REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
Université Abou Bekr Belkaïd de Tlemcen
Faculté de Technologie

MEMOIRE DE FIN D'ETUDE POUR L'OBTENTION DU DIPLOME DE MASTER ACADEMIQUE

Spécialité : « Automatique »
Option : « Automatique et informatique industrielle »
Préparé au Département de Génie Électrique et Électronique

et présenté par

Mr. Abdelli Imed Eddine

Intitulé du mémoire

Fusion de données inertielles et visuelles

Mémoire dirigé par Dr. Choukri Ben Salah

soutenu le 1 Octobre 2020 devant la commission d'examen composée de :

Mr Yacoubi Boumédiène	Maître assistant class A	Univ.Tlemcen	Président
Mr Boumédiène Benyahya	Professeur	Univ.Tlemcen	Examinateur

Année universitaire 2019 - 2020

Dédicace

Nous dédions ce travail à :

- Nos très chers parents
- Nos frères et Soeurs
- Nos familles
- Nos amis

Abdelli Imed Eddine
Tlemcen,le1 Octobre 2020

Remerciement

Je porte à exprimer ma profonde gratitude à ALLAH AZZA WA JAL pour m'avoir donné le souffle de vie, la force, la santé et l'intelligence nécessaire pour accomplir ce travail.

Je voudrai tout d'abord remercier mon encadreur M. choukri Ben Salah, qui a supervisé ce projet. Il m'a fait bénéficier de ses conseils et de ses compétences, confirmant son intérêt pour mon travail, et également mon co-encadreur M.khaldi abdelwahhab pour leurs conseils judicieux.

Je profite de cette occasion pour adresser mes sincères remerciements à mon frère Abdelli Hamza docteur en automatique qui m'aide et soutien tout la durée de la préparation de la thèse.

Je remercie également tous les membres du jury. Je les remercie du soutien et de l'attention qu'ils m'ont apportée.

Enfin à tous les gens de la promo d'Automatique et Informatique Industriel 2019/2020 de l'université d'Abou bekr belkaid Tlemcen.

Résumé

Dans ce projet on a réalisé un système de navigation qui est basé sur la vision. On a opté pour l'utilisation du flux optique intégré dans l'OpenCV pour le traitement d'images en utilisant une Raspberry Pi et une caméra dans le but de déduire les informations de localisation. Le système développé compare la déformation des points des coins détectés des objets dans l'environnement pour déduire les différents variables de déplacement et rotation. Les essais faits sur le système montrent que les résultats de localisations sont conformes avec une bonne précision avec les mesures faites en réalité.

Mots clés : le flux optique, localisation, Raspberry Pi.

Abstract

In this project we realized a navigation system which is based on the vision. We opted to use the optical flow integrated in the OpenCV for image processing using a Raspberry Pi and a camera and succeed in inferring the location information. The developed system compares the deformation of the detected corner points of objects in the environment to deduce the different displacement and rotation variables. The tests made on the system show that the localization results conform with good precision with the measurements made in reality.

Keywords : optical flow, location, Raspberry Pi.

في هذا المشروع أدركنا نظام الملاحة الذي يقوم على الرؤية. اخترنا لمعالجة الصور باستخدام OpenCV استخدام التدفق البصري المدمج في وكاميرا ونجحنا في استنتاج معلومات الموقع. يقارن النظام Raspberry Pi المطور تشوه نقاط الزاوية المكتشفة للكائنات في البيئة لاستنتاج متغيرات الإزاحة والدوران المختلفة. أظهرت الاختبارات التي أجريت على النظام أن نتائج التوطين تتوافق بدقة جيدة مع القياسات التي تم إجراؤها في الواقع.

العلامات : التدفق البصري، والموقع ,Raspberry Pi

Abréviation

ERS	Système de résolution d'erreurs
U.A.V	Unmanned Aerial Vehicle
SD	Secure Digital
HDMI	High-Definition Multimedia Interface
MEMS	Micro-ElectroMechanical Systems
GPS	système de positionnement global
DMP	Plateforme de gestion de données (Data management platform)
SCL	System Control Language
SDA	Serial Data Line
HMM	Hidden Markov Model

Table des matières

Table des figures

Liste des tableaux

INTRODUCTION GÉNÉRALE

Aujourd'hui, les robots peuvent se déplacer sur des roues ou des jambes, peuvent se déplacer sur tous les terrains et aussi aériens, peuvent voler et effectuer des tâches loin des positions de sauvegarde. Les champs d'applications de ces robots sont nombreux et variés aussi bien terrestres qu'extraterrestres, dans le cas des robots d'exploration spatiale. Grâce aux progrès de l'électronique et de la technologie informatique, le développement des robots a ouvert de nouveaux domaines d'application, tels que les applications de divertissement et d'éducation. L'intégration puissante du traitement numérique et l'émergence des systèmes embarqués annoncent un nouveau type de robots dotés de plus en plus de fonctions qui augmentent leur autonomie et leur capacité à s'adapter à des environnements changeants. L'autonomie d'un robot est sa capacité à détecter les obstacles et les éviter, se localiser dans l'espace 2D/3D, naviguer et cartographier son chemin. L'intérêt de rendre automatiques certaines fonctions comme la localisation dans l'espace 2D/3D est très présent dans le développement de véhicules aériens sans pilote, appelés aussi drones ou U.A.V. (Unmanned Aerial Vehicle).

Les robots aériens sans pilote à quatre rotors à décollage vertical suscitent actuellement un grand intérêt dans le domaine civil. La capacité de décoller et d'atterrir verticalement et de rester en vol stationnaire ouvre de nouvelles perspectives dans plusieurs domaines. Bien qu'ils semblent simples, les drones restent difficiles à piloter dans des conditions stables adaptées à leur mission. Certains travaux en cours apportent des solutions pour simplifier le pilotage des petits drones en réduisant les tâches ardues des pilotes Ces tâches seront traitées à l'intérieur de la boucle de contrôle automatique pour trouver la référence imposée par la tâche. La réussite de la mission nécessite de toujours connaître la position ou la direction du drone et son accélération.

Comme nous le savons tous, la position d'un avion n'est pas directement mesurable, mais est estimée sur la base des valeurs de mesure des capteurs aéroportés (tels que les centrales inertielles, caméras, GPS, télémètres, etc.). Les conditions d'utilisation de ces capteurs varient, et sont généralement liées à l'application et à son environnement.

En fait, en raison de la perte de signal, l'utilisation du GPS dans les zones urbaines, par exemple, est très limitée. Les capteurs inertiels sont très sensibles au bruit, et leurs performances dépendent de leur bande passante et des conditions de l'environnement d'application (interférences électromagnétiques du magnétomètre). Afin de localiser la position et la direction à l'aide d'un accéléromètre et d'un gyroscope, une opération intégrale doit être effectuée, mais l'erreur générée du fait de cette opération rend difficile leur insertion dans la boucle de contrôle. Cependant, un positionnement basé sur l'utilisation d'autres capteurs et la technologie de fusion de données visuelles et inertielles est possible.

Les capteurs de vision sont une alternative aux capteurs inertiels classiques (accéléromètres, magnétomètres) et sont utilisés pour estimer la position face à certaines conditions d'utilisation. Par exemple, dans le cas de l'utilisation d'un magnétomètre dans un champ d'interférence, ou lorsque le système est soumis à une forte accélération et que l'accéléromètre ne mesure pas seulement l'accélération causée par la gravité. Les capteurs visuels sont également très sensibles aux changements de luminosité, et leur utilisation est géné-

ralement limitée par la puissance de traitement informatique. Cependant, les évolutions technologiques et les améliorations de la puissance de calcul des processeurs actuels ont permis d'envisager une utilisation plus large de ces capteurs de vision. La vision est un moyen de mesurer et d'estimer le mouvement, c'est un environnement réel, elle est à la base de l'utilisation quotidienne et continue des humains et des animaux. Il y a longtemps, grâce à l'analyse de la dynamique du champ visuel, les scientifiques ont prouvé que les images capturées contenaient les informations de mouvement du système visuel lui-même. Dans ces informations, l'augmentation ou la diminution du flux optique lors de l'observation d'une scène fixe aide les insectes à mieux contrôler la distance aux obstacles.

Le présent travail est organisé de la manière suivante :

Le premier chapitre est dédié au concept flux optique. Nous présentons la définition du flux optique ainsi que les différentes méthodes de calcul. Parmi ces méthodes, on s'intéresse plus à la méthode Lucas-Kanade dont l'avantage principal est le temps de calcul qui est relativement court.

Le deuxième chapitre est consacré aux capteurs inertiels (le capteur MPU6050). Nous allons introduir une discussion générale sur le capteur, puis, nous allons donner une définition de capteur inertiel, ainsi que leurs différents types, aussi diverses utilisations.

Dans le troisième chapitre, nous présenterons la fusion de données, y compris sa définition, ses fonctions, ses niveaux de fusion, son architecture et les bénéfices attendus. Par la suite, nous allons introduir les approches classiques pour la fusion de données.

Le quatrième chapitre est consacré à l'implémentation expérimentale de ce projet sur la réalité. D'abord nous commencerons par une expérience simple où nous expliquons le flux optique, puis nous passons à la deuxième étape pour mesurer la distance, la rotation, l'accélération, c'est ainsi que nous avons localisé l'objet.

Chapitre I

Caméra et calibrations

I.1 Introduction

La calibration de la caméra est un problème crucial pour la vision par ordinateur où de nombreuses tâches nécessitent le calcul d'informations métriques précises à partir d'images. Calibrer une caméra consiste à déterminer la transformation qui mappe les points 3D d'une certaine scène ou d'un certain objet en leur 2D correspondant projections sur le plan image de la caméra. La précision de la recons-truction 3D sera influencée par la véracité et la fiabilité de la caméra et du projecteur dans le sys-tème. Par conséquent, la distorsion de l'objectif de l'appareil photo et le projecteur doivent être pris en compte lors de l'étalonnage.

Dans un premier axe, nous tenterons de présenter les capteurs caméras de façon générale, les images, et d'avoir une vue particulière sur la calibration de la caméra qui fait l'objectif de ce chapitre.

I.2 Le capteur caméra

I.2.1 Définition

Un capteur caméra est un appareil utilisé principalement dans les appareils photo numériques et les appareils d'imagerie autonomes ou intégrés. En règle générale, lorsque la lumière frappe l'objectif d'une caméra, le capteur caméra capte cette lumière, la convertit en un signal électronique, puis la transmet au processeur de la caméra ou du dispositif d'imagerie, qui transforme le signal électronique en une image numérique. [7]

I.2.2 Types de capteurs caméra

Il existe deux principaux types de capteurs caméra :

1- Un capteur CCD

Le CCD (dispositif à couplage de charge) consiste en une juxtaposition matricielle de capacités MOS. Pour obtenir des images en direct, il faut compter le nombre d'électrons piégés dans chaque puits, qui est proportionnel au nombre de photons incidents sur chaque pixel. Pour cela, on com-mande séquentiellement la tension des grilles de chaque capacité MOS au rythme d'une horloge, ce qui va permettre de transférer les électrons d'une capacité vers sa voisine : Nous parlons de registres à décalage. Après n transferts, les charges sera converti en Tension dans le condensateur puis ampli-fiées. Et puis sont codées numériquement à l'aide d'un convertisseur analogique/numérique à l'extérieur de la matrice CCD. [14]

Il existe plusieurs architectures pour les capteurs CCD qui dépendent directement du type d'applications cible. Un système de transmission par bloc complet (full frame transfer visible en fi-gure 2a) ou partiel (frame transfer en figure 2b) est principalement utilisés dans le milieu scientifique. Le système de transmission inter-lignes (interline transfer en

figure 2c) est utilisé dans les caméras pour un grand public et les systèmes de télévision professionnels.

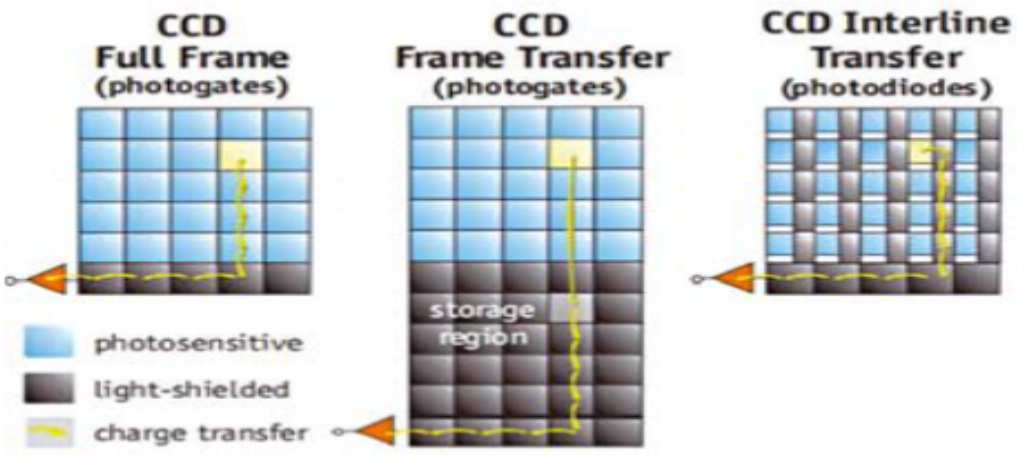


FIGURE I.1 – Schéma des différents types d'architecture de capteurs

2- Un capteur CMOS

La technologie CMOS permet également de transformer la lumière en tension électrique. Le fonc-tionnement est relativement différent et les deux systèmes présente chaque des avantages et des inconvénients.
Le CMOS coute moins cher que le CCD et consomme moins d'énergie.
La gestion des pixels est plus efficace avec le COMS, et les images issues sont plus homogène...

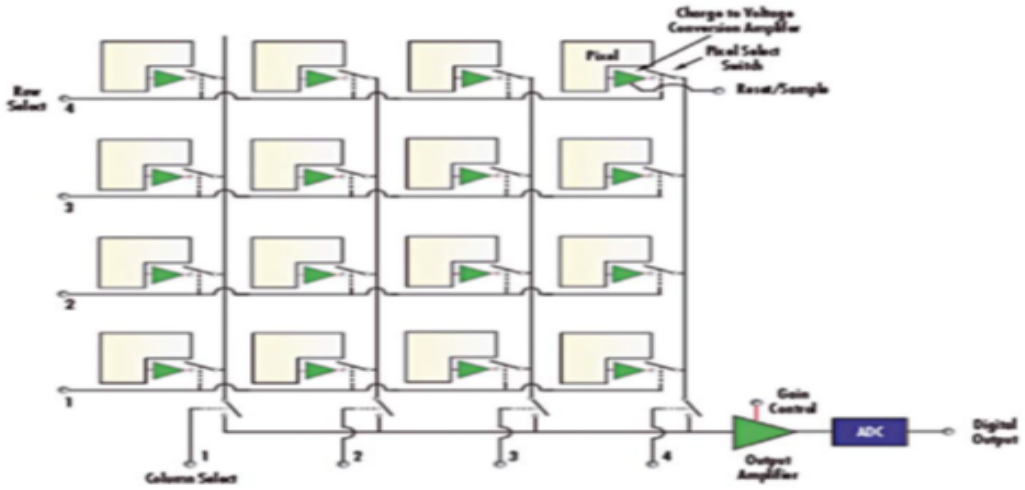


FIGURE I.2 – Schéma de l'architecture d'un capteur CMOS

On pourrait en déduire que le CMOS convient aux applications bas de gamme et

le CCD pour le Broadcast, mais les progrès récent des capteurs CMOS font qu'on les retrouve également dans les dispositifs professionnels.

I.3 Image

L'image est une représentation de quelque chose acquise à l'aide du système de production Images (caméras, caméras, radiographies, scanners, sonar, ...). Elle peut être analogiques (tels que photos, vidéos, etc.) ou numériques (images numérisées suivant divers formats (images compressées ou non...) ou fournis par le capteur Image numérique), dans ce cas, un traitement informatique peut être effectué.

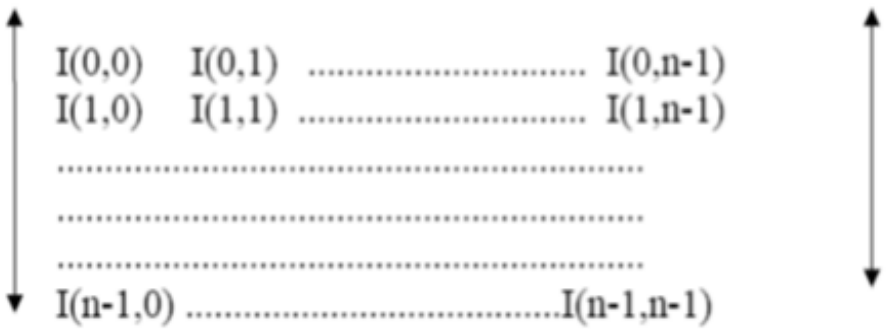
I.3.1 Définition d'une image

Les signaux n-dimensionnels et q-modal sont définis dans l'espace E de dimension $n \in N^*$. c'est Une image avec $n \geq 2$, au point P dans l'espace E, l'image est décrite par le vecteur couleur au point P avec q composante représentant la composante colorimétriques [8]

Par conséquent, l'image peut être considérée comme une fonction spatialement définie $I(X)$ Multi-dimensionnel.
$-X$ est le vecteur de coordonnées définissant la position dans l'espace multidimensionnel (Par exemple, $X = (x1, x2)$en 2D)
$-I(X)$ est la valeur scalaire de l'image unimodale et uu vecteur avec q composante pour les images q-modale. Les images les plus fréquentre sont définies dans l'espace 2D : photo noir et blanc ou cou-leur, radiographie, ... ou 3D : par exemple, une image de tomographie médicale.

I.3.2 Images numériques

Concernant les images numériques, la fonction d'image est échantillonnée pour former des ensembles de points (pixels, éléments PICTure). Généralement, la grille d'échantillonnage est Rec-tangulaire (mais peut aussi être triangulaire ou plus complexe). [24] Donc la représentation matri-cielle suivante :



1- Le pixel Une image numérique est constitué d'un ensemble de points appelés pixels (abréviation de PICture Element) pour former une image. Le pixel représente ainsi le plus petit élément constitutif d'une image numérique. L'ensemble de ces pixels est contenu dans un tableau à deux dimensions consti-tuant l'image :

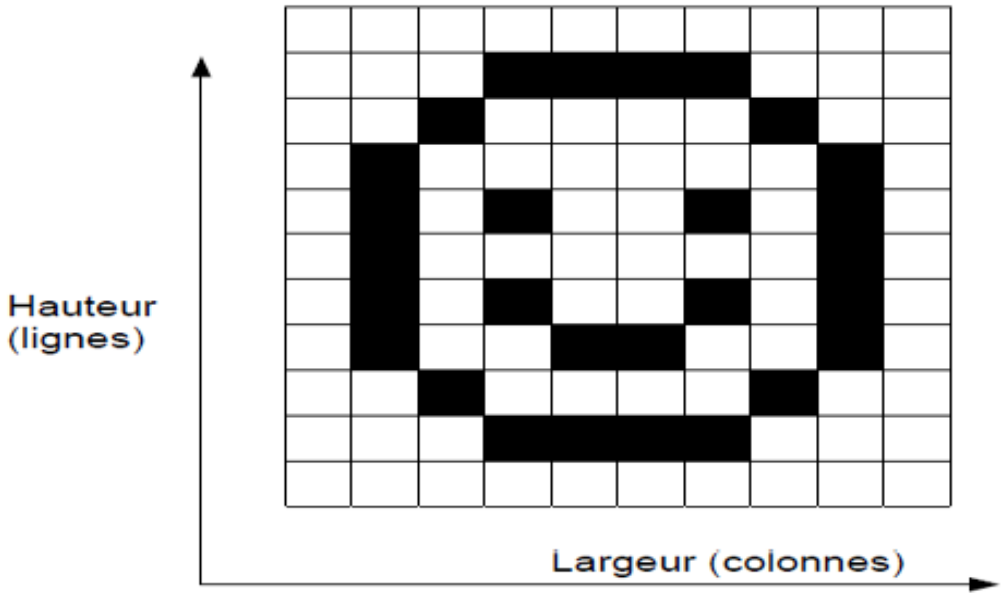


FIGURE I.3 – Ensemble de points pixels

2- Les différents types d'images numériques

Les images noir et blanc Pour les images noir et blanc, on ne considère pas la couleur ici, mais seulement l'intensité Glow (l'exemple classique correspond à une photo en noir et blanc). Dans ces images nous pouvons trouver les images en niveaux de gris ou nous avons un certain niveau de gris, et la plupart des cas, nous avons 256 niveaux de gris, dont :

0 :noir.....127 :gris moyen.....255 :blanc

Ceci est pratique car l'unité d'information est l'octet. Veuillez noter que jute 64 niveaux de gris peu-vent être reconnus par les personnes standard.

Les images couleur Afin de créer des images encore plus riches en couleurs (et donc disposer de plus qu'une palette ¬¬limitée à 256 couleurs), l'idée de mélanger des couleurs primaires en « couches » est arrivée. Il faut savoir qu'il existe deux espaces de représentation des couleurs par mélange, selon qu'on les reproduisent sur un écran d'ordinateur ou sur support papier via une imprimante :

a- L'espace de couleur RGB

Dans le modèle RGB, chaque couleur apparaît dans ses composantes spectrales primaire de rouge, vert et bleu. Le modèle est basé sur le système de coordonnées cartésien. L 'espace couler est un cube dans lequel le rouge, le vert et le bleu occupent trois coins,

le cyan, le magenta et le jaune occu-pent les trois autres coins, le noir est l'origine et le blanc occupe le coin le plus éloigné de l'origine comme présenté dans la figure 4. Dans ce modèle, l'échelle de gris va du noir au blanc Le long de la ligne reliant ces deux points, et chaque couleur est déterminée par un vecteur à trois composants.[26]

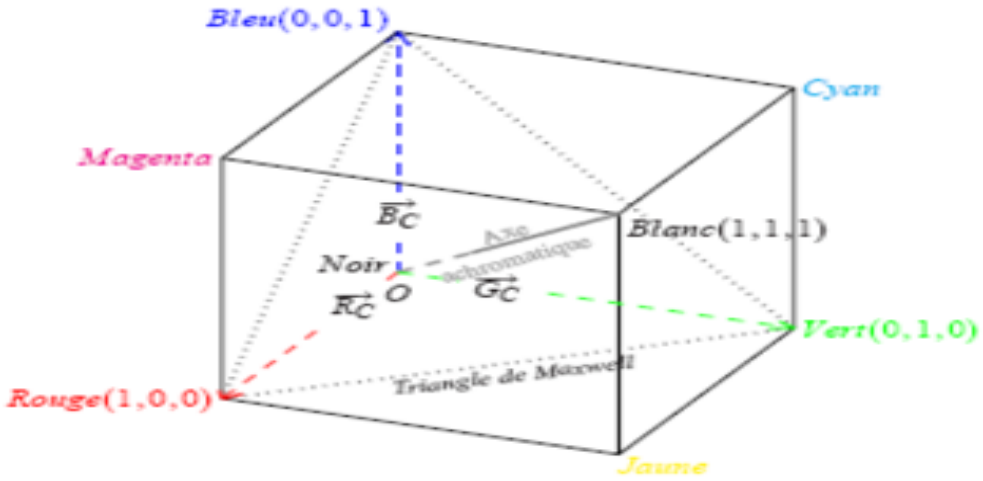


FIGURE I.4 – Cube des Couleurs

b- L'espace de couleur HSV

Ce espace est directement évalué à partir du système de primaire RGB. Le principe de l'es-pace HSV est de caractériser les couleurs de facon plus intuitive, confférmément à la perception naturelle des couleurs, en termes de :
1- teinte (H) : intuitivement, c'est le nom qu'on utilisera pour désigner la couleur, "vert", "mauve", "orange", etc. Idéalement associé à une longueur d'onde, donc à une position sur le cercle de Newton.
2 saturation (S) : c'est le taux de pureté de la couleur, qui doit varier entre la pureté maximale (couleur éclatante) et l'achromatisme (niveau de gris).
3 valeur(V) : c'est la mesure de l'intensité lumineuse de la couleur, qui doit varier entre le noir absolu et le blanc.

Le passage de RGB à HSV se fait par une transformation non linéaire. Plusieurs opérateurs ont été proposés pour la conversion. Voici un exemple [26] :

$$\begin{aligned} V &= max(R,G,B) \\ S &= \frac{V - min(R,G,B)}{V} \end{aligned} \tag{I.1}$$

$$H = \begin{cases} (G-B)/SV & si \quad V = R \\ 2 + (B-R)/SV & si \quad V = G \\ 4 + (R-G)/SV & si \quad V = b \end{cases} \tag{I.2}$$

I.4 Projection Pinhole

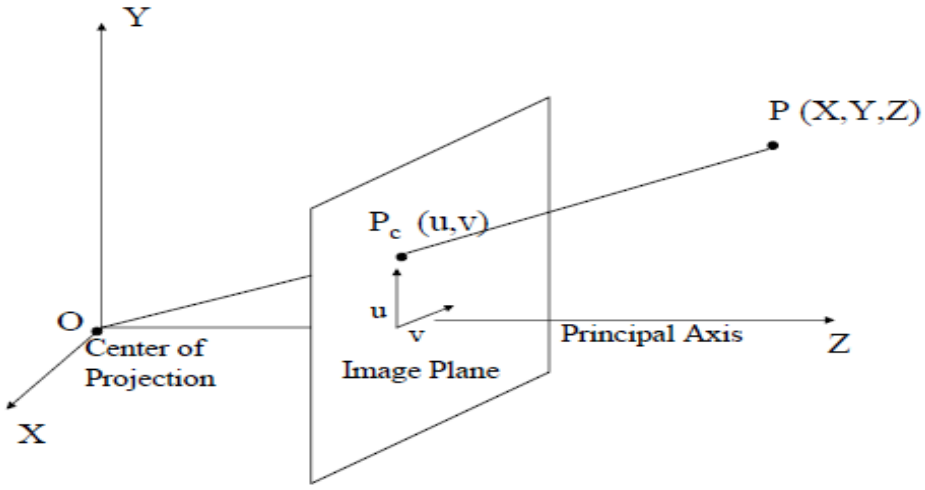


FIGURE I.5 – Projection Pinhole

La figure I.5 montre une caméra avec le centre de projection O et l'axe principal parallèle à l'axe Z. Le plan de l'image est mise au point et donc la distance focale f loin de O. Un point 3D P = (X ; Y ; Z) est imaginé sur le plan image de la caméra à la coordonnée Pc = (u ; v). Nous trouverons d'abord la matrice d'étalonnage de la caméra C qui mappe 3D P en 2D Pc. nous pouvons trouver Pc en utilisant des triangles similaires

$$\frac{f}{Z} = \frac{u}{X} = \frac{v}{Y} \tag{I.3}$$

En utilisant des coordonnées homogènes pour Pc, nous pouvons écrire ceci comme

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{I.4}$$

Vous pouvez vérifier que cela génère bien le point $Pc = (u; v; w) = (\frac{fX}{Z}; \frac{fY}{Z}; 1)$. Notez que P n'est toujours pas dans les coordonnées homogènes.
Ensuite, si l'origine du système de coordonnées de l'image 2D ne coïncide pas avec l'endroit où l'axe Z coupe le plan de l'image, nous devons traduire Pc à l'origine souhaitée. Soit cette traduction dénie par $(t_u; t_v)$. Par conséquent, maintenant $(u; v)$ est

$$\begin{aligned} u &= \frac{fX}{Z} + t_u \\ v &= \frac{fY}{Z} + t_v \end{aligned} \tag{I.5}$$

Cela peut être exprimé sous une forme similaire à l'équation 1 comme

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} f & 0 & t_u \\ 0 & f & t_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{I.6}$$

Maintenant, dans l'équation 2, Pc est exprimé en pouces. Puisqu'il s'agit d'une image de caméra, nous devons l'exprimer en pouces. Pour cela, nous aurons besoin de connaître la résolution de la caméra en pixels / pouce. Si les pixels sont carrés, la résolution sera identique dans les deux directions u et v des coordonnées de l'image de la caméra Cependant, pour un cas plus général, nous supposons des pixels rectangulaires avec une résolution m_u et m_v pixels / pouce respectivement dans les directions u et v. Par conséquent, pour mesure Pc en pixels, ses coordonnées u et v doivent être multipliées respectivement par m_u et m_v. Donc

$$\begin{aligned} u &= m_u \frac{fX}{Z} + m_u t_u \\ v &= m_v \frac{fY}{Z} + m_v t_v \end{aligned} \tag{I.7}$$

Cela peut être exprimé sous forme matricielle comme

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} m_u f & 0 & m_u t_u \\ 0 & m_v f & m_v t_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} \alpha_x & 0 & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} P = KP \tag{I.8}$$

Notez que K ne dépend que des paramètres intrinsèques de la caméra comme sa distance focale, son axe principal et définit ainsi les paramètres intrinsèques de la caméra. Parfois, K a aussi un paramètre de biais s, donné par

$$\begin{bmatrix} \alpha_x & 0 & u_0 \\ 0 & \alpha_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} = K \tag{I.9}$$

Cela intervient généralement si les axes de coordonnées de l'image u et v ne sont pas orthogonaux l'un à l'autre. Notez que K est un matrice triangulaire supérieure 3x3. C'est ce qu'on appelle généra-lement la matrice des paramètres intrinsèques de la caméra.

Maintenant si la caméra n'a pas son centre de projection à (0 ; 0 ; 0) et est orientée de façon arbitraire (pas nécessairement z perpendiculaire au plan de l'image), alors nous avons besoin d'une rotation et d'une translation pour rendre le système de coordonnéesde caméra coïncide avec la configuration de la figure 1. Laisser la caméra se traduire à l'origine du X Y Z coordonnée donnée par T (Tx ; Ty ; Tz). Soit la rotation appliquée pour coïncider l'axe principal avec l'axe Z donnée par une matrice de rota-tion 3x3 R. Ensuite, la matrice formée en appliquant d'abord la translation suivie de la rotation est donnée par la matrice 3x4.

$$E = (R|RT) \tag{I.10}$$

Appelé la matrice de paramètres extrinsèques. Ainsi, la transformation complète de la caméra peut maintenant être représentée comme

$$K(R|RT) = (KR|KRT) = KR(I|T) \tag{I.11}$$

D'où Pc, la projection de P est donnée par

$$Pc = KR(I|T)P = CP \tag{I.12}$$

C est une matrice 3x4 généralement appelée matrice d'étalonnage de caméra complète. Notez que puisque C est 3x4 nous avons besoin de P être en coordonnées homogènes 4D et Pc dérivé par CP sera en coordonnées homogènes 3D. L'exact localisation 2D de la projection sur le plan image de la caméra sera obtenue en divisant les deux premières coordonnées de Pc par le troisième.

I.5 Calibration

I.5.1 Istance Focale

Un objectif est caractérisé par sa distance focale, distance à laquelle se forme l'image d'un point à l'infini, comptée à partir du plan principal image ; ce plan, défini par des considérations d'optique théorique, se place à l'intérieur de l'objectif et se déduit de la constitution de ce dernier

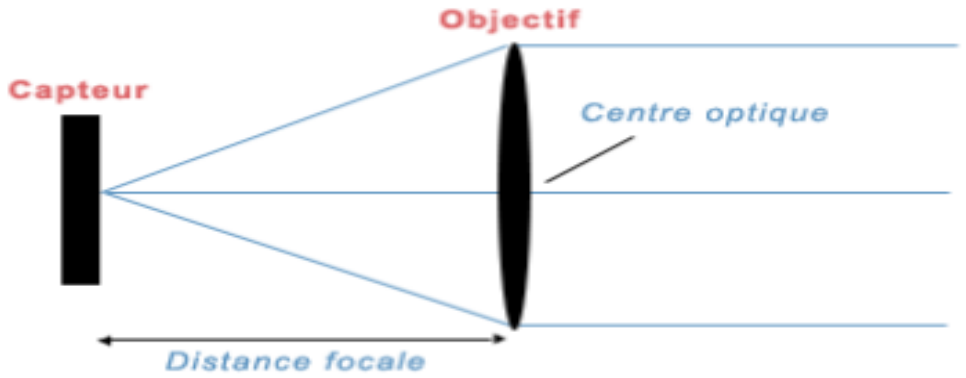


FIGURE I.6 – distance focale

Les objectifs considérés comme normaux par une vielle règle de pratique photographique doivent présenter une distance focale égale à la diagonale de l'image enregistrée, ce qui conduirait à une fo-cale égale de 13 mm por le film 16mm.
En fait, les objectifs normalement utilisés correspond à une focal plus longue : 25 mm pour le film 16mm.
Les objectifs de focale plus courte sont dits grands angulaires et ceux de focale plus longue sont souvent désignés par le terme de télé-objectifs.

I.5.2 Distorsion de l'objectif

Le modèle de projection perspective n'est généralement respecté qu'au centre de l'image. La plupart des objectifs introduisent des déformations de plus en plus visibles lorsque l'on s'approche des bords de l'image [17].
la distorsion de l'objectif produit un déplacement non linéaire de points après leur projections.

FIGURE I.7 – Original distorsion en barillet distorsion en coussin

généralement on modélise distorsion la distorsion d'objectif avec le modèle de distorsion radiale polynomiale :

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6) \begin{bmatrix} x_d \\ y_d \end{bmatrix} \begin{bmatrix} 2P_1 x_d y_d + P_2(r^2 + 2x_d^2) \\ P_1(r^2 + 2y_d^2) + 2P_2 x_d y_d \end{bmatrix} \tag{I.13}$$

$\begin{bmatrix} x_p \\ y_p \end{bmatrix}$:Emplacement du point dans la rétine plan (unitaire f) si la caméra pinhole étaient parfaite

$\begin{bmatrix} x_d \\ y_d \end{bmatrix}$:Emplacement déformé dans le plan rétinien.

$$r^2 = x_d^2 + y_d^2 \tag{I.14}$$

$(k_1 k_2, k_3, P_1, P_2)$ sont les paramètres de distortion

I.5.3 Les paramètres intrinsèques et extrinsèques

Maintenant , nous verrons comment trouver C puis comment le décomposer pour obtenir les para-mètres intrinsèques et extrinsèques. Bien que C avait 12 entrées, l'entrée dans la 3ème ligne et la 4ème colonne est 1. Par conséquent, C a 11 paramètres inconnus. Étant donné C, nous savons que

$$C = (KR|KRT) = (M|MT) \tag{I.15}$$

où KR = M. Notez que, étant donné C, nous pouvons trouver M comme la sous-matrice 3x3 gauche de C. Ensuite, nous utilisons RQ décomposition pour décomposer M en deux 3x3 matrices M = AB, où A est triangulaire supérieur et B est orthogonal matrice (c'est-à-dire $B^T B = I$).*CetrianglesuprieurAcorrespondKetBcorrespondlarotationR.Soitc4dsignentladernire*

$$\begin{aligned} MT &= c4 \\ T &= M^{-1}c4 \end{aligned} \tag{I.16}$$

Ainsi, étant donné C, nous pouvons trouver les paramètres intrinsèques et extrinsèques à travers ce processus. Mais maintenant la question est : comment trouver C pour une caméra générale ? Pour cela, il faut trouver des correspondances entre les points 3D et leurs projections sur l'image de la caméra. Si on connaît un point 3D P_1correspondant à Pc_1 sur la coordonnée image de la caméra,puis

$$Pc_1 = CP_1 \tag{I.17}$$

Ou

$$\begin{bmatrix} u_1 \\ v_1 \\ w_1 \end{bmatrix} = C \begin{bmatrix} X_1 \\ Y_1 \\ Z_1 \\ 1 \end{bmatrix} \tag{I.18}$$

Cependant, notez que les coordonnées d'image de la caméra 2D que nous détectons sont

$$(u_1', v_1') = (\frac{u_1}{w_1}; \frac{v_1}{w1}) \tag{I.19}$$

Notez que trouver C signifie que nous avons trouvé toutes les 12 entrées de C. Ainsi, nous essayons de résoudre pour 12 inconnues. Soit les lignes de C données par r_i, i = 1 ; 2 ; 3. Ainsi

$$C = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \tag{I.20}$$

Puisque nous connaissons la correspondance P_1et Pc_1, nous savons

$$\begin{aligned} u_1' &= \frac{u_1}{w1} = \frac{r_1 P_1}{r_3 P1} \\ v_1' &= \frac{v_1}{w1} = \frac{r_2 P_1}{r_3 P1} \end{aligned} \tag{I.21}$$

Ce qui nous donne deux équations linéaires

$$\begin{aligned} u_1'(r_3 P_1) &= r_1 P_1 \\ v_1'(r_3 P_1) &= r_2 P_1 \end{aligned} \tag{I.22}$$

Notez que dans ces deux équations, seuls les éléments der$_1, r_2$ et $r_3$sont les inconnues. Ainsi, nous constatons que la correspondance de chaque 3D à 2D génère deux équations linéaires. Pour résoudre 12 inconnues, nous aurons besoin d'au moins 6 de ces correspondances.

I.6 La détection des contours avec le filtre de Canny

Les bords ou contours fournissent beaucoup d'information à propos d'une image : ils délimitent les objets présents dans la scène représentée, les ombres ou encore les différentes textures.

Un moyen pour détecter les bords serait de segmenter l'image en objets, mais il s'agit d'un problème difficile. Le filtre de Canny [20] développé en 1986 est une solution plus simple, qui repose sur l'étude du gradient.
Les bords se situent dans les régions de l'image qui présentent de forts changements. En effet, les contours des objets correspondent à des changements de profondeur (on passe d'un objet à un autre situé en arrière-plan), et les ombres et différentes textures à des changements d'illumination.

Mathématiquement, la détection des bords revient donc à chercher les points de l'image où la fonction d'intensité II varie brusquement. Or, nous savons qu'une amplitude du gradient élevée indique un fort changement d'intensité. Le but est donc de chercher les maxima locaux de $||\nabla I||$

La méthode de détection des bords par le filtre de Canny comporte quatre étapes :

Etape 1. Réduction du bruit

Le bruit de l'image peut nous induire en erreur : les pixels aux valeurs aberrantes provoquent des forts changements d'intensité alors qu'ils n'appartiennent à aucun contour.

L'image doit donc être débruitée au préalable avec un filtre adapté. Comme étudié dans la première partie du cours, on utilisera un filtre gaussien pour éliminer le bruit additif, et un filtre médian pour le "poivre et sel" (très rare).

Etape 2. Calcul du gradient de l'image

Le gradient de l'image débruitée est approximé par filtrage de convolution, le plus souvent avec les masques de Sobel.

On calcule ensuite l'amplitude et la direction du gradient en tout point de l'image :

$$\begin{aligned} ||\nabla I(x,y)|| &= \sqrt{(S_x * I(x,y))^2 + (S_y * I(x,y))^2} \\ \theta &= arctan(\frac{S_x * I(x,y)}{S_y * I(x,y)}) \end{aligned} \tag{I.23}$$

Les bords sont repérés par les points de forte amplitude.

Etape 3. Suppression des non-maxima

Les bords trouvés à l'étape précédente sont trop épais. Pour les rendre plus précis, on ne sélectionne que les points pour lesquels l'amplitude du gradient est localement maximale dans sa direction.
Pour cela, on quantifie θ et on trouve les deux voisins de chaque pixel

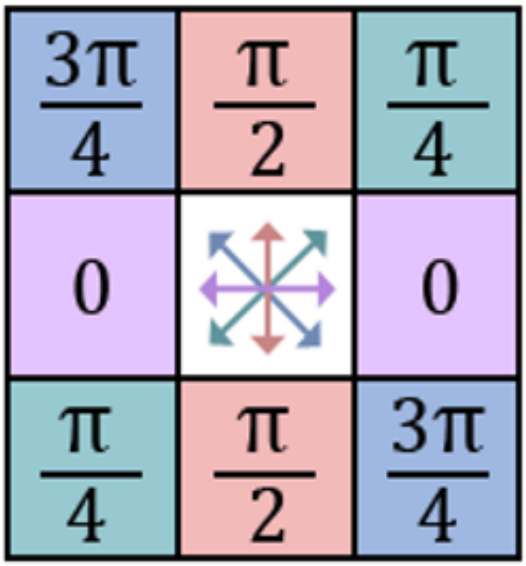


FIGURE I.8 – Quantification de l'angle de la direction du gradient

Par exemple, un pixel (x,y) pour lequel $\theta = 85 \approx \pi 2$ aura pour voisins les pixels $(x, y+1)$ et $(x, y-1)$.
Le pixel courant est retenu que si son amplitude est plus grande que celles de ses deux voisins.

Etape 4. Seuillage

Parmi les points sélectionnés dans l'étape précédente, on ne retient finalement que ceux dont l'amplitude du gradient est supérieure à un certain seuil.
Il est difficile de choisir la valeur d'un "bon" seuil. C'est pourquoi on privilégie le seuillage par hystérésis, qui utilise deux seuils, notés sbassbas et shautshaut.
• Si $||\nabla I(x,y)|| \geq s_{haut}$, alors le pixel (x,y) est situé sur un bord
• Si $||\nabla I(x,y)|| \geq s_{bas}$, alors le pixel (x,y) est situé sur un bord si et seulement si l'un de ses huit voisins aussi
• Sinon, le pixel (x,y) n'est pas sur un bord

I.7 Détection de coin

La détection de COIN est une tâche importante dans divers systèmes de vision par ordinateur et de compréhension d'image. Les applications incluent le suivi de mouvement,

la reconnaissance d'objets et la correspondance stéréo. La détection de coin doit satisfaire un certain nombre de critères importants :
• Tous les vrais coins doivent être détectés.
• Aucun faux coin ne doit être détecté.
• Les coins doivent être bien localisés.
• Le détecteur d'angle doit être robuste vis-à-vis du bruit.
• Le détecteur de coin doit être efficace

De nombreuses recherches ont été menées sur la détection d'angle au cours des dernières années. Moravec[18] a observé que la différence entre les pixels adjacents d'un bord ou d'une partie uniforme de l'image est faible, mais au coin, la différence est significativement élevée dans toutes les directions. Harris [16] a implémenté une technique appelée algorithme de Plessey. La technique était une amélioration de l'algorithme de Moravec.

I.8 Le détecteur de Moravec

Le détecteur de Moravec permet de déterminer les changements d'intensité autour d'un pixel donné.

L'idée est de considérer un voisinage centré en ce pixel de le décaler légèrement dans plusieurs directions, puis de calculer pour chaque déplacement la variation d'intensité. Cela se traduit mathématiquement par la fonction suivante :

$$E_{m,n} = \sum_{(x,y)\in W_{m,n}} [I(x+u, y+v) - I(x,y)]^2 \tag{I.24}$$

$E_{m,n}(u,v)$ représente la différence d'intensité entre le voisinage $W_{m,n}$ centré en le pixel (m,n) et le voisinage $W_{m,n}$ décalé de (u,v).

On applique cette fonction dans les trois situations principales ci-dessous :
Situation 1 : la zone contient un coin, l'intensité change brusquement dans plusieurs directions, donc la fonction EE prend de fortes valeurs dans ces directions
Situation 2 : la zone contient un contour, l'intensité change brusquement si on se déplace horizontalement et très peu verticalement. Ainsi, EE prend de fortes valeurs si on déplace perpendiculairement au contour, et des faibles pour des déplacements le long du contour.
Situation 3 : pas de changement d'intensité : la région est uniforme. EE prend alors de faibles valeurs dans toutes les directions.

I.9 Le détecteur de Harris-Stephens

Le détecteur de Harris-Stephens est une amélioration du détecteur de Moravec. Il apporte trois modifications majeures :

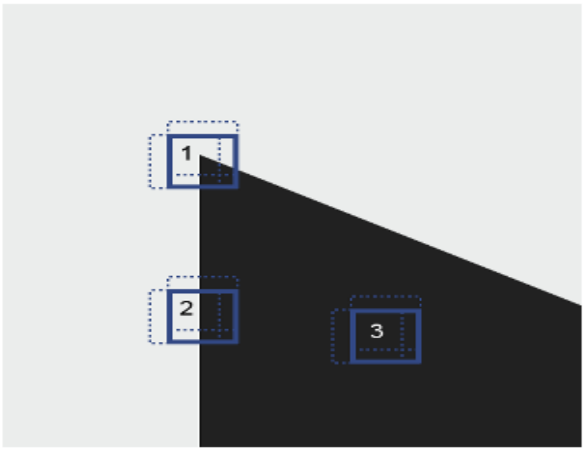


FIGURE I.9 – Détecteur de Moravec

1. La fenêtre carrée $W_{m,n}$ centrée en (m, n) est remplacée par une fenêtre gaussienne :

$$E_{m,n}(u,v) = \sum_{(x,y)\in W_{m,n}} [I(x+u, y+v) - I(x,y)]^2 \qquad W(x,y) = e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{I.25}$$

2. $I(x+u, y+v)$ est approximé par un développement de Taylor au voisinage de (x, y) :

$$I(x+u, y+v) \approx I(x,y) + u\frac{\delta I(x,y)}{\delta x} + v\frac{\delta I(x,y)}{\delta y} \tag{I.26}$$

On obtient donc finalement

$$\begin{aligned} E_{m,n}(u,v) &\approx \sum_{(x,y)} W_{m,n}[u\frac{\delta I(x,y)}{\delta x} + v\frac{\delta I(x,y)}{\delta y}]^2 \\ &\approx [u \quad v] M \begin{bmatrix} u \\ v \end{bmatrix} \end{aligned} \tag{I.27}$$

Avec :

$$M = \sum_{(x,y)} W(x,y) \approx \begin{bmatrix} \frac{\delta I(x,y)^2}{\delta x} & \frac{\delta I(x,y)}{\delta x}\frac{\delta I(x,y)}{\delta y} \\ \frac{\delta I(x,y)}{\delta x}\frac{\delta I(x,y)}{\delta y} & \frac{\delta I(x,y)^2}{\delta y} \end{bmatrix} \tag{I.28}$$

Comme d'habitude, les dérivées partielles sont approximées avec les masques de Sobel. La matrice M décrit le comportement local de $E_{m,n}$.

3. Les coins sont détectés selon un nouveau critère : les valeurs propres $\lambda 1$ et $\lambda 2$ de M.

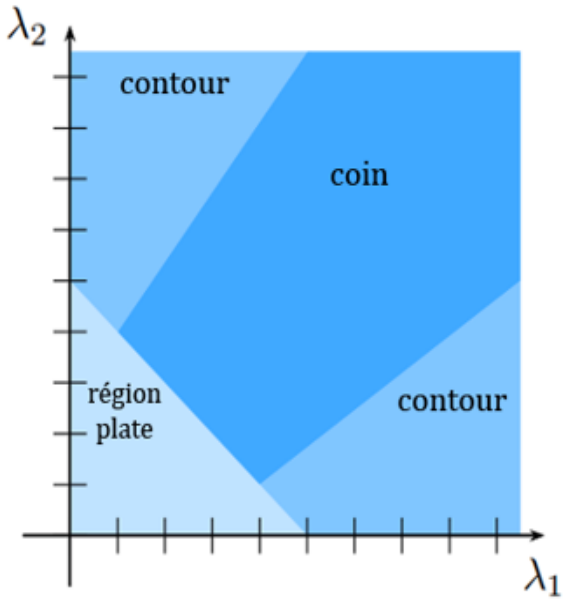


FIGURE I.10 – Détection des coins en fonction des valeurs propres

Trouver les valeurs propres de M peut être fastidieux. Une alternative consiste alors à analyser les valeurs de l'opérateur R :

$$R = det(M) - k.trace(M)^2 = (\lambda_1\lambda_2 - k.(\lambda_1 + \lambda_2)^2) \tag{I.29}$$

Avec k une constante choisie entre 0,04 et 0,06.

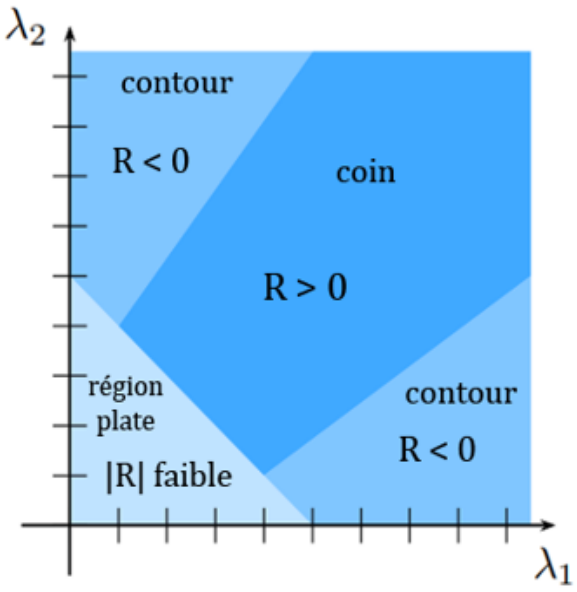


FIGURE I.11 – Détection des coins en fonction de la valeur de R

I.10 Étapes de l'algorithme du détecteur de Harris-Stephens

La détection des coins avec le détecteur de Harris-Stephens se fait donc en quatre étapes :
Etape 1 :
Calcul de la matrice MM pour chaque pixel. Les dérivées partielles sont approximées en filtrant l'image par convolution avec les masques de Sobel.
Etape 2 :
Calcul de $R = det(M) - k.trace(M)^2$ pour chaque pixe
l. **Etape 3 :**
Seuillage de RR. On sélectionne les pixels pour lesquels >s , où s est un seuil à choisir.
Etape 4 :
Suppression des non-maxima de R. Les coins correspondent aux maxima locaux de R : pour les trouver, on applique aux pixels sélectionnés la méthode décrite dans l'étape 3 du filtre de Canny

I.11 Conclusion

Dans ce chapitre nous avons essayé de réaliser une étude bibliographique générale sur les caméras, en rappelant quelques définitions sur la notion de calibration de la caméra, et également la détection des contours et des coins.

Chapitre II

LE FLUX OPTIQUE

II.1 Introduction

Dans le travail actuel de cette thèse, nous explorerons en profondeur les principes de base du flux optique, qui a un effet particulier. Le but de ce chapitre est de présenter le concept de flux optique, de présenter et de discuter des méthodes de calcul associées. Nous aborderons d'abord les différentes méthodes (Horn et Schunck, Lucas-Kanake, Bruhn, etc.), puis enfin étudierons la méthode Lucas-Kanake dans Opencv et nous montrerons des résultats obtenus par cette dernière méthode.

II.2 Le flux optique

Le flux optique est le mouvement d'objets entre des images consécutives de séquence, causé par le mouvement relatif entre l'objet et la caméra. Le problème du flux optique peut être exprimé comme suit :

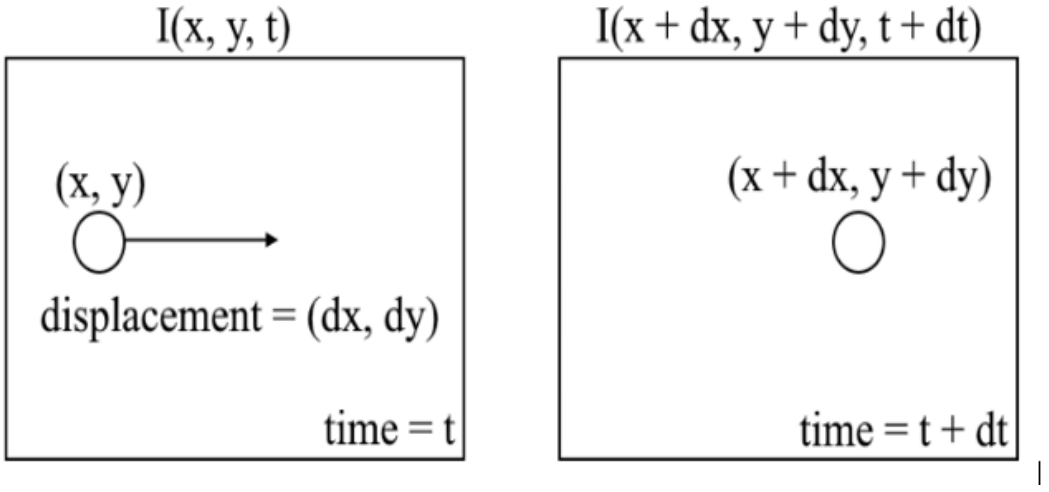


FIGURE II.1 – Problème de flux optique

Où entre des images consécutives, nous pouvons exprimer l'intensité de l'image (I) en fonction de l'espace (x, y) et le temps(t). En d'autres termes, si nous prenons la première image $I(x, y, t)$ et déplacez ses pixels de (dx, dy) plus de temps, on obtient la nouvelle image $I(x + dx, y + dy, t + dt)$.

Tout d'abord, nous supposons que les intensités des pixels d'un objet sont constantes entre des images consécutives.

$$I(x, y, t) = I(x + dx, y + dy, t + dt) \tag{II.1}$$

Deuxièmement, nous prenons l'approximation de la série Taylor de l'ERS et supprimons les termes courants.

$$I(x+\delta x, y+\delta y, t+\delta t) = I(x,y,t) + \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t + ... \tag{II.2}$$

$$\Rightarrow \frac{\partial I}{\partial x}\delta x + \frac{\partial I}{\partial y}\delta y + \frac{\partial I}{\partial t}\delta t = 0 \tag{II.3}$$

Troisièmement, nous divisons par dt pour dériver l'équation de flux optique :

$$\frac{\partial I}{\partial x}u + \frac{\partial I}{\partial y}v + \frac{\partial I}{\partial t} = 0 \tag{II.4}$$

$$f_x u + f_y v + f_t = 0 \tag{II.5}$$

Où $u = dx/dt$ et $v = dy/dt$.

$dI/dx, dI/dy$ et dI/dt sont les dégradés de l'image le long de l'axe horizontal, de l'axe vertical et du temps. Par conséquent, nous concluons avec le problème du flux optique, c'est-à-dire la résolution $u(dx/dt)$ et $v(dy/dt)$ pour déterminer le mouvement dans le temps. Vous remarquerez peut-être que nous ne pouvons pas résoudre directement l'équation du flux optique pour u et v puisqu'il n'y a qu'une seule équation pour deux variables inconnues. Nous allons implémenter certaines méthodes telles que la méthode Lucas-Kanake pour résoudre ce problème.

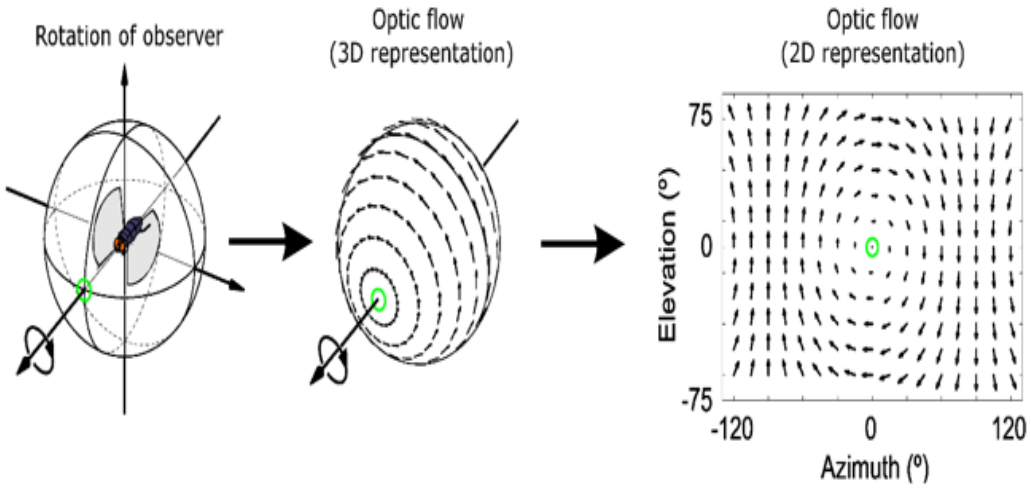


FIGURE II.2 – Flot optique

II.3 Les differentes methodes de calcul du flux optique

II.3.1 La méthode Horn-Schunck

La méthode Horn-Schunck est un algorithme d'estimation de flux optique classique. Il suppose une fluidité dans le flux sur toute l'image. Ainsi, il essaie de minimiser les

distorsions dans l'écoulement et préfère les solutions qui montrent plus de douceur. Dans cette mission, nous implémenterons la version moderne de la méthode Horn-Schunck, en utilisant un schéma pyramidal grossier à fin pour obtenir de meilleures performances, similaires à celles de la méthode Lucas-Kanade. De nombreux algorithmes de flux optique actuels sont construits sur son cadre.

L'objectif est formulé comme une fonctionnelle énergétique globale qui est ensuite minimisée. Soit l'image $p = (x, y)$ et le champ de flux sous-jacentw $(p) = (u(p), v(p), 1)$, où $u(p)$ et $v(p)$ sont l'horizontale et composantes verticales du champ d'écoulement, respectivement. Sous l'hypothèse de constance de luminosité, la valeur de pixel doit être cohérente le long du vecteur de flux et le champ de flux doit être lisse par morceaux. Il en résulte une fonction objective dans le domaine spatial continu.

$$E(u, v) = \int |I_2(P + W) - I_1(P)|^2 + \lambda(|\nabla u|^2 + |\nabla u|^2)dp \tag{II.6}$$

Où ∇ est l'opérateur de gradient, λ pondère la régularisation, $I1$ et $I2$ sont deux images correspondantes.

II.3.2 La méthode Lucas-Kanake

Lucas et Kanade ont proposé une technique efficace pour estimer le mouvement de caractéristiques intéressantes en comparant deux images consécutives [2]. La méthode Lucas-Kanade fonctionne sous les hypothèses suivantes :

1- Deux images consécutives sont séparées par un petit incrément de temps (dt) de sorte que les objets ne sont pas déplacés de manière significative (en d'autres termes, la méthode fonctionne mieux avec les objets se déplaçant lentement).

2- Un cadre dépeint une scène «naturelle» avec des objets texturés présentant des nuances de gris qui changent en douceur.

Premièrement, sous ces hypothèses, nous pouvons prendre une petite fenêtre 3x3 (voisinage) autour des caractéristiques détectées par Shi-Tomasi et supposer que les neuf points ont le même mouvement.

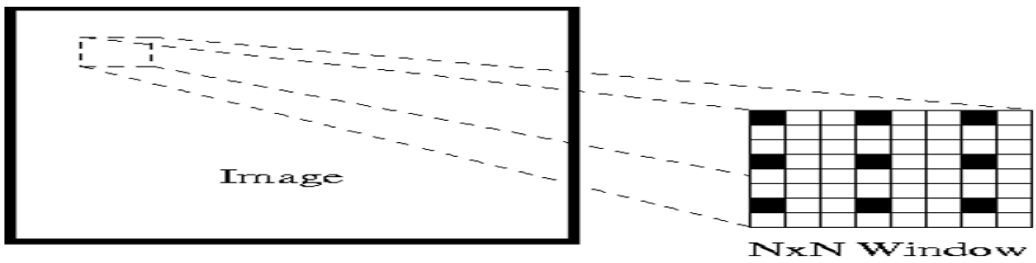


FIGURE II.3 – Lucas-Kanade le flux optique est estimé pour les pixels noirs

Cela peut être représenté par :

$$\begin{aligned} I_x(q_1)V_x + I_y(q_1)V_y &= -I_t(q_1) \\ I_x(q_2)V_x + I_y(q_2)V_y &= -I_t(q_2) \\ I_x(q_n)V_x + I_y(q_n)V_y &= -I_t(q_n) \end{aligned} \tag{II.7}$$

Où $q1, q2, \ldots$,qn désignent les pixels à l'intérieur de la fenêtre (par exemple n = 9 pour une fenêtre (3x3) et $Ix(qi), Iy(qi)$ et $It(qi)$ désignent les dérivées partielles de l'image I par rapport à position (x, y) et temps t, pour le pixel qi à l'instant courant.

Il ne s'agit que de l'équation de flux optique (que nous avons décrit précédemment) pour chacun des n pixels.

L'ensemble d'équations peut être représenté sous la forme matricielle suivante où $Av = b$:

$$\text{A}=\begin{bmatrix} I_x(q_1) & I_x(q_2) \\ I_y(q_1) & I_y(q_2) \\ I_x(q_n) & I_y(q_n) \end{bmatrix},\text{v}=\begin{bmatrix} V_x \\ V_y \end{bmatrix},\text{b}=\begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ -I_t(q_n) \end{bmatrix}$$

Notez que précédemment (voir la section " Définition de flux otique"), Nous avons été confrontés au problème de devoir résoudre deux variables inconnues avec une équation. Nous devons maintenant résoudre deux inconnues (Vx et Vy) avec neuf équations, ce qui est surdéterminé.

Deuxièmement, pour résoudre le problème de surdétermination, nous appliquons least squares fitting pour obtenir le problème suivant à deux équations-deux inconnus :

$$A = \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i)I_y(q_i) \\ \sum_i I_y(q_i)I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i)I_t(q_i) \\ -\sum_i I_y(q_i)I_t(q_i) \end{bmatrix} \tag{II.8}$$

Où $Vx = u = dx/dt$ désigne le mouvement de x dans le temps et $Vy = v = dy/dt$ désigne le mouvement de y dans le temps. La résolution des deux variables complète le problème du flux optique.

FIGURE II.4 – Flux optique clairsemé de chevaux sur une plage

En bref, nous identifions quelques fonctionnalités intéressantes pour suivre et calculer de manière itérative les vecteurs de flux optique de ces points. Cependant, l'adoption de la méthode Lucas-Kanade ne fonctionne que pour les petits mouvements (de notre hypothèse initiale) et échoue lorsqu'il y a un grand mouvement. Par conséquent, l'implémentation OpenCV de la méthode Lucas-Kanade adopte des pyramides.

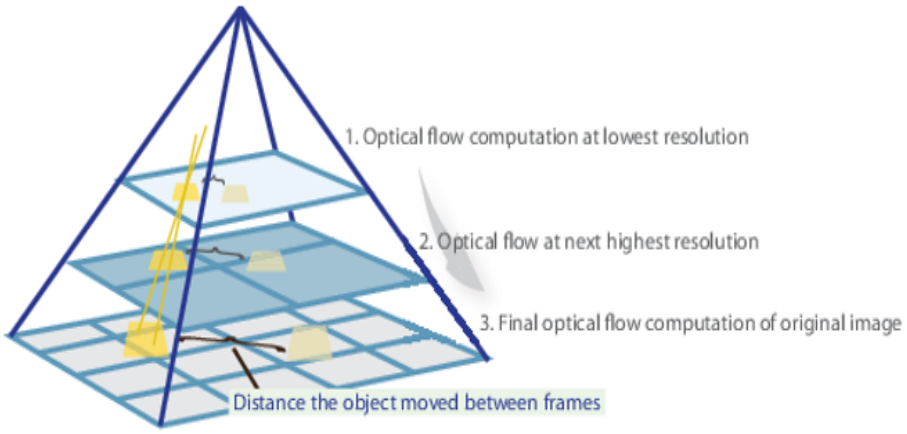


FIGURE II.5 – La méthode Pyramide calcule le flux optique à différentes résolutions

Dans une vue de haut niveau, les petits mouvements sont négligés lorsque nous montons la pyramide et les grands mouvements sont réduits à de petits mouvements - nous calculons le flux optique avec redimensionnement.

II.3.3 La méthode Bruhn et al

Depuis 2004, Bruhn et al. La contrainte de calcul du flux optique a été reformulée en combinant les méthodes de Horn et Schunck ainsi que celle de Lucas et Kanade [11][10]. Il en résulte une méthode à la fois locale et globale, alliant la densité de la méthode de Horn et Schunck à la précision et la robustesse de la méthode de Lucas et Kanade. La méthode consiste à minimiser la quantité suivante :

$$\int\int_{\Omega} \overrightarrow{v}^T J_\sigma(\overrightarrow{\nabla_3 I})\overrightarrow{v} + \lambda^2(||\overrightarrow{\nabla_v x}||_2^2 + ||\overrightarrow{\nabla_v y}||_2^2)dxdy \tag{II.9}$$

On remarque que l'équation de contrainte est proche de celle formulée par Horn et Schunck, à l'exception du premier terme de l'intégrale : $\overrightarrow{v}J_\sigma(\overrightarrow{\nabla_3}\overrightarrow{v})$ qui correspond à la contrainte de Lucas et Kanade. Dans cette équation $\overrightarrow{\nabla_3}$ correspond au gradient spatio-temporel et $J_\sigma = G_\sigma * (\overrightarrow{\nabla_3}\overrightarrow{\nabla_3}I^T)$ étant un noyau gaussien de variance σ avec lequel on effectue la convolution.

Dans ce cas pour $\sigma = 0$ l'équation devient celle du gradient formulée par Horn et Schunck. Pour $\lambda = 0$ l'équation est celle formulée par Lucas et Kanade. C'est donc en faisant varier ces paramètres que l'on peut plus ou moins lisser le champ de flux optique

(paramètre λ) ou améliorer la précision (paramètre σ) sachant que cette dernière option augmente le temps de calcul.

II.4 Flux optique LUCAS-KANADE dans OPENCV

OpenCV fournit tout cela dans une seule fonction, cv2.calcOpticalFlowPyrLK (). Ici, nous créons une application simple qui suit certains points d'une vidéo. Pour décider des points, nous utilisons cv2.goodFeaturesToTrack (). Nous prenons la première image, détectons certains points d'angle de Shi-Tomasi, puis nous suivons de manière itérative ces points à l'aide du flux optique Lucas-Kanade. Pour la fonction cv2.calcOpticalFlowPyrLK (), nous passons l'image précédente, les points précédents et l'image suivante. Il renvoie les points suivants avec des numéros d'état qui ont une valeur de 1 si le point suivant est trouvé, sinon zéro. Nous passons itérativement ces points suivants comme points précédents à l'étape suivante.

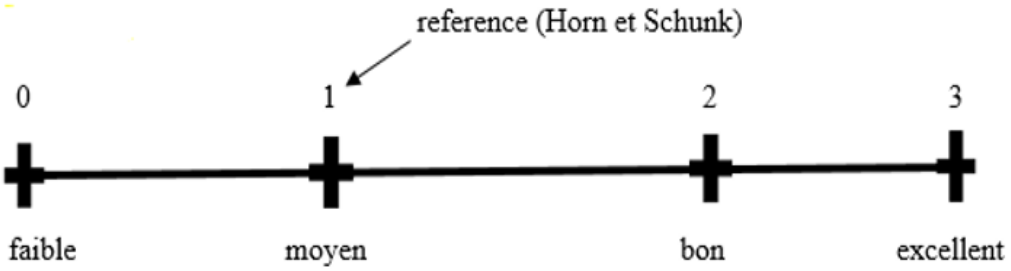
II.5 Conclusion

Dans ce chapitre on a introduit le flux optique et démontré son utilisation. Notre projet vise comme but dans la localisation des objets dans l'environnement. Il y a plusieurs méthodes de calcul du flux optique, on a présenté trois méthodes ; La méthode Horn-Schunck, La méthode Lucas-Kanake et La méthode Bruhn et al.
Finalement on a opté pour la méthode Lucas-Kanake dans l'OpenCV pour son avantage de bonne précision et temps de calcul optimisé. Pour récapituler les différentes caractéristiques des méthodes présentées et pour clarifier nos choix, nous utiliserons le tableau suivant déduit par une étude comparative entre les méthodes [9].

- **0 faible**
- **1 moyen**
- **2 bon**
- **3 excellent**

	Précision	Temps de calcul
Horn et Schunck	1	1
Nagel	0	1
Lucas et Kanade	3	2
Simoncelli	1	2
Bruhn et al.	2	2
Camus.	0	3
Heeger	2	0

TABLE II.1 – Tableau récapitulatif des différentes caractéristiques des méthodes de calcul du flux optique [9]



Chapitre III

LES CAPTEURS INERTIELS

III.1 Introduction

La localisation d'un objet est un sujet prédominant dans la recherche car utilisée dans de multiples domaines (robotique, véhicule intelligents, drones etc...).La localisation d'un objet consiste à déterminer sa position dans un repère donné [23].

Avec le développement rapide de la technologie (MEMS), des capteurs à bas prix pouvant supporter des fortes contraintes dynamique sont maintenant courants, cette thèse a pour objectif de présenter la localisation à partir d'un capteur inertiel.

Après cette présentation de la problématique donnons dans ce qui suite une description des capteurs les plus utilisés dans le domaine de la localisation [21].

III.2 Définition de capteur

Un capteur est un organe de prélèvement d'information qui élabore à partir d'une grandeur physique, une autre grandeur physique de nature différente (très souvent électrique). Cette grandeur représentative de la grandeur prélevée est utilisable à des fins de mesure ou de commande.

Le capteur est la partie de base du système d'acquisition de données. Leurs réalisations se situent dans le domaine des instruments.

III.3 Définition de capteur inertiel

Les capteurs inertiels fournissent aux satellites des mesures relatives à une référence fixe dans l'espace. Le gyroscope fournit l'attitude (vitesse angulaire) par rapport à cette référence, tandis que l'accéléromètre fournit la position. Pour le premier, un mouvement de rotation absolu est détecté, tandis que pour le second, un mouvement de translation accéléré est détecté.

III.4 Les differents types de capteurs

III.4.1 Gyromètres

Le gyroscope est un capteur de mouvement. Par rapport à un référentiel inertiel (ou Galileo), un gyroscope mesure la vitesse de rotation du référentiel selon un ou plusieurs axes. Il existe plusieurs types de gyroscopes : les gyroscopes mécaniques, les gyroscopes optiques et les gyroscopes vibratoires.

a-Gyromètres mécaniques

Le gyroscope mécanique utilise le haut du miroir du gyroscope. En raison de sa vitesse de rotation élevée, il présente une rigidité gyroscopique qui lui permet de rester dans une direction fixe. En raison de la rigidité du gyroscope, tout couple appliqué à ce dernier entraînera un éloignement de l'arbre de la direction initiale à faible vitesse. Le principe est d'estimer le déplacement du véhicule par rapport au sens de référence de l'axe de rotation du moule de broche.

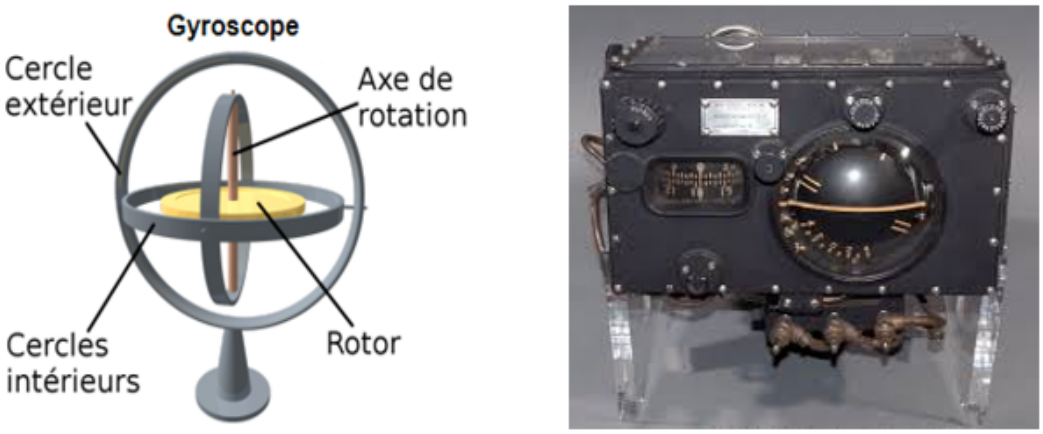


FIGURE III.1 – Gyromètres mécaniques

b-Gyromètres optiques

Les gyromètres optiques peuvent être des gyromètres laser ou à fibre optique, dans les deux cas, ils présentent l'avantage de se dispenser de pièce mécanique en mouvement (pas d'usure), d'avoir de plus grande dynamique de mesure et bande passante, une insensibilité à l'accélération et moins de contraintes concernant la stabilité en température. Les gyromètres laser fonctionnent suivant le principe du laser à cavité résonnante. Les gyromètres à fibre optique reprennent l'effet Sagnac.

Deux ondes parcourant un chemin fermé en rotation (par rapport à un référentiel inertiel) subissent un décalage temporel lorsqu'elles ont été émises et reçues par un émetteur/récepteur fixe par rapport au chemin optique. Le décalage temporel entre les deux rayons lumineux est ainsi proportionnel à la vitesse de rotation du système.

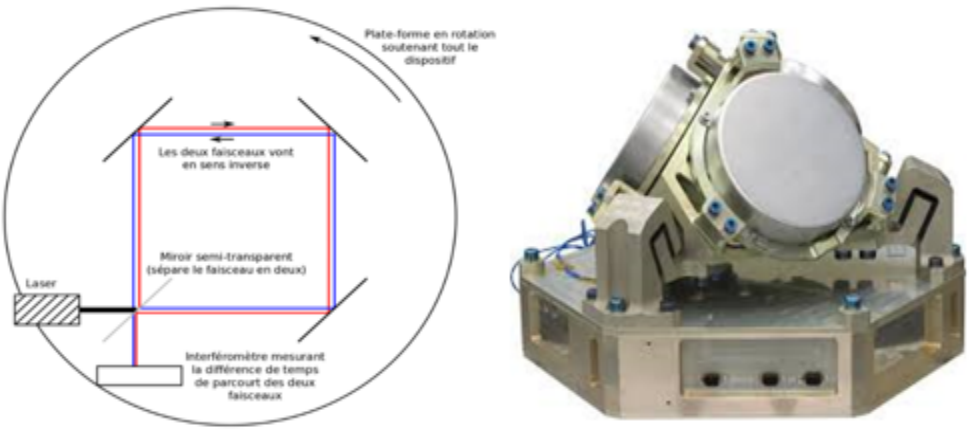


FIGURE III.2 – Gyromètres optiques

c- Gyromètres vibrants

Les gyroscopes vibrants sont très similaires aux gyroscopes mécaniques, sauf qu'il n'y a pas de pièces mobiles. Grâce à la force de Coriolis, ils détectent le déplacement des ondes de vibration dans la structure. Le gyroscope vibrant MEMS en est un exemple. En raison de sa petite taille et de son faible prix, il est très courant dans de nombreux appareils électroniques.

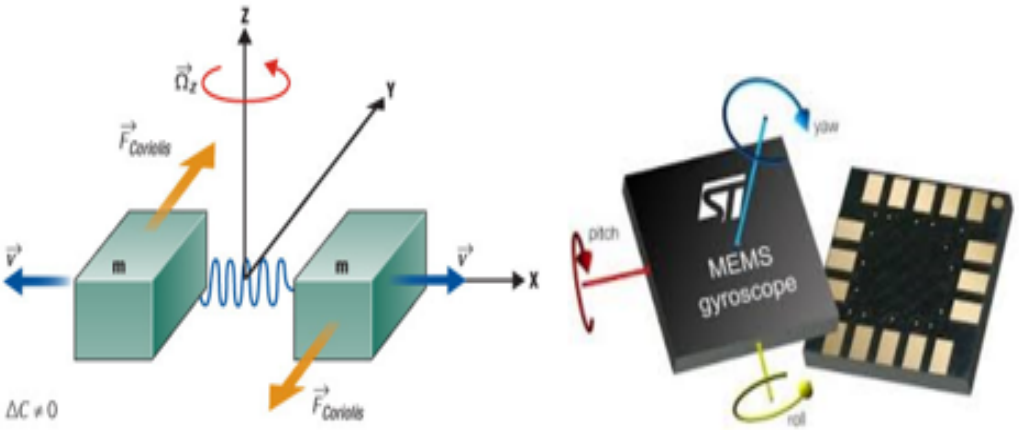


FIGURE III.3 – Gyromètres vibrants

III.4.2 Accéléromètres

Un accéléromètre est un capteur utilisé pour enregistrer en trois dimensions l'accélération linéaire et dynamique d'un objet. Les accéléromètres sont principalement utilisés pour la navigation et le guidage des véhicules de relance atmosphérique ou pour l'identification des micro-vibrations. Le changement de leur tension de sortie se traduit par une

accélération. La règle de base est d'avoir une masse dans le boîtier fixée avec des ressorts. Au fur et à mesure que le boîtier est accéléré, la masse a tendance à rester stationnaire par inertie : elle est donc en mouvement par rapport au boîtier.

Les ressorts contrarient alors le mouvement de la masse par rapport au carter, et le déplacement devient alors proportionnel à son accélération. Les accéléromètres sont capables de mesurer la résultante des forces de surface, mais pas les accélérations d'origine gravitationnelle, car la masse et son logement sont soumis au même champ gravitationnel.

L'accéléromètre est conçu comme un capteur capable de mesurer l'accélération linéaire et la vibration d'un objet en trois dimensions. L'accéléromètre est généralement divisé en deux parties : la partie mécanique, qui est chargée de détecter l'accélération de la masse contenue dans l'équipement ; la partie électronique, dont la tâche est d'interpréter le signal.

Les accéléromètres se trouvent dans de nombreux objets du quotidien, tels que les téléphones intelligents, les voitures, les montres de sport, etc. Veuillez noter qu'il existe de nombreux types d'accéléromètres, selon qu'il s'agit de détection piézoélectrique (pour les capteurs de pression), de détection piézorésistive, de détection de capacité, de détection optique, etc.

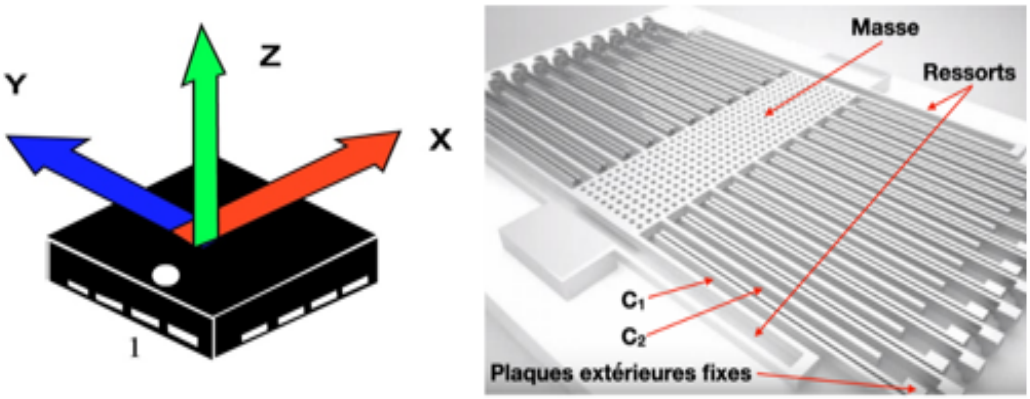


FIGURE III.4 – MEMS Accéléromètres

a-Approche intuitive

Le capteur d'accélération peut être illustré par le système de ressort de masse. Considérons le contraire de ce schéma : à l'équilibre, la position x de la masse m sera utilisée comme référence, donc $x = 0$. Si le support est soumis à une accélération verticale, deux choses se produiront : d'une part, le support se déplacera vers le haut ; en raison de l'inertie de la masse m, le support aura tendance à rester dans sa position initiale, et d'autre part le ressort sera contraint de se comprimer. Plus l'accélération appliquée au support est grande, plus la valeur de x.

Nous pouvons montrer les principes de base de l'utilisation de la dynamique du système non amorti (et considérer le système horizontalement, donc nous ne considérons pas les poids) :$m\ddot{x} + kx = 0$ signifie accélération de masse, m signifie accélération, x signifie position de support (par rapport au cadre de référence Galileo).

Évidemment, l'accélération est proportionnelle à x. En mesurant simplement le déplacement de la masse m par rapport à son support, on peut connaître l'accélération de la masse m.

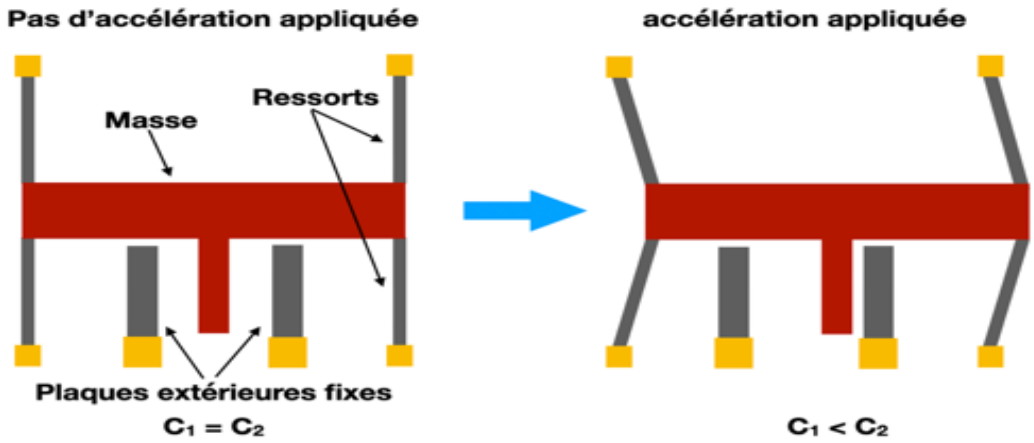


FIGURE III.5 – le principe de fonctionnement d'Accéléromètres

Principe

Le principe de la plupart des accéléromètres est basé sur la loi fondamentale de la dynamique :

$$F = ma \tag{III.1}$$

Avec
F : Force(N).
m : Masse (kg).
a : Accélération (m/s^2) aussi notée γ).

Pour être plus précis, il consiste en l'équivalent entre la force d'inertie de la masse sismique du capteur et la force de rappel appliquée à la masse. Il existe deux séries principales d'accéléromètres : les accéléromètres non servo et les accéléromètres servo.

Principaux paramètres propres à un accéléromètre

En plus des caractéristiques classiques des capteurs, l'accéléromètre peut être caractérisé par les données suivantes :

• son étendue de mesure est exprimée en $g = 9,80665m/s^2$
• sa masse du capteur, la finesse (terme technique correct correspondant)
• sa sensibilité transversale
• son nombre d'axes (1 à 3 axes)
• sa construction mécanique
• La présence d'une électronique intégrée.
• Son prix (en 2007, de 6 euros pour un capteur capacitif non asservi jusqu'à 3 000 euros pour un capteur asservi haut de gamme).

Toutes ces caractéristiques interagissent et caractérisent un principe, une technologie ou un procédé de fabrication.

III.5 Les differents usages des capteurs inertiels

III.5.1 Les applications de Gyromètre

Les gyromètres (ou gyroscopes) sont utilisés :
• Pour la stabilisation d'une direction ou d'un référentiel mécanique, par exemple pour la stabilisation d'une caméra, d'une antenne ou d'un viseur infrarouge d'un autodirecteur de missile,
• Dans les systèmes de guidage des missiles ou fusées,
• Dans les systèmes de pilotage automatique des aéronefs
• En association avec des accéléromètres, pour déterminer la position, la vitesse et l'attitude d'un véhicule (avion, char, bateau, sous-marin, etc.). Dans ce cas, il s'agit d'un équipement appelé centrale à inertiel.

Ces équipements peuvent être complémentaires avec un GPS sauf dans les applications où celui-ci n'est pas utilisable (sous-marins, satellites).

FIGURE III.6 – Applications des Gyromètres

III.5.2 Les applications d'Accéléromètre

Les applications de ce capteur sont très diverses :
• la mesure de vitesse (par intégration)
• la mesure de déplacement (par double intégration)
• le diagnostic de machine (par analyse vibratoire)
• la détection de défaut dans les matériaux (en mesurant la propagation d'une vibration à travers les matériaux)

Néanmoins, elles sont généralement classées en trois grandes catégories :
• Les chocs
• L'accélération vibratoire
• L'accélération de mobiles

III.6 Le capteur MPU6050

Le module MPU-6050 intègre un système micro électromécanique (gyroscope 3 axes) et un accéléromètre 3 axes dans le même circuit (DMP), qui peuvent vous fournir les informations de navigation les plus précises et manipuler votre robot mobile, véhicule à moteur ou véhicule, avion sans pilote

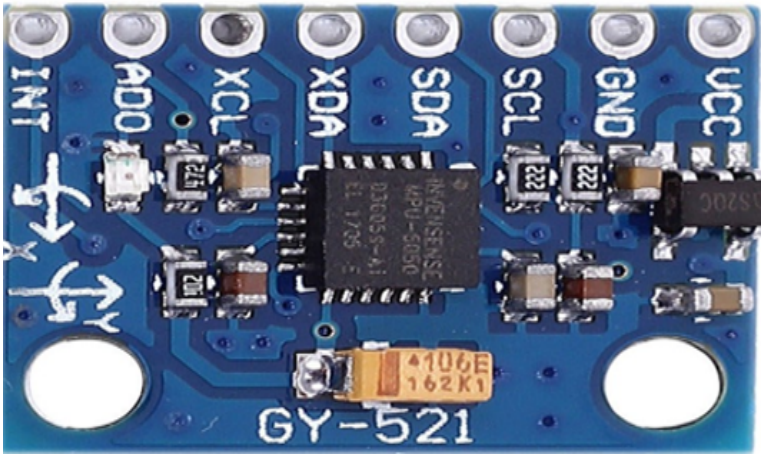


FIGURE III.7 – le capteur MPU 6050

III.6.1 Les Caractéristiques

Le module GY-521 est une carte de dérivation pour les MEMS MPU-6050 (systèmes micro électromécaniques) qui comprend un gyroscope à 3 axes, un accéléromètre à 3 axes, un processeur de mouvement numérique (DMP) et un capteur de température. Le processeur de mouvement numérique peut être utilisé pour traiter des algorithmes complexes

directement sur la carte. Habituellement, le DMP traite des algorithmes qui transforment les valeurs brutes des capteurs en données de position stables. Ce didacticiel ne donne qu'une brève introduction au GY-521 / MPU-6050. En particulier, il montre comment récupérer les valeurs brutes du capteur. Les valeurs des capteurs sont récupérées à l'aide du bus de données série I2C, qui ne nécessite que deux fils (SCL et SDA).

III.6.2 Les Spécifications

Les spécifications de ce capteur sont :
- Modèle : GY-521
- Puce : MPU-6050
- Alimentation : 3.3V - 5V (le module inclut un régulateur de basse tension)
- Communication : Protocole de communication de la CII standard
- Convertisseur de 16 bits AD intégré, sortie de données 16 bits
- Gammes de gyroscope : ± 250, 500, 1000, 2000°/s
- Accéléromètres : ±2, ±4, ±8, ±16g
- Hauteur de Brôches : 2.54mm

III.7 Conclusion

Dans ce chapitre nous avons essayé d'expliquer ce que veut dire le capteur en général et le capteur inertiel en particulier.
Au début on a défini le capteur inertiel en énumérant les différents types (Gyromètres, Accéléromètres) et leur principe. Puis, nous avons présenté les différentes applications de ce capteur.

Enfin, nous avons présenté le capteur MPU6050, ainsi que ses caractéristiques.

Chapitre IV

FUSION DES DONNEES POUR L'ESTIMATION

IV.1 Introduction

La fusion de données vise l'association, la combinaison et l'intégration de multiples sources de données de nature différentes (symboliques, numériques,...) représentant des connaissances et des informations diverses dans le but de fournir une information globale plus fiable et plus complète. Les données fusionnées reflètent non seulement l'information générée par chaque source de données, mais encore l'information qui n'aurait pu être fournie par aucune des sources prises séparément. Le terme de fusion de données s'est étendu à de plus vastes domaines de recherche notamment dans les applications de diagnostic médical, la robotique, la compréhension automatique de documents scientifiques... Tous ces domaines ont en commun le fait de devoir manipuler de grandes quantités de données de natures et de types variés, afin d'obtenir une information de meilleure qualité.

Fusionner l'odométrie et la cartographie permet par exemple de remédier au problème d'intermittence souvent rencontrée dans l'utilisation d'un GPS dans les centres villes. De ce fait, la fusion de données devient particulièrement nécessaire autant d'un point de vue fondamental que pour des applications pratiques.

Dans les paragraphes restants, nous n'introduisons pas en détail certaines des techniques de fusion de données les plus couramment utilisées, et cet article propose également une solution simple et facile à mettre en œuvre pour les algorithmes de fusion de données de capteurs utilisant des filtres. La carte de couplage faible de Kalman (KF) est utilisée pour estimer la position d'objets évoluant dans l'espace tridimensionnel. L'avantage de cette démonstration est qu'elle clarifie la fusion multi-capteurs dans le domaine de la localisation.

IV.2 Fusion de données

IV.2.1 Définition

La fusion de données est « un procédé traitant de l'association, de la corrélation et de la combinaison des données et de l'information provenant de sources uniques ou multiples afin d'aboutir à des estimées affinées des positions et des identités, à une évaluation complète et en temps utile des situations et des menaces, et de leurs importances. Le procédé est caractérisé par une amélioration continue des estimées et des prévisions, et par l'évaluation du besoin en sources additionnelles ou de modification du procédé lui-même, afin d'atteindre des résultats améliorés. »

IV.3 Approches classiques pour la fusion de données

Un grand nombre de techniques de fusion de donnée ont été décrites dans la littérature. Parmi les plus utilisées on peut citer : le filtre de Kalman, les modèles de Markov

cachés, l'estimation bayésienne, la théorie de Dempster-Shafer, la logique floue,... Dans les paragraphes qui suivent, nous donnons une brève description de certaines de ces techniques.

IV.3.1 Les modèles de Markov cachés

Les modèles de Markov cachés, ou HMM pour Hidden Markov Models, sont des modèles statistiques de données séquentielles, qui ont été largement utilisés dans le domaine de l'intelligence artificielle, en particulier dans le domaine de la reconnaissance de la parole.
L'hypothèse Markovienne est à la base de nombreux modèles stochastiques dont les HMM. Cette hypothèse suppose que l'état d'un système peut être totalement déterminé à condition de connaître l'état dans lequel il était à l'instant précédant et les observations faîtes sur le système à l'instant courant. Ces HMM sont particulièrement adaptés à la modélisation de processus stochastiques. Cependant, la gestion de multiples capteurs n'est possible qu'avec un prétraitement de l'ensemble des données issues des capteurs pour les transformer en une observation unique, qui sera l'observation utilisée à chaque pas de temps par le HMM. Il est difficile aussi dans ce cadre de prendre en compte des connaissances sur la dynamique du système que l'on cherche à localiser.

IV.3.2 Les modèles graphiques probabilistes

Les modèles graphiques probabilistes sont classiquement définis comme étant un mariage entre la théorie des probabilités et la théorie des graphes. Les probabilités permettent à es modèles de prendre en compte l'aspe t in certain présent dans les applications réelles. La représentation graphique des modèles offre un cadre intuitif et attractif dans de nombreux domaines d'applications où les utilisateurs ont besoin de " comprendre" le modèle qu'ils utilisent. Les modèles graphiques sont aussi connus sous le nom de réseaux de croyance, réseaux d'indépendance probabiliste ou encore réseaux bayésiens. Ils forment un cadre très efficace pour la fusion de données. Ils permettent de modéliser un problème simplement et d'utiliser des données en provenance de multiples sources, qu'elles soient qualitatives ou quantitatives. Ces données servent à mettre à jour la connaissance et les croyances que l'on a sur le problème.

IV.3.3 Filtre Kalman

Le filtre de Kalman dont on doit le nom à Rudolf Emil Kalman est un estimateur récursif. Cela signifie que pour estimer l'état courant du système : $Xk \in \Re n$, seul l'état précédent : $Xk-1$ et les mesures actuelles :$Yk \in \Re m$ sont nécessaires (et éventuellement l'entrée du filtre $Uk-1 \in \Re l$) Étant donné le système linéaire définit comme suit :

$$X_k = AX_{k-1} + BU_{k-1} + W_k \tag{IV.1}$$

$$Y_k = H_k X_k + V_k \tag{IV.2}$$

On suppose que l'on ne peut pas observer directement le système donné par (I.1), mais on dispose de l'observation$Y_K$donnée par l'équation (I.2), qui est la somme d'un signal $H_K X_K$, et d'un bruit d'observation V_K. Les variables aléatoires W_K et $V_K$représentent respectivement le bruit du modèle et le bruit des observations. Ces deux variables sont supposées indépendantes et suivent la loi normale :

$$\begin{aligned} p(W) &\sim N(0, Q) \\ p(V) &\sim N(0, R) \end{aligned} \tag{IV.3}$$

Dans la pratique, les covariances Q et R changent au cours du temps, cependant nous supposons qu'elles restent constantes. Les coefficients des équations (I.3), sont définis comme suit :$A \in \Re n \times n$,$B \in \Re n \times l$ et $H \in \Re m \times n$. On suppose également que :
• la condition initiale X_0 est gaussienne de moyenne $\bar{X_0}$ et de covariance Q_0^X,
• les bruits W_K et V_K et X_0 sont mutuellement indépendants.

Étant donné l'observation à l'instant k : $Y_{0:K} = (Y_0, Y_1, ..., Y_K)$, l'objectif du filtre de Kalman est d'estimer le vecteur aléatoire X_K à partir de $Y_{0:K}$ de façon optimale et récursive. Si on adopte le critère du minimum de variance1, il s'agit de al culer la loi conditionnelle du vecteur aléatoire X_K sa chant $Y_{0:K}$. Comme le cadre est gaussien, il suffit de al culer la moyenne donnée par :

$$p(X_K|Y_{0:K}) \sim N(\hat{X}_K, P_K) \tag{IV.4}$$

Avec :

$$\hat{X}_K = E[X_K|Y_{0:K}] \tag{IV.5}$$

Et la matrice de covariance donnée par :

$$P_k = E[(X_k - \hat{X}_K)(X_k - \hat{X}_K)^T|Y_{0:K}] \tag{IV.6}$$

On pose également :

$$p(X_K|Y_{0:K-1}) = N(\hat{X}_{\bar{K}}, P_{\bar{K}}) \tag{IV.7}$$

Avec :

$$\hat{X}_K = E[X_K|Y_{0:K-1}] \tag{IV.8}$$

Et :

$$P_{\bar{k}} = E[(X_k - \hat{X}_{\bar{K}})(X_k - \hat{X}_{\bar{K}})^T|Y_{0:K}] \tag{IV.9}$$

Les matrices P_k et $P_{\bar{k}}$ne dépendent pas des observations Y_K , C'est-à-dire :

$$\begin{aligned} P_k = E[(X_k - \hat{X}_K)(X_k - \hat{X}_K)^T|Y_{0:K}] = E[(X_k - \hat{X}_K)(X_k - \hat{X}_K)^T] \\ P_{\bar{k}} = E[(X_k - \hat{X}_{\bar{K}})(X_k - \hat{X}_{\bar{K}})^T|Y_{0:K}] = P_{\bar{k}} = E[(X_k - \hat{X}_{\bar{K}})(X_k - \hat{X}_{\bar{K}})^T] \end{aligned} \tag{IV.10}$$

Supposons maintenant connue la loi conditionnelle du vecteur $p(X_{K-1}|Y_{0:K-1})$. Pour calculer la loi conditionnelle du vecteur $p(X_K|Y_{0:K})$, on procède en deux temps :
1) Prédiction : on calcule la loi conditionnelle du vecteur $p(X_K|Y_{0:K-1})$ en utilisant l'équation (III.1),
2) Correction : on corrige la prédiction en tenant compte de la nouvelle observation Y_K donnée par l'équation (III.2).

La question qui se pose maintenant est de savoir e qu'apporte la nouvelle observation Y_k par rapport aux observations passées $Y_{0:K-1}$?

On pose :

$$e_k = Y_k - E[Y_k - |Y_{0:k-1}] \tag{IV.11}$$

D'après l'équation (III.2)

$$e_k = Y_k - H_k E[X_k - |Y_{0:k-1}] + E[V_k - |Y_{0:k-1}] + Y_k - H_k \hat{X}_{\bar{k}} \tag{IV.12}$$

Compte tenu du fait que V_K et$Y_{0:K-1}$ sont indépendants.

Le processus e_k est un processus gaussien à valeurs dans $\Re m$, appelé processus d'innovation. Les preuves de toutes ces équations peuvent être trouvées par exemple dans [22] et [12]. Ainsi le filtre de Kalman peut être donné par l'algorithme 1.

IV.3.4 Filtre de Kalman étendu

Étant donné le système non linéaire définit comme suit :

$$X_k = f_k(X_{k-1}, U_{k-1}, W_k) \tag{IV.13}$$

$$Y_k = h_k(X_K, V_k) \tag{IV.14}$$

Où X_K,Y_K, W_K et V_K sont des variables définies de la même manière que dans le filtre de Kalman classique. On suppose que les fonctions f_K et h_K sont dérivables.

Algorithme 1 Filtre de Kalman

Initialisation

$\hat{X}_{\bar{0}} = \hat{X}_0 = E[X_0]$
$P_{\bar{0}} = Q_0^X = cov(X_0)$
Prédition
$\hat{X}_{\bar{k}} = A\hat{X}_{k-1} + BU_{k-1}$
$P_{\bar{k}} = AP_{k-1}A^T + Q$
Corretion
$K_k = P_{\bar{k}}H_k^T[H_k P_{\bar{k}H_k^T+R}]^{-1ou}$ K_k est appelé le gain de Kalman.

$\hat{X}_k = \hat{X}_{\bar{k}} + K_k(Y_k - H_k\hat{X}_{\bar{k}})$
$P_k = (I - K_kH_k)P_{\bar{k}}$

Le système donné par les équations (III.13) et (III.14), ne peut pas se résoudre explicitement comme dans le as linéaire/gaussien. L'idée du filtre de Kalman étendu est de linéariser les fonctions f_K et h_K autour de l'estimée courante et d'appliquer la technique du filtre de Kalman classique.

Ainsi, on linéarise f_K autour de $(\hat{X}_{k-1}, \hat{U}_{k-1})$ et l'équation d'état s'écrit :

$$X_k = f_{k-1}(\hat{X}_{k-1}, \hat{U}_{k-1}) + \frac{\partial f}{\partial x}|\hat{X}_{K-1}, \hat{U}_{k-1}(X_{k-1} - \hat{X}_{k-1}) + \frac{\partial f}{\partial U}(|\hat{X}_{K-1}, \hat{U}_{k-1})(U_{k-1} - \hat{U}_{k-1}) + \varepsilon_k^1 \tag{IV.15}$$

De la même manière, on linéarise h_K autour de$(\hat{X}_{k-1})$, et on obtient :

$$X_k = h_k(\hat{X}_{\bar{k}}) + \frac{\partial f}{\partial x}|\hat{X}_{\bar{k}}(X_k - \hat{X}_{\bar{k}}) + \varepsilon_k^2 \tag{IV.16}$$

Notons que $\in_k^1$ et $\in_k^2$ représentent les termes d'ordre supérieur à un. Ils seront négligés par la suite. Le filtre de Kalman étendu peut être donné par l'algorithme 2. D'après [14], on peut s'attendre à de bons résultats avec cette technique de filtrage lorsque l'on est proche d'une situation linéaire ou lorsque le rapport signal/bruit est grand.

IV.3.5 Filtrage particulaire

Les méthodes fondées sur le filtrage particulaire proposent de représenter la loi conditionnelle de l'état par un nombre fini de masses pondérées. Un ensemble de points appelés particules est généré et ha une de ces particules représente un état probable du système (voir la figure 2). Les coefficients de pondération (poids) sur chaque particule sont une mesure du degré de confiance que l'on peut avoir en es dernières pour représenter effectivement l'état. Les particules évoluent suivant l'équation d'état du système (étape de prédiction) et les poids sont ajustés en fonction des observations (étape de correction).

L'idée de base du filtrage particulaire consiste donc à déterminer des approximations sous la forme :

Algorithme 2 Filtre de Kalman Étendu

Initialisation

$\hat{X}_{\bar{0}} = \hat{X}_0 = E[X_0]$
$R_0 = Q_0 = cov(X_0)$
Prédition
$\hat{X}_{\bar{k}} = f(\hat{X}_{k-1}, U_{k-1})$

$P_{\bar{k}} = A_k P_{k-1} A_k^T + B_k Q_{k-1} B_k^T + Q$
Corretion
$\hat{X}_k = \hat{X}_{\bar{k}} + K_k(Y_k - H_k(\hat{X}_{\bar{k}}))$
$P_k = (I - K_k H_k) P_{\bar{k}}$
$K_k = P_{\bar{k}} H_k^T [H_k P_{\bar{k}} H_k^T + R]^{-}1$
$Avec : A_k = \frac{\partial f}{\partial x}|_{(\hat{X}_{k-1}, \hat{U}_{k-1})}, B_k = \frac{\partial f}{\partial u}|_{(\hat{X}_{k-1}, \hat{U}_{k-1})} et H_k = \frac{\partial h}{\partial x}|_{(\hat{X}_{\bar{k}}, \hat{U}_{k-1})}$

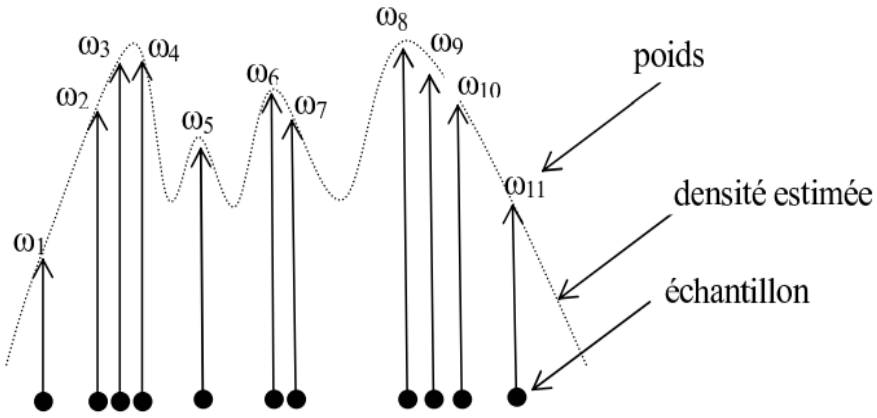


FIGURE IV.1 – Approximation d'une densité par un ensemble fini de masses pondérées

$$p(X_k|Y_{1:k-1}) \approx p^N(X_k|Y_{1:k-1}) = \sum_{i=1}^{N} \omega_{k^-}^i - \delta_{\xi_{k^-}^i}(X_k) \tag{IV.17}$$

$$p(X_k|Y_{1:k}) \approx p^N(X_k|Y_{1:k}) = \sum_{i=1}^{N} \omega_k^i - \delta_{\xi_k^i}(X_k) \tag{IV.18}$$

À chaque instant k, il s'agit don de déterminer un nombre fini de paramètres ($\omega_{k-}^{i:N}$, $\xi_{k^-}^{i:N}$) et ($\omega_k^{i:N}$, $\xi_{k^{i:N}}$). An de calculer es quantités, le filtrage particulaire procède en deux phases. Étape de prédiction : supposons connue la distribution de $p(X_{K-1}|Y_{1:K-1})$ donnée par :

$$p(X_{k-1}|Y_{1:k-1}) = \sum_{i=1}^{N} \omega_{k-1}^i \delta_{\xi_{k-1}^i}(X_{k-1}) \tag{IV.19}$$

On détermine la probabilité conditionnelle $p(X_K|Y_{1:K-1})$ en appliquant la définition suivante :

$$p(X_k|Y_{1:K-1}) = \int_{\Re^n} p(X_k|X_{k-1}) p(X_{k-1}|Y_{1:k-1}) dX_{k-1} \tag{IV.20}$$

En appliquant l'équation de Chapman-Kolmogorov [13] et [15], on obtient :

$$p(X_k|Y_{1:K-1}) = \int p(X_k|X_{k-1}, Y_{1:k-1}) p(X_{k-1}|Y_{1:k-1}) dX_{k-1} \tag{IV.21}$$

Dans [15] et [16], les auteurs donnent la démonstration de l'équation suivante :

$$p(X_k|X_{k-1}, Y_{1:K-1}) = p(X_k|X_{k-1}) \tag{IV.22}$$

Et par conséquent la probabilité conditionnelle $p(X_k|Y_{1:K-1})$ s'écrit comme suit :

$$p(X_k|Y_{1:K-1}) = \int p(X_k|X_{k-1})p(X_{k-1}|Y_{1:k-1})dX_{k-1} \tag{IV.23}$$

On remplace le terme $p(X_{K-1}|Y_{1:K-1})$ par l'équation donnée par (III.19), et on obtient :

$$\begin{aligned} p(X_k|Y_{1:K-1}) &= \sum_{i=1}^{N} \omega_{k-1}^i \int p(X_k|X_{k-1})\delta_{\xi_{k-1}^i}(X_{k-1})dX_{k-1} \\ &= \sum_{i=1}^{N} \omega_{k-1}^i \int p(X_k|X_{k-1})\delta_{\xi_{k-1}^i}(X_{k-1})dX_{k-1} \\ &= \sum_{i=1}^{N} \omega_{k-1}^i p(X_k|X_{k-1} = \xi_{k-1}^i) \end{aligned} \tag{IV.24}$$

On obtient don un mélange de lois $p(X_k|Y_{k-1} = \xi_{k-1}^i)$ qui n'est pas sous forme particulaire. Pour obtenir une approximation de type particulaire on peut échantillonner selon cette loi [13]. Une autre possibilité envisageable, consiste à utiliser :

$$p(X_k|Y_{1:K-1}) = \sum_{i=1}^{N} \omega_{k-1}^i \xi_{k-}^i(X_k) \tag{IV.25}$$

$O : \xi_{k-}^i \sim p(X_k X_{k-1} = \xi_{k-1}^i)$.

Étape de correction : à l'étape de correction, la loi de densité conditionnelle $p(X_k|Y_{1:k})$ peut être déduite à partir du filtre prédit $p(X_k|Y_{1:k-1})$ à l'étape k selon la formule de correction suivante [22] :

$$\begin{aligned} p(X_k|Y_{1:K}) &= \frac{p(Y_k|X_k)p(X_k|Y_{1:k-1})}{\int_{\Re^n} p(Y_k|X_k)p(X_k|Y_{1:k-1})dX_k} \\ &= \sum_{i=1}^{N} \frac{\omega_{k-}^i p(Y_k|X_k)}{\sum_{j=1}^{N} \int \omega_{k-}^j p(Y_k|X_k)\delta_{\xi_{k-}^j}(X_k)dX_k} \delta_{\xi_{k-}^i}(X_k) \\ &= \sum_{i=1}^{N} \frac{\omega_{k-}^i p(Y_k|X_k = \xi_{k-}^i}{\sum_{j=1}^{N} \omega_{k-}^j p(Y_k|X_k = \xi_{k-}^j)} \delta_{\xi_{k-}^i}(X_k) \end{aligned} \tag{IV.26}$$

Ainsi, l'approximation de la densité conditionnelle $p(X_k|Y_{1:k})$ est donnée par :

$$p(X_k|Y_{1:k}) \approx \sum_{i=1}^{N} \omega_k^i \delta_{\xi_{k-}^i}(X_k) \tag{IV.27}$$

Avec :

$$\omega_k^i = \frac{\omega_{k-}^i p(Y_k|X_k = \xi_{k-}^i)}{\sum_{j=1}^N \omega_{k-}^j p(Y_k|X_k = \xi_{k-}^j)} \tag{IV.28}$$

L'algorithme 3 résume l'aspect fonctionnel du filtrage particulaire :

Algorithme 3 Filtrage Particulaire

Prédition

$\mathrm{p}(\mathrm{X}_k|Y_{1:k-1}) \approx \sum_{i=1}^N \omega_{k-}^i \delta_{\xi_{k-}^i}$

Ou

$\omega_{k-}^i = \omega_{k-1}^i$

$\xi_{k-}^i \sim p(X_k|X_{k-1} = \xi_{k-1}^i)$

Corretion

$\mathrm{p}(\mathrm{X}_k|Y_{1:k}) \approx \sum_{i=1}^N \omega_k^i \delta_{\xi_{k-}^i}(X_k)$

Ou

$\omega_k^i = \frac{\omega_{k-}^i p(Y_k|X_k = \xi_{k-}^i)}{\sum_{j=1}^N \omega_{k-}^j p(Y_k|X_k = \xi_{k-}^j)}$

$\xi_k^i = \xi_{k-}^i$

Le filtre particulaire tel que donné ci-dessus présente un défaut important : les poids des particules ont tendance à diverger de sorte que, après un certain nombre de mesures, la plupart des particules ont un poids négligeable. Ce phénomène est connu sous le nom de dégénérescence des poids. Le système de particules est appauvri et don ne peut plus représenter correctement la densité conditionnelle. Afin de remédier à e problème, plusieurs méthodes ont été proposées dans le but de régulariser les poids. Idéalement les poids doivent tous rester proches de 1/N , i.e. les particules sont d'égale importance dans l'approximation. Considérons le critère suivant :

$$N_k^{eff} = \frac{1}{\sum_{i=1}^N (\omega_k^i)^2} \in [1, N] \tag{IV.29}$$

Qui représente le nombre efficace de particules. Lorsque N_k^{eff} est proche de N alors les particules sont d'égale importance. Il y a dégénérescence des poids lorsque N_k^{eff} est proche de 1 (contribution négligeable de es poids dans l'approximation). Un second critère de rééchantillonnage a été proposé par [19]. Ce critère est fondé sur le calcul de l'entropie des pondérations. Pour éviter le phénomène de divergence du filtre, une nouvelle étape dite de ré échantillonnage est introduite. L'idée est de dupliquer les particules de poids fort et d'éliminer les particules de poids faible. Les principales méthodes de sélection de particules comme le résume [13] sont le tirage multinomial, le ré échantillonnage résiduel, le ré échantillonnage déterministe. L'algorithme de filtrage particulaire est réalisé par l'algorithme 4. Notons que plusieurs optimisations de et algorithme existent dans la

littérature.

Algorithme 4 Filtre Partiulaire générique

Initialisation

$\xi^{1:N} \sim p(X_0)$

$\omega^{1:N} \longleftarrow 1/N$

$Pour(k = 1; k \leq NB_{iteration})Faire$

$\tilde{\xi^i} \sim p(X_k|X_{k-1} = \xi^i)pouri = 1 : Nchantillonage$

$\tilde{\omega^i} \longleftarrow \omega^i p(X_k|X_k = \xi^i)pouri = 1 : Nmisejourdespoids$

$\tilde{\omega^i} \longleftarrow \frac{1}{sum(\tilde{\omega}^{1:N})}pouri = 1 : Nnormalisationdespoids$

$N_{eff} \longleftarrow \frac{1}{\sum_{i=1}^{N}(\tilde{\omega}^i)^2}$

$Si(N_{eff}/N \leq seuil)Alors$

$\xi^{1:N} \longleftarrow rchantillonner(\tilde{\omega}^{1:N}; \tilde{\xi}^{1:N})$

$\omega^{1:N} \longleftarrow 1/N$

Si non

$\xi^{1:N} \longleftarrow \tilde{\xi}^{1:N})$

$\omega^{1:N} \longleftarrow \tilde{\omega}^{1:N}$

Fin si

$Sortie(\xi^{1:N;\omega^{1:N}})$

IV.4 CONCLUSION

Dans ce chapitre nous avons essayé de réaliser une étude bibliographique générale sur la fusion de donnée et par la suite, nous avons montré les différentes approches classiques pour effectuer la fusion de données, en rappelant quelques filtres et d'algorithme qui peuvent être appliqués dans la fusion de donnée.

Chapitre V

IMPLEMENTATION EXPERIMENTAL

V.1 Introduction

Ce chapitre est dédié à la mise en œuvre du projet sur site en testant sur des objets pour déterminer et comprendre son efficacité. Tout d'abord, nous avons commencé par une expérience simple dans laquelle nous avons expliqué le fonctionnement du flux optique, puis nous les avons mesurés, mesuré la distance, la rotation, l'accélération et déterminé la cible.

V.2 Principaux composante de projet

V.2.1 Raspberry pi3 (modèle B+)

De nombreux produits peuvent être utilisés. Mais nous avons choisi ce produit raspberry pi3 pour plusieurs avantages qui répondent aux exigences des robotique, comme sa petite taille et sa puissance, et il possède plusieurs entrées et sorties. Facilité d'utilisation par rapport aux autres technologies industrielles et développement rapide de ce game de produits

V.2.2 Définition

Le Raspberry pi est un nano ordinateur de la taille d'une carte de crédit que l'on peut brancher à un écran et utilisé comme un ordinateur standard. Sa petite taille, et son prix intéressant fait du Raspberry pi un produit idéal pour tester différentes choses, et notamment utilisé même dans les projets de recherche. Évidemment, pour sa taille il ne faut pas s'attendre à des performances incroyables, mais pour mettre en ligne des projets à montrer au client ou expérimenter avec linux c'est largement suffisant.

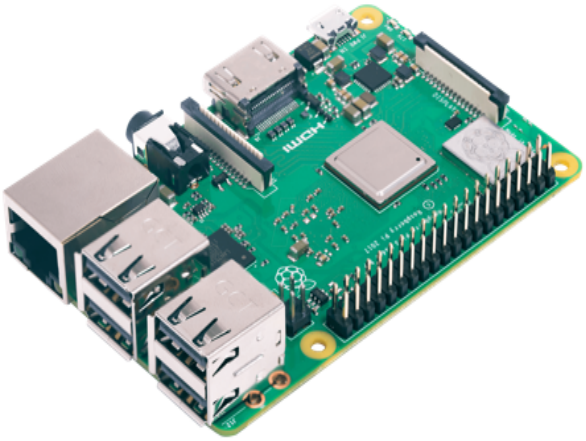


FIGURE V.1 – Raspberry pi3 (modèle B+)

V.2.3 Spécification

Le Raspberry Pi 3 Model B + est la dernière révision de la gamme Raspberry Pi 3.

— Broadcom BCM2837B0, SoC Cortex-A53 (ARMv8) 64 bits à 1,4 GHz
— SDRAM LPDDR2 1 Go
— LAN sans fil IEEE 802.11.b / g / n / ac 2,4 GHz et 5 GHz, Bluetooth 4.2, BLE
— Gigabit Ethernet sur USB 2.0 (débit maximal 300 Mbps)
— Embase GPIO 40 broches étendue
— HDMI pleine grandeur
— 4 ports USB 2.0
— Port de caméra CSI pour connecter une caméra Raspberry Pi
— Port d'affichage DSI pour connecter un écran tactile Raspberry Pi
— Sortie stéréo 4 pôles et port vidéo composite
— Port Micro SD pour charger votre système d'exploitation et stocker des données
— Entrée d'alimentation CC 5 V / 2,5 A
— Prise en charge Power-over-Ethernet (PoE) (nécessite un HAT PoE séparé)

V.3 Les accessoires indispensable (ou très utiles)

V.3.1 La caméra de Raspberry pi

V.3.2 Définition

Le module de caméra est une coutume conçue add-on pour Raspberry Pi. Il se fixe à Raspberry Pi par le biais de l'une des deux petites prises sur la surface supérieure de la carte. Cette interface utilise l'interface CSI dédié, qui a été spécialement conçu pour s'interfacer à des caméras. Le bus CSI est capable de débits de données très élevés, et il exerce exclusivement les données de pixels. Raspberry Pi PAS INCLUS.

Le conseil d'administration lui-même est minuscule, à environ 25mm x 20mm x 9mm. Il pèse également un peu plus de 3g, le rendant parfait pour des applications mobiles ou autres dont la taille et le poids sont importants. Il se connecte à Raspberry Pi au moyen d'un câble ruban court. La caméra est reliée au processeur BCM2835 sur la Pi par l'intermédiaire du bus CSI, une liaison de largeur de bande supérieure, qui porte les données de pixels de la caméra vers le processeur. Ce bus se déplace le long du câble ruban qui relie le bord de l'appareil photo à la Pi.

Le capteur lui-même a une résolution native de 5 mégapixels, et dispose d'une lentille de focalisation à bord fixe. En termes d'images fixes, l'appareil est capable de 2592 x 1944 pixels des images statiques, et prend également en 1080p30, 720p60 et 640x480p60/90 vidéo. L'appareil photo est prise en charge dans la version la plus avancée de Raspbian, système d'exploitation préféré de Raspberry Pi.

Raspberry Pi 3 appareil photo Vision nocturne avec 150 degrés grand angle fisheye 5 Mégapixels 1080p 2 lampe de poche LED module de caméra pour Raspberry Pi 2.

V.3.3 Caractéristiques

— Raspberry Pi module de caméra de vision nocturne.
— Raspberry Pi caméra
— Capteur ov5647 5 mégapixels
— Taille du CCD : 1/10,2 cm
— Meilleure résolution du capteur : 1080p
— 4 trous de vis
— Utilisés pour la fixation
— Longueur focale : 2,1
— Diagonale angle : 130 degrés
— Fournit une alimentation 3.3 V de sortie
— Liste d'emballage : Raspberry Pi caméra à vision nocturne × 1 pcs Lampe torche LED × 2 PCs FCC 15 broches câble × 1 pcs

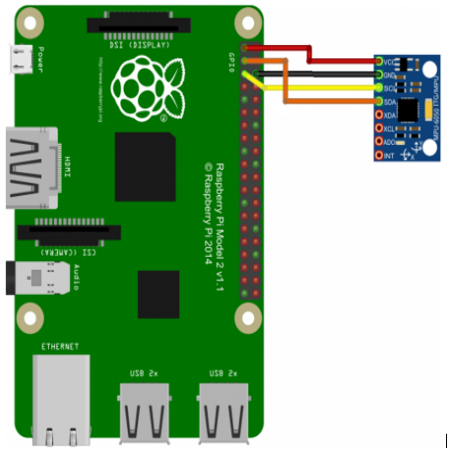
FIGURE V.2 – La caméra de Raspberry pi

V.3.4 Définition d'Opencv

OpenCV (Open Computer Vision) est une bibliothèque graphique, initialement développée par Intel. Elle est spécialisée dans le traitement d'images en temps réel, que ce soit pour de la photo ou de la vidéo. Sa première version est sortie en juin 2000. Elle est disponible sur la plupart des systèmes d'exploitation et existe pour les langages Python, C++ et Java. Depuis 2016 et le rachat de ItSeez par Intel, le support est de nouveau assuré par Intel.

FIGURE V.3 – Opencv

V.4 Test de capteur



Raspberry Pi	MPU 6050
Pin 1 (3.3V)	VCC
Pin 3 (SDA)	SDA
Pin 5 (SCL)	SCL
Pin 6 (GND)	GND

V.4.1 Le programme

```
import smbus            #import SMBus module of I2C
from time import sleep          #import

#some MPU6050 Registers and their Address
PWR_MGMT_1   = 0x6B
SMPLRT_DIV   = 0x19
CONFIG       = 0x1A
GYRO_CONFIG  = 0x1B
INT_ENABLE   = 0x38
ACCEL_XOUT_H = 0x3B
ACCEL_YOUT_H = 0x3D
ACCEL_ZOUT_H = 0x3F
GYRO_XOUT_H  = 0x43
GYRO_YOUT_H  = 0x45
GYRO_ZOUT_H  = 0x47


def MPU_Init() :
    #write to sample rate register
    bus.write_byte_data(Device_Address, SMPLRT_DIV, 7)

    #Write to power management register
    bus.write_byte_data(Device_Address, PWR_MGMT_1, 1)

    #Write to Configuration register
    bus.write_byte_data(Device_Address, CONFIG, 0)

    #Write to Gyro configuration register
    bus.write_byte_data(Device_Address, GYRO_CONFIG, 24)

    #Write to interrupt enable register
    bus.write_byte_data(Device_Address, INT_ENABLE, 1)

def read_raw_data(addr) :
    #Accelero and Gyro value are 16-bit
        high = bus.read_byte_data(Device_Address, addr)
        low = bus.read_byte_data(Device_Address, addr+1)

        #concatenate higher and lower value
        value = ((high << 8) | low)

        #to get signed value from mpu6050
        if(value > 32768) :
                value = value - 65536
        return value


bus = smbus.SMBus(1)    # or bus = smbus.SMBus(0) for older version boards
Device_Address = 0x68   # MPU6050 device address

MPU_Init()

print («  Reading Data of Gyroscope and Accelerometer »)

while True :

    #Read Accelerometer raw value
```

```
    acc_x = read_raw_data(ACCEL_XOUT_H)
    acc_y = read_raw_data(ACCEL_YOUT_H)
    acc_z = read_raw_data(ACCEL_ZOUT_H)

    #Read Gyroscope raw value
    gyro_x = read_raw_data(GYRO_XOUT_H)
    gyro_y = read_raw_data(GYRO_YOUT_H)
    gyro_z = read_raw_data(GYRO_ZOUT_H)

    #Full scale range +/- 250 degree/C as per sensitivity scale factor
    Ax = acc_x/16384.0
    Ay = acc_y/16384.0
    Az = acc_z/16384.0

    Gx = gyro_x/131.0
    Gy = gyro_y/131.0
    Gz = gyro_z/131.0


    print (« Gx=%.2f » %Gx, u'\u00b0'+ « /s », « \tGy=%.2f » %Gy, u'\u00b0'+
« /s », « \tGz=%.2f » %Gz, u'\u00b0'+ « /s », « \tAx=%.2f g » %Ax,
« \tAy=%.2f g » %Ay, « \tAz=%.2f g » %Az)
    sleep(1)
```

V.4.2 Résultats de simulation

```
Python 3.7.3 (/usr/bin/python3)
>>> %cd /home/pi/Desktop
>>> %Run MPU.py
 Reading Data of Gyroscope and Accelerometer
Gx=0.00 °/s      Gy=0.00 °/s      Gz=0.79 °/s      Ax=0.00 g
      Ay=0.00 g   AZ=0.00 g
Gx=-0.71 °/s     Gy=1.94 °/s      Gz=0.10 °/s      Ax=-0.07 g Ay=-
0.01 g      AZ=0.93 g
Gx=-0.73 °/s     Gy=-0.02 °/s     Gz=0.10 °/s      Ax=-0.08 g Ay=-
0.01 g      AZ=0.93 g
Gx=-0.73 °/s     Gy=-0.03 °/s     Gz=0.09 °/s      Ax=-0.08 g Ay=-
0.01 g      AZ=0.93 g
Gx=-0.71 °/s     Gy=-1.95 °/s     Gz=0.08 °/s      Ax=-0.08 g Ay=-
0.01 g      AZ=0.92 g
Gx=-0.71 °/s     Gy=-0.05 °/s     Gz=0.08 °/s      Ax=-0.08 g Ay=-
0.01 g      AZ=0.93 g
Gx=-0.71 °/s     Gy=1.95 °/s      Gz=0.11 °/s      Ax=-0.08 g Ay=-
0.01 g      AZ=0.92 g
```

V.5 LOCALISATION PAR LE FLUX OPTIQUE

V.5.1 Détecter et suivi les coins de l'objet par Le flux optique

V.5.2 Test du flux optique

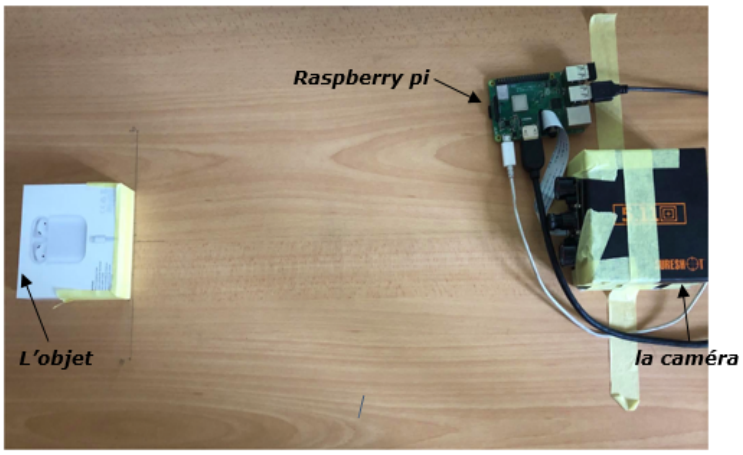


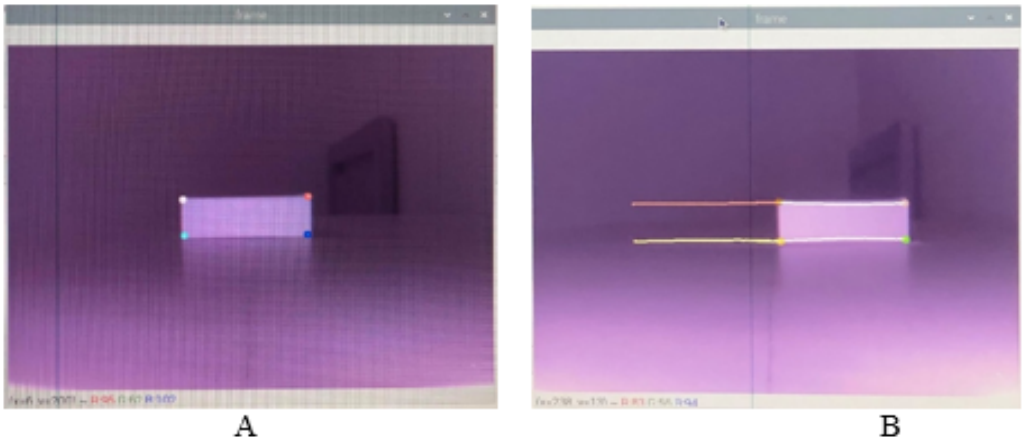
FIGURE V.4 – la réalisation et le test

V.5.3 Le programme

```
import numpy as np
import cv2

cap = cv2.VideoCapture(0)

# paramètres pour la détection des coins ShiTomasi
feature_params = dict( maxCorners = 100,
                       qualityLevel = 0.3,
                       minDistance = 7,
                       blockSize = 7 )

# Paramètres pour le flux optique lucas kanade
lk_params = dict( winSize  = (15,15),
                  maxLevel = 2,
                  criteria = (cv2.TERM_CRITERIA_EPS |
cv2.TERM_CRITERIA_COUNT, 10, 0.03))

# Créez des couleurs aléatoires
color = np.random.randint(0,255,(100,3))

# Prenez la première image et trouvez-y les coins
ret, old_frame = cap.read()
```

```
old_gray = cv2.cvtColor(old_frame, cv2.COLOR_BGR2GRAY)
p0 = cv2.goodFeaturesToTrack(old_gray, mask = None, **feature_params)

# Créer une image de masque à des fins de dessin
mask = np.zeros_like(old_frame)

while(1):
    ret,frame = cap.read()
    frame_gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # calcule le flux optique
    p1, st, err = cv2.calcOpticalFlowPyrLK(old_gray, frame_gray, p0, None,
**lk_params)

    # Sélectionnez les bons points
    good_new = p1[st==1]
    good_old = p0[st==1]

    # dessine les traces
    for i,(new,old) in enumerate(zip(good_new,good_old)):
        a,b = new.ravel()
        c,d = old.ravel()
        mask = cv2.line(mask, (a,b),(c,d), color[i].tolist(), 2)
        frame = cv2.circle(frame,(a,b),5,color[i].tolist(),-1)
    img = cv2.add(frame,mask)

    cv2.imshow('frame',img)
    k = cv2.waitKey(30) & 0xff
    if k == 27:
        break

    # Maintenant, mettez à jour l'image précédente et les points précédents
    old_gray = frame_gray.copy()
    p0 = good_new.reshape(-1,1,2)

cv2.destroyAllWindows()
cap.release()
```

V.5.4 Les résultats de simulation



A B

FIGURE V.5 – les résultats de programme de flux optique

On remarque que la caméra a capturé quatre points sur l'objet, et c'est ce qu'il faut, puis quand on a déplacé cet objet, on a vu que la caméra, ou plus précisément, le flux optique, avait tracé la trajectoire du mouvement de cet objet.

V.5.5 La distance de x, y et z par le flux optique

a- Trouver la distance de x en cm

Pour calculer le déplacement en x, deux facteurs sont nécessaires. Tout d'abord, lorsque $z = 0$, le facteur du mappage entre le déplacement du centre de l'objet dans le monde réel et dans l'image à l'aide des pixels est déterminé. Deuxièmement, le facteur de changement des dimensions de l'objet lorsque z change est détermine (c'est un facteur dynamique, il change pour chaque z). Les deux facteurs sont multipliés par le déplacement du centre pour calculer le déplacement de l'objet sur les axes x.

b- Le programme

```
#La calb est pour la calibration du l'objet sur l'axe x et y quelque soit z
    dcu = sqrt((cux2-cux1)**2+(cuy2-cuy1)**2)
    dco = sqrt((cx2-cx1)**2+(cy2-cy1)**2)
    calb = dcu/dco
# Les dimension d'objet
    ob1 = 9.5
#fx pour la calibration du l'objet sur l'axe x et y a chaque itération
    fx = d1/ob1   #pour x
    dx=ox-ux
    x = dx/fx
```

c- Résultats de simulation

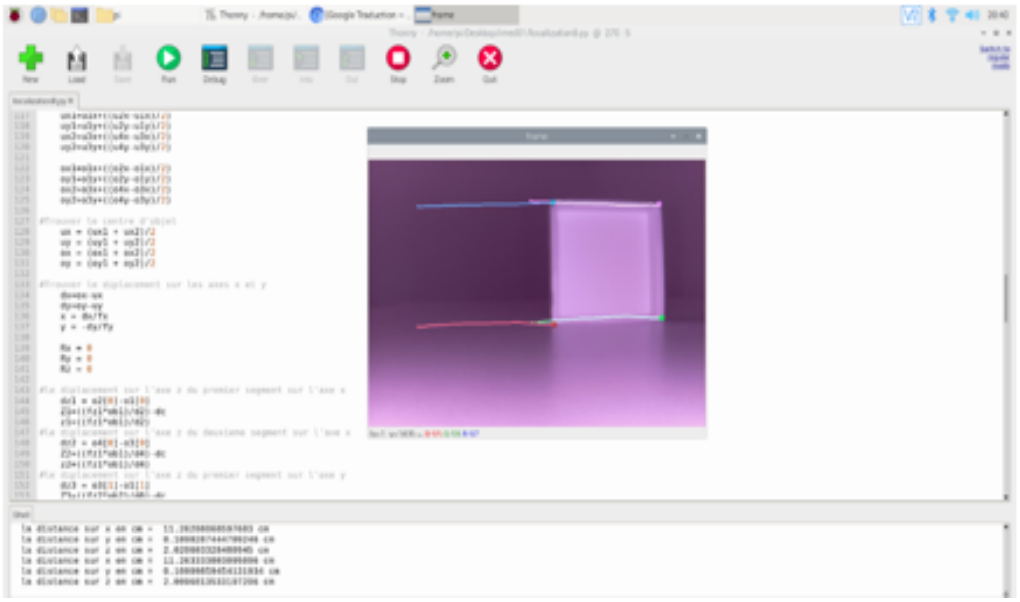


FIGURE V.6 – Le déplacement sur x.

$$\begin{cases} \textbf{La distance de } x \textbf{ en } cm = 11.262887283614107cm. \\ \textbf{La distance de } y \textbf{ en } cm = 0.10904319213753424cm. \\ \textbf{La distance de } z \textbf{ en } cm = 2.0096813530004210cm. \end{cases}$$

Dans ce cas, J'ai déplacé cet objet sur une distance de 11 cm sur l'axe des x, on remarque que le flux optique permet de calculer et mesurer avec précision la distance parcourue par cet objet, et c'est ce que montrent les images et les résultats. **Remarque :** Le signe négatif indique la direction opposée

V.5.6 Trouver la distance de y en cm

Pour le déplacement de l'objet selon l'axe y, c'est le même principe du déplacement de l'objet sur l'axe x.

a- Le programme

```
#La calb est pour la calibration du l'objet sur l'axe x et y quelque soit z
    dcu = sqrt((cux2-cux1)**2+(cuy2-cuy1)**2)
    dco = sqrt((cx2-cx1)**2+(cy2-cy1)**2)
    calb = dcu/dco
# les dimension d'objet
    ob2 = 9.5
# fy pour la calibration du l'objet sur l'axe x et y a chaque itération
    fy = d5/ob2   #pour y
    dy=oy-uy
    y = -dy*(calb/fy)
```

b- Résultats de simulation

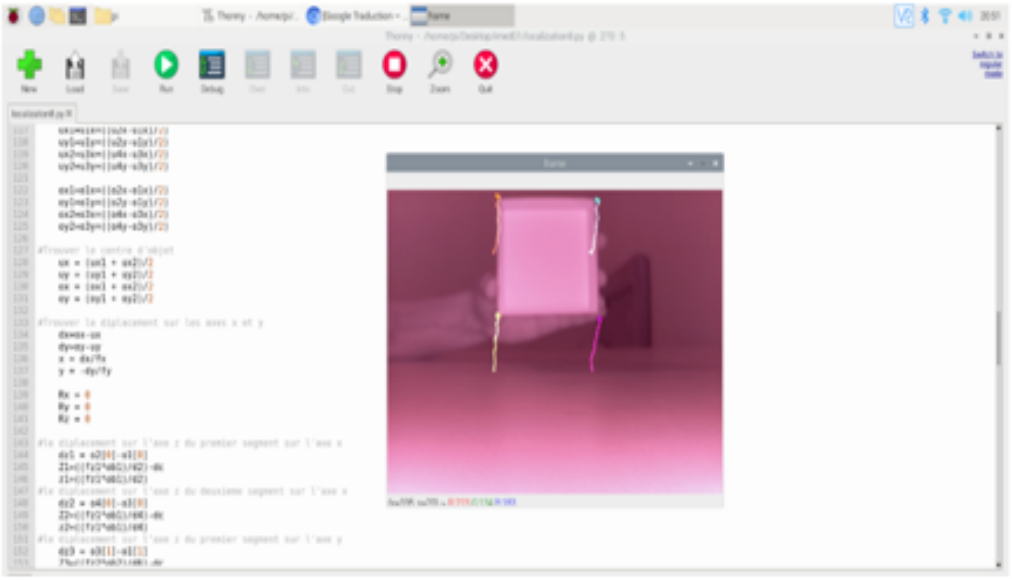


FIGURE V.7 – Le déplacement sur y

$$\begin{cases} \textbf{La distance de } x \textbf{ en } cm = 0.36975829760577805cm. \\ \textbf{La distance de } y \textbf{ en } cm = 10.315595431427301cm. \\ \textbf{La distance de } z \textbf{ en } cm = -1.7364140834904447cm. \end{cases}$$

Dans ce cas, J'ai déplacé cet objet sur une distance de 10 cm sur l'axe des y, nous constatons qu'il y a une perturbation causées par la façon dont l'objet a été transporté, nous avons donc trouvé les résultats satisfaisants et acceptables.

V.6 Trouver la distance de z en cm

Pour calculer le déplacement sur z, nous devons calibrer la caméra. Nous devons d'abord connaître la valeur initiale de la caméra à partir de l'objet et la longueur des bords de l'objet. Puis déduire le nombre de pixels des bords et trouver la distance focale qui sera utilisée plus tard avec uniquement les informations des pixels des bords pour calculer la valeur mise à jour de z.

a- Le programme

```
#fz1 et fz2 sont pour la calibration du l'objet sur l'axe z
    fz1 = (d1*dc)/ob1   #pour Rx et z
    fz2 = (d5*dc)/ob2   #pour Ry et z
    Rx = 0
    Ry = 0
    Rz = 0
#le déplacement sur l'axe z du premier segment sur l'axe x
    dz1 = o2[0]-o1[0]
    Z1=((fz1*ob1)/d2)-dc
    z1=((fz1*ob1)/d2)
#le déplacement sur l'axe z du deuxième segment sur l'axe x
    dz2 = o4[0]-o3[0]
    Z2=((fz1*ob1)/d4)-dc
    z2=((fz1*ob1)/d4)
#le déplacement sur l'axe z du premier segment sur l'axe y
    dz3 = o3[1]-o1[1]
    Z3=((fz2*ob2)/d6)-dc
    z3=((fz2*ob2)/d6)
#le déplacement sur l'axe z du deuxième segment sur l'axe y
    dz4 = o4[1]-o2[1]
    Z4=((fz2*ob2)/d8)-dc
    z4=((fz2*ob2)/d8)
```

b- Résultats de simulation

$$\begin{cases} \textbf{La distance de } x \textbf{ en } cm = -0.73778133771391711cm. \\ \textbf{La distance de } y \textbf{ en } cm = -0.1266069753321641cm. \\ \textbf{La distance de } z \textbf{ en } cm = 9.217537811913602cm. \end{cases}$$

Il en va de même pour la troisième tentative, qui était sur l'axe z, où l'on constate que les résultats sont très proches.

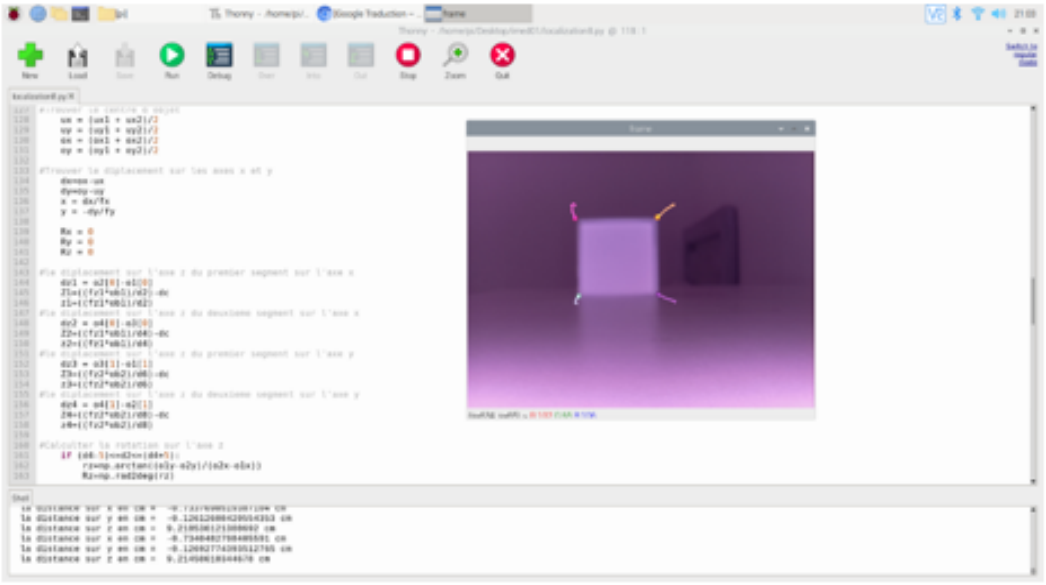


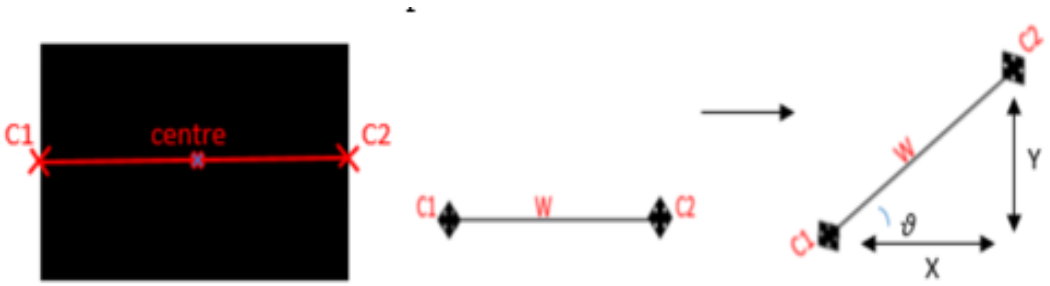
FIGURE V.8 – Le déplacement sur z

V.7 La rotation en x, y et z par le flux optique

V.7.1 Trouver la rotation de z en degré

Pour calculer la rotation dans l'axe z, les valeurs des deux points ($c1$ et $c2$) de la ligne horizontale passant par le centre de l'objet sont déduites. Ensuite, en appliquant la trigonométrie, la rotation est calculée comme suit :$R_z = arctan(\frac{cy1-cy2}{cx2-cx1})$

Là on a besoin seulement deux points



Orientation sur z :$R_z = tan^{-1}\frac{Y}{X}$

a- Le programme

```
        cx1= (o1x+o3x)/2
        cy1= (o1x+o3x)/2
        cx2= (o2x+o4x)/2
        cy2= (o2x+o4x)/2
#Calculer la rotation sur l'axe z
        rz=np.arctan((cy1-cy2)/(cx2-cx1))
        Rz=np.rad2deg(rz)
```

b- Résultats de simulation

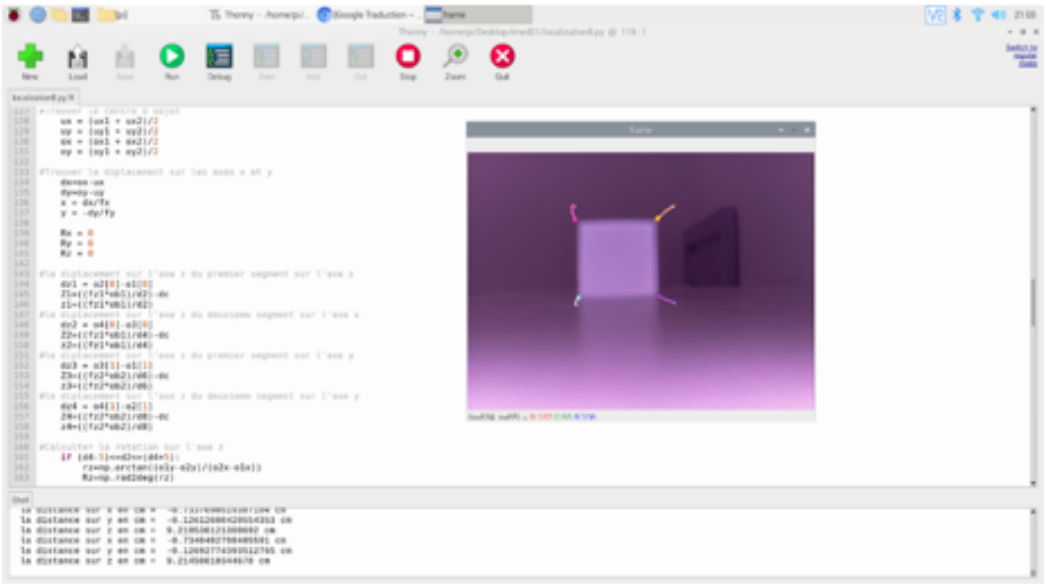


FIGURE V.9 – la rotation sur z

La rotation sur x en degré = 3.4770531977471675°.
La rotation sur y en degré = -0 °.
La rotation sur z en degré = -27.17329 °
.

Dans cette étape, nous avons tourné cette objet sur l'axe Z de 40 degrés, où nous avons remarqué que le résultat était très proche, et nous avons remarqué aussi qu'il y avait des perturbations sur chacun de nous les axes x et y, et nous trouvons cette normale en raison de la façon dont cette chose tourne.

Remarque : Le signe négatif indique la direction opposée

V.7.2 Trouver la rotation de x en degré

Lorsque l'objet tourne autour de l'axe x, le déplacement du bord supérieur de l'objet est différent du déplacement du bord inférieur. En calculant la différence de déplacement (dZ) des bords et en connaissant la longueur de l'objet (ob1), l'angle de rotation (R_x) est déduit en utilisant la trigonométrie comme suit : $R_x = arcsin\frac{dZ}{ob1}$

Les différents scénarios des positions des bords de l'objet lors de la rotation sont considérés dans le programme.
Là on a besoin des quatre points

L'orientation sur x

a- Le programme

```
#Le calcule du diplacement sur l'axe z et la rotation sur l'axe x

    if (Z1 >0 and Z2<0) or (Z1<0 and Z2>0):
        if Z1 >0 and Z2<0 :
            dZ=Z1-Z2
            rx=np.arcsin(dZ/ob1)
            Rx=np.rad2deg(rx)
            Z=Z2+(dZ/2)
        elif Z1<0 and Z2>0 :
            dZ=Z1-Z2
            rx=np.arcsin(dZ/ob1)
            Rx=np.rad2deg(rx)
            Z=Z2+(dZ/2)
    else :
        if abs(Z1) > abs(Z2)+1:
            dZ=Z1-Z2
            rx=np.arcsin(dZ/ob1)
            Rx=np.rad2deg(rx)
            Z=Z2+(dZ/2)
        elif abs(Z2) > abs(Z1)+1:
            dZ=Z2-Z1
            rx=-np.arcsin(dZ/ob1)
```

```
            Rx=np.rad2deg(rx)
            Z=Z1+(dZ/2)
        elif abs(Z1)-1 <= abs(Z2) <= abs(Z1)+1:
            if Z1 >= Z2:
                dZ=Z1-Z2
                rx=np.arcsin(dZ/ob1)
                Rx=np.rad2deg(rx)
                Z=0
            elif Z2 >= Z1:
                dZ=Z2-Z1
                rx=-np.arcsin(dZ/ob1)
                Rx=np.rad2deg(rx)
                Z=0
        elif (Z1-1 <= Z2 <= Z1+1) :
            Ry=0
            Z=0
```

b- Résultats de simulation

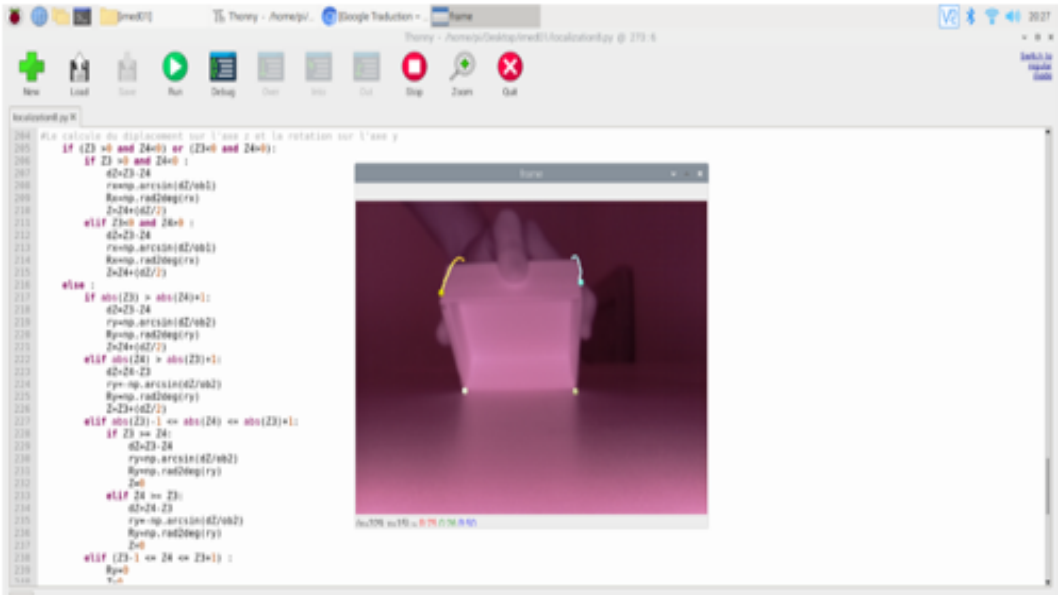


FIGURE V.10 – la rotation sur x

La rotation sur x en degré = -29.86419844591053 °.
La rotation sur$y$ en degré = -3.262653599888042 °.
La rotation sur z en degré = 0 °.

À ce stade, nous avons fait une rotation de 30 degrés sur l'axe des x et avons constaté que les résultats étaient très proches, avec une certaine interférence sur l'axe des y.

V.8 Trouver la rotation de y en degré

Lorsque l'objet tourne autour de l'axe y, le déplacement du bord droit de l'objet est différent du déplacement du bord gauche. En calculant la différence de déplacement (dZ) des bords et en connaissant la largeur de l'objet ($ob2$), l'angle de rotation (R_y) est déduit

en utilisant la trigonométrie comme suit : $R_y = arcsin\frac{dZ}{ob2}$

Aussi les différents scénarios des positions des bords de l'objet lors de la rotation sont considérés dans le programme.
Là on a besoin des quatre points

L'orientation sur y

a- Le programme

```
#Le calcule du diplacement sur l'axe z et la rotation sur l'axe y
    if (Z3 >0 and Z4<0) or (Z3<0 and Z4>0):
        if Z3 >0 and Z4<0 :
            dZ=Z3-Z4
            ry=np.arcsin(dZ/ob2)
            Ry=np.rad2deg(ry)
            Z=Z4+(dZ/2)
        elif Z3<0 and Z4>0 :
            dZ=Z3-Z4
            ry=np.arcsin(dZ/ob2)
            Ry=np.rad2deg(ry)
            Z=Z4+(dZ/2)
    else :
        if abs(Z3) > abs(Z4)+1:
            dZ=Z3-Z4
            ry=np.arcsin(dZ/ob2)
            Ry=np.rad2deg(ry)
            Z=Z4+(dZ/2)
        elif abs(Z4) > abs(Z3)+1:
            dZ=Z4-Z3
            ry=-np.arcsin(dZ/ob2)
            Ry=np.rad2deg(ry)
            Z=Z3+(dZ/2)
        elif abs(Z3)-1 <= abs(Z4) <= abs(Z3)+1:
            if Z3 >= Z4:
                dZ=Z3-Z4
                ry=np.arcsin(dZ/ob2)
                Ry=np.rad2deg(ry)
```

```
                Z=0
            elif Z4 >= Z3:
                dZ=Z4-Z3
                ry=-np.arcsin(dZ/ob2)
                Ry=np.rad2deg(ry)
                Z=0
        elif (Z3-1 <= Z4 <= Z3+1) :
            Ry=0
            Z=0
    if Ry > Rx :
        Z=Z3
    elif Rx > Ry :
        Z=Z1
```

b- Résultats de simulation

FIGURE V.11 – la rotation sur y

La rotation sur x en degré = -3.685527508644382 °.
La rotation sur y en degré = 38.02746742756542 °.
La rotation sur z en degré = 0 °.

Dans la dernière expérience, nous avons tourné de 40 degrés sur l'axe y, et le résultat était satisfaisant car il est difficile de déplacer l'objet sur l'axe y de manière correcte à 100%. Pour cette raison, nous avons remarqué que l'objet est également le long de l'axe x Bougez, mais le mouvement est très court.

V.9 Système de localisation

Ce système de localisation développé peut également être utilisé de manière inversée. En prenant comme référence les objets dans la vue de champ de la caméra du système—considérons que cet objet est une référence fixe, puis en utilisant l'outil de flux optique, lorsque le véhicule de la caméra se déplace, du point de vue de la caméra, il semble que l'objet bouge (cela dépend donc du point de vue). Lorsque l'on connaît l'état initial de l'objet à partir de la caméra et de son environnement, puis lorsque le véhicule se

déplace, l'objet est suivi (notre système développé basé sur le flux optique peut suivre le déplacement en x, y et z et la rotation le long de x, y et axe z de l'objet de référence), et en utilisant la localisation inverse, le véhicule est localisé dans son environnement à l'aide du système développé et de l'état initial de l'objet de référence. Le système développé peut être adapté à différents types de véhicules. Cela peut inclure des véhicules volants ou des véhicules à roues, le système fonctionne tant que nous connaissons les états initiaux.

En d'autres termes, lorsque nous connaissons l'emplacement de l'objet de référence dans son environnement et que nous connaissons l'emplacement de la caméra à partir de l'objet de référence à l'état initial, nous pouvons connaître l'emplacement de la caméra dans l'environnement.

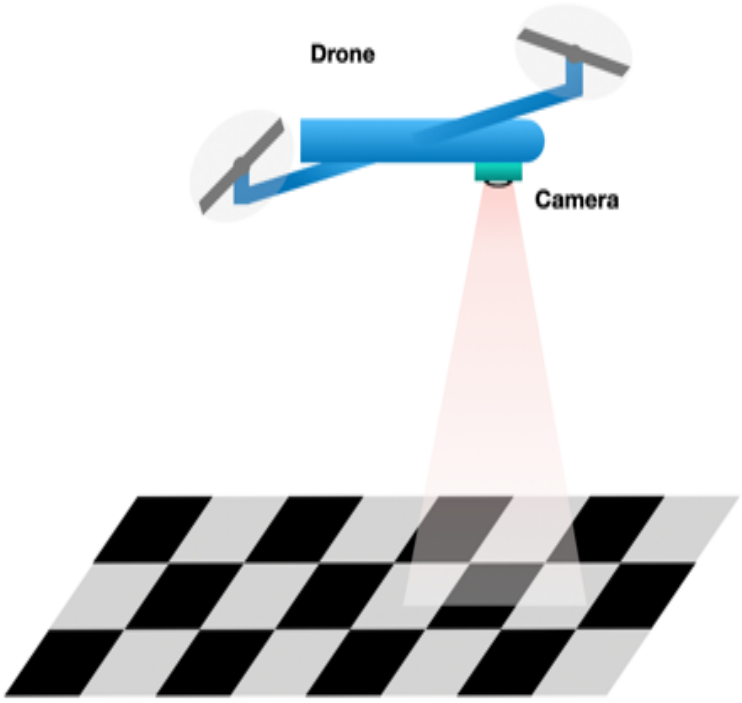


FIGURE V.12 – Localisation

V.10 Conclusion

Dans ce chapitre nous avons réuni l'ensemble des différentes parties du projet en réalisant une localisation avec vision par l'utilisation d'une caméra et un Raspberry Pi. A l'aide d'un OpenCV basé sur le flux optique pour le traitement d'image, nous avons créé des méthodes pour déduire les différentes variables de localisation, et décrire les différentes parties du programme pour réaliser la localisation des objets dans l'environnement et les testés sur un objet cubique.

A la fin, les résultats de la localisation obtenus sont conformes aux mesures faites en réalité avec une bonne précision.

CONCLUSION GÉNÉRALE

Dans ce projet nous avons réalisé un système de localisation basé sur la vision. Nous avons axé nos efforts dans un premier temps à soulever la problématique du flux optique qui consiste à détecter et suivre les coins des objets, dans un second temps, nous avons traité le problème de la localisation par le flux optique. Nous avons commencé par la présentation du flux optique, ces différentes méthodes de calcul en utilisant la méthode de Lucas-Kanade dans OpenCV.

Ensuite, nous avons parlé des capteurs inertiels en générale, et le modèle $MPU6050$ en particulier en expliquant leur principe de fonctionnement. Puis, nous avons exploré la fusion des données, leurs qualifications, bénéfices attendus et aussi les filtres, algorithme pouvant être appliqués dans la fusion de donnée dans ce projet. Finalement, nous avons montré le fonctionnement de notre système et expliquer que les points détectés des coins des objets sont comparés à chaque image du dans l'enregistrement de la caméra.

Les informations de la déformation de ces points nous permettent de faire l'association des données et les opérations de trigonométrie pour extraire avec une bonne précision les informations de la position dans les trois axes X,Y, et Z et aussi la rotation autour des trois axes, le roulis, le lacet et le tangage du dispositif incorporant le système.

Pour démonter le fonctionnement du système on a utilisé une **Raspberry Pi** connectée à une caméra pour l'observation de l'enivrement et un ordinateur pour recevoir et afficher les résultats, **Raspberry Pi** utilise l'OpenCV pour traiter les images et faire toutes les opérations nécessaires pour la localisation à haute performance, ce qui rend le système compact, prend peu d'espace et être capable de s'intégrer dans plusieurs dispositifs comme les drones et les voitures.

Bibliographie

[1] *http ://sesp.esep.pro/fr/pages$_n$anosats/capteurs − inertiels.html.*

[2] *https ://nanonets.com/blog/optical-flow/ ?utm$_s$ource = nanonets.com/blog/utm$_m$edium = blogutm$_c$ontent = optical*

[3] *https ://opencv-python-tutroals.readthedocs.io/en/latest/py$_t$utorials/py$_v$ideo/py$_l$ucas$_k$anade/py$_l$uc*

[4] *https ://www.generationrobots.com/fr/402159-gyroscope-et-accelerometre-3-axes-mpu-6050.html.*

[5] *https ://www.journaldunet.fr/.*

[6] *https ://www.mschoeffler.de/2017/10/05/.*

[7] *https ://www.techopedia.com/definition/30413/image-sensor.*

[8] CHASSERY Jean-Marc COCQUEREZ Jean-Pierre et al BOLON, Philippe. nalyse d'images : filtrage et segmentation 1995.

[9] C. BRAILLON. *Détection d'obstacles par fusion de flux optique et stéréovision dans des grilles d'occupation.* PhD thesis, 2007.

[10] WEICKERT Joachim-et SCHNORR Christoph. Lucas/Kanade meets Horn/Schunck BRUHN, Andrés. Combining local and global optic flow methods. international journal of computer vision. Master's thesis, 2005.

[11] WEICKERT Joachim-FEDDERN Christian et al BRUHN, Andrés. Variational optical flow computation in real time. ieee transactions on image processing. Master's thesis, 2005.

[12] Karim DAHIA. *Nouvelles méthodes en filtrage particulaire-Application au recalage de navigation inertielle par mesures altimétriques.* PhD thesis, Université Joseph-Fourier-Grenoble I., 2005.

[13] Arnaud DOUCET. On sequential simulation-based methods for bayesian filtering. Master's thesis, 1998.

[14] Thomas ESTRUCH. Thomas. comment fonctionnent les capteurs ccd et cmos ?. photoniques.

[15] Jean-François GRANDIN. Direction technique - systèmes de guerre électronique, thales systèmes aéroportés.

[16] Christopher G HARRIS. Stephens, mike, et al. a combined corner and edge detector. in : Alvey vision conference. 1988. p. 10-5244.

[17] Julien Metge. Etude de la calibration et de l'intégration sur mini-drone d'un système caméra-capteurs inertiels et magnétiques et ses applications. traitement du signal et de l'image.

[18] Hans P MORAVEC. Rover visual obstacle avoidance. in : Ijcai. 1981. p. 785-790.

[19] DAHIA Karim-et MUSSO Christian PHAM, D. A kalman-particle kernel filter and its application to terrain navigation. Master's thesis, 2003.

[20] LI Zhanjing-ZHANG Wei et al RONG, Weibin. An improved canny edge detection algorithm. in : 2014 ieee international conference on mechatronics and automation. ieee, 2014. p. 577-582.

[21] Cherif SMAILI. *Fusion de données multi-capteurs a l'aide d'un réseau bayésien pour l'estimation d'état d'un véhicule*. PhD thesis, Université Nancy II, 2010.

[22] François-et SAYS Simon TFabien, LE GLAND. Filtrage particulaire : quelques exemples" avec les mains" et matlab. Master's thesis, 2003.

[23] Jean-Philippe Lauffenburger Michel Basset Thomas, Sébastien Changey and Emmanuel Pecheur. Modélisation d'une unité de mesure inertielle augmentée de capteurs magnétiques a référence absolue. Master's thesis, 2013.

[24] Sami. TOUHAMI. Classification non supervisée de pixels d'images couleur par analyse d'histogrammes tridimensionnels.

[25] MONBET-Valérie et LEGLAND TRAN, Vu Duc. François. filtre de kalman d'ensemble et filtres particulaires pour le modele de lorenz. actes de la manifestation des jeunes chercheurs stic. Master's thesis, 2006.

[26] Rachid ZENNOUHI. Contribution à la segmentation des images couleurs par classification des pixels dans l'espace hsv et application de l'imagerie pour la détection du stress hydrique chez mentha spicata l. 2013.