TELECOM Bretagne

RAPPORT FINAL DU PROJET N°50

Titre du projet

"AUTHENTIFICATION D'HOLOGRAMME DE SÉCURITÉ PAR ACQUISITION ET TRAITEMENT D'IMAGES VARIABLES SUR SMARTPHONE"

Equipe projet composée de
ALVAREZ ETCHEVERRY Florencia
NAVARRO Peris Jorge
MARQUEZ Daniel Jonathan
EL OMRANI Younes
KOUASSI Paterne Eudes V.

Relecteurs
Les membres de l'équipe

Destinataires
Les encadrants techniques
Le groupe de pilotage
Le client

Client
L'entreprise «SURYS» représentée par:
MARC Pic

Encadrant technique
Kevin HEGGARTY
Gouenou COATRIEUX

Groupe de pilotage
Sian ROWLAND
Charlotte LANGLAIS

Version : première version
Date : 14/06/2016

Tables des matières :

1. Introduction

Dans le cadre de la formation des élèves ingénieurs en deuxième année, et ceux en Master première année de Télécom Bretagne, il est organisé un projet ingénieur nommé "PROJET S4" en collaboration avec les entreprises partenaires de l'école. C'est dans ce contexte que nous avons été regroupé en groupe de 5 élèves avec, comme projet, le numéro 50 : *"Authentification d'hologrammes de sécurité par acquisition et traitement d'images variables sur smartphone"*. Ce projet est une proposition de l'entreprise SURYS.

En effet, un **hologramme de sécurité** est un élément (micro ou nano-structure diffractante) imprimé tridimensionnel très difficile à falsifier en raison de la complexité et du coût de la technologie utiliséé pour sa fabrication (holographie). Il est employé pour garantir l'authenticité de documents comme des billets de banque, des passeports, des cartes de crédit, des étiquettes de produits de luxe, des œuvres d'art, des billets de spectacle, etc.

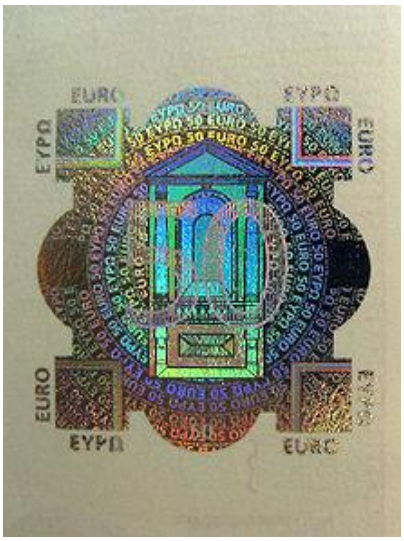


Figure1 : Hologramme de sécurité sur un billet de 50 euros1

2. Contexte et objectif du projet :

2.1. Contexte du projet :

Il faut dire que la protection des documents (billets de banque, papiers d'identité ...) et produits de grande valeur (luxe, médicaments ...) vis-à-vis des contrefaçons est un marché extrêmement important (~$2.5bn/an). L'une des manières les plus efficaces de les protéger est le rajout d'hologrammes de sécurité qui ne peuvent pas être copiés ou imprimés sans un équipement extrêmement sophistiqué et coûteux. Jusqu'à récemment ces hologrammes de sécurité tiraient profit surtout de l'effet visuel produit lors de l'observation par l'œil humain. Par exemple une image ou couleur d'image qui change selon l'angle d'illumination ou d'observation. La disponibilité des capteurs d'images numériques dans les smartphones, et leurs performances qui sont moyennes par rapport aux scanners d'images très sophistiqués,

[1] (2011). Hologramme de sécurité — Wikipédia. Retrieved March 16, 2016, from https://fr.wikipedia.org/wiki/Hologramme_de_s%C3%A9curit%C3%A9.

permettent d'envisager une vérification automatisée des hologrammes de sécurité à l'aide de smartphones.

2.2. Présentation du client :

SURYS est une entreprise à dimension internationale, leader dans son secteur technologique qui développe et commercialise des composants optiques de sécurité, « les hologrammes ». Nous échangeons avec elle, par l'intermédiaire de M. Marc PIC, directeur de la technologie pour l'activité digitale à SURYS.

2.3. Objectifs du projet :

La société SURYS a voulu mettre en place un processus simple et efficace pour l'identification et la reconnaissance des hologrammes qu'elle produit. Ce processus permettra à SURYS de faciliter la vérification de l'authenticité d'un hologramme donné. Par exemple on pourra vérifier à travers ce processus si deux hologrammes identiques à l'œil humain, sont réellement produits par la société SURYS ou s'il s'agit d'une contrefaçon.

L'objectif final du projet est de développer une application téléphone (Android) qui nous permettra d'effectuer cette reconnaissance de l'hologramme conçu par SURYS, ceci à partir de plusieurs images prises de cet hologramme.

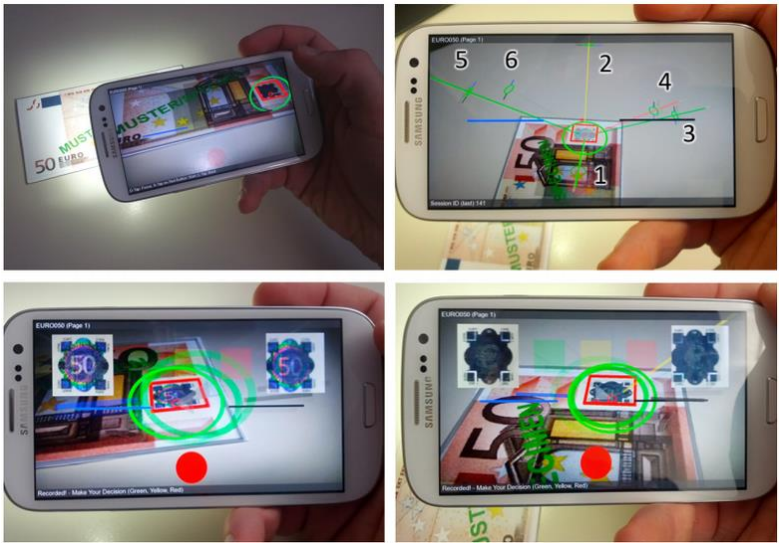


Figure 2 :Prise de photos d'hologrammes avec smartphone2

2.4. Besoins du client et produits du projet :

Le livrable principal de ce projet est l'application Android qui permettra de répondre au besoin du client. En plus de cette application un guide d'utilisation lui sera donné, ainsi que toutes spécifications techniques de l'application en question.

Pendant ce projet, en plus de l'application mobile on a réalisé plusieurs livrables intermédiaires :

[2] Hartl, A. (2013). Mobile Interactive Hologram Verification - Graz University of ... Retrieved from http://data.icg.tugraz.at/~dieter/publications/Schmalstieg_257.pdf.

- **<u>Plan d'expérience pour la prise de photos :</u> Réalisé à l'issue de la tâche 3.1 du lot 3*(Cf. WBS Annexe 1). Il détaille la démarche suivie pour la prise des images de référence des hologrammes.**
- **<u>Base de données des photos de référence :</u> Fichier contenant toutes les photos des hologrammes sur lesquelles on se basera pour effectuer la comparaison pour en déduire l'authenticité, on va les appeler images de référence.**

Voici un exemple d'une image de référence :

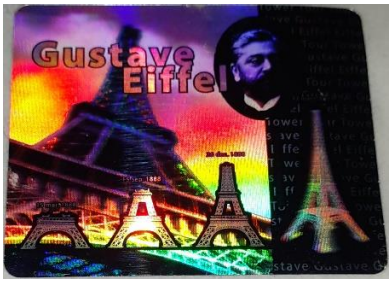


Figure 3 : Image prise à un angle (20,0)

Ces livrables sont donnés au client ainsi qu'aux encadrants techniques et au groupe de pilotage pour leur permettre de suivre le processus adopté pour la réalisation de la solution ainsi que les différents choix qui ont été faits tout au long du projet ainsi que leurs justifications.

2.5. Apport du projet :

Le travail effectué pendant tout ce projet, présente une évolution de la reconnaissance des hologrammes par rapport à ce qui existe au moment présent. Donc présente une valeur ajoutée et un apport de point vue technique et fonctionnelle. Voici les valeurs ajoutées de notre projet:

- Utilisation d'une caméra d'un smartphone :

Selon notre client, jusqu'à présent ils avaient besoins d'un appareil complémentaires sophistiqué d'une grande résolution pour pouvoir réaliser les empreintes des hologrammes avec un smartphone. Ce qui signifie que travailler juste avec l'appareil photo d'un smartphone sera un grand handicap. Puisque la grande résolution donnée par le dispositif additionnel apporte une facilité par rapport à l'analyse des images des hologrammes.

L'identification utilisant seulement la caméra d'un smartphone sera dès lors d'un grand apport fonctionnel.

- Puissance de calcul du processeur et les contraintes du Système d'exploitation mobile

Notre application est destinée à un environnement de capacité très restreinte (comparativement à un ordinateur bureau ou portable). Ce qui nous impose une optimisation dans l'utilisation de la mémoire de travail et de la quantité de calcul, afin d'éviter une monopolisation du processeur ou un plantage complet du système d'exploitation. Cependant les smartphone Android sont de plus en plus puissant en termes de mémoire vive (jusqu'à 2 Go) et en termes de puissance de calcul du processeur (processeur SnapDragon).

Effectuer une identification qui repose sur un traitement d'image et une comparaison de plusieurs images nécessitera une bonne gestion de la mémoire vive du téléphone.

- La facilité d'utilisation de notre application sur smartphone pour un non expert

L'application étant destinée à un large public après mise en production, il faut que nous puissions tenir compte du fait qu'un non expert (dans le domaine du traitement d'images et/ou de la technologie des hologrammes) puisse utiliser aisément cette application sans une quelconque aide extérieure.

2.6. Définitions et acronymes :

Hologramme : Un hologramme est le produit de l'holographie. Il s'agit historiquement d'un procédé de photographie en relief. Aujourd'hui, un hologramme est une image 3D qui est comme " suspendue en l'air ". On appelle aussi, à tort, " hologrammes " les dispositifs que l'on appose sur des cartes bancaires, billets, passeports ou boites de logiciel. Ce sont des dispositifs difficiles à reproduire par des faussaires *Hologram Counterfeiting* qui ne sont pas des hologrammes mais de simples gaufrages. Ils ne contiennent pas d'information 3D. Le relief apparent s'inverse quand on tourne le dispositif de 180° (la tête en bas). Le relief disparaît si l'on tourne de 90°. [1]

3. Conception de la solution finale

3.1. Description de la solution

3.1.1. Fonctionnement de l'application :

La solution développée permet à l'utilisateur de l'application de vérifier l'authenticité d'un hologramme physique qu'il a entre les mains ou étant fixé sur un objet. La procédure à suivre pour la prise de photo n'est pas quelconque, elle doit suivre un scénario que nous avons préalablement défini. Au lancement de l'application Android, appelé ***Holowater,*** l'utilisateur est amené à choisir l'hologramme qu'il veut identifier. Notre application comporte dans sa base de données deux hologrammes produits par la société SURYS. Après avoir choisi le type de d'hologramme à analyser, l'utilisateur lance la procédure et l'appareil photo du téléphone s'affiche à l'écran pour commencer les prises de photos. L'utilisateur est amené à prendre l'hologramme dans 3 positions différentes que nous avons déjà défini.

Après prise de photos l'application effectue le traitement d'images des différentes photos qu'il a prises. Suite à cela l'application affiche le résultat à l'utilisateur : s'il reconnaît l'hologramme analyser ou non (dans le second cas il ne s'agit donc pas du même hologramme que celui de la référence, ou il s'agit juste d'une contrefaçon).

Après prise de photos l'application effectue le traitement d'images des différentes photos qu'il a prises. Suite à cela l'application affiche à l'utilisateur si l'identification de l'hologramme a été faite ou si celle-ci a échoué.

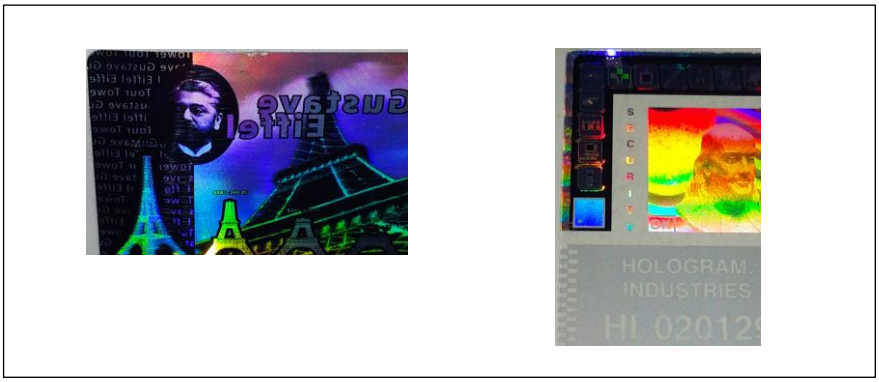


Figure 4 : Les hologrammes inclus dans la base de données de l'application

3.1.2. Spécification technique du traitement :

Quand l'utilisateur appuie sur l'écran pour prendre la photo il est amené à disposer l'appareil suivant les trois positions de références sans quoi la prise de photo ne débute pas. Une fois dans les positions indiquées (voir figure 3), la prise de photo se fait automatiquement. Il faut rappeler que la prise des hologrammes en photo est délicate, ceci est dû à sensibilité des hologrammes, et au changement de couleur aperçue sur la caméra du téléphone avec un mouvement minuscule. Ainsi nous avons 3 photos de l'hologramme à identifier en fin de prise de la procédure de prise de vue.

Les résultats de traitement de chaque position vont être combinés par la suite pour en déduire une décision finale par rapport à l'authenticité de l'hologramme. La décision finale sera prise comme suit : si la vérification de chacune des photos prises séparément est valide (correspondantes aux images de références pour l'hologramme choisi) alors l'authentification est un succès, mais si une seule des trois photos prises n'est pas compatible aux images de références correspondantes à sa position alors on s'arrête là et l'authentification est un échec. La décision finale sera affichée à l'écran du téléphone. Et un nouvel essai sera proposé à l'utilisateur.

3.2. Efficacité de la solution proposée

3.2.1. Eléments déterminant la solution

Pendant le développement de la solution technique plusieurs choix ont été faits. Avant prise de n'importe quelle décision par rapport à ces choix, plusieurs critères ont été mis en place afin de faciliter et justifier nos divers choix.

- **Algorithme de traitement d'images**

Nous avons eu le choix entre plusieurs algorithmes et méthodes pour effectuer nos comparaisons entre les photos prises et les photos de référence. Nous avons utilisé en premier lieu l'erreur quadratique moyenne, et vu les résultats obtenus qui seront détaillés plus tard dans le document, nous sommes passés à d'autres algorithmes plus performants et qui présentent surtout un gain en temps d'exécution.

- **Processus de prise de photo**

Comme l'exécution sera faite sur un téléphone, et étant donné que les performances sur un téléphone sont faibles par rapport à un ordinateur, le temps d'exécution est l'un des critères cruciaux sur lesquels nous nous sommes basés pour déterminer ce processus.

Dans un premier scénario nous avions choisi de définir 5 positions de référence pour la prise de photo. Mais cette solution avait tendance à durer plus qu'il ne fallait (3 minutes) et parfois faisait crasher l'application. En plus nous venions d'opter pour un algorithme de traitement d'image plus robuste que le précédent. Nous avons donc réduire à 3 le nombre de positions de prise de vue. Ce choix a été un compromis entre la précision de l'identification et le temps d'une opération d'authentification de l'hologramme.

- **Nombre de photo de référence**

Un autre compromis fait entre la précision, la robustesse de l'authentification et le temps de traitement est le nombre de photos de référence choisi. Le nombre de positions modifie linéairement le temps de traitement global. Prenons par exemple deux cas :

- Nombre de positions = 3, temps de traitement par photo = 12ms. Cela nous donne 36ms de traitement total.
- Pour un cas identique sauf avec 4 positions de prises on trouvera 42ms de traitement.

Nous avons donc opté pour 3 positions de comparaison avec une photo prise pour chaque position, puisque nous jugeons que l'identification pourra être réalisée avec ce nombre de positions et qu'un ajout d'une position supplémentaire affinerait le résultat et lui donnerait plus de précision, toutefois ceci augmenterait considérablement le temps de traitement global.

3.2.2. Avantages

- **Simplicité d'utilisation**

Notre application apporte une simplicité en ce qui concerne l'identification des hologrammes. Elle ne nécessite pas des prérequis en traitement d'images ou en optique de réfraction pour pouvoir l'utiliser. Elle pourra donc être utilisée par toute personne ayant un hologramme produit par la société SURYS et souhaitant elle-même effectuer l'authentification.

- **Ébauche d'une inclusion de la lecture des tatouages**

Un tatouage est une technique permettant d'ajouter des informations de copyright ou d'autres messages de vérification à un fichier ou signal audio, vidéo, une image ou un autre document numérique. Le message inclus dans le signal hôte, généralement appelé marque ou bien simplement message, est un ensemble de bits, dont le contenu dépend de l'application. La marque peut être le nom ou un identifiant du créateur, du propriétaire, de l'acheteur ou encore une forme de signature décrivant le signal hôte. Le nom de cette technique provient du marquage des documents papier et des billets (selon Wikipédia).

La technique utilisée dans notre application pour effectuer l'identification, et qui sera détaillée par la suite, fait une bonne introduction d'ajout de la fonctionnalité technique qui permettra de lire des tatouages inclus sur l'hologramme physique. En effet à l'issue de l'étape qui précède la comparaison on obtient des hologrammes recadrés et droit en forme de

rectangle. À noter qu'après la prise, dans les différents angles, les hologrammes sont en forme de trapèze à cause de l'inclinaison du téléphone.

Photo originale prise à un angle

Photo recadrée de la photo

Figure 5 : Transformation affine des images des hologrammes

- **Possibilité d'étendre l'identification à d'autres hologrammes**

L'identification jusqu'à maintenant est possible pour deux hologrammes. La démarche suivie étant à une grande partie identique pour les deux hologrammes, sauf quelques petites différences qui seront détaillées plus tard, on peut étendre par la suite l'utilisation de l'application à d'autres hologrammes.

3.2.3. Inconvénients et limitations

- **Nécessité de l'application du processus déjà défini**

Comme cité précédemment l'utilisateur doit suivre un processus prédéfini pour effectuer l'identification de l'hologramme, si ce processus n'est pas respecté, même si l'hologramme appartient bel et bien à la société SURYS, il ne sera pas reconnu par l'application. Donc le respect de scénario et son application correcte est l'un des nécessite pour une bonne identification. Ceci présente un grand défaut d'utilisation, et limite la liberté de l'utilisateur, et le met dans une situation où il doit rester vigilant par rapport au mouvement et l'inclinaison du téléphone pendant la prise.

- **Influence d'une lumière ambiante forte**

L'identification de l'hologramme est réalisée parfaitement dans un endroit obscur. En présence de lumière ambiante l'identification est réalisé aussi mais avec des résultats moins performants que dans le premier cas. Il faut noter que la lumière ambiante présente une complexité supplémentaire par rapport au traitement de l'image, ceci est dû essentiellement à l'incidence inconnue est ingérable de ces rayons, et cette quantité de lumière modifie légèrement les couleurs aperçues sur la photo. La présence d'une lumière ambiante trop forte induit donc une grande probabilité de l'échec de l'identification.

4. Développement de la solution technique :

4.1. Plan d'expérience pour la prise des photos de référence :

La prise des photos de référence a été précédée par un établissement d'un plan d'expérience qui détermine le processus dans lequel ces photos vont être prises. La détermination du plan d'expérience a suivie 3 étapes principales :

1- Prises de photos préliminaires :

Nous avons commencé par prendre plusieurs photos des hologrammes afin de recenser les variantes des photos prises et déterminer les facteurs qui agissent sur l'hologramme. Voici les principaux facteurs et paramètres influençant les photos prises :

- **Lumière ambiante**

La Lumière ambiante est un paramètre très important qui fait que l'allure de la photo change complètement dans un endroit obscur. Dans un endroit illuminé la prise de photo est délicate, et les photos prises dans les mêmes conditions ne donnent pas le même résultat.

- **Inclinaison du téléphone**

L'inclinaison est conditionnée par deux angles. Ces deux angles x et y sont mis à 0 quand le téléphone se met dans une position horizontale. C'est ces deux angles qui définissent la forme de l'hologramme pris en photo. Quand le téléphone est mis en position horizontale la forme de l'hologramme est un rectangle. Sinon la forme tend vers la forme d'un trapèze.

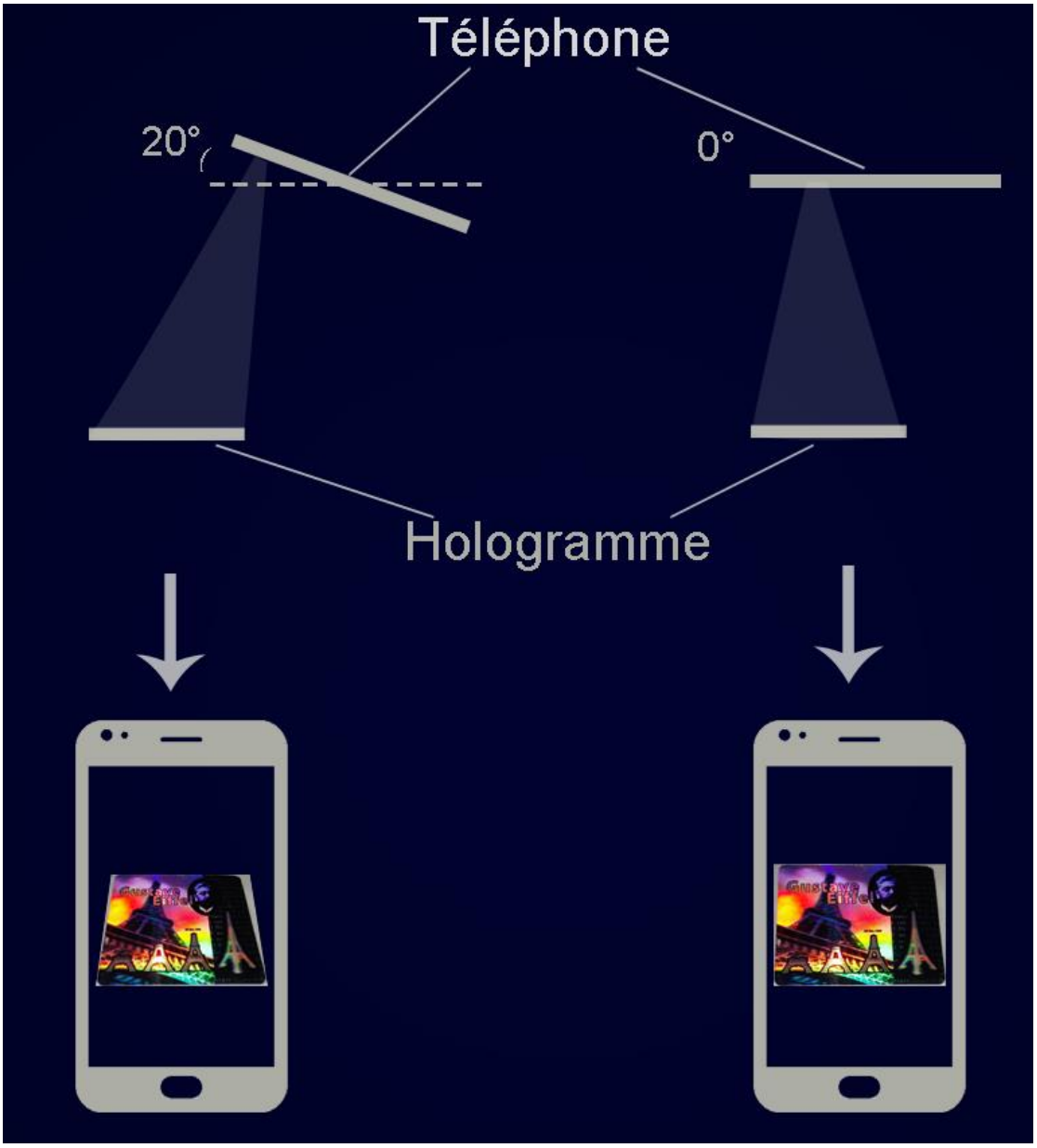


Figure 6 : Inclinaison du téléphone et forme de l'hologramme

Quand on varie ces deux angles les couleurs aperçues sur l'hologramme changent, donc théoriquement la photo prise avec le téléphone change.

- **Distance à partir de l'hologramme**

Cette distance sépare la caméra du téléphone de l'hologramme physique. Cette distance varie proportionnellement avec la taille de l'image de l'hologramme. Pour une meilleure identification de l'hologramme il faut que la taille de l'hologramme pris en photo soit la plus proche de la taille de référence.

- **Flash**

Le flash est considéré comme une source de lumière supplémentaire. Une bonne gestion du flash permet de stabiliser les prises de photo et d'anéantir l'effet de la lumière ambiante qui est considéré comme parasite du moment qu'on n'arrive pas à la gérer.

2- Conclusions tirées des prises :

Dans un deuxième temps, nous avons créé une liste des prises dans laquelle nous avons mentionné les détails de chaque prise, et renseigné les commentaires relatifs à chaque prise.

Dans cette première phase de développement, nous avons utilisé l'erreur quadratique moyenne comme élément de comparaison entre les photos prises. Cet outil effectue la comparaison pixel à pixel des deux photos et somme cette quantité pour tous les pixels de la photo.

Voici les principaux résultats que nous avons tirés de cette première phase :

- **Lumière ambiante**

La lumière ambiante est considérée comme parasite, parce qu'elle apporte une complexité supplémentaire à la prise. Cette lumière est ingérable parce qu'elle est dominante et son angle d'incidence est difficile à déterminer lors de la prise.

On considère dans notre plan d'expérience que pour une bonne prise, il faut éliminer la lumière ambiante au maximum.

- **Flash**

L'utilisation du flash classique dégrade la qualité et la netteté de la photo prise quand le téléphone est en parallèle avec l'hologramme (inclinaison à 0°), c'est-à-dire quand la lumière incidente est verticale, parce qu'il donne une intensité de lumière considérable dans un seul temps, cette quantité rencontre une réflexion sur la surface de l'hologramme et apparaît sur la photo prise.

De ce fait nous allons omettre la prise de photo à la position parallèle et on se basera sur les positions latérales quand le téléphone sera incliné pour effectuer les comparaisons pour en déduire l'authenticité.

Il faut préciser que les angles d'inclinaison combinée avec le flash changent les résultats en fonction de la position de l'hologramme sur l'écran du téléphone.

Figure 7 : Schéma montrant les différentes couleurs apparues en fonction de la position de l'hologramme dans l'écran du téléphone.

Ce schéma montre qu'en fonction de la position de l'hologramme sur l'écran les couleurs de l'hologramme changent. Ceci s'explique par le fait que la lumière aperçue par l'hologramme à chaque fois change et à chaque position l'hologramme dégage les lumières correspondant à cette position.

On déduit que pour avoir une stabilité des prises et une adéquation des couleurs et des images entre plusieurs prises dans la même position l'hologramme doit être situé au milieu de l'écran du téléphone.

NB : Pendant les prises effectuées par l'utilisateur et pour faciliter les prises des images nous dessinerons une zone dans l'écran qui va l'aider à positionner l'hologramme au centre de l'écran.

- **Recadrage**

Le recadrage est l'action qui consiste à avoir deux photos superposées avec des formes identiques et des pixels identiques. Cette action précède le traitement avec l'erreur quadratique moyenne, puisque ce traitement se fait pixel à pixel, d'où la nécessité d'avoir deux photos superposées.

Le recadrage manuel fait avec *Photoshop* dans cette première phase donne des résultats irréguliers, c'est-à-dire pour des expériences identiques, on trouve des résultats différents avec un ordre de grandeur assez élevé. On conclut donc que le recadrage apporte plusieurs défauts à l'image recadrée surtout que la comparaison dans cette première phase a été faite pixel à pixel.

- **L'erreur quadratique moyenne :**

On retrouve les résultats suivants pour les différentes prises :

Type de prise	Ordre de grandeur de l'EQM
Deux photos différentes	10^4
Hologramme dans deux positions différentes	10^3
Hologramme dans la même position avec lumière ambiante avec défaut de recadrage	1000
Hologramme dans la même position sans lumière ambiante avec défaut de recadrage	400
Deux hologrammes dans la même position sans lumière ambiante avec un recadrage correct	10

Vu les résultats imprécis et irréguliers que nous avons trouvé à l'aide de l'erreur quadratique moyenne, nous avons jugé que son utilisation n'est pas efficace, et avons décidé de l'omettre pour le traitement d'images, et somme passé à d'autres outils qui seront détaillés par la suite.

3- Définition du plan d'expérience :

En se basant sur conclusions tirées, nous avons défini le plan d'expérience pour la prise des photos de référence. Les prises seront faites dans les positions suivantes :

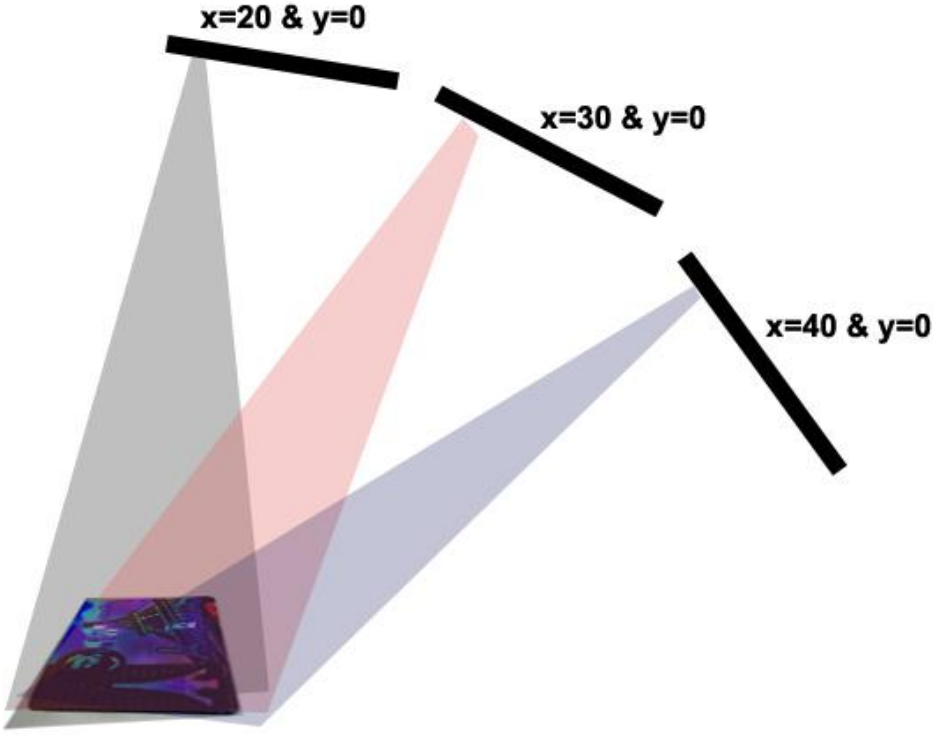


Figure 8 : Les différents angles de prises de photo

Dans chaque prise le téléphone doit être mis exactement dans ces positions. On affichera sur l'écran les angles d'inclinaison du téléphone pour permettre à l'utilisateur d'avoir l'inclinaison requise.

Le processus définit ci-dessous est valable pour les différentes positions :

Processus de prise de photo :

1- On maintient le téléphone fixe dans la position indiqué par le téléphone.
2- Il faut que l'hologramme soit au milieu de l'écran.
3- Après ceci on prend la photo en utilisant une pince en métal pour garder la stabilité du téléphone au moment de la prise.
4- Dans le cas d'un faux mouvement de téléphone on devra refaire cette prise.

4.2. Processus de comparaison :

4.2.1. Introduction au processus de traitement d'images

L'application a été abordée en plusieurs phases : le développement du code du traitement d'images et les tests, tout fait sur ordinateur et sur l'IDE (Environnement de Développement Intégré) *Eclipse*, et la traduction de ce code à *Android*. Le programme a été codé, par conséquent, sur langage *Java*, puisqu'il est le langage sur lequel *Android* est basé. Le programme de traitement d'images a deux objectifs :

1. Trouver un objet déterminé (Feature Detection) dans une image appelé « scénario » à partir d'une image de référence de cet objet. Cet objet est l'hologramme à scanner.

2. Une fois trouvé cet objet dans la photo scénario, faire un traitement de couleur entre les deux images qui va déterminer si l'image de référence et l'objet trouvé dans le scénario ont la même distribution de couleurs dans les mêmes endroits, et ainsi déterminer si sont les mêmes objets.

Pour cela on a utilisé la librairie libre *OpenCV,* qui a centaines de fonctions pour le traitement d'images sur les langages de programmation C++ (native), Python et Java, et va nous permettre le développement de notre application.

4.2.2. Comparaison de forme des hologrammes

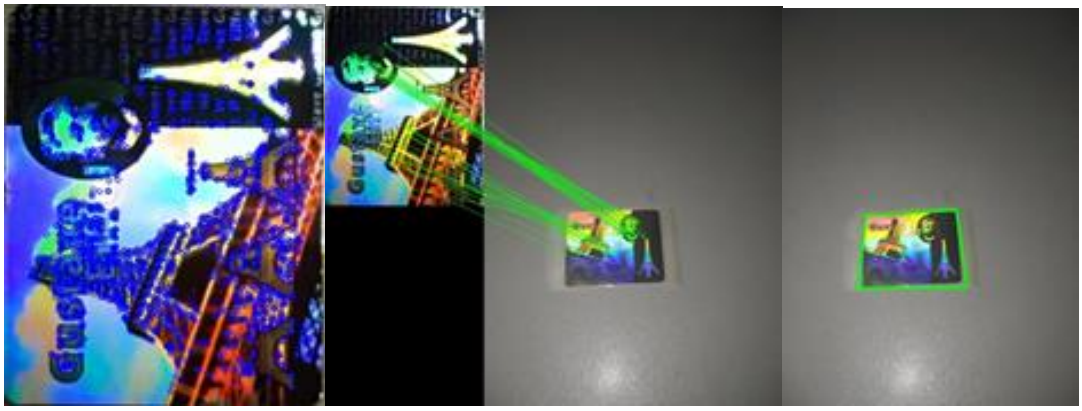
D'abord, l'algorithme utilisé pour le *Feature Detection* était *SURF* (Speeded-Up Robust Features), mais lors de la traduction à *Android*, on a trouvé qu'il est patenté et n'est pas utilisable dans notre application. L'alternative à *SURF* trouvée est l'algorithme *BRISK*, qui fonctionne de manière similaire à *SURF*. Le processus de fonctionnement de *BRISK* est le suivant :

1. Trouver les points clés d'une image, autrement dit « *scale-invariant points* », qui sont des points qui ne sont pas sensibles aux changements de l'écaillage, mouvement et rotation, et partiellement invariant aux changements d'illumination. Pour cette raison, d'abord il fait le *Scale-space keypoint detection*, où il trouve les points d'intérêt d'une image à partir des

couches d'octave de la pyramide d'images[3] en appliquant des filtres de lissage et le redimensionnement des images. Ce calcul est fait après d'une transformation des images de couleur (RGB) à échelle de grises. [4]

2. Grâce au point dernier, on trouve les points clés d'une image, mais cela concerne seulement à leur position dans l'image scénario, donc il faut savoir la différence entre les caractéristiques des keypoints d'une image et l'autre. Faire cette différence est nécessaire parce qu'on pourra seulement trouver l'objet si on compare deux images avec le même objet dans la même position (mêmes pixels), sans rotations et différent angle. Pour cette raison c'est qu'on doit faire le « Keypoint Description », qui va comparer les points clés. Il synthétise, en format vectoriel (de longueur constante) caractéristiques sur les points clés comme par exemple leur intensité dans la direction de leur orientation la plus prononcée. Alors, grâce au Keypoint Description, la recherche d'un objet dans une image scénario est indépendante de sa position dans le scénario, les changements de taille et rotations, donc il le rend plus robuste aux transformations.

Les images ci-dessous démontrent le fonctionnement de l'algorithme *BRISK* sur une image prise avec le téléphone portable d'un Smart Phone :



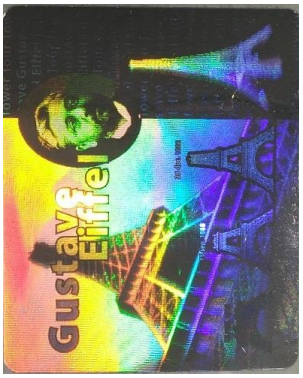
a) Points clés image de référence (en bleu) *b) Keypoint matching* *c) Hologramme trouvé*

4.2.3. Comparaison des couleurs des hologrammes

Une fois que l'hologramme est trouvé dans l'image scénario, on doit procéder à faire un traitement de son couleur. Ce traitement commence par la transformation affine de l'hologramme trouvé dans le scénario, de manière qu'on peut l'extraire et avoir une image complète de l'hologramme.

[3] *La pyramide d'images s'agit d'une succession de lissage et sous-échantillonnage d'une image. Chaque lissage et sous-échantillonnage correspond à une couche d'octave.*

Comme on peut observer dans les images à droite, l'image extraite du scénario est presque la même (en forme) que l'image de référence, donc cela va permettre de pouvoir faire un bon traitement de couleur. La transformation affine nous permet obtenir la rotation (transformation linéal), la translation (addition vectoriel) et les changements d'échelle (transformation linéal) entre les deux images. La librairie *OpenCV* a une fonction *findHomography* qui retourne la transformation affine entre deux ensembles de points (dans notre cas, les points clés des images de référence et scénario), et aussi la fonction *warpPerspective* qui applique la transformation affine calculé avec *findHomography*.Une fois l'image est extraite, de manière qu'on a l'image a), on procède à comparer les couleurs.



a) Image extraite du scénario

Pour cela, on applique une fenêtre glissante qui va parcourir les deux images au même temps, extraire des sous-images correspondantes à la fenêtre et compare les histogrammes de couleur RGB de chaque sous-image. Pour les tests, on a divisé les axes x et y en x/n_1 et y/n_2 fragments, où $n_1 = 4$ et $n_2 = 8$, donc il fait un total de 32 sous-images pour chaque image, donc on aura 64 sous-images en total, dans lesquelles on doit calculer et comparer 3 histogrammes pour chacune (un histogramme pour chaque canal RGB), donc il faut comparer $3\ histogrammes * 64\ sous\text{-}images = 192$ histogrammes par image.

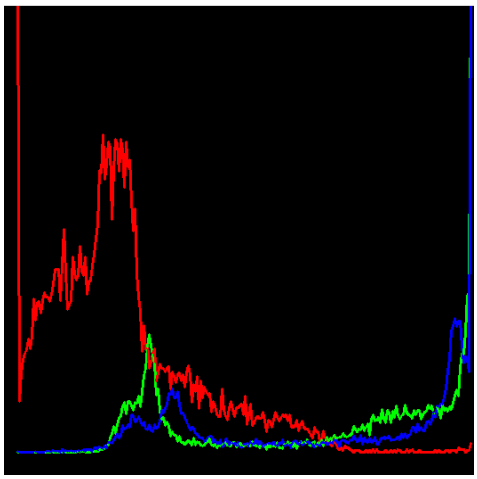
b) Image de référence

Cela peut sembler que le temps de calcul va être très élevé, mais le temps pour trouver seulement l'image (*Feature Detection* et *Keypoint Description*) avec des images scénario avec une taille de 750x1000 pixels et référence avec taille 399x500 pixels est en moyenne de 1025 millisecondes sur un ordinateur, et 1250 ms avec la comparaison de tous histogrammes. C'est un 21,9% du processus du calcul, mais on doit avoir en compte que le nombre de sous-images est très élevé, et si par contre on réduit le nombre de sous-images à $n_1 = n_2 = 4$, le temps de calcul total moyen avec la comparaison d'histogrammes est de 1120 ms, c'est-à-dire, un 9,27% du calcul. Avec le nombre de sous-images on a le compromis entre : avec plus sous-images, plus précision mais plus temps de calcul.

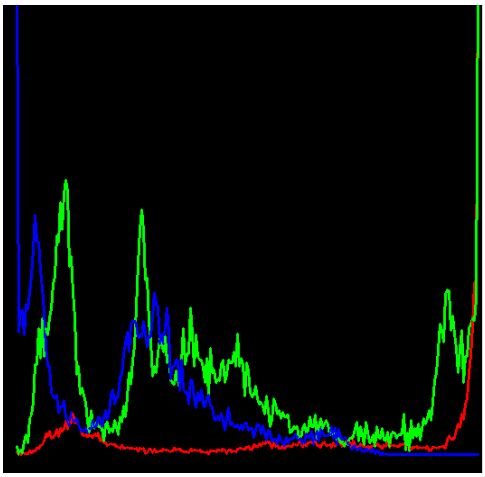
Les images ci-dessous montrent le fonctionnement de la fenêtre glissante avec $n_1 = n_2 = 4$:

a) Fenêtre appliqué sur image référence

b) Fenêtre appliqué sur l'image scénario



c) Histogramme de la sous-image de l'image de référence



d) Histogramme de la sous-image de l'image scénario

Dans les histogrammes, l'axe *X* représente l'intensité du couleur (à gauche l'intensité plus petite et à droite la plus haute), et l'axe *Y* le nombre de pixels avec la même intensité de couleur (en *Y*=0 il y a 0 pixels avec l'intensité *X*). On peut voir que la différence est très grande, donc si on fait la comparaison numérique pour ces sous-images avec *OpenCV*, on obtient :

```
Comparaison Bleus: -0.07433333069568861
Comparaison Verts: 0.44366734286984216
Comparaison Rouges: -0.1264680763325924
```

La méthode utilisée est la corrélation entre eux, et une valeur 1 signifie correspondance totale, et valeurs proches à 0 est différence totale. Ces résultats sont évidents, puisque les images comparées ont les couleurs complètement différents. On a pris comme valeur délimitant 0.6. Donc, si les résultats des comparaisons sont supérieurs à cette valeur pour chaque couleur, les sous-images sont validées, mais si les images ont au moins deux sous-images qui ont 1 histogramme RGB différent, la vérification échoue. Si la comparaison d'hologrammes de deux paires de sous-images a échoué, la vérification de couleur échoue.

Le diagramme suivant montre le fonctionnement de l'algorithme de traitement d'images.

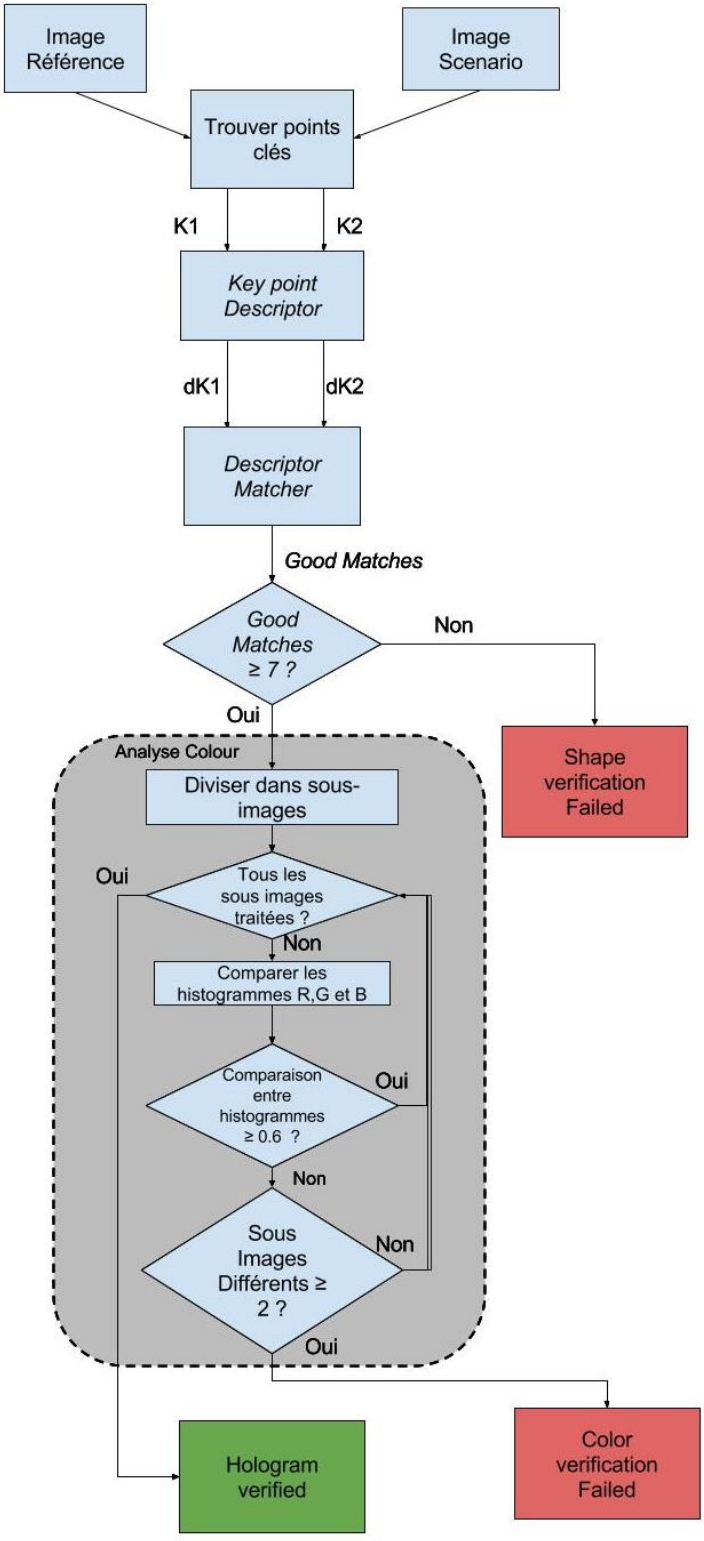
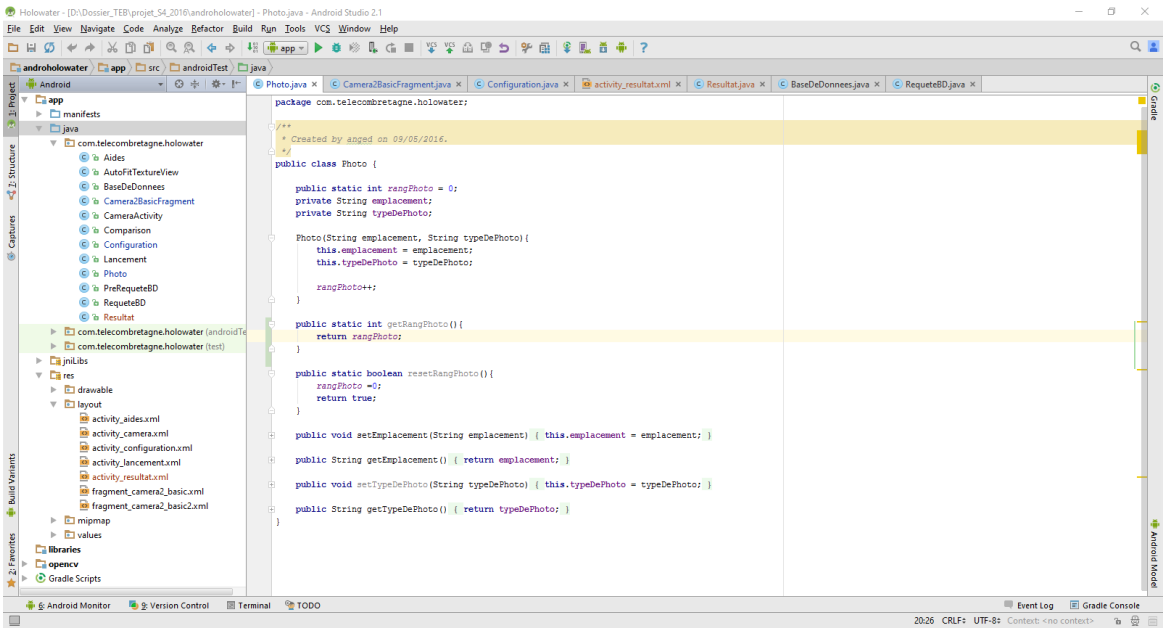


Figure 9 : Processus global de traitement d'image

Où K1 et K2 sont les poins clés de l'image de référence et scénario respectivement, et dK1 et dK2 sont les *Key Points Descriptors*.

4.3. Traduction du code à Android

L'application Android a été développée avec l'éditeur officiel *Android Studio*.

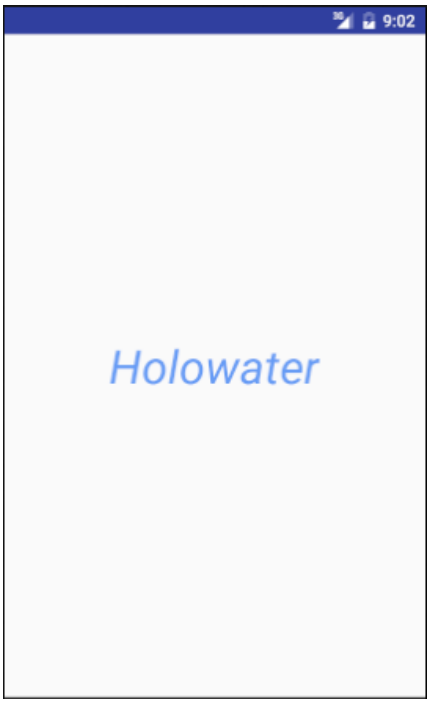


Pour pouvoir intégrer le code de traitement d'image, nous avons intégré la librairie OpenCV dans sa version 3.1 pour Android. Le code de l'application se divise en deux grandes parties: une partie XML (pour les interfaces utilisateurs) et une partie JAVA (pour les traitements).

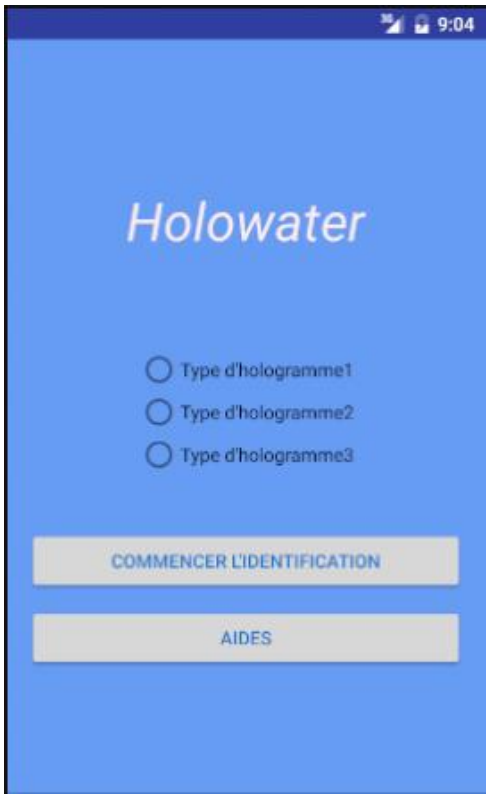
4.3.1. Le XML

L'application dispose de 5 interfaces utilisateurs toutes codées en XML.

1. L'interface de lancement : qui est la première à être lancée et comportant le logo de l'application.



2. Le menu de configuration : il s'agit de la seconde interface à laquelle on arrive directement après l'interface de lancement.



Le bouton « AIDES » permet d'accéder à l'interface d'aide qui indique comment utiliser l'application (cette interface ne sera pas présentée ici).

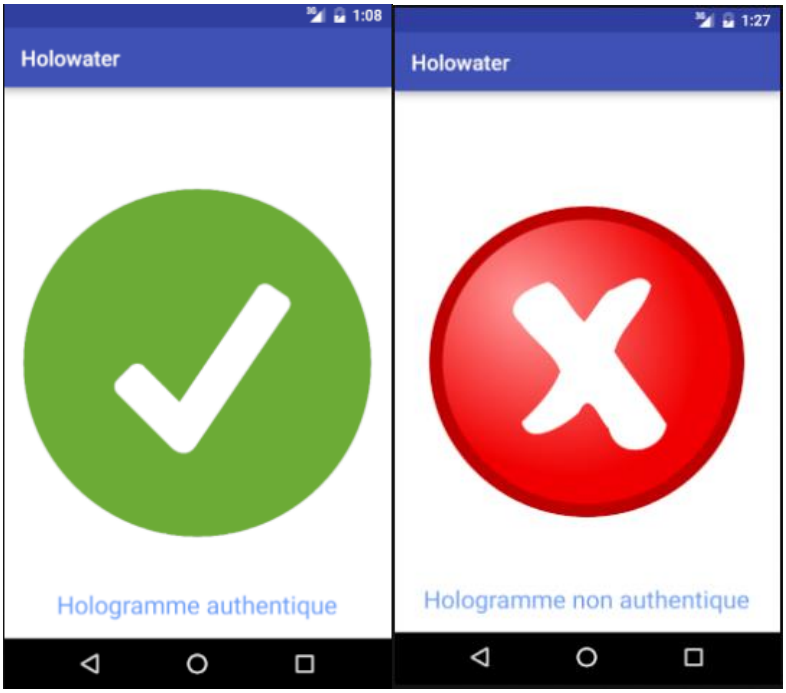
Quant au bouton « COMMENCER L'IDENTIFICATION » il permet d'accéder à l'interface d'authentification de l'hologramme en possession de l'utilisateur. Pour ce faire l'utilisateur doit au préalable choisir le type d'hologramme qu'il veut identifier parmi la liste proposée au-dessus des boutons.

3. L'interface de prise de vue pour l'authentification de l'hologramme : Elle se présente comme suit.

Sur cette interface on à deux boutons. Le bouton « COMMENCER » permet de lancer les prises de vue de l'hologramme. Le second bouton présenté sous forme de point d'interrogation permet d'accéder à l'aide.

Après que la prise de vue vient le traitement qui fait apparaître l'interface du résultats qui permet de savoir si l'hologramme à identifier est authentique ou non.

4. L'interface des résultats : elle n'a que deux présentations suivant l'authenticité de l'hologramme en question. Ce point inclut deux interfaces d'approbation et de désapprobation ce qui nous fait 5 interfaces en totale.



NB : Lorsque l'utilisateur quitte l'interface de résultat, il est ramené à l'interface de configuration. Où il peut soit sortir de l'application soit reconfigurer l'application pour un nouveau test.

4.3.2. Configuration JAVA

Le code JAVA est constitué de 12 classes dont 8 principales :

✓ La classe **« BaseDeDonnees »** : Elle contient le code de création de la base de données, utilisée pour stocker les photos prises lors des prises de vues. A chaque nouvelle analyse, la base de données est vidée de son contenu.

✓ Les classes « PreRequeteBD » et « RequeteBD » : Elles permettent d'inplémenter des méthodes d'accès à la base de données en utilisant notamment le Data Biding avec la classes « Photo ».

✓ La classe « Photo » : C'est cet objet qui permet de lier la photo physique prise et sauvegarder sur le téléphone, aux données relatives à la photo et sauvgardées dans la base de données (Data Biding). Elle a comme attriubuts l'emplacement (chemin local) de la photo sur le téléphone, et la position dans laquelle a été prise la photo (référence par rapport aux différentes positions de prise de vue).

✓ La classe « Comparison » qui contient le code de traitement d'image, à travers sa méthode publique «comparisonBloc(…)» et sa méthode privée « Comparison(…) ». Elle importe des packages provenant de la librairie OpenCV 3.1.

✓ C'est la méthode «comparisonBloc(…)» qui est appelée pour faire le traitement d'image sur les photos prises lors de la prise de vues. Elle renvoie un booléen qui permet de savoir après traitement si l'hologramme à analyser est authentique ou non.

✓ La classe « CameraActivity » : elle hérite de la classe native « Activity » et est liée à l'interface de configuration. C'est elle qui gère tous les traitements de cette interface (interaction avec l'utilisateur).

✓ La classe « Camera2BasicFragment » : C'est un fragment. Elle hérite de la classe native « Fragment ». Elle est utilisée à partir de la classe « CameraActivity ». Elle gère l'interface de prise de vue et gère aussi tous les traitements inhérents à l'authentification de l'hologramme. C'est elle qui appelle la méthode « comparisonBloc (…)» et renvoie le résultat à la classe « Resultat ».

✓ La classe « Resultat » : Elle hérite de la classe native « Activity ». Elle gère l'interface de présentation du résultat de la vérification. Elle permet de modifier dynamiquement le résultat de l'interface suivant que l'hologramme soit authentique ou non (comme présenté précédement).

NB : Le rôle de cette classe (et/ou son interface liée) est discutable, d'autant plus qu'elle ne sert qu'à présenter le résultat du traitement. Cela pourrait être fait directement depuis la classe **« Camera2BasicFragment »**. Ici, elle est utilisée pour les besoins de démonstration du fonctionnement de l'application.

5. Résultats obtenus :

À cause des problèmes eus avec le flash de l'application et la gestion du stockage des images prises par la caméra du Smart Phone, il n'a pas été possible d'avoir des comparaisons à temps réel avec l'application et avec les bonnes images par rapport à celles de référence (même angle et même intensité de lumière).

La résolution des images de référence sera changée, parce que quand une taille de l'ordre de 3120x4160 pixels est choisi, une taille très élevé, le traitement de forme et couleur d'une image avec son image de référence (on rappelle que l'application doit analyser 3 photos scénario avec 3 images de référence) avec un portable BQ Aquaris M5 prend en moyenne 9.5 secondes par photo, donc il fait un total de 28.5 secondes. Ce temps de calcul est très élevé dû à que la résolution de l'image scénario est très élevé (résolution 4k). Ainsi que ce temps est inacceptable pour le traitement et doit être diminué pour ne pas encombrer l'utilisateur avec l'attente du résultat.

Comme on a indiqué dans le point 4.2.3, la comparaison d'histogrammes donne un résultat très petit quand les couleurs de l'image de référence et le scénario, donc quand on a utilisé photos scénarios avec hologrammes très similaires à la photo de référence, les comparaisons son de l'ordre de 0.90. Avec l'ordinateur on a peut faire les calculs et ils ont réussi lors de vérifier si les hologrammes sont très similaires.

6. Problèmes rencontrés :

6.1. Non gratuité de les algorithmes de traitement d'images

Lors de la première de phase du développement qu'on a réalisé avec le langage JAVA sur ordinateur, l'utilisation de la librairie OPENCV n'a pas posé problème, et les tests ont parfaitement fonctionné sur ordinateur. Lors du passage au fonctionnement sur téléphone, avec la traduction de l'algorithme écrit sur ordinateur à Android, l'utilisation de SURF a posé problèmes, au niveau des droits. SURF est un algorithme patenté qui n'est pas utilisable pour fins économiques.

Nous devions chercher une alternative à ce problème, étant donné que notre traitement cœur se faire avec des fonctions de cette librairie, deux solutions s'offraient à nous :

- Achat des droits : Cette solution a été envisageable vu le budget donné par l'école pour la réalisation du projet. Mais vu la procédure administrative qu'on devait exécuter pour se procurer une version payante la libraire SURF, et vu le volume horaire qui nous a resté pour finaliser le projet, cette solution s'est avérée infaisable.
- Solution proposé par le client : Utiliser la version Chercheur de SURF. La version chercheur est proposé aux personne voulant effectuer des recherches en utilisant cette librairie.
- Trouver un équivalent de cette librairie sur Android : C'est cette solution qui a été adoptée. Nous avons trouvé la méthode BRISK, qui nous a permis d'effectuer le même traitement fait avec SURF mais sur Android.

6.2. Deux lots de développement technique :

Sur le Vade-mecum de notre projet est marqué sur le but de ce projet est de développer une application téléphone qui sera capable de lire un tatouage imprimé sur un hologramme. Ce but final était divisé en deux grands lots de développement (Cf. WBS Annexe 1). Le premier lot est la réalisation de l'authentification des hologrammes avec un téléphone. Le deuxième lot est l'écriture des tatouages sur les hologrammes et leur lecture avec l'application

développée. Vu le volume horaire du projet, les encadrants technique nous ont proposé de mettre le deuxième lot de développement comme optionnel.

Nous avons donné que 30h pour ce lot dans notre WBS, contre 400h pour le premier lot. De ce fait nous n'avons pas su comment répartir le travail entre les éléments du groupe pour pouvoir réaliser la première partie à temps et commencer le deuxième lot du développement. Nous étions très confus par rapport à ce sujet pendant les premières semaines du projet, et cela a même affecté la clarté et la netteté des livrables qu'on nous avons écrit, puisque le livrable principal du projet était inconnu, et nous n'avons pas pu décider ou prédire si nous pouvions avoir des résultats concrets dans la 2éme partie.

7. Améliorations potentielles de l'application

7.1. Utilisation d'autres hologrammes :

Dans la version actuelle de l'application, nous avons prédéfini les hologrammes qu'on peut authentifier. Cette prédétermination comme détaillée plus tôt, consiste à inclure les photos de référence de cet hologramme dans la base de données, et d'adapter le traitement d'image selon ses spécificités. L'application ne permet pas d'authentifier des hologrammes différents de ceux qui sont déjà prédéfinis.

Ce qu'on peut améliorer dans l'application, c'est inclure une fonctionnalité qui permet d'ajouter de nouveaux hologrammes à l'application. Donc ceci implique une modification du processus de traitement pour ajouter cette fonctionnalité. Afin de réaliser ceci, plusieurs possibilités d'extension s'offrent à nous :

- L'unification du traitement pour tous les hologrammes, et dans ce cas un ajout d'un hologramme consistera à ajouter ses images de référence à la base de données seulement.
- Procéder par un modèle de plugin qui contiendra les images de références de l'hologramme et son modèle de traitement, ce plugin qui devra être ajouté à l'application nous permettra d'authentifier ce nouvel hologramme.

Il faut que noter que pour ajouter des images de référence d'un nouvel hologramme, il faut prendre ces images en respectant le plan d'expérience détaillé dans la partie 3.3.1.

7.2. Traitement sous 3 angles différents :

Comme indiqué auparavant, la comparaison sera faite dans 3 positions qui sont: x=20° et y =0°, x=30° et y=0°, x=40° et y=0°, ces positions suivent un arc d'angle y nul tout au long de la prise. Cette démarche simplifie en quelques sortes le traitement des photos, puisque on arrive à gérer l'incidence du flash vers l'hologramme, par conséquent gérer les couleurs qu'on aperçoit sur la photo. Cette procédure impose à l'utilisateur de respecter les angles de prises sous peine de l'échec de l'authentification de l'hologramme.

L'inclinaison du téléphone est déterminée par 3 angles (x,y,z). Dans notre démarche on ne varie qu'un seul angle. La complexité du problème augmente quand plusieurs angles sont pris en compte, dès lors d'autres méthodes de traitement d'images s'imposent pour résoudre ce problème.

L'amélioration éventuelle possible pour l'application est la prise en compte des 3 différents angles, donc ceci implique une simplification de l'utilisation de l'application, car l'utilisateur ne sera pas amené à respecter les angles de prises prédéfini, et l'authentification sera réalisée à n'importe quelle position du téléphone. En d'autres termes la contrainte d'angle sera annulée, et c'est à l'application de s'adapter à la position du téléphone pour effectuer le traitement adéquat.

7.3. Etendre à l'utilisation des tatouages :

Notre solution actuelle réalise l'authentification des hologrammes, elle indique s'il s'agit des hologrammes produit l'entreprise SYRUS, ou s'il s'agit d'une falsification. Comme indiqué dans la partie 2.2.2 notre solution actuelle pourra faire l'ébauche du développement de l'application permettant la lecture des tatouages. En effet les tatouages sont des éléments qu'on ajoute sur la photo, et qui sont invisibles à l'oeil humain, donc l'amélioration apportée permettra dans un premier temps, d'indiquer une méthode d'inclusion des tatouages sur les hologrammes. Les travaux effectués sur les tatouages permettent l'écriture des images normales, donc une écriture d'un tatouage sur un hologramme nécessite une grande préalable de recherche. Dans un deuxième temps l'application développée doit être capable d'authentifier l'hologramme et extraire le tatouage qui a été écrit dessus.

Cette partie constituait le 2ème lot du développement technique qu'on devait réaliser, mais faute de temps nous n'avons pas pu le débuter.

7.4. Autre algorithme de comparaison des couleurs

Nous avons utilisé jusqu'à maintenant la corrélation entre les histogrammes de la photo de l'hologramme est sa photo de référence pour déduire si couleurs sont identiques dans les deux photos. Cette méthode repose sur un calcul mathématique pur. L'amélioration éventuelle à cet algorithme est le remplacer par un processus de comparaison des couleurs présent dans la bibliothèque OPENCV, pour pouvoir comparer les performances et choisir le meilleur des deux algorithmes. Faute de temps nous n'avons pas pu implémenter ce processus de comparaison de couleur, et sommes contenté de l'algorithme déjà implémenté fonctionne correctement.

8. Organisation générale du travail :

Le travail durant le projet s'est divisé en deux axes essentiels. Le premier axe est la gestion du projet qui consiste principalement en la rédaction des travaux documentaires concernant la gestion du projet, le suivi des différents lots de développement ainsi qu'à la participation des séances de gestion de projet organisés par les groupes de pilotage. Les principaux responsables de cette partie du projet sont Younes El OMRANI et Daniel MARQUEZ. Le deuxième axe est le développement technique qui a porté sur le développement de l'application Android, cette partie a comme responsables Florencia ECHEVERRY, Jorge PERIS et Paterne KOUASSI.

8.1. Travaux documentaires

La partie documentaire se divise en deux sous-parties : la rédaction des comptes rendus, et la mise en forme des livrables.

8.1.1. Organisation de la rédaction

Le responsable de la documentation informe le reste du groupe des échéances liées au livrables, ceci permet d'anticiper la rédaction de ces divers livrables. Ces documents sont en général rédigés par la totalité du groupe. La révision et la relecture est dès lors confiée à deux membres du groupe. À l'issue de cette étape, le livrable est donné au responsable de la documentation qui se charge d'acheminer le livrable aux destinataires. Dans le cas où le livrable est un document numérique (rapport d'avancement, cahier de charge, plan de management, rapport final…), il le dépose sur le BSCW à l'emplacement approprié. Un mail est ensuite envoyé aux destinataires (encadrant technique et à l'ensemble du groupe de gestion du projet, et/ou au client) afin de les mettre au courant de ce dépôt. Pour le code source final du projet, on le livrera au client par le moyen qui nous sera proposé par ce dernier.

8.1.2. Livrables récurrents

- **Comptes rendus :**

Les comptes rendus font preuves écrites des réunions entre l'ensemble du groupe, entre le groupe et le client, et entre le groupe et les encadrant techniques. Généralement les comptes rendus sont rédigés après les réunions, pour garder une meilleure trace du déroulement de la réunion. L'ensemble des comptes rendus est mis dans un drive partagé entre l'ensemble du groupe et accessibles aux encadrant techniques et au groupe de projet. Les comptes rendus sont écrits par le responsable de la documentation (Daniel MARQUEZ).

- **Rapports d'avancements :**

Les rapports d'avancement sont une synthèse des activités de la semaine passée. Ils décrivent le déroulement des taches du projet de chaque semaine et suivent la structure suivante :

● Bilan et prévision: Dans cette partie, nous présentons dans un premier l'avancée des tâches de la semaine qui se termine par rapport aux prévisions (tâches à achever et le temps pour le faire), ainsi que la gestion des risques de la semaine précédente. Dans un second temps, on présente les prévisions pour les tâches de la semaine à venir.

● Planning de Gantt suivi: Dans cette partie, directement liée à la précédente, on y présente le Gantt du projet ainsi que l'avancement des tâches de manière plus graphique. On peut y voir l'évolution effective des tâches comparée au planning initial.

● Tableau de risques: Dans cette partie, nous présentons les différents risques qui planent sur le projet. Un risque se résume en un intitulé, une gravité en code couleur et un responsable.

● Synthèse de travail réalisé: Dans cette partie, se montre le nombre de heures travailles par chaque lot du projet et par chaque membre du groupe.

8.2. Développement technique

La phase de développement a été divisée en deux parties. Le développement de l'algorithme de comparaison sur ordinateur avec le langage Java et la librairie OPENCV, où les responsables de cette partie sont Florencia ALVAREZ et Jorge PERIS, et la traduction de cet algorithme à Android avec le logiciel Android Studio. Cette partie avait pour responsable Paterne KOUASSI. Le travail de développement a été réalisé avec l'outil Redmine, qui est un système de gestion de versions qui permet à plusieurs personnes de travailler sur le même projet en même temps. Cet outil nous a permis d'avancer le travail technique du projet en parallèle et de permettre à l'ensemble des développeurs de coder sur le même document, et garder un rythme d'avancement constant, sans s'obliger à se rencontrer pour réunir les différentes parties du code pour les tester.

Avant de commencer le développement technique, nous avons segmenté tout le travail technique en modules et blocs complémentaires, pour permettre le développement simultané sans création d'incohérence dans le code global.

9. Conclusion

L'objectif principal du projet «Authentification d'hologrammes de sécurité par acquisition et traitement d'images variables sur smartphone» est de permettre à l'entreprise « SURYS » vérifier facilement, avec un outil courant comme un smartphone, si un hologramme est authentique ou pas. On fait face a de multiples contraintes et problèmes lors de la mise en place de la solution, toutefois nous avons pu résoudre ces problèmes et acheminer notre projet pour une solution fonctionnel répondant exactement au besoin attendu par le client. Malgré les compromis réalisés pendant le projet est les hypothèses faites, la solution que nous avons proposée est dotée d'une certaine robustesse et efficacité.

La solution actuelle fera un bon sujet de développement pour l'inclusion des lectures des tatouages sur hologrammes. Cette amélioration peut être accompagnée de l'amélioration de la solution et de l'annulation des hypothèses et compromis de traitement faits.

10. Annexes :

10.1. WBS :