بسمه تعالی

درس دید کامپیوتری

گزارش پروژه نهایی

نام استاد:

دکتر محمدزاده

نام دانشجو:

محمد کلباسی (96102256)

مقدمات:

در ابتدا نیاز است که ما به صورت خوبی پس زمینه را از خود تصویر جدا کنیم بعد از این کار میتوانیم با اعمالی مانند کانتور و غیره هر جسم غیر پس زمینه که در ویدیو های ما بیشتر خوشبختانه ماشین میباشند را جدا کنیم، به خاطر همین امر نیز ما نیاز نداریم که الگوریتمی طراحی کنیم که ماشین هارا از موارد دیگر جدا کند و میتوانیم از دو ادمی که در ویدو 2 عبور میکنند چشم پوشی کنیم

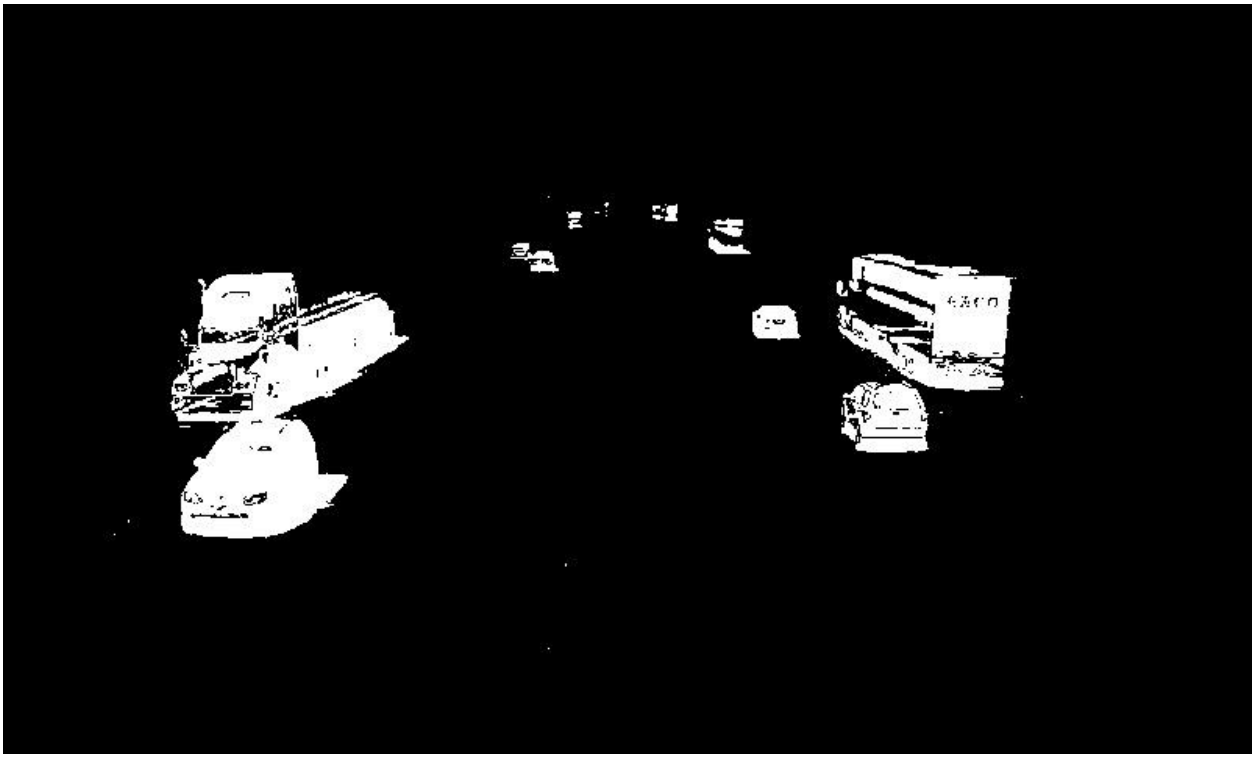
برای ساخت پس زمینه کافی است که مانند متد تمرین اول از همه فریم ها بیایم و یک میانگین بگیریم و انرا به عنوان پس زمینه معرفی کنیم برای سرعت بیشتر میتوان که از هر تعداد فریم یکی را انتخاب کنیم(مثلا از هر 5 فریم یکی) اینگونه همپوشانی ماشین هایی که در میانگین گیری قرار میگیرد کمتر میشود ولی ممکن است در کل وزن فریم هایی که مدت بیشتری ثابت میماندند نیز کمتر شود برای همین نیز ما همه فریم ها را لحاظ میکنیم برای این کار فریم ها را ذخیره کرده و بعد میانگین گیری را در بعد خود انجام میدهیم. در این مرحله fps را نیز پیدا میکنیم که در اینده به ما در امر پیدا کردن زمان ماشین ها کمک میکند. نتیجه برای ویدیو ها به صورت شکل زیر در امده است:

تصویر 1

تصویر 2

نتیجه در هردو مطلوب میباشد در خط راست کمی تیره تر از حالت عادی شده است و هنیتطور چپ ولی تاثیر زیادی در نتایج نخواهند داشت.

حال باید تصویر را جدا کرده و کانتور هارا تولید کنیم. برای جدا ساختن تصاویر متحرک در تمرین اول سه روش معرفی کرده بودیم دو روش مناسب برای این کار یکی این میبود که بیایم و تصاویر را به صورت مطلق از هم کم کنیم و بعد در فضای HSV آنرا برده و ترشلد بزنیم در این حالت نتایج نسبتا خوب بودن حالت دوم این است که محاسبات را در حالت سیاه و سفید قرار بدهیم و بعد ترشلد بزنیم در این حالت تصویر بدست امده نویزی میباشد ولی کامل در از روش اول ماشین هارا تشخیص میداد پس ما میتوانیم با فیلترهایی مانند erosion ,dilation,opening,closing این مشکل را حل کنیم پس این کار را انجام میدهیم دقت کنید که ما از absdiff استفاده کنیم تا مشکلات ناشی از تفاضل دو تصویر برنخوریم این امر باعث میشود که چرخش های بیتی عجیب(تفاضل 2 و 1 حاصل 255 بدهد!) نداشته باشیم برای همین نیز از این تابع استفاده میکنیم با اینکه opening ,closing ترکیبی از دوفیتلر دیگر میباشد چون ما لزوما از کرنل های یکسانی استفاده نمیکیم بهتر است که هر 4 حالت را قرار دهیم از طرف دیگر ترتیب خود قرار گرفتن این عملگر ها نیز برای ما مهم میباشند ما میایم و در هر مرحله انرا رسم میکنیم اگر دو تصویر کم شده را روی ان ترشلد قرار دهیم به صورت زیر میباشد

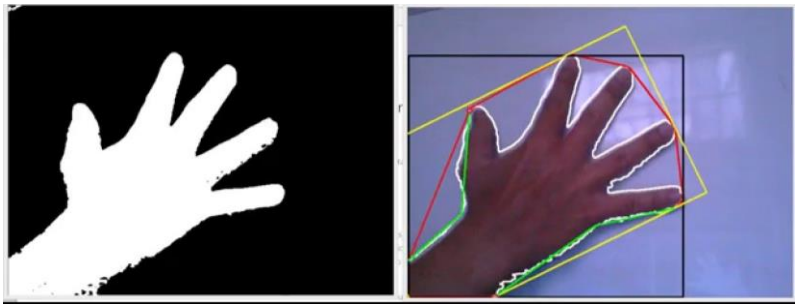


شکل 3: بعد کم کردن و قرار دادن ترشلد

هدف رسیدن به تصویری تمام پر میباشد برای هر کدام میابشد. بیاید اول از dilation استفاده کنیم که تو پر شوند بعد از closing استفاده میکنیم با این منطق که حال که به شدت و به صورت خشنی سعی در پر کردن داشته ایم بیایم و فاصله های اخر را ببنیدم بعد از opening استفاده میکنیم تا ذرات کوچک سفید حذف شوند حال برای اینکه اندازه منطقی تر شود در نهایت از erosionاستفاده میکنیم نتیجه بینهایت ایده ال نیست ولی به صورت کلی برای ما کار را فراهم میکند نتیجه نمونه در فریم 300 در زیر اورده شده است دقت کنید در هنگام اجرای کد این قسمت تمام نتایج همزمان نشان داده میشوند:

شکل 4

اگر خیلی نزدیک به هم باشند اینکه در یک کانتور در نظر گرفته شود برای ما مشکلی ایجاد نمیکند در ظاهر چون به هر حال در خلاصه سازی میباشیم و اگر خیلی نزدیک باشند انگار در یک tube بوده ولی از طرفی این کار برای شمارش ماشین برای ما باعث دردسر خواهد شد برای همین نیز تا حد ممکن سعی میکنیم دقت را بهتر کنیم تا کانتور های تشخیص داده شده ایده ال شوند. حال باید بیایم و کانتور های موجود در یک فریم را پیدا کنیم و از اطلاعات ان برای بروزرسانی استفاده کنیم به راحتی با دستورcv2.findContours میایم و کانتور هارا پیدا میکینم با استفاده از خروجی سلسله مراتبی ان میرویم وکانتوری که کانتور های بیرونی را در بر میگیرد انتخاب میکنیم[1] . بعد میایم و روش طول این کانتور شرطی قرار میدهیم بدیهی است که هر کانتور بیرونی لزوما کانتور دلخواه ما نمیباشد مثلا اگر در همین تصاویر دقت کنید متوجه میشوید که بعضی نقاط سفید نویزی ممکن است برای ما دردسر ایجاد کنند برای همین ترشلد گذاری منطقی برای دو بازه نیاز میباشد حال کا کانتور هارا داریم در ابتدا مرکز های انرا تفکیک میکنیم. حال باید خود اطلاعات کانتور را نیز ذخیره کنیم در ابتدا ما امدیم و با استفاده از convex hull contour خواستیم مشخصات انرا ذخیره کنیم زیرا انتظار رفتار بهتری نسبت به حالت مستطیلی از ان داریم



[1] https://github.com/seraj94ai/vehicle-tracking/blob/master/vehicle%20tracking.py

ولی وقتی نتیجه انرا مشاهده کرده ایم به شدت پخش و برای حالت های نویزی عبور کرده مساحت خیلی بیشتری را به خود اختصاص میداد پس ما از این کار استفاده نمیکنیم و بنابر پیشنهاد تی ای از روش هایی مبنی بر خود کانتور جلو میریم.

حال میاییم و دو کلاس تعریف میکنیم در کلاس اول ما به نوعی برای اینکه متغیر های حیاتی را نگه داریم و بروز کنیم استفاده میکنیم و در کلاس دوم فعالیت های مانند بررسی اینکه دو کانتور (بخوانید tube) ایا یکی میباشند یا خیر.

پس فرض میکنیم حال کانتور ها را داریم و با کلاس ها میخواهیم به شناسایی بپردازیم:(الگوریتمی که در زیر استفاده شده است مبنی بر کد که لینک ان در پانویس اورده شده است[2]) این روش مبتنی بر مقایسه مرکز های دسته های ورودی داده شده به ان میباشد پس باید در ابتدا ما باکس هایی که نمایانگر هر ماشین و مراکز انرا به ان معرفی کنیم همچنین نیاز است که هر tube مشخصه خاص خود را داشته باشد برای همین باید ID مخصوص به خود داشته باشند

ما نیاز به kalman filter داریم که انرا از فایل قرار داده شده استفاده کرده و فراخوانی میکنیم کاربرد این برای بروزرسانی مکان نقطه

مرکز براساس نقطه قبلی که در ان قرار گرفته و بهترین نقطه جدیدی که ما انرا در ان میبینیم میباشد. در ابتدا ما در کلاس Tracked میایم و مشخصات نقطه دنبال شده را قرار که یعنی به نوعی tube های مربوطه را تولید میکنیم .این مشخصات در ابتدا سه متغیر ورودی میباشند که ما با انها مشخص میکنیم که ماکزیمم نقطه کلیدی دو تصویر با هم چه قدر فاصله داشته باشد هنوز انرا میتوانیم همسایه هم گرفته و بهم ربط داده دیگری این است که کدام tube میتواند در چند فریم غایب باشد بدون اینکه ما ان tube را تمام شده بدانیم مثلا ممکن است به خاطر روش تولید کانتور ها و اگر تصاویر را مشاهده کنید بعضی وقتا وسط اجسام به خوبی دیده نمیشود (نمیتوانستیم مقدارdialation را بیشتر کنیم وگرنه تصویر به طور کامل مشخصه را از دست میدادا) برای همین یک کانتور واحد در عین حال ممکن است به صورت موقت دو یا چند کانتور شده و بعد به حالت عادی گردد پس ما باید یک پنالتی در نظر بگیریم و اگر از ان بیشتر شد دیگر فریم را تمام شده(یا نویزی بنا بر پارامتر بعدی مشخص میشود) پارامتر بعدی مشخص کردن این است که هر tube حداقل چند تا فریم مربوطه باید داشته باشد تا اینکه بتوانیم انرا غیر یک tube نویزی معرفی کنیم. این پارامتر ها مربوط به کلاس Tracking میباشند ولی مشخصات اولیه هر tube و frame های ان و ID مربوطه و فریم ابتدایی(برای تشخیص مسیر به ان نیاز خواهیم داشت) و سرعت و فریم هایی که فعال نبوده و مرکز فعلی را نگه میداریم این کلاس Tracked میباشد.

در کلاس Tracking ما عملگر بروز رسانی را تعریف میکنیم. ورودی های این عملگر کانتور های جدید مرکز های جدید و شماره فریم و فریم ریت میباشد. فایده هر کدام در جلو تر اورده شده است. در ابتدا اگر ما از پیش Track هایی (معدل Tube های ما میباشد) نساخته باشیم به تعداد مرکز فعلی میایم و از کلاس Tracked تولید میکنیم و بعد عملیات اینده را انجام میدهیم.

در ابتدا ما برای اینکه بخواهیم هر دو فریم را به هم مربوط کنیم از چیزی که در تمرین 4 نیز به خوبی با ان کار کرده باشیم استفاده کنیم یعنی اینترست پوینت ها ولی مشکلی اساسی که در این امر به ان برخوردیم این بود که در ابتدا هر دو فریم وقتی میامدیم و به ازای کانتور های موجود باید همه را با همه بررسی میکردیم و محاسبه نقاط اینترست پوینت خود امری زمان گیر محسوب میشد از طرف دیگر بین دو کانتور متفاوت ما مچ های متفاوت میگرفتیم یعنی دو کانتور با 5 نقطه به هم مربتط و با یکی دیگر کانتور اولی در 10 نقطه مچ میشد ولی این 10 نقطه در مرز ترشلد ما قرار گرفته بودند در حالی که در حالت اول برای ما دقت بیشتری محاسبه میشد ولی مچینگ کمتری در نهایت داشتیم برای همین نیز خود معیار مناسب بودن در این حالت خیلی جالب برای ما نبود. پس ما روش دوم را در نظر میگیریم. بنا بر ویدیو این فرض که روشنایی پیکسل های جابجا شده به صورت عادی ثابت میباشد فرض خوبی میباشد پس ما از همان اطلاعات مرکزمان استفاده کرده ایم. ما گفتیم که اطلاعات مرکز قبلی را نگه داشته ایم و حال در این تابع مراکز جدید را خواهیم داشت پس ما N تا ترک داریم و M تا مرکز در این مرحله میایم و یک ماتریس M در N تشکیل میدهیم درایه های ان برابر با فاصله این مراکز از هم میباشند. برای اینکه به صورت بهینه

[2] https://github.com/srianant/kalman_filter_multi_object_tracking

پیدا کنیم هر مرکز به کدام ترک(بخوانید tube) نسبت باید داده شود باید یک مسئله بهینه سازی انجام شود و چون مسئله از حالت NP hard نمیباشد ما اردر n^3 برای حل ان روش داریم که هر کدام بهترین را به یک کدام مناسب نسبت بدهیم این الگوریتم Hungarian_algorithm میباشد که در کتابخانه scipy برای ما در بخش optimization موجود میباشد بنا بر این روش ما میتوانیم به ماتریسی برسیم که در ان هر row را به col مربوطه نسبت میدهد که row مفهوم ترک و col مفهوم مرکز جدید را دارد اینرا در لیستی موقت ذخیره میکنیم اگر مقداری نسبت داده نشده باشد مقدار انرا -1 قرار میدهیم تا بتوانیم در جلوتر عملیات لازم را روی ان انجام بدهیم. میایم و روی مقادیر نسبت داده شده حرکت میکنیم اگر مقدار -1 باشد با ان کاری نداریم ولی اگر نه باید چک کنیم که ایا این مراکزی که نسبت داده ایم فاصله ای قابل قبول همانگونه که توضیح داده ایم با یکدیگر دارند یا خیر؟ اخر فاصله بیشتر بود ما به طور کاذب مسئله بهینه سازی را برای این نقاط حل کرده این و مقادیر انرا -1 میکنیم حال باید مقادیر فریم هایی که -1 به انها نسبت داده شده را به اطلاعات دیده نشدن در فریم یکی بیفزاییم . برای این روش ما یکبار این کار را امتحان کرده بودیم که به جای دور زدن در ابتدا مسیر های با فاصله بیشتر از اختلاف را با مقادیر خیلی بزرگ جایگزین کنیم و بعد مسئله را حل کنیم ولی در این حالت جواب ها بسیار بد و بی ربط میشد که به ذات مسئله بهینه سازی برمیگردد برای همین نیز این هزینه محاسباتی بیشتر را میدهیم و با یک فور اصلاح میکنیم به جای اینکه از روش دیگر اصلاح را انجام بدهیم.

حال باید tube هایی را که باید حذف کنیم پیدا میکنیم اگر مقدار دیده نشدن فریم بیش از حد مجاز باشد باید انرا حذف کنیم ولی از طرفی اگر طولش به اندازه کافی باشد در کنار حذف کردن باید انرا ذخیره کنیم در اینجا ما index هایی که باید حذف کنیم را ذخیره کنیم چون فرایند حذف کردن سخت میباشد.

در فرایند حذف کردن در ابتدا اینرا بررسی میکنیم ایا اصلا tube برای حذف شدن وجود دارد یا اینکه لازم نیست اینکار را انجام دهیم با این کار در محاسبات صرفه جویی میکنیم اگر حال وجود داشته باشد باید چک کنیم که ایا همه دارند حذف میشوند یا نه که به خاطر اختلاف یکی بین مقدار i در رنج نسبت داده میشود و طول که مفهوم شروع از یک را در بر دارد باعث نشود که به ارور بر بخوریم. حال میایم و به ترتیب فریم ها را حذف میکنیم فقط دقت کنید باید کانتری قرار دهیم که وقتی یکی را حذف میکنیم دفعه بعدی یکی کمتر از مقدار باید حذف شود را حذف کنیم چون مثلا در ابتدا باید 3 و بعد 5 را حذف کنیم(دقت کنید که حذف به ترتیب از کوچک به بزرگ میباشد) اگر 3 حذف شود دیگر جای 5، 5 نمیباشد و در حالت جدید 4 شده است برای همین این کانتر مهم میباشد. حال که حذف مقادیر ضائد را انجام داده ایم باید به ازای مرکز های جدیدی که پیدا شده است ولی به چیزی نسبت داده نشده اند میایم و به فرم کلاسی که داریم Tracked تعریف میکنیم. در هنگام اضافه کردن به این دقت میکنیم که همواره باید ID را در نهایت با یک جمع کنیم برای حالت بعدی که ای دی نو به خود نسبت دهد. حال به مرحله نهایی میرسیم در این بخش ابتدا کار های اصلی را توضیح میدهیم و در جلو تر موارد امتیازی بیان شده است. به ازای ترک هایی که الان منفی یک نمیباشند میایم و اعمالی را برای اضافه کردم فریم حاضر و تنها با در بر داشتن کانتور مورد نظر داشته باشیم. در ابتدا عملگر KF را روی مرکز جدیدی که باید اضافه کنیم انجام میدهیم و مرکز اصلاح شده را بدست میاوریم سپس این را چک میکنیم که ایا از 10 فریم حضور گذشته است یا نه (این مقدار 10 میتواند متغیر باشد با 10 نسبتا جواب خوبی بدست میاید) دلیل اینکه چرا حتما بعد چند فریم انرا مشخص میکنیم در جلوتر بیان میشود. بعد از ان میایم و یک ماسک تولید کرده به طول هر فریم تصویر و در ان بخش های کانتور را به صورت تو پر سفید میکنیم حال در یک متغیر هرجا که ماسک سفید باشد، مقدار فریم را در ان ذخیره میکنیم پس ما الان یک متغیر داریم که دارد برای ما به صورت تئوری تنها تصویر ماشین در بین کلی سیاهی میباشد این را به متغیر Tracked ما اضفه میکنیم و بعد این را که ما این tube را دیتیکت کردیم با صفر کردن مقدار فریم هایی که انرا در ان ندیدیم انجام میدهیم. حال نکته دیگری که وجود دارد این میباشد که باید زمان را به وجود بیاوریم در اینجا از ورودی فریم ریت استفاده میکنیم در هر ثانیه ما به تعداد فریم ریت فریم داریم پس اگر شماره فریم به ما برسد تبدیل ان به زمان برای ما کار راحتی میباشد. حال باید انرا نشان دهیم ما انرا به تصویر تمام سیاه تنها با جزییات ماشین اضافه میکنیم و رنگ انرا سفید قرار میدهیم دقت کنید که بخاطر ذات کانتور و دلایل گفته شده در قبل از اینکه در گوشه کانتور انرا قرار دهیم پرهیز میکنیم و انرا در همان مرکز تحمین زده شده قرار میدهیم. حال ی یک عملگر پایان دهنده تعریف میکنیم اگر ان احضار شود

میایم و روی ترک های فعلی جاروب میکنیم و اگر حضور به اندازه کافی داشته باشند انرا به tube های نهایی شده اضافه میکنیم. حال به مرحله خلاصه سازی میرسیم در ابتدا میایم و طول بیشترین tube را پیدا میکنیم اگر پس زمینه را داشته باشیم تنها به تعداد tube ماکزیمم فریم کافی است. بعد میایم و اختلاف طول هر tube با ماکزیمم مقدار را پیدا میکنیم این مقدار بازه ای است که اگر در ان باشد حضور کاملش را در ویدیو خواهیم دید به نظر میرسد منطقی باشد که بنا براسا اختلاف های یکسان بیایم و زمان شروع را رابطه دار کنیم ولی در این حالت به یک مشکل خوردیم ما امدیم و با فاصله هایی انهایی که طول یکسان اختلاف دارند را نمایش دهیم ولی به علت تداخل های متعدد با بقیه حالات این روش به جواب خوبی برای ما نرسید پس میایم و روش دیگری را بررسی میکنیم. در این روش که متاسفانه قطعیتی برای ایده ال بودن واب ندارد ولی بیس ان منطقی میباشد این است که ما توضیع رندوم در بازه اجازه وجود هر فریم نسبت میدهیم یعنی مثلا اکر طول بیشترین tube برارب 700 باشد و tube فعلی 200 فریم اشغال کند میتوان از یک تا 500 حضور داشته باشد با توضیع یونیفرم نقطه اغازین را برای ان بین 0 تا 500 توضیع میکنیم با این کار باعث میشود یک پخش همگن و یکنواخت در شروع ها داشته باشیم پس یک ماتریس تولید کرده و نقاط اغاز را در ان قرار میدهیم با اینکار و داشتن پس زمینه کدک مربوط به تولید ویدیو را میساریم طول کل ویدیو به اندازه طولانی ترین tube قرار داده شد بعد در یک حلقه for به ازای اینکه شمارنده فعلی چه موقعیتی نسبت به نقطه شروع هرکدام از tube ها قرار دارد اگر مهلت شروع انها رسیده باشد انها را به پس زمینه اضافه میکنیم تا وقتی که طولشان تمام شود. چالشی که در این مرحله به ان بر خوردیم این میباشد که ما با دستور numpy.where نقاطی در فریم tube مقدار دارند را روی پس زمینه میخواستیم بندازیم ولی گاهی در جاهایی بعضی پیکسل های ان سیاه میشد و این خاصیت ناقص میشد برای همین نیز میایم انرا سیاه و سفید میکنیم و یک پلیت 3 بعد ان همان بعد سیاه و سفید میباشند در واقع برای ما اطلاعات پیکسل ها در این مرحله مهم نیست بلکه وجود داشتن و نداشتن پیکسل مهمه برای همین با اینکار این مشکل مرتفع میشود و بعد به راحتی انرا روی پس زمینه تصویر میکنیم.

برای ویدیو اول نتایج نسبتا معقول میباشند ولی برای ویدیو دوم نتیجه خیلی جالب نمیباشد علت این است که سایه شدید در ناحیه لاین چپ(از دید روبرو) باعث شده است که تنظیم پارامتر ها حساس باشد (ما با ازمایش هایی چند باره بهترین پارامتر ها در این مرحله در حدود (250 ~ 270, 25~35,100) بدست اورده ایم(البته مثلا اگر 250 انتخاب کرده اید بهتر است 25 را انتخاب کنید که به مفهوم ظریف نگری در مقابل گم شدن اطلاعات باشد) این پارامتر ها در ویدیو دوم به خوبی عمل نمیکند از طرف دیگر نیز در قسمت های دور به شدت نویزی به نظر میرسد برای همین نیز در تنظیم پارامتر های dialation باید کمتر باشد ولی این به بهای ناکامل ماندن ماشین ها میباشد پس پارامتر های مناسبی برای این حالت وجود به نظر ندارد ولی نتیجه ای حاصل میشود که انرا اورده این دو نمونه از فریم ها تصاویر زیر میباشند.

تصویر 5

تص.یر 6

موارد امتیازی:

نکته مهم: برای اینکه بخش های امتیازی نیز در خروجی اجرا شود باید Tracking ابتدایی را run کرده ولی اگر بدون بخش های امتیازی بخواهید نام کلاس Tracking2 را به Tracking تغییر داده بعد کد را به اجرا در بیاورید

سرعت نسبی:

برای این کار این امر را در نظر میگیریم که هر دو فریم فاصله مرکز ها دارد معیاری از سرعتشان به ما میدهد چون که فریم ریت برای همه ثابت میباشد انگار عنصر t ثابتی دارند. برای جلوگیری از اینکه به خاطر جابجایی های نامعقول مرکز کانتور ها جلوگیری کنیم روی مقدار سرعت یک ترشلد قرار میدهیم و برای حفظ پیوستگی بهتر سرعت میانگین وزن داری با سرعت قبلی بدست امده در فریم قبل را با سرعت فعلی در نظر میگیریم(به نوعی فیتلر پایین گذر اعمال میکنیم) خروجی را در بالای بخش مربوط به زمان به مشابه ان اضافه میکنیم. مشکل اساسی این بخش بعد خاص عکس و طرز حرکت انها میباشد که روی یک خط راست نیست برای همین نیز این عدد تنها نسبی و دقیق نمیباشد.

رنگ خودرو:

در این بخش ما ابتدا سعی کردیم با اینکار که انرا به فضایHSV برده مقدار رنگ را با ماسک های متغیر بازه رنگ هارا جدا کنیم ولی مشکلی مانند رنگ متغیر کامیون ها و چند رنگه بودن انها و طیف وسیع رنگ ها این کار را عملا غیر ممکن میکرد برای همین نیز ما با عملیات محاسباتی در هر tube میایم و از هر سه کانال میانگین بر روی وزن رنگ های ان میگیریم و انرا به صورت یک مستطیل که نمایانگر رنگ میباشد نشان میدهیم. باز از دقت خیلی عالی برخوردار نمیباشد ولی حدودی از رنگ ماشین به ما میدهد

Dense بودن: اینکار را به صورت ضمنی در متن توضیح دادیم و توجیه کردیم که با توضیع نقاط شروع به صورت یکنواخت و با فرض در اردر هم بودن سرعت ها و خیلی متفاوت نبودن توانستیم نسبتا راه کاری برای این امر انجام دهیم.

حاصل این ها به صورت ویدیو های زیر میباشد:

تصویر 7و8

اگر به شکل ها دقت کنید در وسط انها مستطیل رنگی میبینید که برای نشان دادن رنگ ها به کار رفته است.

خط های دنبال کننده مسیر و شمارش ماشین ها:

برای اینکار در ابتدا باید دقت را به شدت زیاد کنیم ظرافت این امر بسیار بالا میباشد چون برحسب مرکز کانتور ها که گاهی خیلی دور میتوانند باشند از مرکز ماشین داریم قضاوت میکنیم پس دقت را افزایش میدهیم میایم و تابعی مینویسم برای محاسبه زاویه این امر درنهایت برای ما خیلی مهم خواهد بود. ما اندازه ها را نیز حساب میکنیم و چون انتظار داریم بهترین ترک ها طول خوبی را داشته باشند یک ترشلد دیگر نیز در اینجا قرار داده و بهترین ها را جدا میکنیم حال روی فاز انها که KNN میزنیم و به دو کلاستر زاویه ها را تبدیل میکنیم (چون با دانش قبلی میدانیم تنها دو مسیر داریم) حال میایم و فرض میکنیم بهترین مسیر با بیشترین طول میباشد و روی ان با نقطه شروع و زاویه ای که داریم و طول ان نقطه انتهایی را نتیز با یک رابطه مثلثاتی ساده حساب میکنیم بدیهی است که جون به فرم int در میاوریم دقت ان از دست میرود ولی این بهایی است که باید پرداخت کنیم. حال به خاطر اینکه تحمین را خیلی حساس کرده ایم میگوییم از هر 3 ماشین یک ماشین به بیرون راه پیدا کرده و نسبت کانتور های حذف شده و بعد را با ترشلد روی اندازه در نظر میگیریم با فرض اینکه نرخ ماشین ها در هر دو سمت یکسان میباشد این نسبت در 3 در tube هایی که در هر مسیر قرار گرفته اند(بر حسب tag که در KNN به ما میدهد تشخیص میدهیم) تعداد هرکدام را نیز پیدا کرده و رسم میکنیم:

تصویر 9

برای ویدیو دوم خیلی جالب نمیباشد هرچند حدود درستی دارد و میتوان حدس زد تحت تاثیر کدام ماشین میباشد(تعداد ماشین های هر مسیر با شماره مشخص شده است) ولی برای ویدیو اول:

تصویر 10

که یک مسیر را بسیار عالی و مسیر دیگر را به نسبت خوب تشخیص داده است

نکته: دو section نهایی کد برای این بخش ها به کار رفته است و تنها خروجی این بخش را میدهد چون پارامتر های ترشلد برای این دو کمی تفاوت دارد برای همین در دو بخش بخش اول برای ویدیو اول و بخش دوم برای ویدیو دوم حتما دقت کنید که هر بخش را با پس زمینه مربوطه بسنجید بهتر است در این بخش از Tracking2 استفاده کنید چون محاسبات کمتری دارد ولی پارامتر های لازم را برای این امر فراهم میکند