Chapitre 3

Problèmes d'acquisition

Dans cette thèse, nous nous intéressons au problème de modifications virtuelles des scènes réelles. Comme nous l'avons expliqué dans le chapitre précédent, ceci nécessite l'acquisition d'un modèle 3D texturé des scènes réelles. Dans ce chapitre, nous décrivons l'étendue du travail préliminaire et les différents choix que nous avons faits. Cette thèse n'est pas une thèse classique de synthèse d'image. Travailler avec la réalité présente de nombreuses difficultés auxquelles il faut faire face avant de s'attaquer à la résolution des problèmes de recherche qui nous intéressent. La réalité est l'environnement le plus complexe que l'on peut trouver ; le fait de le capturer oblige déjà à se restreindre et à appliquer des approximations. À chaque approximation, il y a une perte de l'information, ce qui peut amener à de nombreuses erreurs. Nous avons dans ce chapitre décrit les différents choix que nous avons faits. Nous espérons que cette description servira de bon point de départ à d'autres chercheurs désirant travailler sur le mélange du virtuel et de la réalité.

Pour avoir une connaissance de la géométrie de la scène réelle, nous avons décidé d'utiliser des outils de reconstruction à base d'images. Nous avons donc besoin de *capturer* l'environnement réel, pour en obtenir des images. Ceci implique de nombreux aspects techniques que nous décrivons dans ce chapitre. Parmi ces aspects, il y a celui du choix du matériel, tel que la caméra servant à la capture, ou les lampes éclairant la scène réelle. Ce choix de matériel a été guidé à la fois par des aspects de coût financier et par la simplicité d'utilisation.

Ensuite, nous avons eu à choisir et à utiliser des systèmes de reconstruction à base d'images. Nous en avons choisi deux. Le premier est un système développé par l'équipe Robotvis à Sophia–Antipolis, appelé *TotalCalib*, qui est en cours de commercialisation par la société *Realviz* [Rea]. Le deuxième système choisi est *Rekon*, un outil de reconstruction à base de contraintes développé dans le laboratoire d'informatique graphique (LIGUM) de l'Université de Montréal. Enfin, il a fallu traiter toutes ces données pour les rendre compatibles à nos systèmes de simulation.

Dans ce chapitre, nous décrivons tous les choix réalisés, ainsi que les difficultés générales auxquelles il faut faire face lorsque l'on travaille avec la réalité.

3.1 Les caméras utilisées

Avant de reconstruire un modèle géométrique de l'environnement réel, nous avons besoin de le *capturer* au moyen d'une caméra produisant des images de l'environnement réel, et de ramener ces données images sur un ordinateur. Nous avions plusieurs choix de capture possibles.

Nous pouvions capturer ces images avec un appareil photographique traditionnel, et numériser ces photographies. Dans ce contexte, la résolution et la qualité des photographies sont bien supérieures à celles d'appareils numériques. De plus, les prix des appareils traditionnels de bonne qualité sont abordables. En revanche, cette solution comporte de nombreux inconvénients. En effet, elle ne facilite pas la tâche des captures. En utilisant un appareil traditionnel, il faudrait prendre toutes les photographies, puis développer la pellicule pour savoir si les photographies correspondent vraiment à nos besoins, et enfin les essayer dans nos systèmes. Si les résultats se révélaient incomplets, il nous faudrait recommencer la série de captures. De plus, les variations des réactions chimiques lors du développement, ainsi que les compensations lors du développement des films font que les photgraphies ne sont pas véritablement représentatives de la réalité. Enfin, le coût engendré par les multiples développements nous a paru compensé par l'achat d'un appareil numérique.

À cause de ces inconvénients, nous avons choisi des caméras numériques. Dans notre objectif de départ, nous désirions développer un système qui fonctionnerait pour des environnements intérieurs. Nous avons tout d'abord choisi une caméra Sony EVI-D31, branchée directement sur une station de travail graphique SGI-O2 munie d'une carte de numérisation vidéo, pouvant être télécommandée en zoom ou en position autour de son axe. Cette caméra est en photographie sur la figure 3.1 (a). Elle est très adaptée à l'outil de reconstruction *TotalCalib* car elle est pratique pour créer des mosaïques, qui sont des données d'entrée au système de reconstruction. Le *déplacement télécommandé* et la possibilité de capture directement sur ordinateur nous ont paru être des avantages significatifs. Cependant, un des inconvénients est que nous utilisions cette caméra branchée sur ordinateur. De plus, la résolution des images (640×480) ne nous suffisait plus.

À cause des inconvénients cités ci-dessus, nous nous sommes également intéressés à d'autres appareils photgraphiques numériques, dont la technologie évolue rapidement. Afin de mieux diriger notre choix, nous avons essayé de définir clairement nos besoins. Parmi ceux-ci se trouvait notre souhait de produire des images de luminance (voir le chapitre 6), à partir d'images prises à différents temps d'exposition. De plus, nous voulions avoir une résolution de bonne qualité. Les appareils répondant à ces besoins sont des appareils manuels traditionnels, montés sur un système de capture numérique. Le coût de tels appareils était au–dessus de notre budget*. Nous avons donc restreint nos choix sur des appareils numériques semi–automatiques, ayant une bonne résolution d'image. L'appareil choisi est le Kodak DC260, montré en photographie sur la figure 3.1 (b). Cet appareil est programmable par des scripts. Les fonctions manuelles sont pour certaines accessibles uniquement au travers de ses scripts. Les scripts doivent être chargés sur l'appareil pho-

*en–dessous de 10,000 FF

(a) (b)

FIG. 3.1: Les deux appareils que nous utilisons pour prendre en photographie des scènes réelles. (a) Caméra Sony EVI-D31. (b) Appareil photographique Kodak DC260.

tographique sur la carte servant aussi à stocker les photographies. Ils sont écrits dans un langage pseudo–code spécifique. Nous avons utilisé ces scripts, et nous décrivons un exemple d'utilisation dans le chapitre 6. Cet appareil s'est révélé très pratique, bien que limité dans ses fonctions manuelles.

3.2 Reconstruction des scènes réelles à partir de photographies

Comme nous l'avons expliqué précédemment, nous avons besoin d'obtenir un modèle géométrique de la scène réelle afin de l'utiliser dans nos systèmes. Dans l'état de l'art (section 2.4.2), nous avons présenté différentes méthodes pour reconstruire des scènes réelles à partir d'images. Ces méthodes pouvaient être classées en deux grands groupes : les méthodes automatiques, et les méthodes semi-automatiques. Nous avons sélectionné deux méthodes semi-automatiques afin d'avoir un meilleur contrôle sur le résultat. De plus, les systèmes choisis sont des systèmes de recherche. Nous désirions ainsi bénéficier d'une collaboration entre les équipes pour adapter les systèmes selon nos besoins. Le premier système utilisé est l'outil de reconstruction *TotalCalib* [BR97] de l'équipe ROBOTVIS de Sophia–Antipolis. Cet outil est basé sur des concepts de vision 3D. Nous le présentons dans la section suivante. Le deuxième outil choisi est le système *Rekon* [POF98] développé dans le laboratoire LIGUM à Montréal. Ce système est basé sur l'intervention de l'utilisateur, qui doit fixer des contraintes géométriques dans le système afin de l'aider dans la résolution. Alors que *TotalCalib* utilise des points et des contraintes épipolaires, l'outil *Rekon* utilise des points, des lignes et des polygones. Le calibrage dans *Rekon* est aidé par des contraintes géométriques telles que le parallélisme, la perpendicularité, et la co–planarité. Nous présentons le système *Rekon* dans la section 3.2.2. Le système *Rekon* a une approche très similaire de l'outil de reconstruction *Façade* [DTM96].

3.2.1 Un premier outil de reconstruction : *TotalCalib*

Le premier outil de reconstruction que nous avons utilisé s'appelle *TotalCalib*. Ce système de reconstruction a été développé par l'équipe ROBOTVIS à Sophia–Antipolis. Son implantation [FLR+97] est basée sur les principes de la vision 3D, et utilise différentes techniques : le calibrage automatique de la caméra [Rob95], la construction de mosaïques [ZFD97], la géométrie épipolaire qui permet d'obtenir un modèle géométrique de la scène [BR97], et enfin l'extraction de textures dépendantes du point de vue.

Le calibrage de la caméra est réalisé en dehors de *TotalCalib*, par une technique développée par Robert [Rob95]. Ce calibrage est semi–automatique. L'utilisateur donne pour six points de référence, leurs coordonnées dans un espace 3D, et leurs coordonnées sur l'image. Ces points doivent être distribués sur une mire de calibrage, comme par exemple sur l'image 3.2. Ces six coordonnées sont ensuite utilisées par le système pour déterminer les paramètres intrinsèques et extrinsèques de la caméra.

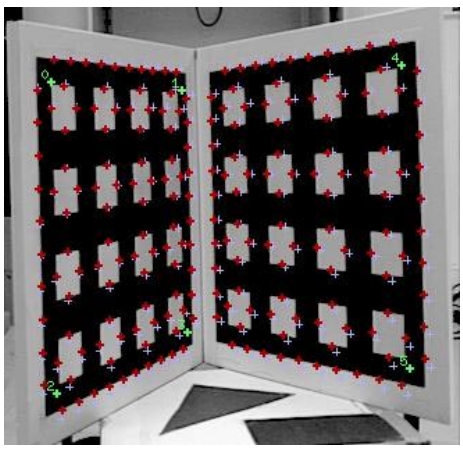


FIG. 3.2: Mire de calibrage, et les six points d'entrée pour retrouver les paramètres de la caméra. (Figure extraite de la page web : `http ://www-sop.inria.fr/robotvis/personnel/lucr/detecproj.html`)

Ensuite, des mosaïques sont reconstruites pour une séquence de trois images [ZFD97]. Un exemple de quatre mosaïques servant à reconstruire un modèle est montré sur la figure 3.3. Pour reconstruire ces mosaïques, un ensemble de 12 images est nécessaire (3 images par mosaïque). *TotalCalib* n'impose pas des mosaïques en entrée. Cependant, l'utilisation de mosaïques permet une reconstruction plus large, permettant d'avoir des polygones couvrant plus qu'une seule image (notamment l'image centrale).

Ces mosaïques sont passées en entrée dans le système *TotalCalib*, qui va permettre de reconstruire le modèle 3D des objets visibles sur ces mosaïques. Ce système montré sur la figure 3.4 combine différentes techniques de vision par ordinateur. L'utilisateur entre les points en cliquant sur une image avec la souris. Pour les premiers points, la correspondance de points est réalisée par l'utilisateur qui associe les points correspondant au même point 3D entre eux. Ensuite, le système est capable de retrouver automatiquement les points correspondants sur les trois autres mosaïques, en utilisant la matrice de projection et la recon-

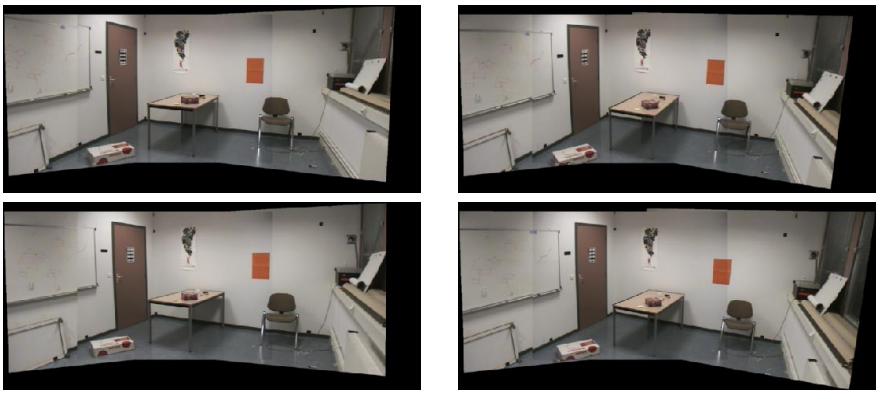


FIG. 3.3: Les quatre mosaïques servant d'entrée au programme TotalCalib.

naissance de discontinuités locales dans l'image. Avec à peu près trente points entrés et mis en correspondance sur les images, le système est capable de retrouver les matrices fondamentales (matrices de projection et les matrices de paramètres intrinsèques), en utilisant une méthode non–linéaire [ZDFL95]. Les polygones sont ensuite reconstruits manuellement, en reliant les points entre eux. Les coordonnées 3D de ces polygones sont retrouvés par le système en utilisant la matrice de projection. Sur la figure 3.4, le système *TotalCalib* est illustré, avec les quatre mosaïques servant à la reconstruction. Sur la fenêtre en haut à gauche, les polygones ont été entrés par l'utilisateur.

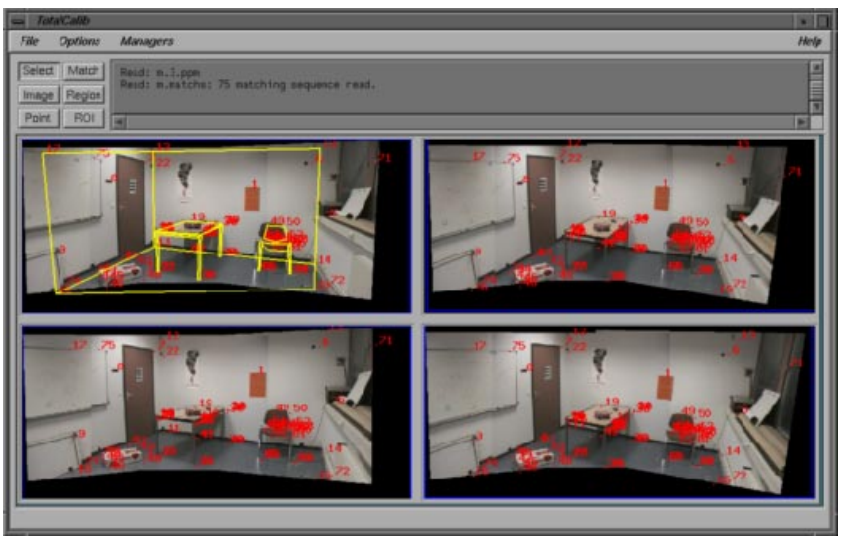


FIG. 3.4: Le système *TotalCalib*. Dans la fenêtre en haut à gauche, des polygones ont été créés à la main par l'utilisateur, en reliant des points précédemment introduits. Ces points ont servi au calibrage des images et pour retrouver les matrices de projection pour chacune des mosaïques.

Les paramètres intrinsèques ne sont pas toujours correctement retrouvés par le système. Dans ce cas, nous utilisons les paramètres intrinsèques préalablement retrouvés en utilisant

le calibrage automatique de la caméra à l'aide d'une mire. Un exemple de modèle reconstruit par *TotalCalib* est présenté sur la figure 3.5.

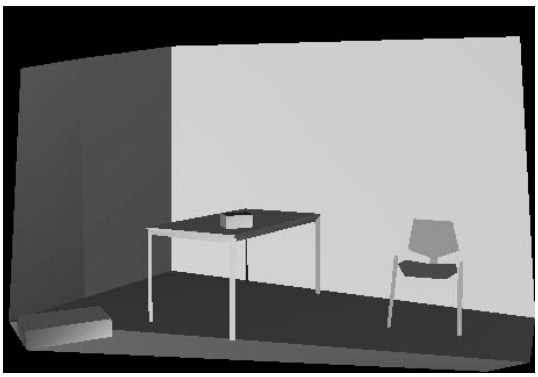


FIG. 3.5: Modèle reconstruit avec le système *TotalCalib*, correspondant aux images présentées sur la figure 3.3.

Des textures dépendantes du point de vue sont extraites [FLR+97] à partir du modèle reconstruit, d'une des mosaïques, et des matrices de projection. Pour chaque polygone reconstruit, l'image originale (mosaïque) est détordue et ré–appliquée sur le polygone. Le modèle texturé est montré sur la figure 3.6 (a). En (b), le modèle texturé est montré depuis un point de vue différent de celui de l'image d'origine. Les textures ainsi extraites contiennent les images des objets qui sont entre le polygone et le point de vue.

(a) (b)

FIG. 3.6: (a) Modèle texturé construit avec *TotalCalib*. (b) Même modèle mais sous un autre point de vue. Cette image met en évidence que les textures sont dépendantes du point de vue.

Comme nous l'avons décrit, le système est semi–automatique. L'intervention de l'utilisateur est réduite à l'introduction de points dans *TotalCalib*, ainsi qu'à la création des polygones sur l'image. Le modèle géométrique ainsi reconstruit représente correctement la réalité. Cependant, le système ne garantit pas les propriétés géométriques, telles que la planarité d'un polygone, la perpendicularité entre polygones ou le parallélisme.

Mise en place

Nous avons exposé ci-dessus la description technique de la méthode de reconstruction. Nous décrivons plus en détail ici, comment nous avons utilisé ces techniques.

Dans un premier temps, nous avons calibré notre caméra, pour une distance focale donnée. Pour toutes les captures, nous avons utilisé la même distance focale (fixée à l'infini). Les mires de calibrage couramment utilisées en vision sont assez chères. Nous avons donc construit notre propre mire, avec un support carton, et un schéma régulier créé par ordinateur et imprimé sur du papier. Cette mire est présentée sur la figure 3.7.

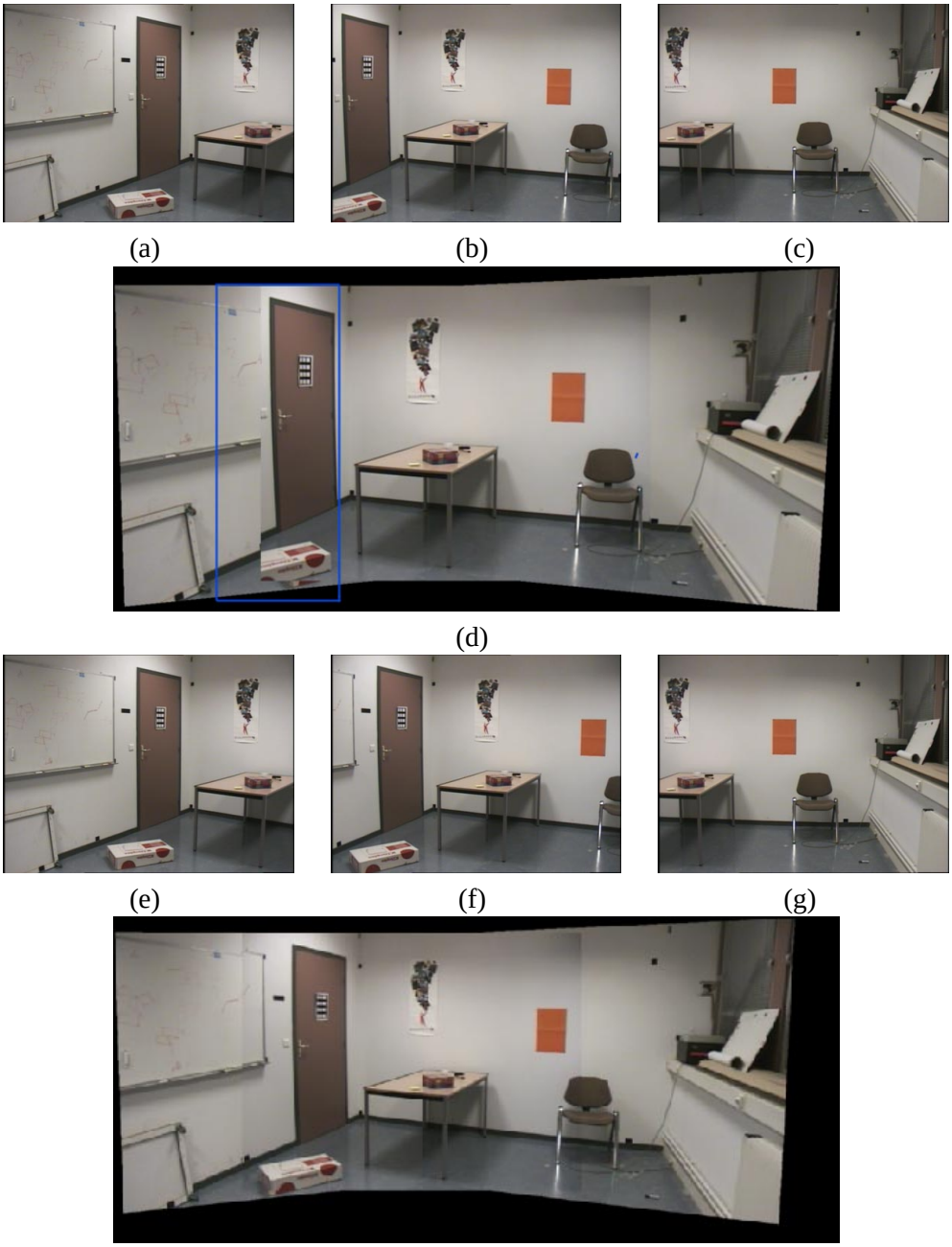
FIG. 3.7: Notre mire de calibrage, construite avec un socle en carton, sur lequel sont fixés deux schémas de grille.

Nous avons ensuite mesuré les six points requis par le programme de calibrage, qui a déterminé les paramètres de notre caméra.

La capture des scènes se faisait ensuite dans les locaux du laboratoire. Comme le montreront les chapitres suivants, nous avions besoin d'avoir une scène close, pour être sûrs d'avoir un éclairage global cohérent. N'ayant ni rideaux opaques, ni stores masquant vraiment la lumière extérieure, les captures se faisaient lorsqu'il faisait nuit. La capture pour le système *TotalCalib* a été effectuée par la caméra Sony présentée dans la section 3.1. L'ordinateur sur lequel était branchée la caméra devait être déplacé pour se trouver dans la salle où nous réalisions les captures.

Pour capturer les mosaïques, nous prenons trois images successives, en télécommandant la caméra pour une rotation horizontale. La capture des mosaïques s'est révélée longue. L'ajustement des images entre elles ne fonctionnait pas à chaque fois, et nécessitait souvent plusieurs essais. Dans la figure 3.8, nous montrons un exemple d'échec d'ajustement des images pour créer une mosaïque. Les trois premières images utilisées, (a), (b), et (c) n'ont pas permis de reconstruire une mosaïque correcte en (d) (voir la région encadrée). Les images ne se recouvraient pas assez. Nous avons repris une nouvelle photographie pour l'image centrale en (f), qui nous a permis d'obtenir une meilleure mosaïque en (h).

Une fois les mosaïques reconstruites, nous utilisons le système *TotalCalib* pour reconstruire la scène. *TotalCalib* s'est révélé être un outil très pratique. Les reconstructions étaient relativement rapides (quelques heures). Il nous a cependant fallu du temps pour



(a) (b) (c)

(d)

(e) (f) (g)

(h)

FIG. 3.8: (a), (b) et (c) ont servi à crée la mosaïque (d). Cette mosaïque n'a pas été ajustée correctement sur la gauche (voir la région encadrée), car les images (a) et (b) ne se recouvrent pas suffisamment. (e), (f), et (g) ont servi à la création de la mosaïque (h), qui cette fois est très bien ajustée. Pour réussir l'ajustement de cette mosaïque, l'image du centre (b) a été remplacée par l'image (f). Les autres images (a) et (e), et (c) et (g) sont identiques.

acquérir les compétences nécessaires, et prendre l'habitude de bien distribuer les points dans les images, ou prendre les mosaïques d'entrée sous des points de vue suffisamment différents. Durant cette phase d'apprentissage, la reconstruction s'étendait alors à plusieurs jours. Pour améliorer le modèle, il fallait parfois reprendre de nouvelles photographies, avec de meilleurs repères et plus d'objets pour varier le placement des points. Le calibrage de la caméra avec la mire s'est avéré très utile. Nous avons pu aider *TotalCalib* à calibrer notre scène.

Un mauvais calibrage avait souvent pour conséquence une mauvaise correspondance entre les textures extraites et les polygones. Étant donné que le modèle est regardé sous un seul point de vue, cela pouvait être acceptable dans une certaine mesure, car l'image générale restait cohérente.

Ce système nous a permis de reconstruire des scènes réelles avec des défauts de reconstruction minimes. La reconstruction générale se fait en plusieurs étapes. Nous devions recommencer du début si par exemple nous n'arrivions pas à calibrer correctement le système en retravaillant la scène, ou en prenant de nouvelles photographies sous un autre point de vue. En général, la capture complète de la phase d'arrangement de la scène, puis la prise de photographies, jusqu'à l'obtention du modèle, nous prenait une semaine. *TotalCalib* est en cours de commercialisation par la société *Realviz* [Rea]. Cet outil n'est plus un outil de recherche, et sa configuration, notamment son interface, est plus adaptée à un utilisateur inexpérimenté.

3.2.2 Un deuxième outil de reconstruction : *Rekon*

Nous avons également travaillé avec un autre outil de reconstruction, qui est le système *Rekon* développé à l'Université de Montréal, dans le laboratoire d'informatique graphique (LIGUM). Il est différent du système *TotalCalib* sur plusieurs plans. *TotalCalib* se veut le plus automatique possible. L'intervention de l'utilisateur est restreinte. *Rekon* au contraire demande souvent à l'utilisateur d'intervenir. Cette intervention permet de réduire le nombre de points nécessaires à une première reconstruction. Alors que *TotalCalib* nécessite une trentaine de points, *Rekon* en demande six. Ces six points doivent cependant être connus parfaitement de l'utilisateur, et si possible non planaires. Ce peut être par exemple des points placés sur les coins d'une boîte. L'utilisateur entre les coordonnées de ces six points à la main. Le système utilise cette correspondance entre le 2D et le 3D pour calibrer la caméra et construire une première matrice de projection. Ce premier calibrage peut être comparé au calibrage sur une mire, comme nous l'avons fait pour *TotalCalib*. Pour réaliser une reconstruction, le système a besoin que pour chaque image, six points soient calibrés. Ce peut être soit des points dont les coordonnées 3D ont été entrées par l'utilisateur, soit des points déjà calibrés dans les autres images. Ce qui veut dire que nous devons avoir deux images au moins contenant les six points de départ entrés par l'utilisateur. On peut ensuite ajouter des nouveaux points 2D sur des images calibrées, et leurs coordonnées 3D seront calculées par le système. Dans la phase de calibrage, l'utilisateur peut entrer des points, des polygones ou des lignes. La mise en correspondance des polygones permet de les étiqueter, et de savoir que le même polygone se trouve sur plusieurs images. Pour obtenir un meilleur calibrage et une reconstruction de meilleure qualité, *Rekon* a besoin d'images d'entrée dont le point de vue est très différent.

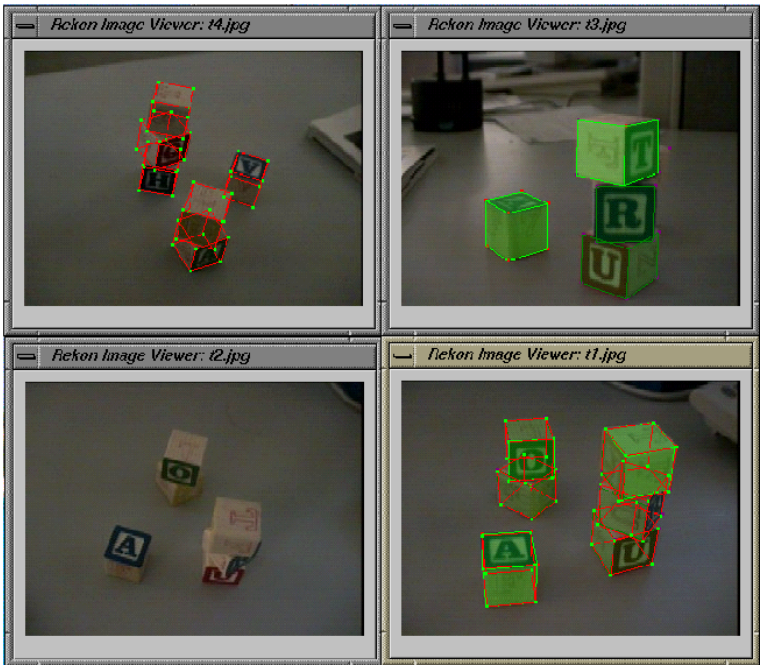


FIG. 3.9: Le système *Rekon*, avec quatre images sur lesquelles l'utilisateur construit un modèle. Le modèle 2D dessiné par l'utilisateur est montré par exemple sur l'image en haut à droite. Le modèle 3D reprojeté sur l'image est montré par exemple en haut à gauche et en bas à droite. (Figure extraite du mémoire de Frasson [Fra99])

Pour améliorer le calibrage, l'utilisateur *aide* le système en lui indiquant les contraintes géométriques qui existent dans la scène. Ce sont des contraintes de planarité, de co–planarité, de perpendicularité et de parallélisme. La contrainte de co–planarité peut s'appliquer entre deux polygones, ou entre un point et un polygone. Cette contrainte est très pratique lorsqu'on veut indiquer au système que deux polygones se touchent. L'utilisateur peut ajouter autant d'images dans le système, qu'il lui en faut pour reconstruire les polygones de la scène. Le système itére autant de fois que c'est nécessaire pour atteindre une convergence.

L'extraction des textures est intégrée directement dans le système. Les textures construites sont indépendantes du point de vue. L'utilisateur choisit un polygone sur une image, et le système sélectionne automatiquement toutes les images où le polygone a été reconstruit. Le système extrait des textures sur chacune des images, et les combine en les pondérant par une valeur de validité. Cette validité est calculée en tenant compte de la visibilité depuis l'œil, de l'angle de vue, et d'autres paramètres.

Le système *Rekon* est comme *TotalCalib* un système complet, qui fournit un modèle géométrique texturé. L'intervention de l'utilisateur est plus importante cependant que pour le système *TotalCalib*.

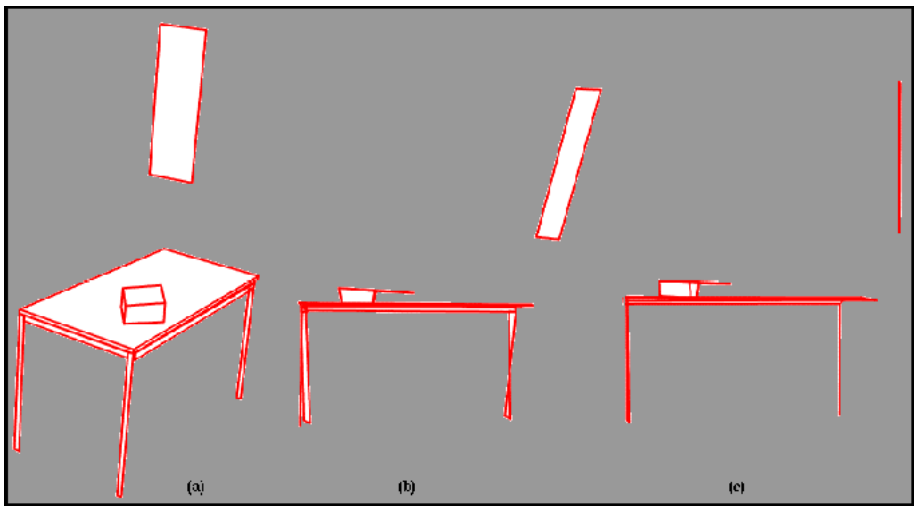


FIG. 3.10: À gauche, on voit la scène reconstruite sous un point de vue. Au milieu, on voit la scène reconstruite sous un point de vue différent mettant en évidence les erreurs de reconstruction. À droite, le système de reconstruction a pris en compte des contraintes géométriques qui ont permis d'améliorer la reconstruction. (Figure extraite de la thèse de Marie–Claude Frasson [Fra99])

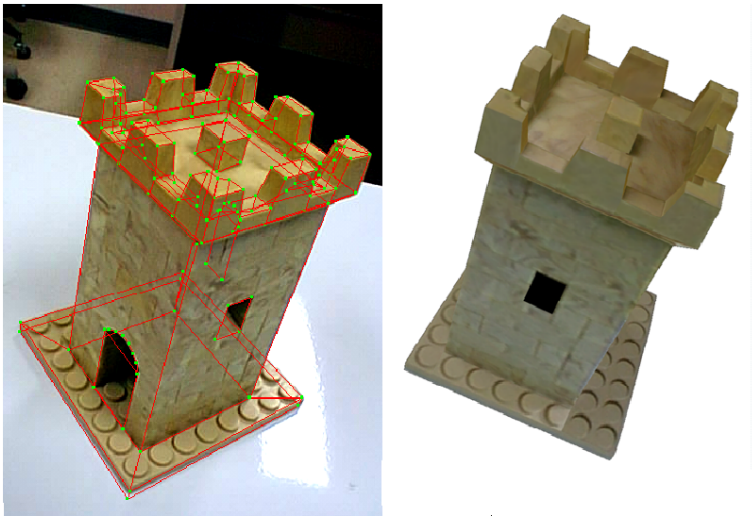


FIG. 3.11: Exemple de modèle reconstruit par *Rekon*. À gauche, le modèle est reprojeté sur l'image lors de la reconstruction. À droite, le modèle est placé sous un nouveau point de vue. Les textures ont été extraites à partir de plusieurs images. Elles sont indépendantes du point de vue. (Figure extraite du mémoire de Frasson [Fra99])

Mise en place

Comme nous l'avons fait pour *TotalCalib*, nous décrivons maintenant notre utilisation de *Rekon*.

La reconstruction d'une scène en utilisant *Rekon* est assez différente de *TotalCalib*. Elle nécessite une connaissance préliminaire de certaines mesures dans la scène. Comme pour *TotalCalib*, nous avons eu à reprendre des photographies pendant la phase de reconstruction, et à conserver la scène quelques jours en état. Les problèmes que nous avons rencontrés étaient divers. Il est assez difficile pour un utilisateur de visualiser les points communs entre les images lors de la prise de photographies. Il est arrivé d'avoir suffisamment de photographies pour visualiser les objets, mais pour lesquels les points communs n'étaient pas assez nombreux. La prise de nouvelles photographies peut cependant être évitée, si on connaît les mesures de certains points. La reconstruction ne prenait que quelques heures. Cependant, si des problèmes apparaissaient, nous pouvions atteindre quelques jours, le temps de reprendre les photographies, ou de nouvelles mesures.

Le système est assez robuste. Les modèles reconstruits sont assez justes, même si l'utilisateur place un point 2D à quelques pixels de sa position exacte. L'algorithme permet une correction des erreurs de l'utilisateur dans le système. Par contre, certaines erreurs de coordonnées 3D, de mauvaises correspondances, ou de mauvaises contraintes géométriques ont une influence assez grande dans le système. Comme l'utilisateur doit intervenir assez souvent, il lui est facile d'introduire des erreurs d'inattention qu'il est ensuite difficile de localiser.

Enfin, nous avons ajouté certaines fonctionnalités dans le système *Rekon*, pour nous permettre d'obtenir le modèle texturé tel que nous en avions besoin. Tout d'abord, l'extraction de textures dépendantes du point de vue a été implantée par Frasson dans le système. Ensuite, les matrices de *Rekon* ont été transformées en des matrices de paramètres de la caméra, que nous savions transformer pour obtenir un point de vue dans nos systèmes (voir section 3.4.1).

3.3 Choix de l'éclairage

Notre objectif est d'estimer les propriétés radiométriques des scènes réelles, afin de modifier leur éclairage. L'estimation des paramètres radiométriques des scènes réelles est un problème très difficile. Cette complexité est due à la variété des types d'éclairage, ainsi qu'à celle des propriétés des surfaces (BRDF complexes). De plus, l'éclairage dépend de la configuration de l'environnement, qui implique plus ou moins d'éclairage indirect. L'état de l'art dans le domaine de l'illumination inverse est très jeune. Il paraît plus judicieux de s'attaquer à des environnements moins complexes afin d'élaborer des méthodes de base et de comprendre les problèmes, avant d'essayer de travailler sur des environnements très complexes.

Nous avons donc fait certains choix sur l'arrangement de nos scènes. Nous avons privilégié d'abord les scènes d'intérieur fermées. Nous prenions soin de n'avoir qu'un éclairage connu, et non un éclairage provenant de l'extérieur comme par des fenêtres, ou par une

porte. Nous avons choisi de préférence des scènes éclairées directement par les lampes, plutôt que par un éclairage indirect. En effet, il est en général plus facile d'estimer des valeurs pour l'éclairage direct que pour l'éclairage indirect. De plus, nous avons choisi des lampes ayant une intensité homogène, pour simplifier notre estimation de l'intensité des lampes. Dans la première méthode de ré–éclairage, présentée au chapitre 4, nous avons utilisé les lampes néons déjà en place. Le support de ces lampes est rectangulaire et il est fixé au plafond. Pour notre deuxième méthode, expliquée au chapitre 5, nous avons utilisé une lampe de jardin, facile à modéliser, car son support est rectangulaire. Cette lampe est une lampe halogène. Pour que l'éclairage de cette lampe s'approche plus à une émission diffuse, nous avons fixé un papier de soie sur le support (voir figure 3.12).

(a) (b)

FIG. 3.12: (a) Lampe utilisée pour éclairer les scènes lors de nos tests. (b) La lampe est recouverte d'un papier de soie, permettant de se rapprocher du caractère diffus.

Enfin, nous avons essayé d'avoir des objets les plus diffus possibles sans pour autant nous limiter à de tels objets. Dans la plupart de nos scènes, les propriétés des surfaces sont très variées.

3.4 Intégration des modèles 3D dans le système de rendu

Nous avons pu construire un modèle 3D texturé d'une scène réelle grâce aux deux outils présentés ci–dessus. Ces deux outils nous ont permis d'obtenir un modèle au format Inventor®, avec des textures RGB, que nous pouvons lire et utiliser dans nos programmes

graphiques. Nous avons cependant réalisé quelques adaptations pour rendre compatibles les données des outils de reconstruction avec celles de nos programmes.

3.4.1 Les matrices de points de vue

Les premières données à adapter sont les matrices de projection fournies par les outils de reconstruction. Les deux matrices fournies sont la matrice de paramètres intrinsèques, et la matrice de paramètres extrinsèques [Fau93]. La matrice de paramètres intrinsèques fournit des indications sur la distance focale, et la projection du centre de la caméra sur l'image. Ces paramètres dépendent uniquement des réglages de la caméra. La forme de la matrice de paramètres intrinsèques est la suivante :

$$\begin{pmatrix} A_u & \gamma & u_0 \\ 0 & A_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{3.1}$$

où A_u, A_v, et γ sont des paramètres indiquant la distorsion d'un pixel, et la distance focale, et u_0 et v_0 sont la projection du centre optique sur l'image. La matrice des paramètres extrinsèques fournit la position relative de la caméra, en termes de rotations (matrice R 3×3) et de translation (vecteur T 3×1). Cette matrice est évidemment différente pour chacune des images utilisées pour la reconstruction. La forme de la matrice est la suivante :

$$\begin{pmatrix} R & T \\ 0 & 1 \end{pmatrix}. \tag{3.2}$$

Les deux matrices fondamentales qui nous intéressent doivent être celles associées à l'image utilisée pour extraire les textures.

Dans nos programmes, nous utilisons la librairie graphique OpenGL®pour réaliser le rendu. Dans OpenGL®, les paramètres utilisés pour définir le point de vue ne sont pas les matrices de vision. Il faut donc convertir les données des matrices typiquement utilisées en vision par ordinateur, décrivant les paramètres intrinsèques et extrinsèques, en paramètres acceptables par OpenGL®.

Les paramètres correspondant aux paramètres intrinsèques sont ceux de *gluPerspective* et *glViewport*.

glViewport a quatre paramètres. Les deux premiers (x,y) définissent le coin en bas à gauche de la fenêtre de vue. Les deux autres $(width, height)$ définissent la taille de la fenêtre de vue. La fenêtre de vue peut se déduire des paramètres intrinsèques. Idéalement, on a :

$$x = \frac{-u_0 \times width}{2} \text{ et } y = \frac{-v_0 \times height}{2}. \tag{3.3}$$

Cette équation provient d'une transformation du repère où sont calculées les coordonnées (u_0, v_0) du centre de la caméra, dans le repère de la fenêtre de vue. Le repère du centre de la caméra varie en abscisses et en ordonnées entre -1 et 1. Le repère associé à la fenêtre varie en abscisses entre 0 et *width*, et en ordonnées entre 0 et *height*. Une fois le changement de repère effectué, il faut appliquer le fait que (u_0, v_0) correspond au centre de la fenêtre de visualisation, et (x,y) au coin en bas à gauche. Les signes négatifs proviennent du fait que

les axes des deux repères ne sont pas orientés de la même façon. Pour avoir une bonne impression lors de la visualisation de la scène, il est préférable d'avoir le rapport $\frac{width}{height}$ égal à celui de la taille de l'image originale.

En réalité, nous n'avons jamais exactement utilisé (x, y) calculés par l'équation (3.3). Cependant, ceci nous donnait un point de départ, et nous ajustions ensuite les paramètres (x, y) directement et manuellement dans le système, jusqu'à que la scène 3D reprojetée soit alignée avec l'image 2D. Nous pensons que ces erreurs sont dues soit aux inexactitudes du calibrage, soit à l'idéalisation de la caméra par OpenGl®.

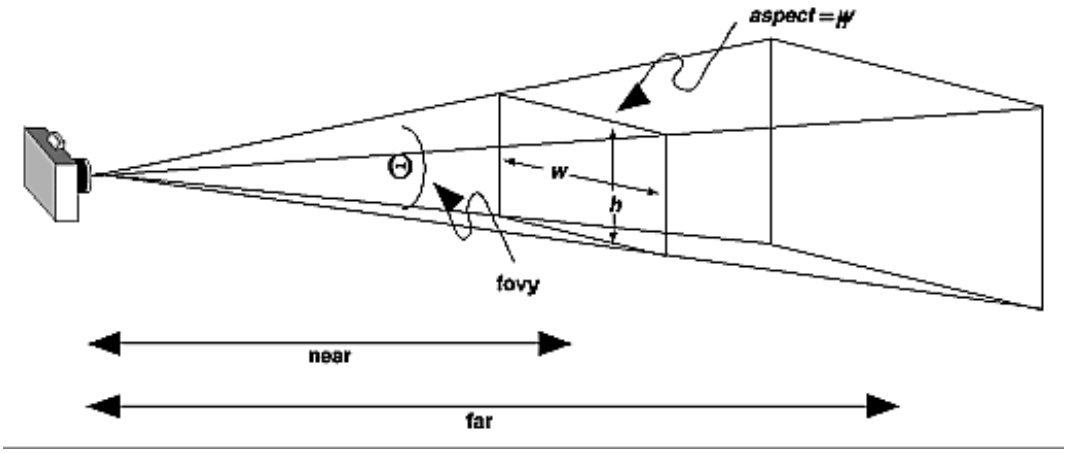


FIG. 3.13: Schéma illustrant les paramètres nécessaires pour simuler la perspective dans nos systèmes graphiques. (Figure extraite du manuel de la librairie graphique OpenGL®) [WNDS99]

Souvent en vision, on ne s'intéresse pas à la vraie profondeur 3D, alors qu'elle est nécessaire dans un système de synthèse d'image. Une façon de la prendre en compte est de donner des valeurs de profondeur de champ. *gluPerspective* permet de fixer les plans de profondeur devant et derrière la scène virtuelle (*near* et *far* sur la figure 3.13). Il faut aussi déterminer l'aspect de la fenêtre, ce qui est simplement donné par $\frac{width}{height}$. Enfin, il faut déterminer le champ de vue ($fovy$, pour *field of view in y* sur la figure 3.13). Ceci se fait en fonction de vecteurs placés sur le rectangle de la fenêtre dans la position la plus proche avant coupure (*near*). Ces vecteurs sont calculés en fonction de la distance focale de la caméra accessible dans la matrice de paramètres intrinsèques, dans la sous–matrice (2×2) de la matrice 3.1 :

$$\begin{matrix} A_u & \gamma \\ 0 & A_v \end{matrix} \tag{3.4}$$

Les coordonnées au milieu de chaque arrête de la fenêtre sont :

$$\begin{aligned} left &= \tfrac{-near}{A_u} \\ right &= \tfrac{-near}{A_u} \\ bottom &= \tfrac{(near \times -\gamma)}{A_v} \\ top &= \tfrac{(near \times \gamma)}{A_v}. \end{aligned} \tag{3.5}$$

Les deux vecteurs utilisés pour calculer l'angle du champ de vue sont :

$$\begin{array}{l} V_1(\frac{right+left}{2}, top, -near) \\ V_2(\frac{right+left}{2}, bot, -near). \end{array} \tag{3.6}$$

L'angle du champ de vue est déterminé par son cosinus donné par le produit scalaire des deux vecteurs normalisés V_1 et V_2.

Les paramètres extrinsèques permettent de positionner la caméra dans l'espace. À partir de la matrice de paramètres extrinsèques, nous avons extrait trois angles de rotations, et un vecteur de translation. Pour extraire les angles, nous utilisons les neuf premiers paramètres de la matrice de paramètres extrinsèques, c'est–à–dire la sous–matrice R (3×3) en haut à gauche de la matrice générale (4×4) de l'équation (3.2). Les trois colonnes de la matrice (3×3) nous indiquent les axes du repère obtenus après rotation. Pour chacun de ces axes, nous avons extrait trois vecteurs normalisés. Ensuite, nous avons obtenu le cosinus des angles en calculant le produit scalaire entre ces vecteurs et les vecteurs originaux $((1,0,0),(0,1,0),(0,0,1))$. Les angles ainsi retrouvés nous indiquent la direction dans laquelle la caméra regarde la scène. À partir de la dernière colonne T de la matrice (4×4), nous obtenons la translation déterminant la position de l'œil dans l'espace. Afin d'obtenir la translation dans le repère original, il faut appliquer la transformation de rotation au vecteur de translation.

En utilisant les paramètres intrinsèques et extrinsèques, nous avons déterminé les paramètres de vue, positionnant l'œil dans l'espace virtuel, et appliquant la perspective et la fenêtre de vue correspondant à notre caméra.

3.4.2 Modèle Inventor

Les modèles 3D texturés, obtenus par *TotalCalib* ou *Rekon*, sont décrit dans le format Inventor®. Ce format est très pratique car il est extensible et permet d'ajouter de l'information dans les fichiers.

Nous avons notamment ajouté des numéros à chaque polygone construit. Nous avons également ajouté une étiquette nous permettant de reconnaître si le polygone était visible sur l'image d'origine. Ceci nous permet de faire une distinction sur la validité des polygones dans notre système. Ceux reconstruits depuis l'image d'origine sont considérés plus importants que ceux non visibles sur cette image, comme par exemple pendant le raffinement de la radiosité hiérarchique.

3.5 Discussion

Dans ce chapitre, nous avons décrit l'aspect matériel de nos recherches. Cet aspect n'est pas à négliger car il représente une quantité de travail considérable. Tous les choix réalisés ont une influence sur les méthodes de recherche présentées dans les chapitres suivants. Alors que certains chercheurs sur le même thème ont privilégié la qualité des équipements (appareils photographiques hauts de gamme, lampes diffuses ou ponctuelles, scan-

ner 3D), nous avons préféré nous placer dans le cas d'un utilisateur ayant un matériel restreint par un certain budget.

Les outils de reconstruction choisis sont des outils de recherche. Il nous a fallu une période d'adaptation pour les manipuler correctement. Comme nous l'avons décrit dans ce chapitre, nous sommes très contents des deux outils de reconstruction utilisés. Ils nous ont permis d'obtenir des modèles 3D texturés de qualité, et facilement intégrables dans nos systèmes.

Dans la suite du document, nous décrivons les méthodes de recherche que nous avons développées pour permettre le ré–éclairage interactif des scènes réelles, capturées et reconstruites par les méthodes décrites dans ce chapitre.