

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique  
Université des Sciences et de la Technologie Houari Boumediene  
Faculté d'électronique et d'informatique  
Département d'informatique



# Rapport de projet

Module : Vision artificielle par ordinateur

Master 1 SII

**Réalisation d'une application de painting  
en utilisant une détection de couleurs**

- Réalisé par :

**BENHADDAD Wissam**

**BOURAHLA Yasser**

29-11-2018

# Table des matières

<b>Table des matières</b>	<b>2</b>
<b>Table des figures</b>	<b>2</b>
<b>Liste des tableaux</b>	<b>1</b>
<b>1 Introduction</b>	<b>2</b>
1.1 Objectifs et problématique . . . . .	2
1.2 Définitions . . . . .	2
1.2.1 Une image sous différents angles . . . . .	2
1.2.2 Notion de video . . . . .	2
1.2.3 Opération sur les images . . . . .	2
1.3 Conclusion . . . . .	3
<b>2 Solution proposées et implémentation</b>	<b>4</b>
2.1 Outils utilisés . . . . .	4
2.1.1 Environnement de travail . . . . .	4
2.1.2 Langage . . . . .	4
2.1.3 OpenCV . . . . .	4
2.1.4 Qt framework . . . . .	4
2.2 Schéma global du système . . . . .	4
2.3 Détection de couleurs . . . . .	5
2.4 Élimination du bruit par regroupement . . . . .	6
2.5 Conclusion . . . . .	8
<b>3 Présentation de l'application</b>	<b>9</b>
3.1 Diagramme d'utilisation . . . . .	9
3.2 Exemples d'utilisation . . . . .	9
3.3 Limitations . . . . .	9
<b>4 Conclusion générale</b>	<b>10</b>

# Table des figures

2.1 Schéma du système . . . . .	5
---------------------------------	---

2.2	Exemple centre de détection . . . . .	6
2.3	Exemple centre de détection avec bruit . . . . .	6
2.4	Exemple distance entre deux carrés collés . . . . .	7
2.5	Distance entre deux pixels représentant l'unité de mesure . . . . .	7
2.6	Exemple de coté d'un groupe de pixels . . . . .	7

## Liste des tableaux

# CHAPITRE 1

## INTRODUCTION

### Objectifs et problématique

Talk about why we are gonna do this stuff.

### Définitions

Some definitions of the concepts, techniques and algorithms that we will use

### Une image sous différents angles

Talk about how an image can be represented (Color spaces, data structures) BRIEFLY

### Notion de video

A video is a succession of images at some frame rate, lol

### Opération sur les images

Notion de filtrage et convolutions

**Filtre moyen**

**Filtre médiane**

**Érosion**

**Dilatation**

NOT SURE IF WE'RE GONNA TALK ABOUT ALL OF THESE ( BAH N3AMROU JEDDOU XD )

**Conclusion**

## CHAPITRE 2

# SOLUTION PROPOSÉES ET IMPLÉMENTATION

## Outils utilisés

Environnement de travail

Langage

OpenCV

Qt framework

## Schéma global du système

Le système se compose principalement de trois modules :

- D'abord le module de détection récupère l'image de la caméra pour y détecter les couleurs recherché et génère des informations sur le curseurs i.e sa position et son état.
- Le module de dessin reçoit les informations sur le curseur et les utilise pour générer l'image du dessin.
- Les deux modules précédents envoient leurs résultats à l'interface graphique pour l'affichage.

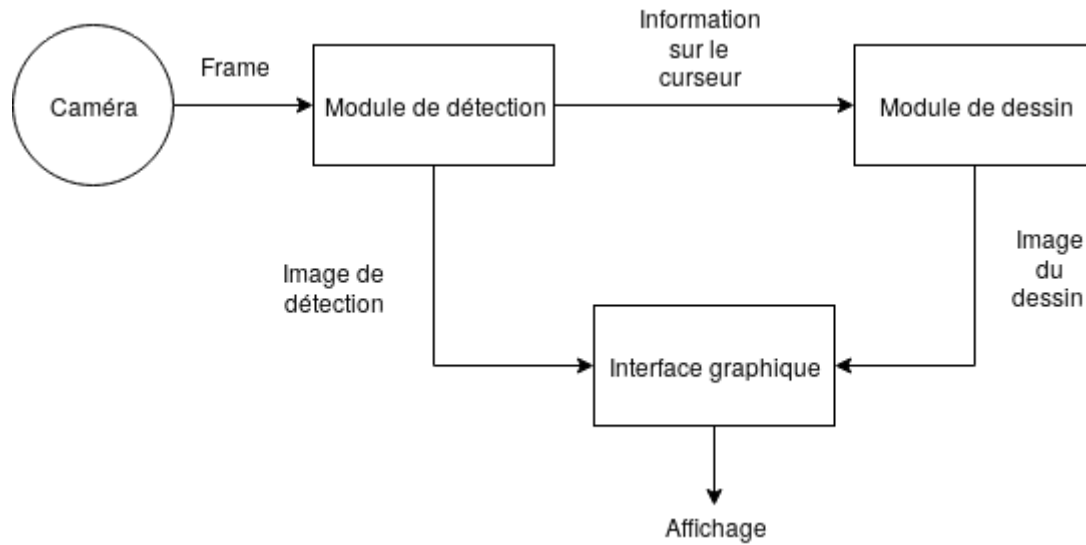


FIGURE 2.1 – Schéma du système

## Détection de couleurs

La détection de couleurs se fait sur une image en HSV, ceci est dû au fait que HSV sépare la valeur de teinte de celle de la luminosité ce qui le rend plus robuste envers les changements de lumière.

Ainsi la détection d'une des deux couleurs se fait en passant par tous les pixels de l'image et décider pour chaque pixel s'il a la couleur recherchée ou non. La décision se fait en testant l'appartenance du pixel à un intervalle de détection. Pour chacune des valeurs H, S, V du pixel, on vérifie si cette valeur appartient à un intervalle défini lors du choix de l'utilisateur de la couleur à détecter comme suit :

Soit la couleur choisie  $H1S1V1$ , et le pixel candidat HSV, les conditions que doit vérifier le pixel sont :

- $H1 - t \leq H \leq H1 + t$  : l'intervalle de teinte dépend fortement de la couleur sélectionnée.
- $S1 * 0.6 \leq S \leq 255$  : l'intervalle de saturation dépend légèrement de la couleur sélectionnée.
- $20 \leq V \leq 255$  : On accepte un intervalle de luminosité large.

Après avoir sélectionné les pixels qui ont la couleur désirée, on calcule le centre de ces pixels. Ainsi la détection résultera en deux centres, un pour chaque couleur. La distance entre ces deux derniers détermine si le curseur est actif. Si la distance est supérieure à un seuil donné le curseur est désactivé, il est actif sinon. Quant à la position du curseur elle est représentée par le centre des deux points sus-cités.

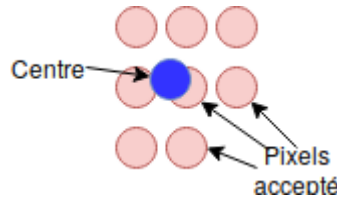


FIGURE 2.2 – Exemple centre de détection

**Le problème** qui se pose en se limitant à la détection par intervalle c'est qu'elle est extrêmement sensible au bruit. Un pixel aberrant jugé appartenir à l'intervalle va fausser la position du centre, et ainsi un suivi médiocre de la couleur.

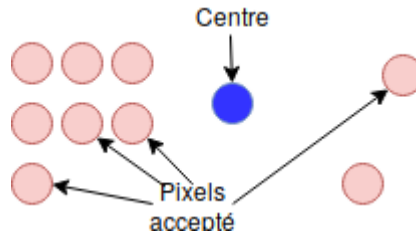


FIGURE 2.3 – Exemple centre de détection avec bruit

## Élimination du bruit par regroupement

Pour remédier au problème rencontré. Nous avons opté à éliminer le bruit en regroupant les pixels proches qui sont acceptés. L'algorithme de regroupement fonctionne comme suit :

- Il prend en paramètre un seuil minimum pour jugé que deux pixels appartiennent au même groupe.
- Au début chaque pixel est dans un groupe à part.
- L'algorithme passe par plusieurs itérations de regroupement, dans une itération chaque groupe est fusionné avec un autre dont la distance est inférieur au seuil modifié\* s'il existe, la fusion des groupes n'est pas parallèle, un groupe peut être fusionné avec un autre créé au cours de la même itération.
- L'algorithme s'arrête quand les groupes ne changent pas après une itération.

**seuil modifié\*** : Comme le seuil donné en entrée est défini pour regrouper deux pixels, il est donc inadapté pour le fusionnement de deux groupes contenant plus d'un pixel. Le seuil doit être donc ajusté en fonction de la taille des groupes. Nous avons choisi d'utiliser la formule suivante :  $nouveauSeuil = (seuil/2) * (racine(tailleG1) + racine(tailleG2))$  avec  $tailleG1$  et  $tailleG2$  les tailles des groupes un et deux respectivement.

L'intuition derrière cette formule, c'est que la distance entre deux centres de carrés collés est égale à  $1/2 * cote1 + 1/2 * cote2$ , avec  $cote1$  et  $cote2$  les cotés des deux carrés.



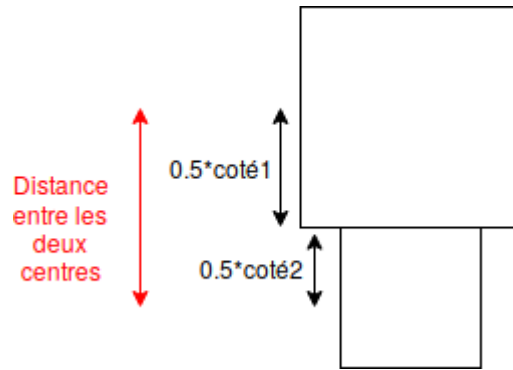


FIGURE 2.4 – Exemple distance entre deux carrés collés

Dans notre cas, la taille du côté d'un pixel est l'unité de mesure de distance. La distance entre deux pixels peut être écrite en fonction de la taille de son côté comme suit :  $(distance/2) * (cotep + cotep)$  avec  $cotep$  la valeur d'un côté de pixel, comme  $cotep=1$  (l'unité de mesure) la formule précédente se réduit en la valeur de distance.

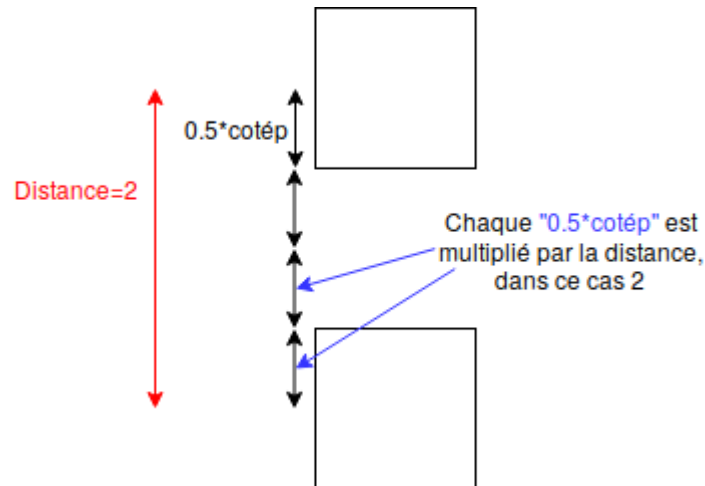


FIGURE 2.5 – Distance entre deux pixels représentant l'unité de mesure

Si on projette cela sur un groupe de pixels carré, la taille de son côté est égale à la racine du nombre de pixels, car le nombre de pixels représente la surface du carré, ainsi la formule devient :  $(distance/2) * (racine(tailleG1) + racine(tailleG2))$ .

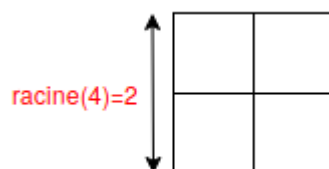


FIGURE 2.6 – Exemple de côté d'un groupe de pixels

## Conclusion

Kheliahli rak ta3ref xD

## CHAPITRE 3

### PRÉSENTATION DE L'APPLICATION

**Diagramme d'utilisation**

**Exemples d'utilisation**

**Limitations**

CHAPITRE 4

CONCLUSION GÉNÉRALE