Titre: Title:	Méthodes d'apprentissage profond pour l'estimation de disparités d'une paire d'images multispectrale
Auteur: Author:	Philippe Duplessis-Guindon
Date:	2022
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Duplessis-Guindon, P. (2022). Méthodes d'apprentissage profond pour l'estimation de disparités d'une paire d'images multispectrale [Mémoire de maîtrise, Polytechnique Montréal]. PolyPublie. https://publications.polymtl.ca/10477/

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/10477/
Directeurs de recherche: Advisors:	Guillaume-Alexandre Bilodeau
Programme: Program:	Génie informatique

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Méthodes d'apprentissage profond pour l'estimation de disparités d'une paire d'images multispectrale

PHILIPPE DUPLESSIS-GUINDON
Département de génie informatique et génie logiciel

Mémoire présenté en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
Génie informatique

Août 2022

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Ce mémoire intitulé :

Méthodes d'apprentissage profond pour l'estimation de disparités d'une paire d'images multispectrale

présenté par **Philippe DUPLESSIS-GUINDON**
en vue de l'obtention du diplôme de *Maîtrise ès sciences appliquées*
a été dûment accepté par le jury d'examen constitué de :

Farida CHERIET, présidente
Guillaume-Alexandre BILODEAU, membre et directeur de recherche
Quentin CAPPART, membre

DÉDICACE

À tous mes amis, mes proches et mes collègues
de m'avoir supporté dans ces temps difficiles ...

REMERCIEMENTS

J'aimerais remercier mon professeur, Guillaume-Alexandre Bilodeau, pour son excellent support tout au long de ma maîtrise. J'apprécie le temps qu'il a mis pour le suivi du projet ainsi que pour l'organisation de l'ensemble des activités du laboratoire. J'aimerais aussi remercier mes collègues qui étaient là lorsque j'avais des questions, et aussi pour m'avoir accueilli au sein du laboratoire. Je remercie donc mes collègues Medhi, Xi, Pankaj, Hughes, Jacob, Rihab, Soufiane, Qingwu, Katia, Andrey, Noreeen and Maude. Pour finir, j'aimerais remercier le Conseil de Recherches en Sciences Naturelles et en Génie du Canada (CRSNG) pour le support financier du projet.

RÉSUMÉ

Ce mémoire présente plusieurs nouvelles méthodes pour estimer la disparité des silhouettes humaines dans une scène à l'aide d'une paire d'images stéréoscopiques du spectre infrarouge thermique et du spectre couleur. Le défi ce problème est de faire correspondre des sous-régions d'images qui ne partagent pas la même information. En effet, le spectre infrarouge thermique et RGB ont quelques informations en commun, telles que les formes générales des silhouettes, mais la principale source d'information du spectre RGB est manquante dans les images infrarouges thermiques (LWIR), soit la couleur. Des méthodes ont été proposées dans des précédents travaux pour résoudre ce problème, mais nous voulons appliquer de connaissances plus récentes pour améliorer les résultats. Trois grandes idées forment notre méthodologie. La première idée a été d'ajouter les masques de segmentation aux sous-régions d'entrée de notre réseau. Le réseau va donc connaître le contexte de la sous-région pour la comparer à la sous-région de l'autre spectre. Le réseau va donc savoir si la sous-région appartient à une instance ou non, et verra si une frontière est présente. La deuxième idée est de modifier l'architecture de l'extracteur de caractéristiques afin d'opter pour une solution plus moderne. Nous avons donc remplacer les deux RNCs du réseau pseudo-siamois par HRNet. HRNet est un extracteur de caractéristique gardant la même résolution tout au long de l'extraction des caractéristiques. Ceci permet donc d'avoir des cartes de caractéristiques avec le plus grand nombre de détails possible. Dans cette même idée, nous avons dû adapter le réseau HRNet étant donné que nous l'utilisons sur des entrées de 36×36, et que l'architecture est initialement faite pour des images de plus grande taille. Nous avons ensuite modifié la sortie de l'architecture HRNet, pour encore une fois garder le plus de caractéristiques possibles. La dernière idée apportée dans ce mémoire a été de combiner les deux premières idées ensemble pour introduire les masques dans le réseau HRNet. Les deux premières idées prises séparément ont donné de très bons résultats. Par contre, la combinaison des deux a donné des résultats peu concluants, mais il était bon de faire l'étude pour prouver l'efficacité des masques selon l'extracteur de caractéristiques. Pour l'ensemble des méthodes présentées, la tête du réseau reste inchangée. La tête qui est présentée est constituée de deux branches avec un fonctionnement identique. Il y a une branche de concaténation et une branche de corrélation. Le principe de ces branches est le suivant. L'opération de fusion est faite sur les deux vecteurs de caractéristiques sortant des extracteurs. Ensuite le vecteur résultant de cette opération de fusion est passé dans un réseau pleinement connecté qui déterminera l'estimation de disparité pour la branche donnée. L'estimation finale sera donnée par la moyenne de ces deux estimations de disparité. Avec ces méthodes, nous avons été capables de générer des résultats qui dépassent l'état de l'art.

ABSTRACT

This work will presents several methods to estimate the disparity between human silhouettes using RGB and infrared images. The main goal of this master thesis is to match two stereo patches from two different spectrum. Between a RGB and a thermal image, there is not much shared information, except from the shapes of the humans in the scene. Methods were proposed for this problem, but we want to improve the solution by bringing new ideas from recent work.

This work presents three proposed ideas that we have implemented. The first idea consists of concatenating segmentation masks to the input patches of the network. The network can know the context of the patch using the masks. The goal of the patch is to identify the boundary in the image. The second idea is to modify the architecture to change the feature extractor for a modern one. We will replace both CNNs in the pseudo-siamese network with an HRNet backbone. We chose HRNet because it is a feature extractor that keeps a high resolution feature map across all the network. The output has a lot of information, and is the same size as the original input patch. Because the HRNet backbone was not designed for 36×36 images, we had to adapt it to fit this input size. We removed the smallest sub-sampling which was too small for such a small input patch. When HRNet was adapted to our network, we then modified the output to see if the result could be improved by concatenating the last high resolution layer of every stage. The last idea was to take this modified pseudo-siamese HRNet network and mix it with the first idea of introducing masks in the network.

We obtained very good results by applying separately both ideas. However, the combination of both did not give convincing results, but it was a good study to see the efficiency of the masks with a high resolution feature extractor. For every method presented, the head of the network stayed the same. The head of the network was made of two branches. One concatenation branch and one correlation branch. A branch takes the output of the fusion operation and passed it to a fully connected layer to output the estimated disparity. The final disparity is given by the mean of both branches.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES SIGLES ET ABRÉVIATIONS

RGB	Red Green Blue - Pour les images en couleurs
LWIR	Long-wave infrared - Pour représenter les images termiques
NIR	Near-infrared
RNC	Réseaux de neurones convolutionnels
CNN	Convolutional neural network - Appellation anglaise de RNC
MHSA	Bloc d'auto attention multi-tête

CHAPITRE 1 INTRODUCTION

La stéréoscopie est utilisée dans plusieurs domaines. Le phénomène stéréoscopique le plus proche de nous est la vision. Nos yeux sont l'équivalent de deux caméras stéréoscopiques, qui nous permettent de voir et de nous donner une estimation de la profondeur des objets dans notre environnement. En effet, tout comme les yeux, les caméras stéréoscopiques permettent d'avoir deux images de la même scène avec deux prises de vue différentes. Souvent les caméras sont alignées sur l'axe des y et ont un décalage donné sur l'axe des x. Ce décalage est important car c'est à l'aide de cette valeur qu'il sera possible de déterminer la distance entre l'objet identifié et les caméras. Les concepts de stéréoscopie sont également utilisés dans le domaine cinématographique pour faire des films et des téléviseurs avec images tridimensionnelles. Une application plus proche du contenu du présent mémoire est un mécanisme d'estimation de distance entre les objets et les caméras dans un véhicule autonome.

Dans le cas des véhicules autonomes, les méthodes utilisées sont dans la même lignée et ont le même but que ce travail. Nous utilisons des caméras stéréoscopiques pour estimer la disparité entre les objets dans la scène. La disparité permet ensuite d'estimer la profondeur, une plus grande disparité correspond en général à un objet plus près des caméras. D'autres moyens sont utilisés pour calculer la distance des objets dans une scène. L'utilisation d'un LIDAR est plus précise que d'utiliser des caméras, mais le LIDAR reste une alternative beaucoup plus dispendieuse. L'option des caméras reste donc plus pertinente pour une raison financière, mais aussi pour des raisons de contexte dans la scène. En effet, les LIDARs ne permettent pas d'interpréter les couleurs, ou les écritures par exemple, ce qui est essentiel pour que le véhicule prenne ses décisions. Ceci rend donc les caméras indispensables dans l'application des véhicules autonomes.

Les méthodes que nous proposons servent à estimer la disparité au pixel près. La plupart des travaux effectués en stéréoscopie sont faits à l'aide d'images de provenance du spectre de la lumière visible, soit les images RGB. Le défi avec ce type d'images est de trouver les correspondances entre les objets des deux images. Un inventaire des difficultés relié à la stéréoscopie sera détaillé dans la section 1.2. Le sujet du présent mémoire est de faire de la stéréoscopie multispectrale. Le terme multispectral désigne l'utilisation de deux images de spectres lumineux différents afin d'estimer la disparité. Dans le cas de la stéréoscopie classique, plusieurs caractéristiques sont partagées entre les deux images, mais ce n'est pas le cas en stéréoscopie multispectrale. En effet, étant donné que le spectre de lumière change entre une image et l'autre de la paire stéréo, les images sont très différentes. Dans notre

cas, les spectres étudiés sont le RGB et le LWIR, soit les images infrarouges thermiques. Les caractéristiques similaires entre les deux spectres sont presque uniquement les formes, étant donné qu'une image est en tons de gris et correspond au rayonnement thermique (l'image infrarouge thermique) et l'autre est en couleur et correspond à la réflexion de la lumière sur une surface (l'image RGB). Encore une fois, les détails des problématiques seront expliqués en détail à la section 1.2.

L'utilisation des caméras thermiques en stéréoscopie permet d'améliorer la précision lorsque le spectre de couleur ne donne pas beaucoup d'information sur la scène. Par exemple, dans une scène où la visibilité est faible, la présence de caméra thermique permet de mieux identifier les silhouettes humaines, et ainsi d'améliorer la précision de l'estimation des disparités.

1.1 Définitions et concepts de base

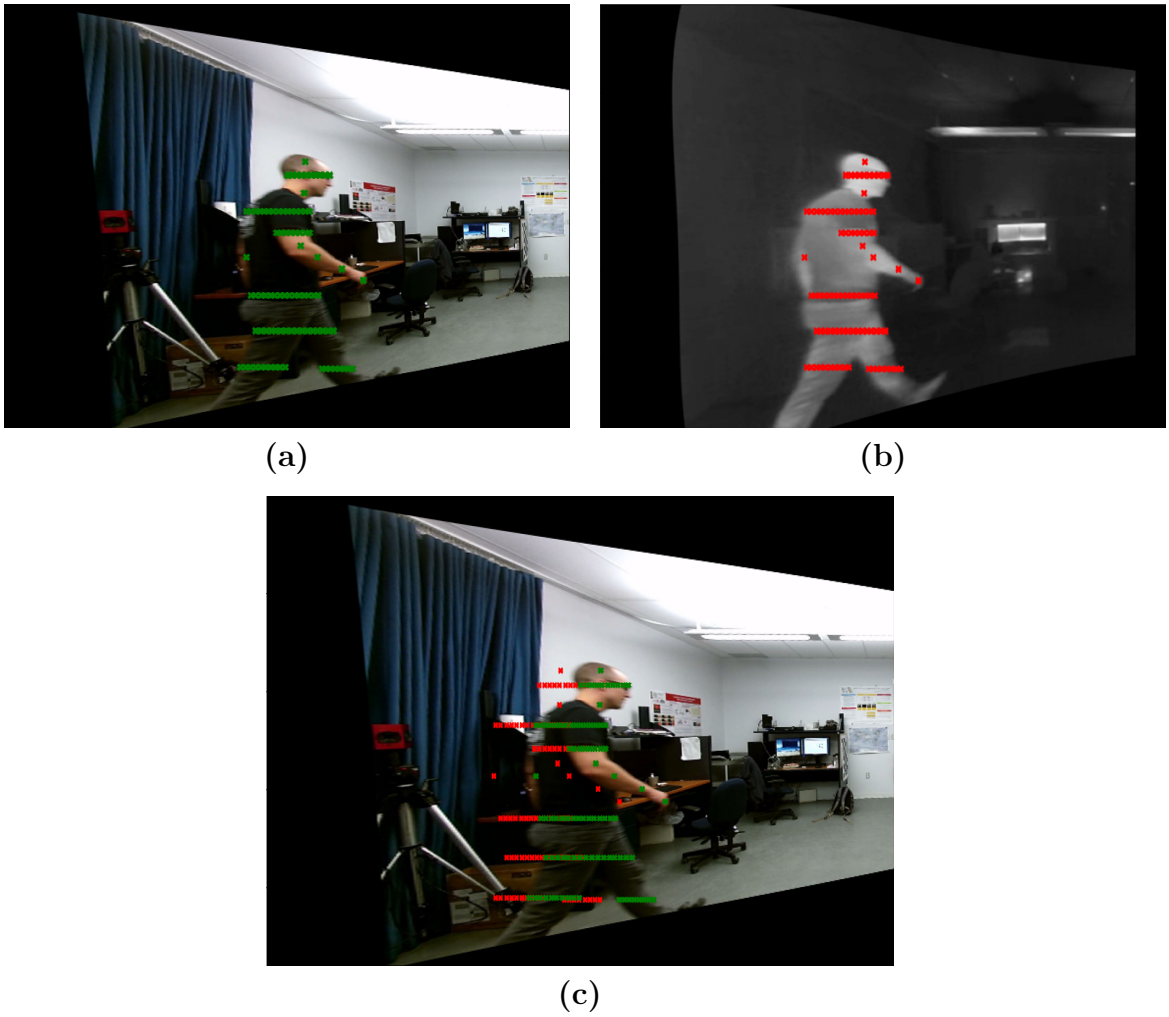
Comme mentionné plus haut, la stéréoscopie est un phénomène que nos yeux utilisent au quotidien. En effet, le fait d'avoir deux yeux nous permet d'avoir une perception plus complète de notre environnement. Si un oeil est fermé, nous pouvons déduire les profondeurs des objets, mais il reste difficile d'avoir une idée précise de notre environnement 3D. Tel est le cas avec l'estimation de distance avec des images. Il y a moyen d'avoir une notion de profondeur en se servant d'une seule caméra, mais pour estimer la vraie profondeur des objets dans une scène, il faut se servir de deux caméras côte à côte, appelée une paire de caméras stéréoscopique.

Le principe de la stéréoscopie est de prendre deux images d'une scène à un moment t afin de pouvoir calculer la distance entre un point dans la scène et les caméras. Le défi est d'identifier un point correspondant sur l'autre image. La meilleure façon pour comprendre ce qu'est le problème de la stéréoscopie est de l'illustrer comme à la figure 1.3. On peut voir sur cette image que les prises de vue sont similaires sans être identiques. En effet, nous apercevons sur l'image de la figure 1.3a que la personne en second plan est en arrière de l'autre personne, mais que dans l'image de la figure 1.3b nous pouvons apercevoir la silhouette de celle-ci. Cette figure illustre un principe d'occlusion qui sera présenté en détail à la section 1.2.

Pour trouver la distance d'un point à la caméra, il faut tout d'abord trouver la disparité associée à un point correspondant entre les deux images. La disparité est la différence entre la position en x d'un point dans une image avec le point correspondant dans l'autre image. Soit une image dans le spectre RGB et une dans le spectre LWIR, la disparité est donnée par la différence de leurs coordonnées en x :

$$d = \mid x_{RGB} - x_{LWIR} \mid \text{ tel que } x_{RGB} \text{ correspond à } x_{LWIR} \tag{1.1}$$

Nous pouvons voir à la figure 1.1c les points de disparités des deux spectres sur l'image RGB. Les points verts correspondent aux données de l'image RGB et les points rouges correspondent aux points correspondant dans l'image LWIR. Ces points sont les annotations fournis par les jeux de données. Le but de ce travail est d'associer les points vert et rouge correspondants. Si nous prenons un exemple du point vert au-dessus de la tête de la personne dans l'image 1.1c, la disparité est la distance en pixel entre le point vert et le point rouge correspondant.



(a) **(b)**

(c)

Figure 1.1 **(a)** Points sur une image RGB du jeu de donnée LITIV 2018 [1]. **(b)** Points sur une image LWIR du jeu de donnée LITIV 2018 [1]. **(b)** Disparités des points entre une image RGB et LWIR affichés sur une image RGB du jeu de donnée LITIV 2018 [1].

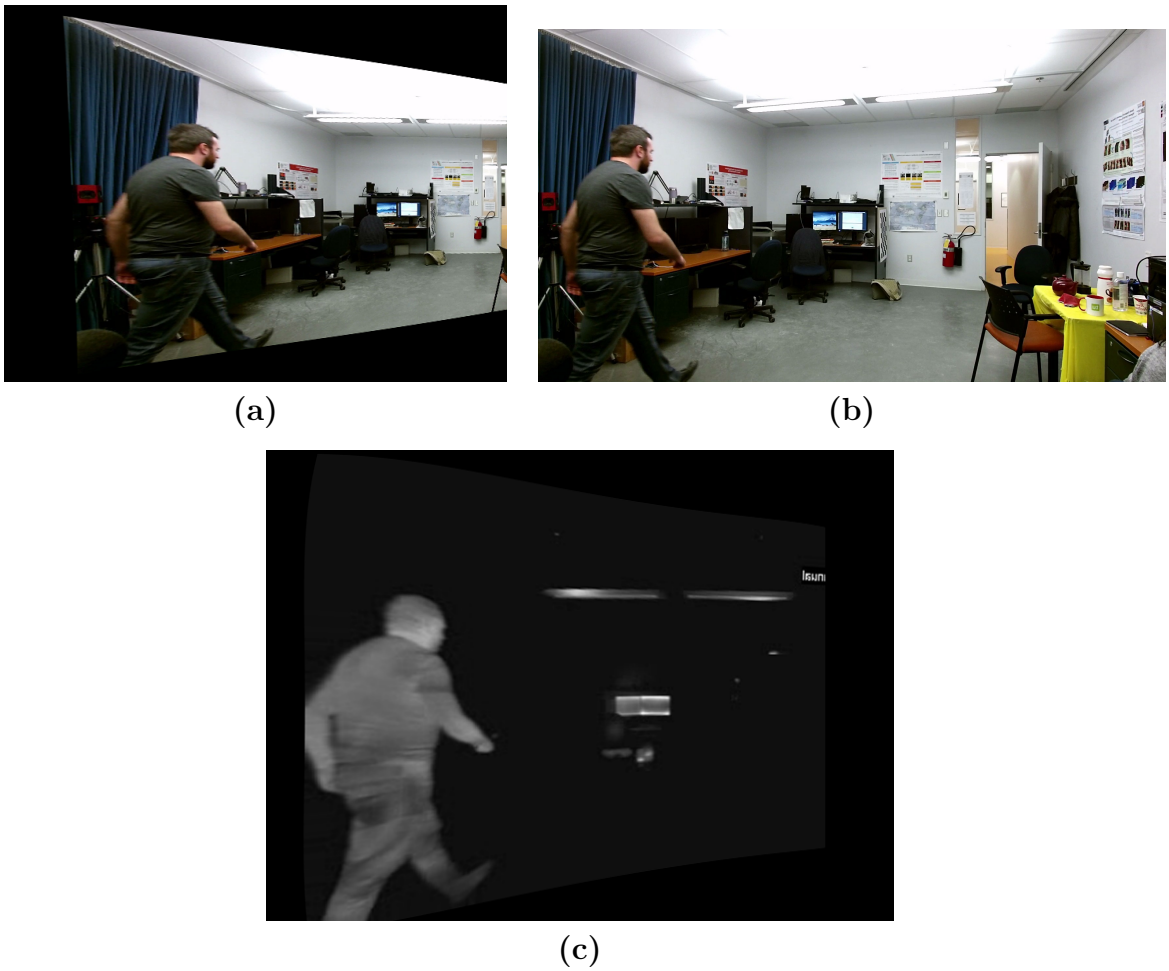
Pour pouvoir calculer la disparité à l'aide de l'équation 1.1, il faut tout d'abord traiter les images afin d'obtenir des images rectifiées. Les images rectifiées sont fournis dans les jeux de données LITIV 2014 [18] et LITIV 2018 [1]. Nous pouvons voir une même image rectifiée à la figure 1.2a et non-rectifiée à la figure 1.2b. Le principe de rectifier une image signifie d'aligner

l'ensemble des points pour faire en sorte que les points correspondant dans les deux images soient à la même coordonnée y dans l'autre image.

Une fois les images rectifiées et la position des deux objets identifiées dans les deux images, calculer la distance entre le point et les caméras est simple. Il suffit d'appliquer la formule suivante :

$$distance = \frac{fB}{d} \tag{1.2}$$

Où f représente la longueur focale de la lentille de la caméra, B la distance entre les deux caméras et d la disparité au point donnée.



(a) **(b)**

(c)

Figure 1.2 **(a)** Image RGB rectifié du jeu de donnée LITIV 2018 [1]. **(b)** Image RGB du jeu de donnée LITIV 2018 [1] **(c)** Image LWIR du jeu de donnée LITIV 2018 [1]

1.2 Éléments de la problématique

Les difficultés de la présente recherche peuvent être présentées en deux parties. Les difficultés reliées à la stéréoscopie classique, et celles attribuées à la stéréoscopie multispectrale. En stéréoscopie multispectrale, les difficultés reliées aux différents spectres s'ajoutent à ceux de la stéréoscopie classique. L'estimation de disparités en contexte multispectrale devient donc plus difficile qu'en stéréoscopie RGB-RGB classique. La section qui suit, présentera premièrement les difficultés associées à la stéréoscopie classique, et ensuite il sera mention des problématiques supplémentaires associées à la stéréoscopie multispectrale.

(a) **(b)**

Figure 1.3 **(a)** Image RGB du jeu de donnée LITIV 2018 [1] formant une occlusion. **(b)** Image LWIR du jeu de donnée LITIV 2018 [1] formant une oclusion.

1.2.1 Difficulté en stéréoscopie classique

En stéréoscopie classique, plusieurs difficultés rendent le problème complexe. Les problèmes récurrents en stéréoscopie sont l'occlusion, les surfaces sans textures et les points de disparités aux frontières.

La première difficulté associée à la stéréoscopie est l'occlusion. L'occlusion consiste à ce qu'une partie de l'image ne soit pas visible dans la seconde image stéréoscopique. Donc, des points correspondants sont manquants ou invisibles. L'occlusion se fait soit sur les bordures des images (points manquants) ou sur des objets de second plan (points invisibles). Les points qui subissent de l'occlusion sont donc causés par d'autres objets dans la scène ou par leurs proximités aux frontières. Il s'agit d'une des plus grandes difficultés du problème. Nous pouvons voir une occlusion au niveau du cou de la personne de second plan sur la figure 1.3b qui n'apparaît pas sur la figure 1.3a.

La seconde difficulté en stéréoscopie est l'absence de texture dans les images. L'absence de texture est un défi, car il est difficile d'identifier des points correspondants si un point est sur une surface de couleur uniforme. En effet, l'ensemble des sous-régions des pixels environnants vont avoir les mêmes ressemblances. Dans le cas de deux images stéréoscopiques en couleur, il sera difficile d'estimer la disparité précise d'un point sur un mur peu éclairé. Par exemple, si la paire d'images à la figure 1.3 était une paire d'images RGB, il serait difficile d'estimer un point sur le mur blanc derrière, ou aussi il serait difficile d'estimer la disparité d'un point sur le chandail noir de la personne en premier plan à la figure 1.3a.

Le dernier problème relié à la stéréoscopie classique est la précision de l'estimation de disparités aux frontières. L'explication de cette difficulté est proche de celle de la première. En effet, un point à la frontière d'une image sera difficile à estimer, car un peu comme l'occlusion, le point sera visible que sur une seule image.

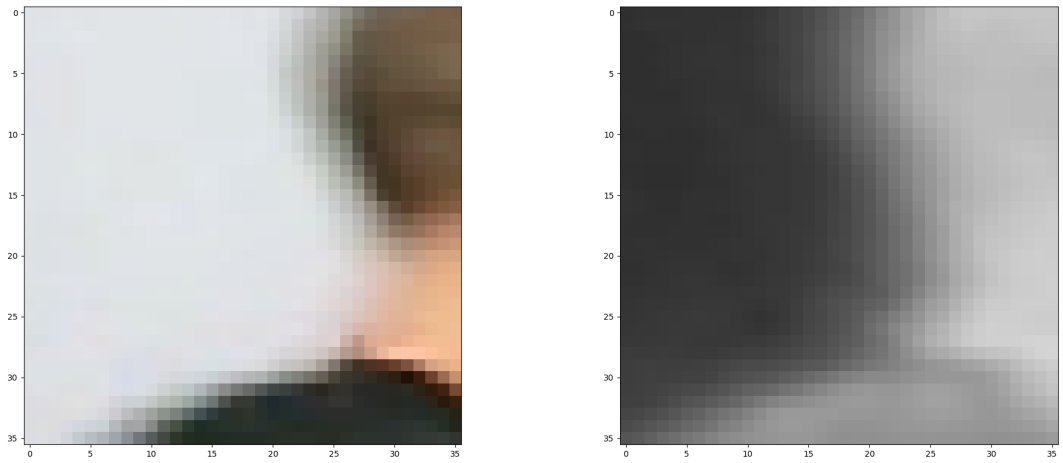
1.2.2 Difficulté en stéréoscopie multispectrale

Comme mentionné précédemment, toutes les difficultés mentionnées pour la stéréoscopie classique sont aussi des difficultés pour la stéréoscopie multispectrale. Quelques difficultés s'ajoutent donc à la liste. L'estimation de la disparité est plus complexe dans ce cas à cause de l'absence de couleurs, le bruit dans la mesure de l'émission de chaleur aux frontières, la différence de textures, et les objets froids.

Le fait de comparer deux spectres augmente la complexité du problème d'estimation de disparités. Les explications reliées aux difficultés sont reliées au spectre LWIR, étant donné que le mémoire porte sur la stéréoscopie RGB-LWIR. Cependant, plusieurs autres spectres sont aussi étudiés dans la littérature. En stéréoscopie multispectrale, les plus populaires sont les spectres RGB-LWIR et RGB-NIR. En effet, plusieurs jeux de données sont disponibles pour ces spectres. D'autres spectres sont étudiés, mais sont significativement moins courants et ont une moins grande présence dans la littérature.

La différence majeure entre la stéréoscopie classique et multispectrale RGB-LWIR est l'absence de couleurs dans le spectre LWIR. Ceci fait en sorte qu'une information significative est manquante entre les deux images, et la présente méthode de stéréoscopie doit utiliser d'autres repères que la couleur pour estimer la disparité. Les corrélations entre les pixels de la sous-région auront donc significativement moins de correspondances.

Une autre difficulté est l'émission de chaleur des corps chauds qui rendent les frontières bruitées. En effet, étant donné que les images LWIR résultent de la chaleur émise dans la scène, les frontières ne vont pas être bien définies dues à la dissipation de chaleur des objets



(a) **(b)**

Figure 1.4 **(a)** Sous-région d'une image RGB du jeu de donnée LITIV 2018 [1]. **(b)** Sous-région d'une image LWIR du jeu de donnée LITIV 2018 [1].

chauds dans la scène. Nous pouvons voir le phénomène autour de la tête de la personne à la figure 1.4. Cette figure consiste à deux images à des positions relatives identiques. Nous pouvons voir à la figure 1.4b que la frontière est moins définie qu'à la figure 1.4a. Ceci est le résultat de l'émission de chaleur captée par la caméra infrarouge.

Étant donné que le spectre LWIR capte des températures, les textures varient entre une surface sur une image LWIR et RGB. En effet, certains motifs d'un chandail, par exemple, ne vont pas apparaître sur une image LWIR. On verra plutôt des textures thermiques montrant des variations de chaleur. Nous pouvons voir sur la figure 1.3 qu'il y a une différence de texture entre les deux sous-images. En effet, la surface du chandail noire est plus texturée sur la figure 1.3b et le logo qui apparaît sur la figure 1.3a est absent sur la figure 1.3b.

Une dernière difficulté est l'estimation de disparité sur les objets froids. En effet, la disparité sera difficile à estimer sur les objets froids, car aucune source de chaleur ne provient de l'objet. Cela revient donc à avoir un objet noir sans texture. Nous pouvons voir la différence d'apparence d'un objet froid dans les figures 1.5a et 1.5b. À la figure 1.5b, nous pouvons voir que la bouteille d'eau que la personne tient dans ses mains est uniformément noire, ce qui veut dire que la bouteille d'eau est froide. L'objet étant froid, il n'y a pas de textures comparées à une image RGB.

(a) **(b)**

Figure 1.5 **(a)** Image RGB contenant un objet froid dans le jeu de donnée LITIV 2018 [1]. **(b)** Image LWIR contenant un objet froid dans le jeu de donnée LITIV 2018 [1].

1.3 Objectifs de recherche

L'objectif du projet est de développer une méthode d'estimation de disparité entre des paires d'images stéréoscopiques couleur-thermique (LWIR). Le projet se basera sur des travaux précédents ayant les mêmes objectifs que les nôtres. Nous allons également adapter à notre problème de nouvelles solutions récemment publiées dans le but d'améliorer la méthode actuelle. Plus précisément, l'objectif de ce mémoire est d'améliorer les techniques d'estimation de disparités sur des paires stéréoscopiques RGB-LWIR. Les objectifs spécifiques sont :

— Améliorer la disparité estimée aux frontières
— Mieux décrire le contenu de chaque images de la paire stéréoscopique pour avoir plus de détails à mettre en commun dans chacune des images de chaque spectre.
— Améliorer l'estimation de disparité sur de petites précisions, et tenter d'avoir une estimation parfaite pour l'estimation sous 5 pixels.

1.4 Plan du mémoire

Le mémoire est constitué de plusieurs sections. Premièrement, dans le chapitre 2, une revue de littérature sera faite sur les techniques d'estimation de disparité classiques, soit les méthodes datant d'avant l'avènement de l'apprentissage machine. Ensuite, la revue de littérature énumérera les techniques de stéréoscopie à l'aide de réseaux de neurones sur des paires d'images en couleur et les techniques multispectrales suivront. La dernière section de la revue de littérature présentera des techniques de stéréoscopie à l'aide des réseaux de neurones à

auto-attention. Après la revue de littérature, le chapitre 3 vous présentera la méthodologie du projet. Le chapitre 4 montrera les expérimentations. Pour finir, le chapitre 5 sera une conclusion qui présentera la synthèse des travaux effectués, des limitations du projet actuel et pour finir des suggestions d'améliorations futures seront présentées.

CHAPITRE 2 REVUE DE LITTÉRATURE

Cette section a pour but de donner une vue d'ensemble des travaux effectués au cours des dernières années en stéréoscopie. Plusieurs types de stéréoscopies sont étudiés, et il sera mention de chacun de ceux-ci dans la présente revue de littérature. Il existe en effet la stéréoscopie entre deux images couleurs du spectre visible (RGB-RGB), mais il y a également de la stéréoscopie multispectrale. La stéréoscopie multispectrale est le fait d'estimer la disparité entre deux images de deux spectres différents. Dans le cas de cette revue de littérature, nous présenterons les deux types de stéréoscopies multispectrales les plus étudiés, soit la stéréoscopie entre une image couleur et une image infrarouge thermique (RGB-LWIR) ainsi que la stéréoscopie entre une image couleur et une image infrarouge proche (RGB-NIR). Les méthodes expliquées seront principalement des méthodes utilisant de l'apprentissage automatique. Il y aura un bref survol des méthodes classiques pour comprendre l'origine des méthodes d'apprentissage automatique.

Deux types d'annotations seront considérés dans cette revue de littérature. Les annotations denses et éparses. Les annotations denses sont des annotations dont les points de disparités sont connus à chaque pixel dans les deux images. Les annotations éparses quant à elles, sont des annotations où seulement quelques données sont présentes dans l'image. Le type d'annotations disponible a généralement un impact sur la conception des méthodes. Par exemple, un apprentissage bout à bout sur une image complète n'est pas possible avec des annotations éparses. Pour les annotations éparses, on doit plutôt analyser les images par sous-régions. Il y a donc des méthodes qui produisent des cartes de profondeur denses, alors que d'autres méthodes produisent des cartes de profondeur éparses.

2.1 Stéréoscopie classique

Cette section fera un aperçu des méthodes stéréoscopiques utilisées avant l'ère des réseaux de neurones. Les méthodes pré-réseaux de neurones sont importantes étant donné qu'elles ont été une source d'inspiration pour les méthodes plus modernes, et encore à ce jour des ressemblances existent entre les deux types de méthodes. Cependant, ces méthodes étant obsolètes, elles ne sont pas mobilisées explicitement pour la compréhension de la méthode qui est proposée dans ce mémoire. Les méthodes plus modernes utilisant des réseaux de neurones seront donc partie intégrante de cette revue de littérature.

2.1.1 Stéréoscopie RGB-RGB

Commençons tout d'abord par présenter des méthodes de stéréoscopie classique dans le spectre RGB-RGB. Une étude importante combinant plusieurs méthodes stéréoscopiques a été faite par Scharstein et al. [19]. Il s'agit d'une taxonomie des algorithmes de stéréoscopie dense. Ils décortiquent les différentes étapes de chaque algorithme classique.

Selon Scharstein et al. la première étape est **le calcul des coûts de correspondance**. Les méthodes de correspondance au niveau des pixels les plus utilisés sont soit la différence au carré (SD) [20–23] ou la différence absolue (AD) [24]. Cette étape sert à extraire les informations des images pour comparer des groupes de pixel entre eux afin de pouvoir trouver des ressemblances.

L'**agrégation des coûts** est la deuxième étape des algorithmes de stéréoscopie selon la taxonomie précédente. L'étape d'agrégation des coûts sert à agréger les coûts dans une certaine région de l'image. L'agrégation des coûts permet de calculer des correspondances entre des groupes de pixels au lieu de calculer des correspondances pixel par pixel. Le résultat de cette agrégation crée des volumes de coûts. Plusieurs méthodes sont utilisées pour agréger les coûts. Il est mention de convolutions gaussiennes, des fenêtres glissantes à taille adaptative [25–28], et des fenêtres glissantes à taille fixe [29]. Ces méthodes changent l'espace dans lequel les coûts sont calculés. En effet, les fenêtres glissantes par exemple, appliquent le coût de correspondance sur une sous-région entourant le pixel dont on veut trouver la disparité. Ceci ajoute donc un contexte au pixel, ajoutant ainsi de l'information permettant d'améliorer l'estimation de disparité.

La troisième étape est **le calcul et l'optimisation des disparités**. Quatre méthodes sont présentées. Les méthodes locales, les méthodes d'optimisation globales, la programmation dynamique et des algorithmes coopératifs. Ce sont toutes des techniques qui sont appliquées à l'espace dans laquelle des coûts sont calculés, soit l'espace établi à l'étape précédente. Les meilleures correspondances sont celles pour lesquelles le coût est minimum.

Hirschmuller et al. [30] ont proposé une méthode appelée *semi-global matching* (SGM). Cette méthode est encore utilisée de nos jours pour trouver les cartes de disparités à la fin des réseaux de neurones.

La quatrième et dernière étape est le **raffinement des disparités**. Cette dernière étape combine plusieurs types de méthodes afin de nettoyer les données pour réduire le bruit en uniformisant les disparités des pixels adjacents.

Avant l'avènement des réseaux de neurones, la stéréoscopie utilisait aussi des points caractéristiques pour estimer la disparité de certains points dans une scène. Une façon de faire

est d'utiliser les points clefs SIFT [31] comme descripteurs de caractéristiques, qui peut remplacer l'étape 1 de la taxonomie de Scharstein et al. [19]. Ces points peuvent ensuite être associés pour trouver la disparité dans une paire image. Cette méthode s'applique en général pour la stéréoscopie éparse.

2.1.2 Stéréoscopie Multispectrale

Les méthodes précédemment mentionnées, telles que la différence au carré (SD) [20–23] ou la différence absolue (AD) [24], ne peuvent pas être utilisées pour la stéréoscopie multispectrale, car ces méthodes se basent sur la valeur des pixels pour trouver des ressemblances. Donc, étant donné que les images RGB ont 3 canaux de couleurs, et que l'image thermique n'en a qu'un seul, ces méthodes de comparaisons de pixels ne fonctionnent pas.

De SIFT [31] a émergé MSIFT [32]. MSIFT modifie SIFT pour l'adapter au problème multispectral, en améliorant la corrélation entre les canaux RGB pour une paire d'images RGB-NIR. Au lieu de se servir de descripteurs de caractéristiques, certaines méthodes éparses et denses utilisent des fenêtres glissantes pour trouver les similitudes dans les images. Tel est le cas de la méthode d'information mutuelle [33], HOG [34], SSD [18], LSS [35] et plus encore. Selon Bilodeau et al. [18], la méthode d'information mutuelle [36] est la méthode par fenêtre glissante la plus efficace pour la stéréoscopie RGB-LWIR. Ils utilisent l'information mutuelle [36] pour former les volumes de coûts, malgré que SGM ne se restreint pas uniquement aux descripteurs d'information mutuelle [36].

2.2 Méthode d'apprentissage profond pour la stéréoscopie

Dans cette section, une revue de littérature sera faite sur les méthodes d'estimation de disparités à l'aide de l'apprentissage profond. Il sera mention des techniques de stéréoscopie RGB, soit la stéréoscopie composée de deux images RGB, et une autre sous-section qui présentera les méthodes multispectrales. Les méthodes multispectrales consistent à faire de l'estimation de disparité entre deux images de spectres différents. Parmi ces spectres se trouvent le spectre visible, communément référé par RGB, le spectre infrarouge thermique, soit LWIR et le spectre infrarouge proche, soit NIR. Les méthodes présentées dans cette section sont des méthodes RGB-LWIR ainsi que des méthodes RGB-NIR.

2.2.1 Stereoscopie RGB

Méthode par sous-régions

Les méthodes par sous-régions sont les méthodes dont les entrées du réseau correspondent à des sous-régions de l'image originale. Donc au lieu de trouver l'ensemble des disparités sur l'image, les disparités sont trouvées en isolant une sous-région autour du point de disparité sur une des deux images, et le but du réseau est de trouver la sous-région correspondante dans l'autre image. La disparité finale sera donc calculée en trouvant la distance entre la position de la sous-région originale, et la sous-région correspondante dans l'autre image. Les méthodes par sous-régions fonctionnent aussi bien avec les annotations denses et éparses.

Plusieurs travaux étudient la stéréoscopie RGB. Le premier travail ayant utilisé les réseaux de neurones convolutifs (RNC) pour résoudre le problème de disparité a été celui de Zbontar and LeCun [2]. Leur méthode consiste à prendre une région de pixels 9×9 sur l'image de droite et de gauche. Le but était d'apprendre les similarités entre les deux régions, et par la suite, étant donné que la position des sous-régions est connue, la disparité peut être établie. Les paires de sous-régions en entrée du réseau sont un mélange de paires d'images correspondantes et d'autres qui ne correspondent pas. Ceci fait en sorte que le réseau identifie si les sous-régions correspondent ou non. Le fonctionnement de la méthode est le suivant. La méthode détermine si la sous-région à la position (x_1, y) correspond à la sous-région (x_2, y). Si c'est le cas, la disparité entre les deux sous-régions sera environ de $|x_2 - x_1|$. L'architecture de leur modèle est illustrée à la figure 2.1. Elle est composée d'une couche de convolution suivie d'une couche complètement connectée pour chaque sous-région de 9×9. Par la suite, les sorties de ces deux sous-réseaux sont concaténées et sont passées dans des couches entièrement connectées. La sortie du réseau est une classification indiquant si les sous-régions sont similaires ou non.

Zbontar and LeCun ont inspiré plusieurs autres méthodes. Une d'entre elles est la méthode proposée par Chen et al. [3]. Elle va dans la même direction que la précédente [2], mais essaye de trouver des différences, et ce avec des sous-régions de tailles différentes. Leur réseau consiste en deux réseaux siamois qui se partagent les poids de leurs RNCs respectifs. Un réseau siamois est utilisé pour les images de tailles originales 13×13 et l'autre pour les images redimensionnées avec un facteur $\times 2$. Le réseau bleu dans l'image 2.2 représente le réseau siamois associé aux images originales. Il est à noter que les deux réseaux siamois ont une architecture identique. Chacun est composé de quatre couches de convolutions. Après la dernière couche de convolution, une opération de corrélation est faite pour trouver la similitude de la sous-région du présent réseau. Avec le résultat de la corrélation des deux réseaux siamois, un vote est fait à savoir si les sous-régions correspondent ou non.

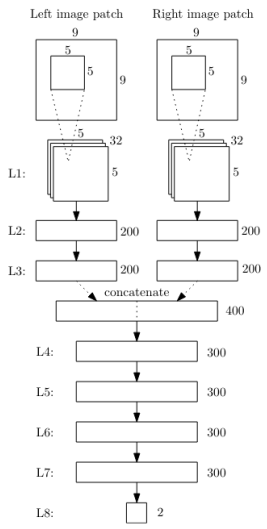


Figure 2.1 Architecture de Zbontar and LeCun [2] © 2015 IEEE.

Malgré le fait que la méthode proposée par Chen et al. [3] soit près de 100 fois plus rapide que les méthodes antérieures, ces algorithmes restent assez lents. Luo et al. [4] ont proposé une méthode qui améliore de façon significative le temps d'inférence. Leur approche consiste à prendre une sous-région de 9×9 de l'image de gauche, et d'une sous-région de la même hauteur pour l'image de droite, mais avec une largeur équivalente à la disparité maximale. Un vecteur de caractéristiques est calculé pour la sous-région 9×9 et un volume de caractéristiques est calculé pour la sous-région plus large. Une fois que les caractéristiques des deux sous-régions ont été extraites, un produit de corrélation est fait entre le vecteur de caractéristique de la sous-région de référence et chacun des vecteurs de caractéristique dans le volume de caractéristiques de la sous-région plus large. Ces produits de corrélations donnent une distribution de probabilité qui permet de trouver la disparité associée à chacun des vecteurs de caractéristique dans le volume précédemment trouvé. Ce vecteur de corrélation final combine le vecteur de corrélation de l'image de droite et celui de gauche pour rendre la stéréoscopie plus efficace. L'architecture de cette méthode est présentée à la figure 2.3.

Les méthodes [2, 3] correspondent à des problèmes de classifications binaires, évaluant les probabilités que les sous-régions correspondent entre elles ou non. Les deux méthodes se basent sur SGM [30] pour la création de la carte de disparité finale.

Vu le succès de la méthode de Chen et al. [3], Park et al. [5] augmentent aussi la taille des sous-régions utilisées pour pouvoir tenir en compte le contexte des pixels qui entourent le point de disparité. Ils augmentent la taille des sous-régions observées avec l'hypothèse que le contexte général de la sous-région est important, et que si les sous-régions sont trop petites,

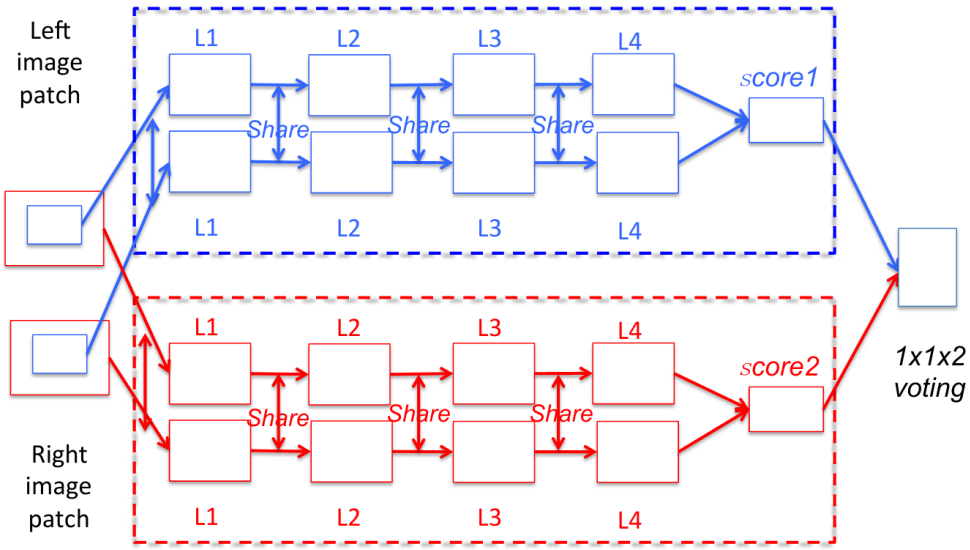


Figure 2.2 Architecture de Chen et al. [3] © 2015 IEEE.

le contexte se perd. Park et al. [5] utilisent des sous-régions de 37×37 pour augmenter l'information associée aux sous-régions, soit des sous-régions considérablement plus grande que les sous-régions utilisées dans les travaux antérieurs. L'architecture utilisée est une version modifiée de celle des pionniers du sujet, soit Zbontar and LeCun [2]. La modification effectuée à l'architecture de Zbontar and LeCun [2] est l'ajout d'un modèle d'agrégation de pixels pyramidal, soit la couche **4P** dans la figure 2.4.

Comme nous avons pu le voir précédemment, deux types de têtes de réseau étaient utilisés. Soit une corrélation ou une concaténation des vecteurs de caractéristiques. Shaked et al. [37] sont les premiers à proposer une méthode combinant ces deux types de tête. Les images de droite et de gauche sont passées dans un réseau siamois, dont les branches sont constituées de couches résiduelles [38] ainsi que de RELU pour extraire les caractéristiques des images. Ces deux vecteurs de caractéristiques sont en premier lieu concaténés, et grâce à une fonction de perte d'entropie croisée binaire, le réseau va déterminer si la sous-région est la même ou non. Dans la seconde branche, un produit croisé est fait entre les deux vecteurs de caractéristiques, et la fonction de perte de Hinge est appliquée pour savoir si les sous-régions correspondent ou non. En utilisant les deux fonctions de pertes, ils disent utiliser une fonction de perte hybride. Nous pouvons voir des similarités entre la concaténation utilisée ici et celle de Zbontar et al. [2] Une similarité est aussi présente entre le produit croisé et la corrélation de Luo et al. [4]

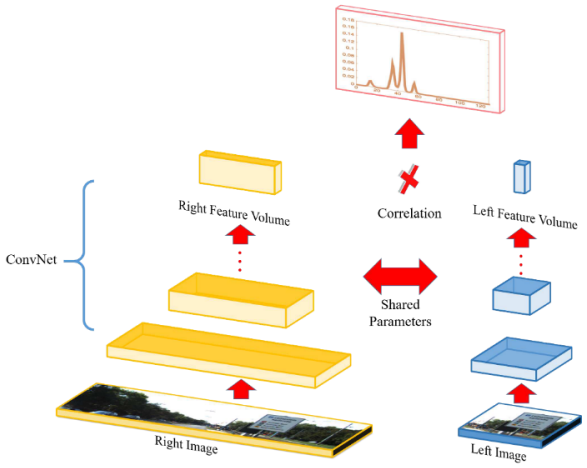


Figure 2.3 Architecture de Luo et al. [4] © 2016 IEEE.

Méthodes bout-à-bout

Les méthodes d'estimation de disparités bout-à-bout sont plus complexes que les méthodes par sous-régions. En effet, afin de limiter le sur-apprentissage, il faut des jeux de donnée plus complets qui sont composés de plus d'annotations par images. En effet, ces méthodes ont comme entrée les images complètes contrairement à un ensemble de sous-régions. Donc, il faut les disparités à chaque pixel pour l'entraînement. Ceci fait en sorte que ce type de méthode permet de créer des cartes de disparité plus précises. L'utilisation de SGM [30] n'est plus nécessaire pour créé la carte de disparité avec ces méthodes bout-à-bout, étant donné que la carte de disparité est la sortie du réseau.

Le premier travail à avoir proposé une architecture de type bout-à-bout était Mayer et al. [39] La contribution majeure qu'ils ont ajoutée est le jeu de données **FlyingThings3D**. Ce jeu de données consiste d'images possédant des disparités connues à chaque pixel, ce qui a permis par la suite à plusieurs travaux d'étudier davantage les méthodes bout-à-bout. Leur méthode qu'ils ont appelée DispNet est inspirée de FlowNet. [40] Le réseau est séparé en deux parties. La partie de compression, et la partie de décompression. La partie de compression contient des convolutions qui résultent à un facteur de réduction final de 64. La décompression fait ensuite un redimensionnent des cartes de disparités de façon graduelle et non linéaire, prenant en considération les caractéristiques prises dans la compression. Le résultat final de ce réseau est donc la carte de disparité à une résolution équivalente à l'image originale.

Kendall et al. [6] ont introduit le réseau GC-Net, qui a été le premier travail à considérer les formes géométriques et le contexte dans leurs estimations des disparités. Ils ont utilisé un réseau siamois pour extraire les caractéristiques des images en entrée, et ils ont utilisé

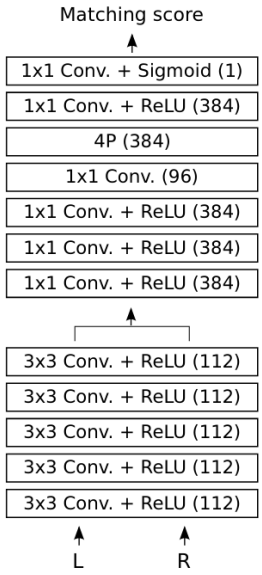


Figure 2.4 Architecture de Park et al. [5] © 2016 IEEE.

des convolutions 3D pour apprendre les vecteurs de caractéristiques. Par la suite, une régression est faite pour donner les cartes de disparité résultantes. La principale contribution de ce réseau est les volumes de coûts 3D que nous pouvons voir à la figure [6]. Ces volumes remplacent l'ancienne méthode de naïvement concaténer les cartes de caractéristiques. Les auteurs disent que la création de volume de coûts 3D permet de maintenir l'information des formes géométriques dans l'image. GC-Net est aussi un des premiers réseaux qui laisse de côté la classification, et approche le problème avec une régression.

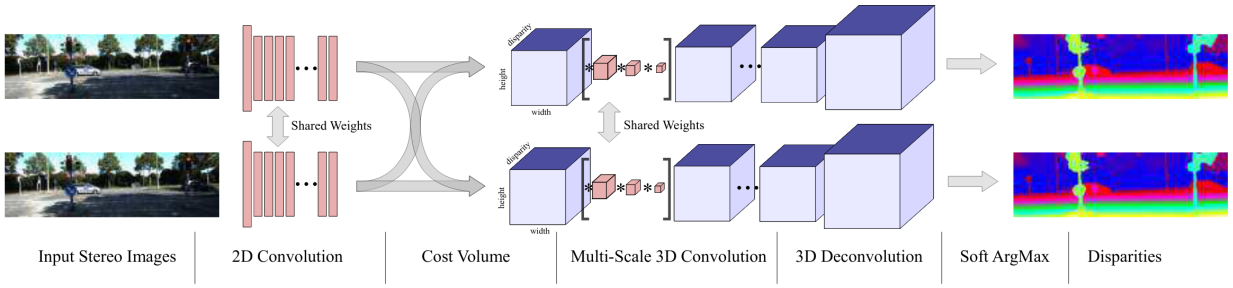


Figure 2.5 Architecture de Kendall et al. [6] © 2017 IEEE.

Plusieurs autres méthodes se sont basées sur les méthodes précédentes, avec quelques variations et améliorations. Par exemple, Chang et al. [7] ont utilisé un module **Spatial Pyramid Pooling** [8] dans leur réseau PSMNet pour extraire les caractéristiques importantes dans les

vecteurs de caractéristiques. Le SPP permet d'extraire des caractéristiques de l'image à plusieurs niveaux. Nous pouvons voir la partie SPP du réseau dans l'encadré rouge de la figure 2.6. Pour la régularisation du volume de coûts et la régression de la disparité, ils ont utilisé un réseau sablier (*Hourglass*). La partie sablier du réseau correspond à la partie de compression et décompression que les méthodes précédentes utilisaient pour générer la carte de disparités finale. Dans le cas de PSMNet, il y a trois compression et décompression, ce qui correspond à un sablier empilé.

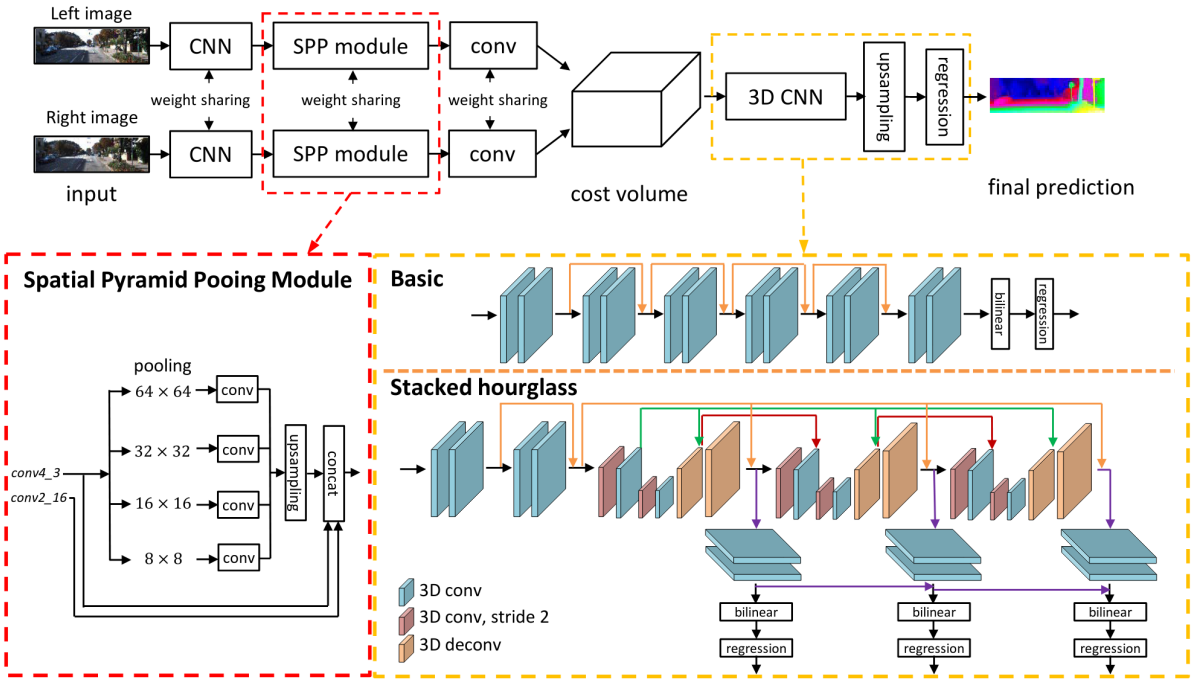


Figure 2.6 Architecture de Chang et al. [7] © 2018 IEEE, utilisant un module SPP [8]

Pang et al. [9] s'inspirent grandement de DispNet [39] pour leur architecture. En effet, la première partie de leur réseau consiste à la même architecture que DispNet pour obtenir la carte de disparité. La deuxième partie du réseau a pour but d'améliorer la précision des estimations de disparités dans les zones difficiles. Cette addition au réseau DispNet peut être vue sur la figure 2.7 avec l'étiquette DispResnet. Il s'agit encore une fois d'une étape de compression et de décompression (réseau sablier). Cette deuxième partie du réseau raffine les disparités en prenant en entrée les éléments suivants dans la figure 2.7 :

— L'image de gauche, représentée par I_L ;
— L'image de droite, représentée par I_R ;
— L'image de gauche modifiée à l'aide de la carte de disparité (d_1), représentée par $\tilde{I}_L$;

— Une carte d'erreur entre l'image de gauche originale et l'image de gauche transformée, représentée par e_L.

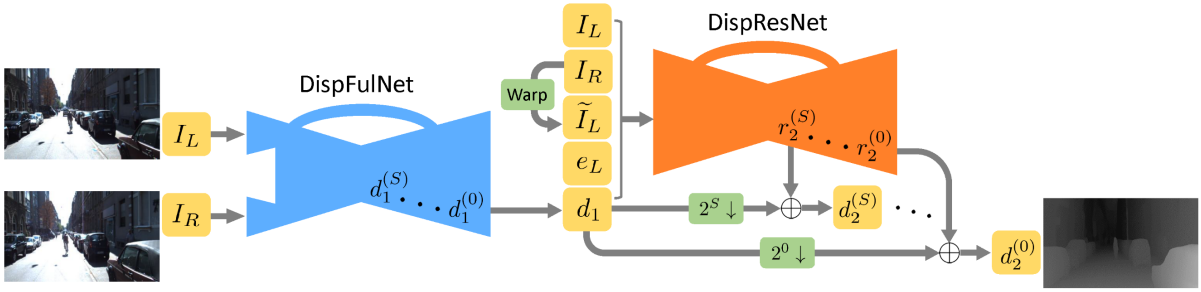


Figure 2.7 Architecture de Pang et al. [9] © 2017 IEEE.

Guo et al. [10] se sont inspirés de Shaked et al. [37] en combinant les opérations de fusion de vecteurs de caractéristiques. En effet, ils utilisent la concaténation ainsi que la corrélation. Leur contribution consiste à créer des volumes de coût pour chaque opération de fusion. Ils appellent leurs opérations de fusion des opérations de corrélation groupée. Le but de la corrélation groupée est de séparer les caractéristiques en groupes, et par la suite, trouver les disparités de ces caractéristiques. Décortiquons maintenant l'architecture corrélation groupée. Considérant que chaque vecteur de caractéristiques à N_c canaux, et que le nombre de canaux finaux désirés est noté comme étant N_g, le nombre de canaux final sera donc de $\frac{N_c}{N_g}$. La corrélation est faite entre chaque groupe g et la disparité est trouvée pour chacun de ces groupes. Nous pouvons voir l'architecture de Guo et al. à la figure 2.8.

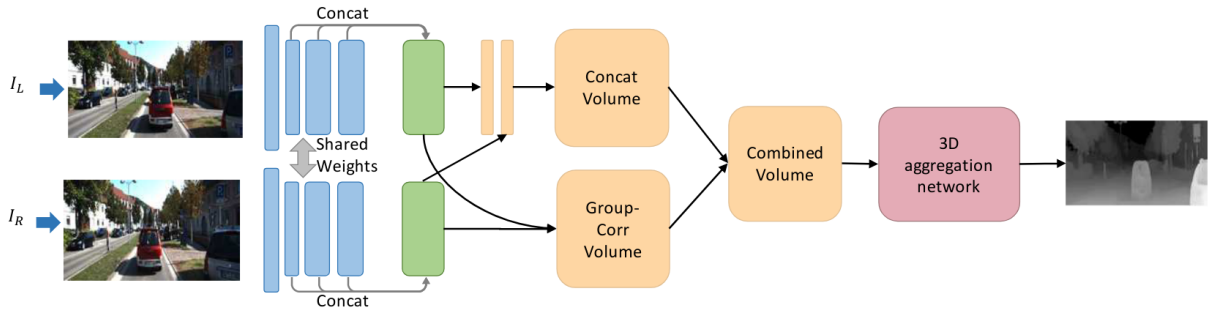


Figure 2.8 Architecture de Guo et al. [10] © 2019 IEEE.

2.2.2 Stéréoscopie Multispectrale

Certaines études ont été faites en stéréoscopie multispectrale. Pistarelli et al. [41] proposent une méthode qui se base sur l'identification des arêtes, étant donné que les arêtes sont l'une des informations les plus similaires entre LWIR et RGB (surtout sur les contours des objets). Ils utilisent Canny [42] pour obtenir les arêtes des images. Ils transforment les images d'arêtes pour les mettre dans un espace commun, soit l'espace de Hough. Ce nouvel espace permet de comparer les sous-régions et d'indiquer si deux points de disparité correspondent.

Baruch et al. [11] propose un modèle constitué de deux RNCs, tel que nous pouvons le voir à la figure 2.9. Le but des deux réseaux est de trouver les points de correspondances. Le premier réseau consiste en un réseau siamois traditionnel, donc les deux RCNs partagent les paramètres. La deuxième partie du RNC est dite asymétrique, car elle est constituée de deux réseaux de convolution à architecture identique, qui extraient les informations uniquement du spectre associé. Cependant les paramètres ne sont pas partagés entre les deux RNCs, laissant des paramètres séparés entre chaque spectre. Ces réseaux extraient les caractéristiques des images en parallèle indépendamment du spectre, ce qui fait en sorte que les réseaux associés à chaque spectre ont leurs paramètres correspondants.

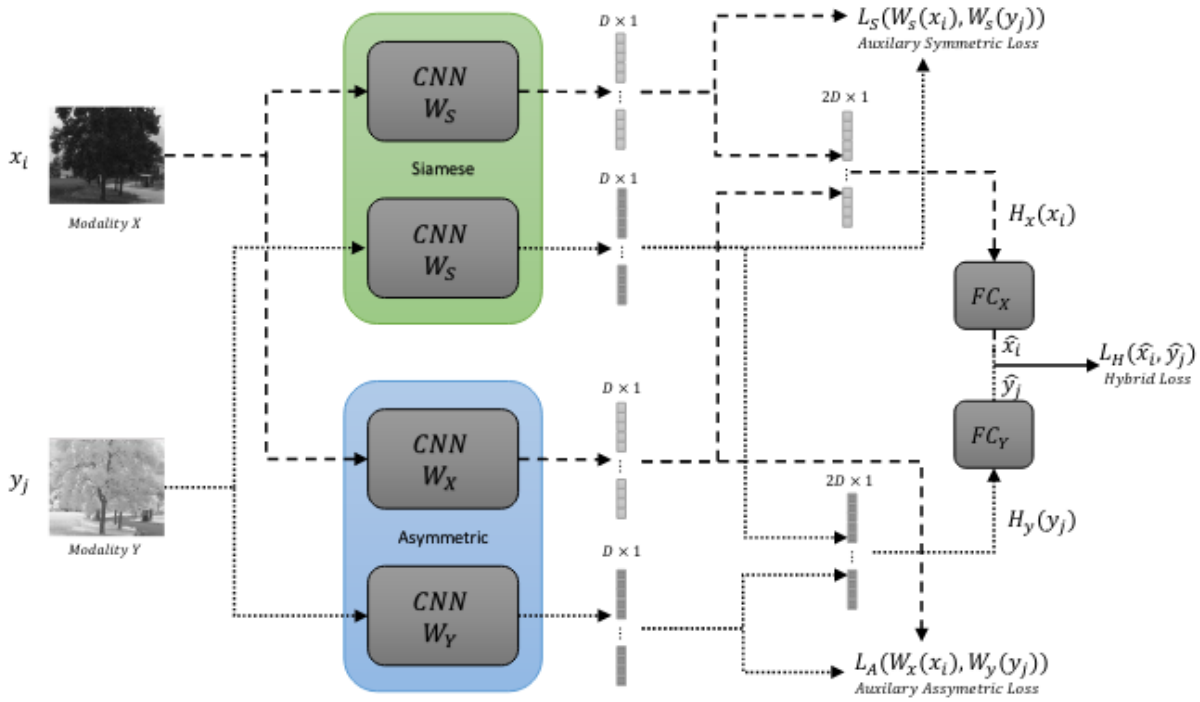


Figure 2.9 Architecture de Baruch et al. [11]

Des études plus récentes ont étudié la stéréoscopie RGB/LWIR. Beaupré et al. [12] ont proposé une méthode qui consiste d'avoir deux réseaux siamois qui partagent les mêmes paramètres. Un réseau a pour but de trouver la disparité pour des images RGB à LWIR, et l'autre pour des images LWIR à RGB. Nous pouvons voir l'architecture du réseau à la figure 2.10. Le principe des réseaux siamois utilisé par Beaupré et al. [12] est similaire à la méthode de stéréoscopie classique de Luo et al. [4]. La première partie du réseau (N_L) compare une petite sous-région d'une image RGB à une sous-région du LWIR de hauteur similaire, mais de la largeur de l'image originale. La sous-région RGB est une petite partie de l'image entourant le point de disparité que le réseau doit estimer. Cette sous-région est ensuite donnée au réseau comme première entrée. L'autre entrée consiste de la sous-région LWIR qui pour sa part est plus large que la précédente. Une fois la totalité des vecteurs de caractéristiques générés, une corrélation est faite entre la sous-région RGB et chaque translation possible dans la sous-région LWIR, qui est de la même largeur que l'image originale. Pour l'autre réseau siamois (N_R), le même principe s'applique, mais la petite sous-région est la LWIR et la sous-région plus large est la RGB. Pour sélectionner la disparité finale, une couche d'addition est mise en place pour additionner les vecteurs de prédictions de chacune des branches. La disparité finale est donnée par la position de la valeur maximale dans ce vecteur.

Un plus récent travail effectué par Beaupré et al. [13] modifie la précédente méthode. Nous pouvons voir l'architecture de ce réseau à la figure 2.11. La principale différence avec le dernier travail est que les poids des réseaux siamois ne sont pas partagés entre les RNCs. Ce changement améliore significativement les résultats. La raison pour laquelle les poids ne sont pas partagés est due à la nature de chacun des spectres, qui n'ont pas du tout les mêmes caractéristiques. Dans la plupart des utilisations actuelles des réseaux siamois, les images en entrée sont similaires en termes de couleur, forme, et contraste. Ce n'est pas le cas pour la stéréoscopie RGB-LWIR. Les deux images n'ont pas beaucoup d'informations en commun. La seule information commune est la forme des objets émettant de la chaleur. Les formes ne sont pas exactement les mêmes étant donné que les images infrarouges capturent la chaleur émise par le corps humain, et forment des bordures moins précises autour des objets en comparaison avec les images RGB.

Les spectres RGB et LWIR ne sont pas les seuls spectres de la lumière étudiés en stéréoscopie. Plusieurs travaux ont été faits en stéréoscopie RGB-NIR. Aguilera et al. [43] est un exemple de stéréoscopie RGB-NIR. Ils ont fait une étude comparant trois différentes architectures RNCs pour voir s'ils sont plus efficaces que les méthodes classiques brièvement expliquées précédemment. La première méthode consiste en un réseau RNC qui prend en entrée les deux images concaténées. La deuxième méthode consiste en un réseau siamois, donc deux RNCs dont les poids sont partagés, et la troisième méthode est d'utiliser un réseau pseudo-siamois,

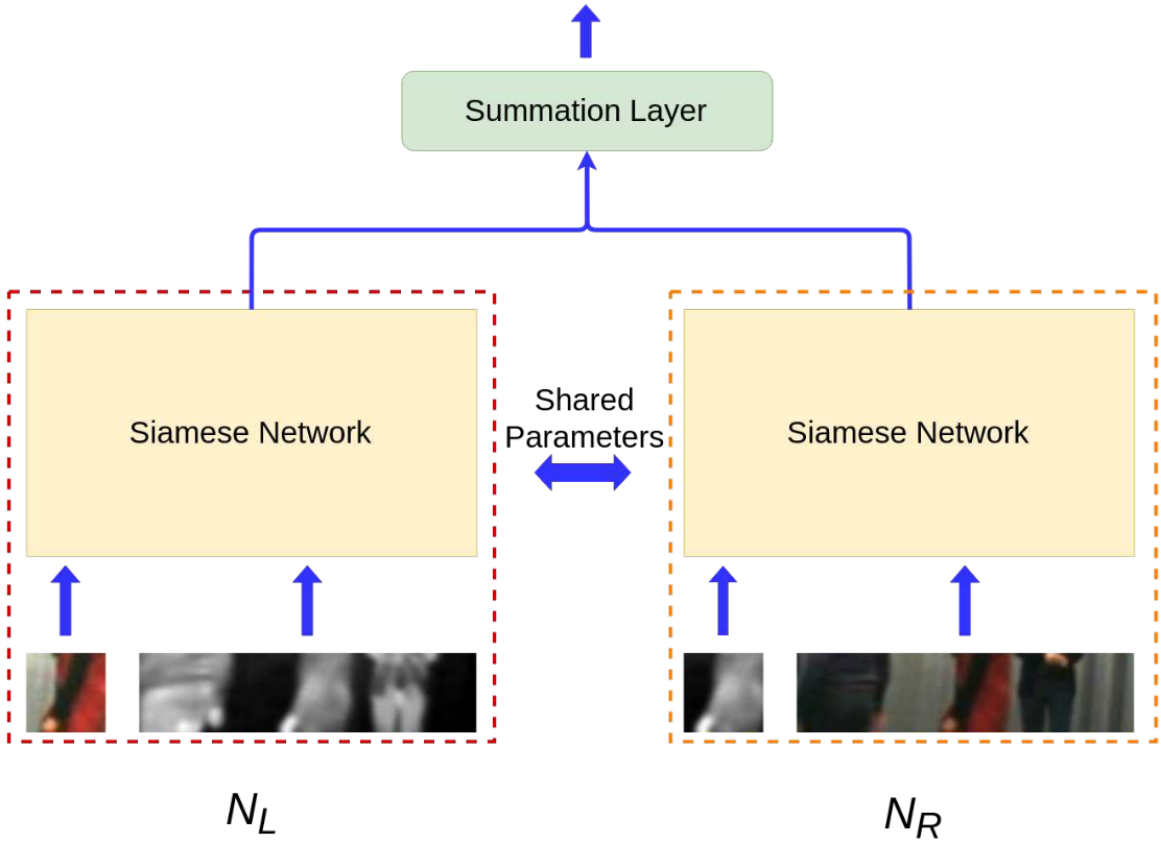


Figure 2.10 Architecture de Beaupre et al. [12] © 2019 IEEE.

soit l'équivalent d'un réseau siamois, mais où les poids ne sont pas partagés. L'architecture RCN classique concaténant les deux images d'entrée s'avère être la plus performante.

Un autre travail effectué par Aguilera et al. [44] introduit le réseau quadruplet. Ce réseau prend deux paires de sous-régions d'images correspondantes en entrée. Ceci aide car le réseau a deux paires d'exemples positifs et 4 pairs d'exemples négatifs. De façon similaire à la stéréoscopie RGB-LWIR, il n'y a pas beaucoup de jeu de données pour le spectre RGB-NIR.

Zhi et al. [45] ont créé un jeu de données pour contrer ce problème. Avec ce jeu de données, ils transforment une image RGB en une image NIR. Ils utilisent cette image nouvellement générée pour faire de l'auto-supervision.

2.3 Réseaux de neurones à auto-attention

Récemment, les RNCs sont de plus en plus remplacés par des réseaux de neurones à auto-attention, soit transformer en anglais. Ce type de réseau a été originalement utilisé dans

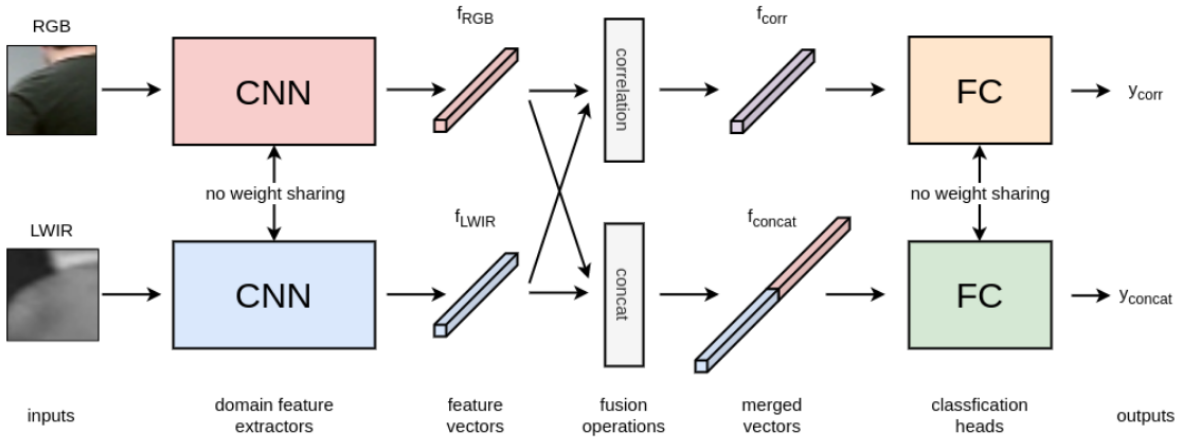


Figure 2.11 Architecture de Beaupre et al. [13] © 2020 IEEE.

le domaine du traitement du langage naturel. Vaswani et al. ont écrit "*Attention is all you need*" [46], soit le travail qui introduit les réseaux de neurones à auto-attention pour le traitement du langage naturel. Ils proposent un mécanisme d'attention pour deux tâches de traduction Anglais-Allemand et Anglais-Français. Les réseaux de neurones à auto-attention ont beaucoup évolué dans le domaine du traitement de langage naturel, et de là se sont inspirés plusieurs chercheurs pour appliquer ces modules d'attention au domaine de vision par ordinateur.

Les premiers à introduire les réseaux de neurones à auto-attention en vision par ordinateurs sont Chen et al. [47]. Ils ont réussi à reproduire les résultats d'un RNC pour un problème de classification. Le premier travail à directement utiliser des réseaux de neurones à auto-attention en vision par ordinateur est le réseau ViT par Dosovitskiy et al. [48]. ViT est un réseau pour la classification d'images. Lors de sa publication, ViT a battu l'état de l'art de façon significative. Depuis ce travail, plusieurs travaux ont émergé, et ce dans plusieurs branches de la vision par ordinateur. Des travaux ont émergé pour la détection d'objets [49,50] en appliquant les réseaux de neurones à auto-attention, qui à leur tour ont battu l'état de l'art des RNCs. D'autres travaux ont aussi été faits en segmentation sémantique [51].

Plusieurs articles ont utilisé des réseaux de neurones à auto-attention pour le calcul du flux optique. Tel est le cas de FlowFormer [52] et CRAFT de Sui et al. [53]. En ce qui a trait de la stéréoscopie directement, Li et al. [14] ont récemment publié un article mettant de l'avant un réseau qui utilise des réseaux de neurones à auto-attention pour remplacer les réseaux de stéréoscopie traditionnels. Leur méthode consiste à extraire les caractéristiques des images stéréoscopiques à l'aide d'un extracteur de caractéristiques sablier, et ensuite

passer ces vecteurs de caractéristiques dans un réseau de neurones à auto-attention, pour remplacer la seconde partie des réseaux traditionnels, soit la partie des opérations de fusions des vecteurs de caractéristiques. Le réseau de neurones à auto-attention remplace donc la concaténation et/ou la corrélation des travaux précédents.

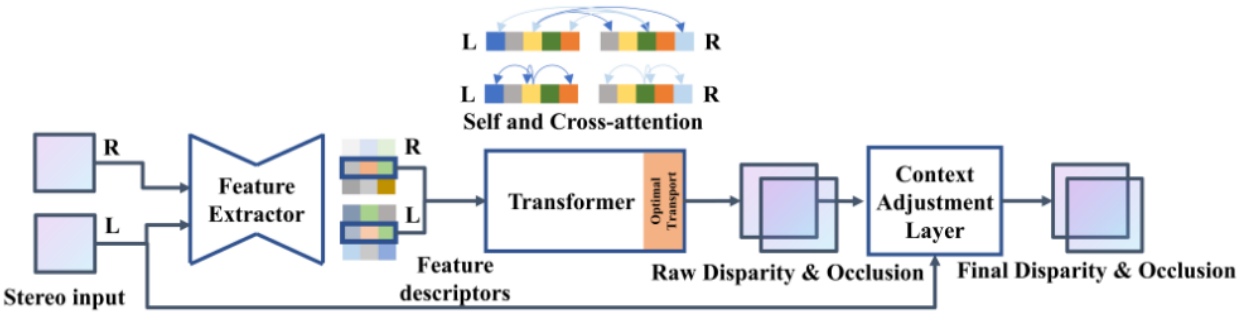


Figure 2.12 Architecture de Li et al. [14] © 2021 IEEE.

Comme présenté précédemment, Li et al. [14] utilisent les réseaux de neurones à auto-attention pour remplacer l'étape des opérations de fusion des vecteurs de caractéristiques. Cependant, certains travaux ont des extracteurs d'images utilisant des réseaux de neurones à auto-attention. Malgré que ce ne soit pas un travail qui fut implémenté pour de la stéréoscopie, Yuan et al. introduisent HRFormer [16] qui s'inspirent de l'architecture de l'extracteur de caractéristiques HRNet de Sun et al. [15].

Avant de parler davantage de HRFormer [16] il est bon de comprendre l'architecture de HRNet [15] présenté à la figure 2.13. HRNet est un extracteur de caractéristiques qui garde la résolution de l'image originale tout au long des étapes de convolutions, contrairement aux réseaux de convolution traditionnels. HRNet extrait les caractéristiques des images à plusieurs résolutions. Il sous-échantillonne d'un facteur de deux à chaque étage du réseau pour ensuite extraire les caractéristiques à la résolution donnée. À la fin de l'extracteur de caractéristiques, l'ensemble des volumes de caractéristiques générés sont redimensionné à la résolution originale, et forme un volume de caractéristiques plus profond contenant l'ensemble de toutes les caractéristiques extraites à toutes les dimensions.

Pour sa part, HRFormer [16] a la même architecture générale que HRNet. L'architecture de HRFormer peut donc être représentée à la figure 2.13. La différence est dans l'architecture des étages individuels. Dans chaque résolution de chaque étage se trouve le bloc illustré à la figure 2.14. Il n'est pas nécessaire d'expliquer l'architecture globale de HRFormer [16], puisqu'il s'agit du même principe que HRNet [15] qui a été expliqué précédemment.

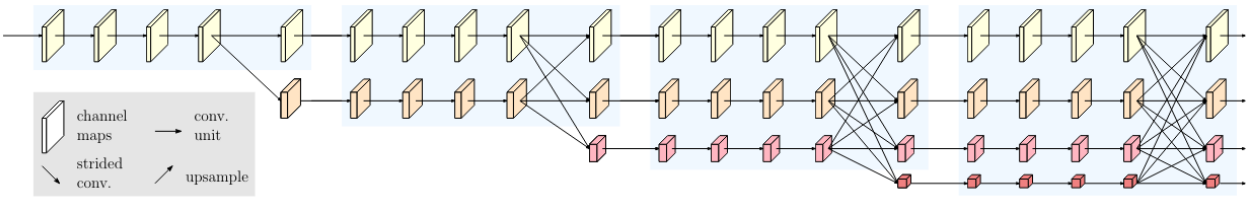


Figure 2.13 Architecture de Sun et al. [15] © 2019 IEEE.

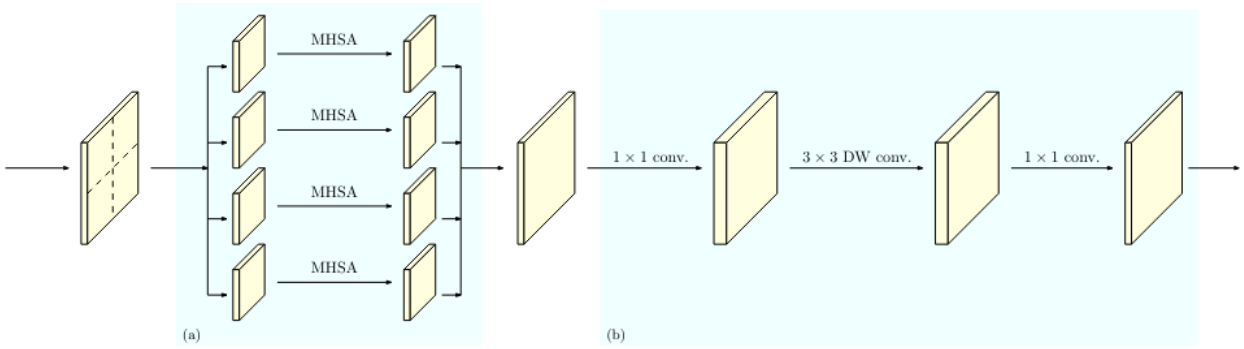


Figure 2.14 Bloc d'un stage dans HRFormer de Yuan et al. [16] © 2021 IEEE (a) représente un ensemble de couche parallèle d'auto attention multi-tête et (b) représente un **FFN**, soit un réseau de convolution transférant les informations vers l'avant.

Au lieu d'avoir uniquement des couches de convolutions, HRFormer [16] introduit des blocs d'auto-attention multi-tête (**MHSA**). Pour optimiser l'espace mémoire, ils séparent la carte de caractéristique d'entrée en plusieurs sous-régions distincte. Dans la couche d'entrée de la figure 2.14, on peut voir que la carte de caractéristiques est séparée en quatre sous-régions. Pour chacune des couches, une couche de neurone à auto-attention **MHSA** extrait des informations. Donc, dans le cas de la figure 2.14 (a) nous voyons que quatre couches **MHSA** sont en parallèle. Celles-ci extraient les caractéristiques de chaque sous-région de la carte de caractéristiques précédente de façon parallèle, sans partager d'informations mutuelles entre les couches de neurones à auto-attention **MHSA**. La fusion de la sortie de ces couches est faite à l'aide de convolution multidimensionnelle qu'on peut voir à la figure 2.14 (b).

CHAPITRE 3 MÉTHODOLOGIE

Le travail de maîtrise présenté se sépare en trois parties. Avant d'expliquer les modifications effectuées, nous allons expliquer en détail l'architecture sur laquelle nous avons basé notre travail. Ensuite, la première partie du travail consiste à ajouter des masques de segmentation à ce réseau pour voir son impact sur les résultats. Ensuite, nous avons modifié le réseau initial pour changer l'extracteur de caractéristiques, et l'adapter à notre problème. Enfin, nous avons modifié cet extracteur de caractéristiques pour améliorer les résultats. Cette section commencera par présenter l'addition des masques. Par la suite, les modifications à l'extracteur de caractéristiques seront présentées. Il sera mention des détails de développement et du cheminement de pensée qui a mené à l'avancement du projet.

3.1 Architecture de base du réseau

Dans cette section, nous allons présenter en détail l'architecture de Beaupré et al. [13]. Les travaux effectués dans ce mémoire ont tous émergé de cette architecture. Pour l'ensemble des modifications, seules les différences avec ce modèle sera présenté. Nous pouvons voir l'architecture de Beaupré et al. à la figure 2.11.

La méthode de Beaupré et al. prend en entrée une sous-région de couleur et une sous-région d'une image infrarouge thermique pour extraire leurs caractéristiques. Ces deux sous-régions sont de taille 36×36, et permettent d'avoir assez d'information à propos du contexte de chacun des points de disparité. Nous pouvons voir les entrées du réseau à gauche de la Figure 2.11.

Les détails sur l'entraînement et la phase de test vous seront présentés dans la section 4. Mais en quelques mots, le réseau peut apprendre en ayant les deux sous-régions d'images 36×36 correspondantes de chacun des spectres, et lors du test, il y a une sous-région 36×36 du spectre RGB qui essaye de trouver son emplacement dans une sous-région plus large LWIR de 36×100. La sous-région plus large permet de réduire le temps d'exécution. La sortir de l'extracteur de caractéristique est ensuite constitué d'un long vecteur de caractéristique. Une opération de fenêtre glissante est fait sur ce long vecteur pour obtenir le vecteur de caractéristique correspondant à une entrée 36×36.

Étant donné que les deux sous-régions sont des images provenant de deux spectres différents, chacun d'entre eux a un extracteur de caractéristiques différent. L'architecture de ces deux extracteurs de caractéristiques est identique, mais les deux extracteurs de caractéristique ne

se partagent pas les poids, ce qui donne de meilleurs résultats [13]. Le fait d'avoir un RNC avec ses propres poids selon le spectre aide à extraire les caractéristiques particulières de chaque spectre des images, et ainsi d'avoir des vecteurs de caractéristiques plus précis selon le spectre. Chaque extracteur de caractéristiques ont comme sortie un vecteur de caractéristiques de 256×1 représenté par F_{RGB} et F_{LWIR} dans la figure 2.11.

Avec les deux vecteurs de caractéristiques, nous appliquons ensuite deux opérations de fusions, soit une opération de concaténation et une de corrélation [37,54]. Les deux méthodes ont leurs avantages et leurs inconvénients. La corrélation est une opération rapide ayant une meilleure efficacité au niveau de l'espace mémoire, mais certaines caractéristiques sont perdues pour chaque spectre lors de l'opération. Pour l'opération de concaténation, aucun élément des vecteurs de caractéristiques n'est perdu. Cependant, cette opération est plus demandante au niveau du temps et de l'espace mémoire.

L'étape de corrélation a comme sortie un vecteur de taille 256×1 représenté par f_{corr} à la figure 2.11. L'opération de concaténation donne cependant un vecteur de sortie de 512×1 étant donné qu'il fusionne les deux vecteurs de caractéristiques. Le vecteur de résultant de la concaténation est représenté par f_{concat} à la figure 2.11.

Chacun des vecteurs de concaténation et de corrélation est l'entrée d'une couche complètement connectée. Chacune des opérations a leurs propres réseaux complètement connectés, et comme pour les RNCs, chacun a des poids séparés, étant donné que ce sont des opérations de fusions complètement distinctes. Chacune des branches a comme sortie un vecteur 2D qui représente un vecteur de classification pour les disparités possibles. Ils sont représentés par y_{corr} et y_{concat} sur la figure 2.11.

3.2 Addition du masque

La première étape du projet a été d'ajouter un masque de segmentation au réseau de Beaupré et al. [13]. Pour cette partie initiale du projet, nous avons écrit et publié un rapport technique sur arXiv [17]. L'architecture de ce réseau adapté est présentée à la figure 3.1. L'addition du masque est faite pour donner des informations sur les frontières des objets et bien repérer chaque silhouette humaine dans la scène. Nous avons remarqué que les méthodes de la littérature ont des difficultés à avoir une estimation de disparités précises aux frontières. Le défi était maintenant de savoir comment utiliser ce masque pour l'inclure dans un réseau de neurones. En d'autres mots, il s'agit de savoir comment générer et introduire les masques dans le réseau. Les deux sous-sections suivantes présenteront en premiers lieux des détails sur la génération des masques, et ensuite la méthodologie pour l'utilisation du masque.

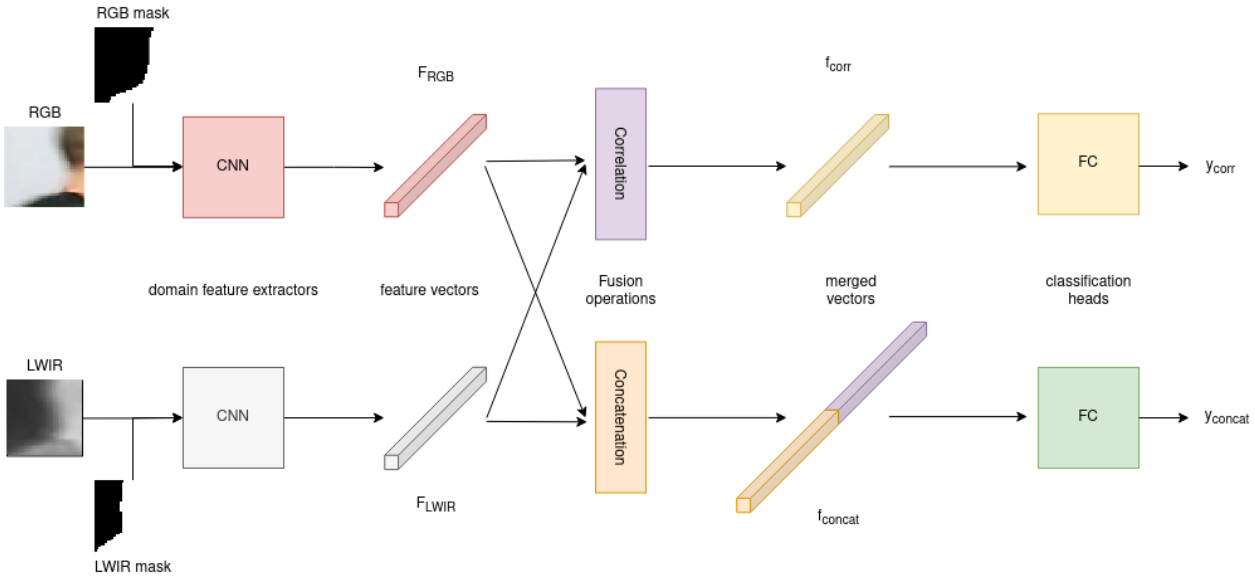


Figure 3.1 Méthode des masques [17]

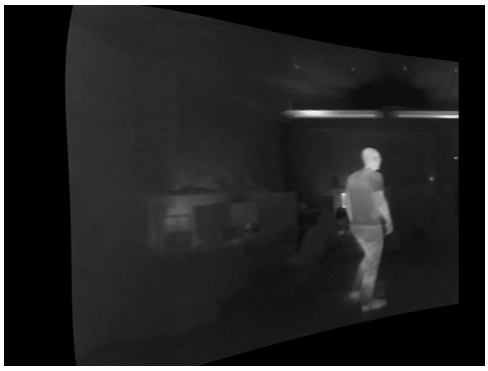
3.2.1 Detectron2 et Mask-RCNN

Pour générer les masques de segmentation, nous utilisons Mask R-CNN de Detectron2 [55] de Facebook AI sur chacune des images en couleurs et thermiques. Cette étape donne des informations reliées aux positions des humains dans la scène, et donne aussi de l'information sur les frontières des silhouettes où certaines disparités apparaissent. Notre hypothèse concernant les masques est que le réseau se concentra sur les silhouettes humaines si les masques sont présents.

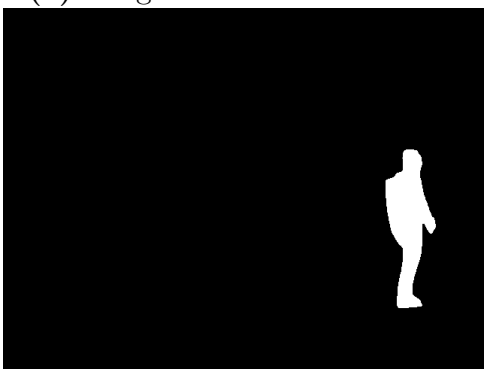
De façon plus détaillée, nous avons utilisé une version de Detectron2 qui inclut Mask R-CNN préentraîné sur le jeu de données COCO [56]. Ce jeu de données inclut la segmentation de 64 classes d'objets. Pour notre part, seulement la classe "personne" nous intéresse, dû à la nature des jeux de données que nous utilisons pour estimer la disparité. Cet algorithme a été utilisé sur les images thermiques de la même façon que sur les images de couleurs. À la figure 3.2f, nous pouvons voir que Detectron2 [55] donne des résultats acceptables pour une segmentation sur des images thermiques, malgré le fait que cet algorithme [55] n'a pas été entraîné sur des images de ce spectre.

L'algorithme de segmentation n'est pas parfait. En effet, la figure 3.3f nous permet de voir une segmentation imparfaite où le masque de deux instances est combiné. Une étude sera faite à la section 4 pour démontrer le bienfait des masques, en dégradant les masques, et en analysant l'impact sur les résultats. Certaines images ont des parties de la silhouette manquante. Tel

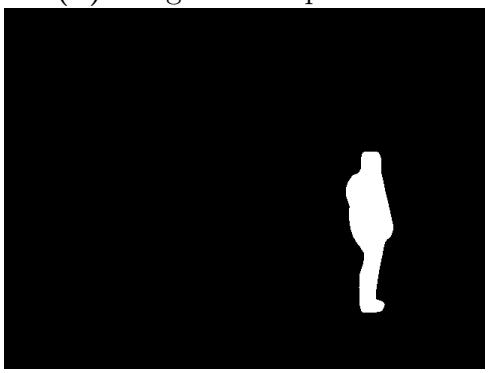
(a) Image couleur rectifiée



(b) Image thermique rectifiée



(c) Masque de l'image couleur



(d) Masque de l'image thermique

(e) Image couleur à 4 canaux

(f) Image thermique à 4 canaux

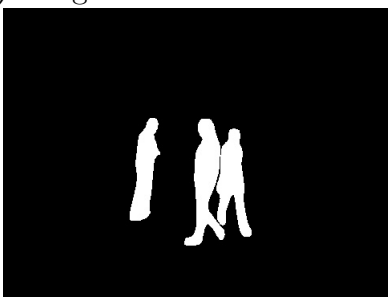
Figure 3.2 Processus d'images RGB et LWIR dans le jeu de donnée

est le cas de la figure 3.4, où nous pouvons voir une situation où une partie du pied est manquante. Le défaut de ce masque n'est pas significatif, car encore une fois, nous voulons un masque qui situe de façon approximative l'emplacement des silhouettes humaines dans la scène. Étant donné les imperfections du masque, nous fournissons l'ensemble des pixels au réseau, et non seulement les pixels correspondants au masque de segmentation. Nous allons voir l'effet de cette érosion du masque dans l'expérience à ce sujet.

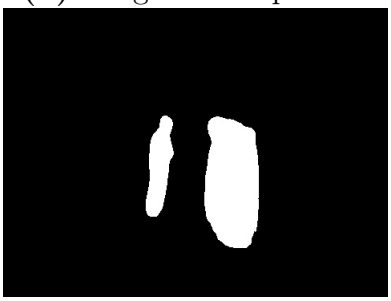
Nous observons également à la figure 3.3f que les deux personnes de droites sont très proches

(a) Image couleur rectifiée

(b) Image thermique rectifiée



(c) Masque de l'image couleur



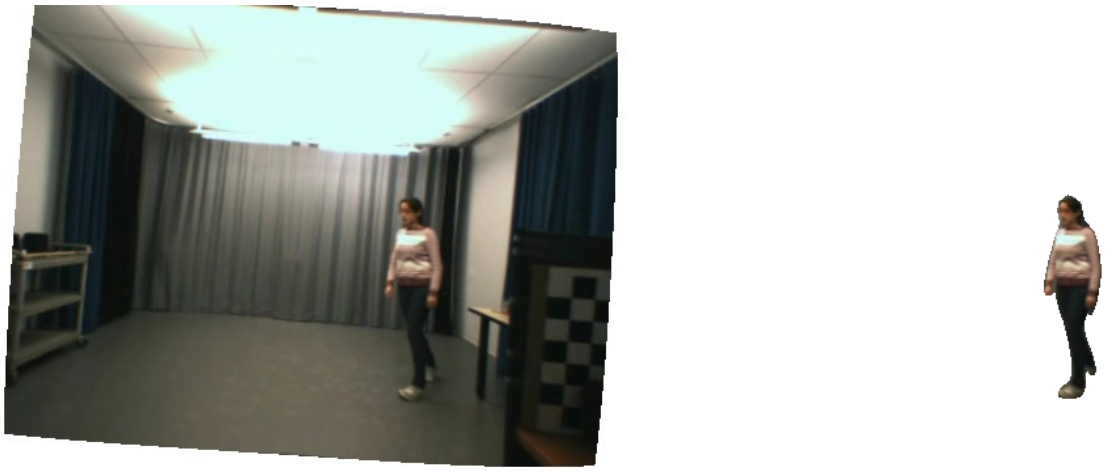
(d) Masque de l'image thermique

(e) Image couleur à 4 canaux

(f) Image thermique à 4 canaux

Figure 3.3 Précision des masques selon le spectre

l'une de l'autre. Ceci fait en sorte que les masques de segmentation sont fusionné pour l'image infrarouge thermique. Si nous comparons cette dernière avec l'image 3.3e nous pouvons voir que la segmentation est plus précise sur une image de couleur. Dans notre méthode, nous ne faisons aucune distinction entre les masques des instances. En effet, Detectron2 [55] retourne le masque de segmentation de chaque objet séparé. Donc si plusieurs personnes ont été détectées, nous concaténons l'ensemble des masques pour générer le masque global, donnant ainsi les masques finaux présentées aux figures 3.3c et 3.3d, par exemple.



(a) **(b)**

Figure 3.4 **(a)** Image RGB original. **(b)** Image RGB avec une erreur de segmentation.

3.2.2 Concaténation du masque

Cette section porte maintenant sur l'utilisation des masques dans notre réseau. Nous avons maintenant à notre disposition un répertoire de masques de chacune des images du jeu de données. Les images dans le répertoire sont donc le résultat du travail relié à la section 3.2.1.

Une fois les masques générés, nous avons réfléchi à trouver la meilleure méthode pour intégrer les masques au RNC. Une des pistes de solution était d'introduire les masques de segmentation au milieu du réseau, mais étant donné que la résolution du vecteur de caractéristique changeait, la meilleure option semblait de l'introduire au début. De ce fait, le masque serait plus longtemps dans le réseau, et le réseau choisira l'impact qu'il aura dans l'extraction des caractéristiques.

Pour introduire les masques au début du réseau, plusieurs options étaient sur la table. La première était d'avoir deux réseaux pseudo-siamois. Un des réseaux prend en entrée une paire d'images stéréoscopiques, et l'autre réseau prend en paramètre les masques. L'idée est ensuite de concaténer les deux vecteurs de caractéristiques résultants. Cependant, à cause du manque de détails dans les masques, la comparaison entre les deux masques serait difficile. En effet, il serait difficile d'estimer la disparité à un point en comparant uniquement les masques de segmentations. Nous avons donc jugé que ce ne serait pas la meilleure option.

Nous avons donc essayé de concaténer les masques aux images en entrée, et ainsi former des images de quatre canaux. Les images en entrée sont donc constituées d'un canal rouge, vert, bleu, et le quatrième canal est le masque binaire généré par Detectron2 [55], tel qu'expliqué à

la section 3.2.1. Ces images à 4 canaux vont ensuite passer dans notre réseau pseudo-siamois. Nous pouvons voir les images à l'entrée du réseau à la figure 3.1. Les images en couleurs et thermiques sont combinées avec les masques avant les extracteurs de caractéristiques. Comme mentionné à la section 3.2.1, les figures 3.2e et 3.2f sont un exemple d'une paire d'images stéréoscopiques à l'entrée de notre réseau. Les masques qui sont concaténés comme quatrièmes canaux des images sont constitués d'une valeur de 0 ou 255. La valeur 255 signifie que le pixel est associé à une instance d'un objet. L'entrée du réseau est donc constitué des images 3.2a et 3.2b qui sont concaténés à leurs masque respectifs 3.2c et 3.2d. Les images 3.2e et 3.2f sont une représentation simplifiée de l'entrée globale. La valeur réelle d'un pixel RGBM appartenant à une instance aux figures 3.2e et 3.2f est donc réellement représenté par $(R_{i,j}, G_{i,j}, B_{i,j}, 255)$. Les pixels n'appartenant pas à une instance sont représentés en blanc, mais leurs valeurs sont en réalité $(R_{i,j}, G_{i,j}, B_{i,j}, 0)$.

3.2.3 Modification à l'architecture du réseau

Nous avons utilisé la même architecture du réseau expliqué à la section 3.1, en ajoutant un quatrième canal en entrée pour les masques de segmentations. Nous pouvons voir à la figure 3.1 que les masques de segmentations sont ajoutés à l'entrée du réseau. Cette modification constitue la principale modification faite sur le réseau par rapport à celui de Beaupré et al. [13].

Nous avons dû augmenter le nombre de canaux à la première couche du réseau pour pouvoir entraîner le réseau avec le masque. Le reste de l'extracteur, ainsi que la tête du réseau n'ont pas été touchés. Nous pouvons voir les détails de la structure de l'architecture au tableau 3.1. La modification apporté à cette architecture est à la couche d'entrée du tableau 3.1. L'architecture du réseau expliqué à la section 3.1 est constitué d'une entrée de $36 \times 36 \times 3$.

3.3 Changement d'architecture du RNC

Une fois que nous avons pu générer des résultats avec les masques, nous avons étudié la modification d'une partie importante du réseau, soit l'extracteur de caractéristiques. Nous sommes donc repartis de l'architecture de base décrite à la section 3.1. Nous avons pris un extracteur déjà existant et avons remplacé la partie RNC du réseau de base et l'avons remplacé par un extracteur plus moderne, soit HRNet [15]. Nous pouvons voir l'architecture modifiée à la figure 3.5. Juste en remplaçant l'extracteur de caractéristique nous avons pu améliorer les résultats. Nous avons dû faire des modifications à l'entraînement pour que cette étape fonctionne, mais les détails seront présentés dans la section 4. Chaque extracteur de

Tableau 3.1 Architecture du modèle proposé. conv : couche convolutive, pc : couche pleinement connectée

Couche	Structure de la couche	Dimension de sortie
Entrée		36 x 36 x 4
RNC		
conv 1	5 x 5, 32	32 x 32 x 32
conv 2	5 x 5, 64	28 x 28 x 64
conv 3	5 x 5, 64	24 x 24 x 64
conv 4	5 x 5, 64	20 x 20 x 64
conv 5	5 x 5, 128	16 x 16 x 128
conv 6	5 x 5, 128	12 x 12 x 128
conv 7	5 x 5, 256	8 x 8 x 256
conv 8	5 x 5, 256	4 x 4 x 256
conv 9	4 x 4, 256	1 x 1 x 256
PC		
pc 1	256/512, 128	1 x 128
pc 2	128, 64	1 x 64
pc 3	64, 2	1 x 2

caractéristique est identique à l'architecture de la figure 3.6.

Le motif principal derrière le choix de cette architecture est de garder la résolution de sortie équivalente à la résolution d'entrée. Tel que mentionné au chapitre 2 dans la revue de littérature, le principe de HRNet est de faire une série de convolutions, et à la fin de chaque étage, diviser par deux la résolution. Après chaque étage, un vecteur de caractéristiques s'ajoute. La sortie du réseau est donc une concaténation de tous ces vecteurs de caractéristiques. L'ensemble des cartes de caractéristique sont redimensionnées pour correspondre à la taille d'entrée originale. Ce vecteur de caractéristiques final a donc un grand nombre de canaux. Le réseau HRNet [15] original effectue une série de convolutions sur ce vecteur de caractéristique finale pour réduire les dimensions de cette carte de caractéristiques. Cependant, notre but de l'introduction de cet extracteur de caractéristiques est d'avoir la meilleure résolution possible. Nous avons donc seulement redimensionné le nombre de canaux pour avoir comme sorti une carte de disparité de taille $36 \times 36 \times 64$. La dernière dimension de la carte de caractéristiques représente le nombre de disparités possible maximum du jeu de données.

3.4 Adaptation de l'architecture HRNet pour notre projet

Pour cette partie, nous ne sommes pas repartis de zéro, mais nous avons continué depuis les modifications de la section 3.3. L'architecture haut niveau de cette modification reste donc

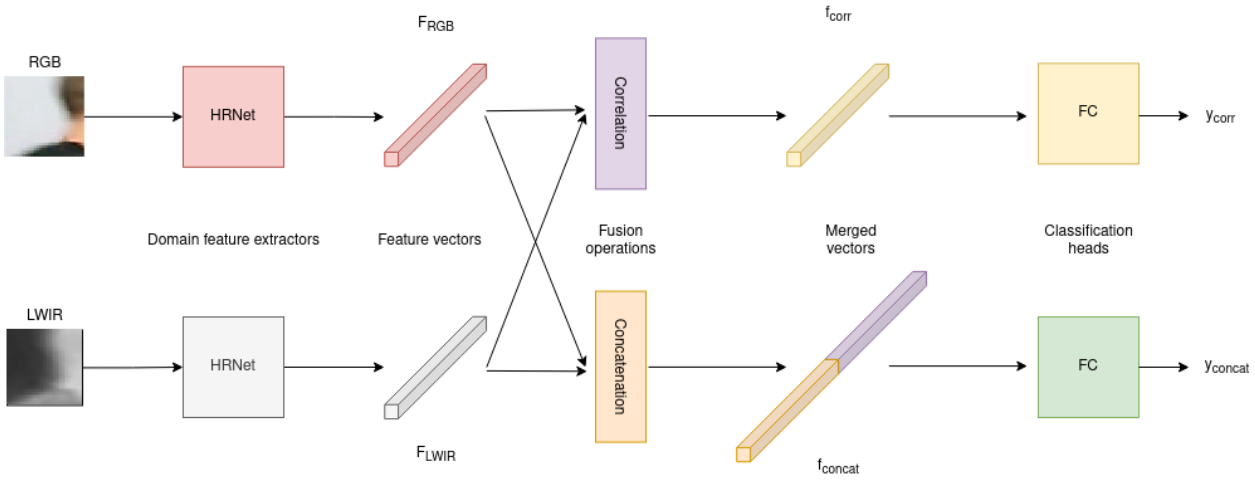


Figure 3.5 Architecture modifié avec l'extracteur de caractéristique HRNet [15]

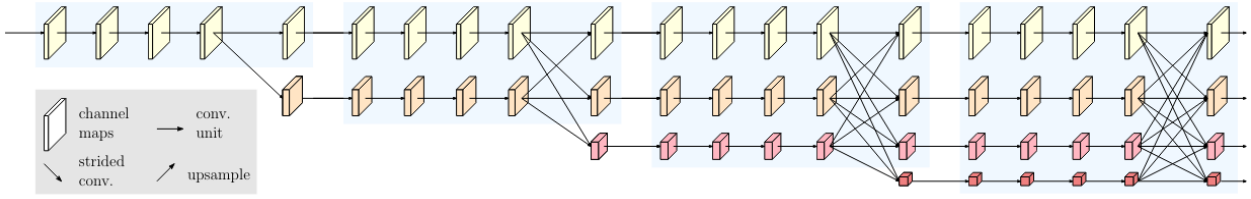


Figure 3.6 Architecture de HRNet © 2019 IEEE.

identique à la section précédente, soit la figure 3.5. Ce que nous avons changé dans cette section est la structure interne de HRNet, pour optimiser l'architecture à notre problème.

Dans l'article original, 4 étages sont utilisés. Un étage est borné par un encadré bleu pâle dans la figure 3.6. Le fait d'avoir 4 étages fait en sorte que la sortie finale est constituée de quatre cartes de caractéristiques avec 2 fois moins de caractéristiques l'une par rapport à l'autre. Dans leurs cas, ils utilisent des images de taille 512×1024, donc il est justifié d'avoir comme dernière carte de caractéristique un sous-échantillonnage d'un facteur de 8. Cependant, dans notre cas, les images d'entrée sont de taille 36×36. Donc réduire la taille avec un facteur 8 est non négligeable sur une sous-région de cette taille. Avec quatre étages, les résolutions sont donc de taille 36×36, 18×18, 9×9 et 5×5.

Ayant remarqué qu'il n'y avait pas une différence significative entre les deux dernières tailles de sous-régions, nous avons décidé de retirer le dernier étage de l'architecture originale. Ceci fait en sorte que nous avons maintenant trois cartes de caractéristiques en sortie, soit de taille 36×36, 18×18 et 9×9. Nous pouvons voir la disposition des étages d'un bloc HRNet à

la figure 3.7. Les blocs 36×36 sont jaunes, 18×18 sont oranges et 9×9 sont rouges. La sortie de notre carte de caractéristiques finale est donc la concaténation des trois dernières cartes de disparité des sous-échantillonnages, qui ont été ajustées pour correspondre à la plus grande résolution. Nous pouvons voir la sortie finale à gauche complètement de l'image 3.7.

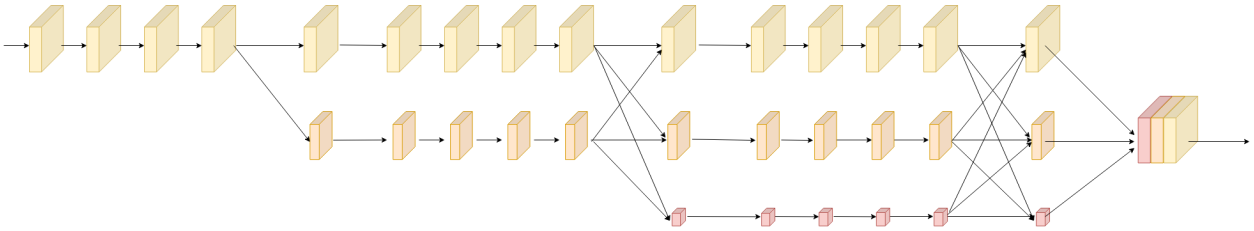


Figure 3.7 Architecture interne adapté de HRNet.

Ce nouvel extracteur de caractéristiques a significativement amélioré les résultats. Ceux-ci vous seront présentés dans la section 4.

Pour l'étape finale de l'intégration de HRNet dans notre réseau, nous avons modifié sa sortie. Encore une fois, cette modification affecte uniquement l'extracteur de caractéristiques, et fait suite à ce qui a été décrit précédemment. Donc comme mentionné plus haut, la sortie de HRNet est une concaténation des 3 sorties de la dernière étape. Cependant, on peut améliorer encore plus les résultats en concaténant la carte de caractéristique de haute résolution de la dernière étage avec celle de l'avant-dernière étage. De ce fait, nous gardons uniquement des caractéristiques de haut niveau. Nous pouvons voir les cartes de caractéristique sélectionnées pour la sortie à la figure 3.8. La sortie finale est une concaténation des cartes de disparités $x2$ et $x3$, représentant respectivement les cartes de disparités des étages 2 et 3. La différence entre la figure 3.7 et 3.8 est le choix de la sortie de HRNet.

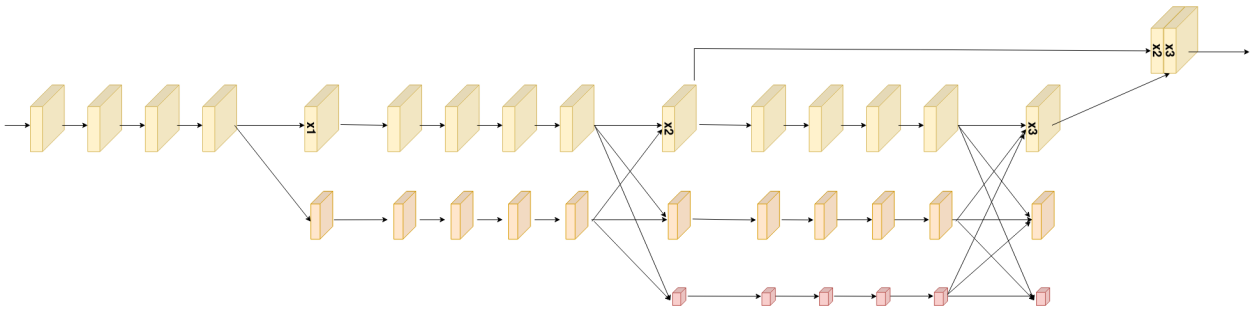


Figure 3.8 Architecture interne modifié de HRNet.

Le grand intérêt de cette méthode est de garder plus de caractéristiques de hautes résolutions. En effet, il est mieux de garder le plus de caractéristiques haute résolution pour perdre le

moins de détails possibles. En prenant des résolutions sous-dimensionnées, nous perdons des informations qui aident à estimer des disparités précises.

3.5 Intégration des masques avec les récentes modifications

Pour agréger l'ensemble de notre recherche, nous avons combiné nos modifications, soit les méthodes des sections 3.2 et 3.4. Donc, du modèle initial avec les masques de segmentation, nous avons modifié la partie expliquée à la section 3.4 pour ajuster le nombre de canaux en entrée. Cette modification est la seule modification effectuée sur HRNet pour introduire les masques. La première couche reste de la même dimension, mais nous ajoutons un canal supplémentaire pour accueillir le masque. Le reste des couches reste intact, ainsi que la sortie du réseau. Le réseau utilisé à cette étape est représenté à la figure 3.9, et l'intérieur du bloc HRNet dans cette figure est l'architecture de la figure 3.8.

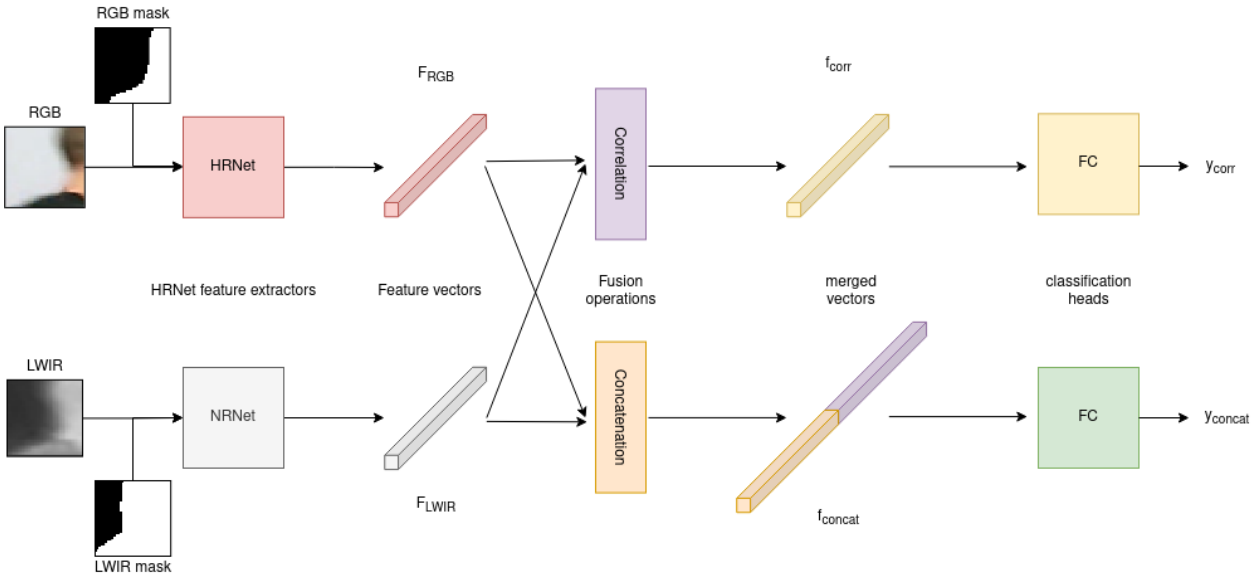


Figure 3.9 Architecture modifiée avec l'extracteur de caractéristique HRNet [15] et les masques de segmentation

CHAPITRE 4 EXPÉRIMENTATIONS

Cette section vous présentera l'ensemble des expérimentations effectué dans ce mémoire. Le code est disponible sur Github à l'adresse en note de bas de page [1]. Dans le fichier *README.md* les différentes versions de code associé aux expériences sont énumérés.

4.1 Jeux de données

Deux jeux de données sont utilisés dans ce projet, et sont utilisés conjointement afin d'entraîner et de tester les réseaux de neurones proposées. Les jeux de données utilisées sont LITIV2014 [18] et LITIV2018 [1]. Nos méthodes ont été testées et entraînées à l'aide de validation croisée en utilisant plusieurs blocs de données. Ceci permet donc d'alterner les données d'entraînement. Nous nous sommes servis des mêmes mélanges de blocs de données que Beaupré et al. [13]. Ces mélanges de blocs de données sont représentés au tableau 4.1 et les mélanges de blocs de données vous seront présenté en détail à la section 4.1.1.

Les deux jeux de données LITIV 2014 et LITIV 2018 sont constitués de plusieurs acteurs se déplaçant dans une scène. L'environnement de base reste similaire entre les vidéos, mais le nombre de personnes ainsi que leurs actions varient.

Nous avons évalué nos méthodes en utilisant la métrique de rappel, comme les méthodes précédentes. L'équation suivante représente notre mesure de performance utilisée pour l'ensemble de nos méthodes :

$$Rappel = \frac{1}{N}\sum_{i=1}^{N} |\hat{d}_i - gt_i| \leq n \tag{4.1}$$

Dans cette équation N représente le nombre de points à évaluer, $\hat{d}_i$ représente l'évaluation de la disparité à un point donnée, gt_i est la valeur réelle de la disparité à ce même point et enfin n représente la précision pour laquelle l'estimation sera considérée comme valide. Dans notre cas, n prends les valeurs de 1, 3 et 5 pixels.

4.1.1 Mélanges de blocs de données d'entraînement

Nous pouvons voir au tableau 4.1 les différents mélanges de blocs de données d'entraînement que nous nous sommes servis pour tester l'entièreté de nos méthodes. Le but de ces mélanges

1. https://github.com/philippeDG/stereoHRNet/

de blocs de données est d'évaluer à quel point nos méthodes peuvent généraliser à des données non observées. En effet, le fait d'avoir plusieurs mélanges de blocs de données permet d'entraîner sur plusieurs vidéos des jeux de données et ensuite de tester sur des vidéos différentes des jeux de données dans le cas où peu de données sont disponibles.

Dans le présent mémoire, il est mention de résultats sur LITIV 2014 et LITIV 2018, comme on peut le voir respectivement aux tableaux 4.5 et 4.6. Lorsque nous faisons mention de résultats pour un jeu de données, cela signifie que le mélange de blocs de données utilisé est testé sur ce jeu de données. Il y a six mélanges de blocs de données en tout, soit trois pour chaque jeu de données. Chaque mélange de blocs de données a une vidéo différente pour la phase de test, donc aucune séquence de ce vidéo a été utilisée lors de l'entraînement. Les trois premiers mélanges de blocs de données sont testés sur LITIV2014. Les mélanges de blocs de données de 1 à 3 au tableau 4.5 sont donc les mélanges de blocs de données 1 à 3 au tableau 4.1. Cependant, les mélanges de blocs de données 3 à 6 dans le tableau 4.1 représentent les mélanges de blocs 1 à 3 pour les résultats sur le jeu de donnés 2018, soit les résultats dans le tableau 4.4.

4.1.2 Prétraitement d'images

Pour le prétraitement d'images, nous avons utilisé la même méthodologie que Beaupré et al. [13]. Étant donnée que les jeux de données ne sont pas si grands, nous n'avons pas le choix de prétraiter et d'augmenter les données pour atteindre de bonnes performances. La première technique d'augmentation de données consiste à établir comme points de données, les points de données adjacents aux points annotés. Cette technique est appelée la duplication croisée [13]. Donc pour les pixels adjacents à un point de disparité donnée, soit tous les pixels à une distance de Manhattan de 1, nous considérons que tout ce groupe de pixels a la même disparité que le centre. Ceci augmente les données d'un facteur de 5. Nous pouvons voir

	Entraînement		Validation	Test
	LITIV 2018	LITIV 2014	LITIV 2014	LITIV 2014
M 1	218 230 (vid04 + vid07 + vid08)	240 167 (vid02 + vid03)	35 378 (vid02 + vid03)	101 433 (vid01)
M 2	218 230 (vid04 + vid07 + vid08)	291 720 (vid01 + vid03)	34 688 (vid01 + vid03)	76 001 (vid02)
M 3	218 230 (vid04 + vid07 + vid08)	320 648 (vid01 + vid02)	34 220 (vid01 + vid02)	61 771 (vid03)
	LITIV 2014	LITIV 2018	LITIV 2018	LITIV 2018
M 4	478 410 (vid01 + vid02 + vid03)	109 620 (vid07 + vid08)	44 226 (vid07 + vid08)	32 192 (vid04)
M 5	478 410 (vid01 + vid02 + vid03)	91 904 (vid04 + vid08)	49 286 (vid04 + vid08)	38 520 (vid07)
M 6	478 410 (vid01 + vid02 + vid03)	99 858 (vid04 + vid07)	41 566 (vid04 + vid07)	38 403 (vid08)

Tableau 4.1 Mélanges de blocs de données utilisés pour notre processus d'entraînement. M : mélanges de blocs de données

l'illustration de cette augmentation à la figure 4.1.

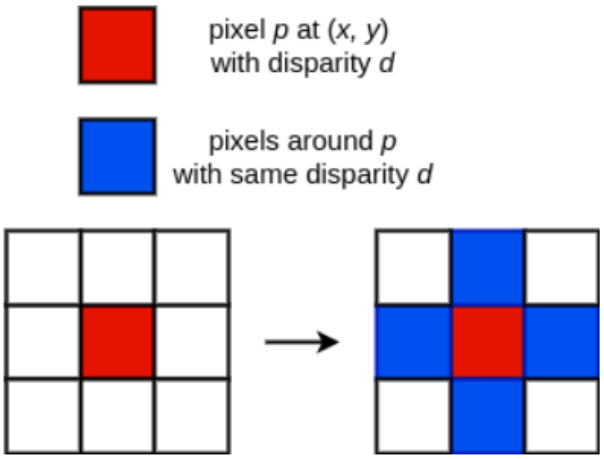


Figure 4.1 Augmentation de données de Beaupre et al. [13] © 2020 IEEE.

Une autre technique d'augmentation de données utilisée est d'ajouter aux données existantes le miroir sur l'axe des y de chaque image. Étant donné que chacune des images est doublée, il y a deux fois plus de données, et si nous combinons les deux méthodes d'augmentation de données, nous avons 10 fois plus de données.

Un dernier prétraitement que nous effectuons est pour la première méthode qui sera présentée. Pour cette méthode, il sera nécessaire de générer les masques dans chacune des images, et ceci va être fait grâce à Detectron2 [55]. Nous pourrons ensuite concaténer ces masques aux images initiales.

4.2 Entraînement

Pour l'entraînement, nous prenons des sous-régions de l'image autour des points de disparités dans chaque image rectifié de chacun des spectres. Nous appelons ces sous-régions P_{RGB} et P_{LWIR}. L'entraînement consiste uniquement à prendre ces deux sous-régions de 36×36 et de les passer dans le réseau pour la phase d'apprentissage. Le temps d'entraînement est d'un peu plus de 12h pour l'ensemble des données. Les expériences ont été fait sur une carte graphique RTX 2080 avec 8 Go de mémoire avec un processeur Intel i7-8700 et 16 Go de mémoire vive.

Comme dans l'article de Beaupré et al. [13], nous utilisons la fonction de perte d'entropie croisée pour les deux branches à la tête de notre réseau, soit la tête de corrélation et la tête

de concaténation. Les fonctions de coût pour chaque branche sont données par :

$$perte_{corr/concat} = -1/N \sum_{i=1}^{N} gt_i log(y_i), \tag{4.2}$$

où N représente le nombre de points de données, gt_i le point de vérité, qui peut avoir la valeur de 0 ou 1 dépendamment si les sous-régions dans le réseau sont correspondantes ou non. y_i est la probabilité de similarité entre les deux sous-régions, qui est représentée par y_{corr} ou y_{concat}.

La fonction de coût total est donnée par la somme des fonctions de perte des deux branches, soit :

$$perte_{totale} = perte_{corr} + perte_{concat}. \tag{4.3}$$

Nous entraînons notre réseau sur 200 époques, comme dans l'article de Beaupré et al. [13]. Pour les réseaux utilisant HRNet, nous entraînons aussi sur 200 époques malgré que ce modèle compte plus de paramètres, mais nous utilisons un réseau HRNet pré-entraîné pour initialiser les poids.

4.2.1 Tailles du lot d'entraînement

Pour la taille du lot d'entraînement pour notre première méthode, nous avons pris la même taille de lot d'entraînement que Beaupré et al. [13]. Ils utilisent une taille de lot de 64. Nous ne voulions pas modifier la taille de lot pour que nos résultats soient le résultat direct de l'addition des masques.

Pour l'extracteur de caractéristiques HRNet, nous avons dû changer la taille du lot d'entraînement, car nous avons eu des problèmes de mémoire. Nous avons donc décidé de trouver un nouveau lot d'entraînement en générant des résultats avec plusieurs tailles de lot différentes. Initialement la taille de lot était de 64. Pour rester dans les multiples de 8, nous avons essayé sur 16 et 24. Étant donnée que nous n'avions pas assez de mémoire pour une taille de lot de 32, nous avons essayé 25, qui était la taille maximale pour avoir assez de mémoire. Nous avons établi que la taille de lot pour HRNet allait être 24, étant donné que la précision était la meilleure des trois tests pour une erreur dans les trois pixels (voir tableau 4.2).

Tableau 4.2 Étude pour avoir la taille de lot optimal sur le premier mélange de blocs avec HRNet.

	Taille de lot de 16	Taille de lot de 24	Taille de lot de 25
n1	75,61	75,98	71.84
n3	98,29	97,12	94.81
n5	99,99	99,89	99.59

4.2.2 Pas du gradient

Pour la valeur du pas du gradient, nous avons opté pour la même mentalité. Nous voulions garder le plus possible les paramètres proches de ceux de Beaupré et al. [13]. Dans leur article, ils ont utilisé un pas de gradient de 0.01 pour l'ensemble des entraînements.

Lors de l'entraînement du réseau avec HRNet, nous nous sommes aperçus que le réseau n'arrivait pas à apprendre suffisamment. Nous pouvions voir lorsque nous entraînions HRNet avec le pas de gradient à $0,01$ que la métrique de rappel stagnait dans les alentours de $0,5$. Cependant, avec le pas de gradient ajusté, nous avons pu obtenir une courbe d'apprentissage satisfaisante, que nous pouvons voir à la figure 4.2. La valeur de pas de gradient utilisée pour la génération de ce graphique, et celle que nous avons utilisée pour entraîner le modèle est de $0,001$.

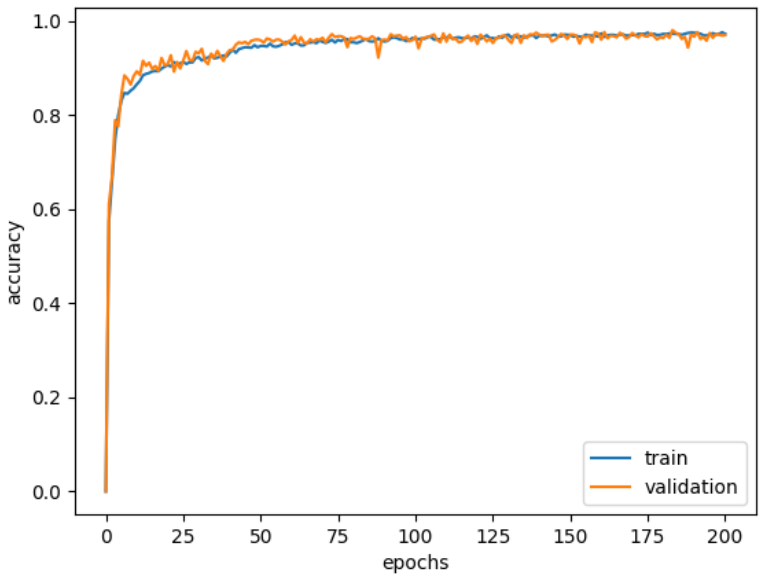


Figure 4.2 Métrique de rappel selon l'époque d'entraînement pour le réseau HRNet modifié avec le pas du gradient ajusté.

4.3 Estimation de disparité / Phase de test

Pour estimer la disparité, nous devons tout d'abord établir la disparité maximale, donnée par d_{max}. Ensuite, nous ajoutons la moitié de cette distance de part et d'autre de la sous-région P_{LWIR}, pour former une sous-région plus large, de hauteur identique de 36 et de largeur $36 + d_{max}$. La largeur est de la sorte pour pouvoir effectuer d_{max} translations d'une sous-région 36×36. La sous-région RGB reste de la même taille.

Après avoir passé ces deux sous-régions dans les extracteurs de caractéristiques, F_{RGB} sera un vecteur de caractéristiques de 256 et F_{LWIR} sera une carte de caractéristiques de $256 \times d_{max}$. Nous passons ensuite chacun des vecteurs de caractéristiques de la carte de caractéristiques LWIR dans les deux branches d'opération de fusion (la tête de notre réseau) avec le vecteur de caractéristiques F_{RGB}. La sortie des opérations de fusions sont f_{corr} et f_{concat}.

Chacun des vecteurs f_{corr} et f_{concat} sont l'entrée d'un réseau complètement connectés. La sortie de ces réseaux complètement connectée correspond à y_{corr} and y_{concat}. Ces vecteurs de sortie sont chacun de taille d_{max}, et chacune des valeurs dans ce vecteur correspond à la probabilité que P_{RGB} correspond à chaque sous-région 36×36 de P_{LWIR}. Donc pour chaque translation d'une sous-région 36×36 dans la sous-région de P_{LWIR} de largeur $36 + d_{max}$, nous avons la probabilité que P_{RGB} corresponde. La disparité de chaque branche est donnée par l'index de la probabilité la plus élevée dans chaque élément du vecteur de sortie correspondant.

$$\hat{d}_{corr/concat} = argmax(d), \tag{4.4}$$

Où d est le vecteur de disparité en sortie, de taille d_{max} de chacune des branches (y_{corr} et y_{concat}). La disparité finale est une moyenne des meilleures disparités de chaque branche $\hat{d}_{corr}$ et $\hat{d}_{concat}$. La disparité finale est donnée par :

$$\hat{d} = \frac{\hat{d}_{corr} + \hat{d}_{concat}}{2}. \tag{4.5}$$

4.4 Addition du masque

Le tableau 4.3 nous donne les résultats de la méthode proposée, soit méthode des masques, comparés à plusieurs méthode de l'état de l'art sur le jeu de données LITIV 2014. Ce tableau est obtenu en calculant la moyenne de chaque mélange de blocs de données. Premièrement, nous pouvons noter que nos résultats sont légèrement inférieurs à ceux de Beaupré et al. [13] pour la précisions $n = 3$, mais supérieurs pour les précisions de $n = 1$ et $n = 5$. À partir du tableau 4.3, nous pouvons voir que notre méthode à des résultats similaires pour les trois

Tableau 4.3 Résultats sur le jeu de données LITIV 2014 comparés aux méthodes de l'état de l'art. Les résultats sont la moyenne des trois vidéos. **Caractères gras : Meilleurs résultats**.

Méthodes	*erreur* ≤ 1 *pixel*	*erreur* ≤ 3 *pixel*	*erreur* ≤ 5 *pixel*
Méthode des masques (la nôtre)	**57.52 ± 2.32**	88.74 ± 0.99	**98,55 ± 0.39**
Domain Siamese CNN [13]	56.25 ± 3.47	**89.95 ± 0.39**	98.53 ± 0.43
CNN Siamois [12]	-	77.9 ± 5.04	-
St-Charles et al. [1]	48.2 ± 3.95	-	-
Information mutuelle [18] (40×130)	-	83.3	-
Information mutuelle [18] (20×130)	-	77.5	-
Information mutuelle [18] (10×130)	-	64.9	-
Fast Retina Keypoint [18](40×130)	-	64.1	-
Local Self-Similarity [1, 18](40×130)	22.6 ± 10.66	73.4	-
Différence des carrées [18](40×130)	-	65.6	-

Tableau 4.4 Résultats sur le jeu de données LITIV 2018 comparés aux méthodes de l'état de l'art. Les résultats sont la moyenne des trois vidéos. **Caractères gras : Meilleurs résultats**.

Méthodes	*erreur* ≤ 1 *pixel*
Méthode des masques	**60.45 ± 4.38**
Domain Siamese CNN [13]	44.40 ± 3.70
St-Charles et al. [1]	42.23 ± 16.27

précisions. Pour la précision $n = 3$, nous voyons que la méthode *Domain Siamese CNN* est légèrement avantageuse, mais pour les précisions d'*erreur de pixel* ≤ 1 ($n = 1$) et d'*erreur de pixel* ≤ 5 ($n = 5$) l'évaluation du rappel s'améliore légèrement avec notre méthode. Ceci montre donc un léger avantage d'utiliser les masques.

Le tableau 4.4 nous montre les résultats sur le jeu de donnes LITIV 2018 [1]. Ce tableau a aussi été obtenu en faisant la moyenne des trois premiers mélanges de blocs de données. Étant donné qu'aucune autre méthode n'a été testée sur des *erreurs de pixels* ≤ 3 ($n = 3$) ou ≤ 5 ($n = 5$) sur le jeu de donnée LITIV 2018 [1], nous montrons uniquement l'*erreur de pixel* ≤ 1 ($n = 1$). Nous pouvons voir une amélioration significative sur la métrique de rappel. L'écart type indique que chacun des mélanges de blocs de données traité avec notre méthode améliore les résulats par rapport aux travaux précédents.

Finalement, si nous détaillons les résultats de chaque mélange de blocs de données des deux jeux de données, nous pouvons voir dans le tableau 4.5 que notre méthode s'améliore par rapport à la méthode *Domain Siamese CNN* [13] qui utilise une architecture similaire à la nôtre, à l'exception des masques. Ceci peut être affirmé grâce aux résultats du tableau 4.6. Nous pouvons voir que notre méthode améliore les résultats pour les mélanges de blocs de données du jeu de données LITIV 2018 [1].

Tableau 4.5 Résultats détaillés sur le jeu de données LITIV 2014 pour une erreur de moins de 1 pixel. M : Mélanges de blocs

	Beaupré et al. [13]	Méthode des masques
M 1	57.38	**57.91**
M 2	52.35	**55.02**
M 3	59.02	**59.63**

Tableau 4.6 Résultats détaillés sur le jeu de données LITIV 2018 pour une erreur de moins de 1 pixel. M : Mélanges de blocs

	Beaupré et al. [13]	Méthode des masques
M 1	48.0	**64.23**
M 2	44.6	**55.64**
M 3	40.6	**61.49**

Les résultats comparatifs entre notre méthode et celle de Beaupre et al. [13] montrent l'efficacité du masque. Les tableaux 4.3 et 4.4 montrent bien l'impact positif général des masques sur l'ensemble des mélanges de blocs étant donné que l'information des frontières est ajoutée.

4.4.1 Étude d'ablation

Le tableau 4.7 montre l'étude d'ablation qui a été faite sur la branche de corrélation et de concaténation. Pour cette étude d'ablation, nous avons modifié la tête du réseau, pour en isoler une des deux opérations de fusion chacun son tour. Cette expérience a pour but de valider la présence des deux branches à la tête du réseau lorsque nous ajoutons les masques. Donc en résumé, chacun des extracteurs de caractéristiques génère les vecteurs de caractéristiques pour chacune des images des deux spectres. Ces vecteurs sont ensuite concaténés pour une expérience, et corrélés pour l'autre. Pour le premier et dernier mélange de blocs, nous pouvons voir que la combinaison des deux branches donne de meilleurs résultats que chacune des deux branches séparées. Cependant, pour le deuxième mélange de blocs, la branche de corrélation donne de meilleurs résultats que la branche de concaténation et que la combinaison des deux branches. Avec ces deux constats, nous pouvons conclure qu'il est idéal de conserver les deux branches de la tête du réseau, car pour la plus grande partie des mélange de blocs et des précisions, la combinaison des blocs améliore les résultats.

Tableau 4.7 Étude d'ablation pour l'ajout des masques sur le LITIV2018. M : Mélanges de blocs

	Branche de corrélation			Branche de concaténation			Méthode des masques		
	n1	n3	n5	n1	n3	n5	n1	n3	n5
M 1	53.33	86.72	97.70	45.95	82.31	**99.14**	**57.91**	**89.51**	98.31
M 2	**62.47**	**93.05**	**99.46**	49.42	83.10	96.93	55.02	87.61	98.35
M 3	58.31	86.68	97.77	56.82	84.76	98.00	**59.63**	**89.1**	**99.0**

4.4.2 Étude de dilatation/érosion des masques

Pour étudier l'efficacité directe des masques, nous avons fait une étude qui dilate ou érode les masques générés par Detectron2 [55] pour voir l'efficacité du masque selon sa précision. Nous avons effectué les tests avec des tailles de lot plus petit pour diminuer le temps d'exécution. Les résultats de l'étude sont montrés au tableau 4.8. À la figure 4.3 nous apercevons le résultat d'une image dilatée et à la figure 4.4 nous apercevons des images ayant subit de l'érosion. Nous pouvons voir que les masques semblent être plus efficaces avec une dilatation de 5 pixels. Ces résultats ne sont que sur le premier mélange de blocs, donc nous ne pouvons pas conclure qu'une dilatation de 5 permet d'avoir de meilleur résultats. Cependant, nous pouvons bien voir que la qualité des masques affectent la précision des résultats, mais le masque ne reste quand même pas une grande influence sur la métrique de rappel, car les résultats ne varient pas beaucoup selon les différentes dilatations/érosions.

(a) **(b)**

Figure 4.3 **(a)** Image RGB à 4 canaux dilatée de 5 pixels **(b)** Image RGB à 4 canaux dilatée de 15 pixels.

(a) **(b)**

Figure 4.4 **(a)** Image RGB à 4 canaux avec érosion de 5 pixels **(b)** Image RGB à 4 canaux avec érosion de 15 pixels.

Niveau de dilatation (px)	n1	n3	n5
-15	56,10	87,61	97,93
-10	58,13	91,24	99,15
-5	58,00	90,25	98,48
0	58,52	90,53	98,69
5	60,32	91,35	99,23
10	57,15	91,16	98,76
15	56,20	88,64	98,18

Tableau 4.8 Métrique de rappel pour les précisions n1, n3 et n5 selon le niveau de dilatation du masque avec le réseau de RNC de base. Les dilatations négatives représentent une érosion des masques.

4.5 Adaptation de l'architecture HRNet pour notre projet

Le changement d'architecture vers HRNet a donné de très bonnes améliorations sur l'ensemble des mélanges de blocs de données. Au tableau 4.9 nous pouvons voir que la moyenne des trois premiers mélanges de blocs pour la précision en dessous d'un pixel est de **75,00**, et dans la méthode avec la concaténation des masques la précision en dessous d'un pixel est de **57,91**.

Pour la précision de trois pixels et moins, nous pouvons voir au tableau 4.9 que la précision était de **88,74**, et que l'état de l'art était à **89,95**. Cependant, avec l'addition de HRNet, nous avons obtenu une précision de **96,24**, et ce avec un écart type de $0,4$.

Pour la précision de moins de 5 pixels, il y a eu une bonne amélioration malgré le fait que pour la méthode du masque, la valeur de rappel était de **98,55** pour les résultats sur le jeu

Tableau 4.9 Comparaison de l'adaptation de HRNet avec la méthode des masques.

		Méthode des masques	Adaptation de HRNet
LITIV 2014	n1	57,52 ± 2,32	**75,00 ± 0,66**
	n3	88,74 ± 0,99	**96,24 ± 0,40**
	n5	98,55 ± 0,38	**99,61 ± 0,23**
LITIV 2018	n1	60,45 ± 4,38	**63,34 ± 7,00**
	n3	87,4 ± 2,00	**92,55 ± 2,26**
	n5	98,72 ± 0,10	**99,69 ± 0,21**

de données LITIV 2014 [18], comme nous pouvons le voir dans le tableau 4.9. Avec notre méthode, nous avons été capables d'obtenir une précision de **99,61**, ce qui dépasse l'état de l'art de près de 1%.

Pour les résultats sur LITIV 2018, nous pouvons voir dans le tableau 4.9 que la moyenne des mélanges de blocs est de $60,45$ pour la méthode des masque à la précision de un pixel et moins. Avec HRNet la précision est de $63,34$. Ce qui donne donc une augmentation de précision sur la précision de un pixel et moins. Pour la précision sous 3 pixels, la précision passe de $87,40$ à $92,55$. Ce qui rend encore une fois la méthode de HRNet plus avantageuse. Pour finir, HRNet améliore également les résultats pour la précision de 5 pixels et moins, passant de $98,72$ à $99,69$. En comparant les résultats et les écarts types, nous pouvons voir que HRNet améliore significativement les résultats comparé à la méthode des masques.

4.5.1 Étude d'ablation

De façon similaire à la section précédente, nous avons généré une étude d'ablation contenant les résultats en séparant les branches de concaténation et de corrélation. Nous pouvons voir cette étude au tableau 4.11. Si nous comparons les résultats des trois premiers mélanges de blocs dans le tableau 4.11 nous pouvons voir que la combinaison des deux branches, soit les données de la colonne de droite, donnent de meilleurs résultats que les résultats de la branche

Tableau 4.10 Résultats pour l'adaptation de HRNet dans notre réseau. M : Mélanges de blocs

	M 1	M 2	M 3	M 4	M 5	M 6	Moyenne 3 premiers M	Moyenne 3 derniers M
n1	75,61	75,08	74,3	68,64	55,4	65,98	75,00 ± 0,66	63,34 ± 7,00
n3	95,97	96,7	96,06	94,65	90,16	92,83	96,24 ± 0,40	92,55 ± 2,26
n5	99,8	99,67	99,36	99,92	99,52	99,64	99,61 ± 0,23	99,69 ± 0,21

de concaténation ou de corrélation pour la plupart des mélanges de blocs. En effet, pour le mélange 3, la branche de concaténation donne de meilleur résultats.

Cependant, pour les trois derniers mélanges de blocs, nous pouvons voir que la corrélation est meilleure que la combinaison des deux branches pour un mélange de blocs. En effet, pour le deuxième bloc, la branche de corrélation donne une meilleur métrique de rappel que la combinaison des deux branches, et ce sur les trois précisions. Pour le dernier mélange de blocs, la branche de corrélation donne une meilleur métrique uniquement sur la précision de moins de trois pixels. Pour les deux autres précisions, la combinaison des deux branches reste meilleure. La combinaison des deux branches reste justifiable étant donné que pour les trois premiers mélanges de blocs les résultats sont significativement meilleurs, et que les résultats que nous venons juste de comparer, et les écarts types sont plutôt grands, ce qui fait en sorte que nous pouvons affirmer que les résultats sont similaires.

Tableau 4.11 Étude d'ablation pour HRNet adapté. M : Mélanges de blocs

	Branche de corrélation			Branche de concaténation			Adaptation de HRNet		
	n1	n3	n5	n1	n3	n5	n1	n3	n5
M 1	73,44	**96,22**	99,76	69,22	93,32	99.11	**75,61**	95,97	**99,80**
M 2	68,23	94,89	99.37	64,88	95,45	99,41	**75,08**	**96,06**	**99,67**
M 3	77,15	95,87	99,66	**80,93**	**97,68**	**99,86**	74,30	96,06	99,36
M 4	68,39	92,79	99.62	60,70	90,72	99,50	**68,64**	**94,65**	**99,92**
M 5	**59,33**	**92,21**	**99.71**	56,07	87,14	97,73	55.40	90,16	99,52
M 6	64,14	**93,06**	99.55	59,95	87,20	97,80	**65.98**	92,83	**99,64**

4.6 Modification de HRNet pour l'amélioration des résultats

En concaténant les deux dernières couches de pleine résolution, nous avons pu obtenir encore de meilleurs résultats (tableau 4.12). En effet, si nous comparons les résultats sur le LITIV2014 de cette modification avec celle de la sortie normale de HRNet, nous pouvons voir que le résultat passe de $96,24$ pour la précision en bas de 3 pixels à $96,94$.

Pour la précision de moins d'un pixel, la précision passe de $75,00$ à $74,14$. Nous pouvons voir une légère baisse, que nous pouvons considérer similaire dû aux écarts types qui se recoupent.

Pour la précision sous 5 pixels, les résultats passent de 99,61 à 99,87, ce qui est une bonne amélioration considérant que le résultat est proche de 100.

Pour les résultats sur le LITIV2018, nous pouvons voir que l'ensemble des résultats au tableau 4.12 sont tous supérieurs aux à ceux des méthodes précédentes.

Tableau 4.12 Comparaison entre la modification de HRNet et les méthodes précédentes.

		Méthode des masques	Adaptation de HRNet	Modification de HRNet
LITIV 2014	n1	57,52 ± 2,32	**75,00 ± 0,66**	74,14 ± 1, 21
	n3	88,74 ± 0,99	96,24 ± 0,40	**96,94 ± 0, 56**
	n5	98,55 ± 0,38	99,61 ± 0,23	**99,87 ± 0,04**
LITIV 2018	n1	60,45 ± 4,38	63,34 ± 7,00	**63,55 ± 5,37**
	n3	87,4 ± 2,00	92,55 ± 2,26	**94,83 ± 2,64**
	n5	98,72 ± 0,10	99,69 ± 0,21	**99,90 ± 0,10**

4.6.1 Étude d'ablation

Dans ce cas-ci, nous n'avons pas fait d'étude d'ablation complète étant donné que les couches de concaténation et de corrélation se sont prouvées à être efficaces grâce à l'étape précédente, soit à la section 4.5.1. Cependant, nous avons testé la précision de la couche de haute résolution selon chaque étages dans HRNet. Nous pouvons voir les résultats au tableau 4.13. Dans ce tableau, $x1$ représente la couche pleine résolution de sortie du premier étage. $x2$ représente celle du deuxième étage, et $x3$ représente la couche de pleine résolution du dernier étage. Nous pouvons voir que les meilleurs résultats sont partagés entre $x2$ et la dernière couche. En effet, pour la précision en dessous de un, le dernier étage a de meilleur performance, mais les précisions en dessous de trois et cinq, l'étage $x2$ est meilleur. Ceci a donc justifié notre choix de modifier la sortie pour concaténer les couches $x2$ et $x3$.

Tableau 4.13 Résultats sur le premier mélange de blocs, selon la profondeur de la couche haute résolution.

	x1	x2	x3
n1	73,04	76,66	**76,83**
n3	95,58	**96,64**	96,35
n5	99,26	**99,84**	99,76

4.7 Intégration des masques avec les récentes modifications

Le tableau 4.15 montre les résultats sur le jeu de donnée 2014 pour l'ensemble des méthodes. Les résultats en gras montrent les résultats de la meilleure méthode selon la précision. La dernière section qui consistait à introduire les masques dans l'architecture HRNet. Le tableau 4.16 montre les résultats obtenus sur les différents mélanges de blocs. Si nous comparons les

résultats avec le tableau de résultats de l'étape précédente, soit le tableau 4.14, nous pouvons voir que les résultats ne se sont pas améliorés. L'extracteur de haute résolution HRNet ne semble pas avoir besoin de masques pour bien performer. Les résultats de cette étude restent cependant meilleurs que le premier réseau d'addition de masque si nous nous référons au tableau 4.15. Ceci montre encore une fois que HRNet est meilleur que le RNC initial.

4.7.1 Étude de dilatation des masques

Suivant la même logique que l'étude de dilatation précédente à la section 4.4.2, nous avons dilaté les masques pour voir si les masques ont un effet positif sur le réseau. Nous pouvons voir que dans le tableau 4.17 que plus le masque est dilaté, plus la métrique de rappel est élevée. Donc si on suit cette logique, un masque dilaté au maximum serait le meilleur résultat. Le masque n'est donc pas utile car un masque dilaté au maximum reviens à avoir un masque complètement associé à une sous-région, ce qui reviens à ne pas avoir de masque.

4.8 Discussion générale

Les dernières sections ont pu montrer les résultats des expériences effectuées. Nous allons maintenant discuter de la pertinence des méthodes, ainsi qu'expliquer les résultats obtenus. Nous allons donc énumérer les architectures que nous avons testées, pour ensuite proposer une architecture finale.

Le choix de nos méthodes sera fait à l'aide des résultats obtenus dans le tableau 4.15. Ce tableau montre les résultats qui ont été testés avec le jeu de données LITIV 2014. Il est bon de se rappeler que pour l'ensemble des mélanges de blocs utilisés pour générer les résultats de ce tableau, LITIV 2018 a aussi été utilisé pour entraîner les modèles. Les résultats du jeu de données LITIV 2018 seront également analysés et vous seront présentés au tableau 4.18. Nous vous présenterons dans la présente section, chaque méthode que nous avons utilisée, et ferons le compte rendu de leurs efficacités. Pour finir, nous allons proposer une méthode finale qui nous semble être la meilleur selon les résultats obtenus.

Tableau 4.14 Résultats pour HRNet avec les sorties concaténées du dernier et de l'avant-dernier stage. M : Mélanges de blocs

	M 1	M 2	M 3	M 4	M 5	M 6	Moyenne 3 premiers M	Moyenne 3 derniers M
n1	75,29	74,25	72,87	69,68	61,29	59,68	74,14 ± 1,21	63,55 ± 5,37
n3	96,6	97,59	96,64	97,74	94,13	92,61	96,94 ± 0,56	94,83 ± 2,64
n5	99,87	99,9	99,83	100	99,9	99,8	99,87 ± 0,04	99,90 ± 0,10

Tableau 4.15 Résultats de toutes les méthodes sur le jeu de données LITIV 2014 comparés aux méthodes de l'état de l'art. Les résultats sont la moyenne des trois vidéos. **Caractères gras : Meilleur résultats** ; *Texte italique : Méthodes apportées dans ce rapport en ordre présenté.*

Méthodes	*erreur* ≤ 1 *pixel*	*erreur* ≤ 3 *pixel*	*erreur* ≤ 5 *pixel*
Réseau HRNet complet avec masques	66,80 ± 8,11	95,57 ±3, 22	99,74 ±0, 26
Sorties de HRNet avec les sorties concaténées	74,14 ± 1,21	**96,94 ± 0,56**	**99,87 ± 0,04**
Adaptation de HRNet	**75,00 ± 0,66**	96,24 ± 0,40	99,61 ± 0,23
Méthode des masques	57.52 ± 2.32	88.74 ± 0.99	98.55 ± 0.39
Domain Siamese CNN [13]	56.25 ± 3.47	89.95 ± 0.39	98.53 ± 0.43
CNN Siamois [12]	-	77.9 ± 5.04	-
St-Charles [1]	48.2 ± 3.95	-	-
Information mutuelle [18] (40×130)	-	83.3	-
Information mutuelle [18] (20×130)	-	77.5	-
Information mutuelle [18] (10×130)	-	64.9	-
Fast Retina Keypoint [18](40×130)	-	64.1	-
Local Self-Similarity [1, 18](40×130)‡	22.6 ± 10.66	73.4	-
Différence des carrées [18](40×130)	-	65.6	-

Tableau 4.16 Résultats pour HRNet avec les sorties concaténé du dernier et de l'avant derniers stage et l'ajout des masques. M : Mélanges de blocs

	M 1	M 2	M 3	M 4	M 5	M 6	Moyenne 3 premiers M	Moyenne 3 derniers M
n1	62,05	76,16	62,18	52,08	44,55	45,45	66,80 ± 8,11	47,36 ± 4,11
n3	91,98	98,21	96,51	95,15	89,75	85,57	95,57 ± 3,22	90,16 ± 4,80
n5	99,44	99,88	99,9	99,96	99,81	98,46	99,74 ± 0,26	99,41 ± 0,83

Pour commencer, parlons de la première méthode apportée dans ce mémoire, soit l'addition des masques au réseau RNC de Beaupré et al. [13] Tel que décrit à la section 4.4, nous avons concaténé aux images en entrée leurs masques de segmentation. Au tableau 4.15 nous pouvons voir les résultats sur *Domain Siamese CNN [13]* qui est la méthodes de Beaupré et al. [13], et la méthode que nous allons comparer, soit la *Méthode des masques*. Pour les trois précisions, les résultats étaient similaires. Cependant, pour les précisions de moins de 1 et 5 pixels, les résultats de l'addition du masque sont légèrement supérieur comparés aux résultats de Beaupré et al. [13]. Pour l'erreur sous 1 et 5 pixels, les améliorations sont de respectivement $1,27$ et 0.02 pour LITIV 2014 et pour l'erreur sous 3 pixels, il y a une baisse de $1,21$ pour LITIV 2014. Cependant, pour LITIV 2018, une hausse de $16,05$ et $5,98$ est observé respectivement pour la précision sous 1 et 3 pixels. Pour la précision sous 5 pixels, une baisse de $0,06$ est vue. De manière générale, les masques donnent une légère amélioration avec le RNC sans toutefois améliorer les résultats de manière significative. Il est normal que la précision sous 5 pixels aille moins d'amélioration étant donné que la valeur de la métrique de rappel est plus proche de 100 que l'erreur sous 1 pixel. Cependant, une amélioration de

Niveau de dilatation (px)	n1	n3	n5
0	62,05	91,98	99,44
5	66,47	94,61	99,76
10	73,65	97,26	99,79
15	74,07	97,64	99,95

Tableau 4.17 Métrique de rappel pour les précisions n1, n3 et n5 selon le niveau de dilatation du masque avec le réseau final HRNet+Masque.

Tableau 4.18 Résultats de toutes nos méthodes sur le jeu de données LITIV 2018. Les résultats sont la moyenne des trois vidéos. **Caractères gras : Meilleur résultats**.

Méthodes	*erreur* ≤ 1 *pixel*	*erreur* ≤ 3 *pixel*	*erreur* ≤ 5 *pixel*
Réseau HRNet complet avec masques	47,36 ± 4,11	90,16 ± 4,80	99,41 ± 0,83
Sorties de HRNet avec les sorties concaténées	**63,55 ± 5,37**	**94,83 ± 2,64**	**99,90 ± 0,10**
Adaptation de HRNet	63,34 ± 7,00	92,55 ± 2,26	99,69 ± 0,21
Méthode des masques	60,45 ± 4,38	87.40 ± 2,00	98,72 ± 0.10
Domain Siamese CNN [13]	44,40 ± 3.70	82,42 ± 7,00	98,78 ± 0,92

0.02 n'est pas significative. Du fait même, l'amélioration sous 5 pixels sur le LITIV 2018 n'est elle non plus significative.

Pour les résultats de l'adaptation de HRNet, à la section 4.5, nous pouvons voir de bien meilleures améliorations. Pour montrer l'efficacité de ce réseau, nous allons comparer la méthode *Domain Siamese CNN* [13] à la méthode *Adaptation HRNet*. Il est bien de rappeler que la présente expérience part du réseau de Beaupré et al. [13], et non de la méthode avec masque. Nous n'avons donc pas de masques pour cette expérience. Le but de cette expérience est d'améliorer les résultats en utilisant un extracteur de caractéristiques plus moderne. Les détails des modifications effectuées pour adapter HRNet à notre modèle ont été présentés à la section [13]. Pour les trois précisions, une amélioration significative s'est fait voir. En effet, nous pouvons voir au tableau 4.15 que pour la précision sous 1 pixel, la métrique de rappel est passée de $56,25$ pour *Domain Siamese CNN* à $75,00$ pour l'*adaptation de HRNet*. Ceci augmentant ainsi la précision de $18,75$. Pour la précision sous 3 pixels, nous voyons une amélioration de $6,29$, donnant une valeur de métrique de rappel de $96,24$ à la présente méthode. Pour la précision sous 5 pixels, la valeur de rappel passe de $98,53$ à $99,61$, ce qui fait en sorte que nous nous rapprochons du score partait de $1,00$. Cette augmentation n'est pas négligeable pour la précision sous 5 pixels, étant donné que le résultat de l'ancienne méthode était déjà à $1,47$ du score parfait. Pour les résultats sur le LITIV 2018, les résultats s'améliorent aussi de la même façon. Ces résultats montrent donc l'impact positif d'un meilleur extracteur de caractéristiques tel que HRNet.

La prochaine méthode modifie la sortie des vecteurs de caractéristiques de HRNet. Les détails des modifications sont expliqués en détail à la fin de la section 3.4. Les résultats reliés à cette section dans les tableaux 4.15 et 4.18 sont identifiés par *sorties de HRNet avec les sorties concaténées*. Si nous faisons un simple rappel de la méthode, nous concaténons les couches de sortie haute résolution des deux derniers étages pour former la sortie finale. Les modifications effectuées font suite aux modifications de la méthode d'*adaptation de HRNet*. Nous pouvons voir une baisse de $0,86$ par rapport à la méthode d'*adaptation de HRNet* pour la précision sous un pixel pour LITIV 2014. Cette baisse n'est cependant pas significative étant donné que l'écart type est de $1,21$. Sur le LITIV 2018 cependant nous voyons une hausse de $0,21$. Pour l'erreur sous 3 pixels, il y a une légère amélioration de $0,7$ sur le LITIV 2014, et sur le LITIV 2018 une amélioration de $2,18$ est vue. L'augmentation est légère, mais l'écart de l'amélioration est plus grand que l'écart type, ce qui signifie que l'ensemble des mélanges de données a engendré une amélioration. Pour la précision sous 5 pixels, une amélioration de $0,26$ est faite, montant le score à $99,87$. La méthode est vraiment efficace pour cette précision étant donné que l'écart type est de $0,04$. Pour le LITIV 2018 le résultat obtenu est similaire. On obtien un résultat de $99,90$ augmentant ainsi le résultat précédent de $0,21$. Nous pouvons conclure que la concaténation augmente la précision sous 5 pixels de façon significative, en sacrifiant de la précision pour celle sous 1 pixel pour le jeu de donnée LITIV 2014. Étant donné que la sortie de ce réseau est constituée de seulement des cartes de caractéristiques de haut niveau, le réseau est donc précis de façon globale pour des précisions plus élevées, mais manque de détails pour avoir une amélioration significative sous 1 pixel.

La dernière méthode proposée est une combinaison de la méthode précédente, soit *sorties de HRNet avec les sorties concaténées* et la *Méthode des masques*. Cette méthode ajoute donc les masques de segmentation aux images d'entrée du réseau HRNet avec les sorties améliorées. Les détails des changements sont à la section 3.5. Nous allons commencer à comparer cette méthode à la *méthode des masques*, pour voir l'effet du masque sur HRNet, et ensuite nous allons comparer l'actuelle méthode des *sorties de HRNet avec les sorties concaténées*, pour voir l'efficacité de HRNet sur la méthode des masques. Donc, pour la première comparaison entre le *réseau HRNet complet avec masques* et la *Méthode des masques*, nous voyons une bonne amélioration pour l'ensemble des précisions. Pour la précision sous 1 pixel sur LITIV 2014, il y a une amélioration de $9,28$, ce qui est bon, mais reste moins que l'amélioration de $17,89$ pour l'ajout de HRNet sur le réseau sans masques. Pour la précision sous 3 pixels, il y a une amélioration de 6.83 entre le *réseau complet avec masque* et la *méthode avec masque*. Une précision de $6,29$ à été aperçu lors de l'ajout de HRNet au réseau RNC de Beaupré et al. [13] Finalement, pour la précision sous 5 pixels, un écart de $1,19$ est aperçu pour l'ajout de HRNet sur la *méthode des masques* et un écart de $1,34$ à été fait lors de l'addition de

HRNet à la méthode de Beaupré et al. [13]. En comparant l'ajout de HRNet sur les réseaux avec masque et sans masques, nous pouvons conclure que HRNet à un moins gros impact sur les résultats sur les réseaux avec les masques de segmentation, qu'il a sur les réseaux sans masques de segmentation.

Maintenant, regardons l'impacte des masques sur le réseau HRNet, en comparant le *réseau HRNet complet avec masques* avec les *sorties de HRNet avec les sorties concaténées*. Pour la première précision, l'ajout du masque sur HRNet a une baisse de $7,34\%$ pour le LITIV 2014 et une baisse de $16,19$ pour le LITIV 2018. Pour la précision sous 3 pixels, il y a une diminution de la métrique de rappel de $1,37$ comparée à la méthode des *sorties de HRNet avec les sorties concaténées* sur le LITIV 2014 et une baisse de $4,67$ pour le LITIV 2018. Pour la précision sous 5, nous avons une diminution de 0.13 pour le LITIV 2014 et pour le LITIV 2018 une diminution de 0.49. La grande diminution de la précision sous 1 pixel et la diminution sous 3 pixels pour les 2 jeux de données nous fait croire que l'addition des masques sur le réseau HRNet n'est pas efficace.

Le réseau que nous considérons optimal avec les résultats obtenu serait donc le réseau des *sorties de HRNet avec les sorties concaténées*. Malgré sa légère réduction par rapport à la méthode d'*adaptation de HRNet*, les deux autres précisions restent supérieur, ainsi que les écarts types, qui rendent la méthode stable selon les mélanges de blocs. De plus, avec les résultats sur LITIV 2018, nous pouvons voir que les trois précisions sont meilleur avec la méthode des *sorties de HRNet avec les sorties concaténées*. De toutes les méthodes testées, nous avons eu les meilleurs résultats avec la méthode des *sorties de HRNet avec les sorties concaténées* sur la précision sous 3 et 5 pixels, avec des précisions respectives de $96,94$ et $99,87$ sur le jeu de donnée LITIV 2014.

CHAPITRE 5 CONCLUSION

Ce dernier chapitre conclura le présent mémoire. Une synthèse des travaux vous sera présentée. Ensuite, des limitations de la solution et ainsi que quelques suggestions sur des améliorations futures seront proposées.

5.1 Synthèse des travaux

Dans ce mémoire, nous avons proposé de nouvelles méthodes pour améliorer l'estimation de disparité sur des silhouettes humaines entre deux images stéréoscopiques en couleur et infrarouge thermique. Les modèles proposés permettent d'obtenir des résultats qui dépassent ceux qui existent dans les travaux précédents.

La première méthode proposée est un réseau pseudo-siamois qui prend en entrée une sous-région d'une paire d'images stéréoscopiques dont une est en couleur et l'autre est une image infrarouge thermique. Les masques binaires dans les silhouettes humaines sont générés et sont concaténés aux sous-régions, indiquant ainsi au réseau si une frontière se trouve dans la sous-région. Ces images à quatre canaux passent ensuite dans un extracteur de caractéristiques, et subissent des opérations de fusion pour finalement savoir la disparité du point central de la sous-région.

La deuxième amélioration proposé a été de modifier l'extracteur de caractéristiques. Nous avons pris un réseau de référence, donc sans l'ajout du masque, et nous avons remplacé le RNC par un extracteur de caractéristiques plus moderne, soit HRNet. Ce réseau a été choisi, car il a la particularité de produire dans caractéristiques de haute résolution. Les résultats ayant été prometteurs, nous avons décidé poursuivre sur cette idée, et nous avons décidé de modifier la sortie du réseau pour avoir le plus de caractéristiques à haute résolution possibles pour augmenter l'information disponible pour calculer les disparités. Pour modifier la sortie, nous avons pris le dernier et l'avant-dernier bloc de caractéristiques hautes résolutions. Ces changements ont fait en sorte que les résultats se sont encore améliorés.

La dernière étape a été de combiner les deux aspects du travail. Donc l'intégrer les masques à l'architecture proposé avec HRNet. Le principe est le même, soit de concaténer un masque de segmentation à l'image d'entrée du réseau pour obtenir une image à quatre canaux. Ensuite, nous avons donné ces images au réseau HRNet, qui a montré dans ce cas résultats mitigés, les masques semblant être moins utiles avec une représentation des images de meilleure qualité.

Le réseau HRNet modifié avec les images d'entrée à trois canaux reste donc la solution avec

les meilleurs résultats.

5.2 Limitations de la solution proposée

Une des limitations est le temps d'exécution de nos méthodes. Bien que le nouvel extracteur de caractéristiques donne de meilleurs résultats, le temps d'entraînement, et le temps de test a augmenté de façon significative. De plus, une autre limitation est le fait que notre méthode soit adaptée à des jeux de données dont les données sont éparses. Étant donné que les données dans les jeux de données sont éparses, il y a moins de points de disparités sur chacune des images. Nous ne pouvons donc pas générer de cartes de disparités précises pour toute l'image en utilisant uniquement ces données.

5.3 Améliorations futures

Certaines améliorations pourraient être appliquées pour améliorer les travaux effectués. Dans ce mémoire, nous ne nous sommes pas concentrés sur la tête du réseau. Nous avons encore une tête de réseau traditionnel. Cependant, une modification que nous pourrions apporter c'est de changer la tête du réseau pour remplacer les branches de corrélation et de concaténation avec un réseau de neurones à auto-attention. Nous avions commencé à faire cela en nous inspirant de la méthode de Li et al. [14]. Cependant, nous avons rencontré des erreurs lors de l'entraînement, ce qui a fait en sorte que nous n'avons pas pu terminer cette partie du projet. Le principe était de mettre à la tête du réseau un réseau de neurones à auto-attention, dont les entrées étaient les deux vecteurs de caractéristiques f_{RGB} et f_{LWIR} générés par HRNet.

Une autre amélioration possible qui pourrait être effectuée serait de changer la nature du problème pour faire un problème de régression au lieu que ce soit une classification. Il n'y aurait pas beaucoup de changement au niveau de l'architecture. Il suffirait de changer la fonction de perte actuelle pour mettre une fonction de perte de régression. Il est difficile de savoir si les résultats s'améliorerait, mais il serait intéressant de comparer cette fonction de perte à celle actuel. Un autre détail qui serait intéressant de voir, mais qui ne serait pas nécessairement une innovation en soi, serait de changer l'espace de couleur des images d'entrées. Par exemple choisir un format de type LAB.

Une dernière amélioration pourrait être de tester la méthode sur d'autres jeux de données. Par exemple, le jeu de données FLIR [57] est constitué d'un ensemble de paires d'images RGB-NIR. Ce n'est pas dans le même spectre de couleur étudier dans ce mémoire, mais il serait intéressant de tester le modèle proposé dans d'autres spectres.

RÉFÉRENCES

[1] B. G.-A. B. R. St-Charles, P.-L., "Online mutual foreground segmentation for multispectral stereo videos," 2019.

[2] J. Žbontar et Y. LeCun, "Computing the stereo matching cost with a convolutional neural network," p. 1592–1599. [En ligne]. Disponible : http://arxiv.org/abs/1409.4326

[3] Z. Chen, X. Sun, L. Wang, Y. Yu et C. Huang, "A deep visual correspondence embedding model for stereo matching costs," dans *Proceedings of the IEEE International Conference on Computer Vision*, 2015, p. 972–980.

[4] W. Luo, A. G. Schwing et R. Urtasun, "Efficient deep learning for stereo matching," dans *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, p. 5695–5703.

[5] H. Park et K. M. Lee, "Look wider to match image patches with convolutional neural networks," *IEEE Signal Processing Letters*, vol. 24, n°. 12, p. 1788–1792, 2016.

[6] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach et A. Bry, "End-to-end learning of geometry and context for deep stereo regression," dans *Proceedings of the IEEE international conference on computer vision*, 2017, p. 66–75.

[7] J.-R. Chang et Y.-S. Chen, "Pyramid stereo matching network," dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, p. 5410–5418.

[8] K. He, X. Zhang, S. Ren et J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition," dans *Computer Vision – ECCV 2014*. Springer International Publishing, 2014, p. 346–361. [En ligne]. Disponible : https://doi.org/10.1007%2F978-3-319-10578-9_23

[9] J. Pang, W. Sun, J. S. Ren, C. Yang et Q. Yan, "Cascade residual learning : A two-stage convolutional neural network for stereo matching," dans *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, p. 887–895.

[10] X. Guo, K. Yang, W. Yang, X. Wang et H. Li, "Group-wise correlation stereo network," dans *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, p. 3273–3282.

[11] E. B. Baruch et Y. Keller, "Multimodal matching using a hybrid convolutional neural network," Thèse de doctorat, Ben-Gurion University of the Negev, 2018.

[12] D.-A. Beaupre et G.-A. Bilodeau, "Siamese cnns for rgb-lwir disparity estimation," dans *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[13] ——, “Domain siamese cnns for sparse multispectral disparity estimation,” 2020.

[14] Z. Li, X. Liu, N. Drenkow, A. Ding, F. X. Creighton, R. H. Taylor et M. Unberath, “Revisiting stereo depth estimation from a sequence-to-sequence perspective with transformers,” dans *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, p. 6197–6206.

[15] K. Sun, B. Xiao, D. Liu et J. Wang, “Deep high-resolution representation learning for human pose estimation,” dans *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, p. 5693–5703.

[16] Y. Yuan, R. Fu, L. Huang, W. Lin, C. Zhang, X. Chen et J. Wang, “Hrformer : High-resolution transformer for dense prediction,” *arXiv preprint arXiv :2110.09408*, 2021.

[17] P. Duplessis-Guindon et G.-A. Bilodeau, “4d-multispectralnet : Multispectral stereoscopic disparity estimation using human masks,” 2022. [En ligne]. Disponible : https://arxiv.org/abs/2204.09089

[18] T. A. S.-C. P.-L. R. D. Bilodeau, G.-A., “Thermal-visible registration of human silhouettes : a similarity measure performance evaluation,” dans *Infrared Physics and Technology*, vol. 64, 2014, p. 79–86.

[19] D. Scharstein, R. Szeliski et R. Zabih, “A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,” dans *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*. IEEE Comput. Soc, p. 131–140. [En ligne]. Disponible : http://ieeexplore.ieee.org/document/988771/

[20] M. J. Hannah, “Computer matching of areas in stereo images,” section : Technical Reports. [En ligne]. Disponible : https://apps.dtic.mil/sti/citations/AD0786720

[21] P. Anandan, “A computational framework and an algorithm for the measurement of visual motion,” *International Journal of Computer Vision*, vol. 2, n°. 3, p. 283–310, 1989.

[22] L. Matthies, T. Kanade et R. Szeliski, “Kalman filter-based algorithms for estimating depth from image sequences,” *International Journal of Computer Vision*, vol. 3, n°. 3, p. 209–238, 1989.

[23] E. P. Simoncelli, E. H. Adelson et D. J. Heeger, “Probability distributions of optical flow.” dans *CVPR*, vol. 91, 1991, p. 310–315.

[24] T. Kanade, H. Kano, S. Kimura, A. Yoshida et K. Oda, “Development of a video-rate stereo machine,” dans *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 3. IEEE, 1995, p. 95–100.

[25] O. Veksler, “Stereo matching by compact windows via minimum ratio cycle,” dans *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, vol. 1. IEEE, 2001, p. 540–547.

[26] M. Okutomi et T. Kanade, “A locally adaptive window for signal matching,” *International journal of computer vision*, vol. 7, n°. 2, p. 143–162, 1992.

[27] T. Kanade et M. Okutomi, “A stereo matching algorithm with an adaptive window : Theory and experiment,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 16, n°. 9, p. 920–932, 1994.

[28] S. B. Kang, R. Szeliski et J. Chai, “Handling occlusions in dense multi-view stereo,” dans *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, vol. 1. IEEE, 2001, p. I–I.

[29] Y. Boykov, O. Veksler et R. Zabih, “A new algorithm for energy minimization with discontinuities,” dans *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer, 1999, p. 205–220.

[30] H. Hirschmuller, “Stereo processing by semiglobal matching and mutual information,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 30, n°. 2, p. 328–341, 2007.

[31] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *International Journal of Computer Vision*, vol. 60, n°. 2, p. 91–110, Nov 2004. [En ligne]. Disponible : https://doi.org/10.1023/B:VISI.0000029664.99615.94

[32] M. Brown et S. Süsstrunk, “Multi-spectral sift for scene category recognition,” dans *CVPR 2011*, 2011, p. 177–184.

[33] P. Viola et W. Wells, “Alignment by maximization of mutual information,” dans *Proceedings of IEEE International Conference on Computer Vision*, 1995, p. 16–23.

[34] N. Dalal et B. Triggs, “Histograms of oriented gradients for human detection,” dans *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, 2005, p. 886–893 vol. 1.

[35] A. Torabi et G.-A. Bilodeau, “Local self-similarity as a dense stereo correspondence measure for themal-visible video registration,” 07 2011, p. 61 – 67.

[36] P. Viola et W. M. Wells III, “Alignment by maximization of mutual information,” *International journal of computer vision*, vol. 24, n°. 2, p. 137–154, 1997.

[37] A. Shaked et L. Wolf, “Improved stereo matching with constant highway networks and reflective confidence learning,” dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[38] K. He, X. Zhang, S. Ren et J. Sun, “Deep residual learning for image recognition,” dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, p. 770–778.

[39] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy et T. Brox, “A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation,” dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, p. 4040–4048.

[40] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. Van Der Smagt, D. Cremers et T. Brox, “Flownet : Learning optical flow with convolutional networks,” dans *Proceedings of the IEEE international conference on computer vision*, 2015, p. 2758–2766.

[41] M. D. Pistarelli, A. D. Sappa et R. Toledo, “Multispectral stereo image correspondence,” dans *International Conference on Computer Analysis of Images and Patterns*. Springer, 2013, p. 217–224.

[42] J. Canny, “A computational approach to edge detection,” *IEEE Transactions on pattern analysis and machine intelligence*, n°. 6, p. 679–698, 1986.

[43] C. A. Aguilera, F. J. Aguilera, A. D. Sappa, C. Aguilera et R. Toledo, “Learning cross-spectral similarity measures with deep convolutional neural networks,” dans *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2016, p. 267–275.

[44] C. Aguilera, A. Sappa et R. Toledo, “Cross-spectral local descriptors via quadruplet network,” *Sensors*, vol. 17, p. 873, 04 2017.

[45] T. Zhi, B. R. Pires, M. Hebert et S. G. Narasimhan, “Deep material-aware cross-spectral stereo matching,” dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[46] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser et I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[47] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan et I. Sutskever, “Generative pretraining from pixels,” dans *International Conference on Machine Learning*. PMLR, 2020, p. 1691–1703.

[48] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, “An image is worth 16x16 words : Transformers for image recognition at scale,” *arXiv preprint arXiv :2010.11929*, 2020.

[49] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov et S. Zagoruyko, “End-to-end object detection with transformers,” dans *European conference on computer vision*. Springer, 2020, p. 213–229.

[50] X. Zhu, W. Su, L. Lu, B. Li, X. Wang et J. Dai, “Deformable detr : Deformable transformers for end-to-end object detection,” *arXiv preprint arXiv :2010.04159*, 2020.

[51] S. Zheng, J. Lu, H. Zhao, X. Zhu, Z. Luo, Y. Wang, Y. Fu, J. Feng, T. Xiang, P. H. Torr *et al.*, “Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers,” dans *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, p. 6881–6890.

[52] Z. Huang, X. Shi, C. Zhang, Q. Wang, K. C. Cheung, H. Qin, J. Dai et H. Li, “Flowformer : A transformer architecture for optical flow,” *arXiv preprint arXiv :2203.16194*, 2022.

[53] X. Sui, S. Li, X. Geng, Y. Wu, X. Xu, Y. Liu, R. Goh et H. Zhu, “Craft : Cross-attentional flow transformer for robust optical flow,” *arXiv preprint arXiv :2203.16896*, 2022.

[54] X. Guo, K. Yang, W. Yang, X. Wang et H. Li, “Group-wise correlation stereo network,” 2019.

[55] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo et R. Girshick, “Detectron2,” https://github.com/facebookresearch/detectron2, 2019.

[56] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick et P. Dollár, “Microsoft coco : Common objects in context,” 2015.

[57] FLIR, “Free teledyne flir thermal dataset for algorithm training.” [En ligne]. Disponible : https://www.flir.in/oem/adas/adas-dataset-form/