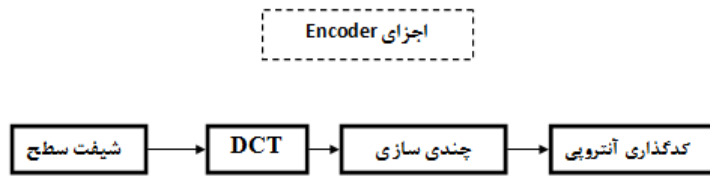
ساختارهای آن به عنوان ساختارهای پایه‌ای استفاده می‌کنند. یکی از روش‌ها کاهش افزونگی در استاندارد JPEG، مبتنی بر DCT است، که در آن از تبدیل DCT به عنوان مبنای استفاده می‌شود. اما در بعضی از استانداردها، از روش‌های پیشگو برای کاهش افزونگی بین پیکسل‌ها استفاده می‌شود.



شکل ۲-۱۲ مراحل الگوریتم JPEG انتقال تصویر به فضای YCbCr جز استاندارد jpeg نمی‌باشد اما اکثر نرم افزارهایی که این استانداردها را پیاده سازی می‌کنند ابتدا این تبدیل را انجام می‌دهند. با انجام این تبدیل، تا حد زیادی حجم تصاویر کاهش می‌یابد. بعد از آن که داده‌ها به این فضا انتقال یافت و نمونه برداری شد، به Encoder ارسال و در آنجا به دنباله‌ای از بیت‌ها تبدیل می‌شوند.

۲-۵-۱- کدگذار (JPEG Encoder)

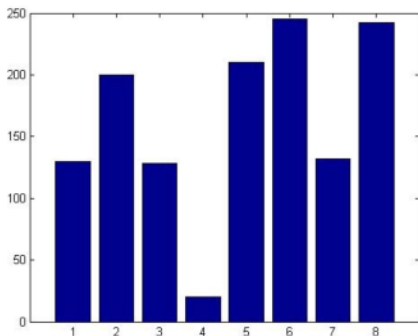
شکل ۲-۱۲ ساختار کدگذار JPEG را نشان می‌دهد. انتقال تصویر به فضای YCbCr جز استاندارد JPEG نمی‌باشد اما اکثر نرم افزارهایی که این استانداردها را پیاده سازی می‌کنند ابتدا این تبدیل را انجام می‌دهند. با انجام این تبدیل، داده‌ها به فضای جدیدی انتقال یافته و نمونه برداری انجام می‌گیرد و به این طریق تا حد زیادی حجم تصاویر کاهش می‌یابد. بعد از این مرحله داده‌ها به Encoder ارسال و در آنجا به دنباله‌ای از بیت‌ها تبدیل می‌شوند. کدگذار JPEG دارای چندین زیر بخش می‌باشد که اجزای آن در شکل ۲-۱۳ به تصویر کشیده شده است.



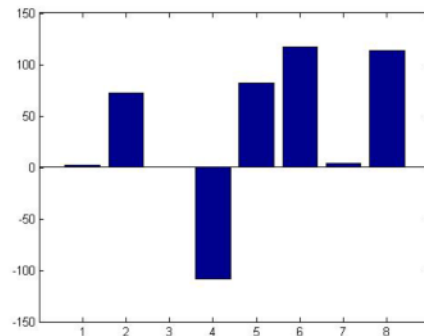
شکل ۲-۱۳ اجزای کدگذار JPEG، ابتدا داده ها شیفت داده می شود، سپس با استفاده از الگوریتم DCT کاهش افزونگی در حوزه فرکانس انجام می گیرد و در نهایت چندی سازی و آنتروپی بر روی آن اعمال می گردد.

الف) شیفت سطح (Level Shift)

اولین گام در کدگذار شیفت دادن داده ها می باشد. در این گام تمامی داده ها از مقداری خاص کم می شود تا اعداد حول صفر متمرکز گردد. اگر به شکل ۲-۱۴ دقت شود در قسمت (الف) یک سیگنال را نشان می دهد. مقادیر این سیگنال در بازه بین ۰ تا ۲۵۵ می باشد بنابراین برای متمرکز کردن این داده ها حول صفر، می توانیم از تمام مقادیر آن ۱۲۸ را کم کنیم و به این طریق داده ها شیفت داده می شود. تصویر (ب) سیگنال شیفت داده را نشان می دهد.



الف) نمونه ای از سیگنال گسسته



ب) سیگنال بعد از شیفت دادن

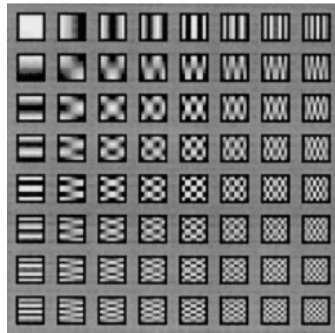
شکل ۲-۱۴ نمونه ای از اعمال شیفت سطح بر روی داده ها

ب) تبدیل گسسته کسینوسی (Transform Coding – Discrete Cosine Transform)

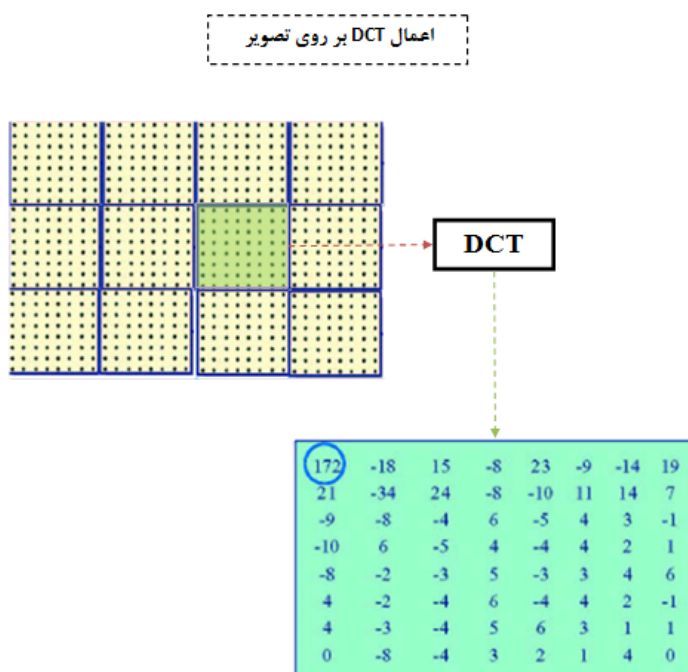
در تصاویر معمولی تغییرات در پیکسل های مجاور به کندی صورت می گیرد. بنابراین در این روش تصویر به بلوک هایی به اندازه $8*8$ تقسیم و بر روی هر کدام از بلوک ها تبدیل DCT انجام می گیرد. بنابراین رابطه تبدیل DCT را می توان به صورت زیر نوشت. تصویر پایه ها DCT در شکل ۱۵-۲ نشان داده شده است. در شکل ۱۶-۲ و شکل ۱۷-۲ اعمال تبدیل گسسته فوریه بر روی تصویر نمایش داده شده است.

$$F(u, v) = \frac{2}{\sqrt{MN}} C(u) C(v) \sum_{x=0}^{N-1} \sum_{y=0}^{M-1} f(x, y) \cos \left[\frac{\pi(2x+1)u}{2M} \right] \cos \left[\frac{\pi(2y+1)v}{2N} \right] \quad (19-2)$$

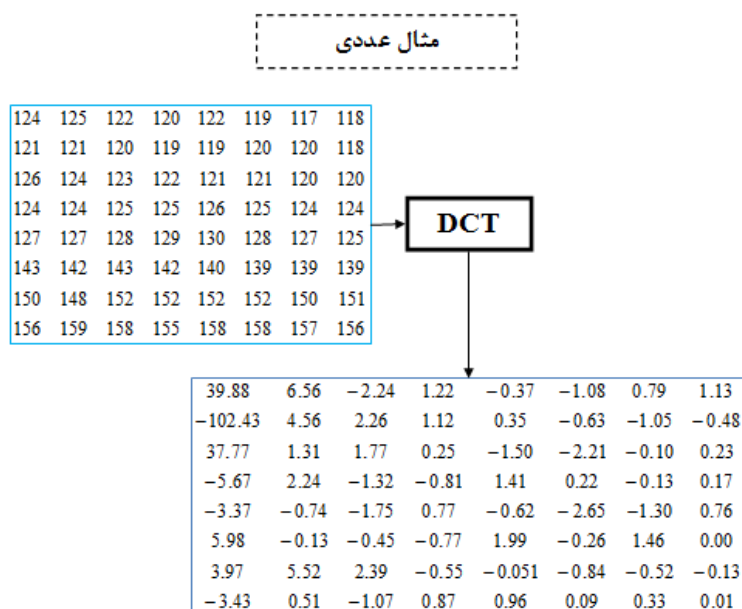
$$\text{where } C(k) = \begin{cases} 1/\sqrt{2} & \text{for } k = 0 \\ 1 & \text{otherwise} \end{cases}$$



شکل ۱۵-۲ تصویر مولفه های تبدیل گسسته کسینوسی



شکل ۲-۱۶ در این تصویر نحوه اعمال تبدیل DCT را بر روی بلوک های تصویر نشان می دهد.



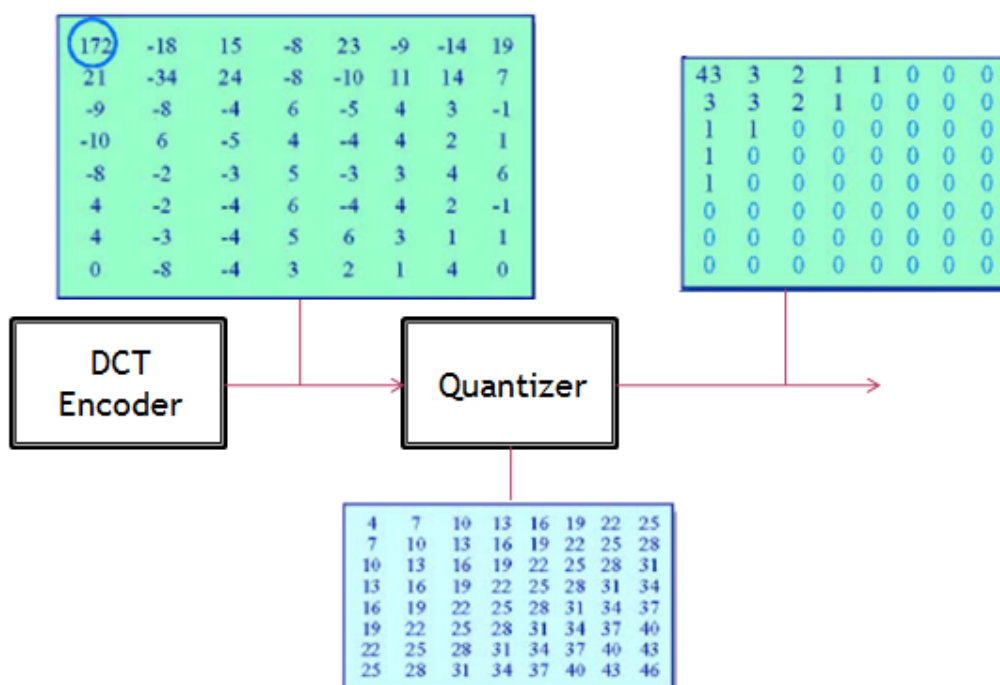
شکل ۲-۱۷ مثال عددی از تبدیل DCT بر روی تصویر

پ) چندی سازی (Quantization)

دومین گام در فشرده سازی بعد از DCT، quantization می باشد. همان طوری که قبلاً اشاره شد چشم انسان نسبت به فرکانس های پایین حساس تر از فرکانس های بالا است. بنابراین گام ها quantization را برای فرکانس های بالا، بزرگ انتخاب می شود. برای فرکانس های پایین لازم است طول گام ها کوچک تر انتخاب شود تا اطلاعات مفید حفظ گردد. در JPEG دو جدول را برای چندی سازی اطلاعات رنگی، تعریف می شود. آنگاه عمل چندی سازی به کمک رابطه (۲۰-۲) انجام می شود.

$$\hat{F}(u,v) = \text{round} \left(\frac{F(u,v)}{Q(u,v)} \right) \quad (20-2)$$

همانطوری که در شکل ۲-۱۸ مشاهده می شود گام های چندی سازی در گوشه سمت چپ بلوک ها کوچک می باشد زیرا در این مکان فرکانس های پایین قرار دارد. گام های چندی سازی در گوشه سمت راست و پایین بزرگ است زیرا فرکانس های بالا در این قسمت قرار دارد.



شکل ۲-۱۸ نحوه چندی سازی درایه ها در این تصویر مثالی عددی را نشان می دهد که چگونه داده ها در استاندارد JPEG کوانتیزه می شوند

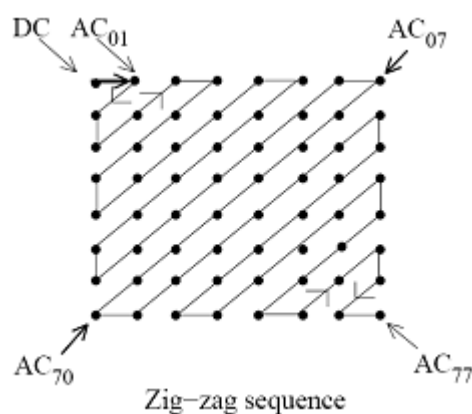
ت) کد گذاری مولفه های DC (Differential Coding of the DC Components)

بعد از چندی سازی باید مولفه (داده واقع در گوشه چپ و بالای هر بلوک) DC را از مولفه های (سایر داده ها به غیر از مولفه DC) جدا کرد زیرا بین مولفه DC بلوک های مجاور هم، وابستگی زیادی وجود دارد. فرض کنید مولفه DC بلوک i ام را، به صورت DC_i و مولفه DC بلوک قبلی به صورت DC_{i-1} نمایش دهیم. آنگاه $Diff_i$ را به صورت (۲-۲۱) بیان می شود.

$$Diff_i = DC_i - DC_{i-1} \quad (2-21)$$

ث) خطی کردن ضرایب AC (Zigzag Scanning of the AC Coefficients)

به هر حال ماتریس ضرایب AC که در مراحل قبلی به وجود می آید دو بعدی می باشد. به همین دلیل لازم است تا به داده یک بعدی تبدیل شود. در این داده یک بعدی، در ابتدای دنباله فرکانس های پایین، و در انتهای دنباله، ضرایب فرکانس های بالاتر قرار دارد. معمولاً می توان ضرایب را همانند شکل ۲-۱۹ مرتب کرد.



شکل ۲-۱۹ نمودار فوق نحوه مرتب کردن ضرایب فرکانس، جهت خطی کردن مولفه ها را نشان می دهد.

ج) کد گذار Run Length Coding of the AC Coefficients

به دلیل این که گام های بزرگ را برای فرکانس های بالا انتخاب می شود بعد از چندی سازی، مقادیر زیادی از ضرایب AC برابر ۰ می گردد. برای این که بتوان از این ویژگی استفاده شود. JPEG کد گذاری Run Length Coding را پیشنهاد کرد که به صورت $(runlength, value)$ بیان می شود و در آن value مقادیر غیر صفر و runlength طول دنباله صفر را نشان می دهد. به عنوان مثال اگر دنباله ضرایب به صورت زیر باشد.

$[54, 45, 0, 0, 0, 0, 23, 0, -30, -16, 0, 0, 1, 0, 0, 0, 0, 0, 0, \dots, 0]$

می توانیم آن را به صورت زیر که حجم کمتری نیاز دارد، نشان داد.

$[(0, 54), (0, 45), (4, 23), (1, -30), (0, -16), (2, 1), EOB]$

EOB، آخر بلوک را نشان می دهد.

چ (کد گذاری آنتروپی (Entropy Coding)

نهایتاً مولفه های DC و AC به encoder ارسال می شود. به دلیل این کد هافمن محاسبات زیادی نیاز دارد، در استاندارد JPEG جداول ثابتی ارائه شده است که می توان از آن، برای کد کردن استفاده کرد. جدول ۲-۶ قسمتی از جدول کد هافمن استفاده شده در استاندارد JPEG، را نشان می دهد.

که تفاضل ضرایب DC به صورت $(size, Amplitude)$ کد خواهد شد که در آن size تعداد بیت های مورد نیاز برای کد کردن را نشان می دهد و Amplitude کد واقعی را نشان می دهد ضرایب AC نیز به صورت زیر نمایش داده می شود.

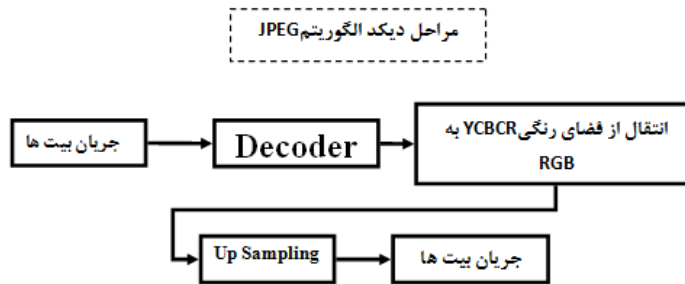
$(runlength, size, amplitude)$

جدول ۲-۶ قسمتی از جدول کد هافمن استفاده شده در استاندارد JPEG

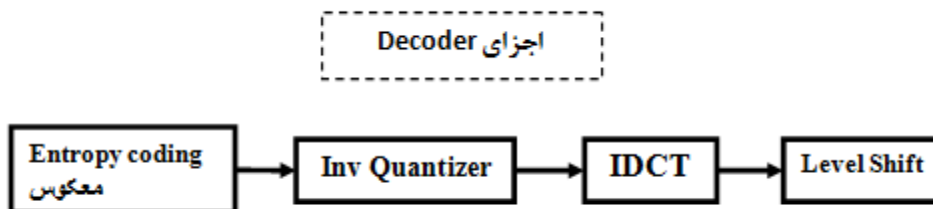
گروه	مقدار	مقادیر بیتی
۱	-۱,۱	۰,۱
۲	-۳,-۲,۲,۳	۰۰,۰۱,۱۰,۱۱
۳	-۷,-۶,-۵,-۴,۴,۵,۶,۷	۰۰۰,۰۰۱,۰۱۰,۰۱۱,۱۰۰,۱۱۰,۱۱۱
۴
۵

۲-۵-۲- کدگشای JPEG Decoder

جریان بیت ورودی باید به ضرایب اولیه برگردد تا بتوان تصویر اولیه را بازیابی کرد. نمودار شکل ۲-۲۰ سیستم مراحل دیکد را نشان می دهد. این مولفه معکوس عملیات رمزگذار را انجام می دهد به همین دلیل از تشریح بیشتر آن خودداری می کنیم. اجزای کد گشا در تصویر ۲-۲۱ آمده است.



شکل ۲-۲۰ مراحل کدگشایی در الگوریتم JPEG



شکل ۲-۲۱ اجزای کدگشا در استاندارد JPEG

۶-۲- استاندارد JPEG۲۰۰۰

با توجه به اثر بلوکی که در روش استاندارد JPEG، روشی به کمک موجک ها ارائه شد که در ادامه با آن آشنا می شویم این روش به JPEG۲۰۰۰ معروف است و دارای گام هایی است که در ادامه معرفی می شود.

۲-۶-۱- تبدیل رنگی معکوس پذیر

از آنجایی که فضای رنگ RGB برای فشرده سازی تصاویر مناسب نمی باشد برای برطرف کردن این مشکل از فضای رنگ YCbCr استفاده می شود. اما از آنجایی که این تبدیل ها ممیز اعشاری است بازیابی آن برابر مقدار اولیه نمی شود برای برطرف کردن این مشکل، در این استاندارد از تبدیل زیر استفاده می شود:

I) forward RCT Transform:

$$Yr = \frac{R + 2G + B}{4} \quad (22-2)$$

$$Ur = B - G \quad (23-2)$$

$$Vr = R - G \quad (24-2)$$

II) Backward RCT Transform:

$$G = Yr - \left(\frac{Ur + Vr}{4} \right) \quad (25-2)$$

$$R = Ur + G \quad (26-2)$$

$$B = Vr + G \quad (27-2)$$

۲-۶-۲ - چندی سازی در JPEG۲۰۰۰

بعد از تبدیل تصویر به فضای موجک، باید آن را چندی سازی نمود تا کمیت آن را کاهش داد. از ویژگی های مهم این استاندارد این است که هم فشرده سازی بدون از دست دادن و هم با از دست دادن را پشتیبانی می کند. این استاندارد روش نوینی را برای چندی سازی ارائه کرده است که دارای رابطه ای به صورت (۲۸-۲) می باشد.

$$q_b(u, v) = \text{sign}[a_u(u, v)] \text{floor} \left[\frac{|a_b(u, v)|}{\Delta_b} \right] \quad (28-2)$$

R_b نسبت معکوس با اهمیت زیر باند ها دارد. هر چه اهمیت زیر باند بیشتر باشد آن را بزرگتر انتخاب و هر چه کوچکتر باشد اهمیت آن کمتر می باشد. از آن برای محاسبه گام چندی سازی استفاده می شود رابطه گام چندی سازی آن به صورت (۲۹-۲) می باشد.

$$\Delta_b = 2^{R_b - \epsilon_b} \left(1 + \frac{\mu_b}{2^{11}} \right) \quad (29-2)$$

با توجه به مقادیری که برای پارامتر ها استفاده می شود، چندی سازی بدون از دست دادن و با از دست دادن اطلاعات را بدست آوریم. اگر $R_b = \epsilon_b$, $\mu_b = 0$, آنگاه $\Delta_b = 1$ و چندی سازی بدون از دست دادن اطلاعات می شود. پارامتر های فوق باید محدودیت ها $0 \leq \mu_b \leq 2^{11}$, $0 \leq \epsilon_b \leq 32$ را برآورده کند.

۲-۷ - جمع بندی (از نتایج پایان نامه)

جدول ۲-۷ رابطه میزان فشرده سازی با ضرایب چندی سازی را با اعمال بر روی یک تصویر خاص نشان می دهد. همانطور که در تصاویر شکل ۲-۲۲ مشاهده می کنید تصویر با ضریب ۱ دقیقاً مانند تصویر اصلی است اما درصد درایه های غیر صفر آن حدود ۱۵٪ است (از درایه های صفر صرف نظر می شود) پس حجم تصویر فشرده شده برابر ۰٫۱۵ درصد تصویر اولیه می باشد. بعد از کوانتیزه، لازم است تا داده های هر بلوک را به صورت خطی در بیاوریم. برای خطی کردن می توان به صورت

زیگزاگ استفاده کرد و در نهایت مقادیر خطی الگوریتم Huffman یا الگوریتم دیگر ارسال می شود به این طریق فشرده سازی به پایان می رسد.

حال می خواهیم میزان اثر کوانتیزه را بر فشرده سازی بدست آوریم. این فشرده سازی، فشرده سازی JPEG می باشد. بعد از انتقال تصویر به حوزه فرکانس، چندی سازی می شود.

جدول ۲-۷ نسبت فشرده سازی با ضرایب کوانتیزه مختلف (این نسبت از حجم تصویر فشرده شده بر حجم تصویر اصلی بدست می آید)

نسبت فشرده سازی	ضریب چندی سازی
۰,۰۷۹۳۷۶	۲
۰,۰۵۱۶۳۶	۴
۰,۰۳۹۳۳۷	۶
۰,۰۳۱۴۳۳	۸
۰,۰۲۷۱۶۱	۱۰



ب) تصویر اصلی



الف) تصویر بازیابی شده بعد از کوانتیزه با ضریب ۱



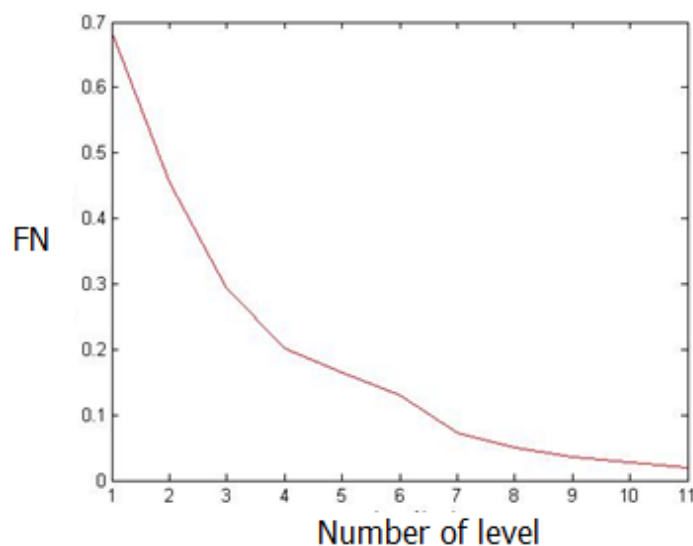
ت) تصویر بازیابی شده بعد از کوانتیزه با ضریب ۲



پ) تصویر بازیابی شده بعد از کوانتیزه با ضریب ۵

شکل ۲-۲۲ برای بررسی اثر DCT و چندی سازی، تصویر را ابتدا با اعمال DCT به حوزه فرکانس ارسال می کنیم بعد با ضرایب های مختلف کوانتیزه می شود و دو باره تصاویر را از حوزه فرکانس به حالت اولیه بر می گردانیم. منظور از ضریب عددی ثابت است که در ماتریس Q ضرب می شود.

همانطور که در استاندارد JPEG مشاهده شد این استاندارد از اثر بلوکی رنج می برد. علت این پدیده اعمال فیلتر بر روی مستطیل ها غیر همپوشان می باشد. بعد از انتقال تصویر به حوزه فرکانس، فرکانس های بالا را که برای چشم انسان از اهمیت کمتری برخوردار است، حذف می کنیم. اما همین عمل موجب بروز مشکلاتی در تصاویر بازیابی شده می شود. برای مقابله با این موضوع ابتدا در استاندارد JPEG۲۰۰۰، بر روی تصویر، فیلتر های پایین گذر و بالاگذر اعمال می شود. به این وسیله مولفه های با فرکانس بالا و پایین استخراج می شود. از آنجایی که فرکانس های پایین را بیشتر حفظ می کند و فرکانس های بالا بیشتر تضعیف می شود. زمانی که تعداد سطوح بیشتر باشد درصد فشرده سازی بیشتر می شود اما کیفیت تصویر کاهش می یابد. همانطور که در تصاویر مندرج در شکل ۲-۲۴ نشان داده شده است. اگر سطوح فشرده سازی بیشتر از ۱۰ سطح شود، کیفیت تصویر کاهش می یابد. در نمودار شکل ۲-۲۳ رابطه بین سطوح فشرده سازی و میزان فشرده سازی در استاندارد JPEG۲۰۰۰ نشان می دهد. با افزایش سطوح درصد فشرده سازی بالا می رود.



شکل ۲-۲۳ رابطه بین سطوح فشرده سازی و میزان فشرده سازی در استاندارد JPEG۲۰۰۰ محور γ معکوس درصد فشرده سازی را نشان می دهد.



ب) سطح فشرده سازی = ۱۰



الف) سطح فشرده سازی = ۹



ت) سطح فشرده سازی = ۸



پ) سطح فشرده سازی = ۷



ج) سطح فشرده سازی = ۶



ث) سطح فشرده سازی = ۵



ح) سطح فشرده سازی = ۴



چ) سطح فشرده سازی = ۳



د) سطح فشرده سازی = ۲



خ) سطح فشرده سازی = ۱

شکل ۲-۲۴ تصاویر بازیابی شده از استاندارد JPEG۲۰۰۰، تعداد سطح تعداد مراحل اعمال فیلتر را نشان می دهد همانطور که مشاهده می شود اگر تعداد سطح فشرده سازی بیشتر از ۸ سطح باشد کیفیت تصویر بازیابی کاهش می یابد. (ادامه تصویر)

فصل سوم

اجزا و استانداردهای سیستم‌های فشرده‌سازی ویدیو

۳-۱- مقدمه

رسانه های ویدیویی معمولاً حجم زیادی از داده ها را تولید می کنند. مثلاً در تلوزیون های HDTV، حجم داده‌ی تولید شده در یک ثانیه از مرز یک گیگا بیت می‌گذرد. همچنین اگر ویدیویی با کیفیت تلوزیون های آنالوگ را در نظر بگیریم، آنگاه نرخ تولید ویدئو برابر ۱۶ مگابیت بر ثانیه خواهد بود که برای ۱ ساعت (۳۶۰۰*۱۶=۵۷۶۰۰) مگابیت خواهد شد. این میزان داده حتی برای ویدئوهایی با کیفیت معادل VHS یا VCD که در آنها، حجم داده‌ی تولید شده تقریباً یک چهارم می شود، بسیار زیاد می باشد.

حجم عظیم اطلاعات ویدئویی، مشکلاتی از قبیل فضای لازم برای ذخیره کردن اطلاعات و پهنای باند کافی برای انتقال آن را به همراه دارد. این مشکل گریبان گیر شبکه های انتقال اطلاعات و سیستم های offline نگهداری اطلاعات می باشد. از طرفی حافظه های جانبی موجود در بازار، توان انتقال چند گیگابیت بر ثانیه را ندارند.

حتی اگر حافظه های جانبی این توان را داشته باشند تمام پهنای باند گذرگاه داده درگیر این انتقال خواهد شد. به همین دلیل نیازمند روش هایی هستیم که با کمترین اطلاعات از دست رفته، بیشترین حجم اطلاعات را منتقل و یا ذخیره کنیم. فشرده سازی ویدئو یکی از مسائل اساسی در ذخیره سازی و انتقال اطلاعات می باشد. یکی از دلایل فراگیر شدن ویدئو دیجیتال در بین اقشار جامعه، استفاده از تکنیک های فشرده سازی ویدئو می باشد.

افزونگی اطلاعات یکی از عواملی است که می توان با حذف آن، حجم قابل توجهی از ویدئو را کاهش داد. افزونگی اطلاعات به دو دسته‌ی افزونگی بین فریم ها و افزونگی درون فریم ها تقسیم می شوند که در ادامه به معرفی آن خواهیم پرداخت.

حجم تصویر(الف) در شکل ۱-۳ نصف تصویر (ب) می باشد، در حالی که این دو تصویر از نظر ما دارای کیفیت برابر می باشند. دلیل این مسئله ریشه در ویژگی های روانشناختی چشم انسان دارد. با اینکه مقادیر نشان دهنده هر کدام از تصاویر با هم اختلاف دارند، چشم انسان توانایی درک این تفاوت را ندارد. ویژگی های چشم انسان (ناتوانی در تشخیص فرکانس های نزدیک به هم یا غیر حساس بودن به مولفه های رنگی در تصویر و...) به طراحان سیستم های فشرده سازی کمک می کند تا حجم اطلاعات را کاهش دهند. به این گونه افزودگی، افزودگی درون فریم می گویند. مانند فرکانس های تکراری، فرکانس نزدیک به هم و یا نزدیک بودن اطلاعات پیکسل های مجاور، که با استفاده از آن می توان حجم یک فریم را کاهش داد .



ب) تصویر بازیابی شده بعد از نمونه برداری کردن

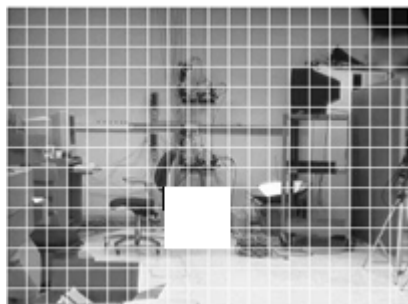


الف) تصویر اصلی

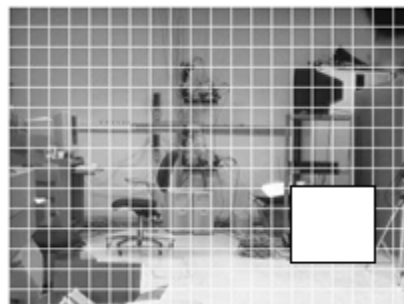
شکل ۱-۳ تصویر ب را از حوزه RGB به حوزه YCBCR انتقال می دهیم و در این فضا از مولفه ها CB و CR نمونه برداری می کنیم حجم داده های جدید یک دوم حجم تصویر ب می شود. برای بدست آوردن تصویر الف دوباره بر روی داده های نمونه برداری شده تبدیل معکوس اعمال می شود تا داده ها را از فضای YCBCR به فضای RGB برگردد اما همانطور که مشاهده می شود کیفیت دو تصویر مشابه می باشند به این نوع افزودگی درون فریم می گویند.

نوع دیگر افزودگی، افزودگی بین فریم می باشد. برای درک این نوع افزودگی به تصاویر ۲-۳ دقت کنید. در این تصویر دو فریم متوالی نشان داده شده اند. همانطور که مشاهده می شود این دو فریم دارای اطلاعات مشترک فراوانی می باشند. معمولا در تصاویر ویدیویی فریم های متوالی دارای پس زمینه

مشترک می باشند و اشیا در درون فریم ها جا به جا می شوند. به این افزونگی (اطلاعات مشابه در فریم های متوالی) افزونگی بین فریم می گویند.



ب) یک فریم در لحظه t



الف) تصویر فریم در لحظه t+1

شکل ۲-۳ تصویر الف و ب دو فریم ویدئویی را در زمان های مختلف نشان می دهد. همانطور که مشاهده می شود قسمت عمده تصویر (پس زمینه) در دو فریم یکسان می باشد و صرفاً تغییرات محدود به ناحیه کوچکی از تصویر است که اشیا در آن حضور دارد.

برای اولین بار در سال ۱۹۹۰ گروهی تحت عنوان CCITT^۱ تلاش کردند تا الگوریتم های فشرده سازی ویدئو را استاندارد سازی کند. هدف نهایی این کار استفاده از این استاندارد برای تلفن و تلفن تصویری بود. این استاندارد با نام H.۳۲۰ معرفی شد که قسمت فشرده سازی ویدئو آن به نام H.۲۶۱ می باشد. این گروه به تلاش های خود جهت استاندارد سازی سایر کاربردها ادامه داد و استاندارد H.۳۲۳ را جهت انتقال اطلاعات بر روی شبکه اینترنت ارائه کرد. از استانداردهای دیگر این گروه می توان به استاندارد H.۳۲۴ اشاره کرد. این استاندارد خطوط کم سرعتی مانند خطوط تلفن (با استفاده از مودم) را در برنامه کاری خود قرار داد. در استانداردهای جدید سعی شده با نرخ بیت کمتر، کیفیت مناسب تری ارائه شود. به عنوان مثال H.۲۶۳ نسخه جدید استاندارد H.۲۶۱ برای فشرده سازی ویدئو می باشد که برای ارائه ویدیویی با کیفیت QCIF به ۲۴ kbps نیاز دارد. این در حالی است که برای رسیدن به کیفیت معادل همین ویدئو در استاندارد H.۲۶۱ به ۶۴ kbps نیاز می باشد.

^۱ Consultative Committee for International Telephone and Telegraph

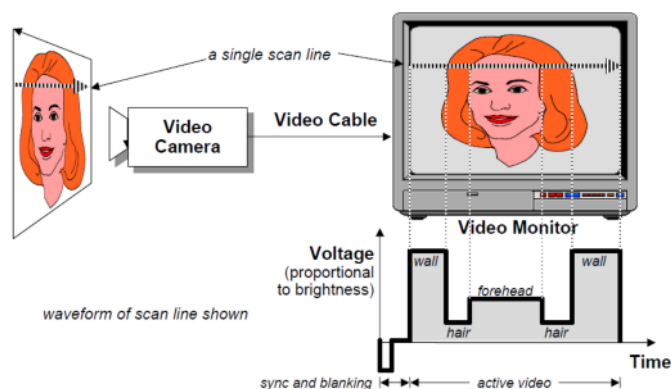
همزمان با تلاش این گروه برای فشرده سازی ویدئو گروه دیگری در ISO به نام MPEG^۲ شروع به استاندارد سازی الگوریتم ها و روش های ذخیره سازی و انتقال داده های ویدیویی نمودند. استاندارد های این گروه معمولاً با پیشوند MPEG شروع می شوند. اولین استاندارد این گروه استاندارد MPEG۱ می باشد که برای فشرده سازی ویدئو به منظور استفاده در CD ها یا VCD مناسب می باشد [۱،۳۲]. این استاندارد برای نمایش ویدئو به نرخ بیت ۱،۵Mbps نیاز دارد. استاندارد دیگری که پس از آن معرفی شد استاندارد MPEG۲ می باشد. هدف این استاندارد فشرده سازی ویدئوهای با کیفیت بالاتر مانند CCIR۶۰۱ می باشد که برای کیفیت فیلم های DVD و کاربرد های HDTV مناسب می باشد. یکی دیگر از استانداردها، استاندارد MPEG۴ می باشد هدف در این استاندارد کد کردن هر کدام از اشیا موجود در ویدئو به صورت مجزا می باشد. در ادامه به معرفی اجزای بعضی از سیستم های فشرده سازی یاد شده در بالا می پردازیم.

۳-۱-۱- نحوه تولید ویدئو

همانطور که در شکل ۳-۳ مشاهده می شود، برای نمونه برداری از دنیای واقعی دوربین در هر لحظه یک خط را پیمایش می کند. با پیمایش هر خط، سیگنال مربوط به آن خط، تولید می شود. مقدار این ولتاژ متناسب با سطح خاکستری بالا و پایین می رود. هر چه سطح خاکستری روشنتر باشد، ولتاژ بالاتر و هر چه تیره تر باشد، میزان ولتاژ کمتری تولید می گردد.

^۲ Motion Picture Expert Group

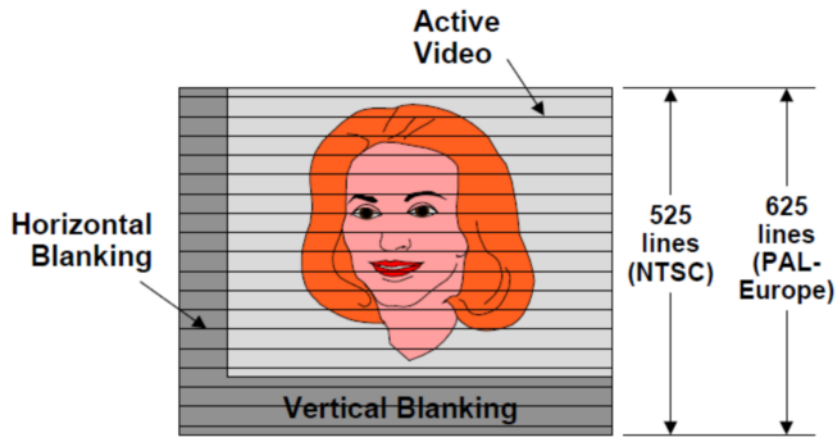
^۳ تصاویر ۳-۳ تا ۳-۶ از مرجع [۳۹] اقتباس شده است.



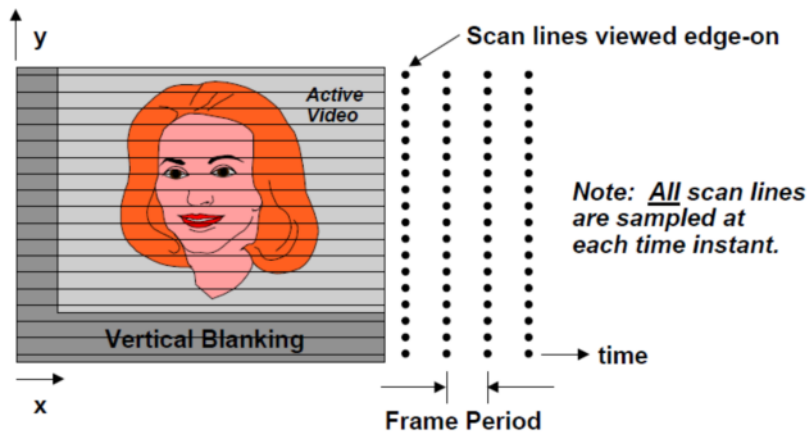
شکل ۳-۳ نحوه تولید تصاویر ویدیویی برای نمونه برداری از دنیای واقعی دوربین در هر لحظه یک خط را پیمایش می کند زمانی که یک خط پیمایش می شود یک سیگنال که آن خط را نمایش می دهد تولید می شود مقدار این ولتاژ متناسب با سطح خاکستری بالا و پایین می رود هر چه سطح خاکستری روشنتر باشد ولتاژ بالاتر می باشد و هر چه تیره تر باشد میزان ولتاژ کمتر می باشد.

برای ایجاد هماهنگی بین سیستم های مختلف ویدیویی دو نوع استاندارد NTSC و PAL وضع شده است. در استاندارد NTSC هر صحنه شامل ۵۲۵ خط و در PAL ۶۲۵ خط مورد استفاده قرار می گیرد که در تصویر ۳-۴ این تفاوت ها نشان داده شده است.

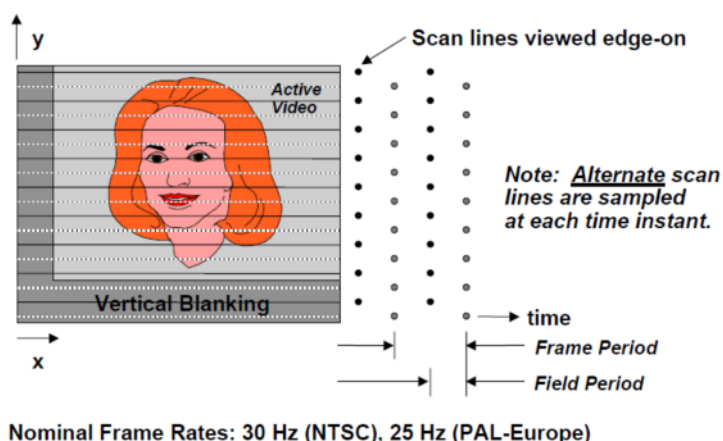
نمونه برداری به دو دسته progressive و interlaced تقسیم می شود. در روش اول تمام خطوط لازم برای نمایش یک فریم در یک دوره زمانی نمونه برداری می شود که در تصویر ۳-۵ نشان داده شده است. عیب این روش را می توان در تعداد زیاد خطوط لازم برای نمونه برداری فریم دانست. برای مقابله با این مشکل روش interlaced ابداع شد. در این روش به جای نمونه برداری از تمام خطوط، همانطور که در تصویر ۳-۶ مشاهده می شود، فریم به صورت یک خط در میان نمونه برداری و ارسال می گردند. با توجه به ویژگی های چشم، انسان متوجه خطوط حذف شده در نمونه برداری نمی شود. برتری این روش نسبت به روش قبل کاهش پنجاه درصدی زمان نمونه برداری، می باشد. در مقابل این روش موجب به وجود آمدن لرزش در صحنه، کاهش جزئیات در مرز خطوط و نیاز به محاسبات پیچیده برای تغییر اندازه در تصویر می باشد.



شکل ۳-۴ تعداد خطوط لازم برای به تصویر کشیدن صحنه در استاندارد های PAL و NTSC برای ایجاد هماهنگی بین سیستم های پخش و نمایش تلویزیونی دو نوع استاندارد NTSC و PAL (برای تعیین تعداد خطوط مورد نیاز برای نمونه برداری) وضع شده است. در استاندارد NTSC هر صحنه شامل ۵۲۵ خط و در PAL ۶۲۵ خط مورد استفاده قرار می گیرد.



شکل ۳-۵ نمونه برداری به صورت progressive. در این تصویر هر خط ستونی نمایش یک فریم را در یک لحظه نشان می دهد هر نقطه نمایش دهنده یک خط نمونه برداری می باشد.



شکل ۳-۶ نمونه برداری به صورت interlaced در این روش برای به تصویر کشیدن صحنه خطوط به صورت یکی در میان نمونه برداری می شود.

۳-۱-۲- معیار ها ارزیابی روش های فشرده سازی ویدئو

الگوریتم ها و استاندارد های فراوانی برای فشرده سازی ویدئو وجود دارد. به همین دلیل لازم است تا معیاری برای ارزیابی این استانداردها ارائه شود. برای این منظور از رابطه Normalized Bit Rate استفاده می شود که به صورت معادله (۳-۱) تعریف شده است.

$$\text{Normalized Bit Rate} = \frac{C * R}{H * V * F} \quad (3-1)$$

در رابطه فوق C، فاکتور رنگ، R، برای Bit Rate و F، برابر تعداد فریم در واحد زمان است و واحد آن فریم بر ثانیه می باشد. دو پارامتر H و V نیز طول و عرض فریم را نشان می دهد. هر چه اندازه این معیار کوچکتر باشد روش فشرده سازی کارآمدتر می باشد [۳۳].

۳-۱-۳- نمونه برداری رنگی

چشم انسان دارای خاصیت هایی می باشد که با بهره گرفتن از این خاصیت ها می توان بر روی ویدئو فشرده سازی انجام داد. یکی از این ویژگی ها کم بودن حساسیت چشم انسان به اطلاعات رنگی نسبت به اطلاعات درخشندگی در تصویر می باشد. بنابراین اگر بتوانیم تصویر را به فضای رنگی انتقال دهیم که در آن درخشندگی از اطلاعات رنگی جدا شود، می توانیم بدون تغییر قابل توجه در کیفیت

تصویر، از اطلاعات رنگی تصویر نمونه برداری کنیم. فضای رنگی مناسب برای این منظور، فضای رنگ YCbCr می باشد. تبدیل که برای انتقال تصویر از فضای RGB به فضای رنگی YCbCr از رابطه زیر استفاده می شود:

$$\begin{pmatrix} Y \\ Cb \\ Cr \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.334 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{pmatrix} * \begin{pmatrix} R \\ G \\ B \end{pmatrix} + \begin{pmatrix} 128 \\ 128 \\ 128 \end{pmatrix} \quad (2-3)$$

بعد از این که داده ها به فضای YCbCr انتقال داده شد لازم است تا از داده های در فضای جدید نمونه برداری شود.

۳-۲- گام های فشرده سازی در استاندارد H.۲۶۱

این استاندارد که می توان سرآغاز استاندارد های سری H دانست در سال های ۱۹۸۸ تا ۱۹۹۰ توسط گروه CCITT معرفی و توسعه داده شد. هدف طراحی این استاندارد، استفاده از آن برای کنفرانس های ویدیویی و تلفن های ویدیویی بر روی خطوط ISDN می باشد. نرخ بیتی که توسط این استاندارد پشتیبانی می شود $64KB/Sec$ می باشد، p می تواند متناسب با کاربرد مقداری بین ۱ تا ۳۰ داشته باشد.

این استاندارد از فریم های $(352 * 288)$ CIF و $(176 * 144)$ QCIF پشتیبانی می کند. در این استاندارد دو نوع فریم P و I معرفی شده است که فریم های نوع I، فریم مرجع هستند. فریم های نوع P با استفاده از فریم های I پیش بینی می شود. در صورت از بین رفتن فریم های مرجع دیگر نمی توان فریم های P را بازیابی کرد.

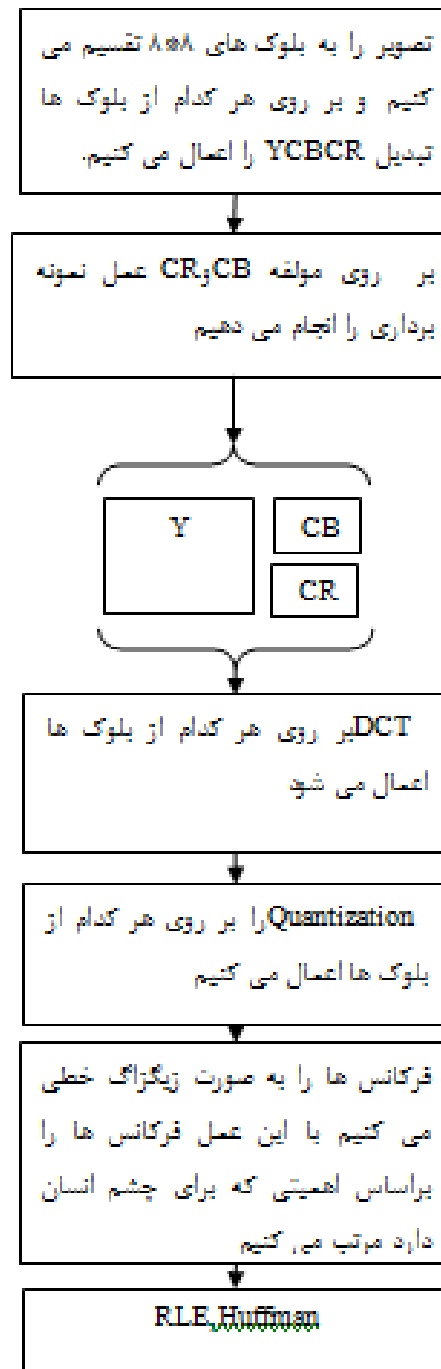
۳-۲-۱- کد کردن فریم نوع I

این فریم در واقع فریم مرجع می باشد و برای بازیابی اطلاعات سایر فریم ها لازم می باشد. در صورتی که اطلاعات این فریم از بین برود، نمی توان فریم های دیگر را (که با استفاده از این فریم

ساخته می شوند) بازیابی کرد. این فریم به صورت مستقل کد می شود و برای بازیابی به اطلاعات خود نیاز دارد. این فریم به صورت بلوکی فشرده سازی می گردد. ابتدا فریم به فضای رنگی YCbCr منتقل و در فضای جدید از داده‌ها نمونه برداری می شود. در گام دوم تصویر به بلوک های 8×8 تقسیم، سپس بر روی هر کدام از بلوک ها تبدیل DCT اعمال می گردد. بعد از تبدیل DCT تصویر از حوزه مکانی به حوزه فرکانس وارد می شود.

بعد از این مرحله لازم است تا خروجی را چندی سازی^۴ کنیم. همانطور که قبلا اشاره شد که چشم انسان به فرکانس های پایین حساس تر از فرکانس های بالا می باشد. بدین جهت لازم است تا چندی سازی به صورتی انجام گیرد که فرکانس های بالا را کمتر از فرکانس های پایین نگه دارد. پس از کوانتیزه کردن اطلاعات، آنها را به صورت خطی مرتب نموده و الگوریتم length run بر روی آن اعمال می شود. در مرحله دیگر برای فشرده سازی بیشتر خروجی این مرحله به الگوریتم های فشرده سازی آنتروپی ارسال می شوند. شکل ۳-۷ مراحل طی شده برای فشرده شدن فریم نوع I را نشان می دهد.

^۴ quantization



شکل ۳-۷ مراحل لازم برای فشرده سازی فریم نوع ۱

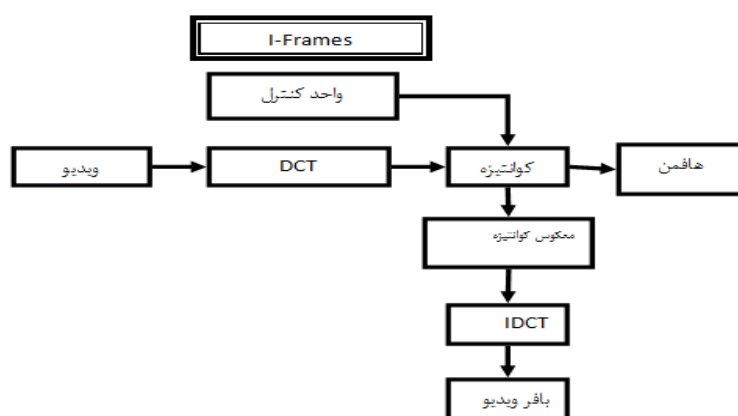
در مرحله دیگر، اطلاعات از حالت دو بعدی به صورت خطی تبدیل می شود. برای اینکار لازم است تا اطلاعات بلوک ها به صورتی خطی شوند تا اطلاعات ارزشمند در کنار هم قرار گیرند. با این کار قسمت های کم اهمیت و مهمتر از هم جدا نموده و قسمت های کم اهمیت تر حذف می شوند.

۳-۲-۲- کد کردن فریم P

این نوع فریم باید بر اساس فریم قبلی خود کد شود. فریم قبلی می تواند یک فریم از نوع I یا از نوع P باشد. وجود همبستگی بین فریم جاری و فریم قبلی مبنای کار در این مرحله می باشد. با استفاده از بردار حرکت و فریم قبلی می توان فریم جاری را با تقریب مناسبی بدست آورد.

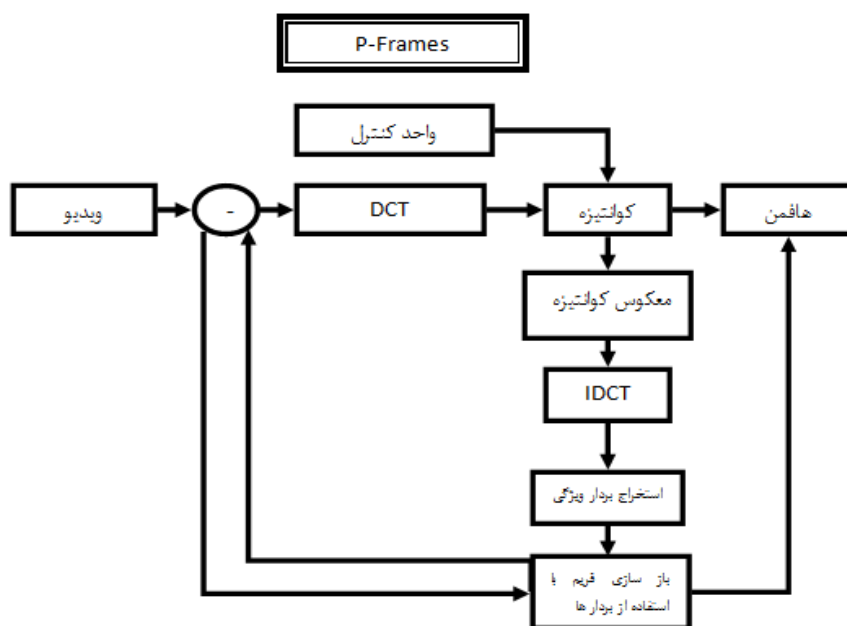
مهم ترین قسمت سیستم های فشرده سازی، رمز گذار می باشد. در دیاگرام شکل ۳-۸ اجزای رمز گذار برای فریم نوع I نشان داده شده است. در این الگوریتم فریم به حوزه فرکانس منتقل شده و در حوزه فرکانس کوانتیزه می شود تا حجم آن کاهش یابد. واحد کنترل برای مدیریت پهنای باند می-باشد. در صورتی که حجم اطلاعات ذخیره شده در بافر خروجی از حد آستانه تجاوز کند، واحد کنترل با تغییر پارامتر واحد کوانتیزه سعی می کند تا جریان بیت ها را کنترل کند.

بعد از کوانتیزه شدن، فریم به الگوریتم هافمن فرستاده می شود تا کدگذاری مناسبی برای آن انتخاب شود. فریم فشرده سازی شده دوباره عمل معکوس کوانتیزه و معکوس تبدیل کسینوسی انجام می شود. این فریم حفظ می شود تا با استفاده از آن فریم های نوع P آتی را تقریب بزنیم.



شکل ۳-۸ Encoder برای فریم نوع I، در این الگوریتم فریم به حوزه فرکانس منتقل می شود و در حوزه فرکانس کوانتیزه می شود تا حجم آن کاهش یابد. بعد از کوانتیزه شدن فریم برای انتقال به الگوریتم هافمن فرستاده می شود. تا کد گذاری مناسبی برای آن انتخاب شود. به این وسیله حجم فشرده سازی بالا می رود. روی فریم فشرده سازی شده عمل معکوس کوانتیزه و معکوس تبدیل کسینوسی انجام می شود. این فریم حفظ می شود تا با استفاده از آن فریم های نوع P را تقریب بزنیم.

در دیاگرام شکل ۳-۹ اجزای Encoder برای فریم نوع P نشان داده شده است.



شکل ۳-۹ Encoder برای فریم نوع P، بعد از کوانتیزه شدن فریم برای انتقال به الگوریتم هافمن فرستاده می شود تا کد گذاری مناسبی برای آن انتخاب شود به این وسیله حجم فشرده سازی بالا می رود. روی فریم فشرده سازی شده عمل معکوس کوانتیزه و معکوس تبدیل کسینوسی اعمال می شود این فریم در حافظه ذخیره می گردد.

۳-۲-۳- ساختار داده ها در استاندارد H.۲۶۱

از آنجایی که فریم مرجع دارای اهمیت می باشد، لازم است تا هر چند وقت یکبار ارسال شود. برای پیدا کردن بلوک مناسب به جای فریم اصلی از فریم decode شده استفاده می شود. در صورتی که بافر خروجی پر باشد میزان چندی سازی بیشتری می شود تا حجم داده های تولیدی کاهش یابد. برای ارسال و ذخیره کردن ویدئو علاوه بر اطلاعات فریمها، باید اطلاعات دیگری نیز ذخیره شود تا با استفاده از آن بتوان ویدئو فشرده شده بازیابی شود.

هر بلوک دارای چندین فیلد می باشد. اولین فیلد، فیلد آدرس است که برای مشخص کردن بلوک ها استفاده می شود. امکان دارد برای یک بلوک هیچ بلوک مناسبی پیدا نشود؛ پس لازم است فیلدی

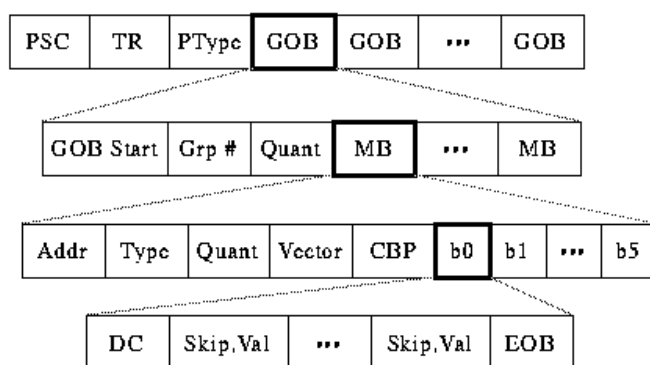
برای این منظور اختصاص داده شود. فیلد دیگر میزان چندی سازی بلوک داده را نشان می دهد. برای ذخیره کردن بردارهای حرکت نیز به یک فیلد داده و فیلد دیگر برای ذخیره کد چندی سازی نیاز است. در نهایت از آنجایی که همه بلوک ها میزان انطباق شان با بلوک اولیه در یک سطح نمی باشد (بعضی از بلوک ها بهتر منطبق می شوند بعضی از بلوک ها انطباق شان مناسب نمی باشد) لازم است تا فریمی را به این منظور اختصاص دهیم. در تصویر زیر فریمی که برای نمایش ماکرو بلاک ها مورد استفاده قرار می گیرد نمایش داده شده است .

Addr	Type	Quant	Vector	CBP	b0	b1	...	b5
------	------	-------	--------	-----	----	----	-----	----

شکل ۳-۱۰ ساختار لازم برای ذخیره ماکرو بلاک

فیلد Addr بعضی از ماکرو بلوک ها دقیقا با آنچه در فریم مرجع واقع شده است منطبق می باشند؛ بنابراین لازم است تا به جای آنها آدرسشان ارسال شود. Type و Quant به ترتیب نوع فریم و میزان چندی سازی را نشان می دهد. فیلد Vector همانطور که از نام آن انتظار می رود برای ذخیره کردن برداری های حرکت می باشد. درون بلوک ها بعضی از بلوک ها فریم جاری با هیچ کدام از بلوک های فریم مرجع به خوبی سازگار نمی باشند، بنابراین لازم است بیتی فرستاده شود که این نوع بلوک ها را نشان دهد. فیلد نهایی (b) مقدار ماکرو بلوک ها را ذخیره می کند.

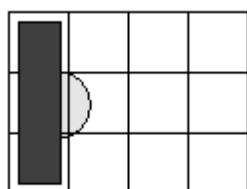
بعد از ساختار دادن به بلاک لازم است تا برای انتقال آن بر روی شبکه، این فریم ها را در درون ساختار دیگر قرار داد. تصویر ۳-۱۱ این ساختار را نشان می دهد. PSC برای تعیین مرز بین فریم ها استفاده می شود. TR برای ایجاد همزمانی بین صدا و تصویر مورد بهره برداری قرار می گیرد. برای تعیین نوع فریم از این فیلد استفاده می شود. هر تصویر به بلوک ها 3×11 تقسیم می شود که به آنها ماکرو بلاک می گویند. از آنجایی که گاهی لازم است تا گروهی از بلوک ها را رد کنیم به هر کدام از گروه ها، بلوکی اختصاص می دهیم تا شماره آن گروه را تعیین کند.



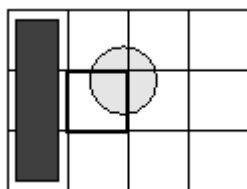
شکل ۳-۱۱ ساختار فیلد برای جریان های H.۲۶۱

۳-۳-۳ استاندارد MPEG

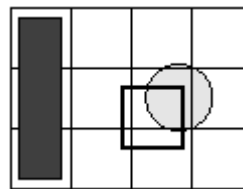
این استاندارد در سال ۱۹۸۸ برای انتقال اطلاعات معرفی شد. این استاندارد برای ارائه کیفیت VHS بر روی CD ها ارائه شده است و دارای سه بخش صدا، تصویر و واحد کنترل می باشد. گاهی اطلاعاتی که ماکرو بلاک لازم دارد در فریم مرجع قبلی وجود ندارد. به تصویر ۳-۱۲ توجه کنید نمی توان بلوک سیاه رنگ را با استفاده از فریم های قبلی نمایش داد. به همین دلیل در استاندارد MPEG۲ فریم نوع B معرفی شد. در این نوع فریم به جای استفاده از یک فریم از دو فریم مرجع استفاده می شود.



پ) فریم قبلی



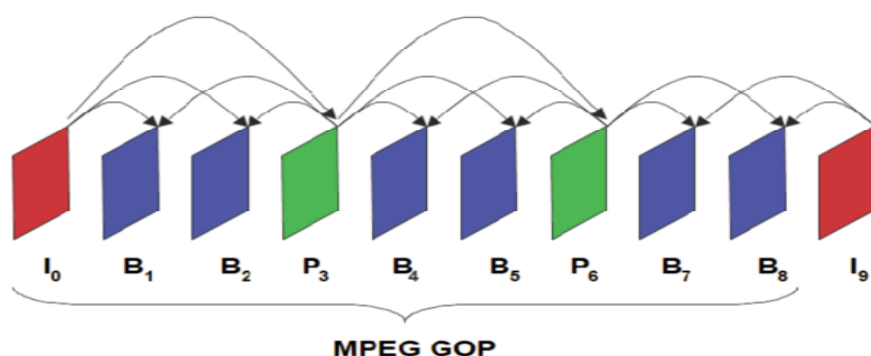
ب) فریم جاری



الف) فریم بعدی

شکل ۳-۱۲ در تصویر فوق نقطه ضعف استاندارد H.۲۶۱ را برای پیش بینی بلوک مناسب با استفاده از یک فریم مرجع را نشان می دهد. در این شکل نمی توان بلوک سیاه رنگ را با استفاده از فریم های قبلی نمایش داد. به همین دلیل در استاندارد MPEG۲ فریم نوع B را معرفی کردند. در این نوع فریم به جای استفاده از یک فریم از دو فریم مرجع استفاده می شود.

فریم B حالت تعمیم یافته فریم P می باشد. برای اینکار فاصله فریم جاری با فریم قبلی و فریم بعدی آن محاسبه شده و میانگین دو مقدار به دست آمده حساب می شود. در استاندارد MPEG، معمولاً هر چند فریم تشکیل یک گروه را می دهند که ساختاری به صورت شکل ۳-۱۳ دارد. این ساختار لزوماً مورد استفاده قرار نمی گیرند.



شکل ۳-۱۳ یک نمونه از گروه ها فریم در استاندارد MPEG

۳-۱- تفاوت های MPEG و H.۲۶۱

به علت بیشتر شدن فاصله با فریم مرجع در استاندارد H.۲۶۱ لازم است تا فضای مورد جستجو برای بردار حرکت بزرگتر انتخاب شود. برای افزایش دقت در محاسبات به جای ۱ پیکسل از کسری از پیکسل استفاده می کنند تا به این طریق دقت بیشتر شود. در ضمن ساختار جریان ها در استاندارد MPEG امکان forward و backward را ارائه می کند.

۳-۲- مراحل الگوریتم MPEG۱

همانطور که در قسمت های قبلی اشاره شد، استاندارد MPEG۱ همزمان با استاندارد های گروه H توسعه داده شدند. در این استاندارد علاوه بر معیارها و مراحل که در استاندارد H.۲۶۱ وجود دارد، معیارها و ویژگی های جدیدی ارائه شد که کارایی این استاندارد را به نسبت هم تایش H.۲۶۱ افزایش دهد. از مهمترین این اختلاف بین این دو نوع استاندارد معرفی فریم نوع B می باشد که برای پیش

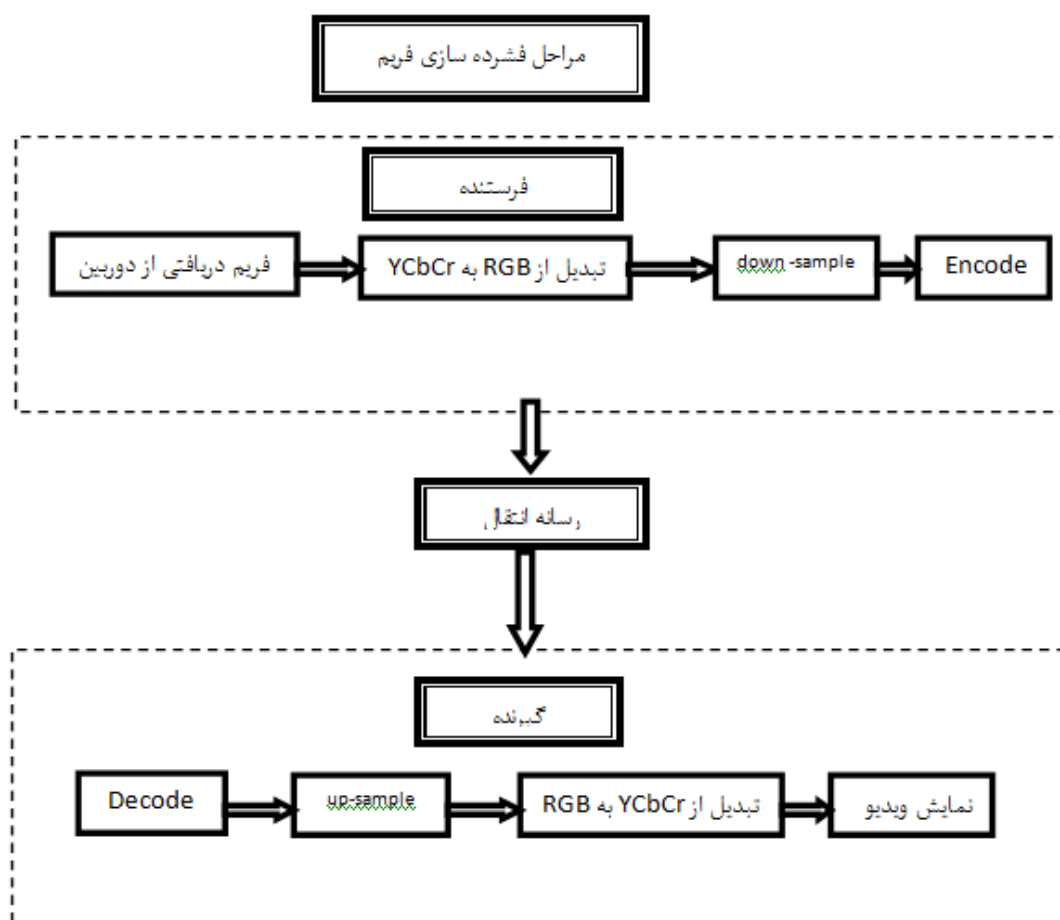
بینی آن می توان از روش درون یابی استفاده کرد. همچنین با معرفی گروه فریم ها این امکان فراهم شد که در صورت از بین رفتن قسمتی از ویدئو تمام ویدئو از بین نرود. علاوه بر موارد ذکر شده، در این استاندارد، خطا از فریم های اولیه به فریم های گروه های دیگر انتشار نمی یابد. استاندارد MPEG دارای ساختارهایی با فیلدها و سرایندهای^۵ زیاد می باشد، که جهت انتقال و همگام سازی رسانه ها و وسایل لازم است، در این قسمت از این فیلدها صرف نظر می شود و به معرفی گام های لازم برای فشرده سازی ویدئو در این استاندارد می پردازیم.

الف) مدل رنگی

همانطور که در قسمت ۱-۳ اشاره شد مدل مرسوم برای نمایش رنگ ها مدل RGB می باشد. اما اشکالی که در این مدل وجود دارد این است که مولفه های رنگی در این سیستم دارای همبستگی کمی می باشند. به همین دلیل نمی توان در این مدل رنگی عمل فشرده سازی را انجام داد.

در استاندارد MPEG ابتدا تصویر از فضای رنگی RGB به فضای رنگی YCbCr انتقال داده می شود. در این فضای رنگی جدید مولفه Y میزان درخشندگی را نشان می دهد و مولفه های Cr و Cb مولفه های رنگی را نشان می دهند. بعد از نمونه برداری، داده ها کد شده و از طریق رسانه انتقال منتقل می شود. در سمت گیرنده، داده های دیکد می شوند مولفه های رنگی up-sample می شوند و از فضای رنگی YCbCr به فضای رنگی RGB منتقل می شوند سپس تصویر نمایش داده می شود مراحل کار را در شکل ۳-۱۴ مشاهده می کنید.

^۵ Heading



شکل ۳-۱۴ مراحل فشرده سازی ویدئو در استاندارد MPEG

ب) نمونه برداری از فضای رنگی

از آنجایی که چشم انسان نسبت به مولفه های رنگی حساسیت کمتری دارد، می توان با تعداد نمونه های کمتری تصویر اصلی را بدون آنکه کیفیت تصویر تغییر کند بازیابی کرد. اما حجم داده ذخیره شده به صورت قابل ملاحظه ای کاهش می یابد. مثلا فرض کنید دارای تصویر با ابعاد 256×256 باشیم آنگاه تصویر رنگی دارای سه ماتریس Y, Cb, Cr خواهد بود که هر کدام از این ماتریس ها ابعاد 256×256 دارد. حال اگر از نمونه برداری $4:2:0$ استفاده کنیم (یعنی از هر ۴ نمونه ۱ نمونه را انتخاب کنیم) ابعاد ماتریس مورد نیاز برای Cb و Cr برابر 64×64 خواهد شد که برای یک فریم

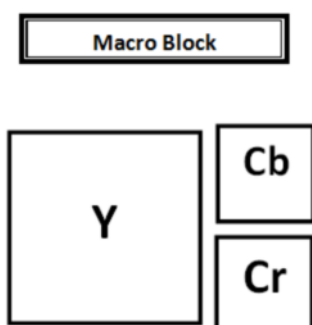
$122880 = (256^2 - 64^2) * 2$ درایه صرفه جویی می شود. اگر نرخ نمونه برداری ویدئو برابر ۳۰

فریم بر ثانیه باشد، آنگاه برای یک ساعت ویدئو میزان صرفه جویی در درایه ها برابر می شود:

$$3600 * 30 * 122880 = 1327104000$$

پ) کد کردن مبتنی بر بلوک

الگوریتم های بیان شده در استاندارد MPEG الگوریتم های مبتنی بر بلوک می باشند. به عبارتی در این روش ها به جای کد کردن تمام فریم، آنها را به بلوک های غیر همپوشانی تقسیم و مراحل بعدی را بر روی این بلوک ها اعمال می کنیم. در استاندارد MPEG معمولا از واحد های دیگری به نام ماکرو بلوک استفاده می شود. هر ماکرو بلوک شامل ۶ بلوک می باشد که از ۴ بلوک Y و دو بلوک رنگی تشکیل شده است تصویر یک ماکرو بلاک را در شکل ۳-۱۵ نشان داده شده است.



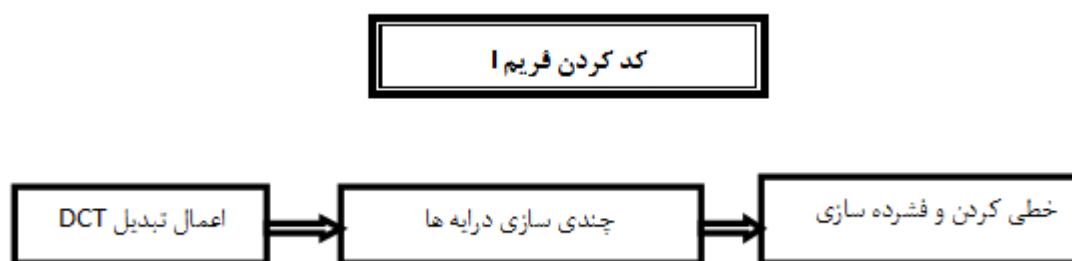
شکل ۳-۱۵ ماکرو بلاک های مورد استفاده در استاندارد MPEG

ت) کد کردن فریم نوع I

فریم های نوع I فریم های مرجع می باشند. این نوع فریم ها دارای محتوی مستقل می باشند. به همین دلیل برای کد کردن این فریم ها معمولا از روش های کد کردن مستقل (کد کردن تصاویر) استفاده می شود. این فریم ها مهم ترین نوع فریم در گروه فریم ها می باشند. زیرا سایر فریم ها را با استفاده از

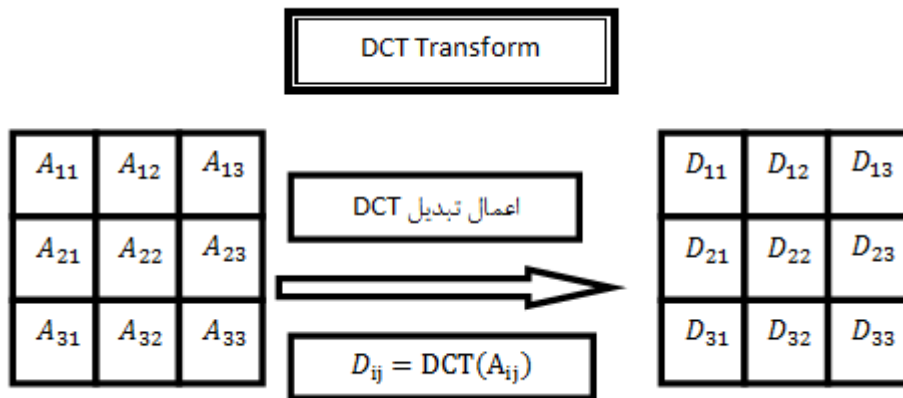
این نوع فریم می توان تخمین زد. اگر خطایی در فشرده سازی این نوع فریم اتفاق بیفتد این خطا، در سایر فریم هایی که با استفاده از این فریم تخمین زده شده اند، انتشار پیدا می کنند.

برای جداسازی فرکانس های بالا و پایین در ماتریس فرکانس ها، فرض می شود که فرکانس ها به صورت زیگزاگ درون ماتریس ذخیره شده است یعنی کمترین فرکانس در ابتدای این ماتریس قرار دارد و هر چه مقدار فرکانس ها به سمت پایانی نزدیک می شود فرکانس بالاتر انتخاب می شود. مراحل کد کردن فریم نوع I در شکل ۳-۱۶ آمده است.



شکل ۳-۱۶ مراحل مورد نیاز برای فشرده سازی فریم نوع I

مراحل طی شده برای انتقال تصویر به حوزه فرکانس در شکل ۳-۱۷ نشان داده شده است. همانطور که در این تصویر مشاهده می شود هر کدام از تصویر به صورت بلوک تقسیم می شود و بر روی هر کدام از بلوک ها تبدیل DCT اعمال می شود تا ماتریس D را نمایش دهد. هر کدام از بلوک های A_{ij} ماتریسی با ابعاد $8*8$ می باشد.

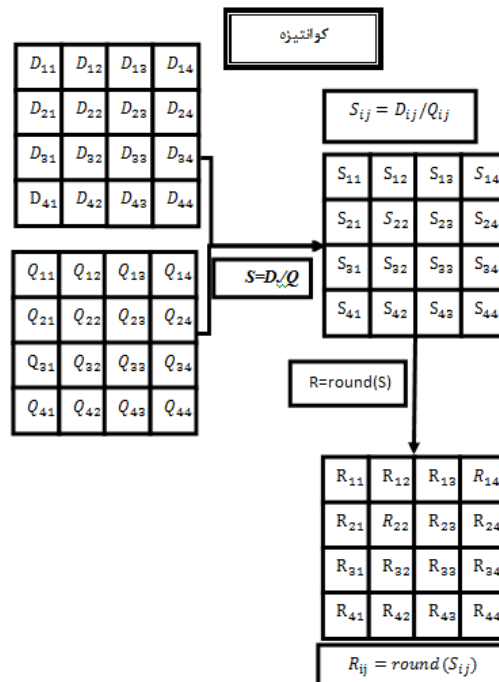


شکل ۳-۱۷ مراحل لازم برای تبدیل DCT بر روی تصویر، ابعاد بلوک ها 8×8 می باشد. همانطور که در این تصویر مشاهده می شود که هر کدام از تصویر به صورت بلوک تقسیم می شود و بر روی هر کدام از بلوک های تبدیل DCT اعمال می شود تا ماتریس D را نمایش می دهد. هر کدام از بلوک های A_{ij} ماتریسی با ابعاد 8×8 می باشد.

بعد از انتقال اطلاعات به فضای فرکانس لازم است تا داده ها را چندی سازی کنیم به قسمی که فرکانس های بالا بیشتر تضعیف شوند. برای این منظور ماتریسی به نام ماتریس Q با درایه های زیر تعریف می گردد.

$$Q = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

مقدار هر کدام از بلوک ها را بر مقادیر ماتریس Q به صورت نظیر به نظیر تقسیم می کنیم. در گام دوم مقدار گرد شده این اعداد را قرار می دهیم. در شکل ۳-۱۸ نمونه ای از چندی سازی نشان داده شده است.



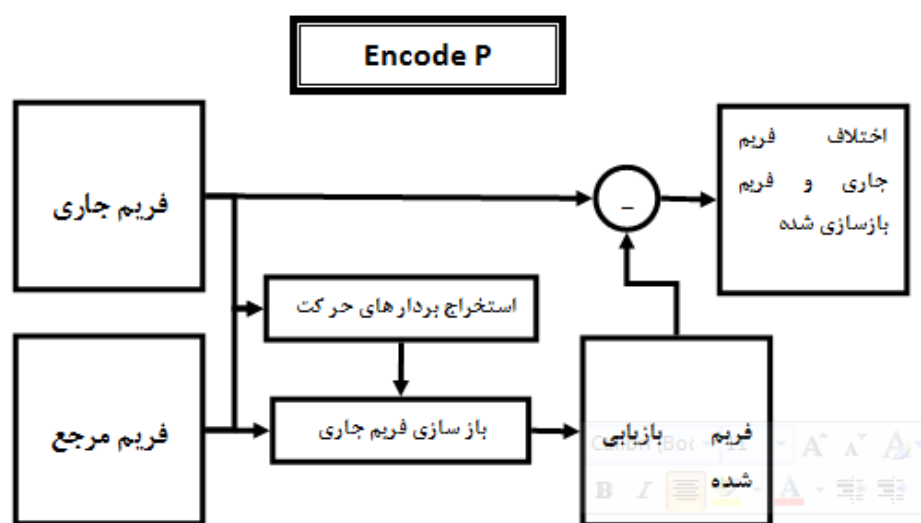
شکل ۳-۱۸ مراحل لازم برای کوانتیزه کردن ماتریس فرکانس برای سادگی ابعاد این ماتریس را 4×4 در نظر گرفتیم در حالی که در سیستم های فشرده سازی ابعاد ماتریس 8×8 می باشد. برای چندی سازی هر عنصر در حوزه فرکانس را بر درایه متناظر آن در ماتریس کوانتیزه تقسیم می کنیم.

ث) نحوه کد کردن فریم نوع P

فریم نوع P فریم هایی هستند که به صورت Inter (temporally) کد می شود منظور از Inter این است که در این روش از ارتباطی که بین فریم جاری و فریم پیشین وجود دارد برای فشرده سازی فریم استفاده می شود. در استاندارد MPEG تاکید دارد که فریم پیشین باید فریمی از نوع I و یا P باشد.

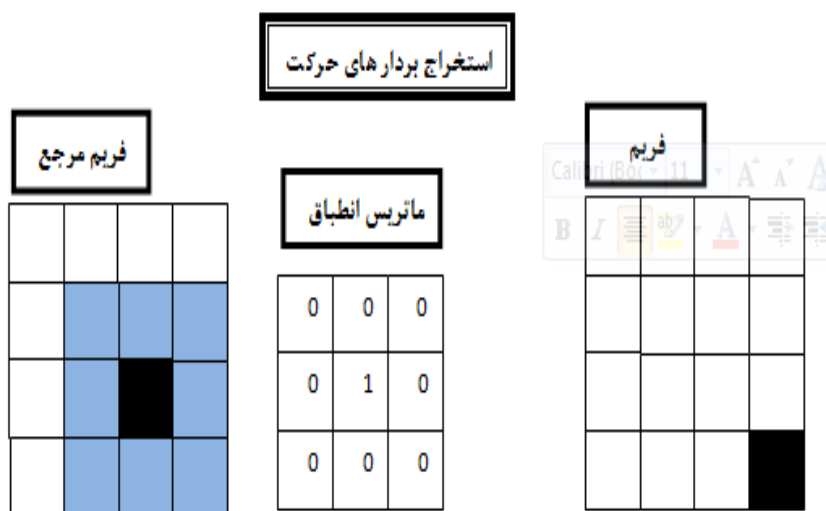
برای استفاده از افزونگی بین فریم ها، باید از بردار های حرکت بهره برد. به این منظور هر کدام از بلاک های فریم جاری، با بلاک های واقع شده در یک پنجره در فریم مرجع مقایسه شده و شبیه

ترین بلاک انتخاب می‌گردد. فرض می‌کنیم این بلاک، بلاکی است که با استفاده از یک بردار حرکت به موقعیت جدید در فریم جاری منتقل شده است. با استفاده از بردارهای حرکت و فریم مرجع، فریم جاری پیش بینی شده و اختلاف فریم جاری و فریم پیش بینی شده محاسبه می‌گردد. در نهایت بردارهای حرکت و اختلاف بین فریم‌ها ذخیره می‌گردد.

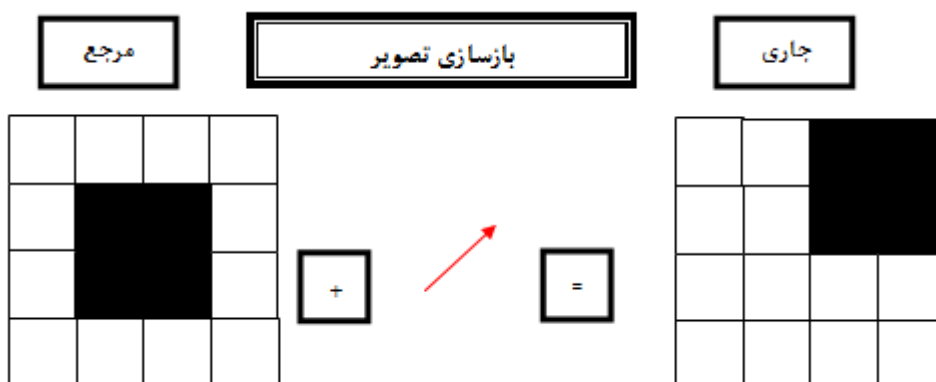


شکل ۳-۱۹ مراحل مورد نیاز برای کد کردن فریم نوع P

در شکل ۳-۱۹ مراحل مورد نیاز برای فشرده سازی فریم P نشان داده شده است. یکی از مهم ترین مراحل، استخراج بردارهای حرکت می باشد؛ همانطور که در مثال در شکل ۳-۲۰ نشان داده شده است. در این روش هر بلوک، با تمام بلوک های موجود در یک پنجره در فریم مرجع مقایسه، و از این میان بلوکی با بالاترین انطباق، انتخاب می شود و با استفاده از آن، بردار حرکت بدست آورده می شود. معیاری که برای این منظور استفاده می شود جمع قدر مطلق اختلاف ها می باشد. گام دیگر در این الگوریتم، بازسازی فریم جاری با استفاده از بردار حرکت و فریم مرجع می باشد. شکل ۳-۲۱ این گام را نشان می دهد. در فریم جاری شیئی وجود دارد که نسبت به فریم مرجع جابه جا شده است. مقدار و جهت این جابه جایی با استفاده از بردار حرکت تعیین می شود. بنابراین اگر فریم مرجع و بردارهای حرکت در دسترس باشد می توان فریم جاری را تخمین زد.



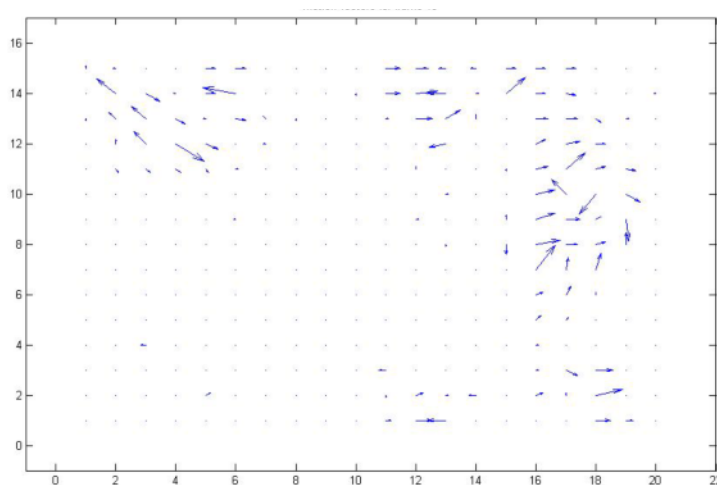
شکل ۲۰-۳ استخراج بردارهای حرکت



شکل ۲۱-۳ بازسازی فریم با استفاده از بردار حرکت و فریم مرجع

این الگوریتم ها پارامترهای فراوانی دارند که کارایی و کیفیت ویدئو به انتخاب این پارامترها بستگی دارد. مثلا پارامتر scale که میزان چندی سازی را تعیین می کند، در صورتی که مقداری کوچک باشد کیفیت تصویر بازیابی شده نسبتا مناسب خواهد بود. اما اگر این مقدار را بزرگ انتخاب کنیم باعث به وجود آمدن اثر بلوکی بر روی تصویر خواهد شد. از طرفی هر چه مقدار این چندی سازی بزرگ انتخاب شود کارایی الگوریتم از جهت بالارفتن درصد فشرده سازی بالا خواهد رفت.

حال می خواهیم بررسی کنیم آیا بردار های حرکت می توانند به درستی جابه جایی را به تصویر بکشند این بردار ها برای چند فریم در شکل ۳-۲۲ آمده است. همانطور که مشاهده می کنید بیشتر بردار های حرکت در قسمت های بدون حرکت تصویر برابر، بردار ضفر هستند. فقط در قسمت بالا سمت چپ از تصویر که علت تغییرات نوری بردار های حرکت به وجود آمده است و سمت راست تصویر که شی در حال حرکت می باشد اثراتی از جابه جایی و بردار های حرکت در جهت حرکت نشان داده می شود. غالب بردار های حرکت جهت حرکت را نشان می دهند.



شکل ۳-۲۲ بردار حرکت برای چند فریم دلخواه

ج) کد کردن فریم نوع B

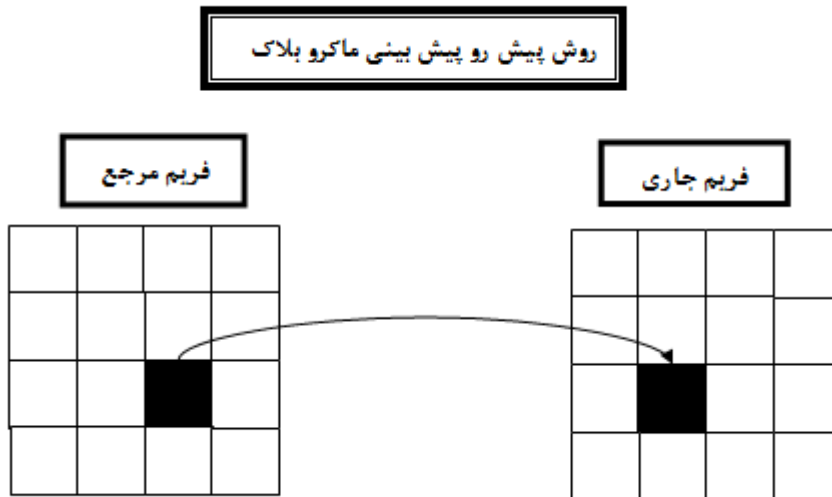
همانطوری که در تصویر ۳-۱۲ نشان داده شد، یکی از نقاط ضعف استاندارد H.۲۶۱ این است که در برخی از مواقع، صرفاً با استفاده از یک فریم مرجع نمی توان بعضی از بلوک های فریم جاری را به درستی پیش بینی نمود. به همین دلیل لازم بود تا روش های دیگری برای پیش بینی بلوک ها ارائه شود. روش های تخمین بلوک را می توان به سه دسته تقسیم نمود.

۱. تخمین پیش رو

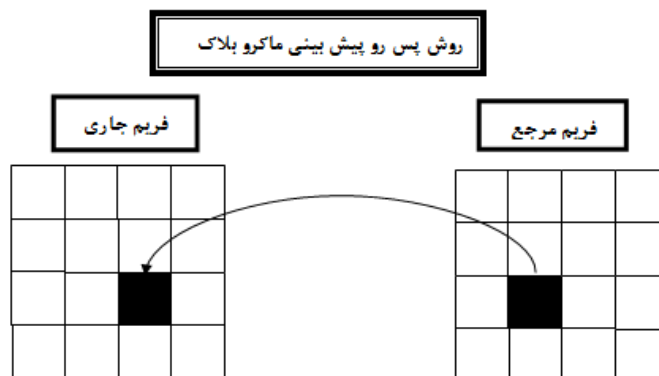
۲. تخمین پسرو

۳. تخمین درونیایی

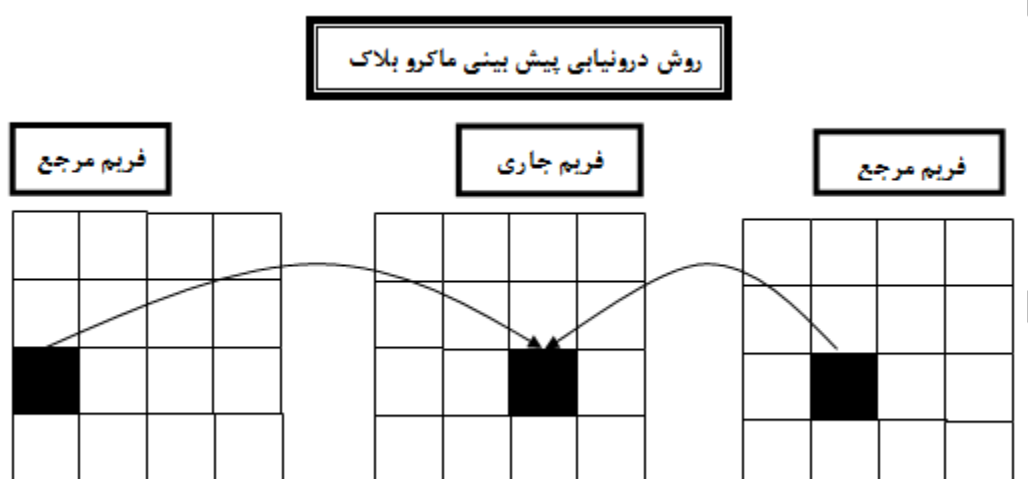
این روش ها را می توان در تصاویر ۳-۲۳ تا ۳-۲۵ مشاهده کرد.



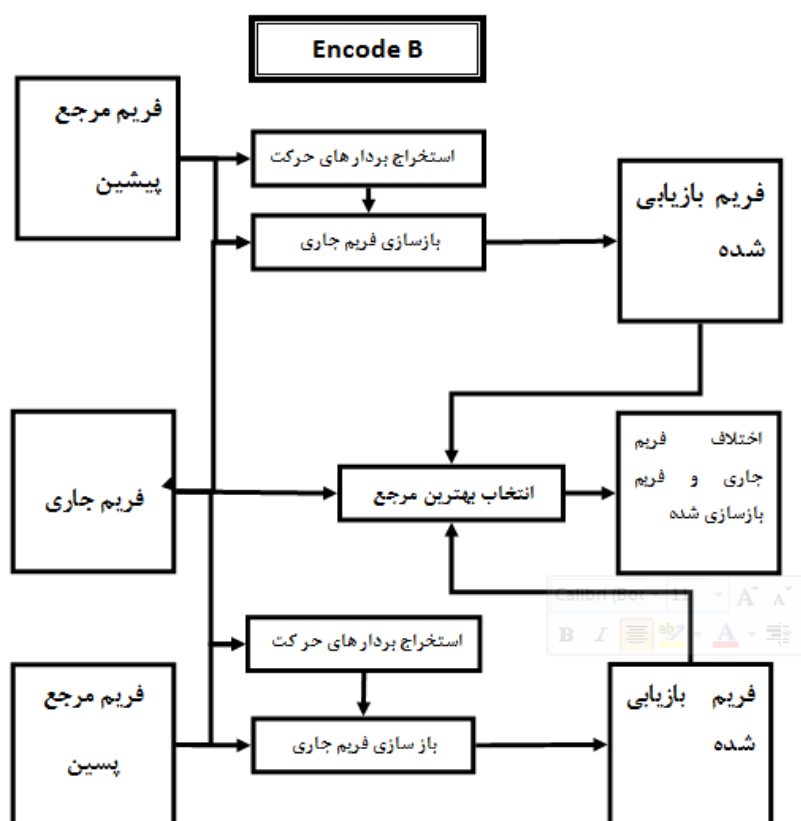
شکل ۳-۲۳ روش پیش بینی ماکرو با استفاده از روش پیش رو



شکل ۳-۲۴ روش پس رو برای پیش بینی ماکرو بلاک



شکل ۳-۲۵ روش درونیابی پیش بینی ماکرو بلاک ها



شکل ۳-۲۶ مراحل کد کردن فریم برای فریم های نوع B

مراحل کد کردن فریم های نوع B در تصویر ۳-۲۶ نشان داده شده است. در کد کردن این نوع فریم برای تطبیق بلوک مورد نظر، بلوک را با دو مرجع مقایسه می کنیم. بنابراین زمان محاسبات به دو

برابر افزایش می یابد. از طرفی به یک بیت اضافه برای هر بلوک نیاز است که تعیین کند کدام فریم مرجع برای بازیابی فریم مرجع استفاده شده است. اما این بالا رفتن هزینه ها برای کاهش خطا (افزایش دقت) و در نتیجه کاهش حجم ذخیره سازی می شود.

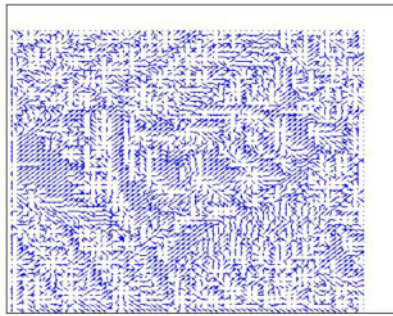
ج) بررسی و خلاصه (از نتایج پایان نامه)

شکل ۳-۲۷ اثر اندازه بلوک را بر میدان حرکت و تصویر بازیابی شده نشان می دهد. هر چه اندازه بلوک ها کوچکتر باشد، میدان ها به دلیل همگون و شبیه شدن بلوک های همسایه منظم تر می شوند. به همین دلیل بردارهای حرکت متناظر با بلوک همسایه همانند می باشند. در مقابل هر چه اندازه بلوک ها بزرگتر باشد، میدان به دلیل ناهمگونی بلوک های بزرگ، نامنظم می باشد.

ویژگی دیگری که مورد بررسی قرار می دهیم اثر اندازه بلوک ها بر زمان مصرفی می باشد. نتایج این بررسی در جدول ۳-۱ آماده است. همانطور که نتایج نشان می دهد. اگر اندازه بلوک ها خیلی کوچک یا خیلی بزرگتر باشد آنگاه زمان مصرفی بالا می رود. زیرا اگر اندازه بلوک خیلی کوچک باشد تعداد بلوک ها زیاد می شود و زمان محاسبات برای پیدا کردن مکان بلوک ها بالا می رود. از طرفی اگر اندازه بلوک خیلی بزرگ باشد آنگاه بلوک ها قسمت های ناهمگونی را شامل می شود و از طرف دیگر برای مقایسه هر کدام از بلوک ها به تعداد محاسبات بالاتری نیاز دارد. مناسب ترین مقدار برای اندازه بلوک در این آزمایش برابر ۱۶ می باشد.

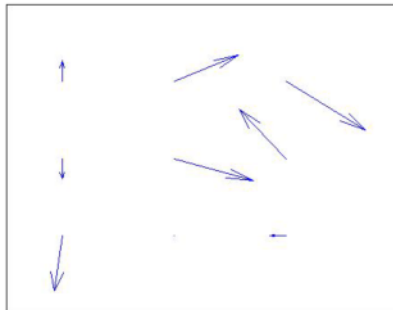
جدول ۳-۱ بررسی اثر اندازه بلوک ها بر زمان کد کردن

اندازه بلوک ها	زمان مصرفی
۴	۳,۳۷۸۲۳۱
۸	۲,۸۳۶۴۶۲
۱۶	۲,۸۲۳۹۰۹
۳۲	۳,۱۱۷۷۸۶
۶۴	۴,۳۲۴۱۴۱



(ب) بردارهای حرکت زمانی که اندازه بلوک ها برابر ۴ است

(الف) ضریب=۳۴,۷۴ و PSNR=۳۴,۷۴



(ت) بردارهای حرکت زمانی که اندازه بلوک ها برابر ۶۴ است

(پ) ضریب=۳۷,۰۶ و PSNR=۳۷,۰۶

شکل ۳-۲۷ بررسی اثر اندازه بلوک بر میدان های بردارهای حرکت در فریم

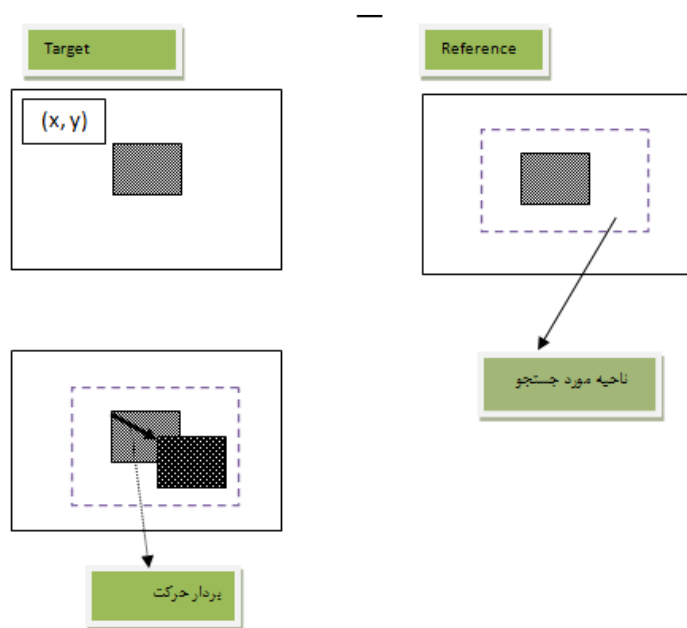
۳-۴- تخمین بردارهای حرکت

رمزگذار بین فریم^۶ یکی از رایج ترین روش ها برای کاهش افزونگی بین فریم های ویدیویی می باشد. در این روش، ابتدا فریم را پیش بینی می کنیم و به جای ارسال فریم اختلاف بین فریم واقعی و فریم پیش بینی شده را ارسال می کنیم. در بسیاری از مواقع این اختلاف ها اندک می باشد. اگر در درون صحنه حرکتی وجود نداشته باشد، اختلاف بین فریم های متوالی کم و نزدیک به صفر می باشد. بنابراین می توانیم مقادیر صفر یا نزدیک به صفر را نادیده بگیریم و به این طریق به حجم بالایی از فشرده سازی دست یابیم.

^۶ INTERFRAME predictive coding

آنچه در قبل فرض کردیم این بود که در صحنه ای که نمونه برداری می شود شی در حال حرکت نباشد. اما در بسیاری از مواقع در صحنه اشیایی در حال حرکت می باشند. اگر موقعیت و بردار حرکت شی را بدانیم، می توانیم با استفاده از فریم های قبلی و این بردار حرکت فریم واقعی را پیش بینی کنیم و به این وسیله حجم فشرده سازی را بالا ببریم. برای این کار لازم است تا روش های مناسبی برای پیش بینی حرکت اشیا داشته باشیم. یکی از روشهای مرسوم این است که فریم را به بلوک هایی تقسیم نموده و در صورتی که اختلاف بین بلوک فریم مرجع و فریم مقصد کمتر از حد آستانه بود، فرض می شود که صحنه در آن بلوک تغییر نکرده است. در غیر این صورت در درون فریم مرجع بلوکی را جستجو می کند که دارای بالاترین شباهت با بلوک فریم مقصد دارد. روش های فراوانی به منظور بالابردن دقت و کارایی این روش پیشنهاد شده است که در ادامه نمونه هایی از آن معرفی می شود.

فرض کنید تصویر مرجع و تصویر مقصد به صورت شکل ۳-۲۸ باشد. هدف پیدا کردن برداریست که نشان دهد بلوک سیاه رنگ در تصویر مرجع به کدام نقطه در فریم مقصد منتقل شده است. برای این منظور لازم است تا ناحیه را در فریم مرجع تعیین کنیم که در آن به جستجو بلوک مطلوب پردازیم. به این ناحیه، ناحیه جستجو می گوییم. اندازه این ناحیه در بالا بردن دقت و کم یا زیاد کردن زمان پردازش تاثیر دارد.



شکل ۳-۲۸ جستجو یک بلوک درون یک ناحیه

یک بلوک از تصویر مقصد در موقعیت (x, y) و $R(x+i+k, y+j+l)$ یک بلوک در موقعیت $C(x+k, y+l)$ را نشان می دهد. برای به دست آوردن میزان شباهت بین بلوک ها، لازم است تا تابع فاصله ای تعریف گردد. برای این هدف معمولاً از تابع مجموع قدر مطلق فاصله ها که رابطه (۳-۳) آمده است استفاده می شود.

$$MAE(i, j) = \frac{1}{N^2} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} |C(x+k, y+l) - R(x+i+k, y+j+l)| \quad (3-3)$$

یک راه حل این است که بلوک موجود در فریم مقصد را با تمام بلوک های فریم مرجع مقایسه نموده و بلوکی را که دارای کمترین اختلاف (بیشترین شباهت) می باشد، به عنوان بلوک مطلوب انتخاب شود. این روش معمولاً زمانبر می باشد؛ به همین دلیل لازم است از روش هایی مانند $D-3$ logarithmic search (LOGS) [۲۲], three-step search (TSS) [۱۱], conjugate direction search (CDS) [۱۰], cross search (CS) [۱۸], new three-step search (NTSS) [۱۲], four-step search (۴SS) [۸], block-based gradient descent search (BBGDS) [۱۳]

محاسبات کمتری دارد استفاده کنیم. روش های یاد شده با انتخاب استراتژی و الگوی مناسب برای جستجو بلوک مناسب سعی می کنند تعداد نقاط لازم برای جستجو را کاهش دهند.

۳-۴-۱- مروری بر روش های تخمین بلوک

انتخاب الگوی برای پیدا کردن بلوک مناسب، در کارایی و سرعت روش تاثیر بسزایی دارد. معمولا از اشکال مربعی برای انتخاب بلوک ها استفاده می کنیم. دقیق ترین روش برای جستجوی بلاک ها، روش جستجو کامل^۷ می باشد. در این روش تمام بلوک های موجود در پنجره جستجو مورد ارزیابی قرار می گیرند و مناسب ترین بلوک (بلوک با بالاترین مشابهت) انتخاب می شود.

روش جستجوی کامل به علت بالا بودن زمان محاسبات، روش مناسبی برای انطباق بلوک نمی باشد. در صورتی که اندازه پنجره جستجو برابر $\pm w$ باشد تعداد نقاط مورد جستجو برابر $(2w + 1)^2$ می باشد. به همین دلیل در کاربرد های عملی از روش های مانند ۳- *D logarithmic search (LOGS)* [۲۲], *three-step search (TSS)* [۱۱], *conjugate direction search (CDS)* [۱۰], *cross search (CS)* [۱۸], *new three-step search (NTSS)* [۱۲], *four-step search (۴SS)* [۸], *block-based gradient descent search (BBGDS)* [۱۳] استفاده می شود. اگر چه این روش ها سرعت را بالا می برند اما همیشه این خطر وجود دارد که در مینیمم های محلی گرفتار شوند. معمولا این روش ها کارایی مناسبی را در کاربرد های عملی از خود نشان داده اند و به همین دلیل از این روش ها در سیستم های فشرده سازی استفاده می شود.

الف) الگوریتم جستجو Unrestricted Center-Biased Diamond

این الگوریتم را به اختصار UCBD می نامیم و مراحل الگوریتم آن به صورت زیر می باشد.

^۷ full search

۱. starting: الگویی که برای جستجو الگو مورد استفاده قرار می گیرد یک لوزی می باشد که ۹ نقطه آن مورد استفاده قرار می گیرد. اگر نقطه بهینه در مرکز الگو باشد به مرحله Ending برو در غیر این صورت به مرحله searching برو. تصویر ۳-۲۹ الگویی را که برای جستجو لازم است را نشان می دهد.

۲. searching: اگر نقطه بهینه بر روی رئوس یعنی $(c, c + 2), (c, c - 2), (c + 2, c), (c - 2, c)$ باشد از جستجو vertex استفاده می کنیم. در غیر این صورت اگر نقاط واقع بر روی لبه ها باشد از جستجو face استفاده می کنیم نقاط face یعنی

$$(c - 1, c), (c + 1, c), (c, c - 1), (c, c + 1).$$

a. vertex search: الگویی که برای این جستجو مورد استفاده قرار می گیرد به صورت

شکل ۳-۳۰ می باشد برای این جستجو ۵ نقطه جدید مورد ارزیابی قرار می گیرد.

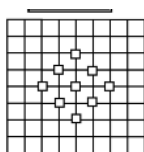
b. face search: الگویی که برای این جستجو مورد استفاده قرار می گیرد به صورت

شکل ۳-۳۱ می باشد برای این جستجو ۳ نقطه جدید مورد ارزیابی قرار می گیرد.

اگر در هر کدام از مراحل جستجو نقطه بهینه در مرکز الگو قرار گرفت. به مرحله Ending برو در غیر این صورت مرحله جستجو را ادامه بدهید.

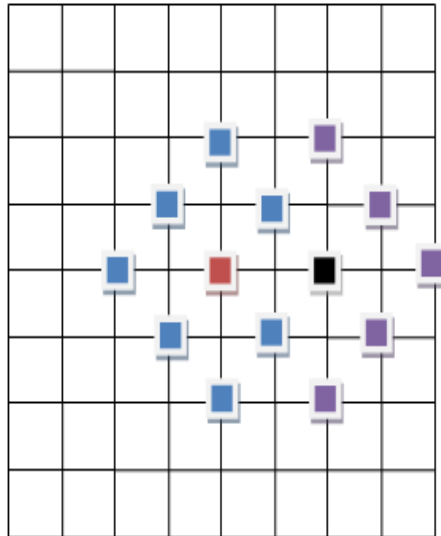
۳. Ending: الگویی که برای این جستجو مورد استفاده قرار می گیرد به صورت شکل ۳-۳۳ می

باشد برای این جستجو ۴ نقطه جدید مورد ارزیابی قرار می گیرد. از بین نقاط، مقدار بهینه را انتخاب می کنیم و به کمک آن بردار حرکت را ترسیم می کنیم.



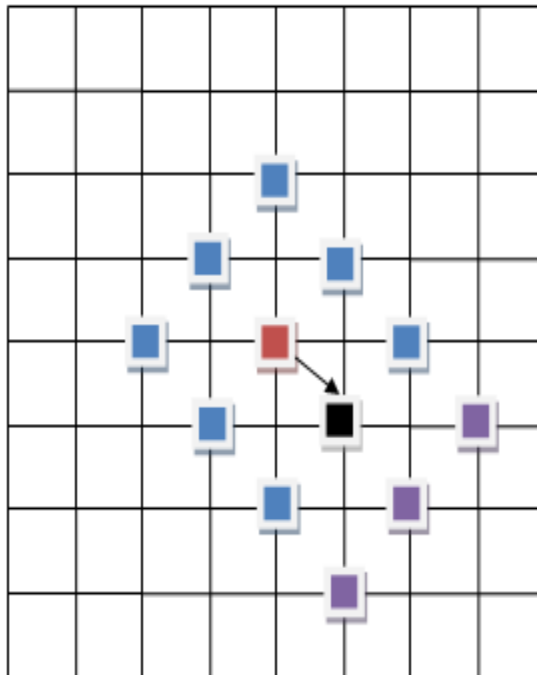
شکل ۳-۲۹ الگوی مورد استفاده برای جستجو در الگوریتم UCBD.

جستجو راس (vertex)

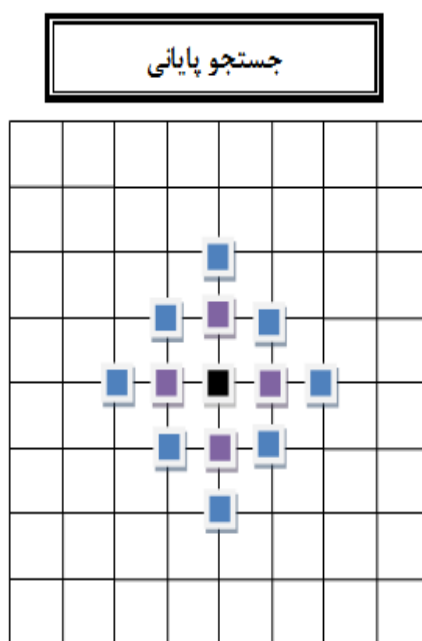


شکل ۳-۳ جستجو بر روی راس در الگوریتم UCBD

جستجو لبه (face)



شکل ۳-۳ جستجو بر روی لبه در الگوریتم

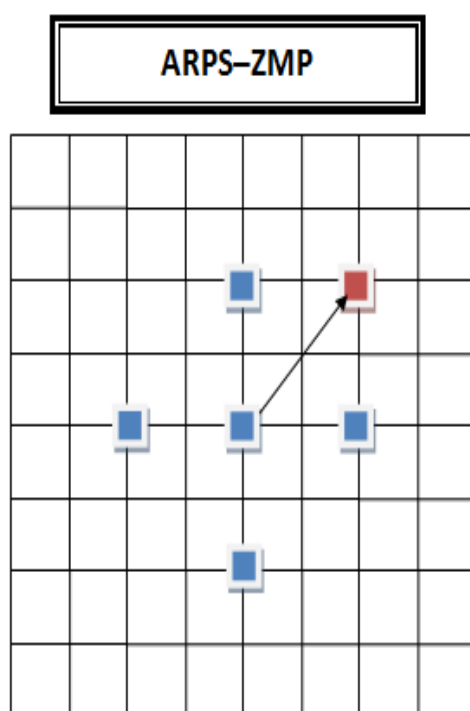


شکل ۳-۳۲ جستجو در مرحله پایانی الگوریتم UCBD

ب) الگوریتم جستجو ARPS-ZMP

بلوک های که در مجاورت هم قرار دارند معمولاً دارای رفتار های مشابه می باشند. بنابراین اگر بردار حرکت را برای بلوک سمت چپ یا بلوک سمت بالا را داشته باشیم می توانیم مقدار بردار حرکت را با استفاده از آنها تخمین بزنیم و به کمک آن در زمان محاسبات صرفه جویی کنیم. این الگوریتم دارای مراحل زیر می باشد [۲۱].

۱. starting: فاصله بین بلوک مرجع و بلوک مقصد را محاسبه می کنیم
۲. مرکز ARP که به وسیله بردار های حرکت نمایش داده می شود را انتخاب کنید در ۴ نقطه به علاوه نقطه ای که توسط بردار حرکت نمایش داده می شود را مورد ارزیابی قرار می دهیم.
۳. مرکز جستجو را در نقطه ای قرار بده که دارای بالاترین شباهت باشد اگر نقطه بهینه در مرکز الگو قرار بگیرد جستجو به پایان می رسد در غیر این صورت جستجو را ادامه می دهیم.



شکل ۳-۳ نقاط مورد جستجو برای الگوریتم ARPS-ZMP

پ) الگوریتم جستجو چهار گام (Four-Step Search Algorithm)

برای جابه‌جایی‌های بزرگ یعنی جابه‌جایی‌هایی که به ± 7 گام، نیاز دارد، در روش ۳SS بر روی شبکه $9*9$ ، نه نقطه مورد ارزیابی قرار می‌گیرد. اما در این روش برای اولین گام ۹ نقطه را بر روی مربعی به ابعاد $5*5$ مورد ارزیابی قرار می‌دهد. اگر نقطه مینیمم در مرکز مربع قرار گیرد اندازه شبکه به $3*3$ کاهش می‌یابد در غیر اینصورت مرکز مربع بر روی نقطه با حداکثر مشابهت قرار گرفته و این کار تا زمانی که مرکز شبکه دارای کوچکترین مقدار اختلاف شود تکرار می‌گردد. مراحل الگوریتم را می‌توان در زیر خلاصه کرد. در شکل ۳-۳ با یک مثال گام‌های طی شده در این الگوریتم را نشان می‌دهیم.

۱. گام اول: در این گام ۹ نقطه را بر روی شبکه ای به ابعاد 5×5 مورد ارزیابی قرار می دهیم و نقطه با حداکثر شباهت را پیدا می کنیم. اگر نقطه در مرکز قرار داشته باشد به گام ۴ برو در غیر اینصورت به گام ۲ برو. (شکل ۳-۳۴)

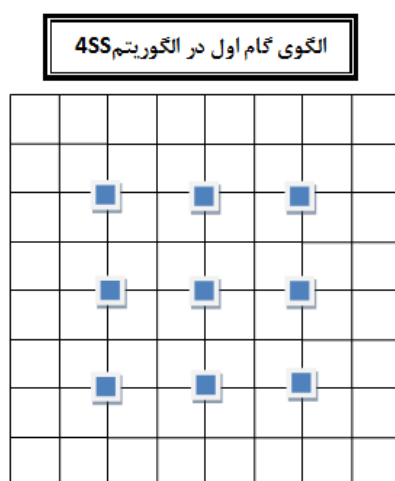
۲. گام دوم: اندازه پنجره مورد جستجو معمولا برابر 5×5 می باشد اما تعداد نقاطی که مورد ارزیابی قرار می گیرد به نقطه که قبلا بالاترین شباهت را داشته است بستگی دارد. (شکل ۳-۳۵)

a. اگر این نقطه یکی از نقاط واقع شده در گوشه های تصویر باشد به ۵ نقطه برای ارزیابی نیاز داریم.

b. اگر این نقاط بر روی محور افقی یا عمودی قرار بگیرد به ۳ نقطه برای ارزیابی نیاز داریم. اگر مینیمم مقدار در مرکز پنجره واقع شده باشد اندازه پنجره مورد ارزیابی 3×3 خواهد شد. اگر مینیمم مقدار در مرکز پیدا شد به گام چهارم برو در غیر اینصورت به گام سوم برویم.

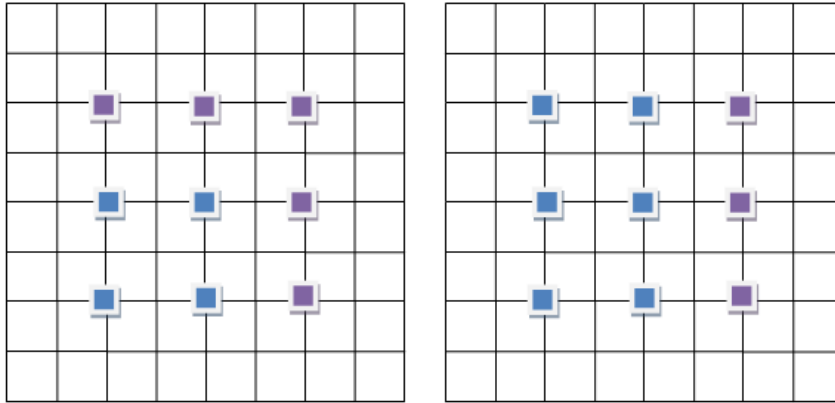
۳. استراتژی این روش همانند روش دو می باشد اما در این مرحله به گام ۴ می رویم. (شکل ۳-۳۵)

۴. اندازه پنجره مورد جستجو به پنجره 3×3 کاهش داده می شود. و جهت بردار حرکت نقطه ای از این نه نقطه است که دارای کم ترین مقدار اختلاف می باشد. (شکل ۳-۳۶)



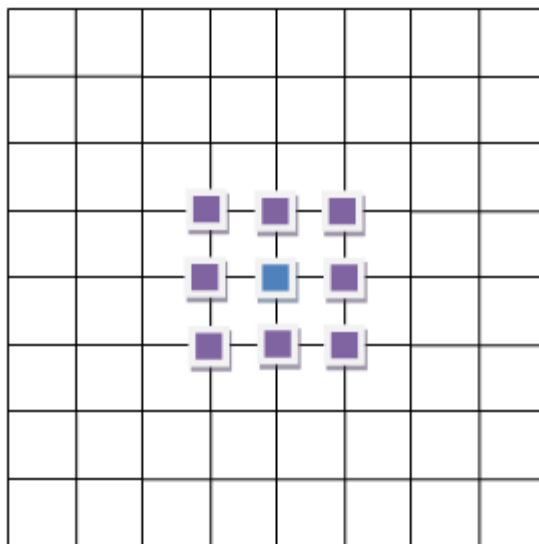
شکل ۳-۳۴ الگوی گام اول

الگوهای گام دوم و سوم در الگوریتم 4SS

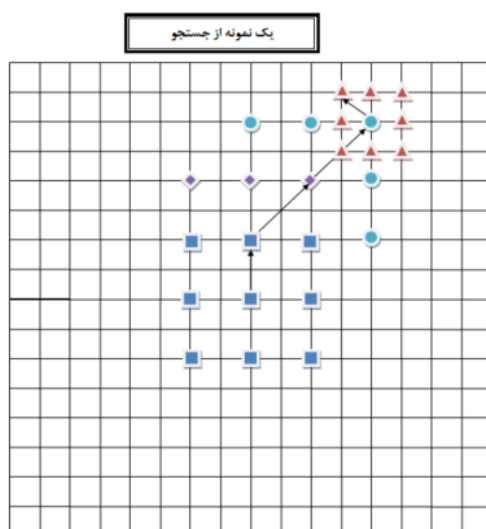


شکل ۳-۳۵ الگوی گام دوم و سوم نقاط آبی نقاطی را نشان می دهد که قبلا مورد ارزیابی قرار گرفته اند و نقاط بنفش نقاط جدید را نشان می دهد.

الگوی گام پایانی در الگوریتم 4SS



شکل ۳-۳۶ الگو در گام نهایی



شکل ۳-۳۷ یک نمونه از جستجو در الگوریتم FSS

ت) الگوریتم جستجو لوزی (DIAMOND SEARCH ALGORITHM)

الگوریتم DS از دو نوع الگو برای جستجو استفاده می کند. این الگوها در شکل ۳-۳۸ نمایش داده شده است. اولین الگو *large diamond search pattern* (LDSP) نامیده می شود که در واقع از ۹ نقطه تشکیل شده است که شکلی به صورت (◇) را نشان می دهد. دومین الگو شکل کوچکتری را تشکیل می دهد و شامل ۵ نقطه می باشد و *small diamond search pattern* (SDSP) نامیده می شود [۱۳].

در استفاده از الگوی LDSP، فرایند جستجو زمانی که نقطه MDB در مرکز قرار بگیرد، ادامه می یابد. سپس جستجو از مرحله LDSP به مرحله SDSP منتقل می شود. در این مرحله نقطه ای که MDB باشد بردار حرکت را نشان می دهد. این الگوریتم را به صورت زیر می توان خلاصه کرد. شکل ۳-۴۰ با یک مثال گام های طی شده در این الگوریتم را نشان می دهد.

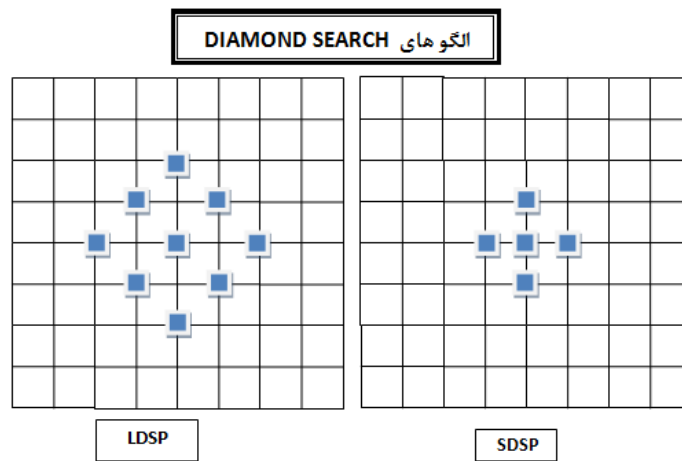
۱. در اولین گام LDSP را در مرکز پنجره جستجو قرار می دهیم. نه نقطه واقع شده را مورد

بررسی قرار می دهیم اگر نقطه ای که دارای کمترین LDSP باشد در مرکز شکل قرار بگیرد به

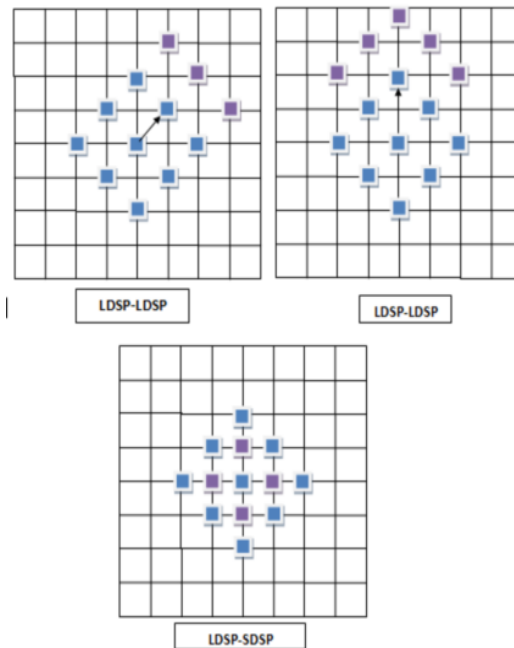
مرحله سوم برو در غیر این صورت به گام دوم برو.

۲. اگر نقطه که MDB در آن واقع شده است در مرکز الگو باشد به گام سوم بروید در غیر این صورت گام دوم را با LDSP ادامه بدهید.

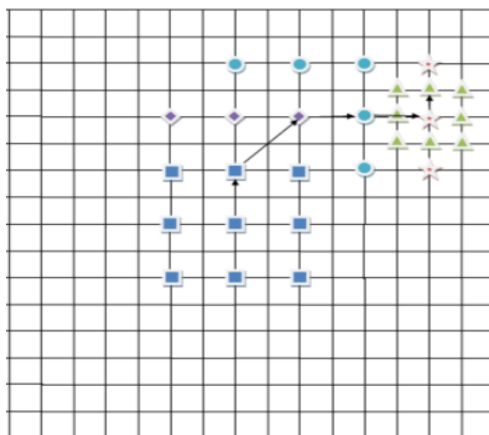
۳. در این مرحله جستجو را از حالت LDSP به وضعیت SDSP منتقل می کنیم. در این وضعیت نقطه ای که در آن MDB واقع می شود بردار حرکت را می توان به کمک آن استخراج کرد.



شکل ۳-۳۸ الگوهای مورد استفاده در الگوریتم Diamond Search



شکل ۳-۳۹ انواع تغییر وضعیت ها در الگوریتم Diamond Search



شکل ۳-۴۰ یک نمونه از جستجو با استفاده از الگوریتم Diamond Search

ث) الگوریتم جستجو کراس (CROSS-SEARCH ALGORITHM(CSA)

مبانی این الگوریتم مانند الگوریتم های ارائه شده در [۱۹] است، اما دارای تفاوت هایی می باشد که نسبت به الگوریتم ها ارائه شده حجم محاسبات کمتری لازم دارد. مهم ترین تفاوت این الگوریتم نسبت به سایر الگوریتم ها این است که این الگوریتم در هر مرحله از محاسبات به تنها ۴ نقطه نیاز دارد. این نقاط می تواند نقاط پایانی شکل (X) یا (+) باشد. مرحله نهایی نیز می تواند هر کدام از الگوها باشد.

نقطه بالا و سمت چپ هر بلوک را می توانیم به عنوان مرجع در نظر بگیریم. بلوک متناظر در فریم قبلی و فریم مرجع به عنوان بلوک (۰،۰) در نظر بگیریم در ادامه جزئیات این الگوریتم را بیان می کنیم. شکل ۳-۴۲ با یک مثال گام های طی شده در این الگوریتم را نشان می دهد

۱. بلوک جاری را با بلوک (۰،۰) مقایسه می کنیم اگر اختلاف کمتر از حد آستانه باشد فرض می کنیم این بلوک، بلوک ثابت است و جستجو را ادامه نمی دهیم در غیر این صورت به گام دوم می رویم.

۲. m و n را P و مقدار دهی اولیه می کنیم معمولاً مقدار $p=w/2$ و $m=n=0$ در نظر می گیریم.

۳. z را مقدار دهی اولیه می کنیم این مقدار برابر $m=n=i=j=0$ می باشد.

۴. اختلاف بلوک ها را در موقعیت $a(i+p, j-p), (i-p, j+p), (i-p, j-p), (i, j)$

$d(i+p, j+p)$ محاسبه می کنیم.

۵. اگر $p=1$ به گام ۶ برو در غیر این صورت اندازه p را نصف کن و به مرحله ۳ برو.

۶. اگر (m,n) یکی از مختصات ها $(i+1, j+1)$ or $(i-1, j-1)$ باشد به گام ۷ برو در

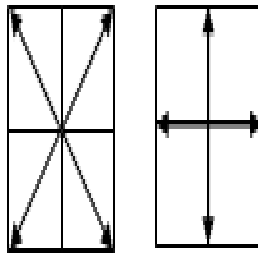
غیر اینصورت به گام ۸ برو.

۷. مختصات های $(m, n+1)$ and $(m+1, n), (m, n-1), (m-1, n)$ را برای پیدا

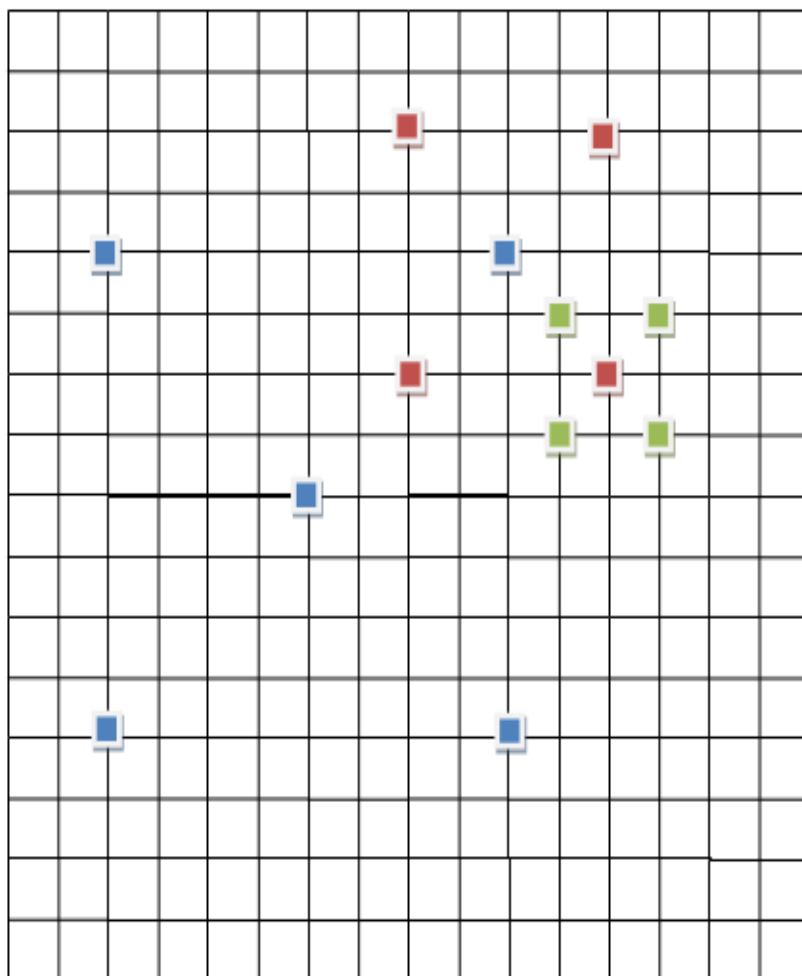
کردن نقطه مینیمم جستجو کن.

۸. مختصات های $(m+1, n+1)$ and $(m+1, n-1), (m-1, n+1), (m-1, n-1)$ را

(۱) را جستجو کن.



شکل ۳-۴۱ الگوهای مورد استفاده در الگوریتم جستجو کراس

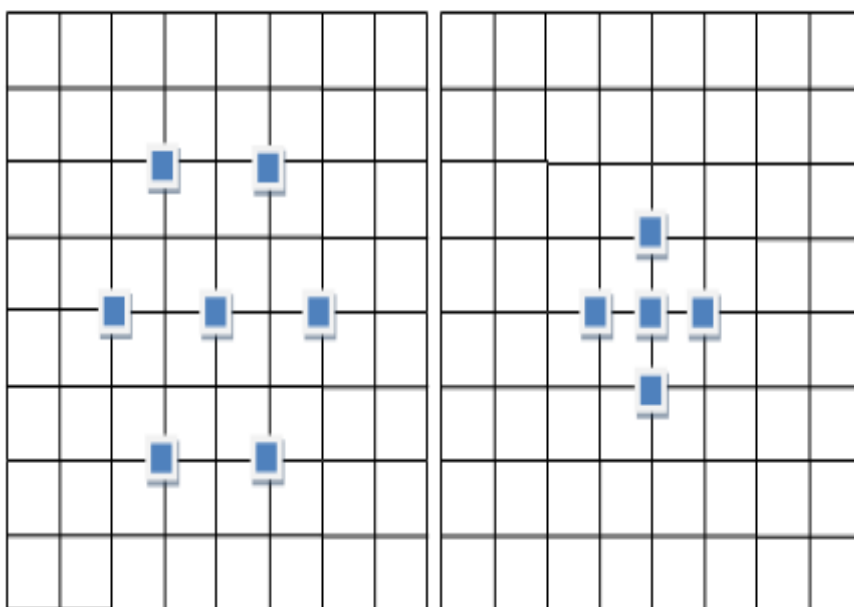


شکل ۳-۴۲ یک نمونه مورد استفاده برای الگوریتم کراس

ج) الگوریتم جستجو شش وجهی (Hexagon-Based Search Pattern)

در این الگوریتم از الگویی شش وجهی (Hexagon) برای جستجو استفاده می کنیم و دارای دو فاز می باشد. در فاز اول از الگوی بزرگتری استفاده می شود. این الگو شامل ۷ نقطه می باشد. اگر MDB در مرکز الگو واقع شود به مرحله دوم جستجو می رویم که در آن از الگو کوچکتری که شامل ۴ نقطه است استفاده می کنیم. توجه کنید زمانی که الگو شش وجهی در حال حرکت در جهتی است که مقدار اختلاف بین بلوک ها کاهش می یابد فقط ۳ نقطه جدید به محاسبات اضافه می شود و لازم نیست همه ۷ نقطه را مورد ارزیابی قرار دهیم. زیرا ۴ نقطه دیگر قبلا محاسبه شده اند. این الگوریتم را می توانیم به صورت زیر خلاصه کنیم [۱۰]:

۱. الگوریتم در اولین گام از الگو بزرگ که دارای ۷ نقطه می باشد استفاده می کند. در این مرحله اگر MDB در مرکز الگو پیدا شد به مرحله پایان برو در غیر این صورت به گام جستجو برو.
۲. در این مرحله که مرحله جستجو می باشد الگو را در موقعیت مناسب قرار بدهیم تا مرکز الگو در موقعیتی باشد که دارای کمترین MDB در مرحله قبل بوده، در این مرحله ۳ نقطه جدیدی که بدست می آید مقدار آن را محاسبه می کنیم. اگر این مقدار مینیمم در مرکز شکل باشد می توانیم به مرحله پایانی برویم در غیر این صورت این گام را ادامه می دهیم.
۳. در این مرحله از الگو بزرگ به الگو کوچک تغییر وضعیت می دهیم در الگو جدید به ۴ نقطه برای بررسی نیاز داریم MDB که در این وضعیت بدست می آید بردار حرکت را نشان می دهد.

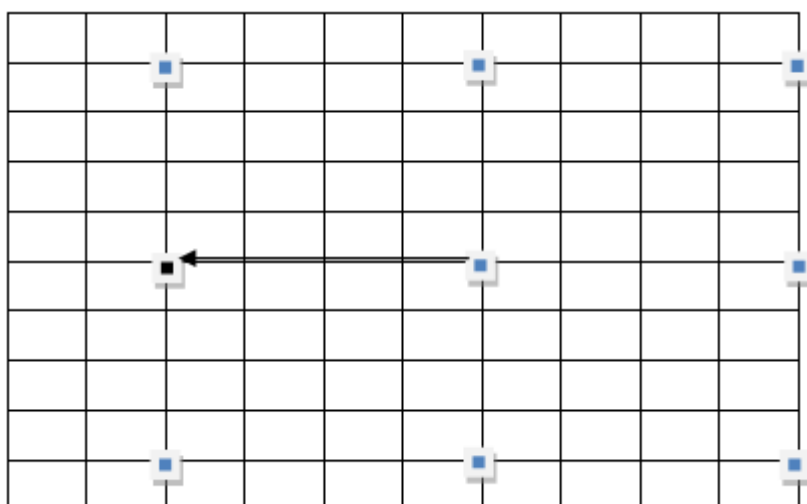


شکل ۳-۴ الگو مورد استفاده در الگوریتم HEXBS

چ) الگوریتم جستجو D-Logarithm ۲

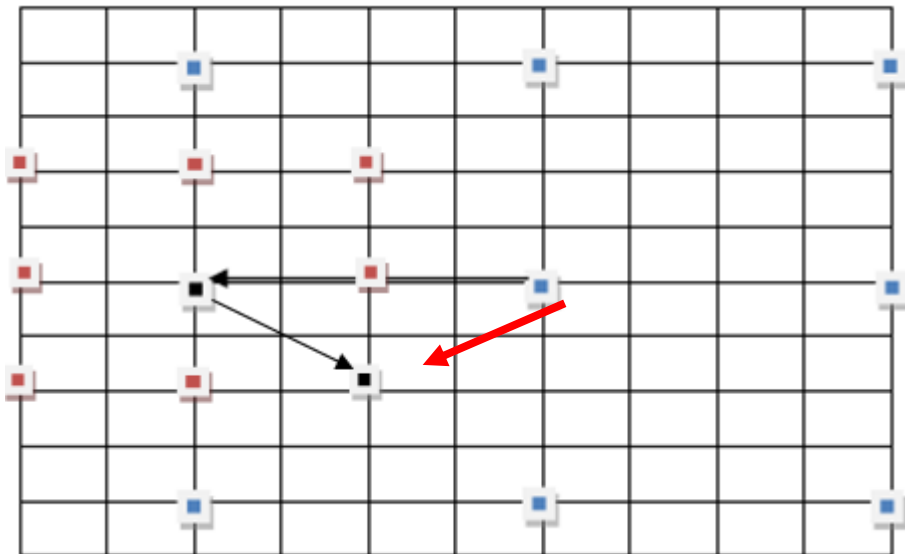
در این روش در ابتدا ۹ نقطه را مورد بررسی قرار می دهیم. در مرحله بعد محاسبات را بر روی مربعی به مرکز نقطه بهترین انطباق ادامه پیدا می کند و در هر مرحله اندازه پنجره کوچک تر می شود تا زمانی که محاسبات به پایان برسد شکل ۳-۴۴ مثالی را نشان می دهد.

گام اول: اندازه گام ها را برابر ۴ فرض کنید. تمام بلوک ها یی را که بر روی نقاط آبی هستند را مورد بررسی قرار می دهیم تا بهترین بلوک را پیدا کنیم فرض کنید نقطه سیاه رنگ بهترین بلوک را نشان دهد. پس بردار حرکت به صورت زیر خواهد بود به عبارتی بلوک از موقعیت مربعی مرکزی به موقعیت بلوک مشکی منتقل شده است.



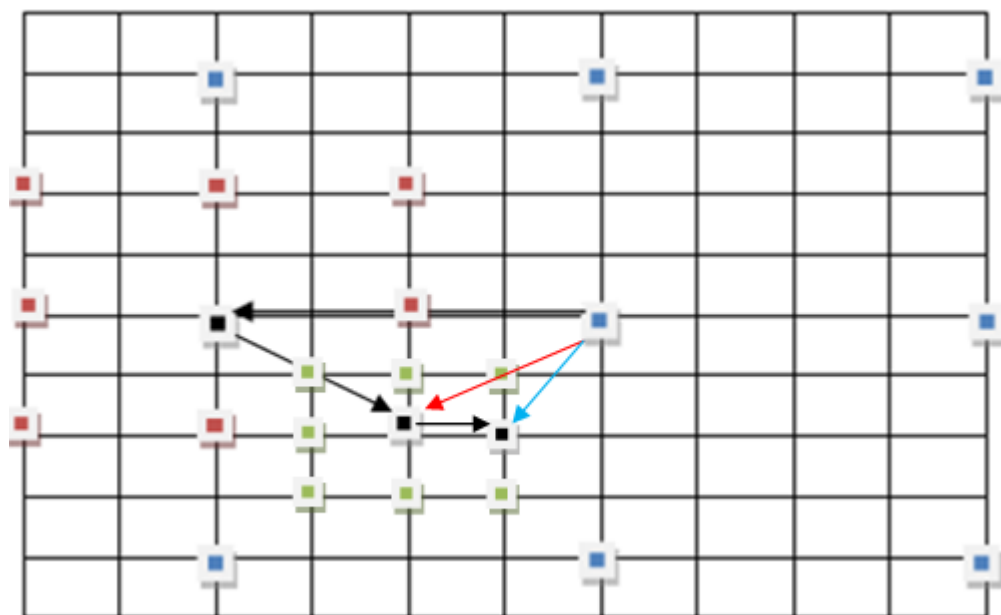
شکل ۳-۴۴ مرحله اول الگوریتم D-Logarithm ۲

گام دوم: اندازه گام ها برابر ۲ فرض کنید. تمام بلوک هایی را که بر روی نقاط قهوه ای هستند را مورد بررسی قرار می دهیم تا بهترین بلوک را پیدا کنیم فرض کنید نقطه سیاه رنگ بهترین بلوک را نشان دهد. پس بردار حرکت نهایی برابر بردار قرمز رنگ هست به عبارتی بلوک از موقعیت مربعی مرکزی به موقعیت بلوک مشکی منتقل شده است. (شکل ۳-۴۵)



شکل ۳-۴۵ مرحله دوم جستجو ۲D-Logarithm

گام سوم: اندازه گام ها را برابر ۱ فرض کنید. تمام بلوک هایی را که بر روی نقاط سبز هستند را مورد بررسی قرار می دهیم تا بهترین بلوک را پیدا کنیم فرض کنید نقطه سیاه رنگ بهترین بلوک را نشان دهد. پس بردار حرکت نهایی برابر بردار آبی رنگ هست به عبارتی بلوک از موقعیت مربعی مرکزی به موقعیت بلوک مشکی منتقل شده است. (شکل ۳-۴۶)



شکل ۳-۴۶ مرحله دوم جستجو

۳-۴-۲- مقایسه الگوریتم های مختلف تخمین بلوک (از نتایج پایان نامه)

در بین روش های نیمه بهینه انطباق بلوک، TSS رایجترین روشی است که مورد استفاده قرار می-گیرد در این روش در گام اول ۹ نقطه اطراف بلوک مورد ارزیابی قرار می گیرد. از بین این بلوک ها بهترین نقطه انتخاب می شود در مرحله بعد ۸ نقطه اطراف این نقطه جستجو می شود و از بین آنها بهترین را برمی گزینیم و جستجو را بر روی نقاط همسایه آن انجام می دهیم تا بهترین بلوک انتخاب شود و به این طریق تعداد نقاط مورد جستجو برابر $9+8+8=25$ می شود.

الگوریتم TSS نسبتا ناکارآمد است زیرا در گام اول تعداد زیادی نقطه مورد ارزیابی قرار می دهد، بررسی هایی که بر روی چند نمونه از ویدئو ها انجام شده نشان می دهد که بالاتر از ۸۰٪ مواقع بلوک ها هدف در درون مربعی به مرکز بلوک جاری و اندازه ۳ می باشد. در صورتی که ویدئو دارای تغییرات کمتری بیشتری باشد این مقدار به ۹۹٪ نیز می رسد [۱۳] به همین دلیل روش جدیدی NTSS ارائه شد در این روش اگر صحنه دارای ثبات باشد ۱۷ نقطه مورد ارزیابی قرار می گیرد در حالی که برای

بلوک ها با تحرک بیشتر این حرکت به ۲۰ تا ۲۲ مورد می رسد. کارایی این روش نسبت به TSS، ۱۸٪ در زمان صرفه جویی می گردد اما روش TSS دقت بیشتری است.

در [۱۰] روش جدیدی (Diamond Search) برای انطباق به نام معرفی شد در این روش به جای بررسی یک شبکه ۹*۹ یک شبکه ۵*۵ مورد بررسی قرار می گیرد این روش بر خلاف روش TSS به جای ۳ گام از ۴ گام استفاده می کند. با مطالعاتی که انجام داده شد در این روش در بهترین حالت ۱۷ مقایسه و برای بدترین حالت از ۲۷ نقطه بررسی استفاده می کند. در بررسی ها نشان داده شده این روش بطور معمول ۶ مقایسه کمتر انجام می دهد و در بدترین حالت ها این تعداد مقایسه ها، به ۲ مقایسه کاهش می یابد.

در الگوریتم UCBD تعداد نقاطی که مورد بررسی قرار گیرد متغیر می باشد دومین نکته ای که لازم است در مورد این الگوریتم توجه شود این است که با توجه به نقطه که بهترین انطباق در آن صورت می گیرد تعداد نقاط جدیدی که مورد مطالعه قرار می گیرد تغییر می کند در صورتی که این نقطه بر روی لبه باشد تعداد نقاط جدید برابر ۳ خواهد شد. اگر نقطه بر روی راس ها واقع شود تعداد نقاط جدید برابر ۵ خواهد شد. بنابراین در اولین گام لازم است تا ۹ نقطه و گام نهایی ۴ نقطه را بررسی کنیم. اگر تعداد مراحل که نقطه راس انتخاب می شوند برابر n_v و تعداد نقاط که بر روی لبه n_e باشد آنگاه کل تعداد نقاط مورد بررسی برابر $N = 9 + 5n_v + 3n_e + 4$ خواهد شد. اگر بطور متوسط تعداد نقاط مورد جستجو را برابر ۴ و تعداد گام ها را برابر ۳ بگیریم کل تعداد مقایسه ها ۲۱ خواهد شد.

در ۲D-logarithm اندازه پنجره اولیه می تواند اندازه های مختلفی داشته باشد اگر اندازه پنجره اولیه برابر W باشد که معمولاً W توانی از ۲ می باشد آنگاه تعداد مراحل جستجو برابر $Number\ of\ step = \log_2^W$ خواهد شد. در گام اول تعداد نقطه مورد بررسی ۹ می باشد و در گام

های دیگر، تعداد نقطه جستجو برابر ۸ می باشد آنگاه کل تعداد نقطه مورد جستجو برابر عبارت زیر خواهد شد.

$$N = (\text{number of steps} - 1) * 8 + 9$$

الگوریتم دیگر CSA می باشد این الگوریتم مانند الگوریتم فوق می باشد زیرا در هر مرحله اندازه پنجره نصف حالت قبل می شود با این تفاوت که در این الگوریتم تعداد نقاط مورد ارزیابی در گام اول برابر ۵ می شود و تعداد نقاط در گام های بعدی برابر ۴ می باشد پس کل تعداد نقاطی که مورد بررسی قرار می گیرد برابر $N = (\text{Numberofstep} - 1) * 4 + 5$ می باشد.

در الگوریتم HEXBS برای گام اول ۸ نقطه مورد بررسی قرار می گیرد و در گام های میانی تعداد نقطه جدید مورد ارزیابی برابر ۳ نقطه می باشد و در نهایت تعداد نقطه استفاده شده برابر ۴ می باشد پس اگر تعداد گام ها برابر Numberofstep فرض کنیم آنگاه کل تعداد مراحل در این الگوریتم برابر $N = (\text{Numberofstep} - 2) * 3 + 8 + 4$ خواهد شد.

فصل چهارم

مدل سازی پس زمینه

۴-۱- مقدمه

همانطور که در فصل دوم ذکر شد که افزودنی را می توان به دو دسته افزودنی درون فریم و افزودنی بین فریم تقسیم نمود. در افزودنی بین فریمی حجم عمده یک فریم را پس زمینه تشکیل می دهد که در فریم های متوالی مشترک می باشد و آنچه تغییر می کند اشیا و تغییرات جزئی در پس زمینه هستند. (که قابل توجه برای چشم انسان نمی باشند مانند امواج دریا یا لرزش های آرام شاخه و برگ درختان) هستند، پس اگر بتوان پس زمینه را مدل نمود، می توان از ذخیره قسمت های مشترک فریم ها صرف نظر کرد و به این طریق به حجم بالایی از فشرده سازی در ویدیو دست یافت.

از کاربرد های دیگر مدل سازی پس زمینه، کاربردهایی مانند مانیتورینگ ترافیک یا سیستم های نظارتی است. در این سیستم ها هدف استخراج اشیا از فریم های ویدیویی است (به عبارتی هدف، جداکردن اشیا از پس زمینه می باشد). یکی از روش های رایج برای این منظور background subtraction می باشد. در این روش، ابتدا مدلی از پس زمینه به دست می آوریم. در گام دوم تفاضل پس زمینه و فریم جاری را محاسبه می کنیم. اگر این اختلاف قابل توجه باشد، می توان آن پیکسل را به عنوان اشیا در نظر گرفت، در غیر این صورت پیکسل مورد نظر جز پس زمینه می باشد.

شایان ذکر است اختلاف بین سایه اشیا و پس زمینه قابل توجه می باشد، بنابراین سایه اشیا نیز به عنوان شی در نظر گرفته می شود و در نتیجه اشیا استخراج شده دارای مشکلاتی می باشند (شی و سایه آن به عنوان دو شی مجزا در نظر گرفته می شوند). از آنجایی که نتایج این مرحله برای پردازش لایه های بالاتر در سیستم های نظارتی (مثلا سیستمی که از شی استخراج شده برای شناسایی افراد استفاده می کند) مورد استفاده قرار می گیرد، خطای این مرحله در کل سیستم منتشر می شود و صحت عملکرد آن تحت تاثیر قرار می دهد. روش های متفاوتی به منظور تشخیص سایه از اشیا وجود دارد؛ در ادامه فصل تلاش می شود تا به معرفی این روش ها بپردازیم. معمولا اشیا استخراج شده دارای

نویزهایی می باشند، بنابراین با استفاده از تکنیک های کاهش نویز اثر آنها را کاهش می دهند، در قسمت انتهایی فصل به معرفی این تکنیک ها پرداخته می شود.

۴-۲- مدل سازی پس زمینه

روش های مدل کردن پس زمینه را می توان به دو گروه تقسیم کرد. در گروه اول مدل پس زمینه، در طول زمان ثابت می باشند. این مدل ها برای محیط های داخلی با شرایط نوری کنترل شده مناسب هستند. مزیت این روش ها سادگی، سرعت بالا و نیاز کم به حافظه می باشد. اما این روش ها دارای دقت پایینی بوده و در سیستم های عملیاتی، منجر به خطا های غیر قابل تحملی می شوند. برای مقابله با این مشکل از متدهایی استفاده می شود که در طول زمان، مدل تغییر می کند تا خطا کاهش یابد.

یکی از ساده ترین روش ها برای مدل کردن پس زمینه این است که، زمانیکه در صحنه هیچ شیئی وجود ندارد از صحنه تصویر گرفته، و آن را به عنوان پس زمینه در نظر بگیریم. برای تشخیص اشیا از پس زمینه، اختلاف فریم جاری را با تصویر گرفته شده محاسبه می کنیم اگر مقدار این اختلاف بیشتر از حد آستانه باشد، می توان آن را به عنوان اشیا و در غیر اینصورت به عنوان پس زمینه حساب کرد. این روش برای محیط های کنترل شده مناسب می باشد؛ ولی برای محیط های بیرونی که شرایط نوری آن در حال تغییر است، دارای خطا زیادی خواهد بود. در محیط های بیرونی یافتن صحنه های بدون شی مشکل است، به همین دلیل لازم است روش هایی ارائه کنیم تا با توجه به تغییرات محیطی انطباق یابد و از طرفی حضور اشیا نتواند در تعیین مدل پس زمینه، اثر جدی داشته باشد.

۴-۲-۱- فیلتر میانه تقریبی

در این روش برای تقریب مقدار پس زمینه، از مقادیری که یک پیکسل در طول زمان به خود می گیرد، استفاده می شود. در این روش فیلتر میانه را بر روی بافری از تصاویر ویدیویی اعمال می کنیم.

از آنجایی که محاسبه میانه بر روی دنباله ای از فریم ها عملیاتی زمان بر می باشد در [۱۵] روشی برای تعیین تقریبی میانه ارائه شده که در ادامه به تشریح آن می پردازیم.

فرض کنید x_t نشان دهنده مقدار پیکسل در زمان جاری بوده و m_t مقدار میانه را تا زمان جاری نشان دهد، برای بروز کردن مقدار میانه با توجه به این مقدار جدید، از رابطه زیر استفاده می شود.

$$m_{t+1} = \begin{cases} m_t + \alpha & x_t > m_t \\ m_t - \alpha & x_t < m_t \\ m_t & x_t = m_t \end{cases} \quad (1-4)$$

پارامتر α نرخ بروزرسانی پس زمینه را نشان می دهد، مقدار بزرگتر نشان دهنده تغییرات شدیدتر در پس زمینه و مقدار کوچکتر نشان دهنده تغییرات کمتر می باشد. در صورتی که مقدار $\alpha = 1$ باشد و حد آستانه را برابر ۱۵ در نظر بگیریم، (مقدار حد آستانه عددی است که اگر مقدار اختلاف بین پیکسل و مقدار میانه کمتر از آن باشد مقدار پیکسل را جزء پس زمینه در نظر می گیریم در غیر اینصورت این مقدار نشان دهنده پیکسلی است که اشیا را نشان می دهد.) به موجب آن اگر شی به مدت ۱۵ فریم در صحنه ثابت بماند، آنگاه شی جز پس زمینه خواهد شد و اگر شی شروع به حرکت کند، ۱۵ فریم طول خواهد کشید تا مقدار میانه اصلاح شود.

در صورتی که مقدار به درستی انتخاب نشود، می توان سایه ای از شی را در پس زمینه یافت، به همین دلیل در این روش لازم است تا مقدار این پارامتر به درستی انتخاب شود. در شرایط محیطی که پس زمینه دارای تغییر زیادی است مقدار این پارامتر بزرگ در نظر گرفته می شود. اگر مقدار تغییرات کم بوده یا اشیا با سرعت کمی در صحنه حرکت کنند، بهتر است مقدار آن کوچک انتخاب شود. در ادامه نحوه به دست آمدن رابطه (۱-۴) توضیح داده می شود. فرض کنید مقادیری را که یک پیکسل در طول زمان دارد برابر $\{x_1, x_2, \dots, x_N\}$ و تابع $e(m)$ با رابطه (۲-۴) نشان داده شود.

$$e(m) = \sum_{i=1}^N |x_i - m_i| \quad (2-4)$$

مینیمم مقدار $e(m)$ زمانی بدست می آید که از مقدار میانه در مکان m_t استفاده کنیم. با استفاده از این رابطه و مشتق می توانیم مقدار میانه جدید را از رابطه (۳-۴) بدست آوریم.

$$m_{t+1} = m_t - \alpha \frac{d \sum |x_i - m_t|}{dm_t} \quad (3-4)$$

در صورتی که به جای N نمونه از یک نمونه استفاده کنیم می توان این رابطه را به صورت (۴-۴) خلاصه کرد.

$$m_{t+1} = m_t - \alpha \frac{d |x_t - m_t|}{dm_t} \quad (4-4)$$

از مزایای این روش می توان به سادگی روش، سرعت و ایمن بودن آن در مقابل نویز ضربه اشاره کرد. اما این روش، نیازمند تنظیم پارامتر به صورت درست می باشد. در صورتی که این پارامتر به درستی انتخاب نشود، در نتایج تولید شده، خطا به وجود می آید.

پیاده سازی و ارزیابی (از نتایج پایان نامه)

روش مذکور بر روی دو نوع ویدیو (در محیط داخلی و خارجی) اعمال شده است که شکل ۴-۱ نتیجه خروجی را بر روی محیط بیرونی نشان می دهد. سطر اول، پس زمینه مدل شده در سه فریم دلخواه، توسط متد میانه نشان داده شده است. سطر دوم و سوم به ترتیب فریم واقعی و شی مطلوب (شی متحرک بدون پس زمینه و سایه) و در سطر نهایی اشیا استخراج شده به صورت خودکار را نشان می دهد.

همانطوری که قبلا بیان شد، برای استخراج خودکار اشیا، اختلاف بین مدل پس زمینه و فریم را بدست می آوریم. پیکسل هایی که در آن اختلاف بدست آمده، اگر از یک حد آستانه بیشتر باشد، به عنوان پیکسل های شی در نظر گرفته می شود. در نتایج بدست آمده، دو نوع خطا وجود دارد اولین نوع این خطا، خطای $\text{false negative (FN)}$ است، این خطا برابر درصد پیکسل هایی است که جز شی

می باشد اما به عنوان پس زمینه در نظر گرفته شده اند. برای محاسبه آن از رابطه (۴-۵) استفاده می شود. این خطا زمانی به وجود می آید که، سطح خاکستری پیکسل شی، در محدوده مورد انتظار مدل پس زمینه، واقع می شود. به همین دلیل لازم است تا علاوه بر سطح خاکستری از ویژگی دیگر نیز استفاده شود.

دومین نوع خطا false positive (FP) می باشد که برای محاسبه آن از رابطه (۴-۶) استفاده می شود. این خطا درصد پیکسل هایی از پس زمینه را نشان می دهد که به عنوان پیکسل های شی در نظر گرفته شده است. این خطا به دلیل عواملی مانند نویز ضربه، حرکت شاخه ها و لرزش برگ ها و سایه اشیا می باشد. (این خطا در تصاویر جدول ۴-۱ قابل مشاهده می باشد). که N_{obR} تعداد پیکسل هایی که به یک شی تعلق دارد و N_{obE} تعداد پیکسل هایی که مدل به عنوان شی در نظر گرفته است. که N_{bkR} تعداد پیکسل هایی که به پس زمینه تعلق دارد و N_{bkE} تعداد پیکسل هایی که مدل به عنوان پس زمینه در نظر گرفته است.

$$false\ negative = \frac{N_{obR} - N_{obE}}{N_{obR}} \quad (۴-۵)$$

$$false\ positive = \frac{N_{bkR} - N_{bkE}}{N_{bkR}} \quad (۴-۶)$$



پ) پس زمینه بدست آمده در

فریم ۲۵



ب) پس زمینه بدست آمده در

فریم ۵۰

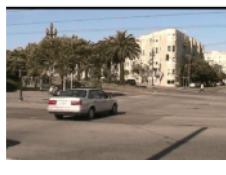


الف) پس زمینه بدست آمده

در فریم ۷۵



ج) فریم ۲۵



ث) فریم ۵۰



ت) فریم ۷۵



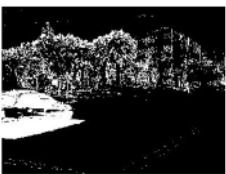
خ) شی مطلوب در فریم ۲۵



ح) شی مطلوب در فریم ۵۰

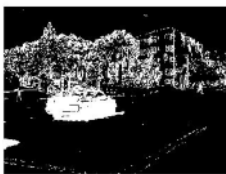


چ) شی مطلوب در فریم ۷۵



ذ) اشیا استخراج شده الگوریتم

توسط میانه در فریم ۲۵



د) اشیا استخراج شده توسط

الگوریتم میانه در فریم ۵۰

شکل ۴-۱ نمونه ای از بکارگیری روش میانه برای مدل سازی پس زمینه جهت تشخیص اشیا $\alpha=0,1$ ، $T=15$ و

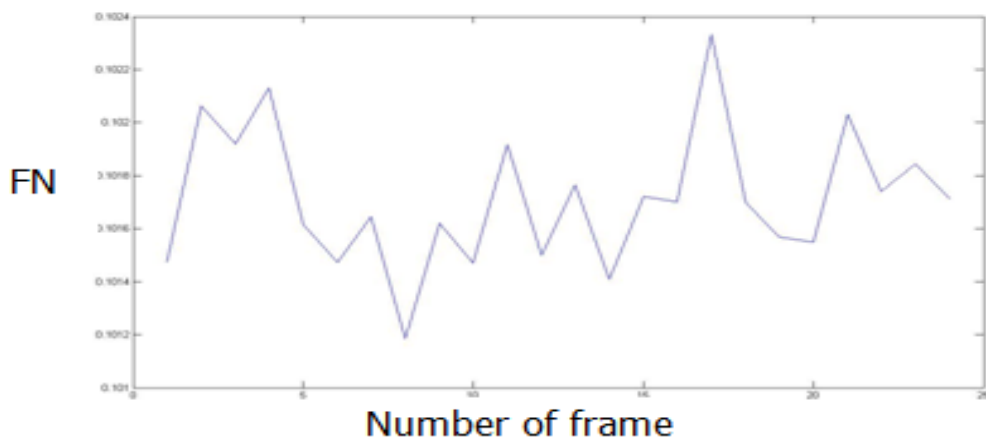
تعداد فریم های مورد استفاده ۱۵۰ می باشد زمان مصرفی برابر ۱۲,۰۴۷۲۹۶ ثانیه می باشد.

با دو پارامتر می توان میزان خطا را برای استخراج شی از پس زمینه حساب نمود. برای جلوگیری از بروز این خطاها سه راهکار وجود دارد:

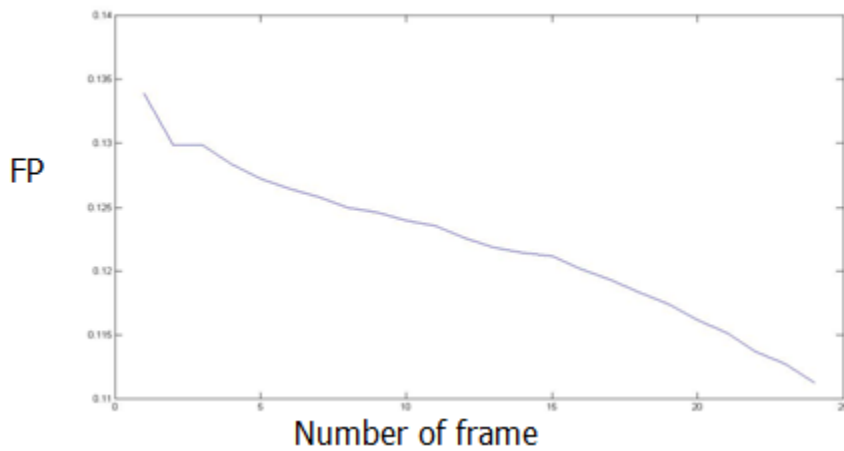
۱. اعمال فیلترها جهت حذف نویز
۲. استفاده از ویژگی های بیشتر برای تشخیص اشیا
۳. تنظیم دقیق تر پارامترها برای بالا بردن دقت روش.

مورد اخیر را برای این الگوریتم مورد بررسی قرار دادیم. این الگوریتم دارای ۳ پارامتر w, α و اندازه حدآستانه و تعداد فریم های مورد استفاده، می باشد. در ادامه اثر هر کدام از این پارامترها را بر روی خروجی نهایی بررسی می کنیم.

اندازه w اثری بر روی خطای FN ندارد (کمترین مقدار خطا $0,1012$ و بیشترین خطا برابر $0,1024$) زیرا علت این خطا به دلیل نزدیکی سطح خاکستری شی و مدل پس زمینه می باشد و برای رفع آن لازم است از ویژگی های دیگر علاوه بر این سطح خاکستری، استفاده کنیم. از طرفی هر چه اندازه w بزرگتر شود میزان خطای FP کاهش می یابد، زیرا مدل پایدارتر می شود.

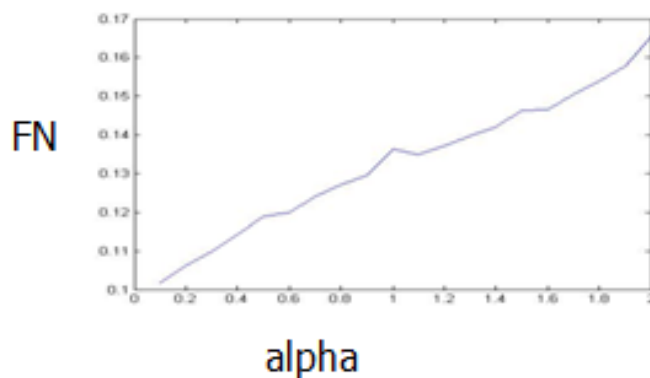


شکل ۲-۴-۲ خطای false negative این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها تعداد فریم های استفاده شده برای مدل سازی پس زمینه را نشان می دهد و محور عمودی میزان خطا را نشان می دهد.

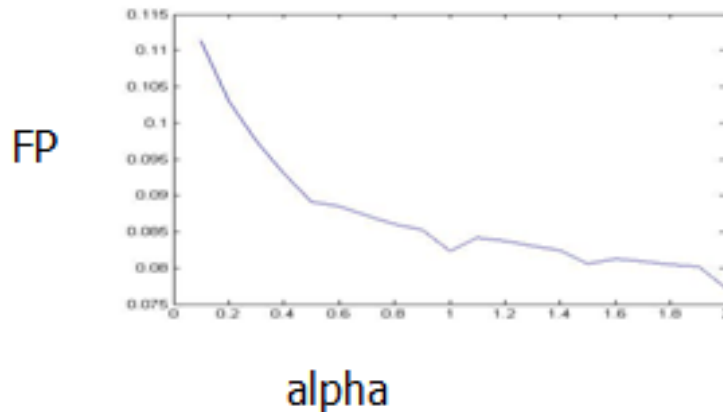


شکل ۳-۴ خطای *false positive* این خطا درصد پیکسل هایی را نشان می دهد که جز به شی نبوده اند اما محاسبه شده اند. محور x ها تعداد فریم های استفاده شده برای مدل سازی پس زمینه را نشان می دهد و محور عمودی میزان خطا را نشان می دهد.

در نمودار های ۴-۴ و ۳-۴ و ۴-۴ اثر α روی خطاهای مذکور، نشان داده شده است؛ هر چه اندازه این پارامتر بیشتر باشد تغییرات مدل زیاد می شود، به همین دلیل مناسب ترین مقدار برای پارامتر α برابر مقدار ۱ می باشد.

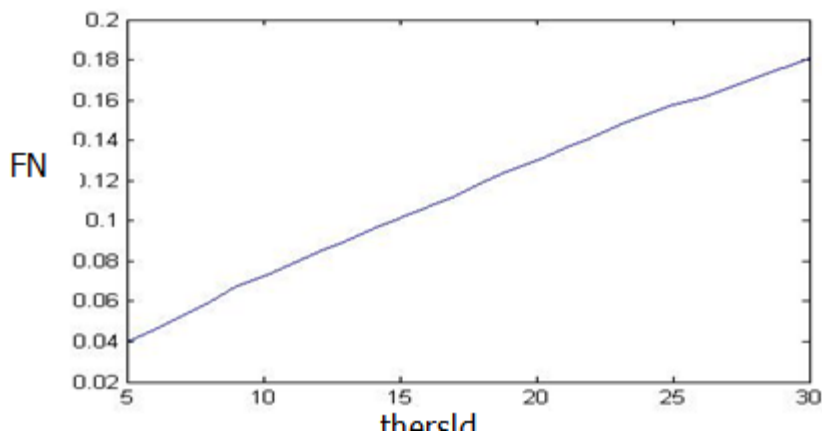


شکل ۴-۴ خطای *false negative* این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها میزان α و محور عمودی میزان خطا FN را نشان می دهد.

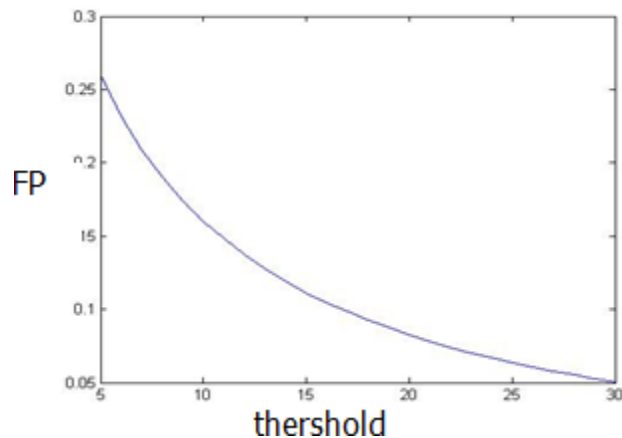


شکل ۴-۵ خطای false positive این خطا درصد پیکسل هایی را نشان می دهد که جز به شی نبوده اند اما محاسبه شده اند. محور x میزان α و محور عمودی میزان خطای FP را نشان می دهد.

معیار نهایی که مورد ارزیابی قرار گرفت، اندازه حد آستانه می باشد. همانطور که در شکل ۴-۷ ملاحظه می شود، هر چه اندازه حد آستانه بیشتر می شود، میزان خطای FP کاهش می یابد؛ زیرا پیکسل هایی که بر اثر لرزش در صحنه به وجود می آید، حذف می شوند. از طرفی میزان خطای FN نیز افزایش می یابد، زیرا تعدادی از پیکسل های متعلق به شی، حذف می شود. نتیجه این آزمایش ها در شکل ۴-۴ و ۴-۷ آمده است.



شکل ۴-۶ خطای false negative این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر گرفته شده است. محور x ها میزان حد آستانه و محور عمودی میزان خطای FN را نشان می دهد.



شکل ۷-۴ خطای *false positive* این خطا برابر درصد پیکسل هایی از شی که به عنوان پس زمینه در نظر

گرفته شده است. محور x ها میزان حدآستانه و محور عمودی میزان خطا را نشان می دهد.

نتایج تحلیل مدل برای محیط درونی در شکل ۸-۴ نشان داده شده است. سطر اول مدل استخراج شده برای پس زمینه را در چند فریم دلخواه، نشان می دهد. در این مدل، شی نیز به عنوان پس زمینه محاسبه شده است، زیرا سرعت شی کم می باشد و این متد نمی تواند به درستی آن را از درون مدل جدا کند. سطر دوم اشیا استخراج شده را به صورت خودکار نشان می دهد. همانطور که ملاحظه می شود در این روش میزان خطای FP نسبت به محیط بیرونی کمتر می باشد و می توان آنها را با استفاده از فیلترها حذف کرد.

در شکل های ۹-۴ تا ۱۲-۴ اثر پارامتر α و حدآستانه را بر خطای های FP و FN نشان می دهد با افزایش این دو پارامتر خطا FP را در محیط خارجی کاهش می دهد اما باعث افزایش خطا FN می شود.



پ) مدل استخراج شده در فریم

ب) مدل استخراج شده در فریم

الف) مدل استخراج شده در

۵۰

۷۵

فریم ۱۰۰



ج) شی استخراج شده در فریم

ث) شی استخراج شده در فریم

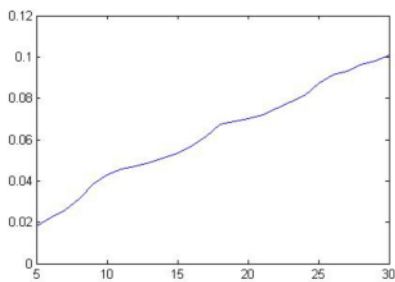
ت) شی استخراج شده در فریم

۵۰

۷۵

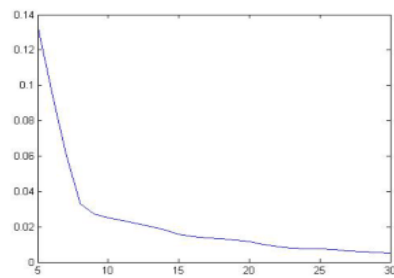
۱۰۰

شکل ۴-۸ نتایج الگوریتم در محیط درونی با $\alpha=0,1$, $T=15$ و تعداد فریم های مورد استفاده ۱۵۰ می باشد. سطر اول مدل استخراج شده برای پس زمینه را در چند فریم دلخواه، نشان می دهد. در این مدل، شی نیز به عنوان پس زمینه محاسبه شده است، زیرا سرعت شی کم می باشد و این متد نمی تواند به درستی آن را از درون مدل جدا کند. سطر دوم اشیا استخراج شده را به صورت خودکار نشان می دهد.



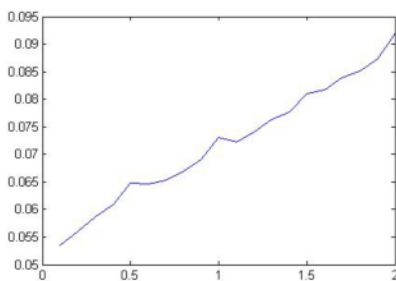
شکل ۴-۱۰ اثر حد آستانه در محیط خارجی بر

خطای FN

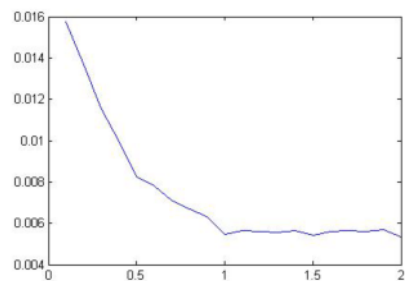


شکل ۴-۱۹ اثر حد آستانه در محیط خارجی بر خطای

FP



شکل ۴-۱۲ اثر alpha در محیط خارجی بر خطای FN



شکل ۴-۱۱ اثر alpha در محیط خارجی بر خطای FP

۴-۲-۲- الگوریتم میانگین برای مدل سازی پس زمینه

ساده ترین روش برای مدل سازی پس زمینه، اولین فریم ویدیو را به عنوان مدل پس زمینه در نظر می گیرد و هر فریم جدید را از پس زمینه کم می کند و با استفاده از مقدار حد آستانه تصمیم می گیرد که آیا پیکسل مربوطه، جز پس زمینه بوده یا پیش زمینه است. اما این الگوریتم بسیار خطا پذیر در مقابل تغییرات در پس زمینه می باشد، به همین دلیل به جای استفاده از یک فریم، از میانگین چندین فریم استفاده می شود. که در ادامه چند نمونه از این الگوریتم ها معرفی می شود. این الگوریتم ها در مرجع [۷] آمده است.

تعاریف و نمادها

۱. I: فریم جاری را نشان می دهد و به صورت یک ماتریس $M \times N$ قابل نمایش می باشد.

۲. B: پس زمینه را نشان داده و به صورت یک ماتریس $M*N$ قابل نمایش می باشد.
۳. $I_{x,y}$: مقدار پیکسل سطر x و ستون y ماتریس I، را نشان می دهد.
۴. $B_{x,y}$: مقدار پیکسل سطر x و ستون y ماتریس B، را نشان می دهد.
۵. I_t : فریم را در زمان t نمایش می دهد.
۶. Pixel process: مقدار یک پیکسل را در طول زمان نشان می دهد فرض کنید $I_{x,y,t}$ نشان دهنده مقدار پیکسل در سطر x و ستون y در زمان t باشد آنگاه Pixel process به صورت زیر می باشد.

$$[I_{x,y,0}, I_{x,y,1}, \dots, I_{x,y,t}]$$

که در آن $I_{x,y,0}$ مقدار پیکسل سطر x و ستون y در زمان 0، $I_{x,y,1}$ مقدار پیکسل سطر x و ستون y در زمان 1 و $I_{x,y,t}$ مقدار پیکسل سطر x و ستون y در زمان t می باشد.

مدل سازی پس زمینه مستقل از زمان (Time Invariant Background Model) TIBM

در این روش فرض می شود که پس زمینه در طول زمان ثابت و بدون تغییر است. با این فرض اولین فریم به عنوان پس زمینه در نظر گرفته می شود؛ به عبارتی $B_{x,y,t} = I_{x,y,0}$ می باشد. از معایب این روش این است که در مقابل نویز ایمن نبوده و در مقابل تغییرات نوری قابل اعتماد نیست. برای محیط هایی که شرایط آن به شدت قابل کنترل است مناسب بوده و به شدت وابسته به فریم اولیه می باشد.

مدل سازی پس زمینه متغیر با زمان (Time Background Model) TBM

ایراد اساسی روش فوق این است که نسبت به زمان ثابت است. برای بر طرف کردن این مشکل TBM پیشنهاد می کند. که مدل را به گونه ای طراحی کنیم که نسبت به تغییرات پس زمینه حساس بوده و به مرور زمان تغییر کند. این روش از معیاری جدیدی به نام AD (Absolute Deference) (قدر مطلق فریم جاری را از پس زمینه جاری نشان می دهد) استفاده می کند که رابطه آن در معادله (۴-۷) آمده است.

$$AD_{x,y,t} = |B_{x,y,t} - I_{x,y,t}| \quad (7-4)$$

اگر مقدار $AD_{x,y,t}$ از یک حد آستانه بیشتر بود مقداری که پیکسل x,y نشان می دهد جز پیش زمینه می باشد. به همین دلیل مقدار $B_{x,y,t}$ برابر مقدار قبلی یعنی $B_{x,y,t-1}$ می باشد. ولی اگر مقدار این اختلاف کمتر از حد آستانه باشد، نشان دهنده این است که مقدار پس زمینه در این پیکسل تغییرات جزئی پیدا کرده، پس مقدار $B_{x,y,t}$ برابر $I_{x,y,t}$ می شود. تمام موارد یاد شده را می توان به صورت رابطه زیر خلاصه کرد.

$$B_{x,y,0} = I_{x,y,0} \quad (8-4)$$

مقدار پس زمینه در مرحله اول برابر فریم اولیه ویدیو می باشد و برای تعریف پس زمینه در زمان k می توان از رابطه زیر استفاده کرد.

$$B_{x,y,k} = \begin{cases} I_{x,y,k} & AD_{x,y,k} < Th \\ B_{x,y,k-1} & AD_{x,y,k} > Th \end{cases} \quad (9-4)$$

این روش به شدت وابسته به مقدار حد آستانه می باشد. مقدارهای بزرگ حد آستانه پیش زمینه را هم در پس زمینه مدل می کند به عبارتی مقدار پس زمینه متأثر از پیش زمینه می باشد. مقدارهای کوچک حد آستانه تغییرات را در پس زمینه مدل نکرده و نسبت به نویز نیز ایمن نمی باشد.

مدل سازی پس زمینه مبتنی بر میانگین LTABM (Long Term Average Background)

(Model)

روش TBM در مقابل نویز ایمن نیست؛ برای برطرف کردن این مشکل، در [7] پیشنهاد شده است که از مقدار میانگین فریم ها، در طول زمان استفاده نماییم. فرض کنید pixel process یک پیکسل دلخواه را داریم، می خواهیم بدانیم مقدار پس زمینه در موقعیت (x,y) در زمان جاری (t) چقدر است. برای محاسبه آن از رابطه زیر داریم:

$$B_{x,y,t} = \frac{1}{t} \sum_{r=0}^t I_{x,y,r} \quad (10-4)$$

به عبارتی مقدار پس زمینه در یک موقعیت، برابر میانگین مقادیر آن در طول زمان می باشد. برای این که بتوان مقدار فرمول فوق را به صورت بازگشتی محاسبه کرد از رابطه (4-11) که معادل رابطه فوق است استفاده می شود.

$$B_{x,y,t} = \left(1 - \frac{1}{t}\right) B_{x,y,t-1} + \frac{1}{t} I_{x,y,t} \quad (11-4)$$

در زمان های اولیه که تعداد کمی فریم برای محاسبه مدل استفاده می شود. سهم هر کدام از فریم ها در محاسبه پس زمینه زیاد می باشد و اگر فریمی دارای نویز باشد در محاسبه پس زمینه اثر خواهد گذاشت. هرچه زمان بگذرد از تعداد بیشتری فریم برای محاسبه مدل پس زمینه، استفاده می شود. مزیت این روش این است که اثر نویز را خنثی می کند؛ اما تغییراتی که در طی زمان در پس زمینه به وجود می آیند در مدل منعکس نمی شود.

مدلسازی پس زمینه مبتنی بر میانگین با طول ثابت MABM (Movie Average)

(Background Model)

برای برطرف کردن مشکل روش TBAM در روش MABM پیشنهاد می شود که برای محاسبه پس زمینه همواره از چند فریم اخیر - که تعداد آنها همواره ثابت است - استفاده نماییم. فرض کنید که مقدار پیکسل x,y در طی زمان برابر $[I_{x,y,-k}, \dots, I_{x,y,-2}, I_{x,y,-1}, I_{x,y,0}, I_{x,y,1}, \dots, I_{x,y,t}]$ باشد. زمان صفر زمان شروع محاسبات است و زمان های منفی مقدار پیکسل ها را پیش از زمان صفر نشان می دهد. فرض کنید برای محاسبه مدل بخواهیم از W فریم استفاده کنیم، آنگاه مقدار پس زمینه برای زمان t از رابطه (4-12) به دست می آید.

$$B_{x,y,t} = \frac{1}{W} \sum_{r=t-W+1}^t I_{x,y,r} \quad (12-4)$$

این روش وابسته به مقدار W می باشد. اگر w را عددی بزرگ در نظر بگیریم تغییرات در پس زمینه به کندی مدل خواهند شد. اگر w کوچک باشد نویزها در محاسبه پس زمینه بیشتر اثر خواهند کرد. از معایب این روش اینست که برای محاسبه رابطه فوق لازم است تا همواره w فریم اخیر را نگه داری کنیم.

مدل سازی پس زمینه مبتنی بر میانگین گوسی (RGABM Running Gaussian Average)

Background (Model)

یکی از معایب روش MABM این است که لازم است تا w فریم جاری را ذخیره کنیم. برای برطرف کردن این مشکل در مقالات [1],[3]، روش RGABM پیشنهاد شده است که در آن مقدار پس زمینه مقداری بین مقدار پس زمینه در زمان قبلی و مقدار آن در زمان جاری می باشد و می توان آن را با رابطه زیر نشان داد.

$$B_{x,y,k} = \begin{cases} (1-\alpha)B_{x,y,k-1} + \alpha I_{x,y,k} & , k > 1, \alpha \in [0,1] \\ I_{x,y}^0 & k = 0 \end{cases} \quad (13-4)$$

مقدار α میزان اثر فریم جاری در مدل را نشان می دهد.

مدل سازی پس زمینه مبتنی بر میانگین گوسی بهبود یافته (IRGABM Improved)

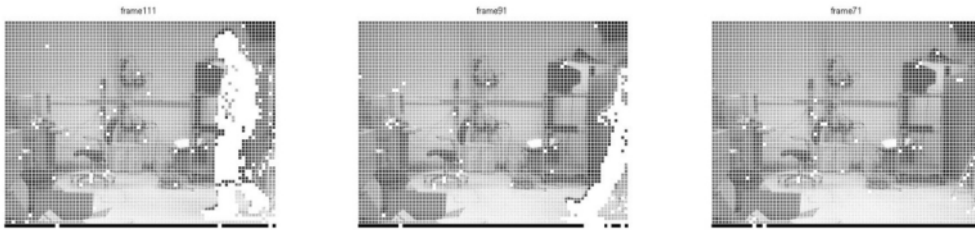
(Running Gaussian Average Background Model)

در این روش [4] برای بروز کردن پس زمینه از رابطه زیر استفاده می شود.

$$B_{x,y,k} = \begin{cases} (1-\alpha_1)B_{x,y,k-1} + \alpha_1 I_{x,y,k} & , if k > 0, AD_{x,y}^k < Th \\ (1-\alpha_2)B_{x,y,k-1} + \alpha_2 I_{x,y,k} & , if k > 0, AD_{x,y}^k > Th \\ I_{x,y}^0 & if k = 0 \end{cases} \quad (14-4)$$

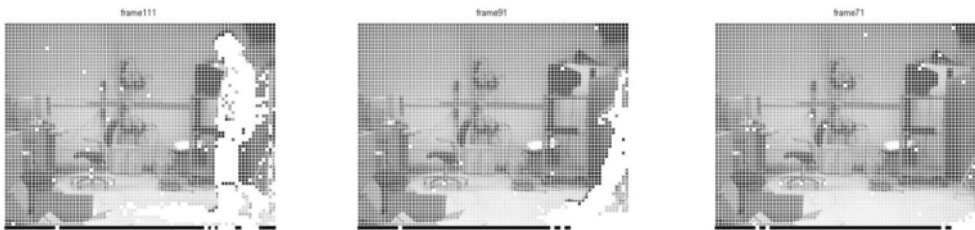
دلیل این کار این است که زمانیکه مقدار تفاضل یک پیکسل از مدل کم باشد، نشان دهنده آن است که پیکسل پس زمینه می باشد. به همین دلیل لازم است تا برای محاسبه پس زمینه از آن استفاده شود تا تاثیر بیشتری را بر روی مدل داشته باشد. زمانی که تفاضل از حد آستانه بیشتر باشد، احتمال این که پیکسل متعلق به پیش زمینه باشد، زیاد بوده و لازم است تا اثر کمتری را در محاسبه پس زمینه داشته باشد.

ITBM



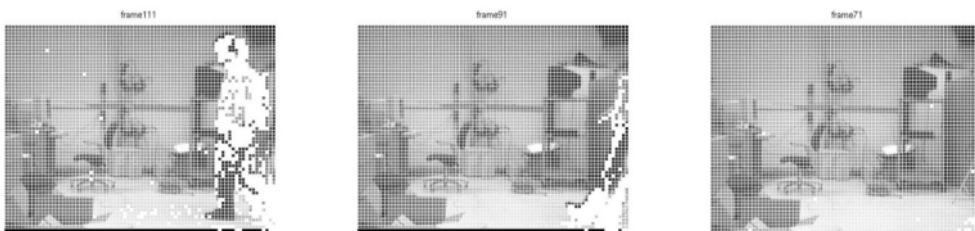
الف) خروجی الگوریتم ITBM برای فریم ۲ (ب) خروجی الگوریتم ITBM برای فریم ۲۰ (پ) خروجی الگوریتم ITBM برای فریم ۵۰

RGABM



ت) خروجی الگوریتم RGABM برای فریم ۲ (ث) خروجی الگوریتم RGABM برای فریم ۲۰ (ج) خروجی الگوریتم RGABM برای فریم ۵۰

IRGABM



چ) خروجی الگوریتم IRGABM برای فریم ۲ (ح) خروجی الگوریتم IRGABM برای فریم ۲۰ (خ) خروجی الگوریتم IRGABM برای فریم ۵۰



شکل ۴-۱۳ اشیا استخراج شده از محیط درون ساختمان با استفاده از الگوریتم های رهیافت میانگین (ادامه تصویر)

ارزیابی الگوریتم میانگین در مدل سازی پس زمینه (از نتایج پایان نامه)

اگر مقدار α کوچک انتخاب شود الگوریتم RGABM و LTABM دارای کارایی یکسانی خواهند بود. الگوریتم هایی که در این دسته قرار دارد الگوریتم های بسیار ساده ای بوده و حجم محاسباتی کمی دارند. به همین دلیل می توان از این روش ها برای محیط هایی که دارای تغییرات بسیار بالایی نمی باشند، استفاده کرد. روش هایی که در این رهیافت مطرح می شوند، معمولاً برای محیط های درون ساختمان مورد استفاده قرار می گیرند. از طرفی کارایی این روش ها به انتخاب درست پارامتر های آن بستگی دارد.

الگوریتم ITBM ساده ترین روش برای استخراج اشیا از پس زمینه می باشد. این الگوریتم برای محیط هایی که پس زمینه تغییرات شدیدی نداشته و در ضمن سرعت شی در فریم های متوالی قابل توجه باشد، استفاده می شود. همانطور که در شکل ۴-۱۳ مشاهده می کنیم، این روش خطای FP نسبتاً بالایی دارد، به عبارتی قسمتی از صحنه که به عنوان پس زمینه بوده، به عنوان شی در نظر گرفته شده است.

الگوریتم های RGABM و IRGABM دارای کارایی مشابهی می باشند. اما میزان خطای FN الگوریتم IRGABM بیشتر است. اگر چه این متد برای مدیریت شرایط پیچیده تری طراحی شده است، اما از

آنجایی که پارامتر های بیشتری دارد باید پارامتر های آن به درستی تنظیم شود. در شرایطی که پارامتر ها بیشتر می شود پیچیدگی سیستم نیز بیشتر می گردد، بنابراین اگر پارامتر ها به درستی انتخاب نشوند کارایی سیستم به شدت تحت تاثیر قرار می گیرد.

در الگوریتم MABM، پس زمینه به صورت حاصل جمع چند فریم در نظر گرفته می شود، به همین دلیل نتیجه کار به جای یک شی، حاصل جمع چند شی در فریم های متوالی است. اگر چه میزان خطای FN این روش و الگوریتم IRGABM نسبت به سایر الگوریتم ها پایین تر است ولی میزان خطای FP آن بالا می باشد.

۳-۲-۴ - الگوریتم PCA برای مدل سازی پس زمینه

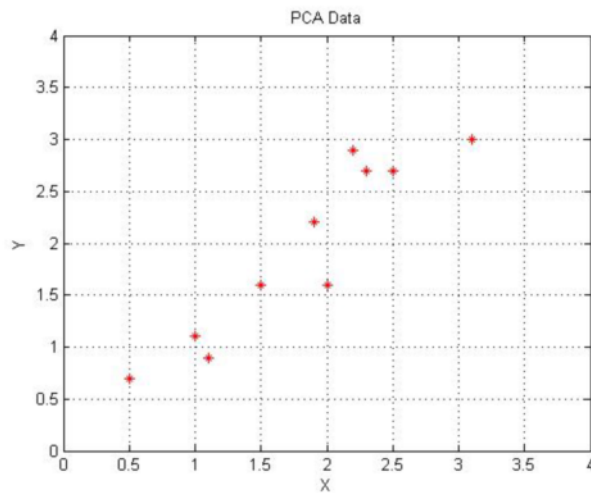
از آنجایی که استخراج الگو از بین داده ها با ابعاد بزرگ دشوار می باشد و از طرفی قادر به نمایش داده های بیشتر از سه بعد نمی باشیم، لازم است تا ابعاد داده ها را کاهش دهیم تا با استفاده از این داده های جدید، الگو ها را استخراج نماییم. PCA ابزاری مناسب برای کاهش ابعاد داده ها و در نتیجه ابزار قدرتمندی برای تحلیل و نمایش داده ها است [۱۴،۳۰].

در ادامه به بیان گام های الگوریتم با استفاده از یک مثال می پردازیم.

الف) کسب داده

در این گام داده ها در قالب مناسب فرمت بندی می شوند تا برای استفاده در مرحله بعدی مناسب باشد. به عنوان مثال، تعدادی داده دو بعدی تولید شده است (علت استفاده از داده های دو بعدی این است که می توان نتیجه کار را ترسیم کرد و درک بصری از مراحل کار داشته باشیم)؛ شکل ۴-۱۴ پراکندگی داده ها استفاده شده را نشان می دهد.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2.5 & 0.5 & 2.2 & 1.9 & 3.1 & 2.3 & 2.0 & 1.0 & 1.5 & 1.1 \\ 2.4 & 0.7 & 2.9 & 2.2 & 3.0 & 2.7 & 1.6 & 1.1 & 1.6 & 0.9 \end{bmatrix}$$



شکل ۴-۱۴ داده های اولیه مورد استفاده شده

(ب) تفریق میانگین از داده ها

برای اینکه PCA به درستی کار کند لازم است تا همه داده ها را از میانگین آن کم کنیم با این عمل داده ها اطراف صفر متمرکز می شود. در مثال فوق اگر میانگین داده ها در محور افقی و عمودی را به ترتیب \bar{x} , \bar{y} بنامیم و داده های ذکر شده را از \bar{x} , \bar{y} کم کنیم داده های قبلی به داده های ذیل تبدیل می شود.

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0.69 & -1.31 & 0.39 & 0.09 & 1.29 & 0.49 & 0.19 & -0.81 & -0.31 & -0.71 \\ 0.76 & -1.24 & 0.96 & 0.26 & 1.06 & 0.76 & -0.34 & -0.84 & -0.34 & -1.04 \end{bmatrix}$$

(پ) محاسبه کواریانس

در این گام ماتریس کواریانس داده های تولید شده، محاسبه می شود. از آنجایی که داده ها دو بعدی بوده، ماتریس کواریانس نیز 2×2 می شود. عناصر غیر قطری مثبت است که نشان می دهد داده های دو محور با هم نسبت مستقیم دارند، یعنی افزایش داده های x باعث افزایش داده های y می شود.

$$cov = \begin{pmatrix} 0.6166 & 0.6384 \\ 0.6384 & 0.7582 \end{pmatrix}$$

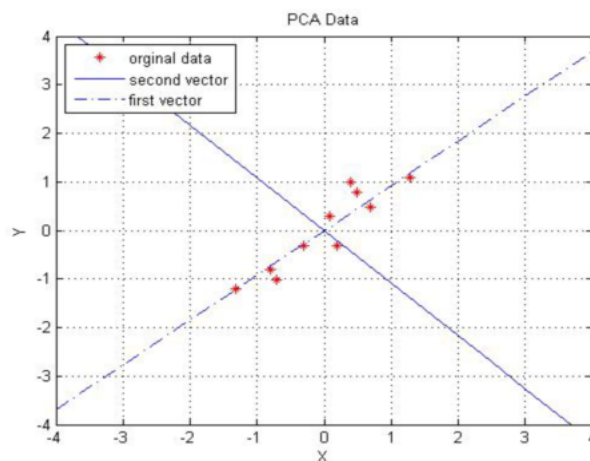
ت) محاسبه مقادیر و بردار های ویژه ماتریس کواریانس

از آنجایی که ماتریس کواریانس یک ماتریس مربعی می باشد می توان بردار ها و مقادیر ویژه آن را استخراج کرد. این داده های جدید دارای اطلاعات بسیار ارزشمندی می باشد. مقادیر ویژه و بردارهای ویژه ماتریس فوق در زیر نمایش داده شده است.

$$eigenValue = \begin{pmatrix} 0.0491 \\ 1.2840 \end{pmatrix}$$

$$eigenVector = \begin{pmatrix} -0.735179 & 0.677873 \\ 0.677873 & 0.735179 \end{pmatrix}$$

توجه داریم که بردار های ویژه باید واحد باشند به عبارتی طول این بردار ها باید برابر ۱ باشد.



شکل ۴-۱۵ داده های اولیه به همراه ترسیم بردار های ویژه

همانطور که در شکل ۴-۱۵ مشاهده می شود بردارهای ویژه بر همدیگر عمود می باشند. نکته مهم تری که می توان مشاهده کرد این است که، یکی از بردار های ویژه درست از وسط داده ها عبور کرده است، مانند اینکه بخواهیم خطی را از درون داده ها عبور دهیم بصورتیکه بهترین انطباق را با داده ها

داشته باشد. این مثال نشان می دهد که چگونه داده ها حول بردار ویژه اول گسترده شده اند. بردار ویژه دوم اطلاعات نسبتاً کمتری نشان می دهد. بیشتر داده ها بر روی بردار اولیه قرار گرفته است و فقط نوساناتی اطراف بردار ویژه اصلی دارد.

ث) استخراج مولفه اصلی مناسب و بدست آوردن بردار ویژگی

در این گام می توان نشان داد که چگونه داده ها فشرده می شوند. با دقت در مقادیر ویژه می توان گفت، بردار ویژه متناظر با بالاترین مقدار ویژه، مولفه اصلی داده را نشان می دهد. ابتدا بردارهای ویژه ماتریس کواریانس را بدست آورده و در گام دوم آنها را به ترتیب مقادیر ویژه شان به صورت نزولی مرتب می کنیم. سپس تعدادی از بردارها را که مقدار ویژه کمی دارند حذف می کنیم. اما این عمل باعث از دست رفتن مقداری از اطلاعات خواهد شد. برای کم کردن میزان از دست رفتن اطلاعات، لازم است به اندازه مقادیر ویژه توجه شود یعنی مقادیر کوچک حذف شوند. با حذف تعدادی از مولفه ها ابعاد داده ها کاهش خواهد یافت. به عبارت دقیق تر اگر داده ها دارای n بعد باشند تعداد بردارهای ویژه و مقادیر ویژه نیز برابر n خواهد شد. اگر p تا از این مولفه ها را انتخاب کنیم ابعاد داده های جدید برابر p خواهد شد.

در نهایت باید تعدادی از بردارها انتخاب شده و به صورت ستونی در ماتریس ویژگی ذخیره می شوند به این ماتریس *feature vector* می گویند. برای مثال داده شده بردار ویژگی به صورت زیر خواهد بود.

$$featureVector = \begin{pmatrix} -0.735179 & 0.677873 \\ 0.677873 & 0.735179 \end{pmatrix}$$

و در صورتی که بخواهیم ابعاد داده ها را کاهش دهیم بردار ویژه با کمترین مقدار را حذف می کنیم و در نتیجه *feature vector* به صورت زیر خواهد شد.

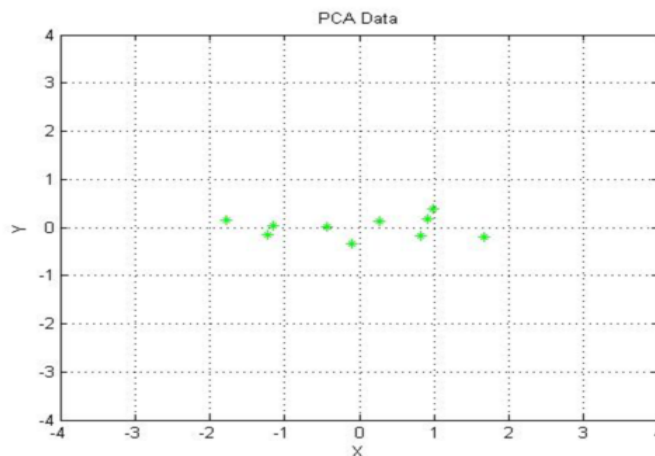
$$featureVector = \begin{pmatrix} -0.735179 \\ 0.677873 \end{pmatrix}$$

ج) بدست آوردن داده های جدید

برای بدست آوردن داده های نهایی، پس از انتخاب مولفه های لازم، بردار ویژگی ترانهاده شده و در بردار داده ها ضرب می شود، بنابراین می توان نوشت.

$$Finaldata = RowFeatureVector * RowDataAdjust \quad (۱۵-۴)$$

RowFeatureVector در واقع ماتریسی است که هر یک از سطر های آن یکی از بردار های ویژگی می باشد و *RowDataAdjust* ماتریسی ستونی است و در هر کدام از ستون ها یک نمونه داده وجود دارد؛ و هر سطر نشان دهنده یکی از مولفه های داده می باشد. نتیجه این عملیات این است که داده ها با استفاده از بردار های ویژه بیان می شود. همانطور که می دانیم، می توانیم داده ها را با استفاده از بردار های مختلفی به تصویر بکشیم، صرفا لازم است تا بردار ها بر هم عمود باشند. زمانی که تعدادی از مولفه ها را کاهش می دهیم، داده ها را صرفا با تعداد کمتری بعد نمایش می دهیم. زمانی که از یک بردار برای نمایش استفاده کنیم، به جای نمایش داده ها بر روی محور های x و y می توانیم آنها را با تصویر آن، که بر روی بردار ویژه اصلی واقع شده است، نشان داد.



شکل ۴-۱۶ داده های بعد از نگاشت با استفاده از بردار های ویژه

همانطور که در شکل ۴-۱۶ مشاهده می شود، داده ها در واقع همان داده های قبلی می باشد با این تفاوت که دستگاه مختصات دوران داده شده و داده ها در طول محور x ها گسترش یافته است. در صورتیکه فقط از یک بردار که دارای بالاترین مقدار است استفاده شود، در این صورت داده ها به صورت یک بعدی در آمده و بر روی محور x ها نگاشت می یابد.

تاکنون سعی کردیم تا داده ها را با استفاده از الگوهایی که در درون داده ها داریم نگاشت دهیم. این الگوها در واقع خطوطی هستند که همراستا با بردارهای ویژگی ماتریس کواریانس داده ها می باشند. در این نمایش، هر کدام از داده ها، با استفاده از میزان مشارکتی که هر کدام از مولفه ها دارند، نمایش داده می شوند. در واقع نمایش اولیه داده ها با استفاده از محورهای x و y ، ارتباطی را که بین داده ها وجود دارند، نشان نمی دهد.

(چ) بازیابی داده های قبلی

برای بازیابی داده ها لازم است تا همه بردارهای ویژگی را داشته باشیم. اگر در مرحله تولید داده های جدید، تعدادی از بردارهای ویژگی را حذف کنیم مقداری از اطلاعات را از دست داده ایم. برای استخراج داده های جدید از رابطه زیر استفاده می شود.

$$FinalData = RowFeatureVector * RowDataAdjust \quad (۴-۱۶)$$

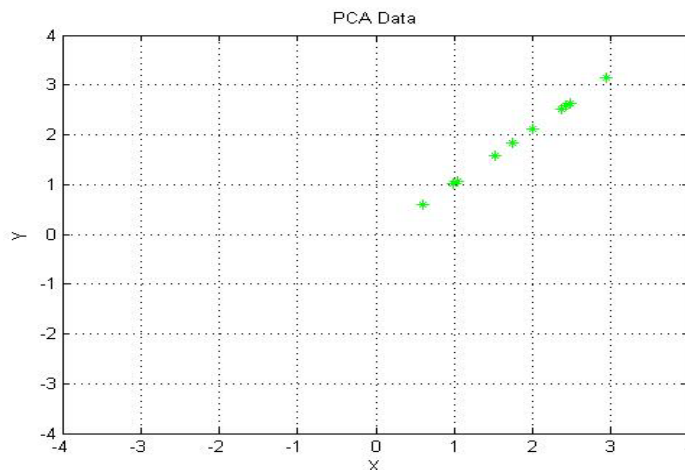
بنابراین برای بازیابی داده های اولیه از رابطه (۴-۱۷) استفاده می شود.

$$RowDataAdjust = RowFeatureVector^{-1} * FinalData \quad (۴-۱۷)$$

اما از آنجایی که بردارهای ویژگی، بردارهای واحد بوده و بر هم متعامد می باشند. پس برای بدست آوردن معکوس ماتریس $RowFeatureVector$ لازم است تا آن را ترانهاد کنیم. به عبارتی معکوس ماتریس $RowFeatureVector$ با ترانهاد آن برابر می باشد. برای تحلیل PCA، از همه داده ها مقدار

میانگین را کسر کردیم پس برای بدست آوردن مجدد داده ها لازم است تا داده های بدست آمده در این مرحله را با میانگین آنها جمع کنیم، پس رابطه (۴-۱۷) به صورت زیر خواهد شد.

$$RowDataA_{djust} = RowFeatureVector^T * FinalData + meanData \quad (۴-۱۸)$$



شکل ۴-۱۷ داده ها بعد از بازیابی

ارزیابی (از نتایج پایان نامه)

زمان محاسبات برای این الگوریتم بسیار زیاد می باشد. فرض کنید اندازه تصویر برابر $۲۴۰ * ۳۲۰$ باشد آنگاه تعداد عناصر ماتریس کواریانس برابر $۵۸۹۸۲۴۰۰۰۰ = (۳۲۰ * ۲۴۰)^۲$ می شود که بیشتر از ۵ میلیارد عنصر می باشد بنابراین عملاً محاسبه ممکن نمی باشد، بنابراین دو راهکار وجود دارد.

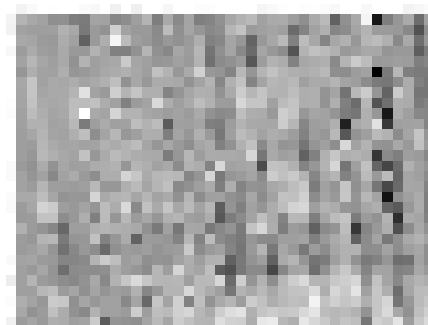
۱. از جبر خطی می دانیم مقادیر ویژه دو ماتریس $C = A * A^T$ و $L = A^T * A$ برابر می باشند اما ابعاد ماتریس L به مراتب کوچکتر می باشد برای بدست آوردن بردار های ویژه ماتریس C ، کافی است تا بردار ویژه ماتریس C را در ماتریس A ضرب کنیم. اما تعداد بردار هایی که در

این روش بدست می آید کم می باشد نمونه ای از اعمال این روش را بر روی یک نمونه، در شکل ۴-۱۸ آمده است.

۲. نمونه برداری: در این روش از داده ها نمونه برداری می کنیم. در این آزمایشات اندازه تصویر را در مقیاس یک شصت و چهارم کاهش دادیم.

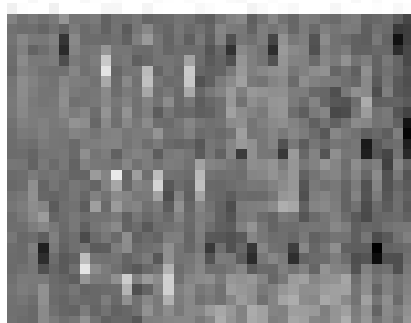


ب) فریم اصلی

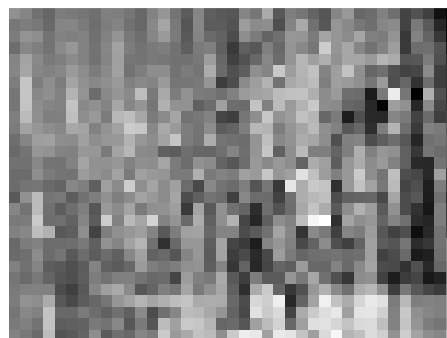


الف) فریم بازیابی شده با استفاده از ۱۱۸۰ بردار

ویژه



ت) فریم بازیابی شده با ۱۱۹۰ بردار از ۱۲۰۰ بردار



پ) فریم بازیابی شده با استفاده از تمام بردار

های ویژه

شکل ۴-۱۸ نتیجه مدل سازی پس زمینه با استفاده روش PCA

همانطور که از تصاویر فوق مشاهده می شود کیفیت تصویر تحت تاثیر تعداد بردارهای استفاده شده است و از طرفی زمان محاسبات این الگوریتم بالا می باشد به همین دلیل عملی نمی باشد.

۴-۲-۴- مدل سازی پس زمینه با استفاده از روش های آماری

یکی از رایج ترین روش های مدل سازی پس زمینه، رهیافت آماری می باشد. در این رهیافت با استفاده از بردار های ویژگی که از فریم های ویدیویی استخراج شده، تلاش می شود تا مدلی آماری از پس زمینه بدست آید. سپس برای تشخیص ورود یک شی، بردار ویژگی با مدل پس زمینه مقایسه می شود در صورتی که این بردار در بازه اطمینان قرار داشته باشد آن پیکسل (بلوک) به عنوان قسمتی از پس زمینه و در غیر اینصورت پیکسل جزئی از شی در نظر گرفته می شود. مدل سازی آماری به دو دسته پارامتریک و غیر پارامتریک تقسیم می شود.

در روش های پارامتریک برای تخمین تابع چگالی در خوشه بندی یا رگرسیون داده ها فرض می شود که تمام داده ها توسط یک مدل تولید می شود. مثلاً برای رگرسیون فرض می شود که تمام داده های ورودی بر روی یک خط واحد قرار دارند و برای هر ورودی، خروجی متناظر آن را می توان از روی مدلی که استخراج شده، تخمین زده می شود. برای خوشه بندی فرض می شود که داده های هر کدام از کلاس ها توسط یک توزیع، مثلاً توزیع نرمال، تولید می شود. بنابراین لازم است تا پارامتر های توزیع تخمین زده شود. امتیاز این روش این است که مثلاً به جای ذخیره کردن هزاران داده برای خوشه بندی داده ها، زمانی که توزیع از نوع، نرمال باشد کافی است میانگین و واریانس داده ها ذخیره شود. از معایب این روش این است که این فرض (فرض در مورد نوع توزیع) همیشه درست نمی باشد و می تواند منبع خطا شود. اگر نتوانیم داده ها را با یک مدل واحد نگاهت دهیم یک راهکار این است که داده ها را به کمک مدل های ترکیبی بیان کنیم.

در حالت غیر پارامتریک تنها فرض این است ورودی مشابه، خروجی مشابه تولید می کند. بنابراین در روش غیر پارامتریک الگوریتم شامل دو مرحله است:

۱. پیدا کردن نمونه های مشابه هم

۲. با کمک یک روش درونیابی مدلی را تخمین می زنیم

تفاوت الگوریتم های مختلف غیر پارامتریک را می توان در تابع اندازه گیری مشابهت و نوع روش درونیابی که مورد استفاده قرار می گیرد، دانست.

در روش پارامتریک فرض می شود که داده ها متعلق به یک مدل استاندارد می باشد و با استفاده از داده ها موجود، می توان پارامترهای مربوطه را بدست آورد. از طرف دیگر در روش های غیر پارامتریک هیچ فرض خاصی برای مدل در نظر گرفته نمی شود و سعی می شود با استفاده از داده های ورودی مدل را پیش بینی نماید.

مدل سازی پس زمینه با استفاده از روش غیر پارامتریک

فرض کنیم نمونه ها به صورت $X = \{x^t\}_{t=1}^N$ است، که هر یک به صورت مستقل از تابع توزیع $P(\cdot)$ به دست می آید. از آنجایی که تابع توزیع را نمی توان به طور دقیق به دست آورد، آن را از روی نمونه های موجود تخمین می زنیم. تخمین زنده های غیر پارامتریک را برای تابع تجمعی به صورت $F(x)$ نمایش می دهیم که در نقطه x می توان آن را به کمک نسبت تعداد نمونه های کوچکتر از x بر روی تمام نمونه ها بدست آورد؛ و به صورت رابطه (۴-۱۹) نمایش داده می شود [۱۹].

$$F(\hat{x}) = \frac{\#\{x^t \leq x\}}{N} \quad (۴-۱۹)$$

$\#\{x^t \leq x\}$ تعداد نمونه های کوچکتر از x بوده و برای تخمین تابع چگالی می توان از رابطه (۴-۲۰) استفاده کرد.

$$\hat{p}(x) = \frac{1}{h} \left[\frac{\#\{x^t \leq x+h\} - \#\{x^t \leq x\}}{N} \right] \quad (۴-۲۰)$$

h : طول بازه ای است که نمونه ها در آن واقع می شود.

هیستوگرام

در روش های پارامتریک فرض می شود که داده ها دارای مدلی خاص باشد؛ اما در بسیاری موارد هیچ یک از مدل های استاندارد با داده ها انطباق ندارند. زمانی که مدل خاصی موجود نباشد از هیستوگرام ها استفاده می شود در این روش بازه ای را که داده ها در آن قرار گرفته است، به چندین زیر بازه تقسیم کرده و تعداد عناصر درون هر یک از زیر بازه ها را می شماریم. بنابراین اگر فرض کنیم که تعداد n_j داده در بازه Δ_j قرار داشته باشد و طول بازه Δ_j و کل تعداد داده برابر N باشد، آنگاه احتمال وجود داده ای در بازه Δ_j برابر $\hat{p}_j = n_j/N\Delta_j$ می شود. طول این بازه ها می تواند یکسان یا متغیر باشد. در ضمن مجموع مساحت بازه ها باید برابر ۱ شود. انتخاب طول بازه مناسب در هیستوگرام تاثیر زیادی بر کارایی این روش دارد برای درک بهتر این موضوع اگر به تصاویر مندرج در شکل ۴-۱۹ دقت شود مشاهده می شود در صورتی که اندازه میله ها بزرگ باشد نمودار زیاد هموار می باشد. تابع چگالی را می توان به صورت (۴-۲۱) یا (۴-۲۲) نمایش داد.

$$\hat{p}(x) = \frac{\#\{x^t \text{ in the same bin as } x\}}{Nh} \quad (۴-۲۱)$$

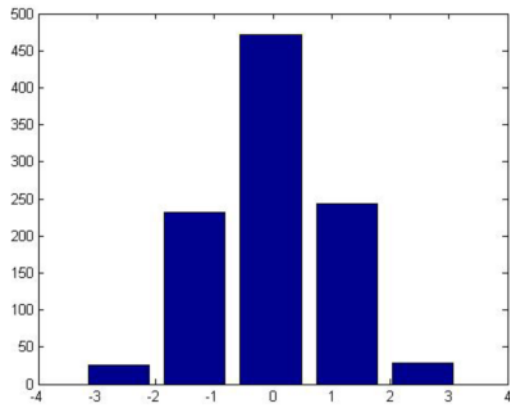
$$\hat{p}(x) = \frac{\#\{x-h < x^t \leq x+h\}}{2Nh} \quad (۴-۲۲)$$

رابطه فوق را می توان به صورت زیر بازنویسی کرد.

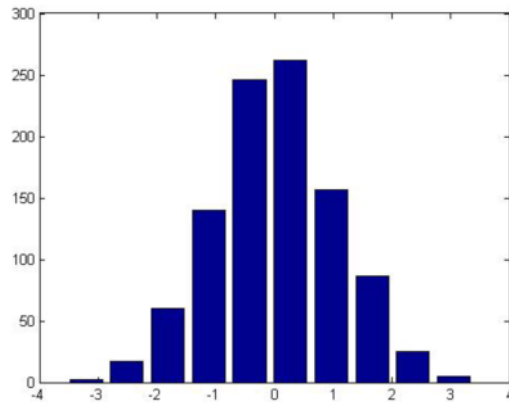
$$\hat{p}(x) = \frac{1}{Nh} \sum_{t=1}^N w\left(\frac{x-x^t}{h}\right) \quad (۴-۲۳)$$

که تابع w دارای رابطه زیر می باشد.

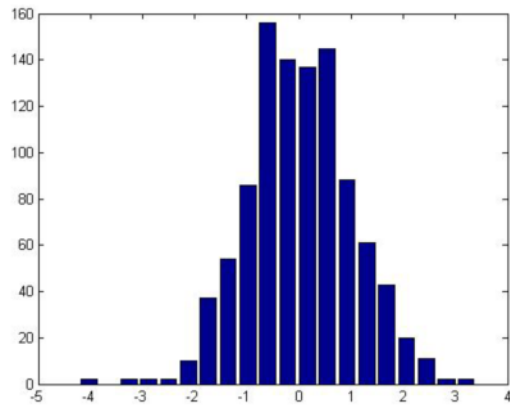
$$w(u) = \begin{cases} \frac{1}{2} \text{ if } |u| < 1 \\ 0 \text{ otherwise} \end{cases} \quad (۴-۲۴)$$



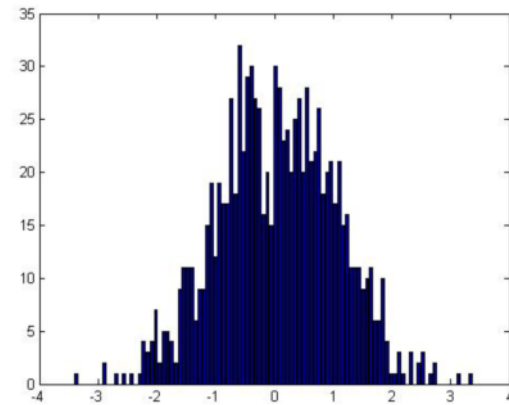
(ب) تعداد بین ها = ۵



(الف) تعداد بین ها = ۱۰



(ت) تعداد بین ها = ۲۰

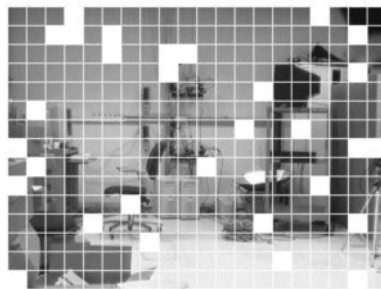


(پ) تعداد بین ها = ۱۰۰

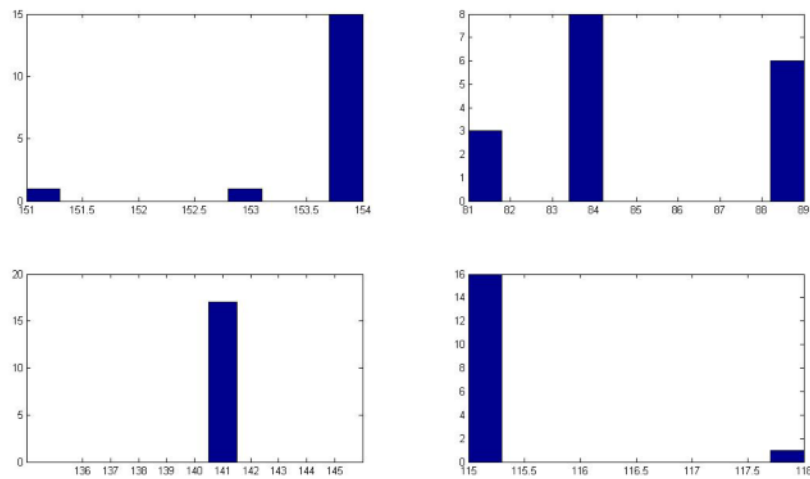
شکل ۴-۱۹ در شکل ابتدا ۱۰۰۰ نقطه را که به صورت تصادفی توسط تابع تصادفی نرمال تولید شدند به کمک روش هیستوگرام سعی در تخمین تابع توزیع چگالی را آنها را هستیم همانطوری که مشاهده می کنید زمانی که تعداد بین ها کم باشد میزان خطاها بالاست.

ارزیابی روش غیر پارامتریک برای مدل سازی پس زمینه (از نتایج پایان نامه)

برای بررسی این روش از مدل سازی پس زمینه آن را بر روی دو نمونه ویدیو اعمال نموده ایم. ابتدا چند بلوک را به صورت تصادفی انتخاب کرده و سطح خاکستری آنها را بررسی می کنیم. در تصویر ۲۰-۴ بلوک هایی را که مورد بررسی قرار گرفته، نشان داده شده است. برای بررسی تغییرات سطح خاکستری در ۱۷ فریم ابتدایی ویدیو، ۳۰ بلوک به صورت تصادفی انتخاب کرده و هیستوگرام آنها را رسم نمودیم همانطور که در شکل ۴-۲۱ مشاهده می شود، در این محیط تغییرات سطح خاکستری در طول ۱۷ فریم تقریباً ثابت می باشد.



شکل ۴-۲۰ این تصویر بلوک های را که برای بررسی نحوه تغییرات سطح خاکستری مورد استفاده قرار می گیرد نشان داده شده است.



شکل ۴-۲۱ این تصویر نمودار هستوگرام ۴ بلوک دلخواه را نشان می دهد همانطور که مشاهده می شود سطح خاکستری در تمام ۱۷ فریم تقریبا ثابت می باشد.

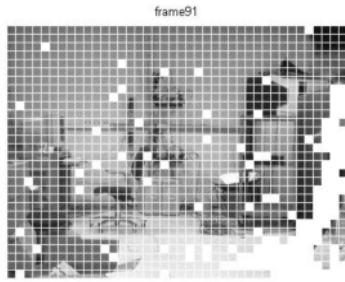
برای استخراج شی در صحنه ابتدا هستوگرام را برای تمام ۳۰۰ بلوک محاسبه (برای کاهش حجم محاسبات به جای تمام نقاط درون بلوک فقط تعدادی از نقاط را انتخاب می کنیم و هستوگرام آنها را رسم می کنیم) سپس بردار ویژگی را برای فریم جدید بدست می آوریم. اگر اختلاف مقدار یک بلوک و مقدار هستوگرام متناظرش، در بازه قابل پذیرش باشد آن پیکسل را به عنوان پس زمینه، در غیر اینصورت به عنوان شی در نظر می گیریم.



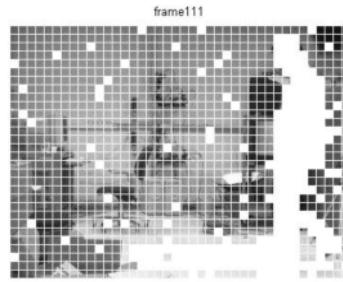
ب) فریم ۹۱

الف) فریم ۱۱۱

شکل ۴-۲۲ اندازه بلوک ها برابر 16×16 می باشد شماره بر روی عنوان تصاویر نشان دهنده شماره فریم در ویدیو می باشد. اشیا استخراج شده توسط الگوریتم مدل سازی پس زمینه با استفاده از هستوگرام

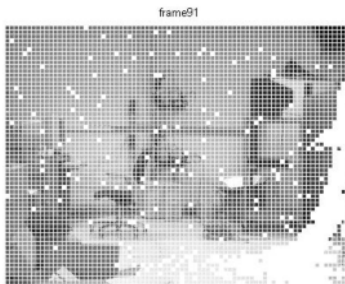


(ب) فریم ۹۱

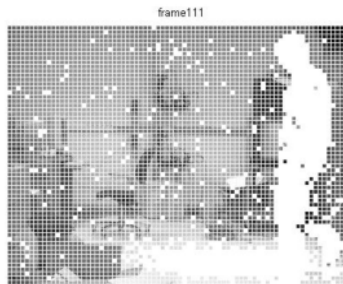


الف (فریم ۱۱۱)

شکل ۴-۲۳ اندازه بلوک ها برابر $8*8$ می باشد شماره بر روی عنوان تصاویر نشان دهنده شماره فریم در ویدیو می باشد.



(ب) فریم ۹۱

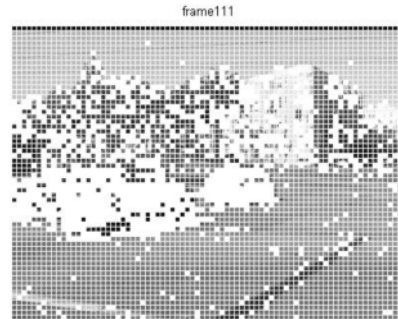
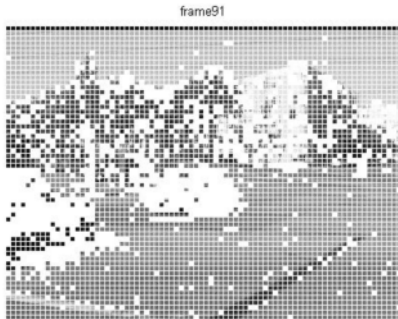


الف (فریم ۱۱۱)

شکل ۴-۲۴ اندازه بلوک ها برابر $4*4$ می باشد شماره بر روی عنوان تصاویر نشان دهنده شماره فریم در ویدیو می باشد. اشیا استخراج شده توسط الگوریتم مدل سازی پس زمینه با استفاده از هستوگرام

نحوه تشخیص اشیا در این روش به این صورت است که مقادیر جدید بلوک با هیستوگرام متناظرش مقایسه می شود؛ احتمال تعلق آن بلوک به پس زمینه، برابر با مقدار احتمال نزدیکترین میله به آن می باشد. در صورتی که فاصله آن از همه میله ها، بیشتر از حدآستانه باشد، احتمال آن برابر صفر و به عنوان شی در نظر گرفته می شود. همانطور که در تصاویر ۴-۲۲ تا ۴-۲۴ مشاهده می شود زمانی که اندازه بلوک ها را بزرگ در نظر می گیریم قسمتی از شی پوشش دادن داده نشده ولی زمان محاسبات

کمتر می شود. با آزمایشات متعدد مشاهده شده که مناسب ترین اندازه بلوک برای پوشش تمام شی، بلوک های ۴*۴ می باشد.



ب) فریم ۹۱

الف) فریم ۱۱۱

شکل ۴-۲۵ نتیجه خروجی روش مدل سازی پس زمینه با استفاده از هیستوگرام برای استخراج اشیا از

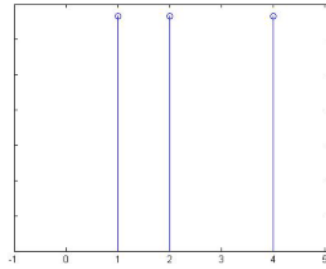
صحنه در محیط برونی

همانطور که در شکل ۴-۲۵ مشاهده می شود میزان خطا برای فریم های اولیه کمتر از فریم های بعدی می باشد. مدل با استفاده از فریم های اولیه ساخته می شود به همین دلیل برای فریم های اولیه میزان خطا کمتر است؛ اما به مرور زمان پس زمینه در محیط های بیرونی تغییر می کند به همین دلیل لازم است در طول زمان، پس زمینه بروز شود تا خطا کاهش یابد.

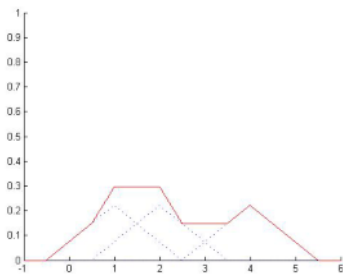
مدل سازی با استفاده از کرنل ها

داده ها غالباً دارای اطلاعاتی از مدل آماری هستند که بر اساس آن تولید شدند. می توان هر نمونه را به صورت یک تابع دلتا در نظر گرفت که مجموع مساحت این میله ها برابر ۱ می شود. همانطور که در شکل ۴-۲۶(الف) ملاحظه می شود سه نمونه داده داریم که در نقاط ۱،۲ و ۴ واقع شده اند اما این تابع ناپیوسته می باشد به همین دلیل لازم است تا روشی استفاده کنیم تا تابع توزیع چگالی پیوسته باشد؛ برای نیل به این هدف می توان از توابع کرنل استفاده کرد. برای مدل سازی تابع توزیع چگالی

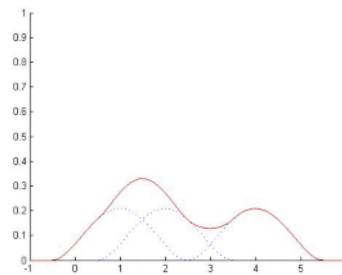
با استفاده کرنل لازم است تا ابتدا میزان عرض تابع کرنل را تعیین کنیم در گام دوم به ازای هر کدام از نقاط، یک کرنل به مرکز آن نقطه تصویر می کنیم ارتفاع کرنل باید به صورتی انتخاب شود که مجموع مساحت کرنل ها برابر ۱ باشند [۲].



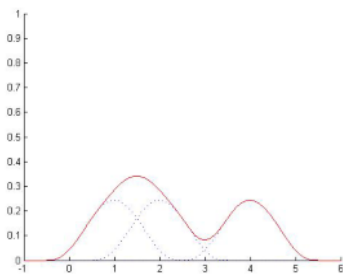
الف) نمونه داده ها



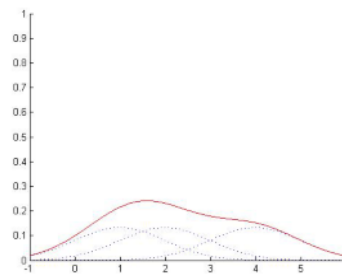
پ) مدل سازی توزیع احتمال با کرنل مثلثی



ب) مدل سازی توزیع احتمال با کرنل *Bivariate*



ث) مدل سازی توزیع احتمال با کرنل *triweight*



ت) مدل سازی توزیع احتمال با کرنل *normal*

شکل ۴-۲۶ مدل سازی توزیع احتمال یا استفاده از چند کرنل استاندارد

اشکال ۴-۲۶ (ب) تا ۴-۲۶ (ث) نمایش توزیع احتمال را برای ۳ نقطه شکل ۴-۲۶ (الف) نشان می دهد. همانطور که در شکل دیده می شود کرنل مثلثی توزیعی ناهموار دارد ولی سایر کرنل ها توزیع های هموارتری را ارائه می دهند. از بین کرنل ها کرنل نرمال، توزیع احتمال هموارتری دارد.

فرض کنید که مقدار کمیتی که می خواهیم برای آن توزیع نرمال طراحی کنیم به صورت $\{x_1, x_2, \dots, x_N\}$ باشد می خواهیم احتمال وقوع کمیت x را با استفاده از کرنل ها تخمین بزنیم برای این منظور می توان از رابطه (۴-۲۵) استفاده کنیم.

$$p(x) = \frac{1}{N} \sum_{i=1}^N K_h(x - x_i) \quad (۴-۲۵)$$

که در رابطه فوق K_h کرنلی با عرض h را نشان می دهد.

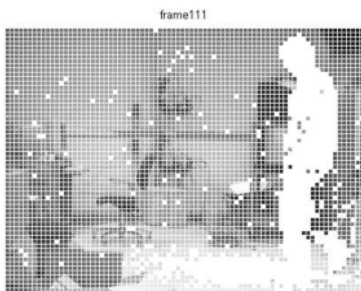
ارزیابی مدل سازی پس زمینه با استفاده از کرنل ها (از نتایج پایان نامه)

در این قسمت کارایی کرنل های مختلف را بر روی مدل سازی پس زمینه و استخراج اشیا مورد بررسی قرار می دهیم. همانطور که در تصاویر درج شده در شکل های ۴-۲۷ تا ۴-۳۲ مشاهده می شود میزان حدآستانه ای که مورد استفاده قرار می گیرد بر روی میزان خطا در فریم تاثیر مستقیم دارد بهترین بازه برای این رهیافت مقدار حدآستانه در بازه $[0,009, 0,05]$ می باشد. زمانی که مقدار حدآستانه را بزرگ در نظر می گیریم بدون توجه به نوع کرنل میزان خطای نوع FP به شدت افزایش می یابد این خطا تا حدی بالا می رود که حجم عمده پس زمینه به عنوان پیکسل های شی در نظر گرفته می شود.

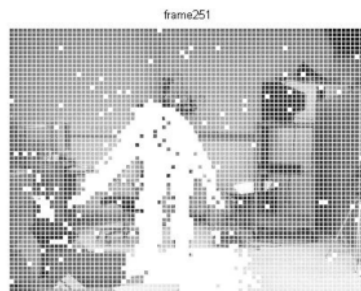
از بین کرنل های موجود بهترین کارایی مربوط به کرنل نرمال می باشد همانطور که در نتایج مشاهده می شود کرنل نرمال دارای کمترین خطای از نوع FP می باشد از طرفی میزان خطای FN آن نیز با سایر کرنل ها یکسان می باشد. از نقاط برتری دیگر این کرنل هموارتر بودن تابع آن نسبت به سایر

روش ها می باشد. اما از معایب این کرنل می توان به زمان محاسبات نسبتا بالای این کرنل در مقایسه با سایر کرنل ها و همچنین حساسیت نسبتا بالای این روش به انتخاب حد آستانه اشاره نمود (همانطور که مشاهده می کنید زمانی که حد آستانه را برابر با ۰,۲ انتخاب می کنیم خطای FP در این روش به شدت بالا می رود به طوری که تمام صحنه سفید می شود).

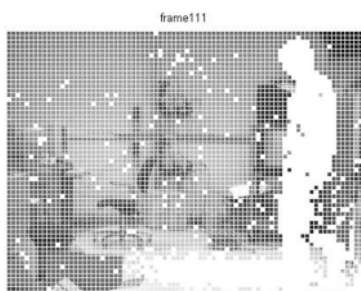
در مرحله نهایی کارایی این روش ها برای مدل کردن محیط های بیرونی مورد ارزیابی قرار گرفته است همانطور که در تصاویر مندرج در شکل ۴-۳۳ مشاهده می شود میزان خطای FN این روش بالا می باشد.



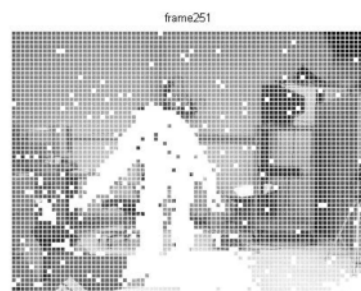
(ب) فریم ۱۱۱ با حد آستانه ۰,۰۹



(الف) فریم ۲۵۱ با حد آستانه ۰,۰۹

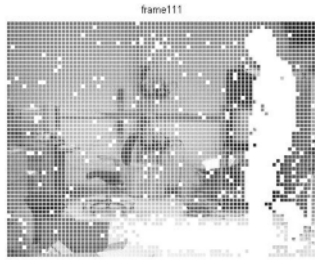


(پ) فریم ۱۱۱ با حد آستانه ۰,۰۵

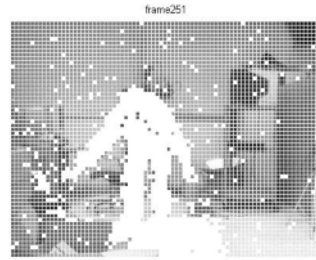


(پ) فریم ۲۵۱ با حد آستانه ۰,۰۵

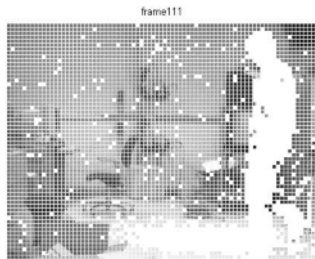
شکل ۴-۲۷ تصاویر استخراج شده از مدل سازی با کرنل با استفاده از کرنل نرمال



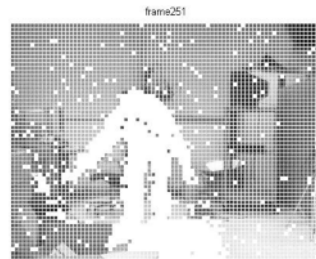
ب) فریم ۱۱۱ با حد آستانه ۰,۰۹



الف) فریم ۲۵۱ با حد آستانه ۰,۰۹



ت) فریم ۱۱۱ با حد آستانه ۰,۰۵



پ) فریم ۲۵۱ با حد آستانه ۰,۰۵

شکل ۴-۲۸ تصاویر استخراج شده از مدل سازی با کرنل با استفاده از کرنل Biweight



ب) فریم ۱۱۱ با حد آستانه ۰,۰۹



الف) فریم ۲۵۱ با حد آستانه ۰,۰۹

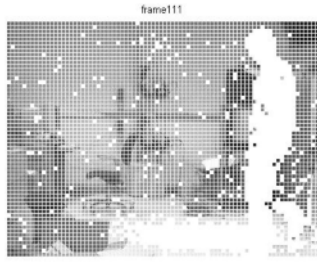


پ) فریم ۱۱۱ با حد آستانه ۰,۰۵

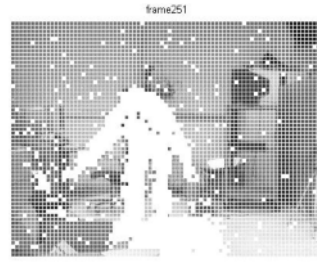


پ) فریم ۲۵۱ با حد آستانه ۰,۰۵

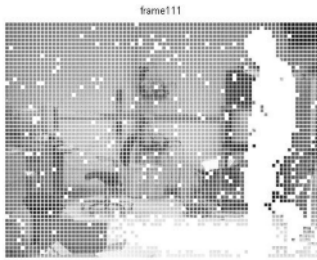
شکل ۴-۲۹ تصاویر استخراج شده از مدل سازی با کرنل با استفاده از کرنل Triweight



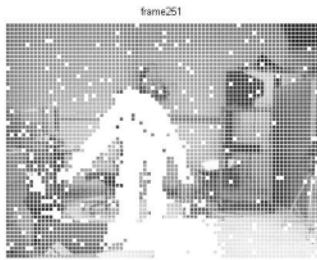
ب) فریم ۱۱۱ با حد آستانه ۰,۰۹



الف) فریم ۲۵۱ با حد آستانه ۰,۰۹



پ) فریم ۱۱۱ با حد آستانه ۰,۰۵



پ) فریم ۲۵۱ با حد آستانه ۰,۰۵

شکل ۳-۴ استخراج شده از مدل سازی با کرنل با استفاده از کرنل Epanechnikov



ب) فریم ۱۱۱ با حد آستانه ۰,۰۹



الف) فریم ۲۵۱ با حد آستانه ۰,۰۹



پ) فریم ۱۱۱ با حد آستانه ۰,۰۵

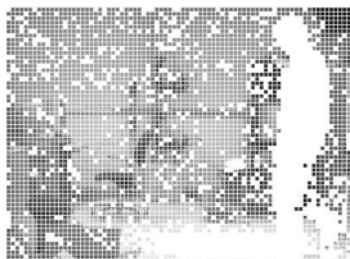


پ) فریم ۲۵۱ با حد آستانه ۰,۰۵

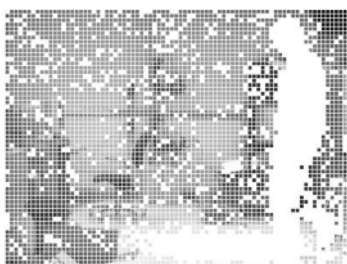
شکل ۳۱-۴ استخراج شده از مدل سازی با کرنل با استفاده از کرنل Triangle



$Width=0,0000001$



$Width=0,000001$

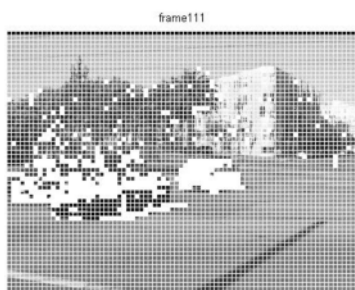


$Width=0,05$

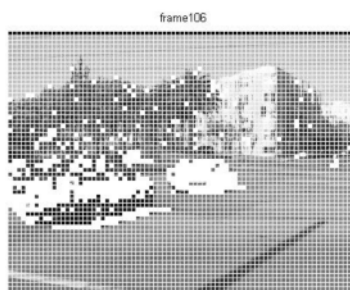


$Width=1,2$

شکل ۳۲-۴ بررسی اثر عرض کرنل بر کارایی مدل سازی با استفاده از کرنل نرمال



الف) فریم ۱۱۱ با حد آستانه ۰,۰۰۵



ب) فریم ۱۰۶ با حد آستانه ۰,۰۰۵

شکل ۳۳-۴ بررسی استخراج اشیا در محیط بیرونی با استفاده از رهیافت مدل سازی با استفاده از کرنل

مدل سازی پس زمینه با استفاده از روش های پارامتریک

در روش های پارامتریک فرض می شود که داده ها دارای یک مدل استاندارد می باشند؛ این مدل می تواند مدلی ساده مانند توزیع یکنواخت، گاما، بتا، یا توزیع نرمال، ترکیبی خطی یا غیر خطی از چند

توزیع استاندارد باشد. در ادامه از توزیع نرمال برای مدل سازی پس زمینه استفاده می کنیم؛ سپس به توضیح روش ترکیبی که در مقاله *stuffer*[۴] معرفی شده است، می پردازیم.

استفاده از توزیع نرمال جهت مدل سازی پس زمینه

یکی از شناخته شده ترین و پر کاربرد ترین مدل در حوزه علوم و مهندسی توزیع نرمال است، این توزیع مدلی پیوسته می باشد و دارای رابطه ای به صورت زیر می باشد.

$$f(x|\mu, \sigma^2) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\left\{\frac{(x-\mu)^2}{2\sigma^2}\right\}} \quad (26-2)$$

که در آن $-\infty \leq x \leq \infty, -\infty \leq \mu \leq \infty, \sigma^2 > 0$. برای نمایش توزیع نرمال فقط کافی است تا میانگین و واریانس توزیع را داشته باشیم [۱۹].

تعیین پارامترها

فرض کنید N نمونه در اختیار داریم می خواهیم تابع توزیع احتمالی بدست آوریم که به کمک آن احتمال رخداد یک مقدار معین را پیش بینی کنیم. برای این کار ابتدا باید نوع تابع توزیع را مشخص کرده سپس پارامترهای آن را محاسبه نماییم. حال فرض کنید که نوع تابع توزیع را بر اساس اطلاعات موجود می دانیم، حال چگونه می توان با استفاده از داده ها پارامترها را تخمین زد؟ برای اینکار معمولا از الگوریتم Maximum Likelihood Estimation استفاده می شود که در ادامه آن را توضیح می دهیم [۸].

Maximum Likelihood Estimation

فرض کنید داده های آماری به صورت (۴-۲۷) نمایش داده شوند

$$X = \{x^t\}_{t=1}^N \quad (27-4)$$

و فرض کنید که نمونه x توسط مدل احتمال (۴-۲۸) تولید شود.

$$x^t \sim p(x|\theta) \quad (۴-۲۸)$$

هدف این است که بر اساس این نمونه های آموزشی و مدلی که حدس زده ایم مقدار θ را تخمین بزنیم. در این مسئله متغیرها را مستقل از هم فرض می کنیم بنابراین رابطه ای به صورت (۴-۲۹) داریم:

$$l(\theta) \equiv p(X|\theta) = \prod_{i=1}^N p(x^i|\theta) \quad (۴-۲۹)$$

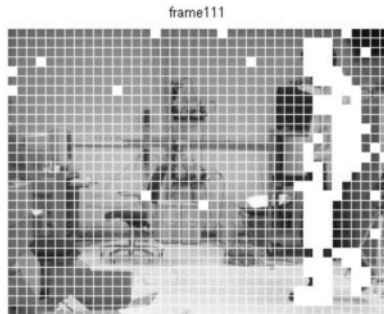
بایستی مقدار θ به نحوی انتخاب شود که مقدار تابع l به ازای نمونه های داده شده ماکزیمم شود برای این کار با محاسبه مقدار مشتق تابع l نسبت به θ و برابر صفر قرار دادن مقدار این توابع می توان نقاط اکسترمم آن را به دست آورد. از آنجایی که محاسبه مشتق برای تابع فوق به علت ضرب چند عامل در هم مشکل است معمولاً از این تابع لگاریتم می گیریم پس رابطه فوق به صورت زیر در می آید که همه عوامل ضرب در آن به جمع تبدیل می شود.

$$L(\theta|X) \equiv \log l(\theta|X) = \sum_{i=1}^N \log p(x^i|\theta) \quad (۴-۳۰)$$

ارزیابی مدل سازی پس زمینه با استفاده از روش های پارامتریک (از نتایج پایان نامه)

ابتدا آزمایشی بر روی محیط درون ساختمان انجام شده است. در ابتدا فریم را به بلوک های ۸*۸ تقسیم کرده و سپس از درون هر کدام از بلوک ها مقدار یک پیکسل را در طول ۱۷ فریم متوالی استخراج می کنیم. با استفاده از این مقادیر مدل آماری را استخراج می کنیم هر فریم جدیدی که وارد می شود مقدار پیکسل های بلوک ها مورد نظر آن را استخراج می کنیم اگر مقدار اختلاف بین میانگین و این مقادیر بیشتر از دو برابر واریانس باشد آن بلوک را به عنوان بلوک متعلق به شی و در غیر اینصورت به عنوان پس زمینه در نظر می گیریم. همین آزمایش را برای بلوک هایی به اندازه ۴*۴

نیز انجام می دهیم نتایج این آزمایشات به ترتیب در اشکال ۴-۳۴ و ۴-۳۵ آمده است همانطور که مشاهده می شود هر چه اندازه بلوک ها کوچکتر می شود دقت در تشخیص اشیا بالاتر می رود.

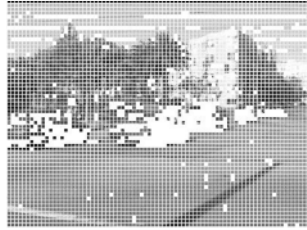


شکل ۴-۳۴ در این تصویر زیر چند نمونه از اشیایی را می بینید که توسط مدل گوسی از پس زمینه جدا شده اند اندازه بلوک ها برابر $8*8$ می باشد.



شکل ۴-۳۵ در این تصویر چند نمونه از اشیایی را می بینید که توسط مدل گوسی از پس زمینه جدا شده اند اندازه بلوک ها برابر $4*4$ می باشد.

دومین محیط مورد بررسی محیط بیرونی است. نتایج این روش در شکل ۴-۳۶ آمده است در این محیط میزان خطای FN زیاد است. به عبارتی تعداد زیادی از پیکسل های شی شناسایی نشده است این مسئله می تواند به دو دلیل رخ می دهد: ۱- پارامترها به درستی انتخاب نشده باشد ۲- مدل دقت کافی ندارد. همانطور که در شکل ۴-۳۷ می بینید زمانی که حد آستانه را تغییر می دهیم میزان خطای FN کاهش می یابد اما همزمان خطای FP افزایش می یابد.



(ب) فریم ۲۵۱



(الف) فریم ۱۵۱

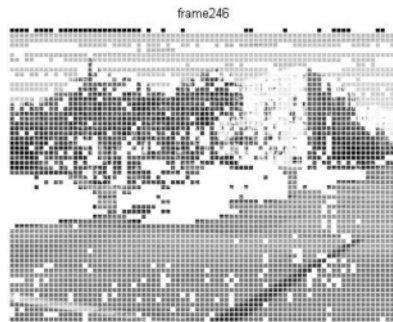


(ت) فریم ۱۱۱



(پ) فریم ۹۱

شکل ۴-۳۶ چند نمونه از اشیایی که توسط مدل گوسی از پس زمینه جدا شده اند. اندازه بلوک ها برابر 4×4 می باشد محیط مورد بررسی خارجی می باشد.



شکل ۴-۳۷ اشیا استخراج شده با استفاده از مدل گوسی با تغییر حد آستانه

مدل سازی پس زمینه با استفاده از *Gaussian Mixture Modeling*

این الگوریتم برای اولین بار در [۴] معرفی شد؛ در این متد برای نمایش پس زمینه از ترکیب چند مدل گوسی استفاده می شود، این روش برای کاربرد های بی درنگ بسیار مناسب بوده و از مزایای آن

می توان به کارایی بالای آن در شرایطی که دارای حرکت های تکراری تصادفی هستند (مانند آبشار، حرکت شاخه) اشاره نمود. در ادامه به بیان مبانی ریاضی این روش می پردازیم.

مدل ریاضی

مقدار یک پیکسل متناسب با شعاع نور بازتاب شده از سطح اشیا موجود در صحنه و نویزها می باشد. بر این اساس پس زمینه ها را می توان در دو دسته تقسیم نمود. دسته اول محیط هایی که پس زمینه آن تحت تاثیر یک شی ثابت قرار دارند مانند محیط های داخلی و دسته دوم محیط های بیرونی که معمولا مقدار پس زمینه می تواند توسط اشیا مختلفی در طی زمان تعیین شود (مانند حالتی که یک شاخه در حال حرکت داریم)، در این شرایط معمولا مقدار پیکسل از چند منبع به وجود می آید که برای نمایش آن لازم است از توابع ترکیبی استفاده کنیم.

فرض کنیم دارای k شی هستیم که هر کدام از این k شی می توانند مقدار پیکسل را تعیین کند و مقادیری را که هر شی نشان می دهد به جای یک مقدار ثابت به صورت یک تابع گوسی نمایش داد نکته دیگر این است که در هر زمان یک شی می تواند مقدار پیکسل را تعیین کند سپس مقدار پیکسل می تواند با استفاده از رابطه آماری زیر بدست آید.

$$p(x|\theta) = \sum_{i=1}^K w_i p_i(x|\theta_i) \quad (31-4)$$

که در رابطه فوق θ بردار است که پارامترهای هر کدام از توابع را نشان می دهد و w_i وزن هر کدام از توابع را نشان می دهد (شی که بیشتر در پس زمینه مشاهده می شود سهم بیشتری در تعیین مقدار پس زمینه دارد) p_i تابع احتمالی است که مقدار شی θ_i را نشان می دهد.

اگر داده ها کامل باشد می توان از روش ماکزیمم کردن احتمال^۱ و زمانی که داده ها ناقص باشند) مانند مسئله کلاسه بندی زمانی که نمی دانیم داده به کدام کلاس تعلق دارد) می توان از روش *Exception likelihood* استفاده کرد این الگوریتم برای اولین بار توسط آقای [۲۳] *demster* معرفی شد.

ارزیابی (از نتایج پایان نامه)

نکته ای که از توضیحات می توان استنباط کرد این است که عملکرد این الگوریتم تحت تاثیر پارامتر های زیادی است؛ یکی از این پارامتر ها تعداد مولفه ها می باشد. در شکل ۴-۳۸ نتایج استخراج شده توسط این الگوریتم در محیط *indoor* مشاهده می شود؛ همانطور که ملاحظه می شود زمانی که تعداد مولفه ها زیاد می شوند خطای نوع *FP* افزایش می یابد اما همزمان با آن خطای نوع *FN* کاهش می یابد. دومین نکته ای که می توان استنباط کرد این است که در فریم های اولیه خطا کم می باشد اما زمانی که به سمت فریم های نهایی نزدیک می شویم خطا افزایش می یابد. علت این پدیده این است که در فریم های اولیه چون شخص در حال تغییر موقعیت می باشد در نتیجه به عنوان پس-زمینه محاسبه نمی شود ولی در فریم های نهایی تدریجا شخص در موقعیت تقریبا ثابت قرار می گیرد به همین دلیل به عنوان جزئی از پس زمینه محاسبه می شود.



(ب) تعداد کرنل ۳

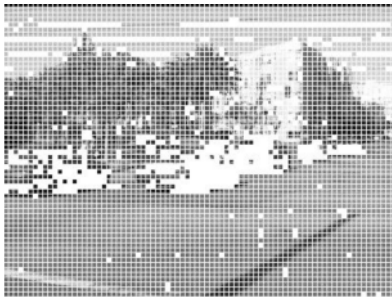


(الف) تعداد کرنل ۵

شکل ۴-۳۸ شباهت استخراج شده توسط الگوریتم *Gaussian Mixture Model* در محیط *indoor*

^۱ Maximum likelihood

در شکل ۴-۳۹ نتایج عملکرد این الگوریتم بر روی نمونه های بیرونی نشان داده شده است. همانند حالت درونی (*indoor*) زمانی که تعداد کرنل ها تغییر می یابد میزان خطای FN و FP تحت تاثیر قرار می گیرند. هر چه اشیا نزدیکتر باشند دقت تشخیص بالاتر و هر چه اشیا در فاصله دور تر قرار می گیرند دقت تمایز کاهش می یابد، زیرا در فاصله های دور، شی نسبتا ثابت است به همین دلیل همراه پس زمینه مدل می شود.



ب) تعداد کرنل ها ۳



الف) تعداد کرنل ها برابر ۵

شکل ۴-۳۹ اشیا استخراج شده توسط الگوریتم Gaussian Mixture Model در محیط *outdoor*

۴-۳- بررسی و مقایسه عملکرد روش های موجود مدلسازی پس زمینه (از نتایج

پایان نامه)

موارد فراوانی وجود دارد که لازم است تا برای ارزیابی الگوریتم های مدل سازی پس زمینه مورد بررسی قرار گیرد. از این میان می توان به حافظه مصرفی، زمان محاسبات و دقت اشاره نمود. برای اندازه گیری میزان دقت الگوریتم از دو نوع خطای FN و FP استفاده می کنیم. فرض می کنیم که اندازه فریم برای تمام الگوریتم ها $M*N$ بوده و فریم های به صورت سطح خاکستری^۲ می باشند.

حافظه مورد نیاز

^۲ gray scale

اولین الگوریتم مورد بررسی الگوریتم *stuffer* می باشد این الگوریتم دارای دو فاز است: فاز استخراج شی از پس زمینه، و فاز بروز رسانی پس زمینه می باشند. علاوه بر این، فرض می کنیم که ابعاد ویژگی هایی که استخراج می کنیم برابر d باشد. در فاز بروز رسانی لازم است تا میانگین، کواریانس و وزن را برای هر کدام از K کرنل مورد مطالعه محاسبه کنیم.

در ادامه میزان حافظه مورد نیاز برای میانگین، کواریانس و ماتریس وزن ها مورد ارزیابی قرار می دهیم.

الف) میانگین

۱ اگر ابعاد داده ها برابر d باشد اندازه میانگین برابر d خواهد شد.

۲ تعداد پیکسل ها برای یک فریم برابر $M*N$ می باشد.

۳ تعداد کرنل ها برابر K می باشد.

پس برای نگهداری کل تعداد درایه ها برای ذخیره ماتریس به $O(K*M*N*d)$ نیاز داریم.

ب) کواریانس

۱ اگر تعداد کرنل برابر k باشد.

۲ تعداد درایه های ماتریس کواریانس برابر $d*d$ می باشد.

۳ تعداد پیکسل ها برای تمام فریم برابر $M*N$ است.

پس کل درایه های مورد نیاز برابر $O(M*N*d*d*k)$ خواهد بود.

پ) ضرایب

۱ برای هر کدام از کرنل ها به یک عدد برای ضرایب نیاز داریم پس کلا به k عدد نیاز

داریم.

۲ تعداد پیکسل ها برای تمام فریم برابر $M*N$ است.

کل فضایی که لازم است تا ضرایب را ذخیره کنیم برابر $O(M*N*k)$ خواهد بود. با در نظر گرفتن موارد الف تا پ، مرتبه حافظه این الگوریتم برابر $O(M*N*k*d*d)$ خواهد بود. برای فاز استخراج اشیا، نیاز است تا میانگین، ماتریس کواریانس و بردار ضرایب را ذخیره کنیم همانند حالت قبل مرتبه حافظه برابر $O(M*N*d*d*k)$ است.

دومین الگوریتم مورد بررسی الگوریتم مدل سازی پس زمینه با استفاده از کرنل می باشد میزان حافظه مورد نیاز برای این الگوریتم متناسب با میزان حافظه مورد نیاز برای هر کرنل می باشد اگر فرض کنیم بطور متوسط برای هر کدام از پیکسل ها به K تا کرنل نیاز داشته باشیم و مرتبه حافظه هر کدام از کرنل ها H و تعداد کل پیکسل ها برای هر فریم نیز برابر $M*N$ می باشد؛ پس میزان حافظه مورد نیاز برای این روش برابر $O(M*N*K*H)$ خواهد بود.

در رهیافت های مدل سازی پس زمینه مبتنی بر میانگین فرض می شود که پس زمینه یک تصویر است. پس همه این متد ها به جز $MABM$ به $O(M*N)$ درایه نیاز دارند در روش $MABM$ اگر تعداد فریم هایی که برای مدل کردن پس زمینه استفاده می شود برابر L باشد (از آنجایی که لازم است همیشه این L فریم اخیر را نگه داریم) بنابراین به $O(L*M*N)$ درایه نیاز داریم.

الگوریتم مدل سازی با استفاده از تابع گوسی حالت ساده شده الگوریتم $stuffer$ می باشد در این روش به جای k کرنل فقط یک کرنل داریم که وزن این کرنل نیز برابر ۱ است بنابراین لازم نیست تا برای آن عددی را ذخیره کنیم. اگر مانند قبل فرض کنیم که برای هر کدام از پیکسل ها d ویژگی را استخراج کنیم، مرتبه حافظه برای این الگوریتم $O(M*N*d*d)$ خواهد بود.

الگوریتم میانه برای هر کدام از پیکسل ها یک مقدار میانه را حفظ می کند و به مرور زمان این مقدار را تغییر می دهد به قسمی که این مقدار برابر میانه *pixel processes* بشود کل پیکسل ها درون یک قاب برابر $M*N$ می باشد پس حافظه ای که این الگوریتم مورد نیاز دارد برابر $O(M*N)$ خواهد بود.

الگوریتم نهایی که مرتبه زمانی آن مورد ارزیابی قرار می گیرد الگوریتم مدل سازی با استفاده از *PCA* می باشد همانطور که فرض کردیم ابعاد فریم ها برابر $M*N$ است پس ماتریس کواریانس دارای $O((M * N)^2)$ درایه است که این حجم حافظه بسیار زیادی می باشد بنابراین پیادسازی این الگوریتم به حافظه زیادی نیاز دارد، پس لازم است تا از فریم ها نمونه برداری شود تا بتوان از این الگوریتم استفاده نمود یا از ماتریس L استفاده شود که به حافظه کمتری نیاز دارد اما اشکال این روش در این است که ناپایدار می باشد.

جدول ۴-۱ حافظه مورد نیاز هر کدام از روش ها را خلاصه می کند.

جدول ۴-۱ در این جدول میزان حافظه مورد نیاز برای هر کدام از روش ها نشان می دهد.

		Memory Requirement	
		Method	Requirement
statistics	approach	Stuffer and Grimson Methods	$O(K*M*N*d*d)$
		Gaussian background modeling	$O(M*N*d*d)$
		Kernel background model	$O(K*M*N*d*d)$;
Non statistics approach	Average methods	BMIT	$M*N$
		IBM	$M*N$
		LTABM	$M*N$
		MABM	$L*M*N$ (L denote number of frames, which are used in background modeling).
		RGABM	$M*N$
	PCA modeling	PCA modeling	$O((M * N)^2)$
		Median background modeling	$O(M*N)$

زمان مصرفی

دومین فاکتوری که مورد ارزیابی قرار داده ایم زمان مصرفی می باشد این فاکتور را می توان از دو جنبه زمان مصرفی واقعی و مرتبه زمانی مورد بررسی قرار داد. منظور از زمان مصرفی واقعی، زمانی است که الگوریتم لازم دارد تا ویدیو نمونه را مورد ارزیابی قرار داده و پس زمینه را مدل و اشیا را از صحنه ها جدا کند. برای محاسبه مرتبه زمانی تعداد عملیات های پایه ریاضی، که هر الگوریتم برای مدل کردن یا استخراج کردن اشیا از درون یک فریم لازم دارد، می شماریم. در ضمن در محاسبه مرتبه زمانی از زمان محاسبات عملیات جمع و تفریق در مقابل عمل ضرب و تقسیم صرف نظر می کنیم.

همانطور که قبلا بیان کردیم الگوریتم *stuffer* دارای دو فاز می باشد فاز بروز رسانی و فاز استخراج اشیا، در فاز بروز رسانی لازم است تا میانگین، ماتریس کواریانس و وزن برای هر کدام از کرنل ها بروز شود. برای بروز کردن میانگین در صورتی که ابعاد بردارها d باشد بروز رسانی آن $O(d)$ عمل ضرب نیاز دارد. تعداد عملیات ضرب برای ماتریس کواریانس برابر $O(d*d*d)$ خواهد شد برای بروز کردن وزن هر کرنل به تعدادی ثابت، ضرب نیاز داریم پس مرتبه زمانی آن برابر $O(1)$ می شود. بنابراین اگر تعداد کل کرنل ها برابر K باشد در کل فریم مرتبه زمانی $O(k*d*d*d*M*N)$ خواهد شد. برای فاز استخراج لازم است تا فاصله هر پیکسل از میانگین هر کدام از کرنل ها محاسبه شده و بررسی شود که آیا در بازه اطمینان قرار دارد یا خیر؟ بنابراین برای محاسبه این الگوریتم به $O(K*d*d*d*M*M)$ محاسبه نیاز دارد.

مرتبه زمانی استخراج با استفاده از کرنل گوسی، حالت ساده تر الگوریتم *stuffer* با یک کرنل می باشد؛ پس زمان محاسبات آن برابر $O(d*d*d*M*N)$ می شود. مرتبه زمانی برای الگوریتم استخراج با استفاده از کرنل گوسی، وابسته به زمان محاسبات کرنل می باشد. اگر محاسبات کرنل برابر H و تعداد متوسط کرنل ها برابر k باشد مرتبه زمانی این روش برابر $O(K*H*M*N)$ خواهد شد.

الگوریتم های *IBM* و *BMIT* نیاز به هیچ عملیات ضربی ندارد پس مرتبه زمانی آن $O(1)$ می شود. الگوریتم *LTABM* دو عملیات ضرب نیاز دارد پس مرتبه زمانی آن برابر $O(M*N)$ خواهد شد. الگوریتم *RGABM* به دو عملیات ضرب نیاز دارد به همین دلیل مرتبه زمانی آن مانند *LTABM* خواهد بود. الگوریتم *IRGABM* همانند الگوریتم *RGABM* می باشد به همین دلیل مرتبه زمانی و میزان حافظه آن یکسان می باشد.

الگوریتم *PCA* حداقل به $O((M * N)^2)$ عملیات ضرب نیاز دارد تا ماتریس کواریانس بدست آید. از طرفی باید بردار های ویژگی این ماتریس استخراج شود که این عملیات به نوع الگوریتم عددی مورد استفاده وابسته می باشد. مرتبه زمانی این الگوریتم (*PCA*) از تمامی الگوریتم های بررسی شده بیشتر است. نتایج در جدول ۴-۲ خلاصه شده است.

جدول ۴-۲- زمان مورد نیاز برای هر الگوریتم ها

		Time consuming	
		Method	Time
statistics approach	approach	Stuffer and Grimson Methods	$O(K*M*N*d*d*d)$ ۳,۳۷۱۹۹۶
		Gaussian background modeling	$O(M*N*d*d*d)$ ۲,۶۵۴۱۱۲
		Kernel background model	$O(K*M*N*H)$ ۴,۲۸۰۴۱۱
Non statistics approach	Average methods	BMIT	$O(1)$ ۰,۰۶۴۸۳۲
		IBM	$O(1)$ ۰,۰۶۴۵۵۳
		LTABM	$O(M*N)$ ۱,۳۶۷۹۰۰
		MABM	$O(M*N)$ ۲,۰۷۸۹۲۸
		RGABM	$O(M*N)$ ۱,۳۶۶۱۶۳
		PCA modeling	$O((M * N)^2)$ -
		Median background modeling	$O(M*N)$ ۲,۰۰۳۴۵۶

دقت

آخرین معیار مورد ارزیابی برای انتخاب الگوریتم دقت الگوریتم در تمایز اشیا از صحنه می باشد این معیار باید تحت شرایط مختلف محیطی (درونی و بیرونی) مورد ارزیابی قرار بگیرند. نتایج بررسی برای

محیط داخلی در جدول ۳-۴ نشان داده شده است. بهترین نتیجه در خطای FN متعلق به الگوریتم $BMIT$ می باشد این خطا برای سایر الگوریتم ها قابل قبول نمی باشد. اگر این خطا بالا باشد نشانه ی این است که قسمتی از شی توسط الگوریتم به اشتباه به عنوان پس زمینه در نظر گرفته شده است؛ به همین دلیل اشیا ناقص می باشند پس کاربرد هایی که اشیا را به صورت کامل لازم دارد مانند $image$ $indexing$ نمی توان مورد استفاده قرار داد. بهترین عملکرد در مورد خطای FP متعلق به الگوریتم $stuffer$ می باشد علت به وجود آمدن این نوع خطا نویز ها و تغییرات ناگهانی در شرایطی محیط می باشد بنابراین در شرایطی که نویز در محیط بالا باشد می توان از این الگوریتم استفاده نمود.

جدول ۳-۴ بررسی دقت چند الگوریتم مدل سازی پس زمینه در شرایط indoor

Accuracy								
Stuffer and Grimson Methods	Frame ۱		Frame ۲		Frame ۳		Frame ۴	
	FN	FP	FN	FP	FN	FP	FN	FP
		۰	۳,۷۷۶۰۵-۰۰۴	۰	۰,۰۱۱۳	۰,۷۵۴۸	۰,۰۲۴۲	۰,۷۳۳۷
BMIT	۰	۰,۰۰۸۰	۰	۰,۰۲۵۳	۰,۱۴۱۸	۰,۰۷۳۹	۰,۲۰۹۵	۰,۰۶۸۴
IBM	۰	۰,۰۰۷۷	۰	۰,۰۰۸۹	۰,۱۴۴۲	۰,۰۴۱۲	۰,۲۲۶۷	۰,۰۳۶۳
LTABM	۰	۶,۵۱۰۴۵-۵	۰	۰,۰۲۹۵	۰,۶۹۶۲	۰,۰۲۶۷	۰,۸۷۵۴	۰,۰۴۳۴
MABM	۰	۰,۰۰۴۱	۰	۰,۰۱۹۴	۰,۱۲۹۱	۰,۰۶۱۵	۰,۲۱۳۵	۰,۰۷۸۱
RGABM	۰	۲,۳۴۳۷۵-۰۰۴	۰	۰	۰,۷۷۷۲	۰,۰۱۸۳	۰,۸۶۹۰	۰,۰۱۲۳

میزان خطای الگوریتم را برای شرایط محیطی بیرونی مورد ارزیابی قرار داده و نتایج آن را در جدول ۴-۴ نشان داده شده است با توجه به نتایج مندرج در جدول مشاهده می شود که درصد خطای FN برای الگوریتم ها مندرج در جدول ۴-۴ بالا می باشد و این نشان می دهد که این الگوریتم ها

برای کاربرد های که نیاز به شی کامل دارند مناسب نمی باشند. اما در این میان الگوریتم *stuffer* خطای قابل قبولی را از نظر خطای *FP* دارد که این نشان می دهد این الگوریتم برای شرایطی که محیط دارای تغییرات می باشد دارای کارایی مناسبی می باشد.

جدول ۳-۴- میزان خطای چندی از الگوریتم را برای شرایط بیرونی نشان می دهد.

Accuracy outdoor								
Stuffer and Grimson Methods	Frame ۱		Frame ۲		Frame ۳		Frame ۴	
	FP	FN	FP	FN	FP	FN	FP	FN
	۰,۰۰۷۹	۰,۹۷۴۸	۰,۰۱۰۴	۰,۷۹۱۱	۰,۰۰۶۸	۰,۹۴۱۶	۰,۰۰۳۹	۰
BMIT	۰,۱۹۸۲	۰,۱۲۷۵	۰,۲۳۱۶	۰,۰۹۵۰	۰,۲۴۳۶	۰,۰۶۶۴	۰,۱۷۰۷	۰
IBM	۰,۱۵۶۵	۰,۱۶۱۹	۰,۱۷۵۲	۰,۰۹۸۴	۰,۱۸۷۴	۰,۰۷۰۱	۰,۱۳۹۲	۰
LTABM	۰,۰۱۷۴	۰,۵۶۹۳	۰,۰۳۹۷	۰,۵۹۳۱	۰,۰۲۷۲	۰,۶۹۲۲	۰,۰۱۸۷	۰
MABM	۰,۱۴۲۲	۰,۱۵۵۱	۰,۲۱۳۳	۰,۱۰۴۰	۰,۱۸۶۰	۰,۱۸۶۰	۰,۱۲۴۸	۰,۱۲۴۸
RGABM	۰,۰۰۶۶	۰,۸۱۸۷	۰,۰۱۲۱	۰,۷۲۲۵	۰,۰۱۲۲	۰,۸۲۶۵	۰,۰۰۷۲	۰

۴-۴- حذف سایه

در هنگام جدا کردن اشیا متحرک از درون فریم های ویدیویی معمولا سایه اشیا متحرک به عنوان شی در نظر گرفته می شوند. البته این نتیجه نیز مورد انتظار می باشد زیرا که سایه دارای اختلاف قابل توجه با صحنه می باشد به همین دلیل اشتباها به عنوان شی برچسب گذاری می شود. بهر حال یک سگمنت بندی مناسب می توان سایه ها را از اشیا جدا کرد. به منظور حذف سایه، از الگوریتم ارائه شده در [۵] تشریح می کنیم. ، زمانی که سایه ای بر روی یک شی می افتد شدت سطح خاکستری تغییر می کند اما مقدار رنگ نرمال شده تغییر نمی کند.

$$\frac{R_s}{R_s + G_s + B_s} \cong \frac{R}{R + G + B}, \frac{G_s}{R_s + G_s + B_s} \cong \frac{G}{R + G + B}, \frac{B_s}{R_s + G_s + B_s} \cong \frac{B}{R + G + B} \quad (۳۲-۴)$$

$$I_s(x, y) = \alpha I(x, y) \quad (۳۳-۴)$$

$I(x, y)$ مقدار شدت سطح خاکستری را در نقطه (x, y) نشان می دهد. بعد از تشخیص سایه ها آنها را حذف می کنیم.

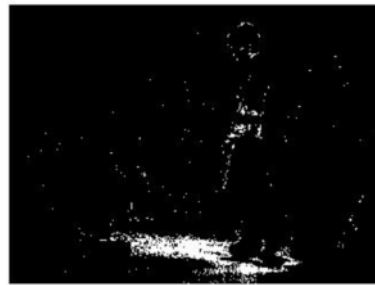
الگوریتم دیگری در [۲] برای حذف سایه بیان شده است این الگوریتم بیان می کند، زمانی که سایه ای بر روی یک سطح می افتد مولفه ی درخشندگی بیشتر از مولفه ی رنگی نور تحت تاثیر قرار می گیرد به عبارتی مولفه رنگی نور تغییر نمی کند. به همین دلیل لازم است به جای نمایش رنگی RGB از نمایش رنگی استفاده کنیم که مولفه ی رنگی را از درخشندگی جدا کند. یکی از نمایش که به صورت مطلوبی این کار را انجام می دهد، HSV می باشد SoH مولفه های رنگی و V میزان درخشندگی را نشان می دهد با بکاربردن این نمایش و با استفاده از رابطه زیر می توان سایه را از تصویر جدا کرد.

$$shadow_{x,y}^k = \begin{cases} 1 & \text{if } D_H < \tau_h, |I_{x,y_s}^k - B_{x,y_s}^k| < \tau_s, v_a \leq \frac{I_{x,y_v}^k}{B_{x,y_v}^k} \leq v_b \\ 0 & \text{otherwise} \end{cases} \quad (۳۴-۴)$$

$$D_H = \min \left\{ \left| I_{x,y_h}^k - B_{x,y_h}^k \right|, 360 - \left| I_{x,y_h}^k - B_{x,y_h}^k \right| \right\} \quad (۳۵-۴)$$

در عبارات فوق $I_{x,y}^k$ پیکسل واقع در موقعیت (x, y) فریم k ام و $B_{x,y}^k$ پیکسل واقع در موقعیت (x, y) پس زمینه را در فریم k ام نشان می دهد زیر اندیس های h, s, v اشاره به مولفه های H, S, V دارد. τ_h, τ_s, v_a, v_b اعدادی در بازه $[0, 1]$ می باشند که با استفاده از آن می توان کران مقادیر را در رابطه (۳۴-۴) کنترل کرد.

برای بررسی کارایی، این الگوریتم را بر روی چند فریم ویدیویی اعمال کردیم که نتایج در شکل ۴-۴۰ مشاهده می شود. همانطور که در این چند نمونه مشاهده می شود علاوه بر سایه اشیا، مقداری نویز وجود دارد. برای مقابله با این پدیده در بخش بعدی به معرفی روش های مقابله با نویز خواهیم پرداخت.



ب) شی استخراج شده با الگوریتم های استخراج پس زمینه

الف) سایه استخراج شده

شکل ۴-۴۰ تصویر اشیا استخراج شده از صحنه به همراه سایه آن یا استفاده از رابطه (۴-۳۴)

الگوریتم دیگری که ارائه شده است فرض می کند که سایه ها در واقع درصدی از نور تابیده شده به پس زمینه را بلوک می کنند به همین دلیل ناحیه که سایه دارد تاحدی تاریک تر می شود. با استفاده از این خاصیت اگر فریم جدیدی وارد شود می توان سایه ها را تشخیص داد. برای تعیین نقاط کاندیدا برای سایه می توان از *normalized cross correlation* که در ادامه توضیح داده می شود استفاده می کرد.

فرض کنید $B(i,j)$ پیکسل پس زمینه را در موقعیت (i,j) نشان می دهد و $I(i,j)$ پیکسل تصویر را در موقعیت (i,j) نشان می دهد. برای هر پیکسل (i,j) که متعلق به اشیا باشد متغیر $T_{i,j}$ را به صورت زیر تعریف می کنیم.

$$T_{i,j} = I(i+n, j+m), -N \leq n \leq N, -N \leq m \leq N \quad (۴-۳۶)$$

آنگاه NCC با استفاده از رابطه زیر محاسبه می شود.

$$NCC(i, j) = \frac{ER(i, j)}{E_B(i, j)E_{T_{ij}}} \quad (37-4)$$

که

$$ER(i, j) = \sum_{-N}^N \sum_{-N}^N B(i+n, j+m) T_{i,j}(n, m) \quad (38-4)$$

$$E_B(i, j) = \sqrt{\sum_{-N}^N \sum_{-N}^N B(i+n, j+m)^2} \quad (39-4)$$

$$E_{T_{ij}} = \sqrt{\sum_{-N}^N \sum_{-N}^N T_{ij}(n, m)^2} \quad (40-4)$$

در نواحی که سایه می باشد باید مقدار NCC بالا (نزدیک به یک) مقدار انرژی کمتر باشد بنابراین برای تعیین پیکسل های کاندیدا می توان از رابطه (41-4) استفاده کرد.

$$NCC(i, j) \geq l_{ncc} \text{ and } E_{T_{ij}} \leq E_B(i, j) \quad (41-4)$$

در رابطه فوق l_{ncc} میزان حد آستانه برای تعیین پیکسل های کاندیدا است. کارایی این الگوریتم وابسته به این پارامتر می باشد اگر مقدار آن را کوچک در نظر بگیریم قسمتی از اشیا را به عنوان پس زمینه در نظر می گیرد از طرفی اگر مقدار حد آستانه را بزرگ بگیریم مقداری از سایه نادیده گرفته می شود. الگوریتم ارائه شده پارامتر N را مورد ارزیابی قرار دادیم همانطور که در تصاویر این شکل 4-- 41 مشاهده می کنید مقدار N بر کارایی اثر جدی دارد. در این قسمت به معرفی دو روش از روش های موجود برای حذف نویز پرداختیم برای مطالعه بیشتر می توانید به مراجع [24, 25, 26, 27, 28, 29] مراجعه کنید.



ب) شی استخراج شده با استفاده



الف) سایه تشخیص داده شده یا $N=3$



ث) سایه تشخیص داده شده یا $N=5$



پ) سایه تشخیص داده شده یا $N=7$

شکل ۴-۴ سایه استخراج شده با استفاده از تصویر الگوریتم NCC

۴-۵- حذف نویز

همانطور که در آزمایش ها مراحل قبلی دیدیم کارایی الگوریتم های آماری مناسب می باشد اما این روش ها نسبت به نویز حساس می باشند. این نویز ها بر روی حجم بالای محاسبات در الگوریتم ها تاثیر می گذارند و در نتیجه باعث به وجود آمدن خطا در خروجی می شوند. به این منظور حذف نویز یکی از قسمت های اساسی در سیستم ها تشخیص اشیا متحرک می باشد. برای حذف نویز چندین دسته الگوریتم وجود دارد که در ادامه به معرفی هر یک می پردازیم.

- عملگر های ریخت شناسی

- (CCL) Connected component labeling and area filtering
- فیلتر میانه و میانگین

۴-۵-۱- حذف نویز با استفاده از الگوریتم های ریخت شناسی

عملگر های ریخت شناسی معمولا بر روی تصاویر باینری اعمال می شوند و دارای یک عنصر ساختاری و یک عملگر مجموعه ای (اجتماع، اشتراک و غیره) می باشند. معمولا این عنصر ساختاری دارای ابعادی به صورت 3×3 است که مرکز آن در مرکز عنصر ساختاری می باشد. عنصر ساختاری بر روی تصویر حرکت داده می شود در صورتیکه شرط عملگر مجموعه برآورده می شود (ضرب نقطه به نقطه عناصر از یک مقدار پیش فرض تجاوز کند) مقدار پیکسلی را که در مرکز عنصر ساختاری قرار می گیرد به مقدار پیش فرض تغییر (۰ یا ۱) می کند. برای عملگر های پایه ای در الگوریتم های ریخت شناسی فرض می کنیم مقادیر عنصر ساختاری برابر ۱ برای foreground و ۰ برای عناصر پس زمینه باشد [۱۵].

هنگام اندازه گیری اشیا درون تصویر، گاهی لازم است تا اشیا را ساده کنیم مثلا حفره های درون اشیا را پر کنیم یا زائده های برآمده را حذف کنیم. حذف پیکسل های مرزی اشیا را عمل erosion (سایش) می نامیم. منظور از پیکسل های مرزی پیکسل هایی از شی می باشند که در همسایگی آن پیکسل هایی از پس زمینه باشد. عمل سایش پیکسل های مرزی اشیا را به عنوان پیکسل پس زمینه برچسب می زند و در نتیجه شی کوچکتر می شود. عمل گسترش بر عکس باعث بزرگتر شدن شی می شود زیرا در این عمل پیکسل های مرزی پس زمینه را به عنوان پیکسل شی در نظر می گیرند. عمل سایش و گسترش را گاهی shrinking و growing نیز می گویند.

اگر تعدادی حفره پس زمینه در شی وجود داشته باشد. می توانیم آنها را با استفاده از عمل گسترش پر کنیم اما عمل گسترش باعث بزرگتر شدن اشیا می شود اگر به تعداد عمل گسترشی که بر روی تصویر اعمال می کنیم عمل سایش اعمال کنیم اشیا تقریبا دوباره به اندازه اولیه خود باز خواهند

گشت بدون اینکه حفره ها برگردانده شوند. توجه کنید عمل سایش اشیا معادل عمل گسترش پس زمینه می باشد و برعکس. بنابراین اگر تعداد پیکسل های پس زمینه کمتر از تعداد پیکسل های شی باشد و بخواهیم عمل گسترش یا سایش را بر روی اشیا انجام دهیم می توانیم متناظر این عمل ها یعنی اعمال سایش و گسترش را بر روی پس زمینه ها اعمال کنیم.

ماسک ها یا عناصر مورد استفاده می تواند به صورت های همسایگی ۸ تایی یا ۴ تایی باشند اگر اندازه ماسک ها 3×3 باشد در هر بار عمل سایش یا گسترش به اندازه یک پیکسل به مرز اشیا افزوده یا کاسته شود. در صورتی که اندازه ماسک ها را بزرگتر بگیریم اندازه مرزی که افزایش یا کاهش می یابد بزرگتر می شود.

سایش Erosion

همانطور که از نام این عملگر مشخص می شود این عملگر موجب سایش مرز می شود. یکی از عناصر ساختاری را در شکل ۴-۴۲ می بینید این عملگر را بر روی تصویر ورودی اعمال می کنیم اگر همه مقادیر در زیر عنصر ساختاری برابر ۱ باشد آن عنصر متعلق به foreground می باشد به همین دلیل مقدار آن را تغییر نمی دهیم اما اگر یکی از نقاط همسایه پیکسل مرکز برابر ۰ باشد (در این تصویر همسایگی ۴ در نظر گرفته شده است.) مقدار پیکسل مربوطه را برابر ۰ قرار می دهیم. همانطور که در تصاویر مشاهده می کنید این عملگر مرز اشیا را منقبض می کند و باعث گسترش حفره در تصویر می شود.

این عمل می تواند بر روی یک شی یا همه اشیا درون تصویر اعمال شود روش کار به این ترتیب است که با استفاده از عنصر پایه ای تمام تصویر را اسکن می کنیم در هر مرحله مرکز کرنل را بر روی یک پیکسل قرار می دهیم اگر تمام نقاط واقع شده در زیر ماسک برابر ۱ باشند پیکسل واقع شده در مرکز کرنل به عنوان پیکسل شی (درون تصویر جدید که نتیجه عمل را نگه می دارد) و در غیر این صورت

به عنوان پیکسل پس زمینه علامت گذاری می شود. پس قسمتی از اشیا که در تصویر جدید باقی می ماند پیکسل هایی هستند که اگر مرکز ماسک را در آن موقعیت قرار دهیم کاملا درون شی قرار خواهند گرفت و هیچ کدام از پیکسل های واقع در زیر ماسک متعلق به پس زمینه نخواهد بود. اندازه نواری که از اشیا بریده می شود تقریبا برابر نصف طول یا عرض ماسک خواهد بود.

عمل سایش پیکسل های ناحیه ای R در تصویر I با استفاده از عنصر ساختاری A را به صورت $R' = R \ominus A$ نشان می دهیم. معمولا ماسک های مورد استفاده متقارن می باشند اما می توانند به صورت غیر متقارن نیز باشند اگر بیشتر عنصر ساختاری به سمت راست متمایل باشند آنگاه عمل سایش باعث می شود مرز های سمت راست بیشتر از سمت چپ سایش یابد. مرکز عنصر ساختاری می تواند در هر کدام از موقعیت ها قرار بگیرد بنابراین الزامی وجود ندارد که مرکز در میانه عنصر ساختاری قرار بگیرد.

گسترش Dilation

این عملگر دوگان عملگر erosion می باشد بر خلاف عملگر erosion این عملگر بر روی پیکسل های پس زمینه عمل می کند یک عملگر بر روی تصویر ورودی اعمال می شود و همانطور که مشاهده می کنید این عملگر باعث گسترش مرز اشیا می شود و حفره های درون تصویر را پر می کند.

این عمل می تواند بر روی یک شی یا همه اشیا درون تصویر اعمال شود روش کار به این ترتیب است که با استفاده از عنصر پایه ای تمام تصویر را اسکن می کنیم در هر مرحله مرکز کرنل را بر روی یک پیکسل قرار می دهیم اگر تمام نقاط واقع شده در زیر ماسک برابر یا \circ باشند آن پیکسل واقع شده در مرکز کرنل به عنوان پیکسل پس زمینه (درون تصویر جدید که نتیجه عمل را نگه می دارد) برچسب گذاری می شود در غیر این صورت به عنوان پیکسل شی علامت گذاری می شود. پس لایه جدیدی

که به شی اضافه و باعث گسترش شی می شود، پیکسل هایی هستند که اگر مرکز ماسک بر روی آن قرار گیرد ماسک با شی در حداقل یک پیکسل همپوشانی خواهد داشت.

عمل گسترش پیکسل های ناحیه ای R در تصویر I با استفاده از عنصر ساختاری A را به صورت $R' = R \oplus A$ نشان می دهیم.

عملگر *opening* و *closing*

همانطور که می دانیم عمل سایش و گسترش به ترتیب باعث افزایش و کاهش مرز اشیا می شود به همین دلیل بعد از عمل سایش لازم است تا گسترش انجام شود به این عمل *opening* می گویند عمل *opening* برجستگی های باریک و اتصالات ضعیف بین اشیا را از بین می برد بدون اینکه تغییر خاصی در مرز اشیا به وجود بیاید. این عملگر با استفاده از نماده ها، به صورت زیر نشان داده می شود.

$$R' = (R \ominus A) \oplus A \quad (42-4)$$

از طرفی می دانیم عمل گسترش باعث بزرگتر شدن اشیا می شود بنابراین لازم است تا بر روی آن عمل سایش را اعمال کنیم به این عمل، عمل *closing* می گوئیم عمل *closing* حفره ها و فرورفتگی هایی اشیا را بدون تغییر خاص درون اشیا پر می کند. نمایش ریاضی این عمل به صورت زیر می باشد.

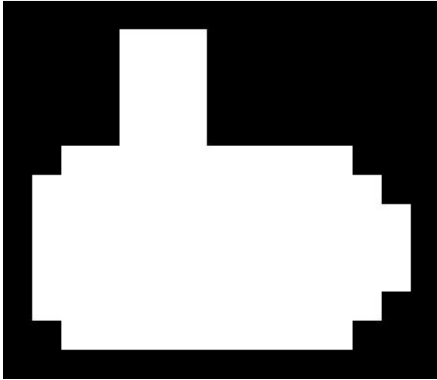
$$R' = (R \oplus A) \ominus A \quad (43-4)$$

در ادامه اثر هر کدام از این روش ها بر روی اشیا استخراج شده برای حذف نویز نشان داده می شود. همانطور که در تصاویر شکل ۴-۵۸ مشاهده می شود عمل سایش، نویز های ضربه را کاهش می دهد اما باعث افزایش حفره ها در درون اشیا می شود از طرفی عمل گسترش، این حفره ها را کوچک می کند اما نویز ها را تقویت می کند در حالی عملیات *opening* و *closing* نتیجه بینابینی را بدست می آورد.

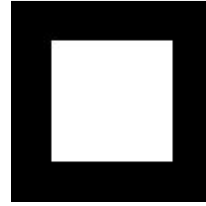
با استفاده متوالی از عملگر سایش^۳ و گسترش^۴ بعضی از نویزها (مانند نویز فلفل نمکی) از تصویر حذف می شود علاوه بر آن در بعضی از مواقع باعث از بین رفتن اتصالات ضعیف در تصویر نیز می شود.

^۳ erosion

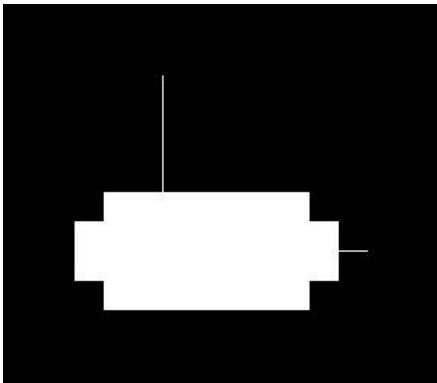
^۴ dilaton



شکل ۴-۴۳ تصویر اصلی

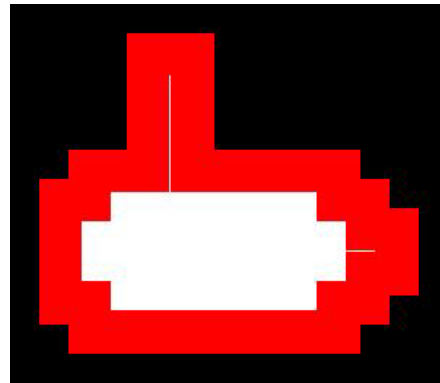


شکل ۴-۴۲ کرنل مورد استفاده برای سایش



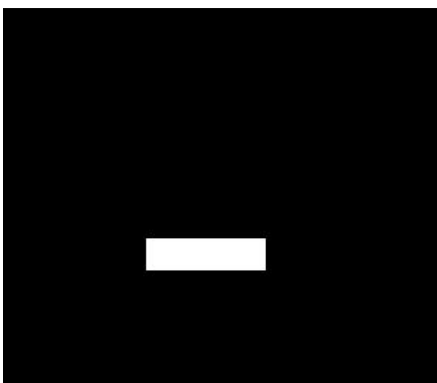
شکل ۴-۴۵ تصویر ۳۹ بعد از یک بار سایش با کرنل

۳۸

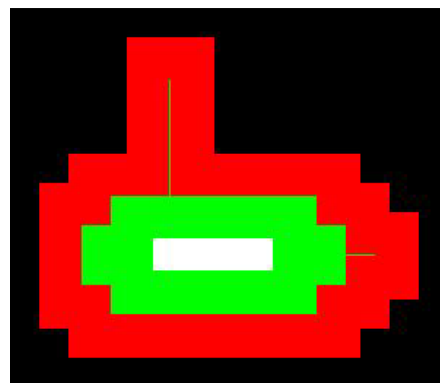


شکل ۴-۴۴ قسمت های قرمز نواحی ساییده شده

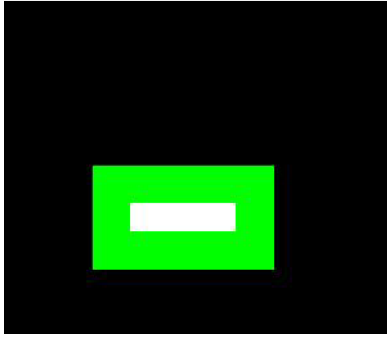
تصویر اصلی را نشان می دهد



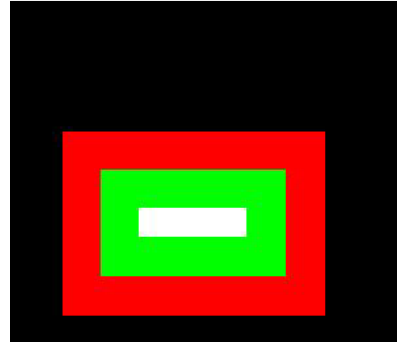
شکل ۴-۴۷ بعد از دو بار سایش



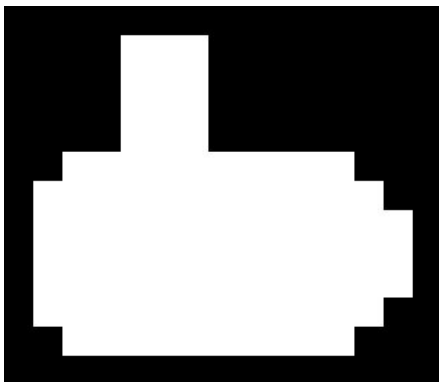
شکل ۴-۴۶ قسمت سبزر بار دوم سایش شده



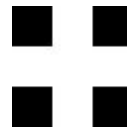
شکل ۴-۴۹ گسترش تصویر با کرنل ۳۹ قسمت سبز قسمت افزوده شده را نشان.



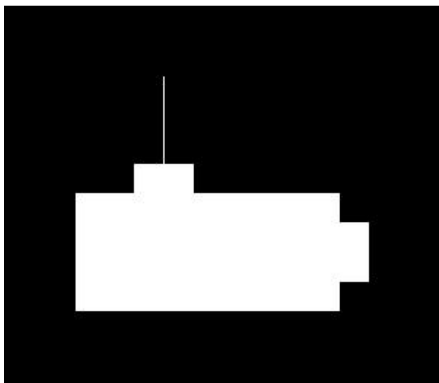
شکل ۴-۴۸ قسمت قرمز افزوده شده برای بار دوم می باشد



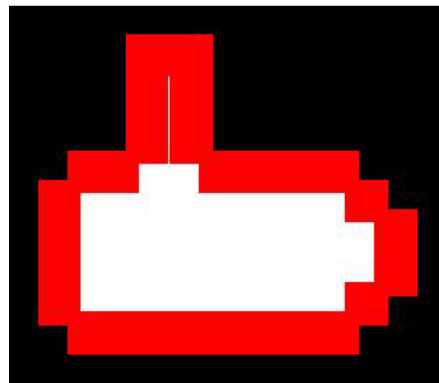
شکل ۴-۵۱ تصویر اصلی



شکل ۴-۵۰ تصویر کرنل مورد استفاده

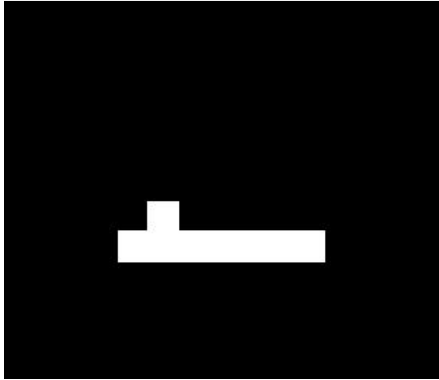


شکل ۴-۵۳ تصویر سایه شده با استفاده از کرنل ۴۶

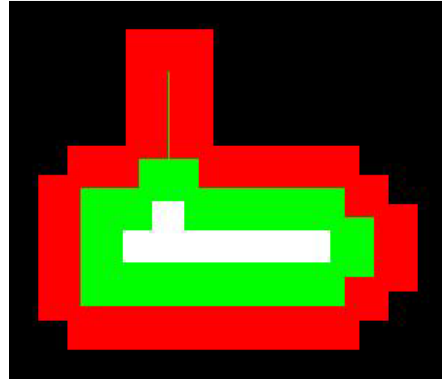


شکل ۴-۵۲ قسمت قرمز سایه شده را نشان

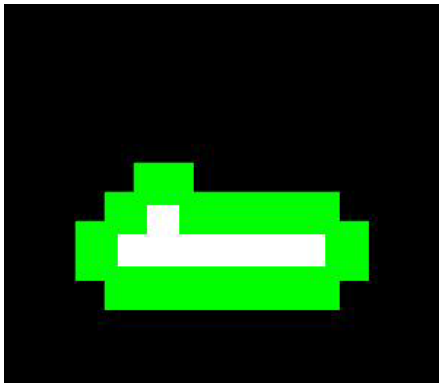
می دهد.



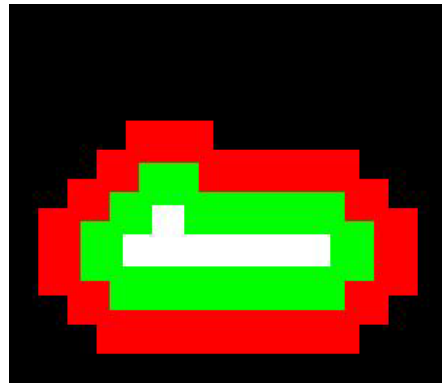
شکل ۴-۵۵ تصویر بعد از دو بار سایده شده



شکل ۴-۵۴ قسمت سبز در مرحله دوم سایده شده



شکل ۴-۵۷ بعد از گسترش با استفاده از کرنل ۴۶

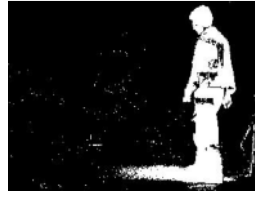


شکل ۴-۵۶ قسمت قرمز بعد گسترش برای بار دوم

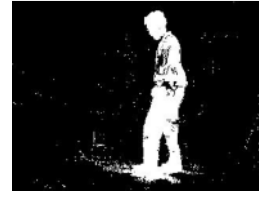
تصویر اصلی



ث) تصویر اصلی فریم ۱۰



ب) تصویر اصلی فریم ۱۰۰

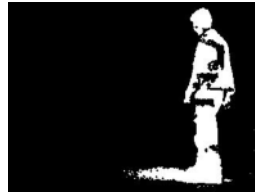


الف) تصویر اصلی فریم ۲۵۰

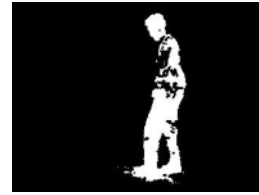
سایش



ج) حذف نویز با سایش ۱۰



ث) حذف نویز با سایش ۱۰۰



ت) حذف نویز با سایش فریم ۲۵۰

گسترش



خ) حذف نویز با گسترش ۱۰



ح) حذف نویز با گسترش ۱۰۰

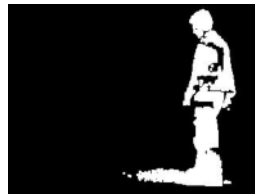


چ) حذف نویز با گسترش ۲۵۰

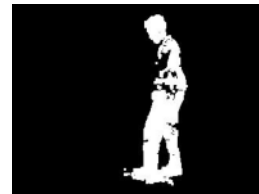
باز کردن



ک) حذف نویز با بازکردن ۱۰



ذ) حذف نویز با بازکردن ۱۰۰



د) حذف نویز با بازکردن ۲۵۰

بستن



م) حذف نویز با بستن ۱۰



ل) حذف نویز با بستن ۱۰۰



گ) حذف نویز با بستن ۲۵۰

شکل ۴-۵۸ استفاده از روش های ریخت شناسی برای حذف نویز

۴-۵-۲ - Connected Component Labeling (CCL) and Area Filter

الگوریتم (CCL) پیکسل ها را در گروه هایی دسته بندی می کند بدین ترتیب که پیکسل هایی که همسایگی هر قرار دارند به عنوان یک مولفه در نظر گرفته می شوند در ادامه گام هایی که لازم است تا طی شود تا مولفه های متصل پیدا شوند تشریح می شود [۸،۳۱].

- تصویر را به صورت raster پویش می کنیم.
- اگر یک پیکسل متعلق به پس زمینه باشد.
 - اگر یکی از پیکسل های سمت چپ، بالا چپ، بالا و یا بالا-راست دارای برچسب باشد آن پیکسل را با این برچسب علامت بزن.
 - در صورتی که خانه های ذکر شده دارای برچسب های متفاوتی بودند یک پیکسل را با یکی از برچسب ها برچسب گذاری کن و سایر خانه ها را با برچسب یکسان برچسب گذاری کنید.
 - در صورتی که هیچ کدام از خانه ها دارای برچسب نباشند به این پیکسل برچسب جدید منتسب کن.
- همه ی پیکسل ها با توجه به قوانین ذکر شده در گام دوم پویش می شوند.
- گروه هایی که دارای برچسب همسان می باشند متعلق به یک کلاس می باشند به همین دلیل آنها را با هم ادغام می کنیم.
- تصویر یک بار دیگر پویش می شود تا گروه ها را برچسب جدید بزند.

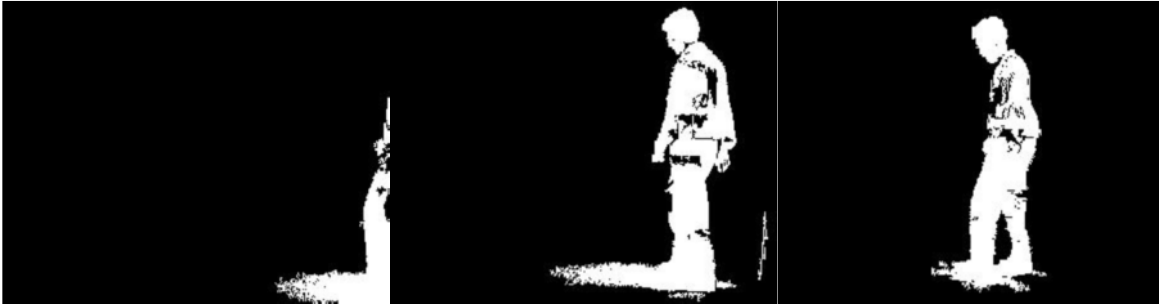


(ب) تصویر ورودی

(الف) تصویر خروجی الگوریتم CCL

شکل ۴-۵۹ بررسی عملکرد الگوریتم CCL بر روی تصویر شامل چند شی

مساحت هر کدام از اشیا بعد از برچسب گذاری محاسبه می شود با توجه به مقدار میانگین اندازه اشیا متحرک مقدار حد آستانه ای انتخاب می شود. این حد آستانه برای حذف نواحی خیلی کوچکتر که احتمالاً نویز می باشد، استفاده می شود. مقدار بهینه حد آستانه، با چندین بار آزمایش بر روی ویدیو های Indoor و outdoor انتخاب می شود. علاوه بر مساحت، اندازه طول و عرض جعبه در بر گیرنده شی و مرکز آن و همچنین متوسط تعداد اشیا در صحنه، در نظر گرفته می شود، ترکیب این معیارها تاثیر زیادی بر کارایی دارد. عملکرد این الگوریتم برای حذف نویز بر روی تصاویر در شکل ۴-۶۰ مشاهده می شود. همانطور که از تصاویر درج شده در این جدول مشاهده می کنید میزان دقت آن روش بالاتر می باشد. برخلاف روش های قبلی (که در آن کاهش خطا FN با افزایش خطا FP و بالعکس همراه است) این روش خطای FP را کاهش می دهد بدون این که در FN اثر بگذارد.



پ) فریم شماره ۱۰

ب) فریم شماره ۱۲۸

الف) فریم شماره ۲۵۰

شکل ۴-۶ حذف نویز با استفاده از الگوریتم CCL

۴-۵-۳- حذف نویز با استفاده از الگوریتم میانه و میانگین

الگوریتم نهایی که به منظور کاهش خطا و نویز استفاده می شود فیلتر میانه و میانگین می باشد. عملکرد فیلتر میانگین به این صورت است که یک ماسک (مثلا ماسک 3×3) بر روی تصویر می لغزد و به جای عنصر که در مرکز ماسک قرار دارد مقدار میانگین مقادیر زیر ماسک درج می شود. عملکرد این الگوریتم بر روی تصاویر در شکل ۴-۶۱ مشاهده می شود. عملا این فیلتر کارایی ندارد زیرا میانگین به شدت تحت تاثیر نقاط اکستریم قرار می گیرد مثلا فرض کنید در زیر ماسک به ابعاد 3×3 که شامل ۹ عنصر می باشد اگر ۸ نقطه واقع در آن برابر مقدار ۰ و یک نقطه برابر ۲۵۵ باشد منطقی است اگر فرض کنیم این مقدار به علت نویز به وجود آمده باشد اما زمانی که میانگین محاسبه شود به علت بزرگ بودن مقدار میانگین فراتر از ۰ می شود به همین دلیل بر روی نتیجه نهایی اثر نامطلوب می گذارد دومین فاکتور که بر کارایی این الگوریتم اثر می گذارد اندازه فیلتر می باشد هر چه اندازه این فیلتر بزرگتر باشد تصویر هموار تر و محو تر می شود هر چه اندازه فیلتر کوچکتر باشد نتیجه نهایی تحت تاثیر مقادیر نویز قرار می گیرد.



الف) فریم شماره ۲۵۰

ب) فریم شماره ۵۰

پ) فریم شماره ۱۰

شکل ۴-۶۱ نتایج عملکرد فیلتر میانگین بر روی تصاویر به منظور حذف نویز

برای مقابله با این مشکل (تحت تاثیر قرار نگرفتن با مقدار نویز) از فیلتر میانه استفاده می شود در این روش به جای استفاده از یک مقدار از مقدار میانه پیکسل های مجاور آن استفاده می شود مقدار میانه تحت تاثیر مقدار نویز قرار نمی گیرد به همین دلیل توقع داریم کارایی الگوریتم بالاتر رود. نتایج این الگوریتم را در شکل ۴-۶۲ مشاهده می شود کارایی این الگوریتم به مراتب بهتر از الگوریتم میانگین می باشد در واقع فیلتر میانه یکی از بهترین الگوریتم ها به منظور مقابله با نویز ضربه می باشد. نقطه ضعف این الگوریتم را می توان در زمان محاسبات بالا برای پیدا کردن میانه دانست که برای مقابله با این مشکل از روش های استفاده می شود که به جای محاسبه مقدار درست میانه از مقدار تقریبی آن استفاده کرده و به این روش زمان محاسبات را کاهش می دهد.



الف) فریم شماره ۲۵۰

ب) فریم شماره ۱۲۸

پ) فریم شماره ۱۰

شکل ۴-۶۲ نتایج عملکرد فیلتر میانه بر روی تصاویر به منظور حذف نویز

فصل پنجم

جدا کردن اشیا از فریم های ویدیویی

۵-۱- مقدمه

همانطور که در فصل دوم اشاره شد، رسانه‌های ویدیویی معمولاً حجم زیادی از داده‌ها را تولید می‌کنند به عنوان مثال در تلویزیون‌های HDTV، در یک ثانیه حجم داده تولید شده از مرز یک گیگا بیت فراتر می‌رود. حتی اگر به کیفیت پایین تری از ویدیو نیاز داشته باشیم باز هم با حجم بالایی از داده مواجه خواهیم شد. این حجم عظیم داده مشکلاتی از قبیل فضای لازم برای ذخیره کردن اطلاعات و پهنای باند کافی برای انتقال آن به همراه دارد.

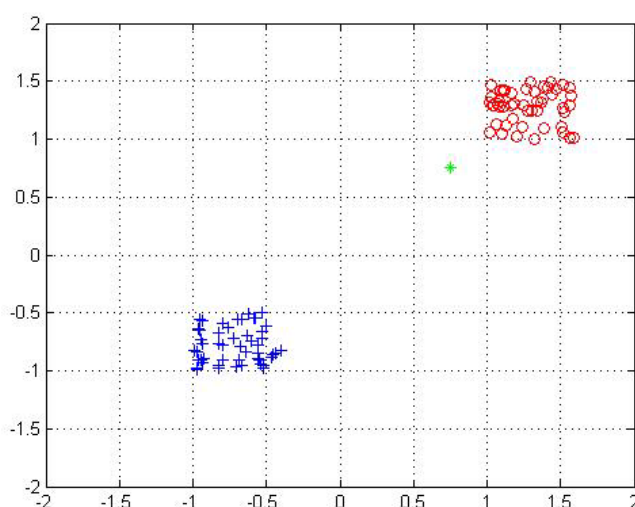
حجم عمده یک فریم را پس زمینه تشکیل می‌دهد که در فریم‌های متوالی مشترک می‌باشد، آنچه تغییر می‌کند اشیا متحرک در صحنه می‌باشند و در پس زمینه تغییرات جزئی وجود دارد که قابل توجه برای چشم انسان نمی‌باشند (مانند امواج دریا یا لرزش‌های آرام شاخ و برگ درختان) بنابراین می‌توان با جدا کردن اشیا از پس زمینه و مدل کردن پس زمینه از ذخیره کردن حجم عمده فریم که مشترک می‌باشد، صرفه نظر کرد و به این طریق به حجم بالایی از فشرده سازی در ویدیو دست یافت.

در فصل سوم به معرفی راهکار رایج برای مدل کردن پس زمینه و استخراج اشیا از پس زمینه پرداخته ایم. در راهکار‌های ارائه شده قبلی دیدیم که روش‌های آماری علی‌رغم کارایی در جداکردن دو گروه اشیا و پس زمینه از هم، سه مشکل عمده ۱. بالا بودن حجم محاسبات برای بدست آوردن مدل، ۲. فرض داشتن مدلی خاص برای رفتار پس زمینه (در حالی که این فرض ممکن است با واقعیت فاصله معنا داری داشته باشد) و در نهایت ۳. در بسیاری از مواقع تعداد داده‌ی ناکافی برای آموزش مدل رنج می‌برد. به همین دلیل لازم است تا از رهیافتی دیگر مانند (رهیافت جداکننده) که این مشکلات را ندارد، استفاده شود.

در حالت ایده آل می‌توان داده‌ها را به دو کلاس پس زمینه و اشیا، که به صورت خطی جداپذیر می‌باشند، در نظر بگیریم. اگر مرز جداکننده تعیین شود، با استفاده از این تابع جداکننده، می‌توان اشیا

را از پس زمینه جدا کرد. این رهیافت نه تنها فرضی در مورد مدل تولید کننده داده‌ها در نظر نمی‌گیرد بلکه زمانی که تابع جداکننده بدست آمد، دیگر لازم نیست مدام آموزش ببیند. در واقع اینکه هر چند فریم یک بار تابع باید بروز شود بستگی به محیط و تغییرات آن دارد که در نتیجه با حجم محاسبات کمتری مواجه می‌شویم. از طرفی برای آموزش این رهیافت لازم نیست تعداد زیادی داده در دست داشته باشیم.

در شکل ۵-۱ چند نمونه داده داریم که با استفاده از دو ویژگی نمایش داده می‌شوند. داده‌های که با علامت (0) و (+) نشان داده شده است، داده‌های متعلق به دو کلاس مختلف می‌باشند. به این داده‌ها، که کلاس آن از قبل تعیین شده، داده‌های آموزشی می‌گویند. حال فرض کنید یک داده جدید (مثلا داده‌ای که با علامت (*) نشان داده شده است) در اختیار داریم. هدف کلاسه بندی این است که تعیین کند این داده به کدامیک از این دو کلاس (از پیش تعیین شده) تعلق دارد [۳].



شکل ۱-۵ داده‌های که با علامت (+) نشان داده شده‌اند، متعلق به کلاس اول. و داده‌های که با علامت (O) نشان داده شده‌اند، متعلق به کلاس دوم می‌باشند. حال یک نقطه جدید داریم که با علامت (*) نشان داده شده است، هدف کلاسه بندی این است که تعیین کند این داده متعلق به کدام کلاس می‌باشد.

می‌خواهیم تابعی را تعریف کنیم که به عنوان ورودی مقادیر (بردار ویژگی) داده جدید را دریافت و در خروجی تعیین کند که این داده متعلق به کدام کلاس می‌باشد. دو رهیافت معروف برای کلاسه بندی، رهیافت آماری و رهیافت جداکننده می‌باشند در ادامه تفاوت این دو رهیافت بیان می‌شود.

در رهیافت آماری، فرض می‌کنیم که داده‌های متعلق به هر کدام از کلاس‌ها، توسط یک مدل تولید می‌شود. در گام دوم با استفاده از داده‌های آموزشی پارامترهای این مدل تخمین زده می‌شوند. به عنوان مثال در داده‌های شکل ۱-۵ فرض می‌کنیم داده‌های متعلق به هر کدام از کلاس‌های + و O، توسط مدل گوسی تولید شده است. حال با استفاده از این داده‌های آموزشی، لازم است تا پارامترهای این مدل (میانگین و ماتریس کواریانس) با استفاده از روش‌هایی مانند maximum likelihood بدست آید. در این مثال دو بردار میانگین و دو ماتریس کواریانس (برای هر کدام از کلاس‌ها) وجود دارد. گام نهایی این است که، با استفاده از دو مدل، تعیین شود داده جدید متعلق به کدام کلاس می‌باشد. برای تعیین کلاس این داده، در رهیافت آماری، ابتدا احتمال تعلق داده جدید به هر کدام از کلاس‌ها محاسبه می‌شود. هر کلاس که احتمال تعلق داده به آن بزرگتر باشد، داده جدید به آن

کلاس منسوب می‌شود. احتمال تعلق داده جدید به کلاس k ام را می‌توان با رابطه (۵-۱) بدست آورد.

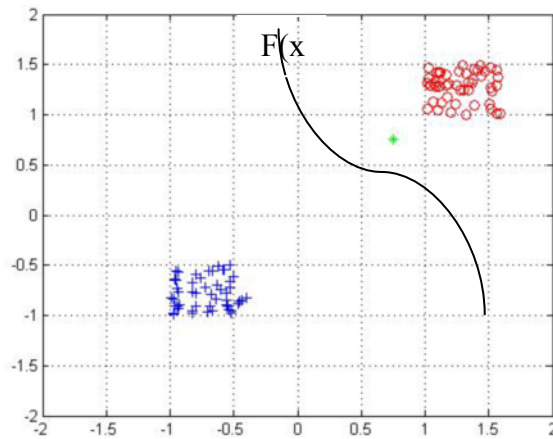
$$P(C_k | x = \text{newdata}) \quad (۵-۱)$$

مشکل روش آماری، اینست که در بعضی از مواقع توزیع فرضی (در مثال فوق فرض کردیم داده‌ها توسط توزیع نرمال تولید شده است) با توزیع واقعی داده اختلاف بسیار زیادی دارد، یا اینکه تعداد نمونه‌ها بسیار کمتر از حدی است که بتوان با کمک آن پارامترهای توزیع را (مثلاً برای توزیع نرمال این پارامتر میانگین و ماتریس کواریانس) با اطمینان انتخاب کرد.

در شکل ۵-۲ منحنی $f(x)$ بین داده‌های دو کلاس کشیده شده که مرزی را تعیین می‌کند که داده‌های متعلق به هر کدام از کلاس‌ها را از هم جدا می‌کند. تابع $F(x)$ به گونه‌ایست که اگر X متعلق به کلاس O باشد مقدار این تابع، عددی منفی می‌شود و اگر داده متعلق به کلاس دوم باشد مقدار خروجی این تابع، عددی مثبت خواهد شد.

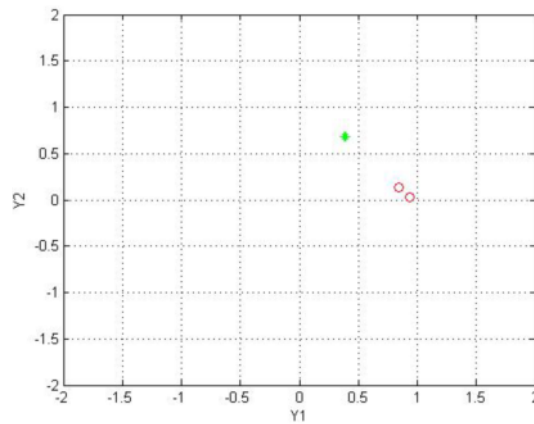
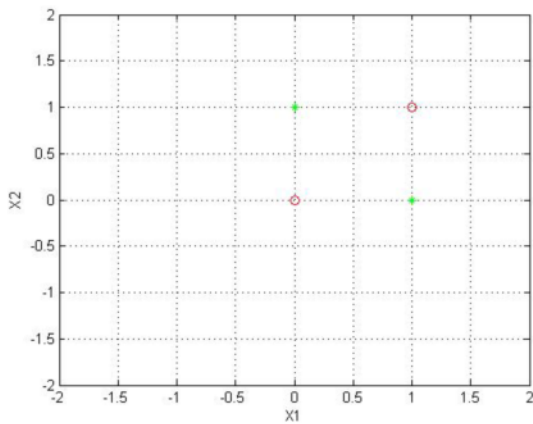
تابع جداکننده می‌تواند دارای اشکال متفاوتی باشد که ساده‌ترین شکل آن تابع خطی است به این معنا که کلاس‌ها را می‌توان با استفاده از خط یا یک ابر صفحه از هم جدا کرد. در بعضی از مواقع داده‌ها با یک تابع خطی جداپذیر نمی‌باشند، اما می‌توان آنها را به فضای جدیدی نگاشت داد که در این فضا به صورت خطی از هم جداپذیر باشند.

داده‌هایی که می‌توان آنها (یا انتقال یافته آنها به فضای جدید) را با استفاده از یک تابع خطی از هم جدا کرد، جداپذیر خطی می‌گویند.



شکل ۲-۵ خطی که بین داده‌های دو کلاس کشیده شده است و می‌توان با استفاده از تابع $F(x)$ آن را نشان داد، مرزی را تعیین می‌کند که داده‌های متعلق به هر کدام از کلاس‌ها را از هم جدا می‌کند.

به عنوان مثال در شکل ۳-۵ داده‌ها اگرچه در فضای X جداپذیر خطی نیستند، اما وقتی به فضای جدید Y نگاشت می‌شوند (شکل ۴-۵) به داده‌های خطی جداپذیر تبدیل می‌شوند.



شکل ۴-۵ فضای اولیه که داده‌ها در آن به صورت خطی جدا نمی‌باشند.

شکل ۳-۵ فضای جدید که داده‌ها در آن به صورت خطی جدا می‌باشند.

برای تعیین مرز جداکننده بین داده‌ها روش‌های فراوانی وجود دارد که در ادامه به معرفی بعضی از آنها می‌پردازیم.

۵-۲- توابع جداکننده

به تابعی که یک بردار را به عنوان ورودی دریافت می‌کند، و به عنوان خروجی، مقداری را که تعیین کننده کلاس آن بردار می‌باشد بر می‌گرداند، تابع جداکننده می‌گویند. از آنجایی که در این پایان-نامه حاضر ۲ کلاس وجود دارد، بحث به روش‌های دو کلاسه محدود می‌شود.

یک جداکننده خطی را می‌توان با کمک رابطه‌ای به صورت (۵-۲) نشان داد [۱۷،۳۳].

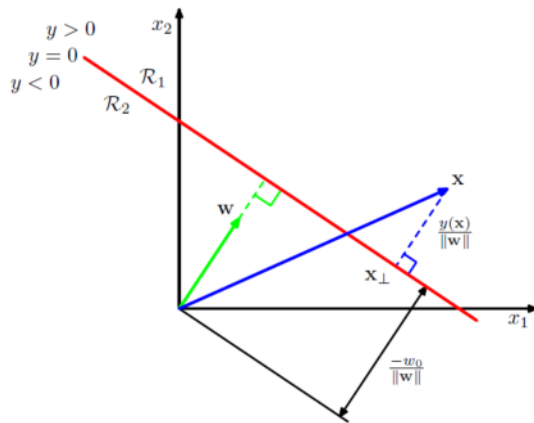
$$y(x, w) = w^T x + w_0 \quad (۵-۲)$$

در رابطه فوق، w برداری می‌باشد که وزن‌ها و w_0 میزان خطا یا bias را تعیین می‌کند. اگر مقدار $y(x) \geq 0$ باشد، آن داده متعلق به کلاس اول، و در غیر این صورت، این داده متعلق به کلاس دوم می‌باشد. سطح متناظر با این رابطه $y(x) = 0$ می‌باشد که سطحی با درجه $D-1$ در فضای D بعد می‌باشد. برای محاسبه فاصله تابع از مبدا داریم:

$$\frac{w^T x}{w} = -\frac{w_0}{w} \quad (۵-۳)$$

زیرا اگر دو نقطه x_A و x_B بر روی سطح مورد نظر قرار داشته باشد آنگاه داریم $y(x_A) = y(x_B) = 0$ و از این نتیجه می‌گیریم $w^T(x_A - x_B) = 0$. اگر یک نقطه بر روی خط جدا کننده باشد آنگاه داریم:

$$y(x) = w^T x + w_0 = 0 \rightarrow w^T x = -w_0$$



شکل ۵-۵ خط قرمز سطح جداکننده را نشان می‌دهد روابط فوق برای فضای دو بعدی به تصویر کشیده شده است.

فاصله یک نقطه از سطح از رابطه $\frac{y(x)}{\|w\|}$ بدست می‌آید. زیرا، اگر x_{\perp} تصویر x بر روی سطح جداکننده فرض کنیم آنگاه داریم.

$$x = x_{\perp} + r \frac{w}{\|w\|} \rightarrow w^T x + w_0 = w^T \left(x_{\perp} + r \frac{w}{\|w\|} \right) + w_0 =$$

$$(w^T x_{\perp} + w_0) = r \frac{w^T w}{\|w\|} = r \|w\| \rightarrow r = \frac{x}{\|w\|} \quad (۴-۵)$$

اگر بخواهیم به صورت فشرده تر این عبارت‌ها را نمایش دهیم، به متغیر ورودی یک مقدار $x = 1$ را اضافه می‌کنیم، آنگاه خواهیم داشت.

$$\tilde{x} = (1, x^T)^T, \tilde{w} = (w_0, w^T)^T \rightarrow y(x) = \tilde{w}^T \tilde{x} \quad (۵-۵)$$

۵-۳ - حداقل مربعات خطا برای کلاسه بندی

حال می‌خواهیم، از حداقل مربعات خطا، برای تعیین پارامترهای تابع جداکننده خطی، استفاده کنیم. فرض می‌کنیم که تعداد کلاس‌ها، برابر k باشد. هر کلاس با استفاده از تابعی با رابطه (۵-۶) نمایش داده می‌شود [۱۷، ۳۳].

$$y_k(x) = w_k^T x + w_{k0} \quad (6-5)$$

پس به بطور کلی رابطه فوق را می‌توان به صورت (7-5) نشان داد.

$$y(x) = \tilde{W}^T \tilde{x} \quad (7-5)$$

داده متعلق به کلاس k ام خواهد بود اگر y_k (عنصر سطر k ام بردار y) دارای بالاترین مقدار باشد.

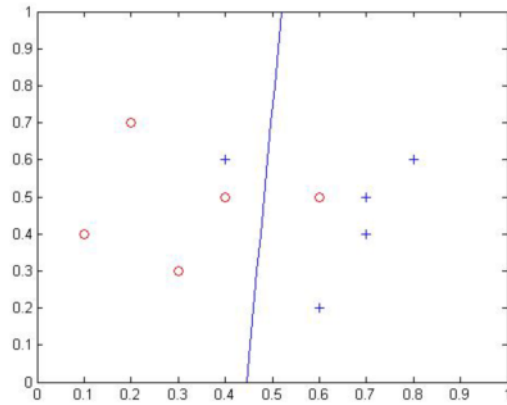
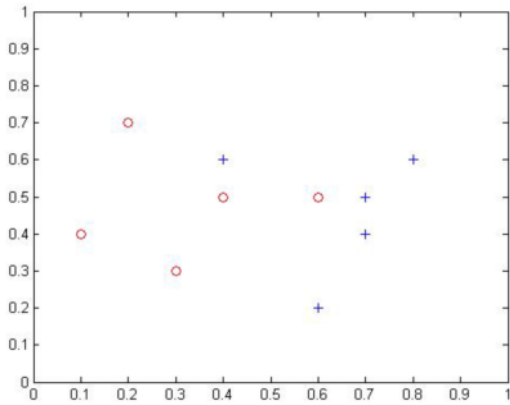
مقدار w باید به صورتی انتخاب شود که، میزان خطا حداقل شود. فرض کنید که، داده‌ها را به صورت $\{x_n, t_n\}, n = 1, \dots, N$ نمایش دهیم، آنگاه تابع حداقل مربع خطا بصورت شکل زیر خواهد بود.

$$E_D(\tilde{W}) = \frac{1}{2} Tr \left\{ \left(\tilde{W}^T \tilde{x} - T \right)^T \left(\tilde{W}^T \tilde{x} - T \right) \right\} \quad (8-5)$$

با مشتق گرفتن از رابطه فوق نسبت به w ، مقدار \tilde{W} از رابطه (9-5) بدست خواهد آمد.

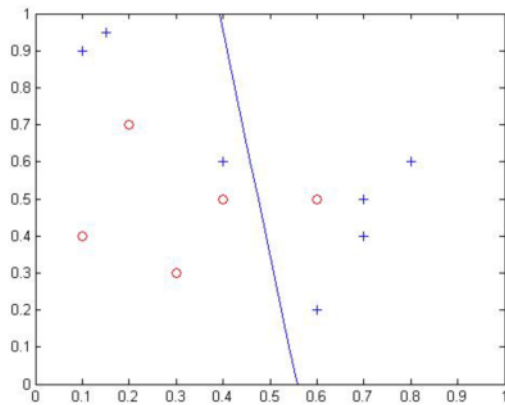
$$\tilde{W} = \left(\tilde{X}^T \tilde{X} \right)^{-1} \tilde{X}^T T \quad (9-5)$$

یکی از معایب این روش، این است که به شدت تحت تاثیر مقادیر کرانه ایست. به عنوان مثال به داده‌های شکل 7-5 دقت کنید. همانطور در شکل 6-5 مشاهده می‌کنیم. روش حداقل مربعات، مرز را به صورت مناسبی تعیین کرده است، اما در شکل 8-5، به علت داده‌های نویز، این روش دچار خطا می‌شود.



شکل ۵-۷ داده‌ها متعلق به دو کلاس می‌باشند داده‌های متعلق به هر کلاس با استفاده از علائم مورد استفاده جدا شده‌اند.

شکل ۵-۶ کلاسه بندی داده‌ها با استفاده از الگوریتم حداقل مربعات



شکل ۵-۸ اثر داده‌های انحرافی بر روی نتیجه حداقل مربعات خطا

۵-۴- الگوریتم پرسپترون

یکی از پرکاربردترین الگوریتم‌ها، برای جداکردن خطی، الگوریتم پرسپترون می‌باشد و برای اولین بار در [۳۴] معرفی شده است. در این الگوریتم ابتدا فرض می‌شود که، فقط دو کلاس وجود داشته

باشد. در قدم اول، بردار ورودی، با استفاده از تابع پایه، نگاشت داده می شود. این مقدار نگاشت داده شده، به عنوان ورودی، به یک تابع ارسال می شود، که می توان آن را با رابطه (۵-۱۰) نشان داد.

$$y(x) = f(w^T x) \quad (10-5)$$

تابع f تابع گام^۱ است که، دارای رابطه به صورت زیر می باشد.

$$f(a) = \begin{cases} 1 & a \geq 0 \\ -1 & a < 0 \end{cases} \quad (11-5)$$

یکی از تفاوت های این الگوریتم با روش سابق در این است که، در روش یادشده در بالا، مقداری که به کلاس دوم داده می شود، برابر ۰ می باشد. اما در الگوریتم پرسپترون مقادیر هدف به صورت $t \in \{-1, 1\}$ می باشد. هدف این الگوریتم انتخاب مقدار w است به نحوی که مقدار تابع خطا به حداقل برسد. یکی از ساده ترین توابع، تابعی است که، تعداد الگوهای نادرست کلاسه بندی شده اند را برمی گرداند. فرض کنید که مقدار w به درستی انتخاب شده باشد. اگر مقدار دلخواه x_n داشته باشیم و این متغیر به کلاس C_1 تعلق باشد، آنگاه خواهیم داشت.

$$w^T \phi(x_n) \geq 0$$

و اگر داده متعلق به کلاس دوم باشد، آنگاه داریم.

$$w^T \phi(x_n) < 0 \quad (12-5)$$

می دانیم که $t \in \{-1, 1\}$. پس در صورتی که یک نمونه به درستی انتخاب شود، مستقل از کلاسی که داده به آن تعلق دارد، خواهیم داشت.

$$w^T \phi(x_n) t_n \geq 0 \quad (13-5)$$

^۱ Step

این تابع زمانی به ماکزیمم مقدار می‌رسد که، همه الگوها به درستی کلاسه بندی شود و تابع خطا به صورت زیر بیان می‌شود.

$$E = - \sum_{n \in M} w^T \varphi(x_n) t_n \quad (14-5)$$

برای تغییر وزن با استفاده از الگوریتم گرادیان می‌توان از رابطه زیر استفاده کرد.

$$w^{t+1} = w^t - \eta \nabla E = w^t + \eta \varphi_n t_n \quad (15-5)$$

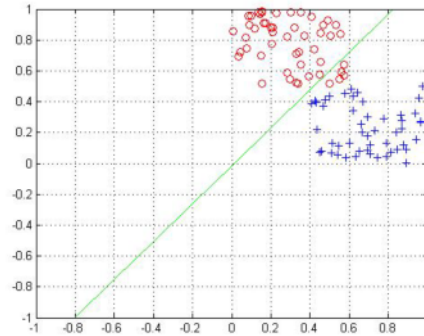
توجه کنید اگر w را در یک ضریب ضرب کنیم، نتیجه نهایی تغییری نمی‌کند. زیرا آنچه در یک بردار مهم است، جهت بردار می‌باشد و اندازه آن اهمیتی ندارد.

الگوریتم پرسپترون در بین داده‌ها گردش می‌کند، اگر نمونه به درستی کلاسه بندی شده باشد، وزن تغییر نمی‌کند اما اگر نمونه، متعلق به کلاس اول باشد و نادرست کلاسه بندی شود، بردار $\varphi(x_n)$ به وزن w اضافه می‌شود و اگر داده به کلاس دوم تعلق داشته باشد، بردار $\varphi(x_n)$ از وزن w کم می‌شود. در هر مرحله، وزن‌ها بهبود می‌یابد و اثر خطا کاهش می‌یابد، زیرا داریم.

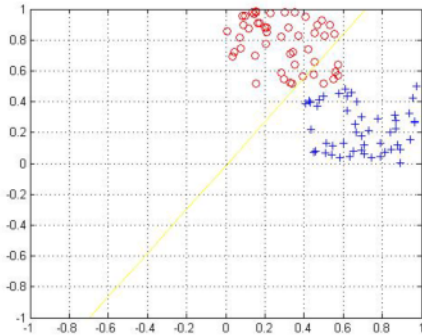
$$-w^{(\tau+1)T} \varphi_n t_n = -w^{(\tau)T} \varphi_n t_n - (\varphi_n t_n)^T \varphi_n t_n < -w^{(\tau)T} \varphi_n t_n \quad (16-5)$$

و از طرفی $\|\varphi_n t_n\|^2 \geq 0$ می‌باشد. البته رابطه بالا نشان می‌دهد، با تغییر وزن، سهم هر کدام از نمونه‌ها در خطا کاهش می‌یابد، اما زمانی که وزن‌ها تغییر می‌کند ممکن است، بعضی از نمونه‌ها که قبلاً درست کلاسه بندی شده است کلاسه بندی نادرست شود. شکل ۵-۹ مراحل اولیه که الگوریتم پرسپترون طی می‌کند تا داده‌ها را از هم جدا کند نشان می‌دهد. در مراحل اولیه تعدادی از داده‌ها به صورت نادرستی کلاسه بندی شده‌اند. اما به مرور زمان، این خطاها کم می‌شود تا به صفر برسد.

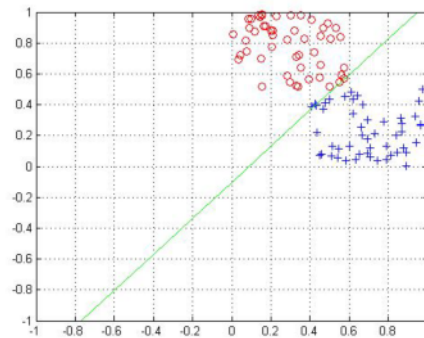
الف) گام ۱



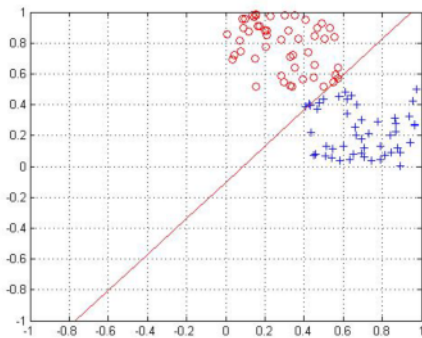
ب) گام ۲



پ) گام ۷



ت) گام ۸



شکل ۵-۹ گام‌های طی شده توسط الگوریتم پرسپترون برای پیدا کردن جداکننده بهینه (در داده‌های ایده آل داده‌ها تقریباً بطور کامل جداپذیر خطی می‌باشند). این الگوریتم در ابتدا مقداری که به بردار جداکننده انتساب می‌کند مقداری تصادفی می‌باشد، به همین دلیل تعدادی از داده‌ها به صورت نادرستی کلاسه بندی شده‌اند. در گام‌های دیگر رفته رفته این خطا کمتر می‌شود به طوری که در گام ۸ کاملاً داده‌ها به درستی کلاسه بندی می‌شود.)

۵-۵- مدل آماری

در ادامه نشان می‌دهیم که، با یک فرض ساده کننده، می‌توان یک مدل جداکننده را، با استفاده از مدل آماری و قانون بیز، محاسبه کرد. برای بدست آوردن مدل آماری، لازم است تا $p(x|C_k)$ و $p(C_k)$ را بدست آورده شود. برای حالت دو کلاسه، و با استفاده از قانون بیز، داریم.

$$p(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(x|C_2)p(C_2)} = \frac{1}{1 + \exp(-a)} \rightarrow a = \ln\left(\frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}\right) \quad (17-5)$$

برای حالتی که بیشتر از دو کلاس وجود داشته باشد، رابطه ای فوق، به صورت (18-5) خواهد بود.

$$p(C_k|x) = \frac{p(x|C_k)p(C_k)}{\sum_j p(x|C_j)p(C_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)} \quad (18-5)$$

با توجه به رابطه فوق داریم.

$$a_k = \ln p(x|C_k)p(C_k) \quad (19-5)$$

برای یک حالت خاص، می توان فرض کرد که توزیع class condition، توزیع نرمال باشد و ماتریس کواریانس برای هر دو کلاس مشترک باشد، آنگاه رابطه class condition، به صورت زیر بازنویسی می شود.

$$p(x|C_1) = \frac{1}{(2\pi)^{\frac{D}{2}} |\Sigma|^{\frac{1}{2}}} \exp\left\{-\frac{1}{2}(x - \mu_1)^T \Sigma^{-1} (x - \mu_1)\right\} \quad (20-5)$$

با در نظر گرفتن رابطه فوق، و استفاده از قانون بییز، تابع احتمال از رابطه (21-5) بدست می آید.

$$p(C_1|x) = \sigma(w^T x + w_0) \quad (21-5)$$

که در رابطه فوق w و w_0 از رابطه (22-5) و (23-5) بدست می آید.

$$w = \Sigma^{-1}(\mu_1 - \mu_2) \quad (22-5)$$

$$w_0 = -\frac{1}{2} \mu_1^T \Sigma^{-1} \mu_1 + \frac{1}{2} \mu_2^T \Sigma^{-1} \mu_2 + \ln\left(\frac{p(C_1)}{p(C_2)}\right) \quad (23-5)$$

برای حالتی که، k کلاس وجود داشته باشد، آنگاه a_k از رابطه (24-5) بدست می آید.

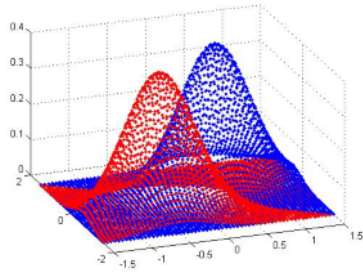
$$a_k(x) = w_k^T x + w_{k0} \quad (24-5)$$

که w_k و w_{k0} از رابطه (25-5) و (26-5) بدست می‌آید.

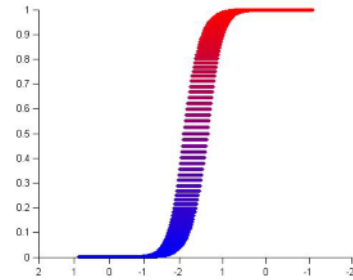
$$w_k = \Sigma^{-1} \mu_k \quad (25-5)$$

$$w_{k0} = -\frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \ln p(C_k) \quad (26-5)$$

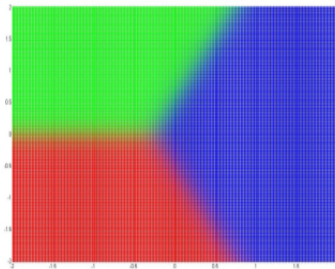
در آزمایش اول ابتدا داده‌هایی تصادفی را، از دو مدل گوسی که در شکل 5-11 مشاهده می‌شوند، انتخاب کردیم. تابع *pastier* این توزیع را، در شکل 5-10، بدست آوردیم. همانطور که مشاهده می‌کنید، شکل آن همانند تابع *sigmoid* می‌باشد. در آزمایش دیگر داده‌ها را از 3 کلاس که دارای توزیع گوسی می‌باشد انتخاب کردیم. زمانی که ماتریس کواریانس سه کلاس برابر می‌باشد (در شکل 5-12) می‌بینیم که مرز جداکننده بین هر کدام از کلاس‌ها به صورت یک خط مستقیم می‌باشد. اما در قسمت‌های بعدی ماتریس کواریانس را تغییر داده، مشاهده می‌شود زمانی که کلاس سوم حول میانگین خود چگال‌تر (شکل 5-14 و 5-15) است، مرز جداکننده بین آن و کلاس‌های دیگر به صورت منحنی تبدیل می‌شود و ناحیه که متعلق به آن کلاس است، کوچکتر از سایر کلاس‌ها می‌گردد. و زمانی که داده‌های آن پراکنده‌تر (شکل 5-16 و 5-17) می‌باشد داده‌ها از میانگین دورتر می‌شود.



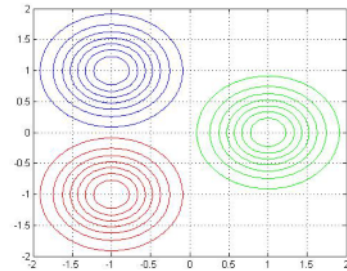
شکل ۵-۱۱ توزیع چگال احتمال perior



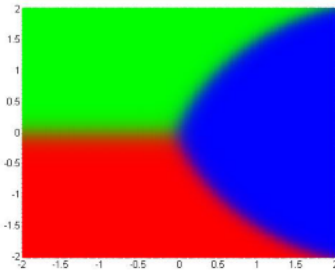
شکل ۵-۱۰ احتمال posterior



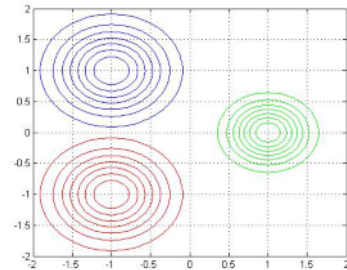
شکل ۵-۱۳ مرز جدا کننده بین داده های دو کلاس



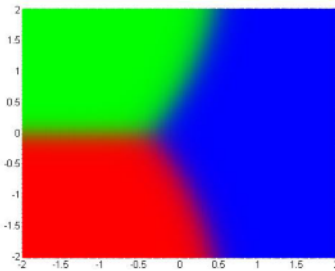
شکل ۵-۱۲ کانتور های ۳ کلاس زمانی که دارای کواریانس مشترک می باشند.



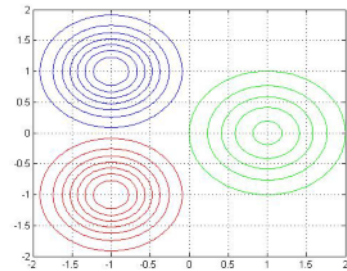
شکل ۵-۱۵ مرز جدا کننده کلاس ها



شکل ۵-۱۴ کانتور ها ۳ کلاس زمانی که کلاس سوم متمرکز تر می باشد



شکل ۵-۱۷ مرز جداکننده بین کلاس

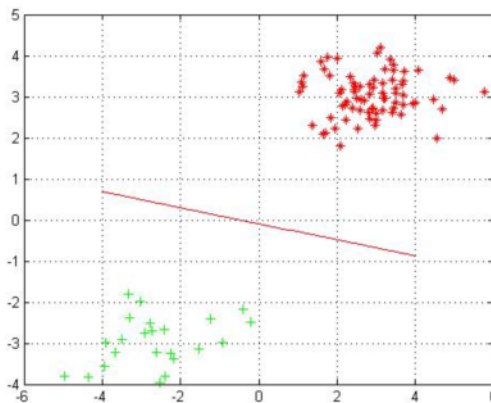


شکل ۵-۱۶ کانتور کلاس ها زمانی که یک کلاس نسبت به دو

کلاس دیگر پراکنده تر می باشد

آزمایش دیگر، استفاده و بررسی این روش برای پیدا کردن مرز جداکننده بین کلاس هایی که جدا پذیر خطی، هستند انجام شد. و همانطور که مشاهده می شود، این روش بر روی داده هایی که جداپذیر خطی می باشد، به درستی عمل می کند. نکته دیگر این که، با تغییر نسبت تعداد داده های دو کلاس به هم، شیب خط جدا کننده تغییر نمی کند، اما خط بر روی صفحه به موازات هم جابه جا می شود، این جابه جایی به نحوی است که، مرز جداکننده به کلاسی که تعداد داده بیشتر دارد، نزدیکتر می شود. نتایج این آزمایش در تصویر ۵-۱۸ نشان داده شده است.

این مدل به نسبت مدل حداقل مربعات خطا دارای دقت بیشتری می باشد اما زمان محاسبات بالایی دارد. برتری دیگر این روش نسبت به روش حداقل مربعات این است که به مرور زمان می توان آن را آموزش داد پس لازم نیست تمام داده ها را همزمان داشت و از طرفی احتمال خطاهای عددی کاهش می یابد.



شکل ۵-۱۸ پیدا کردن مرز جداکننده با الگوریتم regression بر روی داده های ایده آل

اگر میانگین و واریانس هر کدام از کلاس‌ها را داشته باشیم، می‌توان با استفاده از آنها برای حالت‌های خاص، مرز جداکننده را تعیین کنیم. اما چگونه می‌توان میانگین و واریانس را از روی داده‌های آموزشی بدست آورد.

فرض کنید که، دو کلاس وجود داشته باشد. و مدل توزیع، توزیع نرمال باشد. این نمونه‌ها را می‌توان به صورت $\{x_n, t_n\}$ نمایش داد. اگر $p(C_1) = \pi$ برای کلاس اول، مقدار $t=1$ و برای کلاس دوم مقدار $t=0$ باشد. برای کلاس اول داریم.

$$p(x_n, C_1) = p(C_1) p(x_n | C_1) = \pi p(x_n | C_1) \quad (27-5)$$

برای کلاس دوم داریم

$$p(x_n, C_2) = p(C_2) p(x_n | C_2) = (1 - \pi) p(x_n | C_2) \quad (28-5)$$

و تابع Likelihood به صورت زیر نوشته خواهد شد.

$$p(t | \pi, \mu_1, \mu_2, \Sigma) = \prod_{n=1}^N [\pi N(x_n | \mu_1, \Sigma)]^{t_n} [(1 - \pi) N(x_n | \mu_2, \Sigma)]^{(1-t_n)} \quad (29-5)$$

برای بدست آوردن مقدار نهایی π ، از عبارت (30-5) نسبت به π مشتق می‌گیریم.

$$\sum_{n=1}^N \ln \pi t_n + \ln(1 - \pi)(1 - t_n) \quad (30-5)$$

آنگاه داریم.

$$\pi = \frac{1}{N} \sum_{n=1}^N t_n = \frac{N_1}{N} \quad (31-5)$$

که در آن N_1 تعداد عناصر متعلق به کلاس اول و N_2 تعداد عناصر کلاس دوم را نشان می‌دهد. حال برای بدست آوردن میانگین، لازم است تا، عبارت‌هایی که مقدار میانگین کلاس اول در آن اثر دارد، را مشخص کنیم. پس به رابطه (۳۲-۵) می‌رسیم.

$$\sum_{n=1}^N t_n \ln N(x_n | \mu_1, \Sigma) = -\frac{1}{2} \sum_{n=1}^N t_n (x_n - \mu_1)^T \Sigma^{-1} (x_n - \mu_1) + const \quad (32-5)$$

و خواهیم داشت

$$\mu_1 = \frac{1}{N_1} \sum_{n=1}^N t_n x_n \quad (33-5)$$

میانگین کلاس دوم برابر می‌شود با

$$\mu_2 = \frac{1}{N_2} \sum_{n=1}^N (1-t_n) x_n \quad (34-5)$$

برای محاسبه ماتریس کواریانس می‌توان از رابطه (۳۵-۵) استفاده کنیم.

$$-\frac{1}{2} \sum_{n=1}^N t_n \ln |\Sigma| - \frac{1}{2} \sum_{n=1}^N t_n (x_n - \mu_1)^T \Sigma^{-1} (x_n - \mu_1) - \frac{1}{2} \sum_{n=1}^N (1-t_n) \ln |\Sigma| - \frac{1}{2} \sum_{n=1}^N (1-t_n) (x_n - \mu_2)^T \Sigma^{-1} (x_n - \mu_2) = -\frac{N}{2} \ln |\Sigma| - Tr\{\Sigma^{-1} S\} \quad (35-5)$$

که در رابطه فوق S برابر است با

$$S = \frac{N_1}{N} S_1 + \frac{N_2}{N} S_2 \quad (36-5)$$

$$S_1 = \frac{1}{N_1} \sum_{n \in C_1} (x_n - \mu_1) (x_n - \mu_1)^T \quad (37-5)$$

$$S_2 = \frac{1}{N_2} \sum_{n \in C_2} (x_n - \mu_2) (x_n - \mu_2)^T \quad (38-5)$$

با مشتق گرفتن از رابطه فوق مقدار بهینه برای ماتریس کواریانس برابر با $\Sigma = S$ می شود.

۵-۶- Iterative reweighted least squares

یکی از روش‌ها برای بهینه سازی تابع خطا، روش نیوتن می باشد که به صورت رابطه (۵-۳۹) نمایش

داده می شود. استفاده از این متد برای خوشه بندی استفاده کردیم [۳۵].

$$w^{\text{new}} = w^{\text{old}} + H \nabla E \quad (۵-۳۹)$$

اگر بخواهیم از این روش برای کلاسه بندی داده‌ها استفاده کنیم، داریم.

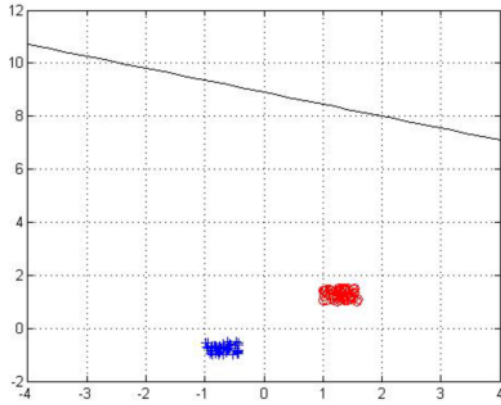
$$\nabla E(w) = \sum_{n=1}^N (y_n - t_n) \varphi_n = \boldsymbol{\varphi}^T (\mathbf{y} - \mathbf{t}) \quad (۵-۴۰)$$

$$H = \nabla \nabla E(w) = \sum_{n=1}^N y_n (1 - y_n) \varphi_n \varphi_n^T = \boldsymbol{\varphi}^T R \boldsymbol{\varphi} \quad (۵-۴۱)$$

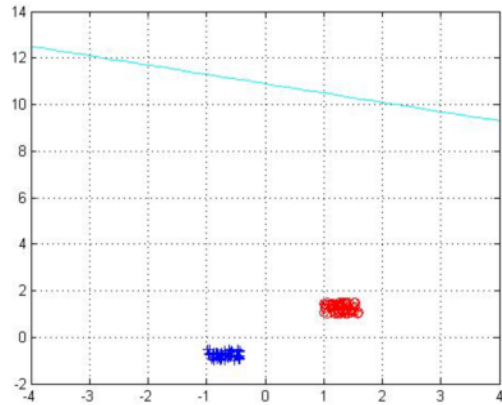
که ماتریس R ماتریسی است که درایه‌های آن به صورت $R_{nn} = y_n(1 - y_n)$ می باشد. برای تغییر وزن‌ها می توان از رابطه (۵-۴۲) استفاده کرد.

$$w^{\text{new}} = w^{\text{old}} - (\boldsymbol{\varphi}^T R \boldsymbol{\varphi})^{-1} \boldsymbol{\varphi}^T (\mathbf{y} - \mathbf{t}) \quad (۵-۴۲)$$

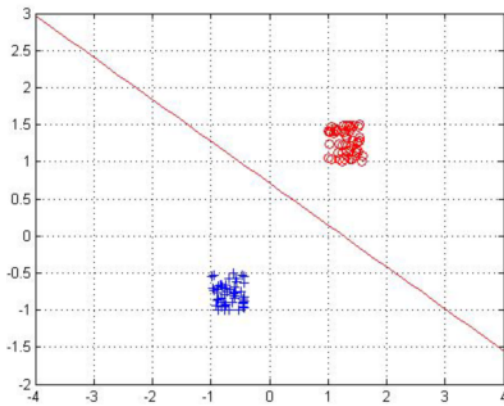
این الگوریتم در ابتدا مقداری تصادفی به بردار جداکننده انتساب می کند به همین دلیل تعدادی از از داده‌ها به صورت نادرستی کلاسه بندی می شود. در گام‌های بعدی رفته رفته این خطا کمتر می شود بصورتی که در گام ۶ داده‌ها کاملاً به درستی کلاسه بندی می شود. این مساله برتری روش را برای پیدا کردن نقاط بهینه نشان می دهد. روند اجرای این الگوریتم را در شکل ۵-۱۹ مشاهده می کنید.



ب) گام ۲۰



الف) گام اول



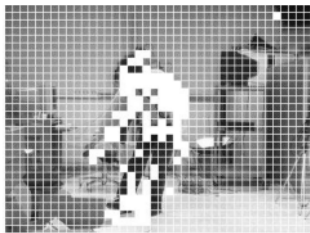
پ) گام ۳۰

شکل ۵-۱۹ گام‌های طی شده توسط الگوریتم Iterative reweighted least squares برای جداکردن مرز جداکننده.

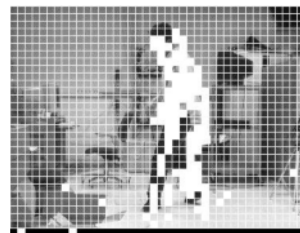
۵-۷- نتایج (از نتایج پایان نامه)

در این بخش ابتدا به معرفی روش کلاسه بندی داده‌های جداپذیر خطی، با استفاده از روش حداقل مربعات پرداختیم. برای آزمایش و ارزیابی این روش داده‌ها را بر روی دو نمونه داده واقعی و داده‌های ایده آل آزمایش کردیم. مشاهده شد که داده‌های ایده آل بطور کامل، جداپذیر خطی می‌باشند هم‌منظوریکه در نتایج بخش اول نشان داده شد این نوع روش به طور مناسبی می‌تواند داده‌ها را کلاسه بندی کند.

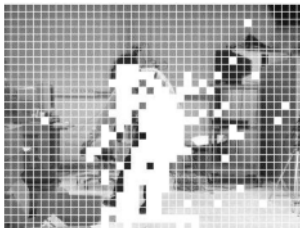
داده‌هایی دیگری که مورد ارزیابی قرار گرفتند، داده‌های واقعی استخراج شده از ویدیو می‌باشند. این نتایج در شکل ۵-۲۰ خلاصه شده است. سطر اول، نتایج الگوریتم‌های استاندارد را نشان می‌دهد. با استفاده از نتایج این الگوریتم‌ها، داده‌های مورد نیاز برای الگوریتم حداقل مربعات فراهم می‌شود و در گام بعدی، پارامترهای حداقل مربعات خطا، با استفاده از داده‌های آموزشی استخراج می‌گردد. نتیجه پایانی در سطر آخر آمده است. همانطور که مشاهده می‌کنید، نتیجه نسبت به الگوریتم‌های استاندارد بهبود داشته است.



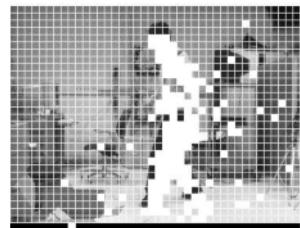
ب



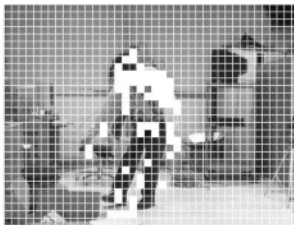
الف



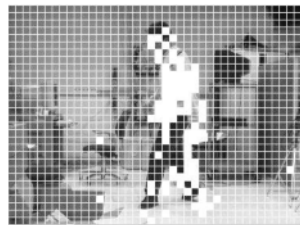
ت



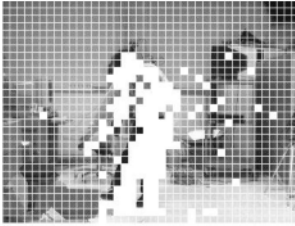
پ



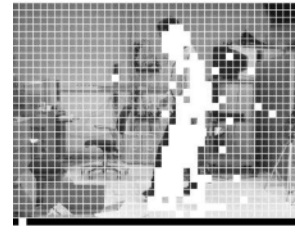
ج



ث



ح



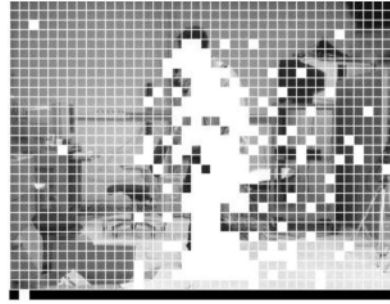
چ

شکل ۵-۲۰ نتایج بدست آمده با استفاده از الگوریتم حداقل مربعات خطا، سطر ۴ نتایج الگوریتم حداقل مربعات خطا را نشان می دهد سایر سطر ها نتایج الگوریتم ها استاندارد می باشد(ادامه تصویر).

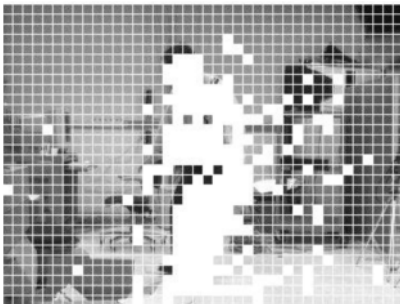
الگوریتم دیگر مورد ارزیابی، الگوریتم پرسپترون می باشد. این نتایج در شکل ۵-۲۱ خلاصه شده است. در قدم اول، الگوریتم با استفاده از داده های آموزشی، *train* می شوند و در گام بعدی، تعدادی فریم به شبکه آموزش داده شده، ارسال می شود. همانطور که در نتایج نشان داده شده در شکل ۵-۲۱ مشاهده می شود، تعدادی از بلوک ها که متعلق به پس زمینه بوده اند، به عنوان پیکسل شی در نظر گرفته شده اند. بیشتر این بلوک ها، بلوک هایی در اطراف اشیا می باشند. از انجایی که این نواحی در زمان های نزدیک، نواحی بوده اند که، اشیا در آن موقعیت قرار داشته است، زمانی که اشیا از این نواحی خارج می شود، شبکه به نادرست، این پیکسل ها را نیز به عنوان پیکسل شی در نظر می گیرد. این مشکل ناشی از ایستا بودن مدل، در طی زمان می باشد، پس لازم است، مدل در گذر زمان تغییرات را بروز کند.



ب



الف



ت



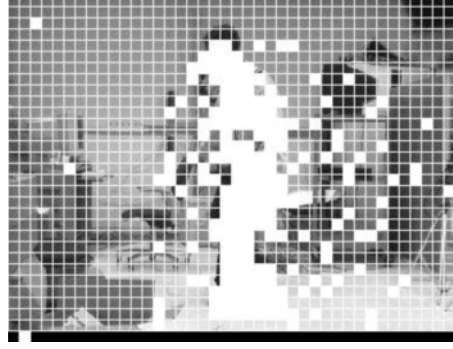
پ

شکل ۵-۲۱ نتایج استخراج اشیا توسط الگوریتم پرسپترون

در قسمت سوم، با الگوریتم *Logistic regression* که، مبتنی بر روش های آماری برای پیدا کردن مرز جداکننده می باشد، آشنا شدیم. نتایج آموزش این الگوریتم با استفاده از داده های موجود را در شکل ۵-۲۲ مشاهده می کنید. این مدل به نسبت مدل حداقل مربعات خطا دارای دقت بیشتری می باشد. برتری دیگر این روش نسبت به روش حداقل مربعات این است که، به مرور زمان می توان آن را آموزش داد پس لازم نیست تمام داده ها را همزمان داشت، و از طرفی، به علت آموزش مرحله به مرحله، احتمال خطاهای عددی کاهش می یابد.



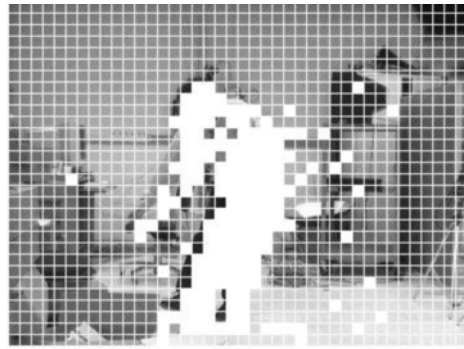
ب



الف



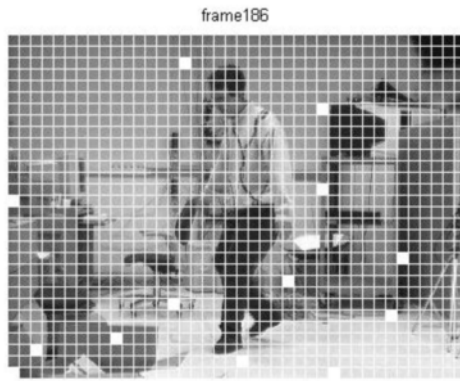
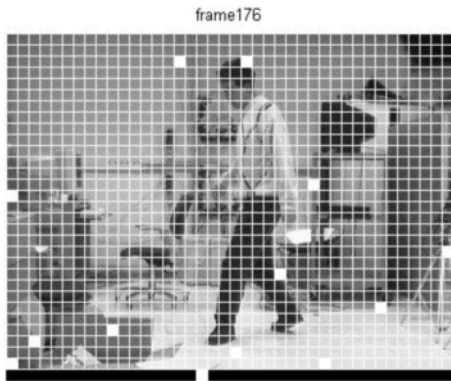
ت



پ

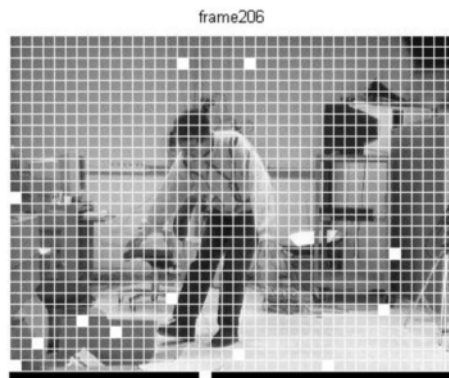
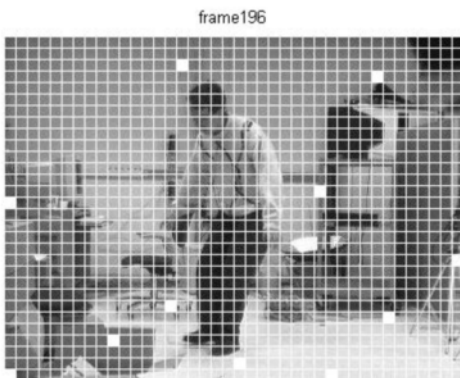
شکل ۲۲-۵ نتایج عمل بر روی داده های واقعی با الگوریتم Logistic regression

الگوریتم نهایی، Iterative reweighted least squares می باشد که نتایج آن در شکل ۲۳-۵ خلاصه شده است. ابتدا الگوریتم با استفاده از داده های آموزشی، *train* می شوند و در گام بعدی تعدادی فریم به شبکه آموزش داده شده، ارسال می شود. همانطور که در نتایج ثبت شده در شکل ۲۳-۵ مشاهده می شود، این الگوریتم اصلا نتوانسته نقاط متعلق به اشیا را پیدا کند. زیرا این الگوریتم مبتنی بر محاسبات عددی ماتریسی فراوان می باشد، و از آنجایی که این عمل، باید چندین بار تکرار شود، زمانی که داده ها غیر ایده آل باشد احتمال بوجود آمدن ماتریس *singular* بالا می رود. در نتیجه خطا، آنقدر بزرگ می شود که عملا نمی توان از این الگوریتم استفاده کرد.



ب

الف



ت

پ

شکل ۵-۲۳ نتایج بر روی داده‌ها ویدیو با استفاده از الگوریتم Iterative reweighted least squares

فصل هشتم

بهبود سیستم فشرده سازی MPEG۲

۶-۱- مقدمه

از آنجایی که رسانه های ویدیویی معمولا حجم زیادی از داده ها را تولید می کنند این حجم عظیم اطلاعات مشکلاتی از قبیل فضای زیاد برای ذخیره کردن اطلاعات و پهنای باند بالا برای انتقال را به همراه دارد. به همین دلیل در بسیاری از کاربردها ضرورت دارد تا حجم داده های تولید شده توسط سیستم های ویدیویی کاهش یابد.

در فصل سوم با استاندارد ها و سیستم های فشرده سازی ویدیو آشنا شدیم. دیدیم عمده حجم محاسباتی در سیستم های فشرده سازی رایج محدود به فرایند انطباق بلوک می باشد. در سیستم های فشرده سازی فرض می شود که بلوک های درون فریم ها، در طول زمان جابه جا می شوند. بنابراین لازم است تا میزان جابه جایی تک تک بلوک های فریم مرجع را در فریم جاری بدست آورد. برای بدست آوردن این جابه جایی باید همه بلوک های موجود در فریم جاری که ممکن است یک بلوک از فریم مرجع به آن منتقل شده باشد، جستجو شود تا بهترین بلوک پیدا شود. این پدیده زمانبر خواهد بود اگر تابعی که بلوک ها را با هم مقایسه می کند نادرست انتخاب شده باشد، یا بلوک هایی که لازم نیست مورد ارزیابی قرار گیرند مورد جستجو قرار بگیرند.

انتخاب تابعی نادرست برای مقایسه بلوک ها به دو صورت می تواند موجب افزایش حجم محاسبات شود:

- تابع در درون خود حجم محاسباتی بالایی داشته باشد مثلا درون تابع از توابع لگاریتمی یا مثلثاتی استفاده شود.
- تابع بدرستی نتواند بلوک ها را انتخاب کند که در این صورت حجم داده ها بیشتر شده و به تبعه آن حجم محاسبات بیشتر می شود.

مشکل دوم (یعنی زمانیکه بلوک هایی که لازم نیست ارزیابی شوند مورد بررسی قرار می گیرند) معلول دو عامل است.

- در بسیاری از بلوک‌های تغییر قابل توجه ای وجود ندارد به همین دلیل لازم نیست مورد ارزیابی قرار گیرند.

- جستجو ناآگاهانه انجام می‌گیرد، مثلاً در بسیاری از مواقع با استفاده از اطلاعات که از بلوک‌های مجاور داریم می‌توانیم محدوده جستجو را محدود کنیم و با استفاده از این اطلاعات حجم محاسبات را کاهش دهیم.

در این فصل، به بیان بهینه‌سازی‌هایی که بر روی سیستم فشرده‌سازی MPEG۲ انجام داده ایم، می‌پردازیم. بهینه‌سازی ویدیو از جنبه‌های مختلفی می‌تواند مورد ارزیابی قرار گیرد. یکی از این جنبه‌ها کاهش زمان فشرده‌سازی و جنبه دیگر، افزایش درصد فشرده‌سازی می‌باشد. همچنین معیارهای مشابهت، که در استانداردهای فشرده‌سازی استفاده می‌شود، مورد بررسی قرار می‌گیرند، تا اهمیت آن بر کارایی مشاهده شود.

۶-۲- معیارهای شباهت و کارایی آن بر فشرده‌سازی ویدیو

همانطوری که در فصل‌های قبل گفته شد، یکی از مهم‌ترین قسمت‌های سیستم‌های فشرده‌سازی ویدیو، پیدا کردن بلوک‌های متحرک و پیدا کردن بردار حرکت (نشان دهنده جابه‌جایی بلوک متحرک در صحنه) می‌باشد. در فصل چهارم، الگوریتم‌های جستجو بلوک را معرفی کردیم، این الگوریتم‌ها که به منظور پیدا کردن بردار حرکت طراحی شده‌اند، در درون خود نیاز دارند تا بلوک‌ها را با هم مقایسه کنند تا نزدیک‌ترین بلوک از بلوک‌های موجود در پنجره‌ی جستجو را با بلوک جاری پیدا کنند.

اما چگونه میزان مشابهت دو بلوک را با هم اندازه بگیریم. برای پیدا کردن میزان مشابهت بین دو بلاک، روش مرسوم در فشرده‌سازی ویدیو تابع Hamming می‌باشد اما آیا این معیار مناسب‌ترین معیار برای پیدا کردن میزان مشابهت می‌باشد؟ مزیت‌ها و معایب هر کدام از روش‌ها چیست؟ در این بخش سعی می‌کنیم تا با بررسی چند تابع و مقایسه چند روش با تابع Hamming مزایا و معایب هر کدام از روش‌ها را مورد بررسی قرار دهیم.

۶-۲-۱- تعاریف توابع مشابهت

در این قسمت در ابتدا خواص و ویژگی‌های هر تابع^۱ similarity لازم دارد، را بیان می‌کنیم و در نهایت به معرفی چند تابع که دارای این خاصیت باشد می‌پردازیم. ورودی این تابع، دو بردار از اعداد حقیقی و خروجی آن، عددی حقیقی است، که میزان اختلاف دو بردار را نشان می‌دهد. طبیعتاً هر چه تفاوت‌ها بیشتر باشد، این عدد باید بزرگتر باشد و عدد کوچک نشان دهنده شباهت بیشتر می‌باشد. با توجه به این مسئله برای تعریف این تابع می‌توان نوشت:

$$d : X * X \rightarrow R \quad (۱-۶)$$

دومین خاصیت این است که، فاصله هر بردار با خودش باید کوچکتر یا مساوی تر از فاصله هر بلوک دیگر از آن باشد. این خاصیت را می‌توان با رابطه (۲-۶) نشان دهیم.

$$\exists d_0 \in R : -\infty < d_0 \leq d(x, y) < \infty, \forall x, y \quad (۲-۶)$$

$$d(x, x) = d_0 \quad (۳-۶)$$

سومین خاصیت می‌گوید که، اگر برداری وجود داشته باشد که فاصله آن با بردار x برابر d باشد، آنگاه آن بردار باید برابر با بردار x باشد به بیان ریاضی

$$d(x, y) = d_0 \rightarrow x = y \quad (۴-۶)$$

و خاصیت نهایی نامساوی مثلثی می‌باشد این خاصیت بیان می‌کند.

$$d(x, y) + d(y, z) \geq d(x, z), \forall x, y, z \quad (۵-۶)$$

در ادامه چند تابع برای اندازه‌گیری فاصله بین دو بردار معرفی می‌کنیم که دارای خواص فوق می‌باشد. یکی از روابط کلی رابطه (۶-۶) می‌باشد.

^۱تابع شباهت

$$d_p(x, y) = \left(\sum_{i=0}^l w_i |x_i - y_i| \right)^{1/p} \quad (6-6)$$

در رابطه فوق p یک عدد صحیح بوده و x_i نشان دهنده مولفه i ام بردار x است. همچنین y_i مولفه i ام بردار y و w_i وزن مولفه های i ام را نشان می دهد. این وزن نشانگر میزان اهمیت مولفه است بطوریکه هر چه اندازه این وزن بیشتر باشد اهمیت این مولفه بیشتر خواهد بود. حال اگر که w_i را برابر ۱ در نظر بگیریم و $p=1$ آنگاه تابع hamming بدست می آید. این تابع معمولاً به عنوان معیار مشابهت در الگوریتم های انطباق بلوک استفاده می شود و دارای رابطه (6-7) به صورت زیر می باشد.

$$\sum_{i=1}^n |x_i - y_i| \quad (7-6)$$

اگر w_i را برابر ۱ در نظر بگیریم و $p=2$ آنگاه تابع اقلیدسی بدست می آید. این تابع معمولاً به عنوان معیار مشابهت در الگوریتم های انطباق بلوک استفاده می شود و دارای رابطه (6-8) به صورت زیر می باشد.

$$\sum_{i=1}^n (x_i - y_i)^2 \quad (8-6)$$

فاصله اقلیدسی (6-9) را می توان به صورت کلی تر مانند رابطه (6-9) نمایش داد. که در آن B یک ماتریس مثبت معین می باشد.

$$d(x, y) = \sqrt{(x - y)^T B (x - y)} \quad (9-6)$$

در جدول 6-۱ چند تابع شناخته شده را با روابط آن نشان می دهد.

جدول ۱-۶ توابع مختلف برای اندازه‌گیری میزان مشابهت

نام تابع	فرمول
Euclidian	$\sum_{i=1}^n (x_i - y_i)^2$
Chebychev	$\max_i x_i - y_i $
Hamming	$\sum_{i=1}^n x_i - y_i $
Cosine	$\frac{\sum_{i=1}^n x_i y_i}{\sqrt{\sum_{i=1}^n x_i^2} \sqrt{\sum_{i=1}^n y_i^2}}$
chi-square	$\sqrt{\sum_{i=1}^n \frac{x_i^2}{x} + \sum_{i=1}^n \frac{y_i^2}{y}}$
Jensen	$xp = \frac{x}{\sum_{i=1}^n x_i}, yp = \frac{y}{\sum_{i=1}^n y_i}$ $\frac{\sum_{i=1}^n (xp_i \log xp_i + yp_i \log yp_i - (xp + yp)_i \log (xp + yp)_i)}{2}$
Monkowski	$\sqrt[p]{\sum_{i=1}^n (x_i - y_i)^p}$

۶-۲-۲- آزمایش

همانطور که می‌دانیم، فریم‌های متوالی دارای اطلاعات مشابه می‌باشند. به عبارتی پس زمینه در بسیاری از مواقع در فریم‌های متوالی مشابه است و تنها تفاوتی که بین فریم‌ها وجود دارد بلوک‌هایی از تصویر است که در آن شیء واقع شده است. برای اینکه بتوان از این نوع افزونگی استفاده کرد در سیستم‌های فشرده‌سازی فریم‌ها را به بلوک‌هایی تقسیم می‌کنیم، اگر بلوک‌های متناظر در بین فریم جاری و فریم مرجع دارای اختلاف قابل توجهی باشند، احتمالاً آن فریم شامل یک یا چند

شی بوده اند، به همین دلیل لازم است تا شبیه ترین بلوک را با بلوک جاری پیدا کنیم. معیار مرسوم اندازه گیری میزان شباهت فریم ها، مجموع قدر مطلق اختلاف بین بلوک ها است. در این فصل، تابع های دیگر را نیز از جهت حجم فشرده سازی و زمان مورد ارزیابی قرار دادیم، و آنها را با تابع مجموع قدر مطلق اختلاف، مقایسه کردیم تا نقاط قوت و ضعف آنها را بسنجیم.

در این آزمایش از چهار تابع $Euclidian$ ، $hamming$ ، $p=3$ ، $Monkowski$ و $chebychev$ برای اندازه گیری مشابهت بین بلوک ها، استفاده کردیم. و زمان کدینگ و دیکد را برای هر کدام در

جدول ۶-۱ نشان داده‌ایم، همان طور که می بینید زمان دیکد و کد برای معیار مشابهت $Euclidian$ و $hamming$ دقیقاً برابر می باشند. اما حجم فشرده سازی زمانی که از فاصله اقلیدسی استفاده می کنیم کمتر می باشد.

کمترین زمان کدینگ مربوط به تابع $chebychev$ می باشد. علت این است که در این روش بعد از محاسبه کردن اختلاف بین بلوک ها، فقط لازم است تا ماکزیمم را بدست آوریم. این تابع حجم محاسباتی کمی نسبت به توابع دیگر دارد، اما همانطور که می بیند زمانی که از این تابع استفاده می کنیم، حجم فشرده سازی بالا می رود، و از طرفی زمان دیکد نیز به علت این تابع بلوک ها را به درستی منطبق نمی کند، افزایش می یابد.

بالا ترین زمان دیکد و کمترین حجم فشرده سازی متعلق به $Monkowski$ می باشد. علت بالا بودن زمان دیکد این است که، این تابع نسبت به توابع دیگر پیچیده تر بوده و نیاز به محاسبات زیادتری دارد. این روش علاوه بر این که نیاز به زمان محاسبات بالا دارد. حجم فشرده سازی آن نیز کم می باشد. به همین دلیل این روش، برای فشرده سازی ویدیو مناسب نمی باشد.

از دیگر نکاتی که با آزمایش‌ها بدست می‌آید بیشتر بودن زمان کدینگ نسبت به زمان دیکد است. بطوریکه در بعضی از مواقع، زمان محاسبات کدینگ دو برابر محاسبات دیکد می‌باشد. علت این پدیده اینست که در کد کردن، عملیات انطباق بلوک که عملیات زمان‌بری است وجود دارد، در حالی که این عملیات، در دیکد وجود ندارد.

جدول ۶-۲ زمان مورد نیاز برای کد و دیکد بر روی ۴۰ فریم

تابع فاصله	زمان کدینگ	زمان دیکد
Euclidian	۱۵	۶
Hamming	۱۵	۶
Monkowski p=۳	۲۰	۹
Chebychev	۱۴	۱۰

جدول ۶-۳ حجم فشرده سازی بر روی ۴ فریم

تابع فاصله	حجم فشرده شده بر حسب kb
Euclidian	۱۰,۷۲۲
Hamming	۱۰,۷۵۰
Monkowski p=۳	۱۳,۷۴۱
Chebychev	۱۰,۸۰۸

۶-۳- نتیجه

در فریم‌های ویدیویی متوالی معمولاً همبستگی زیادی بین فریم‌ها وجود دارد. برای این که بتوان به فشرده‌سازی مطلوب دست یافت، لازم است تا روش‌هایی ابداع کنیم که از این همبستگی‌ها، که اصطلاحاً افزونگی موقت نامیده می‌شود، استفاده کنیم. یکی از موثرترین روش‌ها برای کاهش

افزونی بین فریمی، استفاده از انطباق بین بلوک‌ها است، که در استاندارد‌های ویدیویی مورد استفاده قرار می‌گیرد. فرایند انطباق بلوک، یکی از زمانگیرترین بخش‌ها در سیستم‌های فشرده‌سازی می‌باشد، به همین دلیل به روش‌های دقیق و کارآمدی نیاز داریم.

اگرچه معیار مرسوم برای اندازه‌گیری انطباق بین بلوک‌ها، مجموع قدر مطلق اختلاف می‌باشد. اما بهترین معیاری که می‌توان برای این منظور استفاده کرد تابع اقلیدسی چرا که بالاترین حجم فشرده‌سازی را دارد و از زمان‌کد و دیکد مناسبی برخوردار است. بدترین عملکرد مربوط به تابع Monkowski می‌باشد که زمان پردازش بالا و حجم فشرده‌سازی پایین دارد. از لحاظ زمانی تابع hamming با تابع اقلیدسی برابری می‌کند اما تابع اقلیدسی حجم فشرده‌سازی بهتری دارد.

۴-۶- بهبود انطباق بلوک با استفاده از مدل سازی پس زمینه

در سیستم‌های فشرده‌سازی، قسمت انطباق الگو میزان جابه‌جایی بلوک جاری را نسبت به بلوک متناظرش در فریم مرجع، محاسبه می‌کند. به علت آنکه این فرایند یکی از زمانبرترین قسمت‌های فشرده‌سازیست الگوریتم‌های فراوانی برای آن ارائه شده است. تمامی این الگوریتم‌ها تعداد نقاط جستجو را کاهش می‌دهند. اما مشکل همه‌ی این الگوریتم‌ها این است که، بدون در نظر گرفتن شرایط محیطی، برای تمام بلوک‌های موجود در فریم عمل جستجو را انجام می‌دهند، در حالی که در بسیاری از مواقع فقط ناحیه محدودی از فریم دارای تغییر می‌باشد که می‌توانیم جستجو را به همین ناحیه محدود کنیم. در این بخش با استفاده از مدل پس زمینه، ناحیه‌ای که تغییرات در آن انجام می‌گیرد، را پیدا و جستجو را به این ناحیه محدود کرده‌ایم. نتایج نشان می‌دهد، این روش بدون کاهش دقت، زمان جستجو و حجم محاسبات را کاهش می‌دهد.

۶-۴-۱- پیدا کردن ناحیه فعال ۲

از آنجا که در محیط های داخلی فقط در قسمت های محدودی از فریم ها تغییرات انجام می گیرد. بنا براین می توانیم جستجو را به این ناحیه ها محدود کنیم. به همین دلیل لازم است تا ناحیه ی فعال را پیدا کنیم. برای این منظور ابتدا مدلی از پس زمینه را بدست می آوریم و با استفاده از اختلافی که بین فریم جاری و پس زمینه وجود دارد، ناحیه فعال را پیدا می کنیم. در ادامه با جزئیات روش آشنا می شویم.

۶-۴-۲- مدل سازی پس زمینه

برای بعضی از کاربرد ها مانند مانیتورینگ ترافیک یا سیستم ها نظارتی لازم است تا اشیا و اشخاصی را از درون فریم های ویدیویی استخراج کنیم. در ابتدا مدلی را از پس زمینه به دست می آوریم (به صورت ساده مدل پس زمینه را می توان تصویری از صحنه بدون اشیا متحرک در نظر گرفت). برای استخراج اشیا، تفاضل پس زمینه و فریم جاری را محاسبه می کنیم، اگر این اختلاف قابل توجه باشد، می توان آن پیکسل را به عنوان پیکسلی از اشیا متحرک در نظر گرفت.

در این بخش برای مدل سازی پس زمینه از روش Running Gaussian Average Background Model استفاده کردیم. در این روش ابتدا فریم اول دنباله فریم به عنوان مدل در نظر گرفته می شود و این مدل با مرور زمان با استفاده از رابطه (۶-۱۰) اصلاح می شود، تا تغییرات محیط را پوشش دهد.

$$B_{x,y}^k = \begin{cases} (1-\alpha)B_{x,y}^{k-1} + \alpha I_{x,y}^k, & k > 1, \alpha \in [0,1] \\ I_{x,y}^0, & k = 0 \end{cases} \quad (6-10)$$

$I_{x,y}^k$ پیکسل واقع در سطر و ستون (x,y) در فریم k ام، α ضریبی است که میزان اثر فریم جاری و پس زمینه جاری را در پس زمینه جدید نشان می دهد و $B_{x,y}^k$ پیکسل واقع در سطر و ستون (x,y) در مدل پس زمینه در k امین فریم را نشان می دهد.

ناحیه که تغییرات آن محسوس می باشد. ۲

۳-۴-۶- تعیین ناحیه فعال

برای این که بتوانیم ناحیه فعال را پیدا کنیم اختلاف بین مدل و فریم جاری را بدست می آوریم، اگر اختلاف بیشتر از حد آستانه باشد، آن را جز به اشیا متحرک در نظر می گیریم. شکل ۱-۶ نمونه خروجی این مرحله را نشان می دهد.



شکل ۱-۶ ابتدا اختلاف بین پس زمینه و فریم جاری بدست می آوریم اختلاف بالاتر حد آستانه را نگه می داریم.

همانطور که می بینیم، در این تصویر نویز ضربه به وجود دارد. برای برطرف کردن آن می توانیم از فیلتر میانگین یا میانه استفاده کنیم. از این میان الگوریتم میانه عملکرد مناسب تری دارد. نتیجه اعمال فیلتر را بر شکل ۱-۶ در شکل ۲-۶ مشاهده می کنید. همانطور که می بینید نویزها تا حد زیادی حذف شده است.



شکل ۲-۶ نتیجه اعمال فیلتر میانه بر روی تصویر ۱-۶

۵-۶- آزمایش

الگوریتم جدید را با الگوریتم های رایج از جهت کیفیت تصویر بازیابی شده، زمان محاسبات و حجم محاسبات مورد ارزیابی قرار دادیم نتایج در ادامه آمده است.

۵-۶-۱- زمان محاسبات

هر دو الگوریتم مورد مطالعه را بروی دنباله ای از فریم ها، با تعداد فریم مختلف اعمال کردیم. زمان هر کدام از محاسبه الگوریتم ها در جدول ۴-۶ خلاصه شده است. همانطور که مشاهده می شود همواره زمان الگوریتم جدید کمتر از زمان محاسبات الگوریتم ها مرسوم می باشد. نمودار شکل ۳-۶ این داده ها را بر روی صفحه ترسیم می کند، هر چه تعداد فریم ها بیشتر شود اختلاف زمان محاسبات الگوریتم جدید نسبت به الگوریتم های دیگر بیشتر می شود. شیب الگوریتم جدید کندتر است یعنی با افزایش تعداد فریم ها، زمان با نسبت کمتری افزایش می یابد

جدول ۴-۶ مقایسه زمان محاسبات الگوریتم جدید با الگوریتم های رایج

تعداد فریم	زمان جستجو ۲-D logarithmic search	UCBD	۴SS	CSD	زمان جستجو برای بلوک مطلوب الگوریتم جدید
۱۰	۱,۰۱۴۳	۰,۷۷۸۸	۱,۰۰۲۳	۱,۱۲۲۷	۰,۱۴۴۶
۲۰	۱,۹۳۴۹	۱,۵۰۶۲	۲,۰۳۳۷	۲,۲۰۶۸	۰,۲۲۹۴
۳۰	۲,۹۸۵۵	۲,۲۲۲۴	۲,۹۳۹۱	۳,۳۵۲۶	۰,۳۵۱۲
۴۰	۳,۷۷۸۹	۳,۰۲۱۴	۳,۹۱۴۶	۴,۵۹۳۶	۰,۴۷۷۴
۵۰	۵,۲۱۶۰	۳,۸۴۲۴	۵,۰۳۷۶	۵,۷۷۲۰	۰,۵۶۸۲
۶۰	۸,۳۶۰۱	۵,۰۱۵۵	۵,۹۶۸۵	۶,۷۸۳۷	۰,۷۶۴۷
۷۰	۷,۵۰۷	۵,۸۱۵۳	۷,۲۹۸۶	۹,۱۸۴۰	۰,۸۹۹۱

پس از بررسی در نتایج مشاهده کردیم که از بین ۳۰۰ بلوک که در درون فریم ها وجود دارد بطور متوسط حدود ۵ ماکرو بلاک وجود دارد که در درون آن ها تغییرات وجود دارد که فقط در این

الگوریتم جستجو انجام می‌گیرد و سایر بلوک‌ها مورد بررسی قرار نمی‌گیرد حال به صورت دقیق‌تر میزان زمان صرف شده توسط هر کدام از الگوریتم‌ها را بررسی می‌کنیم.

الگوریتم ۲-D logarithmic search

۱. برای مرحله اول ۱۰ نقطه مورد بررسی قرار می‌دهد.
۲. مرحله دوم ۹ نقطه مورد بررسی قرار می‌دهد.
۳. مرحله سوم ۹ نقطه مورد بررسی قرار می‌دهد.
۴. در نتیجه برای یک بلوک ۲۸ مقایسه انجام می‌شود.
۵. پس کل فریم ۲۸ بار مورد بررسی قرار می‌گیرد.

الگوریتم CSD

این الگوریتم همانند ۲-D logarithmic search می‌باشد.

الگوریتم UCBD

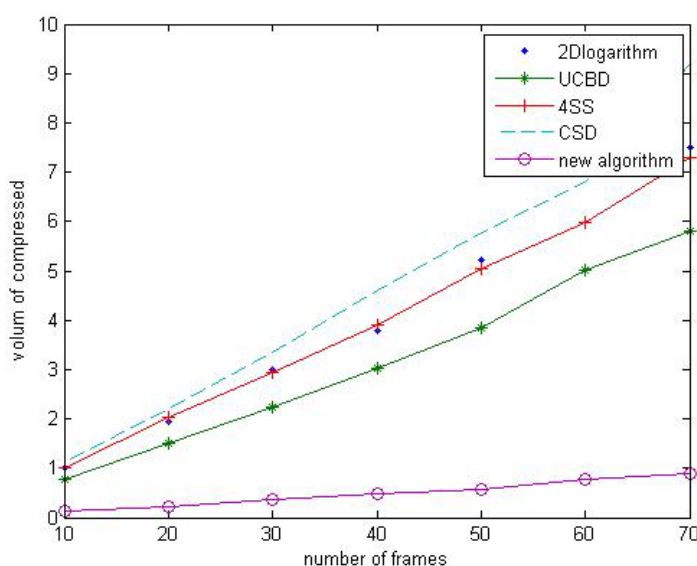
۱. در صورتی که نقطه مورد بررسی بر روی لبه باشد تعداد نقاط جدید برابر ۳ خواهد شد.
۲. اگر نقطه بر روی راس‌ها واقع شود تعداد نقاط جدید برابر ۵ خواهد شد.
۳. بنابراین در اولین گام لازم است تا ۹ نقطه و گام‌نهایی ۴ نقطه را بررسی کنیم.
۴. اگر تعداد مراحل که نقطه راس انتخاب می‌شوند برابر n_v و تعداد نقاط که بر روی لبه n_e باشد آنگاه کل تعداد نقاط مورد بررسی برابر $N = 9 + 5n_v + 3n_e + 4$ خواهد شد.
۵. اگر بطور متوسط تعداد نقاط مورد جستجو را برابر ۴ و تعداد گام‌ها را برابر ۳ بگیریم کل تعداد مقایسه‌ها ۲۱ خواهد شد.

الگوریتم ۴SS

۱. در این روش در بهترین حالت ۱۷ مقایسه و برای بدترین حالت از ۲۷ نقطه بررسی استفاده می‌کند.
۲. در بررسی‌ها نشان داده شده این روش بطور معمولاً ۶ مقایسه کمتر انجام می‌دهد و در بدترین حالت‌ها این تعداد مقایسه‌ها ۲ مقایسه کمتر انجام می‌دهد.

اما در الگوریتم جدید ما عملیات زیر داریم.

۱. بروز کردن پس زمینه که ۲ عملیات ضرب و ۱ عملیات جمع دارد پس کلا ۳ بار فریم مورد بررسی قرار می‌گیرد.
۲. تفریق فریم از پس زمینه ۱ محاسبه و اعمال حد آستانه ۱ مرور کل تصویر لازم است.
۳. از آنجا که فیلتر میانه را 3×1 در نظر گرفته ایم این عمل نیز ۳ جستجو لازم دارد. مشاهده می‌شود که عملیات سر بار ۸ بار جستجو کل فریم را نیاز دارد در عوض تعداد بلوک‌هایی که مورد جستجو قرار می‌گیرد. زیر $0,01$ می‌باشد. در نتیجه حجم محاسبات کاهش می‌یابد.



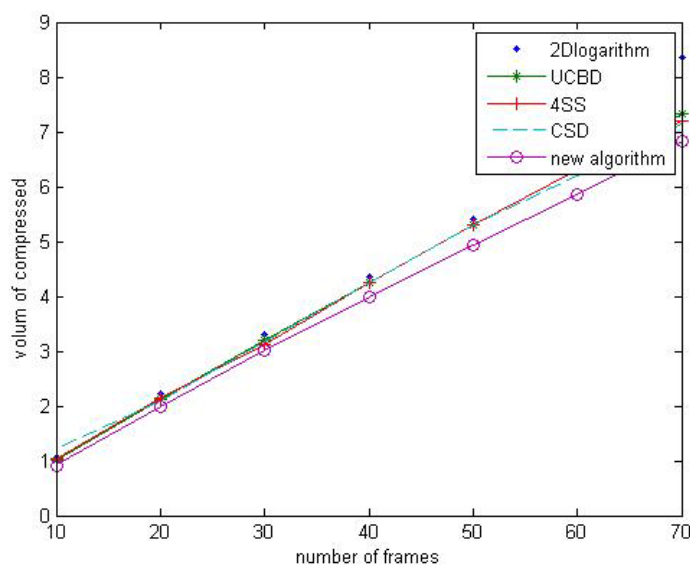
شکل ۳-۶ مقایسه الگوریتم جدید با الگوریتم رایج

۲-۵-۶- حجم فشرده سازی

دومین معیار مورد ارزیابی میزان فشرده سازی می‌باشد. همانطور که در جدول ۳-۶ مشاهده می‌کنید حجم فشرده سازی نسبت به الگوریتم‌های سابق کاهش یافته است. به همین طریق نمودار شکل ۴-۶ نشان می‌دهد همیشه حجم فشرده سازی جدید کمتر است.

جدول ۵-۶ مقایسه حجم فشرده الگوریتم جدید با الگوریتم های رایج

تعداد فریم	زمان جستجو 2-D logarithmic search	UCBD	4SS	CSD	زمان جستجو برای بلوک مطلوب الگوریتم جدید
۱۰	۱,۰۱۴۳	۰,۷۷۸۸	۱,۰۰۲۳	۱,۱۲۲۷	۰,۱۴۴۶
۲۰	۱,۹۳۴۹	۱,۵۰۶۲	۲,۰۳۳۷	۲,۲۰۶۸	۰,۲۲۹۴
۳۰	۲,۹۸۵۵	۲,۲۲۲۴	۲,۹۳۹۱	۳,۳۵۲۶	۰,۳۵۱۲
۴۰	۳,۷۷۸۹	۳,۰۲۱۴	۳,۹۱۴۶	۴,۵۹۳۶	۰,۴۷۷۴
۵۰	۵,۲۱۶۰	۳,۸۴۲۴	۵,۰۳۷۶	۵,۷۷۲۰	۰,۵۶۸۲
۶۰	۸,۳۶۰۱	۵,۰۱۵۵	۵,۹۶۸۵	۶,۷۸۳۷	۰,۷۶۴۷
۷۰	۷,۵۰۷	۵,۸۱۵۳	۷,۲۹۸۶	۹,۱۸۴۰	۰,۸۹۹۱



شکل ۴-۶ مقایسه حجم فشرده الگوریتم جدید با الگوریتم های رایج

۳-۵-۶- کیفیت تصویر

سومین معیار مورد ارزیابی اثر هر کدام از روش ها بر کیفیت تصاویر می باشد تصاویر نتایج هر یک از الگوریتم ها در شکل های ۵-۶ تا ۵-۹ آورده شده است. مشاهده می شود که کیفیت خروجی الگوریتم با هم برابری می کند.

۶-۶- نتیجه

یکی از زمانبرترین قسمت‌ها در فشرده‌سازی، انطباق بلوک می‌باشد. به همین دلیل الگوریتم‌هایی برای برطرف کردن این مشکل ارائه شده است. این الگوریتم‌ها تلاش می‌کند تا با کمتر کردن تعداد نقاط مورد جستجو، حجم محاسبات را کاهش دهند. در الگوریتم‌های رایج سعی می‌شود بدون در نظر گرفتن شرایط محیط تعداد محاسبات کاهش داده شود اما عیب کار آنجاست که برای تمام بلوک‌ها عمل جستجو را انجام می‌دهند. از آنجایی که فقط تعداد اندکی از بلوک‌ها تغییرات قابل توجه دارند، در این فصل ما با استفاده از روش‌های مدل‌سازی پس‌زمینه، نواحی فعال را تعیین کردیم و با محدود کردن جستجو به این نواحی در نتایج مشاهده شد که در این روش جدید بدون از دست دادن دقت می‌تواند حجم داده و زمان محاسبات را کاهش داد.



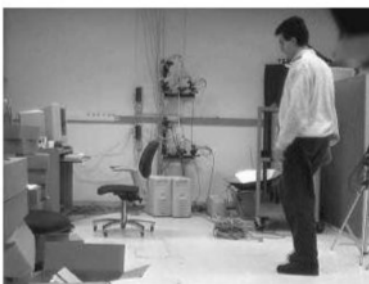
شکل ۶-۶ الگوریتم جدید



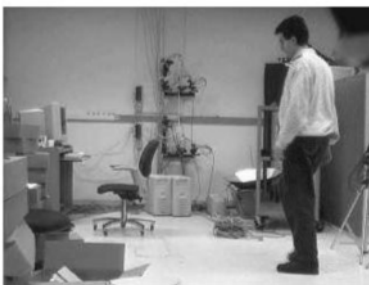
شکل ۶-۵ الگوریتم 2D logarithm



شکل ۶-۸ شکل بازیابی شده از الگوریتم UCBD



شکل ۶-۷ شکل بازیابی شده از الگوریتم CSD



شکل ۶-۹ شکل بازیابی شده از الگوریتم 4SS

فصل ہفتم

نتائج و پیشہدات

۷-۱- جمع بندی

رسانه های ویدیویی معمولا حجم زیادی از داده ها را تولید می کنند این حجم عظیم اطلاعات مشکلاتی از قبیل فضای لازم برای ذخیره کردن اطلاعات و پهنای باند کافی برای انتقال آن به همراه دارد. بنابراین فشرده سازی مسئله ای فراتر از یک امر لوکس می باشد. اصولا دنیای ویدیو دیجیتال حضورش را در بین اقشار جامعه و فراگیر شدنش را مدیون تکینک های فشرده سازی می باشد.

در حقیقت آنچه به ما این امکان را می دهد، تا به وسیله آن بتوان داده ها را با حجم بیشتری منتقل کرد، افزونگی اطلاعات می باشد. این افزونگی ها می توانند به دو دسته افزونگی بین فریم ها و افزونگی درون فریم ها تقسیم نمود در افزونگی بین فریم ها، حجم عمده یک فریم را پس زمینه تشکیل می دهد که در فریم های متوالی مشترک می باشد، آنچه تغییر می کند اشیا و تغییرات جزئی در پس زمینه می باشند (که توسط چشم انسان قابل تشخیص نمی باشند، مانند امواج دریا یا لرزش های آرام شاخه و برگ درختان) پس اگر بتوان پس زمینه را مدل کرد، می توان از ذخیره کردن حجم عمده پس زمینه فریم که تکراری هستند، صرفه نظر کرد و به این طریق به حجم بالایی از فشرده سازی در ویدیو دست یافت. این تز سعی دارد، با استفاده از مدل سازی پس زمینه از افزونگی بین فریم ها بهره گیرد تا سیستم فشرده سازی تصاویر ویدیویی را بهینه نماید.

در سیستم های رایج فشرده سازی ویدیو برای بهره گیری از افزونگی بین فریم ها از انطباق بلوک استفاده می کنند. سیستم فشرده سازی برای رسیدن به این هدف، ابتدا هر فریم را به بلوک هایی تقسیم می کنند هدف این است تا بین بلوک های فریم جاری و فریم مرجع یک تناظر برقرار شود. برای پیدا کردن تناظر بین یک بلوک از فریم جاری با بلوک های فریم مرجع، این بلوک را با مجموعه

ای از بلوک های فریم مرجع مقایسه می کنیم تا شبیه ترین بلوک را با بلوک جاری پیدا کنیم. ساده ترین راه حلی که به نظر می رسد این است که این بلوک را با تمام بلوک های فریم مرجع مقایسه کنیم، اما این فرایند به شدت زمان بر می باشد. بنابراین نیاز به روش هاست که با بررسی حداقل تعداد بلوک ها، بتواند سریع بلوک مطلوب را پیدا کند. الگوریتم های فراوانی برای رسیدن به این منظور وجود دارد. که ما به بررسی و مقایسه تعدادی از آنها پرداختیم، این الگوریتم ها عبارتند:

۱. الگوریتم جستجو (Unrestricted Center-Biased Diamond)

۲. الگوریتم جستجو (ARPS-ZMP)

۳. الگوریتم جستجو سه گام (Three-Step Search Algorithm)

۴. الگوریتم جستجو لوزی (DIAMOND SEARCH ALGORITHM)

۵. الگوریتم جستجو کراس ((CROSS-SEARCH ALGORITHM(CSA

۶. الگوریتم جستجو شش وجهی (Hexagon-Based Search Pattern)

۷. الگوریتم جستجو (2-D logarithm)

برای ارزیابی و مقایسه این الگوریتم ها، تعداد مقایسه ای که هر کدام از آنها به طور متوسط نیاز دارد تا بلوک مناسب را پیدا کند محاسبه شد. در نتایج نشان داده شد که الگوریتم های Unrestricted Diamond، Three-Step Search Algorithm، ARPS-ZMP، SEARCH ALGORITHM، Hexagon-Based Search، CROSS-SEARCH ALGORITHM، Pattern و 2-D logarithm به ترتیب به ۲۱، ۲۲، ۱۷، ۱۷، ۲۱ و ۳۳ مقایسه نیاز دارد. این تعداد مقایسه ها با این فرض بدست آمد که به طور متوسط هر کدام از الگوریتم ها ۴ گام جستجو انجام دهند.

ایرادی که بر روش های فوق وارد است این است که در این الگوریتم ها تمام بلوک ها مورد ارزیابی قرار می دهد چه در آنها تغییری اتفاق افتاده باشد چه اتفاقی نیفتاده باشد. در حالی که در بسیاری از

مواقع پس زمینه تغییر نسبتاً عمده ای ندارند. اگر بتوان پس زمینه را مدل کرد می توان از جستجو بلوک های آن صرف نظر کرد و به این طریق حجم محاسبات را کاهش دهیم. در این تز به بررسی چند الگوریتم مدل سازی پس زمینه پرداختیم این الگوریتم را می توان به دسته های زیر تقسیم کرده.

۱. مدل سازی پس زمینه با استفاده از الگوریتم میانه

۲. مدل سازی پس زمینه با استفاده از الگوریتم میانگین

• (Time Invariant Background Model) TIBM

• (Time Background Model) TBM

• (Long Term Average Background Model) LTABM

• (Movie Average Background Model) MABM

• Running Gaussian Average Background (Model) RGABM

• (improved Running Gaussian Average Background Model) IRGABM

۳. مدل سازی پس زمینه الگوریتم PCA

۴. مدل سازی پس زمینه با استفاده از روش های آماری

• الگوریتم های غیر پارامتریک

✓ مدل سازی با استفاده از هیستوگرام

✓ مدل سازی با استفاده از کرنل

• الگوریتم پارامتریک

✓ مدل سازی با استفاده از توزیع گوسی

✓ مدل سازی با استفاده از Mixture of Gaussian

این الگوریتم ها جهت زمان محاسباتی (مرتبه زمانی و زمان اجرا بر روی داده‌های واقعی)، میزان حافظه مورد نیاز، دقت در استخراج اشیا از پس زمینه در این پایان نامه مورد بررسی قرار گرفته‌اند. از جهت زمان محاسبات الگوریتم های میانه و میانگین نسبت به الگوریتم های دیگر برتری اما از جهت دقت، الگوریتم های آماری به خصوص در محیط های بیرونی کارایی مناسب تری دارند حجم حافظه مورد نیاز برای پیاده‌سازی الگوریتم های آماری به نسبت الگوریتم های میانگین و میانه بیشتر می باشد. میزان حافظه مصرفی در الگوریتم PCA در حدی زیاد است که عملاً نمی توان از آن استفاده کرد.

در روش های رایج مدل سازی ، روش های آماری علاوه بر کارایی در تقسیم کردن فریم ها به دو گروه اشیا و پس زمینه، از سه مشکل عمده (۱- بالا بودن حجم بالای محاسبات برای بدست آوردن مدل ۲- فرض مدلی خاص برای رفتار پس زمینه، در حالی که این فرض با واقعیت فاصله معنا داری داشت. ۳- و در نهایت در بسیاری از مواقع تعداد داده ای که برای آموزش مدل استفاده می شد ناکافی می باشد) رنج می برد. به همین دلیل لازم است تا از رهیافت دیگر (رهیافت جداکننده) که این مشکلات را ندارد، استفاده شود. برای مقابله با مشکلات فوق، با استفاده از رهیافت جداکننده، اشیا را از پس زمینه جدا می کنیم. در حالت ایده آل می توان داده ها را به دو کلاس پس زمینه و اشیا، که به صورت خطی جداپذیر می باشند، در نظر بگیریم. اگر مرز جداکننده تعیین شود، با استفاده از این تابع جداکننده، می توان اشیا را از پس زمینه جدا کرد. الگوریتم های جداکننده مورد بررسی در این پایان نامه عبارتند:

۱. حداقل مربعات خطا

۲. پرسپترون

۳. جدا کننده خطی با استفاده مدل آماری

۴. Iterative reweighted least squares

برتری این رهیافت این است که رهیافت نه فرضی در مورد مدل که داده ها را تولید می کند در نظر می گیرد. در ضمن زمانی که تابع جداکننده بدست آمد، دیگر لازم نیست مدام آموزش ببیند (اینکه هر چند فریم تابع باید بروز شود وابسته به محیط و تغییرات آن دارد) پس حجم محاسبات کمتری دارد. از طرفی برای آموزش آن لازم نیست تعداد زیادی داده در دست داشته باشیم.

از آنجایی که اختلاف بین سایه اشیا و پس زمینه قابل توجه می باشد، بنابراین معمولاً سایه اشیا نیز به عنوان شی در نظر گرفته می شود و در نتیجه اشیا استخراج شده دارای مشکلاتی می باشند (معمولاً شی و سایه آن به عنوان یک شی در نظر گرفته می شوند). روش های متفاوتی به منظور تشخیص سایه از اشیا وجود دارد. برای حذف سایه دو الگوریتم HSV و الگوریتم NCC استفاده کردیم که الگوریتم NCC دارای دقت مناسب تری می باشد.

اشیا استخراج شده دارای نویز هایی می باشند، بنابراین لازم است، با استفاده از تکنیک های مرسوم اثر آنها را کاهش دهیم، در نهایت به معرفی این تکنیک ها می پردازیم این تکنیک ها را می توان به دسته ای زیر تقسیم کرد.

۱. میانه

۲. میانگین

۳. CCL

۴. الگوریتم های ریخت شناسی (باز کردن، بستن، گسترش و سایش)

از این میان کارایی الگوریتم میانه و CCL مناسبتر از دو الگوریتم دیگر بود.

برای انطباق بلوک معیاری که برای اندازه گیری میزان شباهت بین بلوک استفاده می شود تابع همینگ می باشد در این تز ما به بررسی و مقایسه چند معیار دیگر با معیار همینگ پرداختیم. کمترین زمان کدینگ مربوط به تابع chebychev می باشد. علت این است که در این روش بعد از محاسبه کردن اختلاف بین بلوک ها، فقط لازم است تا ماکزیمم را بدست آوریم. این تابع حجم

محاسباتی کمی نسبت به توابع دیگر دارد، اما همانطور که می بیند زمانی که از این تابع استفاده می کنیم، حجم فشرده سازی بالا می رود، و از طرفی زمان دیکد نیز بالا می رود، بالا رفتن دیکد به این علت است که، این تابع بلوک ها را به درستی منطبق نمی کند، بنابراین زمان پردازش بالا می رود. بالا ترین زمان دیکد و کمترین حجم فشرده سازی متعلق به Monkowski می باشد. علت بالا بودن زمان دیکد این است که، این تابع نسبت به توابع دیگر پیچیده تر می باشد و نیاز به محاسبات زیادتری دارد. این روش علاوه بر این که نیاز به زمان محاسبات بالا دارد. حجم فشرده سازی آن نیز کم می باشد. به همین دلیل این روش، برای فشرده سازی ویدیو مناسب نمی باشد. یکی از دیگر نکاتی که می توان از آزمایش ها بدست آوریم این است که، زمان کدینگ نسبت به زمان دیکد بیشتر می باشد. و این مقدار در بعضی از مواقع، زمان محاسبات کدینگ دو برابر محاسبات دیکد می باشد. علت این پدیده این می باشد که در کد کردن، عملیات انطباق بلوک که عملیات زمان بری است وجود دارد، در حالی که این عملیات، در دیکد وجود ندارد.

و در نهایت، با استفاده از روش های مدل سازی پس زمینه نواحی که در آن تغییرات قابل توجه اتفاق می افتد را تعیین، و جستجو را به این ناحیه محدود می کنیم. الگوریتم جدید را با چهار الگوریتم جستجو دیگر (2-D logarithmic search، UCBD، 4SS و CSD) مقایسه کردیم. نتایج نشان داد که به طور متوسط برای هر فریم الگوریتم های 2-D logarithmic search، CBD، 4SS و CSD به ترتیب به ۰،۱۰۶۲، ۰،۰۷۸۰، ۰،۱۰۰۳ و ۰،۱۱۵۶ ثانیه و در حالی که الگوریتم جدید ما به ۰،۰۱۲۴ ثانیه نیاز دارد. اهمیت این بهبود زمانی مشخص می شود که بدانیم اگر فرکانس نمونه برداری برابر ۳۰ فریم بر ثانیه باشد آنگاه در یک ساعت (۱۰۸۰۰۰) فریم داریم آنگاه زمان محاسبات برای یک ساعت ویدیو در الگوریتم های 2-D logarithmic search، UCBD، 4SS و CSD به ترتیب ۱۰۸۳۲، ۸۴۲۴، ۱۱۴۷۰ و ۱۲۴۸۵ ثانیه در حالی که الگوریتم جدید ما ۱۳۳۹ ثانیه نیاز دارد که تقریباً یک دهم زمان محاسبات الگوریتم های یاد شده می باشد. نتایج همچنین نشان می دهد درصد فشرده سازی نیز به

نسبت الگوریتم های یاد شده بهبود قابل توجهی می کند تقریباً ۵۰٪ بهبود می یابد. از جهت کیفیت نیز این الگوریتم ها هم سطح می باشند.

۲-۷- کارهای آینده

۱. یکی از زمان برترین فرایندها در سیستم های فشرده سازی ویدیو، فرایند انطباق بلوک می باشد. بنابراین بهینه سازی بر روی این مولفه می تواند حجم محاسبات را کمتر کند. این بهینه سازی ها می تواند در سه قسمت دسته بندی کرد.

- انتخاب معیار مناسب برای مقایسه شباهت دو بلوک با هم برای تعیین میزان دو این معیار علاوه بر اینکه باید از بین بلوک ها مناسب ترین بلوک را انتخاب کند باید حجم محاسبات کمتری داشته باشد.

- انتخاب هوشمندانه بلوک ها بسیاری از بلوک ها که در الگوریتم های انطباق بلوک جستجو می شوند لازم نیست بررسی گردند. مثلاً بلوک هایی که در مجاورت هم قرار می گیرند دارای الگوی حرکتی مشابه می باشند از این اطلاعات می توان استفاده کرد تا محدوده جستجو را محدود کرد.

- بسیاری از نواحی دارای تغییرات خاصی نیستند بنابراین لازم نیست که جستجو در آن نقاط انجام بگیرد. اما مسئله این است که چگونه می توان این نواحی را تعیین کنیم.

۲. در روش های جدا کننده اگر محیط دارای تغییرات سریع باشد لازم است تا به ازای هر چند فریم وزن های و پارامتر های الگوریتم بهینه شوند تا بدین منظور بایستی پاسخی برای سوالات زیر یافته شود:

- به ازای هر چند فریم لازم است تا وزن های تغییر کند؟

- چگونه وزن های شبکه آموزش دیده، را با حداقل تغییرات به گونه تنظیم کنیم، تا دوباره دقت آن بهبود بیابد؟
- از چه پارامتر های محیطی، می توان استفاده کرد تا میزان پویایی محیط را درک کرد؟

مراجع و منابع

[1] <http://mpeg.org/>

[2] Ahmed Elgarni, L. Ramani Duraiswami, David Harwood AND Larry S. Davis (2002) "**Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance**". IEEE pp 1101-1113.

[3] Bishop, C. M. (2006). "**Pattern Recognition and Machine Learning**". Springer.

[4] C. Stauffer, W.E.L. Grimson(2000) "**Learning Patterns of Activity Using Real-Time Tracking**". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 22, No. 8.

[5] Cavallaro, A. Salvador, E. Ebrahimi (2004). "**Detecting shadows in image sequences**". Visual Media Production(CVMP)", pp 160-174.

[6] Ce Zhu, Xiao Lin and LapPui Chau(2002) "**Hexagon-Based Search Pattern for Fast Block Motion Estimation**" IEEE Transactions on Pattern Analysis and Machine Intelligence, VOL. 24, NO. 8.

[7] Chen, Shu-Te Su and Yung-Yaw(2006) "**Moving Object Segmentation Using Improved Running Gaussian Average Background Model**." Digital Image Computing: Techniques and Applications, pp 24-30.

[8] Feig L. K. Liu (1996). "**A block-based gradient descent search algorithm for block motion estimation in video coding**". IEEE Trans. Circuits Syst. Video Technol, vol. 6, pp. 419-423.

[9] GHANBARI. M (1990). "**The Cross-Search Algorithm for Motion Estimation**". IEEE Transaction On Communication, VOL. 38, NO. 1.

[10] Jain J. R. Jain A. K. (1987). "**Displacement measurement and its application**". IEEE Transaction On Communication, VOL. 35, NO. 1.

[11] L. Hanzo, P. J. Cherriman and J. Streit (2007). "**Video Compression and Communications**". WILEY

[12] Ma, L. M. Po and W. C. (1996). "**A novel four-step search algorithm for fast block motion estimation**". IEEE Trans. Circuits Syst. Video Technol, vol. 6, pp. 312-317.

[13] Ma, Shan Zhu and Kai-Kuang (2000). "**A New Diamond Search Algorithm for Fast Block-Matching Motion Estimation**". IEEE TRANSACTIONS ON IMAGE PROCESSING, VOL. 9, NO. 2.

[14] Oliver, N., B. Rosario, and A. Pentland (1999). "**A Bayesian Computer Vision System for Modeling Human Interactions**". Int'l Conf. on Vision Systems Springer.

[15] Rafael C. Gonzalez, Richard E. Woods (2001). "**Digital Image Processing**". Prentice Hall

[16] Schofield, N. McFarlane and C. (1999). **"Segmentation and tracking of piglets in images"**. Machine Vision and Applications ,pp 187-193.

[17] Sergios Theodoridis, Konstantinos Koutroumbas (2003). **"PATTERN RECOGNITION"** Elsevier (USA).

[18] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro (1991). **"Motion compensated interframe coding for video conferencing"**. Telecommunication Conf

[19] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein (2001) **"Introduction to Algorithms"**, Second Edition. MIT Press.

[20] Wendy L. Martinez, Angel R. Martinez (2002). **"Computational Statistics Handbook with Matlab"**. CRC Press.

[21] Yao Nie, Kai-Kuang Ma (2002). **"Adaptive Rood Pattern Search for Fast Block-Matching Motion Estimation"**. IEEE TRANSACTIONS ON IMAGE PROCESSING , VOL. 11, NO. 12.

[22] Yun Q. Shi, Huifang Sun (2008). **"Image and Video compression for Multimedia Engineering"**. CRC Press.

[23] A. P. Dempster, N. M. Laird, and D. B. Rubin. (1977) **"Maximum likelihood from incomplete data via the EM algorithm"**. Journal of the Royal Statistical Society. Series B (Methodological), pp:1-38

[24] P. Blauensteiner, H. Wildenauer, A. Hanbury, and M. Kampel. (2006) **"On colour spaces for change detection and shadow suppression"**. In Proc. of the 11th CVWWS Computer Winter Vision Workshop, pp 117-123, February 7-8,

[25] T. Horprasert, D. Harwood, and L.S. Davis. (1999) **"A statistical approach for real-time robust background subtraction and shadow detection"**. In Proceedings of IEEE ICCV'99 FRAME-RATE Workshop

[26] Erum A. Khan and Erik Reinhard. (2004) **"A survey of color spaces for shadow identification"**. In APGV '04: Proceedings of the 1st Symposium on Applied perception in graphics and visualization, pp 160-166, New York, NY, USA, ACM Press.

[27] P. Kumar, K. Sengupta, and A. Lee. (2002) **"A comparative study of different color spaces for foreground and shadow detection for traffic monitoring system"**. In Intelligent Transportation Systems, 2002. Proceedings. The IEEE 6th International Conference on, pp 100-106,

[28] Nicolas Martel-Brisson and Andre Zaccarin. (2005) **"Moving cast shadow detection from a gaussian mixture shadow model."** In CVPR '05: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)- Volume 2, pp 743-748

[29] Andrea Prati, Ivana Mikic, Mohan M. Trivedi, and Rita Cucchiara. (2003) **"Detecting moving shadows: Algorithms and evaluation"**. IEEE Trans. Pattern Anal. Mach. Intell., pp 918-923,

[30] Piccardi, M. (2004) **"Background subtraction techniques: a review."** Systems, Man and Cybernetics, IEEE International Conference, Vol 4, , pp.3099-3104.

[31] Jain, R., Kasturi, R., Schunk, B.G. (2003), **"Machine Vision"**, McGraw-Hill Inc., pp. 63- 69.

[٣٢] A.N.Netravali and B.G.Haskell,(١٩٩٥) "**Digital Pictures representation , compression, Standard**", ٣nd ed,Plenum Press,

[٣٣] Y.Wang, J.Ostermann and Y.-Q.Zhang, (٢٠٠٢) "**video processing and communication**", Prentice Hall,

[٣٤] Duda, R. O. and P. E. Hart (١٩٧٣). "**Pattern Classification and Scene Analysis**". Wiley.

[٣٥] Rosenblatt, F. (١٩٦٢). "**Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms**". Spartan.

[٣٥] Bishop, C. M. and I. T. Nabney (٢٠٠٨). "**Pattern Recognition and Machine Learning: A Matlab Companion**". Springer. In preparation.

[٣٦] Gonzalez,Woods,Eddins(٢٠٠٩)."**Digital Image Processing Using Matlab**" [Gatesmark Publishing](#).

[٣٧] Wei-Yi Wei(٢٠٠٤). "**Image Compression Tutorial**" National Taiwan University, Taipei, Taiwan, ROC

[٣٨] <http://www.dogma.net/markn/index.html>

[٣٩] Michael Isnardi(١٩٩٩) "**MPEG-٢ Video Compression**" Sarnoff Corporation

Abstract

The video devices usually produce huge data. There is a challenge for saving or transporting this massive data. The video compression systems are devised to figure out this problem. These systems use interframe redundancy to compress this enormous data.

For detecting the redundancy, frames are divided to some same size blocks afterward block-matching component of compression system examines these blocks, to avoid from saving repeated block. The block-matching component is very time consuming component in the video compression system. In addition, if this component does not operate properly subsequently compression percent would be reduced. However, in common video compression systems such as MPEG2, explore all the blocks as a result these systems need enormous computation.

In these thesis by using background modeling, initially active regions are identified, the active region means a region of frame which moving objects are placed in it, after that we limit the block-matching operation to these regions. To detect active regions, in first step, a model of background is identified and noises are removed, then output of previous stage submits to another component. In this component, pixels of the frame are classified to three categories: object, shadow and background class. Finally by employment these information, the active regions are detected and the block-matching operations are limited to these regions. The outputs of proposed approach are judged against of the four methods (2D- logarithm search, UCBD, FSS and CSD). The results demonstrate the new optimal system, is very faster than another methods. Hence, the experiments express volume of output of new method is half of volume of output of non-optimal systems.

Keywords: *Video compression, Background Modeling, Image Compression, Similarity Measure and MPEG.*



Shahrood University of Technology

Faculty of Computer and IT

Video Image Compression using of Background Modeling

Mehdi Sedighi

Supervisor

DR. Hamid Hassanpour

Advisor

DR. Ali Akbar Pouyan

Date: Bahman ۸۹

