*République Algérienne Démocratique et Populaire*

*Ministère de l'enseignement Supérieur et de la Recherche Scientifique Université de Jijel*

*Faculté des Sciences Exactes et Informatique*
*Département d'Informatique*

**Mémoire de fin d'études pour l'obtention du diplôme de Master en Informatique**

**Option : Intelligence Artificielle**

**Thème**

# Construction d'images Panoramiques sur Smartphone

**Réalisé par :**

Hamadéne Atika

**Encadré par :**

Dr. Mokhtar Taffar

Promotion 2017/2018

# *Remerciements*

Louange à Allah, que le salut et la bénédiction soient sur son Prophète.

Je remercie avant tout Allah le tout puissant qui m'a donné le courage et la force pour continuer ce travail.

Je présente mes remerciements et gratitudes à ma famille, qui m'a largement soutenu durant tout mon cursus.

Mes sincères remerciements vont à mon encadreur , Mr Mokhtar Taffar qui m'a accompagné durant cette année de mémoire, avec ses conseils, son aide et ses encouragements précieux.

Je remercie très vivement les membres de jury pour me faire l'honneur de bien vouloir juger mon travail.

Je tiens à remercier tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

# Dédicace

***A mon très cher Papa***

***A ma  très chère Maman***

***A ma famille***

***A mes amies***

***Et tous ceux qui ont un impact sur ma vie***

***Je dédie ce modeste travail***

*Atika*

# Table des matières

# Liste des figures

# Liste des tableaux



# Liste des abréviations

**SURF**.....Speeded-Up Robust Features ee

**SIFT**. ..... Scale Invarient Feature Transformationation

**ORB**. ..... Orienté FAST et Rotated BRIEF

**SDK.**.....Softwart Developement Kit scn

**NDK**. ..... Native Developement

# Introduction générale

## 1. Généralité

Durant ces deux dernières décennies, le domaine de la vision par ordinateur et de l'analyse d'image/vidéo a connu une avancée fulgurante, tant des algorithmes et approches remarquables, voir révolutionnaires, utilisés que des appareils smart et miniatures exploités. Une ère informatique dont le développement de logiciels/applications nécessite de plus en plus de créativité et de techniques innovantes destinées à différents types d'appareils (PC, Tablette, Mini-Tablette, Smartphone) fonctionnant sous divers environnements et couvrant un très large spectre de domaines si ce n'est pas tous les domaines dans les prochaines années.

Parmi ces techniques, on retrouve la construction panoramique qui permet de représenter la scène dans sa géométrie globale naturelle, avec un angle de vue horizontal (ou plan horizontal), dit angle de panorama (Pan, en anglais), qui peut varier de 270° à 360°, et un angle de vue vertical (plan vertical), appelé angle d'inclinaison (Tilt, en anglais), qui peut s'étendre jusqu'à 120 degrés, selon le type de la caméra utilisée.

L'utilisation des images panoramique est largement répandue dans des domaines telles que la robotique, la vision par ordinateur, la surveillance et la réalité virtuelle. Elles sont également utilisées dans des applications commerciales comme le divertissement, la TV interactive, l'immobilier et le tourisme virtuel. Ces dernières années, les systèmes panoramiques de formation d'image ont sensiblement progressé. Non seulement les professionnels peuvent créer et afficher des panoramas, mais il existe une grande quantité de logiciels disponibles, dont certains gratuits. Chacun d'entre nous, avec un ordinateur et un simple appareil photo numérique, peut créer des panoramas. Même les grandes compagnies sont présentes sur ce marché comme Apple avec son logiciel QuickTime2 ou Canon avec le logiciel PhotoStitch3 qui permettent un accès facile à la création d'image panoramique.

Une image panoramique est une image de grand angle visualisant l'environnement autour de l'appareil photo. Habituellement, l'image panoramique entoure complètement l'appareil photo ou la caméra sur le plan horizontal et approximativement 120° sur le plan

vertical ou 180° pour créer une sphère complète. Un exemple de représentation panoramique est donné ci-après.

**Figure 1** . ***Exemple de représentation panoramique (360° suivant le plan horizontal et 120° sur le plan vertical)***

Une image panoramique ou mosaïque est l'alignement de plusieurs images en une seule image pour représenter une même scène. On retrouve la mosaïque d'image dans d'autres domaines tels que l'imagerie médicale (IRM, Scanner, appareil dentaire), les applications d'infographie, la réalité augmentée, la télédétection (caméras de télésurveillances, images satellitaires, …) et le divertissement par des applications de loisirs (pour faire des photographies de vacances).

Bien qu'elle paraisse facile, la technique de construction de mosaïque d'image est assez complexe et fait intervenir plusieurs méthodes (de nature géométrique et photométrique) pour réussir une image mosaïque parfaite. La construction et l'exploitation d'une image panoramique passent par 4 grandes étapes : l'acquisition, la projection des prises de vue dans

l'image panoramique, l'amélioration (correction) du rendu et, enfin, l'immersion et visualisation.

L'étape de projection des prises de vue ou des images dans un panorama comporte à elle seule plusieurs phases : calcul du modèle de transformation mathématique, réaliser les transformations géométriques, calcul de l'homographie, la mise en correspondance (assemblage), le recalage et les alignements.

## 2. Travaux connexes

La construction des images panoramique est un problème difficile qui continue à mobiliser la communauté scientifique [1] [2][3]. Plusieurs solutions sont proposées dans la littérature et de nombreux articles proposent une synthèse de ces méthodes. Le principe de base consiste à mettre en correspondance des zones de l'image en fonction de leurs propriétés radiométriques ou géométriques. Les méthodes sont habituellement classées en deux catégories : denses ou éparses.

Les méthodes denses cherchent à estimer les paramètres de transformation en exploitant l'intensité de l'ensemble des pixels contenu dans la zone de recouvrement. Les méthodes éparses n'utilisent pas tous les points mais seulement quelques points particuliers (coins, extrema locaux) ou des primitives géométriques (lignes, cercles…). Dans le cas où il n'y a pas de déformation des images, les méthodes denses sont réputées plus précises, mais elles sont souvent moins rapides, moins robustes aux changements de luminosité et à la présence d'objet en mouvements.

Il existe plusieurs méthodes pour capturer des panoramas. La solution la plus simple consiste à utiliser un appareil photo classique monté sur un trépied et en faisant tourner manuellement l'appareil photo autour de son centre optique. Les professionnels auront plutôt tendance à utiliser une caméra motorisée de type PTZ (Pan, Tilt, Zoom). Le principe est exactement le même, à la différence que les rotations de la caméra autour de son centre optique sont pilotées et contrôlées par un ordinateur. Ceci permet de déterminer avec plus de précision les conditions de la prise de vue. L'un des inconvénients de cette approche, qu'elle soit manuelle ou pilotée, est qu'il faut du temps pour parcourir l'ensemble de la scène. La représentation panoramique ne peut pas être réalisée en temps réel. A un instant donné, seule une fraction de la scène est vue par la caméra. Des solutions existent pour capturer l'ensemble de la scène en temps réel. La plus simple consiste à utiliser des équipements photographiques appelés omnidirectionnels. Dans le cadre de notre projet, pour faute de moyens et d'équipements spécialisés, nous nous limiterons à l'utilisation de simples images prises à l'aide d'une caméra embarquée dans un Smartphone fonctionnant sous Andoid.

Pour réaliser les différentes étapes de la construction de panorama de nombreuses approches ont vu le jour.

## 3. Problématique

Comme discuter dans les sections précédentes, les algorithmes de mosaïcage se basent essentiellement sur les étapes de : enregistrement, couture, assemblage et réglage des recouvrements. L'usage de caméras professionnelles pour les systèmes de vidéosurveillance offres des images ou vidéos bien calibrées, avec des paramètres de caméra bien contrôlés. Cependant, s'agissant d'images ou de vidéos provenant d'un appareil photo de Smartphone, l'algorithme de construction de panorama devra répondre à plusieurs contraintes géométriques de stabilité, de vitesse de prise de vue et même de variations photométriques ou de luminosité.

L'usage d'une seule caméra comme source d'images ou du flux vidéo pour construire un panorama peut réduire la difficulté algorithmique liée à la phase d'assemblage et limite les déformations dans l'image mosaïque résultante. Toutefois, la construction vidéo panoramique est encore plus complexe que celles des images panoramiques et ça l'est encore plus lorsqu'il y a un ou des objets se déplaçant dans une scène formée de plusieurs images. C'est pour cela que les nouveaux systèmes de vidéosurveillance fonctionnant dans un environnement multi-caméra ou non sont plus intelligents : doter d'un fond panoramique construit à partir d'images provenant d'une caméra mobile ou de plusieurs caméras, ils permettent de détecter et suivre un objet à travers ces caméras ; le panorama se construit au fur et à mesure que l'objet détecté se déplace ou la/les caméra(s) bougent.

## 4. Motivation

Dans le cadre de notre projet, nous allons aborder les différentes étapes qui entrent dans le processus de réalisation et d'exploitation d'une image panoramique à partir d'images déjà acquises ou en temps réel : l'acquisition, la projection des prises de vue dans l'image panoramique, l'amélioration du rendu, la visualisation ; ainsi que l'application de détection et de suivi des objets en mouvement dans un panorama.

Dans ce travail de master, une fois qu'on dressera un état de l'art sur la construction panoramique. On présentera une méthode de construction de panorama sur un système mobile. On utilisera l'algorithme d'extraction des caractéristiques SURF Et ORB. Le

détecteur SURF est simple à mettre en œuvre et offre une rapidité supplémentaire par rapport aux autres détecteurs, ce qui permet de l'utiliser en temps réel dans une vidéo.

## 5. Plan du mémoire

Ce mémoire est structuré comme suit :

- ✓ Le premier chapitre est consacré à la présentation des images panoramiques et leurs domaines d'utilisations.
- ✓ Dans le deuxième chapitre, on présentera les méthodes de constructions des images panoramiques et l'utilité de ces images.
- ✓ Dans le troisième chapitre, on s'orientera les environnements de développement mobiles en particulier l'Android.
- ✓ Le quatrième chapitre fera l'objet de notre conception du système.
- ✓ Le cinquième chapitre fera l'objet de notre partie pratique, l'application dédié à la résolution de la problématique et le résultat obtenu après exécution.
- ✓ Enfin on clôture le manuscrit par une conclusion générale sur le sujet.

# Chapitre 1 : Les Panoramas

## 1.1Introduction

De nos jours les Smartphones offrent divers services aux utilisateurs. L'appareil photo n'est plus un luxe ,chaque smartphone est impérativement doté d'une caméra proposant des filtres et des opérations tel que les images panoramiques. Dans ce chapitre nous allons présenter la terminologie générale d'une image panoramique, ensuite découvrir les différents domaines d'applications de panoramas[14], et enfin on va citer brièvement les approches pour construire une image panoramique.

## 1.2 Panoramas

### 1.2.1Définition

Une image ''Panoramique'' ou ''Mosaïque'' est une image de grand angle qui montre un champ de vision horizontal au moins aussi large que celui de la combinaison des deux yeux humains. Pour obtenir un panorama, plusieurs images sont nécessaires ; ce panorama résulte de la combinaison (au sens superposition) de ces images. Chaque image représente une partie de la même scène avec un degré d'inclinaison vertical ne dépassant pas un certain angle par rapport aux autres images, et dont le résultat est une vue en largueur horizontale de la scène.

Il existe des caméras spécialisées conçues pour capturer une large scène, mais n'importe quelle caméra peut être utilisée pour produire des panoramas. L'imagerie panoramique a donné naissance à la vidéo-panoramique. Basé sur le même concept, elle concerne les vidéo .

## 1.2.2 Image Panoramique Traditionnelle

A l'origine, l'imagerie panoramique se faisait avec des caméras classiques sans l'aide des ordinateurs[14]. Traditionnellement, quelques méthodes furent utilisées dont on en cites quelques-unes :

- **Méthode Artificielle** : elle consistait à masquer la partie inférieure ou supérieure de l'image pour donner la fausse impression d'un large champ de vision.
- **Panorama Segmenté** : ce type de panorama consiste en un groupe d'images qui se chevauchent d'environ un tiers avec l'image suivante et précédente.
- **Panorama Rotationnel :** ce panorama est réalisé grâce à des caméras qui pivotent autour d'un point.

## 1.2.3 Exemple de panoramas

### 1.2.3.1 Panorama à deux images

Un panorama peut être construit à partir de deux images au moins, on retrouve une même partie de l'image présente dans deux images.

**Figure 1.1 : Deux images différentes représentant une même scène**

Le résultat est l'image mosaïque, assemblée à partir de ces deux images, qu'on peut voir sur la figure suivante **Figure1.2**

**Figure 1.2** ***:Image panoramique à partir de deux images***

### 1.2.3.2 Panorama à plusieurs images

On peut faire un panorama à partir de plusieurs images . Dans l'exemple suivant , on a plusieurs images pour représenter le panorama final comme le montre la **figure1.3**

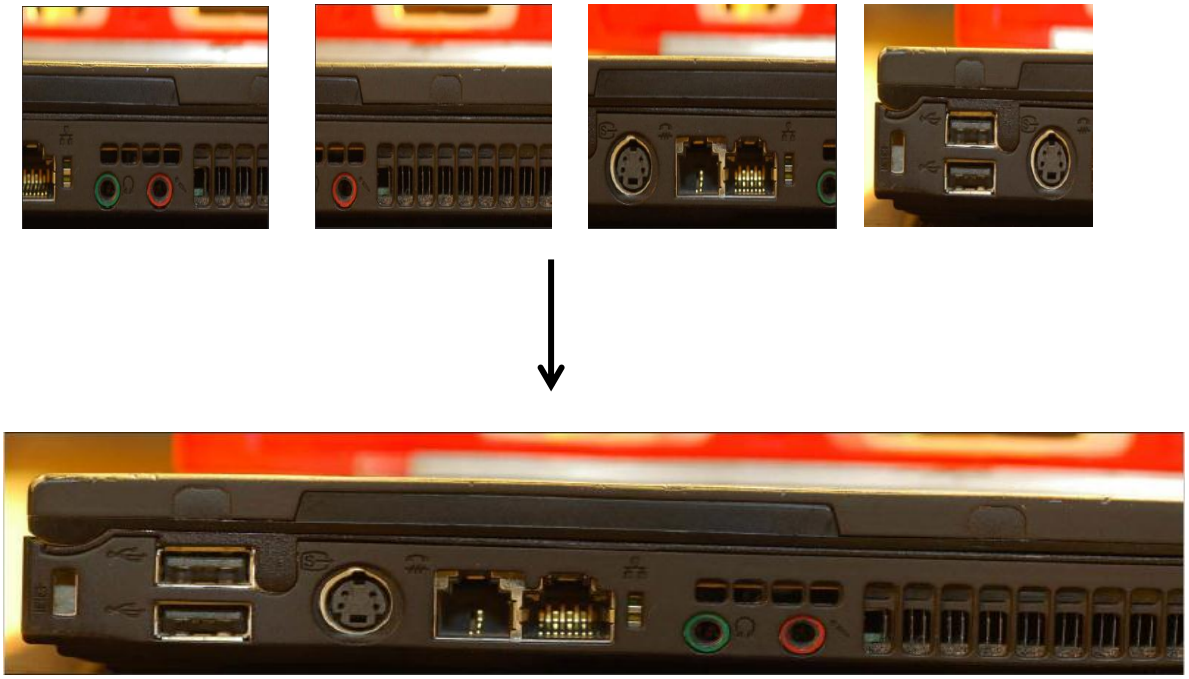


**Figure 1.3** ***: Image panoramique à partir de plusieurs images***

#### 1.2.3.3 Panorama à partir d'une séquence vidéo

Une autre approche de l'imagerie panoramique peut être l'utilisation d'une séquence vidéo comme source de capture d'image. Les applications vidéo ont généralement été conçues pour la création, l'accès et la relecture de contenu. De nos jours, il est également possible d'utiliser la vidéo pour permettre de nouvelles applications. Dans ce cas, les informations de mouvement peuvent être extraites de la séquence d'images capturée par la caméra pour créer un générateur de panorama en temps réel.[14]

Les séquences vidéo ont généralement une résolution inférieure à celle des images fixes. Cependant, ils ont une grande zone de chevauchement entre les images consécutives. La forte redondance des images nous permet de construire le panorama et de compenser une résolution inférieure.

Prendre une séquence vidéo pour créer un panorama est un processus très intuitif. Le point de départ de la vidéo est le début de la mosaïque et peut ensuite être déplacé dans les directions souhaitées avec des transitions douces. Pour obtenir une image panoramique à partir d'une vidéo, une séquence est généralement enregistrée afin de la traiter ultérieurement. Chaque image de la vidéo est séparée et traitée comme une image unique qui se chevauche par exemple autour de 90% de la précédente.

En plus des avantages d'interactivité, l'utilisation de l'imagerie vidéo basse résolution peut être défendue sur des aspects purement techniques. En mode basse résolution, la sensibilité de la caméra peut être meilleure car la taille effective des pixels est plus grande, réduisant ainsi les exigences d'éclairage et améliorant la tolérance au flou de mouvement.

## 1.3 Domaines d'Application des Images Panoramiques

Contrairement à ce qu'on peut penser, les panoramas sont présents dans différents domaines d'application comme dans :

### 1.3.1 Photo graphie panoramique

Devenu un moyen de divertissement grâces aux réseaux sociaux, prendre une photo est la chose la plus répondue de nos jours , le panorama permet dans ce cas d'offrir au téléspectateur une vue unique de plusieurs images tout en conservant l'information de ces images .

### 1.3.2 La Radiologie panoramique

Dans le domaine médicale on retrouve la radiographie panoramique souvent associée à le panoramique dentaire avec lequel on a une mise à plat de l'arc dentaire, d'une oreille à l'autre .

### 1.3.3 La Video-surveillance

De nos jours les systèmes de vidéos surveillance emploient le stitching panoramique pour de multiples caméras .

### 1.3.4 En Télédétection ou Imagerie Satellitaire

Plusieurs vues satellitaires ou aériennes doivent être fusionnées pour représenter une carte couvrante d'une région.

## 1.4 Les Modèles de Panoramas

Un panorama eut être représenté sous différentes formes géométriques et ce dépendant de son utilisations[14] . On distingue :

### 1.4.1 Planaire

Un panorama planaire est traditionnellement une image à grand angle réalisé par l'assemblage de nombreuses images.On reconnait dedans le point de début et le point de fin . Ces panoramas sont les plus répondus dans la photographie et le meilleur choix quand on veut publier. **La figure 1.4** montre un panorama planaire

**Figure 1.4 :** ***Image panoramique sous le modèle planaire***

### 1.4.2 Circulaire

On obtient un panorama circulaire quand on effectue une conversion de coordonnées d'un panorama planaire vers des coordonnées polaires. Pour un résultat à 360 degré, tout le champ de vison est convertie. Ces panoramas présentent une déformations dans les deux axes . La **figure 1.5** montre le panorama circulaire

Fi**gure 1.5:** ***Image panoramique sous le modèle circulaire***

### 1.4.3 Cylindrique :

L'image est prévue pour être visionnée comme si elle était courbée autour d'un point tel la face interne d'un cylindre alors un panorama cylindrique est nécessaire. Ce type d'image peut être aperçu horizontalement avec une boucle allons de droite à gauche . Pour les réaliser on a besoin de logiciels de retouches. La **figure 1.6** montre un panorama cylindrique

**Figure 1.6 :** ***Panorama cylindrique***

### 1.4.4 Sphérique

Comme type de panorama est conçu pour être perçu à partir du centre hypothétique d'une sphère . avec ces panoramas la partie la plus basse et la plus haute de l'image peuvent etre visionnée Pour se faire des logiciels sont nécessaires .

**Figure1.7 :Panorama Sphérique**

### 1.4.5 Cubique

Chaque face du cube est représentée comme un panorama planaire , pour se faire six différentes images sont nécessaires pour chaque face du cube .Dans La **figure 1.8** on retrouve un exemple de panorama cubique .

:

**Figure** 1.**8 : Panorama Cubique**

## 1.5 Types d'algorithmes pour la construction panoramiques

Pour La construction des panoramas se basent essentiellement sur les algorithmes de **Stitching** qui sont séparé en deux catégories : les algorithmes de construction par lots (Batch Type Algortihm) et les algorithmes de construction à temps réels( Reel-Time Algortihm). [14]

### 1.5.1 Batch Type Algortihm

Ces algorithmes sont assez populaires et ont été utilisés dans les ordinateurs personnels depuis déjà quelques années grâce à des logiciels conçu spécialement pour cette tache. Le point de départ est une séries d'images déjà prises auparavant qui sont supposée être assemblée en une image panoramique . On verra par la suite les étapes pour effectuer cet assemblage.

### 1.5.2 Real-Time Algorithm

Contrairement aux précédents ces algorithmes sont relativement plus récents**.** Ils demandent beaucoup de ressources car le point de départ est un cadre de l'image qui s'agrandit au fur et à mesure que la construction du panorama se fait. On peut voir cette approche dans l'option panoramique des Smartphones.

## 1.6 Approches d'Assemblage

Pour Typiquement il existe deux approches d'assemblage d'images pour les deux type de construction panoramique vu en dessous , l'approche directe (**direct approach)** et l'approche basée sur les caractéristiques (**feature based approach**).

### 1.6.1 Approche Directe

Basée sur les pixels ,l'approche directe prend en compte toutes les informations de l'image. Tous les pixels des images à assembler sont utilisés et sont comparés entre eux, ce qui en fait une technique très complexe. Elle fonctionne très lentement et la rend indésirable pour les applications en temps réel où le temps est un facteur énorme. Elle est grandement affectée par les différences d'exposition du même objet dans différentes images à assembler. Un autre inconvénient majeur des techniques d'assemblage direct est qu'elles nécessitent beaucoup d'initialisation, ce qui signifie qu'il doit y avoir beaucoup d'interactions humaines pour définir les paires d'images correspondantes et pour s'assurer que l'assemblage a lieu correctement. Par conséquent, cette méthode est généralement évitée dans les logiciels commerciaux pour l'assemblage d'images.

### 1.6.2 Approche basée Sur Les Caractéristiques

Cette approche repose sur le détecteur de caractéristique. Tous les principaux points caractéristiques d'une paire d'images sont comparés à ceux de toutes les entités d'une autre image en utilisant des descripteurs locaux ,qui lorsqu'il reçoit une image, il l'analyse pour extraire ses métadonnées. Des informations telles que la relation entre les points de l'image et les relations géométriques des pixels de l'image forment les caractéristiques extraites . Parmi les plus populaires, on peut citer la transformation des caractéristiques invariantes en échelle (SIFT), les fonctions robustes accélérées (SURF). Les techniques basées sur cette approche fonctionnent beaucoup mieux, mais le temps de couture peut varier considérablement en fonction de la méthode exacte utilisée. De plus elle ne nécessite aucune interaction humaine car elles reposent sur la correspondance automatique des points spécifiques et supposent que le reste est cohérent autour de ces points. Ces techniques sont souvent les mieux adaptées, c'est pour cela qu'on va s'orienter vers elles dans la suite de ce manuscrit.

## 1.7 Conclusion

Dans ce chapitre, on a présenté les images panoramiques , les differents modéles de reprèsentations de ces images ainsi que les domaines d'applications . Nous avons aussi parcouru les apprcohes d'assemblage d'images . Dans le chapitre suivant ,nous allons voir de plus près ces étapes principalement avec l'approche basée sur les caractéristiques .

# Chapitre 2 : La Construction Panoramique

## 2.1 Introduction

L'approche basée sur les caractéristiques est considérée comme la mieux adaptées afin de réussir à assembler des images et réaliser un panorama. Dans ce chapitre, nous allons présenter les différentes étapes qui existent dans cette approche pour la construction panoramique .

## 2.2 La construction panoramique

Essentiellement pour construire un panorama **Le Recalage (Registration)** et **la Fusion d'images (Blending)** sont les étapes les plus importantes qui influencent directement la construction de l'image mosaïque . Mais dans la littérature on peut trouver d'autres étapes souvent incluses dans ces deux dernières comme acquisition des images, extractions des caractéristiques, le calcule d'homographie, l'alignement, et l'enveloppement de l'image[4][5][6] .

### 2.2.1 Recalage

En traitement d'image, le recalage est une technique qui consiste en la « mise en correspondance d'images ». Il se réfère donc à l'établissement d'une correspondance géométrique entre une paire d'images représentant la même scène.

Afin d'enregistrer un ensemble d'images ,les comparer ou les combiner, il est nécessaire d'estimer les transformations géométriques qui alignent les images par rapport à une image de référence dans cet ensemble. L'ensemble peut être constitué de deux images ou plus prises sur

une seule scène à des moments différents.. Le recalage est la 1ere étape de la construction d'image mosaïque à partir d'un nombre d'images acquis .

### 2.2.3 Fusion

La fusion ou assemblage des images implique l'exécution des ajustements définis lors de la phase d'étalonnage.Les couleurs sont ajustées entre les images pour compenser les différences d'exposition.

**L'étape d'assemblage** a pour but de superposer les images alignées sur un canevas plus grand en fusionnant les valeurs de pixel des parties superposées et en conservant les pixels où aucun chevauchement ne se produit. Les erreurs propagées par des défauts d'alignement géométriques et photométriques entraînent souvent des discontinuités d'objets indésirables et une visibilité des joints au voisinage de la limite entre deux images. Ainsi, un algorithme de mélange doit être utilisé pendant ou après l'étape de couture afin de minimiser les discontinuités dans l'apparence globale de la mosaïque.

### 2.3.3 Calcule d'homographie

Dans le cas où les longueurs et les angles liées aux parties communes ne sont pas conservées, on doit appliquer une transformation projective c'est L'homographie [6].

Une telle transformation ne conserve ni les longueurs, ni les angles et peut s'écrire sous forme matricielle:

$$\begin{pmatrix} x' \\ y' \\ w' \end{pmatrix} = \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\text{soit} \quad \mathbf{x}' = H\mathbf{x}$$

Sont les coordonnées homogènes des pixels de l'image de d´épart et de l'image déformée. Pour déformer une image, il suffit donc d'appliquer une homographie `a tous ses pixels.

$$\mathbf{x} = (x, y, 1)^\top \text{ et } \mathbf{x}' = (x', y', w')^\top = \left(\frac{x'}{w'}, \frac{y'}{w'}, 1\right)^\top$$

Une homographie est une matrice 3 × 3 et possède donc 9 éléments indépendants. Cette matrice est utilisée en coordonnées homogènes, elle est donc invariante par changement d'échelle et on peut alors la multiplier par une constante réelle non nulle sans changer ses effets sur une image. Pour trouver la matrice d'homographie il faut résoudre le système suivant :

$$\begin{pmatrix} 0 & 0 & 0 & -w'_1x_1 & -w'_1y_1 & -w'_1w_1 & y'_1x_1 & y'_1y_1 \\ w'_1x_1 & w'_1y_1 & w'_1w_1 & 0 & 0 & 0 & -x'_1x_1 & -x'_1y_1 \\ 0 & 0 & 0 & -w'_2x_2 & -w'_2y_2 & -w'_2w_2 & y'_2x_2 & y'_2y_2 \\ w'_2x_2 & w'_2y_2 & w'_2w_2 & 0 & 0 & 0 & -x'_2x_2 & -x'_2y_2 \\ 0 & 0 & 0 & -w'_3x_3 & -w'_3y_3 & -w'_3w_3 & y'_3x_3 & y'_3y_3 \\ w'_3x_3 & w'_3y_3 & w'_3w_3 & 0 & 0 & 0 & -x'_3x_3 & -x'_3y_3 \\ 0 & 0 & 0 & -w'_4x_4 & -w'_4y_4 & -w'_4w_4 & y'_4x_4 & y'_4y_4 \\ w'_4x_4 & w'_4y_4 & w'_4w_4 & 0 & 0 & 0 & -x'_4x_4 & -x'_4y_4 \end{pmatrix} \begin{pmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \\ h_5 \\ h_6 \\ h_7 \\ h_8 \end{pmatrix} = \begin{pmatrix} -y'_1w_1 \\ x'_1w_1 \\ -y'_2w_2 \\ x'_2w_2 \\ -y'_3w_3 \\ x'_3w_3 \\ -y'_4w_4 \\ x'_4w_4 \end{pmatrix}$$

C'est ce système qu'il faut résoudre et une fois H trouve, on n'a plus qu'à faire et l'appliquer sur l'image à déformer et la projeter dans le plan de l'image fixe.

$$H = \begin{pmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & 1 \end{pmatrix}$$

### 2.3.4 Extraction Des points d'Intérêts

Les points d'intérêts sont les points caractéristiques dans une image, qui se distinguent par rapport aux autres informations de l'image. Elles correspondent à des doubles dis- continuités de la fonction d'intensités. L'ensemble de points d'intérêts forme souvent la zone d'intérêt dans l'images.

**Figure 2.1 : Exemple de points d'intérêts**

Lorsque On souhaite créer une image panoramique à partir de deux images, la première étape consiste à extraire des points caractéristiques dans chaque image img1 et Img2. Par la suite, ces points seront associés deux à deux et permettront le raccordement des deux images. Pour pouvoir être associés, il faut que les points détectés dans la zone de recouvrement entre les images soient les mêmes.

On ne détecte pas les mêmes points => aucune chance de les associer

On détecte les mêmes points => on va pouvoir les associer

Pour extraire ces points , de nombreux détecteurs ont vu le jour parmi les quel on va citer quelques un dans la suite de chapitre .

### 2.3.5 Alignement

L'alignement peut être nécessaire pour transformer une image en fonction du point de vue de l'image avec laquelle elle est compostée. L'alignement en termes simples est un changement du système de coordonnées, de sorte qu'il adopte un nouveau système de coordonnées qui génère une image correspondant au point de vue requis. Les types de transformations qu'une image peut traverser sont la traduction pure, la rotation pure, une transformation de similarité qui comprend la traduction, la rotation et la mise à l'échelle de l'image à transformer, la transformation Affine ou projective.

## 2.4 Détecteurs de point d'intérêts

### 2.3.1 SIFT

L'algorithme SIFT est un algorithme de détection de caractéristiques de bas niveau qui détecte les caractéristiques distinctives (également appelées «points-clés») des images. Le descripteur SIFT est invariant par rapport aux transformations, rotations et transformations d'échelle dans le domaine de l'image et aux transformations de perspective et aux variations d'illumination robustes à modérées

Le descripteur SIFT est le plus couramment utilisé pour l'extraction de points clés. Ce qui a fait le succès de ce descripteur, c'est qu'il est peu sensible au changement d'intensité, de mise à l'échelle et de rotation, ce qui fait de lui un descripteur très robuste. Il est basé sur les différences de gaussiennes.[7]

### 2.3.2 SURF

Un peu plus récemment (2006 et 2008), une variante appelé **SURF** est apparue. Le principal avantage de cette technique est que l'extraction des points d'intérêts se fait plus rapidement, notamment grâce à l'utilisation d'images intégrales. Les points d'intérêt ne sont pas extraits exactement de la même manière que **SIFT** et ne sont pas non plus caractérisé de la même manière.

### 2.3.3 ORB (Orienté FAST et Rotated BRIEF)

ORB [8], construit sur le détecteur de points FASTkey et le descripteur BRIEF [9], est utilisé pour l'extraction en temps réel des principales caractéristiques ponctuelles. FAST et ses variantes sont très efficaces pour trouver des points clés raisonnables. En outre, cette technique trouve une application dans de nombreux systèmes en temps réel tels que la réalité augmentée mobile, le suivi en parallèle et la cartographie. [10], [11]

FAST utilise un seul paramètre, à savoir le seuil d'intensité entre le pixel central et ses pixels adjacents sur un anneau circulaire. FAST a une contrainte de non-inclusion d'un opérateur d'orientation tel que les histogrammes du dégradé dans SIFT [12] et SURF. Le principe de la technique FAST consistant à donner de grandes réponses le long des bords sert également de goulot d'étranglement.

BRIEF est une technique qui utilise un descripteur de chaîne de bits, dérivé d'un patch d'image créé à partir d'un espace échantillon de tests d'intensité binaire. La distance de Hamming, qui est très efficace pour calculer une chaîne binaire, est utilisée dans l'évaluation du descripteur ORB. BRIEF a une limitation est qu'il n'est pas invariant à la rotation.

Afin de gérer le problème de rotation, ORB utilise le centroïde d'intensité pour calculer une direction pour chaque FAST.

La méthode des SURF utilise le fast-Hessien pour la détection de points d'intérêts et une approximation des ondelettes de Haar pour calculer les descripteurs.

## 2.5 conclusion

Dans ce chapitre nous avons présenté les principaux étapes de la construction panoramique ainsi que les méthodes utilisées dans ces étapes , comme le détecteur SIFT , SURF et ORB . Notre travail s'oriente réellement vers l'implémentation de l'application mobile, dans le chapitre qui va suivre on va s'orienter vers l'environnement de développement de ces applications mobiles .

# Chapitre 3 : Environnement de Développement

## 3.1 Introduction

La révolution des Smartphones ne cesse d'exploser qu'elle soit dans leurs designs ou dans les systèmes d'exploitations et les interfaces graphiques qu'ils offrent .

Dans ce chapitre on va aborder les systèmes d'exploitation mobiles que l'industrie des Smartphones ait connu jusqu'à ce jour. On va présenter plus particulièrement le système d'Android[15][16][17] ,son architecture, son environnement et son outils de développement Android Studio.

## 3.2 Les Systèmes d'exploitation mobiles

Il existe actuellement uniquement deux types de systèmes d'exploitation, les Unix (Android, iOS) et Windows. Chaque système d'exploitation dispose de ses propres langages de développement et de ces propres documentations pour créer des applications compatibles. Aujourd'hui,seuls 6 acteurs se partagent 99% du marché des systèmes d'exploitation dans le monde.

**Android** : le système d'exploitation de Google qui équipe la majorité des Smartphones et tablettes d'aujourd'hui, on le découvrira d'avantages par la suite.

**iOS** : le système d'exploitation d'Apple qui équipe exclusivement les iPhone et iPad.

**BlackBerry Os** : BlackBerry OS : le système d'exploitation développé par BlackBerry qui équipe exclusivement les téléphones et Smartphones BlackBerry.

**Symbian OS (Nokia)** : Symbian est le système d'exploitation historique des premiers téléphones Nokia et Motorola. Cet OS est désormais de moins en moins répandu.

**Tizen Os Samsung** : le système d'exploitation Tizen a longtemps équipé les propriétaires de téléphones Samsung.

## 3.3 Système d'Android

### 3.3.1 Android

Android est le système d'exploitation qui prend le lead à l'échelle mondiale .Il est proposé par de nombreuses marques de Smartphones à travers le monde. Il est actuellement l'OS le plus utilisé dans le monde faisant tourner des smartphones, tablettes, montres connectées, liseuses électroniques, télévisions interactives, et bien d'autres. C'est un système open source qui utilise le noyau Linux. Il a été créée par Android, Inc. qui fut rachetée par Google en 2005. Depuis sa sortie , Android a connu de nombreuses versions ,plus performantes les unes que les autres . Dans le tableau ci-dessous on retrouve ces versions :

| Sortie | Version | Nom | API |
|---|---|---|---|
| 09/2008 | 1.0 | Apple Pie | 1 |
| 02/2009 | 1.1 | Banana Bread | 2 |
| 04/2009 | 1.5 | Cupcake | 3 |
| 09/2009 | 1.6 | Donut | 4 |
| 01/2010 | 2.1 | Eclair | 7 |
| 05/2010 | 2.2 | Froyo | 8 |
| 12/2010 | 2.3-2.3.2 | Gingerbread1 | 9 |
| 02/2011 | 2.3.3-2.3.7 | Gingerbread2 | 10 |
| 05/2011 | 3.1 | Honeycomb1 | 12 |
| 07/2011 | 3.2 | Honeycomb2 | 13 |
| 12/2011 | 4.0 | IceCreamSandwich | 15 |
| 07/2012 | 4.1 | JellyBean1 | 16 |
| 11/2012 | 4.2 | JellyBean2 | 17 |
| 07/2013 | 4.3 | JellyBean3 | 18 |
| 11/2013 | 4.4 | KitKat | 19 |
| 11/2014 | 5.0 | Lollipop | 21 |
| 03/2015 | 5.1 | Lollipop | 22 |
| 10/2015 | 6.0 | Marshmallow | 23 |
| 08/2016 | 7.0 | Nougat | 24 |
| 10/2016 | 7.1 | Nougat2 | 25 |
| 08/2017 | 8.0 | Oreo | 26 |

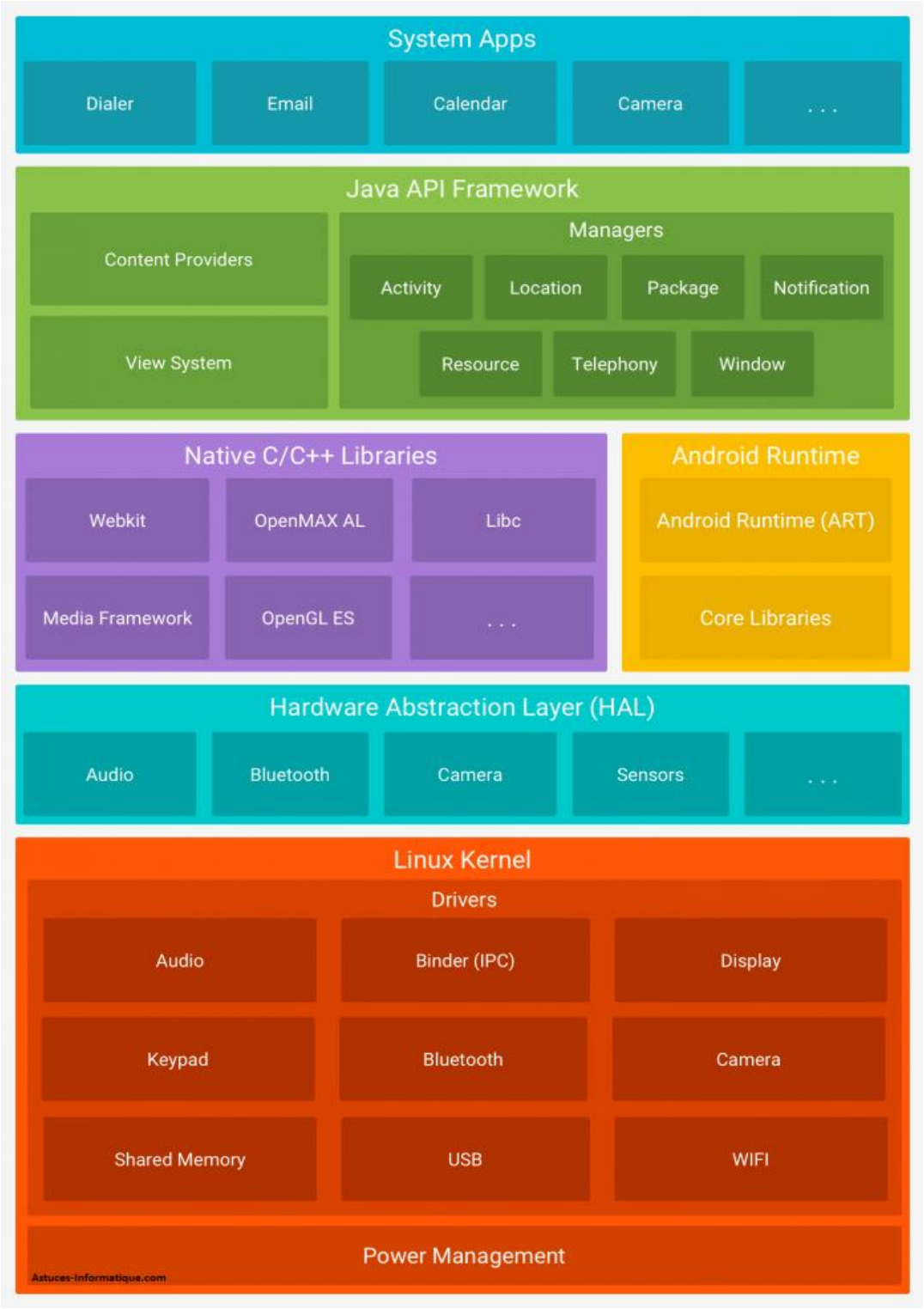
| | | | |
|---|---|---|---|
| 09/2017 | 8.1 | Oreo | 27 |
| 08/2018 | 9.0 | Pie | 28 |

**Tableau 3.1 : Versions d'Android**

Il faut noter que ces versions changent fréquemment ,que pour pouvoir développer sous android ,il faut prendre en compte l'API du système utilisé avec celui cible .On verra d'avantages de détails la dessus dans le chapitre d'implémentation .

### 3.3.2 Architecture d'un système Android

Le système d'exploitation Android est une pile de composants logiciels qui est grossièrement divisé en cinq sections et quatre couches principales, comme indiqué ci-dessous dans le diagramme d'architecture.




. **Figure 3.1 : Architecture Android**

- **Noyaux Lunix**

Au fond des couches est Linux - Linux 3.6. Ceci fournit une couche d'abstraction entre le matériel(la quincaillerie) de dispositif et il contient tous les périphériques essentiels comme l'appareil photo le clavier, l'affichage etc. Aussi , le noyau gère les services du système, comme la sécurité, la gestion de la mémoire et des processus.

- **Les bibliothèques (Libraries)**

En interne, Android inclut un ensemble de bibliothèques C et C++ utilisées par de nombreux composants de la plateforme Android. Ces bibliothèques sont en réalité accessibles au développeur par l'intermédiaire du framework Android. En effet, le framework Android effectue, de façon interne, des appels à des fonctions C/C++ beaucoup plus rapides à exécuter que des méthodes Java standard. La technologie Java Native Interface (JNI) permet d'effectuer des échanges entre le code Java et le code C et C++. La liste ci-dessous énumère quelques-unes des bibliothèques disponibles dans Android :

-***Bibliothèque système C***. Implémentation (dérivée de BSD) de la bibliothèque standard C (libc), optimisée pour les systèmes Linux embarqués.

-***Bibliothèques multimédias***. Basées sur StageFright, elles permettent le support de nombreux formats audio et vidéo, tels que MPEG4, H.264, MP3, AAC, AMR, JPG et PNG …etc .

-***SurfaceFlinger***. Permet l'accès au sous-système d'affichage.

-***LibWebCore***. Moteur de rendu de pages Internet basé sur ***Webkit***. Cette bibliothèque est donc principalement utilisée dans le navigateur et dans les vues web embarquées (WebView).

-***Skia***. Moteur graphique 2D.

-***Bibliothèques 3D***. Implémentation basée sur OpenGL ES 1.0 API et plus récemment OpenGL ES 2.0.

-***FreeType***. Rendu des polices de caractères.

-***SQLit***. Base de données légère et puissante.

- **Android Runtime(ART)**

Il s'agit de la troisième section de l'architecture et disponible sur la deuxième couche en bas. Cette section fournit un composant clé appelé Dalvik Virtual Machine, qui est une sorte de machine virtuelle Java spécialement conçue et optimisée pour Android.

La **VM Dalvik** utilise des fonctionnalités principales de Linux comme la gestion de la mémoire et le multi-threading, intrinsèque dans la langue Java. La VM Dalvik permet à toute application Android de fonctionner dans son propre processus, avec sa propre instance de la machine virtuelle **Dalvik.** . Il faut noter qu'à partir de la version 5.0 sortie en 2014, l'environnement d'exécution ART (**Android RunTime**) remplaça la machine virtuelle Dalvik.

Avec **ART**, contrairement à la machine virtuelle java, les fichiers .apk ne sont plus lancés directement mais décompressés et lancés avec de nouvelles bibliothèques et API ; les applications prennent ainsi plus de place (+20 %), mais les gains en performance et en autonomie des batteries sont conséquents.

**Le runtime d'Android** fournit également un ensemble de bibliothèques de base qui permettent aux développeurs d'applications Android d'écrire des applications Android en utilisant un langage de programmation Java standard .

- **Framework d'application**

La couche Framework d'application fournit de nombreux services de niveau supérieur aux applications sous la forme de classes Java. Les développeurs d'applications sont autorisés à utiliser ces services dans leurs applications.Le cadre Android comprend les services clés suivants:

***Activity Manager*** – Contrôle tous les aspects du cycle de vie de l'application et de la pile d'activité.

***Fournisseurs de contenu*** – Permet aux applications de publier et de partager des données avec d'autres applications.

***Resource Manager*** – Fournit l'accès à des ressources non intégrées au code telles que les chaînes, les paramètres de couleur et les mises en page de l'interface utilisateur.

***Notifications Manager*** – Permet aux applications d'afficher des alertes et des notifications à l'utilisateur.

***View System*** – Un jeu extensible de vues utilisé pour créer des interfaces utilisateur d'application.

## 3.4 Outils De Développement Android

Pour développer des applications sous Android , il existe plusieurs environnement avec un ensemble d'outils nécessaire. parmi ces outils on retrouve :

### 3.4.1 Le JDK (Java DeveloppmentKit)

Les applications développées pour Android étant essentiellement écrites en langage java, un langage de programmation orienté objet qui a la particularité d'être très portable. Cela signifie qu'un programme Java, fonctionnant sur Windows (par exemple), pourra facilement tourner sur Mac ouGNU/Linux.

Cette petite prouesse vient du fait que Java s'appuie sur une machine virtuelle pour s'exécuter (appelée la JVM). Pour avoir une JVM ,il faut télécharger le JRE. Ce dernier contient, en plus de la JVM, des bibliothèques Java standards. La JVM ne lit pas directement le code Java. Elle lit un code compilé (le byte code). Pour passer du code Java, que le développeur écrit, au code compilé, lu par la JVM, des outils spéciaux sont nécessaires. Ces outils sont inclus dans le JDK. De plus, le JDK contient le JRE (et donc la machine virtuelle), ce qui est bien pratique.

### 3.4.2 Le SDK (Software Development Kit)Android

Les applications Android sont développées en Java, mais un appareil sous Android ne comprend pas le Java tel quel, il comprend une variante du Java adaptée pour Android . Un SDK, c'est-à-dire un kit de développement logiciel, est un ensemble d'outils que met à disposition un éditeur afin de permettre de développer des applications pour un environnement précis. Le SDK Android permet, donc, de développer des applications pour Android et uniquement pour Android.

### 3.4.3 L'IDE Eclipse

Eclipse est un environnement de développement intégré. C'est un logiciel qui permet d'écrire un programme beaucoup plus facilement qu'avec le simple Bloc-notes. Outre la coloration du code, il permet d'apporter des outils très pratiques pour compiler vos programmes, les déboguer, etc. Il peut être utilisé pour programmer avec n'importe quel type de langage, mais nous l'utiliserons pour faire du Java. De plus, Eclipse est conçu pour pouvoir être complété avec des plugins (extension). Ainsi, il existe un plugin pour développer des applications Android que nous verrons dans la partie suivante.

### 3.4.4 Le plugin ADT pourEclipse

Google fournit un plugin pour Eclipse, nommé ADT (Android Development Tools), la fonction principale de ce plugin est de créer un pont entre Eclipse et le SDK Android.

Jusqu'à Mai 2013, pour développer des applications pour Android, Google mettait en avant l'utilisation d'Eclipse couplé avec le Plugin ADT (Android Development Tools). Cette première solution a tout de même permis à Google de posséder le Store d'application le plus riche. Eclipse est un IDE qui a été développé par IBM puis est passé open source en 2001, la Fondation Eclipse gère maintenant l'IDE. Eclipse possède les avantages d'être modulable ainsi que multi plate-forme.

### 3.4.5 L'environnement Android Studio

Google a créé un environnement de développement complet pour la création d'application mobile Android nommé Android Studio. Il est open source et disponible gratuitement, permettant de réaliser des projets sur différents types de support, tablette ou Smartphone. Il permet principalement d'éditer les fichiers Java et les fichiers de configuration d'une application Android .

C'est durant la Google I/O de 2013, que Google a montré la première version d'Android Studio. En Access preview au départ pour sa version 0.1, puis passé en béta en juin 2014 pour la version 0.8, jusqu'à l'arrivé à la version 1.5.cet IDE n'a pas été développé de zéro mais est basé sur l'IDE de JetBrains, IntelliJ IDEA. Cette société propose de nombreux IDE pour différents langages (PHP Storm, RubyMine, ...) mais qui sont tous payant. Dans sa dernière version, Android Studio offre toutes les possibilités nécessaires pour développer une application Android complète.

En plus d'être pratique , il est L'IDE officiel supporté par Google pour développer sous Android, c'est pour cette raison que l'on va l'utiliser pour notre application par la suite.

## 3.4 Caractéristiques d'une Application Android

### 3.5.1 Composantes d'une Application Android

Une application Android se compose de plusieurs éléments. Dans ce qui suit, nous allons essayer de découvrir les plus importants :

- ***Les Activités***

Une activité est la composante principale pour une application Android. Elle représente l'implémentation et les interactions des interfaces. Plusieurs choix se proposent pour mettre en place l'interface visuelle :

-Utiliser un fichier XML pour décrire l'interface.

-Créer les éléments de l'interface à l'intérieur du code java.

- ***LesServices***

Un Service est, en fait, un programme tournant en tâche de fond et n'ayant pas d'interface graphique. L'exemple commun illustrant cette notion, est celui du lecteur mp3.Un lecteur mp3 ne nécessite pas, pour la plupart du temps, d'interface graphique et doit tourner en tâche de fond, laissant la possibilité aux autres applications de s'exécuter librement. Un service peut être lancé à différents moments : Au démarrage du téléphone.Au moment d'un événement (arrivée d'un appel, SMS, mail, etc…).Lancement de l'application.Action particulière dans application.

- **Les Broadcast Receivers**

Un BroadcastReceiver, comme son nom l'indique, permet d'écouter ce qui se passe sur le système ou sur votre application et de déclencher une action que vous aurez prédéfinie. C'est souvent par ce mécanisme que les services sont lancés.

- **Les Content providers**

Les Content Provider sont, comme l'exprime leur nom, des gestionnaires de données. Ils permettent de partager l'information entre applications. Vous pouvez accéder : Aux contacts

stockés dans le téléphone.A l'agenda. Aux photos. Ainsi que d'autres données depuis votre application grâce aux content providers.

- **Les Intents**

Les Intents sont des objets permettant de faire passer des messages contenant de l'information entre composants principaux. La notion d'Intent peut être vue comme une demande de démarrage d'un autre composant, d'une action à effectuer.

### 3.5.2 Cycle de vie d'une Application Android

Le diagramme d'état suivant présente les principaux états du cycle de vie d'une activité Android, il est suivi d'une description des principales méthodes événementielles du cycle de vie d'une activité .

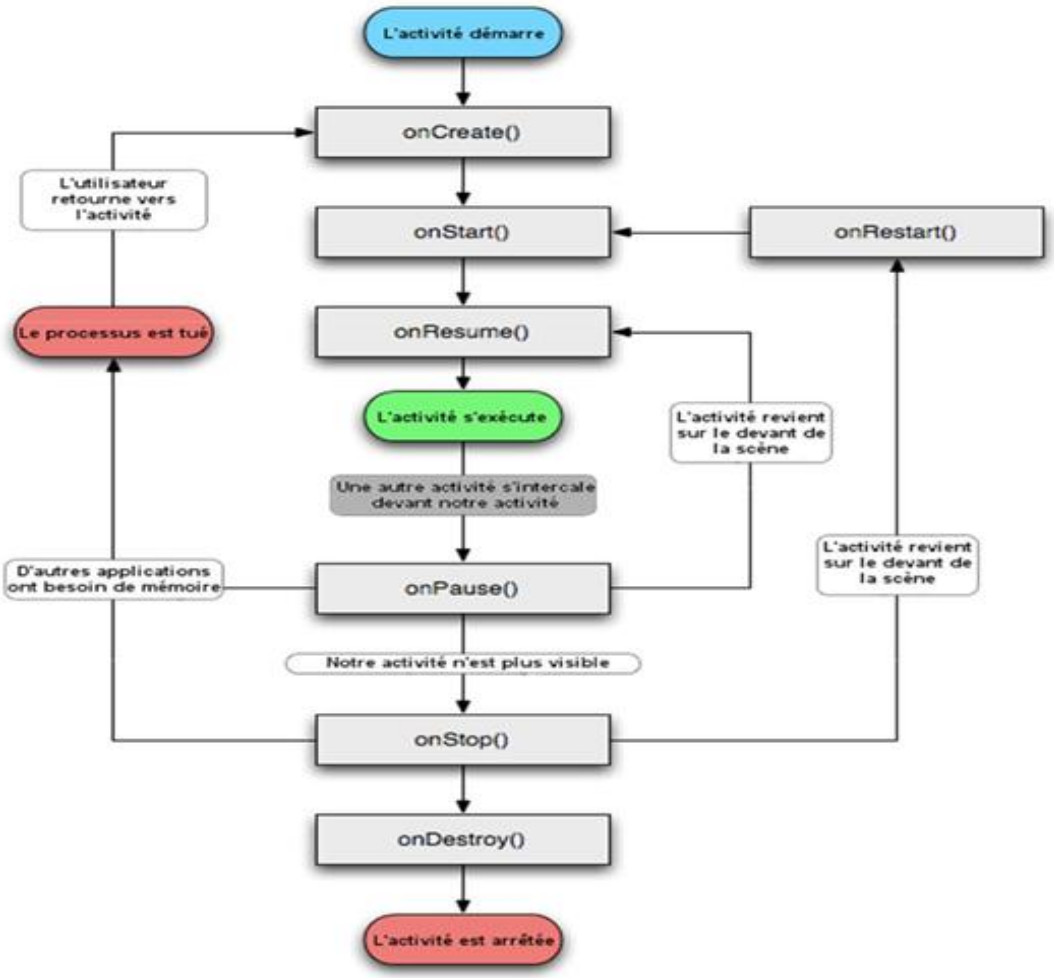


**Figure 3.2 : Cyclé de vie d'une application Android**

La méthode onCreate () est appelée à la création de votre activité. Elle sert à initialiser l'activité ainsi que toutes les données nécessaires à cette dernière. Quand la méthode OnCreate () est appelée, on lui passe un Bundle en argument. Ce Bundle contient l'état de sauvegarde enregistré lors de la dernière exécution de l'activité.

La méthode onStart () signifie le début d'exécution de l'activité (début du passage au premier plan). Si l'activité ne peut pas aller en avant plan, pour une quelconque raison, elle sera transférée à OnStop ().

La méthode *onResume ()* est appelée lorsque l'activité commencera à interagir avec l'utilisateur juste après avoir été dans un état de pause.

La méthode *onPause()* est appelée au passage d'une autre activité en premier plan. L'intérêt d'un tel appel est de sauvegarder l'état de l'activité et les différents traitements effectués par l'utilisateur. A ce stade, votre activité n'a plus accès à l'écran, vous devez arrêter de faire toute action en rapport avec l'interaction utilisateur (désabonner les listeners).

La méthode *onStop()* est appelée quand l'activité n'est plus visible quel que soit la raison. Dans cette méthode vous devez arrêter tous les traitements et services exécutés par votre application.

La méthode *onDestroy ()* est appelée quand votre application est totalement fermée (Processus terminé). Les données non sauvegardées sont perdues

### 3.5.2 Problèmes Rencontrés

Durant nos quelques mois de travail, nous avons rencontré des soucis ,que nous jugeons important à signaler pour les futur lecteurs de ce mémoire ,afin de contribuer à la bonne installation et mise en œuvre de l'environnement pour les futures étudiants ,qui souhaiteraient se lancer dans l'application mobile .

Afin de les éviter nous proposons de :

- Toujours prendre en considérations les versions utilisées,

- S'assurer du bon fonctionnement des pilotes dans le gestionnaires de périphérique de l'ordinateur utilisé ,

-Et enfin , suivre toutes les étapes proposées dans les tutoriels et cours en faisant très attention lors de n'importe quelle configuration .

## 3.6 Conclusion

Dans ce chapitre nous avons présenté quelque outils de développement Android ,donné un aperçu sur le système d'exploitation mobile Android et ainsi permettre aux nouveaux développeurs de mieux cerner l'environnement .

# Chapitre 4 : Conception du Système

## 4. 1 Introduction

Rappelons d'abord que le but de notre travail est de réaliser une application dédiée à la construction de panoramas sous Android. Dans les chapitres précédents on a présenté d'une part les différentes méthodes qui existent pour réaliser des panoramas et le système d'exploitation mobile Android de l'autre .Les étapes de construction sont toujours les même que le système soit destiné pour un logiciel ou pour une application, il faut juste s'avoir s'adapter lors de l'implémentation de l'algorithme du processus dans le langage choisi.

Dans ce chapitre nous allons montrer brièvement l'architecture générale de notre système et les processus de réalisation du système.

## 4.2 La conception Globale du système

Comme déjà expliqué , on veut mettre au point un système qui , en entrées reçoit plusieurs images et renvoie en sortie une image panoramique .l'ensemble de processus effectué sur les images , consistent en les étapes de construction panoramique déjà vu dans les chapitres précèdent .

Si le système reçoit seulement deux images en entrée, alors les opérations sont realisées sur ces deux images , pour un résultat en image panoramique ,

Et si les images entrées sont plusieurs, alors pour chaque deux images le traitement est effectué et une fois assemblée, l'image résultat est à sont tour traité l'image suivante ,jusqu'à ce qu'il n'y ait plus d'images a ajouter au panorama qui sera l'image finale .

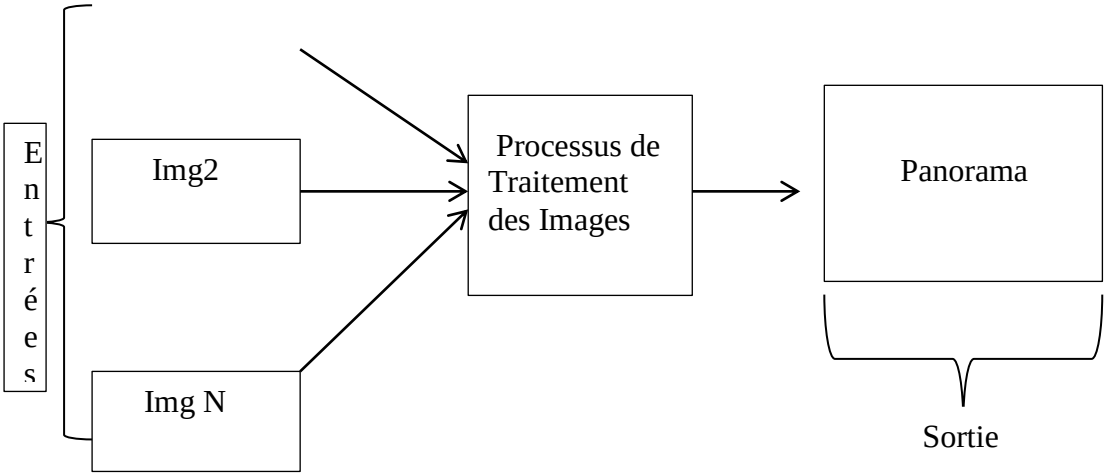



**Figure4.1 : Schéma représentatif du système**

**PseudoAlgorithme du processus() :**

E {}= Ensemble Image , Image_resulat

,Processus_traitement // Fonctioon

Début :

Tant que l'ensemble d'image n'est pas vide

Image_resultat←Processus_traimtement(I1,I2)

Supprimer I1,I2 de E

Ajouter Image_resulatt à E

Fin tant que

Afficher ImageResluatt

Fin

## 4.3. Processus de Traitement

Le processus de traitement contient toutes les opérations à réaliser sur les images

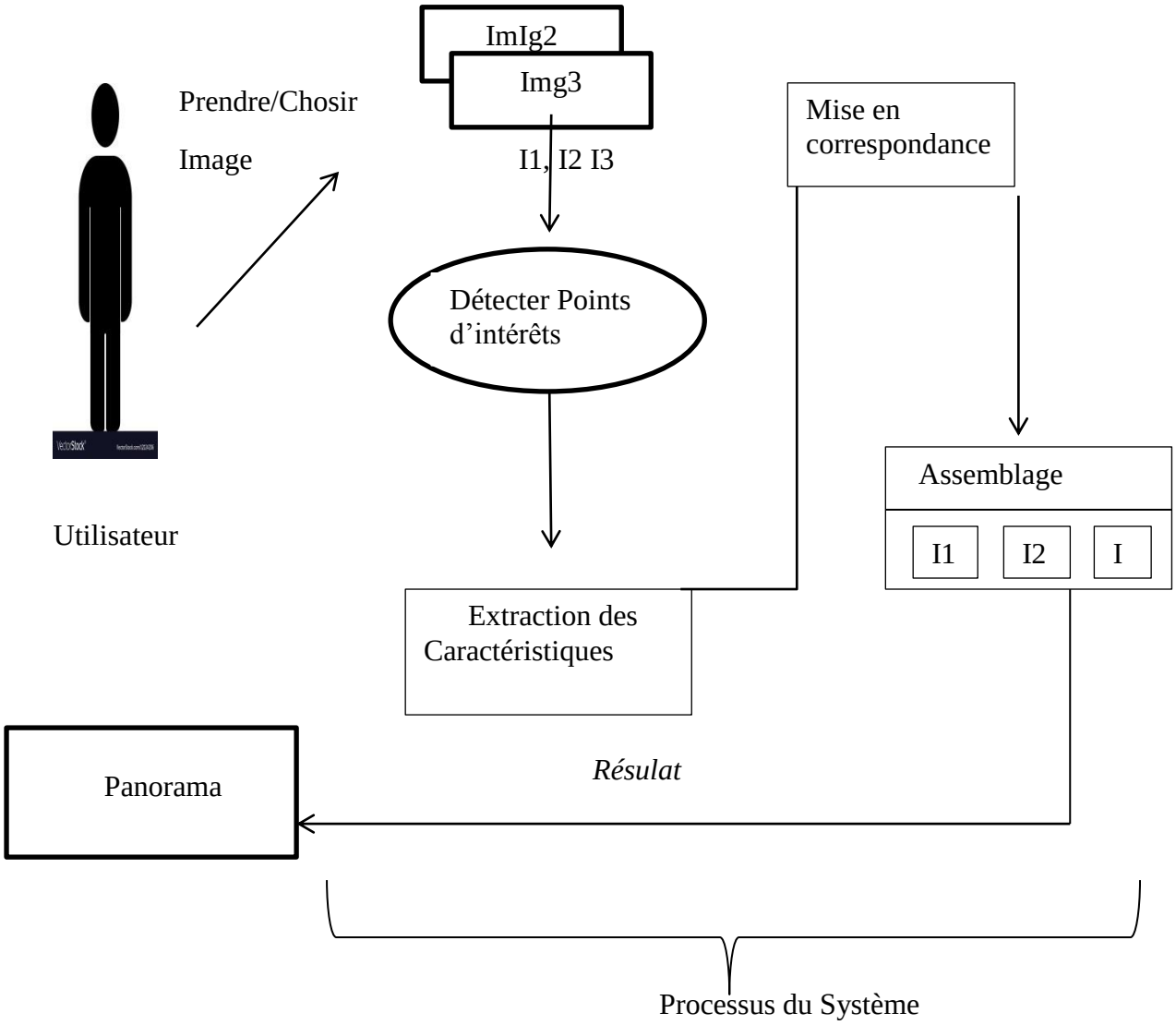



Figure 4.2 : Représentation du processus

Notre processus de traitement effectue les taches à réalisées par le système ,. Toutes les étapes du Stitching sont réalisés dans le processus :

- Le système reçoit les images entrées par l'utilisateur ;
- Les images sont analysées pour trouver les points d'intérêts de chaque images ; avec le détecteur surf ,les caractéristiques seront extraites ;

- Les caractéristiques extraites seront comparées , si les images contiennent les mêmes valeurs de caractéristiques , alors le système va entamer le processus de fusion, dans le cas contraire ,le système ne fait rien : Images non destiné à etre assemblée
- Le processus de fusion va permettre de fusionner les images en une seule image ,qui sera l'image résultat Panorama .
- Si l'image Résultat n'est pas tout a fait parfaite, un transformation est réalisée .
- Le résultat que l'utilisateur verra sera le panorama final des images entrées .

## 4.5 Conclusion

Dans ce chapitre on a présenté brièvement la conception de notre système ,car notre travail se concentre sur l'application ,les étapes théoriques de la procédure sont expliquées mais pour la conception on va se concentrer beaucoup plus sur l'implémentation .

# Chapitre 5 Implémentation et présentation de l'application

## 5.1 Introduction

Pour notre application l'outils de développement Android studio a lui seul ne suffit pas pour effectuer le traitement sur les images . Il servira certes de plateforme de programmation ,qui va nous permettre d'effectuer l'exécution sur un smartphone , mais aussi va nous permettre d'ajouter les bibliothèques nécessaires comme OpenCV4Android (Open Computer Vision pour Android),JavaCV(Java Computer Vision) pour travailler sur les images et construire le panorama.

Ce chapitre sera consacré donc à la présentation de notre application. On va y montrer la configuration de l'environnement de développement Android Studio avec les outils nécessaires pour le traitement d'images [14] et les différentes UI de l'application et enfin on va présentera notre application finale .

## 5.2 Les Outils requis

### 5.2.1 Outils

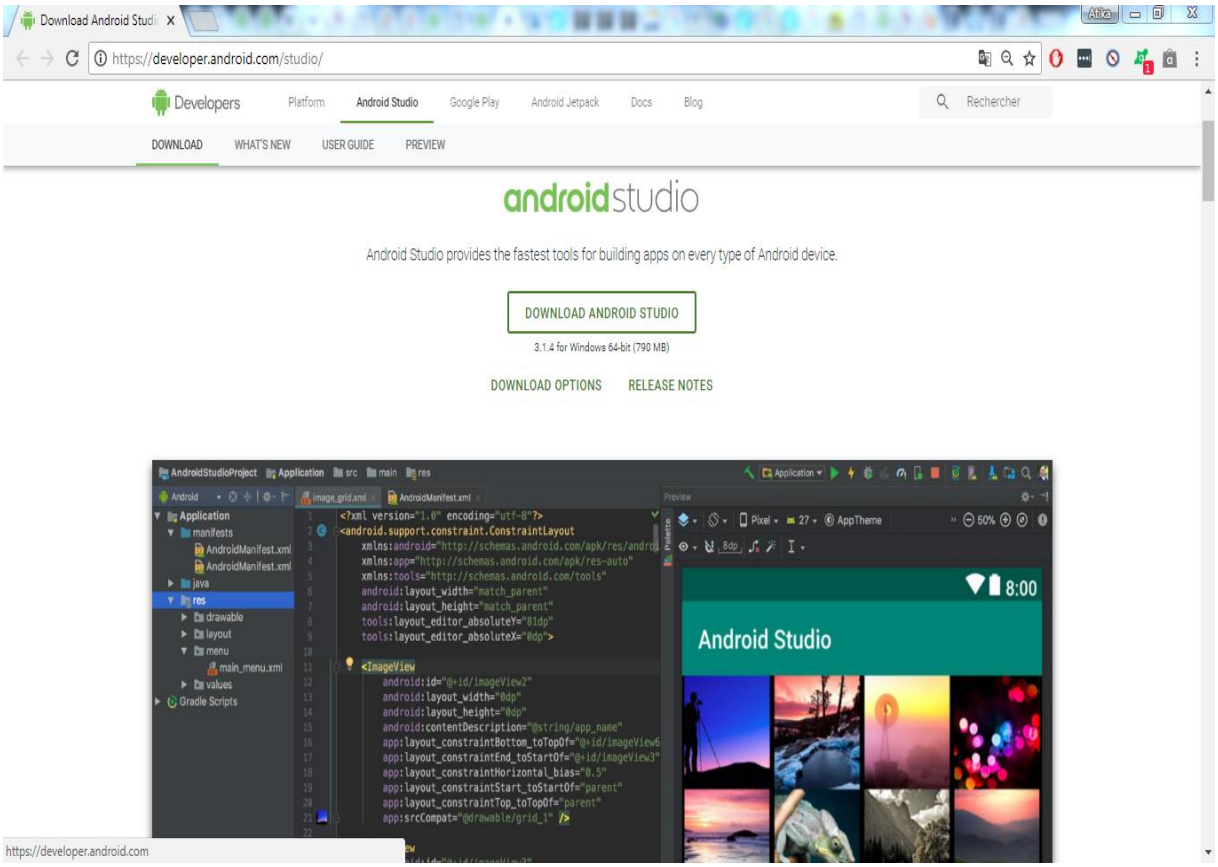
Pour notre application ,comme expliqué dans le chapitre précèdent, on va se servir d' Android sutdio , mais aussi d'autre environnement . Ce qui suit représente la liste entière de ces outils :

#### 5.2.1.1 Android Studio

***Téléchargement et installation***

Pour installer Android Sudio, il faut se rendre sur la page de téléchargement de l'IDE (https://developer.android.com/studio/) . Normalement, la page détecte automatiquement le système d'exploitation et propose de télécharger l'archive compatible (pour Mac, Windows ou Linux),qui est la dernière version de l'IDE qui existe sur le marché.

Pour télécharger une différente version , plus ancienne par exemple , il faut cliquer sur *Download Options* et sélectionner alors la version de votre choix




. **Figure 5.1 : Site Android Studio**

Pour notre travail, on dispose de la version 3.1.3 ,voici la fenêtre d'accueil de l'application :

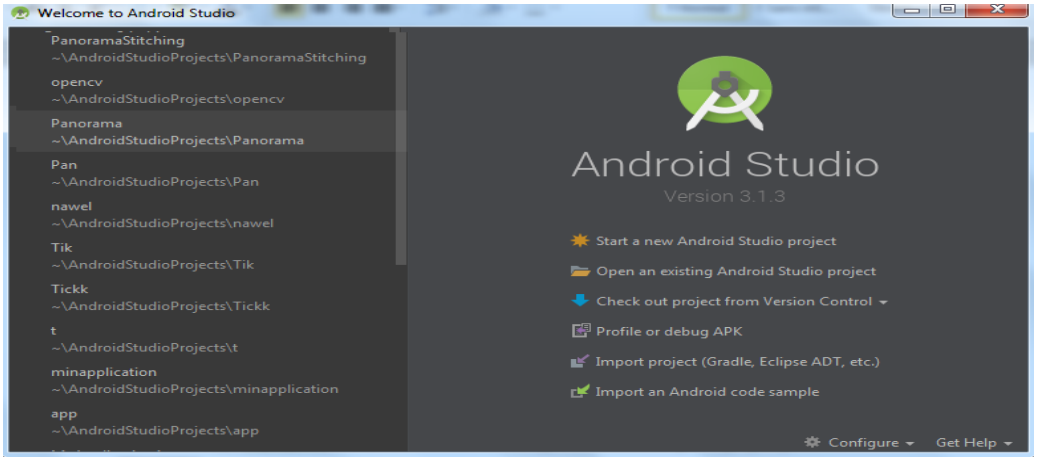



**Figure 5.2 : Acceuil Android Studio**

#### 5.2.1.2 OpenCV4Android

OpenCV est une bibliothèque graphique libre, spécialisée dans le traitement d'images en temps réel. met à disposition de nombreuses fonctionnalités très diversifiées permettant de créer des programmes partant des données brutes pour aller jusqu'à la création d'interfaces graphiques basiques.

OpenCV4Android comme son nom l'indique , c'est celle qu'on peut peut utiliser pour le développement mobile . La verison utilisée pour ce projet est la Opencv3.4.1

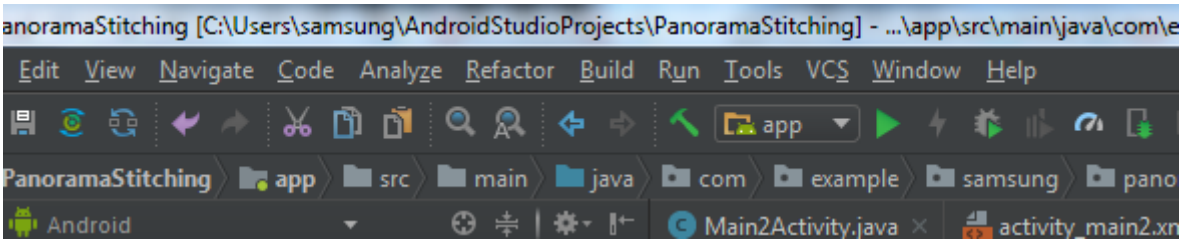
La bibliothèque est téléchargeable sur le site « https://opencv.org/platforms/android/ »

#### 5.2.1.3 NDK (Native Developement Kit)

Le NDK (Native Developpement Kit – Kit de développement natif) est un outil permettant d'implémenter des parties natives (en C ou C++) pour vos application. Cela vous permet par exemple, de créer du code commun (bibliothèque) entre plusieurs applications ou entre des applications Android.

On a beosin de rédiger du code en C++ ,car l'opencv java ne dispose pas de ce dont nous avons besoin pour effectuer le sttiching . le native nous permettera de faire appel a la fonction principale de notre application [14].

Pour le télécharger il suffit d'aller sur Tools → SDK Manager →



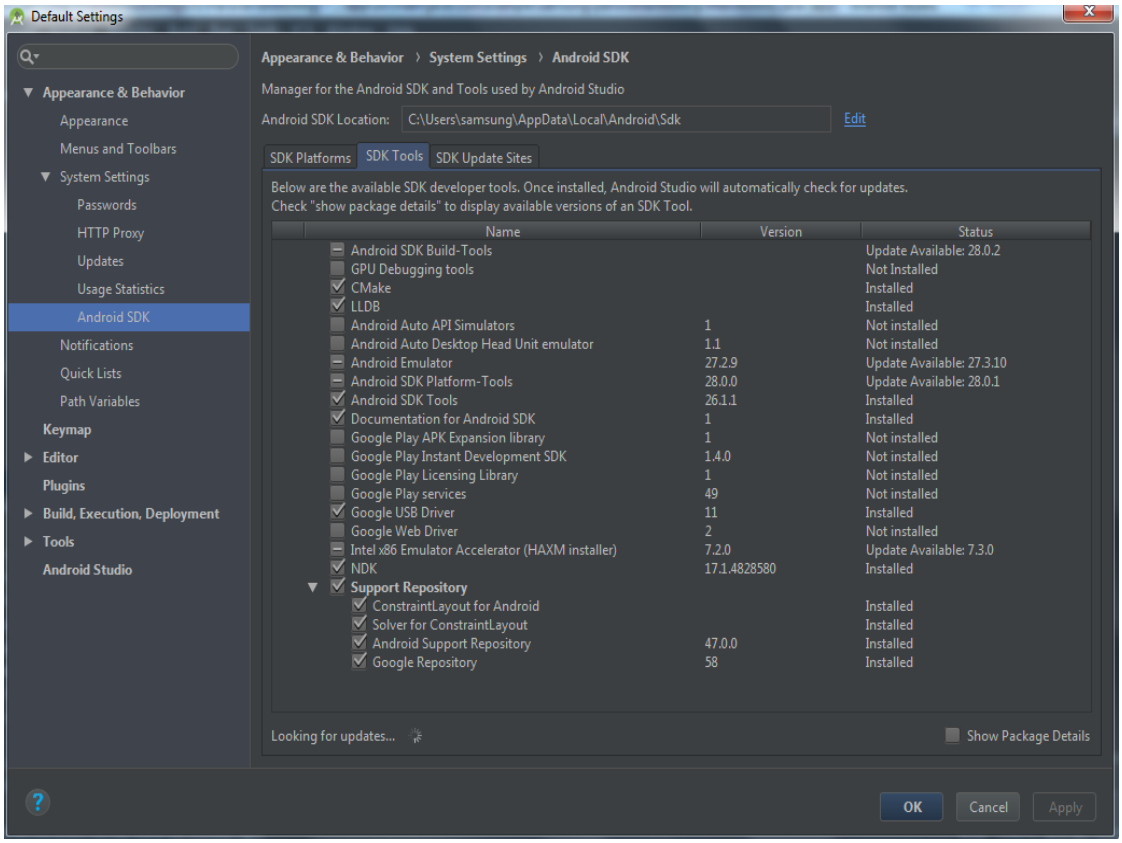



**Figure 5. 3 : Installation de NDK**

## 5.2.2 Configuration de l'OpenCV sous Android studio

Pour configurer l'opencv sous android il faut suivre les étapes illustrée la dessous :

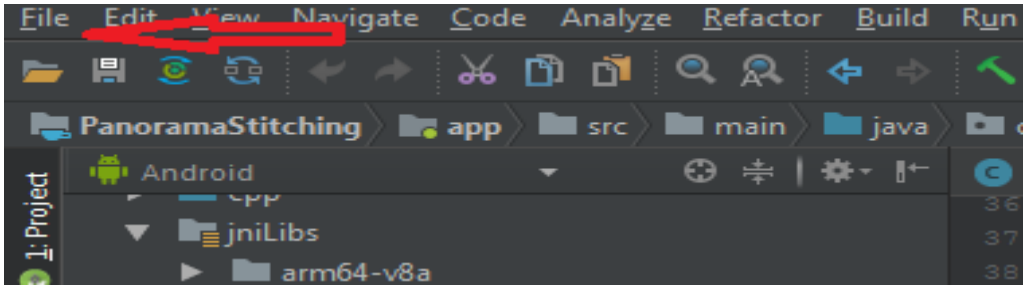



**Figure 5. 4 Menu android studio**

File→ New → Import Module : la fenêtre suivante s'affiche :

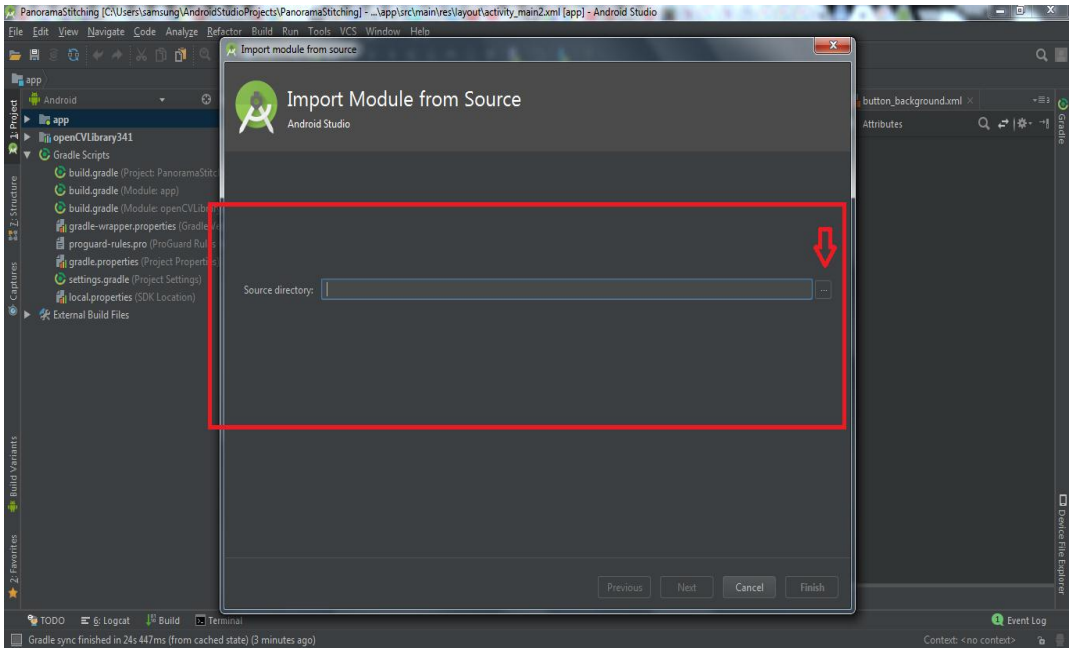



**Figure5.5 : Fenêtre d'importation du module**

Choisir après le module opencvfo4android déjà téléchargement ,dossier java

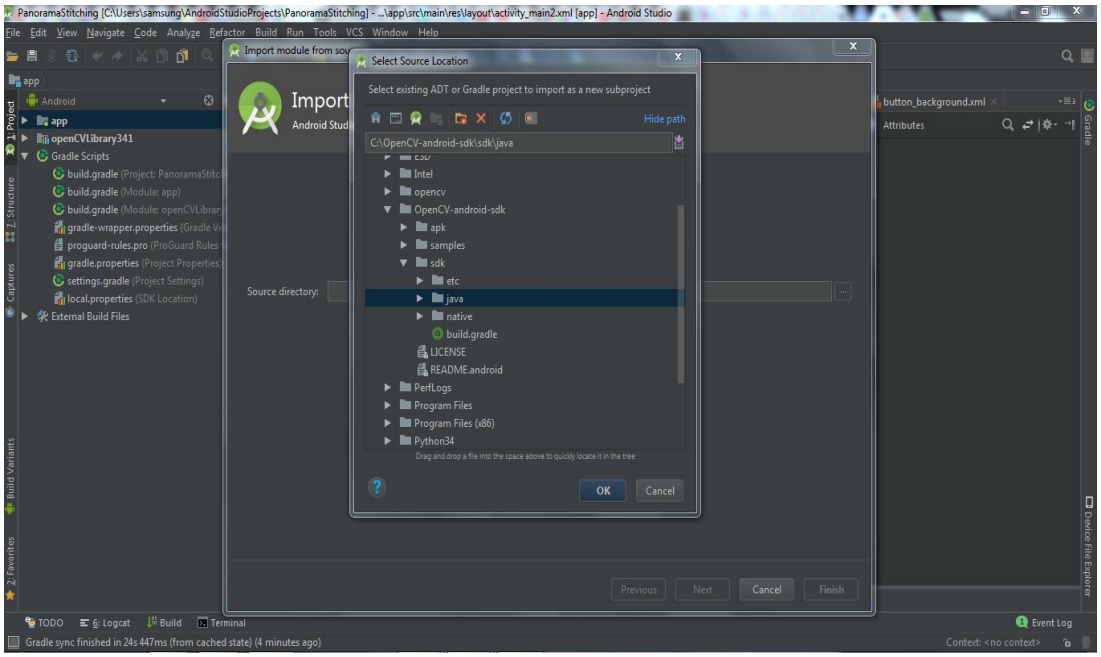



**Figure 5. 6 Sélection du module opencv/sdk/java**

Ensuite, il faut aller sur la table de dépendance et l'ajouter

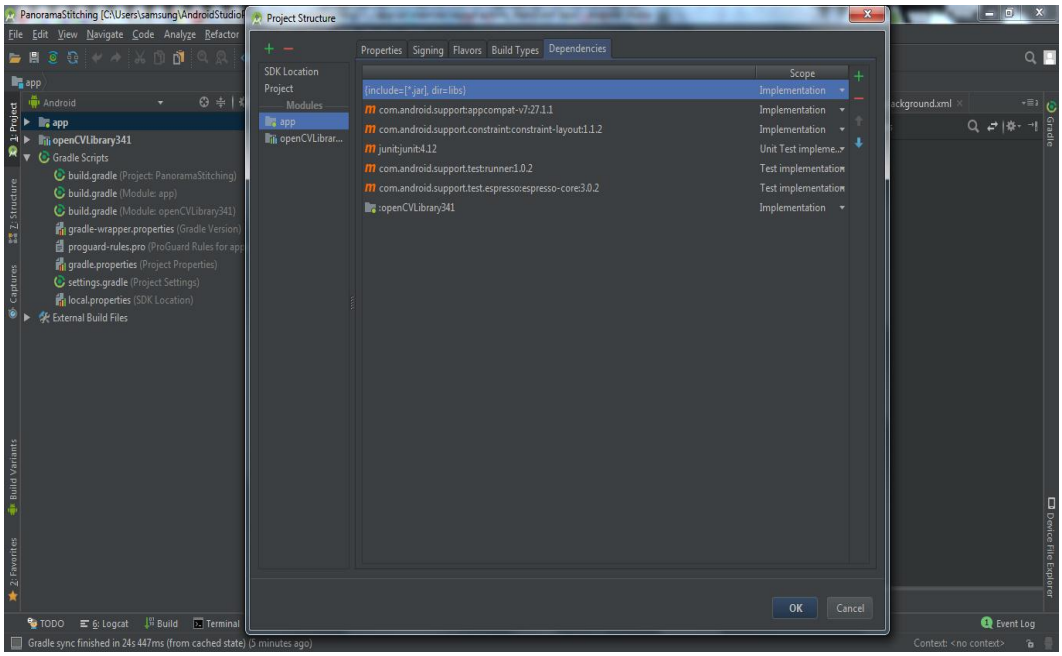



Figure 5.7 : Fenêtre de la structure du projet

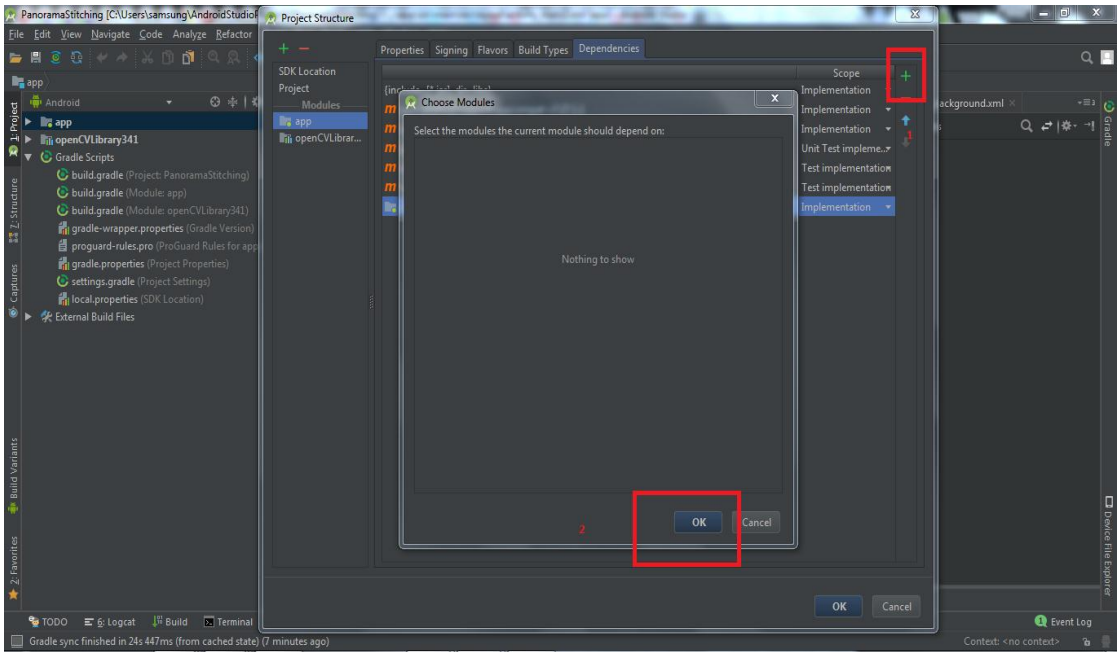



Figure 5.8 : ajouter le module a la table d'independance

Une fois ces étapes réalisée, il suffit juste de changer le Gradle(Opencv) et le mettre à jour avec celui de l'application, qui dépend de la version ciblée de l'android,et de synchroniser le projet . Les figures suivantes le montrent

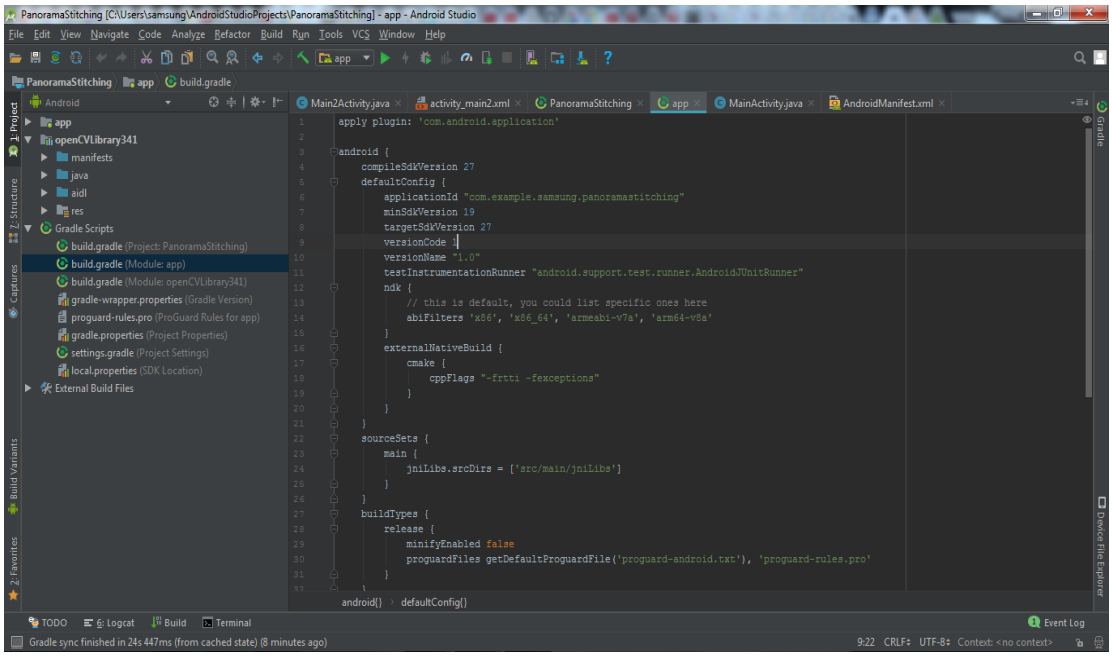



**Figure 5.10 : Valeur du Gradle cible de l' application**

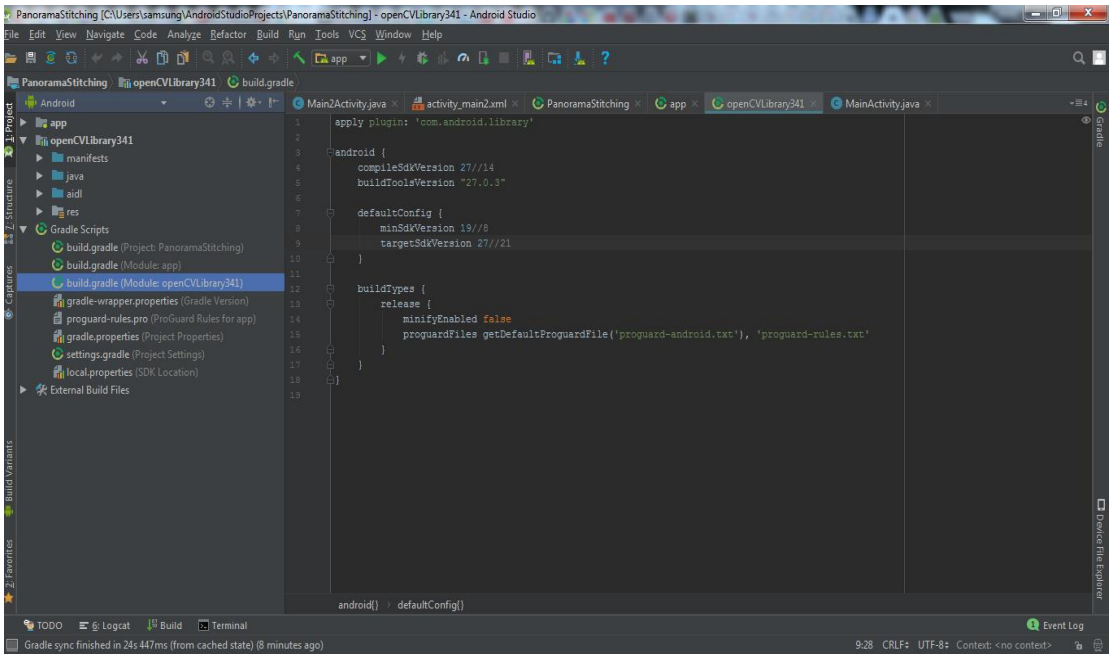



**Figure 5 .11. : Fenêtre représentant du Gradle OpenCV**

En principe, après ces étapes l'openCv est ajouté a notre projet android ,comme montrée dans la figure suivante :

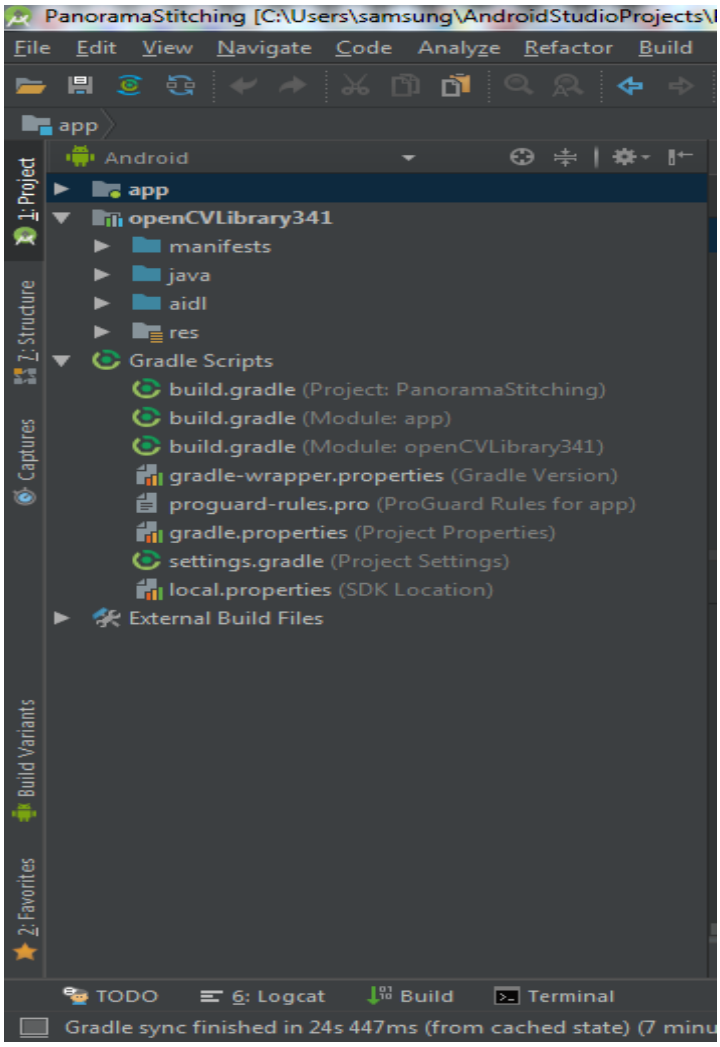



**Figure 5.12 : Capture montrant l'Opencv ajoutée avec succès**

## 5.2.2 Ajout de NDK au projet

Ainsi après l'installation du NDK sur l'android studio ,il nous faut encore passer par quelques étapes afin de pouvoir faire appel au fonctions écrites en c++ a partir de la classe principale de note code java .

Les étapes seront expliquées dans ce qui va suivre :

1- Créer un dossier, sous App/src/main/ de notre projet, on va le nommer jniLibs ,

2- Ajouter les dossiers qu'on voit sur la figure à partir de opencv4android/dsk/native/libs , en les copiant et les collant sur le dossier jniLibs crée .

Le résultat sera comme suit :

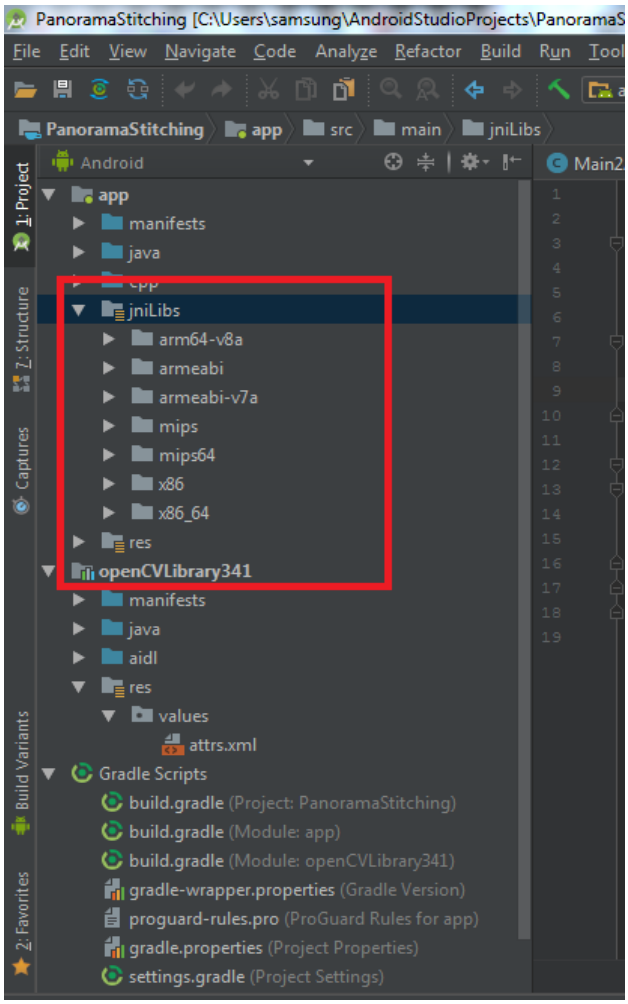



**Figure5.13 : Capture d'écran du dossier NDK à ajouter .**

```
static {
    System.loadLibrary( libname: "native-lib");
    System.loadLibrary( libname: "opencv_java3");
}
ImageButton btc;
private static final int CAPTURE_IMAGE_ACTIVITY_REQUEST_CODE = 100;
private static final int CAMERA_REQUEST = 1888;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    TextView tv = (TextView) findViewById(R.id.sample_text);

    if (!OpenCVLoader.initDebug()) {
        tv.setText(tv.getText() + "\n OpenCVLoader.initDebug(), not working.");
    } else {
        tv.setText(tv.getText() + "\n OpenCVLoader.initDebug(), WORKING.");
        //DRS 20160822c Added 1
        tv.setText(tv.getText() + "\n" + validate( matAddrGr: 0L,  matAddrRgba: 0L));
    }

    ImageButton btc = (ImageButton)findViewById(R.id.btc);

    ImageButton btnCamera = (ImageButton)findViewById(R.id.btnCamera);
```

Figure 5.14 : chargement de la classe native dans la classe java

```
Main2Activity.java   activity_main2.xml   PanoramaStitching   app   openCVLibrary341   MainActivity.java
apply plugin: 'com.android.application'

android {
    compileSdkVersion 27
    defaultConfig {
        applicationId "com.example.samsung.panoramastitching"
        minSdkVersion 19
        targetSdkVersion 27
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
        ndk {
            // this is default, you could list specific ones here
            abiFilters 'x86', 'x86_64', 'armeabi-v7a', 'arm64-v8a'
        }
        externalNativeBuild {
            cmake {
                cppFlags "-frtti -fexceptions"
            }
        }
    }
    sourceSets {
        main {
            jniLibs.srcDirs = ['src/main/jniLibs']
        }
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
        }
    }
android{} > defaultConfig{}
```

Figure 5.15 : Ajouter dans le gradle (app) les libs ajouter dans le dossier jniLibs

Modifier le cmake pour réaliser le ndk avec notre opencv

```
# For more information about using CMake with Android Studio, read the
# documentation: https://d.android.com/studio/projects/add-native-code.html

# Sets the minimum version of CMake required to build the native library.

cmake_minimum_required(VERSION 3.4.1)

#set(pathToOpenCv ${CMAKE_SOURCE_DIR}C:/OpenCV-android-sdk)

 set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=gnu++11")

 include_directories(C:/OpenCV-android-sdk/sdk/native/jni/include)
 add_library(native-lib SHARED
             src/main/cpp/native-lib.cpp )

 add_library( lib_opencv SHARED IMPORTED )
 set_target_properties(lib_opencv PROPERTIES IMPORTED_LOCATION
                       ${CMAKE_SOURCE_DIR}/src/main/jniLibs/${ANDROID_ABI}/libopencv_java3.so)

find_library( # Sets the name of the path variable.
              log-lib

              # Specifies the name of the NDK library that
              # you want CMake to locate.
              log )

 target_link_libraries(native-lib log lib_opencv)
```

Figure 5.16 : CmakeText avec les librairies ajoutées

-**Remarque :** On a rencontré des problèmes lors de l'ajout du NDK , il faut penser à vérifier l'architecture du système android cible, sur le quel on va débuguer et exécuter l'application .on peut rencontrer certains bugs a cause de la verison ndk et sa compatibilité avec le **sdk** .

## 5.3 Interface Graphique

Notre interface application , nous permet de prendre plusieurs photos ou de choisir des photos à partir de la galerie[18] .

Avant pour pouvoir l'exécuter sur smartphone , il faut y effectuer ce réglage :

-Aller dans Paramètres → Option Développeurs→ cocher la case « »Déboguer »

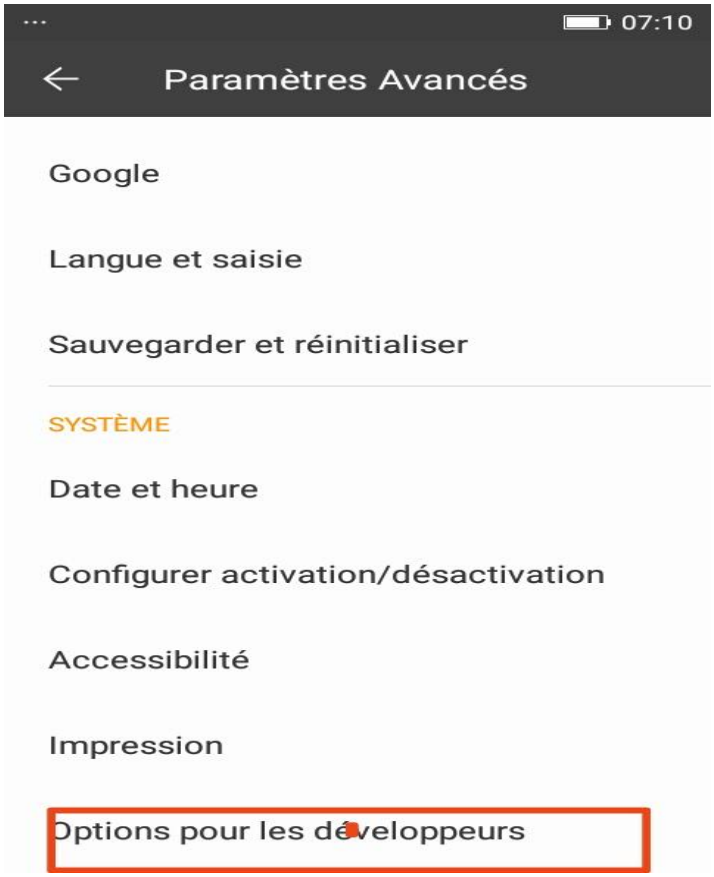



**Figure 5.17 ScreenShopt du réglage**

**Remarque** : L'emplacement de « Options pour les développeurs » dépend de la version d'Android utilisée .

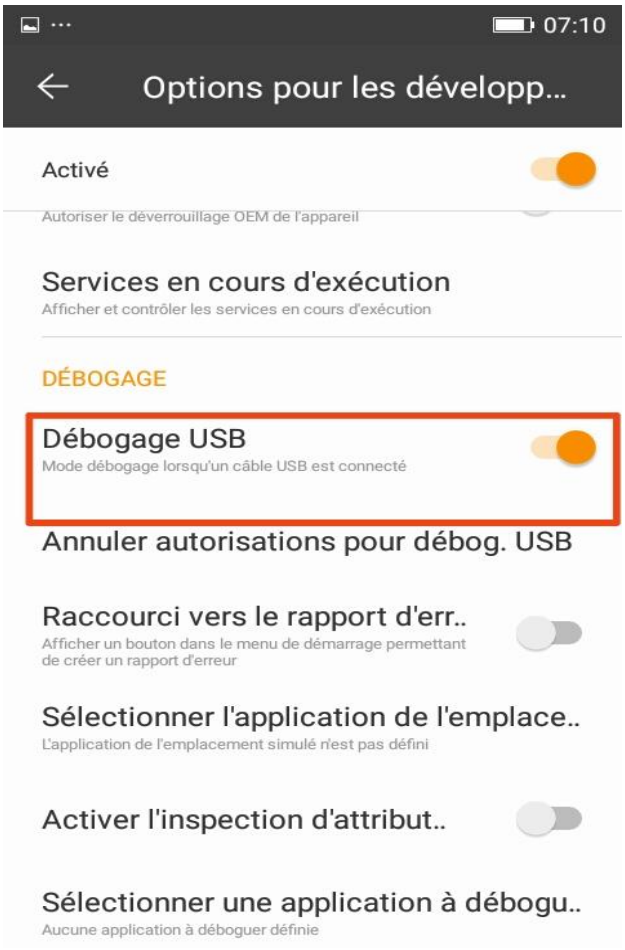



**Figure 5.18 : ScreenShot operation Deb 2**

On a implémenté la fonction suivante pour effectuer un assemblage de plusieurs images :

- La fonction prend en paramètre l'adresse des images envoyé depuis la classe principale java, après traitement nécessaire, elle sera renvoyé toujours sous forme d'adresse pour être convertie en image et afficher

```
std::string hello2 = "Fonction Opencv Teste";

return env->NewStringUTF(hello2.c_str());
}
JNIEXPORT jint JNICALL
Java_com_example_samsung_panoramastitching_MainActivity_panorama(JNIEnv *env, jobject,
                                                                 jlongArray imageAddressArray,
                                                                 jlong outputAddress) {
    jsize a_len = env->GetArrayLength(imageAddressArray);
    jlong *imgAddressArr = env->GetLongArrayElements(imageAddressArray,0);
    vector<Mat> imgVec;
    for(int k=0; k<a_len; k++)
    {
        Mat &curimage = *(Mat*)imgAddressArr[k];
        Mat newimage;
        curimage.copyTo(newimage);
        float scale = 500.0f / curimage.rows;
        resize(newimage, newimage, Size((int)(scale*curimage.cols), (int)(scale*curimage.rows)));
      //("Image height %d width %d", newimage.rows, newimage.cols);
        imgVec.push_back(newimage);
    }
    Mat &result = *(Mat*)outputAddress;

     Stitcher stitcher = Stitcher::createDefault();
   Stitcher::Status status = stitcher.stitch(imgVec, result);

    //("Result height %d width %d", result.rows, result.cols);

    //cv::imwrite(filepath1, result);

    return status;
}
}
```

**Figure 5.19 : Aperçue du code source en c++**

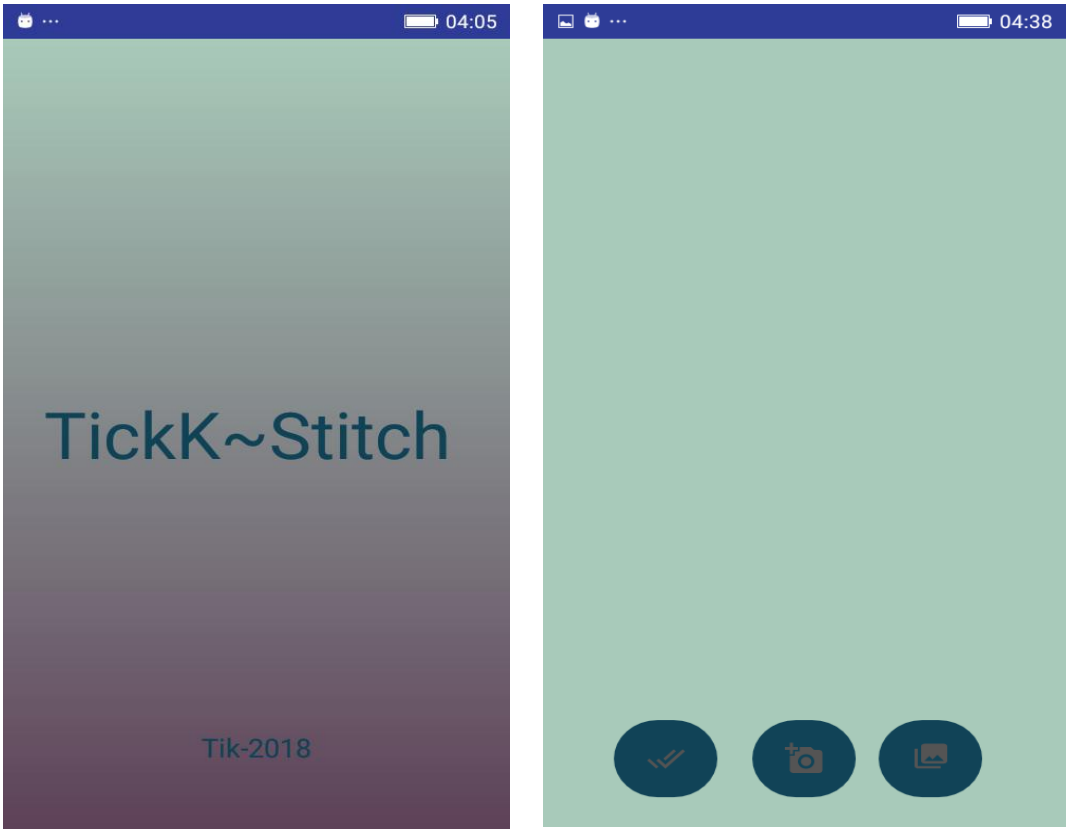



**Figure 5.20 : Accueil de l'application**

Avec le bouton :  on peut sélectionner les images qu'on veut utiliser pour réaliser le panorama ,comme suit :

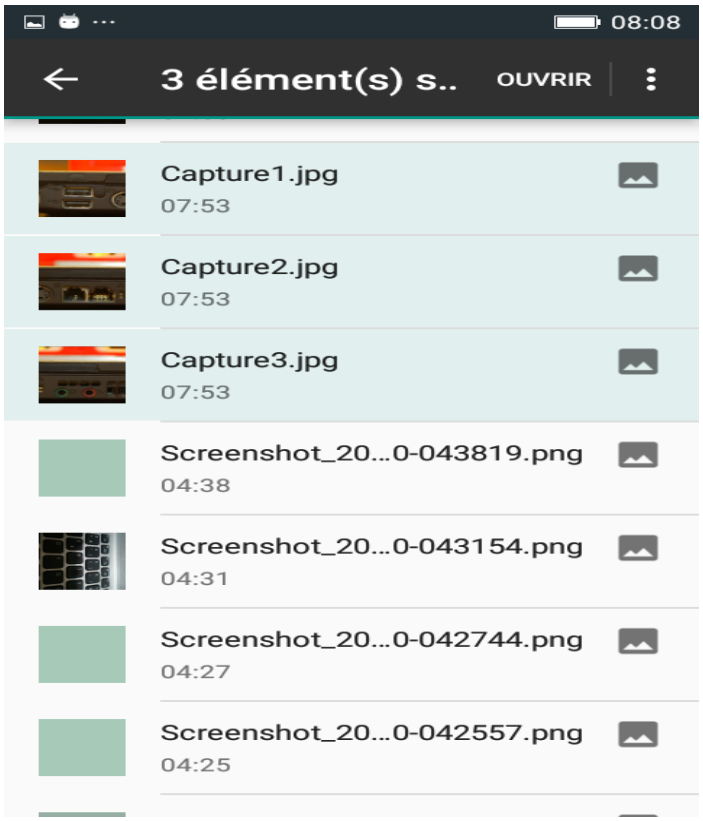



**Figure : 5.21 Aperçue de la sélection d'images avec TickkStitch .**

L'implémentation de cette méthode est assez simple sous java :

```
70          ////////////////////
71          super.onActivityResult(requestCodEe, resultCodEe, datAa);
72          if(requestCodEe == 1) {
73              if(resultCodEe == Activity.RESULT_OK) {
74                  if(datAa.getClipData() != null) {
75                      int count = datAa.getClipData().getItemCount();
76                      for(int i = 0; i < count; i++){
77                          Uri imageUri = datAa.getClipData().getItemAt(i).getUri();}
78
79                  }
80              } else if(datAa.getData() != null) {
81                  String imagePath = datAa.getData().getPath();
82
83              }
84          }
85      }
86      @Override
87      public boolean onOptionsItemSelected(MenuItem item) {
88          int id = item.getItemId();
89
90          if (id == R.id.action_load_image && read_external_storage_granted) {
91              Intent photoPickerIntent = new Intent(Intent.ACTION_PICK);
92              photoPickerIntent.setType("image/*");
93              startActivityForResult(photoPickerIntent, SELECT_PHOTO);
94              return true;
95          } else if(!read_external_storage_granted) {
96              Log.e( tag: "MainActivity",  msg: "ERREUR !");
97              return true;
98          }
        MainActivity
```

**Figure : 5.22 Code java de sélection d'images .**


**Figure : 5.23 Panorama réalisé avec de l'application TickkStitch** .

**5.4 Conclusion :** Dans ce chapitre, on a essayé de présenter les configurations de l'android studio et l'application finale de notre travail. Cette application ***TickkStitch,*** nous offre la possibilité de prendre des photos et réaliser un panorama a partir de ces dernières, ou bien de les choisir , pour les assembler .

# Conclusion générale

Notre travail se situe dans la vision numérique , on a essayé de proposer une application sur les Smartphones pour le traitement d'images. L'implémentation sur un système mobile peut etre un challenge pour certain , à cause de l'environnement de travail d'un coté et d'un autre la nature du travail a effectuer qui est l'implémentation pour un système exploitation mobile .

Pour des développements futurs de notre travail. Une extension de notre travail peut etre faite a fin d'incorporer ce travail sur dans une vidéo de la télésurveillance . Il serait interessant d'accomplir la tache suivante : une fois, construite, notre image mosaïque sera utilisée comme modèle d'arrière-plan pour un ensemble de scènes-vidéo prises à partir d'un Smartphone. Notre système sera également doté d'un algorithme de détection basée sur le caractéristiques SURF pour modéliser les objets d'une scène. Une fois, modéliser et détecter, un objet peut être suivi dans le panorama. Notre système final constituera un système de vidéo surveillance de construction de panorama et de détecter/suivre de tout objet qui entre dans ce panorama

On a rencontré de nombreux problèmes , qu'on a réussi à surmonter . En perceptive on a pu :

- Comprendre l'algorithme de stitching ,
- Découvrir le NKD , Opencv4android ,
- Utiliser les fonctions en c++ dans un langage java et apprendre à envoyer des paramètres entre classe de langage différents .

Enfin , contribuer peut être à un travail futur dans le même cadre et utiliser ce programme de base pour de meilleures projets.

## Références


[1] He, K., Chang, H, and J. Sun. *Rectangling Panoramic Images via Warping*. ACM Transactions on Graphics – SIGGRAPH 2013 Conference. 2013

[2] Ostiak, P. Implementation of HDR panorama stitching algorithm.

[3] Szeliski, R. *Image Alignment and Stitching: A Tutorial.*Microsoft Research. 2004

[4] :Xiong, Y. and K. Pulli. *Fast Panorama Stitching for High-Quality Panoramic Images on Mobile Phones.* IEEE Transactions on Consumer Electronics. 2010.

[5] : M. Brown and D.G. Lowe, "Recognising Panoramas," Proc. of the Ninth IEEE International Conference on Computer Vision, 2003, pp. 1218-1225.

[6] : Jean Philippe Tardif Sebastien "Vision par ordinateur : calibration et homographie""https://www.iro.umontreal.ca/~roys/ift6145H06/calib4.pdf

[7] : L Younes "Comprendre et paramétrer l'algortihme SIFT " www.univ-reims.fr/site/laboratoire-labellis/centre.../32697.pdf »

[8] : G. Gao, K. Jia, A new image mosaics algorithm based on feature points matching, in: International Conference on Innovative Computing, Information and Control, 2007, pp. 471471.

[9] : T. Botterill, S. Mills, R. Green, Real-time aerial image mosaicing, in: International Conference of Image and Vision Computing New Zealand (IVCNZ), 2010, pp. 1–8.

[10] L. Yao, Image mosaic based on SIFT and deformation propagation, in: IEEE International Symposium on Knowledge Acquisition and Modeling Workshop, 2008, pp. 848–851.

[11] : G. Jun-Hui, Z. Jun-Hua, A. Zhen-Zhou, Z. Wei-Wei, L. Hui-Min, An approach for X-ray image mosaicing based on Speeded-up robust features, in: International Conference on Wavelet Active Media Technology and Information Processing (ICWAMTIP), 2012, pp. 432–435.

[12] : Brown, M. and D. Lowe. *Automatic Panoramic Image Stitching using Invariant Features*. International Journal of Computer Vision. 2007.

[13] : Bordallo M. « *Panorama Imaging for Mobile Phones* ». University of Oulu, Department of Electrical and Information Engineering, 2010.

[14] Salil Kapur , Nizhar Thakar "Mastering Opencv For Android"

[15] :P.Froicois,"Etude sur les besoins de compétence dans le développement d'applications mobiles", Octobre 2013.

[ 16] :Damien G, Julien C , Emmanuel R : Programmation Android De la conception au déploiement. EYROLLE Edition 2010.

[17]: Wei-Meng Lee : Android Application development cookbook: 93 Recipes for Building Winning Apps, John Wiley & Sons Inc. Edition 2013.

[18] : Reto Meter : Professional Android for Application Development, John Wiley & Sons Inc. Edition 2013.

# Résumé

Les applications mobiles sont devenues un moyen de création de services. Elles permettent de consulter du contenu, de fournir un service adapté, de jouer et faire des photographies de vacances ou même de prendre directement des photos/vidéo d'un évènement marquant du quotidien. Dans les domaines de la vision par ordinateur et la vidéo-surveillance, beaucoup de recherches et d'avancées technologiques industrielles ont été réalisées ces dernières années. On est désormais capable à travers un Smartphone de vérifier l'état de la maison, de sécuriser les lieux à distance, de détecter automatiquement une intrusion, de contrôler un outil ménager, de reconnaitre, localiser et suivre le déplacement d'un objet dans une scène panoramique directement sur Smartphone, et tant d'autres applications innovantes faisant combiner diverses approches liées à la détection/reconnaissance d'objets, la vidéosurveillance et le développement d'applications sur mobiles.

Dans ce mémoire, nous nous intéressons à la construction de panorama sur Smartphone utilisant Androïd. Par ce travail, nous présentons une approche utile à la formation d'image panoramique à partir de plusieurs images prises séparément ou.

Ainsi, nous proposons une application pour créer des panoramas. La construction de ce modèle de fond se base sur plusieurs algorithmes dont principalement le stitching (assemblage) et la fusion (mixtion). Notre système ainsi réalisé a été testé sur un ensemble d'images.

**Mots-clés**

Vision par ordinateur, image panoramique, construction de mosaïque, modèle de fond, Android Studio, émulateur Smartphone.

# Abstract

Mobile apps have become a way to create services. They allow you to consult content, provide an adapted service, play and take holiday photographs or even take photos / videos of a major event. In the fields of computer vision and video surveillance, a lot of research and technological advances have been made in recent years. We are now able through a Smartphone to check the state of the house, to secure the premises remotely, to automatically detect an intrusion, to control a household tool, to recognize, locate and track the movement of an object in a panoramic scene directly on Smartphone, and so many other innovative applications combining various approaches related to the detection / recognition of objects, video surveillance and the development of mobile applications.

In this thesis, we are interested in the construction of panorama on Smartphone using Android. Through this work, we present a useful approach to panoramic image formation from several images taken separately or.

Thus, we propose an application to create panoramas. The construction of this basic model is based on several algorithms including mainly stitching (assembly) and fusion (mixing). Our system thus realized was tested on a set of images.

**Keywords :**

Computer Vision, panoramic image, mosaic construction, background template, android studio, smartphone emulator.