
بسم الله الرحمن الرحيم

دانشکده : برق

گروه : برق – مخابرات سیستم

تعیین بی‌درنگ سرعت وسائل نقلیه بر روی یک برد توسعه بر مبنای پردازنده ی *ARM* با استفاده از پردازش ویدیو

دانشجو : سید امیر آقایان

استاد راهنما :

دکتر حسین خسروی

پایان نامه ارشد جهت اخذ درجه کارشناسی ارشد

زمستان ۹۶

شماره ۱۵۳۴/ت/۰۰
تاریخ: ۹۶/۱۱/۴

باسمه‌تعالی

دانشگاه صنعتی شاهرود

مدیریت تحصیلات تکمیلی

فرم شماره (۳) صورتجلسه نهایی دفاع از پایان نامه دوره کارشناسی ارشد

با نام و یاد خداوند متعال، ارزیابی جلسه دفاع از پایان نامه کارشناسی ارشد خانم / آقای **سید امیر آقایان** با شماره دانشجویی **۹۴۰۲۵۱۴** رشته **مهندسی برق- مخابرات** گرایش **سیستم** تحت عنوان : **تعیین بی درنگ سرعت وسایل نقلیه بر روی یک بردتوسعه برمبنای پردازنده یARM با استفاده ازپردازش ویدیو** که درتاریخ **۱۳۹۶/۱۱/۰۴** با حضور هیأت محترم داوران در دانشگاه صنعتی شاهرود برگزار گردید به شرح ذیل اعلام می گردد:

قبول (با درجه: عالی) ☑ مردود ☐

نوع تحقیق: نظری ☑ عملی ☐

عضو هیأت داوران	نام ونام خانوادگی	مرتبه علمی	امضاء
۱ـ استادراهنمای اول	حسین خسروی	استادیار	
۲- استادراهنمای دوم			
۳- استاد مشاور			
۴- نماینده تحصیلات تکمیلی	ولی ا... شیخی	استادیار	
۵- استاد ممتحن اول	سید محمد سجاد ایوری	دانشیار	
۶ـ استاد ممتحن دوم	هادی گرایلو	استادیار	

نام و نام خانوادگی رئیس دانشکده:

تاریخ و امضاء و مهر دانشکده:

تبصره: در صورتی که کسی مردود شود حداکثر یکبار دیگر (در مدت مجاز تحصیل) می تواند از پایان نامه خود دفاع نماید (دفاع مجدد نباید زودتر از ۴ ماه برگزار شود).

تقدیم اثر

ماحصل آموخته‌هایم را تقدیم می‌کنم به آنان که مهر آسمانی‌شان آرام بخش آلام زمینی‌ام است.

به استوارترین تکیه‌گاهم،دستان پرمهر پدرم

به سبزترین نگاه زندگیم، چشمان سبز مادرم

که هرچه آموختم در مکتب عشق شما آموختم و هرچه بکوشم قطره‌ای از دریای بی‌کران مهربانیتان را سپاس نتوانم بگویم.

تشکر و قدردانی

از جناب آقای دکتر حسین خسروی بسیار سپاسگزارم چرا که بدون راهنمایی‌های ایشان انجام این پایان نامه بسیار مشکل می‌نمود. همچنین لازم می‌دانم از اساتید بزرگوار جناب آقای دکتر هادی گرایلو و جناب آقای دکتر سید علی سلیمانی ایوری به منظور تقبل مسئولیت داوری این پایان‌نامه صمیمانه سپاسگزاری نمایم.

تعهد نامه

اینجانب سید امیر آقایان دانشجوی دوره کارشناسی ارشد رشته برق – مخابرات سیستم دانشکده برق و رباتیک دانشگاه صنعتی شاهرود نویسنده پایان نامه تعیین بی‌درنگ سرعت وسائل نقلیه بر روی یک برد توسعه بر مبنای پردازنده ی ***ARM*** با استفاده از پردازش ویدیو، تحت راهنمائی جناب آقای دکتر خسروی متعهد می‌شوم.

- تحقیقات در این پایان نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است .
- در استفاده از نتایج پژوهشهای محققان دیگر به مرجع مورد استفاده استناد شده است .
- مطالب مندرج در پایان نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است .
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می باشد و مقالات مستخرج با نام « دانشگاه صنعتی شاهرود » و یا « *Shahrood University of Technology* » به چاپ خواهد رسید .
- حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان نامه تأثیرگذار بوده اند در مقالات مستخرج از پایان نامه رعایت می گردد.
- در کلیه مراحل انجام این پایان نامه ، در مواردی که از موجود زنده (یا بافتهای آنها) استفاده شده است ضوابط و اصول اخلاقی رعایت شده است .
- در کلیه مراحل انجام این پایان نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده است اصل رازداری ، ضوابط و اصول اخلاق انسانی رعایت شده است .

تاریخ

امضای دانشجو

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد . این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود .
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی باشد.

چکیده

در سال‌های اخیر، با افزایش تعداد خودروها، برقراری امنیت جاده‌ها از اهمیت بالایی برخوردار شده است. یکی از مهم‌ترین دلایل تصادفات جاده‌ای، رعایت نکردن سرعت مطمئنه در جاده‌ها است. به منظور کنترل ترافیک، اعمال قانون و به عنوان یک راه پیش‌گیری، تخمین سرعت خودروها دارای اهمیت است.

در این پایان‌نامه، هدف تعیین بی‌درنگ سرعت وسائل نقلیه به کمک روش‌های بینایی ماشین و پردازش تصویر است. برای تشخیص سرعت خودرو دو روش با استفاده از پردازش ویدیو و به کمک کتابخانه‌ی بینایی ماشین ***OpenCV*** و زبان برنامه‌نویسی ***C++*** مورد آزمون قرار گرفته است.

روش اول مبتنی بر استخراج مدل پس‌زمینه و الگوریتم خودهمبستگی است. در این روش در ابتدا مدلی از پس‌زمینه به کمک الگوریتم مخلوط گوسی به دست می‌آید. سپس محدوده‌ای از تصویر را به منظور اندازه‌گیری سرعت انتخاب و یکسو می‌نماییم. توسط عملیات همبستگی بین ناحیه‌ی انتخاب شده‌ی متناظر در مدل پس‌زمینه و تصویر اصلی، ورود خودرو به این ناحیه را تشخیص داده و در نهایت بر اساس زمان حضور خودرو در این ناحیه و همچنین میزان مسافت پیموده شده در این ناحیه (طول ناحیه) و ضریب تبدیل پیکسل بر ثانیه به کیلومتر بر ساعت، سرعت خودرو به دست می‌آید.

در روش دوم، از الگوریتم‌های پیچیده‌ی ردیابی صرف‌نظر کرده‌ایم. در ابتدا ناحیه‌ی مطلوب برای تشخیص سرعت خودرو بر روی تصویر انتخاب شده و سپس این ناحیه توسط تبدیلات افکنشی یکسو می‌شود. پیش‌زمینه، شامل اشیاء متحرک، توسط تفاضل قاب‌های متوالی به دست می‌آید. بازه‌ی جابه‌جایی پیموده شده‌ی هر خودرو توسط جابه‌جایی مرکز ثقل طی قاب‌های متوالی در محدوده‌ی تعیین شده، اندازه‌گیری می‌شود. زمان پیموده شدن این مسافت نیز از طریق شمارش تعداد قاب‌ها به دست می‌آید. در نهایت با تقسیم بازه‌ی جابه‌جایی پیموده شده بر زمانِ مربوط به هر خودرو و ضرب آنها در ضریبی که پیکسل بر ثانیه را به کیلومتر بر ساعت تبدیل می‌کند، سرعت خودرو به دست می‌آید.

این دو روش بر روی لپ‌تاپ لنوو مدل ***z510*** با پردازنده‌ی ***Intel core i5*** با فرکانس کاری ۲,۵ گیگاهرتز و رم ۶ گیگابایتی و همچنین بر روی برد توسعه‌ی ***Odroid XU4*** با پردازنده‌ی ۸ هسته ای با فرکانس کاری ۲ گیگاهرتز و رم ۲ گیگابایت پیاده سازی شد. زمان اجرای برنامه در روش اول بر روی لپ تاپ۵۵ میلی ثانیه و بر روی برد توسعه ۱۰۵ میلی ثانیه اندازه‌گیری شد. خطای سرعت اعلام شده ۴/۹۹% با انحراف معیار ۴/۸۳% و اختلاف ۲/۲۶± کیلومتر بر ساعت گردید. در روش دوم زمان

اجرای برنامه بر روی لپ تاپ ۳۱ میلی ثانیه و بر روی برد ۴۸ میلی ثانیه گردید. همچنین خطای سرعت اعلام شده ۳/۳۱% با انحراف معیار ۳/۲۸% و اختلاف ۱/۳۹± کیلومتر بر ساعت شد.

کلمات کلیدی– پردازش ویدئو، تبدیل افکنشی، تخمین سرعت، تفاضل قاب ها، عملیات ریخت شناسی.

لیست مقالات مستخرج از پایان نامه

آقایان، س.ا و خسروی، ح.، "تخمین بی درنگ سرعت خودرو از طریق دوربین به‌کمک ردیابی مرکز ثقل و پیاده‌سازی آن روی برد توسعه‌ی ***XU4***"، دهمین کنفرانس بینائی ماشین و پردازش تصویر ایران، دانشگاه صنعتی اصفهان، آذر ماه ۱۳۹۶.

فهرست مطالب

فهرست اشکال

فهرست جداول

فهرست نمودارها

فصل اول: مقدمه

۱–۱ مقدمه

در سال‌های اخیر، با افزایش تعداد خودروها، برقراری امنیت جاده‌ها از اهمیت بالایی برخوردار شده است. بدین منظور تحقیقات زیادی برای نظارت بر جریان‌های ترافیکی صورت گرفته است. هوشمندسازی سیستم‌های ترافیکی راه حلی مناسب برای افزایش دقت و صرفه جویی در وقت و هزینه است. از این رو محققان زیادی راه حل‌های وسیعی را به همین منظور پیشنهاد داده‌اند. سامانه-ی نظارتی بصری هوشمند، شامل نظارت بی درنگ اشیای ثابت یا گذرا درون یک محیط خاص است. هدف اصلی این سیستم‌ها فراهم نمودن یک تفسیر خودکار از صحنه‌ها و درک و پیش بینی کنش و واکنش‌های شی مشاهده شده بر اساس اطلاعات به دست آمده از سنسورها است. فرآیند اصلی در یک سیستم نظارتی هوشمند شامل تشخیص و شناسایی شی در حال حرکت، ردیابی، تحلیل های رفتاری و بازیابی است. این فرآیندها اصولا تکنیک‌های بینایی ماشین، تحلیل‌های الگویی، هوش مصنوعی و مدیریت داده را اعمال می‌کنند.

با افزایش تقاضا برای امنیت جامعه نیاز برای فعالیت‌های نظارتی در بسیاری از محیط‌ها شامل فرودگاه‌ها، دریا و خطوط راه آهن و مکان‌های عمومی، بیش از پیش شده است.

یکی از مهم‌ترین دلایل تصادفات جاده‌ای، رعایت نکردن سرعت مطمئنه در جاده‌ها است. به منظور اعمال قانون و به عنوان یک راه پیش‌گیری، تخمین سرعت خودروها دارای اهمیت زیادی است. سرعت خودرو می‌تواند توسط تکنولوژی‌های تخریبی[۱] یا غیر تخریبی[۲] اندازه‌گیری شود که در ادامه به بیان هر یک از این روش‌ها و مزایا و معایب هر یک می‌پردازیم.

۱–۲ تعریف مساله، مشکلات موجود و ضرورت انجام تحقیق

به منظور تعیین سرعت وسائل نقلیه، دو تکنولوژی تخریبی و غیر تخریبی، به کار می‌رود. سیستم‌های تخریبی، معمولا بر اساس آشکارسازهای حلقه‌ی سلفی هستند که حساس و دقیق هستند اما هزینه‌ی نصب بالایی دارند و می‌تواند به جاده آسیب برساند. سیستم‌های غیر تخریبی اغلب بر اساس سنسورهای لیزری، مادون قرمز، رادار داپلری یا سنسورهای صوتی و دوربین هستند. استفاده از تجهیزات راداری و لیزری در مقایسه با یک سامانه‌ی دوربین با دقتی مشابه، گران قیمت‌تر هستند. همچنین تشخیص سرعت بیش از یک خودرو به طور هم‌زمان توسط یک روش مبتنی بر تجهیزات راداری و لیزری امکان‌پذیر نیست در حالی که به کمک دوربین و الگوریتم‌های پردازشی می‌توان سرعت چندین خودرو را به طور هم‌زمان اندازه‌گیری کرد. همچنین ساخت یک بستر مناسب بر روی

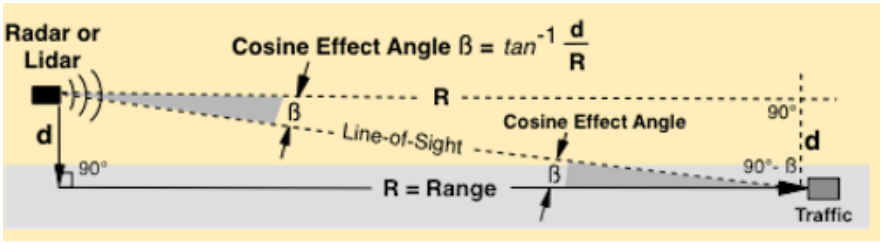
[۱] ***Intrusive***

[۲] ***Non- intrusive***

جاده برای قرارگیری تجهیزات راداری و لیزری بسیار ضروری است در حالی که نصب یک دوربین ویدئویی ملاحظات خاصی ندارد. یک مزیت دیگر به کارگیری دوربین این است که در کنار محاسبه‌ی سرعت، اطلاعات مختلف دیگری از وضعیت ترافیکی نیز می تواند ارائه شود. اگر اطلاعات سرعت بتواند از داده‌ی تصویر در دسترس استخراج شود ما به یک سیستم غیر تخریبی می‌رسیم که به طور قابل توجهی هزینه‌ها را کاهش می‌دهد. اطلاعات مربوط به سرعت خودروها ممکن است صرفا برای کنترل رانندگانی که از حد سرعت تجاوز می کنند نباشد بلکه برای سیستم‌های کنترل ترافیک به کار روند.

در سرعت سنج‌های لیزری دستی، به وجود یک اپراتور نیاز است که از جمله مشکلات آن بروز خطرات جانی مخصوصا در بزرگراه‌های پرتردد است. در این سرعت سنج‌ها مبنای اندازه‌گیری سرعت، فاصله‌ی زمانی بین دو بازگشت متوالی اشعه‌های لیزری است. در سرعت سنج‌های لیزری محل استقرار تجهیزات باید ثابت باشد و همچنین یکی از مشکلات اصلی استفاده از لیزر خطای کسینوس زاویه‌ی لیزر با جهت حرکت خودرو است به این معنا که بهترین موقعیت برای استفاده از آن رو به روی خودرو است. اگر موج به صورت زاویه‌دار به خودرو برخورد نماید سرعت محاسبه شده توسط لیزر در کسینوس زاویه‌ی بین لیزر و حرکت ماشین ضرب می‌شود و در بدترین حالت که عمود بر خودرو است لیزر سرعت صفر را نشان می‌دهد***[1]*** .

خطای کسینوس زاویه‌ی لیزر با جهت حرکت خودرو از طریق فرمول زیر قابل محاسبه است***[2]***:



شکل ۱-۱ خطای کسینوس زاویه‌ی لیزر با جهت حرکت خودرو ***[2]***

سرعت اندازه‌گیری شده = $V_m = v_0 \cos\beta = v_0 \frac{R}{(R^2+d^2)^{0.5}}$ (معادله ۱-۱)

در فرمول بالا ***V0*** سرعت اصلی هدف، β زاویه ی اثرکسینوسی، ***R*** فاصله ی هدف تا رادار و ***d*** نیز فاصله ی آنتن تا وسط خط هدف می باشد.

در سرعت سنج‌های راداری اصول کار بر مبنای پدیده‌ی داپلر است. موج رادیویی از دستگاه خارج می‌شود، به جسم برخورد کرده و با مقداری تغییر (بسته به سرعت جسم) به دستگاه باز می-گردد. با محاسبه‌ی زمان رفت و برگشت موج و مقدار تغییر فرکانس آن، فاصله و سرعت جسم محاسبه می‌گردد. از جمله مشکلات اصلی این سرعت‌سنج‌ها، در ادامه آورده شده است:

۱- محل و زاویه‌ی آنتن: به دلیل آنکه معیاری دقیق برای محدوده‌ی تحت نظر رادار وجود ندارد و رادار نیز تفکیک زاویه‌ای ندارد، در صورت وجود چند شی روبروی آنتن رادار، نمی‌توان اطمینان حاصل کرد که سرعت داده شده توسط رادار مربوط به کدام شی است.

۲- دید عبوری: حتی زمانی که محل نصب و زاویه‌ی رادار بسیار دقیق تنظیم شده باشد، به دلیل آنکه امواج رادیویی تا حدی از اشیاء عبور می‌کنند، ممکن است بازتاب قوی‌تری با شی پشتی ایجاد شود تا با شی جلویی. پس به جای سرعت جسم نزدیکتر، سرعت جسم پشتی نشان داده می‌شود.

۳- تداخل اشیا: در صورتی که رادار بر روی جسم متحرکی نصب شده باشد، سرعت محاسبه شده توسط رادار نسبی بوده، تغییر سرعت‌های ناگهانی شی حامل رادار در سرعت محاسبه شده توسط رادار تاثیر ناگهانی گذاشته، خطای خروجی رادار را بالا می‌برد و آن را غیر قابل استناد می‌کند.

۴- خطای کسینوس: مانند خطای سرعت سنج لیزری، کسینوس زاویه‌ی مابین جهت حرکت شی و جهت موج رادار در سرعت واقعی ضرب شده و سرعتی کمتر از سرعت واقعی گزارش داده می‌شود.

۵- جهش مجدد: موج رادیویی تنها یک بار بازتاب نمی‌کند. به این معنا که ممکن است قسمتی از موج فرستاده شده ابتدا به شی اول و سپس به شی دوم برخورد کرده و در انتها به آنتن رادار برسد. در چنین حالتی سرعتی متشکل از مجموع یا اختلاف دو سرعت در رادار دیده می‌شود.

۶- خطای بازتاب: وابسته به نوع و محیط نصب، در صورتی که محافظی در جلوی رادار قرار داده شود (برای مثال شیشه‌ی محافظ یا شیشه‌ی جلوی خودرو)، ممکن است بازتاب اولیه‌ای بر روی جنس محافظ انجام شده و مقداری از پالس موج به پشت رادار فرستاده شود و به اشتباه سرعت جسم پشت رادار را نمایش دهد.

۷- خطای تابلوی راهنما: همان خطای بازتاب مجدد در حالت وجود تابلوی فلزی بزرگ در محدوده‌ی روبروی رادار است که ایجاد خطا در سرعت‌سنجی می‌کند.

۸- تداخل رادیویی: به دلیل استفاده‌ی رادار از امواج رادیویی، امواج رادیویی دیگر با آن تداخل کرده، سرعت‌های بدست آمده را غیر قابل استناد می‌کند. برای مثال، بیسیم پلیس، دکل‌های فشار قوی برق و دستگاه‌های مایکروویو در رادار ایجاد خطا می‌کنند.

۹- تداخل پروانه : این اصطلاح به دلیل تاثیر چرخش پروانه‌ی خنک کننده‌ی خودرو بر رادار بوجود آمده و خاص پروانه نیست. به صورت کلی رادار سرعت جسم روبروی خود را تعیین

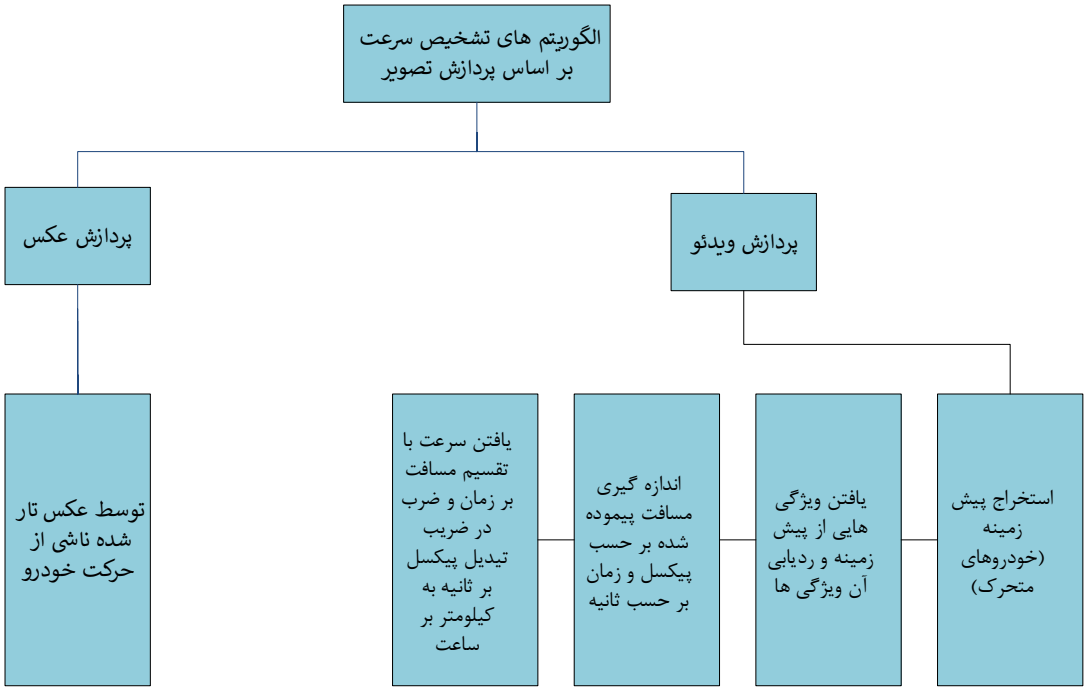
می‌کند. این سرعت ممکن است سرعت خودروی مقابل، سرعت پروانه‌ی آن خودرو یا سرعت لرزش بار داخل آن خودرو باشد***[1]*** .

در این پایان‌نامه، به منظور تعیین سرعت وسائل نقلیه، دو روش مبتنی بر سامانه‌های دوربینی پیشنهاد شده است. در این روش‌ها از الگوریتم‌های بینایی ماشین و پردازش تصویر استفاده شده و بر روی برد توسعه‌ی ***Odroid XU4*** بر مبنای پردازنده‌ی ***ARM*** پیاده‌سازی شده اند.

۱–۳ راه حل‌های موجود

در رابطه با موضوع تخمین سرعت به کمک پردازش تصویر کارهای زیادی از گذشته تا به امروز صورت گرفته است. در ادامه به معرفی روش‌های موجود تخمین سرعت وسائل نقلیه به کمک پردازش تصویر و کارهای انجام شده در این رابطه توسط سایر افراد، می‌پردازیم. در فصل ۲ به بررسی دقیق‌تر این روش‌ها خواهیم پرداخت.

در دیاگرام بلوکی شکل ۱-۲ انواع روش‌های تخمین سرعت خودرو در بلوک دیاگرام زیر آورده شده است.



شکل ۱-۲ بلوک دیاگرام مربوط به انواع روش‌های تخمین سرعت خودرو

نقطه‌ی تمایز این روش‌ها در نحوه‌ی یافتن اشیای پیش‌زمینه (خودرو) و همچنین در روش‌های ردیابی آن‌ها به منظور اندازه‌گیری سرعت می‌باشد.

چهار روش به منظور یافتن اشیای پیش‌زمینه مطرح شده است که عبارتند از:

- تفاضل مدل پس‌زمینه
- تفاضل دو قاب متوالی
- تفاضل سه قاب متوالی
- ترکیب روش‌های تفاضل مدل پس‌زمینه و تفاضل قاب‌های متوالی

همچنین برای ردیابی خودروها از روش‌های زیر بهره گرفته شده است:

- شار نوری
- فیلتر کالمن
- الگوریتم ***Hungarian***
- الگوریتم تطبیق بلوکی و همبستگی

ویژگی‌های که به منظور ردیابی انتخاب می‌شوند، عبارتند از:

- نقاط گوشه‌ای و لبه‌ها
- مرکز ثقل
- متن پلاک
- ویژگی‌های ***SIFT***

۱-۴ راه حل پیشنهادی

در این پایان‌نامه، هدف تعیین بی‌درنگ سرعت وسائل نقلیه به کمک روش‌های بینایی ماشین و پردازش تصویر است. دو روش برای این منظور به کمک کتابخانه‌ی بینایی ماشین ***OpenCV*** و زبان برنامه‌نویسی ***C++*** مورد آزمون قرار گرفته است.

در ابتدا روشی مبتنی بر استخراج مدل پس‌زمینه و الگوریتم خودهمبستگی[۱] ارائه شده است. در این روش در ابتدا مدلی از پس‌زمینه به کمک الگوریتم مخلوط گوسی[۲] به دست می‌آید. سپس محدوده‌ای از تصویر را به منظور اندازه‌گیری سرعت انتخاب می‌نماییم. به منظور افزایش دقت و رفع اعوجاج غیر خطی تصویر، بر روی این ناحیه تبدیلات افکنشی صورت می‌گیرد. سپس به کمک انجام

[۱] ***correlation***

[۲] ***mixture gaussian***

عملیات همبستگی بین ناحیه‌ی انتخاب شده‌ی متناظر در مدل پس‌زمینه و تصویر اصلی، ورود خودرو به این ناحیه را تشخیص داده و در نهایت بر اساس زمان حضور خودرو در این ناحیه و همچنین میزان مسافت پیموده شده در این ناحیه (طول ناحیه) سرعت خودرو تعیین می‌شود. در روش دوم به منظور بی‌درنگ بودن، از الگوریتم‌های پیچیده‌ی ردیابی صرف‌نظر کرده‌ایم. این روش، شامل ۴ بخش کلی است. در ابتدا چهار گوشه از تصویر دارای پرسپکتیو، به صورت دستی انتخاب شده و با اعمال تبدیل افکنشی[۱] بر روی ناحیه‌ی مذکور، تصویر یکسو شده[۲]ای از این ناحیه به دست می‌آید. چهار نقطه‌ی دیگر نیز به منظور تعیین محدوده‌ی مورد نظر برای تشخیص سرعت به طور دستی بر روی تصویر یکسو شده در نظر گرفته می‌شود. سپس به کمک روش تفاضل قاب‌ها، پیش‌زمینه[۳]ی تصویر (خودرو-های در حال حرکت) تعیین می‌شود. بعد از آن با آستانه‌گذاری[۴] مناسب، تصویر اولیه‌ای از پیش‌زمینه به دست می‌آید. با عملیات ریخت‌شناسی، نویزها و اشیای غیر مرتبط در تصویر را حذف نموده و همچنین از تعداد حفره‌های موجود در اشیای پیش زمینه تا حدودی کم می‌نماییم. در ادامه بیرونی-ترین مرز مربوط به اشیای پیش‌زمینه را به دست آورده و آن‌ها را به طور کامل پر می‌کنیم. بدین ترتیب پیش‌زمینه‌ای شامل چندضلعی‌های توپر که هر یک نمایان‌گر یک خودرو متحرک است، به دست می‌آید. در مرحله‌ی بعدی، مرکز ثقل هر یک از چندضلعی‌ها را در قاب‌های متوالی ویدئو به دست آورده، مسیر حرکت خودرو را رسم نموده و مسافت را می‌یابیم. فاصله‌ی زمانی سپری شده را با شمارش تعداد قاب‌های ویدئو به دست می‌آوریم. در مرحله‌ی آخر، سرعت خودروها را باید تعیین کنیم. برای این کار مسافت پیموده شده به واحد کیلومتر و زمان سپری شده نیز به واحد ساعت تبدیل می‌شود. با تقسیم مسافت پیموده شده بر حسب کیلومتر بر زمان سپری شده بر حسب ساعت، سرعت خودرو بر حسب واحد کیلومتر بر ساعت، تخمین زده می‌شود.

در شکل ۱-۳، نمودار کلی مربوط به مراحل کار، نمایش داده شده است:

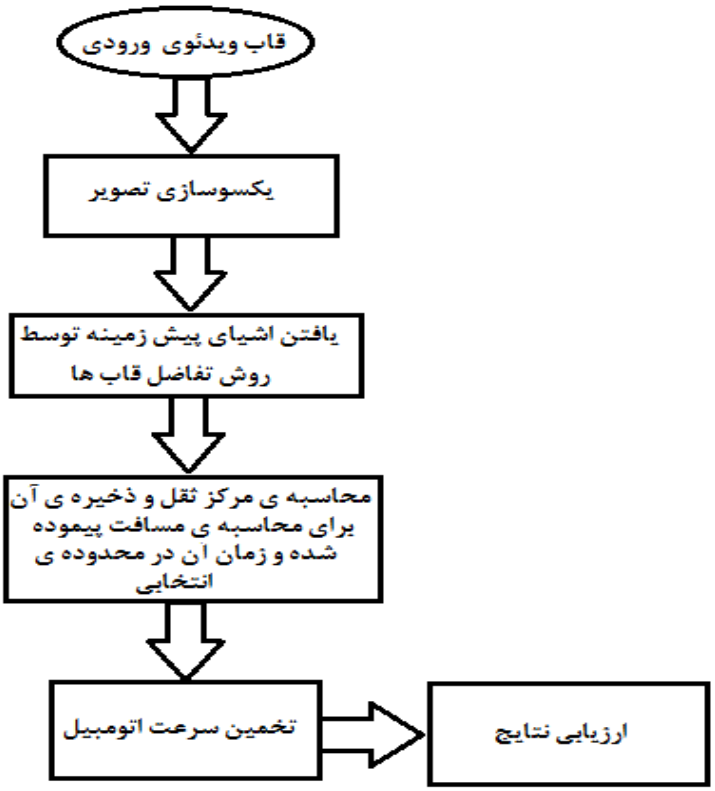
---

[۱] ***Projection Transform***

[۲] ***Rectified***

[۳] ***foreground***

[۴] ***Thresholding***



شکل ۱-۳ نمودار کلی مراحل کار در روش پیشنهادی

۱–۵ مروری بر فصول پایان نامه

این پایان‌نامه دارای ۵ فصل است. در این فصل، پس از بیان مقدمه به تعریف مسئله، مشکلات موجود و ضرورت انجام تحقیق پرداختیم. سپس راه حل‌های موجود را معرفی کردیم. بعد از آن مختصری از روش پیشنهادی خود و بلوک دیاگرام کلی روش پیشنهادی بیان گردید. در فصل دوم مروری بر کارهای انجام شده‌ی قبلی صورت می‌گیرد و با جزئیات بیش‌تری نتایج حاصل شده‌ی هر یک آورده می‌شود. در فصل سوم شرح کاملی از روش پیشنهادی خود در این پایان‌نامه به همراه موضوعات و تئوری‌های لازم می‌آوریم. در فصل چهارم نتایج مربوطه به روش پیشنهادی خود را آورده و آن را با کارهای قبلی مقایسه می‌نماییم. سرانجام در فصل پنجم نتیجه‌گیری و پیشنهاد راه‌کارهایی برای آینده، ارائه می‌گردد.

فصل دوم: مروری بر کارهای انجام شده‌ی قبلی

۲–۱ مروری بر کارهای انجام شده‌ی قبلی

همان طور که اشاره گردید، در رابطه با موضوع تخمین سرعت به کمک پردازش تصویر، کارهای زیادی از گذشته تا به امروز صورت گرفته است. در ادامه به معرفی کامل برخی از روش‌های تخمین سرعت وسائل نقلیه به کمک پردازش تصویر که توسط سایر افراد صورت گرفته است، می پردازیم.

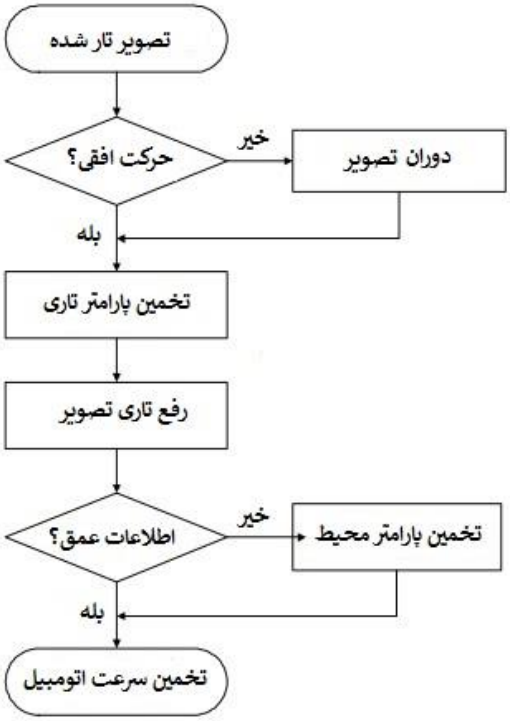
در روش ارائه شده در [۳]، به کمک یک عکس گرفته شده از حرکت خودرو، به منظور تخمین سرعت خودرو استفاده می‌شود. به دلیل حرکت نسبی بین دوربین و شی در حال حرکت طی زمان گرفتن عکس، تاری در قسمت متحرک عکس پدید می‌آید. از روی این تاری ایجاد شده در عکس می توان برای تخمین سرعت شی متحرک استفاده نمود. برای هر بازه‌ی زمانی ثابت، جابه‌جایی خودرو در تصویر با میزان تاری ناشی از پردازش تصویر، متناسب است. لذا پارامتر تاری حرکت (طول حرکت و جهت) و موقعیت نسبی بین دوربین و شی تعیین شده و سرعت خودرو بنا به هندسه‌ی تصویربرداری تعیین می‌شود.

در ابتدا پارامترهای تاری و ناحیه‌ی تقریبی به دست می‌آید. سپس تاری عکس رفع شده و سایر پارامترها را از آن به دست می‌آوریم. سرانجام سرعت با توجه به هندسه‌ی تصویرنگاری، موقعیت دوربین و پارامترهای تاری محاسبه می‌شود.

سرعت تخمین زده شده در این مقاله برای ترافیک‌های محلی و بزرگراه‌ها، با سرعت واقعی حدود ۵ درصد اختلاف داشت. در این روش زاویه‌ی دوربین باید به فرمی باشد که از خودرو تصویری افقی بگیرد.

بسته به پردازش تصویر، تخریب تصویر توسط تاری حرکت می‌تواند به صورت تغییرناپذیر با مکان و تغییرپذیر با مکان، طبقه‌بندی شود. تخریب تغییر ناپذیر با مکان مربوط به موردی است که مدل تخریب عکس در تصویر به مکان بستگی نداشته باشد که این نوع از تاری ناشی از حرکت دوربین طی فرآیند تصویربرداری است. بازسازی تاری حرکت تغییرناپذیر با مکان یک مسئله‌ی کلاسیک بوده و روش‌های زیادی برای آن ارائه شده است. نوع تاری تغییرپذیر با مکان در مواردی اتفاق می افتد که یک شی با سرعت بالا توسط یک دوربین ثابت تصویربرداری می شود. بازسازی تصویری که دچار تاری تغییرپذیر با مکان شده است یک مسئله‌ی سخت تری نسبت به حالت تغییرناپذیر با مکان است و توسط محققان کمی جواب داده شده است. در ادامه تعیین سرعت خودرو توسط یک تصویر تارشده بررسی می‌شود.

در شکل ۲-۱ دیاگرام بلوکی کلی این روش را مشاهده می‌نمایید:



شکل ۲-۱ دیاگرام بلوکی روش تخمین سرعت توسط تصویر تار شده در مرجع [۳]

این روش دارای میانگین خطای ۵٪ است. این روش بر روی یک کامپیوتر با پردازنده‌ی ***Pentium M 1.6 GHz*** و رم ۵۱۲ مگابایتی اجرا شده و زمان پردازش آن برابر با ۳۰۰ میلی‌ثانیه شده است.

نتایج رفع تاری در ۲-۳ آورده شده است:

شکل ۲-۳ تصویر بازسازی شده در مرجع [۳]

شکل ۲-۲تصویر اصلی (تار شده) در مرجع [۳]

در [۴]، به کمک پردازش قاب‌های ویدئویی، سرعت هم به صورت برخط[۱] و هم برون خط[۱] اندازه‌گیری می‌شود. این عملیات پردازشی در ۴ مرحله صورت می‌گیرد. مرحله‌ی اول مربوط به

[۱] ***online***

شناسایی خودرو است که از الگوریتم هیبرید بر مبنای ترکیب تکنیک کم کردن پس‌زمینه با یک الگوریتم تفاضل ۳ قابی است. مرحله‌ی دوم مربوط به ردیابی خودرو است که خود شامل سه مرحله‌ی بخش‌بندی تصویر، برچسب‌گذاری شی و استخراج مرکز خودرو است. مرحله‌ی سوم مربوط به اندازه گیری سرعت می باشد که از روی شمارش تعداد قاب‌های مصرفی برای عبور خودرو از محدوده‌ی مورد نظر به دست می آید. مرحله چهارم نیز مربوط به گرفتن عکس از خودروای است که از حد سرعت تعیین شده تجاوز می‌کند.

در [۵]، از سه تکنیک برای محاسبه‌ی سرعت بهره گرفته شده است و این سه تکنیک در انتها با هم مقایسه شده‌اند. تکنیک اول بر مبنای روش همبستگی متقابل نرمال شده[۲] می باشد. دومین تکنیک بر مبنای تفاضل قاب‌ها و به کارگیری مرکز ثقل و سومی نیز بر مبنای شار نوری[۳] است.

در جدول‌های ۲-۱ میزان خطای هر روش آورده شده است: (در این‌جا مسافت پیموده شده‌ی خودرو بین قاب مرجع و قاب جاری محاسبه می‌گردد.)

جدول ۲-۱ نتایج تعیین سرعت مربوط به روش همبستگی متقابل نرمال شده در مرجع [۵]

شماره ردیف	قاب جاری	قاب مرجع	سرعت (کیلومتر بر ساعت)	خطا
۱	۱۹	۱۴	۴۵/۹	٪۱۲
۲	۲۰	۱۵	۳۶/۵۸	٪۷
۳	۲۱	۱۶	۳۸/۷۹	٪۳
۴	۲۲	۱۷	۳۸/۲۸	٪۴
۵	۲۳	۱۸	۳۷/۶۰	٪۶
۶	۲۴	۱۹	۳۷/۴۳	٪۶
۷	۲۵	۲۰	۴۶/۴۱	٪۱۳

[۱] ***offline***

[۲] ***normalized cross correlation***

[۳] ***Optical Flow***

جدول ۲-۲ نتایج تعیین سرعت مربوط به روش تفاضل قاب‌ها و به کارگیری مرکز ثقل در مرجع [۵]

شماره ردیف	شماره قاب	قاب مرجع	سرعت (کیلومتر بر ساعت)	خطا
۱	۱۹	۱۴	۳۳/۵۸	٪۱۶
۲	۲۰	۱۵	۲۹/۸۳	٪۲۵
۳	۲۱	۱۶	۴۴/۳۹	٪۱۰
۴	۲۲	۱۷	۵۳/۶۸	٪۲۴
۵	۲۳	۱۸	۴۶/۹۳	٪۱۳
۶	۲۴	۱۹	۳۲/۶۴	٪۱۷
۷	۲۵	۲۰	۳۶/۷۴	٪۸

جدول ۲-۳ نتایج تعیین سرعت مربوط به روش شارنوری در مرجع [۵]

شماره ردیف	شماره قاب	قاب مرجع	سرعت (کیلومتر بر ساعت)	خطا
۱	۱۹	۱۴	۴۳/۵۰	٪۱۶
۲	۲۰	۱۵	۴۰/۳۳	٪۲۵
۳	۲۱	۱۶	۴۰/۳۰	٪۱۰
۴	۲۲	۱۷	۴۲/۶۲	٪۲۴
۵	۲۳	۱۸	۴۱/۰۹	٪۱۳
۶	۲۴	۱۹	۴۱/۰۷	٪۱۷
۷	۲۵	۲۰	۳۷/۱۳	٪۸

درصد خطا نشان می‌دهد که روش همبستگی متقابل نرمال شده و جریان نوری نتایج بهتری را ایجاد می‌کند. این روش‌ها در شرایط روز خوب می‌باشند اما در شب ممکن است نتایج متفاوت باشد. تغییر در شکل شی نیز ممکن است بر روی نتایج اثر بگذارد.

در [۶] روش تعیین سرعت بر مبنای سه گام اصلی است. این سه گام عبارتند از : کم کردن پس زمینه، استخراج ویژگی و ردیابی خودرو. سرعت با شمارش قاب‌های مصرفی برای پیمودن مسافت توسط خودرو و نرخ قاب‌ها و میزان جابه‌جایی خودرو تعیین می‌گردد.

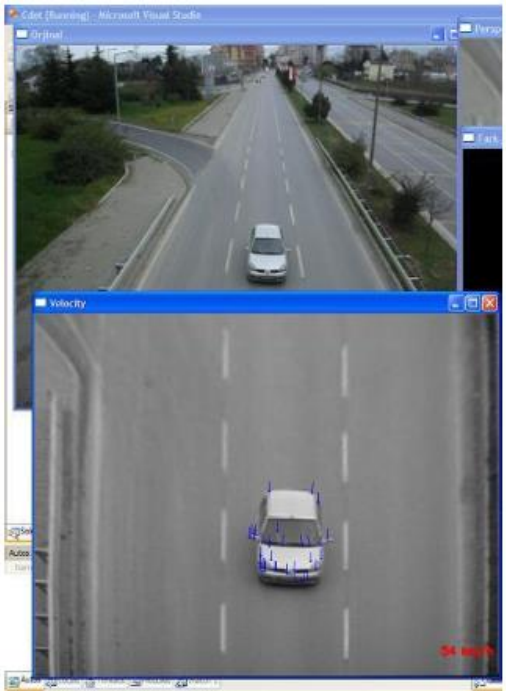
این روش برای تخمین سرعت خودرویی است که به سمت دوربین می‌آید. سرعت توسط ردیابی حرکت خودرو طی توالی‌های تصویر به دست می‌آید. در ابتدا ویدئو به قاب‌ها تبدیل می‌شود. برای تشخیص خودرو در حال حرکت تفاضل پس زمینه صورت می‌گیرد. با میانگین‌گیری روی تمام قاب‌ها، پس زمینه بدون شی متحرک استخراج می شود. پس زمینه ی به دست آمده برای عملیات آستانه یابی و ریخت شناسی به کار می رود. روش اجزای متصل به منظور تشخیص شی و مرکز ثقل آن شی به کار می رود. مرکز ثقل طی چندین قاب ردیابی می شود. سرعت توسط مسافت پیموده شده توسط خودرو و نرخ قاب ویدئو محاسبه می شود.

در [۷]، مطالعه‌ی دقیقی برای توسعه‌ی یک سیستم بی‌درنگ به منظور نظارت بر جریان ترافیکی و یافتن سرعت وسایل نقلیه توسط دوربین‌های ویدئویی صورت گرفته است. در این‌جا فرض شده است که جاده‌ی مورد مطالعه مسطح و مستقیم است و همچنین دوربین در پایین یک پل قرار گرفته است و طول یک خط در تصویر نیز مشخص است. به منظور تخمین سرعت یک خودرو در حال حرکت توسط یک دوربین فیلم برداری، یکسوسازی تصویرهای ویدئویی را به منظور حذف اثرات پرسپکتیو انجام می‌دهیم و یک ناحیه ی مطلوب[۱] را برای ردیابی خودروها تعیین می‌کنیم. بردار سرعت یک تعداد کافی از نقاط مرجع بر روی تصویر خودرو در هر قاب ویدئویی، مشخص می‌شود. برای این منظور تعداد کافی نقطه از خودرو انتخاب شده و این نقاط باید به طور دقیقی در حداقل دو قاب ویدئویی متوالی دنبال شوند. در گام بعدی به کمک بردار مسافت نقاط ردیابی شده و همچنین مدت زمان گذشته، بردار سرعت آن نقاط محاسبه می شود. بردار سرعت محاسبه شده در سیستم مختصات تصویر ویدئویی تعریف می‌شود و بردار مسافت توسط واحد پیکسل اندازه‌گیری می‌شود. سپس دامنه‌ی بردارهای محاسبه شده در فضای تصویر به فضای شی منتقل می‌شود تا مقدار واقعی این دامنه ها پیدا شود.

[۱] ***ROI***

نتایج حاصل از یکسوسازی و ردیابی نقاط ویژگی و بردارهای ویژگی را به ترتیب در شکل‌های ۲-۴ و ۲-۵ مشاهده می‌نمایید:

شکل ۲-۴ نتایج حاصل از یکسوسازی تصویر در مرجع [۷]

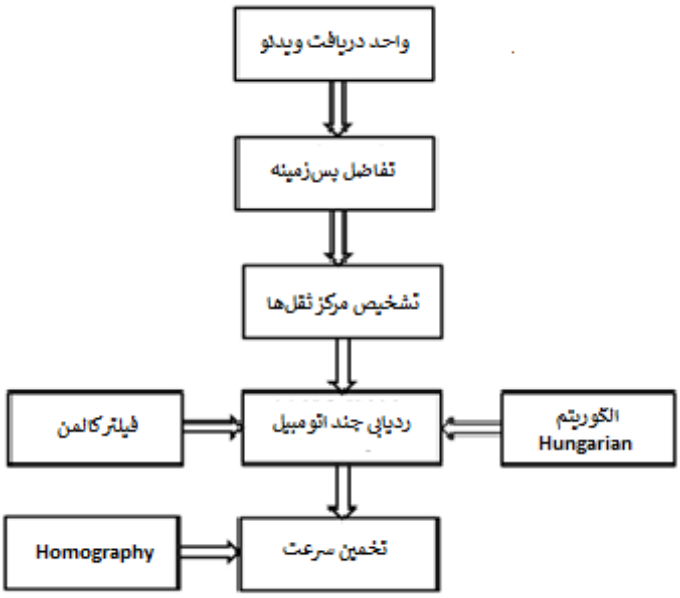


شکل ۲-۵ ردیابی نقاط ویژگی پایدار انتخاب شده توسط الگوریتم شار نوری و تخمین سرعت به کمک مسافت پیموده شده و نرخ قاب‌ها در مرجع [۷]

در [۷]، از یک دوربین با نرخ ۳۰ قاب بر ثانیه و تفکیک پذیری ***640*** در ***480*** استفاده شده است. تصاویر ثبت شده به صورت خاکستری هستند. به منظور بی درنگ شدن برنامه‌ای به زبان ***C++*** توسعه داده شده است. زمان پردازش هر قاب بر روی لپ تاپی با پردازنده‌ی ***Intel Core i7 2.6 GHz*** و رم ۸ گیگابایتی، حدود ۳۰ میلی ثانیه گزارش شده است. خطای سرعت محاسبه شده تقریبا ***±2 km/h*** می‌باشد.

در [۸]، سرعت وسائل نقلیه از طریق دوربین‌های ترافیکی کالیبره نشده به صورت مستقل از نما و به صورت بی‌درنگ، تخمین زده می‌شود. در این مقاله یک راه حل جامع برای پیاده‌سازی یک ماژول پردازشی بر روی دوربین‌های ترافیکی که قادر به ردیابی و تخمین سرعت هر وسیله‌ی نقلیه‌ای است، ارائه شده است. یک راهکار ترکیبی برای ردیابی چند وسیله‌ی نقلیه به کار گرفته شده است که از فیلتر کالمن و الگوریتم ***Hungarian*** برای رفع انسدادها استفاده می‌کند. در ادامه یک روش تخمین سرعت توصیف شده است که به اندازه‌ی کافی برای کار با دوربین با هر زاویه‌ای بدون نیاز به کالیبراسیون و برای حداقل ارتفاع ۷ متر، قدرتمند می باشد. سیستم بر روی دنباله‌های ویدئویی تولید شده بر روی کامپیوتر همانند آنچه که در دنیای واقعی رخ می‌دهد تست شده است و تخمین سرعت با ماکزیمم خطای کم تر از ۳ کیلومتر بر ساعت به دست آمده است.

در این مقاله، یک ماژول هوشمند به منظور ردیابی چندین خودرو و تخمین سرعت آن‌ها به صورت بی‌درنگ توسعه یافته است که از دوربین‌های ترافیکی به منظور تولید و انتقال اطلاعات به سرور استفاده می‌کند. ترکیب فیلتر کالمن و الگوریتم ***Hungarian*** به منظور بهبود عملکرد ردیابی استفاده می‌شود. قدرت تخمین سرعت در تغییرناپذیریش نسبت به وضع دوربین و وجود کالیبراسیون های بسیارکم، نهفته است. تنها ۴ نقطه روی یک خط جاده باید طی راه اندازی اولیه‌ی ماژول انتخاب شود. این ۴ نقطه یک ناحیه‌ی مطلوب را فراهم می‌کند که ماژول درون آن پردازش انجام می‌دهد و باعث کاهش پیچیدگی محاسبات شده و همچنین باعث می‌شود که تنها بر روی خودروهای در حال حرکت تمرکز شود. این موضوع به چندین لاین توسعه یافته که می‌توانند به طور همزمان ردیابی شوند. در شکل ۲-۶ بلوک دیاگرام مربوط به ماژول پردازشی را مشاهده می‌نمایید:



شکل ۲-۶ بلوک دیاگرام مربوط به روش پیشنهادی مرجع [۸]

سیستم پیشنهادی که در بالا مشاهده شد، بر روی برد ***BeagleBoard- xM*** با نرخ ۱۰ قاب بر ثانیه و بر روی ***Lenovo T440 ThinkPad*** با نرخ ۳۰ قاب بر ثانیه، پیاده‌سازی شده است. مرحله‌ی اول شامل تفاضل پس‌زمینه است که در آن اشیای پیش‌زمینه مثل خودروها به صورت تصاویر باینری استخراج می‌شود. خروجی این مرحله به بلوک تشخیص مرکز ثقل برای به دست آوردن مرکز ثقل خودروها می‌رود و سپس این مرکز ثقل‌ها باید ردیابی شوند. این ماژول فیلتر کالمن را به همراه الگوریتم ***Hungarian*** به کار می گیرد. مرحله‌ی آخر، ماژول تخمین سرعت است که از یک مبدل بر مبنای ***homography*** که مختصات های ردیابی خودرو را به فضای دیگری تبدیل می‌کند، کمک می-گیرد.

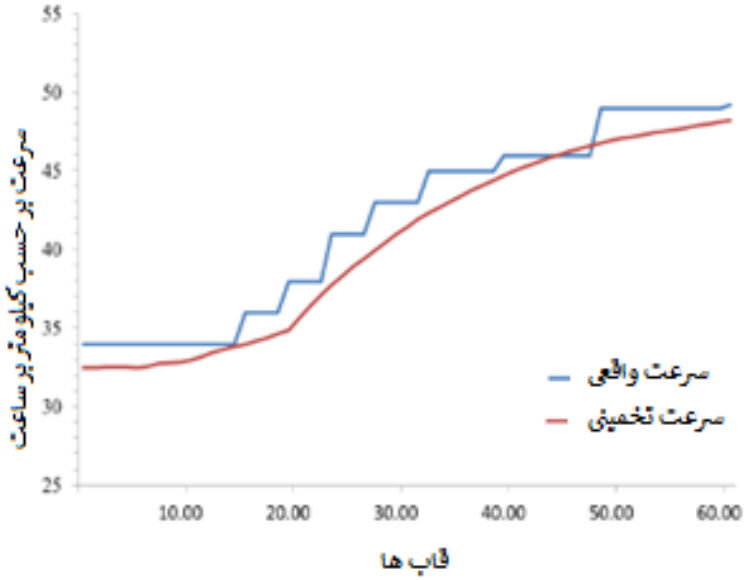
تصاویر ویدئویی با نرخ قاب ۳۰ تایی توسط دوربین ***Sony DCR-SX45E*** و با تفکیک پذیری ***768*** در ***576*** گرفته شده است. نتایج مربوط به این روش در جدول ۲-۴ آورده شده است:

جدول ۲-۴ نتایج مربوط به تعیین سرعت در مرجع [۸]

شماره ردیف	سرعت واقعی (کیلومتر بر ساعت)	سرعت تخمینی (کیلومتر بر ساعت)	خطا (کیلومتر بر ساعت)	درصد خطا
۱	۲۰	۲۰/۵۲	۰/۵۲	۲/۶۰
۲	۲۵	۲۴/۳۴	-۰/۶۶	-۲/۶۴
۳	۳۰	۲۸/۵۴	-۱/۴۶	-۴/۸۶
۴	۳۵	۳۳/۶۱	-۱/۳۹	-۳/۹۷
۵	۴۰	۳۹/۰۵	-۰/۹۵	-۲/۳۸
۶	۴۵	۴۶/۳۶	۱/۳۶	۳/۰۲
۷	۵۰	۵۱/۹۴	۱/۹۴	۳/۸۸
۸	۵۵	۵۳/۴۹	-۱/۵۱	-۲/۷۵
۹	۶۰	۵۸/۲۷	-۱/۷۳	-۲/۸۸

جدول ۲-۵ زمان پردازش مربوط به یک قاب در روش مرجع [۸]

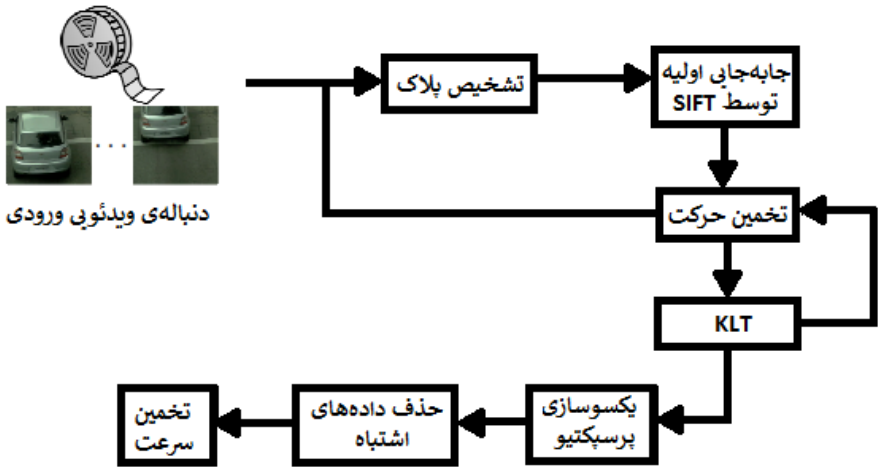
ماژول	زمان پردازش (میلی ثانیه)
تفاضل پس‌زمینه	۲۱/۲۴۶
تشخیص مرکز ثقل‌ها	۳/۰۱۲
ردیابی چند خودرو	۰/۰۲۳
تخمین سرعت	۰/۰۰۳
کل	۲۴/۲۸۴



نمودار ۲-۱ مقایسه‌ی سرعت واقعی و تخمین زده شده در مرجع [۸]

در مرجع شماره ی [۹]، یک سیستم جدید برای تخمین سرعت خودرو از ویدئوی گرفته شده از جاده های شهری توصیف شده است. سیستم مورد نظر از تشخیص متن برای مکان یابی پلاک خودرو های عبوری استفاده کرده و سپس از پلاک به منظور انتخاب ویژگی‌های پایدار برای ردیابی استفاده می‌کند. سپس ویژگی‌های مربوط به ردیابی برای رفع اعوجاج پرسپکتیوی تصویر فیلتر می‌شود. سرعت خودرو با مقایسه‌ی مسیر ویژگی‌های ردیابی با معیارهای شناخته شده‌ی دنیای واقعی تخمین زده می‌شود.

در این مقاله یک سیستم جدید برای تخمین سرعت خودرو از ویدئوهای گرفته شده از جاده‌های شهری، توصیف می‌شود. بلوک دیاگرام سیستم را در شکل ۲-۷ مشاهده می‌نمایید:



شکل ۲-۷ بلوک دیاگرام مربوط به روش ارائه شده در مرجع [۹]

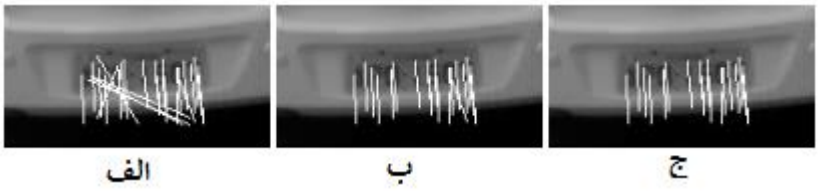
در ابتدا یک تشخیص دهنده‌ی متن به منظور تشخیص پلاک خودروهای در حال عبور استفاده می‌شود. سپس ویژگی‌های پایدار درون نواحی تشخیص، توسط یک ترکیبی از الگوریتم های ***SIFT*** و ***KLT*** ردیابی می‌شوند. بعد از فیلتر کردن ناهماهنگی‌ها، سرعت خودرو با مقایسه‌ی مسیر ویژگی‌ها با معیارهای دانسته ی دنیای واقعی تخمین زده می شود که به ما اجازه می‌دهد که اعوجاج پرسپکتیوی را یکسو کنیم و یک رابطه‌ی متر بر پیکسل به دست آوریم. قبل از این، سیستم مربوطه، سرعت خودرو را با ردیابی گوشه‌ها و ویژگی‌های ناحیه‌ی استخراج شده از پلاک تخمین می‌زد.

برای ارزیابی این سیستم ویدئوهای گرفته شده تحت شرایط عملکردی واقعی مرتبط با داده‌ی صحه‌گذاری شده توسط یک آشکارساز حلقه‌ی سلفی، استفاده گردید. این سیستم به دقت ۰/۸۷ و یک فراخوانی ۰/۹۲ برای تشخیص پلاک رسیده است. سرعت خودرو که با یک خطای متوسط ۰/۵۹ کیلومتر بر ساعت تخمین زده می‌شود، در بازه ی ۳-/۲+ کیلومتر بر ساعت قرار می‌گیرد. انحراف معیار خطا ۱/۶۳ کیلومتر بر ساعت و حداکثر مقدار خطا برای کل داده‌ها ۳/۲۴- و ۳/۹۱+ کیلومتر بر ساعت بود. تصاویر گرفته شده توسط دوربین دارای تفکیک پذیری[۱] ***768*** در ***480*** و با نرخ ۳۱/۲۵ قاب بر ثانیه بودند.

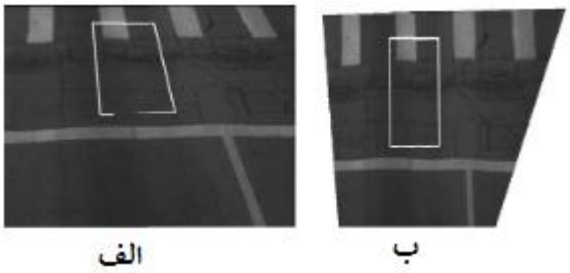
نتایج مربوط به این روش را در شکل ۲-۸ مشاهده می‌نمایید:

[۱] ***Resolution***

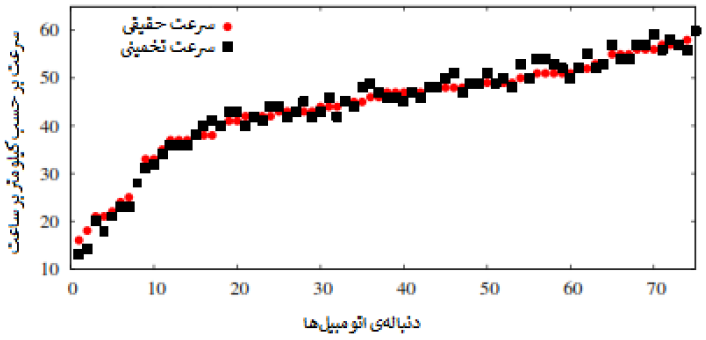
شکل ۲-۸ تشخیص پلاک از روی شناسایی متن در مرجع [۹]



شکل ۲-۹ حذف داده‌های اشتباه[۱] الف) بردارهای حرکتی ب) حذف بردارهای اشتباه (۴ بار تکرار) ج) نتیجه‌ی نهایی (۵ بار تکرار). (مرجع [۹])



شکل ۲-۱۰ الف) یکسوسازی تصویر الف) تصویر اصلی دارای اثر پرسپکتیوی ب) تصویر یکسو شده. (مرجع [۹])

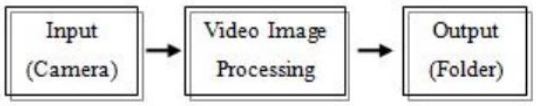


نمودار ۲-۲ مقایسه‌ی سرعت واقعی و تخمین زده شده در مرجع [۹]

[۱] ***Outlier rejection***

در مرجع شماره‌ی [۱۰]، الگوریتم تشخیص بی‌درنگ سرعت خودرو توسط تکنیک بردار حرکت توصیف شده است. سیستم مانیتورینگ توسط دوربین‌های ویدئویی مراقبتی، علاقه‌مندی زیادی را در میان انجمن‌های تحقیقاتی مخصوصا در مانیتورینگ سرعت خودرو، به دست آورده است. جدا از تشخیص سرعت خودرو این الگوریتم می‌تواند به منظور مانیتور کردن شرایط ترافیکی در طول جاده یا بزرگ‌راه استفاده شود. دوربین‌های ویدئویی مراقبتی موجود به ندرت برای اندازه‌گیری سرعت خودرو و تخمین خودرو به کار می‌روند. یک برنامه‌ی متلب مرتبط با الگوریتم مورد نظر با تصاویر و سری‌های ویدئویی بی درنگ، پیشنهاد و توسعه داده شده است. توسعه‌ی الگوریتم تشخیص سرعت خودرو بر مبنای تابع بردار-مقدار[۱] و تکنیک بردار حرکت است که سرعت خودرو در حال حرکت را تخمین می‌زند.

تکنیک استخراج بلوکی و تفاضل بلوکی، به عنوان یکی از ساده‌ترین تکنیک‌های تشخیص حرکت شناخته می‌شود. این تکنیک با کدگذاری دنباله‌ی ویدئویی برای تشخیص سرعت خودرو بر اساس تصاویر فعلی و قبلی، سازگار است. هر تصویر به بلوک‌های مربعی غیر همپوشان برای مقایسه‌ی بلوک های مربوطه در قاب کنونی و تصویر قبلی، تقسیم می‌شود. هر یک از بلوک‌ها برای تخمین سرعت خودرو از هم کم می‌شوند. بلوک دیاگرام تشخیص حرکت را در شکل ۲-۱۱ مشاهده می‌نمایید:

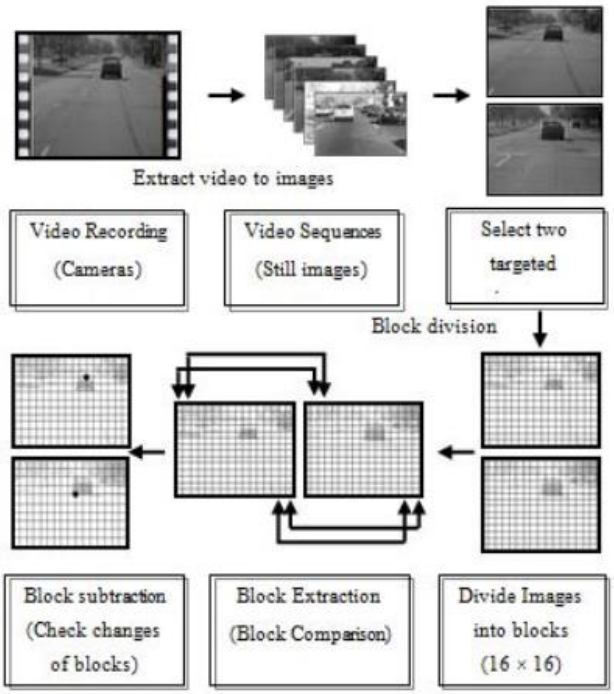


شکل ۲-۱۱بلوک دیاگرام کلی روش ارائه شده در مرجع [۱۰]

سیستم نشان داده شده در بالا شامل ۳ مرحله برای تشخیص سرعت خودرو است و خروجی آن نیز سرعت تخمینی است. دوربین ویدئویی مراقبتی به عنوان یک ورودی برای این سیستم به منظور گرفتن ویدئو و ذخیره‌ی آن در یک پوشه‌ی تعیین شده در هارد دیسک، به کار گرفته شده است. متلب به عنوان بستر اصلی برای توسعه‌ی الگوریتم تشخیص سرعت خودرو است. تابع بردار-مقدار به منظور تحلیل تغییرات در بخش بلوک بر مبنای دو تصویر متوالی استفاده می‌شود.

در این مرجع، یک الگوریتم جدید تشخیص سرعت خودرو به کمک نرم افزار متلب ارائه شده است. این روش بر مبنای دنباله‌ی ویدئویی بی‌درنگ بوده که از یک الگوریتم آفلاین (داده‌های آموزش داده شده از قبل)، برای کاهش زمان پردازش استفاده می‌کند.

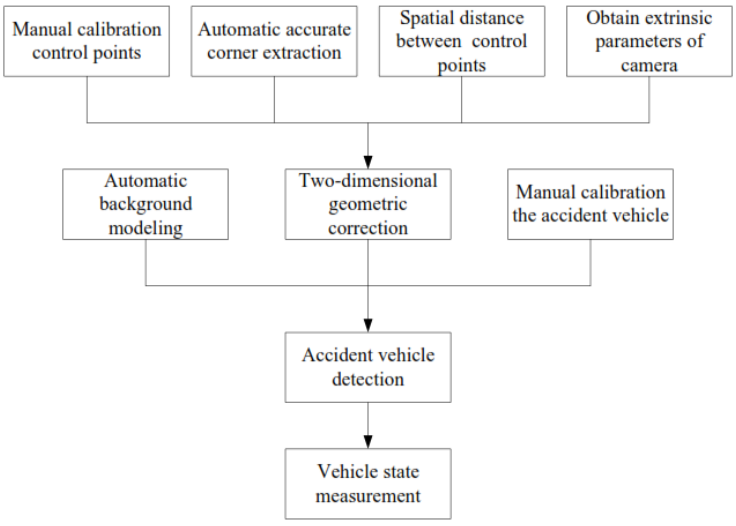
[۱] ***Vector-Valued***



شکل ۲-۱۲ مراحل کار الگوریتم در مرجع [۱۰]

در مرجع شماره‌ی [۱۱]، یک روش جدید به کمک عملیات دستی برای تحقیق و بررسی صحنه-ی تصادفات جاده‌ای ارائه شده است. در این روش، مشخصات صحنه‌ی تصادف شامل اطلاعات مربوط به لاین‌ها و خودروهای تصادفی، برای تعیین وضعیت خودروهای تصادفی به کار گرفته می‌شود. توسط عملیات دستی، وضعیت یک خودرو تصادفی، شامل سرعت و مسیر حرکت خودرو، می‌تواند قاطع تر و دقیق تر تخمین زده شود.

در شکل ۲-۱۳ دیاگرام کلی الگوریتم به کار گرفته شده را مشاهده می نمایید:



شکل ۲-۱۳ دیاگرام کلی الگوریتم به کار رفته در مرجع شماره‌ی [۱۱]

در روش پیشنهادی همان طور که ملاحظه می‌کنید، ۴ مرحله‌ی اصلی وجود دارد. در مرحله‌ی اول، مختصات چندین نقطه در حوزه‌ی تصویر و در حوزه‌ی دنیای واقعی به صورت دستی تعیین می‌شود. سپس می‌توان رابطه‌ی مسافت بین فضای تصویر و فضای واقعی را به منظور اصلاح هندسی دو بعدی تخمین زد. در مرحله‌ی دوم، مدل پس‌زمینه تخمین زده می‌شود و برای تشخیص حرکت استفاده می‌شود. از آن جایی که اشیای در حال حرکت زیادی در صحنه‌ی تصادف وجود دارند، خودروهای تصادفی به صورت دستی در مرحله‌ی سوم برچسب‌گذاری می‌شوند. در مرحله‌ی چهارم، منحنی سرعت و بردارهای حرکت خودروهای تصادفی به طور خودکار استخراج می‌شود.

در مرحله‌ی اول که مرحله‌ی اصلی کار است، اصلاح دو بعدی هندسی انجام می‌دهیم. اصلاح دو بعدی برای بردن مختصات تصویر خودرو به دنیای واقعی صورت می‌گیرد. خوشبختانه در اغلب موارد فرض بر این است که جاده مسطح بوده و خودروها در سطح مسطحی در حال حرکت هستند. در این حالت ما به یک تبدیل دو بعدی به دو بعدی نیاز داریم. فرمول مربوط به تبدیل افکنشی در زیر آمده است:

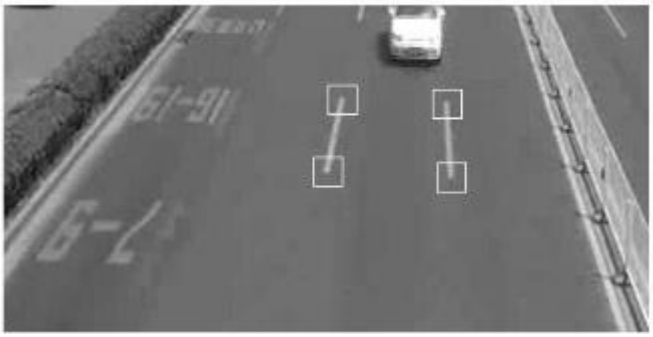
$$q = H_p p$$ (معادله ۲-۱)

که در آن H_p ماتریس نگاشت افکنش است. همچنین $P_z=q_z=1$ می‌باشد و داریم:

$$\begin{bmatrix} q_x \\ q_y \\ q_z \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$ (معادله ۲-۲)

این ماتریس شامل ۹ المان است ولی ۸ پارامتر مستقل دارد لذا با ۴ جفت نقطه‌ی کنترلی قابل حل است.

در شکل ۲-۱۴ این ۴ جفت نقطه تعیین شده است. همچنین فاصله‌ی خطوط جاده را نیز می‌دانیم.



شکل ۲-۱۴ نقاط مربوط به اصلاح دو بعدی تصویر در مرجع [۱۱]

نوبت به اصلاح نقاط کنترلی و استخراج نقاط گوشه‌ای رسیده است. در این‌جا از آشکارساز گوشه-ی هریس و استفن استفاده می‌شود. ماتریس G به صورت زیر محاسبه می‌شود.

(معادله ۲-۳)

$$G = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$$

که I_x و I_y مشتقات جزئی در راستای x و y هستند و جمع بر روی همسایه‌های مرکزی ویژگی‌ها زده می شود. λ_1 و λ_2 دو مقدار ویژه از ماتریس G هستند و هنگامی نقاط را به عنوان گوشه قبول داریم که داشته باشیم:

(معادله ۲-۴)

$$\min(\lambda_1, \lambda_2) > \lambda$$

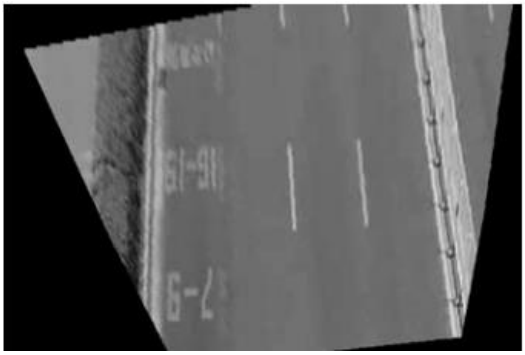
که λ یک آستانه‌ی از قبل تعریف شده است. نتیجه‌ی تشخیص لبه در شکل ۲-۱۵ نشان داده شده است:

شکل ۲-۱۵ استخراج گوشه ها در مرجع [۱۱]

بعد از تعیین این نقاط تبدیل افکنشی صورت می‌گیرد. در شکل ۲-۱۶ این تبدیل را مشاهده می‌نمایید:

شکل ۲-۱۶ تصویر بعد از تبدیل افکنشی در مرجع [۱۱]

روش تفاضل قاب‌ها بر مبنای پس‌زمینه‌ی ثابت بر این اساس است که ابتدا تصویری بدون خودرو انتخاب می‌شود و به عنوان پس‌زمینه لحاظ می‌گردد و سپس تفاوت دنباله‌های تصویر می‌تواند توسط کم کردن تصاویر شامل خودروهای در حال حرکت و پس‌زمینه‌ی ثابت به دست آید. در شکل ۲-۱۷، پس‌زمینه‌ی ثابت را مشاهده می‌نمایید:



شکل ۲-۱۷ تصویری از پس زمینه ی ثابت در مرجع [۱۱]

فرض کنید زمان پس‌زمینه t_0 باشد. مقدار خاکستری نقطه‌ی (x, y) در تصویر پس زمینه را توسط $I_0 = (x, y, t_0)$ بیان می‌کنیم. در لحظه‌ی t، مقدار خاکستری نقطه‌ی (x, y) در تصویر به $I_t = (x, y, t)$ تغییر خواهد کرد. در لحظه‌ی $t+1$، مقدار خاکستری نقطه‌ی (x, y) در تصویر به $I_{t+1} = (x, y, t+1)$ تغییر خواهد کرد. تفاضل خاکستری بین دو لحظه‌ی t_0 و t مطابق با معادله‌ی زیر است:

$$\Delta I_t(x,y) = |I_t(x,y) - I_0(x,y)| \quad \text{(معادله ۲-۵)}$$

در شکل ۲-۱۸ حاصل این تفاضل را مشاهده می‌نمایید:

شکل ۲-۱۸ تصویر حاصل از تفاضل خاکستری در مرجع [۱۱]

اطلاعات حرکت به کمک باینری کردن تصویر تفاضلی صورت می‌گیرد. معادله‌ی مربوطه در ادامه آورده شده است که مقدار آستانه‌ی T از الگوریتم آتسو محاسبه می‌شود:

$$\Delta I(x,y) = \begin{cases} 1, & \Delta I(x,y) \geq T \\ 0, & \Delta I(x,y) < T \end{cases} \quad \text{(معادله ۲-۶)}$$

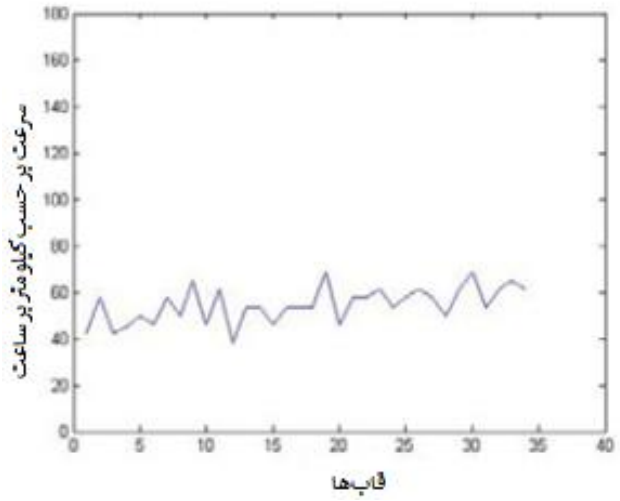
نتیجه‌ی حاصل از باینری کردن تصویر در شکل ۲-۱۹ آورده شده است و از آن جایی که چندین شی در حال حرکت در صحنه وجود دارند، لذا خودروهای تصادفی را به طور دستی برچسب‌گذاری می‌کنیم:

شکل ۲-۱۹تصویر حرکت باینری شده در مرجع [۱۱]

حال نوبت به اندازه‌گیری وضعیت خودرو است. فرض کنید $x_{max} = \{x_1, x_2, \dots, x_n\}$ برابر با مختصات x از جلوی یک خودرو و n نیز اندیس قاب باشد. f نرخ ویدئو، L_{Actual} طول واقعی خط جاده بر حسب متر و L_{image} نیز طول خط جاده بر حسب واحد پیکسل می‌باشد. همچنین عدد 3.6 به منظور تبدیل m/s به Km/h ضرب شده است. سرعت خودرو تصادفی بین قاب‌های K و $K+1$ می تواند توسط فرمول زیر محاسبه شود:

$$v_k^{k+1} = \left| \frac{(x_{k+1} - x_k) L_{Actual}}{L_{image}} \right| \times f \times 3.6 \qquad \text{(معادله ۲-۷)}$$

در این مقاله نرخ ویدئو 25 قاب بر ثانیه و ابعاد ویدئو بعد از تبدیل افکنشی $603x903$ پیکسل است. طول لاین‌ها در زمینه‌ی واقعی 6.1 متر و در زمینه‌ی تصویر 143 پیکسل بود. منحنی سرعت خودرو تصادفی مطابق با فرمول فوق محاسبه گردید و در شکل ۲-۲۰ آن را مشاهده می‌نمایید:



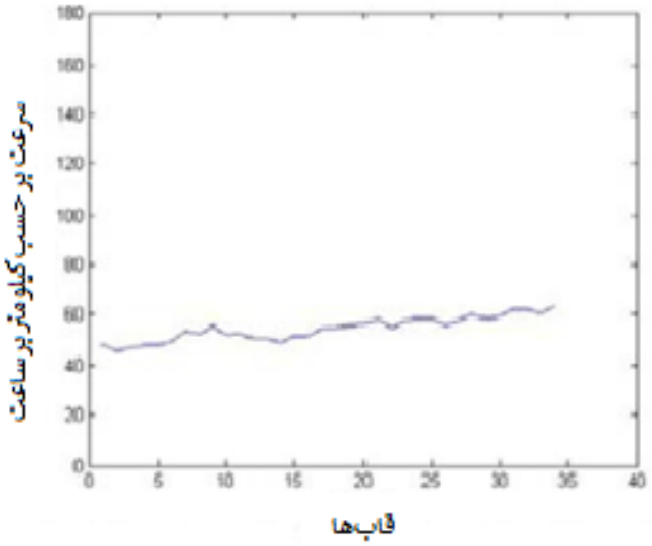
شکل ۲-۲۰ منحنی سرعت خودرو تصادفی در مرجع [۱۱]

با تحلیل منحنی بالا وضعیت حرکتی خودرو تصادفی را ثبت می‌نماییم. با این حال، منحنی سرعت که به صورت قاب به قاب اندازه‌گیری شده است، به دلیل عدم دقیق بودن دوربین، پایدار نیست. برای کاهش این اثر در تحلیل وضعیت خودرو تصادفی، منحنی سرعت خودرو تصادفی توسط یک فیلتر پایین‌گذر نرم می‌شود. مطابق با مسافت پیموده شده در بازه‌های زمانی متوالی، سرعت به صورت زیر بیان می‌شود:

$$v_k^{k+1} = \left| \frac{(x_{k+\Delta x} - x_k) L_{Actual}}{\Delta x \times L_{image}} \right| \times f \times 3.6$$

(معادله ۲-۸)

که مقدار Δx می‌تواند مطابق با موقعیت واقعی تنظیم شود. در این مورد، Δx برابر با ۵ بوده و منحنی نرم[۱] شده در شکل ۲-۲۱ نشان داده شده است:



شکل ۲-۲۱ منحنی نرم شده در مرجع [۱۱]

نتایج عملی نشان می دهد که روش پیشنهادی می‌تواند وضعیت خودرو تصادفی را به دقت ثبت نماید. نرخ ویدئو در این آزمایش ۲۵ قاب بر ثانیه و اندازه‌ی ویدئو پس از یکسوسازی ***603*** در ***903*** پیکسل بود. طول هر خط از جاده در دنیای واقعی ۶/۱ متر و معادل ۱۴۳ پیکسل گردید.

در مرجع شماره‌ی [۱۲]، یک روش جدید بر مبنای پردازش تصویر برای تشخیص خودکار و بی-درنگ سرعت خودرو توسط دوربین فیلم برداری، ارائه شده است. بر مبنای هندسه‌ی نوری[۲]، در بخش اول یک روش ساده برای نگاشت دقیق مختصات از حوزه‌ی تصویر به حوزه‌ی دنیای واقعی ارائه گردیده است. بخش دوم بر روی تشخیص خودرو در دنباله‌ی ویدئویی تمرکز می‌کند. آزمایش نشان

[۱] ***Smooth***

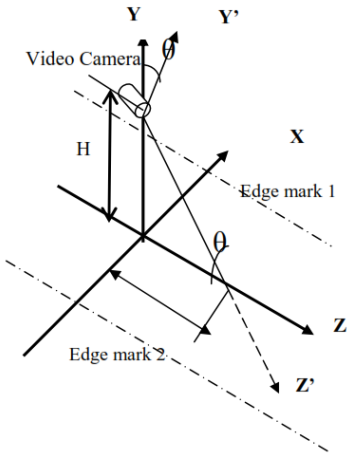
[۲] ***Optic***

می‌دهد که تنها یک دوربین ویدئویی دیجیتال و یک کامپیوتر برای مانیتور کردن همزمان سرعت خودروها در چند لاین نیاز است. خطای میانگین سرعت خودرو تشخیص داده شده زیر ۴ درصد است.

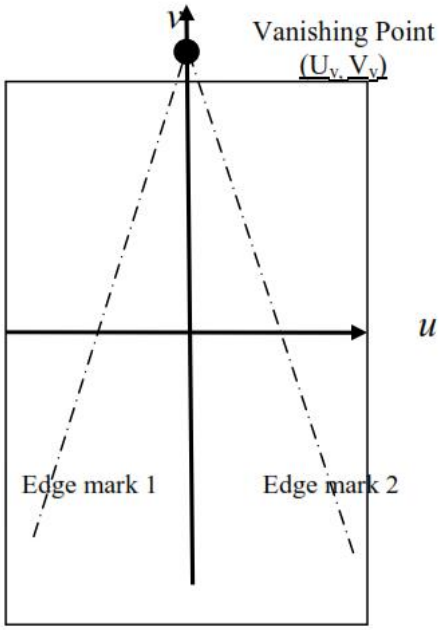
در این مقاله الگوریتم جدیدی ارائه شده است که از مزیت پردازش تصویر دیجیتال و دوربین‌های نوری به منظور تشخیص دقیق و خودکار سرعت خودرو به طور بی درنگ استفاده می‌کند. الگوریتم تنها به یک دوربین ویدئویی و یک کامپیوتر پردازشی برای کارکرد نیاز دارد و به طور همزمان می‌تواند سرعت خودرو را در چند لاین با دقت بالا و خطای کمتر از ۴ درصد به طور بی درنگ تشخیص دهد. الگوریتم تنها نیازمند این است که دوربین به طور مستقیم بالای بخش مورد نظر جاده (در حداقل ۵ متر بالای جاده برای اطمینان از دقت مطلوب) با یک زاویه‌ی خاص نسبت به بزرگ‌راه قرار گیرد. کالیبراسیون بسیار ساده است و مستقیما در قاب‌های ویدئویی بر اساس موقعیت یک نقطه‌ی ***vanishing***، یک خودرو با طول و عرض دانسته شده و اطلاعاتش (موقعیت لبه‌ی بالایی و پایینی) در یک تصویر نمونه، صورت می گیرد. کالیبراسیون به هیچ اطلاعاتی در مورد دوربین از جمله فاصله‌ی کانونی نیاز ندارد. تنها مشخصه‌ای از دوربین که مورد نیاز است، نرخ ویدئویی یا همان تعداد قاب بر ثانیه است.

نوبت به نگاشت مختصات از حوزه‌ی تصویر به حوزه‌ی دنیای واقعی می‌رسد. نگاشت ما یک نگاشت دو بعدی به دو بعدی است که دنیای تصویر را به دنیای واقعی نگاشت می‌دهد.

در ابتدا به چگونگی تنظیم کردن دوربین ویدئویی وقتی تصاویر ویدئویی از جاده گرفته می‌شود نگاه می‌اندازیم. همان طور که در شکل ۲-۲۲ نشان داده شده است، دوربین در ارتفاع H از سطح جاده و با یک زاویه ی انحراف محور لنز (θ) از راستای مستقیم جاده قرار گرفته است. می‌توان به راحتی تابع نگاشت بین حوزه‌ی تصویر و حوزه‌ی دنیای واقعی را بر اساس هندسه‌ی لنز به دست آورد.



شکل ۲-۲۲ مختصات دنیای واقعی و تنظیم دوربین ویدئویی در مرجع [۱۲]



شکل ۲-۲۳ مختصات حوزه ی تصویر در مرجع [۱۲]

در هندسه‌ی لنز، یک دوربین می‌تواند به صورت یک لنز محدب با فاصله‌ی کانونی f همان طور که در شکل ۲-۲۳ مشاهده می‌شود، ساده‌سازی شود. در این جا فرض می‌شود که فاصله‌ی بین یک شی و صفحه‌ی لنز برابر با z و فاصله‌ی بین یک شی و محور اپتیکی لنز برابر با y است. همچنین تصویر شی دارای یک ارتفاع v و مسافتش تا صفحه‌ی لنز برابر با w است.

$$\frac{1}{f}=\frac{1}{z}+\frac{1}{w}$$ (معادله ۲-۹)

$$\frac{v}{y}=\frac{w}{z}, \frac{v}{w}=\frac{y}{z}$$ (معادله ۲-۱۰)

از آن جایی که فاصله‌ی کانونی دوربین ویدئویی معمولا خیلی کوچک است (۵ سانتی‌متر یا کم‌تر) و فاصله ی بین یک خودرو و لنز بیش‌تر از ۵ متر است ($z>>f$ و $w \equiv zf/(z-f) \approx f$)، لذا صفحه‌ی تصویربرداری یعنی جایی که آرایه‌ی **CCD** قرار می‌گیرد، خیلی نزدیک به نقطه‌ی کانونی است. در واقع بعد از این که دوربین تنظیم شد، w ثابت است.

در این‌جا سه سیستم مختصات داریم. اولی سیستم مختصات دنیای واقعی (x, y, z) است که x، y و z به ترتیب راستای عرضی در سطح جاده، راستای عمود بر سطح جاده و راستای طولی یا رو به جلو در سطح جاده را ارائه می‌دهند. دومی، سیستم مختصات دوربین (x', y', z') است که x' راستای افقی در صفحه‌ی لنز، y' راستای عمود بر راستای x' در صفحه‌ی لنز و z' نیز راستای محور اپتیکی را ارائه می‌دهند. سومین سیستم مختصات، سیستم حوزه‌ی تصویر(u, v) است. u برای راستای افقی و v راستای عمودی (واحد پیکسل) است. همان طور که در معادله‌ی بالا نشان داده شده است، یک رابطه-

ی نگاشت مستقیمی بین سیستم مختصات دوربین و سیستم مختصات تصویر وجود دارد. اما آنچه که نیاز است، رابطه‌ی بین سیستم مختصات دنیای واقعی و سیستم مختصات تصویر است. در ابتدا رابطه-ی بین سیستم مختصات دنیای واقعی و سیستم مختصات دوربین را تحلیل می‌کنیم. برای ساده‌سازی تحلیل‌ها، فرض می‌شود که محور x موازی با محور x' است. داریم:

$$x' = x \quad \text{(معادله ۲-۱۱)}$$

$$y' = z\cos\theta + H\sin\theta \quad \text{(معادله ۲-۱۲)}$$

$$z' = z\sin\theta - H\cos\theta \quad \text{(معادله ۲-۱۳)}$$

که H ارتفاع دوربین از سطح جاده و θ نیز زاویه‌ی بین محور z و z' است. با ترکیب این معادله با معادله‌ی ۲-۱۰، داریم:

$$\frac{v}{w} = \frac{y'}{z'} = \frac{z\sin\theta - H\cos\theta}{z\cos\theta + H\sin\theta} \quad \text{(معادله ۲-۱۴)}$$

$$\frac{u}{w} = \frac{x'}{z'} = \frac{x}{z\cos\theta + H\sin\theta} \quad \text{(معادله ۲-۱۵)}$$

که (u, v) موقعیت در مختصات تصویر و (x, y, z) سیستم مختصات دنیای واقعی و w نیز فاصله ی بین صفحه‌ی CCD و صفحه‌ی لنز است که بعد از این که دوربین تنظیم شد، ثابت می‌ماند.

وقتی $z \to \infty$ یا (u, v) به یک نقطه‌ی تکی همگرا شود به آن نقطه‌ی $Vanishing$ می‌گویند که مربوط به نقطه‌ی فاصله‌ی بی نهایت در سطح جاده است. در این جا (Uv, Vv) برای ارائه‌ی نقطه‌ی $Vanishing$ به کار می‌رود.

$$U_v = \lim_{z\to\infty} \frac{wx}{z\cos\theta + h\sin\theta} = 0 \quad \text{(معادله ۲-۱۶)}$$

$$V_v = \lim_{z\to\infty} \frac{w(z\sin\theta - h\cos\theta)}{z\cos\theta + h\sin\theta} = w\tan\theta \quad \text{(معادله ۲-۱۷)}$$

معادله‌ی فوق نشان می‌دهد که بعد از این که دوربین تنظیم گردید، نقطه‌ی $Vanishing$ ثابت می‌شود. مختصات عمودی‌اش با $\tan\theta$ متناسب است. می‌توان ویژگی نقطه‌ی $Vanishing$ را به منظور کالیبره کردن نگاشت بین سیستم مختصات تصویر و سیستم مختصات دنیای واقعی استفاده کرد.

در یک تصویر، یافتن نقطه‌ی ***Vanishing*** توسط گرفتن نقطه‌ی سطح مقطع[1] از دو لبه‌ی لاین‌ها همان طور که در شکل فوق مشاهده می‌نمایید، خیلی راحت است. از این اطلاعات می‌توان به منظور کالیبره کردن تابع نگاشت بین مختصات‌های تصویر و مختصات‌های دنیای واقعی استفاده نمود.

فرض می‌شود که یک نقطه در سطح جاده دارای مختصات دنیای واقعی (***x, 0, z***) بوده و مختصات (***u, v***) مربوط به حوزه‌ی تصویر باشد بر اساس دو معادله‌ی قبلی داریم:

$$\frac{vtan\theta}{V_v} = \frac{y'}{z'} = \frac{zsin\theta - Hcos\theta}{zcos\theta + Hsin\theta}$$ (معادله ۲-۱۸)

و این معادله می تواند به معادله ی زیر تغییر نماید:

$$z = \frac{2HV_v}{sin2\theta\,(V_v - v)} - Htan\theta$$ (معادله ۲-۱۹)

که ***H*** ارتفاع محل دوربین ویدئویی از سطح جاده و θ نیز زاویه‌ی بین محور اپتیکی (لنز) دوربین و سطح جاده است. به این دلیل که هر دو پارامتر ***H*** و θ بعد از این که دوربین تنظیم شد، ثابت می‌مانند، معادله‌ی فوق را به صورت زیر می‌توان ساده نویسی کرد:

$$z = \frac{C_1}{V_v - v} + C_2$$ (معادله ۲-۲۰)

که در آن داریم:

$$C_2 = -Htan\theta\,,\; C_1 = \frac{2HV_v}{sin2\theta}$$ (معادله ۲-۲۱)

که C_1 و C_2 ثابت‌های مربوط به تنظیم دوربین هستند و از آن جایی که C_2 تنها یک ثابت تبدیل در راستای مستقیم جاده است، می توان آن را توسط حرکت دادن مبدا مختصات دنیای واقعی به اندازه‌ی C_2، حذف نمود. معادله‌ی بالا را می‌توان به صورت زیر ساده‌تر کرد:

$$z = \frac{C_1}{V_v - v}$$ (معادله ۲-۲۲)

به طور مشابه می توان ***x*** را از معادلات قبل به صورت زیر به دست آورد:

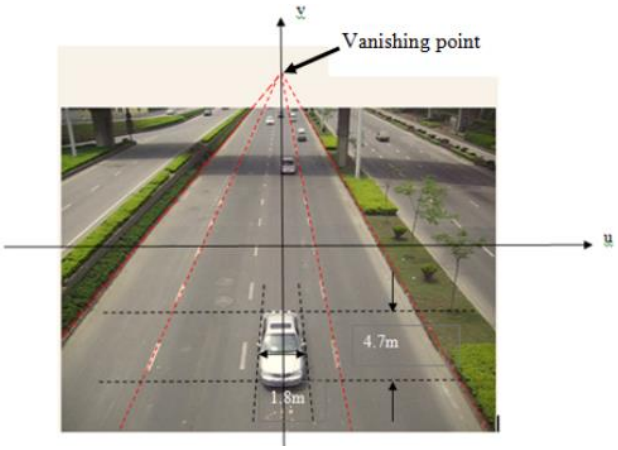
$$x = \frac{(zcos\theta + hsin\theta)utan\theta}{V_v} = hcsc\theta\,\frac{u}{V_v - v} = C_3\frac{u}{V_v - v}$$ (معادله ۲-۲۳)

و تابع نگاشت بین مختصات دنیای واقعی و مختصات تصویر می‌تواند به صورت زیر نوشته شود:

[1] ***cross-section***

$$\begin{cases} x = \dfrac{C_3 u}{V_v - v} \\ z = \dfrac{C_1}{V_v - v} \end{cases}$$ (معادله ۲-۲۴)

که ***(x, z)*** مختصات دنیای واقعی از یک نقطه درون صفحه‌ی سطح جاده است. ***z*** راستای مستقیم جاده و ***x*** راستای عرضی جاده است. ***(u, v)*** نیز مختصات حوزه‌ی تصویر همان نقطه است که ***u*** امتداد بعد افقی و ***v*** امتداد بعد عمودی است. V_v مختصات عمودی نقطه‌ی ***Vanishing*** سطح جاده است. همان طور که در شکل ۲-۲۴ نشان داده شده است، می‌توان نقطه‌ی ***Vanishing*** و یک خودرو با اندازه‌ی مشخص (طول و عرض) را برای کالیبره کردن V_v، C_1 و C_2 به کار برد.



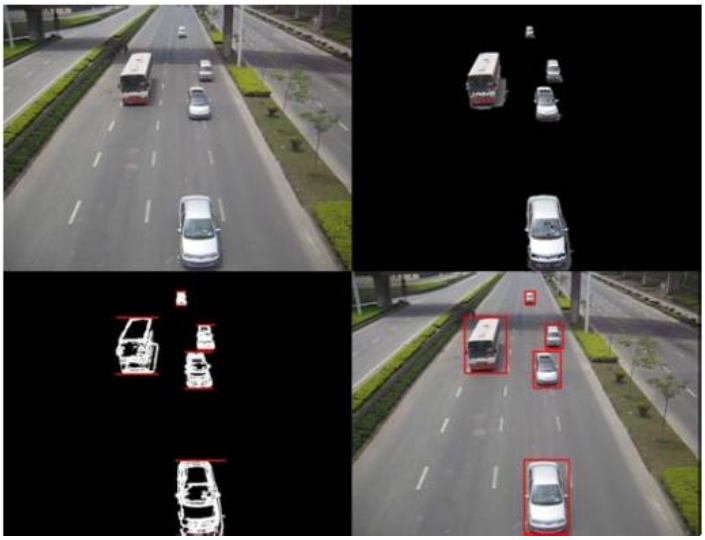
شکل ۲-۲۴ نقطه‌ی ***Vanishing*** و کالیبراسیون در مرجع [۱۲]

کالیبراسیون به صورت آنچه که در ادامه آورده شده است، انجام می‌شود. همان طور که در شکل بالا نشان داده شده است، در ابتدا باید موقعیت نقطه‌ی ***Vanishing*** یعنی (U_v, V_v) سطح جاده را از تقاطع لاین‌ها به دست آوریم. سپس از یک خودرو با طول (L) و عرض (W) شناخته شده در تصویر به منظور انجام کالیبراسیون استفاده می‌کنیم. در ابتدا کوچک‌ترین مستطیل در بر گیرنده‌ی خودرو را رسم می‌کنیم. مختصات‌های گوشه‌ی بالا سمت چپ (u_1, v_1) و گوشه‌ی پایین سمت راست (u_2, v_2) از مستطیل را به دست آورده و داریم:

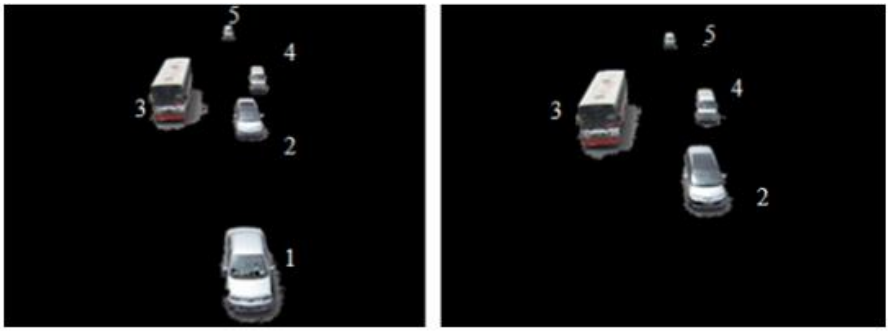
$$\begin{cases} C_1 = \dfrac{W(V_v - v_2)}{u_2 - u_1} \\ C_2 = \dfrac{L(V_v - v_1)(V_v - v_2)}{v_1 - v_2} \end{cases}$$ (معادله ۲-۲۵)

بعد از این مراحل نوبت به تشخیص خودرو در تصاویر ویدئویی می‌رسد. در این‌جا از روشی مشابه با روش [۱۳] برای به دست آوردن پس‌زمینه‌ی سازگار استفاده می‌کنیم که برای بخش‌بندی دقیق خودرو ضروریست. سپس از تصاویر ویدئویی برای مقایسه با تصویر پس‌زمینه به منظور استخراج پیش-

زمینه استفاده می‌شود (شکل ۲-۲۵). برای ردگیری موقعیت خودرو از روشی مشابه با [۱۴] و [۱۵] استفاده می‌شود.



شکل ۲-۲۵ خودرو های تشخیص داده شده در مرجع [۱۲]



شکل ۲-۲۶ نتایج ردیابی خودرو در مرجع [۱۲]

الگوریتم مورد نظر در ***Visual c++*** و بر روی لپ تاپ لنوو مدل ***w500*** با پردازنده‌ی اینتل ***Centrino 2 vPro*** با فرکانس کاری ***2GHz*** و به کمک دوربین ***Canon*** به منظور آزمایش اولیه، پیاده‌سازی گردیده است.

از ۳ خودرو با سرعت ثابت ***80 Km/h*** و برای ۱۰ بار اجرای برنامه و هر یک در یک بخش از جاده استفاده شده است. در ۳۰ مورد، سرعت متوسط تشخیص داده شده ***78 Km/h*** با انحراف معیار ***1.3 Km/h*** بوده است که خطایی برابر با ***3.3 Km/h*** یا خطای ***4*** درصد را می‌دهد.

در مرجع شماره‌ی [۱۶]، الگوریتم تشخیص سرعت خودرو و کاربردش برای سیستم نظارتی هوشمند توسط ایستگاه های کاری کامپیوتری در خط انتخاب شده‌ی جاده توسط روش تفاضل قاب-

ها، ارائه شده است. تفکیک پذیری زمانی معمول ۳۰ میلی‌ثانیه تا ۴۰ میلی‌ثانیه در دوربین‌های مرسوم به کار گرفته شده است.

این مقاله الگوریتم تشخیص سرعت خودرو را ارائه می‌دهد که بعدا در سیستم بینایی یکپارچه[۱] پیاده‌سازی خواهد شد. الگوریتم پیشنهادی توسط تکنیک تفاضل قابی توسعه پیدا کرده است و فرمول بندی سرعت خودرو، از معادله‌ی حرکتی‌اش گرفته شده است. عملکرد الگوریتم پیشنهادی بر روی دنباله‌ی ویدئویی واقعی تست شده است. این مقاله الگوریتم تشخیص سرعت خودرو و کاربردهایش در سیستم‌های نظارتی هوشمند را عنوان می‌کند.

بخش دوم مقاله، فرمول‌بندی سرعت خودرو را شرح می‌دهد. بخش سوم نحوه‌ی به دست آوردن زمان و تکنیک تشخیص شی در حال حرکت را توصیف می‌کند. در بخش چهارم در مورد نتایج عملی بحث می‌شود و در بخش پنجم نیز در مورد عملکرد الگوریتم پیشنهادی نتیجه‌گیری می‌شود.

فرض کنید که یک خودرو تنها یک شی دارای جرم است. با چشم‌پوشی از نیروهایی که روی آن اثر می‌کند، می‌توان سرعت خودرو را با بررسی معادله‌ی حرکتیش به دست آورد. یک خودرو در حال حرکت را که برای n قاب ویدئویی ردیابی می‌شود، در نظر بگیرید. در این صورت یک مجموعه از n تا مختصات دو بعدی در دسترس است ($[\lambda_1, \lambda_2, \dots, \lambda_n] \epsilon z^2$). این مختصات‌های دو بعدی در واقع مختصات‌های سه بعدی در فضای دنیای واقعی‌اند که به فضای صفحه تبدیل شده‌اند. ($[\gamma_1, \gamma_2, \dots, \gamma_n] \epsilon R^3$). با گرفتن هر دو نقطه‌ی λ_n و λ_{n-1}، سرعت می‌تواند به صورت نرخ تغییرات در جابه‌جایی با توجه به تغییر در زمان ($\Delta\tau$) محاسبه شود.

$$v_\tau = \frac{\lambda_n - \lambda_{n-1}}{\tau_n - \tau_{n-1}} \qquad \text{(معادله ۲-۲۶)}$$

$$v_\tau = \lim_{\Delta\tau \to 0} \frac{\Delta\lambda}{\Delta\tau} = \frac{dL}{d\tau} \qquad \text{(معادله ۲-۲۷)}$$

از آن جایی که شی در حال حرکت (خودرو) از قاب ویدئویی استخراج می‌شود، لذا λ_n مختصات پیکسل تصویر را ارائه می‌دهد ($\lambda_n = (x, y)$). در نهایت سرعت خودرو بر حسب پیکسل بر ثانیه گزارش می‌شود. این برای سرعت در دنیای واقعی مطلوب نیست زیرا واحد آن متر بر ثانیه (یا کیلومتر بر ساعت) است. بنابراین مختصات پیکسل که از تصویر به دست آمده است، در ابتدا باید به مختصات دنیای واقعی بر حسب متر تبدیل شود. به بیانی دیگر، کافیست رابطه‌ی یک پیکسل با مسافت را بر

[۱] ***Embedded***

حسب متر در دنیای واقعی بیابیم. برای این که دقیقا این کار را انجام دهیم، یک فرآیند کالیبراسیون نیاز است. بدین طریق می‌توان نسبت بین پیکسل‌ها و طول بر حسب متر را تعریف کرد.

فرض کنید که نسبت دنیای واقعی به فضای پیکسل توسط $\frac{\omega}{\varsigma}$ داده شده باشد، در این صورت سرعت از رابطه ی زیر به دست می آید:

(معادله ۲-۲۸)

$$v_\tau = \frac{(\lambda_n - \lambda_{n-1})\frac{\omega}{\varsigma}}{\tau_n - \tau_{n-1}} ms^{-1}$$

ضمنا این رابطه تا زمانی درست است که هیچ گونه اعوجاج پرسپکتیوی نداشته باشیم. به بیان دیگر طول بر حسب پیکسل و فاصله در مختصات دنیای واقعی خطی باشد. مشخصه‌ی خطی حفظ می‌شود اگر نسبت $\frac{\omega}{\varsigma}$ در تمام تصویر ثابت بماند. بنابراین دوربین باید به طریقی تنظیم گردد که مسیر پیمایش خودرو شرایط فوق را حفظ کند.

فرمول‌بندی سرعت در قسمت فوق تعریف شد. مرحله‌ی بعدی به دست آوردن λ و τ از دنباله‌ی ویدئویی است. بدون این دو پارامتر، نمی‌توان سرعت را تخمین زد.

تنها اختلاف زمانی برای محاسبه‌ی سرعت نیاز است و زمان مطلق سیستم مهم نیست. زمان می‌تواند از نرخ دنباله‌ی ویدئویی (تعداد قاب‌ها بر ثانیه) به دست آید. معمولا، یک دنباله‌ی ویدئویی دارای نرخی بین ۲۵ قاب بر ثانیه تا ۳۰ قاب بر ثانیه است. زمان بین دو قاب متوالی (تفکیک پذیری زمانی)، $\frac{1}{25}$ یعنی ***40*** میلی‌ثانیه یا $\frac{1}{30}$ یعنی ***33.33*** میلی‌ثانیه است. تفکیک پذیری زمانی محدود به نرخ ویدئویی دوربین است. هم اکنون الگوریتم پیشنهاد شده توسط دوربین معمولی که دارای این تفکیک پذیری زمانی است، تست شده است. بنابراین بیشینه‌ی سرعتی که می توان اندازه‌گیری نمود به دو فاکتور بستگی دارد: ۱) تفکیک پذیری زمانی و ۲) تفکیک پذیری جابه‌جایی بر حسب پیکسل در تصویر.

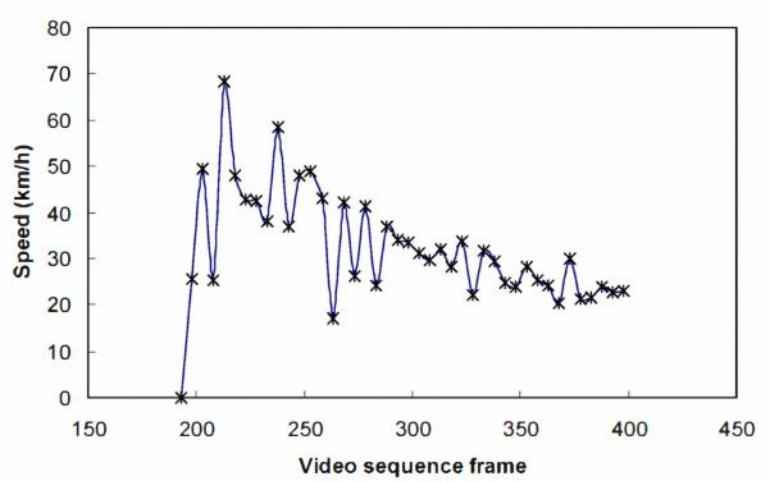
هدف اصلی از بخش بندی حرکت، جداسازی نواحی خودرو مورد نظر از پس‌زمینه است. روش‌های خیلی زیادی برای به دست آوردن نواحی حرکت وجود دارد که از جمله‌ی آن‌ها روش تفاضل زمانی قاب، کم کردن پس‌زمینه و شار نوری است. در کاربردهای بی درنگ، از روش شار نوری به دلیل پیچیدگی در محاسبات اجتناب می‌شود.

تفاضل قاب ساده‌ترین تکنیک تشخیص حرکت است و این تکنیک در الگوریتم پیشنهاد شده به تصویب رسیده است. این تکنیک از قاب ویدئویی در زمان τ استفاده کرده و آن را از قاب ویدئویی در زمان $\tau - 1$ کم می‌کند. اگر تفاضل قاب تنها از یک قاب قبلی استفاده کند، حافظه‌ی زیادی را

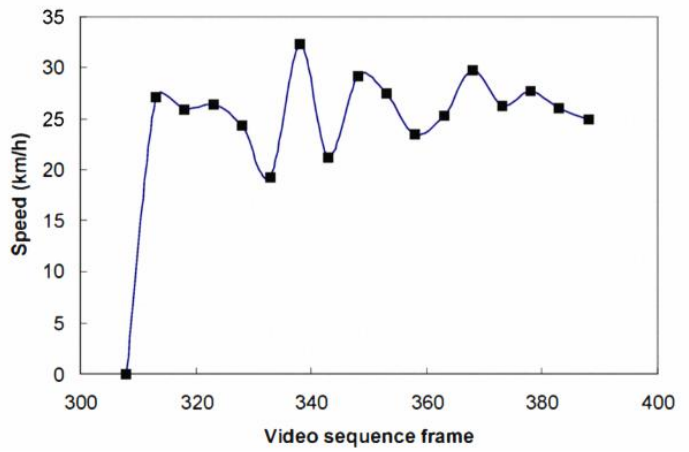
مصرف نمی‌کند و لذا برای کاربردهای یکپارچه[1] مناسب است. علاوه بر این، مزایای دیگر روش تفاضل قاب این است که به تغییرات روشنایی حساس نیست و به مقداردهی اولیه‌ی پس زمینه نیازی ندارد.

یک قاب خاکستری در زمان τ به صورت $I_\tau(x,y)\epsilon z^2$ با ابعاد $m\times n$ ($0<x<m$ و $0<y<n$) تعریف می‌شود. بنابراین تفاضل قاب به صورت زیر تعریف می‌شود:

$$D_\tau = argmax(I_\tau - I_{\tau-1})$$ (معادله ۲-۲۹)



نمودار ۲-۳ سرعت خودرو هدف در قاب‌های دنباله‌ی ویدئویی در مرجع [۱۶]



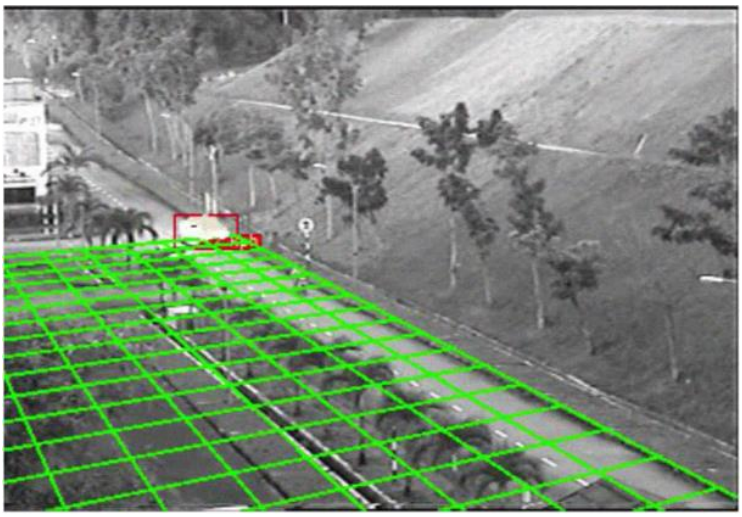
نمودار ۲-۴ سرعت موتورسیکلت هدف در قاب‌های دنباله‌ی ویدئویی در مرجع [۱۶]

شکل‌های ۲-۲۷ و ۲-۲۸، یک نمایی از الگوریتم پیشنهادی برای تشخیص سرعت خودرو را نشان می‌دهد. برای هر دنباله‌ی ویدئویی، اشیای در حال حرکت با ناحیه‌ی مستطیلی علامت‌گذاری

[1] ***Embedded***

می‌شوند. در شکل ۲-۲۷، برای تصویرسازی، ازیک دنباله‌ی ویدئویی ۷ ثانیه‌ای تصویر استخراج شده است. همچنین شکل ۲-۲۷ حرکت یک خودرو هدف را که با ناحیه‌ی مستطیلی مشخص شده است در راستای لاین جاده با سرعت تخمین زده شده‌ی ***17 Km/h*** نشان می‌دهد. در شکل ۲-۲۸ یک تصویر در ۳ بعد به همراه پرسپکتیو ارائه شده است که محاسبه‌ی سرعت را نشان می‌دهد.

شکل ۲-۲۷ مثالی از تخمین سرعت ارائه شده به صورت یک قاب ویدئویی در مرجع [۱۶]



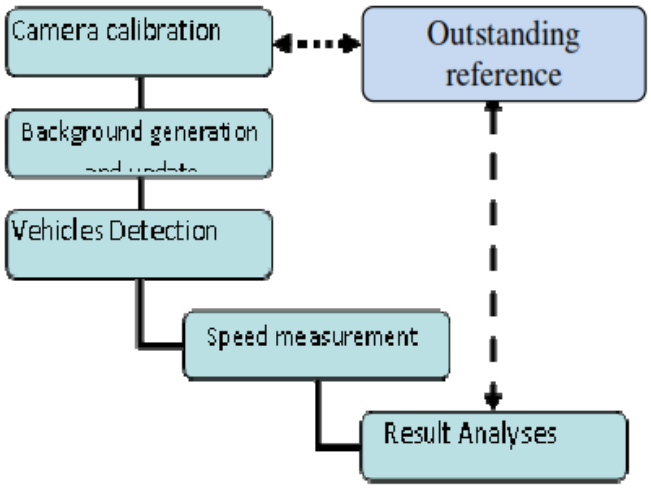
شکل ۲-۲۸ تخمین سرعت ارائه شده به صورت یک قاب ویدئویی در ۳ بعد با پرسپکتیو در مرجع [۱۶]

در مرجع شماره‌ی [۱۷]، هدف معرفی الگوریتمی برای تشخیص سرعت خودرو در دنباله‌ی تصاویر ویدئویی توسط روش ***CVS***[۱] است. در این مقاله فیلم‌های ترافیکی توسط یک دوربین ساکن در آزادراه، جمع‌آوری شده است. دوربین بر اساس معادلات هندسی، کالیبره شده است. سیستم طراحی

[۱] ***Combination of Value and Saturation***

شده قابلیت توسعه به دیگر کاربردهای مرتبط با ترافیک را نیز دارد. میانگین خطای سرعت تخمینی خودرو در تفکیک پذیری‌ها و دنباله‌های ویدئویی مختلف، حدود ۷ کیلومتر بر ساعت بوده است.

در این مقاله یک الگوریتم جدید پیشنهاد شده است که از ویدیوی دیجیتال، پردازش تصویر و بینایی ماشین برای تشخیص خودکار سرعت خودرو به روشی دقیق، استفاده می‌کند. معماری سیستم را در شکل ۲-۲۹ مشاهده می‌نمایید:



شکل ۲-۲۹ معماری سیستم در مرجع [۱۷]

این الگوریتم برای تشخیص سرعت خودرو تنها به یک دوربین ویدئویی و یک کامپیوتر با پردازنده‌ی ۲ هسته‌ای به همراه نرم افزار متلب که بر روی کامپیوتر نصب است، نیاز دارد. این روش تنها به نصب دوربین به طور مستقیم در بالای جاده بستگی دارد. کالیبراسیون دوربین که بر اساس هندسه و مدل‌های تحلیلی است، ساده بوده و در این مقاله توضیح داده شده است. به علاوه، کالیبراسیون به هیچ اطلاعاتی درباره‌ی دوربین نیاز ندارد و تنها مشخصات دوربین مانند نرخ و ابعاد ویدئو که از طریق نرم افزار به دست می‌آید، مورد نیاز است.

پردازش تصویر در این مقاله یک کار مهم و دیگر بخش پیچیده است. این کار نیازمند پردازش داده مانند حذف و استخراج پس‌زمینه، مکان‌یابی و تشخیص خودروهای در حال حرکت، حذف سایه-ی خودرو، اعمال فیلتر برای اصلاح تصویر و محاسبه‌ی سرعت خودرو و ... است.

داده‌هایی که از دوربین ویدئویی گرفته می‌شود، ترکیبی از قاب‌های تصویر پشت سر هم هستند و هر قاب نیز شامل تعدادی پیکسل بوده که دو نوع داده را شامل می‌شوند یکی پس زمینه و دیگری پیش‌زمینه. پس زمینه شامل اشیای ساکن مانند خودروهای پارک شده، سطح جاده، ساختمان‌ها یا هر نوع شی ساکن دیگر و همچنین شرایط آب و هوایی و زمان روز یا شب است. پیش‌زمینه اشیای در حال حرکت مانند عابرین پیاده، خودروهای در حال حرکت یا هر شی در حال حرکت را شامل می-

شود. به منظور فهمیدن سرعت خودروهای در حال حرکت، گام اول استخراج و حذف پس‌زمینه است. بنابراین پیش‌زمینه که دارای داده‌ی با ارزشی است، می‌تواند استخراج شود و برای اطلاعات مورد نیاز مانند سرعت، دسته‌بندی و تعداد خودروها استفاده شود.

همان طور که در بالا ذکر شد، استخراج و حذف پس‌زمینه یکی از بخش‌های مهم تشخیص خودرو است که برای به دست آوردن داده‌های پیش‌زمینه بسیار مناسب است. علاوه بر این، اگر تفاضل تصویر پس‌زمینه بدون این که تصویر پس‌زمینه را به روزرسانی نماییم صورت گیرد، منجر به نتایج نادرست و غیرمناسب می‌شود. بنابراین روش پویای استخراج پس‌زمینه، موقعی که تغییرات محیطی در یک صحنه وجود دارد با دقت بهتری می‌تواند پس‌زمینه را تخمین بزند.

الگوریتم‌های استخراج پس‌زمینه به چندین نوع تقسیم می‌شود. رایدر (۱۹۹۵) از تخمین سازگار پس‌زمینه و تشخیص پیش‌زمینه توسط فیلترکالمن استفاده کرده است. بایلو (۲۰۰۵) تخمین پس‌زمینه را به کمک توزیع گاوسی برای بخش‌بندی تصویر استفاده کرده است. شای (۲۰۰۲) فیلتر میانه‌ی سازگار را برای تولید پس‌زمینه معرفی کرد و اسد و سید (۲۰۰۹) تخمین ریخت‌شناسی[۱] پس-زمینه را به کار بردند.

در این مقاله روش فیلتر میانگین برای ایجاد پس‌زمینه استفاده شده است زیرا پس‌زمینه در رنگ اصلی نیاز است و همچنین یک راه موثر برای ایجاد پس‌زمینه مخصوصا در تغییرات نور کم و وضعیت دوربین ثابت بوده و نتایج دقیقی را ایجاد می‌کند. معادله‌ی استفاده شده در این مقاله در زیر توصیف شده است:

$$K_{xy}(t_n) = \frac{\sum_{m=0}^{j-1} f_{xy}(t_{n-m})}{j}$$ (معادله ۲-۳۰)

که n تعداد قاب‌ها، $f_{xy}(t_n)$ مقدار پیکسل (x, y) در n امین قاب، $k_{xy}(t_n)$ مقدار میانگین پیکسل (x, y) در n امین قاب است که طی j قاب قبلی میانگین‌گیری شده و j نیز تعداد قاب‌هایی است که برای محاسبه‌ی میانگین مقدار پیکسل‌ها استفاده می‌شود.

در این جا یک روش که ترکیب اشباع و مقدار یا ***CVS*** نامیده می‌شود به کار گرفته می‌شود. روشی که برای استخراج پیش‌زمینه بر مبنای تفاضل قاب‌ها استفاده می‌گردد، در ادامه توصیف شده است:

$$N_{xy}(t_n) = \begin{cases} 1, \text{ Foreground} & |f_{xy}(t_n) - f_{xy}(t_{n-1})| > T \\ 0, \text{ Background} & |f_{xy}(t_n) - f_{xy}(t_{n-1})| < T \end{cases}$$ (معادله ۲-۳۱)

[۱] ***morphological***

که $N_{xy}(t_n)$ برابر با مقدار پیش‌زمینه یا پس‌زمینه‌ی تصویر در پیکسل (x, y) در n امین قاب و T نیز آستانه‌ی به کار رفته برای تمییز بین پیش‌زمینه و پس‌زمینه است.

عیب این روش این است که آستانه‌ای ثابت به تمام پیکسل‌ها اعمال می‌شود و هنگامی که شی دارای سرعت کمی است، مقدار تفاضل قاب‌ها کوچک شده و لذا استفاده از آستانه‌ی کوچک می‌تواند لکه‌های پیش‌زمینه را استخراج کند و لکه‌های پس‌زمینه‌ی بیش‌تری به عنوان لکه‌های پیش‌زمینه معرفی می‌شوند. از سوی دیگر، آستانه‌ی زیاد باعث تشخیص ناقص پیش‌زمینه می‌شود. همچنین همین سختی در مورد استخراج پیش‌زمینه به روش گاوسی نیز وجود دارد به طوری که انحراف استانداردی به منظور تعیین آستانه به کار گرفته می‌شود و لکه‌های پیش‌زمینه ممکن است به اشتباه به عنوان پس‌زمینه محاسبه شود. در این مقاله، روش ***CVS*** برای محاسبه‌ی پیش‌زمینه پیشنهاد شده است.

نتایج مهمی که در این‌جا به دست آمد، این بود که در هر قاب، رنگ یا ***Hue*** تاثیرگذار نیست و تقریبا ثابت است اما در طول روز تناوب شدت نور باعث تغییر و تاثیرگذاری بر روی قاب‌ها می‌شود. بنابراین، ضروریست که از مدل رنگ سازنده و موثری مانند ***HIS***[۱] به جای فضای رنگ ***RGB*** استفاده شود. اما از آن‌جایی که مدل ***HIS*** در نرم افزار متلب تعریف نشده است، از مدل ***HSV***[۲] به جای آن استفاده شده است.

روشی که در این جا برای تشخیص شی ارائه شده است، اولین بار است که در آن پس‌زمینه بر اساس مقدار و اشباع رنگ استخراج می‌شود و سپس از قاب جاری که آن نیز بر اساس مقدار و اشباع است، کم شده و سرانجام نتایج با یکدیگر ترکیب می‌شود. فلوچارت مربوطه در ادامه قابل ملاحظه است. روش پیشنهادی تشخیص پیش‌زمینه بر اساس مقدار و اشباع فضای رنگ است.

بعد از استخراج پیش زمینه تصاویر جمع آوری شده به تصاویر باینری تبدیل می‌شوند زیرا عملیاتی همچون تشخیص لبه، حذف نویز و گسترش[۳] و لیبل‌گذاری شی، در قالب باینری مناسب‌تر هستند.

در هر قاب سرعت خودرو توسط موقعیت خودرو در آن قاب محاسبه می‌شود لذا گام بعدی یافتن کادر در بر گیرنده‌ی خودرو[۱] و مرکز ثقل آن است. مرکز ثقل خودرو برای یافتن مسافت پیموده شده-

[۱] ***Hue, Saturation, intensity***

[۲] ***Hue, Saturation, Value***

[۳] ***dilation***

ی خودرو در حال حرکت در قاب‌های متوالی مهم بوده و همچنین از آن‌جایی که نرخ قاب حرکات گرفته شده مشخص است، امکان محاسبه‌ی سرعت وجود دارد.

این اطلاعات باید به طور متوالی در یک سلول آرایه‌ای هم اندازه با تصاویر گرفته شده توسط دوربین، ذخیره شود زیرا مسافت طی شده توسط مرکز ثقل به منظور تشخیص سرعت خودرو مورد نیاز است. برای فهمیدن مسافت طی شده توسط پیکسل، فرض می‌شود که پیکسل دارای مختصاتی به صورت زیر است:

$$i = (a, b), \quad i - 1 = (e, f)$$ (معادله ۲-۳۲)

که موقعیت مرکز ثقل در قاب ***i*** و ***i-1*** برای یک خودرو با مختصات ***(a, b)*** و مختصات ***(e, f)*** نشان داده شده است.

مسافت پیموده شده طی یک قاب توسط خودرو برابر است با:

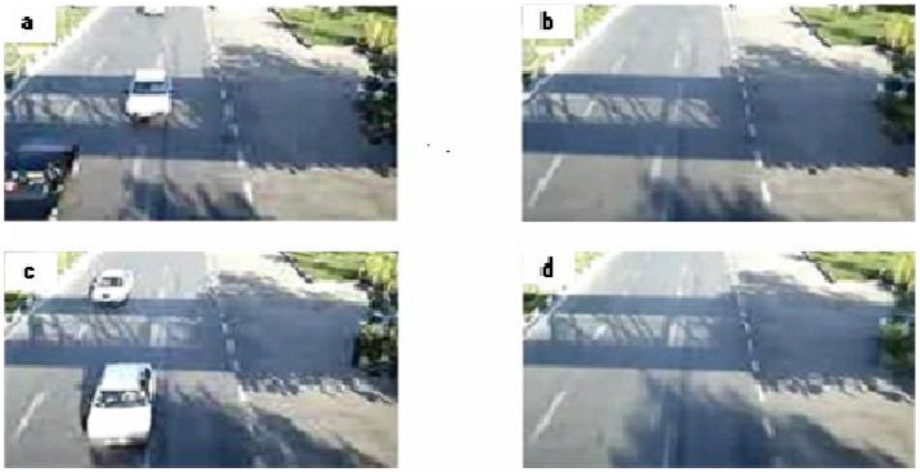
$$d_i = \sqrt{(a-e)^2 + (b-f)^2}$$ (معادله ۲-۳۳)

و اگر دنباله‌ی تصاویر ۲۵ قاب بر ثانیه باشد، زمان بین دو قاب متوالی برابر با ***0.04*** ثانیه است و سرانجام سرعت را می‌توان از معادله‌ی زیر به دست آورد:

$$V = K\frac{\Delta x}{\Delta t}$$ (معادله ۲-۳۴)

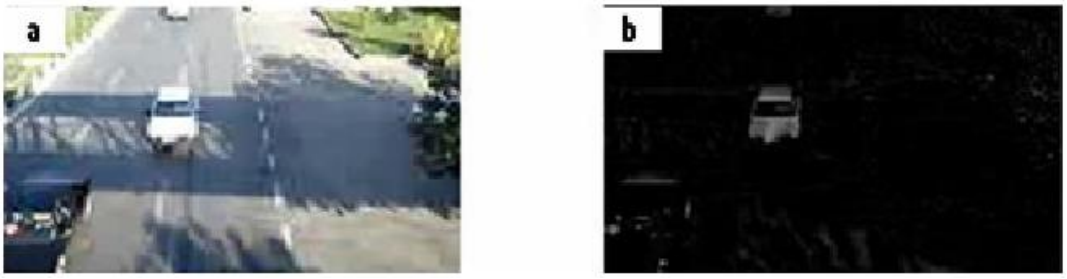
که ***K*** ضریب کالیبراسیون است.

عملیات ریخت‌شناسی، شکل تصویر را حفظ کرده و آن را ساده می‌کند و کیفیت شی را نیز افزایش می‌دهد.



[1] ***blobs bounding box***

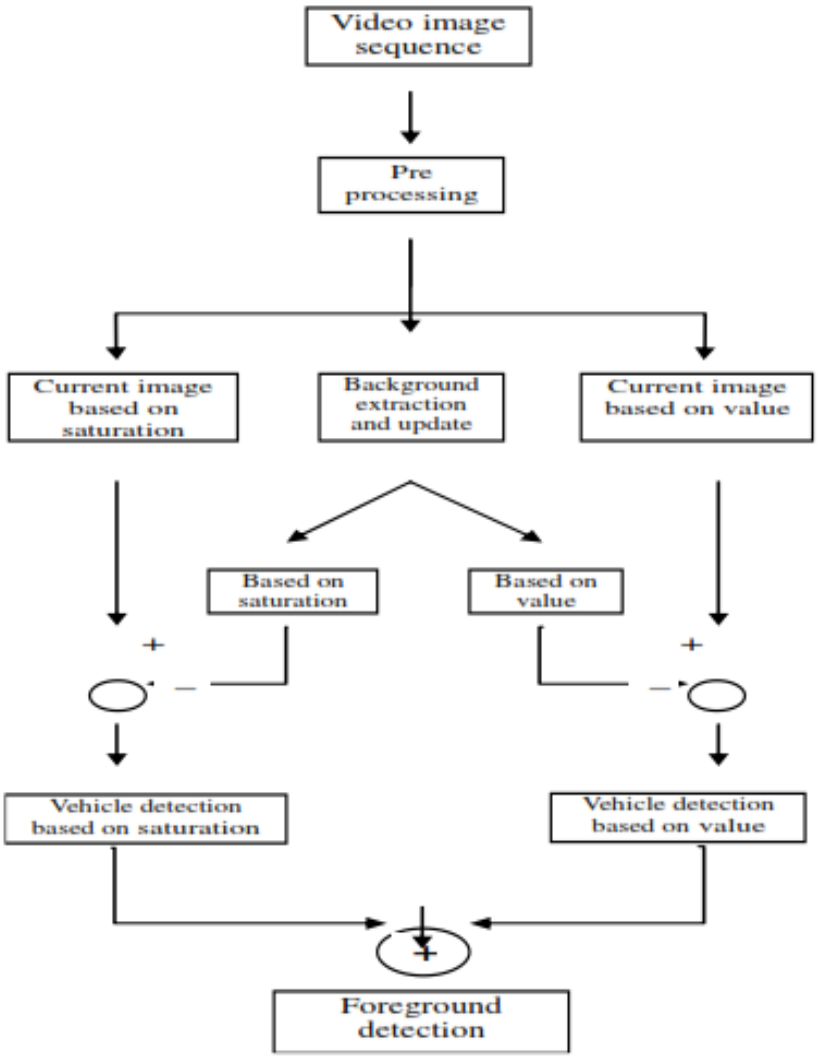
شکل ۲-۳۰ نتایج استخراج پس‌زمینه. (***a***) قاب شماره‌ی ۲۲ ویدئو. (***b***) قاب شماره‌ی ۲۲ پس‌زمینه (***c***) قاب شماره‌ی ۳۰ ویدئو (***d***) قاب شماره‌ی ۳۰ پس‌زمینه در مرجع [۱۷]



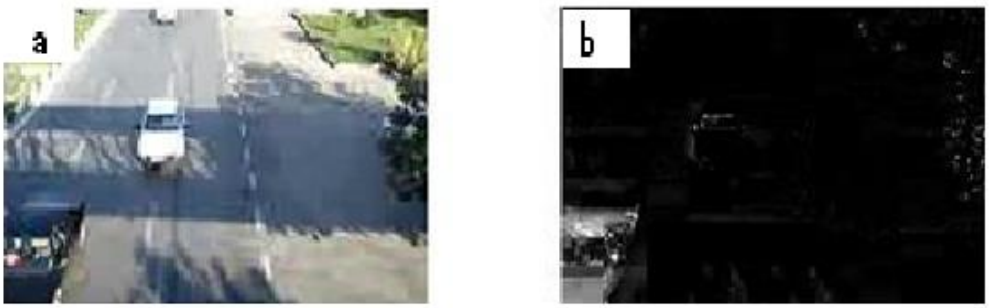
شکل ۲-۳۱ تشخیص شی در حال حرکت بر اساس مقدار. (***a***) قاب شماره‌ی ۲۲ ویدئو (***b***) شی در حال حرکت بر اساس مقدار در مرجع [۱۷]

شکل ۲-۳۲ پس زمینه بر اساس اشباع (***a***) قاب شماره ی ۲۲ ویدئو (***b***)قاب بر اساس اشباع (***c***)پس زمینه بر اساس اشباع در مرجع [۱۷]

شکل ۲-۳۳ پس زمینه بر اساس مقدار (***a***) قاب شماره‌ی ۴۳ ویدئو (***b***) قاب بر اساس مقدار (***c***) پس‌زمینه بر اساس مقدار در مرجع [۱۷]

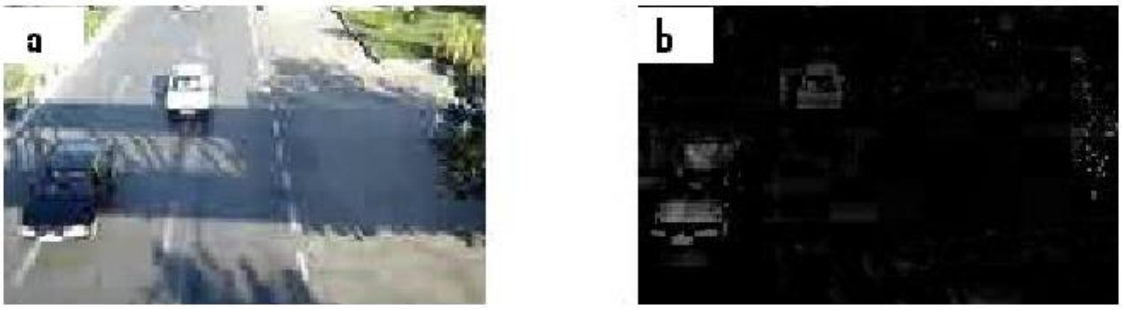


شکل ۲-۳۴ فلوچارت تشخیص پیش‌زمینه در مرجع [۱۷]

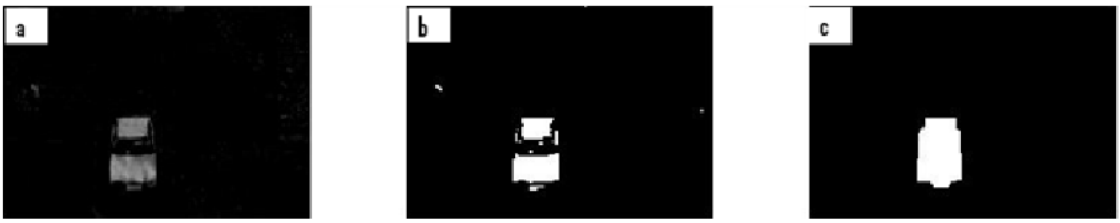


شکل ۲-۳۵ تشخیص شی در حال حرکت بر اساس اشباع. (***a***) قاب شماره‌ی ۲۲ ویدئو (***b***) شی در حال حرکت بر اساس اشباع در مرجع [۱۷]

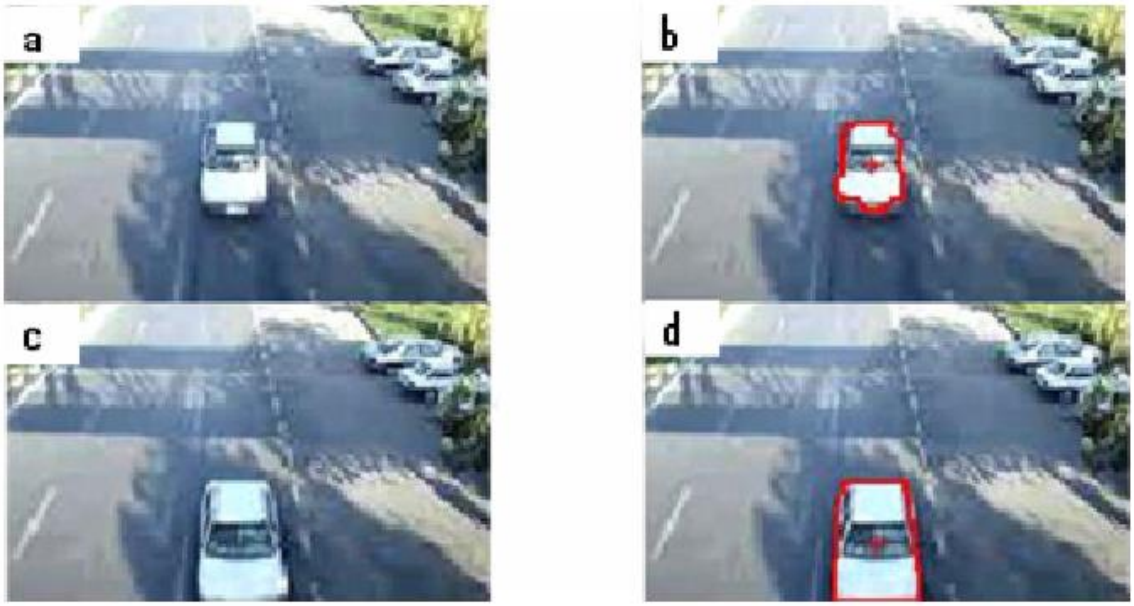
عملیات ریخت شناسی عموما برای مواردی چون بهبود ساختار شی، پیش پردازش تصویر، بخش‌بندی شی و اندازه‌گیری مساحت و محیط استفاده می‌شود.



شکل ۲-۳۶ تشخیص شی در حال حرکت بر اساس اشباع و مقدار (***a***) قاب شماره‌ی ۲۲ ویدئو (***b***) شی در حال حرکت بر اساس اشباع و مقدار در مرجع [۱۷]



شکل ۲-۳۷ حذف نویز و عملیات گسترش. (***a***)شی در حال حرکت. (***b***) تصویر باینری. (***c***) بعد از حذف نویز و پر کردن سوراخ ها در مرجع [۱۷]



شکل ۲-۳۸ کادر مرزی و مرکز ثقل. (***a***) قاب شماره‌ی ۴۵. (***b***) کادر مرزی و مرکز ثقل در قاب شماره‌ی ۴۵. (***c***) قاب شماره‌ی ۵۱ ویدئو. (***d***) کادر مرزی و مرکز ثقل در قاب ۵۱ در مرجع [۱۷]

در این مقاله، از تکنیک‌های باز کردن[۱]، بستن[۱]، فرسایش[۲] و گسترش[۳] برای پر کردن سوراخ‌های درون شی در حال حرکت استفاده شده است که عموما طی پردازش تصویر و ویدئو اتفاق می‌افتد و باعث اختلال و خطا در گام نهایی که به دست آوردن اطلاعات ترافیکی است، می‌شود.

[۱] ***opening***

عملیات گسترش برای توسعه‌ی حاشیه‌ی ناحیه‌ی شی در حال حرکت به کار می‌رود. پیکسل-های پیش‌زمینه در حالی که سوراخ‌های درون آن نواحی کوچک‌تر می‌شود، بزرگ می‌شوند. از سوی دیگر، عملیات فرسایش حاشیه‌ی پیکسل‌های پیش‌زمینه را فرسایش می‌دهد و مرزهای اشیای در حال حرکت در اندازه و شکل کوچک‌تر می‌شوند و سوراخ‌ها از لحاظ اندازه و تعداد، بیش‌تر می‌شوند. عملیات ***opening*** شامل یک عملیات فرسایش و سپس یک گسترش است. یک ***closing*** عکس عمل یک ***opening*** بوده و عملکرد اصلی این عملیات حذف نویز است. اثر اصلی ***opening***، حذف اشیای کوچک که در حقیقت اشیای در حال حرکت نیستند، از پیش‌زمینه است و آن‌ها را در پس‌زمینه قرار می‌دهد در حالی که ***closing*** سوراخ‌های کوچک درون اشیای در حال حرکت را حذف می‌نماید.

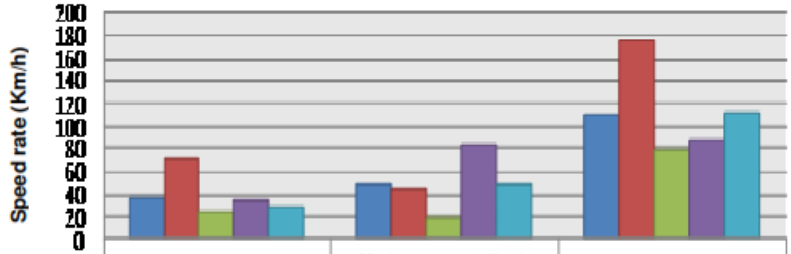
سرانجام برای تشخیص سرعت همان‌طور که پیشنهاد داده شد، کادر مرزی و مرکز ثقل تعیین می‌شود. نتایج حاصل شده برای دو قاب متوالی در شکل بالا نشان داده شده است.

سرانجام برای روش ارائه شده در مرجع [۱۷]، نتایج در جدول ۲-۶ آورده شده است:

جدول ۲-۶ نتایج تجربی سامانه‌ی تعیین سرعت در مرجع [۱۷]

Real speed (Km/h)	X_1 (pixel) (x, y)	X_2 (pixel) (x, y)	$\Delta x = X_1 - X_2$ (pixel)	Δx (m)	Error	System detection speed (Km/h)
40	(64,76)	(74,69)	12.2	0.81	0.03	38.6
50	(63,47)	(79,42)	16.76	1.12	0.01	50.4
60	(64,79)	(85,84)	21.58	1.44	0.07	64.7

جدول ۲-۷ نتایج تجربی تعیین سرعت با مدل‌های پس زمینه‌ی مختلف در مرجع [۱۷]



	Low speed (Km/h)	Medium speed (Km/h)	High speed (Km/h)
Mean filters (CVS)	38.06	49.3	110
Median filter	70.7	44.5	177
Framed different	24.5	18.1	78
Gaussian model	34.7	83	87.1
Real speed	30	50	112.7

[۱] ***closing***

[۲] ***erosion***

[۳] ***dilation***

برنامه در محیط متلب پیاده‌سازی شده است و قادر به پردازش ۱۲ قاب در ثانیه بر روی یک پردازنده‌ی دو هسته‌ای با فرکانس ۲ گیگاهرتز است.

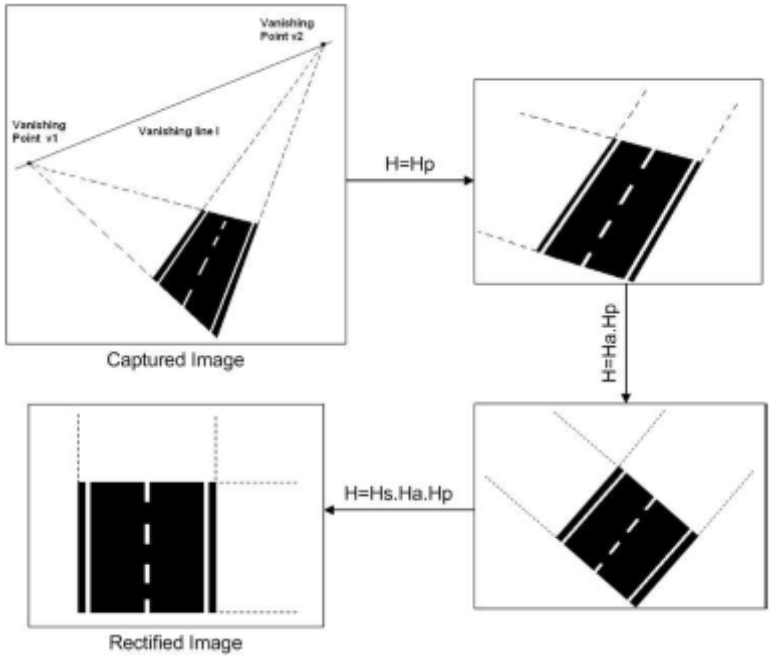
در مرجع شماره‌ی [۱۸]، روش خودکاری به منظور تخمین سرعت خودرو و تعیین تعداد خودرو ها (شدت ترافیک) در هر لاین توسط تصاویر اصلاح شده، ارائه شده است. این روش نیازمند دانستن دو طول از سطح زمین بوده و می‌تواند به بزرگراه‌هایی که دارای لاین‌های نسبتا صافی در نواحی نزدیک به دوربین هستند نیز اعمال شود. یک فاکتور مقیاس نیز تعیین می‌شود تا سرعت خودرو بر حسب واحد واقعی به دست آید. این ضریب مقیاس، فواصل بر روی صفحه‌ی تصویر را به فواصل بر روی صفحه‌ی زمین مرتبط می‌کند و می‌تواند توسط تعیین دوره و موقعیت خطوط راه راه بزرگراه بر روی تصویر اصلاح شده‌ی پس‌زمینه به دست آید. هنگامی که این خطوط مکان‌یابی می‌شوند، امکان مکان-یابی مرز لاین‌های بزرگراه و همچنین تعیین سرعت متوسط خودرو در هر لاین وجود دارد.

تبدیل افکنشی مربوط به شکل‌دهی تصویر، مشخصات هندسی خاصی از جمله طول، زاویه و نسبت مساحت‌ها را تحریف می‌کند در نتیجه به کارگیری دنباله‌های تصاویر یا ویدئویی در نظارت‌های ترافیکی مخصوصا برای تخمین سرعت خودرو چالش برانگیز است. برای حل این مشکل می توان از تصاویر اصلاح شده استفاده نمود که مشخصات هندسی از دست رفته‌ی تصاویر را بازیابی می‌کند. طی سال‌ها، چندین روش برای اصلاح تصاویر توسعه داده شده است که در مراجع [۱۹] و [۲۰] و [۲۱] مشاهده می‌شود. در این مقاله از روش ارائه شده در مرجع [۲۲] برای اصلاح تصویر استفاده شده است. این روش به تخمین دو نقطه‌ی محوشدگی[۱] و دانستن دو زاویه بر روی صفحه‌ی زمین نیازمند است. به دلیل وجود تعداد خطوط عمود و موازی زیاد به خاطر طبیعت ساختار جاده، این پارامترها به آسانی می توانند به دست آیند. در یک روش کلی، این روش تبدیل افکنشی را توسط انتشار سه ماتریس یا تبدیل، تخمین می‌زند.

$$\mathrm{H} = H_s . H_a . H_p$$ (معادله ۲-۳۵)

که H_s تبدیل تشابه را ارائه می‌دهد، H_a ماتریس همگر و H_p تبدیل افکنش خالص است. در شکل ۲-۳۹ فرآیند افکنش را مشاهده می کنید:

[۱] ***vanishing***



شکل ۲-۳۹ مراحل فرآیند اصلاح تصویر در مرجع [۱۸]

در شکل ۲-۴۰ لبه‌های تشخیص داده شده از پیش‌زمینه توسط آشکارساز لبه‌ی سوبل نشان داده شده است. خطوطی که در شکل ۲-۴۰ مشاهده می‌شود، از فرآیند تشخیص لبه به دست آمده است.

شکل ۲-۴۰ (***a***) لبه های شناسایی شده توسط الگوریتم سوبل (***b***) خطوط مشخص شده توسط لبه‌های ارائه شده در مرجع [۱۸]

وقتی که تصویر اصلاح می‌شود، امکان اندازه‌گیری مسافت پیموده شده توسط خودرو در دو قاب متوالی بر روی صفحه‌ی زمین، وجود دارد. برای انجام این کار باید ضریب مقیاس را به دست آوریم. برای یافتن این ضریب از دوره‌ی خطوط راه راه روی سطح جاده در سطح زمین و تصویر اصلاح شده استفاده می‌کنیم.

برای یافتن محل خودرو در قاب‌ها باید آن‌ها را از پس‌زمینه متمایز کنیم. به این فرآیند بخش-بندی تصویر می‌گوییم. برای اندازه‌گیری سرعت خودروها، باید آن‌ها را ردیابی کرد. در این مقاله، از

فیلتر کالمن برای ردیابی استفاده می‌شود. توسط فیلتر کالمن، مکان بعدی خودرو در قاب‌های متوالی مشخص می‌شود. فرمول‌های زیر برای تخمین موقعیت به کار گرفته می‌شود:

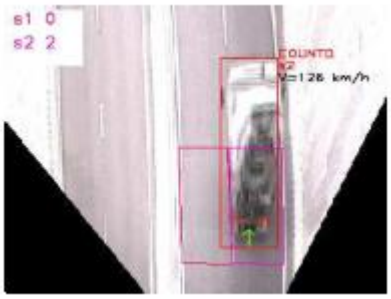
$$\hat{x}(k) = \emptyset(k-1).x(k-1) \quad \text{(معادله ۲-۳۶)}$$

$$\hat{x}(k) = [x, v_x, a_x, y, v_y, a_y, w, v_w, h, v_h]^T \quad \text{(معادله ۲-۳۷)}$$

$$\hat{z}(k) = H.\hat{x}(k) \quad \text{(معادله ۲-۳۸)}$$

$$\hat{z}(k) = [x, v_x, y, v_y, w, h]^T \quad \text{(معادله ۲-۳۹)}$$

$\hat{x}(k)$ حالت تخمین زده شده، φ حالت مدل انتقالی، (x, y) موقعیت خودرو، v_x و v_y سرعت و a_x و a_y شتاب خودرو هستند. l و h عرض و ارتفاع و v_l و v_w نیز تغییرات عرض و ارتفاع است. در شکل ۲-۴۱ مثالی از اعوجاج شکل خودرو به دلیل فرایند اصلاح آورده شده است:

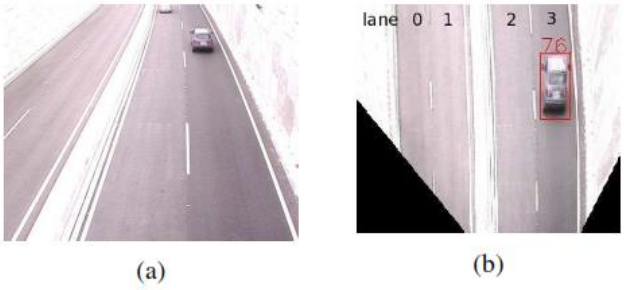


شکل ۲-۴۱مثالی از اعوجاج شکل خودرو به دلیل فرایند اصلاح در مرجع [۱۸]

در این بخش چندین نتیجه از الگوریتم اصلاح و تخمین سرعت خودرو ارائه شده است. شکل ۲-۴۲ نتیجه‌ی حاصل از این دو را نشان می‌دهد.

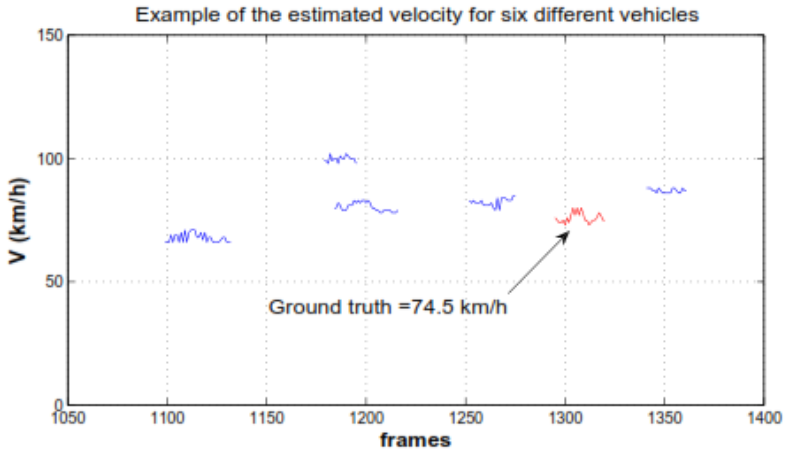
شکل ۲-۴۲مثالی از تصویر گرفته شده در سمت چپ و تصویر اصلاح شده در سمت راست در مرجع [۱۸]

سرعت واقعی خودرو توسط ***GPS*** اندازه‌گیری شده و در نتیجه خطا با مقایسه‌ی سرعت تخمین زده شده و سرعت اصلی به دست آمده است. خطای گزارش شده، ۲ درصد بوده است.



شکل ۲-۴۳ تصویر گرفته شده و اصلاح شده در مرجع [۱۸]

در نمودار ۲-۵ سرعت اندازه‌گیری شده‌ی چندین خودرو را مشاهده می‌نمایید:



نمودار ۲-۵ سرعت اندازه‌گیری شده‌ی چندین خودرو در مرجع [۱۸]

در مرجع شماره‌ی [۲۳]، یک روش برای تخمین سرعت خودرو توسط ردیابی حرکت به کمک دنباله‌ی تصاویر ارائه شده است. ردیابی حرکت به کمک الگوریتم ***Lucas-Kanade-Tomasi*** انجام می شود. معادلات سرعت نیز برای یک مدل فضای حالت دینامیکی فرمول‌بندی می‌شود. فیلتر کالمن و توسعه یافته‌ی کالمن برای تخمین سرعت شی و تخمین مکان شی در قاب‌های بعدی به کار رفته است. این الگوریتم پیشنهاد شده در دنیای واقعی به کار گرفته شده است و به کمک یک دوربین کالیبره نشده سرعت خودروها را در قاب‌های ویدئویی تخمین می‌زند. مزیت اصلی این روش این است که ساده و مقاوم بوده و زمان، پیچیدگی و ابهام کمی در تخمین سرعت خودرو دارد.

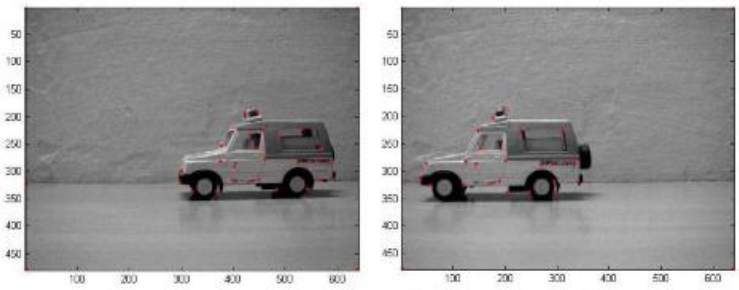
در دنباله‌ی تصاویر، نویز باعث جابه‌جایی روشنایی از مقدار اصلی خود می‌شود. مشکل اصلی پردازش تصویر حذف نویز از تصاویر با حفظ ویژگی‌هایش است. لذا قبل از انجام پردازش و شناسایی

ویژگی‌ها باید عملیات پیش پردازش انجام دهیم. اغلب تکنیک‌های پیش پردازش به منظور حذف نویز و بهبود تصویر است. بهبود تصویر در این‌جا شامل تیزتر کردن ویژگی‌های تصویر است.

سیستم بر مبنای بینایی ماشین اشیا را بر اساس ویژگی‌هایشان تشخیص می‌دهد. استخراج ویژگی اولین گام از تحلیل حرکت است.

لبه‌ها و گوشه‌ها به عنوان ویژگی‌های پایدار تعیین می‌شوند. لبه‌ها نسبت به تغییرات روشنایی در تصویر مقاوم‌اند. از معمول‌ترین آشکارسازهای لبه، الگوریتم کنی[۱] و سوبل[۲] هستند. در این مقاله از آشکارساز سوبل استفاده می‌شود. همچنین برای تشخیص گوشه‌ها از الگوریتم هریس[۳] استفاده شده است.

نقاط به دست آمده از روش‌های ذکر شده باید در دنباله‌ی ویدئویی دنبال شوند. برای این منظور روش‌های همبستگی[۴]، ردیابی مرکز ثقل و ردیابی بر اساس ویژگی‌ها وجود دارد. در این‌جا از الگوریتم آخری و از روش ***KLT*** استفاده می‌شود. این الگوریتم ویژگی‌هارا در دنباله‌های ویدئویی ردیابی می کند. در شکل ۲-۴۴ نمونه ای از کاربرد این الگوریتم و ردیابی نقاط ویژگی را مشاهده می‌کنید:



شکل ۲-۴۴ ردیابی ویژگی توسط ردیاب ***KLT*** در مرجع [۲۳]

در گام اول نویز تصویر حذف می‌شود و نقاط گوشه‌ای نیز توسط آشکارساز گوشه‌ی هریس تعیین می‌شوند. این نقاط توسط نقاطی به رنگ قرمز مشخص می‌شوند. این نقاط توسط الگوریتم ***KLT*** و از یک قاب به قاب دیگر ردیابی می‌شوند.

افکنش کروی یک ابزار برای تعیین ۳ بعدی حرکت روی دنباله‌ی ویدئویی است. به کمک فرمول های زیر این افکنش صورت می‌گیرد:

---

[۱] ***Canny***

[۲] ***Sobel***

[۳] ***Harris***

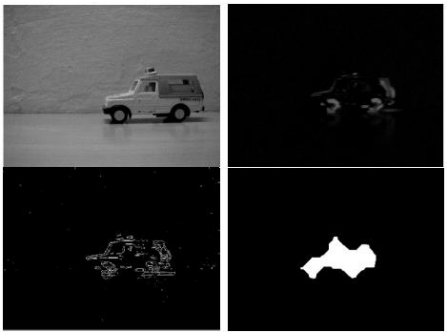
[۴] ***Correlation***

$$P_c = \frac{P_i}{\|P_i\|}$$ (معادله ۲-۴۰)

$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix}, P_i = \begin{bmatrix} X_i \\ Y_i \\ F \end{bmatrix}, P_c = \begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix}$$ (معادله ۲-۴۱)

که x, y, z مختصات نقطه در فضای واقعی، X_i, Y_i, F مختصات در صفحه‌ی تصویر و x_c, y_c, z_c نیز مختصات واحد کروی هستند.

در این‌جا از تبدیلات هندسی به جای کالیبره کردن دوربین استفاده می‌شود. همچنین از عملیات ریخت‌شناسی برای بهبود تصاویر ویدئویی بهره گرفته شده است. در شکل‌های ۲-۴۵ و ۲-۴۶ نحوه‌ی تشخیص ویژگی‌ها و ردیابی آن‌ها نشان داده شده است:



شکل ۲-۴۵ نتایج ردیابی مرکز ثقل در مرجع [۲۳]

شکل ۲-۴۶ عملیات بر روی یک دنباله‌ی ویدئویی ترافیکی در مرجع [۲۳]

در شکل ۲-۴۷ نتایج حاصل از الگوریتم را مشاهده می‌نمایید:

Frame 1st	Frame 5th
(126,213)	(107,209)
(119,221)	(108,220)
(137,223)	(128,220)
(129,231)	(121,228)

(a)

	Vehicle 1	Vehicle 2
Frame 1	(174.00,138.00)	(222.74,128.26)
Frame 2	(172.50,132.00)	(221.53,123.64)
Frame 3	(170.50,125.00)	(219.05,118.90)

(b)

	Length in pixels	Scale Factor (m/pixels)
Vehicle 1	26.33	0.1889
Vehicle 2	22.00	0.2273

(c)

	Speed (in km/hr)
Vehicle 1	73.76
Vehicle 2	62.15

شکل ۲-۴۷ پارامترهای خودرو در مرجع [۲۳]

در مرجع شماره‌ی [۲۴]، هدف تشخیص و ردیابی خودرو از روی دنباله‌ی ویدئویی توسط روش شار نوری و تفاضل پس‌زمینه است. همچنین این مقاله دو روش مختلف برای تخمین سرعت خودروها را با هم مقایسه می‌کند. این سیستم قابلیت توسعه به سایر کاربردهای مرتبط با ترافیک را دارا است. برای تخمین سرعت خودرو این سیستم از پردازش تصویر استفاده می‌کند. حرکت خودرو در طول قاب‌ها توسط الگوریتم شار نوری و تکنیک تفاضل پس زمینه تشخیص داده شده و ردیابی می‌شود. مسافت پیموده شده توسط خودرو به کمک جابه‌جایی مرکز ثقل طی قاب‌های متوالی محاسبه شده و لذا سرعت خودرو تخمین زده می‌شود. بسیاری از تکنیک‌های ردیابی حرکت پیشنهادی بر اساس تطبیق الگو، ردیابی لکه[۱] و ردیابی کانتور هستند. یک تکنیک مشهور تخمین و ردیابی حرکت، شار نوری است که به طور گسترده برای عملی کردن سیستم مراقبتی ترافیکی استفاده و تست نشده است. بنابراین برای تحلیل قابلیت اطمینان و قابلیت عملی کردن این روش، این مقاله ایده‌ی پیاده سازی شار نوری در سیستم مراقبتی ترافیکی را پیشنهاد می‌دهد و همچنین عملکرد آن را مورد ارزیابی قرار می‌دهد. نتایج ردیابی به کمک شار نوری نشان می‌دهد که جریان نوری یک تکنیک عالی برای ردیابی حرکت شی در حال حرکت و دارای پتانسیل عالی برای پیاده‌سازی درون سیستم مراقبتی ترافیکی است.

در این مقاله، به منظور پیاده‌سازی کار پیشنهاد شده برای تشخیص و ردیابی خودرو، دو الگوریتم مختلف استفاده شده است. الگوریتم به کار برده شده الگوریتم شار نوری و تکنیک تفاضل پس زمینه است. کاری که در این‌جا انجام می‌شود، تشخیص و ردیابی یک خودرو در دنباله‌های ویدئویی گرفته

[۱] ***Blob Tracking***

شده توسط یک دوربین است. فرض کنید که بخش‌های جاده مسطح و مستقیم است. خودروها در قاب‌های متوالی تشخیص داده شده و ردیابی می‌شوند. مسافت پیموده شده و سرعت خودرو در طول قاب‌ها توسط تشخیص مرکز ثقل خودروها محاسبه می‌شود. در ادامه دو الگوریتم به همراه نتایج‌شان به طور دقیق پوشش داده می‌شود.

قبل از انجام هر گونه عملیات پردازش تصویر، کیفیت قاب‌ها خیلی مهم است. در تکنیک تفاضل پس زمینه، فاز پیش پردازش به منظور بهبود کیفیت قاب‌ها به همراه ملاحظاتی صورت می‌گیرد و ویدئو با ۳ نوع نویز نمک و فلفل، نویز گاوسی و نویز متناوب تست می‌شود. این نویزها شانس بیش‌تری دارند که کیفیت قاب‌های ویدئویی را کاهش دهند. هر نوع نویزی توسط تکنیک‌های فیلترگذاری مختلفی رفع می‌شود و بهترین فیلتر برای نویزهای مختلف در نظر گرفته شده است. برای تشخیص خودروها عملیات زیر صورت می‌گیرد:

۱- ایجاد یک دنباله از یک ویدئوی ورودی
۲- گرفتن تصویر فعلی و تصویر قبلی
۳- تفاضل‌گیری بین آن‌ها
۴- انتخاب آستانه
۵- انجام عملیات فیلترگذاری برای حذف نویز از تصویر

شکل ۲-۴۸ یک قاب ایجاد شده از ویدئو در مرجع [۲۴]

شکل ۲-۴۹ نتیجه‌ی تفاضل پس‌زمینه در مرجع [۲۴]

هنگامی که خودرو توسط گام‌های بالا تشخیص داده شد، باید مرکز ثقل خودرو تشخیص داده شده را محاسبه کرده و یک مستطیل محاط شده بر آن باید رسم شود که در این جا از تابع

regionprops مربوط به متلب استفاده شده است. به منظور دسته‌بندی شی پیش‌زمینه از پس‌زمینه، عملیات بخش‌بندی توسط تکنیک مدل پس زمینه با تفاضل قاب صورت گرفت که زمان پردازش کم-تری را گرفت. این مرحله عبارت است از:

۱- خواندن اولین قاب ورودی و در نظر گرفتن آن به عنوان قاب پس‌زمینه
۲- تبدیل قاب پس‌زمینه به خاکستری
۳- تنظیم مقدار آستانه
۴- تنظیم متغیرهایی برای اندازه‌ی قاب مانند عرض و ارتفاع قاب پس‌زمینه
۵- انجام پردازش‌های زیر از دومین قاب شروع می‌شود و تا آخرین قاب ویدئو ادامه می‌یابد.
۶- خواندن قاب
۷- تبدیل قاب به خاکستری
۸- یافتن تفاضل قاب‌ها بین قاب فعلی و قاب قبلی
۹- $Diff\_frame = frame\ t - frame\ t\ -1$
۱۰- دسته‌بندی پیکسل‌ها بر اساس این که مربوط به پیش‌زمینه است یا پس‌زمینه

اگر مقدار ***Diff_frame*** از آستانه بیش‌تر باشد، پیکسل مورد نظر، مربوط به پیش‌زمینه است و در آرایه‌ی برداریِ پیش‌زمینه‌ی جدید ذخیره می‌شود و در غیر این صورت مقدار بردار پیش‌زمینه‌ی مربوطه برابر با صفر می‌شود. حال قاب فعلی به عنوان قاب قبلی در نظر گرفته شده و قاب بعدی به عنوان قاب فعلی می باشد. این عملیات تا زمانی که به قاب آخر ویدئو برسیم، ادامه می‌یابد.

روشی که در مورد جابه‌جایی تصویر تصمیم‌گیری می‌کند و فهم آن نیز آسان است، روش مبتنی بر ویژگی است. این روش ویژگی‌هایی همچون لبه‌های تصویر، گوشه‌ها و سایر ساختارها که به خوبی در دو بعد قرار می‌گیرد را پیدا کرده و آن‌ها را از یک قاب تا قاب بعدی ردیابی می‌کند. عمل استخراج ویژگی، اگر به خوبی انجام شود، باعث کاهش میزان اطلاعاتی که باید پردازش شود می‌گردد.

یافتن مساحت شی، مرکز ثقل و پارامتر مستطیل محاط توسط تابع ***regionprops*** در متلب صورت می گیرد. مستطیل محاط و مرکز ثقل بر روی شی قرار می‌گیرد.

به عنوان روش دیگر، الگوریتم شار نوری و فاز تشخیص خودرو را در ادامه مورد بحث قرار می-دهیم. تشخیص شی توسط تحلیل لکه‌ها[۱] و بردارهای حرکتی از دنباله‌ی قاب‌های ویدئویی محاسبه می‌شود. وقتی شی تشخیص داده می‌شود، ویژگی‌ها محاسبه می‌شود. یافتن شار نوری توسط لبه‌ها دارای مزیت‌های زیادی نسبت به تشخیص ویژگی‌های دو بعدی است.

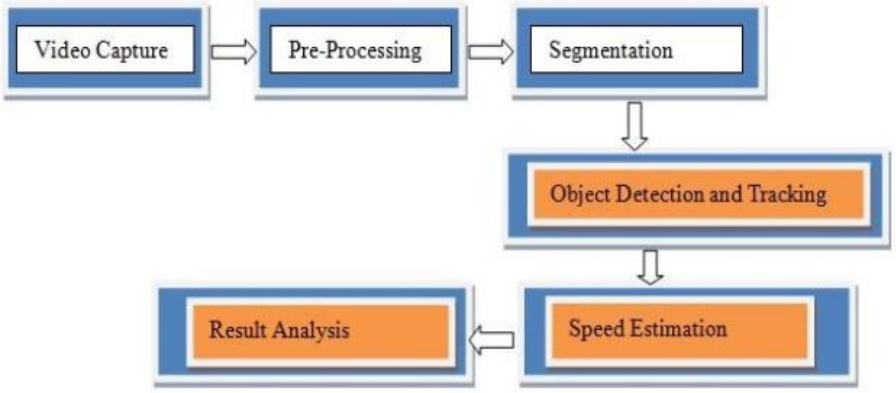
[۱] ***Blobs***

تابع شار نوری، بردار حرکت را که اطلاعات دقیقی درباره‌ی جهت حرکت خودرو می‌دهد، محاسبه می‌نماید. این کار برای محاسبه‌ی حرکت پیکسل‌های یک دنباله‌ی تصویر به کار می‌رود و یک ارتباط پیکسلی انبوه (نقطه به نقطه) را فراهم می‌کند. مسئله تعیین این موضوع است که پیکسل یک تصویر در زمان t در زمان $t+1$ در کجا قرار دارد. در تعداد زیادی از کاربردها، این روش برای تشخیص اشیای در حال حرکت استفاده می‌شود.

محاسبات شار نوری بر مبنای دو فرض است و در ادامه به طور دقیقی به صورت مدل ریاضی توضیح داده می‌شود.

ردیابی شی به فرآیند دنبال کردن شی در حال حرکت در قاب‌های متوالی اتلاق می‌شود. عمل ردیابی توسط استخراج ویژگی اشیا در یک قاب انجام می‌شود و در دنباله‌ی قاب‌ها اشیا را پیدا می-کند.

در شکل ۲-۵۰ بلوک دیاگرام سیستم را مشاهده می‌نمایید. یک دنباله‌ی ویدئویی گرفته شده توسط یک دوربین ویدئویی به عنوان ورودی به این سیستم اعمال می‌شود. بعد از انجام عملیات پیش-پردازش و بخش‌بندی، اشیای شناسایی شده ردیابی می‌شوند و سرعت آن‌ها تخمین زده شده و نتایج حاصل شده مورد ارزیابی قرار می‌گیرند.



شکل ۲-۵۰ معماری سیستم در مرجع [۲۴]

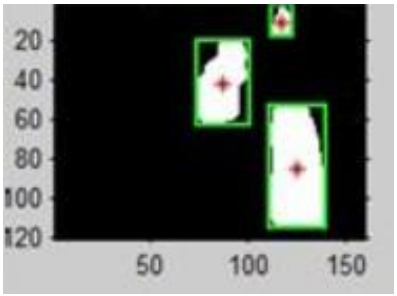
در شکل ۲-۵۱ تفاضل پس‌زمینه بر روی تصویر انجام شده است و تفاضل پیش‌زمینه بر روی باقی مانده‌ی تصویر اعمال شده است. تفاضل بین قاب فعلی و قاب قبلی گرفته شده است و بعد از این مرحله تصویر تحت آستانه‌گذاری قرار می‌گیرد. آستانه‌گذاری توسط روش اتسو انجام می‌پذیرد. اشکال ۲-۵۱ و ۲-۵۲ نتایج حاصل از هر بخش را نشان می‌دهند.

شکل ۲-۵۱ تفاضل بین قاب فعلی و قاب قبلی در مرجع [۲۴]

شکل ۲-۵۲ آستانه گذاری تصویر خروجی در مرجع [۲۴]

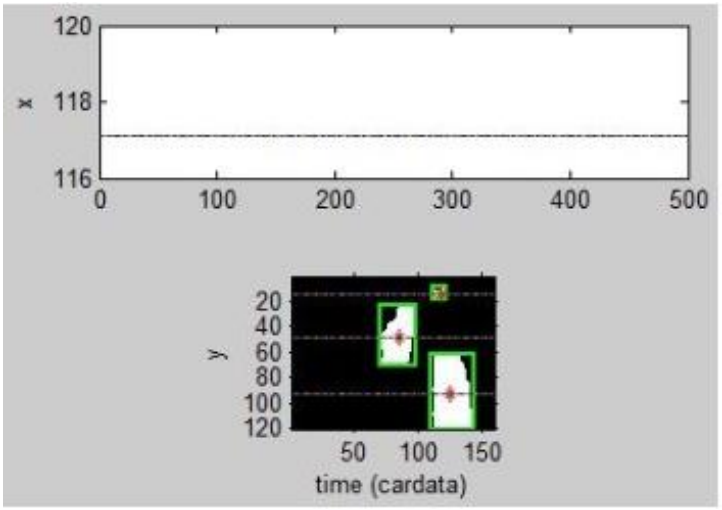
هنگامی که آستانه‌گذاری صورت گرفت، تخمین اولیه‌ای از پیش‌زمینه داریم. حال نوبت به انجام عملیات ریخت‌شناسی می‌رسد تا نویز موجود در تصویر حذف شود. برای این منظور از فیلتر میانه بر روی تصویر آستانه‌گذاری شده استفاده می‌شود. شکل‌های ۲-۵۳ و ۲-۵۴ به ترتیب تصویر فیلتر شده و خودرو تشخیص داده شده و ردیابی شده را نشان می‌دهند.

شکل ۲-۵۳ اعمال فیلتر میانه بر روی تصویر در مرجع [۲۴]



شکل ۲-۵۴ مرکز ثقل و مستطیل محاط شده بر روی خودرو تشخیص داده شده در مرجع [۲۴]

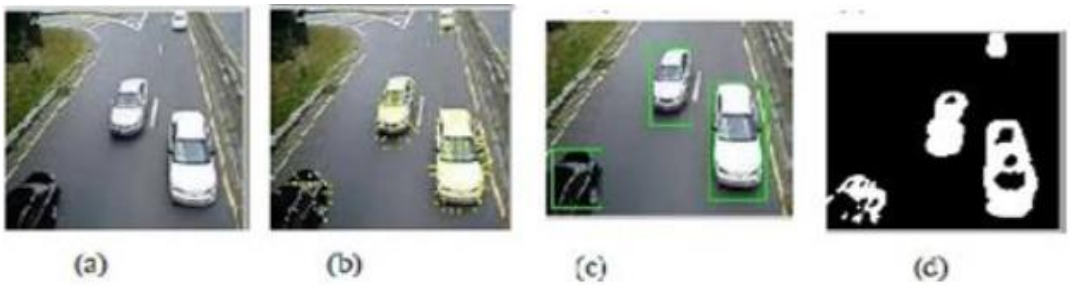
مسیر حرکت خودرو ردیابی شده را می‌توان در شکل ۲-۵۵ مشاهده نمود.



شکل ۲-۵۵ مسیر حرکت خودروهای ردیابی شده در مرجع [۲۴]

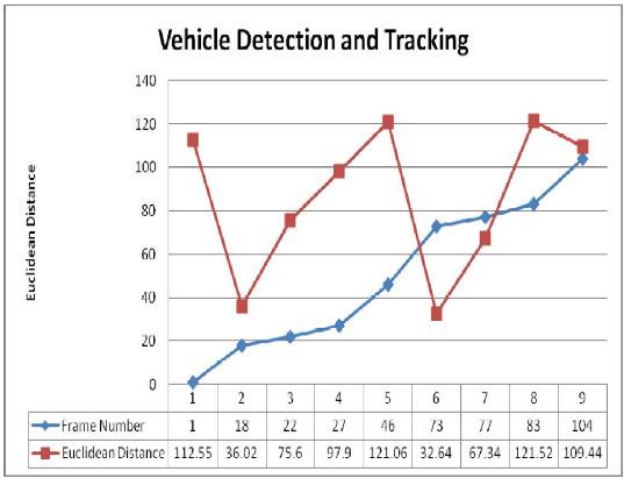
تکنیک بخش بندی برای در یک گروه قرار دادن اشیا مشابه توسط تفاضل پس‌زمینه و به کمک تفاضل قاب صورت می‌گیرد. این روش بهترین تکنیک برای بخش‌بندی اشیای در حال حرکت است.

ردیابی شی در ویدئو توسط اعمال روش شار نوری برای تنظیم بردار حرکتی اشیای در حال حرکت و سپس یافتن آستانه‌ی هر شی، انجام می‌شود. نتایج عملی در شکل ۲-۵۶ نشان داده شده است:



شکل ۲-۵۶ تشخیص و ردیابی شی توسط شار نوری. ***a***) ویدئوی اصلی. ***b***) روش شار نوری برای تنظیم بردار حرکت ***c***) شی، تشخیص و ردیابی اشیایی که از مقدار آستانه بیش‌تر هستند و اشیای در حال حرکت تلقی می‌شوند. ***d***) پیش‌زمینه‌ی شی در حال حرکت شناسایی شده. در مرجع [۲۴]

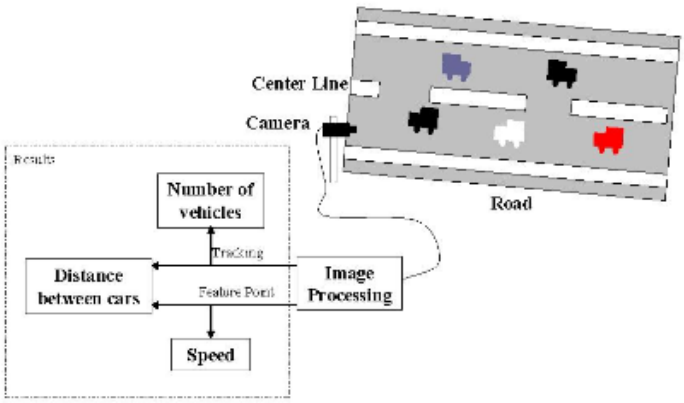
فاصله‌ی اقلیدسی از تفاضل مرکز ثقل محاسبه می‌شود. در نمودار ۲-۶ ردیابی خودرو را در شماره‌ی قاب معین مشاهده می‌کنید.



نمودار ۲-۶ خودرو شناسایی و ردیابی شده با شماره ی قاب مربوطه در مرجع [۲۴]

در مرجع شماره‌ی [۲۵]، یک الگوریتم پیشنهاد داده شده است که نقاط ویژگی خودروها را استخراج و ردیابی می‌کند. این اطلاعات برای اندازه‌گیری سرعت خودروهای در حال حرکت، شمارش تعداد خودروها، اندازه‌گیری فاصله‌ی بین دو خودرو، طبقه‌بندی جهت‌های حرکت و بررسی وضعیت ترافیک، ضروری است. عملگر هریس برای تشخیص لبه‌ها و گوشه‌ها استفاده شده است و سپس نقاط ویژگی استخراج شده توسط تطبیق بلوکی در قاب‌های ویدئویی متوالی ردیابی می‌شوند. خودروهای زیادی می‌تواند در یک زمان به صورت خودکار ردیابی شوند زیرا اطلاعات از دنباله‌های ویدئویی به دست می‌آید. به عنوان یک مثال این مقاله نشان می‌دهد که چگونه می‌توان سرعت خودرو را محاسبه کرد.

در این مقاله، خودروهای در حال حرکت به همراه شکلشان استخراج می‌شوند و ردیابی ناحیه‌ی خودرو تنها توسط سیگنال‌های ویدئویی صورت می‌گیرد. نقاط ویژگی در ناحیه‌ی خودرو توسط عملگر هریس استخراج می‌شود. تشخیص نقاط ویژگی پایدار توسط الگوریتم تطبیق بلوکی تحقق می‌یابد و نتایج ردیابی خوبی را ایجاد می‌کند.

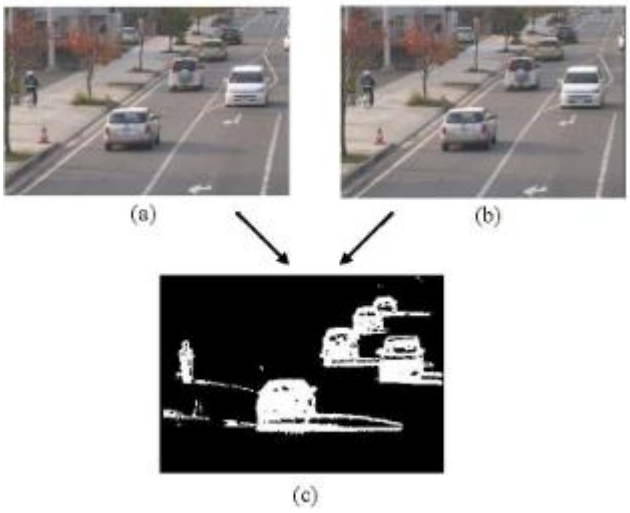


شکل ۲-۵۷ سیستم مراقبتی ویدئویی در مرجع [۲۵]

سیستم اصلی برای نظارت در شکل بالا نشان داده شده است. تصویر خودروهای در حال حرکت توسط یک دوربین ویدئویی ثابت مستقر در یک سمت جاده، گرفته شده است به طوری که دوربین دارای ارتفاع ۴ متر بوده و به نحوی نصب شده است که راستای دید آن تقریبا موازی با جریان ترافیک است. سیگنال ویدئویی توسط الگوریتم پیشنهاد شده مورد پردازش قرار می‌گیرد. به عنوان یک نتیجه، اطلاعات خودروهای در حال حرکت می‌تواند مورد بررسی قرار بگیرد مثلا تعداد خودروها، سرعت حرکت، فاصله‌ی بین دو خودرو و ... . در این مقاله، سرعت خودروهای خروجی هنگام عبور از جلوی دوربین ثابت، اندازه‌گیری می‌شود.

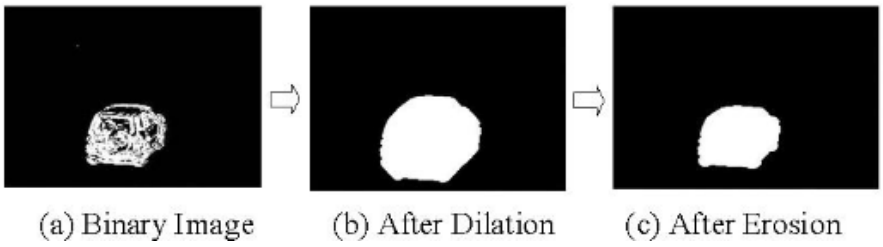
برای استخراج صرف خودروهای متحرک از صحنه‌ی پویا[1]، بخش‌های متغیر که نشان دهنده‌ی اشیای متحرک هستند توسط باینری کردن تفاضل بین دو قاب به ترتیب در زمان t و $t+1$ تشخیص داده می‌شود. در ابتدا، قبل از محاسبه‌ی تفاضل، تصویر رنگی ***RGB*** به تصویر روشنایی (خاکستری) تبدیل می‌شود. تبدیل به شکل زیر صورت می‌گیرد:

$$I(x,y,t) = 0.299R(x,y,t) + 0.587G(x,y,t) + 0.114B(x,y,t)$$ (معادله ۲-۴۲)



شکل ۲-۵۸ تصویر تفاضلی باینری شده (***a***) قاب زمان t ، (***b***) قاب زمان $t+1$ ، (***c***) تصویر تفاضلی (در مرجع ***[25]***)

[1] ***Dynamic***



شکل ۲-۵۹ پردازش‌های گسترش و فرسایش (***a***) تصویر باینری (***b***) بعد از گسترش (***c***) بعد از فرسایش (در مرجع [۲۵])

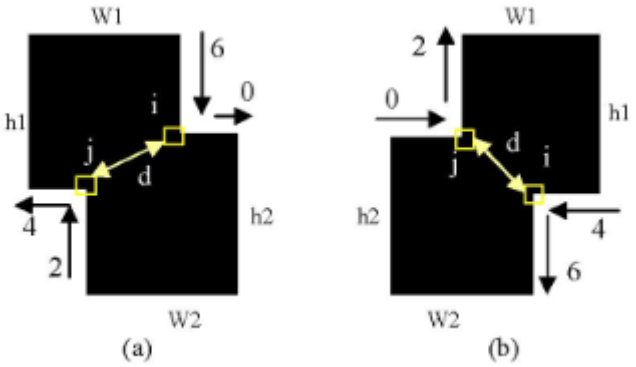
در مرحله‌ی بعدی، تفاضل بین قاب‌های خاکستری شده مطابق با شکل زیر انجام می‌شود.

$$f(x,y,t+n) = |I(x,y,t+n) - I(x,y,t)|$$

مثالی از تفاضل در شکل ۲-۵۸ نشان داده شده است که اشیای در حال حرکت استخراج و توسط نقاط سفیدی مشخص شده‌اند.

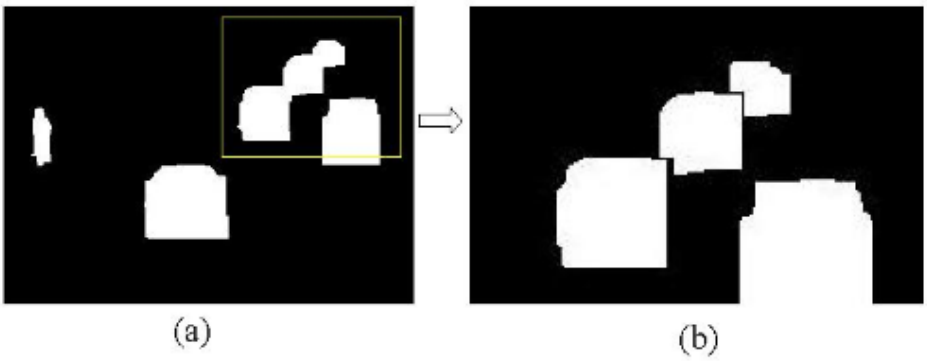
فرض کنید که تنها خودروهای خروجی برای اندازه‌گیری سرعت انتخاب شده‌اند. اشیای متحرک می‌توانند توسط تفاضل بین قاب‌ها استخراج شوند اما اطلاعات مزاحم و یا غیر مرتبط مانند دوچرخه-ها، خودروهای ورودی و سایه‌های خودروها نیز استخراج می‌شوند. استفاده از یک پنجره، به حذف سایر اطلاعات غیرمرتبط کمک می‌کند.

شکل‌دهی به ناحیه‌ی خودرو برای یافتن شکل دقیق کل ناحیه‌ی هر خودرو ضروریست. گسترش و فرسایش، روش‌های معمول برای نرم کردن و یا حذف بخش‌های نویزی و حفره‌های کوچک اطراف یا درون اشیای استخراج شده به صورت تصویر باینری شده هستند.



شکل ۲-۶۰ الگوهای مرسوم خودروهای روی هم افتاده[۱] در مرجع [۲۵]

[۱] ***Overlapped***



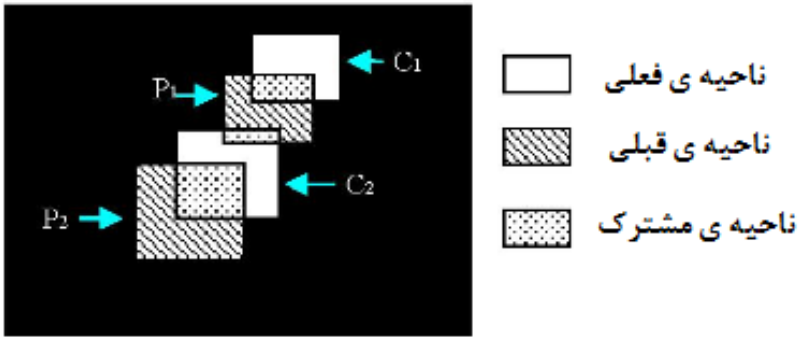
شکل ۲-۶۱جداسازی اشیا. (***a***) ناحیه‌ی تشخیص داده شده‌ی خودرو ، (***b***) بزرگنمایی جزئی تصویر (***a***) (در مرجع [۲۵])

تغییرات مهم بین دو قاب توسط نقاط سفید در شکل ۲-۵۹ (***a***) نشان داده شده است. عمل گسترش به اندازه‌ی ***m*** بار انجام می‌شود تا تمام پیکسل‌های درون ناحیه‌ی خودرو سفید شوند یعنی حفره‌ها در ناحیه حذف یا رنگ شوند. محدوده‌ی اطراف ناحیه توسط کدهای زنجیره‌ای بعد از پردازش گسترش جست‌وجو می‌شود تا ببینیم که آیا ناحیه بسته است یا خیر. بسته بودن موقعی تایید می‌شود که کد زنجیره‌ای به نقطه‌ی شروع جست‌وجو برگردد. سپس حفره‌های درون ناحیه می‌توانند رنگ شوند. بعد از این عملیات فرسایش ***m*** بار اعمال می‌شود تا ناحیه‌ی سفید به سایز اولیه‌ی هر خودرو کاهش یابد. به طور تجربی ***m*** برابر با ۶ انتخاب می‌شود.

اگر دو یا چند خودرو به صورت یک ناحیه ترکیب شده باشند، به دست آوردن اطلاعات درباره‌ی خودروها خیلی دشوار است. مثلا هنگام شمارش تعداد خودروها. تکنیک جداسازی برای محاسبه‌ی تعداد خودروهای در حرکت یکی بعد از دیگری پیشنهاد می‌شود که بر اساس اطلاعات لبه روی خودروها است. در این مقاله، یک تکنیک برای جداسازی نواحی روی هم افتاده توسط کد زنجیره‌ای پیشنهاد شده است. دو الگوی معمول خودروهای روی هم افتاده در شکل ۲-۶۰ برای نمونه نشان داده شده است.

گشتن به دنبال یک تقاطع قائمه مانند ***i*** و ***j*** توسط کد زنجیره‌ای همان طور که در شکل ۲-۶۰ نشان داده شده است، آسان است. اگرفاصله‌ی ***d*** بین ***i*** و ***j*** کم‌تر از حداقل هر کدام از عرض یا ارتفاع ناحیه‌ی خودرو باشد، ناحیه‌ی رو هم افتاده می‌تواند با یک خط بین ***i*** و ***j*** جدا شود. برای جداسازی بهتر، یک حاشیه‌ی مناسب همان طور که در شکل ۲-۶۱ مشاهده می‌نمایید ایجاد شده است.

شکل ۲-۶۲ پنجره و خودروهای استخراج شده (***a***) پنجره برای ترافیک تک‌راه (***b***) خودروهای خروجی (در مرجع [۲۵])



شکل ۲-۶۳ بررسی پیوستگی برای اشیای متحرک در مرجع [۲۵]

یک پنجره بر روی جاده بر روی نمایشگر کامپیوتر فراهم می‌شود که قبل از هر چیزی به عنوان یک نقطه‌ی شروع برای محاسبه‌ی سرعت است. خودروهایی که تنها در موقعیت پنجره واقع می‌شوند به حساب می‌آیند که این باعث راحت‌تر شدن حذف اطلاعات غیر مرتبط و البته به جز سایه‌های مربوط به خودروها، می‌شود.

شکل ۲-۶۲ (***a***) موقعیت پنجره (مشکی) در تصویر را نشان می‌دهد. این نشان می‌دهد که خودروهای خروجی در حال عبور از پنجره به آسانی انتخاب شده است. شی متحرک مانند دوچرخه‌ها و خودروهای ورودی همان طور که در شکل ۲-۶۲ (***b***) مشاهده می‌شود، دور ریخته می‌شوند.

برای ردیابی، یک ناحیه‌ی خودرو در قاب قبلی توسط $P_i(I = 1,2,\ldots)$ و در قاب فعلی توسط $C_j(I = 1,2,\ldots)$ همان طور که در شکل ۲-۶۳ ملاحظه می‌کنید، تعیین شده است. ناحیه‌ی فعلی که C_2 نام دارد دارای دو ناحیه‌ی مشترک بوده که دو خودرو مختلف را در قاب قبلی نشان می‌دهد. اما ناحیه‌ی P_2 از همان ناحیه فرض می‌شود زیرا تعداد بیش‌تری پیکسل مشترک با P_2 وجود دارد. S در معادله‌ی زیر محاسبه می‌شود که نشان دهنده‌ی یک ناحیه‌ی روی هم افتاده مربوط به تعداد پیکسل-

های P_i و C_j هستند. وقتی که S بین P_i و C_j ماکزیمم می‌شود، این دو ناحیه به صورت یک شی تعریف می‌شود. ناحیه‌ی تخمین زده شده همان ناحیه‌ی ردیابی شده است.

$$S_{P\to C} = \frac{A(P_i \cap C_j)}{A(P_i)} \quad \text{(معادله ۲-۴۳)}$$

$$S_{C\to P} = \frac{A(P_i \cap C_j)}{A(C_j)} \quad \text{(معادله ۲-۴۴)}$$

برای تعیین سرعت خودروهای در حال حرکت، مسافت پیموده شده و زمان صرف شده با یک مرجع قرار گرفته در ناحیه‌ی استخراج شده‌ی خودرو، اندازه‌گیری می‌شود. با این حال، خطای اندازه-گیری معمولا اتفاق می‌افتد زیرا شکل و یا اندازه‌ی ناحیه‌ی خودرو تشخیص داده شده ممکن است از یک قاب تا قاب دیگر تغییر کند. استخراج دقیق و پایدار نقطه‌ی ویژگی برای تشخیص دقیق‌تر سرعت خودروها بسیار مهم است. نقطه‌ی ویژگی در این مقاله توسط عملگر هریس استخراج شده است.

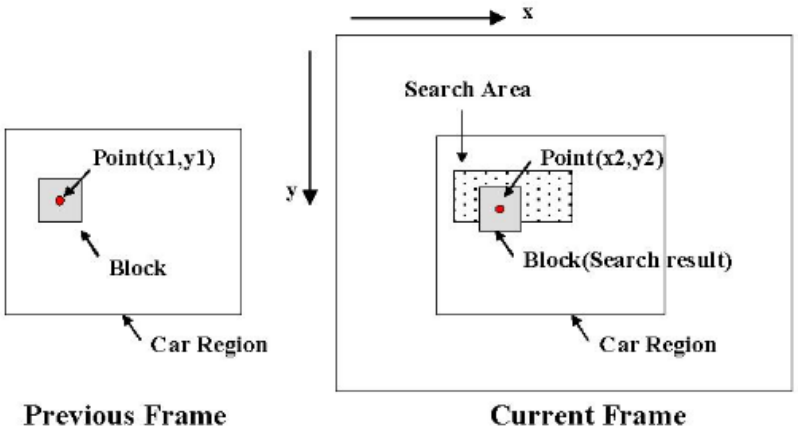
عملگر هریس که در این مقاله استفاده شده است، یک تکنیک برای استخراج نقطه‌ی ویژگی بر اساس همبستگی سیگنال‌های تصویر است به طوری که مقدار خروجی همبستگی برای نقطه‌ی ویژگی موجود در لبه‌ها و گوشه‌ها زیاد می‌شود. عملگر هریس در ادامه تعریف می‌شود:

$$A = \sum_{(x,y)\in N_0} \nabla I(x,y)^T \nabla I(x,y) = \sum_{(x,y)\in N_0} \begin{bmatrix} \frac{\partial I(x,y)}{\partial x} \\ \frac{\partial I(x,y)}{\partial y} \end{bmatrix} \begin{bmatrix} \frac{\partial I(x,y)}{\partial x} & \frac{\partial I(x,y)}{\partial y} \end{bmatrix} \quad \text{(معادله ۲-۴۵)}$$

$$Harris = detA - k(trA)^2 = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 \quad \text{(معادله ۲-۴۶)}$$

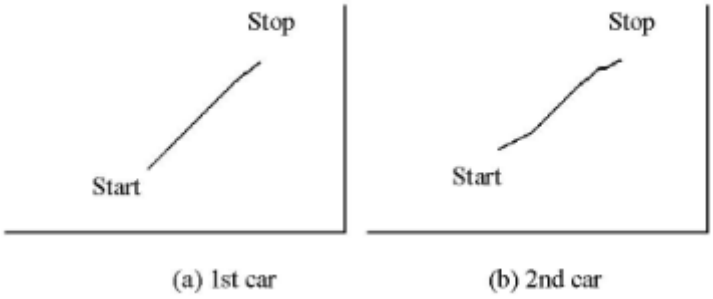
که k یک ثابت است. تنها تعداد مشخصی نقاط ویژگی انتخاب می‌شود و همچنین مقادیر مثبت و ماکزیمم محلی داده شده در معادله‌ی فوق در مرتبه‌ی نزولی مرتب شده‌اند.

نقطه‌ی ویژگی استخراج شده توسط عملگر هریس تنها در ناحیه‌ی خودرو استخراج می‌شود. نقطه‌ی ویژگی برای هر شی گاهی اوقات از یک قاب به قاب دیگر در صحنه‌های پویای واقعی فرق می-کند. برای تشخیص سرعت خودرو با دقت بیش‌تر، نقطه‌ی ویژگی استخراج شده باید پایدار باشد به این معنی که در همان موقعیت شی مربوطه حتی اگر در حرکت باشد، پیدا شود. در این مقاله، نقطه‌ی ویژگی در ناحیه‌ی خودرو توسط روش تطبیق بلوکی همان طور که در شکل ۲-۶۴ مشاهده می‌نمایید، به منظور ردیابی انتخاب می‌شود.



شکل ۲-۶۴ اصول تطبیق بلوکی در مرجع [۲۵]

به عنوان مثال، شکل ۲-۶۵ نتایج ردیابی را برای دو خودرو متفاوت در سیگنال ویدئویی یکسان نشان می‌دهد. اولین خودرو نشان داده شده در شکل ۲-۶۵ (***a***) توسط ردیابی برای ۸۳ قاب و خودرو دوم در شکل ۲-۶۵ (***b***) برای ۷۲ قاب به دست آمده است. این دو مسیر تغییرات موقعیت هر خودرو در جاده را طی قاب‌های به کار رفته برای محاسبه، نشان می‌دهد.



شکل ۲-۶۵ مسیر مربوط به نقاط ویژگی ردیابی شده در مرجع [۲۵]

شکل بالا قسمت (***a***) نشان می‌دهد که مسیر تقریبا یک خط مستقیم است. این حاکی از آن است که اولین خودرو به خوبی ردیابی شده است. قسمت (***b***) شکل بالا نشان می‌دهد که نتیجه لزوما یک خط مستقیم ناست که حاکی از آن است که نتیجه گاهی اوقات اشتباه است.

در این‌جا، دو روش برای تشخیص سرعت خودروها استفاده شده است. در روش اول، سرعت از یک فاصله‌ی مشخص و محاسبه‌ی زمان اتلافی هنگام عبور از آن محاسبه شده است. تعداد پیکسل‌ها در فاصله‌ی مشخص شده از قبل معلوم است و لذا طول معادل بر پیکسل‌ها می‌تواند محاسبه شود. زمان برابر با تعداد قاب‌ها ضرب در ۳۳ میلی ثانیه است. در روش دوم، طول یک بخش قابل اطمینان از مسیر حرکت استفاده می‌شود.

الف) روش ۱: بر اساس طول مسیر حرکت

نقطه‌ی ویژگی در ناحیه‌ی خودرو برای تشخیص سرعت به کار می‌رود و بعد از عبور خودرو از پنجره، تعداد پیکسل‌ها بین نقطه‌ی شروع و نقطه‌ی پایان مسیر از مختصات مسیر شمارش می‌شود. بعد از ضرب ضریب فاصله بر پیکسل در تعداد پیکسل‌های مسیر، سرعت می‌تواند اندازه‌گیری شود زیرا هم مسافت و هم زمان هر دو را می‌دانیم.

نتایج اندازه‌گیری سرعت برای دو خودرو در ادامه آورده شده است. برای خودرو اول، ۸۳ قاب مصرف شده که از قاب ۶۴ ام تا ۱۴۷ ام محاسبه شده است و معادل مسافت ۳۲ متر است. بنابراین سرعت محاسبه شده برابر با ۴۱ کیلومتر بر ساعت است. برای خودرو دوم، ۷۲ قاب مصرف شده که از قاب ۱۲۹ تا ۲۰۱ است. مسافت ۲۵ متر بوده و سرعت محاسبه شده نیز ۳۸ کیلو متر بر ساعت است.

ب) روش ۲) بر اساس بخش قابل اطمینان مسیر

اگر مسیر از نقطه‌ی شروع تا نقطه‌ی پایان، یک خط مستقیم نباشد، نتایج کم و بیش دچار خطا است. بنابراین تنها بخش مستقیم هر مسیر برای محاسبه‌ی سرعت دو خودرو در حال حرکت استفاده می‌شود زیرا صاف بودن مسیر برابر با قابلیت اطمینان بالا است.

نتیجه‌ی محاسبه شده‌ی آزمایش در ادامه آورده شده است. برای خودرو اول، تمام مسیری که تعیین شده مستقیم بوده زیرا یک خط مستقیم را بر مسیر ایجاد کرده و خطایی برابر با یک پیکسل اختلاف را ایجاد کرد. بنابراین نتایج مشابه با روش ۱ است. برای دومین خودرو، ۵۲ قاب با شروع از قاب ۱۴۳ و پایان با قاب ۱۹۳ مصرف شده است. مسافت بخش مستقیم ۱۸ متر است. بنابراین، سرعت محاسبه شده برابر با ۳۹ کیلومتر بر ساعت است.

سرعت خودروهای در حال حرکت برای صحنه‌ی پویای یکسان توسط دو روش اندازه‌گیری شد. برای خودرو اول، هر دو روش تقریبا نتیجه‌ی یکسانی را نشان می‌دهد که در مقایسه با سرعت واقعی خیلی دقیق است. برای دومین خودرو با اعمال روش دوم، یک خطای تشخیص قابل ملاحظه‌ی ۲ کیلومتر بر ساعت (۵ درصد) به ۱ کیلومتر بر ساعت (۲,۵ درصد) کاهش یافت.

روش اول نتایج نسبتا خوبی را می‌دهد در حالی که روش دوم بهبود آشکاری را خصوصا برای دومین خودرو، نشان می‌دهد. این نتیجه نشان می‌دهد که مستقیم بودن مسیر دقت بیش‌تری را ایجاد می‌کند و بیان می‌کند که اندازه‌گیری سرعت بر اساس پردازش سیگنال ویدئویی به طرز قابل ملاحظه‌ای در کاربردهای کنترل ترافیک مفیدند.

فصل سوم: روش پیشنهادی

۳-۱ مقدمه

در این پایان‌نامه، هدف تعیین بی‌درنگ سرعت وسائل نقلیه به کمک روش‌های بینایی ماشین و پردازش تصویر بر روی یک برد توسعه بر مبنای پردازنده‌ی ***ARM*** است. در این‌جا دو روش برای این منظور به کمک کتابخانه‌ی بینایی ماشین ***OpenCV*** و زبان برنامه‌نویسی ***C++*** مورد آزمون قرار گرفته است. این دو روش بر روی برد توسعه‌ی ***Odroid XU4*** بر مبنای پردازنده‌ی ***ARM*** شرکت سامسونگ پیاده‌سازی شده است. بردهای توسعه به دلیل توان مصرفی بسیار کم و قابل حمل بودن، دارای مزیت-های بسیار زیادی نسبت به کامپیوترهای شخصی بر اساس معماری ***X86*** شرکت اینتل هستند. یک سیستم توسعه[۱]، به رایانه‌ای اطلاق می‌شود که دارای عملکردی اختصاصی برای کنترل یک سیستم بزرگ‌تر مکانیکی یا الکتریکی می‌باشد. این رایانه به عنوان بخشی از یک سیستم کامل که شامل سخت افزار و بخش‌های مکانیکی است، درون آن تعبیه شده است. امروزه سیستم‌های توسعه ابزارهای روزمره‌ی زیادی را کنترل می‌کنند. بخش اعظمی از تمام ریزپردازنده‌هایی که به تولید می‌رسد در سیستم‌های یکپارچه به کار می‌رود.

سیستم‌های توسعه دارای هسته‌های پردازشی هستند که می‌تواند میکروکنترلر، پردازنده‌ی ***ARM***، ***DSP*** و مانند آن‌ها باشد. امکاناتی که هر پردازنده در اختیار کاربر قرار می‌دهد شامل انواع ارتباطات، میزان حافظه‌ی قابل پشتیبانی، تعداد هسته‌های پردازشی و فرکانس کاری آن‌ها از جمله مهم‌ترین مواردی هستند که باعث تمایز عملکردی هسته‌های پردازشی نسبت به هم می‌شود.

از ویژگی‌های سیستم‌های توسعه‌ی معمول وقتی با رقبای همه منظوره‌ی خود مقایسه شوند، می توان به مواردی مانند مصرف توان کمتر، اندازه‌ی کوچک‌تر و در نتیجه قابل حمل بودن در مناطق عملیاتی، طیف کاربری قوی‌تر و هزینه‌ی کمتر به ازای هر واحد، اشاره کرد.

به منظور پیاده‌سازی این روش‌ها در ابتدا با توجه به منابع مصرفی برنامه و همچنین برخی از ملاحظات محیطی مانند رطوبت و دما، برد ***Odroid XU4*** انتخاب گردید. در گام بعدی، سیستم عاملی بر روی برد نصب شد. سپس بسته‌های نرم افزاری مورد نیاز از جمله نرم افزار ***QT*** و کتابخانه‌ی متن باز ***OpenCV*** بر روی برد نصب شد. پس از اتمام مراحل ذکر شده، الگوریتم‌های پردازش تصویر مورد نظر بر روی برد کامپایل، ساخته و در نهایت اجرا گردید. در ادامه مشخصات برد ***Odroid XU4*** آورده شده است.

[۱] ***Embedded***

دو روش پیشنهادی به منظور تعیین سرعت خودرو ارائه می‌شود. در ابتدا روشی مبتنی بر استخراج مدل پس‌زمینه و الگوریتم خودهمبستگی[۱] ارائه می‌کنیم. در این روش در ابتدا مدلی از پس-زمینه به کمک الگوریتم ترکیب گوسی[۲] به دست می‌آید. سپس محدوده‌ای از تصویر را به منظور اندازه‌گیری سرعت انتخاب می‌نماییم. به منظور افزایش دقت و رفع اعوجاج غیر خطی تصویر، بر روی این ناحیه تبدیلات افکنشی صورت می‌گیرد. سپس به کمک انجام عملیات همبستگی بین ناحیه‌ی انتخاب شده‌ی متناظر در مدل پس‌زمینه و تصویر اصلی، ورود خودرو به این ناحیه را تشخیص داده و در نهایت بر اساس زمان حضور خودرو در این ناحیه و همچنین میزان مسافت پیموده شده در این ناحیه (طول ناحیه) که مقدار آن در دنیای واقعی مشخص است، سرعت خودرو تعیین می‌شود. در روش دوم، الگوریتم جدیدی با استفاده از پردازش ویدئو، برای تشخیص بی درنگ سرعت خودرو ارائه شده است. به منظور بی درنگ بودن روش پیشنهادی، از الگوریتم‌های پیچیده‌ی ردیابی صرف‌نظر کرده‌ایم. این روش، شامل ۴ بخش کلی است. در بخش ۱، چهار نقطه به صورت دستی از تصویر انتخاب می‌شود و سپس با اعمال تبدیل افکنشی بر روی ناحیه‌ی انتخاب شده تصویر یکسو شده‌ای از این ناحیه به دست می‌آید. چهار نقطه‌ی دیگر نیز به منظور تعیین محدوده‌ی مورد نظر برای تشخیص سرعت به طور دستی بر روی تصویر یکسو شده در نظر گرفته می‌شود. در بخش ۲ به کمک روش تفاضل قاب‌ها پیش‌زمینه ی تصویر تعیین می‌شود. سپس با آستانه‌گذاری مناسب، تصویر اولیه‌ای از پیش زمینه به دست می‌آید. با عملیات ریخت‌شناسی نویزها و اشیای غیر مرتبط در تصویر را حذف نموده و همچنین از تعداد حفره‌های موجود در اشیای پیش زمینه تا حدودی کم می‌نماییم. در ادامه بیرونی‌ترین مرز مربوط به اشیای پیش‌زمینه را به دست آورده و آن‌ها را به طور کامل پر می‌کنیم. بدین ترتیب پیش‌زمینه‌ای شامل چندضلعی‌های توپر که هر یک نمایان‌گر یک خودرو متحرک است، به دست می‌آید. در بخش ۳، مرکز ثقل هر یک از چندضلعی‌ها را در قاب‌های متوالی ویدئو به دست آورده، مسیر حرکت خودرو را رسم نموده و مسافت را می‌یابیم. فاصله‌ی زمانی سپری شده را با شمارش تعداد قاب ها به دست می‌آوریم. در بخش ۴ سرعت محاسبه می‌شود. برای این کار مسافت پیموده شده به واحد کیلومتر و زمان سپری شده نیز به واحد ساعت تبدیل می‌شود. با تقسیم مسافت پیموده شده بر حسب کیلومتر بر زمان سپری شده بر حسب ثانیه، سرعت خودرو تخمین زده می‌شود. در ادامه به معرفی دقیق‌تر هر یک از این روش‌ها و تئوری‌های مربوط به آن‌ها پرداخته و نتایج هر بخش بر روی مجموعه‌ی داده‌ی مرجع [۲۶] مورد آزمون قرار گرفته است.

در پیوست شماره‌ی ۱، مشخصات برد توسعه‌ی ***Odroid XU4*** و نحوه‌ی کار با آن آورده شده است.

---

[۱] ***correlation***

[۲] ***mixture gaussian***

۳-۲ توصیف روش‌های پیشنهادی به همراه موضوعات و تئوری های مربوطه

در این بخش به معرفی هر یک از روش‌های پیشنهادی و تئوری‌های مربوط به آن‌ها می‌پردازیم.

۳-۲-۱ روش اول: مبتنی بر استخراج مدل پس‌زمینه و الگوریتم خودهمبستگی

۳-۲-۱-۱ *بلوک دیاگرام روش اول*

در شکل ۳-۳ بلوک دیاگرام کلی این روش را مشاهده می‌نمایید:

شکل ۳-۱ نمودار کلی مراحل کار در روش اول

۳-۲-۱-۲ *استخراج مدل پس زمینه*

در ابتدا باید مدل پس زمینه استخراج شود. برای این منظور از مدل ترکیب گاوسی وفقی بهبود یافته[1] بهره گرفته می‌شود. از آن‌جایی که صحنه‌ی اصلی با گذشت زمان دچار تغییر می‌شود لذا نیاز است که مدل پس‌زمینه دائما به روزرسانی شود. در شکل‌های ۳-۴ تا ۳-۷، مدل استخراج شده از پس‌زمینه را مشاهده می‌نمایید:

[1] ***Improved adaptive Gausian mixture model***

شکل ۳-۲ تصویر پس زمینه در قاب شماره‌ی ۲۳ - اثر برخی از خودروهای موجود در صحنه‌ی اصلی در بالای پس‌زمینه وجود دارد.

شکل ۳-۳ تصویر پس زمینه در قاب شماره‌ی ۱۷۹ – اثر ناشی از خودروها که در قاب ۲۳ وجود داشت، به دلیل به روزرسانی پس‌زمینه در این-جا ناچیز شده است.

شکل ۳-۴ تصویر پس زمینه در قاب شماره‌ی ۳۸۰ – اثر ناشی از خودروها که در قاب‌های قبلی وجود داشت، به دلیل به روزرسانی پس‌زمینه در این‌جا تقریبا از بین رفته است.

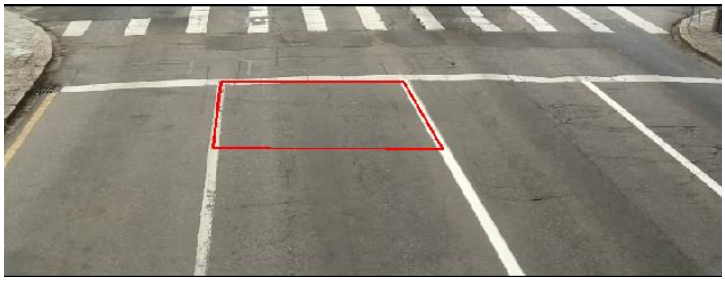
شکل ۳-۵ تصویر پس‌زمینه در قاب شماره‌ی ۱۲۸۴ – از آن‌جایی که خودروها در صحنه‌ی اصلی متوقف شده‌اند و مدل پس‌زمینه نیز دائما در حال به روزرسانی است، لذا تصویر کم‌رنگی از خودروها به پس‌زمینه اضافه می‌گردد. با حرکت خودروها و دورشدن از صحنه‌ی اصلی اثر خودروها رفته رفته از پس‌زمینه حذف خواهد شد.

۳-۲-۱-۳ یکسوسازی

برای انتخاب ناحیه‌ی تشخیص سرعت و به منظور یکسوسازی صحنه‌ی اصلی و پس‌زمینه، ۴ نقطه از تصویر را انتخاب می‌کنیم. چهار نقطه‌ی مرجع طوری در نظر گرفته می شود که در دنیای واقعی یک مستطیل را شکل دهند. تبدیل افکنشی با استفاده از نقاط مرجع بر روی قاب اعمال می-

شود و تصویر یکسو شده‌ای از ناحیه‌ی انتخابی به دست می‌آید. در پیوست شماره‌ی ۲، به مفاهیم و تئوری‌های مربوط به تبدیلات افکنشی[۱] می‌پردازیم.

در شکل‌های ۳-۱۱، ۳-۱۲ و ۳-۱۳ ناحیه‌ی انتخابی از صحنه‌ی اصلی و پس زمینه و تصاویر یکسو شده‌ی این دو ناحیه را مشاهده می‌نمایید:



شکل ۳-۶ ناحیه‌ی انتخابی برای یکسوسازی و تعیین ناحیه‌ی تشخیص سرعت

شکل ۳-۷ ناحیه‌ی انتخابی یکسونشده از الف) صحنه‌ی اصلی و ب) پس‌زمینه

شکل ۳-۸ ناحیه‌ی انتخابی یکسوشده از الف) صحنه‌ی اصلی و ب) پس‌زمینه

[۱] ***Projection Transformation***

۳-۲-۱-۴ محاسبه‌ی همبستگی ناحیه‌ی صحنه‌ی اصلی و پس زمینه

همان‌طور که در بخش یکسوسازی مشاهده می‌شود، تصویر مربوط به صحنه‌ی اصلی و پس‌زمینه به خاکستری تبدیل گشته است. حال باید همبستگی دو تصویر مربوط به صحنه‌ی اصلی و پس‌زمینه را محاسبه نمود. تا زمانی که خودروی در صحنه‌ی اصلی وجود ندارد، صحنه‌ی اصلی بیش‌ترین میزان شباهت را به پس‌زمینه دارد و حاصل همبستگی تصویر صحنه‌ی اصلی و پس‌زمینه نزدیک به ۱ می-شود. با وارد شدن خودرو به صحنه‌ی اصلی، مقدار همبستگی کاهش می‌یابد و تا زمانی که خودرو در صحنه حضور دارد این مقدار با یک فاصله دارد. در این‌جا مقدار آستانه برای همبستگی به منظور تشخیص ورود خودرو به صحنه به صورت آزمون و خطا برابر با مقدار ۰/۸۵ در نظر گرفته شده است بدین مفهوم که اگر مقدار همبستگی صحنه‌ی اصلی و پس‌زمینه از ۰/۸۵ کم‌تر گردید یعنی خودرو وارد صحنه شده است و تا زمانی که این مقدار به بیش‌تر از ۰/۸۵ نرسیده است، خودرو از صحنه خارج نشده است.

همبستگی دو تصویر $f_1(x, y)$ و $f_2(x, y)$ با ابعاد m در n به صورت زیر محاسبه می‌شود:

اگر μ_1 و μ_2 به ترتیب میانگین تصویر f_1 و f_2 باشد و σ_1 و σ_2 نیز به ترتیب مقدار انحراف معیار تصویر f_1 و f_2 باشد، در این صورت داریم:

(معادله ۳-۱)
$$\text{کواریانس: } \sigma_{XY} = E\left[(f_1 - \mu_1)(f_2 - \mu_2)\right] = \frac{(f_1 - \mu_1)(f_2 - \mu_2)}{m \times n}$$

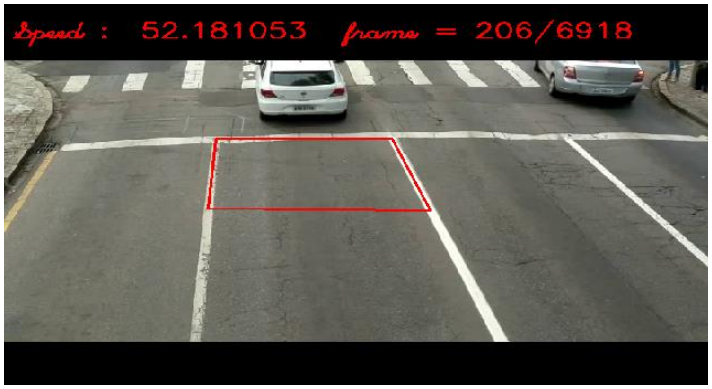
(معادله ۳-۲)
$$\text{همبستگی: } \rho_{xy} = \frac{\sigma_{XY}}{\sigma_X \sigma_Y}$$

مقدار همبستگی همان‌طور که ملاحظه می‌کنید یک کمیت اسکالر بین ۰ تا ۱ است. هر چه مقدار این کمیت به یک نزدیک‌تر باشد میزان شباهت دو تصویر بیش‌تر است.

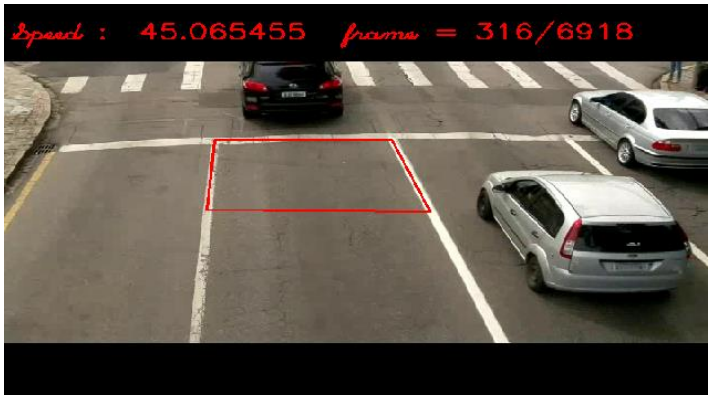
حال نوبت به محاسبه‌ی زمان حضور خودرو در صحنه است. همان‌طور که گفته شد، در این‌جا مقدار آستانه‌ی همبستگی را برابر با ۰,۸۵ انتخاب کردیم. برای به دست آوردن زمان حضور خودرو، کافیست تعداد قاب‌هایی که به ازای آن مقدار همبستگی کم‌تر از ۰,۸۵ است را شمارش نماییم. با داشتن تعداد قاب‌های مصرفی و نرخ ویدئو بر حسب قاب بر ثانیه، می‌توان زمان حضور خودرو در صحنه را بر حسب ثانیه بیان نمود.

۳-۲-۱-۵ محاسبه‌ی سرعت

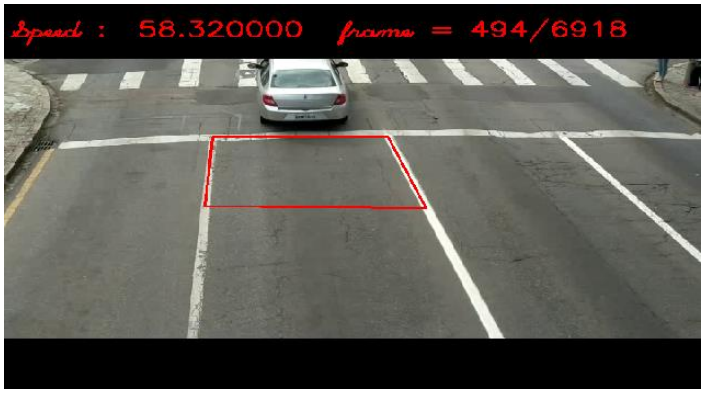
به منظور محاسبه‌ی سرعت به دو کمیت جابه‌جایی و زمان نیاز داریم. زمان حضور خودرو در صحنه را در بخش قبلی به کمک مفهوم همبستگی و با شمارش تعداد قاب‌ها به دست آوریم. برای به دست آوردن بازه‌ی جابه‌جایی هر خودرو باید طول ناحیه‌ی انتخابی را بر حسب واحد کیلومتر بدانیم. برای یافتن سرعت کافیست بازه‌ی جابه‌جایی بر حسب کیلومتر را بر زمان حضور خودرو در صحنه بر حسب ساعت تقسیم نماییم. در شکل‌های ۳-۱۴ تا ۳-۲۰ نتایج مربوط به تشخیص سرعت را در ناحیه‌ی مورد نظر مشاهده می‌نمایید:



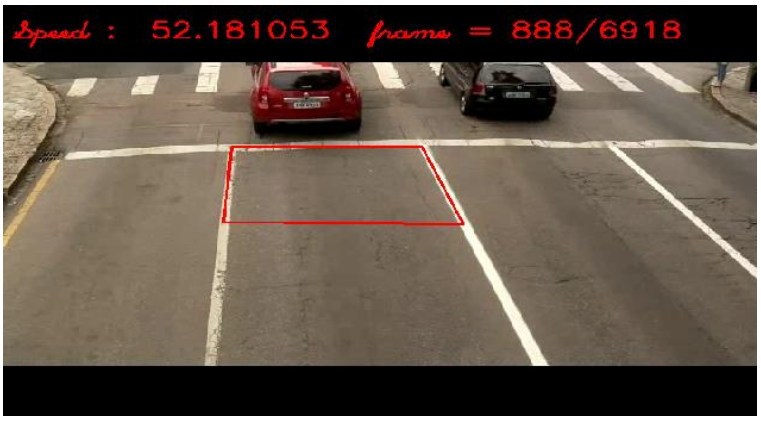
شکل ۳-۹ سرعت خودرو در قاب شماره‌ی ۲۰۶ – سرعت واقعی برابر با ۵۲,۱۳ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۵۲,۱۸ کیلومتر بر ساعت



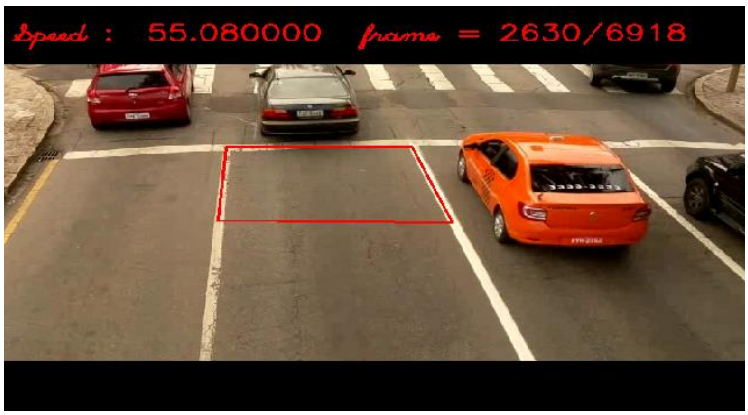
شکل ۳-۱۰ سرعت خودرو در قاب شماره‌ی ۳۱۶ – سرعت واقعی برابر با ۴۸,۸۳ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۴۵,۰۶ کیلومتر بر ساعت



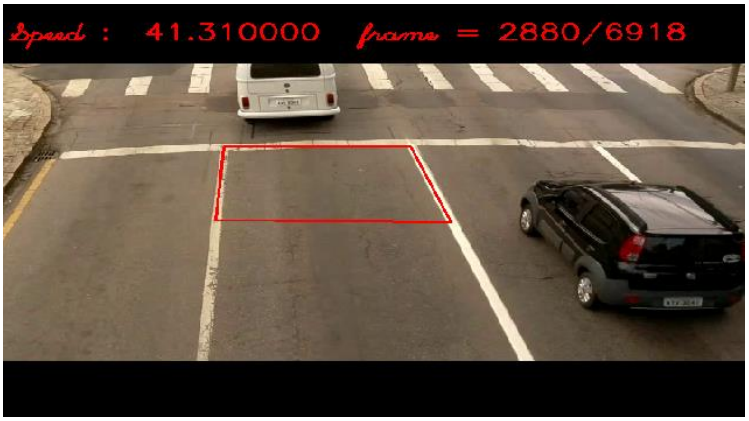
شکل ۳-۱۱ سرعت خودرو در قاب شماره‌ی ۴۹۴ – سرعت واقعی برابر با ۵۸,۳۳ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۵۸,۳۲ کیلومتر بر ساعت



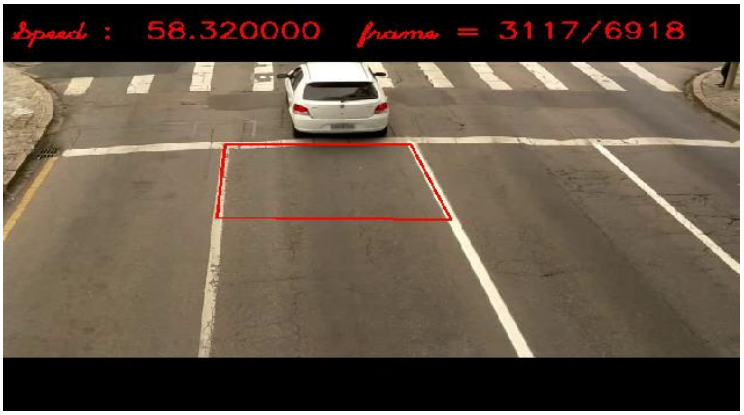
شکل ۳-۱۲ سرعت خودرو در قاب شماره‌ی ۸۸۸ – سرعت واقعی برابر با ۵۰,۸۴ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۵۲,۱۸ کیلومتر بر ساعت



شکل ۳-۱۳ سرعت خودرو در قاب شماره‌ی ۲۶۳۰ – سرعت واقعی برابر با ۵۱,۰۲ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۵۵,۰۸ کیلومتر بر ساعت



شکل ۳-۱۴ سرعت خودرو در قاب شماره‌ی ۲۸۸۰ – سرعت واقعی برابر با ۵۰,۰۳ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۴۱,۳۱ کیلومتر بر ساعت



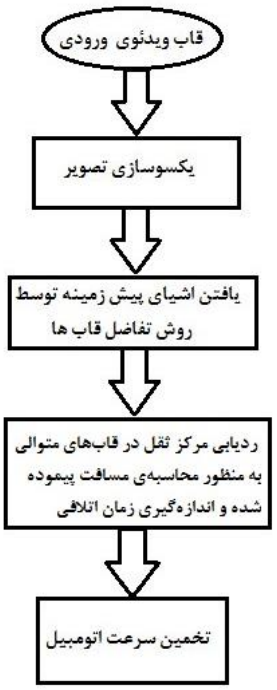
شکل ۳-۱۵ سرعت خودرو در قاب شماره‌ی ۳۱۱۷ – سرعت واقعی برابر با ۵۲,۹۷ کیلومتر بر ساعت – سرعت تخمین زده شده برابر با ۵۸,۳۲ کیلومتر بر ساعت

روش اول نسبت به لرزهای جرئی دوربین مقاوم است اما سایه‌ی مربوط به سار خودروها می‌تواند باعث ایجاد خطا در سرعت تخمین زده شده شود. همچنین ابعاد خودرو هر چه بیش‌تر باشد، زمان حضور بیش‌تر اعلام شده و سرعت به درستی اندازه‌گیری نمی‌شود. به منظور حذف اثر ابعاد خودرو در اندازه‌گیری سرعت، روش دوم ارائه شده است.

۳-۲-۲ روش دوم: مبتنی بر تفاضل قاب‌های متوالی و دنبال کردن مرکز ثقل

۳-۲-۲-۱ *بلوک دیاگرام روش دوم*

در شکل ۳-۲۱، نمودار کلی مربوط به مراحل کار در روش دوم، نمایش داده شده است:



شکل ۳-۱۶ نمودار کلی مراحل کار در روش دوم

۳-۲-۲-۲ بخش ۱: یکسوسازی تصویر

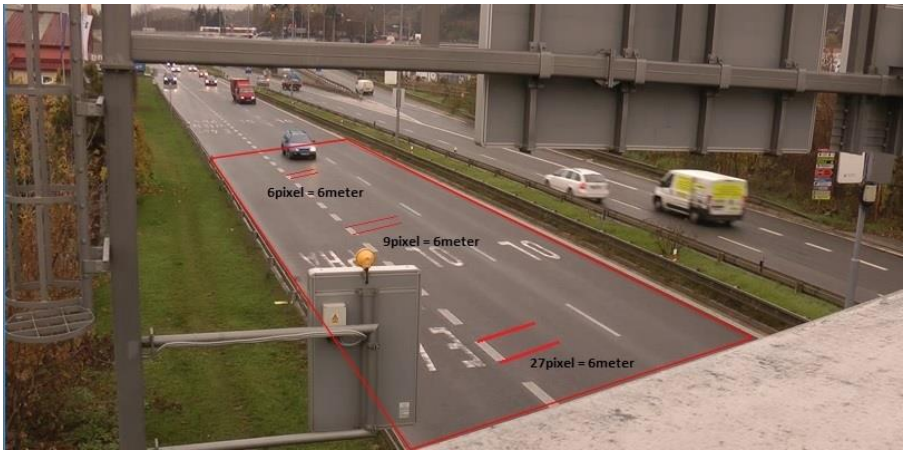
در ابتدا دو قاب متوالی از ویدئو خوانده می‌شود. بر روی قاب شماره‌ی ۱، چهار نقطه‌ی مرجع طوری در نظر گرفته می شود که در دنیای واقعی یک مستطیل را شکل دهند. مثلا نقاط ابتدایی و انتهایی خطوط راه‌راه وسط جاده که با هم موازیند، انتخاب مناسبی می‌تواند باشد. تبدیل افکنشی با استفاده از نقاط مرجع بر روی هر دو قاب اعمال می‌شود و تصاویر یکسو شده‌ای از ناحیه‌ی انتخابی به دست می‌آید. پس از یکسوسازی تصویر، به کمک ۴ نقطه‌ی مرجع، محدوده‌ای بر روی تصویر یکسوشده به منظور تشخیص سرعت مشخص می‌شود.

علت یکسوسازی تصویر در این‌جا، رفع اعوجاج غیرخطی صحنه است. همان‌طور که در شکل ۳-۲۲ مشاهده می‌نمایید، در تصویر یکسو نشده‌ی ۳-۲۲، اندازه‌های دنیای واقعی با پیکسل‌های صحنه رابطه‌ی غیرخطی دارند و لذا نمی‌توان با شمارش پیکسل‌ها اندازه‌های واقعی را اعلام کرد لذا نیاز است که تصویر را یکسو کرده تا بتوان رابطه‌ای خطی را بین پیکسل‌های تصویر و فواصل دنیای واقعی ایجاد کرد.

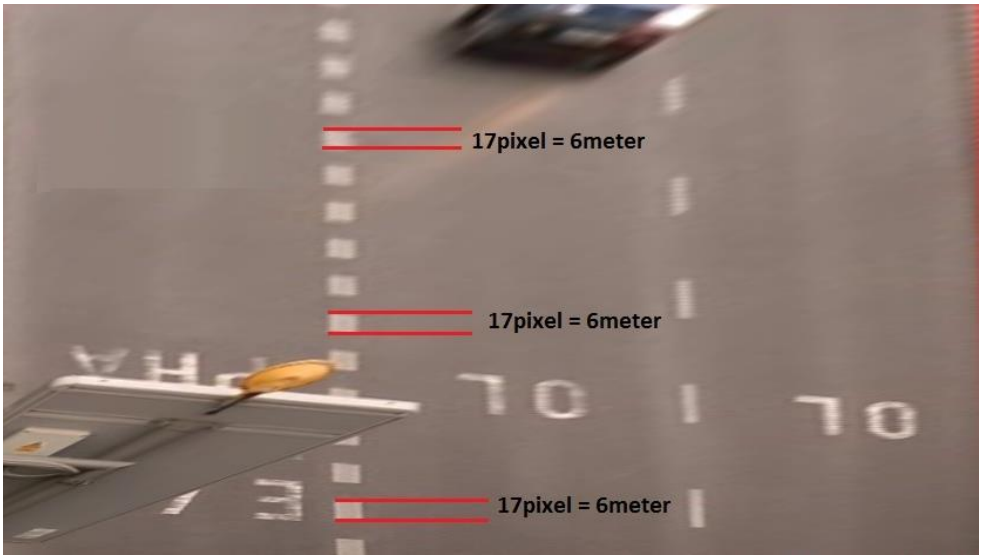
شکل ۳-۱۷ تصویر پرسپکتیو شده

شکل ۳-۱۸ تصویر یکسو شده

همان‌طور که مشاهده می‌کنید، در تصویر پرسپکتیو شده (تصویر اصلی)، اشیای صحنه (شامل خطوط جاده و خودروها) با فاصله گرفتن از دوربین کوچک و کوچک‌تر می‌شوند در حالی که در تصویر یکسو شده خطوط جاده و خودروها دارای اندازه‌ی یکسانی هستند و لذا از تصاویر یکسو شده می‌توان رابطه‌ی خطی‌ای بین تعداد پیکسل‌ها و فواصل واقعی به دست آورد. این موضوع را در مورد خطوط جاده در شکل ۳-۲۵ مشاهده می‌نمایید:

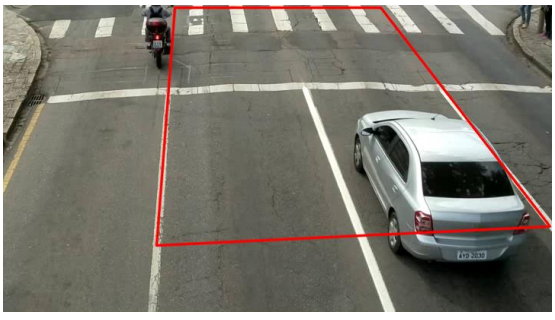


شکل ۳-۱۹ رابطه‌ی غیر خطی بین پیکسل‌های تصویر و اندازه‌های دنیای واقعی قبل از رفع اعوجاج



شکل ۳-۲۰ رابطه‌ی خطی بین پیکسل‌های تصویر و اندازه‌های دنیای واقعی پس از رفع اعوجاج

در شکل‌های ۳-۲۵ و ۳-۲۶ به ترتیب نتایج مربوط به انتخاب نقاط مرجع و یکسوسازی تصویر نمایش داده شده است.

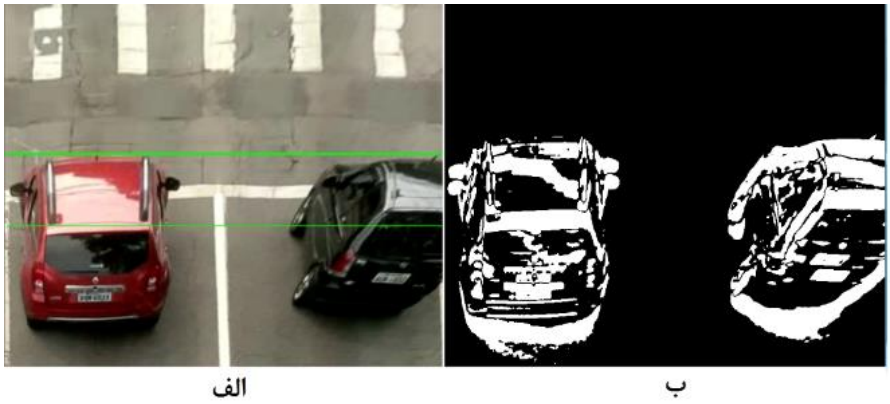


شکل ۳-۲۱ ناحیه‌ی انتخابی توسط ۴ نقطه‌ی مرجع

شکل ۳-۲۲ تصویر یکسوشده به همراه محدوده‌ی تعیین شده برای اندازه‌گیری سرعت

۳-۲-۲-۳ بخش ۲ : یافتن اشیای پیش‌زمینه توسط روش تفاضل قاب‌ها

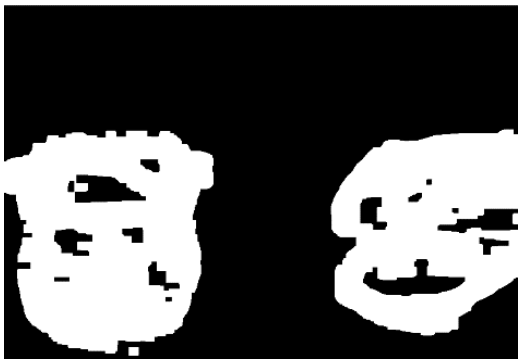
پس از یکسوسازی تصویر و تعیین ناحیه‌ی اندازه‌گیری سرعت بر روی تصویر، نوبت به یافتن اشیای پیش‌زمینه[۱] (خودروهای متحرک) است. از آن جایی که بی درنگ بودن الگوریتم دارای اهمیت است، لذا از روش تفاضل دو قاب به جای روش‌های پیچیده‌ی تشخیص پیش‌زمینه استفاده شده است. قبل از انجام عملیات تفاضل‌گیری، فیلتر گاوسی با اندازه‌ی ۵ در ۵ را بر روی قاب قبلی و فعلی اعمال می‌کنیم. علت این موضوع حذف تغییرات شدید در تصویر است. برای یافتن پیش‌زمینه، به کمک قدر مطلق تفاضل دو قاب متوالی یکسو شده در بخش ۱ و آستانه‌گذاری مناسب، تصویر اولیه‌ای از پیش-زمینه به دست می‌آید. در واقع در این روش قاب قبلی به عنوان پس‌زمینه برای قاب فعلی در نظر گرفته می‌شود که با تفاضل آن‌ها از یکدیگر و آستانه‌گذاری مناسب، شی متحرک به دست می‌آید. در شکل ۳-۲۸ نتیجه‌ی مربوط به این عملیات نمایش داده شده است.



شکل ۳-۲۳ روش تفاضل قاب‌ها برای تخمین پیش‌زمینه. الف) تصویر اصلی ب) تصویر پیش‌زمینه

[۱] ***Foreground***

بعد از یافتن تصویر اولیه‌ای از پیش‌زمینه، به کمک عملیات ریخت‌شناسی[۱] گسترش[۲] و فرسایش[۳]، نویزها و اشیای غیر مرتبط احتمالی حذف شده و همچنین بخش زیادی از حفره‌های تصویر پوشانده می‌شود. این عملیات تا حدی که به تصویر قابل قبولی از پیش‌زمینه برسیم، ادامه می‌یابد. در این‌جا از دو عملیات گسترش و یک عملیات فرسایش به صورت پشت سر هم، با اندازه‌ی ۷ در ۷ و با تکرار دو بار، استفاده شده است. در شکل ۳-۲۹ نتیجه‌ی حاصل از ریخت‌شناسی آورده شده است.



شکل ۳-۲۴ نتیجه انجام عملیات ریخت‌شناسی بر روی تصویر

سپس بیرونی‌ترین مرز مربوط به اشیای پیش‌زمینه تعیین شده، چند ضلعی محاط بر آن را به دست آورده و درون آن را پر می‌نماییم. نتیجه‌ی در شکل ۳-۳۰ آورده شده است.

شکل ۳-۲۵ نتیجه حاصل از پرکردن بیرونی‌ترین مرز دو شی موجود در تصویر

[۱] ***Morphology***

[۲] ***dilation***

[۳] ***Erosion***

۳-۲-۲-۴ بخش ۳ : محاسبه‌ی مرکز ثقل، مسافت و زمان سپری شده

پس از شناسایی خودروهای متحرک، نوبت به محاسبه‌ی مرکز ثقل هر خودرو به منظور ردیابی خودرو در قاب‌های متوالی و یافتن مسیر حرکت آن است. از روی مسیر حرکت خودرو، مسافت پیموده شده توسط خودرو، به دست می‌آید. روش‌های معمول ردیابی بر اساس نقاط ویژگی دارای پیچیدگی‌های محاسباتی بوده و بسیار زمان‌بر هستند. از آن‌جایی که تصویر یکسو شده است و همچنین محدوده‌ی اندازه‌گیری سرعت محدود است، لذا با نزدیک شدن خودرو به دوربین یا با دور شدن از آن، شکل خودرو شناسایی شده دچار تغییر بسیار کمی می‌شود و در نتیجه مرکز ثقل آن نقطه‌ای ثابت خواهد بود بنابراین در این بخش، به منظور ردیابی بی درنگ خودرو در قاب‌های متوالی، می‌توان در هر قاب مرکز ثقل خودروها را به دست آورده و مسیر حرکت خودرو را به دست آورد. برای به دست آوردن مرکز ثقل هر خودرو، از مفهوم گشتاور[۱] استفاده می‌کنیم و مرکز ثقل هر چندضلعی را در قاب‌های متوالی تصویر به دست می‌آوریم. در ادامه به تئوری مربوط به محاسبه‌ی مرکز ثقل از طریق مفهوم گشتاور می‌پردازیم.

گشتاور میانگین وزن‌دار روشنایی پیکسل‌های تصویر می‌باشد. یکی از مشخصات ساده‌ی تصویر یا ناحیه‌ای از تصویر که توسط گشتاور قابل محاسبه است، مرکز ثقل و اطلاعات مربوط به دوران است. برای یک تابع دو بعدی $f(x, y)$، گشتاور از مرتبه‌ی $p+q$ به صورت زیر تعریف می‌شود:

$$M_{pq} = \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} x^p y^q f(x,y) d_x d_y \quad \text{(معادله ۳-۳)}$$

برای یک تصویر خاکستری با مقادیر روشنایی $I(x, y)$ مقدار گشتاور به صورت زیر به دست می‌آید:

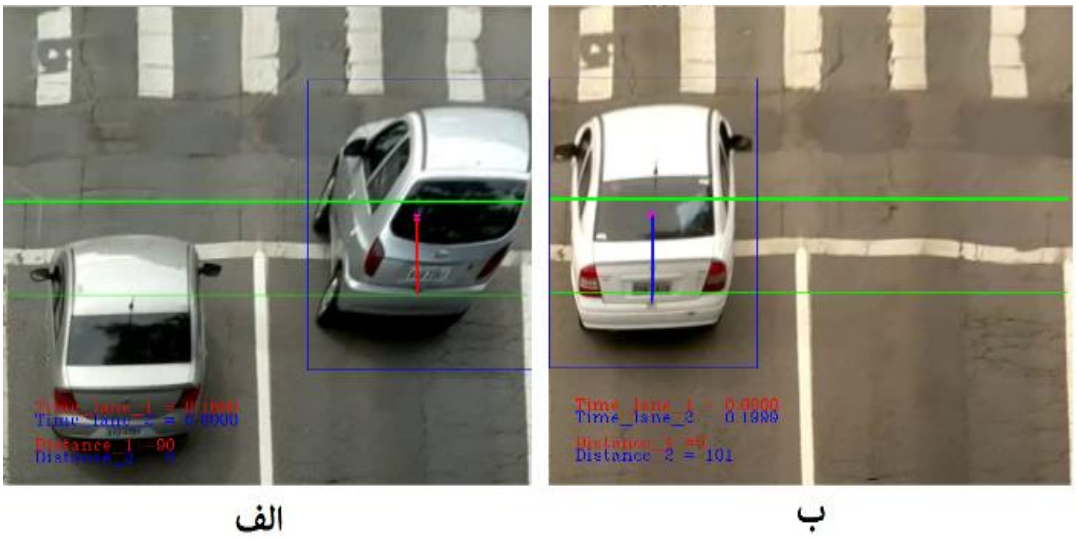
$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y) \quad \text{(معادله ۳-۴)}$$

به کمک مفهوم گشتاور می‌توان مشخصاتی از تصویر مانند مساحت، مرکز ثقل و ... را به دست آورد.

- مساحت (برای تصاویر باینری) یا مجموع سطح خاکستری (برای تصاویر خاکستری) : M_{00}
- مرکز ثقل: $\{\bar{x}, \bar{y}\} = \left\{\frac{M_{10}}{M_{00}}, \frac{M_{01}}{M_{00}}\right\}$

[۱] *moment*

همان‌طور که اشاره گردید، مرکز ثقل هر خودرو را در قاب‌های متوالی تصویر به دست می‌آوریم. با تکرار این عمل، مسیر حرکت خودرو از روی جابه‌جایی مرکز ثقل رسم می‌شود. بازه‌ی جابه‌جایی خودرو از طریق مجموع فواصل اقلیدسی مرکز ثقل‌های متوالی در هر قاب بر حسب واحد پیکسل، به دست می‌آید. همچنین با شمارش تعداد قاب‌های مصرفی در این مسیر و تقسیم تعداد قاب‌ها بر نرخ ویدئو بر حسب قاب بر ثانیه، فاصله‌ی زمانی مسیر طی شده بر حسب ثانیه، محاسبه می‌شود. در شکل ۳-۳۱ مسیر حرکت خودرو و مسافت پیموده شده به همراه مدت زمان اتلافی آن به دست آمده است.



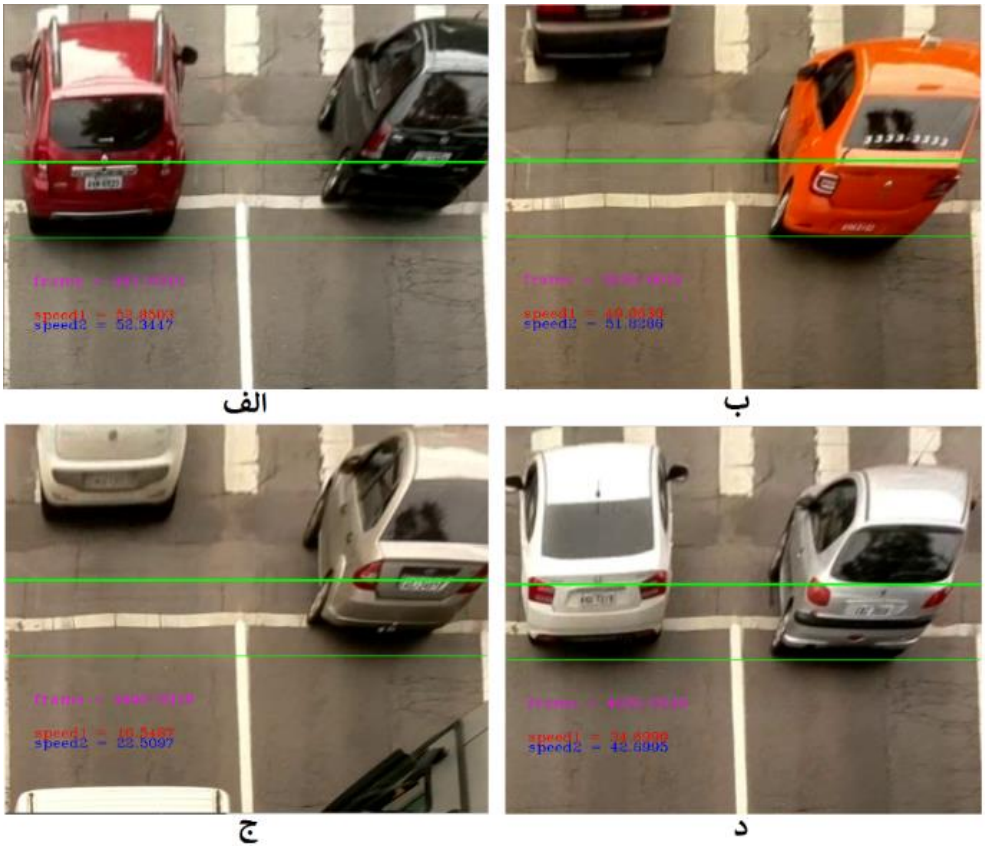
الف ب

شکل ۳-۲۶ نتیجه‌ی حاصل از به دست آوردن مرکز ثقل، رسم مسیر حرکت خودرو، محاسبه‌ی مسافت پیموده شده بر حسب تعداد پیکسل-ها و زمان سپری شده بر حسب ثانیه. الف)نتایج مربوط به خط سمت راست جاده (زمان اتلافی: ۰,۱۶ ثانیه و مسافت ۹۰ پیکسل). ب) نتایج مربوط به خط سمت چپ جاده (زمان اتلافی: ۰,۱۹ ثانیه و مسافت ۱۰۱ پیکسل).

۳-۲-۲-۵ بخش ۴: تخمین سرعت خودرو

بعد از یافتن بازه‌ی جابه‌جایی هر خودرو و مدت زمان آن، به محاسبه‌ی سرعت می‌پردازیم. مسافت پیموده شده بر حسب پیکسل است و لذا باید مسافت پیموده شده برحسب کیلومتر بیان شود. برای این کار باید فاصله‌ی دو نقطه‌ی دلخواه از تصویر واقعی را بر حسب کیلومتر بدانیم. با شمارش تعداد پیکسل‌های بین آن دو نقطه، ضریب تبدیلی به دست می‌آید که رابطه‌ای خطی بین تعداد پیکسل‌ها و فاصله بر حسب کیلومتر بیان می‌کند. با ضرب کردن مسافت پیموده شده بر حسب پیکسل در این ضریب تبدیل، مسافت پیموده شده بر حسب کیلومتر به دست می‌آید. برای محاسبه‌ی سرعت کافی است مسافت پیموده شده بر حسب کیلومتر را بر زمان اتلافی بر حسب ساعت تقسیم کنیم. بدین ترتیب سرعت خودرو تخمین زده می‌شود.

در شکل ۳-۳۲، نمونه‌ای از سرعت‌های تخمین زده شده‌ی چند خودرو را مشاهده می‌نمایید:

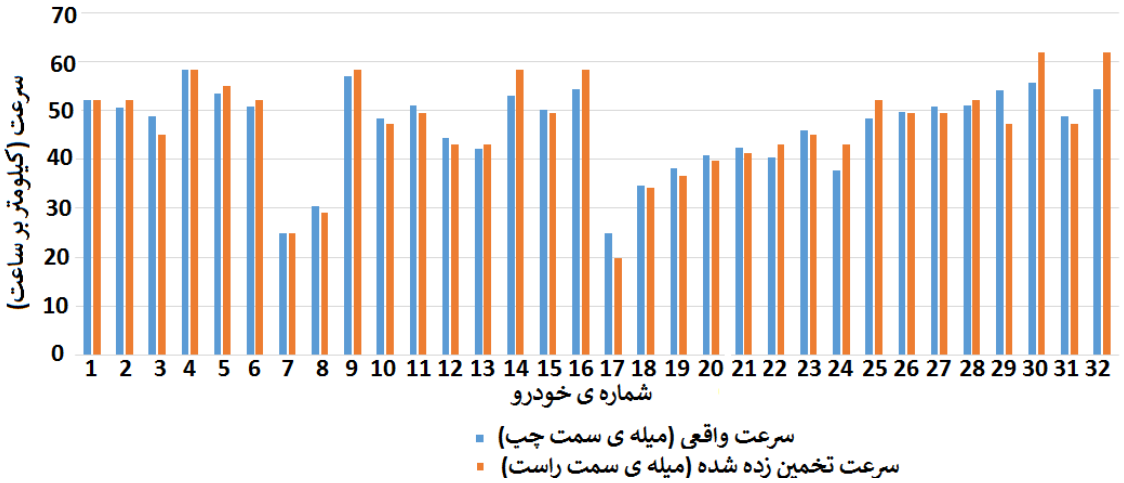


شکل ۳-۲۷ نتیجه‌ی تشخیص سرعت خودروها در خطوط ۱ (سمت چپ) و ۲ (سمت راست) جاده. الف) سرعت محاسبه شده‌ی خودرو در خطوط ۱و ۲ جاده در قاب شماره‌ی ۸۸۵ (به ترتیب ۵۲/۸۵ و ۵۲/۳۴). ب) سرعت محاسبه شده‌ی خودرو در خطوط ۱و ۲ جاده در قاب شماره-ی ۲۶۳۹ (به ترتیب ۴۹ و ۵۱/۸۳). ج) سرعت محاسبه شده‌ی خودرو در خطوط ۱و ۲ جاده ر قاب شماره‌ی ۴۶۹۲ (به ترتیب ۱۶/۵۱ و ۲۲/۵). د) سرعت محاسبه شده‌ی خودرو در خطوط ۱و ۲ جاده در قاب شماره‌ی ۴۹۳۰ (به ترتیب ۳۴/۶۹ و ۴۲/۶).

فصل چهارم: نتایج عملی و مقایسه با کارهای قبلی

۴-۱ نتایج حاصل از روش اول: مبتنی بر استخراج مدل پس‌زمینه و الگوریتم خودهمبستگی

با آزمایش روش اول بر روی چندین ویدئو در شرایط مختلف، معین شد که خطای میانگین تخمین سرعت خودرو برابر با ۴/۹۹ درصد و انحراف معیار استاندارد خطا نیز ۴/۸۳ است. همچنین سرعت به‌طور میانگین با ۲/۲۶± کیلومتر بر ساعت اختلاف برای هر خودرو گزارش می‌شود. در نمودار ۴-۱، نمودارهای مربوط به سرعت واقعی و سرعت تخمین زده شده به تصویر درآمده است:



نمودار ۴-۱ مقایسه‌ی سرعت تخمین زده شده با سرعت اصلی در هر خودرو در روش اول

در جدول ۴-۱ سرعت تخمین زده شده و سرعت اصلی مربوط به چند خودرو نوشته شده است:

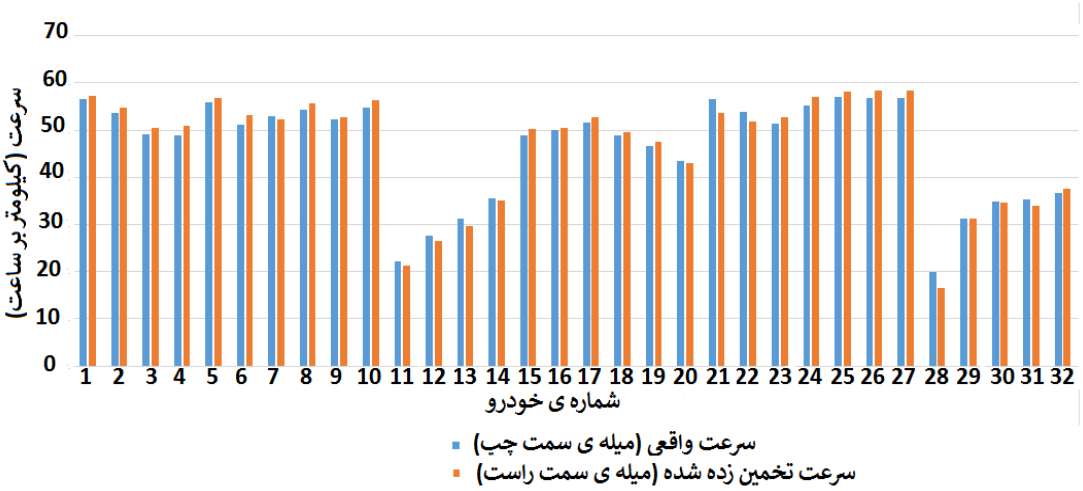
جدول ۴-۱ مقایسه‌ی سرعت تخمین زده شده و سرعت اصلی چند خودرو در روش اول

درصد خطا	سرعت تخمین زده شده	سرعت اصلی خودرو
۲/۱۸۶	۵۸/۳۱۷	۵۷/۰۷
۲/۸۶	۵۵/۰۸	۵۳/۵۵
۳/۰۰۹	۴۹/۵۷۲	۵۱/۱۱
۱۴/۰۵۳	۶۱/۹۶۵	۵۴/۳۳
۱۹/۹۸	۱۹/۸۳	۲۴/۷۸
۱/۵۹۴	۳۴/۱۸۶	۳۴/۷۴
۲/۷۷	۴۳/۱۰۳	۴۲/۳۳
۲/۱۸۶	۵۸/۳۲	۵۷/۰۷
۷/۶۹۷	۵۲/۱۷۹	۴۸/۴۵

به منظور پیاده سازی این روش، برنامه‌ای به زبان ***C++*** و به کمک کتابخانه‌ی ***OpenCV*** توسعه داده شده است. برنامه بر روی لپ‌تاپ لنوو مدل ***z510*** با پردازنده‌ی ***Intel core i5*** مدل ***4200M*** با فرکانس کاری ۲٫۵ گیگاهرتز و رم ۶ گیگابایتی و پردازنده‌ی گرافیکی ***Nvidia Geforce GT 740M*** با حافظه‌ی ***2GB***، اجرا شد. زمان اجرای برنامه برای هر قاب ویدئویی با اندازه ی ***720 x 576*** و برای ویدئو با نرخ قاب ۲۵ قاب بر ثانیه، به طور متوسط ۵۵ میلی ثانیه اندازه‌گیری شد. همچنین برنامه بر روی برد توسعه‌ی ***Odroid XU4*** با پردازنده‌ی ۸ هسته ای ۶۴ بیتی با فرکانس کاری ۲ گیگاهرتز و رم ۲ گیگابایت نیز آزمایش گردید. زمان اجرای برنامه با همان شرایط فوق بر روی این برد ۱۰۵ میلی ثانیه به طول انجامید.

۴–۲ نتایج حاصل از روش دوم: مبتنی بر تفاضل قاب‌های متوالی و دنبال کردن مرکز ثقل

با آزمایش روش دوم بر روی چندین ویدئو در شرایط مختلف، مشخص شد که خطای میانگین تخمین سرعت خودرو برابر با ۳/۳۱ درصد و انحراف معیار استاندارد خطا نیز ۳/۲۸ است. همچنین سرعت به‌طور میانگین با ۱/۳۹± کیلومتر بر ساعت اختلاف برای هر خودرو گزارش می‌شود. در نمودار ۴-۲، نمودارهای مربوط به سرعت واقعی و سرعت تخمین زده شده به تصویر درآمده است:



نمودار ۴-۲ مقایسه‌ی سرعت تخمین زده شده با سرعت اصلی در هر خودرو

در جدول ۴-۲ سرعت تخمین زده شده و سرعت اصلی مربوط به چند خودرو نوشته شده است:

جدول ۴-۲ مقایسه‌ی سرعت تخمین زده شده و سرعت اصلی چند خودرو

درصد خطا	سرعت تخمین زده شده	سرعت اصلی خودرو
۲/۳۴	۵۷,۹۷	۵۶,۶۵
۰/۸۳	۵۴,۱۸	۵۳,۷۴
۱/۳۱	۵۰,۴۵	۵۱,۱۲
۰/۰۹	۵۴,۲۷	۵۴,۲۳
۳/۰۲	۲۱,۴۶	۲۲,۱۳
۷/۱۵	۳۳,۰۲	۳۵,۵۶
۰/۷۴	۴۲,۷	۴۲,۳۹
۱/۵۸	۵۶,۱۷	۵۷,۰۷
۳/۵۶	۵۰,۶۳	۴۸,۸۹

این روش نیز، به زبان ***C++*** و به کمک کتابخانه‌ی ***OpenCV*** بر روی برد توسعه‌ی ***XU4*** و لپ تاپ لنوو ***Z510*** پیاده سازی شده است. زمان اجرای برنامه بر روی لپ‌تاپ لنوو برای هر قاب ویدئویی با اندازه‌ی ***720 x 576*** و با نرخ ۲۵ قاب بر ثانیه، به طور متوسط ۳۱ میلی ثانیه اندازه‌گیری شد. همچنین زمان اجرای برنامه بر روی برد توسعه‌ی ***Odroid XU4*** با همان شرایط فوق ۴۸ میلی ثانیه گردید که برای ویدئو با نرخ ۲۰ قاب بر ثانیه، به صورت بی‌درنگ است.

۴-۳ مقایسه‌ی دو روش پیشنهادی

در این بخش مقایسه‌ای بین دو روش بیان شده خواهیم داشت. نتایج این مقایسه در جدول ۴-۲ مشاهده می‌شود:

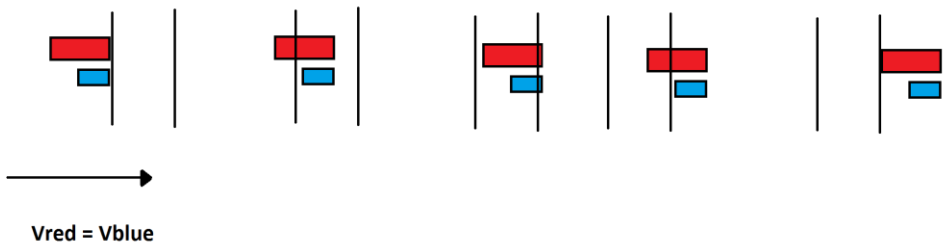
همان‌طور که مشاهده گردید روش اول نسبت به روش دوم دارای دقت پایین‌تر و زمان پردازش بیش‌تری است.

مزیت روش اول در این است که تا حدودی نسبت به لرزش‌های جزئی دوربین مقاوم است.

در روش اول که از الگوریتم همبستگی استفاده می‌شود، بزرگترین ایرادی که وجود دارد، تاثیر ابعاد خودرو بر الگوریتم است که در این‌جا در نظر گرفته نشده است. مسافت پیموده شده توسط تمام

خودروها یکسان است (محدوده‌ی تعیین شده) ولی زمان حضور هر خودرو در صحنه باید اندازه‌گیری شود. برای خودروهای کوچک زمان حضور در صحنه با دقت خوبی تخمین زده می‌شود ولی برای خودروهای بزرگتر مانند ون‌ها، اتوبوس‌ها، کامیون‌ها و ... این زمان حضور به دلیل ابعاد زیاد آن‌ها به درستی تخمین زده نمی‌شود و سرعت اعلام شده دقیق نیست. همچنین اگر خودرو در صحنه متوقف شود زمان حضور آن زیاد می‌شود و سرعت آن نزدیک به صفر اعلام می‌شود هرچند دوباره حرکت کند و از صحنه بیرون رود. همچنین حضور سایه‌ی سایر خودروها در صحنه می‌تواند باعث بروز خطا در اندازه‌گیری زمان حضور و لذا تخمین سرعت شود. از آن‌جایی که در هر قاب عمل همبستگی بین پس‌زمینه و صحنه‌ی اصلی صورت می‌گیرد، لذا زمان پردازش این الگوریتم کمی زیاد است. همچنین ایراد دیگری که در روش اول وجود دارد این است که در یک زمان فقط می‌توان سرعت یک خودرو را گزارش داد.

در شکل ۴-۱ موضوع مربوط به ابعاد خودرو و محدوده‌ی تعیین سرعت در روش اول را مشاهده می‌کنید:



شکل ۴-۱ تاثیر ابعاد خودرو بر سرعت در روش پیشنهادی اول

همان طور که مشاهده می‌کنید، سرعت هر دو خودرو برابر است و در یک زمان وارد صحنه می‌شوند. بعد از گذشت مدت زمان مشخص، خودرویی که دارای ابعاد کوچک‌تر است، زودتر از خودروی بزرگ‌تر صحنه را ترک می‌کند و خودروی بزرگ‌تر مدت زمان بیش‌تری در صحنه حضور دارد. از آن‌جایی که مسافت پیموده شده توسط هر دو خودرو برابر است، لذا سرعت خودرو با ابعاد بزرگ‌تر، کم‌تر از خودرو با ابعاد کوچک‌تر اعلام می‌گردد هر چند سرعت هر دو برابر است.

مزیت روش دوم در این است که برای ردیابی خودرو و همچنین برای یافتن اشیای پیش‌زمینه، از روش‌های پیچیده و زمان‌بر موجود استفاده نمی کند و لذا روشی بی درنگ و در عین حال دقیق در اندازه‌گیری سرعت است.

ایراد روش دوم در این است که برای خودروهایی که دارای سایه‌های بزرگی هستند، به دلیل تغییرات شدید پیش‌زمینه، جابه‌جایی مرکز ثقل زیاد بوده و این روش از دقت خوبی برخوردار نیست.

همچنین اگر خودرو دارای ابعاد بزرگی باشد به طوری که خودرو قبل از ورود به محدوده‌ی تعیین سرعت یا هنگام خروج از آن، به طور کامل در صحنه حضور نداشته باشد، در این صورت به دلیل عدم محاسبه‌ی درست مرکز ثقل، سرعت تخمین زده شده دقیق نیست. لذا بهتر است در این روش، محدوده‌ای که برای تعیین سرعت انتخاب می‌شود در قسمت بالا و پایینش به اندازه‌ی طول بزرگ‌ترین خودرو موجود در صحنه، فاصله وجود داشته باشد. همچنین ثابت بودن دوربین در حین اندازه‌گیری سرعت موضوع بسیار مهمی است..

جدول ۴-۳ مقایسه‌ی کیفی دو روش پیشنهادی

روش / پارامترها	روش اول: همبستگی	روش دوم: ردیابی مرکز ثقل
حساسیت به سایه‌ها	افتادن سایه‌ی خودرو مجاور یا خود خودرو بر روی محدوده‌ی تعیین سرعت باعث خطا می‌شود.	افتادن سایه‌ی خودرو مجاور بر روی محدوده‌ی تعیین سرعت باعث خطا می‌شود.
حساسیت به لرزش دوربین	نسبت به لرزش جزئی دوربین حساس نیست.	حتی نسبت به لرزش جزئی دوربین نیز حساس است.
تعیین سرعت هم‌زمان چند خودرو	امکان تعیین سرعت تنها یک خودرو در یک زمان وجود دارد.	امکان تعیین سرعت چندین خودرو به صورت هم‌زمان وجود دارد.
زمان پردازش	متوسط	کم
دقت تخمین سرعت	متوسط	زیاد
حساسیت به تغییرات شدید صحنه	پایین	بالا
حساسیت به ابعاد خودروها	بالا	پایین

جدول ۴-۴ مقایسه‌ی کمی دو روش پیشنهادی

روش / پارامترها	روش اول: همبستگی	روش دوم: ردیابی مرکز ثقل
ابعاد ویدئو	***720 × 576*** پیکسل	***720 × 576*** پیکسل
زمان پردازش روی لپ تاپ لنوو ***Z510***	۵۵ میلی ثانیه	۳۱ میلی ثانیه
زمان پردازش روی برد ***Odroid XU4***	۱۰۵ میلی ثانیه	۴۸ میلی ثانیه
میانگین خطا	۴,۹۹ درصد	۳,۳۱ درصد
انحراف معیار خطا	۴,۸۳ درصد	۳,۲۸ درصد
میانگین اختلاف سرعت تخمین زده شده و واقعی	±۲,۲۶کیلومتر بر ساعت	±۱,۳۹کیلومتر بر ساعت

و سرانجام در جدول زیر به طور خلاصه نتایج هر یک از روش‌های ارایه شده در فصل دوم با نتایج دو روش پیشنهادی ما مقایسه شده است.

مرجع	روش پیشنهادی	میانگین خطا	ماکزیمم اختلاف مقدار واقعی و تخمینی	انحراف معیار خطا	زمان پردازش	ابعاد ویدئو	نرخ ویدئو	بستر اجرایی
روش پیشنهادی اول	همبستگی متقابل	٪۴,۹۹ یا ±۲,۲۶***Km/h***	۷,۶۳***Km/h***	٪ ۴,۸۳	۵۵***ms***	***720 × 576***	۲۵	***Intel Core i5 -2.5 GHz و 6GB RAM***
					۱۰۵***ms***	***720 × 576***	۲۵	***Odroid XU4 8core 2GHz 2GB RAM***

روش پیشنهادی دوم	ردیابی مرکز ثقل	۳,۳۱ ٪ یا ±۱,۳۹*Km/h*	۵,۶۷*Km/h*	٪ ۳,۲۸	۳۱*ms*	*720 × 576*	۲۵	*Intel Core i5 2.5 GHz و 6GB RAM*
					۴۸*ms*	*720 × 576*	۲۵	*Odroid XU4 8core 2GHz 2GB RAM*
[*3*]	رفع تاری یک عکس گرفته شده	٪ ۵	------	-----	*ms*۳۰۰	------	*0*	Pentium M 1.6 GHz 512 MB RAM
[4]	شار نوری	٪ ۳,۸۶	------	٪۲,۸	------	------	------	------
[5]	الگوریتم تطبیق بلوکی	٪ ۲,۵	------	-----	------	------	------	------
[*7*]	ردیابی نقاط ویژگی	------	*±1-2 km/h*	-----	*ms*۳۰	*640 × 480*	*30*	*Intel Core i7 2.6 GHz و 8GB RAM*
[8]	ترکیب فیلتر کالمن و الگوریتم *Hungarian* برای ردیابی	٪۳,۲۲	۳*Km/h*	٪۰,۷۸	*ms*۱۰۰	*768 × 576*	۳۰	*Beagl eBoard- xM*
					۲۴,۲۸ *ms*	*768 × 576*	۳۰	*Lenovo T440 ThinkPad*
[9]	استفاده از پلاک به منظور ویژگی‌های پایداری برای ردیابی	۰,۵۹*Km/h*	۳,۹۱*Km/h*	۱,۶۳ *Km/h*	------	*768 × 480*	*31.25*	------

[12]	روش استافر برای یافتن مدل پس-زمینه‌و روش کوچیارا و فروند برای ردیابی	٪ ۴	------	۱,۳ ***Km/h***	------	------	------	**CPU اینتل *Centrino 2 vPro - 2GHz***
[17]	روش ***CVS***	۷***Km/h***	------	-----	۸۳,۳۳ ***ms***	------	------	**پردازنده‌ی ۲ هسته‌ای**
[***18***]	لبه‌یابی سوبل و الگوریتم ردیابی کالمن	٪ ۲	------	-----	------	------	------	------

فصل پنجم: نتیجه گیری و پیشنهاد راهکار آینده

۵–۱ نتیجه گیری و پیشنهاد راهکار آینده

در این پایان نامه به ارائه‌ی دو روش برای اندازه‌گیری سرعت خودرو پرداختیم. در روش اول، در ابتدا مدلی از پس‌زمینه به کمک الگوریتم ترکیب گوسی به دست آمد. سپس محدوده‌ای از تصویر را به منظور اندازه‌گیری سرعت انتخاب کردیم. به منظور افزایش دقت و رفع اعوجاج غیر خطی تصویر، بر روی این ناحیه تبدیلات افکنشی صورت گرفت. سپس به کمک انجام عملیات همبستگی بین ناحیه‌ی انتخاب شده‌ی متناظر در مدل پس‌زمینه و تصویر اصلی، ورود خودرو به این ناحیه را تشخیص داده شد و در نهایت بر اساس زمان حضور خودرو در این ناحیه و همچنین میزان مسافت پیموده شده در این ناحیه (طول ناحیه) سرعت خودرو تعیین گردید. در روش دوم، در ابتدا قاب گرفته شده یکسوسازی گردید سپس به کمک روش تفاضل قاب‌ها، تصویر اولیه‌ای از پیش‌زمینه حاصل شد. توسط عملیات ریخت‌شناسی و یافتن بیرونی‌ترین مرز اشیای پیش‌زمینه، تصویر دقیق‌تری از پیش‌زمینه ارائه گردید. سپس مرکز ثقل هر شی در هر قاب ویدئویی محاسبه شد و مسیر حرکت هر خودرو و در نتیجه مسافت پیموده شده توسط آن به دست آمد. به کمک زمان سپری شده و مسافت پیموده شده، تخمینی از سرعت زده شد.

مزیت روش اول در این است که تا حدودی نسبت به لرزش‌های جزئی دوربین مقاوم است. بزرگترین ایرادی که در روش اول وجود دارد، تاثیر ابعاد خودرو بر الگوریتم است که در این‌جا در نظر گرفته نشده است. مسافت پیموده شده توسط تمام خودروها یکسان است (محدوده‌ی تعیین شده) ولی زمان حضور هر خودرو در صحنه باید اندازه‌گیری شود. برای خودروهای کوچک زمان حضور در صحنه با دقت خوبی تخمین زده می‌شود ولی برای خودروهای بزرگتر مانند ون‌ها، اتوبوس‌ها، کامیون-ها و ... این زمان حضور به دلیل ابعاد زیاد آن‌ها به درستی تخمین زده نمی‌شود و سرعت اعلام شده دقیق نیست. همچنین اگر خودرو در صحنه متوقف شود زمان حضور آن زیاد می‌شود و سرعت آن نزدیک به صفر اعلام می‌شود هرچند دوباره حرکت کند و از صحنه بیرون رود. همچنین حضور سایه‌ی سایر خودروها در صحنه می‌تواند باعث بروز خطا در اندازه‌گیری زمان حضور و لذا تخمین سرعت شود. از آن‌جایی که در هر قاب عمل همبستگی بین پس‌زمینه و صحنه‌ی اصلی صورت می‌گیرد، لذا زمان پردازش این الگوریتم کمی زیاد است. همچنین ایراد دیگری که در روش اول وجود دارد این است که در یک زمان فقط می‌توان سرعت یک خودرو را گزارش داد. به منظور بهبود این الگوریتم، باید ابعاد هر خودرو استخراج شده و تحت یک ضریب در معادله‌ی تعیین سرعت آورده شود. همچنین به منظور خنثی کردن اثر سایه بر عملکرد الگوریتم، نیاز است که سایه‌ی خودروها را توسط الگوریتم‌های حذف سایه حذف کنیم.

مزیت روش دوم در این است که برای ردیابی خودرو و همچنین برای یافتن اشیای پیش‌زمینه، از روش‌های پیچیده و زمان‌بر موجود استفاده نمی کند و لذا روشی بی درنگ و در عین حال دقیق در اندازه‌گیری سرعت است. ایراد روش دوم در این است که برای خودروهایی که دارای سایه‌های بزرگی هستند، به دلیل تغییرات شدید پیش‌زمینه، جابه‌جایی مرکز ثقل زیاد بوده و این روش از دقت خوبی برخوردار نیست. همچنین اگر خودرو دارای ابعاد بزرگی باشد به طوری که خودرو قبل از ورود به محدوده‌ی تعیین سرعت یا هنگام خروج از آن، به طور کامل در صحنه حضور نداشته باشد، در این صورت به دلیل عدم محاسبه‌ی درست مرکز ثقل، سرعت تخمین زده شده دقیق نیست. لذا بهتر است در این روش، محدوده‌ای که برای تعیین سرعت انتخاب می‌شود در قسمت بالا و پایینش به اندازه‌ی طول بزرگ‌ترین خودرو موجود در صحنه، فاصله وجود داشته باشد. همچنین ثابت بودن دوربین در حین اندازه‌گیری سرعت موضوع بسیار مهمی است.

در جدول ۵-۱ نتایج مربوط به این دو روش آورده شده است:

جدول ۵-۱ نتایج حاصل شده از دو روش پیشنهادی

روش / پارامترها	روش اول: همبستگی	روش دوم: ردیابی مرکز ثقل
ابعاد ویدئو	***720 × 576*** پیکسل	***720 × 576*** پیکسل
زمان پردازش روی لپ تاپ لنوو ***Z510***	۵۵ میلی ثانیه	۳۱ میلی ثانیه
زمان پردازش روی برد ***Odroid XU4***	۱۰۵ میلی ثانیه	۴۸ میلی ثانیه
میانگین خطا	۴,۹۹ درصد	۳,۳۱ درصد
انحراف معیار خطا	۴,۸۳ درصد	۳,۲۸ درصد
میانگین اختلاف سرعت تخمین زده شده و واقعی	۲,۲۶±کیلومتر بر ساعت	۱,۳۹±کیلومتر بر ساعت

در آینده هدف توسعه‌ی سامانه‌ای است که قادر به حذف سایه‌ی مربوط به هر خودرو برای اندازه‌گیری دقیق‌تر سرعت باشد. همچنین برای بهبود روش اول می‌توان ابعاد خودروها را استخراج کرده و به عنوان ضریبی آن را در محاسبه‌ی سرعت لحاظ کنیم. همچنین اندازه‌گیری سایر پارامترهای ترافیکی نیز از طریق تحلیل سرعت‌های اندازه‌گیری شده و شمارش تعداد اتومبیل‌ها امکان‌پذیر می‌باشد.

پیوست‌ها

۶-۱ مشخصات برد توسعه‌ی *Odroid XU4* و نحوه‌ی کار با آن

در این بخش به بررسی مشخصات برد توسعه‌ی ***Odroid-XU4*** می‌پردازیم.

برد ***ODROID-XU4*** یک نسل جدید از تجهیزات محاسباتی و دارای سخت افزاری قدرتمند و بهینه با ابعاد کوچک است. این برد به صورت منبع باز بوده و می‌توان انواع سیستم عامل‌ها از جمله ***Android 4.4(Kitkat)، 5.0 Lollipop، 7.1 Nougat*** و ***Ubuntu16.04*** را بر روی آن اجرا نمود.

با دارا بودن ***eMMC 5.0، USB 3.0*** و ***Ethernet*** یک گیگابایتی، این برد با سرعت بسیار زیادی قادر به انتقال اطلاعات است.

تصویری از این برد را در شکل ۶-۱ مشاهده می نمایید:

شکل ۶-۱ نمایی از برد توسعه‌ی ***Odroid XU4***

مشخصات این برد در جدول ۶-۱ لیست شده است:

جدول ۶-۱ مشخصات برد ***Odroid xu4***

CPU	***Exynos 5422 Octa big.LITTLE ARM Cortex-A15 @ 2.0 GHz quad-core and Cortex-A7 quad-core CPUs***
GPU	***Mali-T628 MP6 (OpenGL ES 3.0/2.0/1.1 and OpenCL 1.1 Full profile)***
RAM	***2 GB LPDDR3 RAM at 933 MHz (14.9 GB/s memory bandwidth) PoP stacked***
Storage	***microSD card slot, eMMC5.0 HS400 Flash Storage***
USB	***1 × USB 2.0 A Host*** ***2 x USB 3.0 Host***
Video out	***HDMI connector 1.4a output Type-A***
Audio in	***-***
Audio out	***HDMI***
Network	***10/100/1000 Ethernet(8P8C)***
Peripherals	***expansion ports for GPIO, UART, I²C, I²S, SPI bus, PWM,ADC***
Power Source	***5 V 4 A DC input (5.5 x 2.1 mm barrel connector)***
PCB Size	***83 x 59 x 18 mm***
OS	***Linux (Ubuntu, Kali Linux), Android***

مزایا:

- این برد تقریبا صنعتی بوده و در محدوده‌ی دمایی ۱۰- تا ۴۵ درجه کار می‌کند.
- دارای رم نسبتا بالای ***2GB*** و از نوع ***DDR3*** است.
- دارا بودن ***Ethernet*** یک گیگابایت
- در ایران موجود بوده و به راحتی قابل تهیه است.

معایب:

- دارا نبودن ارتباط وای فای و بلوتوث بر روی برد

- دارا نبودن حافظه‌ی ***NAND***

از بین سیستم عامل‌های موجود سیستم عامل اوبونتو بر روی برد نصب شد. برای نصب سیستم عامل بر روی برد دو روش وجود دارد:

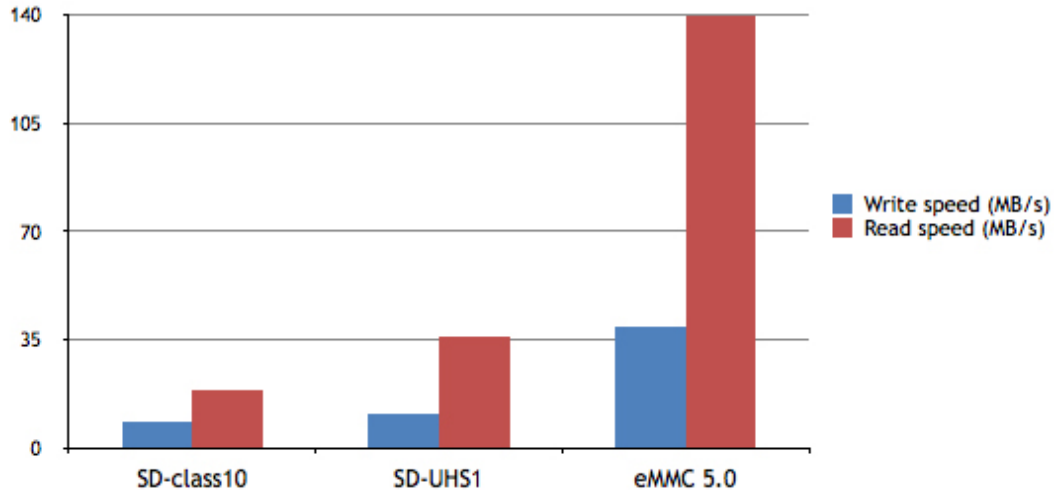
۱) ریختن سیستم عامل مورد نظر بر روی ***eMMC*** و قرار دادن وضعیت بوت شدن بر روی ***eMMC*** از روی برد

۲) ریختن سیستم عامل مورد نظر بر روی ***SD*** و قرار دادن وضعیت بوت شدن بر روی ***SD*** از روی برد

تذکر: مطابق با گفته‌ی سایت سازنده‌ی برد، برای این که زمان بوت شدن و همچنین خواندن و نوشتن از روی حافظه ی خارجی سریع تر شود، بهتر است از ***eMMC*** ورژن ۵ و همچنین از ***SD*** ورژن ***UHS-I*** استفاده شود.

در جدول ۶-۲ و نمودار ۶-۱ مقایسه ای از حافظه های خارجی و سرعت خواندن و نوشتن در آن ها آورده شده است:

نمودار ۶-۱ نمودار مربوط به سرعت خواندن و نوشتن انواع حافظه های خارجی



جدول ۶-۲ مقایسه ای از سرعت خواندن و نوشتن انواع حافظه های خارجی

	SD-class10	***SD-UHS1***	***eMMC 5.0***
Write speed (MB/s)	***8.5***	***10.8***	***39.3***
Read speed (MB/s)	***18.9***	***35.9***	***140***

پس از ریختن سیستم عامل بر روی ***SD***، وضعیت کلید بوت روی برد را بر روی ***SD*** قرار داده و ***SD*** را در سوکت مربوطه قرار می‌دهیم. سپس یک سر کابل ***HDMI*** را به برد و سمت دیگر آن را نیز به مانیتور یا ***LCD*** متصل می‌نماییم. حال تغذیه‌ی برد را متصل می‌کنیم. بعد از مدتی سیستم عامل بالا آمده و پس از ***Log in*** شدن صفحه‌ای مشابه شکل ۶-۲ مشاهده می‌شود:

شکل ۶-۲ محیط مربوط به سیستم عامل ***UBUNTU 16.04***

در گام بعدی برای نصب نرم افزارها و بسته‌های[۱] نرم افزاری مورد نیاز به صورت برخط[۲]، لازم است که برد را به اینترنت متصل نماییم. برای این منظور می توان از دو روش به اینترنت متصل شد:

۱- به کمک کابل شبکه (***LAN***) و ارتباط ***Ethernet***

۲- به کمک ماژول گیرنده ی ***WI-FI*** پشتیبانی شده توسط برد

در روش اول، کابل شبکه را به کانکتور ***Ethernet*** متصل کرده و برد به صورت خودکار به اینترنت متصل می‌شود. علامت دو فلش بالا و پایین در گوشه‌ی بالا سمت راست صفحه ظاهر می‌گردد.

[۱] ***Package***

[۲] ***online***

در روش دوم، از ماژول ***WIFI Module 4*** که در زیر تصویر آن را مشاهده می‌کنید و سازگار با این برد است استفاده می‌نماییم:

در ابتدا این ماژول ***WI-FI*** را به یکی از درگاه های ***USB*** برد متصل می کنیم. در بالا سمت راست صفحه بر روی علامت ***WIFI*** کلیک کرده و نام مودم مورد نظر را پیدا و بر روی آن کلیک می-کنیم. برای اتصال به این شبکه رمز عبور آن پرسیده می‌شود. با زدن رمز عبور به صورت خودکار برد به اینترنت متصل می‌شود.

حال برای کار با برد و نصب کتابخانه‌های مورد نظر، لازم است بسته‌های نرم افزاری مورد نیاز بر روی سیستم عامل نصب شود.

هنگامی که یک برنامه قرار است بر روی یک کامپیوتر اجرا شود برای این که به درستی کار کند به منابع دیگری نیاز دارد. هنگامی که یک نرم افزار نصب می‌شود ممکن است هزاران فایل برای اجرای برنامه نیاز باشد. حتی محل قرارگیری فایل‌ها و همچنین نوع معماری پردازنده‌ای که کامپیوتر از آن استفاده می‌کند نیز مهم است. ***UBUNTU*** از بسته‌های نرم افزاری برای ذخیره‌ی هر آن چه که یک برنامه‌ی خاص برای اجرا به آن نیاز دارد، استفاده می‌کند. بسته‌های نرم افزاری شامل مجموعه‌ای از فایل‌هاست که در درون یک فایل قرار گرفته است. علاوه بر این فایل‌ها که برای اجرای برنامه نیاز است، فایل‌های خاص دیگری به نام اسکریپت‌های نصب[۱] وجود دارد که فایل هایی را در جایی که نیاز است کپی می‌کنند.

دو نوع بسته‌ی نرم افزاری وجود دارد: سورس پکیج[۲] و باینری پکیج[۳]. سورس پکیج به بسته‌های نرم افزاری گفته می‌شود که فقط شامل سورس کد می‌باشند و اگر کد از طریق درستی کامپایل شود می‌توانند به طور عمومی بر روی هر نوع ماشینی استفاده شوند. و اما باینری پکیج، به بسته‌های نرم افزاری گفته می شود که منحصرا برای یک نوع خاصی از کامپیوتر یا معماری ساخته شده اند. با نوشتن دستور ***uname -m*** در ترمینال لینوکس، می توان به معماری کامپیوتر خود پی برد. معماری برد ***Odroid xu4*** از نوع ***ARMhf*** می‌باشد.

---

[۱] ***Installation Scripts***

[۲] ***Source Package***

[۳] ***Binary Package***

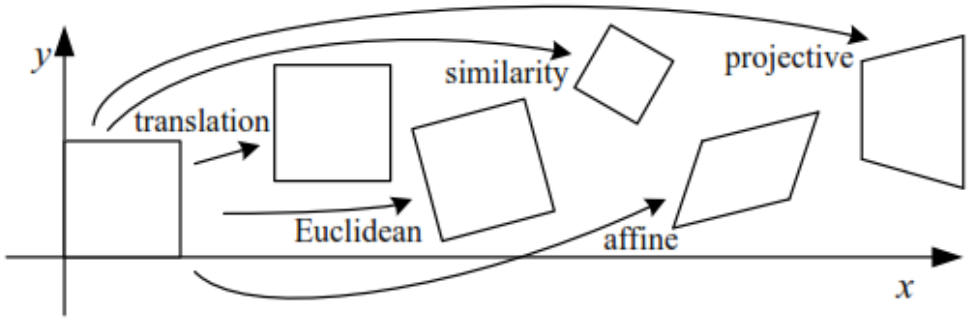
پس از اتصال برد به اینترنت، نیاز است که بسته‌های نرم افزاری مورد نیاز بر روی برد نصب شود. به این منظورکافیست دستور ***sudo apt-get install*** را در محیط ترمینال نوشته و سپس نام بسته‌ی نرم افزاری مورد نظر خود را بعد از آن بنویسیم تا بسته‌ی نرم افزاری مورد نظر از طریق اینترنت دانلود و نصب شود.

پس از نصب بسته‌های نرم افزاری مورد نیاز، باید نرم افزار ***QT*** و کتابخانه‌ی ***OpenCV*** را بر روی برد نصب نمود. از آن‌جایی که هیچ نسخه‌ی از قبل ساخته[۱] شده‌ای از ***QT*** و ***OpenCV*** بر روی ***Odroid XU4*** وجود ندارد لذا نیاز است که سورس آن‌ها را دانلود کرده و جداگانه آن‌ها را کامپایل کرده و بسازیم. به کمک ابزار ***Cmake*** این کار را انجام می‌دهیم.

[۱] ***Prebuild***

۶-۲مفاهیم و تئوری‌های مربوط به تبدیلات افکنشی

به دلیل موقعیت قرارگیری دوربین نسبت به صحنه‌ی مورد نظر، همواره شاهد اثر پرسپکتیوی در تصاویر می‌باشیم. در حالت کلی بسته به موقعیت دوربین و صحنه، تبدیلات دوران، انتقال، تغییر مقیاس، همگر[۱] و افکنش[۲] روی صحنه ممکن است رخ دهد. در شکل ۶-۳ این تبدیلات و اثرش بر روی صحنه را مشاهده می‌نمایید:



شکل ۶-۳ مجموعه تبدیلات هندسی دو بعدی روی تصویر

تبدیلات هندسی رابطه‌ی مکانی بین پیکسل‌ها را در یک تصویر تغییر می‌دهد. در پردازش تصاویر دیجیتال، تبدیل هندسی شامل دو عملیات اصلی است:

۱- تبدیل مکانی: نحوه‌ی قرارگیری پیکسل‌ها را در یک تصویر مشخص می‌کند.
۲- درون‌یابی پیکسل‌ها در سطح خاکستری: شامل تخصیص مقدار خاکستری به پیکسل‌های تصویر تبدیل شده است.

فرض کنید تصویر f با مختصات پیکسل (x,y) تحت تبدیل هندسی قرار می‌گیرد و تصویر g را با مختصات (x', y') ایجاد می‌کند. این تبدیل به صورت زیر بیان می‌شود:

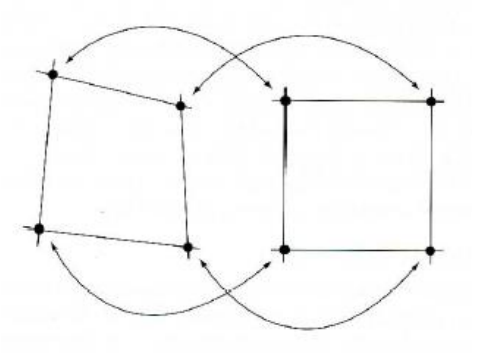
$$x' = r(x,y) \quad \text{(معادله ۶-۱)}$$

$$y' = s(x,y) \quad \text{(معادله ۶-۲)}$$

[۱] ***Affine***

[۲] ***Projective***

که $r(x, y)$ و $s(x, y)$ تبدیلات مکانی هستند که تصویر g را ایجاد می‌کنند. به عنوان مثال اگر $r(x, y) = x/2$ و $s(x, y)=y/2$ باشد، تصویر g از لحاظ طول و عرض نصف تصویر f است. اگر $r(x, y)$ و $s(x, y)$ معلوم باشند، می‌توان تصویر f را با اعمال تبدیل معکوس بر روی تصویر g به دست آورد. به دست آوردن توابعی که قادر به توصیف تبدیلات هندسی انجام شده بر روی تمامی تصاویر باشند، کار غیر ممکنی است لذا روشی که اغلب برای به دست آوردن ماتریس تبدیل مورد استفاده قرار می‌گیرد، استفاده از مجموعه پیکسل‌های ورودی (تصویر اعوجاجی) و خروجی (تصویر اصلاح شده) است. مکان ین پیکسل‌ها در خروجی مشخص است.



شکل ۶-۴ تبدیل هندسی

فرض کنید فرآیند اعوجاج هندسی در نواحی چهارضلعی توسط یک جفت معادله‌ی دو خطی[۱] مدل شده است.

$$x' = C_1x + C_2y + C_3xy + C_4 \quad \text{(معادله ۶-۳)}$$

$$y' = C_5x + C_6y + C_7xy + C_8 \quad \text{(معادله ۶-۴)}$$

توسط ۸ نقطه‌ی مختلف این معادله قابل حل است (ضرایب C_1 تا C_8). این ضرایب اعوجاج هندسی که باعث تبدیل تمام پیکسل‌ها به درون ناحیه‌ی چهارضلعی می‌شوند را تشکیل می‌دهند.

از آن‌جایی مختصات x' و y' ممکن است غیرصحیح باشند، لذا برای این‌که مقدار سطح خاکستری آن را به دست آوریم، نیاز است که عملیات درون‌یابی[۲] صورت گیرد و مقدار سطح خاکستری آن پیکسل به دست آید. سه روش معمول برای این کار عبارتند از: نزدیک‌ترین همسایه‌ها[۳] (راحت‌ترین روش)، دوخطی[۴] وکانولوشن مکعبی[۱].

---

[۱] ***bilinear***

[۲] ***gray-level interpolation***

[۳] ***Nearest neighbor***

[۴] ***Bilinear***

ماتریس انتقال را با H_T نشان داده و داریم:

(معادله ۶-۵)

$$H_T = \begin{bmatrix} 1 & 0 & a \\ 0 & 1 & b \\ 0 & 0 & 1 \end{bmatrix}$$

این ماتریس پس از ضرب شدن در پیکسل مورد نظر، طول و عرض آن را به اندازه‌ی a و b شیفت می‌دهد.

مثال: در این‌جا نقطه‌ی $(10, 20)$ به اندازه‌ی ۵ واحد در راستای x و y شیفت داده می‌شود.

(معادله ۶-۶)

$$\begin{bmatrix} 1 & 0 & 5 \\ 0 & 1 & 5 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 10 \\ 20 \\ 1 \end{bmatrix} = \begin{bmatrix} 15 \\ 25 \\ 1 \end{bmatrix}$$

ماتریس تغییر مقیاس را با H_S نشان داده و داریم:

(معادله ۶-۷)

$$H_S = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

این ماتریس پس از ضرب شدن در پیکسل مورد نظر، طول و عرض آن را در a و b ضرب می‌کند.

مثال: در این‌جا نقطه‌ی $(10, 20)$ با ضریب ۵ در راستای x و y تغییر مقیاس می‌دهد.

(معادله ۶-۸)

$$\begin{bmatrix} 5 & 0 & 0 \\ 0 & 5 & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 10 \\ 20 \\ 1 \end{bmatrix} = \begin{bmatrix} 50 \\ 100 \\ 1 \end{bmatrix}$$

ماتریس دوران را با H_r نشان داده و داریم:

(معادله ۶-۹)

$$H_r = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

این ماتریس پس از ضرب شدن در پیکسل مورد نظر، آن را به اندازه‌ی زاویه‌ی θ دوران می‌دهد.

مثال: در این‌جا نقطه‌ی $(10, 20)$ به اندازه‌ی ۹۰ درجه دوران داده می‌شود.

(معادله ۶-۱۰)

$$\begin{bmatrix} \cos(\frac{\pi}{2}) & -\sin(\frac{\pi}{2}) & 0 \\ \sin(\frac{\pi}{2}) & \cos(\frac{\pi}{2}) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 10 \\ 20 \\ 1 \end{bmatrix} = \begin{bmatrix} -10 \\ 20 \\ 1 \end{bmatrix}$$

---

[1] ***cubic convolution***

ماتریس تبدیل افکنش را با H_P نمایش داده و داریم:

$$H_P = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \quad \text{(معادله ۶-۱۱)}$$

توجه: سایر تبدیل‌ها حالت خاصی از تبدیل افکنش است.

ماتریس H_P را می‌توان توسط ۴ نقطه از صحنه، تعیین نمود. به کمک روابط زیر و با داشتن ۴ نقطه از صحنه می‌توان ماتریس ***Projection*** را محاسبه نمود.

$$\begin{pmatrix} x'_1 \\ x'_2 \\ x'_3 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad \text{(معادله ۶-۱۲)}$$

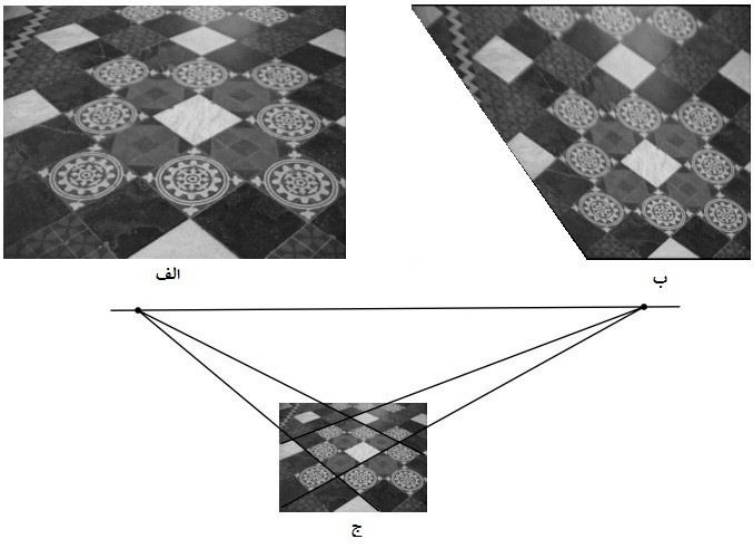
$$x' = \frac{x'_1}{x'_3} = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + h_{33}} \quad \text{(معادله ۶-۱۳)}$$

$$y' = \frac{x'_2}{x'_3} = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + h_{33}} \quad \text{(معادله ۶-۱۴)}$$

$$x'(h_{31}x + h_{32}y + h_{33}) = h_{11}x + h_{12}y + h_{13} \quad \text{(معادله ۶-۱۵)}$$

$$y'(h_{31}x + h_{32}y + h_{33}) = h_{21}x + h_{22}y + h_{23} \quad \text{(معادله ۶-۱۶)}$$

در شکل ۶-۵ نمونه‌ای از تبدیل افکنش را مشاهده می‌نمایید که خطوط غیر موازی صحنه به خطوطی موازی تبدیل شده اند.

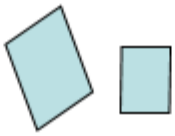
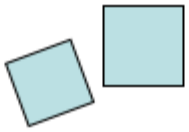
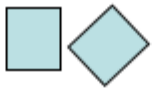


شکل ۶-۵ نمونه‌ای از رفع اعوجاج پرسپکتیوی – الف) تصویر دارای پرسپکنیو ب) تصویر یکسو شده ج) اثر پرسپکنیو بر روی خطوط موازی

در تبدیل همگر، خطی بودن و نسبت فاصله‌ها حفظ می‌شود و در نتیجه‌ی این تبدیل، تمام نقاط روی یک خط در ورودی، در خروجی نیز روی یک خط خواهند ماند. در این تبدیل طول و زاویه‌ی بین خط‌ها لزوما حفظ نمی‌شود. تبدیلات انتقال، تغییر مقیاس و دوران حالت خاصی از این تبدیل می‌باشند. ماتریس همگر که حالت خاصی از ماتریس $\boldsymbol{H_p}$ است را با $\boldsymbol{H_a}$ نشان داده و داریم:

$$H_a = \begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$$ (معادله ۶-۱۷)

در ادامه انواع تبدیلات هندسی را به همراه ماتریس تبدیلشان مشاهده می‌کنید:

Projective 8dof	$\begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix}$		**8 درجه آزادی (2 درجه مقیاس، 2 درجه چرخش، 2 درجه انتقال، 2 درجه خط در بی نهایت)**
Affine 6dof	$\begin{bmatrix} a_{11} & a_{12} & t_x \\ a_{21} & a_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		**4 درجه آزادی (1 درجه مقیاس، 1 درجه چرخش، 2 درجه انتقال)**
Similarity 4dof	$\begin{bmatrix} sr_{11} & sr_{12} & t_x \\ sr_{21} & sr_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		**6 درجه آزادی (2 درجه مقیاس، 2 درجه چرخش، 2 درجه انتقال)**
Euclidean 3dof	$\begin{bmatrix} r_{11} & r_{12} & t_x \\ r_{21} & r_{22} & t_y \\ 0 & 0 & 1 \end{bmatrix}$		**3 درجه آزادی (1 درجه چرخش، 2 درجه انتقال)**

که $T = \begin{bmatrix} t_x \\ t_y \\ 1 \end{bmatrix}$ ماتریس انتقال، $R = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$ ماتریس دوران، $\boldsymbol{S}$ ضریب تغییر مقیاس و ماتریس $A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$ نیز ماتریس همگر است.

مراجع

[1] "Traffic enforcement camera," Wiki Pedia. [Online]. Retrieved January 6, 2018 Available: https://en.wikipedia.org/wiki/Traffic_enforcement_camera.

[2] "Cosine Effect Error." [Online]. Retrieved January 6, 2018

Available: http://copradar.com/preview/chapt2/ch2d1.html.

[3] H. Y. Lin, K. J. Li, and C. H. Chang, "Vehicle speed detection from a single motion blurred image," Image Vis. Comput., vol. 26, no. 10, pp. 1327–1337, 2008.

[4] O. Ibrahim, H. ElGendy, and A. M. ElShafee, "Speed Detection Camera System using Image Processing Techniques on Video Streams," Int. J. Comput. Electr. Eng., vol. 3, no. 6, pp. 771–778, 2011.

[5] S. Jhumat and R. K. Purwar, "Techniques to Estimate Vehicle Speed," vol. 3, no. 6, pp. 6875–6878, 2014.

[6] S. Joshi, "Vehicle Speed Determination Using Image Processing," Int. J. Res. Manag. Sci. Technol., vol. 2, no. 1, pp. 57–60, 2014.

[7] M. S. Temiz, S. Kulur, and S. Dogan, "Real Time Speed Estimation From Monocular Video," ISPRS - Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci., vol. 3, no. September, pp. 427–432, 2012.

[8] Y.G. Anil Rao,N. Sujith Kumar, H. S. Amaresh, and H.V.Chirag, "Real-Time Speed Estimation of Vehicles from Uncalibrated View-Independent Traffic Cameras.",2015.c

[9] D. C. Luvizon, B. T. Nassu, and R. M. Department, "Vehicle Speed Estimation By Licence Plate Detection And Tracking" IEEE Int. Conf. Acoust. Speech Signal Process., pp. 6563–6567, 2014.

[10] S. S. S. Ranjit, S. A. Anas, S. K. Subramaniam, K. C. Lim, A. F. I. Fayeez, and A. R. Amirah, "Real-Time Vehicle Speed Detection Algorithm using Motion Vector Technique," pp. 67–71, 2012.

[11] H. Tan, J. Zhang, and J. Feng, "Vehicle speed measurement for accident scene investigation" Proc. - IEEE Int. Conf. E-bus. Eng. ICEBE 2010, pp. 389–392, 2010.

[12] J. Wu, Z. Liu, J. Li, C. Gu, M. Si, and F. Tan, "An algorithm for automatic vehicle speed detection using video camera," 2009 4th Int. Conf. Comput. Sci. Educ., pp. 3519–3522, 2009.

[13] Chris Stauffer,W.E.L Grimson.Adaptive Background Mixture Models for Real time Tracking. Computer Vision and Pattern Recognition, 246-252, 1999.

[14] Cucchiara R. Image Analysis and Rule-based Reasoning for a Traffic Monitoring System. IEEE Transactions on Intelligent Transportation Systems, 1(2):119-130, 2000.

[15] Freund, P., N. Robust Tracking of Position and Velocity with Kalman Snakes. IEEE Trans Pattern Analysis and Machine Intelligence, 22 (6): 564-569,2000.

[16] H. A. Rahim et al., "2010 - Vehicle Speed Detection Using Frame Differencing for Smart Surveillance System" no. Isspa, pp. 630–633, 2010.

[17] A. G. Rad, A. Dehghani, and M. R. Karim, "Vehicle speed detection in video image sequences using CVS method," Int. J. Phys. Sci., vol. 5, no. 17, pp. 2555–2563, 2010.

[18] C. Maduro, K. Batista, P. Peixoto, and J. Batista, "Estimation of vehicle velocity and traffic intensity using rectified images," Proc. - Int. Conf. Image Process. ICIP, pp. 777–780, 2008.

[19] D. J. Dailey and F. W. Cathey, "A novel technique to dynamically measure vehicle speed using uncalibrated roadway cameras", in Proceedings of IEEE, 2005.

[20] B. Bose and E. Grimson, "Ground plane rectification by tracking moving objects," in Proceedings of the Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2003.

[21] Lazaros Grammatikopoulos, George Karras, and Elli Petsa, "Automatic estimation of vehicle speed from uncalibrated video sequences," in International Archives of Photogrammetry and Remote Sensing, 2002.

[22] D. Liebowitz and A. Zisserman, "Metric rectification for perspective images of planes," in Proceedings of Computer Vision and Pattern Recognition, 1998.

[23] V. K. Madasu and M. Hanmandlu, "Estimation of vehicle speed by motion tracking on image sequences," IEEE Intell. Veh. Symp. Proc., pp. 185–190, 2010.

[24] P. P. S. S and C. Manisha, "25e Optical Flow and Background Subtraction," Proc. Int. Conf. Adv. Comput. Sci. Appl., 2013.

[25] S. Zhu and T. Koga, "Feature Point Tracking for Car Speed Measurement," APCCAS - IEEE Asia Pacific Conf. Circuits Syst., no. 1, pp. 1144–1147, 2006.

[26] A Video-Based System for Vehicle Speed Measurement in Urban Roadways Retrieved January 6, 2018

Available: http://www.dainf.ct.utfpr.edu.br/~rminetto/projects/vehicle-speed

Abstract

In this thesis, a real time algorithm for speed estimation of on-road vehicles is proposed. To determine the speed of vehicles, two video processing methods are proposed and evaluated using the OpenCV library and C ++ programming language. The first method is based on extraction of background model and correlation algorithm. In this method, the mixture of Gaussian is used to obtain background model. Then we select an area of the image for speed measurement. In order to increase the accuracy and eliminate the nonlinear distortion of the image, a projective transformation is applied to deperspective this area. Presence of a vehicle is verified using the correlation between this area in the background model and the upcoming frame. Finally, speed is estimated by dividing the length of this area by the presence time of the vehicle in it. In the second method, we develop a real-time algorithm, so we avoided complicated tracking algorithms. In this method, at first, the area of interest for speed estimation is determined. This area is rectified by the projective transformation. The initial model of the foreground, including movable objects, is obtained by the difference of successive frames. By morphological operations, irrelevant objects are eliminated so that moving objects become more coherent. The distance traveled by each vehicle is measured by the displacement of the center of gravity in successive frames within the specified area. Travel time is obtained by counting the number of frames. Eventually, car speed is obtained by dividing the displacement by time for each car and multiplying the result by a factor that converts pixels per second to km/s.

The both methods are implemented on Lenovo z510 laptop with a 2.5 GHz Intel Core i5 processor and 6 GB of RAM, as well as on an Odroid XU4 with 2 GHz 8-core processor and 2 GB of RAM. The runtime was measured in the first method was 55 ms on laptop and 105 ms on XU4 board. The reported speed error was 4.99% with a standard deviation of 4.83% and a difference of ± 2.26 km / h. In the second method, the runtime on the laptop was 31 ms and on the XU4 Board was 48 ms. Also, the reported speed error was 3.31% with a standard deviation of 3.28% and a difference of ± 1.39 km / h.

Keywords—Video Processing; Projection Transform; Speed Estimation; Difference of Frames, Morphological Operation

Department of Electrical and Robotics Engineering

M.Sc Thesis in Communication Systems Engineering

Real Time Vehicle Speed estimation on an Embedded ARM-Based Board Using Video Processing

Seyed Amir Aghayan

Supervisor:

Dr. Hossein Khosravi

January 2018