De la reconstruction par photogrammétrie à un environnement de réalité virtuelle immersif

Maxime Lhuillier
Université Clermont Auvergne, CNRS, Institut Pascal, F-63000 Clermont Ferrand, France
maxime.lhuillier@free.fr

Résumé

Il y a plusieurs étapes pour générer un environnement de réalité virtuelle (RV) à partir d'images : choisir les conditions expérimentales (scène, caméra, trajectoire, météo), prendre les images, reconstruire un modèle texturé avec un logiciel de photogrammétrie, et importer le modèle 3D dans un moteur de jeux. Cet article se concentre sur un post-traitement de l'étape de photogrammétrie, principalement pour les environnements extérieurs qui ne peuvent pas être reconstruits en utilisant un drone. Puisque la RV a besoin d'un modèle 3D dans un bon système de coordonnées (avec une échelle correcte et un des axes qui est vertical), une méthode simple est proposée pour calculer cela. Dans les expériences, on reconstruit d'abord des environnements immersifs urbains et naturels avec une caméra 360 (Gopro Max) fixée sur un casque, puis on importe dans Unity les modèles 3D avec des bons systèmes de coordonnées, et enfin on explore les scènes comme un piéton grâce à un casque de RV (Oculus Quest).

Mots Clef

Réalité virtuelle (RV), photogrammétrie, direction verticale, échelle, caméra 360, transformée de Hough.

Abstract

There are several steps to generate a VR environment from images: choose experimental conditions (scene, camera, trajectory, weather), take the images, reconstruct a textured 3D model thanks to a photogrammetry software, and import the 3D model into a game engine. This paper focuses on a postprocessing of the photogrammetry step, mostly for outdoor environments that cannot be reconstructed by UAV. Since VR needs a 3D model in a good coordinate system (with a right scale and one of the axes that is vertical), a simple method is proposed to compute this. In the experiments, we first reconstruct both urban and natural immersive environments by using a helmet-held (Gopro Max) 360 camera, then import into Unity the 3D models in good coordinate systems, last explore the scenes like a pedestrian thanks to a VR headset (Oculus Quest).

Keywords

Virtual Reality (VR), Photogrammetry, Vertical Direction, Scale, 360 Camera, Hough Transform.

1 Introduction

La RV a besoin d'un modèle 3D dans un bon système de coordonnées : l'échelle doit être physiquement plausible et l'un des trois axes doit être vertical. Une bonne échelle est nécessaire pour avoir un rendu stéréoscopique (une image différente pour chaque oeil), et une vitesse de déplacement réaliste pour l'utilisateur dans l'environnement de RV. De plus, la direction verticale doit être connue pour afficher en droites verticales, dans l'écran monté sur la tête, des droites verticales en 3D quand l'utilisateur regarde la scène avec des poses habituelles de la tête (avec des angles de roulis et tangage nuls). Sinon, les composants verticaux de la scène comme les murs et façades sembleront obliques, et l'immersion ne sera pas réaliste. Pire, le risque de cybernausée peut augmenter car il y a un conflit de capteur entre la verticale vue par les yeux et la verticale détectée par le système vestibulaire [19].

Malheureusement, l'étape standard de Structure-from-Motion des logiciels de photogrammétrie ne garantit pas un bon système de coordonnées. (Il y a d'autres raisons pour lesquelles un modèle 3D obtenu par photogrammétrie n'est pas directement utilisable en RV, comme la simplification de triangulation, mais elles ne sont pas traitées ici.) Elle peut seulement reconstruire une scène à une similitude près (avec 3+3+1 degrés de liberté : rotation, translation et échelle), si la caméra a un seul centre comme une caméra perspective [8]. Si la caméra a plusieurs centres, comme la plupart des caméras 360 composées de plusieurs caméras monoculaires, et si la distance entre les caméras composantes est connue, la scène est en théorie reconstructible à une transformation Euclidienne près (avec 6 degrés de liberté : rotation et translation). Mais l'échelle estimée est imprécise si cette distance est petite.

En pratique, l'échelle et la verticale sont choisies manuellement grâce à une interface graphique de logiciels comme dans Blender [2] ou Unity [6], mais cela peut être difficile à faire pour des grands environnements. Autrement, on peut calculer la direction verticale à l'orientation près (vers le bas ou le haut) en supposant que le mouvement de la caméra est approximativement sur un plan horizontal pendant l'acquisition des images, puis l'orientation est définie par l'utilisateur ou une hypothèse supplémentaire (ex : le sol est projeté en bas des images) en multipliant la verticale par -1 si besoin. Enfin, on applique

une rotation sur le modèle 3D de sorte que la direction verticale devienne un vecteur de la base canonique de $\mathbb{R}^3$. Bien que ce genre de méthode soit souvent suffisante, il y a des scènes pour lesquelles les verticales ainsi obtenues sont trop imprécises. Cet article montre des exemples : un village touristique construit sur un sol incliné et un canyon basaltique avec un sol très peu planaire.
On propose ici une méthode simple mais nouvelle pour calculer à la fois l'échelle et la direction verticale (orientation incluse) à partir d'un résultat de photogrammétrie, en supposant que

1. la hauteur de la caméra (par rapport au sol) pendant l'acquisition est en gros constante et connue
2. la surface reconstruite par photogrammétrie est complète et fermée : ses triangles recouvrent le plafond (au sens large : feuillage, ciel ...) et le sol
3. on sait si la hauteur de la caméra est plus petite que la différence entre les hauteurs du plafond et du sol (c'est le cas pour la plupart des scènes extérieures)
4. la scène est anisotrope : la densité de la distribution des normales de la surface est plus grande dans un voisinage des directions horizontales et dans un voisinage de la direction verticale qu'ailleurs.

La première hypothèse est facile à vérifier, par exemple pour une caméra montée sur un casque, en ajoutant un décalage à la taille de l'utilisateur qui prend les images. La seconde suppose que les triangles du plafond ne sont pas supprimés par le logiciel de photogrammétrie, même s'ils recouvrent le ciel. Si les images sont prises au niveau du sol avec une caméra 360, les points 3D sont reconstruits tout autour de la caméra, qui se retrouve dans l'enveloppe convexe des points 3D. Dans ce cas, la surface reconstruite par les méthodes standard (avec Delaunay 3D et contraintes de visibilité) sont fermées. Cela implique que la troisième hypothèse est vraie pour la plupart des scènes extérieures : la distance sol-caméra est inférieure à la distance plafond-caméra grâce aux composants élevés de la scène (bâtiments, arbres, ...). Pour une scène intérieure, la hauteur du plafond doit être connue. Enfin, on justifie l'hypothèse d'anisotropie. Le sol a une aire importante et sa pente est modérée. D'autres composants de la scène ont des aires importantes et (presque) verticales comme les façades dans les scènes urbaines ou les troncs dans les scènes naturelles. Donc on s'attend à ce que les normales soient le plus souvent voisines de la verticale ou des directions horizontales, respectivement.
Cet article est une amélioration et une traduction de [15].

2 Travaux précédants

Les logiciels de photogrammétrie ne calculent ni l'échelle ni la direction verticale, sauf si leur étape de Structure-from-Motion tient compte de mesures prises par un capteur supplémentaire (sec. 2.1). On fait ensuite un rapprochement avec une méthode précédante qui utilise aussi des transformées de Hough dans la sec. 2.2.

2.1 Capteurs supplémentaires

Les mesures obtenues par des capteurs GPS (global positioning system) et IMU (inertial measurement unit) peuvent être utilisées avec de l'imagerie terrestre [9] et aérienne [7], si ces capteurs et la caméra sont rigidement liés. L'ajustement de faisceaux [20] raffine simultanement des paramètres de poses de cameras et de points 3D en minimisant une somme d'erreurs de reprojections, qui peut être augmentée par un terme GPS [12] et un terme IMU [16]. Le GPS permet d'obtenir une reconstruction géoréférencée, ce qui implique que l'échelle et la direction verticale sont connues. Cependant, les mesures GPS ne sont pas fiables dans les lieux où les satellites GPS sont masqués par la scène (cas terrestre), par exemple par des bâtiments dans les canyons urbains. Les mesures IMU sont des accélérations de translation (gravité incluse) et des vitesses angulaires en 3D. La verticale peut être calculée lorsque les capteurs sont immobiles, puis l'échelle peut être calculée par intégration dans le temps (seulement quelques secondes à cause de la dérive) lorsque les capteurs se déplacent. La méthode que l'on propose ici n'a pas besoin d'un capteur supplementaire.

2.2 Transformée de Hough pour cylindres

Dans un contexte différent, une méthode [18] estime des cylindres à partir d'un nuage de points 3D obtenus en scannant un site industriel. Cette méthode est basée sur la transformée de Hough pour traiter les cas de faux points et d'instances multiples. Cependant, chaque cylindre a 5 paramètres (2 pour la direction du cylindre, plus 2 pour la position de son axe, plus 1 pour le rayon) et une transformée de Hough en 5D est incalculable en pratique. Donc cette méthode a deux étapes : d'abord trouver des directions de cylindres potentielles avec une transformée de Hough 2D, puis estimer des positions et rayons en utilisant des transformées de Hough 3D. La méthode que l'on propose utilise aussi une transformée de Hough 2D, mais avec des différences : il y a seulement une direction (la verticale), on ne fait pas l'hypothèse de cylindre à base circulaire dans la scène, la donnée d'entrée est une surface triangulée, les votes sont pondérés, l'espace de recherche est réduit à une partie d'une (hémi)sphère unité, enfin les calculs de verticale et d'échelle sont liés entre eux.

3 Choisir le système de coordonnées

Soit $\mathbf{v} \in \mathbb{R}^3$ la direction verticale pointant vers le ciel dans le système de coordonnées utilisé par l'étape de photogrammétrie. Un espace de recherche pour $\mathbf{v}$ est d'abord défini dans la sec. 3.1. Puis on estime $\mathbf{v}$ à l'orientation près dans la sec. 3.2 : on calcule un $\epsilon\mathbf{v}$ avec $\epsilon \in \{-1, +1\}$ dans l'espace de recherche. Enfin, on trouve la bonne orientation (sec. 3.3) et on change le système de coordonnées du modèle 3D (sec. 3.4).

3.1 Espace de recherche

Grâce au Structure-from-Motion, on connait les positions successives $\mathbf{l}_i \in \mathbb{R}^3$ de la caméra (images clés seulement) pendant l'acquisition des images dans le système de coordonnées utilisé par la photogrammétrie. On obtient d'abord une estimation grossière $\mathbf{v}_0$ de $\epsilon\mathbf{v}$ par une analyse en composantes principales (ACP) : $\mathbf{v}_0$ est le vecteur singulier correspondant à la plus petite valeur singulière de la matrice de covariance des $\mathbf{l}_i$. On a $\mathbf{v} \in \{+\mathbf{v}_0, -\mathbf{v}_0\}$ si le mouvement de la caméra est planaire et horizontal. L'espace de recherche de $\epsilon\mathbf{v}$ est un sous-ensemble de la sphère unité de $\mathbb{R}^3$: les vecteurs unités $\mathbf{u}$ formant un angle avec $\mathbf{v}_0$ qui sont inférieurs à un seuil α. Puisque l'on veut traiter le cas des sols avec des pentes modérées, un α suffisamment grand est nécessaire. De plus $\alpha \leq \pi/2$ car un hémisphère contient toutes les directions non-orientées.

3.2 Verticale non-orientée

La direction verticale est détectée avec une transformée de Hough 2D. L'espace de recherche de $\epsilon\mathbf{v}$ est d'abord échantillonné en un ensemble fini de (vecteurs) candidats verticaux. Puis chaque triangle de la surface vote pour tout les candidats verticaux qui sont presque parallèles au triangle. (La normale du triangle et un candidat vertical sont presque orthogonaux.) Les candidats verticaux d'un triangle sont donc inclus dans un grand cercle de la sphère unité. De plus, les votes de chaque triangle sont pondérés par l'aire du triangle. Ainsi le résultat des votes est invariant par subdivision des triangles. Enfin, $\epsilon\mathbf{v}$ est le candidat vertical qui maximise les votes.

On explique maintenant comment cette méthode donne le résultat attendu grâce à l'hypothèse d'anisotropie (sec. 1). Une aire importante de la surface est formée des triangles (presque) verticaux. Chaque triangle vertical vote pour une partie d'un grand cercle, et tout ces grands cercles ont exactement deux points en commun : la verticale définie à son orientation près. (Car un grand cercle est dans un plan vertical qui est parallèle à celui du triangle.) Donc les votes proches de la direction verticale recherchée sont élevés. Cependant, les votes proches des directions horizontales sont aussi élevés car le sol est en gros horizontal et a lui aussi une aire importante. (Chaque triangle horizontal vote pour toutes les directions horizontales.) Il y a donc un risque d'obtenir un mauvais $\epsilon\mathbf{v}$ si l'un des "votes horizontaux" (i.e. somme des votes pour une direction horizontale ou presque) est plus grand que les "votes verticaux" (définition similaire). Cependant, l'angle entre $\mathbf{v}_0$ et toute direction horizontale est plus grande que l'angle entre $\mathbf{v}_0$ et la verticale, puisque $\mathbf{v}_0$ est en gros vertical. On voit maintenant qu'un α suffisamment petit rejette les votes horizontaux et garde les votes verticaux. Puisque les votes élevés restants sont limités aux votes verticaux, le vecteur $\epsilon\mathbf{v}$ finalement estimé est vertical.

Dans les expériences, le compromis est $\alpha = \pi/4$ pour accepter une pente de sol modérée et pour rejeter les votes des triangles du sol. De plus, on échantillonne l'espace de recherche de sorte que les candidats verticaux correspondent aux pixels d'une image : le candidat vertical d'un pixel se projete sur ce pixel avec une caméra perspective [8]. Cette caméra est centrée en zéro comme la sphère unité, son champs de vue est 2α, elle projette $\mathbf{v}_0$ sur le centre de l'image. Donc chaque triangle vote pour les (candidats verticaux des) pixels d'un segment de droite.

3.3 Verticale orientée

Maintenant $\epsilon\mathbf{v}$ est connu mais ϵ et $\mathbf{v}$ sont inconnus. On connait donc la fonction $\epsilon \mapsto \mathbf{v}_\epsilon$. Pour chaque valeur possible $\epsilon \in \{-1, +1\}$, on examine les triangles qui sont "en dessous" de la trajectoire de caméra (pendant l'acquisition) selon $\mathbf{v}_\epsilon$. De façon plus détaillée, pour chaque position de caméra $\mathbf{l}_i$, on collecte dans une liste L_i le(s) triangle(s) qui intersecte(nt) la demi-droite DD_i d'origine $\mathbf{l}_i$ et de direction $-\mathbf{v}_\epsilon$. Puis on calcule la moyenne m_ϵ de la distance entre $\mathbf{l}_i$ et l'intersection(s) entre DD_i et tout triangle dans L_i (pour tous les i). En pratique, on réduit ces calculs 3D en calculs 2D en tournant la surface et la trajectoire de caméra de sorte que l'axe Oz soit parallèle à $\mathbf{v}_\epsilon$. On utilise aussi des buckets pour accélérer les tests d'intersection.

Grâce à l'hypothèse 2 (sec. 1), m_{-1} et m_{+1} sont les distances caméra-sol et caméra-plafond, mais on ne sait pas qui est qui. Grâce à l'hypothèse 3, la distance caméra-sol est la plus petite. Donc $\mathbf{v} = \mathbf{v}_\epsilon$ avec ϵ tel que $m_\epsilon < m_{-\epsilon}$.

3.4 Changement du système de coordonnées

On tourne le modèle 3D de sorte que l'axe Oz ait pour direction $\mathbf{v}$, avec une rotation d'axe $\mathbf{r}/||\mathbf{r}||$ et d'angle $\arcsin(||\mathbf{r}||)$ tels que $\mathbf{r} = \mathbf{v} \wedge (0 \quad 0 \quad 1)^\top$. On le dilate aussi en le multipliant par $h/\min\{m_{-1}, m_{+1}\}$, avec h la hauteur (en mètres) de la caméra pendant l'acquisition, grâce à l'hypothèse 1.

4 Autres détails

Les détails donnés ici ne sont pas le sujet principal de l'article mais sont utiles pour les expérimentateurs et ré-implémenteurs. La sec. 4.1 décrit des spécificités de la caméra utilisée. La sec. 4.2 se concentre sur les capacités de déplacement en RV avec l'aide d'une "surface de mouvement" (sec. 4.3) pour les sols non-planaires.

4.1 Photogrammétrie avec la Gopro Max

La Gopro Max 360 est peu encombrante et a une bonne résolution (détails plus bas) tout en étant grand public. Elle est composée de deux caméras fisheyes qui pointent dans des directions opposées (fig. 1). Cependant, contrairement aux autres caméras 360, les images obtenues ne sont ni équirectangulaires, ni les images fisheye originales[1], mais des images de "cubemap équiangulaire" [17]. Avec un cubemap standard, la sphère de vue est projetée sur les 6 faces d'un cube unité : une demi-droite (rayon

[1] Il y a aussi une vidéo d'images fisheyes à faible résolution et fort taux de compression, mais c'est insuffisant pour la photogrammétrie.

lumineux) d'origine le centre du cube est projetée sur le cube en l'intersection de la demi-droite avec l'une des 6 faces, puis les 6 images obtenues sur les 6 faces sont concaténées dans une image composite (l'image du cubemap). Autrement dit, l'image de chaque face est celle d'une caméra perspective. Avec un cubemap équiangulaire, chacune de ces caméras perspectives est remplacée par une caméra équiangulaire de même champs de vue et même centre. Ceci permet de mieux répartir la résolution avant l'étape de compression.

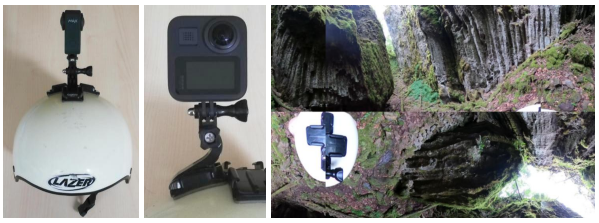


Figure 1: Une Gopro Max 360 et une image de cubemap.

Le microprogramme de la Gopro Max assemble donc chaque paire d'images fisheye originales en une image de cubemap équiangulaire, puis les compresse dans un fichier vidéo. Comme une image de cubemap équiangulaire est la projection d'une caméra centrale, et que la Gopro Max est en réalité non-centrale (environ 5cm entre les centres des fisheyes), il y a une approximation centrale. Il faut ensuite convertir chacune de ces images pour un modèle de projection supporté par des logiciels de photogrammétrie (liste non exhaustive ici), par exemple : une image équirectangulaire pour [5], ou une image de cubemap standard pour [1], ou plusieurs images de fisheye avec un modèle de projection classique avec distortion radiale pour [4]. Pour ces deux raisons (approximation centrale et conversion d'images), les images données en entrée au logiciel de photogrammétrie sont un peu dégradées.

En pratique, les images de cubemap équiangulaires sont obtenues à partir de fichier(s) avec le suffix ".360", de la façon suivante. Chaque fichier est limité à 4Go et contient deux vidéos MP4 4096×1344 à 30Hz synchronisées, d'au plus 8 minutes. Une image de cubemap est la juxtaposition de deux images synchronisées de ces vidéos, chacune d'elle a trois faces de cube. D'abord, on extrait et concatène (en temps) en deux fichiers MP4 sans perte de qualité, avec deux lignes de commandes ffmpeg [3]

```
ffmpeg -f concat -i list.txt -map 0:0 -c copy cube1.MP4
ffmpeg -f concat -i list.txt -map 0:5 -c copy cube2.MP4
```

et avec un fichier list.txt qui contient les noms des fichiers ".360". Ensuite, on extrait les images successives en décompressant simultanément cube1.MP4 et cube2.MP4 avec ffmpeg de façon classique, et en juxtaposant les deux images obtenues simultanément.

4.2 Capacités de déplacement en RV

Le but est la visualisation en RV de modèles 3D immersifs reconstruits par photogrammétrie, qui peuvent être de grande taille, dans le sens où la longueur de trajectoire de la caméra d'acquisition fait plusieurs centaines de mètres. Il faut que l'utilisateur de RV

1. se déplace librement sur le sol comme un piéton
2. puisse suivre la trajectoire de la caméra d'acquisition pour une exploration naive, là ou la qualité visuelle est bonne
3. ait une bonne conscience spatiale avec un risque de cyber-nausée faible.

On détaille maintenant chacun de ces points.
On veut (1) dans le cas d'un sol non planaire. Pour cela, on contraint les positions des yeux de l'utilisateur à être sur une surface de mouvement qui approxime le sol avec un décalage vertical. Grâce à l'hypothèse 1 en sec. 1, on définit la surface de mouvement par extrapolation de la trajectoire de la caméra (plus de détails dans la sec. 4.3).
Pour (2), on augmente le modèle 3D texturé avec un petit cube gris à chaque position d'acquisition (images clés seulement) de sorte que l'utilisateur de RV peut voir et suivre la trajectoire d'acquisition sans perdre de temps à chercher où sont les parties de la scène où la qualité visuelle est bonne. L'utilisateur peut aussi s'en éloigner si son objectif est d'évaluer visuellement la qualité de reconstruction. Plus la distance entre l'utilisateur et cette trajectoire est grande, plus les erreurs de reconstruction apparaissent clairement.
Enfin, il faut choisir une méthode de déplacement pour un environnement de RV de grande taille [21] qui vérifie (3). L'utilisateur a une bonne conscience spatiale de l'environnement de RV, par exemple s'il est capable de retrouver un objet déjà vu pendant un parcours de cet environnement, s'il ne se perd pas facilement. On sait que la conscience spatiale dépend de la méthode de déplacement en RV : elle est en général meilleure pour un mouvement continu (une translation qui dure dans le temps) que pour un mouvement discontinu (un saut brutal) entre deux positions dans l'environnement. Malheureusement, le risque de cyber-nausée est plus élevé pour un mouvement continu que pour un mouvement discontinu. Il faut donc faire un compromis entre les deux conditions de (3). Le meilleur compromis n'est pas le sujet ici et on implémente des capacités de déplacements similaires à ceux de jeux de RV : translations continues avec le joystick de gauche, mouvements discontinus (rotations en ajoutant $\pm\pi/12$ à l'angle de lacet, sauts) avec le joystick de droite. La translation continue est aussi utile à l'utilisateur qui veut juger le résultat de photogrammétrie avec la parallaxe de mouvement dans les images de RV. La rotation permet de ré-initialiser une direction moyenne pour la suite de l'exploration du modèle 3D reconstruit. L'utilisateur peut tourner la tête et rester assis.

4.3 Surface de mouvement

La surface de mouvement extrapole la trajectoire d'acquisition calculée par photogrammétrie. Notons (x_i, y_i, z_i) les positions de la caméra d'acquisition dans le système de coordonnées utilisé par le moteur de jeux pour le modèle 3D. Dans le cas d'Unity [6] que l'on utilise, l'axe Oy est vertical et pointe vers le ciel. Comme le sol est en gros horizontal, on définit la surface de mouvement par une fonction $y = f(x, z)$. Puis le joystick de gauche met à jour les valeurs de x and z, ce qui déplace la caméra virtuelle sur la surface de mouvement grâce à f. Soit

$$f(x,z) = a + \frac{\sum_i w_{ci} y_{ci}}{\sum_i w_{ci}} \text{ avec } w_i = e^{-\frac{\sqrt{(x-x_i)^2+(z-z_i)^2}}{b}}, \quad (1)$$

a un décalage vertical de la caméra virtuelle par rapport à la caméra d'acquisition, b est un facteur d'échelle qui règle la décroissance du poids w_i en fonction de la distance entre (x, z) and (x_i, z_i), et c est utilisé pour sauter des positions et accélérer le calcul de f à chaque image (en supposant que i croit avec l'instant d'acquisition). Les valeurs de a et b sont en mètres. La valeur de b est un compromis : suffisamment petit pour la précision d'extrapolation, suffisamment grand pour éviter des effets d'escalier. Un a non nul est utile pour éviter que les cubes gris occultent le champs de vue de l'utilisateur de RV.

On termine avec une remarque sur les systèmes de coordonnées. Celui calculé dans l'article est main-droite (comme d'habitude) avec un axe Oz vertical vers le ciel. Celui de Unity est main-gauche avec un axe Oy vertical vers le ciel. Il y a deux conséquences. Premièrement, il faut convertir la table de positions (x_{ci}, y_{ci}, z_{ci}) dans le programme C# dont Unity a besoin pour calculer f : il y a une rotation de $-\pi/2$ autour de l'axe Ox et un changement de signe de la coordonnée x pour passer de main-droite à main-gauche. On convertit donc (x_{ci}, y_{ci}, z_{ci}) en $(-x_{ci}, z_{ci}, -y_{ci})$. Deuxièmement, on utilise Unity pour tourner de $-\pi/2$ autour de l'axe Ox les modèles 3D, juste après leur importation dans Unity. Le changement de signe de x est fait implicitement ici. On obtient ainsi des positions d'acquisition et un modèle 3D consistants dans Unity.

5 Expérimentations

5.1 Jeux de données

Le premier modèle 3D "Basalt" est reconstruit à partir d'une vidéo 360 prise en marchant pendant 27 minutes dans un site géologique en utilisant une Gopro Max fixée sur un casque. Les deux côtés du chemin sont composés de prismes de basalte, dont les hauteurs sont d'environ 5-20m, et qui sont dégradés par la végétation et l'humidité. La texture est favorable à la photogrammétrie, à l'exception des quelques endroits avec une lumière faible due à l'étroitesse du canyon (parfois moins de 1m) ou avec de la végétation très proche de la caméra. Le chemin est principalement composé de rochers, qui peuvent être glissants et sont entourés de végétation basse. Son angle de pente est inférieur à $\pi/4$, à l'exception de quelques parties où la marche est remplacée par de l'escalade avec les mains ou une descente sur les fesses. Ce chemin a été sélectionné soigneusement avant l'acquisition, car le site est compliqué (une sorte de labyrinthe avec des obstacles et des parties dangereuses) et car la trajectoire doit inclure des boucles fermées de sorte que le Structure-from-Motion réduise les erreurs de dérive.

Nom	#tri	#loc	h	long.	pente-long.
Basalt	2.9M	3840	1.8m	819m	>30%,496m
City	3.4M	3132	1.55m	1.6km	>10%,240m

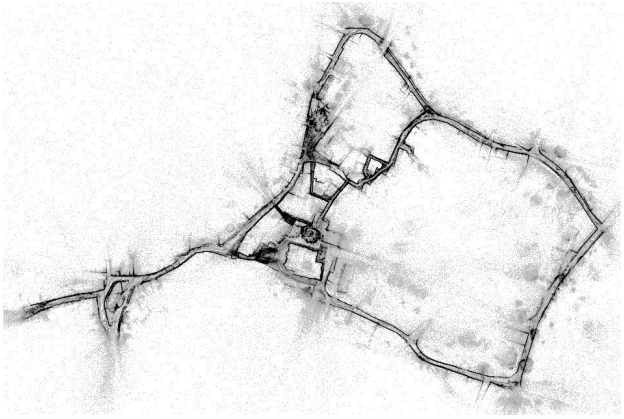
Table 1: Caractéristiques du jeux de données : nombres de triangles et de positions $\mathbf{l}_i$ (sélectionnées par le Structure-from-Motion), distance caméra-sol (i.e. hauteur de caméra) h, longueur de trajectoire, pente-longueur (ex. : pente d'au moins 10% sur 240m de la trajectoire d'acquisition).

Le second modèle "City" est reconstruit à partir d'une vidéo 360 prise en marchant pendant 21 minutes dans des rues d'un village médiéval en utilisant deux Gopro Max, qui sont rigidement montées de chaque côté de la tête près des oreilles (un seul fisheye par Gopro est utile). Ce modèle est intéressant pour expérimenter la méthode proposée sur un sol plus standard avec une pente modérée. Il inclue des petits chateaux, une église, des arbres et des porches. Le chemin est choisi avant l'acquisition grâce à une carte, de sorte à ce qu'il inclue plusieurs boucles. Les jours et heures d'acquisitions sont choisis de sorte à réduire le nombre de touristes dans les rues. (Les rues sont interdites à la plupart des voitures.) Comme pour le premier modèle, la météo est choisie de sorte à ce qu'il y ait suffisamment de lumière et que la végétation ne se déforme pas.

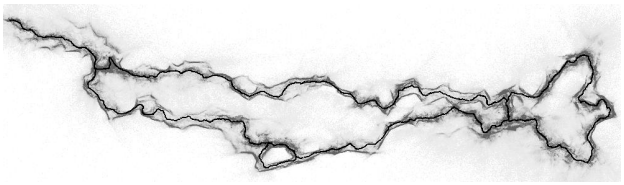
La table 1 donne des caractéristiques de Basalt et City, incluant les longueurs de trajectoires et les angles de pente, qui sont estimés grâce au changement proposé de système de coordonnées. Il faut noter que ces environnements ne sont pas triviaux : les longueurs de trajectoires d'acquisition sont de plusieurs centaines de mètres avec quelques millions de triangles (reconstruits avec [13] et [14], puis simplifiés) et des pentes non négligeables pour le sol. La fig. 2 montre des vues de dessus globales des points reconstruits et des trajectoires de caméra.

5.2 Comparaisons

On compare dans un premier temps des méthodes qui calculent la direction verticale à l'orientation près : l'ACP (sec. 3.1) et la transformée de Hough (sec. 3.2). On rappelle que la première initialise la seconde, et la première seule suffit pour un mouvement presque horizontal de caméra. Leurs résultats sont $\mathbf{v}_0$ et $\epsilon\mathbf{v}$, respectivement. La comparaison est faite par projection orthogonale du nuage de points reconstruit telle que la direction de projection est $\mathbf{v}_0$ ou $\epsilon\mathbf{v}$, et en examinant les composants de la scène



(a) Points de City



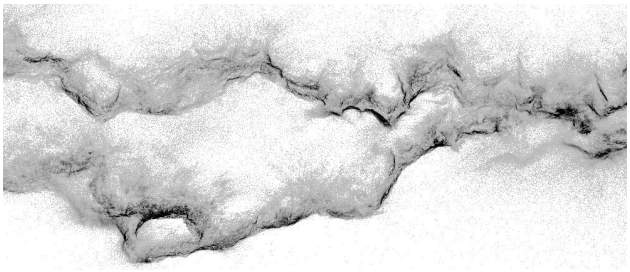
(b) Points de Basalt et trajectoire de caméra

Figure 2: Vues de dessus globales des points reconstruits de City et Basalt. Plus le niveau de gris est sombre, plus la densité de points est grande. La trajectoire de la caméra d'acquisition est aussi montrée en noir pour Basalt.

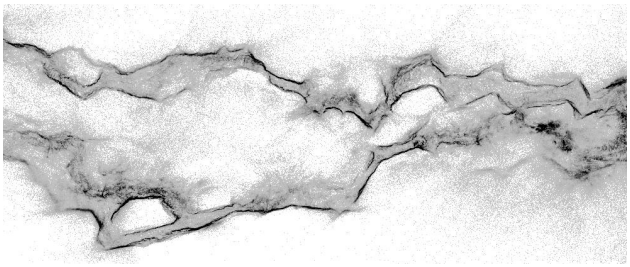
qui ont une surface verticale. Ces composants doivent être projetés en lignes ou courbes. La fig. 3 montre les projections de Basalt, qui est composé de couloirs (ou canyons) dont les deux côtés sont essentiellement verticaux. Les côtés du canyon sont plus parallèles à la direction estimée par la transformée de Hough qu'à celle estimée seulement par ACP, car de nouvelles courbes sombres apparaissent avec $\epsilon\mathbf{v}$. La comparaisons est moins évidente pour City car il vaut zoomer pour voir une différence entre ACP et Hough (fig. 4). On voit que les segments de droites sont moins bruités et plus sombres pour $\epsilon\mathbf{v}$ que pour $\mathbf{v}_0$. De plus, il y a d'autres segments de lignes (ou courbes pour Basalt) qui ont une largeur plus faible pour $\epsilon\mathbf{v}$ que pour $\mathbf{v}_0$. Plus cette largeur est faible, plus la verticale est précise. La conclusion est donc la même pour City et Basalt : $\epsilon\mathbf{v}$ est plus précis que $\mathbf{v}_0$. L'angle entre $\mathbf{v}_0$ et $\epsilon\mathbf{v}$ est égal à 18 degrés pour Basalt et 4 degrés pour City. Cela explique pourquoi la comparaison est plus facile pour Basalt que pour City.

Il y a une seconde façon de comparer qui montre que $\epsilon\mathbf{v}$ est plus précis que $\mathbf{v}_0$. Dans la fig. 5, chaque image du modèle 3D de City est affichée par une caméra perspective [8] avec un tangage nul et un roulis nul, par rapport à une direction verticale de référence qui est $\mathbf{v}_0$ ou $\epsilon\mathbf{v}$. Dans ce cas, les droites verticales en 3D sont projetées en droites verticales, si la direction de référence est une estimation précise de la verticale. C'est le cas pour $\epsilon\mathbf{v}$ mais pas pour $\mathbf{v}_0$.

Troisièmement, on discute les espaces d'accumulation de



(a) Projection parallèle à $\mathbf{v}_0$ (ACP)

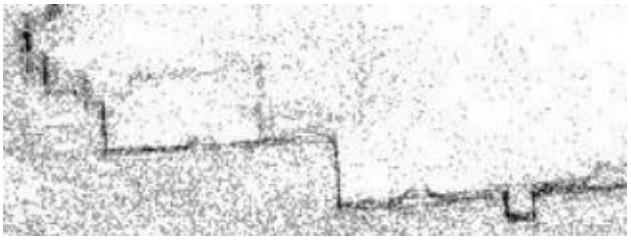


(b) Projection parallèle à $\epsilon\mathbf{v}$ (Hough)

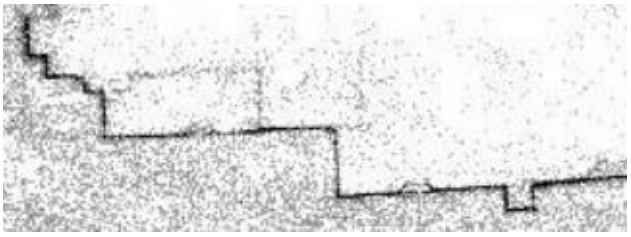
Figure 3: Vues de dessus de points de Basalt. Les côtés du canyon sont plus parallèles à la direction estimée par la transformée de Hough qu'à celle estimée par ACP.

Basalt et City grâce à la fig. 6. Ils sont utilisés par la transformée de Hough pour compter les votes pour les candidats verticaux. Les taches sombres ressemblent à des ellipsoides d'incertitudes et suggèrent que le calcul de $\epsilon\mathbf{v}$ est plus fiable pour City que pour Basalt. Cela est consistant avec le fait que les murs de basalte sont moins verticaux que les murs de bâtiments. La résolution de l'espace d'accumulation est un compromis : pas trop grand à cause du bruit, pas trop petit pour la précision. On utilise ici des images 100×100 pour un intervale d'angles $[-\pi/4, +\pi/4]^2$, et donc on a environ 0.9 pixel par degré. On voit aussi que l'espace d'accumulation de City est plus anisotropique que celui de Basalt : le premier a des bandes légèrement sombres qui sont presque orthogonales. Cela peut être expliqué de la façon suivante : les rues montrées dans la fig. 2 ont en gros deux directions orthogonales, les surfaces des murs et façades (qui sont parallèles ou orthogonales à leur rues) ont donc deux normales orthogonales qui votent dans les bandes.

Quatrièmement, la table 2 détaille les estimations de l'orientation et d'échelle en sec. 3.3. On rappelle que l'on distingue les distances caméra-sol et caméra-plafond dans $\{m_{-1}, m_{+1}\}$ en comparant m_{-1} et m_{+1}. Puisque $m_{+1} < m_{-1}$ pour City et Basalt, et grâce à l'hypothèse 3 (sec. 1), m_{+1} est la distance caméra-sol et les modèles 3D sont dilatés de façon à ce que m_{+1} devienne égal à h. La table donne aussi l'écart type σ_ϵ associée à la moyenne m_ϵ. On voit que l'hypothèse de hauteur constante (Sec. 1) est mieux vérifiée pour City que pour Basalt. Cela confirme l'observation : l'acquisition de City est toujours faite debout, alors que pour Basalt il faut parfois



(a) Projection parallèle à $\mathbf{v}_0$ (ACP)



(b) Projection parallèle à $\epsilon\mathbf{v}$ (Hough)

Figure 4: Vues de dessus de points de City. Les murs et façades sont plus parallèles à la direction estimée par Hough qu'à celle estimée par ACP.

Name	m_{-1}	σ_{-1}	m_{+1}	σ_{+1}	σ_{+1}/m_{+1}
Basalt	28.2	15.1	5.6	0.99	0.177
City	24.7	11.0	7.71	0.23	0.0298

Table 2: Moyennes et écarts types des distances caméra-sol et caméra-plafond dans le système de coordonnées donné en entrée. La distance caméra-sol est m_{+1}.

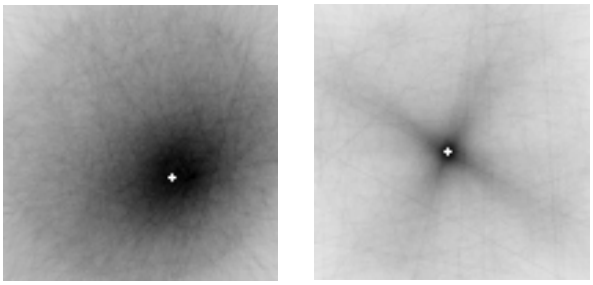
marcher accroupi pour passer sous des obstacles (branches d'arbres).

Enfin, les fig. 7 et 8 montrent des captures d'écrans prises par un casque de RV (Oculus Quest en mode autonome) pendant l'exploration (sec. 4.2) de Basalt et City, après le changement de système de coordonnées (sec. 3.4). La direction de normale pointant vers l'extérieur est aussi codée avec des couleurs : blanc ou noir pour la verticale vers le ciel ou le sol, rouge-vert-bleu pour horizontal. Le réglage de la surface de mouvement est ($a = +0.3$, $b = 1$, $c = 4$) pour Basalt et ($a = -0.2$, $b = 4$, $c = 10$) pour City. Des applications de RV pour huit scènes (dont les deux de l'article) et divers casques (avec un cable relié à un PC et compatibles avec Windows Mixed Reality ou OpenVR/SteamVR, ou bien des Oculus Quest/Go en mode autonome) sont récupérables sur internet [10, 11] pour des déplacements continus et discontinus.

5.3 Limitations de la méthode

Si les parties verticales (d'une scène) sont toutes parallèles, alors tous ses votes sont identiques le long d'un seul grand cercle de la sphère unité, et on ne peut pas estimer la verticale qui maximise les votes. Une scène plane par morceaux doit avoir au moins deux segments de plans

Figure 5: Vues de City par une caméra perspective avec tangage et roulis nuls. Les droites verticales en 3D sont projetées en droite verticales si la direction de référence verticale est $\epsilon\mathbf{v}$ (en bas), mais pas si c'est $\mathbf{v}_0$ (en haut).



(a) Accumulateur de Basalt (b) Accumulateur de City

Figure 6: Espace d'accumulation ($[-\pi/4, +\pi/4]^2$) de la transformée de Hough. Plus le niveau de gris est sombre, plus le vote est important. Les croix blanches montrent $\epsilon\mathbf{v}$. Le centre de l'espace d'accumulation est $\mathbf{v}_0$.

verticaux avec des normales non orientées distinctes. On évite donc les trajectoires dans une rue rectiligne dont les façades et murs sont tous parallèles à la rue.

6 Conclusion

Cet article estime la direction verticale et l'échelle d'une surface triangulée reconstruite par photogrammétrie à partir d'imagerie terrestre, avec des hypothèses explicites. La méthode est simple mais nouvelle : d'abord calculer la verticale non orientée avec une analyse en composantes principales de la trajectoire d'acquisition suivie par une transformée de Hough 2D, puis obtenir la verticale orientée et l'échelle en projetant la trajectoire sur la surface selon les deux directions verticales opposées possibles. L'article donne aussi des détails sur l'acquisition et la visualisation de deux environnements immersifs de RV (obtenus par photogrammétrie et la méthode proposée) : un canyon basaltique et un canyon urbain dont les longueurs de trajectoires sont 800m et 1.6km. Des applications de RV sont récupérables sur internet pour diverses scènes et casques.

References

[1] 3df zephyr. https://www.3dflow.net.

[2] Blender. https://www.blender.org.

[3] Ffmpeg. https://ffmpeg.org.

[4] Meshroom. https://alicevision.org.

[5] Metashape. https://www.agisoft.com.

[6] Unity. https://unity.com.

[7] B. Grayson, N. Penna, J. Mills, and D. Grant. Gps precise point positioning for uav photogrammetry. *The Photogrammetric Record*, 33(164):427–447, 2018.

[8] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision (chapter: 3D reconstruction of cameras and structure)*. Cambridge University Press, 2000.

[9] B. Klingner, D. Martin, and J. Roseborough. Street view motion-from-structure-from-motion. In *International Conference on Computer Vision*. Computer Vision Foundation, 2013.

[10] M. Lhuillier. Photogrammetric scans for vr (with continuous motions). https://maximelhuillier.fr.

[11] M. Lhuillier. Photogrammetric scans for vr (with discontinuous motions). https://steamcommunity.com/profiles/76561198051374313/myworkshopfiles/.

[12] M. Lhuillier. Fusion of gps and structure-from-motion using constrained bundle adjustments. In *Conference on Computer Vision and Pattern Recognition*. IEEE, 2011.

[13] M. Lhuillier. Surface reconstruction from a sparse point cloud by enforcing visibility consistency and topology constraints. *Computer Vision and Image Understanding*, 175, 2018.

[14] M. Lhuillier. Local-convexity reinforcement for scene reconstruction from sparse point clouds. In *International Conference on 3D immersion*, 2019.

[15] M. Lhuillier. From photogrammetric reconstruction to immersive vr environment. In *International Conference on 3D immersion*, 2021.

[16] J. Michot, A. Bartoli, and F. Gaspard. Bi-objective bundle adjustment with application to multi-sensor slam. In *International Symposium on 3D Data Processing, Visualization, and Transmission*, 2010.

[17] D. Newman and D. Stimm. This is gopro max: Tech, specs + more, 2019. https://gopro.com/en/us/news/max-tech-specs-stitching-resolution.

[18] T. Rabbani and F. Van Den Heuvel. Efficient hough transform for automatic detection of cylinders in point clouds. In *Laser Scanning*. ISPRS, 2005.

[19] N. Tian, R. Clement, P. Lopes, and R. Boulic. On the effect of the vertical axis alignment on cybersickness and game experience in a supine posture. In *IEEE Conference on Games*. IEEE, 2020.

[20] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment - a modern synthesis. In *Vision Algorithms: Theory and Practice*, pages 298–372. Springer-Verlag, 1999.

[21] T. Weissker, A. Kunert, B. Frohlich, and A. Kulik. Spatial updating and simulator sickness during steering and jumping in immersive virtual environments. In *IEEE Conference on Virtual Reality and 3D User Interfaces*. IEEE, 2018.

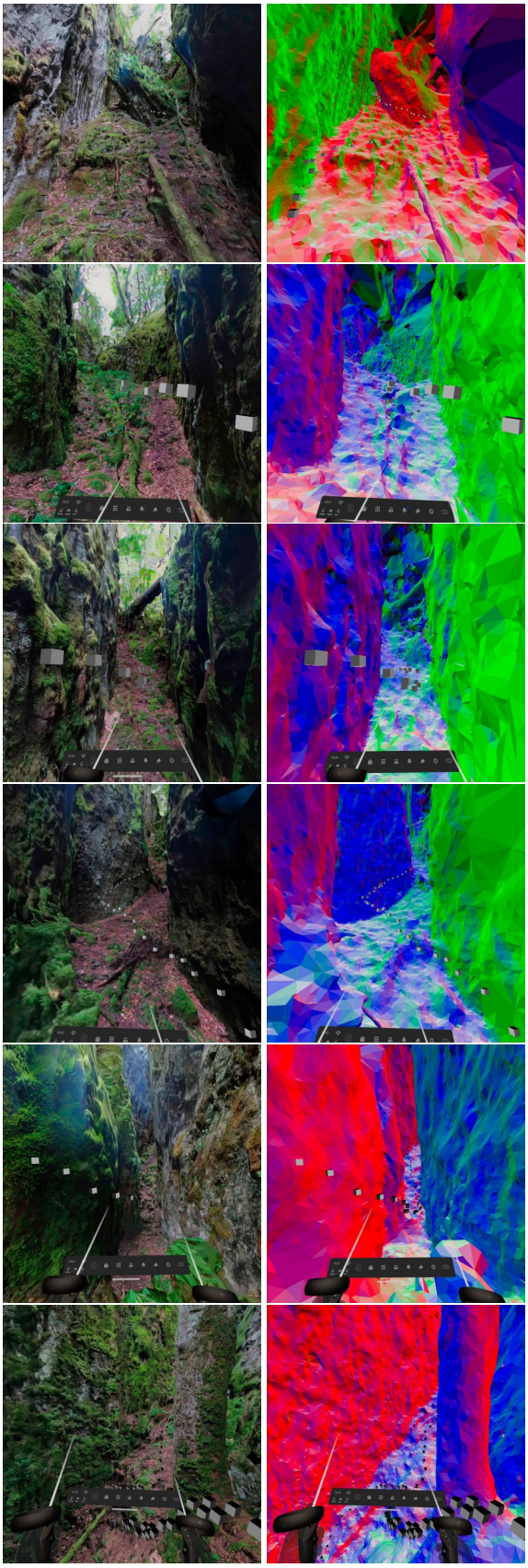


Figure 7: Captures d'écran de Basalt par un Oculus Quest.

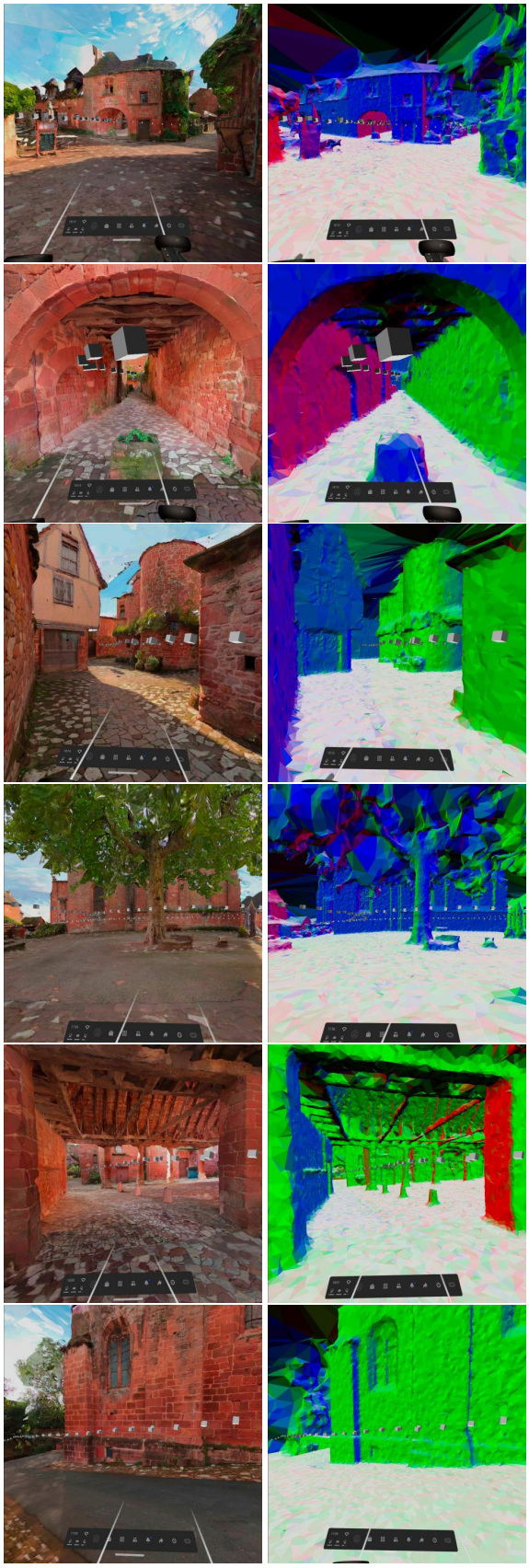


Figure 8: Captures d'écran de City par un Oculus Quest.