به نام خدا

پروژه پردازش تصویر((درس تخمین پارامتر))

دانشگاه امیرکبیر

دانشکده هوافضا

امیر حسین نقدیان

نام استاد: دکتر سبزه پرور

برای تشخیص حرکت اجسام میتوانیم از کتاب خانه های مختلفی در openCV استفاده کنیم، الگوریتم Lucas-Kanade گزینه مناسبی برای optical flow است فقط باید به نکات زیر توجه کنید.

ابتدا نقاط مناسی برای track اشیا باید به پیدا کنید اصطلاحا به این عملیات corner detection گفته میشود. الگوریتم انتخابی برای این بخش Shi Tomasi است.

توابع مجاز هر بخش در openCV به این صورت است:

1- **cv2.calcOpticalFlowPyrLK()**
2- **cv2.goodFeaturesToTrack()**

پارامتر های ورودی هر کدام از توابع بالا در pdf مربوط به کد توضیح داده شده است.

حتما توجه داشته باشید فریم هایی که به این توابع پاس میدهید نباید به صورت چند بعدی باشند پس فریم های دریافتی را به صورت زیر به gray scale تبدیل میکنیم:

```
cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
```

Dense optical flow

در روش dense برای تشخیص حرکت اشیاء بر خلاف روش Luca Kanade ما از تمام نقاط می خواهیم استفاده کنیم و سپس object tracking را انجام دهیم و از همه مهم تر در این روش میخواهیم intensity بین دو فریم را به دست بیا وریم.

برای انجام این قسمت باید مراحل زیر را طی کنیم:

1- پارامتر های ورودی در قسمت pdf کد توضیح داده شده است.
2- فریم اول فیلم را خوانده و به gray scale تبدیل میکنیم برای این کار از تابع مربوطه استفاده میکنیم.

```
cv2.cvtColor(frame1,cv2.COLOR_BGR2GRAY)
```

3- حتما باید یک mask به صورت hsv-color تعریف کنیم و saturation را برابر 255 قرار میدهیم توضیح کامل مدل hsv در لینک زیر موجود است:
https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&cad=rja&uact=8&ved=2ahUKEwjyn4m6h7brAhVxQRUIHZUoBdEQFjAAegQIDRAB&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FHSL_and_HSV&usg=AOvVaw1ZPPY_-vZ9sHQYhSswXtuD

4- از تابع مربوطه برای dense optical flow استفاده میکنیم می توانیم مقادیر پیش فرض را برای این تمرین اعمال کنیم :

```
cv2.calcOpticalFlowFarneback(prvsImg , nextImg , None , 0.5 , 3 , 15 , 3 , 5 , 1.2 , 0)
```

5- خروجی این تابع به صورت flow object است کاری که ما باید در این بخش انجام دهیم این است که با استفاده از تابع مربوطه خروجی را به polar coordinate تبدیل کنیم. به عبارتی دیگر چون مدل ما به صورت مخروطی است پس باید یک مقدار saturation و یک زاویه به دست آوریم.

```
mag, ang = cv2.cartToPolar(flow[: , : , 0] , flow[: , : , 1] , angleInDegrees=True)
```

6- حالا میتوانیم با مشخصات به دست آمده mask را مقدار دهی کنیم.
7- قبل از نمایش فیلم حتما تصویر به دست آمده را به مدل BGR تبدیل کنید.

GMG

در کتاب خانه openCV برای به دست آوردن پس زمینه فیلم میتوانیم از این مدل استفاده کنیم. این مدل ترکیبی از دو روش Bayesian interface و Kalman filters است.

برای به دست آورده پس زمینه دو مرحله انجام می شود.

1- برای هر پیکسل بسته به آن که مقادیر رنگی چه مدت باقی به ماند مقدار دهی انجام میشود. مقادیر رنگی که مدت زمان بیشتری باقی بمانند به عنوان background انتخاب میشوند.
2- برای کاهش نویز در مرحله ی اول مقادیر پیش زمینه باید فیلتر میشود.
3- در حالت اجرا باید کمی منتظر بمانید تا تغیرات الگوریتم اعمال شود.

مقادیر ورودی:

initializationFrames:

قسمت اول مشخص میکند برای به دست آوردن پس زمینه از چه تعداد فریم باید استفاده شود. هر چقدر تعداد فریم ها بیشتر باشد مدل اولیه در این فاز تثبیت شده تر است. فریم نهایی همیشه سیاه است.

:decisionThreshold

در این قسمت مقدار Threshold پیکسل ها که به عنوان پس زمینه یا پیش زمینه طبقه بندی میشوند مشخص میشود. همان طور که در قسمت قبل اشاره شد در فاز اول پیکسل ها جمع آوری میشوند تا مقادیر رنگی از لحاظ پایداری مشخص شوند. تمام مقادیری که مقدار آن‌ها از Threshold کمتر باشند به عنوان پس زمینه انتخاب میشوند. اگر مقدار بالایی برای این بخش انتخاب شود ممکن objects در هر فریم از بین بروند.

به این مسئله حتما توجه داشته باشید که GMG به Noise حساس است پس برای کاهش noise از توابع مربوطه در openCV استفاده کنید اما سایز kernel را باید به درستی انتخاب کنید در غیر این صورت اشیاء متحرک تصویر را از دست می‌دهید.

```
kernel = cv2.getStructuringElement(cv2.MORPH_ELLIPSE,(3,3))
cv2.morphologyEx(mask, cv2.MORPH_OPEN, kernel)
```

MOG

در این مدل از k Gaussians distributions استفاده میشود. به این صورت که تفاوتی که در distribution مشخص میشود پیش زمینه و پس زمینه را در تصاویر طبقه بندی میکند.

مقادیر ورودی:

History:

تعداد فریم‌ها را در این مدل مشخص میکند و هر چقدر مقدار آن پایین بیشتر باشد حساسیت مدل نسبت تغیرات ناگهانی نور بالا میرود.

Nmixtures:

تعداد توزیع گاسی را مشخص میکند هرچقدر مقدار این پارامتر بیشتر باشد زمان اجرا هم به شدت افزایش میابد.

backgroundRatio:

مقدار Threshold برای طبقه بندی پس زمینه و پیش زمینه مشخص میکند. مقدار کم ممکن است الگوریتم را در زمینه تشخیص اشیا با مشکل رو به رو کند.

noiseSigma:

پارامتر آخر هم سطح نویز قابل قبول را مشخص میکند.

MOG2

این الگوریتم برای رفع مشکلات اگوریتم MOG ارئه شد ((یک مقدار ثابت برای توزیع ها استفاده میشد)). این الگوریتم عملکرد بهتری در مورد پیچیدگی رنگ ها در هر frame ارائه کرد.

مقادیر ورودی:

History:

مانند الگوریتم قبل تعداد فریم‌ها را مشخص میکند.

varThreshold:

مقدار وزن پیکسل‌ها را در فریم فعلی با مقادیر موجود ارتباط میدهد. مقادیر کمتر در این مدل فریم ممکن است object را به خوبی تشخیص ندهد.

detectShadows:

برای تشخیص سایه در فریم ها استفاده میشود. فعال کردن این بخش ممکن است زمان اجرا را بالا ببرد.

مقایسه الگوریتم ها:

از دو جهت میتوانیم این مقایسه را انجام دهیم:

1- فریم ها درست طبقه بندی شده اند یا خیر
2- زمان اجرا.

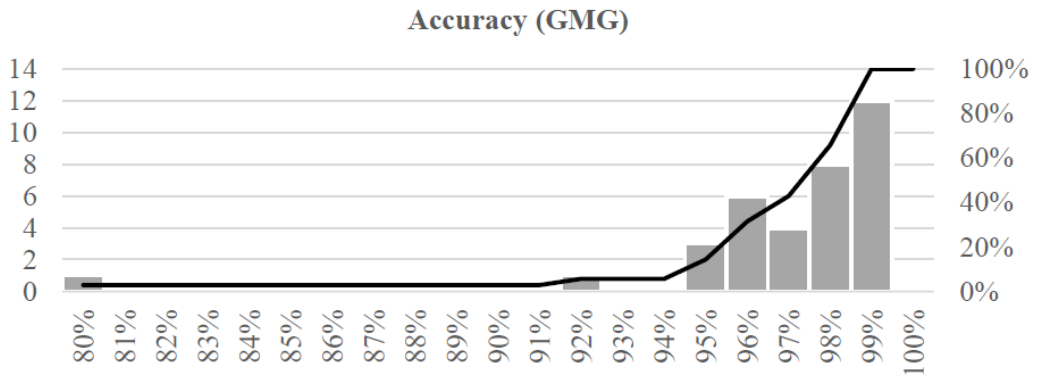
می توانیم با تشکیل Confusion Matrix طبقه بندی را در دیتای مورد نظر برسی کینم.

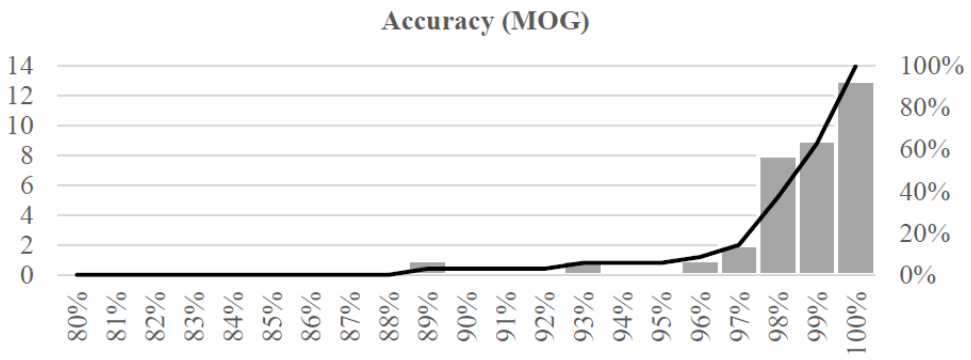
Table 1 – Confusion Matrix and utilized metrics.

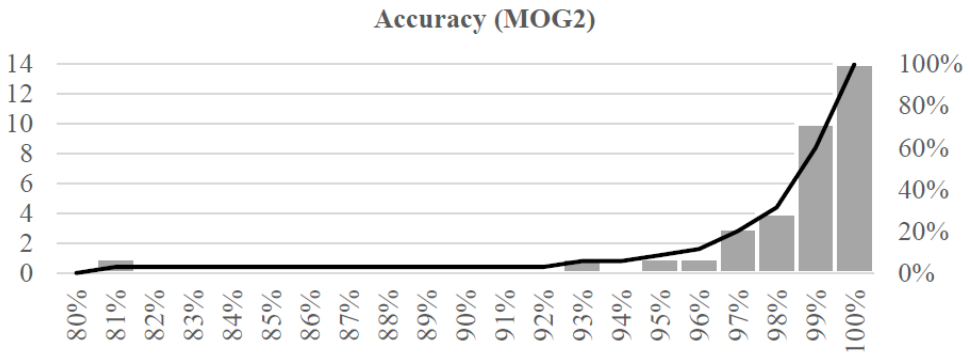
Model	Ground-Truth	
	Foreground (Positive)	Background (Negative)
Foreground (Positive)	TP	FP
Background (Negative)	FN	TN
Total	P	N

$Accuracy = \frac{VP+VN}{P+N}$
$Precision = \frac{VP}{VP+FP}$

Accuracy و Precision به دست آمده را برای مقایسه هر کدام از این الگوریتم‌ها میتوان استفاده کرد.

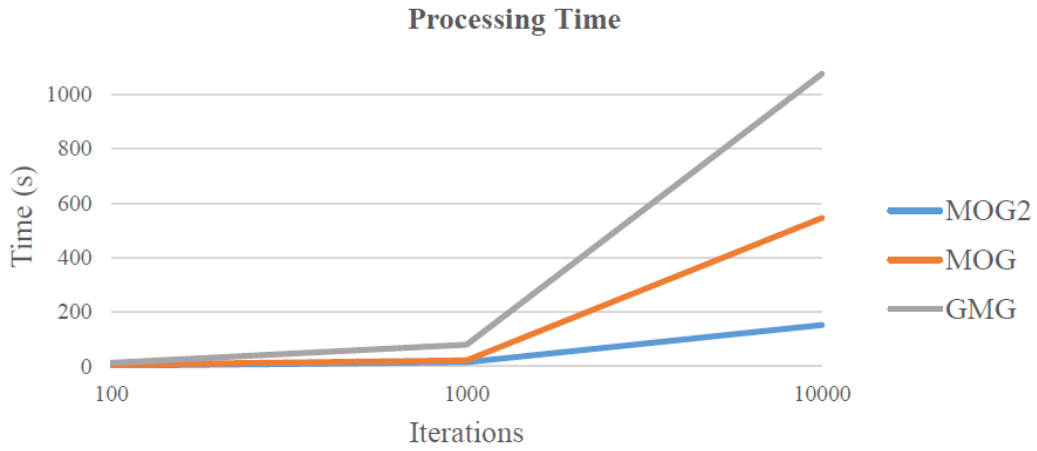




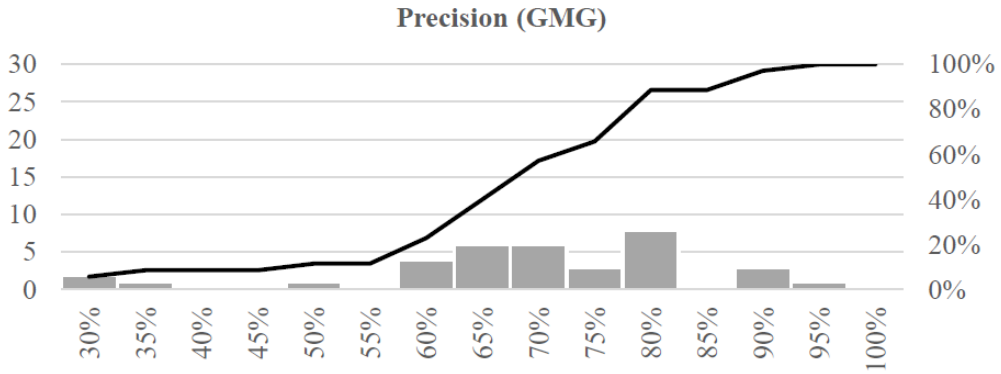


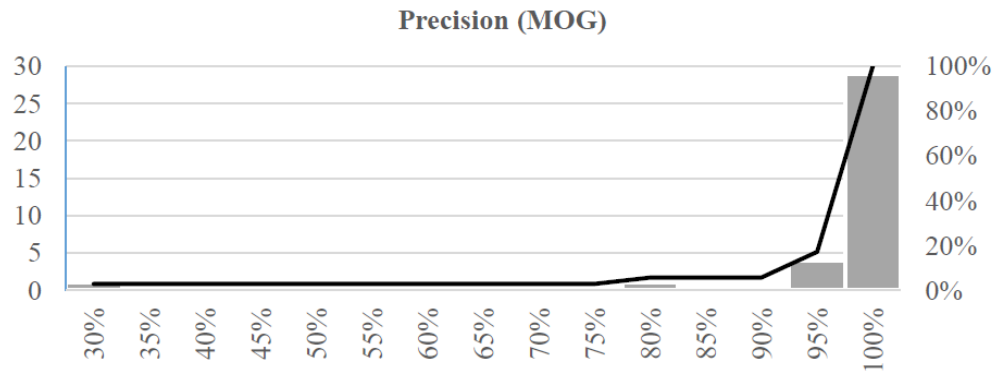
نتایج اشاره شده بسته به ویژگی های دیتا ممکن است متفاوت باشد ولی در حالت کلی میتوانیم به عملکرد هرکدام از الگوریتم ها پی ببریم.

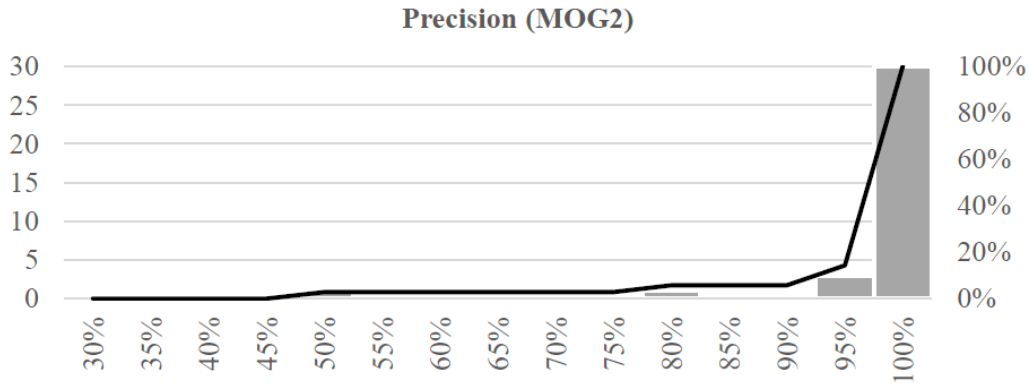
همان طور که ملاحظه میکنید با برسی accuracy نمی توانیم نتیجه گیری کنیم زیرا هر کدام از این الگوریتم‌ها دقت مناسبی را ارائه می دهند.



در زمان اجرای کد این مسئله کاملا بدیهی است که الگوریتم GMG زمان اجرای بالایی دارد







از لحاظ درستی طبقه بندی، الگوریتم GMG نسبت به noise کمی حساس است و ممکن است داده ها را به درستی طبقه بندی نکند. اما الگوریتم MOG2 و MOG عملکرد بهتری در طبقه بندی داده ها از خود نشان دادند.