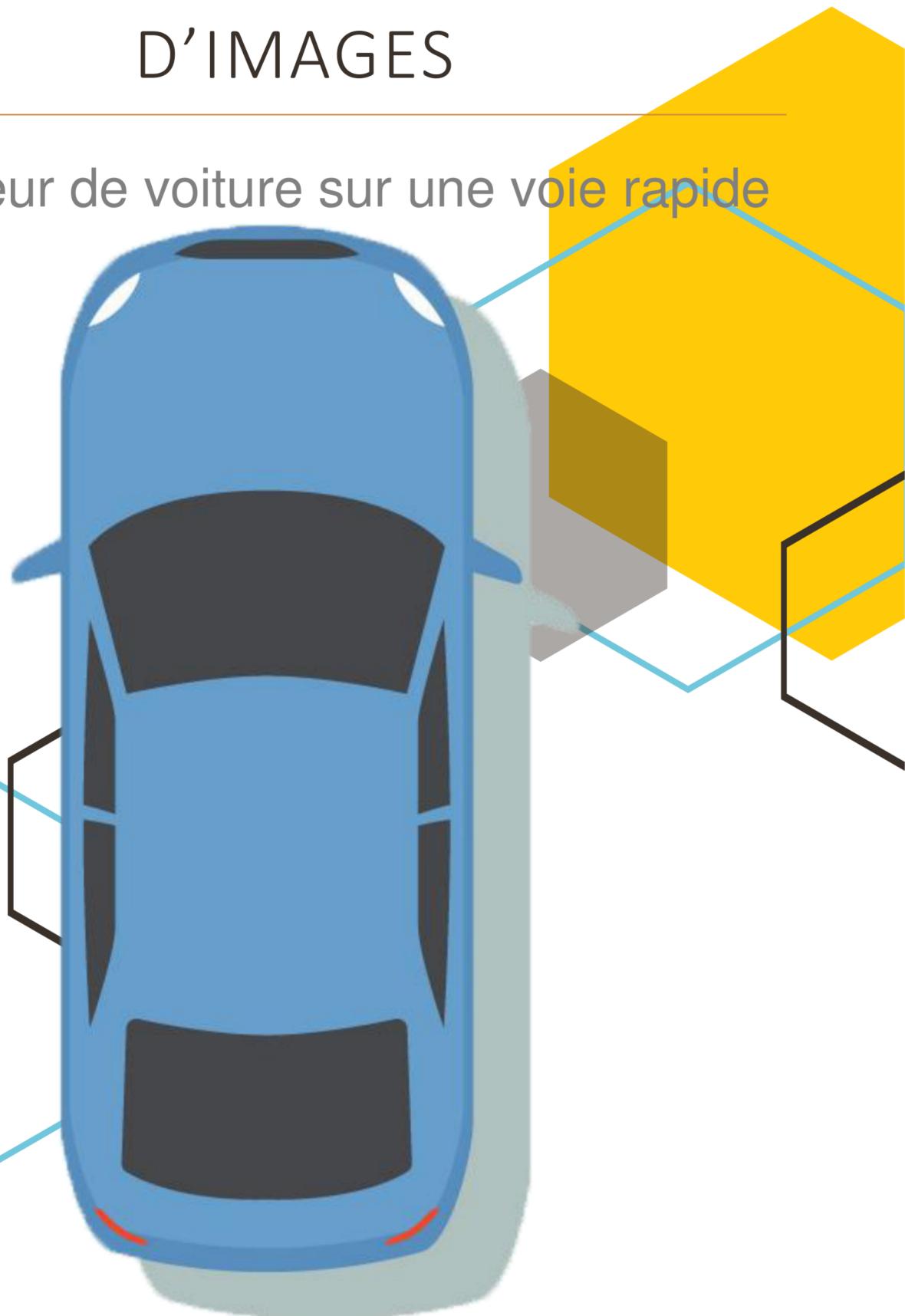


---

# TP N°2 : TRAITEMENT D'IMAGES

---

Compteur de voiture sur une voie rapide



# TABLE DES MATIÈRES

1	Introduction.....	1
2	Détection et Isolation des deux voies rapides.....	1
2.1	Isolation de la couleur verte.....	1
2.2	Amélioration de l'image .....	2
2.3	Tracer des bordures .....	3
3	Suivi de véhicule : .....	4
3.1	Différences d'images.....	4
3.2	Détection de véhicule.....	5
3.3	Ajout d'un carré rouge autour des véhicules .....	6
3.4	Compteur de véhicule .....	6
4	Conclusion .....	8
5	Table des légendes .....	8

## 1 Introduction

L'objectif de ce TP est de fournir une application capable de compter le nombre de voiture passant sur deux voies rapides. Pour atteindre cet objectif, il va nous falloir détecter les voies, dans un premier lieu. Ensuite, il nous faudra repérer et suivre les véhicules dans l'objectif final de les compter.

## 2 Détection et Isolation des deux voies rapides

### 2.1 Isolation de la couleur verte

Afin de détecter les voies rapides, l'idée première est de chercher à isoler la couleur correspondant à ces dernières. Cependant, les couleurs composants la route ne sont pas homogènes, et sont donc compliquées à isoler. Pour contrer ce problème, nous avons remarqué que le reste de l'environnement s'apparente à la couleur verte. Nous avons donc sélectionné la couleur verte à l'aide de la fonction :

```
int low_H = 20;  
int low_S = 25;  
int low_V = 25;  
int high_H = 85;  
int high_S = 255;  
int high_V = 255;  
  
inRange(m_hsv, Scalar(low_H, low_S, low_V), Scalar(high_H, high_S, high_V), m_filtered);
```

Ensuite, nous avons inversé les pixels pour faire apparaître la route :

```
bitwise_not(m_filtered, m_filtered_inv);
```

Nous avons obtenu le résultat suivant :

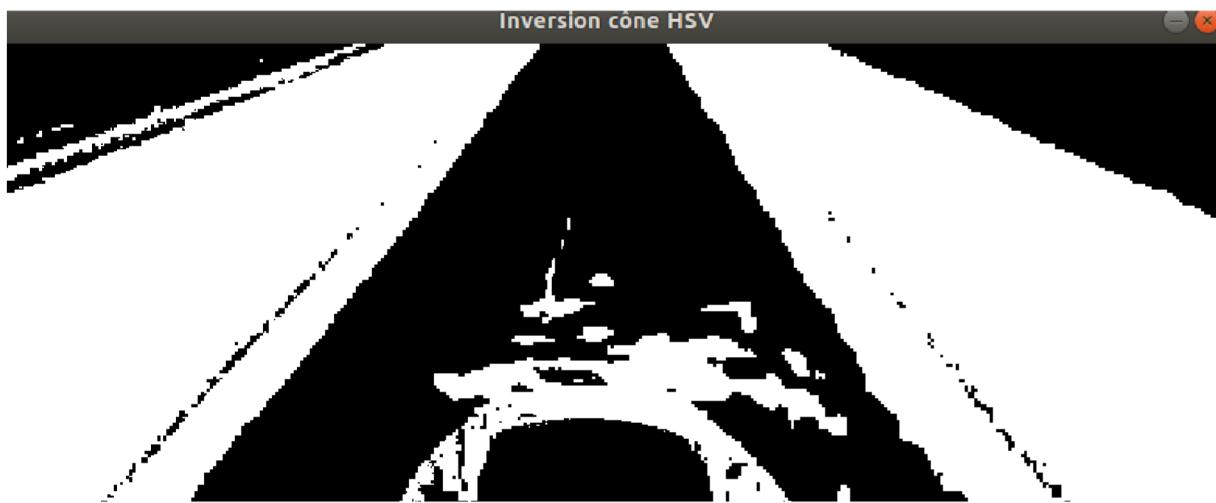


Figure 1: première détection des voies rapides

## 2.2 Amélioration de l'image

Même si à l'œil il nous est possible de reconnaître clairement les deux voies, l'image était toujours composée de résidus indésirables. Pour les retirer nous avons procédé à une fermeture suivie d'une ouverture :

```
morphologyEx(m_filtered_inv,m_closed,MORPH_CLOSE,getStructuringElement(MORPH_ELLIPSE,Size(10, 10)));
```

```
morphologyEx(m_closed, m_opened, MORPH_CLOSE, getStructuringElement(MORPH_ELLIPSE, Size(12, 12)));
```

Nous avons obtenu le résultat suivant :



Figure 2: Voies rapides après fermeture + ouverture

On remarque que l'image est beaucoup plus nette et désormais exploitable.

## 2.3 Tracer des bordures

Par la suite, il nous fallait démarquer ces voies à l'aide de bordures. Pour ce faire, nous avons récupéré les contours, dans un premier temps :

```
Canny(m_opened, m_edge, 0, 0, 3);
```



Figure 3: isolation des contours

Puis nous n'avons conservé uniquement les lignes droites :

```
HoughLinesP(m_edge, m_road, 1, 3.14/180, 60, 100, 100);
```

Enfin nous les avons tracé :

```
for (size_t i = 0; i < m_road.size(); i++) {  
    Vec4i l = m_road[i];  
    line(m_frame_final, Point(l[0], l[1]), Point(l[2], l[3]), Scalar(0, 0, 255), 3, LINE_AA);  
}
```

Nous obtenons le résultat suivant :



Figure 4: Voies rapides bordée de lignes

Le résultat était très satisfaisant. En effet, les voies rapides sont bien délimitées par des bordures.

### 3 Suivi de véhicule :

#### 3.1 Différences d'images

Pour suivre les véhicules, nous avons décidé de procéder par différence d'images puisque nous avions une image de base constante. Nous avons dupliqué les images pour les comparer entre elles :

```
Mat m_Sframe_compt = m_frame.clone();

if(first_image2 == 1){

    m_frame_initial = m_frame.clone();
    cvtColor(m_frame_initial, m_Sgray_initial, COLOR_BGR2GRAY);
}

first_image2++;
```

Après avoir binarisé l'image nous avons fait la différence entre les deux à l'aide de la fonction :

```
absdiff(m_Sgray_initial, m_Sgray, m_Sdiff);
```

Le flux vidéo ressemblait alors à :



Figure 5: détection de mouvement

Le résultat était très satisfaisant. Nous pouvions passer à l'étape suivante à savoir repérer les véhicules.

### 3.2 Détection de véhicule

Pour se faire nous voulions isoler les contours des véhicules. A l'images de la première partie nous avons procédé de la manière suivante :

Application d'un flou pour rendre la détection de contours plus simple :

```
medianBlur(m_Sdiff, m_Sgauss,5);
```

Binarisation + fermeture + dilatation + ouverture :

```
threshold(m_Sgauss, m_Sbinarize, 12, 255, THRESH_BINARY);

morphologyEx(m_Sbinarize, m_Sopen, MORPH_OPEN, getStructuringElement(MORPH_ELLIPSE, Size(4,4)));
morphologyEx(m_Sopen, m_Sdilate, MORPH_DILATE, getStructuringElement(MORPH_ELLIPSE, Size(5,5)));
morphologyEx(m_Sdilate, m_Sclose, MORPH_CLOSE, getStructuringElement(MORPH_ELLIPSE, Size(3, 3)));
```

A ce stade nous avions ce flux :

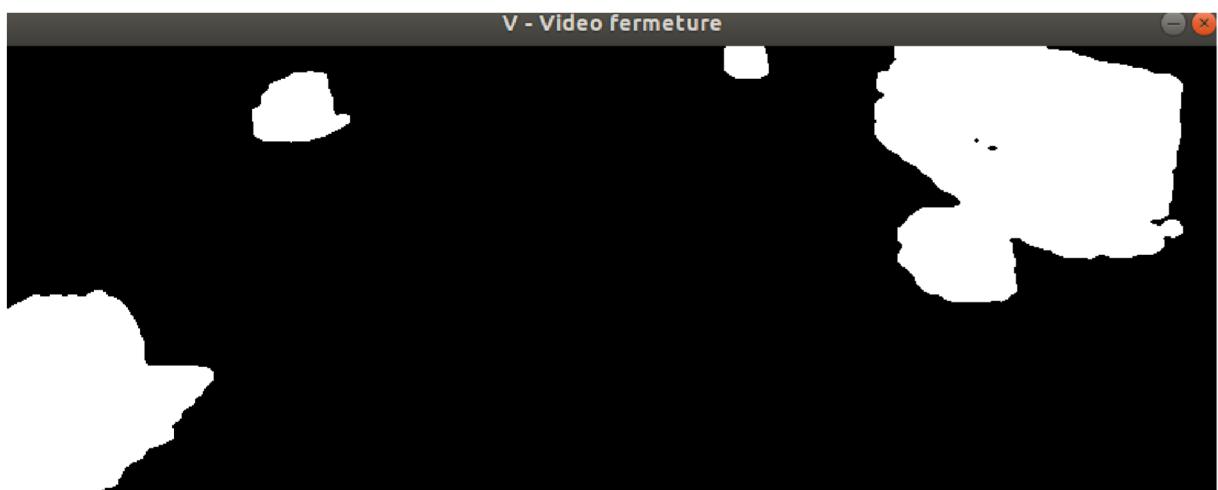


Figure 6: Isolation des véhicules sur les voies rapides

Les véhicules étaient bien isolés, nous pouvons en tracer et repérer les contours :

```
Canny( m_Sclose, m_Scanny, 100 , 200);

findContours(m_Scanny, contours, hierarchy, RETR_TREE, CHAIN_APPROX_SIMPLE );
```

### 3.3 Ajout d'un carré rouge autour des véhicules

Pour chaque contour nous avons récupéré son centre. De plus, si la taille du contour était suffisamment élevée, nous associons le centre du carré rouge à celui du contour.

```
approxPolyDP( contours[i], contours_poly[i], 3, true );

boundRect[i] = boundingRect( contours_poly[i] );
int x = boundRect[i].x + boundRect[i].width/2;
int y = boundRect[i].y + boundRect[i].height/2;
if(arcLength(contours[i], true) > 120){

    Point center;
    center.x = x;
    center.y = y;
```

### 3.4 Compteur de véhicule

Enfin, comme chaque contour correspond à un véhicule nous incrémentions un compteur pour suivre le nombre de véhicule passant sur la voie rapide. Plus précisément, à l'aide des centres, nous détectons si le véhicule est sur la voie de droite ou celle de gauche. Nous avons donc 2 compteurs distinct.

```
int* count = &count_gauche;

if (center.x > m_frame.size().width / 2){

    count = &count_droite;
}

if(center.y <= (120 + range) && center.y >= (120 - range)){

    *count += 1;
    cout << "gauche = " << count_gauche << " droite =" << count_droite << endl;
}
```

Finalement, nous avons obtenu le flux ci-dessous :

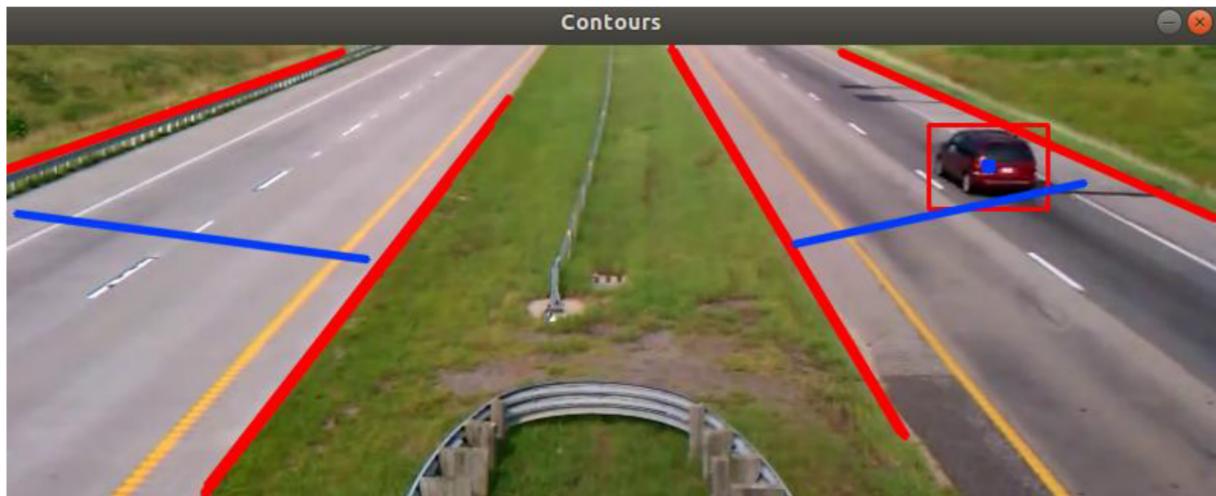


Figure 7: Rendu final, suivi+encadrement des véhicules

Le véhicule est bien tracké, le carré rouge ainsi que le point bleu, représentant le centre du contour, sont bien présents.

Pour ce qui s'agit du comptage nous avons le résultat suivant :

```
martin@martin-VirtualBox: ~/tp_traitement (copie)/TP2
Fichier Édition Affichage Rechercher Terminal Aide
gauche = 9 droite =2
gauche = 10 droite =2
gauche = 10 droite =3
gauche = 10 droite =4
gauche = 10 droite =5
gauche = 10 droite =6
gauche = 11 droite =6
gauche = 12 droite =6
gauche = 13 droite =6
gauche = 13 droite =7
gauche = 13 droite =8
gauche = 14 droite =8
gauche = 15 droite =8
gauche = 16 droite =8
gauche = 16 droite =9
gauche = 16 droite =10
gauche = 16 droite =11
gauche = 16 droite =12
gauche = 16 droite =13
gauche = 17 droite =13
gauche = 17 droite =14
gauche = 17 droite =15
Unable to read device
```

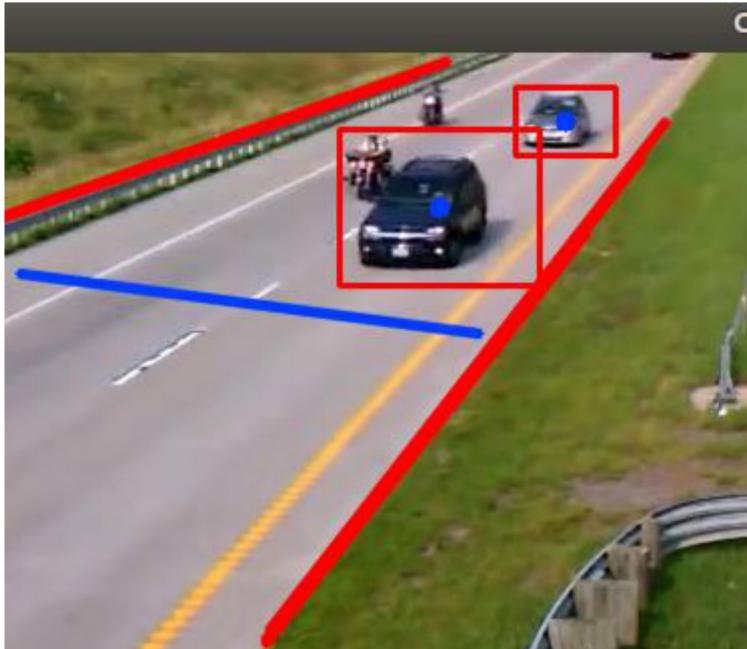
Figure 8: Résultat du comptage automatique

Nous comptons 17 véhicules à gauche et 15 à droite.

## 4 Conclusion

Après comptage manuel du nombre de véhicules nous sommes censés trouver :

17 véhicules à gauche et 9 à droite. Malgré que le compte de gauche soit correcte, en réalité, notre programme compte plusieurs fois certains véhicules et en omet d'autres. En effet, l'impact de cette faille est nettement visible en comparant les résultats de la voie de droite.



Ci-contre nous pouvons observer que la voiture noire et la moto sont confondus, il n'en résulte alors qu'un seul contour.

Figure 9: exemple de limite de notre méthode

En conclusion, notre technique de comptage comporte des limites malgré un fonctionnement parfois satisfaisant. Utiliser les contours pour isoler les véhicules n'est pas sans faille. D'autre méthodes telles que, repérer et compter les régions ou encore implémenter une intelligence artificielle pourraient conduire à de meilleurs résultats.

## 5 Table des légendes

Figure 1: première détection des voies rapides .....	1
Figure 2: Voies rapides après fermeture + ouverture .....	2
Figure 3: isolation des contours .....	3
Figure 4: Voies rapides bordée de lignes .....	3
Figure 5: détection de mouvement.....	4
Figure 6: Isolation des véhicules sur les voies rapides .....	5
Figure 7: Rendu final, suivi+encadrement des véhicules .....	7
Figure 8: Résultat du comptage automatique.....	7
Figure 9: exemple de limite de notre méthode .....	8