République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Mouloud Mammeri de Tizi Ouzou
Faculté du Génie Electrique et Informatique
Département d'informatique

Mémoire de fin d'études
En vue de l'obtention du diplôme Master
Domaine : **Mathématiques et Informatique**
Filière : **Informatique**
Spécialité : **Réseaux, Mobilité et Systèmes Embarqués**

Thème :

« Reconnaissance automatique de plaque d'immatriculation véhicules pour smart parkings »

Soutenu le : 22 juillet 2019

Présenté par :
Mlle AIT BENAMAR Katia Dehbia
Mr. OUHEB Massinissa

Proposé et dirigé par :
Mr. M. DAOUI
Membres du jury :
Mr. I. FILALI
Mme R. HADAOUI

Promotion 2018/2019

Remerciements

Nous tenons à remercier chaleureusement notre promoteur Mr DAOUI pour son suivi et ses orientations et lui exprimer notre reconnaissance pour le temps qu'il nous a consacré.

Nous remercions également les membres du jury d'avoir accepté de juger notre travail.

Nous exprimons notre profonde gratitude à tous les enseignants du département informatique ainsi qu'aux enseignants de la spécialité RMSE.

Nous remercions aussi Mr ALIANE pour tout ce qu'il fait pour les étudiants.

Nos remerciements vont enfin à nos deux familles ainsi qu'à toute personne ayant contribué de près ou de loin à l'élaboration de ce modeste travail.

Dédicace

« Je dédie ce travail à mes chers parents, je ne les remercierai jamais assez pour tous leurs sacrifices, à mes chers frères Rayan et Mohand, à mes tantes et à tous mes proches. »

Katia Dehbia

« Je dédie ce modeste travail à mes chers parents qui m'ont soutenu jusqu'à la fin, à mes chers frères et sœurs, à tous ceux qui me sont proches, à Zaidi Bilal qui nous a aidé dans la réalisation de la maquette »

Massinissa ⵎⴰⵙⵙⵉⵏⵉⵙⴰ

À tous nos amis (Tous les membres de la GDG et WTM Tizi Ouzou, Thanina, Mira, Kenza, Koukou, Sara, Meli, Momo, Yacine, Silia, Samah, Fifi, Moh, Aghiles, Rabah, Massiouen, Ania, Rafik, Djoudjou, Dihia, Amel, Zaki, Sonia, Rymouch, Djouher, Hichem, Yasmine Asma, Arezki, Chahrazed Loubna, Sihem, Meryouma).

SOMMAIRE

Table des figures

Table des tableaux

Liste des abréviations

ML : Machine Learning

KNN : k-Nearest Neighbor

SVM : Support Vector Machine

EDM : Euclidian Data Mining

CRM : Customed Relationship Management

OCR : Optical Caracter Recognition

RSB : Rapport Signal Bruit

LAN : Local Area Network

FTP : File Transfer Protocol

HTTP : HyperText Transfer Protocol

XML : eXtensible Markup Language

POP : Post Office Protocol

IMAP : Internet Message Access Protocol

ALPR : Automatique Licence plate Recognition

ARM : Advanced RISC Machine

RISC : Reduced instruction set computing

RPi : Raspberry pi

LDR: Light Dependent Resistor

LED : Light Emitting Diode

BSD : Berkeley Software Distribution

OS : Operating System

SQL : Structured Query Language

SGBD : Système de Gestion de Bases de données

Introduction générale

De nos jours, la hausse du nombre de véhicules induite par la croissance rapide de la population dans les zones urbaines a provoqué l'augmentation considérable de la demande d'infrastructures de stationnement. Les parkings de voitures sont actuellement une des ressources stratégiques d'une ville intelligente car ils facilitent la mobilité des personnes. Les conducteurs doivent toujours stationner pour se rendre à leurs lieux de destination. Ces systèmes sont généralement gérés par l'homme, ce qui induit des difficultés dans leur gestion et des problèmes de sécurité.

Le problème des parkings déjà existants est la gestion des véhicules, les véhicules ne sont pas identifiés automatiquement et il n'y a pas de système d'accès ne nécessitant aucune action du conducteur : appuis sur bouton, demande de tickets, ...etc. Aussi un des problèmes majeurs est la perte de temps, le conducteur n'est pas sûr de trouver effectivement une place libre, la recherche d'une place libre lui prend du temps et de l'attention. Un aspect important de la problématique est la sécurité, aucun moyen n'est mis en place pour récupérer l'historique des entrées-sorties.

Actuellement, il existe plusieurs techniques de stationnement automatisées conçues à l'aide de technologies telles que la reconnaissance automatique de plaques d'immatriculation (Automatic Licence Plate Recognition en anglais), les réseaux de capteurs, etc. Les systèmes de parking intelligent incluent également des moyens pour louer une place de stationnement et automatiser le paiement pour le temps de stationnement. Le numéro d'immatriculation d'un véhicule représente un moyen efficace pour les identifier, vue que c'est une information unique pour chaque voiture. Ainsi, il est primordial d'identifier automatiquement les plaques d'immatriculation des véhicules pour la gestion et la sécurité d'un parking. En particulier, cette information peut être utilisée pour permettre l'accès automatique au parking, elle peut être aussi utilisée pour d'autres raisons, comme la surveillance des passages aux frontières et aux péages, la rechercher des véhicules suspects ou encore la lutte contre la criminalité. Ceci rend l'identification automatique de ces plaques décisive et inévitable à travers un système de reconnaissance des plaques d'immatriculation.

Un système ALPR est composé généralement en trois grandes parties :

- Acquisition de l'image,
- Localisation de la plaque d'immatriculation dans l'image,
- Identification des numéros de la plaque (reconnaissance).

L'objectif principal de notre travail s'inscrit dans l'optique de réaliser un module de reconnaissance de plaques d'immatriculation des véhicules entrants et sortants pour un parking intelligent. Ainsi, le parking sera automatisé, un portail s'ouvre automatiquement dans le cas où des places sont disponibles après la lecture de matricule. Ensuite, un compteur de temps permet de récupérer le temps de stationnement afin de calculer le montant à payer à la sortie. D'autres capteurs sont utilisés pour la gestion du Parking (identification des places libres, aiguillage des conducteurs, etc…).

Notre mémoire s'articule sur les quatre (4) chapitres suivants :

Chapitre 1 : Vue d'ensemble du Machine Learning : Dans ce chapitre nous parlerons d'une manière générale de l'apprentissage automatique et de ses différents algorithmes puisque nous avons utilisé un algorithme ML. Nous avons fait appel à ces techniques pour la reconnaissance des caractères d'une plaque d'immatriculation.

Chapitre 2 : Traitement d'image : Ce chapitre est consacré aux différents processus utilisés dans le traitement d'image et la détection de contours.

Chapitre 3 : Conception : vise à illustrer concrètement les différentes étapes de conception de notre projet et la démarche utilisée dans l'extraction et la reconnaissance de la plaque d'immatriculation.

Chapitre 4 : Réalisation : Nous avons cité l'environnement de développement utilisé pour la réalisation de notre système ainsi que de son fonctionnement, puis présenté quelques tests.

Chapitre 1

Vue d'ensemble du Machine Learning

1.1 Introduction

L'image dans une machine est représentée comme un tableau de nombres indiquant des informations sur chaque pixel : luminosité, couleur, etc... . Des méthodes complexes sont nécessaires pour la reconnaissance d'objets dans les images où l'apparence de l'objet en question peut varier infiniment et peut être altérée par des interférences. Il n'est donc pas possible de réaliser un algorithme classique de reconnaissance d'objets qui permet d'obtenir des résultats exacts dans toutes les situations. C'est pourquoi nous opterons pour réaliser notre objectif pour le machine learning (se traduisant par apprentissage automatique).

Quand on parle d'apprentissage, deux stratégies principales existent : l'apprentissage déductif et l'apprentissage inductif. Dans l'apprentissage déductif, on démarre d'une connaissance que l'enseignant transmet à l'apprenant, ce dernier acquière ce savoir théorique et le consolide à travers des applications, des travaux pratiques, etc… . L'enseignement inductif quant à lui démarre de l'observation, de l'expérience et du questionnement. Dirigé par l'enseignant, l'apprenant abouti à des connaissances nouvelles, fait les relations entre les observations et les résultats obtenus et déduit progressivement des règles. C'est dans cette catégorie que s'inscrit le machine learning. [22]

Dans ce chapitre, nous définirons ce qu'est le machine learning, ses fondements, ses domaines d'application, la différence de cette approche avec l'approche classique, quelques types, quelques algorithmes du machine learning les plus utilisés, leurs fonctionnements et enfin leurs avantages et leurs limitations.

1.2 L'apprentissage automatique

L'apprentissage automatique est une branche de l'intelligence artificielle (IA). L'intelligence artificielle est axée sur la compréhension de l'intelligence humaine et sur la façon de la répliquer dans des machines (systèmes ou agents). ML vise la découverte automatique des régularités dans les données grâce à l'utilisation d'algorithmes informatiques et à leur généralisation dans des données nouvelles mais similaires. Son objectif principal est l'étude et la conception de systèmes pouvant « apprendre à partir des données » et son objectif est l'apprentissage inductif (apprentissage par des exemples). [23]

Étant donné une tâche T et une mesure de performance P, on dit qu'un programme informatique apprend à partir d'une expérience E si les résultats obtenus sur T, mesurés par P, s'améliorent avec l'expérience E. [1]

1.3 Fondements de l'Apprentissage Automatique

L'apprentissage automatique est fondé sur des méthodes qui proviennent de diverses sciences, notamment :

- L'informatique.

- Les mathématiques : algèbre linéaire, la logique, etc…

- La théorie des probabilités.

- La théorie statistique de l'estimation.

- L'inférence grammaticale. [3]

1.4 Domaines de l'Apprentissage Automatique

L'apprentissage automatique est utilisé dans de nombreux domaines, principalement le data mining et l'intelligence artificielle.

- Le data mining (la fouille de données) est le processus d'extraction de la connaissance à partir d'une très large quantité de données stockées dans un grand nombre de bases de données (hétérogènes ou homogènes). Il utilise des algorithmes complexes combinant le machine learning, les systèmes de base de données, les statistiques et ceci dans le but de construire un modèle qui lui sera exploitable et plus intéressant. [24]

Exemples :

- Application Educational Data Mining (EDM) qui extrait des informations à partir de données dans le milieu éducatif. Il permet de prédire les résultats d'un étudiant et de prendre des décisions sur quoi et comment enseigner ainsi que de développer de nouvelles techniques d'enseignement.

- Application Customer Relationship Management (CRM) qui a pour but d'acquérir et de fidéliser les clients à une entreprise et cela à partir de toutes les données collectées et analysées sur ces derniers (les clients). [25]

- Applications de l'intelligence artificielle, tels que : la robotique, la vision par ordinateur, l'extraction des contenus sémantiques à partir des images et des vidéos.

Exemples :

- Voiture autonome (sans conducteur).

- Logiciel de reconnaissance de plaques d'immatriculation.

1.5 L'approche Machine Learning

Dans l'approche traditionnelle (classique), un problème n'étant pas simple, le programme permettant de le résoudre devient une longue liste de règles complexes assez difficiles à maintenir.

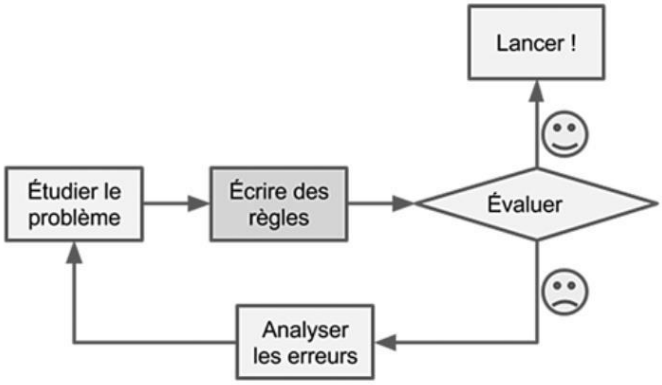


Figure 1 Approche traditionnelle [2]

Par contre, avec des techniques de machine learning, le programme apprend automatiquement quels sont les éléments qui constituent de bons prédicteurs.
Par exemple, un programme qui identifie les courriels frauduleux en détectant des associations de mots aux fréquences inhabituelles dans des exemples de courriels frauduleux comparés à des exemples de courriels licites, le programme est beaucoup plus court, plus facile à maintenir et a plus de chances de fournir de bons résultats.

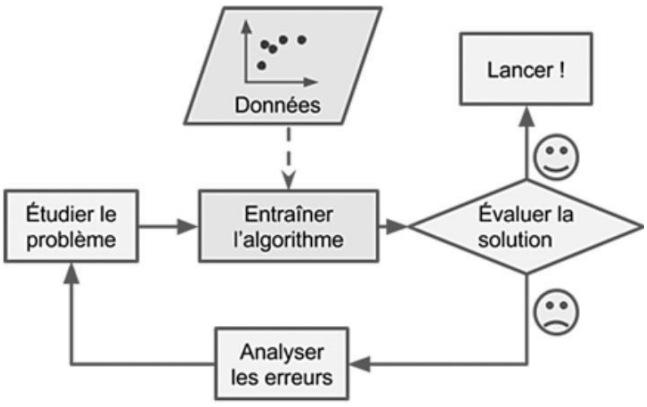


Figure 2 Approche de Machine Learning [2]

L'apprentissage automatique est excellent pour :

- Les problèmes pour lesquels les solutions existantes requièrent beaucoup d'ajustements manuels ou de longues listes de règles : un algorithme d'apprentissage automatique permet de simplifier le code en donnant de meilleurs résultats.
- Les problèmes complexes qui ne trouvent pas de solutions avec l'approche traditionnelle.
- Les environnements qui subissent beaucoup de changements : un système d'apprentissage automatique peut s'adapter à de nouvelles données.
- L'exploration des problèmes complexes et des gros volumes de données. [2]

1.6 Types de systèmes d'apprentissage automatique

Il existe de nombreux types de systèmes d'apprentissage automatique, mais si on les classe en fonction du type de supervision qu'il requièrent durant la phase d'entrainement, les deux principaux types sont l'apprentissage automatique supervisé et l'apprentissage automatique non supervisé.

La différence majeure entre les deux types est le fait que l'apprentissage supervisé se fait sur la base de modèles d'entrée et de sortie connues. C'est à dire, nous avons une connaissance de ce que devrait être les valeurs de sortie de nos échantillons au préalable. Par conséquent, l'apprentissage supervisé vise à apprendre la fonction qui, à partir d'un échantillon de données connu et des résultats souhaités, permet d'avoir la relation observable entre entrée et sortie.

Par contre, l'apprentissage non supervisé n'a pas de données d'entrée ni de résultats préalablement connus. Il doit donc déduire les caractéristiques et la structure de la population d'entrée.

1.6.1 Apprentissage supervisé

Les algorithmes de type apprentissage supervisé permettent de résoudre un problème à partir de données d'entrainement qui comportent les résultats souhaités.

Ils traitent deux principaux types de problèmes : les problèmes liés à la classification et ceux liés à la régression.

Les problèmes de classification surviennent quand l'objectif est d'avoir une sortie de type « Catégorie ». Par exemple, une application qui permet de prédire si un patient a une maladie précise à partir de ses données médicales, la sortie est donc soit « malade » ou « pas malade ».

Les problèmes de régression surviennent quand la variable de sortie est une valeur réelle, telle que « devise » ou « poids ». Par exemple, une application qui permet de prédire la hausse de la devise.

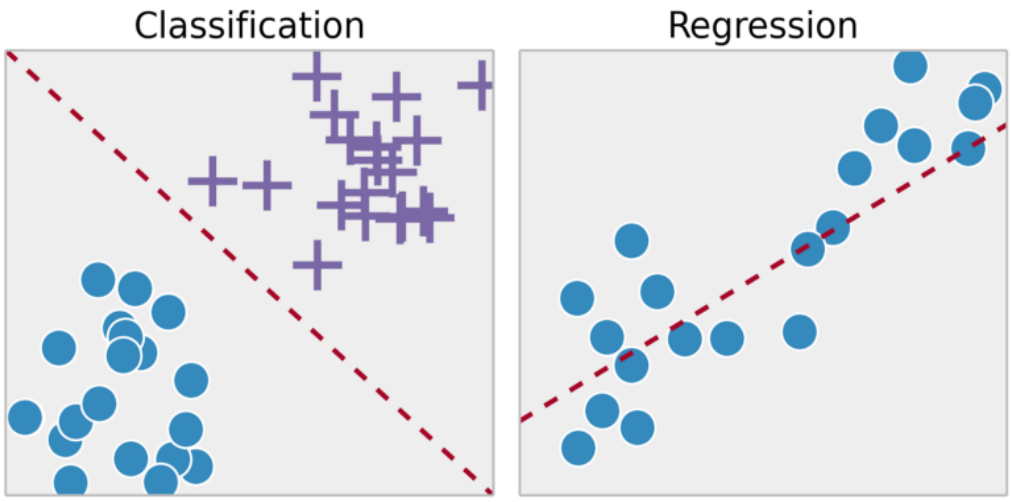


Figure 3 Classification et régression [2]

Dans la régression et la classification, l'algorithme se base sur les données d'apprentissage pour trouver des relations spécifiques dans les données d'entrée qui permettront de produire des données de sortie correctes. Les sorties résultantes dépendent directement des données d'apprentissage. Cependant, même si le modèle est supposé vrai, cela ne veut pas dire que les résultats sont toujours corrects dans des situations réelles. Les étiquettes de données bruyantes ou incorrectes réduisent l'efficacité de l'algorithme.

Parmi les algorithmes les plus importants dans l'apprentissage supervisé, nous citons : les K plus proches voisins (K-NN), les arbres de décisions, les forêts aléatoires, les réseaux de neurones, la régression linéaire et la régression logistique.[3]

1.6.2 Apprentissage non supervisé

Les tâches les plus courantes dans l'apprentissage non supervisé sont le regroupement, l'apprentissage de la représentation et l'estimation de la densité. Dans tous ces cas, l'objectif est de trouver la structure des données sans utiliser d'étiquettes fournies explicitement. Certains algorithmes courants comprennent la mise en cluster, l'analyse en composantes principales et les auto-encodeurs. Étant donné qu'aucune étiquette n'est fournie, il n'existe aucun moyen spécifique de comparer les performances du modèle dans la plupart des méthodes d'apprentissage non supervisées.

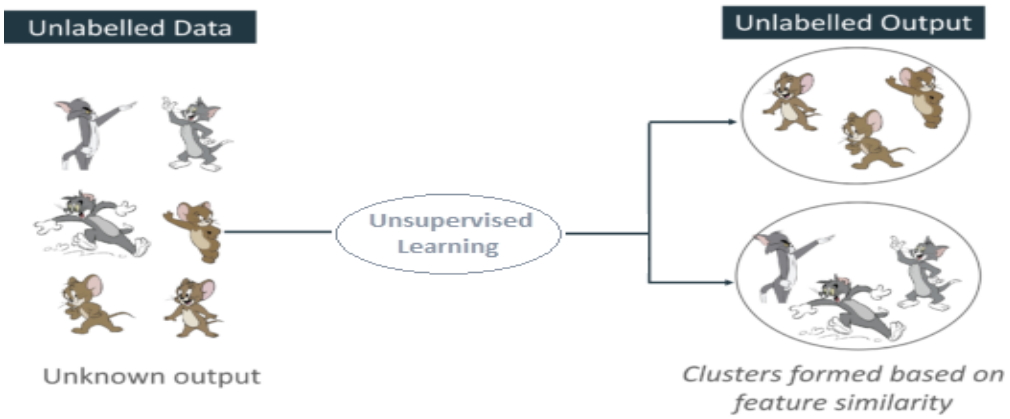


Figure 4 Unsupervised Learning [3]

L'analyse exploratoire et la réduction de la dimensionnalité constituent deux cas d'utilisation courants pour l'apprentissage non supervisé. L'apprentissage non supervisé est très utile en analyse exploratoire car il permet d'identifier automatiquement la structure dans les données. Par exemple, si un analyste essayait de segmenter les consommateurs, des méthodes de classification non supervisées constitueraient un excellent point de départ pour leur analyse. Dans les situations où il est impossible ou peu pratique pour un humain de proposer des tendances dans les données, un apprentissage non supervisé peut fournir des informations initiales qui peuvent ensuite être utilisées pour tester des hypothèses individuelles.

La réduction de la dimensionnalité, qui fait référence aux méthodes utilisées pour représenter des données en utilisant moins de colonnes ou d'entités, peut être obtenue par le biais de méthodes non supervisées. Dans l'apprentissage de la représentation, nous souhaitons apprendre les relations entre les caractéristiques

individuelles, ce qui nous permet de représenter nos données en utilisant les caractéristiques latentes qui relient nos caractéristiques initiales. Cette structure est souvent représentée avec beaucoup moins de fonctionnalités qu'au départ, ce qui rend le traitement des données beaucoup moins intensif et élimine les fonctionnalités redondantes. [4]

1.7 Les principaux algorithmes

Chaque problème présente des spécificités. Il faut trouver la méthode d'apprentissage la plus efficace pour chaque type de problème. Il faut aussi trouver le nombre d'exemples d'entrainement qu'il faut fournir à un programme d'apprentissage pour être certain qu'il apprenne avec efficacité.

Avant d'implémenter un algorithme de machine learning pour un problème donné, il faut étudier son domaine de validité, son domaine d'application et son fonctionnement. Ce qui suit est une brève liste d'algorithmes de machine learning les plus utilisés, leurs principes de fonctionnement, leurs contextes d'utilisation, leurs avantages et leurs limitations.

1.7.1 La régression linéaire

La régression linéaire est un algorithme simple de machine learning supervisé excellent comme exemple pédagogique. La fonction de prédiction qui relie les variables prédictives $x_1, \ldots, x_p$ au résultat cible a la forme : $f(x) = a_1 x_1 + \ldots + a_p x_p$. L'apprentissage du modèle consiste à calculer les coefficients a et b qui minimisent les erreurs de prédiction sur un ensemble de données d'apprentissage. Le plus souvent l'erreur est définie comme la somme des carrés des écarts entre les valeurs prédites $f(x^{(i)})$ et les valeurs observées $y^{(i)}$. Le fait de prendre le carré évite la compensation entre les erreurs positives et négatives.[28]

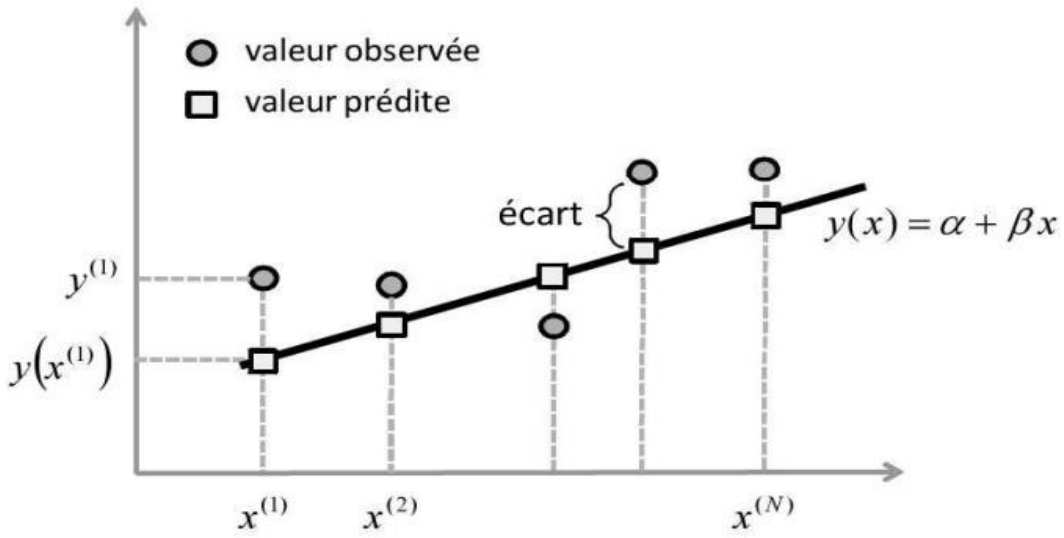


Figure 5 La régression linéaire [2]

Le principal avantage est que le modèle est en général simple à interpréter puisqu'il existe une expression mathématique explicite qu'il suffit de calculer, aucun algorithme numérique complexe n'intervient. Aussi, le calcul d'une prédiction est très rapide. Toutefois il n'est applicable que s'il existe une relation linéaire, il est aussi sensible aux bruits et aux erreurs.

1.7.2 La régression logistique

La régression logistique est un modèle de classification linéaire utilisé quand la variable cible y n'a que deux valeurs possibles, par exemple 0 ou 1. Comme il s'agit d'un modèle linéaire, on utilise une fonction de score S des variables prédictives : $S(x) = a_1x_1 + \ldots + a_px_p$. L'idée est de chercher des coefficients $a_1, \ldots, a_p$ tel que le score $S(x)$ soit positif lorsque les chances d'appartenir au groupe 1 sont grandes et tel que le $S(x)$ soit négatif lorsque les chances d'appartenir au groupe 0 sont grandes.

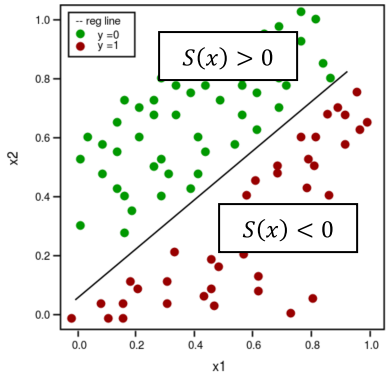


Figure 6 La régression logistique [3]

Bien qu'il soit simple et rapide, la phase d'apprentissage peut s'avérer longue car l'opération numérique d'optimisation des coefficients $a_i, \ldots, a_p$ est complexe. L'algorithme est aussi a priori limité aux variables cibles binaires. [6]

1.7.3 Les k plus proches voisins

L'algorithme des k plus proches voisins (K-NN pour K Nearest Neighbors) est un algorithme d'apprentissage supervisé utilisé notamment pour la classification. L'idée est que chaque observation de l'ensemble d'entraînements représentée par un point dans un espace comportant autant de dimensions qu'il y a de variables prédictives, on suppose une notion de distance dans cet espace et l'on cherche les points les plus proches de celui que l'on souhaite classer. [5]

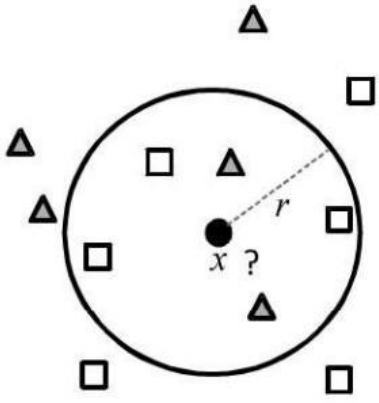


Figure 7 Les k plus proches voisins [3]

Avantages

• On ne présuppose rien sur la distribution des données si ce n'est qu'il existe une notion de similarité.

• Conception relativement simple.

Inconvénients

• L'algorithme est très sensible au bruit.

• L'apprentissage est uniquement local et ne tient pas compte des effets collectifs à longue distance.

• Lorsque le nombre de variable est grand (> 10), le calcul de la distance peut s'avérer très coûteux.

1.7.3.1 Choix de la valeur K

Pour sélectionner le K qui convient, il est nécessaire d'exécuter plusieurs fois l'algorithme K-NN avec différentes valeurs de K et choisir le K qui réduit le nombre d'erreurs rencontrées tout en maintenant la capacité de l'algorithme à prédire avec précision de nouvelles données.

Quelques points importants à prendre en compte : Lorsque nous diminuons la valeur de K à 1, nos prévisions deviennent moins stables. Supposons que $K = 1$ et que nous ayons à classer un caractère, par exemple le caractère « 7 » et qu'il soit entouré de plusieurs caractères « 7 » et d'un caractère « 1 », mais que ce « 1 » est le plus proche voisin. Dans ce cas, puisque $K = 1$, K-NN prédit de manière incorrecte que le caractère est un « 1 ».

Inversement, à mesure que nous augmentons la valeur de K, nos prévisions deviennent plus stables en raison du vote à la majorité de la moyenne, l'algorithme est donc plus susceptible de faire des prévisions plus précises (jusqu'à un certain point).

Finalement, nous commençons à être témoin d'un nombre croissant d'erreurs. C'est à ce stade que nous savons que nous avons poussé la valeur de K trop loin.

Dans les cas où nous votons à la majorité parmi les étiquettes (par exemple, en choisissant le mode dans un problème de classification), nous faisons généralement de K un nombre impair pour avoir un départage.

1.7.4 Les arbres de décision

Les arbres de décision sont des algorithmes de l'apprentissage supervisé utilisables aussi bien pour la classification que pour la régression. Les arbres de décision ne reposent pas sur un modèle probabiliste mais sur des méthodes algorithmiques.

L'idée de base consiste à classer une observation au moyen d'une succession de questions concernant les valeurs des variables prédictives x_i de cette observation. Chaque question est représentée par un nœud de l'arbre de décision, chaque branche sortante du nœud correspond à une réponse possible à la question posée. La classe de la variable cible est alors déterminée par la feuille dans laquelle parvient l'observation à l'issue de la suite de questions.

La phase d'apprentissage consiste donc à trouver les bonnes questions à poser pour ranger correctement un ensemble $x^{(1)},..,x^{(N)}$ de N observations dotées d'étiquettes $y^{(1)},..,y^{(N)}$, mais aussi en minimisant le nombre de ces questions. Les variables explicatives en entrée peuvent être aussi bien qualitatives que quantitatives.

Les arbres de décisions présentent de nombreux avantages puisqu'ils prennent compte des interactions entre les variables et que le nombre de classes résultantes n'est pas limité. Cependant, le critère de segmentation affecté au premier nœud possède une très grande influence sur le modèle de prédiction, si bien qu'un jeu de données peu représentatif pourra mener à des conclusions totalement erronées.

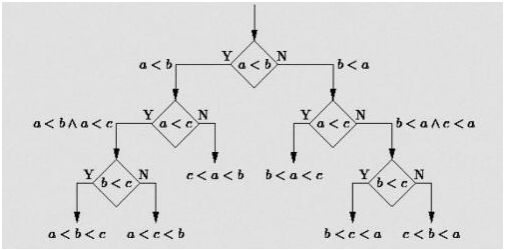


Figure 8 Arbre de décision [3]

1.7.5 Les machines à vecteurs de support

Les SVM sont des algorithmes de classification binaire non linéaires extrêmement puissants. Le principe des SVM consiste à construire entre deux groupes d'observations une bande séparatrice non linéaire de largeur maximale et à l'utiliser pour faire des prédictions. L'astuce consiste à trouver une transformation φ non linéaire qui représente les points originaux dans un espace de dimension plus grande où il sera plus facile de trouver un séparateur linéaire.

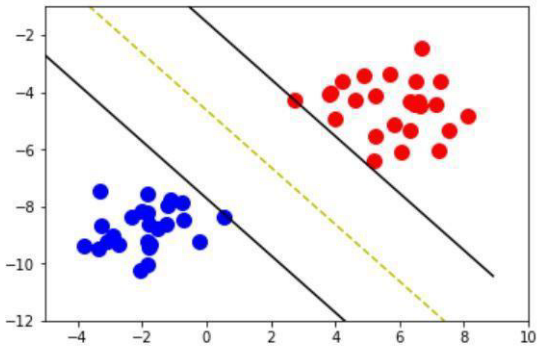


Figure 9 Les SVM [3]

L'avantage des SVM est qu'ils permettent de traiter des problèmes complexes de classification non linéaire.

1.8 Conclusion

Dans ce chapitre, nous avons défini l'apprentissage automatique comme étant des systèmes pouvant « apprendre » à partir de données d'entrainement et pouvant trouver des relations entre les données entrés et les sorties. Nous avons ensuite cité ses principaux domaines d'application qui sont le data mining et l'intelligence artificielle. Nous avons exposé les raisons pour lesquelles nous avons opté pour une approche machine learning au lieu de l'approche traditionnelle. Ensuite, nous avons expliqué deux types majeurs de machine learning qui sont l'apprentissage supervisé et l'apprentissage non supervisé et expliqué les différences entre ces deux types.

À la fin de ce chapitre, nous avons décrit quelques algorithmes du machine learning les plus utilisés, dont les machines à vecteurs de support (SVM), les arbres de décisions, les K plus proches voisins (K-NN), etc… . Nous avons expliqué leurs principes de fonctionnement et enfin leurs avantages et leurs limitations.

Nous opterons dans notre travail pour l'algorithme des K plus proches voisins dont nous détaillerons l'implémentation dans le chapitre 3.

Dans le chapitre qui suit, nous passerons aux concepts de base et aux techniques de traitement d'image.

Chapitre 2

Le traitement d'image

2.1 Introduction

Dans notre cas, nous nous intéressons à la reconnaissance de caractères dans une image, ce que l'on appelle OCR (Optical Character Recognition). OCR se défini comme la conversion d'une image comportant du texte (scanné, écrit à la main ou imprimé) ou bien la reconnaissance et l'extraction de texte se trouvant sur la photo d'un document ou la photo d'une scène en données de type texte pouvant être stockées et exploitées par la machine. OCR est largement utilisé dans de nombreuses applications permettant d'extraire des informations depuis des images de papiers d'identité, des reçus informatisés ou encore des photos de voitures. Ces informations à leurs tours servent à d'autres applications de gestions, et notamment dans notre cas aux applications de gestion de parkings.

La croissance exponentielle des capacités des processeurs est pour une grande part bénéfique au développement du traitement d'images. Ce domaine est par nature très gourmand en puissance de calcul et son développement a longtemps été freiné par l'insuffisance des moyens de calcul disponibles. Ce frein est aujourd'hui en train de s'estomper.

Dans ce chapitre nous allons voir ce qu'est le traitement d'image à travers diverses définitions et illustrations, voir aussi les différents types de processus de traitement et de filtrage tout en passant par les méthodes de détection de contours et de segmentation.

2.2 Définition de l'image

L'image est une représentation d'une personne ou d'un objet par la peinture, la sculpture, le dessin, la photographie, le film, etc… . C'est aussi un ensemble structuré d'informations qui, après affichage sur l'écran, ont une signification pour l'œil humain. Elle peut être décrite sous la forme d'une fonction $f(x, y)$ de brillance analogique continue, définie dans un domaine borné, tel que $x\ et\ y$ sont les coordonnées spatiales d'un point de l'image et f est une fonction d'intensité lumineuse et de couleur. Sous cet aspect, l'image est inexploitable par la machine, ce qui nécessite sa numérisation. [7]

2.3 Image Numérique

Les images manipulées par un ordinateur sont numériques (représentées par une série de bits). L'image numérique est l'image dont la surface est divisée en éléments de taille fixe appelés cellules ou pixels, ayant chacun comme caractéristique un niveau de gris ou de couleurs prélevé à l'emplacement correspondant dans l'image réelle, ou calculé à partir d'une description interne de la scène à représenter. La numérisation d'une image est la conversion de celle-ci de son état analogique en une image numérique (échantillonnage) représentée par une matrice bidimensionnelle de valeurs numériques $I(n, m)$ où : n, m sont les coordonnées cartésiennes d'un point de l'image et $I(n, m)$ le niveau de gris ou de couleur en ce point.

2.3.1 Caractéristiques d'une image numérique

L'image est une collection structurée d'informations déterminé par les paramètres suivants :

2.3.1.1 Pixel

Le pixel est la contraction de l'expression anglaise " Picture elements". Le pixel, étant le plus petit point de l'image, c'est une entité calculable qui peut recevoir une structure et une quantification. Dans une image couleur (R.V.B), un pixel peut être représenté sur trois octets, un octet pour chacune des couleurs : rouge (R), vert (V) et bleu (B).

2.3.1.2 Dimension

C'est la taille de l'image. Cette dernière se présente sous forme de matrice dont les éléments sont des valeurs numériques représentatives des intensités lumineuses (pixels). Le nombre de lignes de cette matrice multipliée par le nombre de colonnes nous donne le nombre total de pixels dans une image.

2.3.1.3 Résolution

C'est la clarté ou la finesse de détails atteinte par un moniteur ou une imprimante dans la production d'images. Sur les écrans d'ordinateurs, la résolution est exprimée en nombre de pixels par unité de mesure (pouce ou centimètre). On utilise aussi le mot résolution pour désigner le nombre total de pixels affichables horizontalement ou verticalement sur un moniteur; plus grand est ce nombre, meilleure est la résolution.

2.3.1.4 Histogramme de l'image

On veut souvent avoir une information sur la distribution des valeurs des pixels (niveaux) dans une image, pour cela on utilise souvent une table qui donne le nombre de pixels de chaque niveau dans l'image. Cette table, souvent représentée graphiquement, est appelée Histogramme de l'image, notée $h(x)$. Dans cet exemple, on veut représenter l'image de taille 256×256 pixels par son histogramme et pour cela, on commence par calculer la table histogramme, ensuite on trace l'histogramme de l'image comme montré dans la figure (10). x : présente la valeur

du pixel dans l'image. $y = h(x)$: présente le nombre de fois répété de cette valeur du pixel dans l'image. [9]

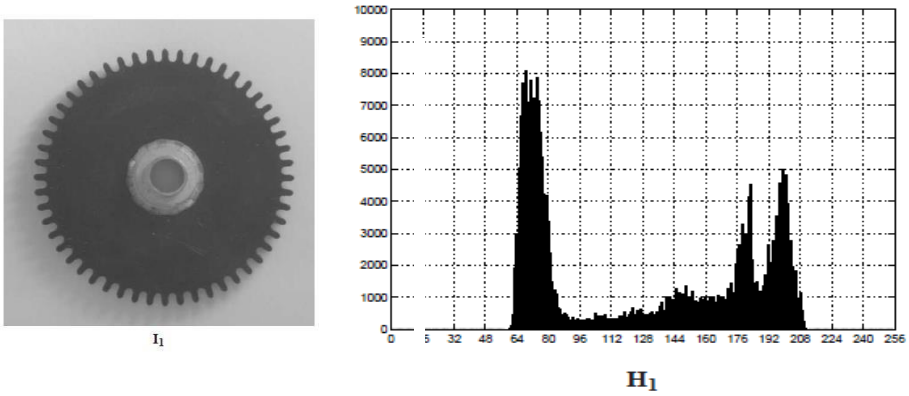


Figure 10 Exemple d'histogramme d'une image. [1]

2.3.1.5 Luminance

C'est le degré de luminosité des points de l'image. Elle est définie aussi comme étant le quotient de l'intensité lumineuse d'une surface par l'aire apparente de cette surface. Pour un observateur lointain, le mot luminance est substitué au mot brillance qui correspond à l'éclat d'un objet [7]. Une bonne luminance se caractérise par :

- Des images lumineuses (brillantes).
- Un bon contraste : il faut éviter les images où la gamme de contraste tend vers le blanc ou le noir, ces images entraînent des pertes de détails dans les zones sombres ou lumineuses.
- L'absence de parasites.

2.3.1.6 Contraste

C'est l'opposition marquée entre deux régions d'une image, plus précisément entre les régions sombres et les régions claires de cette image. Le contraste est défini en fonction des luminances de deux zones d'images. Si L1 et L2 sont les degrés de

luminosité respectivement de deux zones voisines A1 et A2 d'une image, le contraste C est défini par le rapport :

$$C = \frac{L1 - L2}{L1 + L2}$$

2.3.1.7 Bruit

Un bruit dans une image est considéré comme un phénomène de brusque variation de l'intensité d'un pixel par rapport à ses voisins. Le bruit peut provenir de différentes causes :

- Qualité de l'échantillonnage,
- Capacité des capteurs ou une mauvaise utilisation de ces derniers,
- Environnement lors de l'acquisition.

2.3.1.8 Images en couleur

Nous pouvons représenter les couleurs à l'aide de leurs composantes primaires. Les systèmes émettant de la lumière (écrans d'ordinateurs, etc…) sont basés sur le principe de la synthèse additive : les couleurs sont composées d'un mélange de rouge, vert et bleu (modèle R.V.B).

2.3.1.9 Image à niveaux de gris

Le niveau de gris est la valeur de l'intensité lumineuse en un point. La couleur du pixel peut prendre des valeurs allant du noir au blanc en passant par un nombre fini de niveaux intermédiaires. Donc pour représenter les images à niveaux de gris, on attribue à chaque pixel de l'image une valeur correspondant à la quantité de lumière renvoyée. Cette valeur peut être comprise par exemple entre 0 et 255. Chaque pixel est représenté par un octet. Le nombre de niveaux de gris dépend du nombre de bits utilisés pour décrire la « couleur » de chaque pixel de l'image. Plus ce nombre est important, plus les niveaux possibles sont nombreux. [26]

On peut obtenir une image à niveaux de gris à partir d'une image en couleur en calculant la somme pondérée des couleurs comme dans la formule suivante :

$$gris = int(round(0.299 \cdot Rouge + 0.587 \cdot Vert + 0.114 \cdot Bleu))$$

2.4 Filtrage

Un filtrage est l'application d'une transformation (appelée filtre) à toutes les parties d'une image numérique. Le principe du filtrage est de modifier la valeur des pixels d'une image dans le but d'améliorer son aspect. Il s'agit de créer une nouvelle image en se servant des valeurs des pixels de l'image d'origine. Le filtrage ne manipule que les données de l'image numérisée. On ne fait aucune opinion sur ce que symbolise l'image. Le résultat obtenu dépend beaucoup de la qualité du signal de l'image d'origine. Si le signal est très dégradé, le filtrage seul sera très peu efficace, dans ce genre de situation le filtrage est juste une première étape dans un traitement plus complexe. Il existe plusieurs filtres utilisés en traitement d'image :

- Filtrage linaires
- Filtrage non linaires [8]

2.4.1 Les filtres linéaires

Les filtres linéaires convertissent un ensemble de données d'entrée en un ensemble de données de sortie suivant une opération mathématique nommée convolution. Quand il s'agit de données numérisées comme dans le cas du traitement d'image, la relation entre les valeurs des pixels de sortie et celle d'entrée est décrite par une matrice de nombres souvent carrée appelée matrice de convolution. Le temps de calcul est généralement réduit lorsqu'on veut fractionner un filtre en deux filtres dont la convolution mutuelle permet de le reconstituer. Cette technique est utilisée en particulier pour créer un filtre à deux dimensions à partir de deux filtres à une seule dimension (vecteurs) dans les deux sens, horizontal et vertical.[10]

$f'(x,y) = h(x,y) \times f(x,y)$; f' : résultat de convolution

2.4.1.1 Filtre passe-bas

Ce type de filtre consiste à atténuer les composantes de l'image possédant une fréquence haute (pixels foncés). Ce filtrage est souvent utilisé pour réduire le bruit de l'image, on parle dans ce cas de lissage.

- **Le lissage** : est l'opération de filtrage visant à éliminer le bruit d'une image. C'est des filtres passe-bas qui coupent plus ou moins les plus hautes fréquences. Ils sont utilisés pour atténuer les bruits. [8]

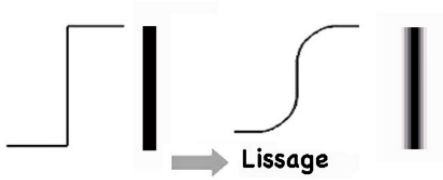


Figure 11 Effet de lissage. [1]

- **Filtre moyen** : Le filtre moyen est un filtre passe-bas dont le principe est de faire la moyenne des valeurs des pixels avoisinants. Le résultat de ce filtre est une image plus floue.

- **Filtre gaussien** : La fonction Gaussienne est très souvent utilisée dans les distributions statistiques. Les propriétés de diminution de bruit des filtres Gaussien peuvent être utilisées en combinaison avec d'autres filtres qui au contraire engendrent du bruit.
  Un filtre gaussien est donné par discrétisation de la fonction gaussienne sur un voisinage de $(0, 0)$. Ici σ est l'écart-type et la moyenne est nulle.

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

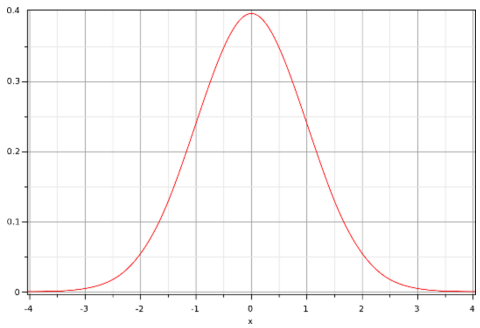


Figure 12 Fonction gaussienne. [4]

Si $\sigma = 0.8$ on a le filtre 3×3 suivant :

$$\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

Si $\sigma = 1$ pour un filtre 5×5 donne environ :

$$\frac{1}{300}\begin{pmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 18 & 30 & 18 & 4 \\ 6 & 30 & 48 & 30 & 6 \\ 4 & 18 & 30 & 18 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{pmatrix}$$

La taille du filtre gaussien est gouvernée par σ qui doit être proportionnel à l'écart-type du bruit. En général un filtre gaussien avec $\sigma < 1$ est utilisé pour réduire le bruit, et si $\sigma > 1$ c'est dans le but de fabriquer une image qu'on va utiliser pour faire un masque flou personnalisé. Il faut noter que plus σ est grand, plus le flou appliqué à l'image sera important. [11]

2.4.1.2 Filtre passe-haut

Les filtres passe-haut atténuent les composantes de basse fréquence de l'image et permettent d'appuyer les détails et le contraste. Ils sont utilisés pour la détection de contours, pour augmenter la netteté et augmenter le contraste.

2.4.1.3 Limites du filtrage linéaire

Bruit impulsionnel : L'image fait apparaître des valeurs parasites aberrantes qui ne dépendent pas des valeurs initiales de l'image. Le filtrage linéaire diffuse les valeurs aberrantes sur les pixels voisins.

Figure 13 Exemple cas de poivre et sel. [1]

2.4.2 Les filtres non linéaires

Le filtrage non linéaire fait intervenir les pixels voisins suivant une loi non linéaire, le filtre le plus classique et plus utilisé est le filtre médian. Comme les filtres de convolution, les filtres non-linéaires opèrent sur un voisinage donné.

> **Filtre médian** : La médiane est la valeur qui partage une population en deux parties de même effectif lorsqu'on trie celle-ci par valeurs ordonnées de la variable considérée. Ce filtre est utilisé pour atténuer des pixels isolés d'une valeur très différente de leur voisinage, il affecte à chaque pixel la valeur médiane de ses voisins de façon à éliminer les points isolés. Ce filtre n'introduit pas de flou sur les contours des particules contrairement aux lissages. [9]

2.5 Les prétraitements

On regroupe souvent sous le terme de prétraitement toutes les opérations qui sont appliquées aux images, indépendamment de leur usage futur, pour leur assurer une bonne qualité. Elles concernent donc essentiellement les corrections de contraste et la suppression du bruit. Les prétraitements sont souvent appliqués dans des cas où l'on destine les images à de nombreuses applications différentes dont on ignore souvent les besoins exacts.

2.5.1 Les traitements photométriques ou colorimétriques

2.5.1.1 Binarisation, seuillage

Le seuillage est une technique qui repose sur une mesure quantitative d'une grandeur. Il permet de classer les pixels selon un seuil fixé. Si un pixel à une valeur supérieure au seuil fixé, il prendra la valeur 255 (blanc), et si sa valeur est inférieure au seuil fixé, il prendra la valeur 0 (noir).

Figure 14 Différents niveaux de seuillage. [4]

2.5.1.2 Augmentation de contraste

Une famille de traitements a pour objectif de donner à l'image un plus grand contraste. Ces méthodes procèdent de trois façons différentes :

1. **Par étirement d'histogramme :**
   On cherche à exploiter au mieux la dynamique de l'histogramme, tout d'abord en utilisant toute l'échelle de gris ou de couleurs disponible (par une table de transcodage), mais aussi en modifiant l'histogramme de façon à le transformer en un histogramme de référence.

2. **Par filtrage passe-haut de l'image :**
   L'idée de base ici est de réduire l'importance du terme continu et des basses fréquences. Un point délicat de ces méthodes est de définir le gabarit du filtre utilisé.

3. **Par des méthodes locales :**
   Ces méthodes modifient localement l'histogramme pour conserver toujours une bonne dynamique, même dans des zones de fort contraste.

2.5.2 Suppression des bruits

Il compte parmi les sujets les plus délicats du traitement des images. Nous verrons tout d'abord les approches linéaires, puis les méthodes non-linéaires.

2.5.2.1 Filtrages linéaires

Dans de nombreux cas on considère que l'image non dégradée $f(x, y)$ est affectée d'un bruit additif $n(x, y)$:

$$g(x, y) = f(x, y) + n(x, y)$$

Le cas d'un bruit multiplicatif est beaucoup plus difficile, il est par exemple traité dans le cas de l'imagerie radar.

Dans les conditions les plus simples, on suppose le signal stationnaire et on adopte donc souvent une approche par filtrage linéaire. Intuitivement, et en l'absence de toute autre information, on recherche des filtres passe-bas éliminant les variations a très haute fréquence du signal.

En pratique, ces filtres ne sont pas très bons car ils dégradent les contours, ils sont plus efficaces si le signal est effectivement stationnaire (et donc sans contours) et si le bruit est gaussien. Les plus utilisés sont le filtre de moyenne (sur des petites fenêtres de 3×3 ou 5×5 pixels) et le filtre gaussien. [12]

On utilise aussi la propriété des filtres gaussiens à noyau autoreproducteur en filtrant successivement par des gaussiennes étroites pour obtenir des gaussiennes larges. On a aussi beaucoup utilisé des approximations splines des fonctions gaussiennes, plus simples à calculer.

Figure 15 Filtrage linéaire. [4]

La figure est une image bruitée par un bruit gaussien à gauche, au centre, filtrage passe-bas sur une fenêtre 3×3 et à droite sur une fenêtre 7×7.

2.6 Détection de contours

Un contour se concrétise par une rupture d'intensité dans l'image suivant une direction donnée, il existe plusieurs méthodes pour détecter cette rupture. Dans la plupart des cas la même méthodologie est employée, elle s'applique en deux étapes : la première permet de localiser les contours à partir d'un calcul de Gradient, de Laplacien ou de Canny dans des directions privilégiées. La seconde étape va permettre d'isoler les contours du reste de l'image à partir d'un seuillage judicieux. Plusieurs méthodes permettent de déterminer le Gradient ou le Laplacien d'une image, il en est de même des techniques de seuillage. Ces deux étapes sont indépendantes, il existe donc un grand nombre de combinaisons pour le calcul du Gradient et des opérations de seuillage aboutissant à la mise en évidence des contours.

La détection de contour est une étape préliminaire à de nombreuses applications de l'analyse d'images. Les contours constituent en effet des indices riches, au même titre que les points d'intérêts, pour toute interprétation ultérieure de l'image. Les contours dans une image proviennent des :

- Discontinuités de la fonction de réflectance (texture, ombre).
- Discontinuités de profondeur (bords de l'objet).

Ils sont caractérisés par des discontinuités de la fonction d'intensité dans les images. Le principe de la détection de contours repose donc sur l'étude des dérivées de la fonction d'intensité dans l'image : les extrema locaux du gradient de la fonction d'intensité et les passages par zéro du Laplacien. La difficulté réside dans la présence de bruit dans les images. [14]

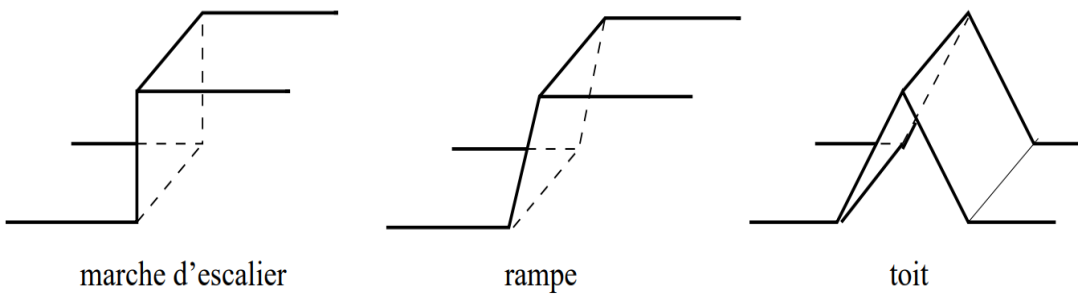


Figure 16 Différents types de contours.[1]

2.6.1 Approche Gradient

En considérant l'image dans un repère orthogonal (Oxy) tel que (Ox) désigne l'axe horizontal et (Oy) l'axe vertical, le Gradient de l'image (ou plutôt de la luminance) en tout point ou pixel de coordonnées (x, y) est désigné par :

$$\vec{\nabla} f(x,y) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right)$$

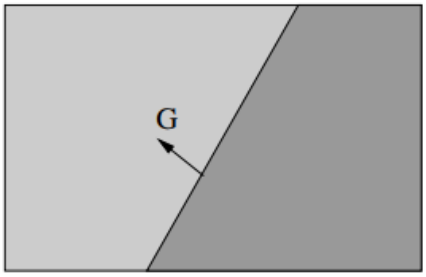


Figure 17 Direction du gradient G. [1]

Le module du gradient permet de quantifier l'importance du contour mis en évidence, c'est-à-dire l'amplitude du saut d'intensité relevé dans l'image :

$$\|\vec{\nabla} f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$

La direction du gradient permet de déterminer l'arête présente dans l'image. En effet, la direction du gradient est orthogonale à celle du contour :

$$\alpha_0 = \arctan\left(\frac{\partial f/\partial y}{\partial f/\partial x}\right)$$

Le principe de la détection de contours par l'utilisation du gradient consiste à calculer d'abord le gradient de l'image dans deux directions orthogonales puis le module du gradient. Il s'agira ensuite d'effectuer une sélection des contours les plus marqués, c'est-à-dire les points de plus fort contraste par un seuillage adéquat, les directions des contours étant orthogonales à la direction α_0 déterminée en tout pixel de l'image. [14]

2.6.1.1 Quelques opérateurs gradient

Les filtres de Roberts

Les filtres de Roberts sont une approche discrète de la dérivée de pas 1 d'une fonction c'est-à-dire le gradient de cette fonction.

Si $f(x, y)$ represente un pixel dans une image, alors les amplitudes des gradients en x et en y peuvent s'écrire respectivement :

$$Gx = f(x+1,y) - f(x,y),$$
$$Gy = f(x,y+1) - f(x,y).$$

Cela revient à convoluer l'image avec les deux filtres $Rx = [-1\ 1]$ et $Ry = transposé([-1\ 1])$.

L'amplitude du gradient peut être alors calculée de plusieurs façons :

$$G1(x,y) = sqrt(Gx\text{^}2 + Gy\text{^}2),$$
$$G2(x,y) = max(abs(Gx), abs(Gy))$$
$$ou\ G3(x,y) = abs(Gx) + abs(Gy).$$

et la direction du gradient est donnée par :

$$\alpha_0 = D(x,y) = arctan(Gy/Gx).$$

Or le bruit peut aussi être une brusque variation locale des niveaux de gris : ces filtres sont donc très sensibles au bruit car ils accentuent, par dérivation, le bruit présent dans l'image. De plus, ces filtres donneront un contour épais si celui-ci est un contour de type "rampe".

Les filtres de Prewitt et de Sobel

Les filtres de Prewitt et de Sobel sont aussi des opérateurs de dérivation mais on y a introduit un opérateur de lissage.

Gx	Gy	
$-1\ \ 0\ \ 1$	$-1\ -2\ -1$	
$-2\ \ 0\ \ 2$	$0\ \ 0\ \ 0$	*Opérateur Sobel*
$-1\ \ 0\ \ 1$	$1\ \ 2\ \ 1$	

Gx			Gy		
-1	0	1	-1	-1	-1
-1	0	1	0	0	0
-1	0	1	1	1	1

Opérateur Prewitt

L'amplitude peut être calculée de la même manière que pour les filtres de Roberts.

Ces filtres sont moins sensibles au bruit que ceux de Roberts car le fait d'introduire un moyennage local sur le domaine couvert par le masque diminue leur sensibilité. Le filtre de Sobel donne une meilleure estimation que celui de Prewitt car la série 1 2 1 est approximativement une gaussienne. [15]

Figure 18 (a) image originale, (b) filtre de Robert, (c) filtre de Prewitt, (d) filtre de Sobel [4]

2.6.2 Approche Laplacien

Les opérateurs de Gradient vus précédemment exploitent le fait qu'un contour dans une image correspond au maximum du gradient dans la direction orthogonale au contour. Or le passage par zéro de la dérivée seconde d'une rupture d'intensité permet également de mettre en évidence le contour. La dérivée seconde est donc déterminée par le calcul du Laplacien :

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

On considère que les points de contours sont localisés aux passages par zéro de Laplacien. Si le calcul du Laplacien était exact il suffirait de sélectionner les points M tels que :

$$\nabla(M) = 0.$$

Figure 19 Détection de contours. [1]

A droite image originale, à gauche affichage des contours grâce au passage par zéro du Laplacien.

2.6.3 Approche de Canny

L'approche développée par Canny consiste en la recherche de l'opérateur de dérivation optimal sous forme de filtre à réponse impulsionnelle finie (RIF). Cette approche peut être résumée comme suit :

Soit un signal $A(x)$ monodimensionnel représentant un saut d'amplitude U_0 noyé dans un bruit blanc stationnaire $N(x)$ de moyenne nulle et de densité spectrale N_0^2.

$$A(x) = U_0 U(x) + N(x)$$

Le signal de sortie est :

$$C(x) = A \times h(x) = \int_{-\infty}^{\infty} A(t)h(x-t)dt$$

C'est un filtre optimal de réponse impulsionnelle $h(x)$ qui satisfait les trois contraintes suivantes :

- Bonne détection : Le rapport RSB (Rapport Signal/Bruit) doit être maximisé.

$$RSB = \frac{U_0 \int_0^{\infty} h(x-t)dt}{N_0\left[\int_{-\infty}^{\infty} h^2(t)dt\right]}$$

- Bonne localisation : Les points détectés doivent être aussi près que possible du centre du contour véritable (à maximiser)

$$RSB = \frac{U_0|h'(0)dt|}{N_0\left[\int_{-\infty}^{\infty} h'^2(t)dt\right]^{1/2}}$$

- Unicité de la réponse : On utilise le critère de Canny. On veut minimiser la densité d0 des passages par 0 de la réponse du bruit.

3.9 Segmentation

Essentiellement, la segmentation est un processus qui consiste à découper une image en régions connexes présentant une homogénéité selon un certain critère, comme par exemple la couleur. L'union de ces régions doit redonner l'image initiale. On regroupe généralement les algorithmes de segmentation en trois grandes classes:

- **Segmentation basée sur les pixels** : Travaille sur des histogrammes de l'image. Par seuillage, clustering ou clustering flou, l'algorithme construit des classes de couleurs qui sont ensuite projetées sur l'image. La segmentation est implicite puisqu'on suppose que chaque cluster de l'histogramme correspond à une région dans l'image. En pratique, ce n'est pas forcément le cas et il faut séparer les régions de l'image qui sont disjointes bien qu'appartenant au même cluster de couleur.

- **Segmentation basée sur les régions** : correspond aux algorithmes d'accroissement ou de découpage de région. L'accroissement de région est une méthode bottom-up : on part d'un ensemble de petites régions uniformes

dans l'image (de la taille d'un ou de quelques pixels) et on regroupe les régions adjacentes de même couleur jusqu'à ce qu'aucun regroupement ne soit plus possible. Le découpage de région est le pendant top-down des méthodes d'accroissement : on part de l'image entière que l'on va subdiviser récursivement en plus petites régions tant que ces régions ne seront pas suffisamment homogènes.

- **Segmentation basée sur les contours** : s'intéresse aux contours des objets dans l'image. La plupart de ces algorithmes sont locaux, c'est à dire fonctionnent au niveau du pixel. Des filtres détecteurs de contours sont appliqués à l'image et donnent généralement un résultat difficile à exploiter sauf si les images sont très contrastées. Les contours extraits sont la plupart du temps morcelés et peu précis, il faut alors utiliser des techniques de reconstruction de contours par interpolation ou connaître à priori la forme de l'objet recherché. [16]

3.10 Conclusion

Dans ce chapitre, nous avons introduit quelques concepts de base qui servent de fondement à la compréhension des différentes techniques de traitement d'image, à savoir quelques définitions et caractéristiques sur les images numériques tout en passant par les notions de filtrage (linéaire et non-linéaire).

Les prétraitements d'images permettent d'améliorer la qualité de l'image et de supprimer les bruits en vue de traitements ultérieurs.

A la fin de ce chapitre, nous avons présenté les différentes approches qui ont été élaborées pour la détection de contours sur les images en niveaux de gris en appliquant différents opérateurs, et nous avons clôturé avec des notions fondamentales sur la segmentation.

Dans le chapitre qui suit, nous passerons à la conception de notre application.

Chapitre 3

Conception

3.1 Introduction

La détection de caractères constitue une étape principale dans les systèmes de reconnaissance de texte et dans plusieurs applications telle que la classification. Toutefois, c'est une tâche peu facile étant donnée les variations de tailles, de styles, d'orientations et d'alignements, ainsi que la complexité de l'arrière-plan de l'image.

Notre projet consiste en la réalisation d'un module capable de lire les plaques d'immatriculation de véhicule dans un smart parking basé sur les méthodes de détection de caractères dans une scène. Nous allons alors présenter les procédés qui nous permettrons de formaliser les étapes préliminaires du développement de notre système. Cette phase de conception nous permettra de décrire le fonctionnement futur du système afin d'en faciliter la réalisation.

Dans ce chapitre, nous allons présenter en premier lieu notre problématique, la démarche utilisée pour reconnaitre la plaque d'immatriculation ainsi que les différentes fonctionnalités utilisées pour automatiser le parking. Aussi, nous allons voir l'architecture proposée pour ce système et la base de données utilisée.

3.2 Problématique

En premier lieu, la problématique est de réaliser le module de reconnaissance automatique de plaques d'immatriculation, ce dernier trouvera son utilité au niveau d'un parking intelligent pour le contrôle des accès et le calcul du montant pour le péage. Il peut aussi renforcer la sécurité des parkings contre le vol de voitures. La réalisation de ce module nécessite de trouver la suite des différents traitements à effectuer sur l'image afin d'en extraire les caractères. Ensuite, l'adaptation de l'algorithme K-NN à notre cas d'utilisation afin de reconnaitre ces caractères.

En deuxième lieu, la gestion des véhicules dans le parking s'intègre dans la problématique. Nous citons notamment :

- La disponibilité de places libres.
- Le temps de recherche d'une place.
- Contrôle automatique des portiques.
- Le calcul du montant à payer.
- L'historique des entrées aux parking qui représente une source d'informations pour une enquête policière par exemple.

Nous allons expliquer dans ce qui suit les solutions que nous proposons pour répondre à ces objectifs ainsi que les étapes de conception de ces solutions.

3.3 Solutions proposées

Nous avons conçu un système de reconnaissance de plaques d'immatriculations embarqué communiquant avec un serveur en utilisant le réseau local du parking. Ainsi, nous avons formé une architecture client/serveur en local. Le serveur qui représente la centrale de calcul qui sera relié à la base de données et le client qui n'est d'autre que le système embarqué qui s'occupera de la reconnaissance et de l'automatisation des accès. La communication entre ces deux derniers sera effectuée via les sockets.

3.3.1 Architecture générale

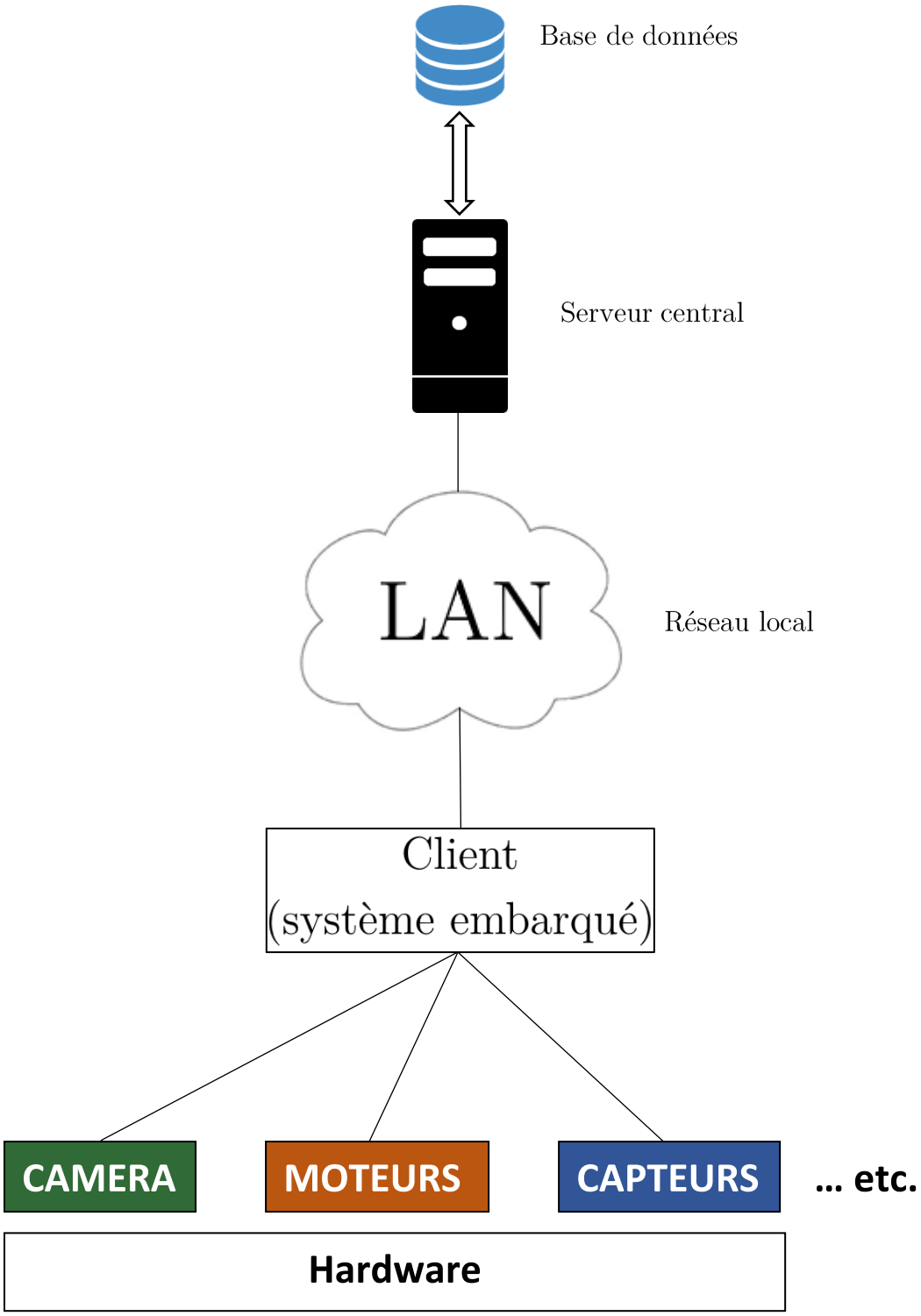


Figure 20 Architecture générale du système.

3.4 La reconnaissance automatique de plaques d'immatriculation

La reconnaissance automatique de plaques d'immatriculation est une technologie basée sur la détection de texte permettant de lire les plaques d'immatriculation de véhicules. L'ALPR est largement utilisé partout pour des applications dans la police, dans des installations de gestion du trafic, des agences de péage sur des routes, ainsi que des services de gestion de stationnement et de parkings.

Dans notre cas, nous utilisons cette technologie pour la gestion d'un parking intelligent pour lire et identifier les plaques des véhicules algériens entrants et sortants.

3.4.1 Génération des données d'entrainement

L'idée de d'utiliser des données de formation dans des programmes d'apprentissage automatique est un concept simple, mais c'est aussi un élément fondamental du fonctionnement de ces technologies. Les données de formation sont un ensemble initial de données utilisées pour aider un programme à comprendre comment appliquer des technologies telle que la classification pour apprendre et produire des résultats satisfaisants. Ces données qu'on appel en anglais « training data ».

Vu que les plaques algériennes ne contiennent que des chiffres alors nous avons rassemblé des images de chiffres en noir et blanc (binaire) de dimension 30 pixels de hauteur et 20 pixel de largeur avec des polices variées (voir le tableau 1).

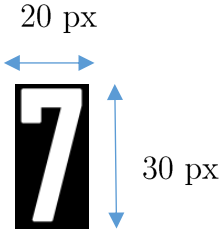


Figure 21 Dimensions utilisées pour les images d'entrainement.

Le but de cette étape est de générer les données d'apprentissage (training set), une étiquette sera attribuée pour chaque image du même chiffre. Afin de faciliter la tâche lors de l'entrainement, les images sont transformées sous forme de flattened images (images aplaties sous forme d'un vecteur a une dimension). Elles sont ensuite enregistrées avec leurs étiquettes respectives dans des fichiers dans la mémoire de stockage.

Nous obtenons donc un ensemble de données prêt à être exploité. Notre algorithme tire des enseignements des données étiquetées. Après avoir compris les données, l'algorithme détermine l'étiquette (classe) à attribuer aux nouvelles données.

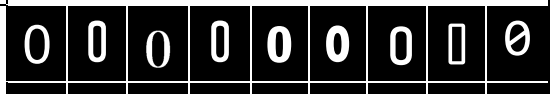
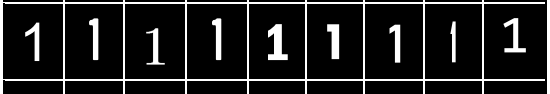
Classification	**Images de chiffres**
« 0 »	
« 1 »	
« 2 »	
« 3 »	
« 4 »	
« 5 »	
« 6 »	
« 7 »	
« 8 »	
« 9 »	

Tableau 1 Tableau des différentes polices utilisées pour générer les données de la classification

3.4.2 Processus utilisé dans reconnaissance

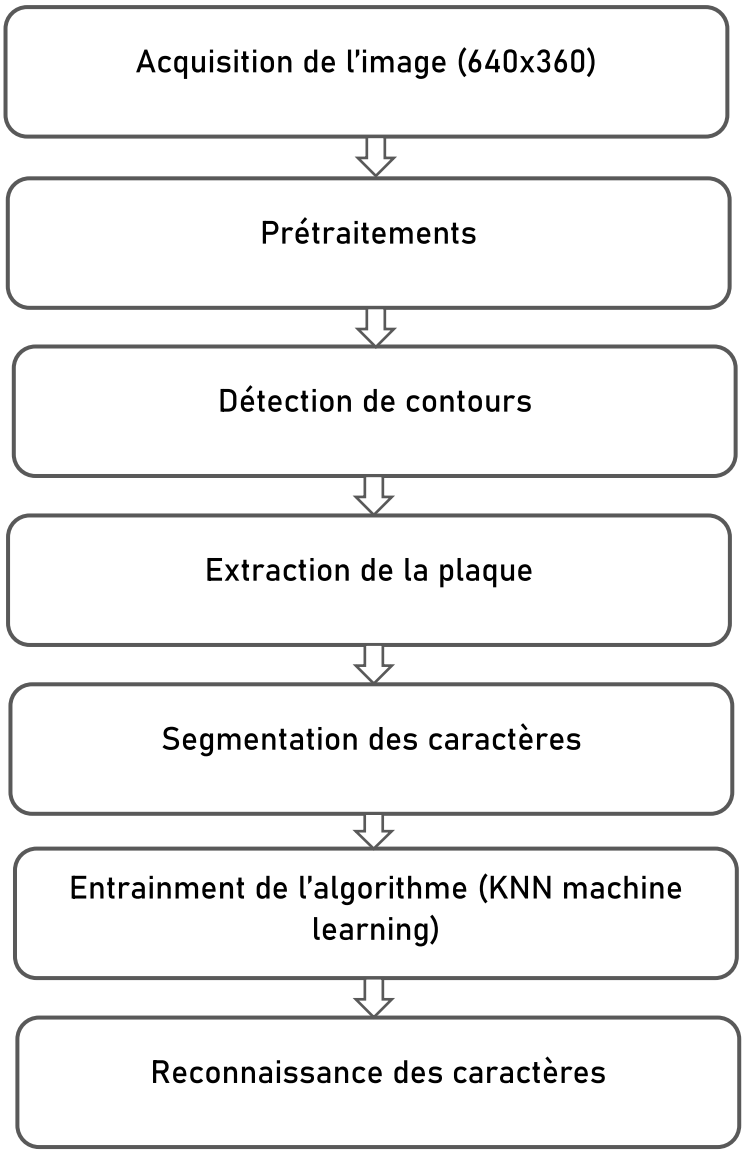


Figure 22 Schémas de reconnaissance.

3.4.2.1 Acquisition de l'image

Figure 23 Image RGB (640 X 360) capturée du véhicule.

Une caméra est embarquée sur le portique du parking pour pouvoir capturer directement la plaque d'immatriculation. Nous utilisons dans notre cas le module caméra embarqué sur notre système pour capturer une image à chaque fois qu'une voiture s'approche à l'entrée ou à la sortie. Une fois le véhicule est à bonne distance, une image est enregistrée.

3.4.2.2 Prétraitements

Comme nous l'avions décrit dans le chapitre précèdent, le prétraitement vise à améliorer les caractéristiques d'une image. Dans un premier temps, nous transformerons notre image en niveau de gris, ce qui nous donne l'avantage de travailler sur une dimension de niveaux de gris de 0 à 255. La formule pour obtenir une image en niveaux de gris est de calculer pour chaque pixel la somme pondérée de ces trois niveaux de couleur (RGB) de la manière suivante :

$$gris = int(round(0.299 \cdot R + 0.587 \cdot G + 0.114 \cdot B))$$

Figure 24 Image en niveaux de gris

a. **Rehaussement du contraste** :

Le renforcement du contraste a pour but de restaurer les contours francs de l'image et la rendre plus interprétable par la machine. Cela se fait en changeant les valeurs initiales de façon à utiliser toutes les valeurs possibles, ce qui permet d'augmenter le contraste entre les cibles et leur environnement.

b. **Lissage par le filtre Gaussien :**

Également appelé flou Gaussien, c'est un effet utilisé pour réduire le bruit de l'image, il est utilisé dans cette étape de prétraitement afin d'améliorer les structures d'image à différentes échelles.

c. **Seuillage :**

Le seuillage d'image est la méthode la plus simple de segmentation d'image. À partir d'une image en niveau de gris, nous l'utilisons pour créer une image comportant uniquement deux valeurs (noir ou blanc) que nous utiliserons plus tard afin de faciliter la détection de contours.

Par la présence de conditions d'éclairage différentes selon les zones dans l'image, nous optons pour le seuillage adaptatif contrairement au seuillage classique vu dans le chapitre précédent. En cela, l'algorithme calcule le seuil pour de petites régions de l'image. Nous obtenons donc des seuils différents pour différentes régions de la même image, ce qui nous donne de meilleurs résultats pour les images à éclairage variable.

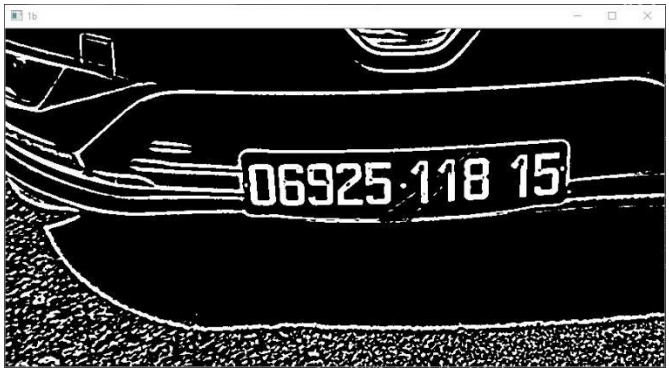


Figure 25 Image obtenue avec le seuillage adaptif

3.4.2.3 Détection de contours

Les contours sont parmi les caractéristiques visuelles les plus importantes et les plus utilisées dans la détection des objets. Sachant que les caractères de texte contiennent de forts contours, nous pouvons utiliser cette propriété pour détecter leurs présences dans une région donnée. Ainsi, le processus de détection de contours réduit considérablement la quantité d'informations à analyser dans l'image. C'est le processus de recherche des forts contrastes dans notre image binarisée. La bibliothèque OpenCV nous offre la possibilité de sauvegarder ces contours.

Figure 26 Représentation des contours

3.4.2.4 Extraction de la plaque

Caractéristiques des plaques algériennes :

Comme nous nous intéressons aux plaques d'immatriculation algériennes, voici quelques détails sur les immatriculations en Algérie :

- La plaque est composée de dix à onze chiffres. En la parcourant de droite à gauche : les 2 premiers chiffres désignent la wilaya, les 2 suivants l'année de la mise en marche du véhicule, le chiffre suivant permet de distinguer le type du véhicule et les 5 ou 6 derniers chiffres quant à eux désignent le numéro d'enregistrement.
- La plaque est sous forme de rectangle et le style des caractères varie d'une plaque à une autre, ce qui veut dire que le style des caractères n'est pas standard. La distance entre les caractères n'est pas standard, elle dépend du constructeur de la plaque.

Figure 27 Plaque d'immatriculation algérienne.

Cette phase de détection nécessite de tester un grand nombre de régions candidates, ces régions ont été préalablement détectées dans l'image, chacune d'elles étant détectée par son contour peut représenter un caractère ou non.

Figure 28 Régions susceptibles d'être un caractère (chiffre).

Une fois les zones de texte détectées, il est important de filtrer un certain nombre de fausses alarmes qui correspondent souvent à des parties de l'image ne contenant pas de texte mais dont les caractéristiques sont similaires à des zones de texte. Dans cette étape, nous visons à corriger les mauvaises classifications du texte et non texte.

Le principe consiste à former une liste de caractères qui se correspondent mutuellement, pour cela nous allons utiliser les propriétés géométriques des contours fermés des régions candidates :

1. La distance entre deux caractères (leurs points centrales) afin qu'ils ne soient ni trop loin ni trop proches, éliminer les contours entrelacés comme les trous dans les chiffres.

$$\min < Distance < max$$

2. L'angle entre deux caractères, pour ne laisser que les contours qui sont sur même ligne droite.

$$|\theta| < \omega$$

3. S'assurer que les caractères ont approximativement la même taille : calculer le rapport de changement dans la surface, la largeur et la hauteur entre deux contours.
   Soit A1, A2, w1, w2, h1, h2 respectivement leurs surfaces, largeurs et hauteurs :

$$\frac{|A1-A2|}{A1} < S$$

$$\frac{|w1-w2|}{w1} < W$$

$$\frac{|h1-h2|}{h1} < H$$

Figure 29 Contours des chiffres correspondants à la plaque d'immatriculation.

Pour extraire la plaque il suffit de calculer ses propriétés géométriques (centre, largeur, hauteur et inclinaison) à partir du premier et dernier caractère trouvé :

Soit le centre (x, y) , la largeur l , la hauteur h, et l'inclinaison α et soit (x_i, y_i) , $i = 1,2$ les coordonnées des centres des deux caractères et h_1, h_2 leurs hauteurs respectivement.

$$x = \frac{x_1 + x_2}{2}, \quad y = \frac{y_1 + y_2}{2}$$

$l = (x2 - x1) \times frl$, avec frl : facteur de rembourrage largeur

$h = \left(\frac{h_1 + h_2}{2}\right) \times frh$, avec frh : facteur de rembourrage hauteur

$$\alpha = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right)$$

Figure 30 Plaque localisée puis découpée.

3.4.2.5 Segmentation des caractères

Afin de s'assurer de l'exactitude de nos caractères, il est impératif de refaire les étapes précédentes avant de procéder au découpage des caractères.

Cette étape a pour intérêt de segmenter les caractères dans la région candidate sélectionnée (plaque d'immatriculation extraite) de manière à ce que chaque caractère puisse être envoyé individuellement dans l'étape OCR.

Figure 31 Segmentation des contours de caractères.

3.4.2.6 K-NN training

Le K-NN est l'abréviation de K Nearest Neighbors. C'est un algorithme qui va nous servir dans la classification. Il est appelé k plus proches voisins en français car le principe de ce modèle consiste en effet à choisir les k données les plus proches du point étudié afin d'en prédire sa valeur.

Nous avons opté pour cet algorithme car il est assez simple d'un point de vue conceptuel et illustre parfaitement les problématiques classiques qui en découlent. On va s'en servir pour illustrer le traitement des données générées dans les sections précédentes.

Notre objectif sera d'entraîner un modèle qui sera capable de reconnaître les chiffres. Après avoir chargé les trainings data, un objet permettant à l'algorithme K-NN de reconnaitre les caractères (chiffres) est généré.

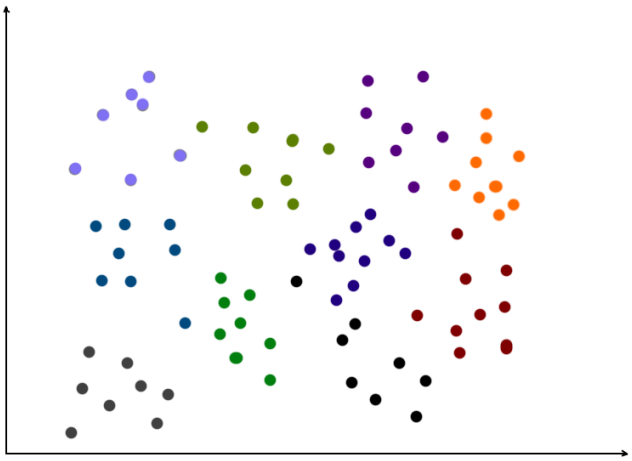


Figure 32 Illustration de l'ensemble des données.

3.4.2.7 Reconnaissance des caractères

Le modèle généré dans l'entrainement va permettre à l'algorithme de k-plus proches voisins de rechercher la correspondance la plus proche des données.

Le choix de l'hyperparamètre k :

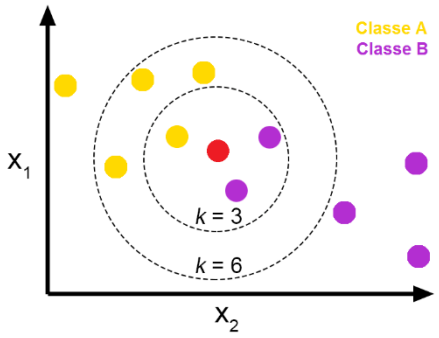


Figure 33 Choix de k.

k est un hyperparamètre, ce qui veut dire qu'il ne va pas pouvoir être appris automatiquement par l'algorithme à partir des données d'entraînement. Les hyperparamètres permettent de caractériser le modèle (complexité, rapidité de convergence, etc...). Du coup c'est à nous que revient la tâche de l'optimiser. En règle générale, moins on utilisera de voisins (un nombre k petit) plus on sera sujet au sous apprentissage (underfitting). D'autre part, plus on utilise de voisins (un nombre k grand) plus sera fiable dans notre prédiction. Cependant, si on utilise k nombre de voisins avec k = i (i étant le nombre de données ou d'observations) on risque d'avoir un sur-apprentissage (overfitting) et par conséquent un modèle qui se généralise mal sur des observations qu'il n'a pas encore vues.

En pratique, choisir la valeur de $k = sqrt(i)$, où i représente le nombre d'échantillons dans un ensemble de données d'apprentissage. De plus, il faut conserver la valeur de k impair pour éviter toute confusion entre deux classes de données.

Après avoir segmenté les contours des caractères et avant la phase de reconnaissance, il est très important de redimensionner chaque image de caractère sur $20 \times 30\, pixel$ comme le training data afin d'avoir des résultats nets. On peut schématiser ainsi le fonctionnement de reconnaissance en l'écrivant en pseudocode suivant :

```
Début

   Pour une nouvelle observation i dont on veut prédire sa
   variable de sortie a faire :

      1. Calculer toutes les distances de cette observation
         i avec les autres observations du modèle M
      2. Retenir les k observations du modèle M les proches de
         i en utilisation la fonction de calcul de distance
         d = √(x² + y²)
      3. Prendre les valeurs de a et des k observations retenues
      4. Calculer la valeur la plus représentée des x retenues

      Retourner la valeur calculée dans l'étape 4 comme étant
      la valeur qui a été prédite par KNN pour l'observation i

Fin
```

Après avoir parcouru et reconnu chaque caractère dans la plaque, le numéro d'immatriculation sera sauvegardé dans la base de données.

Figure 34 Illustration du résultat final de la lecture de la plaque grâce à OpenCV.

3.4.3 Problèmes rencontrés

La détection des plaques d'immatriculation est un type particulier des systèmes de détection de texte dans une image. Cependant, les plaques d'immatriculation ont deux propriétés principales qui devraient contribuer à l'amélioration des performances des systèmes de la détection. Tout d'abord, les couleurs possibles d'une plaque de véhicule sont limitées, deuxièmement le nombre de caractères dans une plaque d'immatriculation est fixé. Depuis que les plaques d'immatriculation des véhicules sont basées sur des normes propres à chaque pays, les systèmes de détection des plaques d'immatriculation sont spécifiques et adaptés à ces différentes normes. Cependant, des problèmes classiques de traitement d'image demeurent toujours tels que :

- La mauvaise résolution de l'image, soit parce que la plaque est trop loin ou parfois résultant de l'utilisation d'une caméra de mauvaise qualité.
- Les images floues, particulièrement à cause du mouvement.
- Le mauvais éclairage et le faible contraste à cause de la surexposition, de la réflexion ou de l'ombre.
- Les objets qui cachent une partie de la plaque, bien souvent une barre de remorquage ou la saleté sur la plaque.
- Plaque cassée ou endommagée.
- La différence de polices des caractères de la plaque, populaire pour les plaques de vanité. Certains pays ne permettent pas de telles plaques, ce qui élimine le problème.

Autre problème tel que :

Le temps de calcul : les méthodes basées sur les outils d'apprentissage consomment un temps important de calcul. Ces méthodes s'appliquent sur la totalité de l'image. La phase d'apprentissage et de classification devient plus difficile lorsque la taille de l'image ainsi que celle des caractères est plus grande. Donc la grande variation de tailles des caractères dans les images représente l'un des obstacles dans l'utilisation des outils d'apprentissage pour la détection de texte.

3.5 Conception du smart parking

Le système embarqué comporte une partie matérielle et une partie logicielle. Lors de la phase de conception, l'aspect matériel doit être pris en considération.

3.5.1 Besoins à satisfaire

Pour notre cas de figure, le principal objectif est d'améliorer l'expérience de l'utilisateur dans le parking. Parmi les besoins à satisfaire, nous citons :

- Réduire le désagrément de l'utilisateur au niveau de l'entrée au parking : Certaines entrées au parking nécessitent l'appuis de l'utilisateur sur un bouton pour lever la barrière et récupérer un ticket d'entrée.
- L'identification automatique de l'utilisateur à sa sortie : éviter le désagrément de garder le ticket et de le présenter à la sortie.
- Le gain de temps.
- L'orientation de l'utilisateur vers la place libre.
- La sécurité.
- Interface homme machine efficace.
- Projet réalisable à moindre coût.

L'entrée du parking :

Un affichage permet à l'utilisateur de savoir au préalable s'il y a des places libres dans le parking, ce qui lui évite une perte de temps inutile. S'il y a des places libres, le véhicule est identifié grâce à sa plaque d'immatriculation avec l'ALPR vu dans la première partie de ce chapitre. Ensuite la barrière lui laisse l'accès automatiquement.

L'intérieur du parking :

Afin d'orienter l'utilisateur à l'intérieur du parking et lui faire ainsi gagner du temps, les affichages à des endroits stratégiques lui permettent de savoir dans quelle direction se trouve les places libres. Pour améliorer la visibilité, des voyants lumineux au niveau de chaque place, par exemple une lumière verte pour signaler que la place est libre sera visible même depuis une longue distance. Le parking prévoit aussi par mesure de sécurité des détecteurs de flammes en cas d'incendie.

La sortie du parking :

Le véhicule est identifié grâce à sa plaque d'immatriculation avec l'ALPR. Une IHM lui affiche le montant à régler dès qu'il s'approche de la sortie. Le montant réglé, la barrière s'élève le laissant quitter le parking. L'IHM doit être ergonomique, et ne doit présenter que les informations vraiment utiles à l'utilisateur.

L'un des principaux problèmes qu'on rencontre lors de la conception d'un projet d'automatisation est le coût du projet. Plusieurs options peuvent répondre à un problème donné, le choix d'une solution simple et efficace est à favoriser contrairement à une solution sophistiquée qui reviendra coûteuse à réaliser et à entretenir une fois arrivé à la phase de mise en service.

Les données :

Les données utiles à la gestion du parking sont relatives aux utilisateurs, à leurs entrées dans le parking et aux places de parking. On suppose ici, que l'on ne tient pas compte des informations sur l'utilisateur tels que son nom et son prénom. Aussi, nous pourrions à l'aide de capteurs ou de caméras savoir qu'elle place l'utilisateur a prise précisément, cependant nous allons ignorer cette information pour notre application.

3.6 Conclusion

Durant cette étape de conception, les besoins auxquels doit répondre le système ont été décrits à l'aide des différentes procédures proposés dans notes architecture.

Par la suite, la conception du système a permis d'exposer l'ensemble des fonctionnalités qu'on a défini. Ce qui permettra de répondre aux besoins déjà définis.

Le chapitre suivant sera consacré à la réalisation du système en expliquant les outils utilisés ainsi que les moyens mis en œuvre pour son implémentation.

Chapitre 4

Réalisation

4.1 Introduction

Dans le chapitre précédent, nous avons exposé la problématique et ainsi conçu des solutions. L'étape suivante est consacrée à la réalisation, où nous détaillerons l'environnement matériel et logiciel de notre travaille, ainsi que la mise en œuvre et les résultats des tests effectués, et enfin nous citerons des perspectives.

Nous avons travaillé sur une carte Raspberry Pi 3 modèle B+, avec le système d'exploitation Raspbian. Nous avons implémenté l'algorithme K-NN avec Python avec l'aide de Numpy et d'OpenCV et nous avons utilisé le module camera V1.3 de Raspberry Pi.

L'application permettra d'indiquer à l'utilisateur s'il y a des places libres disponibles, de détecter un véhicule à l'entrée du parking grâce à des capteurs. La caméra placée sur le portail permettra d'extraire la plaque d'immatriculation à l'aide de l'algorithme K Nearest Neighbors et laisser l'accès au véhicule. À l'intérieur du parking, des lumières colorées et des affichages le guident vers la place libre. À sa sortie, le véhicule sera détecté, ou une caméra y sera aussi placé. Le véhicule sera identifié par sa plaque d'immatriculation, et le prix à payer sera affiché à l'utilisateur.

4.2 Environnement matériel

L'utilisation d'ordinateurs mono-carte dans les systèmes embarqués est très répandue et beaucoup d'individus et d'organisations ont développé et publié des produits entièrement fonctionnels basé sur des ordinateurs à mono-carte. Les ordinateurs mono-carte populaires disponibles sur le marché sont : Raspberry Pi, Banana Pi, BeagleBone et Cubieboard.

4.2.1 Raspberry Pi

Le Raspberry Pi est un ordinateur complet mono carte (single-board). Il possède un processeur, une RAM, des Entrées/Sorties et les ports réseau pour l'interfaçage des périphériques. Raspberry Pi est de petite taille, peu coûteux, il a été développé par la Raspberry Pi Foundation au Royaume-Uni. L'intention derrière sa création était de promouvoir l'enseignement des compétences informatiques de base dans les écoles, et le premier sert bien cet objectif. Raspberry Pi a bien étendu son empreinte au-delà de sa destination en pénétrant sur le marché des systèmes embarqués et la recherche.

Les modèles Raspberry Pi (A, A +, B et B +) sont basés sur le SoC (système sur puce) Broadcom BCM2835, qui comprend un processeur ARM11 700 MHz. La 2éme génération de RPi utilise un processeur ARM Cortex-A7 à quatre cœurs, les premiers RPi multicœur. La 3éme génération de RPi utilise un processeur ARM Cortex-A53 à quatre cœurs et la quatrième génération utilise le processeur ARM Cortex-A72. [17]

Nous utiliserons le Raspberry Pi 3 modèle B + pour exécuter l'algorithme de vision par ordinateur. Voici Les spécifications de celui-ci :

Alimentation à prévoir : 5 Vcc/maxi 2,5 A via prise micro-USB.
CPU : ARM Cortex-A53 quatre cœurs 1,4 GHz.
Wi-Fi : Dual-band 2,4 et 5 GHz, 802.11b/g/n/ac (Broadcom BCM43438).
Bluetooth : 4.2 (Broadcom BCM43438).
RAM : 1 Go.
Ethernet Jusqu'à 300 Mbps.
USB : 4 ports USB 2.0.

Bus : SPI, I2C.
Support pour cartes micro-SD : 1.
Sorties audios :
- **HDMI** avec gestion du 5.1.
- **Jack** 3,5 mm en stéréo.
Sortie vidéo : HDMI
Dimensions : 86 x 54 x 17 mm.
Poids : 50 g.

4.2.2 Module caméra

Nous avons utilisé le module de caméra version 1.3 spécialement conçu pour Raspberry Pi. Il est équipé d'un capteur Omni vision 5647 de 5MP. Le module se connecte à Raspberry Pi, à l'aide d'un câble ruban à 15 broches, à l'interface série CSI (MIPI Camera Serial Interface) à 15 broches, spécialement conçue pour l'interfaçage avec les caméras. Le bus CSI a un débit extrêmement élevé et transmet exclusivement des données de pixels au processeur BCM2835. Le module mesure environ 25 mm x 20 mm x 9 mm et pèse un peu plus de 3 g, ce qui le rend parfait pour les applications mobiles ou autres où la taille et le poids sont importants.

En termes d'images fixes, l'appareil photo est capable de capturer des images de 2592 x 1944 pixels et prend également en charge l'enregistrement vidéo 1080p à 30 images par seconde, 720p à 60 images par seconde et l'enregistrement vidéo 60/90 de 640x480p. [18]

Le module camera prend en charge les formats d'image suivants : JPEG (accéléré), JPEG + RAW, GIF, BMP, PNG, YUV420, RGB888, et les formats vidéo : h.264 brut (accéléré).

4.2.2.1 Picamera

Picamera est un paquet python qui fournit une interface de programmation au module caméra du RPi. On l'installe en utilisant.

```
sudo apt-get install python-picamera
```

Le programme suivant montre rapidement l'utilisation de base du paquet picamera pour capturer une image :

```
import picamera
import time
with picamera.PiCamera() as cam:
cam.resolution=(1024,768)
cam.start_preview()
time.sleep(5)
cam.capture('/home/pi/Desktop/rmse.jpg')
```

Nous devons d'abord importer les modules time et picamera. La méthode *cam.start_preview*() démarre l'aperçu et *time.sleep*(5) attend 5 secondes avant que *cam.capture*() capture et enregistre l'image dans le fichier spécifié.

4.2.2.2 Picamera et OpenCV

Le code suivant illustre l'utilisation de picamera avec OpenCV ; il affiche un aperçu pendant 3 secondes, capture une image et l'affiche à l'écran à l'aide de *cv2.imshow*().

```
import picamera
import picamera.array
import time
import cv2
with picamera.PiCamera() as camera:
     rawCap=picamera.array.PiRGBArray(camera)
     camera.start_preview()
     time.sleep(3)
     camera.capture(rawCap,format="bgr")
     image=rawCap.array
cv2.imshow("Test",image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

4.2.3 Capteur de distance

Les capteurs de distance (ou capteurs de proximité) fonctionnent généralement en émettant un signal quelconque (par exemple un laser, une LED infrarouge, des ondes ultrasonores), puis en lisant comment il a changé à son retour. Ce changement peut concerner l'intensité du signal renvoyé, le temps nécessaire au signal pour revenir, etc.

Termes relatifs à la détection de distance :

Résolution : fait référence au plus petit changement de distance qu'un capteur est capable de détecter.

Taux de mise à jour : généralement mesuré en Hz, entre en jeu avec des objets en mouvement. Plus la vitesse de rafraîchissement est rapide, plus le capteur recevra de lectures par seconde, informations importantes si le capteur se déplace vers un objet fixe à une vitesse élevée.

Plage : La plage est une distance, du minimum au maximum, à laquelle un capteur est capable de retourner des lectures précises. [18]

4.2.3.1 Le capteur HC-SR04

Le capteur HC-SR04 utilise les ultrasons pour déterminer la distance d'un objet. Il offre une excellente plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière du soleil ou des matériaux sombres, bien que des matériaux comme les vêtements puissent être difficiles à détecter.

Figure 35 Capteur HC-SR04

Caractéristiques :

• Dimensions : 45 mm x 20 mm x 15 mm.

• Plage de mesure : 2 cm à 400 cm.

• Résolution de la mesure : 0.3 cm.

• Angle de mesure efficace : 15 °.

Broches de connexion :

• Vcc = Alimentation +5 V DC.

• Trig = Entrée de déclenchement de la mesure (Trigger input).

• Echo = Sortie de mesure donnée en écho (Echo output).

• GND = Masse de l'alimentation.

Distance de l'objet :

La distance parcourue par un son se calcule en multipliant la vitesse du son, environ 340 m/s, par le temps de propagation, soit : d = v · t (distance = vitesse · temps). On sait aussi que le son fait un aller-retour. La distance vaut donc la moitié.

Le capteur HC-SR04 est intéressant. Pour un coût très bas, il donne des résultats étonnants de précision. Il possède un taux d'erreur inférieure à 2 %.

4.2.4 Les servomoteurs

Un servomoteur est un système motorisé capable d'atteindre des positions prédéterminées, puis de les maintenir. La position dans le cas d'un moteur rotatif est une valeur d'angle. Cependant, il présente un grand inconvénient, c'est le coût. Pour cela, une alternative existe qui consiste à utiliser un moteur à courant continu combiné à un capteur de fin de course.

4.2.4.1 Servomoteur SG90

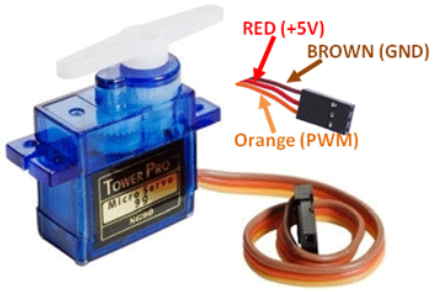


Figure 36 Servomoteur SG90

Caractéristiques :

- Dimensions : 22 x 11.5 x 27 mm.
- Poids : 9 gr.
- Tension d'alimentation : 4.8v à 6v.
- Vitesse : 0.12 s / 60° sous 4.8v.
- Couple : 1.2 Kg / cm sous 4.8v.
- Amplitude : de 0 à 180°.

Broches de connexion :

La correspondance des fils est la suivante :

- Rouge : Vcc = Alimentation +5 V DC.
- Marron : GND = Masse de l'alimentation.
- Orange : PWM = Signal de commande.

Contrôler un servo-moteur :

Le principe de base est assez simple. Il suffit d'envoyer une impulsion et c'est le temps que durera cette impulsion qui déterminera l'angle du servo-moteur. Ce temps d'impulsion est de quelques de quelques millisecondes et doit être répété à intervalle régulier (50Hz soit toutes les 20 ms).

4.2.5 Les afficheurs

Plusieurs choix existent pour afficher les données : les afficheurs 7 segments, les afficheur LCD, les matrices LED, ou même à partir des écrans connectés par HDMI au Raspberry Pi. Nous utiliserons les afficheurs pour indiquer à l'utilisateur le nombre de places libres dans le parking.

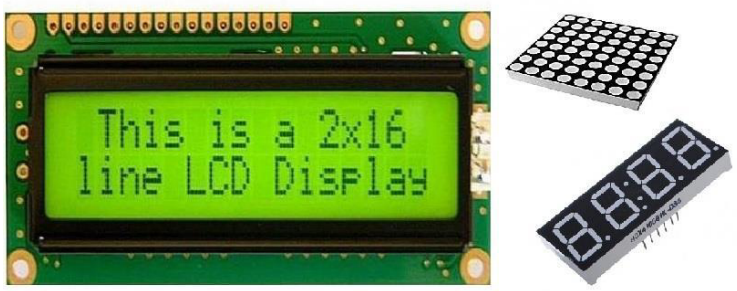


Figure 37 Afficheurs.

4.2.5.1 Afficheur LCD I2C :

L'Afficheur LCD 2 lignes de 16 caractères interface I2C, Il se compose de deux parties : un écran LCD "classique" et au dos un module d'interface I2C. La communication avec une carte se fait avec le protocole I2C sur deux lignes dénommées SCL et SDA. Il faut ajouter les lignes d'alimentation Vcc et GND. [18]

Figure 38 Afficheur LCD 2 lignes de 16 caractères, interface I2C

Caractéristiques :

- Alimentation : 5 Vcc.
- Interface I2C (adresse 0x27).
- Caractères blancs sur fond bleu.
- Contraste ajustable via potentiomètre.
- Dimensions : 80 x 38 x 18 mm.

4.2.5.2 Afficheur 7 segments

Les afficheurs 7 segments : sont un type d'afficheur très utilisés, les caractères s'écrivent en allumant ou en éteignant des segments, au nombre de sept. Dans un afficheur 7 segments, les segments sont généralement désignés par les lettres allant de A à G plus éventuellement un point noté DP.

2 types d'afficheurs existent :

Afficheur à anode commune : toutes les anodes sont reliées et connectées au potentiel haut.
La commande du segment se fait par sa cathode mise au potentiel bas.

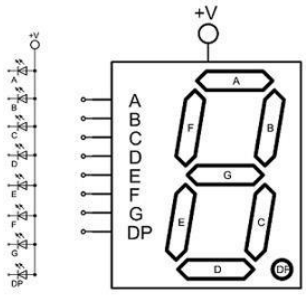


Figure 39 Afficheur à anode commune

Afficheur à cathode commune : toutes les cathodes sont reliées et connectées au potentiel bas. La commande du segment se fait par son anode mise au potentiel haut.

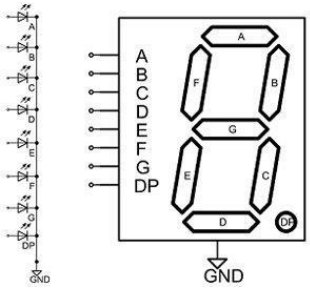


Figure 40 Afficheur à cathode commune

4.2.6 Capteurs de flammes

La sortie varie en présence d'une flamme (la photodiode est sensible spectre lumineux généré par une flamme).

LED1 : indique que le capteur est alimenté en tension.
LED2 : indique qu'une flamme est détectée.

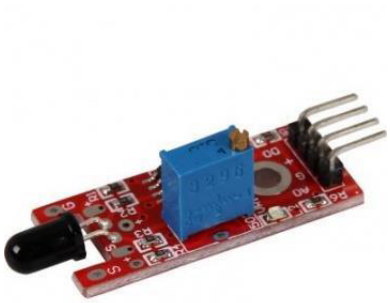


Figure 41 Capteur de flammes.

Broches de connexion :

• Vcc = Alimentation +5 V DC.

• GND = Masse de l'alimentation.

• Sortie numérique : Emission d'un signal lorsqu'une flamme est détectée.

• Sortie analogique : mesure directe du capteur.

4.2.7 Photorésistance

Une photorésistance ou LDR est un composant électronique dont la résistivité varie en fonction de la quantité de lumière à laquelle elle est exposée : plus elle est éclairée, plus sa résistivité baisse. Elle présente l'avantage d'avoir une sensibilité élevée et un coût faible. [18]

Etant donné que le parking est éclairé même la nuit, en plaçant des capteurs de lumière au niveau d'une place de parking, la non détection de lumière s'interprète par l'occupation ce celle-ci (la place).

Figure 42 Photorésistance.

Schéma de connexion :

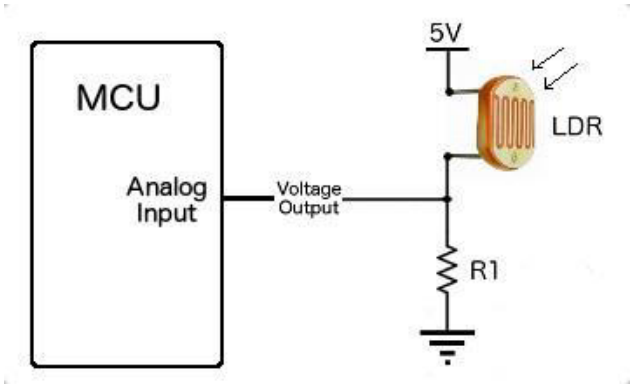


Figure 43 Schéma de connexion d'un capteur de luminosité (photorésistance)

4.2.8 Les LEDs

Une LED (Light-Emitting Diode) est un dispositif optoélectronique capable d'émettre de la lumière lorsqu'il est parcouru par un courant électrique.

Nous les utiliserons pour indiquer si une place est occupée ou pas. Ce qui aidera au guidage des utilisateurs.

Figure 44 LEDs.

4.2.8.1 LED RGB

Le principe de fonctionnement est simple. Les changements de couleurs sont obtenus par le mélange de 3 couleurs primaires. Le rouge, le vert et le bleu. D'où l'abréviation RGB pour R (red – rouge), G (green – vert) et B (blue – bleu). Les leds flexibles RGB contiennent ces 3 couleurs primaires.

Schéma de connexion :

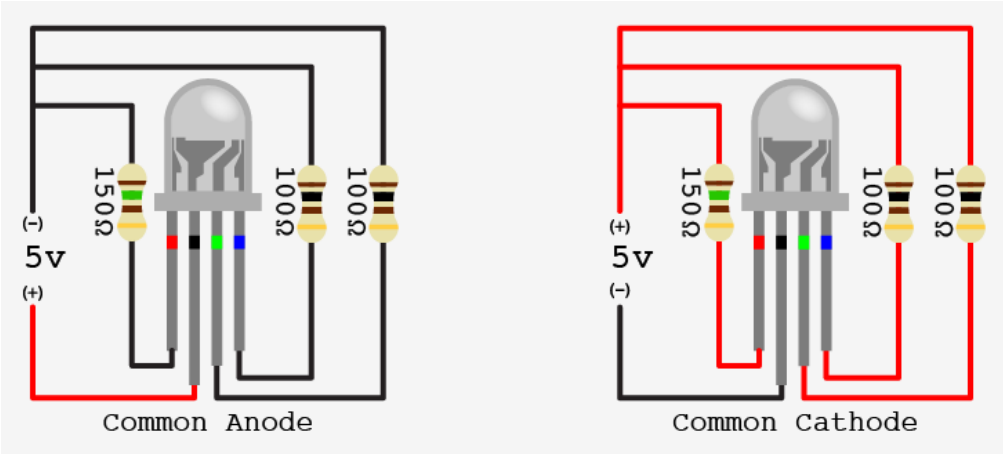


Figure 45 Schéma de connexion d'une LED RGB

4.3 Environnement logiciel

4.3.1 Système d'exploitation

Raspberry Pi utilise principalement des systèmes d'exploitation de type Unix, basés sur le noyau Linux, comme les variantes de Debian et Fedora. Les modèles Raspberry Pi A, A +, B et B + sont basés sur la puce de la famille ARM11, qui fonctionne sur le jeu d'instructions ARM v6. Le jeu d'instructions ARM v6 ne supporte pas Ubuntu et Windows.

Cependant, le Raspberry Pi 2 récemment lancé est basé sur ARM Cortex A7, qui est capable d'exécuter Windows 10 et Ubuntu (Snappy Core). Le système d'exploitation le plus utilisé avec la RPi est Raspbian. Raspbian est une variante non officielle de Debian armhf (ARM hard float) qui est compilé pour s'exécutera sur les ordinateurs Raspberry Pi.

Il s'agit d'un système d'exploitation libre basé sur Debian optimisé pour l'architecture matériel de Raspberry Pi. Raspbian est plus qu'un système d'exploitation pur. Il vient avec plus de 35 000 paquets et logiciel précompilé pour Raspberry Pi.

4.3.2 Python

Python est un langage de programmation interprété, multiparadigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramasse-miettes et d'un système de gestion d'exceptions.

Le langage Python est placé sous une licence libre proche de la licence BSD8 et fonctionne sur la plupart des plates-formes informatiques, des smartphones aux ordinateurs, de Windows à Unix avec notamment GNU/Linux en passant par MacOS, ou encore Android, iOS, et peut aussi être traduit en Java ou .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser.

Il est également apprécié par certains pédagogues qui y trouvent un langage où la syntaxe, clairement séparée des mécanismes de bas niveau, permet une initiation aisée aux concepts de base de la programmation. Python est un langage de programmation généraliste de haut niveau qui peut être appliqué à différentes classes de problèmes.

Le langage est livré avec une grande bibliothèque standard qui couvre des domaines tels que le traitement des chaînes (expressions régulières, Unicode, calcul des différences entre les fichiers), les protocoles Internet (HTTP, FTP, SMTP, XML-RPC, POP, IMAP, CGI), le génie logiciel (tests unitaires, journalisation, profilage, analyse de code Python) et interfaces du système d'exploitation (appels système, systèmes de fichiers, sockets TCP / IP. [19]

4.3.3 OpenCV

OpenCV (Open Source ComputerVision) est une bibliothèque de fonctions programmé pour la vision par ordinateur. Il a été initialement développé par le centre de recherche Intel Russia à Nizhny Novgorod, et il est actuellement maintenu par Itseez.

C'est une bibliothèque multi-plateforme, ce qui signifie qu'elle peut être implémentée et exploité sur différents systèmes d'exploitation. Il se concentre principalement sur les traitements sur les images et les vidéos. En plus de cela, il a plusieurs fonctionnalités d'interface graphique et de gestion des événements pour l'expérience de l'utilisateur.

OpenCV a été publié sous licence BSD (Berkeley Software Distribution) et par conséquent, il est gratuit pour une utilisation académique et commerciale. Il a des interfaces pour les langages de programmation les plus populaires, tels que C/ C++, Python et Java, et il fonctionne sur une variété des systèmes d'exploitation, y compris Windows, Android et les systèmes d'exploitation de type Unix.

OpenCV était à l'origine une initiative d'Intel Research visant à développer des outils d'analyse d'images. En août 2012, le soutien à OpenCV a été repris par une fondation à but non lucratif, OpenCV.org, qui le développe actuellement. Il maintient également un site pour les développeurs et un site pour les utilisateurs d'OpenCV. [20]

Figure 46 Logo Opencv

4.3.4 Numpy

NumPy est un paquet fondamental qui peut être utilisé pour calculer scientifiquement avec Python. C'est une bibliothèque de matrices pour l'algèbre linéaire. NumPy peut également être utilisé comme un conteneur multidimensionnel efficace de données génériques. Des types de données arbitraires peuvent être définis et utilisés. NumPy est une extension du langage de programmation Python.

Il ajoute un support pour les grands tableaux multidimensionnels et matrices, avec une grande bibliothèque de fonctions mathématiques de haut niveau pouvant être utilisées pour exploiter ces tableaux. Nous utiliserons des tableaux NumPy pour représenter des images et effectuer des opérations mathématiques complexes. NumPy est livré avec de nombreuses fonctions incorporés pour toutes ces opérations. Nous n'avons donc pas à nous soucier des opérations de base sur les tableaux. Nous pouvons nous concentrer directement sur les concepts et le code pour la vision par ordinateur. Toutes les structures de tableau OpenCV sont converties vers et à partir de tableaux NumPy. Donc quel que soit l'opération qu'on veut effectuez dans NumPy, on peut combiner NumPy avec OpenCV. [21]

4.3.5 Systèmes de Gestion de Base de données

Comme dans toute application, nous avons eu à stocker de manière fiable des données et à les présenter de manière utile aux utilisateurs. Nous avons donc besoin d'un SGBD pour notre système. Voici un tableau comparatif des différents SGBD : [27]

PostgreSQL	PostgreSQL est la base de données à utiliser pour les gros projets. Stable et très puissant, il permet de gérer des Gigabytes de données sans problème.
MySQL	Mysql est l'un des SGBDR les plus utilisés au monde. Il est gratuit et très puissant. Il possède la double licence GPL et propriétaire depuis son rachat par *Sun Microsystem* eux-mêmes racheté par *Oracle* (concurrent direct de MySQL). Le logiciel reste cependant entièrement gratuit et libre. Il répond à une logique client/serveur, c'est à dire que plusieurs clients (ordinateurs distants) peuvent se connecter sur un seul serveur qui héberge les données.
MariaDB	Le créateur de MySQL a créé MariaDB suite au rachat de MySQL pour continuer le projet en open source.
SQLite	SQLite est une bibliothèque écrite en C. SQLite est parfait pour les petits projets. Sa particularité est d'être intégré directement à un programme et ne répond donc pas à la logique client-serveur. Le logiciel pèse moins de 300 ko et peut donc être intégré à des projets tournant sur de petits supports comme les smartphones. Souvent aucune installation n'est nécessaire pour l'utiliser.
Oracle	Oracle Database est sous licence propriétaire, c'est à dire payant. Il est souvent utilisé pour les projets à gros budget nécessitant de réaliser des actions complexes.
Microsoft SQL Server	Produit Microsoft ne tourne que sur un OS Windows, payant n'apporte rien de plus que les logiciels concurrents libre de droit. Si vous avez trop d'argent à la limite...

Tableau 2 Tableau comparatif des différents SGBD

SQLite est une base de données stockée dans un seul fichier sur le disque. SQLite est intégré à Python, mais il n'est conçu que pour l'accès par une seule connexion à la fois. Par conséquent, il n'est pas adapté à notre application. PostgreSQL et MySQL sont deux des bases de données open source les plus courantes pour le stockage des données d'applications Python. Nous opterons donc pour MySQL.

4.4 Mise en œuvre

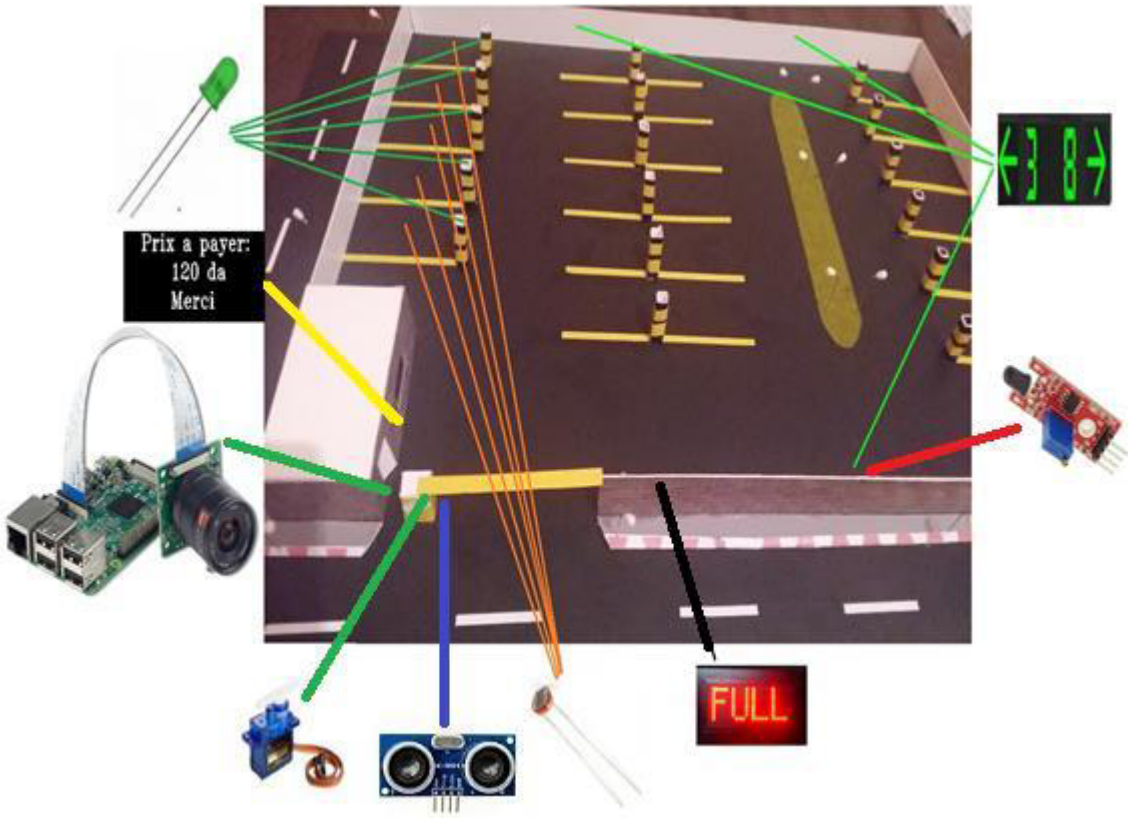


Figure 47 Mise en œuvre.

La mise en œuvre consiste principalement à mettre en place le module ALPR et à la gestion du parking. Une caméra sera placée à l'entrée et une deuxième à la sortie. Elles servent à la capture de la photo de la plaque d'immatriculation afin d'ensuite procéder à son traitement. Le véhicule est détecté grâce à des capteurs de distance. L'afficheur LCD est également placé à l'entrée afin d'indiquer les informations relatives au parking : Plein, vide, nombre de places libres restantes. La barrière est rattachée à un servo moteur, qui ouvre le passage au véhicule.

A l'intérieur, des LEDs rgb placé au niveau de chaque place indiquent s'il y'a des places libres dans chaque allée (lumière rouge : place occupé, lumière verte : place libre). Les photorésistances détectent si la place est libre ou non. Nous ajoutons aussi des afficheurs au bout de chaque allée affichant le nombre de place libre de part et autre afin d'aider l'utilisateur à trouver rapidement une place. Enfin par respect aux normes de sécurité, le capteur de flammes détecte la présence d'incendie.

4.5 Tests

Nous avons testé notre module de reconnaissance de plaques d'immatriculation sur un ensemble de 30 images, le tableau suivant illustre les résultats obtenus :

Tests	Réussite	Echecs	Taux de réussite	Taux d'échec
30	26	4	86%	14%

Tableau 3 Tableau récapitulatif des tests

Observations sur les tests échoué :

Test 6 : Caractère « 1 » non détecté.

Test 7 : Plaque d'immatriculation non détecté.

Test 19 : Caractère « 3 » non détecté ; Caractère « 7 » mais « 1 » détecté ; Caractère « 0 » mais « 5 » détecté.

Test 24 : Plaque d'immatriculation non détecté.

Ces résultats nous donnent donc un taux de réussite de 86%. Les erreurs ont pour cause principale les limites du module camera et donc la qualité de l'image.

Ce qui suit sont les images ou l'exécution de l'algorithme a été un échec.

Figure 48 Test 6

Figure 50 Test 7 - Test 24

Figure 49 Test 19

Quelques exemples de résultats réussis :

Figure 51 Test 5

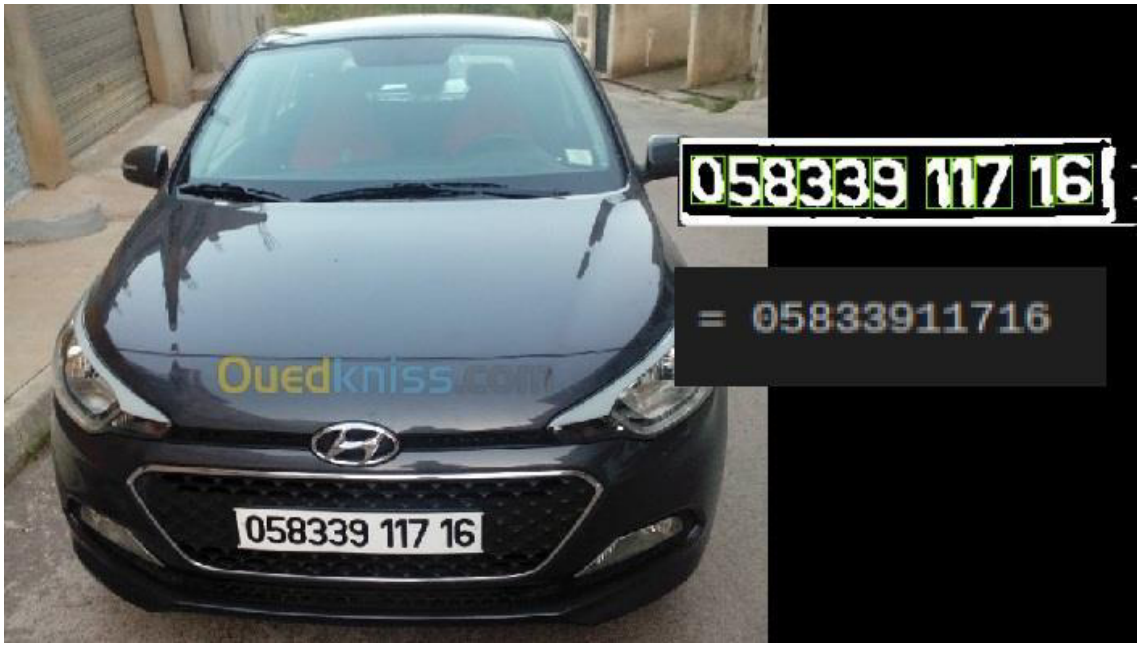


Figure 52 Test 20

4.6 Conclusion

Nous avons dans ce chapitre décrit l'environnement matériel et logiciel de notre projet. Nous avons réalisé les objectifs que nous nous sommes fixés au départ d'implémenter un algorithme de Machine Learning pour traitement d'image dans le cas d'un smart parking.

Nous avons effectué plusieurs tests et nous nous sommes confrontés à certaines difficultés relatives notamment à la qualité de du module caméra à notre disposition, nous avons néanmoins eu des résultats relativement bons avec cet algorithme.

Notre projet même si nous le présentons sous forme de maquette miniature peut parfaitement être mis en place dans un parking existant et contribuer à améliorer l'expérience de ses utilisateurs.

Conclusion générale

Notre travail avait pour but initial de développer un module de caméra capable de lire et reconnaitre les plaques d'immatriculation de véhicules dans un système de parking automatisé et d'assurer une bonne gestion de parking. Au début, une image est capturée puis la région de la plaque est extraite. Après avoir séparé l'insigne, les caractères sont séparés individuellement par segmentation des contours. Ensuite, la méthode de classification du Machine Learning est utilisée pour reconnaitre les caractères de la plaque. Enfin, le numéro d'immatriculation obtenu est enregistré dans la base de données.

Certaines des difficultés possibles qui peuvent survenir sont : Image floue ou mauvaise résolution, plaque cassée, couverte ou sale, mauvais éclairage, temps de traitement (image à grande résolution), limite matériel (hardware), etc...

Ce projet a fait objet d'une expérience intéressante qui nous a permis de renforcer et d'enrichir nos connaissances concernant le traitement d'image et les systèmes autonomes tout au long de notre cursus universitaire. Nous avons aussi pu acquérir de nouvelles notions dans le domaine du Machine Learning et l'intelligence artificielle. Plusieurs technologies ont été nécessaires pour la réalisation de notre projet, citons : Le langage Python, la bibliothèque de traitement d'image Opencv, la carte Raspberry Pi, etc...

A travers cette réalisation, nous avons pu :

- ❖ Tester notre capacité d'analyse de sorte à soulever une problématique, puis de proposer des solutions en se basant sur des démarches scientifiques.
- ❖ Concrétiser nos connaissances théoriques.
- ❖ Voir l'importance d'adapter une bonne démarche de développement.
- ❖ S'ouvrir au milieu professionnel.

Nous avons proposé des solutions afin d'atteindre les objectifs fixés au départ. Nous espérons que ce travail sera à l'origine d'une mise en place réelle de notre application.

Toutefois des perspectives d'améliorations de notre projet restent toujours envisageables. À cet effet, nous évoquons ci-dessous les points manquants comme des perspectives futures de l'application :

- Utiliser les réseaux de neurones (Deep Learning) pour améliorer la précision de la reconnaissance.
- Intégration de ce module de reconnaissances de plaques d'immatriculation à d'autres applications.
- Concevoir une application mobile avec géolocalisation de réservation de places et suivie en temps réel du temps et coût de stationnement sur plusieurs smart parkings et aussi des places de stationnements dans les rues.
- Améliorer le module de reconnaissance afin d'identifier le type de véhicule ainsi que les plaques d'immatriculation étrangères.

Références

[1] : Mokhtar TAFFAR, Initiation à l'apprentissage automatique, Support de cours, 2009.

[2] : Amélien Géron, Machine learning avec scikit-learn, Dunod, 2017.

[3] : Pirmin Lemberger, Big data et machine learning Manuel du data scientist, Dunod, 2015.

[4] : I. Bloch, Gousseau, H. Matre, Matignon, Le traitement des images tome 2, Polycopié du cours. 2004.

[5] : BENDAOUD Mohammed Habib. Développement de méthodes d'extraction de contours sur des images à niveaux de gris, Thèse de Doctorat, 2017.

[6] : Scott E Umbaugh, Digital image processing and analysis, CRC Press, 2010.

[7] : Hinde ANOUAL. Application à la détection des plaques d'immatriculation marocaines, Thèse de Doctorat, 2012.

[8] : Bryan WC Chung, Pro processing for images and computer vision with OpenCV, Apress, 2017.

[9] : D.Boukhlouf, Généralités sur le traitement d'images, Mémoire de Master, 2005.

[10] : Halina Kwasnicka, lakhmi C. Jain, Innovations in intelligent image alalysis, Springer, 2011.

[11] : Bryan WC Chung, Multimedia programming with pure data, Packt publishing, 2013.

[12] : Hector Cuesta, Practical data analysis, Packy publishing, 2016.

[13] : Alan Parkin, Digital imaging primer, Springer, 2016.

[14] : Nisha Gupta, Automatic Number Plate Recognition using raspberry pi2, Mémoire de Master, 2016.

[15] : Bhavith Kumar, Shravan Kanagokar, Trevor Loren, U. Akshay Kini, Vijetha U, SmartPark Smart Parking Lot System, American Journal of Intelligent Systems, 2017.

[16] : Puneeth G, Rohit Anand, Saurabh Jaiswal, ANPR Camera Based Smart Parking Management System. International, Journal of Engineering Research in Electronics and Communication Engineering (IJERECE), 2017.

[17] : Eben Upton et Gareth Halfacree, Raspberry Pi User Guide, WILEY, 2013.

[18] : Rushi Gajjar, Raspberry Pi Sensors, Packt publishing, 2015,

[19] : Gérard Swinnen, Apprendre à programmer avec Python, Creative commons, 2005.

[20] : Alexander Mordvintsev et Abid K, OpenCV-Python Tutorials, Documentation Release 1, 2017.

[21] : Joe Minichino, Joseph Howse, Learning OpenCV 3 Computer Vision with Python Second Edition, Packt publishing, 2015,

[22] : Les modèles d'apprentissage Déductifs et Inductifs, Site web, Publié 15 Mai 2012. Récupéré le 10 Mai 2019 sur https://pourledeveloppementrh.wordpress.com/2012/05/15/les-modeles-dapprentissage-deductifs-et-inductifs/

[23] : Divya Singh, How to Make Machine Learning Models for Beginners, Site web, publié le 4 juin 2019, Récupéré le 10 Mai 2019 sur https://pourledeveloppementrh.wordpress.com/2012/05/15/les-modeles-dapprentissage-deductifs-et-inductifs/

[24] : Data mining, Site web, Récupéré le 16 juin 2019 sur : https://en.wikipedia.org/wiki/Data_mining

[25] : Rajkumar P, Domaines de l'apprentissage automatique, Site web, Publié le 10 Aout 2014, Récupéré de 13 Mai 2019 sur https://bigdata-madesimple.com/14-useful-applications-of-data-mining/

[26] : Frederic Devernay, Détection de contours, Site web, Publié le 22 mars 2005, Récupéré le 20 Mai 2019 sur http://devernay.free.fr/cours/vision/

[27] : Base de donnée et Python, Site web, Récupéré le 30 Mai 2019 sur https://python-django.dev/page-database-data-base-donnees-query-sql-mysql-postgre-sqlite

[27] : Régression linéaire, Site web, Récupéré le 22 juillet 2019 sur https://fr.wikipedia.org/wiki/R%C3%A9gression_lin%C3%A9aire