Projet : Mise en correspondance stéréo

Xuan-Vinh HO et Nhu-Hoang HO

May 8, 2022

Contents

1. Introduction

En réalité, la raison pour laquelle nous pouvons percevoir la profondeur est grâce à nos yeux. Lorsque nous regardons des objets proches avec un seul œil, nous voyons une différence entre les deux perspectives. Mais quand nous regardons quelque chose de loin, nous ne verrons pas de différence. Ces différences sont automatiquement traitées dans notre cerveau et nous pouvons en percevoir la profondeur. Les animaux qui ont les yeux alignés à l'extrême droite et à l'extrême gauche ne peuvent pas percevoir la profondeur car ils n'ont pas de perspectives communes, à la place, ils ont une perspective grand angle. Certains animaux secouent la tête ou courent vite pour percevoir la profondeur, c'est ce qu'on appelle la structure du mouvement.

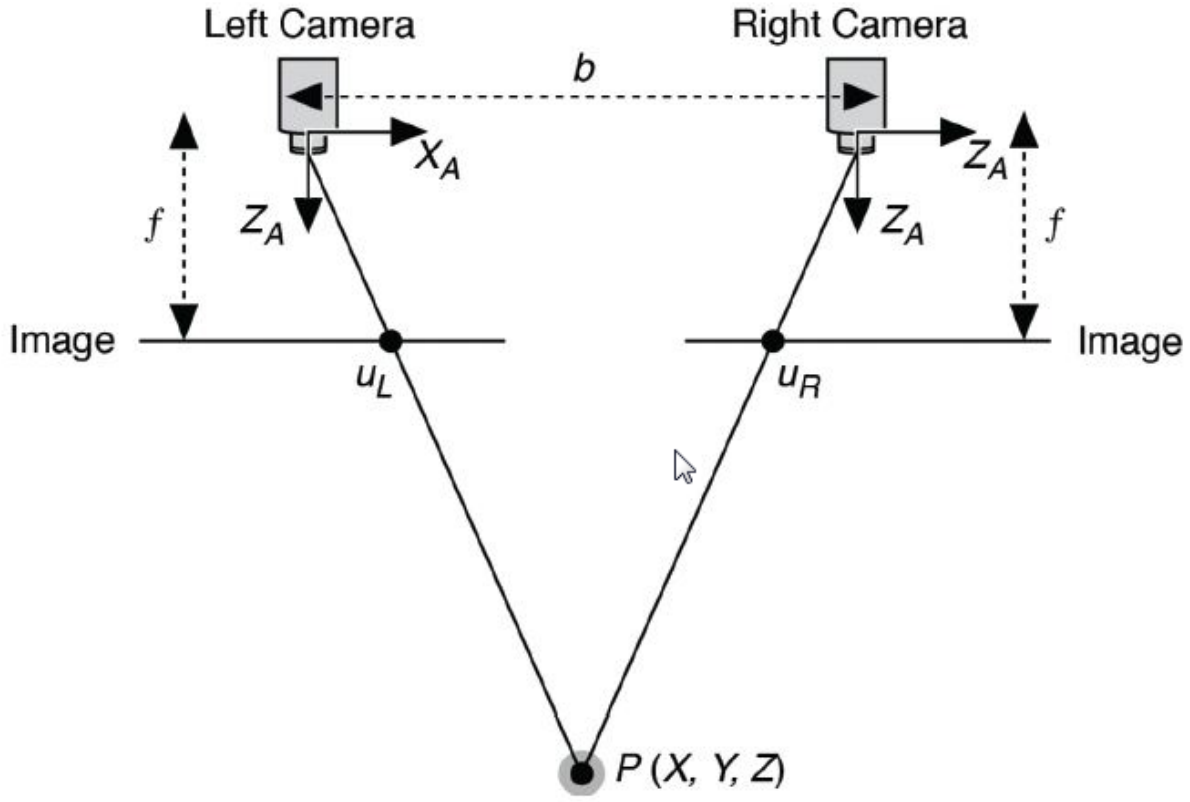


Figure 1: Fonction de pondération équivalente

Dans la stéréovision, si deux caméras sont alignées verticalement, l'objet observé sera dans les mêmes coordonnées verticalement (même colonne dans les images), nous ne pouvons donc nous concentrer que sur les coordonnées x pour calculer la profondeur puisque les objets proches auront une différence plus élevée dans l'axe x . Mais pour y parvenir, nous devons calibrer les caméras pour corriger les distorsions de l'objectif. Après l'étalonnage, nous devons rectifier le système. La rectification est essentiellement un calibrage entre deux caméras. Si nous calibrons et rectifions bien nos caméras stéréo, deux objets seront sur le même axe y et le point observé $P(x, y)$ se trouvera dans la même rangée de l'image, $P1(x_1, y)$ pour la première caméra et $P2(x_2, y)$ pour la deuxième caméra. À partir de là, c'est la seule différence entre les calculs de pixels et de profondeur.

Donc, l'objectif de ce projet est d'implémenter un algorithme de mise en correspondance stéréo pour trouver la carte de disparité. L'algorithme sera testé sur les images du jeu de données qui est simple à prendre en main sans utiliser l'agorithme d'apprentissage automatique. Les images sont déjà rectifiées de sorte à ce que les lignes de même hauteur

soient les lignes épipolaires conjuguées. Nous utiliserons principalement les images cones et teddy de ce dataset, de résolution 450 × 375 et au format png.

Chaque paire d'images à traiter porte les noms `im2` et `im6`, respectivement pour les vues gauches et droites. Les disparités vérité correspondants à ces images comme référence sont données respectivement dans les fichiers disp2 et disp6. Elles sont codées en valeur absolue, c'est-à-dire que la vérité est toujours donnée en disparités positives, et par des entiers prenant leur valeur dans l'intervalle [0 ; 255].

Pour la sélection de l'algorithme, nous avons décidé de choisir l'algorithme block-matching grâce à sa simplicité.

2. L'algorithme de block-matching

2.1. Motivation

Nous avons mentionné que si nous regardons un objet proche avec l'œil gauche fermé et vice versa, l'emplacement change en fonction de la perspective de l'œil ouvert. Et la différence est de plus en plus importante lorsque l'objet est plus éloigné. Cela signifie que si la valeur de disparité est plus petite, l'objet est plus proche. Le processus de rectification peut être fait pour chaque point, mais il n'est pas efficace car:

- C'est un processus lent.
- Si la rectification n'est pas assez bonne, il n'est pas idéal de travailler avec chaque point.

Nous devons réduire l'erreur et le traitement ponctuel n'aidera pas dans ce cas. C'est la raison pour laquelle nous utilisons l'algorithme de block-matching.

2.2 Explication et critères de correspondance

L'algorithme de block-matching est un moyen de localiser des macroblocs correspondants dans une séquence de trames vidéo numériques à des fins d'estimation de mouvement. L'objectif principal des algorithmes d'estimation de mouvement basés sur des blocs est de déterminer l'amplitude et la direction du mouvement entre un macrobloc de la trame actuelle et le bloc candidat le mieux adapté de la trame de référence. Le critère d'appariement le plus couramment utilisé qui mesure la distorsion d'erreur entre le macrobloc de renommée actuelle et les blocs candidats dans la trame de référence est la somme des différences absolues (SAD). La SAD entre un macrobloc de taille MN avec un coin supérieur gauche à (p, q) et un bloc candidat de taille MN avec un coin supérieur gauche à $(p + x, q + y)$ est défini comme suit équation:

$$SAD(x, y) = \cdot \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |I\,(p + i, q + j) - R\,(p + x + i, q + y + j)|$$

où $I(.,.)$ et $R(.,.)$ désignent les valeurs de pixels de l'image actuelle et de l'image de référence. Les coordonnées des vecteurs de mouvement x et y sont définies dans l'équation suivante:

$$(x, y) = \arg \min_{(\widehat{x},\widehat{y})\in R} SAD\,(\widehat{x}, \widehat{y})$$

$$R = \{(\widehat{x}, \widehat{y})\,| - s \leq \widehat{x}, \widehat{y} \leq d\}$$

R et d représentent la plage de recherche, Le critère SAD implique $(M \times N) - 1$ opérations d'addition, $M \times N$ opérations absolues et $M \times N$ opérations de soustraction, c'est-à-dire qu'un calcul SAD nécessite environ $3 \times M \times N$ opérations.

Outre SAD, nous utilisons également la somme des différences au carré (SSD), avec la seule différence est d'utiliser l'opérateur carré au lieu de l'opérateur absolu, ce qui nous donne une meilleure précision:

$$SSD(x, y) = \cdot \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} \left(I\left(p+i, q+j\right) - R\left(p+x+i, q+y+j\right)\right)^2$$

Au cours du processus de l'algorithme, nous appliquons la méthode **sub-pixel estimation** pour calculer la disparité avec une meilleure précision et la méthode **image pyramid** pour réduire les dimensions de l'image. Ces 2 méthodes seront expliquées plus loin.

2.3 Choix de hyper-parametres

Les deux hyper-paramètres utilisés dans cet algorithme sont la taille de bloc (`block_range`) et la valeur de disparité maximum (`disparity_range`). Afin de déterminer les meilleurs valeurs de ces paramètres, nous avons effectué une recherche exhaustive dans un sous-ensemble de l'espace des hyper-paramètres spécifié manuellement, 7 à 15 pour la taille de bloc et 30 à 70 pour la valeur de disparité maximum dans nos cas, et de comparer les résultats pour ces valeurs. De part cette étude, on peut considerer que les parametres optimum sont des blocs de taille 11 pour une disparité max de 64. Cependant, nous utiliserons la pyramide d'image qui est une méthode pour réduire l'image 2 fois, donc la taille de bloc optimum est 7 et la valeur de disparité maximum optimum est 32.

2.4 Résultats obtenus

Nous avons exécuté les algorithmes mentionnés ci-dessus sur les images cones et teddy et utilisé la vérité terrain pour analyser les résultats.

Résultats sur les images **cones**:

	Qualité		Rapidité	ratio qualité/rapidité	
précision	> 1px	> 2px		> 1px	> 2px
SAD	53.45%	63.92%	24.14s	2.21	2.65
SSD	56.11%	65.31%	28.77s	1.95	2.27

Résultats sur les images **teddy**:

	Qualité		Rapidité	ratio qualité/rapidité	
précision	> 1px	> 2px		> 1px	> 2px
SAD	52.91%	62.09%	27.77s	1.90	2.24
SSD	53.79%	62.68%	29.20s	1.84	2.15

Les figures suivantes montrent les cartes de profondeur calculées par 2 méthodes de block matching pour les images **cones** et **teddy**:

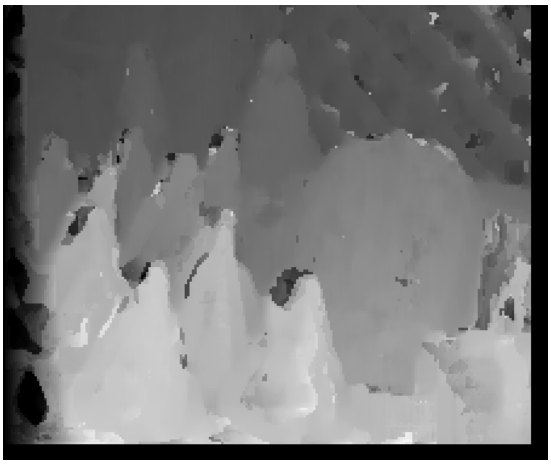
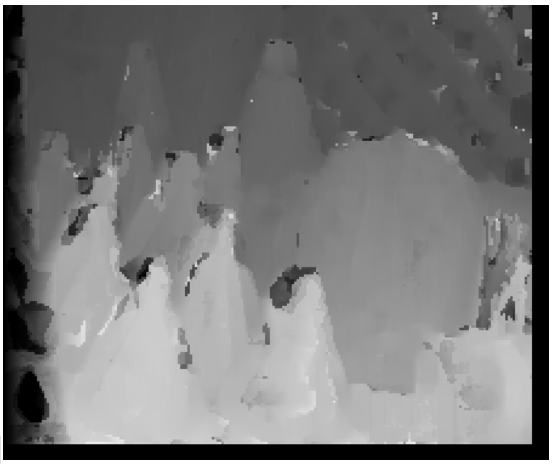
[a]  [b] 

Figure 2: Carte des disparité des images cones avec (a) SAD (b) SSD

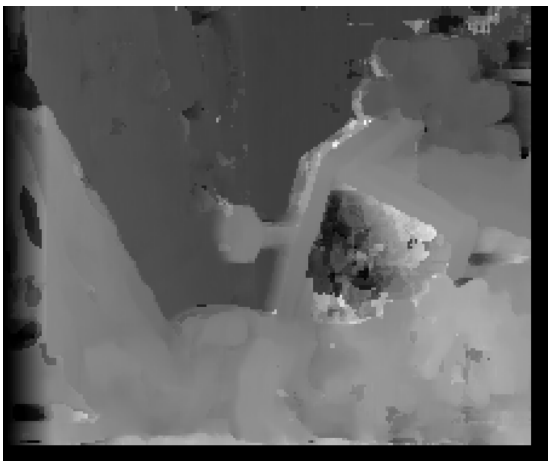
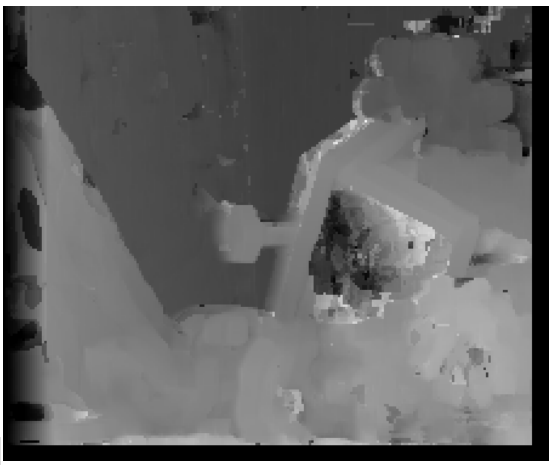
[a]  [b] 

Figure 3: Carte des disparité des images cones teddy avec (a) SAD (b) SSD

3. Méthode d'optimisation

3.1. Sub-pixel estimation

Il s'agit d'une méthode pour améliorer la précision du calcul de la disparité. Au lieu d'obtenir la valeur de disparité calculée directement à partir de la boucle et de l'algorithme, nous utilisons une étape supplémentaire pour obtenir un résultat plus précis. Sur la base de deux valeurs de disparité voisines et de la critère correspondante, nous avons le système d'équation suivant:

$$\begin{aligned} ax_1^2 + bx_1 + c &= y_1 \\ ax_2^2 + bx_2 + c &= y_2 \\ ax_3^2 + bx_3 + c &= y_3 \end{aligned}$$

Avec x_1, x_2, x_3 correspondant aux valeurs de disparité et y_1, y_2, y_3 correspondant aux critères. Sur la base de ce système d'équations, nous pouvons calculer les valeurs des coefficients a, b, c et obtenir une fonction quadratique comme suit:

$$f(x) = ax^2 + bx + c$$

Enfin, calculez les valeurs $disparity_{min}$ et $critère_{min}$ au point minimum de la fonction. La méthode de calcul initiale ne donne que des valeurs entières, ce qui rend la carte des disparités moins lisse que prévu. De plus, il existe de nombreux cas où la critère que nous obtenons n'est pas exacte bien qu'il s'agisse de la valeur minimale. Par conséquent, cette méthode nous aide à obtenir une meilleure valeur de disparité et une carte de disparité plus lisse tout en gardant un temps d'exécution raisonnable car la complexité temporelle de la Sub-pixel Estimation est O(1).

Ci-dessous se trouvent trois points, marqués par des x rouges, qui sont également espacés dans la direction x et se trouvent tous sur une parabole.

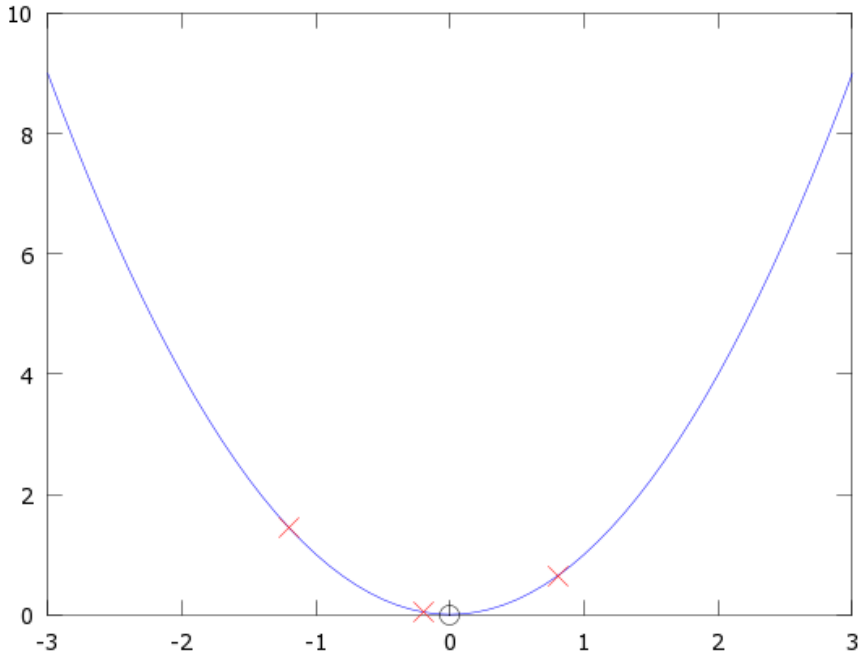


Figure 4: 3 points (rouge) sur une parabole et le point optimum (noir)

3.2. Pyramid (image processing)

Selon Wikipédia, la représentation pyramidale ou pyramidale est un type de représentation de signal à plusieurs échelles développé par les communautés de la vision par ordinateur, du traitement d'image et du traitement du signal, dans lequel un signal ou une image est soumis à un lissage et à un sous-échantillonnage répétés. La représentation pyramidale est un prédécesseur de la représentation à l'échelle de l'espace et de l'analyse multirésolution. Normalement, on travaillait avec une image de taille constante. Mais à certaines occasions, nous devons travailler avec (les mêmes) images dans différentes résolutions. Dans ce cas, nous devrons créer un ensemble de la même image avec différentes résolutions et rechercher des objets dans chacun d'eux. Ces ensembles d'images avec différentes résolutions sont appelés pyramides d'images (car lorsqu'elles sont conservées dans une pile avec l'image de résolution la plus élevée en bas et l'image de résolution la plus basse en haut, cela ressemble à une pyramide).
Il existe deux types de pyramides d'images: Pyramide Gaussienne et Pyramides Laplaciennes. Dans notre projet, nous nous concentrerons sur le premier type.

3.2.1 Gaussian Pyramid

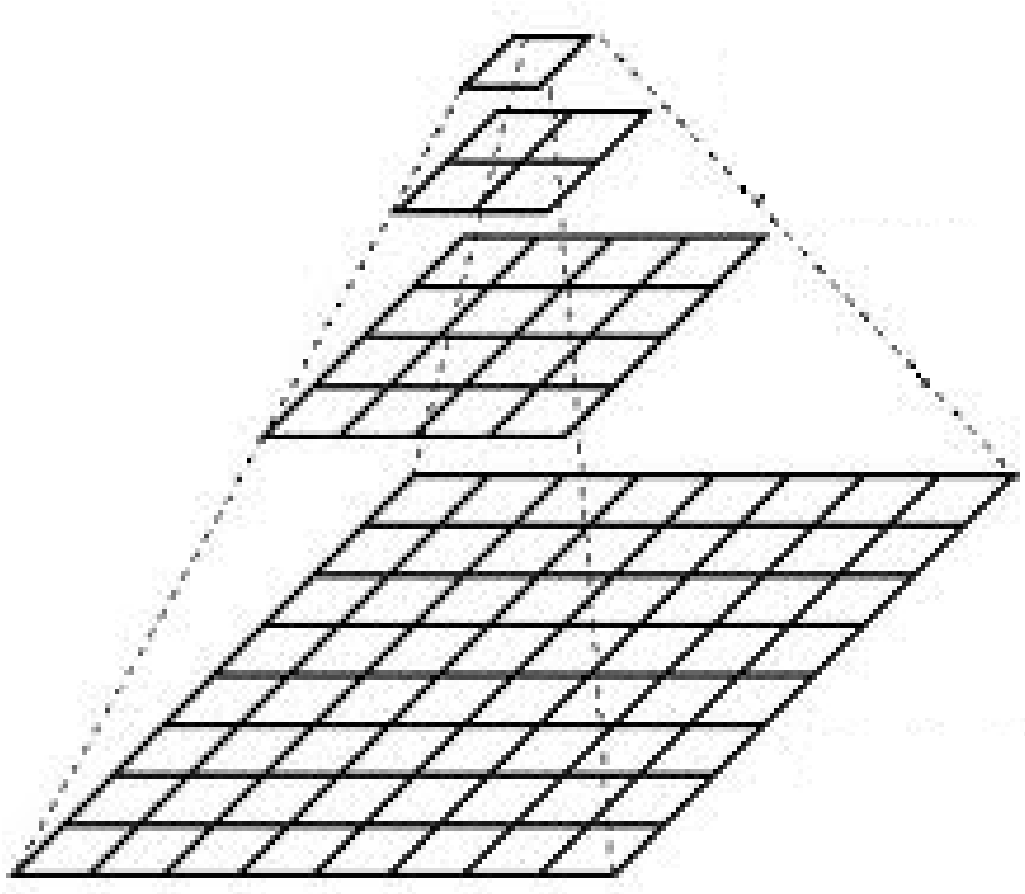


Figure 5: Gaussian Pyramid

L'objectif de cette méthode est de développer une représentation pour décomposer des images en informations à plusieurs échelles, extraire des caractéristiques ou des structures d'intérêt et atténuer le bruit. Comme l'explication précédente, à partir de la source de l'image, nous allons créer des images beaucoup plus petites. La relation de ces images est

décrite par la fonction suivante :

$$\bar{p}_{l+1} = p_l * g_\sigma$$
$$p_{l+1} = \bar{p}_l(x2, 2y)$$

Avec $p_l(x2, 2y)$ et $p_{l+1}(x2, 2y)$ sont deux niveaux de la pyramide et g_σ est un filtre gaussien à 2 dimensions. Dans le cadre continu, une pyramide équivariante d'échelle peut être construite avec n'importe quel filtre; cependant, à moins que l'image ne soit limitée en bande et que le filtre soit choisi avec soin, le coût de calcul peut être prohibitif. Le filtre gaussien est particulièrement adapté à cette tâche car il est séparable et sa convolution avec lui-même s'apparente à une mise à l'échelle. Dans la mise en œuvre de la pyramide gaussienne, le filtre gaussien est approximé par une fonction de pondération équivalente normalisée et symétrique. Une fonction de pondération équivalente est choisie pour assurer une contribution égale, c'est-à-dire que tous les nœuds d'une couche donnée contribuent de manière égale à la construction de la couche suivante. Un filtre de longueur 5 est une fonction de pondération équivalente s'il a la structure suivante:

$$[c \quad b \quad a \quad b \quad c]$$

Avec $b = 1/4$ et $c = b - a/2$.

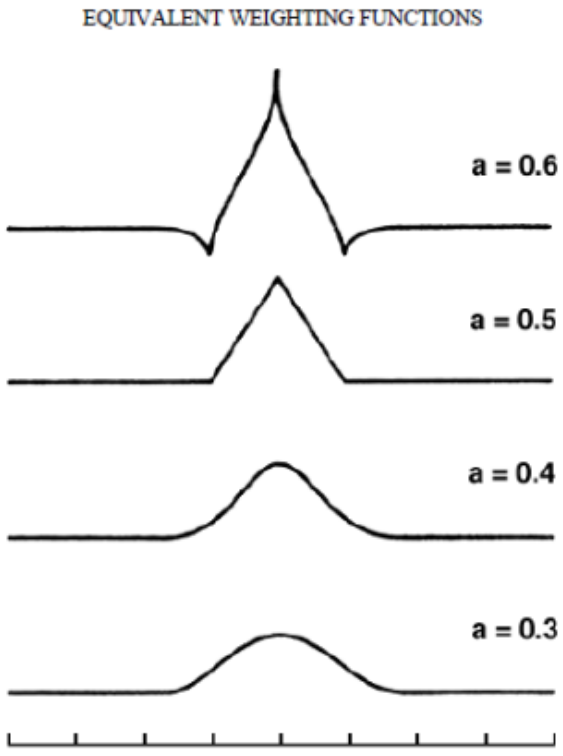


Figure 6: Fonction de pondération équivalente

D'après cette figure, les plus gaussiennes correspondent à quand a = 0,4.

3.2.2. Laplacian pyramid

Une pyramide laplacienne est très similaire à une pyramide gaussienne mais enregistre la différence d'image des versions floues entre chaque niveau. Seul le plus petit niveau n'est

pas une image de différence pour permettre la reconstruction de l'image à haute résolution en utilisant les images de différence sur des niveaux supérieurs. Cette technique peut être utilisée dans la compression d'image. Puisque le laplacien est un filtre passe-haut, à chaque niveau de cette pyramide, nous obtiendrons une image de bord en sortie. Le laplacien peut être approximé en utilisant la différence de gaussien. Donc, ici, nous allons profiter de ce fait et obtenir la pyramide laplacienne en soustrayant les niveaux de la pyramide gaussienne. Ainsi, le laplacien d'un niveau est obtenu en soustrayant ce niveau dans la pyramide gaussienne et en développant la version de son niveau supérieur dans la pyramide gaussienne. Ceci est illustré dans la figure ci-dessous.

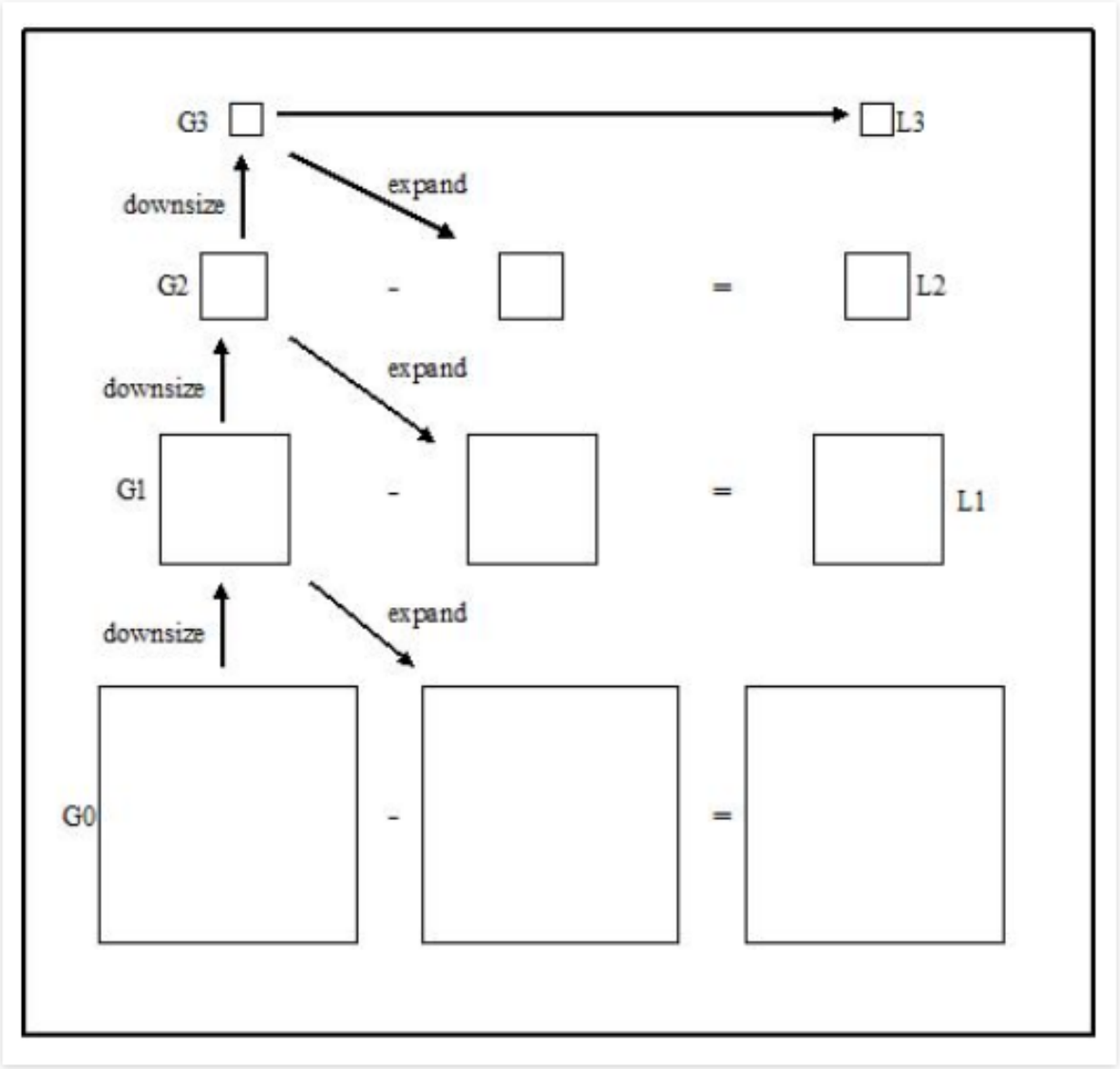


Figure 7: Figure relationnelle

3.3 Augmenter la taille de l'image

Selon la section précédente, nous pouvons voir la relation entre la pyramide gaussienne, la pyramide laplacienne et l'image agrandie. Il est clair que nous pouvons déduire l'image étendue à partir des deux images générées à travers les matrices pyramidales. Cependant, le problème se pose que si nous sommes dans la position de n'importe quelle image et que nous voulons doubler sa taille, nous ne pouvons pas avoir deux images créées par la matrice pyramidale et ne pouvons donc pas être redimensionnées par cette méthode. Par

conséquent, une méthode est nécessaire pour augmenter directement la taille de l'image. Dans notre implémentation, nous avons utilisé cette méthode pour récupérer la carte de disparité de la même taille que l'image d'entrée.

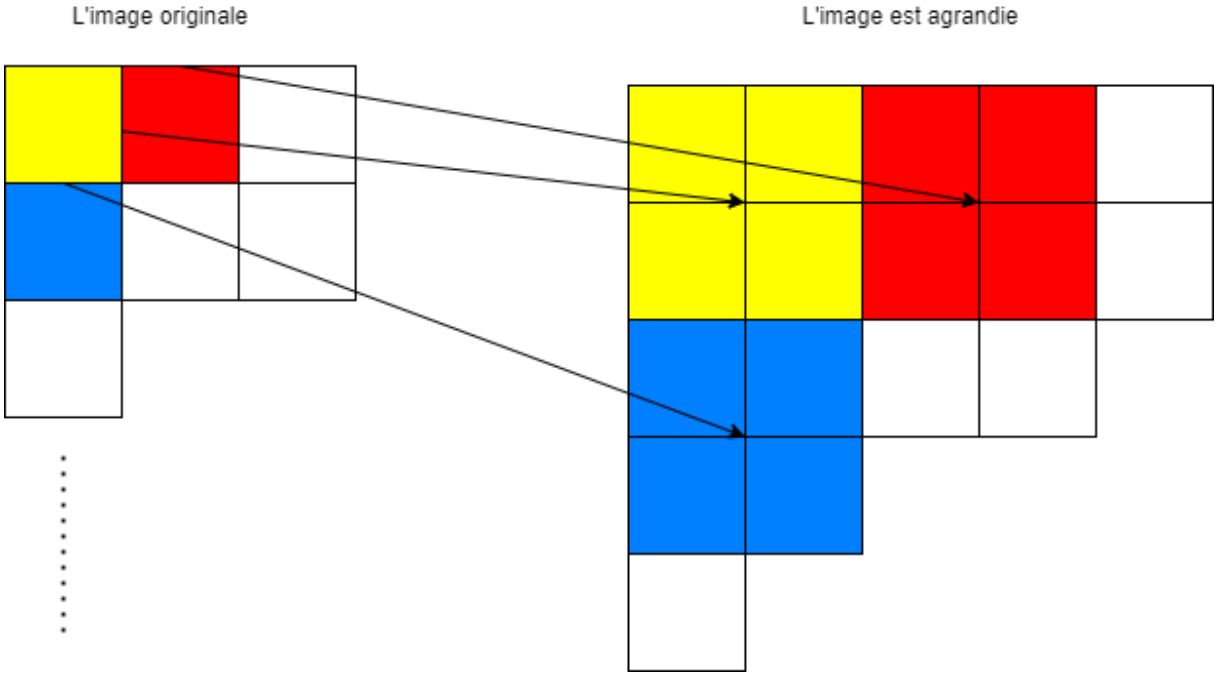


Figure 8: Agrandir l'image