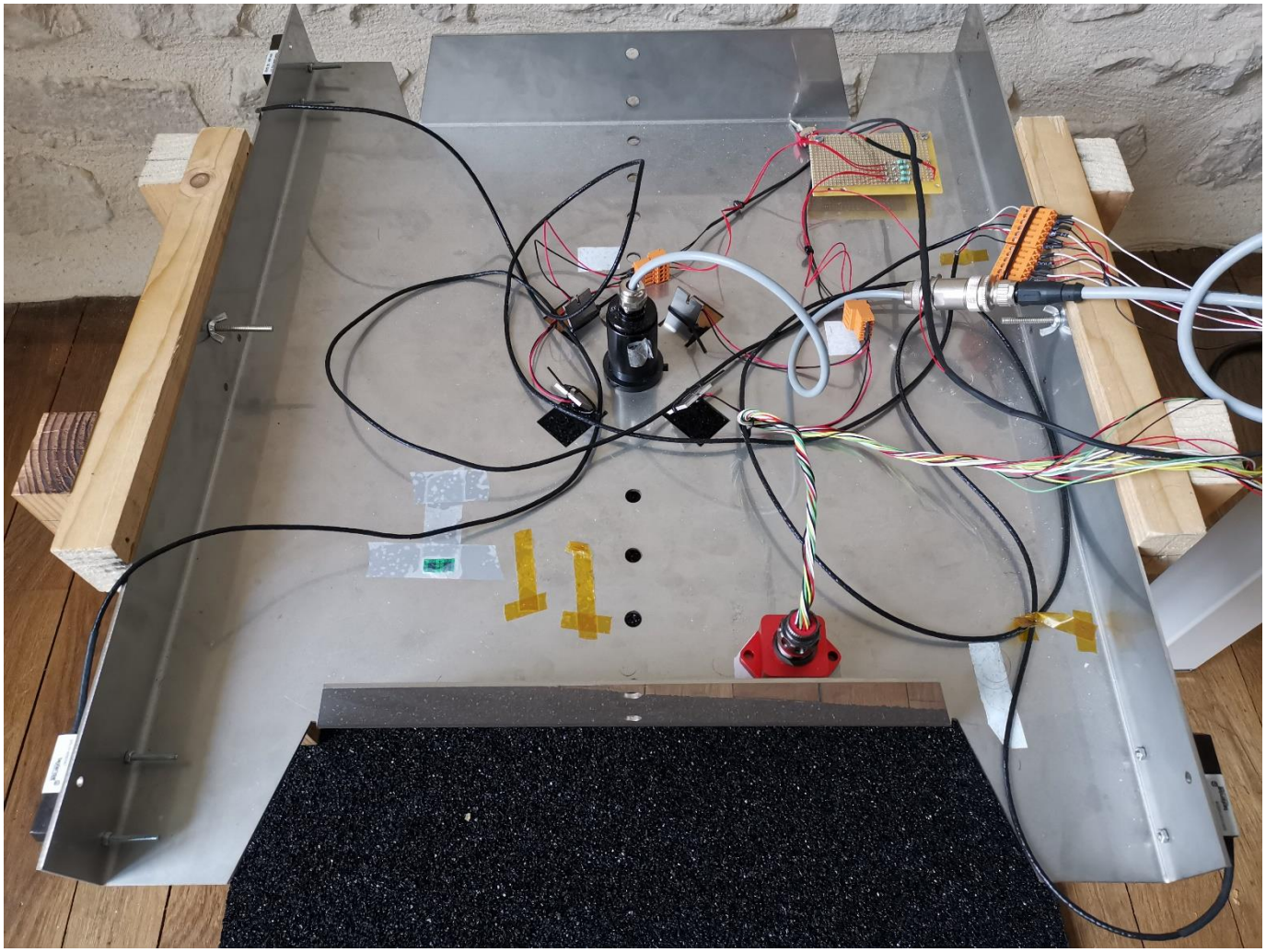


RHS Caméra
Rapport de stage

Texys
ZA des Chamonds
Rue Edouard Branly
58640 Varennes-Vauzelles
+33(0)386 212 718
+33(0)386 212 449

GAURIER Florian
Florian.gaurier@ensta-bretagne.org

Sommaire

Introduction

Texys est une entreprise qui a pour but de développer des technologies basées sur de l'électronique embarquée dans des domaines techniques très spécifiques et contraignants. Le développement des capteurs doit répondre aux contraintes de l'embarqué (Autonomie, encombrement) et à un temps de développement/d'adaptation de technologie rapide.
Texys travaille principalement avec le domaine de la compétition motorisée, qui est un secteur de pointe pour développer les technologies qui pourront être à disposition de l'ensemble des utilisateurs de véhicules motorisés. Ainsi, il n'est pas rare de voir une technologie développée dans les sports se diffuser dans l'ensemble de la gamme de véhicules. [Réf. nécessaire]

Dans le domaine des courses de véhicules, les capteurs doivent satisfaire des contraintes exigeantes de température, d'encombrement/poids mais aussi de fiabilité et de temps de réponse. Afin d'améliorer les performances des pilotes, il y a notamment une donnée essentielle : la hauteur du véhicule au sol, abrégé en RHS (Ride Height Sensor).

Les capteurs de hauteur de caisse (Ride Height Sensors) sont nécessaires dans le domaine de la course motorisée afin de pouvoir estimer le comportement dynamique d'un véhicule. Il s'agit foncièrement d'un simple capteur de distance. On en distingue de deux types : triangulation ou ToF (Time of Flight). Ces deux méthodes sont plus amplement exploitées par le projet RHS. Ce qu'il faut retenir c'est que la triangulation permet une précision importante mais une faible vitesse de lecture, de plus la méthode fait intervenir des limitations d'encombrement minimal à respecter pour des raisons d'optique tandis que le ToF permet une grande vitesse, un faible encombrement mais une précision moindre en raison des distances en jeu dans notre domaine d'application.

Afin d'obtenir le comportement dynamique d'un véhicule (hauteur de caisse + angles de roulis et tangage) il est nécessaire que le véhicule dispose d'au moins 3 capteurs. L'idée du RHS Caméra est donc de réunir ces 3 capteurs en 1. Il se base sur une méthode par triangulation, plus précise que le ToF.

Il est important de noter que l'ensemble des capteurs de distance par triangulation font intervenir un capteur (CMOS) linéaire. Autrement dit, un capteur 1D. Le RHS Caméra se base donc sur un capteur 2D pour obtenir un plan dans lequel l'ensemble des sources lasers (3 minimum) sont présentes afin de miniaturiser le système cité plus haut.

La technologie du RHS Caméra est *a priori* un capteur RHS par triangulation avec une surface d'élément sensible 2D au lieu de 1D.

Chapitre 1 : Etude de faisabilité

1. Analyse théorique

1.1. Description du projet

1.1.1. Technologies envisagées

Dans le domaine de l'automobile, l'utilisation de caméras est de plus en plus démocratisée puisqu'elles permettent d'effectuer un contrôle de l'environnement et donc d'assister le conducteur voire même permettre la création de voitures autonomes par des algorithmes de détection et de classification de l'environnement filmé. Le but de certaines d'entre elles est notamment d'estimer la position du véhicule par rapport à son environnement, elles disposent donc d'éléments servant à la détection de distance.

On recense les techniques de ToF (Time of Flight) dont le principe est d'émettre un signal (acoustique ou électromagnétique) à un instant t_0 connu que l'on reçoit à t_0+dt. Cette mesure permet ensuite, connaissant la vélocité de l'onde, d'en déduire la distance parcourue. Cependant, dans le cadre d'un RHS, la distance parcourue est de l'ordre de la dizaine de centimètres. Les ondes électromagnétiques nécessitent alors des détections de l'ordre de 30ps, ce qui ne peut pas correspondre aux contraintes de taille imposées. La solution acoustique est sensible aux changements de pression et de température de l'air (changeant ainsi la vitesse de l'onde).

On peut voir aussi apparaître les technologies de stéréoscopie qui permettent d'établir une carte de profondeur à l'aide de deux caméras, mais l'encombrement est important tout comme le calcul.

Enfin, la solution privilégiée par Texys est celle de la triangulation : par la connaissance de la position d'un spot lumineux on en déduit géométriquement les données qui nous intéressent.

1.1.2. Principe de fonctionnement

Le projet RHS Caméra a été pensé de manière à réaliser une mesure de hauteur sur un capteur CMOS 2D, développant ainsi comme une extension du laser Baumer. On a ainsi une relation entre la position du véhicule et celle du spot.

La complexité vient du fait que le raisonnement sur le capteur 2D doit en réalité s'appliquer sur un modèle 3D. Et même si *a priori* le tangage et le roulis peuvent être décorrélés en partant du principe que tout mouvement est une combinaison linéaire des actions de roulis, tangage et hauteur (donc que l'on peut raisonner sur un axe puis l'autre indépendamment), il existe en réalité des non-linéarités dans le modèle comme développé dans les parties suivantes. Ces non-linéarités ne permettent pas un calcul immédiat des données attendues. Il faut donc séparer le problème en sous-problèmes correspondants et plus simples à traiter.

L'idée du projet est donc de vérifier la faisabilité et de créer un produit correspondant à un capteur 2D de hauteur, permettant de retrouver également le tangage et le roulis. Une ouverture est également possible concernant l'analyse de la texture du sol afin de déterminer le vecteur vitesse réel du véhicule. Cette dernière partie n'est pas étudiée présentement, mais peut faire l'objet d'un développement ultérieur pour un capteur tout en un.

Ce dernier axe se baserait sur les techniques utilisées dans le cinéma pour la modélisation du mouvement de la caméra et des éléments à l'image, notamment sur les matrices d'homographie qui permettent de retrouver les matrices de rotation et le vecteur de translation d'un objet mobile par rapport à un environnement fixe.

1.1.3. Spécifications

Ce projet possède des spécifications de bases et d'autres contraintes rajoutées plus tard. Cette partie cherche à les rassembler.

Critère	Min.	Max.
Température (°C)	???°C	125°C
Distance (mm)	60mm	400mm
Distance spot (m)	Non Spécifié	4m
Angle roulis (°)	-40°	40°
Angle tangage (°)	-40°	40°
Fréquence des données (Hz)	100Hz	
Précision hauteur (mm)		1
Précision angle (°)		1

1.2. Modèle physique adopté

1.2.1. Paramétrisation

Nous supposons que le RHS Caméra est fixé sous un véhicule, nous nous intéressons au mouvement du RHS Caméra : les angles et la hauteur déterminés sont ainsi pris au centre optique (Position de la lentille) de la caméra. Cette installation suppose aucune déformation le long du véhicule, en effet dans cette configuration le RHS Caméra est disposé n'importe où sous le véhicule. L'angle formé par le véhicule par rapport au sol est alors le même à l'avant et à l'arrière, sauf s'il y a déformation du fond plat.

Nous modélisons le RHS Caméra de la manière suivante :

- La piste est assimilée à un plan fixe et plat de repère $R_0(O, \overrightarrow{x_0}, \overrightarrow{y_0}, \overrightarrow{z_0})$, $\overrightarrow{y_0}$ est dans le sens de parcours de la piste.
- Le repère $R_1(O, \overrightarrow{x_1}, \overrightarrow{y_1}, \overrightarrow{z_1})$ est formé par une rotation d'angle θ autour de l'axe $\overrightarrow{y_0}$ PUIS une rotation d'angle ϕ autour de l'axe $\overrightarrow{y_{0'}}$ résultant de la première rotation. (Repère associé à la première rotation : $R_{0'}(O, \overrightarrow{x_{0'}}, \overrightarrow{y_{0'}}, \overrightarrow{z_{0'}})$)
- L'axe $(O, \overrightarrow{z_1})$ est la droite orthogonale définissant le plan du véhicule. Ce plan du véhicule est supposé confondu au plan de la lentille du RHS Caméra. Le plan du véhicule est définit par son éloignement à O par la hauteur h.
- Le plan du CMOS est parallèle au plan du véhicule et éloigné de la distance focale f de la caméra.

Pour suivre un formalisme mathématique et être plus explicite :

- Soit $R_0(O, \overrightarrow{x_0}, \overrightarrow{y_0}, \overrightarrow{z_0})$ Le repère associé au sol. y_0 est dans l'axe longitudinal de la piste.
- Soit $R_{0'}(O, \overrightarrow{x_{0'}}, \overrightarrow{y_{0'}}, \overrightarrow{z_{0'}})$ Le repère R_0 après une rotation d'angle θ autour de l'axe $\overrightarrow{y_0}$.
- Soit $R_1(A, \overrightarrow{x_1}, \overrightarrow{y_1}, \overrightarrow{z_1})$ Le repère associé au véhicule/la caméra. Il est réalisé par une rotation d'angle ϕ autour de l'axe $\overrightarrow{y_{0'}}$.
- Soit $R_2(A, \overrightarrow{x_2}, \overrightarrow{y_2}, \overrightarrow{z_2})$ Le repère associé au positionnement des lasers autour de la caméra. Il est réalisé par une rotation d'angle β autour de l'axe $\overrightarrow{z_1}$.
- Soit $R_{2'}(i, \overrightarrow{x_{2'}}, \overrightarrow{y_{2'}}, \overrightarrow{z_{2'}})$ Le repère associé au positionnement des lasers par rapport à la normale à la caméra où $i \in \{F, H, I, D\}$. Il est réalisé par une rotation d'angle α autour de l'axe $\overrightarrow{y_2}$.

Pour aller d'un repère à l'autre on forme ainsi 4 matrices de rotation. Notées respectivement $R_\theta, R_\varphi, R_\beta, R_\alpha$. Ainsi, pour aller d'un repère à l'autre, on multiplie successivement les matrices entre elles par l'ordre des rotations définies précédemment.
Celles-ci valent :

$$R_\theta = \begin{pmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{pmatrix} ; \; R_\varphi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & -\sin(\varphi) \\ 0 & \sin(\varphi & \cos(\varphi) \end{pmatrix}$$

$$R_\beta = \begin{pmatrix} \cos(\beta) & -\sin(\beta) & 0 \\ \sin(\beta) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{pmatrix} ; \; R_\alpha = \begin{pmatrix} \cos(\alpha) & 0 & -\sin(\alpha) \\ 0 & 1 & 0 \\ \sin(\alpha) & 0 & \cos(\alpha) \end{pmatrix}$$

Ainsi, on définit la géométrie des points de la manière suivante :

$$\overrightarrow{OA} = h * \overrightarrow{z_1}$$
$$\overrightarrow{OF} = \overrightarrow{OA} + e * \overrightarrow{x_2}$$
$$\overrightarrow{OD} = \overrightarrow{OA} - e * \overrightarrow{x_2}$$
$$\overrightarrow{OH} = \overrightarrow{OA} + e * \overrightarrow{y_2}$$
$$\overrightarrow{OI} = \overrightarrow{OA} - e * \overrightarrow{y_2}$$

Les points F, D, H et I sont définis par l'angle β avec un déphasage respectif de 90° d'un point au suivant. On note F', D', H', I' les points projetés orthogonalement au plan $(A, \overrightarrow{x_1}, \overrightarrow{y_1})$ de composante selon $\overrightarrow{z_0}$ égale à 0. De même on définit les points F_α, D_α, H_α, et I_α de la manière suivante :

$$\overrightarrow{FF_\alpha} = -d_F * \overrightarrow{z_{2'}}$$
$$\overrightarrow{II_\alpha} = -d_I * \overrightarrow{z_{2'}}$$
$$\overrightarrow{HH_\alpha} = -d_H * \overrightarrow{z_{2'}}$$
$$\overrightarrow{DD_\alpha} = -d_D * \overrightarrow{z_{2'}}$$

/!\ : On part du principe que les points définissent un repère $R_{2'}$ qui leur est propre pour éviter une lourdeur au niveau de l'écriture. (Dans l'absolu, on devrait dénoter une rotation β_F, β_I, β_D et β_H ainsi qu'un repère associé à chacune d'entre elles.)

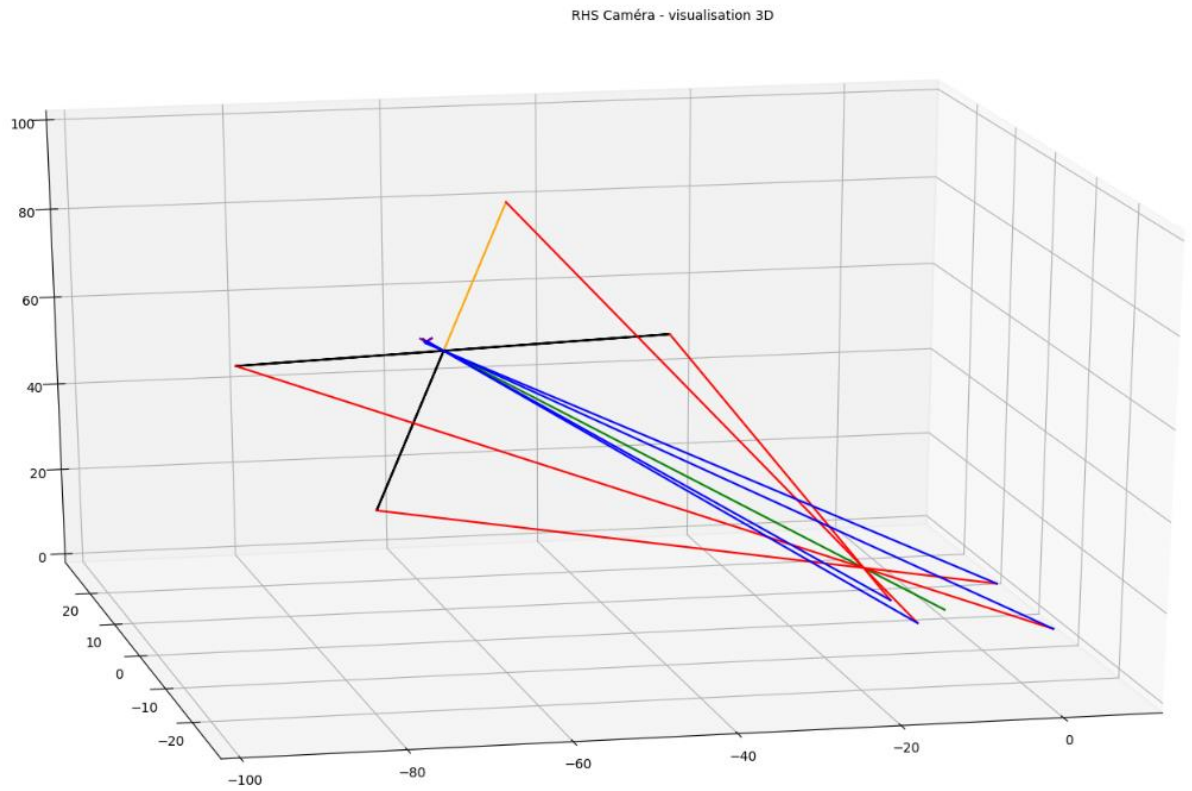


Figure 1 : Représentation 3D du RHS Caméra. En rouge les faisceaux laser ; en bleu les rayons émis par le spot laser (au sol) et passant par le centre optique de la caméra ; en vert l'axe optique de la caméra ; en jaune l'axe x du repère de la caméra après une rotation d'un angle ϑ=45° ; en noir les autres axes $(-x, y, -y)$ de ce même repère. Cette visualisation a été réalisée avec un angle ϑ = 45° et une hauteur h=90. Unités en mm.

On dénommera 1,2,3,4 les points $F_\alpha, I_\alpha, H_\alpha, D_\alpha$ vus depuis la caméra et repérés dans le sens de lecture conventionnel. Il existe un plan parallèle au plan de la caméra, dans lequel se forme l'image filmée. Ce plan est celui dans lequel est présent le capteur CMOS et est séparé du point A d'une distance f (équivalent d'une focale en pixels). On désignera alors d_2^i la distance au centre de l'image en pixels. Cette distance est facilement obtenue par le fait que l'on ait accès à l'abscisse et l'ordonnée du point. On obtient le schéma suivant :

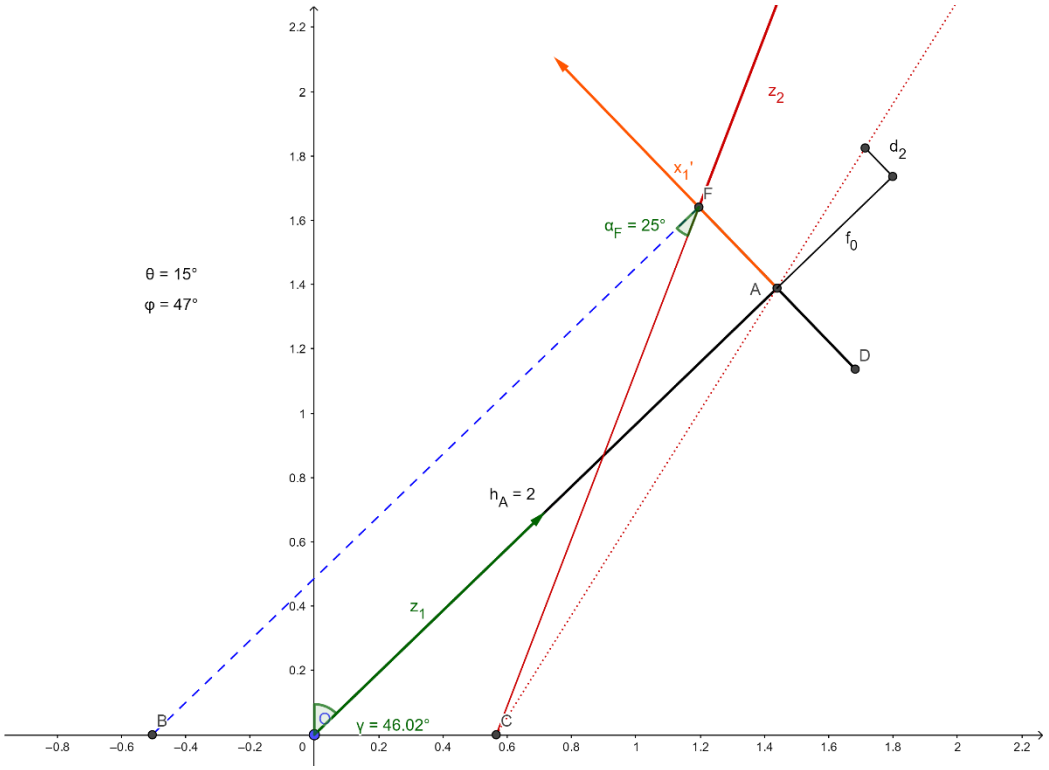


Figure 2 : Plan de coupe laser-sol. En rouge : le laser. En pointillés rouge : le rayon lumineux issu du spot laser passant par le centre optique A de la caméra. d_2 est en réalité d_2^i.

On remarque que l'angle formé (γ) ne possède pas de relation linéaire avec θ et φ (ou leurs cosinus/sinus/tangente).

De là, notre modèle physique est complet. Il faut alors établir une relation mathématique entre l'ensemble des d_2^i et h, θ et φ.

Nous allons procéder en deux étapes. Pour cela, nous allons déterminer l'ensemble des d_i en fonction des d_2^i puis h, θ et φ en fonction des d_i.

Par le schéma 2 on a la relation :

$$d_2^i = \frac{f * (d_i * \sin(\alpha) + e)}{d_i * \cos(\alpha)}$$

$$\boxed{d_i = \frac{f * e}{d_2^i * \cos(\alpha) - f * \sin(\alpha)}} \quad (1)$$

Il nous faut alors déterminer d_i en fonction de h, θ et φ. Pour cela, il faut tracer le plan $(O, \overrightarrow{x_1}, \overrightarrow{z_1})$ cf. figure précédente.

Soit le repère $R_3(O, \overrightarrow{x_3}, \overrightarrow{y_3}, \overrightarrow{z_3})$ défini par une rotation d'angle γ du repère R_2, autour de l'axe $\overrightarrow{y_2}$, de manière à ce que $\overrightarrow{x_3}$ soit dans le plan $(O, \overrightarrow{x_0}, \overrightarrow{y_0})$. On a la relation :

$$\boxed{d_i = \frac{e * \sin(\gamma) + h * \cos(\gamma)}{\cos(\alpha + \gamma)}} \quad (2)$$

On cherche à déterminer sin(γ) et cos(γ) :

$$\overrightarrow{x_3}.\overrightarrow{z_0} = \sin(\gamma) * \cos(\varphi) * \cos(\theta) - \sin(\theta) * \cos(\gamma) * \cos(\beta) - \cos(\gamma) * \cos(\theta) * \sin(\varphi) * \sin(\beta) = 0$$

D'où :

$$\boxed{\tan(\gamma) = \frac{\cos(\beta) * (\tan(\theta) + \tan(\beta) * \sin(\varphi))}{\cos(\varphi)}} \quad (3)$$

En injectant (3) dans (2) on obtient pour chacun des spots :

$$\boxed{\left([e + d_i * \sin(\alpha)] * \cos(\beta) \quad [e + d_i * \sin(\alpha)] * \sin(\beta) \quad 1 \right) \begin{pmatrix} \frac{\tan(\theta)}{\cos(\varphi)} \\ \tan(\varphi) \\ h \end{pmatrix} = d_i * \cos(\alpha)} \quad (4)$$

On a ainsi 4 spots, donc quatre relations de la sorte pour 3 inconnues. On devrait bien avoir une expression des données recherchées. Cependant, une écriture de l'équation précédente plus simple est possible. On utilise l'équation (1) dans l'équation (4) et on obtient :

$$\boxed{\left(d_2^i * \cos(\beta) \quad d_2^i * \sin(\beta) \quad \frac{d_2^i - f * \tan(\alpha)}{e} \right) \begin{pmatrix} \frac{\tan(\theta)}{\cos(\varphi)} \\ \tan(\varphi) \\ h \end{pmatrix} = f} \quad (5)$$

Cette relation, bien plus simple fait intervenir un terme f qui est la distance focale. Etant donné que d_2^i sera exprimé en pixels, cette distance focale le sera également.

/!\ IMPORTANT : les β des équations précédentes sont associés au déphasage d'un point (F, H, D, I) dans le repère R_2. Ainsi, on a $\beta_F=\beta$, $\beta_H=\beta+90°$ etc.

1.3. Détection de spot

Dans la partie précédente, nous avons établi une relation entre les données renvoyées par le capteur et la distance au centre de l'image en pixels. (Cf. (5)) Cependant, il faut connaître les coordonnées du spot en pixels, et donc réaliser un algorithme de détection de spot. Voici les conditions dans lesquelles l'algorithme devra fonctionner :

- Extérieur ⇒ Luminosité variable.
- Détection sur différentes textures : route, bandes blanches et rouges.

La couleur apparente d'un objet est influencée par la luminosité ambiante. De plus, travaillant sur différentes textures, l'aspect du spot est différent sur chacune d'entre elles. Il faut donc trouver un domaine chromatique le plus adapté à la situation. On pourrait penser à trois d'entre eux : RGB, HSV ou YUV. Le RGB étant sensible aux changements de luminosité, il n'est pas adapté à ce que l'on cherche. Le HSV, après essais, donne des résultats instables et on reste peu précis quant au choix de la teinte (Hue en anglais). Enfin, il y a le YUV qui a l'avantage d'être moins sensible aux changements de variation. Ce domaine permet de représenter sur un canal la teinte en niveaux de gris, et sur les deux

canaux restants, la différence entre cette teinte et deux autres composantes du RGB (typiquement le rouge et le bleu).

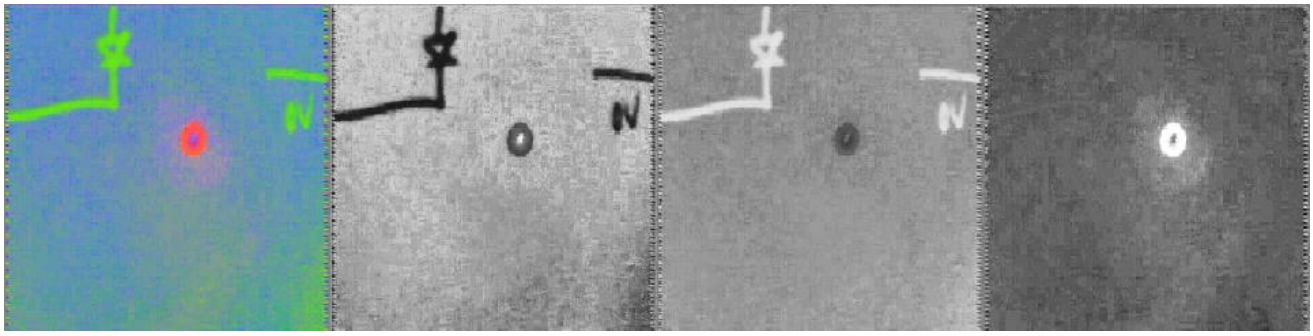


Figure 3 : YUV 3 canaux (gauche). Canal Y (centre gauche). Canal U (centre droite). Canal V (droite). L'image d'origine a été prise après contrasteet saturation.

Lors d'une luminosité forte, le spot laser n'est pas toujours détectable et est noyé dans le bruit. Pour remédier à ce problème, on réduit le contraste et on augmente la saturation, ce qui permet de faire ressortir le point de l'image, un simple filtrage permet ensuite de réduire le bruit. Cependant, il se peut que le point soit toujours noyé dans le bruit, il est alors nécessaire de faire d'autres transformations afin de régler ce problème.

Enfin, on est amenés à travailler sur le degré de « rouge » par les transformations suivantes : [SOURCE]

$$V_i = V - U - Y$$
$$V_i = V_i - \overline{V_i}$$
$$U_i = V_i - U$$
$$Y_i = V_i - Y$$
$$DV_i = V_i + U_i + Y_i$$

DV_i représente ainsi le degré de « rouge » (du canal V en réalité) par rapport aux autres canaux. On a alors ce type d'image :

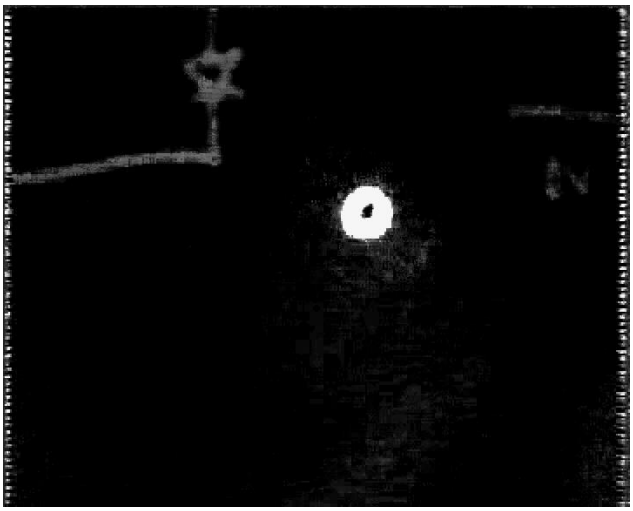


Figure 4: image en DVi. (Degré relatif du canal V dans les aux autres)

Il suffit alors de seuiller l'image pour avoir le spot.

2. Solutions pratiques et choix technologiques de la maquette

2.1. Mécanique et modèle réel

Pour confirmer la validité des formules et déterminer la faisabilité de la solution envisagée, il faut disposer d'une maquette appropriée et donc fixer les différents paramètres. Ces paramètres sont :

- L'angla α fixant la direction du faisceau laser. (Repéré par rapport à la normale au plan caméra)
- L'écartement *e* entre la caméra et le laser.
- L'angle β fixant la position du laser (et donc du spot) dans le plan de la caméra.
- Le nombre de lasers.

Concernant le nombre de lasers, nous l'avons fixé à 4. En effet, nous avons 3 inconnues à déterminer, il faut donc trois données. Le quatrième laser est là pour assurer la symétrie de l'installation mais aussi sert à confirmer/améliorer la mesure réalisée. (Calcul au sens des moindres-carrés à prévoir)

Nous avons imposé que la distance maximale de détection d'un spot soit inférieure à 4m afin de conserver une certaine liberté dans le choix de l'angle α ainsi que de permettre une détection d'un angle de roulis à plus de 60° (record du monde à 68°). Ainsi, avec un angle α à 25°, à la hauteur et l'angle de roulis au maximum, respectivement 0.4m et 70°, on a un spot à 3.8m.

Pour le choix de β on a deux configurations types : croix ou carré. La première a l'avantage d'avoir l'ordonnée de deux spots égale à zéro sur l'image formée et de même en abscisse sur les deux autres spots. Le réglage est donc plus simple qu'une configuration carré, et le calcul (basé sur la distance au centre de la caméra) aussi. La configuration carré permet en revanche que chacun des spots aide à déterminer l'ensemble des angles et de la distance quelle que soit la configuration. En effet, sur le montage croix, lors d'un tangage ou d'un roulis pur, seuls 2 spots sont utilisés. Enfin, le montage croix ne fait pas intervenir d'angle β, rendant les calculs plus simples mais imprécis car le montage réel aura un angle β proche de 0 mais il existera.

On a choisi un montage carré, pensant à l'origine que le problème était linéaire. C'est-à-dire que le mouvement final des points serait une combinaison linéaire de la hauteur, tangage et roulis.

Enfin, l'écartement *e* a été fixé à 3.5cm. Bien que peu représentatif du produit final, il a été majoré de manière à pouvoir réaliser une maquette simple et sans trop de contraintes de fabrication.

Ces valeurs déterminées, il a été possible de créer une maquette servant à installer un dispositif de référence (laser Baumer) en même temps que le dispositif à évaluer (caméra + lasers). La maquette suivante a été réalisée.

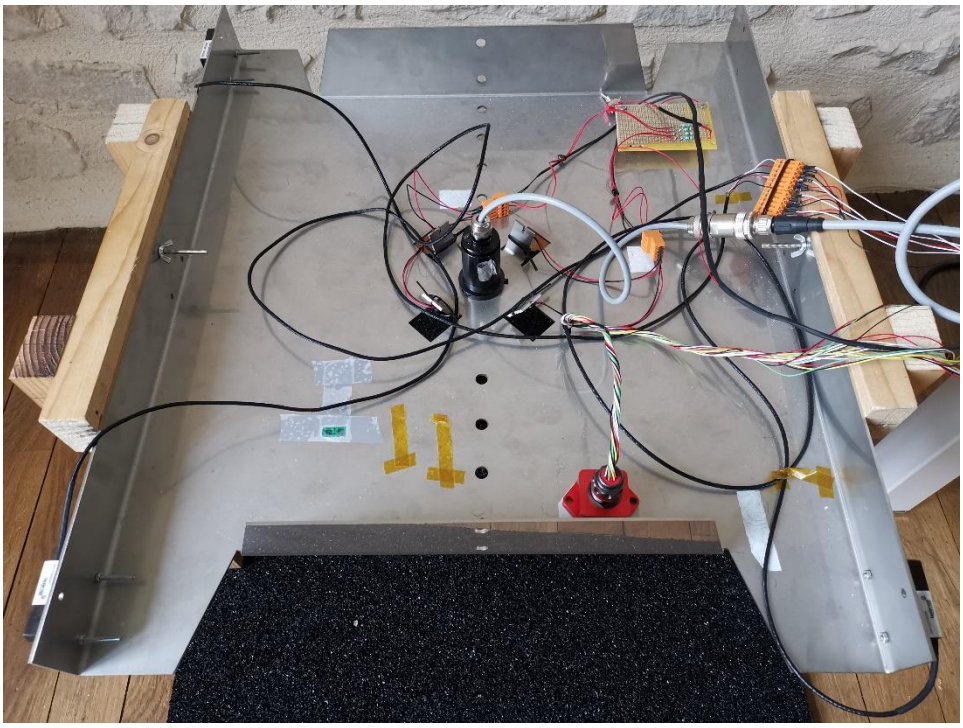


Figure 5: Maquette du RHS. En périphérie les Baumer en référence. Au centre la caméra et les lasers.

Le choix d'une taille aussi importante est due à l'impact que pourraient avoir les faisceaux laser de la référence dans le champ de la caméra. Il a donc été décidé que les Baumer ne devaient pas être présents dans le champ de la caméra à la hauteur maximale soit 40cm, d'où une maquette aussi imposante.

Pour cette maquette, il a fallu choisir des lasers et une caméra. La caméra a été choisie en raison de ses caractéristiques de température (jusqu'à 105°C). Cependant, ce n'est pas la caméra définitive en raison de la résolution de l'image et de la qualité des couleurs. De plus, les données de FOV et de focale ne sont pas spécifiées. Il a donc fallu utiliser d'autres informations ou réaliser des expériences afin de les déterminer. Les lasers ont été choisis pour émettre dans le rouge ($\lambda = 635$nm), peu coûteux en énergie (5mW), peu encombrants et bien visibles sur le sol.

2.2. Logiciel

On a donc trois traits immédiats au logiciel : la gestion des interactions entre les différentes parties, la recherche du point et le calcul du point. Nous avons établi qu'il en nécessitait plus.

Nous ne détaillerons ici que les algorithmes de recherche de point, de calibration et du RHS OS en raison du peu d'intérêt qu'il existe sur le display et le calcul.

2.2.1. Recherche de point

Nous avons brièvement expliqué la recherche de point dans la section 1.3 mais elle détaillait surtout le principe général. Nous allons ici détailler le fonctionnement de l'algorithme de recherche de point au stade de maturité avancée, n'incluant pas toutes les fonctionnalités mais ayant une

structure qui restera sensiblement invariante au cours du projet. (Sauf modification majeure de la technologie employée)

On dispose d'une image (Le plus souvent 640*480) dans laquelle sont présents quatre spots. Chacun de ces spots est présent dans une partie de l'image et ne peut aller dans une autre. Nous avons donc découpé l'image en quatre selon des zones par défaut. (Réglables par l'utilisateur) Puis, on sélectionne dynamiquement la partie de l'image dans laquelle se trouve le spot : on suppose que la variation de position d'un spot est quasi-continue[1] ce qui permet d'utiliser la position précédente afin de redimensionner la fenêtre. On met toutefois en place un critère déterminant qui est la somme de la distance parcourue sur les n dernières mesures. Si cette somme est supérieure à un seuil (configurable par l'utilisateur) alors on recalcule la position du spot sur l'ensemble de la première sous-zone. Il est également important de noter que le redimensionnement est effectué pour chacun des points ! En aucun cas l'image est découpée et stockée suite à un redimensionnement préliminaire puis cette découpe réutilisée.

Une fois l'image redimensionnée, on continue le pré-traitement par une diminution de contraste et une augmentation de saturation avec des valeurs choisies expérimentalement et de manière qualitative. L'image est alors filtrée et convertie dans le domaine colorimétrique YUV. Le filtrage utilise un filtre médian (appliqué 1 fois).

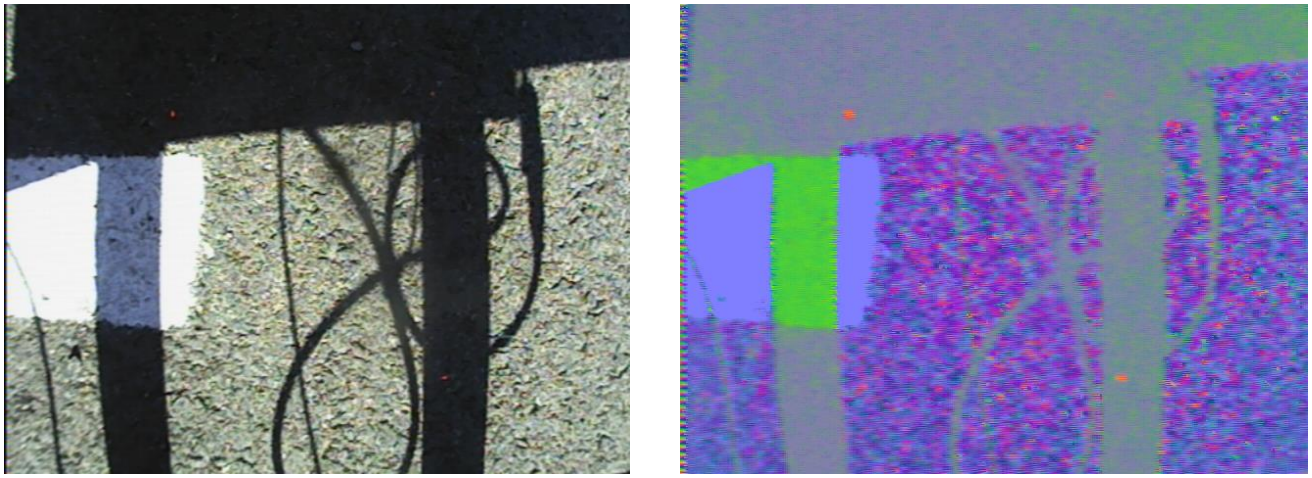


Figure 6: Image prise avec la caméra en plein soleil (à gauche) et la même en YUV non filtrée après contraste et saturation (à droite).

A partir de ce moment, on dispose de l'image YUV saturée, contrastée et filtrée. On a ainsi réduit les artéfacts de bruit et accentué la présence de couleurs à dominante rouge sur le canal V. On obtient cependant des images bruitées en plein soleil sur l'extérieur.

On remarque sur l'image en YUV que les parties au soleil (excepté la bande blanche) sont bien plus bruitées que les parties à l'ombre ou réfléchissant la lumière du soleil. Une détection est donc plus facile à l'ombre qu'au soleil.

[1] La définition mathématique de continuité ($\forall \varepsilon > 0 \ \exists \eta > 0 \ \forall x \in I \ |x - a| < \eta \ \Rightarrow |f(x) - f(a)| < \varepsilon$) n'est pas applicable ici car résultant d'un échantillonnage sur le nombre d'images par seconde renvoyées par la caméra. L'hypothèse exacte serait donc « on suppose le changement de position minime devant le temps mort existant entre deux images ». L'énoncé serait donc long à écrire…

On vient ensuite calculer le degré de « V » par le processus de calcul détaillé dans la partie 1.3 ce qui nous donne une image en noir et blanc (1 canal) sur laquelle on voit clairement le spot avec éventuellement du bruit. Un seuil est appliqué (défini comme étant k*max de l'image), pour cela voir Figure 7 ci- après. Enfin, on recherche la position du point sur l'image seuillée utilisée comme masque et appliquée sur l'image saturée en RGB. La position du point est simplement déterminée comme étant la moyenne des positions des pixels non nuls après avoir effectué la somme des valeurs sur chacun des axes.

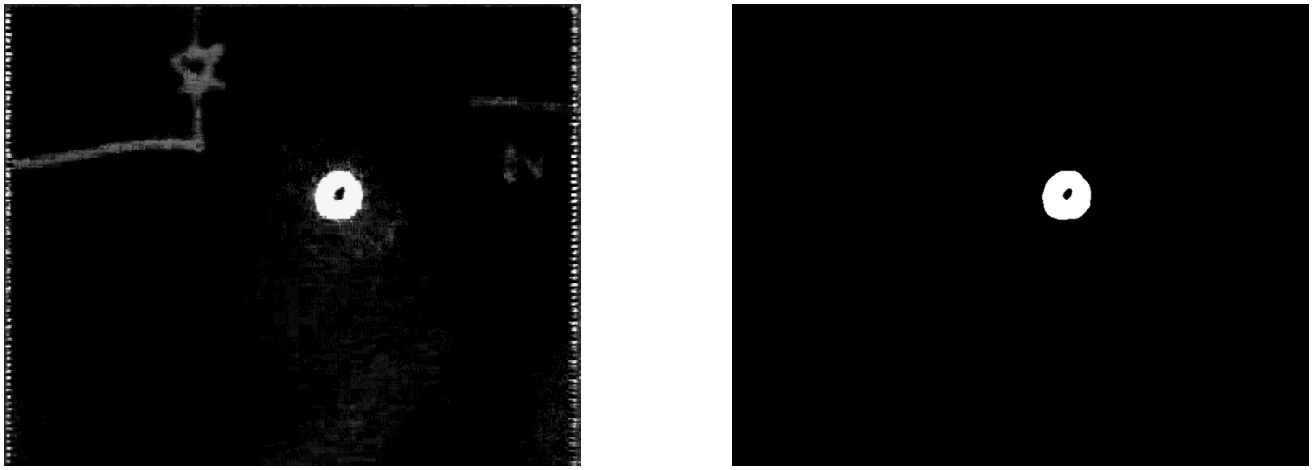


Figure 7: Image DVi (à gauche) et la même image seuillée. (A droite)

2.2.2. Calibration

La calibration a dans un premier temps été mise en place pour le phénomène de distorsion impliqué par la lentille de la caméra et donc pour essayer de redresser l'image déformée. (Et ainsi obtenir une résolution uniforme tout au long de l'image. Cette calibration utilise des fonctions déjà implémentées dans OpenCV et vient détecter le profil type d'un quadrillage type échiquier, en extrait les coordonnées des carrés et permet ensuite d'en déduire les paramètres nécessaires à la distorsion inverse.

Plus tard, nous nous sommes aperçus de l'utilité d'une calibration pour le calcul afin de prendre en compte les défauts de géométrie qui pourraient exister. On pense notamment à l'angle α qui a eu une imprécision de montage importante sur la maquette que l'on a réalisé, sans compter un éventuel défaut d'alignement de la lentille avec l'axe du laser.

On a donc deux types de calibration, dont le deuxième procède en deux étapes.

Pour réaliser la première calibration, on dispose donc l'échiquier dans des positions aléatoires devant la caméra et de préférence balayant l'ensemble du FOV. On capture une image à chaque position. Nous avons également remarqué que le nombre d'images idéal semble se situer entre 10 et 30 images. (Sinon l'erreur augmente) De plus, les calibrations en distorsion que nous avons faîtes ont été faîtes de manière approximative, en tenant l'échiquier à la main, et donc n'ayant pas de dispositif fixe permettant l'immobilisation totale ou un balayage de la zone optimale.

Une fois les images obtenues, les données des carrés sont extraites de chaque image et calculées par les algorithmes de la bibliothèque OpenCV pour en sortir les paramètres.

La deuxième calibration est prévue pour prendre en paramètres deux logs de caméra (un sans angle de roulis/tangage et avec une variation de hauteur uniquement, et l'autre avec tous les paramètres qui varient) et deux logs de référence correspondants.

A ce moment, l'utilisateur doit spécifier à quel moment choisir l'origine et la fin de l'acquisition, puisqu'en réalité, il y a un délai entre l'activation du log de référence et du log de la caméra en raison d'une activation manuelle. Un signal simultané sur les deux logs est alors émis, typiquement une variation soudaine de hauteur.

Une fois les logs acquis et chargés en mémoire et étant donné que la fréquence d'acquisition de la référence est plus importante que celle de la caméra, il a fallu procéder à une compression des données de la référence, impliquant un problème d'hystérésis (expliqué dans les parties suivantes) lors du tracé des graphes de hauteur en fonction de la position du spot sur la caméra.

Une fois ces graphes obtenus, il a été possible d'effectuer une régression linéaire pour obtenir la première étape de la calibration en calcul. Puis, à l'aide des formules analytiques, on en déduit la deuxième calibration nécessaire.

2.2.3. RhsOS

Le RHS OS a pour objectif de gérer les interactions de l'ensemble des modules, d'en être le chef d'orchestre. Il possède alors chacun des modules en tant qu'objets et chacun initialisés avec les paramètres par défaut spécifiés à l'initialisation du RHS OS.

L'idée est donc de commencer par une calibration (chargement ou calcul) puis d'initialiser la capture avec les images à afficher et enfin, réaliser la capture avec le chemin passé en paramètres. Cela permet également d'enregistrer les données, d'afficher les différentes images avec des barres de réglage en direct sur les différents paramètres (notamment contraste et saturation), d'utiliser une capture live, pré-enregistrée ou une image.

2.3. Calibration

2.3.1. Intérêt d'une calibration

La calibration est déterminante dans ce projet. En effet, lors de la réalisation de la maquette, malgré des angles α à 25°, nous avons vu des écarts importants d'un laser à l'autre. En théorie, on aurait dû voir un carré centré sur l'axe optique de la caméra. En pratique, les lasers n'étaient pas du tout à une distance au centre similaire les uns par rapport aux autres : il y a donc un défaut d'alignement du laser. Ainsi, on ne se retrouve plus avec un α unique et des β déphasés de 90° mais bien des α_i et des β_i qu'il faut déterminer. Le but de la calibration est donc de déterminer principalement ces angles (qui ne sont pas censés varier au cours du temps car imposés par la géométrie du produit).

De plus, la lentille de la caméra n'est pas parfaite : celle dont on dispose réalise une distorsion en barillet. Cela a pour impact de changer la résolution d'un pixel ainsi que la position réelle de celui-ci. Sur les bords de l'image on observe donc une distorsion plus importante qui se traduit par le fait qu'un pixel à cet endroit représente une distance au sol plus importante qu'un pixel au centre. Et donc la résolution n'étant plus linéaire, on observe une incertitude de la mesure croissante à mesure que l'on s'éloigne du centre de la caméra.

L'intérêt d'un algorithme de compensation de distorsion est qu'il permet en quelques images calibrées sur une mire de trouver les paramètres intrinsèques de la caméra (et donc invariants dans

le temps) afin de les réutiliser pour une distorsion inverse de l'image qui sera certes rognée par rapport à l'image de départ mais qui a pour avantage d'avoir une résolution constante pour chaque pixel.

2.3.2. Fonctionnement de la calibration

L'algorithme de calibration est construit de telle sorte à ce que les paramètres intrinsèques au système soient déterminés et puissent être enregistrés dans des fichiers spécifiés par l'utilisateur. Il peut être exécuté ou non selon les besoins de l'utilisateur. Ainsi, l'utilisateur peut choisir de réaliser jusqu'à trois formes de calibration : la calibration de distorsion, servant à compenser une distorsion d'image ; la linéarisation du système, et enfin la régression linéaire.

Dans la section 1.2 nous avons établi que le calcul de h, ϕ et θ se réalisait à l'aide d'une relation relativement simple exprimée en (5) que nous redonnons ici :

$$\begin{pmatrix} d_2^i * \cos(\beta) & d_2^i * \sin(\beta) & \frac{d_2^i - f * \tan(\alpha)}{e} \end{pmatrix} \begin{pmatrix} \frac{\tan(\theta)}{\cos(\varphi)} \\ \tan(\varphi) \\ h \end{pmatrix} = f$$

Nous pouvons écrire cette équation sous une autre forme. En effet, lors d'une calibration, nous connaissons les données d'entrée et de sortie, on cherche alors à revenir aux paramètres internes en fonction de y et x. Ecrivons l'équation (5) sous une autre forme :

$$k_1 x_1 y + k_2 x_2 y + (k_3 y + k_4) x_3 = 1 \quad (6)$$

Avec :

$$k_1 = \frac{\cos(\beta)}{f} \; ; \; k_2 = \frac{\sin(\beta)}{f} \; ; \; k_3 = \frac{1}{f * e} \; ; \; k_4 = \frac{\tan(\alpha)}{e}$$

Nous pouvons alors, à l'aide de mesures de référence (Baumer ; mesure directe…) déterminer l'ensemble des x_i. Cependant, le système n'est pas linéaire, il est donc nécessaire de procéder en plusieurs étapes.

Les x_i étant reliés directement aux angles de roulis, tangage et à la hauteur, ces trois données sont indépendantes. On peut également établir un environnement contrôlé où on annule les angles de roulis et tangage. (6) devient alors :

$$k_3 y + k_4 = \frac{1}{x_3} \quad (7)$$

Cette relation peut faire l'objet d'une régression linéaire après acquisition de suffisamment de données représentatives et donc de connaître k_3 et k_4.

On peut alors écrire l'équation (6) sous la forme :

$$\frac{1 - k_3 x_3}{y} = k_1 x_1 + k_2 x_2 + k_3 x_3 \quad (8)$$

Connaissant le membre de gauche ainsi que les x_i, on peut établir une régression linéaire multiple. On dispose de n mesures, donc n équations similaires que l'on peut écrire sous la forme :

$$Y = X * K$$

Avec

$$Y = \begin{pmatrix} \frac{1 - k_3 x_{31}}{y_1} \\ | \\ \frac{1 - k_3 x_{3n}}{y_n} \end{pmatrix}; X = \begin{pmatrix} x_{11} & x_{21} & x_{31} \\ | & | & | \\ x_{1n} & x_{2n} & x_{3n} \end{pmatrix}; K = \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}$$

La régression linéaire multiple s'écrit alors :

$$K = (X^T X)^{-1} X^T Y$$

On a alors l'ensemble des paramètres du système.

2.3.3. Procédure de calibration

1. Calibration de distorsion : l'utilisateur de prend n images (5<n<30) d'une mire (en général un damier) selon différentes positions aléatoires dans le champ de la caméra à calibrer. Une fois les images prises, l'algorithme calcule les paramètres nécessaires à la distorsion inverse et les enregistre dans un fichier déterminé par l'utilisateur.
2. Linéarisation du système : nous avons vu dans la partie précédente qu'on pouvait déterminer certaines grandeurs par annulation des angles de roulis et tangage. Il faut donc positionner le plan de la caméra parallèle à celui du sol et effectuer un balayage sur la zone de mesure (de 60 à 400mm). Plus le balayage est lent, plus il y a de données et plus le dispositif reste horizontal, meilleure sera la mesure.
3. Régression linéaire : ici peu importe le mouvement effectué tant qu'il est lent et reste dans la plage de mesures désirées. On peut toutefois noter que pour plus d'efficacité un balayage intégral de la zone de mesure est à privilégier selon le roulis puis le tangage. (Ou inversement) Une seule acquisition est nécessaire, mais en plusieurs étapes.

A noter que l'étape 1) est indépendante des étapes 2) et 3) mais qu'en revanche l'étape 2) est nécessaire à la réalisation de l'étape 3.

2.3.4. Problèmes et erreurs résultantes

Lors de la calibration en distorsion, nous avons remarqué que la qualité, le nombre des images et que le balayage de la mire dans l'image avait une forte incidence sur la qualité finale de l'image. En effet, si l'ensemble du FOV n'est pas exploité lors du balayage de la mire, il se peut que la distorsion inverse ne soit pas appliquée correctement partout. De même pour le nombre d'images : s'il s'avère trop important, les images de mauvaise qualité vont impacter la mesure. Précision : la calibration a été faîte « à la main », à savoir sans dispositif fixe.

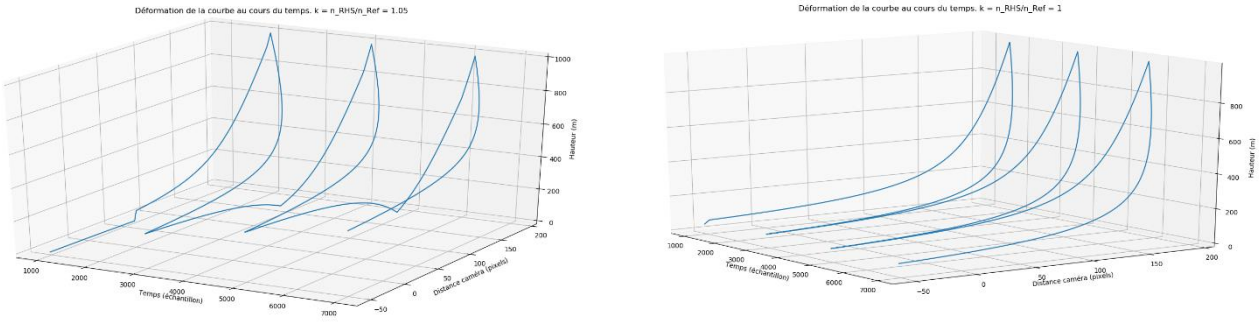


Figure 8 : Hauteur en fonction du temps et de la position du point (en pixels). A gauche avec un nombre de valeurs caméra/référence à 1.05 contre 1 à droite.

Lors de la régression linéaire, aux premières mesures nous avons vu un phénomène d'hystérésis. Celui-ci est provoqué par le suréchantillonnage de la mesure de référence. Notre référence travaillant à 100Hz et la caméra à environ 25Hz, nous avons dû compresser le signal de référence. De plus, les origines des signaux ne sont pas les mêmes de l'un à l'autre, ce qui implique un déphasage. Enfin, le temps entre deux mesures pour la caméra est variable. Tout ceci permet d'expliquer l'apparition d'une hystérésis.

Après simulation, on obtient un écart mesuré de 0.26% sur l'angle α. C'est un écart minime, mais purement théorique. En pratique, on observera des écarts plus grands.

3. Evolutions du projet

Cette partie cherche à recenser les différentes pistes explorées qu'a pu connaître le projet dans cette partie d'étude de faisabilité, permettant d'expliquer les choix expérimentaux (paramètres géométriques) ou d'orientation technologique.

En début de projet la première chose réalisée a été l'étude qualitative du positionnement du laser par rapport à la caméra. Texys étant spécialisée dans les capteurs de petite dimensions, l'écartement du laser à la caméra semblait aller de soi : le plus proche possible avec une marge de manœuvre. Ainsi le choix s'est porté sur 35mm.

L'inclinaison du laser est venue naturellement en partant du raisonnement que lors du croisement des lasers au centre optique, l'inclinaison de 1° du véhicule résulte en une plus grosse variation de la position du spot que lorsque les lasers ne se croisent pas. On améliorerai donc la sensibilité du capteur. On a dû se fixer une limite maximale à ne pas dépasser, estimant qu'on ne pourrait pas capter un spot à plus de 4m. Après simulation sur Excel, il a semblé (de manière qualitative) que le meilleur angle était de 25°, d'où le choix de $\alpha = 25$ °.

La recherche de la position du point a rapidement été au cœur du projet. Une première idée était donc simplement de rechercher le spot sur les valeurs de luminosité. Cependant, les conditions d'illumination étant variables, l'apparence lumineuse du spot le serait aussi. Il a donc fallu trouver une méthode ainsi qu'un domaine dans lesquels l'aspect du point ne changerai pas quelles que soient les conditions d'illumination. Le domaine HSV est un de ces domaines. Cependant la faible connaissance sur la teinte du spot ne permettait pas d'avoir de résultats probants. Le domaine YUV a

alors été choisi et donnait les résultats les plus intéressants à condition de changer le contraste et la saturation de l'image pour conserver le caractère dominant du point quelles que soient les conditions.

Petite précision, il a également été envisagé des algorithmes de détection de contours, mais la spécification du contours précis du spot est floue (au sens propre comme littéral), ainsi que la qualification d'un contours correspondant à un spot. Cela pourrait amener à penser à la transformée de Hough qui permet la détection de formes (notamment de cercles) mais l'imprécision sur le contour d'un spot lumineux a écarté cette hypothèse.

En parallèle de l'écriture d'un script de recherche du point, une recherche dans la littérature a alors été effectuée pour trouver une méthode de positionnement à partir de points laser, aussi bien sur la détection du point que sur le traitement de celui-ci. Malheureusement, la plupart de la littérature (et probablement des recherches) portait sur la segmentation d'une image. Autrement dit la classification d'éléments présents sur l'image. Cette classification pouvait se faire selon différentes méthodes : composantes connexes, formes caractéristiques (ex. : panneau de signalisation) ou encore des méthodes plus mathématiques. Certaines d'entre elles ont été retenues pour un éventuel approfondissement et/ou une réutilisation dans le cadre d'un autre projet ainsi que leur intérêt scientifique. Il est également important de noter que ce temps de recherche n'est pas perdu pour ce projet car comme déjà énoncé, il existe la possibilité d'extrapoler la position du véhicule à partir de la texture du sol.

Une des premières choses à faire a été de déterminer quelle méthode employer pour obtenir les données de roulis, tangage, lacet. Nous avons ainsi prospecté de nombreux articles sur l'obtention des mouvements de caméra à partir des flux optiques jusqu'à prendre connaissance des matrices d'homographie. Cette méthode, utilisée dans le traitement d'images au cinéma permet de reconstruire un environnement 3D à partir d'une succession d'images en cherchant les points 3D fixes de cette image (Ex. : le pied d'une table). Un avantage de cette méthode est qu'elle permet de retrouver la position de la caméra dans la scène. Elle fait appel aux matrices d'homographie.

Après plusieurs recherches, nous nous sommes aperçus que cette méthode ne convenait pas en raison d'un simple fait : pour utiliser les matrices d'homographie, il faut avoir des éléments fixes dans un repère fixe. Or, nous pensions utiliser cette méthode en prenant comme points fixes les spots lasers mais ils ne sont absolument pas fixes dans le repère d'une piste ou d'une route. La méthode n'est donc pas applicable et rend l'utilisation des matrices d'homographie inutile dans le cas souhaité.

Ces recherches ont toutefois permis de trouver des méthodes pour corriger l'effet d'une distorsion par une lentille. En effet, toute caméra possède une distorsion de son image en raison de la présence de lentilles devant son capteur. Il existe des méthodes de traitement d'image permettant de corriger ce phénomène et les matrices d'homographies sont une première approche de cette problématique.

Cela ouvre cependant la voie à un RHS Caméra-Position, qui – sous condition de connaître des points caractéristiques de la route – permettrait de retrouver le vecteur vitesse réel.

A noter que d'autres technologies ont pu être envisagées comme la stéréoscopie donnant une image de profondeur, ou encore comme cité précédemment l'homographie, écartée par un manque de compétence technique. Les technologies IR ont également été envisagées, mais l'apparition de grains en environnement lumineux et le manque d'informations (1 seul canal contre 3 en RGB) ont poussé à rester sur le domaine visible.

Il a paru cohérent de réaliser une calibration sur la distorsion due à la lentille utilisée sur la caméra. En effet, celle-ci présentant un grand angle, la résolution pixel par pixel n'est pas linéaire. Pour la rendre linéaire, un algorithme de distorsion inverse doit être mit en place, et donc une

calibration permettant de sauvegarder et charger à volonté les paramètres servant à cette distorsion inverse.

Une fois tout cela fait, les résultats ont commencé à venir et sont présents dans la partie Résultats et conclusion de ce chapitre.

Résultats et conclusion

Une fois la maquette réalisée et l'algorithme prêt, nous avons réalisé une série de deux mesures. Une première en hauteur uniquement, et une seconde en roulis + hauteur. Suite à la première mesure, nous avons déterminé les paramètres permettant de réaliser la régression linéaire multiple. Cette première mesure nous donne accès au tracé de la distance au sol en fonction de la distance au centre de l'image, laser par laser.

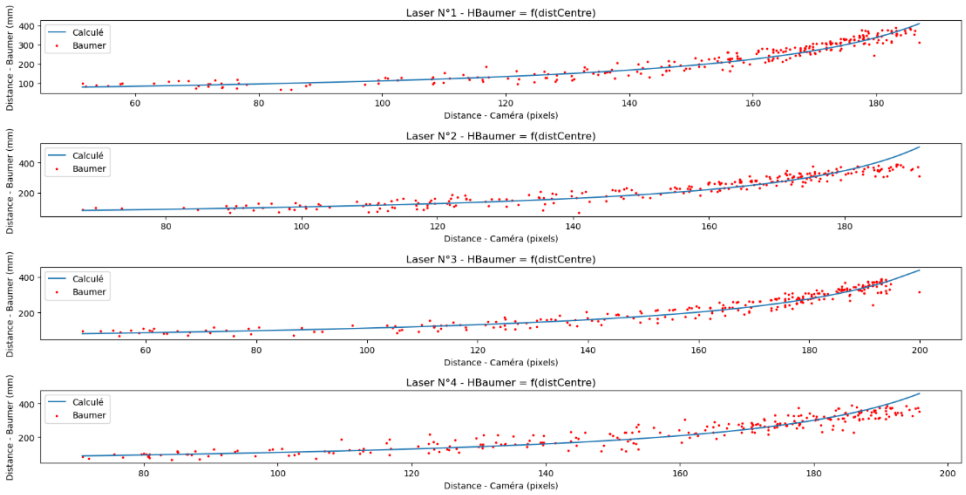


Figure 9 : Tracé de la distance en fonction de la position du spot en pixels. En bleu : la courbe après régression linéaire.

La seconde mesure a permis de déterminer l'ensemble des paramètres suivants. Enfin, nous avons retracé h, θ et φ calculés à partir des paramètres obtenus.

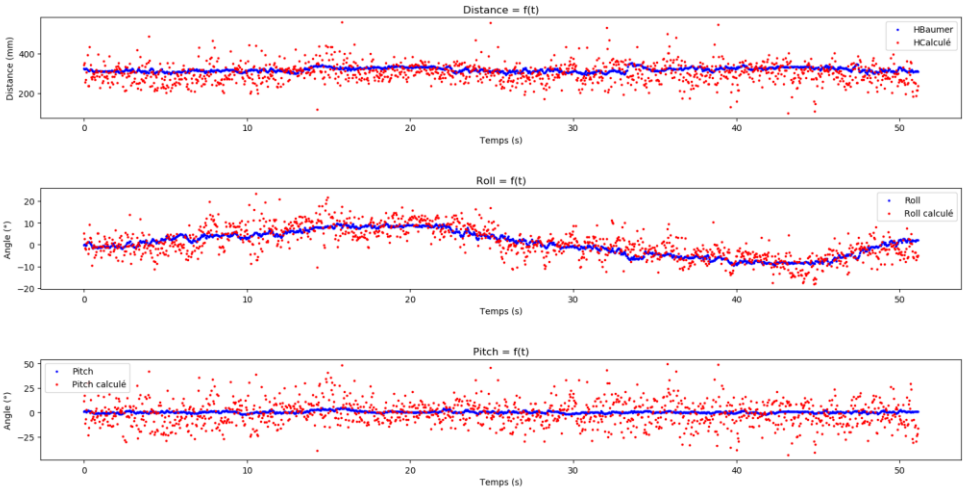


Figure 10 : Courbe de mesure du RHS Caméra (rouge) comparée à la référence (bleu).

Nous avons une mesure très bruitée et inexploitable sur un cas réel, bien que les variations globales semblent correctes. Il faudrait une fréquence d'acquisition bien supérieure pour éventuellement arriver à réduire le bruit par une moyenne glissante.

Ce bruit pourrait être réduit par une meilleure détection des spots. Que ce soit logiciel ou matériel. En effet, nous avons précisé que la caméra utilisée sur la maquette propose des couleurs pâles et peu accentuées. Une amélioration matérielle pourrait aider grandement à la détection et à la précision de cette détection. De plus, les possibilités d'amélioration détaillées dans la section 3 permettraient d'améliorer cette précision.

Il semblerait que ce soit une limitation matérielle qui impacte le plus la précision du dispositif mais il ne faut pas perdre de vue qu'il n'est peut-être pas possible d'obtenir une précision voulue avec la fréquence souhaitée. Ce que l'on peut observer sur les images ci-dessous.

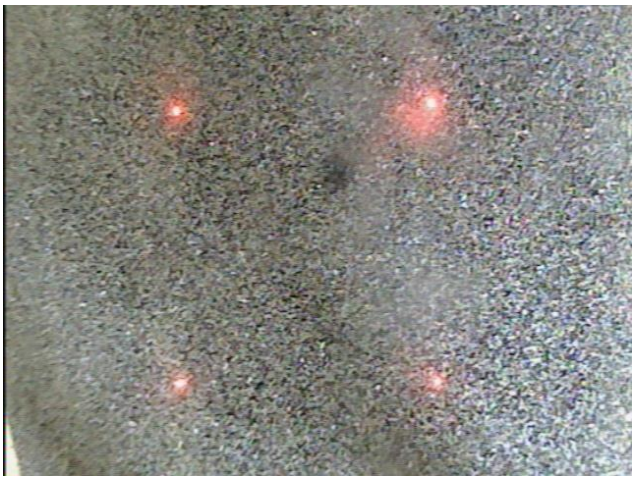


Figure 11: Image provenant de la caméra utilisée

Figure 12: Photo de la scène prise avec un portable.

Le RHS Caméra n'est globalement rien d'autre qu'un projet RHS avec un capteur CMOS 2D. Il semble donc raisonnable de penser que l'on puisse obtenir la précision du RHS mais que le principal facteur limitant est le temps de calcul et donc la fréquence de sortie du RHS Caméra.

Nous tâcherons donc de réaliser un prototype avec le matériel adéquat afin d'améliorer la précision de la mesure. Puis nous verrons également s'il est nécessaire de mettre en place des solutions logicielles pour l'amélioration de la précision.

Il peut également être intéressant d'étudier la modélisation du système et de simuler différents paramètres afin de trouver ceux qui sont optimaux pour le produit final.

Chapitre 2 : Réalisation d'un prototype

1. Choix des composants électroniques et architecture hardware

2. Architecture logicielle

Nous avons les modules suivants : *RhsOS*, *calibration*, *capture*, *processing*, *calculation* et *displayer*. Ils sont chargés respectivement :

- De l'interaction entre les différents modules et de l'entité mère de toutes.
- De la calibration.
- De la capture et enregistrement vidéo.
- De la recherche de spot.
- Du calcul à partir des coordonnées des spots et de l'affichage des images à l'écran.

Il est également important de noter qu'un module « simulation » est également prévu, mais qu'il ne fera pas partie d'une utilisation finale. L'architecture choisie est également plus fonctionnelle. Chaque module se charge de la tâche assignée au départ et ne prend aucun autre module en compte si ce n'est le RhsOS. (Nous verrons un exemple dans ce cas par la suite) Nous allons donc détailler le fonctionnement du logiciel par les fonctions principales. Certaines fonctions peuvent également faire appel à d'autres. (Par exemple la calibration à la capture vidéo.)

Il est également important de noter que le langage utilisé (Python 3) ne sera pas celui implémenté sur l'architecture finale du produit. Cependant, ce langage s'interface bien avec d'autres langages type C ou C++ permettant d'éviter une migration totale du logiciel dans un autre langage de programmation (et donc limiter l'impact de difficultés d'adaptation de la syntaxe d'un langage à l'autre). On pourra donc imaginer un traitement de la vidéo, de recherche de points et de calcul en C (ou langage approprié) tandis que le reste sera géré en Python, le temps n'étant un facteur déterminant que dans ces phases d'acquisition, recherche et calcul.

2.1. Principe

L'architecture logicielle s'articule autour de deux principaux modules que l'on nommera « recherche de spot » et « calcul ». Le premier est certainement le plus lourd et réalise le traitement de l'information brute de la caméra pour en extraire l'information utile : la position x, y de chacun des spots. C'est également la partie la plus sensible en termes de précision : une erreur minime impacte grandement les résultats. (Cf. chapitre 3) La partie calcul forme les informations client à partir des informations utiles fournies par le premier algorithme.

Enfin, un ensemble d'autres modules sont nécessaires pour faire fonctionner l'ensemble du logiciel. Ces modules annexes vont du « système d'exploitation » du RHS Caméra à la simulation en passant par la calibration.

Nous recensons actuellement 12 modules dans l'ensemble permettant de configurer et de simuler le RHS Caméra. Les modules nécessaires au fonctionnement sont :

- *Calculation* : module pour le calcul du roulis/tangage et de la hauteur de caisse à partir d'une régression multi-linéaire.
- *Calibration* : module servant à calibrer la caméra pour réduire les effets de distorsion dus à la lentille et pour obtenir une valeur expérimentale des constantes de la régression multi-linéaire.
- *Capture* : module pour la capture vidéo à partir d'un périphérique externe (live) ou d'une vidéo pré-enregistrée.
- *Displayer* : module servant à afficher en direct l'image avec les valeurs calculées. NON FONCTIONNEL
- *Processing* : module réalisant la recherche de spot.
- *rhsOS* : module coordonnant l'ensemble des interactions entre les différents modules.

2.2. Recherche de spot « *processing* »

La recherche de spot est articulée en différentes actions successives. Elle s'effectue tout aussi bien dans le module du système d'exploitation que dans la partie qui lui est consacrée. Une image contient 4 spots à retrouver dans un des quarts de l'image. On applique ainsi les mêmes étapes à chaque quart d'image :

1) Redimensionnement de l'image : cette étape sert à diminuer le nombre de pixels et donc le temps de traitement de la recherche de spot. Ce redimensionnement est dans un premier temps statique (On passe d'une image pleine à un quart d'image) puis dynamique : on détermine le nouveau cadre en fonction de la position précédente du spot.
2) Traitement de l'image et conversion : l'image est ensuite traitée en réhaussant le contraste et en diminuant la luminosité (valeurs expérimentales non optimisées) puis convertie en YUV. L'image YUV est ensuite séparée en 3 canaux (Y ; U et V) puis un traitement similaire à l'article « Vision based distance measurement system using single laser pointer design for underwater vehicle » est appliqué. L'image est également filtrée durant ce processus afin de réduire le bruit de conversion.
3) Recherche du spot : l'image renvoyée est alors seuillée pour faire disparaître tout artefact pouvant gêner la détection et une recherche de spot est alors opérée. Cette recherche est pour l'instant « simple » : elle prend la moyenne des indices dont la valeur est supérieure à un seuil. C'est une méthode rapide, mais sensible aux fausses détections. Une recherche par pondération d'indices selon la valeur est également possible.

Chacune de ces étapes possède des valeurs de paramétrage fixées expérimentalement et selon le « bon sens ». (Taille de redimensionnement fixe et dynamique ; seuils ; valeur de contraste et luminosité)

2.3. Calcul « *calculation* »

L'algorithme de calcul fait intervenir le modèle précédemment expliqué ainsi que l'ensemble des formules précisées. Il existe deux cas (comparés au chapitre 3) selon les formules employées.

Dans chacun des cas, l'algorithme prend en paramètre les positions des pixels exprimés dans le repère de l'image et calcule par régression multilinéaire l'ensemble des données dynamiques nous intéressant.

L'erreur de l'algorithme de calcul est négligeable sur les plages d'utilisation conventionnelles des véhicules. (Cf. Chapitre 3)

2.4. Annexes

En soit, seul le *rhsOS* est nécessaire au fonctionnement logiciel du RHS Caméra, à condition qu'un remaniement soit fait et que l'image lui soit directement fournie. Le module *Capture* sert simplement à faciliter la répétition de captures vidéo, aussi bien en direct qu'une vidéo pré-enregistrée.

Le module *displayer* est pensé pour permettre l'affichage configurable des différentes étapes du traitement pour débogage ou validation d'étapes. Ce module n'est plus fonctionnel et pourrait être implémenté en brut au sein du *rhsOS* pour outil d'évaluation temporaire. Cependant, ce n'est pas une utilisation recommandée.

Le module *calibration* n'est actuellement plus utilisé (bien que nécessaire à l'importation) car bloquant pour le reste du logiciel en termes d'exécution au démarrage : l'algorithme de calibration se débrouille pour chercher, selon la configuration utilisateur, les valeurs par défaut à utiliser dans le module *calculation* pour l'évaluation des différents paramètres dynamiques attendus. Il a été mis de côté à partir du moment où l'on a cherché à évaluer la précision de l'ensemble de l'algorithme. Il sera cependant utile pour compenser la déformation d'une lentille et éventuellement d'obtenir des paramètres statiques expérimentaux et non théoriques. Un remaniement de ce module est également à prévoir.

Enfin, les modules de simulation *compareCalcul, comparePoints, compareIntegrated* cherchent à évaluer (au prix d'un lourd calcul) respectivement l'impact en termes de précision de l'algorithme de calcul puis de l'algorithme de recherche de point et enfin de l'ensemble intégré. Chacun de ces modules fait appel au module *RHSobject* qui simule le comportement d'un véhicule selon une loi donnée conformément au modèle détaillé précédemment. (Exemple : sinus en tangage) Ce dernier module cherche à modéliser informatiquement le comportement des lasers selon le positionnement du véhicule sur la route en prenant en compte notamment la focale de la caméra etc.

Conclusion

Ce chapitre 2 a plus eu une vocation de recherche d'informations sur la mise en place d'un RHS Caméra et sur la correction de défauts dans l'algorithme. Il a permis de découvrir l'existence de microcontrôleurs et d'implémentations Python dans l'embarqué ; de faire une revue des capteurs CMOS disponibles sur le marché et enfin de comprendre l'intérêt de modules caméra déjà réalisés : la réalisation de modules caméra n'est pas dans le domaine d'expertise de Texys/H2 Motronics, et par conséquent, le coût de développement et d'acquisition de ces connaissances serait important en termes de temps.

Chapitre 3 : Validation

Dans les chapitres précédents, la validation de faisabilité du projet n'a reposé que sur la présence de courbes bruitées et peu exploitables. Cette faisabilité a été établie par l'expérimentation et la faisabilité du projet n'a aucunement été validée par la théorie si ce n'est par la recherche de formules et expliquées dans le début du chapitre 1.

Ce chapitre n'a pas vocation à réfuter ou à remplacer l'étude de faisabilité mais bien à confirmer le socle théorique et pratique sur laquelle elle se repose ainsi qu'à fournir la validation des hypothèses émises quant aux résultats mitigés obtenus en chapitre 1.

Cette phase de réalisation possède un gros avantage sur les phases précédentes : de la littérature sur le sujet a été amassée ainsi que des connaissances d'expérimentation et enfin, le matériel à disposition est de meilleure qualité qu'auparavant et l'on dispose surtout de plus de connaissances sur celui-ci. (Focale de la PiCamera, taille des pixels etc.)

Nous étudierons donc la performance des deux parties principales de l'algorithme RHS, à savoir :

- La recherche des points sur l'image
- L'interprétation de la position des points : calcul de la hauteurs et des angles de roulis, tangage.

Pour le reste du chapitre nous définirons deux types de paramètres :

- Paramètres géométriques qui définissent la mécanique du système. (α, β, f et e)
- Paramètres dynamiques qui définissent la position du système. (h, θ et φ)

On définit les critères de performance de l'algorithme de calcul et de celui de recherche de point comme étant respectivement « l'erreur dynamique » et « l'erreur de positionnement ». L'erreur de positionnement possède une composante par point.

$$\varepsilon_{dyn} = x - \hat{x}$$
$$\varepsilon_{pos}^{i} = d_2^i - \widehat{d_2^i}$$

Où x est le vecteur à 3 dimensions des paramètres dynamiques réels (h, φ et θ) et $\hat{x}$ le vecteur à 3 dimensions des paramètres dynamiques estimés par l'algorithme de calcul.

Où d_2^i est définit comme dans le chapitre 1 et $\widehat{d_2^i}$ la valeur estimée par l'algorithme de recherche de point.

L'algorithme RHS, à savoir l'algorithme résultant de l'intégration de l'algorithme de calcul et de celui de recherche de point, aura comme critère de performance l'erreur dynamique précédemment définie. La différence avec l'algorithme de calcul est que les paramètres dynamiques estimés dépendront également des distances CMOS estimées par l'algorithme de recherche de point, ce qui appuie l'intérêt d'une connaissance de l'erreur de positionnement.

On ne définit pas de critère unique pour l'erreur de positionnement afin de pouvoir conserver un regard précis sur une potentielle source d'erreurs.

1. Validations théoriques

1.1. Première approche

1.1.1. Simulation

La simulation se base sur le modèle mathématique expliqué dans le premier chapitre. Nous allons chercher à complexifier le modèle de simulation au fur et à mesure de la validation des étapes de manière à simuler la réalité au plus près possible (Dans la limite du raisonnable en termes de temps) afin de voir l'influence des différents paramètres et chercher une solution aux problèmes existants.

La simulation permet également d'évaluer différents paramètres géométriques pour une optimisation mécanique du système et indépendante du type de caméra utilisé.

A partir des positions simulées des points laser dans le plan de la caméra, on en détermine une image qui sert d'image de base pour l'algorithme de recherche de points. (Prise en compte de la résolution du capteur et de ses dimensions : la taille d'un pixel peut remplacer les 2 paramètres) On a également accès à la position de ces points dans le plan de la caméra, ce qui donne la possibilité d'évaluer directement l'algorithme de calcul.

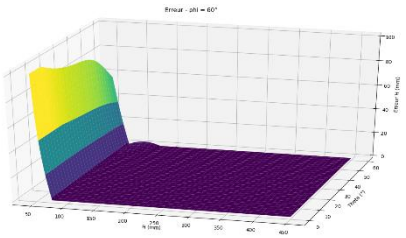
1.1.2. Algorithme de calcul

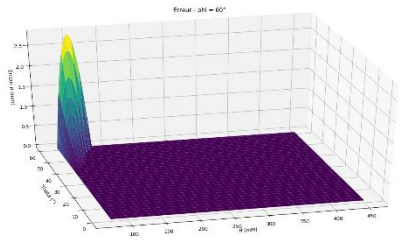
Nous allons évaluer la précision de l'algorithme de calcul. Pour cela, nous avons tout d'abord corrigé et validé le fonctionnement des formules trouvées chapitre 1. Nous avons également implémenté les deux solutions de calcul sous la forme « complete » ou « direct » selon que l'on utilise respectivement la formule (4) ou (5).

Dans une première partie, on suppose les paramètres géométriques connus, tandis que dans la seconde on passera par un processus de calibration. Cependant, une mesure réelle sera nécessaire pour déterminer l'efficacité d'une calibration comparée à la connaissance théorique des paramètres.

La simulation effectuée a été effectuée avec comme paramètres géométriques ; ils sont supposés connus :

$$\alpha = -25°; \beta = 45°;\ f = 5\ mm; e = 35\ mm$$
$$L1 \times L2 = 3.68\ mm \times 2.76\ mm$$
$$resX \times resY = 768 \times 576$$





Avec L1 et L2 les dimensions du capteur CMOS. (Valeurs prises sur le capteur CMOS de la PiCamera) et resX, resY les dimensions en pixels. En réalité, seules les dimensions d'un pixel et la résolution en pixels sont nécessaires.

Nous avons réalisé le calcul d'erreur dynamique pour chaque $x \in \mathrm{I} = [50\mathrm{mm}\,;450\mathrm{mm}] \times [0°;60°] \times [0°;60°]$ avec un pas de 1mm et 1°.

Une première région d'erreur peut être définie de 50 à 75mm. La valeur de 75mm est due à la configuration mécanique choisie : il s'agit de la distance entre la lentille de la caméra (point A) et le point de croisement de l'ensemble des lasers. Cette distance c_{laser} est définie comme :

$$c_{laser} = \frac{e}{\tan(|\alpha|)} = 75.05\ mm$$

On retrouve bien cette limite. Il existe alors deux possibilités pour diminuer la distance minimale de détection :

- Changer les paramètres géométriques. Par exemple en ne croisant pas les faisceaux ou en déterminant un autre range minimum.
- Revoir les formules pour essayer de faire apparaître non plus la valeur absolue de α mais bien la valeur réelle.

1.1.3. Algorithme de recherche de spot

La recherche de spot est un élément critique du RHS Caméra : c'est l'algorithme qui sert à renvoyer des données primaires. La qualité de ces données est donc primordiale. L'algorithme doit donc fonctionner par tout temps, toute condition qu'un circuit peut rencontrer et surtout renvoyer des données stables.

L'algorithme regroupe deux parties essentielles :

- La mise en forme de l'image
- La recherche du spot

Nous avons catégorisé 3 types de mise en forme de l'image (bien qu'il puisse en exister d'autres). Nous les avons baptisés de la manière suivante :

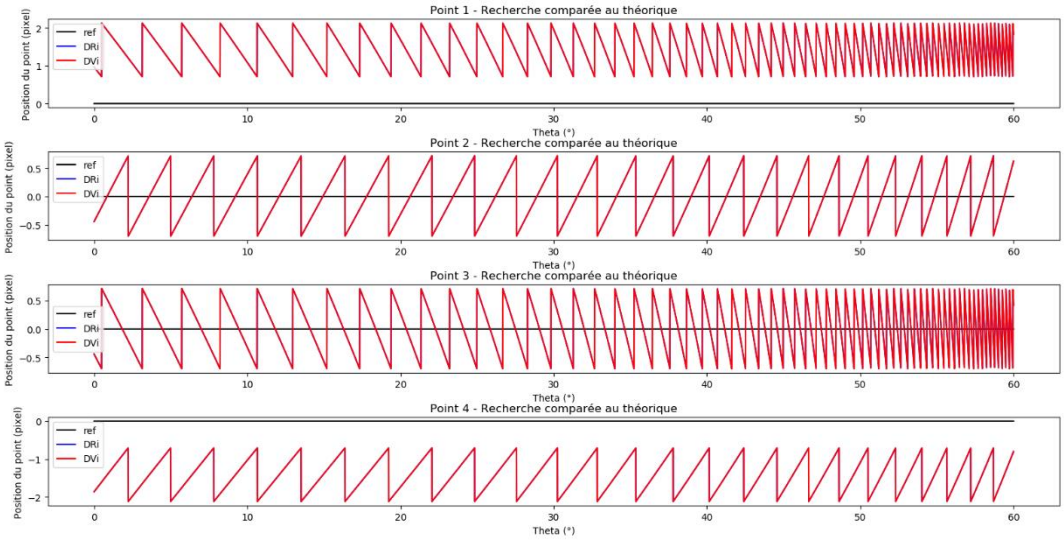
- Méthode YUV : basée sur la recherche du spot sur le canal V d'une image YUV. C'est la première version de l'algorithme mise en place.
- Méthode DRi : basée sur les formules d'un article scientifique [ref.], elle cherche à évaluer le degré de rouge dans une image via un calcul sur les différents canaux permettant la création d'une image noir et blanc.
- Méthode DVi : version la plus aboutie, elle est basée sur la méthode DRi appliquée à une image passée dans le domaine YUV et cherchant donc à déterminer le degré de V dans l'image.

Toutes ces méthodes nécessitent que l'image soit correctement contrastée et saturée auparavant. Au cours des différentes expérimentations, nous avons remarqué que ces valeurs de contraste/saturation ne semblaient pas varier énormément. Ce réglage impacte également le bruit de l'image, ce qui nécessite de la filtrer. (Typiquement filtre médian mais peut être configuré)

Enfin, la méthode de recherche en elle-même peut être différente. La méthode actuellement utilisée est par moyenne sur les indices, ce qui suppose que l'image envoyée ne possède que le spot. Des méthodes de recherche logarithmique, ou encore en moyenne pondérée sur les indices existent

mais leur implémentation possède un coût calcul et de temps de développement importants, et nous préférons privilégier d'autres pistes d'améliorations telle que la qualité de la caméra ou de l'image.

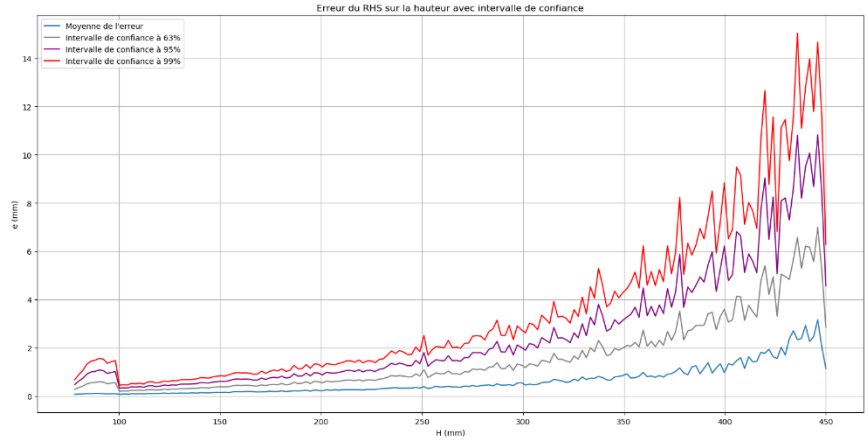
Nous avons ainsi réalisé un comparatif des méthodes DRi et DVi sur une image parfaite. Il n'y a pas de différence. On observe ainsi une symétrie sur les différentes courbes due à la géométrie du système. On voit également une erreur systématique jusqu'à $\pm 1.5\ pixel$ et une erreur de précision de $\pm 1\ pixel$. A noter que cette simulation a été effectuée sur une consigne en roulis pure [0° ; 60°] avec 10 000 points et les mêmes paramètres géométriques que décrits dans la partie sur l'algorithme de calcul.

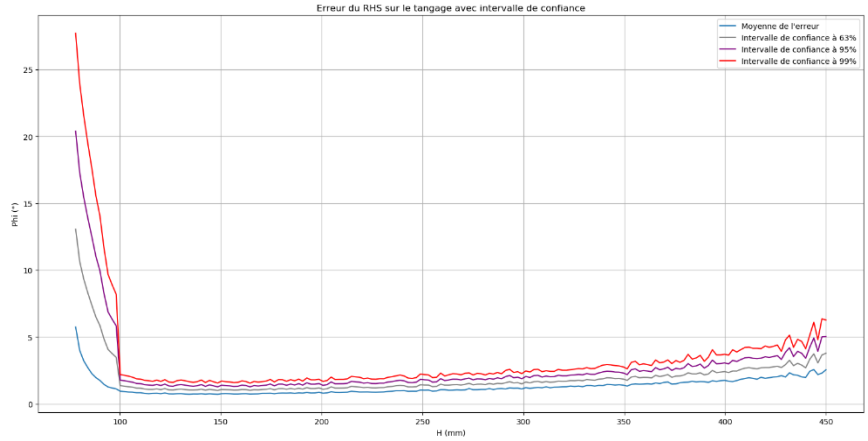


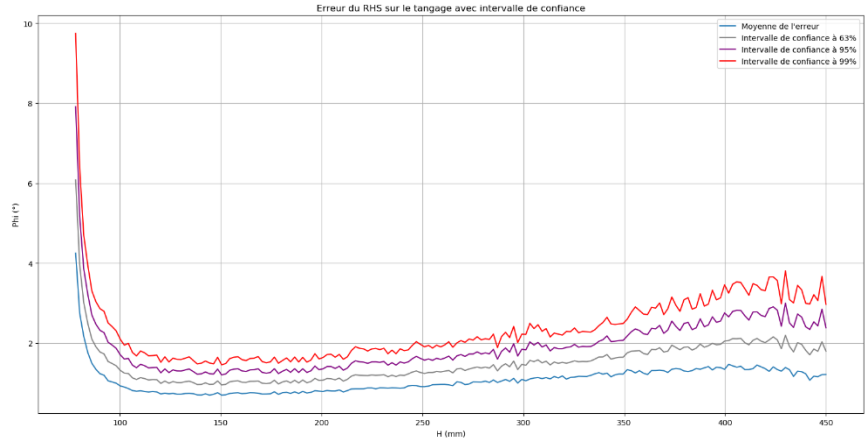
On a donc une confirmation du fonctionnement de l'algorithme de recherche de point sur une image parfaite, à savoir : noir avec des spots en forme de carré rouge. Le manque de précision est dû à la précision de l'ordre du pixel renvoyée par l'algorithme de recherche (moyenne d'indices). Si l'on veut augmenter cette précision, il faut donc changer la méthode de recherche.

1.1.4. Intégration : Algorithme RHS

Une fois ces calculs effectués, nous avons mis bout à bout les deux algorithmes pour en ressortir les données finales théoriques. Cette étape de vérification permet d'identifier l'influence d'une imprécision de l'algorithme de recherche sur les données finales, et de prendre également en considération les problèmes calculatoires pouvant exister. (Cf. Partie précédente)







Nous pouvons ainsi observer un accroissement de l'erreur avec l'augmentation de la distance. La courbe obtenue est obtenue de la manière suivante :

- Création d'un tableau de dimensions $nT \times nP \times nH$ (Dimension 3) où nT, nP et nH sont respectivement les tailles d'échantillons de θ, φ et h. Le calcul a été réalisé sur une plage de 50-450mm ; 0-60° pour chaque angle avec un pas de mesure de 1mm et 1°.
- Calcul de la moyenne de l'erreur sur θ et φ sur le tableau renvoyé. (Moyenne sur $nT \times nP$ valeurs)
- Affichage de la moyenne sur les nH valeurs restantes.

Les courbes obtenues sont à prendre avec précaution puisque pour avoir une idée représentative de l'erreur en fonction des paramètres il faudrait afficher $e = f(h, \theta, \varphi)$, donc de disposer d'un affichage de dimension 4, ce qui est impossible sur un seul graphique ; ou alors par une évolution temporelle sur un des paramètres.

Les courbes affichées ne sont donc qu'une représentation/projection de la réalité et n'illustrent en rien la complexité du problème. (Domaine de définition des angles…)
Les courbes ont été choisies de manière à être le plus illustratives quant à la précision du RHS Caméra.

On s'aperçoit ainsi d'une augmentation de l'erreur en termes de précision à mesure que la distance augmente. Au vu de l'évolution des courbes 3D on peut estimer qu'il existe en effet une augmentation de l'imprécision avec l'augmentation de la distance. Cette imprécision est également augmentée par l'augmentation des angles de roulis et tangage.

On n'atteint pas les performances attendues (1° / 1mm). Cependant, on observe des données correctes pour une distance inférieure à 250mm.

On observe également l'influence de l'erreur de l'algorithme de calcul pour des valeurs inférieures à 100mm comme dans la partie précédente.

1.1.5. Conclusion

Nous avons donc évalué les différentes parties de l'algorithme RHS, tout d'abord indépendamment puis de manière intégrée. L'évaluation indépendante a permis de mettre en évidence les différents paramètres que l'on peut faire varier pour améliorer la précision de l'algorithme global.

On observe ainsi qu'il y a deux parties distinctes dans l'algorithme qui sont le calcul à partir de la position des spots et la recherche des spots.

Le calcul est principalement limité par la disponibilité des spots puisqu'il en faut au moins 3 pour pouvoir déterminer les paramètres dynamiques. Il est également influencé par les paramètres géométriques qui vont alors déterminer la gamme d'utilisation du capteur. Cette gamme, bien que déterminée en théorie, est une limite qui ne pourra jamais être atteinte en réalité.

La recherche de point possède quant à elle 4 axes d'amélioration :

- Les paramètres géométriques.
- La qualité d'image initiale. (A savoir définie par la caméra)
- La qualité d'image en pré-traitement. (Filtrage, méthode DVi…)
- La méthode de recherche du point. (Moyenne d'indice, moyenne pondérée…)

Cependant, la technologie utilisée, à savoir des transformations successives sur l'ensemble de l'image, n'a pas été remise en question. Il existe d'autres méthodes telles que la recherche par réseau de neurones ou la catégorisation d'image à l'aide de blobs… Ces méthodes nécessitent une connaissance plus approfondie du traitement d'images.

Enfin, l'intégration des deux algorithmes a produit des données insatisfaisantes. L'observation de ces données remet ainsi en cause le projet RHS Caméra. Cependant, plusieurs moyens d'action existent au vu de la multi-paramétrisation du système.

Premièrement, une revue de l'algorithme de recherche de point peut être envisagée, prioritairement sur la méthode de recherche du point.

Deuxièmement, une nouvelle paramétrisation géométrique (optimisation des paramètres) peut être effectuée et a été une première voie d'amélioration. En effet, l'ajout de l'angle α est issu d'une intuition en début de projet pour augmenter la sensibilité du système. Cependant, il semblerait que la valeur d'angle ait été trop importante jusqu'à présent.

Une simulation avec un angle $\alpha = 0°$ est prévue pour observer l'influence sur les données. Il est également important de noter que le lancement d'une simulation dure près d'1h et est fonction de la résolution choisie. Il est donc inenvisageable de procéder à une optimisation par itération sur les paramètres géométriques.

Conclusion

L'étude théorique a révélé que la précision sur les paramètres dynamiques ne pouvait être clairement évaluée comme une combinaison linéaire des erreurs sur l'algorithme de recherche et celui de traitement. Une précision de l'ordre de 1mm et 2° a été obtenue sur une plage d'environ 75-250mm et de 0° à 60° pour un réglage d'angle α de -25°. L'amélioration de ces résultats fait intervenir différents facteurs. Ils sont de deux sortes : géométriques et algorithmiques.

L'amélioration du RHS Caméra par géométrie nécessite une étude amont plus approfondie permettant d'évaluer avec précision les paramètres géométriques optimaux pour obtenir la précision et le range souhaités, cette étape est très lourde en termes de calculs et nécessite plusieurs jours (voire semaines) de traitement et d'analyse. Une méthode d'optimisation basée sur des calculs mathématiques du type calcul variationnel.

Cependant, il faut également mener sur le même front une étude d'amélioration algorithmique. En effet, cette amélioration concerne l'algorithme de recherche de spot : un algorithme fiable et stable quelques soient les conditions environnementales est l'élément le plus déterminant quant à la précision du RHS Caméra en fonctionnement. Nous pouvons améliorer la précision par la méthode de recherche du centre (moyenne d'indices ; moyenne pondérée…) ou par l'amélioration du traitement en lui-même pour augmenter le rapport signal/bruit. Cette partie peut être réalisée en simulations mais nécessitera toujours une confirmation expérimentale.

2. Validation pratique

2.1. Evaluation de la vitesse de traitement

Le cahier des charges fait apparaître une fréquence de fonctionnement de 100Hz, nous devons donc nous assurer de la faisabilité technique de cette exigence. Pour cela, il a été décidé d'utiliser la Raspberry Pi4 avec le module caméra Pi Camera pour réaliser cette évaluation : c'est un moyen simple et rapide d'évaluer un programme embarqué.

2.1.1. Implémentation

Les premiers essais ont été effectués rapidement, avec juste quelques corrections de bugs dus à l'importation des algorithmes sur la Raspberry Pi 4, disposant d'un quad core cadencé à 1.4 GHz. (quad core64-bit ARM-Cortex A72)

Le programme écrit tourne sous CPython 3.7. Il fait appel à 5 modules : *rhsOS, capture, processing, calculation, calibration*. Afin de fonctionner correctement ces modules font appel aux modules *opencv*, *numpy*, *keyboard* et *time*.

Les programmes réalisés dans le cadre du projet permettent d'acquérir l'ensemble des informations souhaitées par l'utilisateur de manière modulaire. Ainsi, on peut choisir quelles sont les données à acquérir, de même pour les vidéos issues des différents traitements d'image réalisés. On peut également choisir dans quel dossier faire figurer tous ces éléments. Il est également possible d'utiliser une caméra (live) ou une vidéo enregistrée en le spécifiant à l'algorithme.

Après une première mesure (données non disponibles) nous avons estimé à environ 2Hz la fréquence d'acquisition des paramètres dynamiques. Cette fréquence faible s'explique en partie par la définition des fenêtres de calcul. En effet, avec la présence de 4 lasers et un angle de déphasage caméra $\beta = 45°$, chaque spot est présent dans un quart de l'image. La recherche du point s'effectuait donc sur le quart entier de l'image. Or dans des versions précédentes du programme, la recherche était dynamique : elle dépendait de la position du point précédent.

L'utilisation de ce fonctionnement nécessite donc d'expliciter une hypothèse de petites variations. En effet, on suppose que la position du point à instant $t_0 + dt$ est suffisamment proche de la position du même spot à l'instant t_0. Autrement dit, la variation du point est suffisamment petite considérant la taille de la fenêtre de redimensionnement et la fréquence d'acquisition.

Il se peut néanmoins que le point soit perdu. Dans ce cas, on part du principe que le point sélectionné sera à un endroit aléatoire dans l'image. A ce moment, le cas des petites variations ne s'applique plus. Afin de déterminer si l'on est toujours dans ce cadre théorique, nous effectuons un seuillage par calcul de distance cumulée sur $N = 5$ valeurs. Cette distance est définie comme la somme des distances de la position d'un spot à sa position précédente. Soit x_i et y_i les positions en pixel dans l'image du spot suivi à l'instant i. On a alors le critère S_i (mvtLastPts dans le programme) qui s'écrit :

$$S_i = \sum_{j=i-5}^{i} \sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2}$$

Si notre critère dépasse une certaine valeur (expérimentale arbitraire), alors nous recalculons sur l'ensemble du quart d'image.

Nous pouvons également noter que le dimensionnement du quart d'image est possible. Il y a ainsi 3 niveaux de zoom sur l'image :

- L'image originale.
- L'image pour 1 spot « statique ».
- L'image pour 1 spot « dynamique ».

Le premier niveau de zoom est défini par la qualité de l'image. (Dans les mesures effectuées 640×480) Les deux autres peuvent être choisis par l'utilisateur. L'image statique est définie comme la région de l'image dans laquelle la probabilité de trouver le spot est égale à 1. L'image dynamique sert juste à améliorer la vitesse de calcul.

Nous avons ainsi corrigé et réimplémenté cette fonctionnalité puis nous avons effectué des mesures, avec et sans affichage (du nombre d'images prises) puis avec et sans acquisition d'image. (1 seule sur l'ensemble du traitement).

2.1.2. Résultats

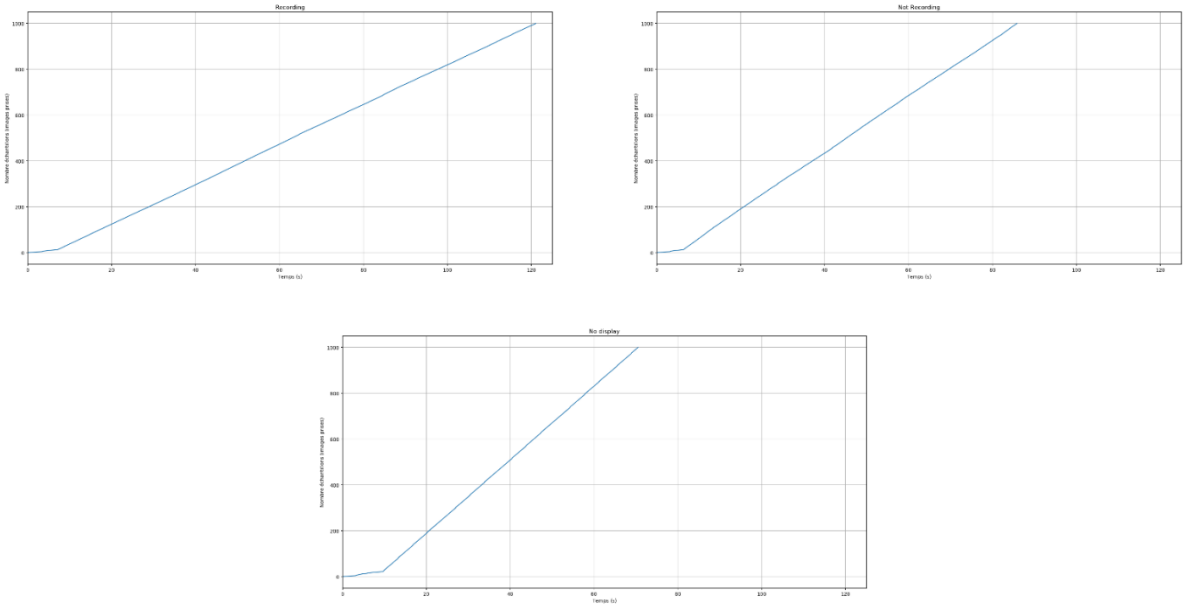
Nous observons ainsi dans chaque cas un régime transitoire qui résulte de la mise en place du redimensionnement et du critère de seuillage. Ce régime transitoire fait appel à un calcul sur les images statiques de chaque spot, d'où une lenteur car on doit évaluer une taille d'image importante. Ensuite, lorsque l'image dynamique est en fonctionnement, on observe une hausse de la fréquence.

Cette fréquence résultante est bien sûr fonction de l'affichage à l'écran mais également de l'acquisition de données. Nos mesures ont été faites sur 3 échantillons de 1000 images à chaque fois. Les mesures ont été effectuées à la suite (peu de variations de lumière sur l'image d'un essai à l'autre) et ont eu la même configuration exceptée :

- Mesure 1 : « No display ». Aucun affichage, acquisition des données de temps, paramètres dynamiques, positions des spots en pixels.
- Mesure 2 : « Not recording ». Affichage du nombre d'images traitées, acquisition des données de temps, paramètres dynamiques, positions des spots en pixels.
- Mesure 3 : « Recording ». Enregistrement d'une vidéo contenant l'image issue du capteur, affichage du nombre d'images traitées, acquisition des données de temps, paramètres dynamiques, positions des spots en pixels.

Nous observons respectivement des fréquences de régime transitoire et continu :

- Mesure 1 : « No display ». Transitoire : 2.31Hz ; Continu : 20.10Hz.
- Mesure 2 : « Not recording ». Transitoire : 2.66Hz ; Continu : 15.89Hz.
- Mesure 3 : « Recording ». Transitoire : 2.30Hz ; Continu : 10.90Hz.



Echantillons en fonction du temps : En haut à gauche « Recording » ; en haut à droite « Not recording » ; en bas « No display ».

Nous voyons donc l'influence importante qu'a l'acquisition des données ainsi que l'affichage de messages en plein fonctionnement. Il faut donc impérativement limiter ces messages ou faire tourner les traitements en tant que « tâches de fond » et ne pas inclure les messages dans le traitement.

Nous avons également des jeux de données pour analyser la stabilité de la recherche de point. Cependant, les algorithmes de calcul sont complètements faussés (et donc les paramètres dynamiques renvoyés) en raison d'une absence de calibration et d'un manque de normalisation sur l'essai : l'ensemble des paramètres ne peut être contrôlé finement.

Conclusion

Nous avons donc réussi à faire fonctionner le RHS Caméra avec une fréquence d'acquisition jusqu'à 20Hz. Ce résultat est engageant mais est limité par de très nombreux facteurs. Le premier d'entre eux est la résolution de l'image qui influence directement le nombre de pixels présents et sur lesquels faire le calcul. Une solution de redimensionnement à centrage dynamique a été conçue pour limiter le calcul effectué étant donné que la majorité de l'information présente dans l'image est une information inutile : autant ne pas la traiter.

Il faut également noter que la lumière aura un impact sur la fréquence. En effet, une scène illuminée va résulter en une instabilité sur les spots et donc risque de provoquer un décrochage de l'algorithme de recherche. Ce décrochage risquant de déclencher le seuil déterminant la validité de l'hypothèse de petites variations.

L'utilisation de CPython, bien qu'efficace dans beaucoup de cas et intéressant pour une rapidité de développement logiciel, engendre une augmentation des temps de calcul. C'est un problème connu et chiffré [1].

Nous réalisons une recherche de point sur 4 spots. Il serait donc judicieux de paralléliser le processus et donc d'utiliser soit plusieurs microcontrôleurs soit un micro multicœurs pour réussir à

obtenir les positions des spots en même temps et ainsi économiser du temps. Cependant, l'utilisation de ces technologies n'est pas dans le domaine de compétence de Texys.

Il y a également la caméra utilisée qui ne fonctionne pas jusqu'à 100Hz. (Fonctionnement normal à 60/90Hz pour cette résolution)

Enfin, l'utilisation d'un OS (ici Debian – système basé Linux) interfère avec le calcul à cause de l'utilisation de ressources via des services tournants en fond.

Il est également important de noter que les résultats présentés ici ne sont aucunement fiables en termes de justesse des données renvoyées comme expliqué précédemment : les paramètres géométriques sont connus avec trop peu de certitude. De plus, il se peut qu'il y ai nécessité d'augmenter la résolution pour permettre un calcul précis des paramètres dynamiques.
L'acquisition d'images nécessite l'utilisation d'une autre caméra qu'une PiCamera. Celle-ci ne peut servir qu'en termes de validation de précision faible vitesse ainsi qu'une première estimation pour la vitesse de traitement mais ne saurait faire foi pour un traitement à haute vitesse.

En conclusion, la faisabilité technique y est à condition de mettre les ressources en termes de temps de développement et d'argent : Framos, un expert du développement de modules de caméra assure qu'il faut au moins un an pour réaliser un module caméra. Un kit d'évaluation complet coûte environ 1000€.

3. Pistes d'amélioration

3.1. Technologie

La première piste d'amélioration se base sur le type de technologie employée pour l'obtention des paramètres dynamiques. Nous avons prévu une technologie par triangulation laser pour évaluer la hauteur de caisse. Il existe également des technologies de vision stéréoscopique, cependant les questions restent les mêmes que pour un RHS Caméra Triangulation : quelle précision et quelle vitesse d'acquisition ? Cette fois-ci il faudrait traiter deux flux vidéo à 100 images par seconde ou plus, avec reconstruction 3D de l'environnement à cette fréquence ainsi qu'une bonne précision et du pré-traitement nécessaires (filtrage etc).

Nous pouvons également ré-évaluer l'impact d'une recherche de spot dans l'infrarouge. En effet, les premiers tests non concluants ont été réalisés avec une caméra de surveillance grand public qui cherche à adapter son image en fonction du flux lumineux etc. On obtient donc un flux vidéo donc le contraste, l'obturation de la caméra… varient, ce qui ne permet pas de conclure à un test probant. Nous avons réalisé le test en plein été au soleil sur un sol bitumeux et avons aperçu une granulosité du spot. D'autant plus que la caméra infrarouge est alors sur un seul canal. Il se peut que le soleil ou la chaleur du sol perturbent la caméra infrarouge. (Il faut donc trouver le bon spectre) Malgré tout, nous n'avons pas suivi cette idée car le dispositif de détection IR dont nous disposions ne montrait pas de rapport signal sur bruit suffisamment important. De plus que des capteurs de ce type ne sont pas les plus répandus.

On peut penser en effet à appliquer un filtre de Kalman pour améliorer les mesures à condition d'exprimer le système sous forme d'une équation différentielle.

Il y a également la possibilité de donner des informations concernant la validité d'une mesure. En effet, grâce à la calibration, nous sommes en mesure de déterminer des courbes théoriques et grâce aux mesures de précision, un intervalle de confiance qui ensemble nous permettent de déterminer la validité ou non d'un point spécifié.

3.2. Hardware

La maquette a déjà permis de relever quelques soucis de conception qu'il faut éviter. En effet, lors de la pose des lasers, nous n'avions pas de référence. Il faut alors disposer de montures réglables afin de positionner au mieux les lasers, puis d'immobiliser le système une fois réglé jusqu'à un nouveau réglage (ou définitivement selon ce qui est choisi). Cela permet un réglage fin lors de la pose des lasers. En effet, à chaque position du véhicule, correspond une distance des lasers à l'axe. Ainsi, lors du réglage de ceux-ci, il suffira de se mettre à une certaine distance, centrer la caméra sur une mire et positionner les lasers sur des points précis de cette mire, assurant un angle optimal. Limitant la nécessité d'une calibration de β et de la régression linéaire et donc réduire les erreurs dues à la mesure. Cependant, cela ne rejette pas totalement l'intérêt de ces deux calibrations qui non seulement diminueront l'incertitude mais qui peuvent être nécessaires si un changement de géométrie survient lors de la vie du produit. (Usure, choc, vibrations…)

Autre piste d'amélioration : le circuit FPGA qui permet une grande modularité et un traitement de calcul/vidéo très efficace et optimisé selon les besoins, mais qui nécessite une grande expertise et qui est plus volumineux qu'un circuit classique.

La puissance et l'efficacité des lasers peut également être revus et un programme modulant l'intensité du laser pour créer un rapport signal sur bruit favorable à la détection du spot. (Ex . : PWM)Pour prévenir tout risque d'aveuglement, il est également possible de prévoir une LED au lieu d'un laser, qui sera moins dangereuse pour la santé.

3.3. Logiciel

En application logicielle, une migration de l'ensemble du RHS du Python vers C++ est recommandée pour une optimisation des calculs. Python n'a servi que d'outil d'évaluation et ne peut clairement pas être utilisé dans un tel produit.

D'autres traitements d'image permettraient peut-être d'arriver au résultat escompté en moins d'étapes et donc permettraient un gain en précision et en temps. Pour cela, une prospection correcte de l'ensemble des méthodes connues pour la recherche d'un spot rouge dans une image en conditions extérieures variables est nécessaire. Notamment par temps de pluie.

Il existe plusieurs méthodes de recherche d'un élément dans une image. Certaines méthodes sont évoluées comme la recherche par « blobs » ou par détection de contours… Nous n'avons pas sélectionné ou même approfondi les recherches concernant ces méthodes en raison de contraintes de temps et de nécessité de formation pour comparer l'ensemble des méthodes disponibles.

Nous avons pour l'instant choisi d'utiliser la recherche par « seuillage » qui consiste à transformer l'image source de manière à « seuiller » l'image (i.e. affecter une valeur par défaut pour tout pixel n'ayant pas une valeur dans l'intervalle spécifié) pour en récupérer l'information désirée.

Le calcul sur l'image seuillée est réalisé par la moyenne est pixels non nuls. Il serait plus judicieux de réaliser la somme pondérée de ces pixels, permettant une précision plus importante du point.

L'instabilité du spot est principalement due à l'apparition ou à la disparition d'éléments qui ne doivent pas ou qui doivent être seuillés. Typiquement, les éléments de couleur orange/jaune sont détectés par l'algorithme qui les fait apparaître dans l'image seuillée. De manière inverse, le centre du spot n'est pas toujours visible car trop lumineux, donc blanc et donc ne présentant pas de degré de rouge.

Une première solution a été apportée en utilisant la transformée de Fourier 2D de l'image et en réalisant un filtre passe-bande, mais celui-ci s'est avéré peu concluant car selon le cas considéré, il fallait utiliser différents filtres et il est coûteux en temps de calcul.

Une autre solution serait de mettre en place une régulation sur la luminosité ambiante, pour déterminer des valeurs de saturation et de contraste dynamiques. Bien que ce soit réalisable, la multiplicité des échantillons ainsi qu'un critère de qualité de l'image sont nécessaires pour déterminer la loi adaptée. L'idéal serait donc de disposer d'un environnement à luminosité contrôlée (avec plus de deux niveaux de luminosité autres qu'éteint et allumé). C'est une solution envisageable, mais dans un futur plus éloigné.

Nous pensons que ce problème pourrait être au final résolu avec un capteur CMOS de meilleur qualité. Nous entendons par « meilleure qualité » un capteur avec une plage dynamique (HDR) plus importante. Et fait donc ressortir la composante rouge. La caméra de la maquette a en effet des couleurs pâles.

Conclusion

Dans ce chapitre, nous avons précisé la faisabilité théorique et pratique du projet. Nous avons ainsi mis en évidence l'influence des paramètres géométriques sur les deux algorithmes du RHS Caméra : recherche de point et calcul. Nous les avons évalués d'abord séparément puis de manière intégrée. Nous pouvons dissocier deux sources d'erreur : l'erreur de calcul due à une limitation de la plage de validité de la mesure en fonction des paramètres géométriques et l'erreur d'approximation dans la recherche du spot. Nous avons alors deux axes d'amélioration de la précision du RHS Caméra : d'une part les paramètres géométriques qui permettent d'établir la plage de mesure et la précision théorique maximale et d'autre part l'algorithme de recherche de spot, permettant d'améliorer la stabilité du signal brut (position du spot) et donc la stabilité de la mesure. A noter que la stabilité de l'algorithme n'a pas pu être évalué sur des images bruitées simulant une forte luminosité.

Nous avons évalué la vitesse de calcul pratique sur un support embarquable (Raspberry Pi). Nous avons déterminé que la fréquence maximale de traitement était d'environ 20Hz pour un cahier des charges stipulant 100Hz. La limitation hardware actuelle pousse ainsi le projet à être mis en stand-by en attendant l'évolution des technologies, tout en effectuant une veille technologique.

Bibliographie

[1] Chefi, Ahmed. « Conception d'un micro capteur d'image CMOS à faible consommation d'énergie pour les réseaux de capteurs sans fil », s. d., 140.

[2] Fadhil, Dr Ahmed Freidoon, et Maryam Mohammed Fatih. « Laser Spot Detection and Tracking », 2016, 10. « (PDF) Vision Based Distance Measurement System Using Single Laser Pointer Design for Underwater Vehicle ». *ResearchGate*. Consulté le 9 octobre 2021. https://www.researchgate.net/publication/297799780_Vision_based_distance_measurement_system_using_single_laser_pointer_design_for_underwater_vehicle.

[3] Pereira, Rui, Marco Couto, Francisco Ribeiro, Rui Rua, Jácome Cunha, João Paulo Fernandes, et João Saraiva. « Energy Efficiency across Programming Languages: How Do Energy, Time, and Memory Relate? » In *Proceedings of the 10th ACM SIGPLAN International Conference on Software Language Engineering*, 256-67. Vancouver BC Canada: ACM, 2017. https://doi.org/10.1145/3136014.3136031.

[4] Rebert, Martin, David Monnin, Sophie Kohler, et Christophe Cudel. « Comparaison de décompositions de la matrice homographique et essentielle pour l'estimation du mouvement de caméra », s. d., 9.

[5] Shah, Karnik. « How to Implement an Image Sensor Interface Using EZ-USB FX3 in a USB Video Class (UVC) Framework », n° 001 (s. d.): 74.