EpipolarNVS : Exploiter la géométrie épipolaire pour la synthèse de nouveaux points de vues

Gaétan Landreau, Mohamed Tamaazousti

EpipolarNVS : Exploiter la géométrie épipolaire pour la synthèse de nouveaux points de vues

Gaétan LANDREAU[1,2] Mohamed TAMAAZOUSTI[2]

[1] Meero, 75002 Paris - France
[2] Université Paris-Saclay, CEA-LIST, F-91120 Palaiseau - France

gaetan.landreau@meero.com

Résumé

La synthèse de nouvelles vues peut être adressée selon une multitude d'approches. Dans le scénario le plus contraignant, le nôtre dans ce travail, une unique image est utilisée pour générer un nouveau point de vue. Cependant, nous montrerons que dans un tel contexte, les solutions apportées par les derniers travaux en apprentissage profonds n'intègrent pas la transformation relative associée au changement de pose de la caméra de la manière la plus optimale et complète qu'il soit. Or, la géométrie épipolaire permet d'interpréter la transformation relative [R|t] comme une information qui peut se stocker sous la forme d'une image, rendant en partie compte de la relation existante entre l'image source et cible. Nous montrons que la géométrie épipolaire permet de bénéficier d'un a priori fort quant à la transformation 3D qui a eu lieu entre ces deux points de vues, et introduisons donc une manière innovante d'encoder une telle information via une image RVB.

Mots Clef

Synthèse de nouvelles vues, Apprentissage profond, Géométrie épipolaire

Abstract

Novel-view synthesis (NVS) can be tackled through different approaches. The most challenging scenario, the one where we stand in this work, only considers a unique source image to generate a novel one from another viewpoint. However, we argue that in such a framework, the latest learning-based solutions do not integrate the relative camera transformation in an optimal and complete way. On the other hand, epipolar geometry interprets the relative transformation [R|t] as an information that can be stored through an image. It directly accounts on the physical relationship that exists between the source and target image. We argue that epipolar geometry allows to get a strong a priori regarding the transformation that took place and therefore introduce a new innovative way to encode such information via an RGB image.

Keywords

Novel View Synthesis, Deep Learning, Epipolar geometry

1 Introduction

La synthèse de nouvelles vues consiste en la génération d'une image associée à un nouveau point de vue en s'appuyant sur une ou plusieurs images sources ainsi que les poses de caméra associées. Cette tâche a de passionnantes applications, tant d'un point de vue vision par ordinateur (stabilisation vidéo, mise en scènes 3D virtuelle) que sur des sujets de réalité virtuelle et augmentée.

Ce problème peut être adressé de différentes manières, selon les données accessibles : plusieurs images source, cartes de profondeur, vérités terrain 3D tel que des nuages de points, pose de caméra (parfaites ou bruitées) etc. Nous considérons dans ce travail l'une des configurations les plus complexes en synthèse de nouvelles vues : celle qui consiste à générer une image selon un nouveau point de vue en ne s'appuyant que sur une unique image source ainsi que la transformation relative entre les poses de la caméra.

Les principaux efforts actuels se concentrent en priorité sur l'apparent réalisme des nouveaux points de vues générés, principalement autour des méthodes NeRF [11, 19, 1, 2]. En revanche, très peu de travaux adressent un autre problème inhérent à la synthèse de nouvelles vues : celui de trouver la manière la plus optimale de fournir au réseau de neurones une information de transformation relative de pose de caméra. D'une manière générale, la pose de caméra dans les travaux en synthèse de nouvelles vues à partir d'une unique image est très souvent réduite à ses paramètres extrinsèques. Le processus optique inhérent à la formation d'une image sur un plan, encodé via la matrice intrinsèque, est donc régulièrement omis des informations fournies aux modèles d'apprentissage profond. Conditionnée avec une information purement extrinsèque dans la littérature existante, nous désirons à travers ces travaux proposer une approche originale permettant de mieux rendre compte de la transformation 3D de pose de la caméra qui a eu lieu entre les images source et cible.

Tandis que la manière la plus élémentaire pour intégrer

une information de pose consiste à encoder directement les matrices de rotations et de translations relatives comme des vecteurs caractéristiques, nous affirmons, au regard des derniers travaux de l'état de l'art en estimation de cartes de profondeur à partir d'une image unique[23], qu'une telle méthode est sous optimale. Nous proposons donc dans ce travail une solution pour encoder la transformation relative d'une caméra comme une image RVB, en s'appuyant sur des considérations de géométries épipolaires. L'image encodée de cette transformation, d'une résolution spatiale similaire à l'image RVB fournie en entrée au réseau, permets de compenser l'écart qui existait en terme de dimensionnalité entre l'espace RVB et celui des matrices de transformation.

La contribution que nous proposons dans ces travaux se fait donc à trois niveaux distincts :

- Une stratégie pour encoder la transformation relative de la caméra en une image RVB, construite grâce à des concepts de géométrie épipolaire.
- Une architecture de réseau de neurones qui s'appuie sur cet encodage de pose.
- Une fonction de coût dite "spectrale" qui se focalise sur la préservation des hautes fréquences afin de récupérer au mieux les détails fins et complexes.

2 Travaux existants

Synthèse de nouvelles vues Dans la mesure où les modalités impliquées en synthèse de nouvelles vues sont nombreuses (images, séquence vidéo, nuages de point 3D, cartes de profondeur, pose de caméra plus ou moins complète etc), de d'intéressantes approches ont été développées pour adresser ce problème de synthèse de nouvelles vues. L'une des dernières tendances, les architectures de type NeRF [11], sont incroyablement puissantes pour générer des scènes selon un point nouveau point de vue, non observé à l'entraînement, et cela avec un réalisme stupéfiant [19, 12, 2]. Cependant, ces architectures sont en un sens sur-entraînées sur une scène unique dans la mesure où elles n'ont que peu de capacité générative. Ce défaut tend à être progressivement effacé par de récents travaux en synthèse de nouvelles vues [22, 10], qui indirectement induisent une capacité générative à ces modèles. MINE [10] est une méthode basée sur de multiples plans (dite MPI pour 'Multiplane Images') nécessitant des cartes de disparités précises comme vérité terrain. PixelNeRF [22] quant à lui fonctionne avec plusieurs vues en entrée, ce qui permet de raffiner et d'améliorer la qualité de la vue prédite. Un autre axe majeur de travail concerne la synthèse de nouvelles vues pour la navigation en intérieur sur des jeux de données tels que RealEstate10K [24] ou Matterport3D [3]. Les résultats sont très encourageants mais au détriment d'architectures complexes et exigeantes sur le type d'informations requis comme vérité terrain pour l'entraînement [21, 14, 13].

Modélisation de la pose de caméra Les paramètres extrinsèques d'une caméra sont parfaitement définit à travers 3 paramètres de rotation et 3 de translation. La matrice intrinsèque est quant à elle définit selon un nombre variable de paramètres (selon le type de modèle retenu pour modéliser le capteur physique de la caméra). Les paramètres extrinsèques de la caméra sont souvent les seules informations fournis à un réseau de neurons pour la synthèse de nouvelles vues.

Encodage de la transformation entre les poses de la caméra Une des solutions les plus évidentes pour encoder la transformation relative existante entre la position initiale et finale de la caméra consiste à simplement calculer la différence des matrices extrinsèque associées [16]. Une autre approche, plus fine, projette l'encodage précédent dans un espace de dimension plus importante [9, 14]. Ces deux dernières possibilités peuvent être d'ailleurs simplifiées si l'on considère un encodage *one-hot* pour les poses discrétisées. Enfin, l'un des travaux les plus proches du nôtre [23] encode la localisation de la caméra (paramétrisée par un angle de roulis et de tangage en considérant une hauteur fixe par rapport au sol) comme une image 2D. En se basant simplement sur la différence des poses $P_{diff} = P_{target} - P_{source} \in \mathbb{R}^v$, [16] duplique ce vecteur sur toute la surface de l'image (selon l'axe associé aux canaux RVB), pour donner en entrée du réseau une image de résolution $\mathbb{R}^{H \times W \times (3+v)}$. D'une manière différente, [9] adopte une stratégie consistant à concaténer le vecteur de pose (encodée dans un espace latent) avec celui associé à l'image d'entrée (qui est donc passée dans un encodeur de type CNN), avant l'entrée dans le décodeur, qui produira l'image selon un nouveau point de vue. Finalement, [21] a conçu une méthode de synthèse de nouvelles vues à partir d'une image unique en s'appuyant uniquement, pour la supervision, sur un nuage de points 3D. La transformation de la caméra est donc intégrée directement dans le réseau en rendant le nuage de points 3D transformé.

Synthèse de nouvelles vues à partir d'une image unique Une telle configuration est plus difficile à appréhender en synthèse de nouvelles vue dans la mesure où seule l'information la plus essentielle est fournie au réseau pendant l'entraînement et l'inférence. Seulement quelques travaux [16, 9, 22] se sont attaqués à cette configuration si complexe. Pouvoir gérer les poses de caméra discrètes (de ShapeNet [4]) et continues (de Synthia [15] ou KITTI [7]) comme le font [16, 9] suppose de devoir ajuster la taille du vecteur latent encodant la transformation relative. C'est l'un des avantages majeur de notre méthode : les transformations discrètes et continues sont gérés exactement de la même manière dans notre stratégie d'encodage grâce à la géométrie épipolaire. Cette propriété permet ainsi, à l'inférence, et à minima sur les poses discrètes, de prédire des points de vues qui n'étaient pas représentés dans le jeu de données d'entraînement.

3 Méthode

3.1 Encodage de la transformation relative

Géométrie épipolaire. On considère $I_s \in \mathbb{R}^{H\times W\times 3}$ l'image RVB source et I_t l'image cible. La matrice intrinsèque $K \in \mathbb{R}^{3\times 3}$ modélise une caméra idéale, sans défauts optique, connu en anglais sous le nom de 'pihole model'. La transformation rigide qui rend compte du changement de point de vue de la caméra entre les images source et cible est entièrement décrite par une rotation $R \in SO(3)$ et une translation $T \in \mathbb{R}^{3\times 1}$:

$$\begin{cases} R = R_t R_s^T \\ T = t_t - R t_s \end{cases} \tag{1}$$

où (R_s, t_s) et (R_t, t_t) correspondent respectivement aux matrices extrinsèques de l'image source et cible.
La géométrie épipolaire [8] tient compte de la géométrie projective qui relie deux points de vue distincts et possède diverses applications telles que le problème de 'Structure from Motion' [17]. La géométrie épipolaire, étant donné une paire de caméras et leurs poses correspondantes, vise à décrire la relation qui existe entre le repère monde 3D et les coordonnées 2D des pixels présent sur le capteur d'image. La matrice fondamentale $\mathrm{F} \in \mathbb{R}^{3\times 3}$ est une matrice qui décrit entièrement une telle relation 3D/2D et peut être obtenue par :

$$\mathbf{F} = K^{-T}[T]_X R K^{-1} \tag{2}$$

avec $[.]_X$ la matrice antisymétrique de n'importe quel vecteur 1D. Étant donné la position d'un pixel [1] $p_s \in I_s$, une telle matrice fondamentale $\mathbf{F} \in \mathbb{R}^{3\times 3}$ permets de définir :

$$\mathcal{P}_{p_s} = \{p_t \in I_t | p_t^T \mathbf{F} p_s = 0\} \tag{3}$$

comme un jeu fini de pixels de I_t se trouvant le long d'une droite épipolaire définit par $l = \mathbf{F}p_s$. D'un point de vue purement géométrique, une telle droite correspond au rendu d'un rayon dans un repère 3D (sur le plan associé à l'image de I_t) passant par le centre de la caméra associé à I_s et par p_s.
La matrice fondamentale $\mathbf{F}$ établit une correspondance pixel-droite via une équation linéaire qui implique à la fois les poses source et cible de la caméra. Dans la mesure où nous utilisons la géométrie épipolaire pour encoder la transformation du point de vue, une stratégie d'échantillonnage doit être définie afin de déterminer les pixels de l'image I_s qui vont être utilisés pour calculer ces lignes épipolaires. Au lieu d'échantillonner aléatoirement l'emplacement de ces pixels sur les $H \times W$ possibilités, les pixels sont échantillonnés selon une grille $\mathbf{G}_r$ qui couvre toute l'image, le paramètre *r* contrôlant la finesse de la grille :

$$\mathbf{G}_r = \left\{ p = (p_x, p_y) \in I \middle| \begin{array}{ll} p_x \equiv 0 & (\mathrm{mod}\ H/r) \\ p_y \equiv 0 & (\mathrm{mod}\ W/r) \end{array} \right\} \tag{4}$$

1. Pour des raisons de simplification, l'utilisation des coordonnées homogènes est ici implicitement utilisée.

Stratégie d'encodage. Notre module encode le mouvement relatif de la caméra qui existe entre les vues source et cible en tirant largement parti de la géométrie épipolaire. La sortie de l'encodeur est transmise à l'une des branches du réseau de neurones comme une image latente, qui représente donc implicitement la transformation qui s'est produite, et est appelée $E_{s\to t}$. Un aperçu des différentes étapes pour calculer $E_{s\to t}$ est présenté à travers le pseudo-code [2] dans l'algorithme 1.

Comme présenté dans l'algorithme 1, chaque ligne épipolaire rapportée sur $E_{s\to t}$ a une couleur associée au pixel échantillonné sur I_s. Un tel choix d'implémentation donne des informations RVB supplémentaires au réseau concernant la couleur qui doit être générée, même si les problèmes d'illuminations sont pas considérés.
Le procédé d'encodage décrit ici est ainsi capable de coder toute forme de transformation du point de vue de la caméra, qu'elle soit discrète ou continue, et ceux sans aucune adaptation structurelle. En effet, la plupart des travaux existants doivent changer la façon dont la transformation du point de vue est encodée puisque les poses continues ne sont pas traitées de la même manière que les poses discrètes, pour lesquelles une stratégie d'encodage *one-hot* peut être utilisée.

Algorithm 1 Module d'encodage épipolaire

procedure ENTRÉE : $(I_s, F, \mathbf{G}_r)$
  $E_{s\to t} = zeros(H,W,3)$
  for p_G in $\mathbf{G}_r$ **do**
    $colRGB = I_s[p_G]$
    Construire $\mathcal{P}_{p_G}$
    $\forall p \in \mathcal{P}_{p_G},\ \ E_{s\to t}[p] = colRGB$
  Sortie $E_{s\to t}$

Tous les pixels nuls sont exclus de $\mathbf{G}_r$ dans notre stratégie d'encodage. Ceci est particulièrement vrai pour les images issues de ShapeNet [4] puisque l'arrière plan est noir et uni et n'apporte donc aucunes informations pertinentes quant aux droites épipolaires associées. La figure 1 représente le résultat que l'on peut attendre d'un tel encodage sur une paire d'image source/cible issues de ShapeNet [4].

Stratégie d'encodage étendue. La génération d'une nouvelle vue sur les données de Synthia [15] et KITTI [7] est, en quelque sorte, d'un point de vue de la transformation relative de la caméra, redondante et délicate à gérer pour notre stratégie d'encodage. En effet, les images contenues dans ces jeux de données ont été enregistrées grâce à une voiture roulant dans les rues (virtuelles ou réelles), et la plupart des transformations de caméra considérées sont des mouvements de translation. De tels changements de point de vue entre les vues source et cible sont mal gérés par

2. Certaines valeurs de pixels sur $E_{s\to t}$ peuvent être écrasées avec un tel échantillonnage. Nous observons en pratique que cela n'a pas d'impacts significatifs sur notre encodage.

FIGURE 1 – De gauche à droite : Image source I_s, Image cible I_t et $E_{s\to t}$. Issu de la classe ShapeNet [4] *Chaise*. Les pixels utilisés sont issus d'un échantillonage avec une grille $\mathbf{G}_{25}$ et sont en rouge sur I_s.

la géométrie épipolaire puisque la notion de profondeur est perdue. Nous avons donc étendu notre stratégie d'encodage initiale pour ces jeux de données via l'addition d'un quatrième canal qui permet de rendre compte de cette notion de profondeur.
On note :

$$\Delta_t = |t_t - t_s| = \left[\Delta t^1, \Delta t^2, \Delta t^3\right]^T \in \mathbb{R}^3 \qquad (5)$$

Nous considérons ici une différence de valeurs absolues dans l'équation (5) puisque les coordonnées des plans 2D où la voiture se déplace ne sont pas uniformisées entre les différentes scènes de KITTI [7] et Synthia [15]. La valeur d'intérêt réelle dans notre cas est appelée δ_t et est obtenue via :

$$\delta_t = sign(\Delta_t^{p_M}) \times |\Delta_t^{p_M}| \qquad (6)$$

avec

$$p_M = \arg\max |\Delta_t| \qquad (7)$$

L'expression de δ_t satisfait deux contraintes pertinentes dans ce cas :

- Sa valeur (scalaire) tient compte de la direction principale de la voiture et donne une idée de la distance parcourue par la voiture entre les images source et cible.
- Le signe de δ_t permets de savoir si les vues source et cible correspondent à une mouvement qui va vers l'avant ou l'arrière.

Ces informations permettent au réseau de mieux appréhender la direction du mouvement qui doit être prise en compte. La valeur de δ_t est finalement stockée dans un quatrième canal de $E_{s\to t}$, uniquement où les différentes droites épipolaires se trouvent sur les 3 premiers canaux.

3.2 Architecture globale et fonction de coût

L'architecture globale est présentée en Figure 2. Une telle architecture s'inspire de celle introduite dans [9], à minima en ce qui concerne la structure d'encodeur/décodeur basée sur une architecture de type 'U-Net' avec une stratégie d'attention. Cependant, la façon dont les informations du changement de point de vue sont fournies au réseau change radicalement pour notre modèle. Tandis que [9] concatène deux vecteurs latents dans la couche la plus profonde du réseau, nous affirmons qu'un tel choix est sous-optimal, du moins pour les informations de pose de caméra discrètes de ShapeNet [4]. Nous encodons donc plutôt cette transformation de pose de caméra grâce à $E_{s\to t}$, comme une image qui alimente donc deuxième encodeur de type CNN. Ce dernier encodeur produit un second vecteur latent qui sera concaténé à celui associé à l'image source, avant que l'ensemble ne soit consommé par le décodeur.

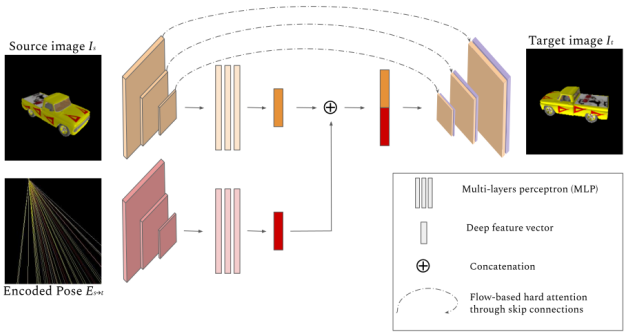

FIGURE 2 – Schéma général de l'architecture proposé. Le réseau prend en entrée l'image source I_s et $E_{s\to t}$ à travers deux encodeurs distincts, qui produisent chacun un vecteur latent, tous deux étant concaténés avant que l'ensemble ne soit fourni au décodeur. Le réseau de neurones s'appuie également sur des connections avec des mécanismes d'attention, introduits dans [9].

La fonction de coût totale est une somme pondérée d'une erreur moyenne (MAE), notée $\mathcal{L}_1$ et utilisée dans [9], ainsi que d'un second terme, appelé $\mathcal{L}_{spectral}$ et directement inspiré de travaux relatifs à la super-résolution [6]. Le dernier terme de coût est ainsi exclusivement dédié à la préservation des hautes fréquences. En effet, en considérant un filtre Gaussien 2D, noté w_{gauss}, une image I peut être décomposée en deux composantes notées I^{LF} et I^{HF}, contenant respectivement les basses et hautes fréquences de I :

$$\begin{cases} I^{LF} = I \circledast w_{gauss} \\ I^{HF} = I - I^{LF} \end{cases} \qquad (8)$$

où $\circledast$ représente l'opération de convolution 2D. La fonction de coût totale utilisée pendant l'entraînement est donc donnée par :

$$\mathcal{L} = \mathcal{L}_1 + \lambda \mathcal{L}_{spectral} = |\hat{I}_t - I_t| + \lambda \left(\hat{I}_t^{HF} - I_t^{HF}\right)^2 \qquad (9)$$

où $\hat{I}_t$ désigne l'image prédite par le générateur et λ est un coefficient de pondération des deux termes.

4 Expérimentations

Jeux de données. Nous avons testé notre méthode sur les mêmes jeux de données que [9, 16] : sur les classes *Chaise* et *Voiture* de ShapeNet [4] ainsi que sur les

images tirées de scènes réelles KITTI [7] et synthétiques de Synthia [15]. Les résultats reportés pour [9] sont légèrement différents de ceux originellement publiés dans la mesure où nous avons considéré un scénario de rendu plus complexe sur tous les jeux de données : les poses de caméra sont bruités sur ShapeNet [4], les bords d'images sur Synthia [15] et KITTI [7] ne sont pas supprimés. En effet, les images originales de Synthia et KITTI ont été redimensionnées à 256×256 sans avoir au préalable réalisé un crop central dans l'image (ce qui permet d'éliminer les éléments les plus difficiles à synthétiser sur la nouvelle vue).

Entraînement - Inférence. L'entraînement du modèle a été fait sur 100 000 itérations, en utilisant la même procédure d'entraînement que [9]. Dans la mesure où l'encodage de la pose relative à la transformation entre les deux points de vue est assez peu coûteux en calcul, les 'batch' d'images sont crées à la volée pendant l'entraînement. Les résultats d'inférences fournis ont été moyennés sur 100 itérations, avec une taille de batch égale 16.

Métriques. La MAE, le score SSIM [20] et le rapport signal sur bruit (PSNR) sont utilisés pour l'évaluation de notre modèle.

4.1 Résultats qualitatifs

Nous présentons en Figure 3 un exemple issu de chacun des quatre jeux de données considérés. Alors que notre modèle parvient à déduire et à prédire correctement la taille de l'objet en fonction de la vue cible sur les classes *Voiture* et *Chaise*, la nouvelle vue produite par [9] ne parvient pas à le faire. En effet, on voit en particulier sur l'exemple de la première ligne de la Figure 3 que la voiture dans l'image générée à la même taille que dans l'image source, ce qui n'est pas attendu pour l'image cible. Les résultats sur ces deux classes sont, d'une manière générale, plus nets et plus réalistes, tant du point de vue colorimétrique que structurel. Les roues de la voiture ou le dossier de la chaise sont ainsi mieux synthétisés avec notre méthode.
Dans la mesure où [9] intègre la pose de la caméra (son élévation) sous forme de pas discrétisés, la transformation du point de vue perd de sa cohérence spatiale 3D. Notre méthode utilise d'autre part un encodage qui tient mieux compte du changement de pose désiré par l'utilisateur.

Les résultats de notre méthode sur les images de scènes réelles et synthétiques sont présentés sur les troisième (Synthia [15]) et quatrième (KITTI [7]) lignes de la Figure 3.
Il est intéressant dans un premier temps de voir le mouvement global qu'a eu la voiture entre la vue source et la vue cible sur Synthia [15]. Bien que ces résultats restent flous, l'architecture proposée permet d' halluciner le mouvement général du bus vers l'avant. Le travail existant le plus proche prédit dans cette configuration ci une image

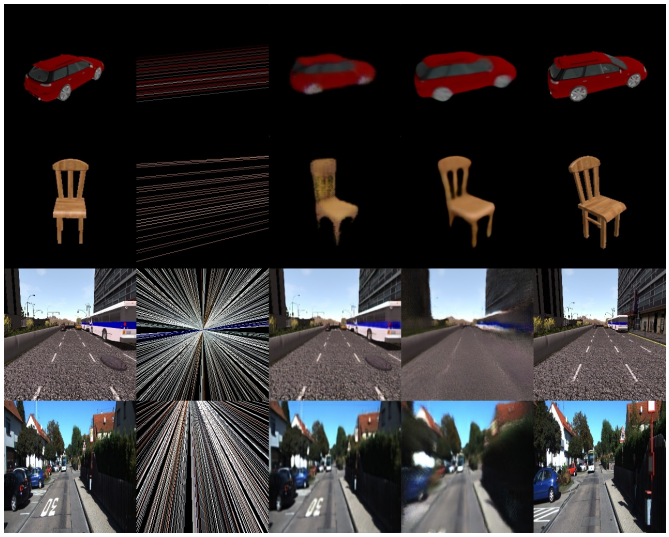

FIGURE 3 – Résultats d'inférence sur les quatre jeux de donnés considérés. L'image $E_{s\to t}$ a été obtenue en utilisant la grille d'échantillonage $\mathbf{G}_{15}$. Respectivement de gauche à droite : l'image source I_s, $E_{s\to t}$, la prédiction $\hat{I}_t$ de [9], la prédiction du modèle proposé $\hat{I}_t$, l'image cible I_t. D'une manière générale, notre méthode produit des résultats qui sont plus cohérents par rapport à l'image cible. Cela montre que notre stratégie d'encodage de la transformation relative est plus favorable que celle proposée dans [9].

sensiblement identique à l'image source, certes plus nette mais qui ne contient pas le déplacement relatif du bus qui a eu lieu. Un scénario sensiblement identique est proposé sur KITTI [7]. Tandis que le signe "30" au sol a disparu entre les deux vues, notre méthode réussit à capturer un tel changement là où [9] échoue de la même manière. Il convient de souligner l'importance de notre stratégie d'encodage dans les résultats présentés, notamment vis à vis de la forme globale des objets où la perspective change drastiquement entre I_s et I_t (sur le toit de la maison, dans le coin en haut à gauche).

4.2 Résultats quantitatifs

Certains résultats sont reportés ci-dessous sur les quatre jeux de donnés considérés. Les tables 1 and 2 résument respectivement les différentes métriques calculées pour ShapeNet [4] et sur les scènes d'environnement extérieurs (Synthia [15] et KITTI [7]). Tandis que notre méthode bat les méthodes existantes sur la MAE, nous obtenons des résultats extrêmement compétitif sur la métrique SSIM, dans la mesure où seul [16] (approche multi-vues) obtient de meilleurs résultats.

Cependant, nous précisons ici un aspect important concernant les résultats reportés. Seule notre méthode et celle de [9] ont été ré-entraînées sur les jeux de données étendus et

plus complexes que nous avons présentés précédemment. Les résultats des travaux restants [16, 25, 5, 18] sont donc ceux originellement publiés par les auteurs.

Modalité	Méthode	Voiture			Chaise		
		MAE (↓)	SSIM (↑)	PSNR (↑)	MAE (↓)	SSIM (↑))	PSNR (↑)
Multi-vues	[16]	0.078	0.935	-	0.141	0.911	-
Vue unique	[18]	0.139	0.875	-	0.223	0.882	-
	[25]	0.148	0.877	-	0.229	0.871	-
	[5]	0.119	0.913	-	0.202	0.889	-
	[22]	-	0.900	23.17	-	**0.911**	**23.72**
	[9]	0.026	0.892	21.18	0.045	0.865	17.89
	Nous	**0.016**	**0.928**	**24.23**	**0.032**	0.901	19.55

TABLE 1 – Performances sur ShapeNet [4]. Les meilleurs résultats pour la modalité d'image unique sont en gras tandis que les deuxième meilleurs résultats sont soulignés.

Les résultats sur Synthia [15] et KITTI [7] sont reportés sur le Tableau 2. Les résultats du modèle proposé restent compétitifs par rapport à l'état de l'art. Il est important de souligner que notre méthode (ainsi que [9]) a été entraînée sur des scénarios plus complexes quant à la composition des images.

Modality	Method	Synthia			KITTI		
		MAE (↓)	SSIM (↑)	PNSR (↑)	MAE (↓)	SSIM (↑)	PNSR (↑)
Multi-views	[16]	0.118	0.737	-	0.163	0.691	-
Single-view	[18]	0.175	0.612	-	0.295	0.505	-
	[25]	0.221	**0.636**	-	0.418	0.504	-
	[9]	0.065	0.632	**19.81**	0.087	0.602	16.84
	Ours	**0.065**	0.631	19.44	**0.082**	**0.609**	**17.11**

TABLE 2 – Performance sur Synthia [15] et KITTI [7]. Les meilleurs résultats pour la modalité d'image unique sont en gras tandis que les deuxième meilleurs résultats sont soulignés.

5 Limitations et travaux futurs

La stratégie d'encodage présentée dans ce papier est une solution innovante pour intégrer la transformation relative d'une caméra dans un réseau de neurones dédié à la synthèse de nouvelles vues. Cependant, cette méthode souffre de certaines limitations qui pourraient être adressées pour améliorer la qualité des résultats produits ainsi que le temps de traitement requis pour générer une nouvelle vue. En effet, dans la mesure où nous calculons ces images RVB $E_{s \to t}$ "à la volée" durant l'entraînement, notre méthode est plus lente que [9]. Un second axe de recherche qu'il conviendrait d'étudier concerne les données requises en entrée de notre modèle, qui pourraient être plus souples. En effet, il pourrait nous être souligné que notre méthode nécessite d'avoir également accès aux paramètres intrinsèques de la caméra, en plus des paramètres extrinsèques.
Une dernière piste de travail intéressante à creuser concerne la création d'une fonction de coût supplémentaire, qui pourrait être qualifiée de cyclique, où il s'agirait de définir $E_{t \to s}$, en s'appuyant donc sur l'image générée par le réseau pendant l'entraînement et la transformation inverse à celle originellement fournie au réseau.

6 Conclusion

En adressant le problème de la synthèse de nouvelles vues en apprentissage profond, nous présentons dans ce papier une nouvelle méthode novatrice pour encoder le déplacement relatif d'une caméra entre sa position initiale et finale. Nous nous appuyons sur la géométrie épipolaire afin d'encoder cette transformation comme une image RVB, notée $E_{s \to t}$ et composée de plusieurs droites épipolaires. Comparativement à la méthode standard consistant à ne fournir au réseau de neurones que les paramètres extrinsèques associés aux caméras source et cible, nous affirmons que notre méthode est mieux conçue pour la tâche adressée, à savoir à la synthèse de nouvelles vues à partir d'une image unique. En effet, cette stratégie permet d'aider le réseau à mieux comprendre la corrélation entre l'image source et le déplacement de caméra demandé par l'utilisateur. Les résultats expérimentaux présentés dans le cadre de ce travail sur différents jeux de données confirme cela. Ces résultats tendent à prouver que cette nouvelle stratégie d'encodage est, plus robuste sur des déplacements complexes où les changements perspectives sont importants, et permet de générer des nouvelles vues plus détaillées comparativement à l'état de l'art.

Références

[1] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf : A multiscale representation for anti-aliasing neural radiance fields. In *ICCV*, 2021.

[2] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360 : Unbounded anti-aliased neural radiance fields. In *CVPR*, 2022.

[3] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d : Learning from rgb-d data in indoor environments. In *3DV*, 2017.

[4] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, and al. Shapenet : An information-rich 3d model repository. *arXiv*, 2015.

[5] Park Eunbyung, Yang Jimei, Yumer Ersin, and al. Transformation-grounded image generation network for novel 3d view synthesis. In *CVPR*, 2017.

[6] Manuel Fritsche, Shuhang Gu, and Radu Timofte. Frequency separation for real-world super-resolution. In *ICCV*, 2019.

[7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving ? the kitti vision benchmark suite. In *CVPR*, 2012.

[8] Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[9] Kim Juhyeon and Young Min Kim. Novel view synthesis with skip connections. In *ICIP*, 2020.

[10] Jiaxin Li, Zijian Feng, Qi She, Henghui Ding, Changhu Wang, and Gim Hee Lee. Mine : Towards continuous depth mpi with nerf for novel view synthesis. In *ICCV*, 2021.

[11] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, and al. Nerf : Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[12] Michael Niemeyer, Jonathan T. Barron, Ben Mildenhall, Mehdi S. M. Sajjadi, Andreas Geiger, and Noha Radwan. Regnerf : Regularizing neural radiance fields for view synthesis from sparse inputs. In *CVPR*, 2022.

[13] Chris Rockwell, David F. Fouhey, and Justin Johnson. Pixelsynth : Generating a 3d-consistent experience from a single image. In *ICCV*, 2021.

[14] Robin Rombach, Patrick Esser, and Björn Ommer. Geometry-free view synthesis : Transformers and no 3d priors. In *ICCV*, 2021.

[15] German Ros, Laura Sellart, Joanna Materzynska, and & al. The SYNTHIA Dataset : A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[16] Shao-Hua Sun, Minyoung Huh, Yuan-Hong Liao, Ning Zhang, and Joseph J Lim. Multi-view to novel view : Synthesizing novel views with self-learned confidence. In *ECCV*, 2018.

[17] Mohamed Tamaazousti, Vincent Gay-Bellile, Sylvie Naudet Collette, Steve Bourgeois, and Michel Dhome. Nonlinear refinement of structure from motion reconstruction by taking advantage of a partial knowledge of the environment. In *CVPR*, 2011.

[18] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Single-view to multi-view : Reconstructing unseen views with a convolutional network. *ArXiv*, 2015.

[19] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus : Learning neural implicit surfaces by volume rendering for multiview reconstruction. In *NeurIPS*, 2021.

[20] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment : from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.

[21] Olivia Wiles, Georgia Gkioxari, Richard Szeliski, and Justin Johnson. SynSin : End-to-end view synthesis from a single image. In *CVPR*, 2020.

[22] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF : Neural radiance fields from one or few images. In *CVPR*, 2021.

[23] Yunhan Zhao, Shu Kong, and Charless Fowlkes. Camera pose matters : Improving depth prediction by mitigating pose distribution bias. In *CVPR*, 2021.

[24] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification : Learning view synthesis using multiplane images. In *SIGGRAPH*, 2018.

[25] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *ECCV*, 2016.