République Algérienne Démocratique et Populaire
الجمهوريـــة الجزائريـــة الديمقراطيـــة الشعبيـــة
Ministère de l'enseignement supérieur et de la recherche scientifique
وزارة التعليـــم العـــالي والبحث العلمـــي
Université Mohamed el-Bachir el-Ibrahimi Bordj Bou Arreridj
جـــامعة محمد البشير الابراهيمي برج بوعريريج

Faculté des Sciences et la Technologie

Département d'Electronique

Rapport de fin d'études

Présenté pour obtenire

LE DIPLOME DE LICENCE

Spécialité : ELECTRONIQUE INDUSTRIELLE

Thème :

Détection et lecteur de Qr code avec OpenCV

Réalisé par :

Mohamadi Redha Badr Eddine

Laati Taha Mohamed Ayyoub

Laksi Aymen

Encadré par : Dr. Sid Ahmed Soumia

Promotion : 2020/2021

Remerciements

Louange à الله,seigneur de l'univers.

Au terme de ce travail nous tenons tout d'abord à exprimer nos profondes gratitudes à notre encadreuse

Dr.SID AHMED SOUMIA.

Qui nous a guidés tout au long de ce travail.

Nous remercions toutes les personnes ayant contribué et faciliter la réalisation de ce travail dans de bonnes conditions.

Dédicaces

Nous dédions ce modeste travail

À nos très chers parents

À toute Nos familles et nos amis

À tous ceux qui nous ont Encouragé et

Soutenu …

ملخص :

تعتبر الرؤية الحاسوبية وتسمى أيضا **الرؤية االاصطناعية أو الرؤية الرقمية** من فروع الذكاء الاصطناعي التي تمكن الجهاز من فهم ما يراه عند توصيله بكاميرا واحدة او اكثر. ،هدفه الرئيسي هو بناء تطبيقات ذكية قادرة على فهم محتوى الصور كما يفهمها الانسان.حيث من الممكن أن تأخذ بيانات الصور عدة أشكال كالصور المتعاقبة **فيديو**، المشاهد من عدة كاميرات ،بيانات ذات عدة أبعاد مأخوذة من جهاز تصوير طبي.تتنوع التطبيقات التي تسعى إليها رؤية الكمبيوتر مثال الكشف عن الاشخاص بكاميرات الشوارع. وقد أظهر التعاون بين مجالي دراسة الرؤية الفيزيولوجية والحاسوبية تطورا في تعميق الفهم لكلا المجالين . هذا المشروع يسمح لنا بدراسة كيفية كشف وقراءة رمز الاستجابة السريعة بإستخدام Pyzbar, OpenCV ونحن متحمسون لرؤية كيف يتم الكشف عنه بطريقة ذكية.

الكلمات المفتاحية : كشف وقراءة رمز الاستجابة السريعة ،مكتبة pyzbar معOpenCV.

Résumé :

La vision par ordinateur (aussi appelée vision artificielle ou vision numérique) est l'un des domaines de l'informatique, dont l'objectif principal est de construire des applications intelligentes capables de comprendre le contenu des images comme une personne les comprend. Là où les données d'image peuvent prendre de nombreuses formes telles que des images successives (vidéo), des scènes de plusieurs caméras, des données de plusieurs dimensions extraites d'un appareil d'imagerie médicale. La collaboration entre les domaines des études physiologiques et de la vision par ordinateur a montré une avancée dans l'approfondissement de la compréhension des deux domaines. Ce projet nous permet d'étudier comment détecter et lire le code QR en utilisant Open CV, Pyzbar.

Mots clés : détection et lecture du code QR, la bibliothèque Pyzbar avec Open CV.

Summary:

Computer vision is a branch of AI that enables a device to understand what it sees when it is connected to one or more cameras. It seeks to automate the tasks performed by the human visual system, as well as related to the automatic extraction and analysis of useful information from one or more images. Computer vision seeks various applications, such as detecting people with street cameras. This project allows us to study how to detect and read the QR code using Open CV, Pyzbar.

Keywords: detection and reading QR code, Pyzbar and OpenCV.

Table des matières

Liste des figures :

Introduction générale

La vision par ordinateur est une branche de l'intelligence artificielle dont le but est de permettre à une machine de comprendre ce qu'elle «voit » lorsqu'on la connecte à une ou plusieurs caméras. Basé sur un ensemble de méthodes et de technologies, la vision par ordinateur permet d'automatiser une tâche spécifique à partir d'une image.

La vision par ordinateur extrait l'information des images et reconnaît des concepts spécifiques. Elle peut donc effectuer une variété de tâches telles que la reconnaissance de visages ou de caractères dans une image, la détection de l'emplacement d'un objet dans une image ou la classification d'images. La détection d'objets consiste à rechercher un élément particulier et à le localiser dans une image, à l'aide d'une « boîte ».

OpenCV (**Open** source **C**omputer **V**ision Library) est une bibliothèque open source développée par Intel Corporation. Le manuel officiel est disponible gratuitement sur le site Internet d'OpenCV. Il a été conçu pour divers objectifs tels que l'apprentissage automatique, la vision par ordinateur, l'algorithme, les opérations mathématiques, la capture vidéo, le traitement d'images, etc., Android, iOS).

Le but de ce travail n'est pas d'avoir une compréhension théorique approfondie du domaine de la vision et des algorithmes de détection, mais de mettre en pratique certaines fonctions d'OpenCV afin de modéliser et concevoir une application de pour détecter un Qr Code puis le lire.

Notre projet est décomposé en deux parties principales :

Partie I :

Cette partie est la partie théorique du projet ; nous donnons donc d'abord quelques définitions où nous parlerons du domaine de la vision par ordinateur et ces domaines d'application, puis nous parlons des caractéristiques des images numériques car dans notre projet nos données d'entrée sont des images, et dernièrement nous expliquerons la structure de la bibliothèque OpenCV et pourquoi avons-nous particulièrement besoin de cette bibliothèque pour réaliser notre projet

Partie II :

Cette partie est la partie expérimentale du projet ; dans cette partie, nous parlerons de notre application et nous expliquerons chaque étape et les fonctions nécessaires de la bibliothèque OpenCV dont nous aurons besoin pour obtenir le résultat souhaité.

Et nous terminerons par la conclusion générale.

PARTIE I

I. LA VISION PAR ORDINATEUR :

La vision par ordinateur (aussi appelée vision artificielle ou vision numérique) est une branche de l'intelligence artificielle (IA) dont le principal but est de permettre aux ordinateurs d'identifier et de comprendre les objets et les personnes dans les images et les vidéos. La vision par ordinateur cherche à exécuter et à automatiser les tacher qui répliquent les capacités humaines, elle cherche à répliquer la manière dont les êtres humains voient et la façon dont les êtres humains donnent un sens à ce qu'ils voient [1]. Pour nous, humains, cela est assez simple et nous le faisons depuis que nous sommes enfants, mais pour les ordinateurs cela peut être une tache assez complexe.

Dans ce mémoire, nous nous sommes particulièrement intéressés au domaine de la vision par ordinateur en tant que branche de l'IA mais on ne peut pas l'aborder sans parler « comprendre » du domaine du traitement d'image et les caractéristiques des images numériques.

I.I. QU'EST-CE QU'UNE IMAGE :

Une image est une représentation planaire d'une scène ou d'un objet situé en général dans un espace tridimensionnel, elle est issue du contact des rayons lumineux provenant des objets formants la scène avec un capteur (caméra, scanner, rayons X, …). Il ne s'agit en réalité que d'une représentation spatiale de la lumière. L'image est considérée comme un ensemble de points auquel est affecté une grandeur physique (luminance, couleur). Ces grandeurs peuvent être continues (image analogique) ou bien discrètes (images digitales). [2]

Les images numériques : Sont constituées d'un ensemble de pixels (picture éléments), juxtaposés en lignes et en colonnes. Le pixel, (qui correspond à un point ou petit carré), est le plus petit élément que l'on peut trouver dans une image. Chaque pixel possède des caractéristiques propres, couleurs, luminosité, brillance, qui permettent de les différencier et de composer les images.

Figure 1 : Représentation matricielle d'une image numérique

I.I.1. Types d'image :

L'image est une représentation visuelle d'une personne ou d'un objet par la peinture, dessin, photographie, film…etc. elle peut être en plusieurs types :

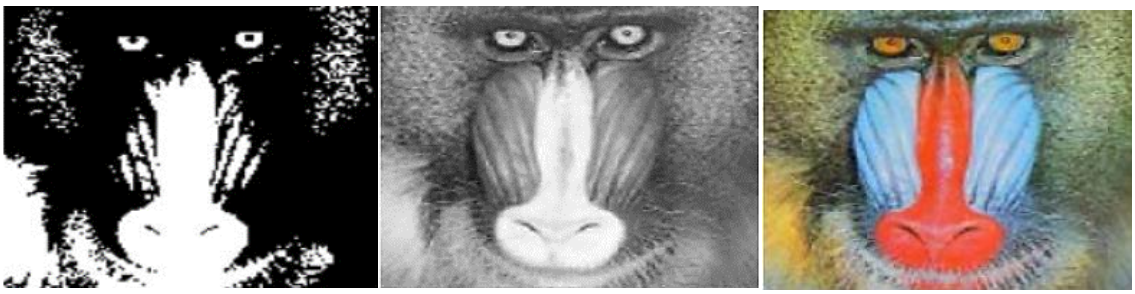
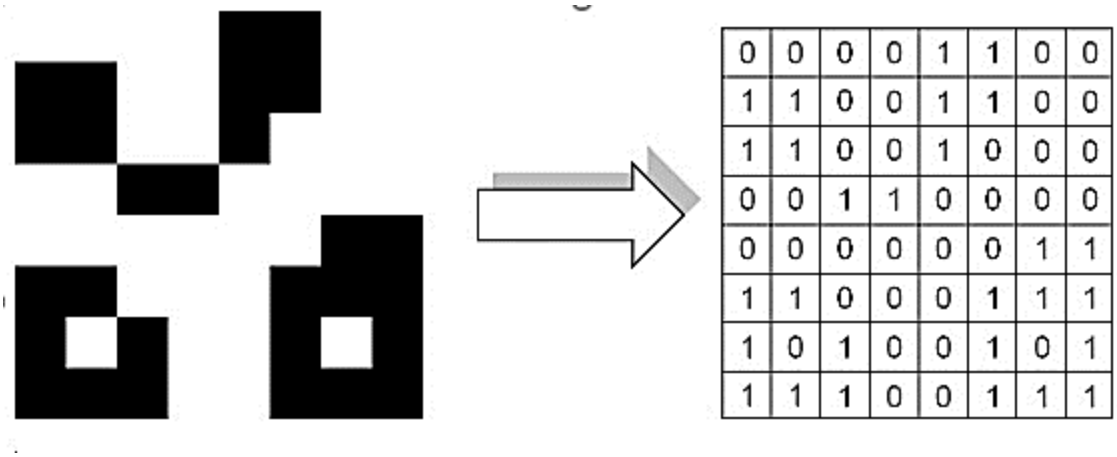


Image binaire Image en niveau de gris Image couleur (RGB)

Figure 2: Les trois types d'image.

A. Image binaire (noire et blanc) :

Les images binaires sont les plus simples., est une image numérique qui a seulement deux valeurs possibles pour chaque pixel. En règle générale, les deux couleurs utilisées pour une image binaire sont en noir et blanc. L'image de 10000 pixels codée occupe donc 10000 bits en mémoire. [4]



0	0	0	0	1	1	0	0
1	1	0	0	1	1	0	0
1	1	0	0	1	0	0	0
0	0	1	1	0	0	0	0
0	0	0	0	0	0	1	1
1	1	0	0	0	1	1	1
1	0	1	0	0	1	0	1
1	1	1	0	0	1	1	1

Figure 3. Une image binaire, avec le tableau de valeurs correspondant.

B. Image en niveau de gris :

Une image numérique en niveaux de gris est un tableau de valeurs. Chaque case de ce tableau, stocke un niveau de gris, allant de 0 (noir) à 255 (blanc) (**Figure 5)**. La couleur est codée souvent sur un octet soit 8 bits ce qui offre la possibilité d'obtenir 256 niveaux de gris (0 pour la noire et 256 pour le blanc. On peut aussi le faire avec 16 niveau de gris (4 bits).

Figure 4. Les niveaux de gris (0-255).

- **Codage d'une image en niveaux de gris**

Si on code chaque pixel sur 2 bits on aura 4 possibilités (noir, gris foncé, gris clair, blanc). L'image codée sera très peu nuancée. [6]

- Codage en 8 bits par pixel (bpp) => 2^8= 256 possibilités

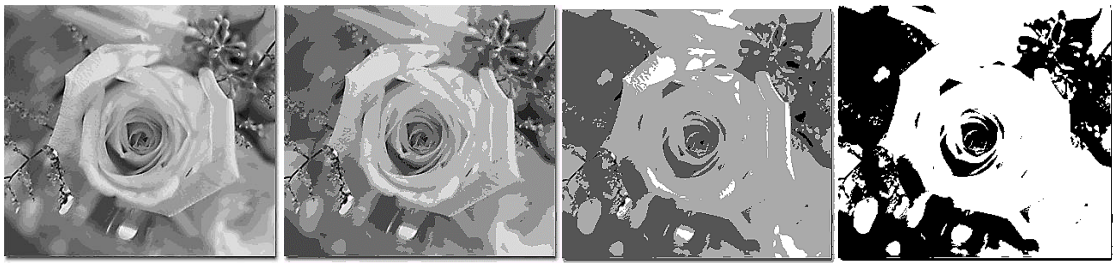
Chaque pixel peut avoir 256 nuances de gris possibles.

- Codage en 16 bits par pixel (bpp) => 2^16= 65536 possibilités

Chaque pixel peut avoir 65536 nuances de gris possibles.

Le nombre de niveaux de gris ou de couleurs a des conséquences sur la qualité de l'image (voir **Figure V**).

256 niveaux de gris 128 niveaux de gris 64 niveaux de gris 32 niveaux de gris



16 niveaux de gris 8 niveaux de gris 4 niveaux de gris 2 niveaux de gris

Figure 5 : Variation du nombre de niveaux de gris pour la même image.

C. Image couleur (RGB)

Chaque pixel possède une couleur décrite par la quantité de rouge (R), vert (G) et bleu (B).

Figure 6 : Espace de couleur RGB.

Chacune de ces trois composantes est codée sur l'intervalle [0, 255], ce qui donne 255^3 =16 777 216 couleurs possibles. Il faut 24 bits pour coder un pixel.

Figure 7 : Image couleur.

- **Codage d'une image en couleurs 8 bits**

Dans ce cas on attache une palette de $2^8 = 256$ couleurs à l'image. Chaque code (de 0 à 255) désigne une couleur choisie parmi les 16 millions de couleurs de la palette RVB (voir ci-après) de manière pertinente ; c'est à dire qu'un programme recherche les couleurs les plus adaptées.

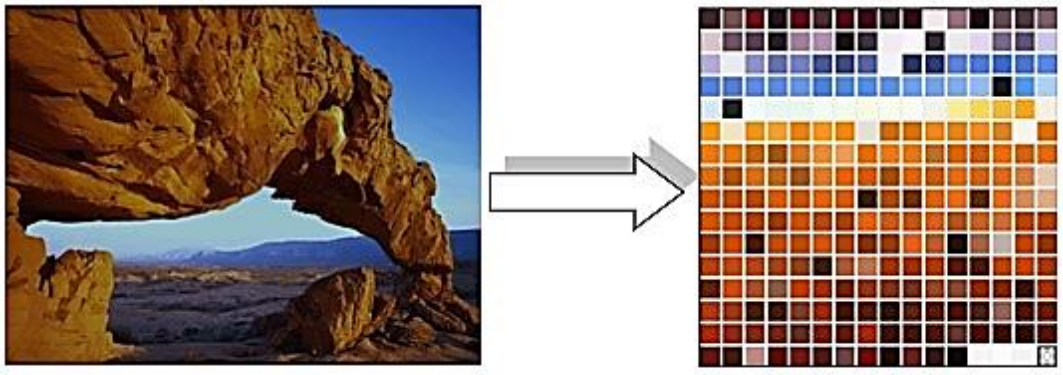


Figure 8 : Image codée sur 8 bits.

I.I.2. Principaux formats graphiques

A. BMP (BitMaP)

Nous avons vu auparavant qu'une image était constituée d'un ensemble de points, nommés pixels, de position et de couleur différente. C'est ce qu'on appelle une image du type « bitmap », ou en français « tableau de données binaires », ce format a été créé par Microsoft.

B. JPEG

Le format JPEG est un format de compression des images graphiques. Il est sans doute le mode de compression le plus efficace qui soit, avec un bon compromis entre gain d'espace disque, temps de compression/décompression et qualité des images. Ainsi une image brute de 2Mo n'occupera après conversion en JPEG que 130 à 400 Ko, selon la qualité d'image voulue.

C. TIFF

Le format TIFF, conçu à l'origine par la compagnie Adlus est un format matriciel. Le format TIFF offre l'avantage d'occuper moins d'espace disque, grâce à son propre algorithme de compression appelé LZW.

D. GIF

Ce format a été créé pour faciliter le transport des images. Il est développé par Compuserve, un des premiers fournisseurs d'accès Internet, est particulièrement bien adapté (vu sa forme compacte), aux applications de transmission de données. Les fichiers sont au format pixel et peuvent gérer jusqu'à 256 couleurs.

I.I.3. Caractéristiques de l'image :

A. Pixel :

Pixel est une abréviation du mot "Picture élément", qui est une unité de surface utilisée pour définir la base d'une image numérique. Il implémente un point donné (x, y) sur le plan image. L'information représentée par le pixel est l'échelle de gris (ou la couleur) obtenue à partir de la position correspondante dans l'image réelle.

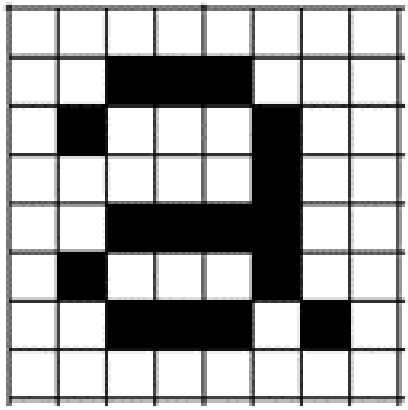


Figure 9 : groupe de pixel formant la lettre A.

B. Dimension et Résolution :

Dimension est la taille de l'image. Il se présente sous la forme d'une matrice dont les éléments sont des valeurs représentatives de l'intensité lumineuse (pixels). Le nombre de lignes de cette matrice multipliée par le nombre de colonnes donne le nombre total de pixels dans l'image. En revanche, la résolution est la clarté ou la finesse des détails obtenus par un écran ou une imprimante lors de la génération d'une image. Sur un écran d'ordinateur, la résolution est exprimée en Le nombre de pixels par unité de mesure (pouces ou centimètres). Le terme résolution est également utilisé pour désigner le nombre total de pixels horizontaux et verticaux sur l'écran. Plus le nombre est grand, meilleure est la résolution. [3]

C. Contraste :

C'est une nette opposition entre deux zones de l'image de contraste a une bonne dynamique des valeurs de gris dans toute la plage de valeurs possibles, avec un blanc très clair et un noir foncé. En revanche, dans les images à faible contraste, la plupart des pixels ont des valeurs de gris très similaires. [3]

D. La texture :

Une texture est une région dans une image numérique qui a des caractéristiques homogènes. Ces caractéristiques sont par exemple un motif basique qui se répète .la texture est composée de Texel, l'équivalent des pixels [3].

E. Bruit :

Le bruit (parasite) dans une image est considéré comme un phénomène où l'intensité d'un pixel par rapport à ses voisins change soudainement, et il provient de l'éclairage de l'optique du capteur et de l'électronique. C'est une sorte de parasite et représente certains dysfonctionnements (poussière, petits nuages, chute instantanée de la force électrique sur le capteur, etc.) Il provoque des taches de petite taille, dont la répartition sur l'image est aléatoire. [3]

Figure 10 : Image bruitée

F. Contour :

Le contour représente la limite entre les objets de l'image, ou le niveau de gris représente la limite entre deux pixels qui sont significativement différents. La texture décrit la structure de ceux-ci. L'extraction de contour comprend l'identification de points dans l'image qui séparent deux textures différentes. [3]

Figure 11 : Contour d'une image

G. Luminance :

C'est le degré de luminance des points de l'image, elle représente le quotient de l'intensité lumineuse d'une surface par l'aire appartenant de cette surface. Souvent le mot luminance est substitué au mot brillance qui correspond à l'éclat d'un objet. [3]

H. Histogramme :

L'histogramme des niveaux de gris ou des couleurs d'une image est fonction de la fréquence à laquelle chaque niveau de gris (couleur) apparait dans l'image affichée. Il fournit de nombreuses informations sur la distribution des niveaux de gris (couleurs) et vous pouvez voir dans quelles limites la plupart des niveaux de gris (couleurs) sont distribués lorsque l'image est trop sombre. [3]

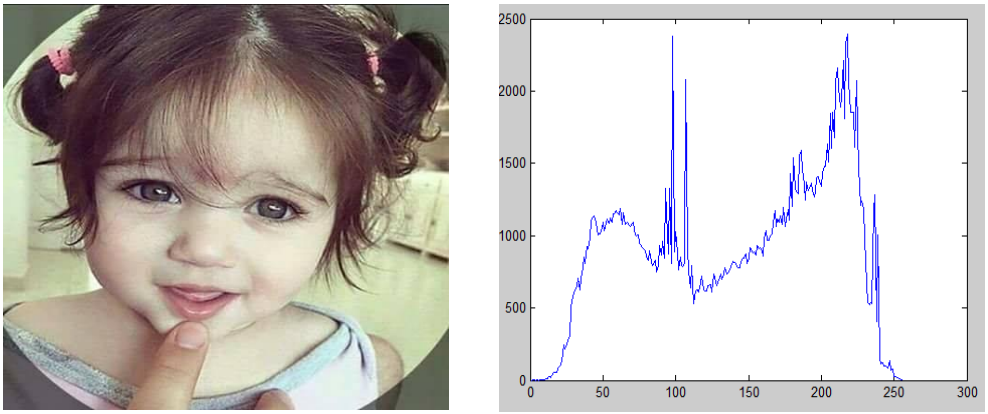


Figure 12 : Image avec histogramme.

I. Poids de l'image :

Tout d'abord, le poids d'une image se mesure en octets (ou avec l'un de ses multiples : Ko, Mo, Go). Un octet est une unité de mesure informatique. Pour simplifier, il vous indique la place que va occuper un fichier électronique sur un disque dur (qu'il soit traditionnel ou SSD).

Ensuite, le poids d'une image sera fonction de nombreux facteurs, du format (.jpeg, .png, .bmp…) au niveau de couleurs possible (8 bits, 14 bits, etc.), en passant par sa définition (en pixels), la compression utilisée ou encore le nombre de couches.

Si chaque couleur primaire est codée sur 8 bits (on parlera aussi de la profondeur de la couleur) cela veut dire que chaque pixel utilisera 1 octet pour indiquer la valeur du Rouge, 1 octet pour indiquer celle du Vert et 1 octet pour celle du bleu. Soit 3 octets par point.[4]

J. Transparence :

Au sens propre, la transparence est le caractère de ce qui est transparent, qui se laisse traverser par la lumière en laissant voir les formes et les couleurs. La translucidité, quant à elle, ne transmet que la lumière.[5]

I.II. Qu'est-ce que la vision par ordinateur

La vision par ordinateur peut être définie comme la science des machines, des robots, des systèmes informatiques et de l'intelligence artificielle qui analysent des images et des vidéos, reconnaissent des objets et agissent en conséquence. Il s'agit donc d'une branche de l'intelligence artificielle dont l'objectif est d'analyser des informations contenues dans des images ou des vidéos. (Figure 13) [6]

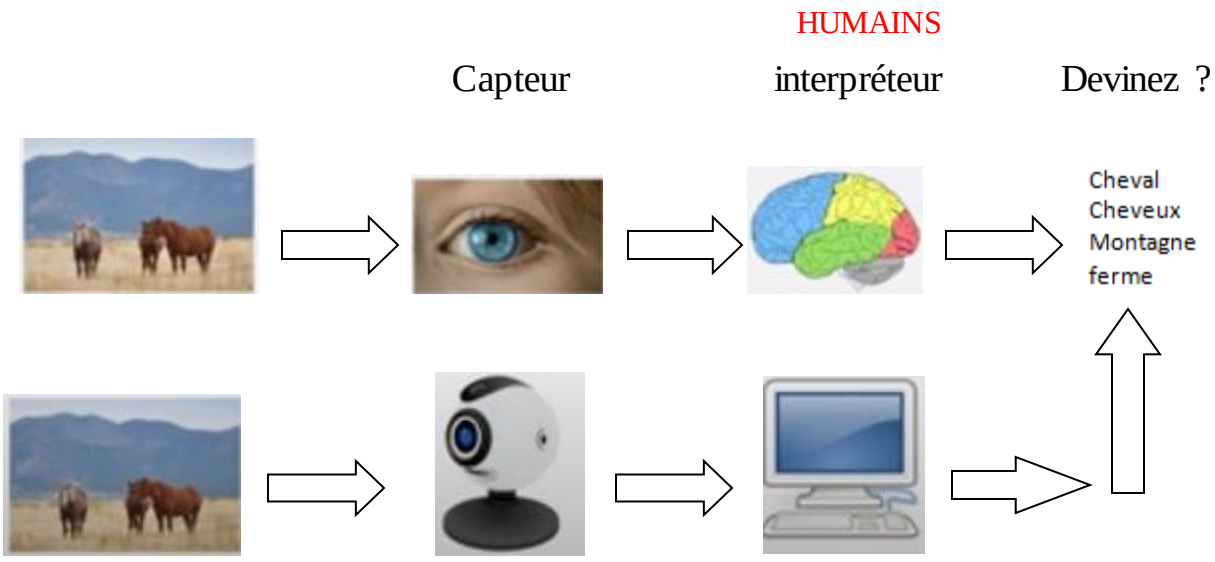


Figure 13: Qu'est-ce que la vision par ordinateur.

Du point de vue de l'ingénierie, il cherche à automatiser les tâches que le système visuel humain peut effectuer. "La vision par ordinateur concerne l'extraction, l'analyse et la compréhension automatiques des informations utiles d'une image unique ou d'une séquence

d'images. Elle implique le développement d'une base théorique et algorithmique permettant la compréhension visuelle automatique".

I.II.1. DOMAINES D'APPLICATION DE LA VISION PAR ORDINATEUR :

À quoi peut bien être utile un ordinateur qui « voit » ? Les domaines d'application sont immenses, nous listons ici quelques cas d'usage :

- ✓ Imagerie médicale
- ✓ Manufacturing / Supply Chain
- ✓ Sécurité / Sûreté
- ✓ Recherche scientifique
- ✓ Informatique de gestion

A. Imagerie médicale :

L'application de l'informatique à l'imagerie médicale représente l'une des plus grandes promesses de la vision par ordinateur. Imaginez par exemple un radiologue qui serait assisté par un logiciel permettant de détecter des anomalies sur un cliché. Le gain de temps pour le radiologue et le gain en sécurité pour le patient peuvent être énormes s'il est possible de réaliser un logiciel suffisamment fiable.

Ces logiciels ultra-spécialisés commencent à apparaitre, et permettent déjà de repérer des éléments sur des radios avec une acuité égale voire supérieure à celle de l'humain. Il s'agit là d'une véritable révolution dans le domaine de la médecine ! L'imagerie médicale bénéficie ainsi des caractéristiques intrinsèques de l'algorithme : pouvoir être exécuté à grande échelle, avec un résultat constant. En somme, c'est comme si le médecin se dotait d'un assistant qui n'est jamais fatigué et qui donne son analyse avec toujours la même acuité. Plus de patients traités avec moins de risques d'erreurs, n'est-ce pas là une noble cause ? [7]

Figure 14 : Un logiciel de diagnostic médical (Imagerie par Résonance Magnétique).

B. Manufacturing / Supply Chain :

L'industrie et la distribution sont des métiers qui ne sauraient se passer d'un œil vigilant. Dans une usine, le contrôle qualité est très souvent réalisé visuellement, il s'agit d'une activité qui présente un risque potentiel en cas d'inattention, et qui pourtant est fondamentale à la bonne marche d'une chaine de fabrication ; et s'il était possible de multiplier les points de contrôle pour détecter les défauts le plus tôt possible dans le processus de fabrication ? Les économies en termes de taux de rebut et de réactivité sont énormes.

La vision par ordinateur permet de détecter des anomalies avec un processus rigoureux, prouvé et répétable. Ainsi, plus de risque d'inattention et un repérage au plus tôt des défauts potentiels. [7]

Par exemple la figure 15 ci-dessous montre l'intégration de la vision industrielle dans une chaîne de contrôle proposée par IntervalZero qui a investi dans de nouveaux partenariats afin de fournir une solution de vision industrielle en temps réel à ses clients RTX64.

Pour en savoir plus sur leurs solutions de vision, vous pouvez télécharger leur livre sur[1].

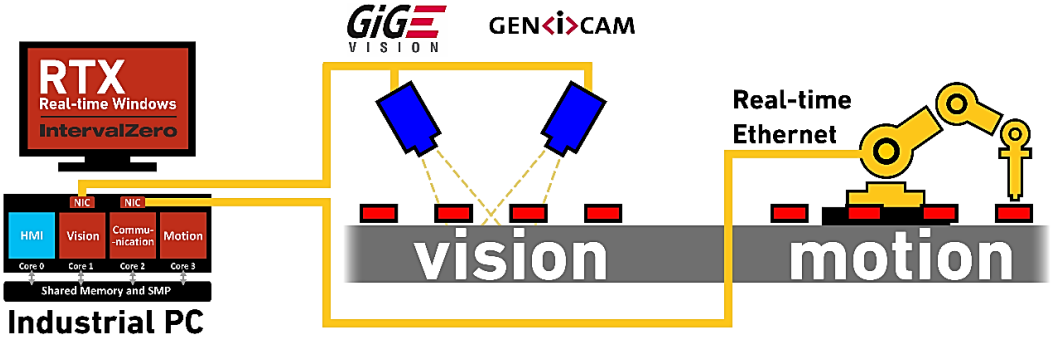


Figure 15 : Une plateforme unique pour les systèmes de contrôle industriels[1]

C. Sécurité/ Sureté :

Tout comme l'industrie et la distribution, la sécurité se doit d'avoir un oeil vigilant, tout particulièrement aujourd'hui.

Dans les rues, les magasins, les musées, les banques il y a des caméras de surveillance qui filment et enregistrent des images. La reconnaissance visuelle permet d'analyser les images enregistrées et d'y repérer des visages et des personnes.

Une inspection visuelle automatisée avec des drones est possible et entraine un gain de temps prodigieux lors de la détection d'un délit. [7]

Figure 16 : Exemple d'application de reconnaissance faciale, installé pour des raisons de sécurité.

[1] https://www.intervalzero.com/french/vision-industrielle/

D. Recherche scientifique :

La vision par ordinateur permet d'identifier des choses invisibles à l'oeil humain. Il est possible de repérer de petits éléments dans une grande image et ainsi prévenir le diagnostic de maladies qui se détectent habituellement trop tard.

De plus, la vision assistée par ordinateur facilite les études et réduit les erreurs humaines liées notamment à l'inattention. [7]

E. Informatique de gestion :

La vision par ordinateur peut classer un grand nombre d'images en temps réel, ce qui aide à la collecte et à l'organisation des données indispensables à l'informatique de gestion.

Pour finir, la vérification des données peut être faite par inspection visuelle automatisée, ce qui entraine encore une fois un gain de temps. [7]

Figure 17: Image montrant Informatique de gestion aujourd'hui.

II. OpenCV :

II.I. Historique :

Officiellement lancé en 1999, le projet OpenCV est développé initialement par Intel pour optimiser les applications gourmandes en temps processeur. Cela faisait partie d'une série de projets tel que l'affichage d'un mur en 3 dimensions. Cette bibliothèque est distribuée sous licence BSD.

Les principaux acteurs du projet sont l'équipe de développement de bibliothèque de chez Intel ainsi qu'un certain nombre d'experts dans l'optimisation de chez Intel Russie.

Les objectifs de base du projet étaient :

- Faire des recherches sur la vision par ordinateur en vue de fournir un logiciel libre et optimisé.

- Établir une infrastructure commune s'appuyant sur les développeurs pour obtenir un code plus lisible et transférable.

- Continuer à développer en rendant le code portable et permettre des performances optimisées gratuites avec une licence qui est libre de toutes contraintes commerciales.

La première version alpha d'OpenCV fut présentée lors de la conférence IEEE sur la vision par ordinateur et la reconnaissance de formes en 2000. Après cela, cinq versions bêta ont été publiées entre 2001 et 2005 et la première version 1.0 a été publiée en 2006.

Au milieu de l'année 2008, OpenCV obtient l'appui de la société de robotique Willow Garage et la bibliothèque est encore développée à ce jour. Une version 1.1 est sortie en Octobre 2008 et un livre écrit par deux auteurs d'OpenCV, publié par O'Reilly Media est sorti sur le marché ce même mois.

La deuxième version majeure d'OpenCV née en octobre 2009. Il s'agit d'OpenCV 2 incluant des changements majeurs au niveau du langage C++ servant à faciliter le développement de nouvelles fonctions et améliorant les performances. [8]

II.II. Qu'est-ce que OpenCV ?

Initialement développée par Intel, OpenCV (Open Computer Vision) est une bibliothèque graphique. Elle est spécialisée dans le traitement d'images, que ce soit pour de la photo ou de la vidéo.

Sa première version est sortie en juin 2000. Elle est disponible sur la plupart des systèmes d'exploitation et existe pour les langages Python, C++ et Java ; Sous licence BSD (Berkeley Software Distribution Licence), OpenCV peut être réutilisé librement, en tout ou partie, pour être intégré au sein d'un autre projet.

C'est notamment cette notion qui fait qu'OpenCV est très populaire et à la base de nombreux logiciels de traitements d'images/vidéos. Elle est aujourd'hui développée, maintenue, documentée et utilisée par une communauté de plus de 40 000 membres actifs ! [8]

Figure 18 : Logo d'OpenCV

II.III. Les principaux modules d'OpenCV 4.2.0 :

OpenCV a une structure modulaire. Les principaux modules d'OpenCV sont énumérés :

- **CxCore (Cœur d'OpenCV)**

Les fonctionnalités de base :

Permet de manipuler les structures de base, réaliser des opérations sur des matrices, dessiner sur des images, sauvegarder et charger des données dans des fichiers XML…

- **Imgproc (traitement d'image)**

Ce module comprend des algorithmes de traitement d'image de base, y compris le filtrage d'image, transformations d'image, les conversions d'espace couleur et etc.

- **Imgcodecs**

Ce module gère la lecture et l'écriture des images. Lorsque vous traitez par exemple une image d'entrée et que vous souhaitez créer une image de sortie au format jpg ou png, vous pouvez l'enregistrer dans ce format avec une simple commande.

- **Videoio (E / S vidéo)**

Le module d' E / S vidéo OpenCV est un ensemble de classes et de fonctions permettant de lire et d'écrire des séquences vidéo ou d'images.
Fondamentalement, le module fournit les classes cv :: Vidéo Capture et cv :: Vidéo Writer comme interface à 2 couches à de nombreuses API d'E / S vidéo utilisées comme backend.

- **Highgui (librairie d'interfaçage graphique)**

Entrés-sorties et interface utilisateur :

OpenCV intègre sa propre bibliothèque haut-niveau pour ouvrir, enregistrer et afficher des images et des flux vidéo. Celle-ci contient aussi un certain nombre de fonctions permettant de réaliser des interfaces graphiques

- **Vidéo (Traitement de flux vidéo)**

Ceci est une vidéo module d'analyse qui inclut des algorithmes de suivi (tracking) d'objets, des algorithmes de soustraction de fond et etc.

- **Calib3d(Calibrage de la caméra et reconstruction 3D)**

Calibration, estimation de pose et stéréovision :

Ce module contient des fonctions permettant de reconstruire une scène en 3D à partir d'images acquises avec plusieurs caméras simultanément.

- **Features2d(Cadre des fonctionnalités 2D)**

Ce module concerne principalement l'extraction de descripteurs.

- **Objdetect (Détection d'objets)**

Cela comprend l'objet de détection et de reconnaissance des algorithmes pour les objets standard.

- **Dnn (Réseaux de neurones profonds)**

Ce module contient :

- API pour la création de nouvelles couches, les couches construisent des briques de réseaux neuronaux ;
- Ensemble de couches intégrées les plus utiles ;
- API pour construire et modifier des réseaux neuronaux complets à partir de couches ;
- Fonctionnalité de chargement de modèles de réseaux sérialisés à partir de différents Framework.

La fonctionnalité de ce module est conçue uniquement pour les calculs de passes aller (c'est-à-dire les tests de réseau). Une formation réseau n'est en principe pas prise en charge.

- **Ml (librairie d'apprentissage automatique)**

La bibliothèque d'apprentissage automatique (MLL) est un ensemble de classes et de fonctions pour la classification statistique, la régression et le regroupement des données. La plupart des algorithmes de classification et de régression sont implémentés en tant que classes C ++. Comme les algorithmes ont différents ensembles de fonctionnalités (comme une capacité à gérer des mesures manquantes ou des variables d'entrée catégorielles), il existe un petit terrain d'entente entre les classes. Ce terrain d'entente est défini par la classe cv :: ml :: Stat Model dont toutes les autres classes ML sont dérivées.

- **Flann (Clustering et recherche dans des espaces multidimensionnels)**

Cette section documente l'interface d'OpenCV avec la bibliothèque FLANN. FLANN (Fast Library for Approximate Nearest Neighbours) est une bibliothèque qui contient une collection

d'algorithmes optimisés pour la recherche rapide des voisins les plus proches dans de grands ensembles de données et pour des entités de grande dimension.

- **Photo (Computationnels Photography)**

Figure 19. Assemblage d'images

- **Gapi (Graph API)**

OpenCV Graph API (ou G-API) est un nouveau module OpenCV destiné à rendre le traitement d'image régulier rapide et portable. Ces deux objectifs sont atteints en introduisant un nouveau modèle d'exécution basé sur des graphes. [8]

G-API est un module spécial d'OpenCV - contrairement à la majorité des autres modules principaux, celui-ci agit comme un cadre plutôt que comme un algorithme CV spécifique. G-API fournit des moyens pour définir des opérations CV, construire des graphiques (sous forme d'expressions) en l'utilisant, et enfin implémenter et exécuter les opérations pour un backend particulier.

Figure 20: Exemple de pipeline s'exécutant sur l'exemple de vidéo "vtest.avi"

PARTIE II

Partie 2 APPLICATION

I. INTRODUCTION :

Dans cette partie nous allons écrire un programme en Python dans l'IDE (Integrated Development Environment) PyCharm pour détecter et décoder un Qr code dans image ou à partir d'une scène vidéo capturée par une webcam en utilisant la bibliothèque visuelle OpenCV, ainsi que bien évidemment de la librairie pyzbar.

II. Description du projet :

Le but de ce travail PFC est de mettre en pratique certaines fonctions d'OpenCV ; voir comment utiliser les bibliothèques OpenCV , NumPy et Pyzbar. Créons un programme d'authentification qui scanne les codes QR à partir d'une caméra ou une webcam, L'idée est simple, nous avons une liste de Qr code approuvé (Base de données) et nous vérifions si la personne après avoir montré son code Qr si elle est autorisée ou non.

La plupart des programmeurs python sont familiers avec ces bibliothèques. OpenCV est une bibliothèque open source de vision par ordinateur et d'apprentissage automatique. C'est une bibliothèque utile pour le traitement d'images. Nous utilisons cette bibliothèque dans notre projet pour traiter chaque image d'une vidéo capturée par un appareil. Nous utilisons Pyzbar pour lire et décoder les codes QR à partir d'une image détectée.

Pour atteindre notre objectif, nous allons suivre les étapes suivantes pour écrire le code :

1. Importer les bibliothèques : OpenCV et Numpy
2. Importer decode de pyzbar

Partie1 :

1. Charger l'image qui contient un Qr Code.
2. Utilisez la fonction de décodage $decode()$ et montrer ce qu'elle retourne en utilisant la fonction $print()$;

Partie2 :

1. Lire le flux vidéo de la webcam et définir sa la largeur et sa hauteur.
2. Utiliser la fonction de décodage $decode()$ pour décoder le Qr Code capturé par la webcam

3. Créer une boîte englobante (bounding box) autour de Qr Code à l'aide de polygone retourné par la fonction de décodage. Pour ce faire, vous devez :
    - Convertir le polygone en un tableau NumPy d'entiers $int32$
    - Remodeler le tableau à l'aide de la fonction $reshape()$
    - Utilisez la fonction $polylines()$pour dessiner un polygone sur l'image.
4. Afficher le code sur l'image captée (précisément sur le Qr Code capturé par la webcam) à l'aide de la fonction : *putText ()*. Pour éviter la rotation du texte lors de la rotation de la caméra ; vous utilisez les points du rectangle ($barcode.rect$).

Figure 21: Résultat voulu.

Partie3 :

Dans cette partie, nous voulons écrire un programme d'authentification. L'idée est simple, nous avons une liste de Qr code approuvé et nous vérifions si la personne après avoir montré son code Qr elle est autorisée ou non, par exemple, vérifier si elle est autorisée à entrer dans le laboratoire.

Pour ce faire, nous allons :

1. Créer un fichier DataFile.txt et écrire tous les codes autorisés dans ce fichier
2. Lire toutes les données de DataFile.txt et les stocker dans une liste ; utiliser la commande $file.read().splitlines()$

3. Si le Qr code capturé par la webcam existe dans le fichier DataFile.txt afficher le message ="Autorise" en vert sinon le message 'Non_Autorise' à l'aide de la fonction : *putText ()*

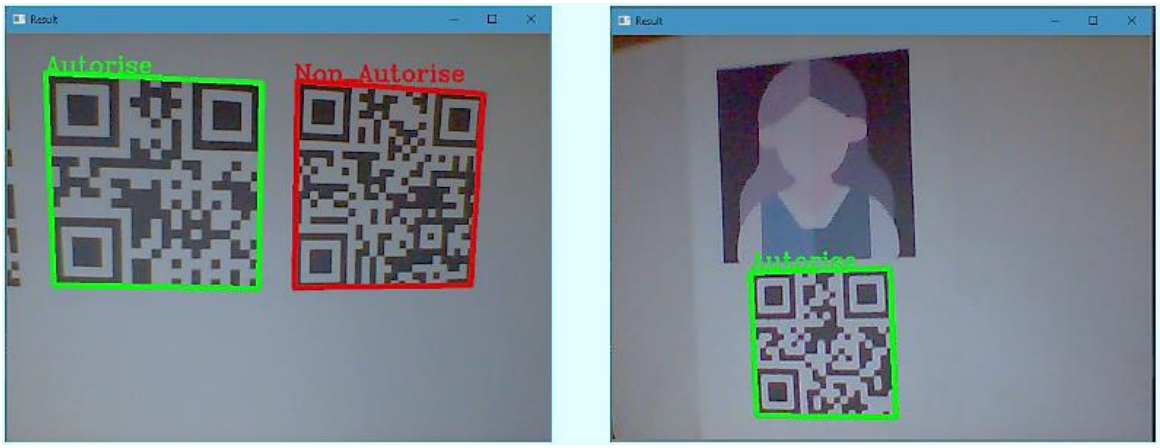


Figure 22:Résultat voulu (Partie 3).

III. Fonctions utiles de la bibliothèque OpenCV :

Un des buts d'OpenCV est d'aider les gens à construire rapidement des applications sophistiquées de vision à l'aide d'infrastructure simple de vision par ordinateur. La bibliothèque d'OpenCV contient plus de 500 fonctions, Il est possible grâce à la « licence de code ouvert » de réaliser un produit commercial en utilisant tout ou partie d'OpenCV. Cette librairie est disponible dans de nombreux langages, Python, Java, Matlab, C#, C++, JavaScript, etc.... Dans notre projet, nous avons choisi de programmer avec le langage de programmation python.

III.1 Pourquoi python ?

Le langage de programmation Python a été créé par Guido van Rossum en 1990 et est rendu disponible sous licence libre. Son développement est aujourd'hui assuré par la Python Software Fondation, fondée en 2001. Il s'agit d'un langage interprété fonctionnant sur la plupart des plateformes informatiques (notamment Linux, Windows et macOS).

Python est un langage qui peut s'utiliser dans de nombreux contextes et s'adapter à tout type d'utilisation grâce à des bibliothèques spécialisées. Il est devenu très rapidement populaire, principalement en raison de sa simplicité et de sa lisibilité du code. Il permet au programmeur d'exprimer des idées en moins de lignes de code sans réduire la lisibilité.

Partie 2 APPLICATION

Dans ce projet, nous utiliserons OpenCV- Python qui est un API Python pour OpenCV, combinant les meilleures qualités de l'API OpenCV C ++ et du langage Python. Nous citons les fonctionnalités utilisées pour la réalisation de ce travail de mémoire.

Comme nos données d'entrée sont des videos, notre première fonction nous aidera à les lire :

- **VideoCapture:**

Syntaxe: $cv2.VideoCapture(video\_path\ or\ device\ index\ )$.
Paramètres:
video_path: emplacement de la vidéo dans votre système sous forme de chaîne avec leurs extensions comme .mp4, .avi, etc.

device index : C'est juste le numéro pour spécifier la caméra. Ses valeurs possibles, càd 0 ou -1.

Valeur de retour : Objet de capture vidéo.

- **cap.set() :**

Syntaxe : $cv2.VideoCapture.set(propid,valeur)$

Paramètres :

Propid : Identificateur de propriété de cv2::VideoCaptureProperties ou un des indicateurs supplémentaires pour les backends d'API d'E/S vidéo.

Valeur : Valeur de la propriété.

Valeur de retour : true si la propriété est prise en charge par le backend utilisé par l'instance VideoCapture.

- **cap.read() :**

Syntaxe : $cv2.VideoCapture.read\ (\ [,image]\ )$

Saisit, décode et renvoie l'image vidéo suivante.

Paramètres :

Image : l'image vidéo est renvoyée ici. Si aucune image n'a été saisie, l'image sera vide.

Valeur de retour : false si aucune image n'a été saisie.

- **open() :**

Syntaxe : $open(,)$

Paramètres :

fichier : objet de type chemin (représentant un chemin d'accès au système de fichiers) mode (facultatif) : lors de l'ouverture d'un fichier. S'il n'est pas fourni, il est par défaut

Valeur de retour : renvoie un objet fichier qui peut être utilisé pour lire, écrire et modifier le fichier. Si le fichier n'est pas trouvé, il lève ‘’FileNotFoundErrorexception’’.

- **read():**

Syntaxe : $file.read()$

Paramètres :

Size : C'est le nombre d'octets à lire à partir du fichier.

Valeur de retour : Cette méthode renvoie les octets lus dans la chaîne.

- **splitlines():**

Syntaxe : $splitlines([garde])$

Paramètres : prend au maximum 1 paramètre.

Garde (facultatif) : Si gardiens est fourni et True, des sauts de ligne sont également inclus dans les éléments de la liste. Par défaut, les sauts de ligne ne sont pas inclus.

Valeur de retour : renvoie une liste de lignes dans la chaîne.

S'il n'y a pas de caractères de saut de ligne, il renvoie une liste avec un seul élément (une seule ligne).

- **decode():**

Syntaxe : $decode(encodage, erreur)$

Paramètres :

Encodage : Spécifie l'encodage à partir duquel le décodage doit être effectué.

Erreur : décide comment gérer les erreurs si elles se produisent, par exemple 'strict' déclenche une erreur Unicode en cas d'exception et ignore les erreurs survenues.

Valeur de retour : Renvoie la chaîne d'origine à partir de la chaîne encodée.

- **np.array():**

Syntaxe : $numpy.array\,(Objet\,, dtype\ =\ None\,, *\,, copie\ =\ True\,, ordre\ =\ 'K'\,, subok\ =\ False\,, ndmin\ =\ 0\,, like\ =\ None\,)$

Paramètres :

Objet : Un tableau, tout objet exposant l'interface du tableau, un objet dont la méthode __array__ renvoie un tableau ou toute séquence (emboîtée).

Dtype : type de données, Le type de données souhaité pour le tableau. S'il n'est pas fourni, le type sera déterminé comme le type minimum requis pour contenir les objets dans la séquence.

Copie : booléen, facultatif Si vrai (par défaut), alors l'objet est copié. Sinon, une copie ne sera effectuée que si __array__ renvoie une copie, si obj est une séquence imbriquée ou si une copie est nécessaire pour satisfaire l'une des autres exigences (dtype, order , etc.).

Ordre : {'K', 'A', 'C', 'F'}, facultatif Spécifiez la disposition de la mémoire de la baie. Si l'objet n'est pas un tableau, le tableau nouvellement créé sera dans l'ordre C (ligne majeure) à moins que 'F' ne soit spécifié, auquel cas il sera dans l'ordre Fortran (colonne majeure). Si object est un tableau, ce qui suit est valable.

Subok : booléen , facultatif Si True, alors les sous-classes seront transmises, sinon le tableau renvoyé sera forcé d'être un tableau de classe de base (par défaut).

Ndmin : entier , facultatif Spécifie le nombre minimum de dimensions que le tableau résultant doit avoir. Les uns seront pré-pendus à la forme selon les besoins pour répondre à cette exigence.

comme : Objet de référence pour permettre la création de tableaux qui ne sont pas des tableaux NumPy. Si un tableau transmis en tant que likesupporte le __array_function__protocole, le résultat sera défini par celui-ci. Dans ce cas, il assure la création d'un objet tableau compatible avec celui passé via cet argument.

Valeur de retour :

out :ndarray Un objet tableau satisfaisant aux exigences spécifiées.

- **reshape():**

Syntaxe : $numpy.reshape\ (\ a\ ,nouvelle\ forme,\ ordre\ =\ 'C'\ )$

Paramètres :

a : Tableau à remodeler.

Nouvelle forme : int ou tuple de ints, La nouvelle forme doit être compatible avec la forme d'origine. S'il s'agit d'un entier, le résultat sera un tableau 1D de cette longueur. Une dimension de forme peut être -1. Dans ce cas, la valeur est déduite de la longueur du tableau et des dimensions restantes.

Ordre : *{'C', 'F', 'A'}, facultatif* Lisez les éléments de *a* en utilisant cet ordre d'index et placez les éléments dans le tableau remodelé en utilisant cet ordre d'index. « C » signifie lire / écrire les éléments en utilisant un ordre d'index de type C, avec le dernier index d'axe changeant le plus rapidement, de nouveau au premier index d'axe changeant le plus lentement. « F » signifie lire/écrire les éléments en utilisant un ordre d'index de type Fortran, le premier index changeant le plus rapidement et le dernier index changeant le plus lentement. Notez que les options 'C' et 'F' ne tiennent pas compte de la disposition de la mémoire du tableau sous-jacent, et se réfèrent

uniquement à l'ordre d'indexation. « A » signifie lire / écrire les éléments dans un ordre d'index de type Fortran si *a* est *contigu* en Fortran en mémoire, dans un ordre de type C dans le cas contraire.

Valeur de retour :

reshaped_array : ndarray Ce sera un nouvel objet de vue si possible ; sinon, ce sera une copie. Notez qu'il n'y a aucune garantie de la disposition de la mémoire (C- ou Fortran-contigu) du tableau renvoyé.

- **cv2.polylines() :**

Est utilisée pour dessiner un polygone sur n'importe quelle image.

Syntaxe : $cv2.polylines(image, [pts], estFermé, couleur, épaisseur)$

Paramètres :

Image : C'est l'image sur laquelle le cercle doit être tracé.

pts : tableau de courbes polygonales. npts : tableau de compteurs de sommets de polygones. ncontours : nombre de courbes.

estFermé : indicateur indiquant si les polylignes dessinées sont fermées ou non. S'ils sont fermés, la fonction trace une ligne du dernier sommet de chaque courbe à son premier sommet.

couleur : C'est la couleur de la polyligne à dessiner. Pour BGR, nous passons un tuple.

épaisseur : C'est l'épaisseur des bords de la polyligne.

Valeur de retour : Elle renvoie une image.

- **cv2.putText() :**

La méthode est utilisée pour dessiner une chaîne de texte sur n'importe quelle image.

Syntaxe : $cv2.putText(image, text, org, font, fontEchelle,$

$couleur$ [,$Épaisseur$ [,$ligneType$[,bas à $gauche\ Origine$]]])

Paramètres :

image : C'est l'image sur laquelle le texte doit être dessiné.

text : chaîne de texte à dessiner.

org : Ce sont les coordonnées du coin inférieur gauche de la chaîne de texte dans l'image. Les coordonnées sont représentées sous forme de tuples de deux valeurs, c'est-à-dire (valeur de coordonnée X , valeur de coordonnée Y).

font : indique le type de police. Certains types de polices sont FONT_HERSHEY_SIMPLEX, FONT_HERSHEY_PLAIN, , etc.

fontEchelle : facteur d'échelle de police multiplié par la taille de base spécifique à la police.

couleur : C'est la couleur de la chaîne de texte à dessiner. Pour BGR , nous passons un tuple. ex : (255, 0, 0) pour la couleur bleue.

épaisseur : C'est l'épaisseur de la ligne en px .

ligneType : C'est un paramètre facultatif. Il donne le type de ligne à utiliser.

bas à gauche Origine : il s'agit d'un paramètre facultatif. Lorsque c'est vrai, l'origine des données d'image se trouve dans le coin inférieur gauche. Sinon, il est dans le coin supérieur gauche.

Valeur de retour : Elle renvoie une image.

- **cv2.imshow() :**

Syntaxe : $cv2.imshow("nom\ de\ fenêtre", image)$.

Paramètres :

nom de fenetre : Une chaîne représentant le nom de la fenêtre dans laquelle l'image à afficher.

image : C'est l'image qui doit être affichée.

Valeur de retour: il ne renvoie rien.

- **WaitKey() :**

L'Attente avant la destruction de la fenêtre, si on met (0), la fenêtre s'affichera indéfiniment tant qu'on n'aura pas cliqué dessus et appuyer sur une touche.

IV. ENVIRONNEMENT DU TRAVAIL \ LES ETAPES D'IMPLEMENTATION :

IV.1 Le Hardware :

Un ordinateur acer avec les caractéristiques suivantes :

➢ Processeur : intel® Core™ i3 CPU M 370 @ 2.40GHz 2.40 GHz.

➢ RAM: 4.00 Go (3.68 Go utilisable).

➢ Disque Dur: 500 Go.

IV.2 Le Software :

➢ Système d'exploitation : Windows 7 professionnel.

➢ Logiciel : PyCharm..

➢ La bibliothèque : OpenCV 4.2.0.

IV.3 Installation et configuration OpenCV avec PyCharm :

Premièrement on doit télécharger PyCharm depuis son site officiel https://www.jetbrains.com/fr-fr/pycharm/ , le logiciel doit être compatible à notre system d'exploitation(Windows).

Afin de vérifier que PyCharm est bien installé et fonctionne correctement, nous allons lancer PyCharm à l'aide du raccourci sur le bureau (ou par le menu Démarrer). (Figure 23, 24)

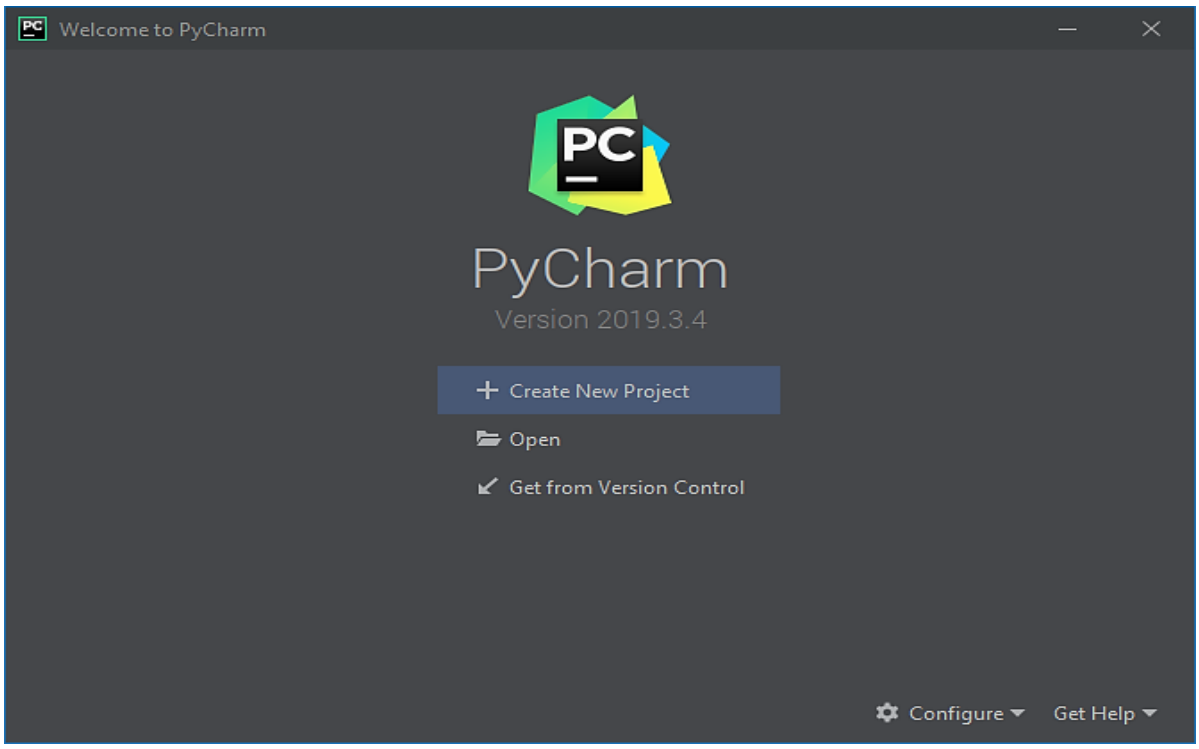


Figure 23 : Ouverture de PyCharm sur Windows 7

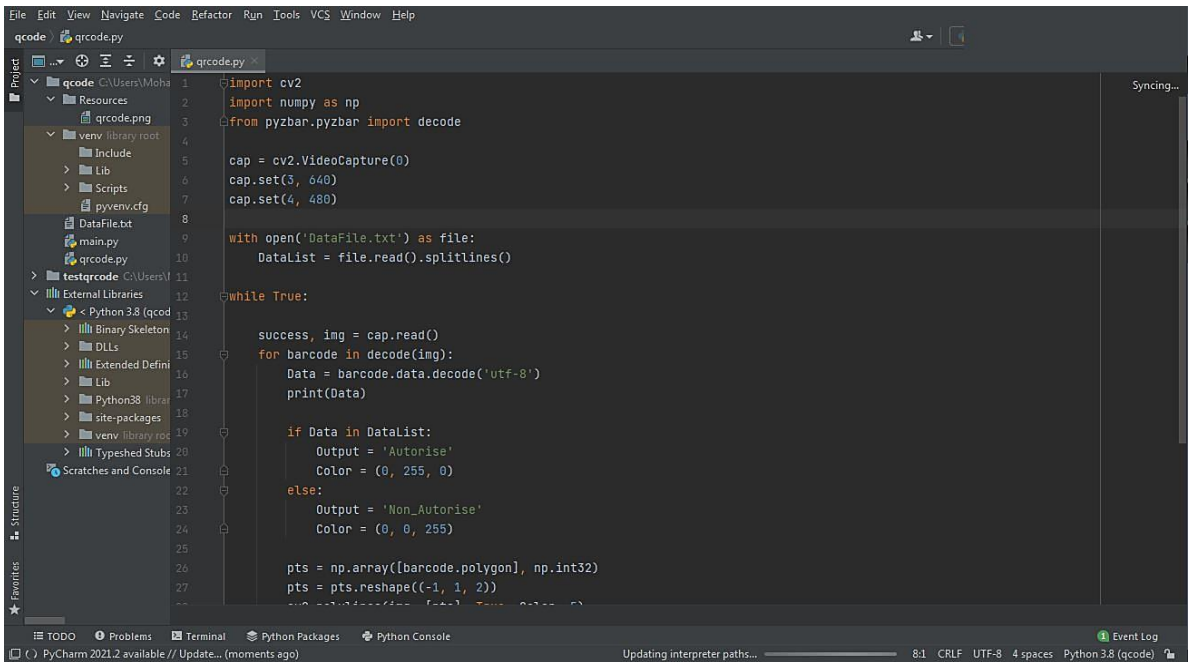


Figure 24 : Surface de PyCharm

Apres l'installation de PyCharm nous devons installer OpenCV 2.4.0, pour ce faire :

1. Cliquer sur l'onglet File → Settings ;

2. Appuyer sur Project Interpreter ;

3. Appuyer sur le symbole (+) ; une ne nouvelle fenêtre s'affiche à l'écran comme vous pouvez le voir sur la Figure 25.

4. taper" opencv - python " dans la barre de recherche., spécifié la version 4.2.0 ; OpenCV-Python est une bibliothèque conçue pour résoudre les problèmes de vision par ordinateur. (Figure 26)

5. Appuyer sur install package.

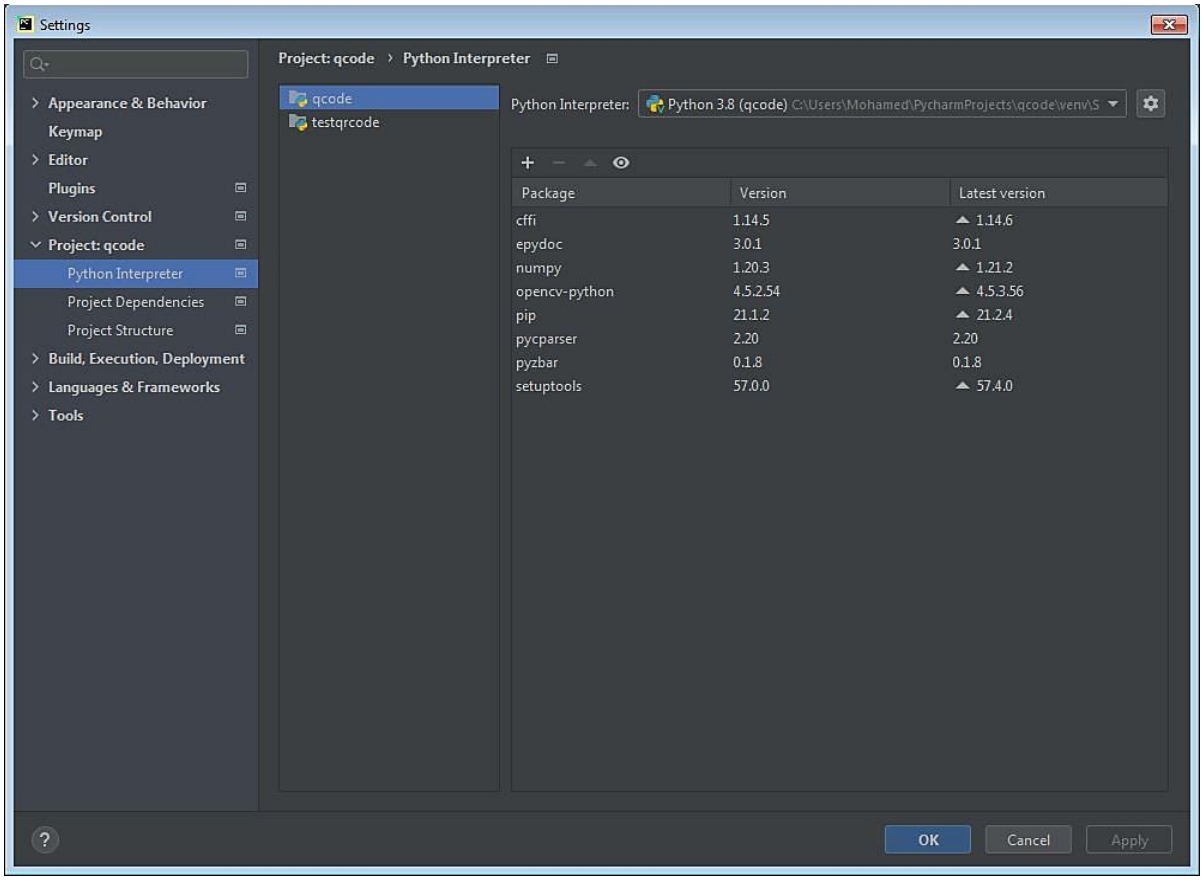


Figure 25 : Installation OpenCV (1)

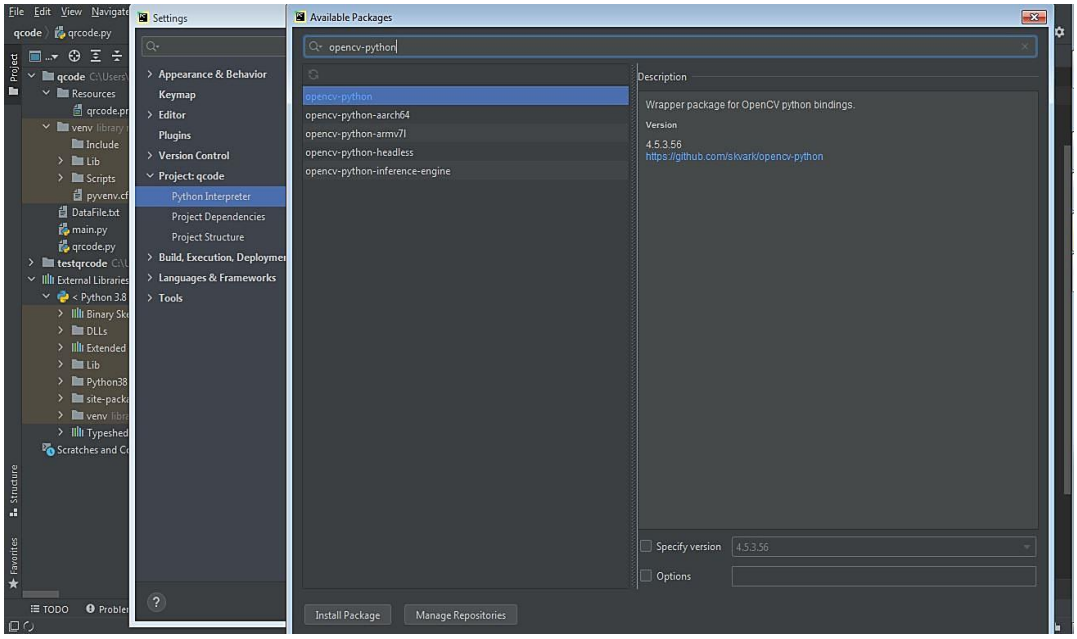


Figure 26 : Installation OpenCV (2)

V. Implémentation et résultats :

Nous commençons par importer le module cv2 et Numpy, afin que nous puissions lire notre image de test comme un tableau Numpy. Nous allons également importer la fonction decode du pyzbar module, que nous allons utiliser pour détecter et décoder le Qr Code.

```
Import cv2

import numpy as np

from pyzbar.pyzbar import decode
```

Après cela, nous allons lire l'image de test à partir du système de fichiers. Nous allons faire cela avec un appel à la fonction *imread* , en passant en entrée une chaîne avec le chemin vers le fichier. Vous pouvez utiliser cet outil en ligne gratuit pour générer un Qr Code avec le contenu que vous souhaitez. Dans notre cas, j'ai créé un Qr Code avec les données. Vous pouvez vérifier ci-dessous sur la Figure 27 le Qr Code utilisé dans nos tests.

Figure 27: L'image qui contient le Qr Code

```
img=cv2.imread('Resources/image1.png')
```

Ensuite, pour décoder le Qr Code dans l'image, il suffit d'appeler la fonction de décodage que nous avons importée, en passant en entrée l'image.

Cette fonction renvoie un tableau d'objets de la classe ***Decoded*** . Chaque élément du tableau représente un Qr Code détecté. Cela signifie que nous pouvons utiliser la bibliothèque avec des images avec plusieurs Qr Code.

```
code=decode(img)
```

Dans notre cas, notre image n'a qu'un seul Qr Code, il est donc prévu que ce tableau ne contienne qu'un seul élément. Néanmoins, pour un code plus robuste, nous allons parcourir le tableau avec une boucle for in, ce qui signifie que cela fonctionnera dans le cas où un, plusieurs Qr Code ou aucun Qr Code ne sont trouvés.

```
for barcode in decode(img):
```

Nous imprimerons également le contenu décodé à partir du Qr Code (attribut de données) Dans l'image de test, j'ai utilisé un Qr Code .

```
print(barcode.data)

print(myData)
```

Enfin, nous récupérons ensuite les données de chaque Qr Code. Ces données sont de types 'bytes', nous allons les convertir en 'string' en utilisant la fonction. $decode(« utf-8 »)$.

```
myData=barcode.data.decode('utf-8')
```

Voilà le programme de la première partie :

```
import cv2
import numpy as np
from pyzbar.pyzbar import decode
img=cv2.imread('Resources/image1.png')
code=decode(img)
for barcode in decode(img):
    print(barcode.data)
    myData=barcode.data.decode('utf-8')
    print(myData)
```

Pour créer un programme d'authentification basé sur une liste de Qr code on crée un fichier ***DataFile.txt*** et on écrit tous les codes autorisés dans ce fichier. Pour lire toutes les données de ***DataFile.txt*** et les stocker dans une liste on utilise la commande $file.read().splitlines()$

```
while True:
    success, img = cap.read()
    for barcode in decode(img):
        Data = barcode.data.decode('utf-8')
        print(Data)
        if Data in DataList:
            Output = 'Autorise'
```

```
            Color = (0, 255, 0)
        else:
            Output = 'Non_Autorise'
            Color = (0, 0, 255)
```

Si le Qr code capturé par la webcam existe dans le fichier DataFile.txt affiche le message "***Autorise***" en vert sinon le message "***Non_Autorise***" en rouge.

```
pts = np.array([barcode.polygon], np.int32)
pts = pts.reshape((-1, 1, 2))
cv2.polylines(img, [pts], True, Color, 5)
```

Pour Crée une boîte englobante (bounding box) autour de Qr Code à l'aide de polygone retourné par la fonction de décodage. Pour ce faire, nous devons :

- Convertir le polygone en un tableau $NumPy$ d'entiers $int32$.
- Remodeler le tableau à l'aide de la fonction $reshape()$.
- Utiliser la fonction $polylines()$ pour dessiner un polygone sur l'image.

```
pts2 = barcode.rect
cv2.putText(img, Output, (pts2[0], pts2[1]), cv2.FONT_HERSHEY_SIMPLEX, 0.9, Color, 2)
```

Ensuite, pour afficher le code sur l'image captée (précisément sur le Qr Code capturé par la webcam) on utilise la fonction : $putText()$. Et pour éviter la rotation du texte lors de la rotation de la caméra on utilise les points du rectangle $barcode.rect$.

```
cv2.imshow('Result', img)
cv2.waitKey(0)
```

Pour finaliser, nous afficherons le résultat, déjà avec le rectangle autour du Qr Code, dans une fenêtre à l'aide de la fonction $cv2.imshow()$.

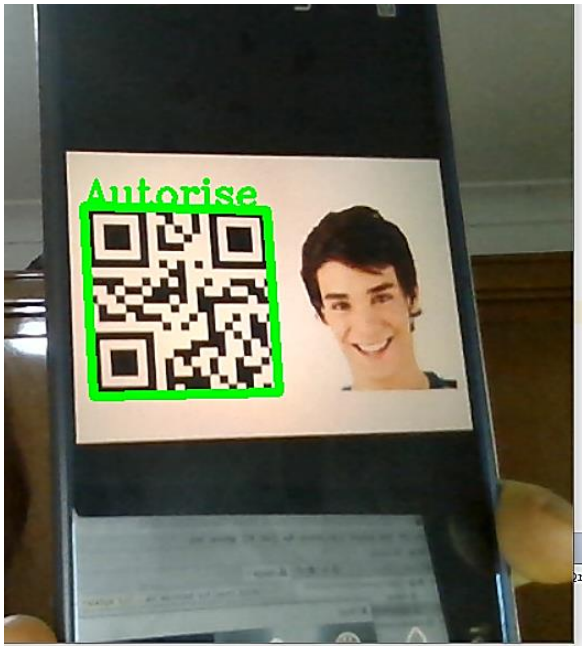


Figure 28 : Le résultat final (1)

Vous pouvez utiliser un outil en ligne gratuit pour générer un Qr Code avec le contenu que vous souhaitez. Dans notre cas, j'ai créé un Qr Code avec les données. Vous pouvez créer un code Qr en utilisant ce site : https://qrcode.tec-it.com/fr.

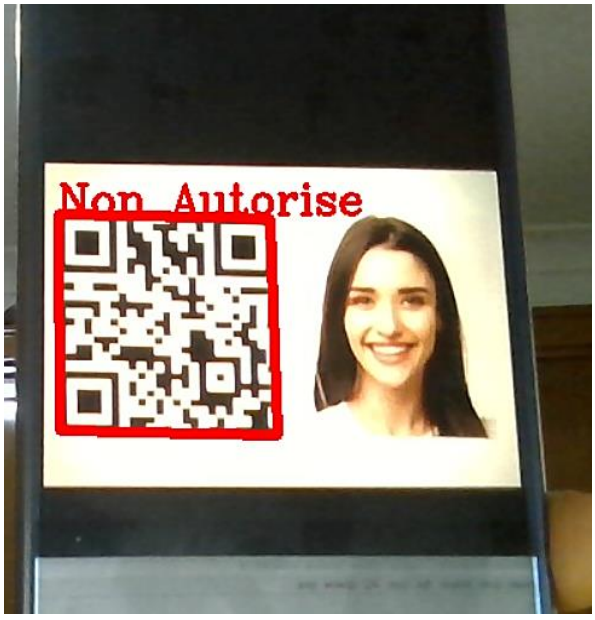


Figure 29 : Le résultat final (2)

V.1 Le programme final :

```
import cv2
import numpy as np
from pyzbar.pyzbar import decode

cap = cv2.VideoCapture(0)

cap.set(3,640)

cap.set(4,480)
with open('myDataFile.text') as f:
    myDataList = f.read().splitlines()
while True:
    success, img = cap.read()
    for barcode in decode(img):
        myData = barcode.data.decode('utf-8')
        print(myData)
        if myData in myDataList:
            myOutput = 'Non_Autorise'
            myColor = (0, 0, 255)
        else:
            myOutput = 'Autorise'
            myColor = (0,255,0)
        pts = np.array([barcode.polygon],np.int32)
        pts = pts.reshape((-1,1,2))
        cv2.polylines(img,[pts],True,myColor,5)
        pts2 = barcode.rect
        cv2.putText(img,myOutput,(pts2[0],pts2[1]),cv2.FONT_HERSHEY_SIMPLEX
                    0.9,myColor,2)
    cv2.imshow('Result',img)
    cv2.waitKey(0)
```

VI. CONCLUSTION :

Dans ce projet, nous avons vu comment utiliser la bibliothèque pyzbar avec OpenCV afin de lire un Qr Code. Le code QR montrait de nombreux avantages qu'il offre et la taille des données qu'il peut stocker, et le code QR se compose d'unités noires disposées sous une forme spécifique sur un fond carré blanc qui mène le scanner pour montrer les données qu'il symbolise. Le processus de détection du code QR est simple et peu coûteux. Nous avons également pris conscience de l'importance de cette technologie et de sa pénétration dans notre vie quotidienne.

CONCLUSION GÉNÉRALE :

Tout au long de la préparation de notre projet de fin d'études, nous avons essayé de mettre en pratique les connaissances acquises durant nos études universitaires et cela dans le but de réaliser une application détection et lecteur de QrCode avec OpenCV et Pyzbar.

Nous avons d'abord eu une initiation théorique à la discipline de la vision par ordinateur, puis nous nous sommes familiarisés avec deux bibliothèques utilisées en Python pour le traitement d'images et la gestion de tableaux, à savoir OpenCV et NumPy.

La bibliothèque OpenCV est un élément important dans la réalisation et le développement de notre projet et cela revient à la richesse de différentes fonctions qui le compose surtout dans le traitement des images et des vidéos.

Le code QR peut être utilisé de différentes manières selon les besoins. Nous avons construit un système d'authentification qui autorise l'accès des personnes déjà inscrites dans un fichier.

Le module QrCode permet la détection et le décodage de QR code grâce à un flux vidéo venant d'une caméra. Pour l'authentification par QrCode, nous avons choisi d'utiliser une variante en Python du programme Pyzbar. Pyzbar nous a permis, contrairement à d'autres lecteurs de QrCode que nous avons essayé, de bénéficier des avantages suivants :

- Multiplateforme (utilisé sous Windows mais disponible pour linux,)
- Performant (scan des QrCodes instantanément, même ceux qui sont peu visibles)
- Flexible (scan aussi bien des codes-barres que des QrCode).

REFERENCES:

Brahmbhatt, S. (2013). Practical OpenCV. Apress.

Radu, H et Olivier, L. (1995). Vision par ordinateur outils fondamentaux.

Marc, M. (2013). Les QR Codes en bibliothèque : un exemple de médiation numérique au service des usagers.

Fouzia, H. (2015-2016). La détection et suivi des objets en. Université Abou Bakr Belkaid– Tlemcen: Mémoire de fin d'études.

WEBOGRAPHIE :

[1] : https://azure.microsoft.com/fr-fr/overview/what-is-computer-vision/.

|2] : http://dspace.univ-tlemcen.dz/bitstream/112/6831/1/Segmentation-des-Images%20.pdf.

[4] : https://pentaxklub.com/le-poids-dune-image/.

[5] : https://www.toupie.org/Dictionnaire/Transparence.htm.

[6] : https://medium.com/@veilleunitec/

[7] : https://www.numericube.com/la-vision-par-ordinateur.

[8] : https://opencv.org/releases-old/opencv/.