

UNIVERSITÉ DE LIÈGE
Faculté des sciences
Sciences géographiques



Modélisation tridimensionnelle de fragments urbains par voie photogrammétrique

Mémoire présenté par :

Julien ERNST

pour l'obtention du titre de :

licencié en sciences

géographiques

option géomatique et

géométrie

Année académique :

2005 - 2006

Remerciements

Nous voici, enfin, arrivés à la fin de ce mémoire... Il est grand temps de penser à vous remercier, vous tous qui m'avez aidé. Mes remerciements vont tout spécialement,

Au Professeur Roland Billen pour ses précieux conseils, son suivi et son implication durant toute l'élaboration de ce mémoire,

Au Professeur Albert Collignon pour ses conseils judicieux en matière de photogrammétrie qui m'ont permis d'explorer de nouvelles pistes de recherches,

Au Professeur Jacques Teller pour ses indications et les données qu'il a mises à ma disposition, elles m'ont permis de trouver plus facilement la bonne marche à suivre dès le début,

À Monsieur Pierre Hallot pour l'aide et le soutien qu'il m'a apportés tout au long de ce mémoire par ses conseils et ses idées, et d'avoir manié avec tant d'habileté son logiciel de topographique,

À Monsieur Benoît Schumacker pour ses précieux conseils en matière de programmation et d'utilisation du compilateur,

À tous les membres de l'Unité de Géomatique pour leurs remarques constructives durant les réunions de staff,

À Benjamin Theys pour le prêt de son appareil photographique et les quelques conseils qu'il a pu me fournir.

Enfin, ces remerciements ne seraient pas complets si nous ne remercions pas nos proches, mes parents, Pauline, Céline, pour leur soutien et pour avoir lu et relu ces nombreuses pages. Et enfin, un merci tout particulier à Mélanie pour son soutien, sa patience et surtout ses prouesses graphiques.

Table des matières

1. Introduction	7
2. Identification des contraintes	9
2.1. Analyse du titre.....	9
2.1.1. Levé tridimensionnel.....	9
2.1.2. Fragments urbains.....	9
2.2. Analyse de l'existant.....	10
2.2.1. Le LEMA.....	10
2.2.2. Cahier des charges.....	11
3. État de l'art	13
3.1. Introduction.....	13
3.2. Les techniques d'acquisition en milieu urbain.....	13
3.2.1. Mesures laser.....	13
3.2.2. Mesures laser autoportées.....	14
3.2.3. Mesures photogrammétriques.....	14
3.2.4. Mesures photogrammétriques autoportées.....	16
3.3. Caractéristiques de la DLT.....	16
3.3.1. Positif.....	17
3.3.2. Négatif.....	17
3.3.3. Analyse des caractéristiques.....	18
3.4. Les outils de traitement.....	18
3.4.1. Arpenteur.....	18
3.4.2. PhotoModeler.....	22
3.4.3. FotoG.....	24
3.4.4. Elcovision.....	25
3.5. Conclusion.....	26
4. Théorie de la DLT	28
4.1. Introduction.....	28
4.1.1. Historique.....	28
4.2. La Direct Linear Transformation 3D (DLT).....	29
4.2.1. Principes de base.....	29
4.2.2. Théorie.....	29
4.2.3. Modélisation des erreurs optiques.....	31
4.3. Résolution.....	35
4.3.1. Détermination des paramètres.....	35
4.3.2. Reconstruction tridimensionnelle.....	39
4.4. Compensation globale.....	41
4.4.1. Compensation à partir du nuage de points.....	41
4.4.2. Cas de la DLT.....	41
5. Mise en oeuvre de la DLT	43
5.1. Introduction.....	43
5.2. Technique de validation.....	43
5.2.1. Mesures laser.....	43
5.2.2. Méthode de mesure.....	44
5.3. Méthodologie.....	45
5.3.1. Prises de vues.....	46
5.3.2. Mesure des points de calage.....	46
5.4. Calcul.....	47
5.4.1. Traitement des données.....	47

5.4.2. Exploitation des données	48
5.4.3. Identification du problème	49
5.5. Optimisation de calcul	49
5.5.1. Introduction.....	49
5.5.2. Méthode de Gauss Jordan	50
5.5.3. Méthode de <i>Gréville</i>	52
5.5.4. Décomposition <i>QR</i>	55
5.5.5. Conclusion	58
5.5.6. La décomposition <i>QR</i> et les moindres carrés.....	59
5.6. Développement d'une application	62
5.6.1. Introduction.....	62
5.6.2. Contrôle de la qualité de l'ajustement	62
5.6.3. Structure de l'application.....	64
5.6.4. Fichiers types	69
5.6.5. Les principales fonctions	73
6. Terrain et application	81
6.1. Introduction.....	81
6.2. Travail de terrain.....	82
6.2.1. Visite préliminaire	82
6.2.2. Première campagne de mesures.....	84
6.2.3. Deuxième campagne de mesures.....	84
6.2.4. Troisième campagne de mesures	87
6.3. Traitement des images	88
6.3.1. Conversion.....	88
6.3.2. Digitalisation des points de calage.....	90
6.4. Comparatif des précisions obtenues.....	90
6.4.1. Les données	91
6.4.2. Quid de la méthode du <i>LEMA</i>	94
6.4.3. L'analyse de la variance	95
6.4.4. Comparaisons multiples de Scheffé.....	98
6.4.5. Analyse des résultats.....	99
6.4.6. Position des points de vues	99
6.5. Validation des choix de compensation globale.....	100
6.6. Cas concret de restitution	101
6.6.1. Visite du site de mesure	102
6.6.2. Mesure des points de calage	104
6.6.3. Digitalisation.....	104
6.6.4. Détermination des paramètres et restitution	105
6.6.5. Traitement des points restitués	106
6.6.6. Analyse	107
7. Conclusion	108
7.1. Conclusion générale.....	108
7.2. Perspectives.....	109
Bibliographie	110

Table des figures

Figure 1. Zone levée par le Laboratoire du LEMA divisée en catégories de précision	11
Figure 2. Exemple de visualisation d'un nuage de points en milieu urbain.....	14
Figure 3. Organigramme présentant la structure de l'application ARPENTEUR	20
Figure 4. Mesures des points de calage pour l'orientation interne.....	21
Figure 5. Détermination des paramètres de la DLT par ARPENTEUR	22
Figure 6. Interface graphique : Elcovision.....	25
Figure 7. Exemple de données Exif	26
Figure 8. Projection centrale	30
Figure 9. Visualisation graphique des erreurs dues à la lentille.....	34
Figure 10. Correction d'une image à partir du modèle de distorsion à 5 paramètres.....	35
Figure 11. Représentation de différentes courbes de Gauss en fonction de la moyenne μ et la variance σ^2	38
Figure 12. Leica TCRA 1103 +	44
Figure 13. Schéma explication de la méthode du rabattement.....	44
Figure 14. Site de mise en œuvre de la DLT.....	45
Figure 15. Représentation graphique de l'indice de convergence : <i>Gauss Jordan</i>	52
Figure 16. Représentation graphique de l'indice de convergence : <i>Gréville</i>	54
Figure 17. Représentation graphique de l'indice de convergence : <i>QR</i>	58
Figure 18. Représentation graphique de l'indice de convergence : décomposition <i>QR</i>	61
Figure 19. Hôtel de ville de Verviers.....	82
Figure 20. Ancien commissariat de police de la ville de Verviers.....	83
Figure 21. Position des points de vue pour le test de la DLT	87
Figure 22. Amélioration interactive du contraste.....	89
Figure 23. Importation des images Bitmap dans <i>IDRISI</i>	90
Figure 24. Fichier des points de calage du cliché 181	92
Figure 25. Fichier des paramètres du cliché 181.....	92
Figure 26. Fichier des points de calage du cliché 183	93
Figure 27. Fichier des paramètres du cliché 183.....	93
Figure 28. Fichier des résidus associés aux clichés 181 et 183.....	94
Figure 29. Représentation de la façade sur l'ancien commissariat de police qui a servi au test de la DLT	95
Figure 30. Position (approchée) des points de vue autour de l'hôtel de ville	103
Figure 31. Exemple de modèle 3D texturé pouvant être obtenu.....	107

Table des tableaux

Tableau 1. Résumé des techniques utilisées par les logiciels étudiés	27
Tableau 2. Caractéristiques des tableurs permettant la résolution des paramètres de la DLT et la restitution	48
Tableau 3. Caractéristiques techniques des appareils de prises de vue.....	86
Tableau 4. Distance entre le point reconstruit avec le DLT et la vérité terrain.....	97
Tableau 5. Table récapitulative de l'analyse de la variance.....	97
Tableau 6. Écart entre la position mesurée et calculée des points de vue.....	100
Tableau 7. Écart sur les coordonnées restituées à partir de deux couples stéréoscopiques.....	101
Tableau 8. Caractéristiques techniques de l'appareil	103
Tableau 9. Géocodification utilisée pour la restitution de l'hôtel de ville de Verviers.....	105
Tableau 10. Exemple de résidus sur 4 points de contrôle	106

1. Introduction

La modélisation tridimensionnelle d'objet en milieu urbain est un domaine de recherche en pleine expansion. L'évolution des techniques d'acquisition et les méthodes de traitements permettent d'adopter une méthodologie simplifiée, non seulement au niveau du travail de terrain, mais aussi lors de la phase ultérieure consacrée aux traitements.

L'acquisition des données est une spécialité des Géomètres tandis que ce sont les Architectes et les Urbanistes qui exploitent celles-ci. Dans ce travail, nous avons mis en avant les spécificités de chacun, et c'est dans le cadre d'un échange inter Facultaire qu'a germé l'idée de ce travail.

À la suite d'une demande du Laboratoire d'Etude Méthodologique Architectural (*LEMA*) de la Faculté des Sciences Appliquées de l'Université de Liège, nous, spécialistes topographes et Géomètres, avons été sollicités pour mettre nos connaissances à leur service dans le but développer une méthode de levé photogrammétrique en milieu urbain.

Au cours de ce mémoire, nous aborderons successivement les différents aspects théoriques de la méthode ainsi que le choix d'une application, le cas échéant nous la développerons nous-mêmes, elle répondra ainsi à toutes les contraintes que nous avons fixées. Ensuite, nous exploiterons les différentes caractéristiques de l'application afin de mettre en œuvre la méthode, il s'agira de la partie pratique du mémoire.

Au cours de ce mémoire, nous allons adopter une méthode de conception qui se rapproche d'une *démarche mixte* (DONNAY 2005) en commençant par réaliser des choix sur base d'une analyse préliminaire, l'état de l'art, quant à la méthode la mieux adaptée à une application photogrammétrique en milieu urbain pour des Architectes.

Après une première analyse des différentes références abordant le thème choisi, nous allons mettre au point un premier prototype jetable de l'application. Une évaluation de ce dernier nous permettra d'identifier et de trouver des solutions aux problèmes éventuels qui pourraient être rencontrés. Le processus d'identification et de résolution se poursuit en boucle jusqu'à ce que l'ensemble des problèmes soient identifiés et résolus. La phase de prototypage est terminée.

C'est en se basant sur ce prototype et sur l'expérience que nous avons acquise durant son élaboration que nous allons développer une application finale. Elle devra remplir les conditions établies par les Ingénieurs Architectes et posséder les fonctions nécessaires à la détermination de paramètres, notamment sur la précision de l'ajustement, qui seront utiles pour la suite de notre analyse.

À la suite de cette étape de conception, nous aborderons, dans la seconde partie de ce mémoire des considérations plus pratiques.

Nous allons d'abord mener une campagne de test. Il s'agira alors de quantifier la précision de l'ajustement et de la restitution en fonction de certains paramètres. Pour cela, nous utiliserons les fonctions disponibles dans l'application afin de diminuer la durée des traitements.

La dernière partie de ce travail sera consacrée au développement d'une méthodologie propre à l'utilisation de la DLT en milieu urbain. Les différentes étapes y seront expliquées et illustrées en insistant sur la durée de chacune d'elles. Pour en permettre une bonne compréhension, il nous semble très important d'illustrer les différentes étapes de cette méthodologie. C'est dans cette optique que nous présenterons un exemple complet de restitution d'un édifice architectural.

2. Identification des contraintes

2.1. Analyse du titre

Afin de commencer débiter dans ce mémoire de la meilleure manière, une bonne compréhension du titre nous paraît importante pour identifier les contraintes et les objectifs qui y sont définis. Nous relèverons plus particulièrement les deux termes suivants : *levé tridimensionnel* et *fragments urbains*.

2.1.1. Levé tridimensionnel

La finalité de ce mémoire consistera à présenter un modèle en trois dimensions de l'objet étudié. Cependant, cette problématique ne sera abordée que de la dernière section de notre mémoire (cf. § 6.6). L'ensemble des développements antérieurs s'attacheront plutôt à décrire les moyens plutôt que les résultats.

2.1.2. Fragments urbains

Dans l'expression *fragments urbains*, nous retiendrons essentiellement le deuxième terme : *urbains*. C'est en effet lui qui détermine et impose la plus grande contrainte sur notre travail futur. En effet, mettre en œuvre un levé en milieu urbain impose à l'opérateur de s'adapter sans cesse en fonction de la configuration des lieux, l'exiguïté des rues, le manque de recul, la circulation automobile, et bien d'autres paramètres. Quant à *fragments*, il nous apporte simplement un complément d'information sur l'extension spatiale du levé. Un fragment étant une partie d'un tout, un fragment urbain est donc une fraction d'une zone urbaine.

2.2. Analyse de l'existant

2.2.1. Le LEMA

Rappelons que c'est dans un projet de synergie avec le laboratoire *LEMA* (Laboratoire d'Etude Méthodologique Architectural) de l'Institut de Génie Civil de l'Université de Liège que s'inscrit ce mémoire. Son but est d'apporter aux Ingénieur Architectes une ou plusieurs techniques de levé, de préférence, par voie photogrammétrique.

Ils utilisent déjà une méthode dont voici très brièvement les fondements. Elle suppose que le plan qui contient la façade soit vertical. Les coordonnées qui doivent être déterminées dans ce plan vertical sont alors calculées par extrapolation à partir d'une mire placée sur la façade dans le plan du cliché. La position planimétrique dans le système Lambert belge de la construction est calculée à partir de mesures sur le plan cadastral. Quand à la position altimétrique, sa détermination reposait sur un nivellement de quelques dizaines de points situés au pied des façades des bâtiments.

Ce n'est évidemment pas une technique que l'on ne retrouve pas dans nos manuels de géométrie. Elle a cependant l'avantage de permettre le levé de très grandes zones en un minimum de temps.

Avec cette méthode, le *LEMA* a entrepris le levé tridimensionnel d'une zone importante d'un quartier de la vieille ville de Verviers (Annexe 14). Ce levé a été mené en considérant trois catégories de précision, ces catégories conditionnaient la méthodologie du levé (Figure 1).

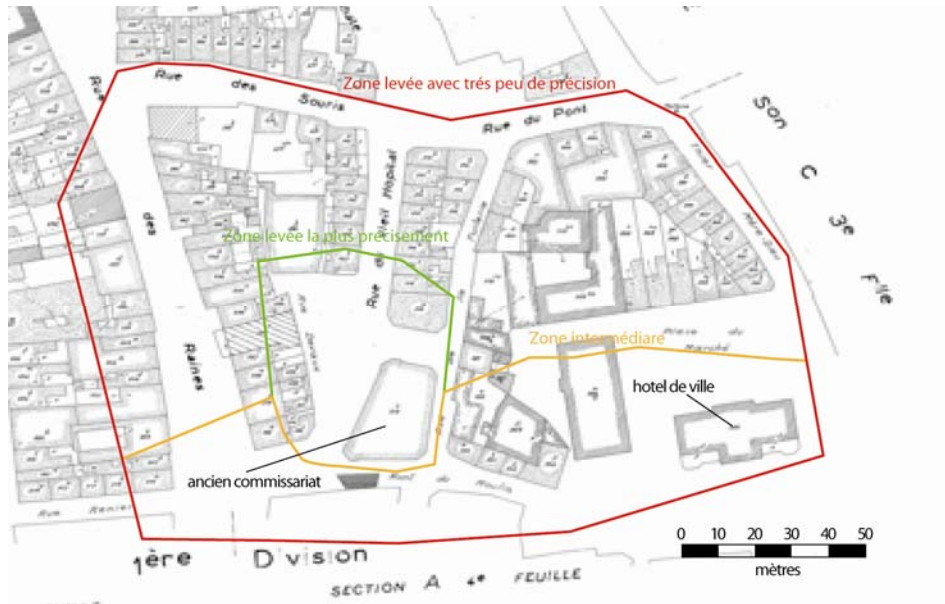


Figure 1. Zone levée par le Laboratoire du LEMA divisée en catégories de précision

(Source : Extrait du plan cadastral, Verviers 2^e division section B 4^e feuille)

2.2.2. Cahier des charges

À la suite de cette description des travaux réalisés par le *LEMA*, nous allons établir un cahier des charges. Il nous servira à baliser notre travail et nos recherches tout au long de l'élaboration de ce mémoire.

Le travail qui va être réalisé ici, s'attachera à proposer une méthode de restitution photogrammétrique tridimensionnelle en milieu urbain. Elle doit limiter au maximum le temps passé sur le terrain, les contraintes qui y sont liées et particulièrement celles qui sont inhérentes aux méthodes de prises de vue. En ce qui concerne les exigences en terme de précision, elles devront absolument rester inférieures à vingt centimètres et cela en toutes conditions.

2.2.2.1. Choix d'une technique photogrammétrique

Dans la première partie de notre état de l'art, nous allons faire un inventaire des différentes techniques de levés en milieu urbain (cf. § 3.2). À la suite de quoi nous ferons un choix de la méthode la mieux adaptée en fonction des différentes contraintes que nous avons identifiées.

La technique choisie sera éprouvée dans des conditions identiques à celles rencontrées lors du levé effectué par le *LEMA*. Nous nous rendrons donc dans les anciens quartiers de Verviers pour réaliser nos mesures. Et nous développerons une méthodologie spécifique pour une utilisation optimale de notre méthode en milieu urbain (cf. § 6).

2.2.2.2. Proposition d'une solution logicielle

Nous étudierons ensuite les différentes solutions de techniques de traitement existantes. Cette étude constituera la seconde partie de notre état de l'art (cf. § 3.4). Et le cas échéant, nous développerons un petit applicatif supportant l'ensemble des tâches qui constituent la reconstruction d'objets tridimensionnels. Toutefois, elle ne permettra pas l'obtention des coordonnées bidimensionnelles dans le système de coordonnées image.

Nous simplifierons donc au maximum son implémentation afin de pouvoir permettre sa possible intégration comme composant d'un autre logiciel. Par exemple, nous pourrions envisager de l'intégrer dans l'application de photogrammétrie aérienne développée par le laboratoire SURFACES : *DDPS* pour *Digital & Didactic Photogrammetric Software* (DA COL & SCHUMACKER 2002 ; <http://www.geo.ulg.ac.be/ddps>). Elle offrirait ainsi une solution logicielle complète combinant la photogrammétrie aérienne et terrestre.

2.2.2.3. Comparaison de techniques

Enfin, ce travail ne serait pas complet si nous ne validions pas notre travail et la précision de notre levé. Nous comparerons aussi la précision de notre levé avec le modèle tridimensionnel du Laboratoire du *LEMA* (cf. § 6.4).

Un autre critère de comparaison est le temps de mise en œuvre de la méthode. Pour cela, nous réaliserons la description complète du levé d'un bâtiment, nous analyserons la méthodologie et le temps nécessaire à chaque étape de la restitution (cf. § 6.6).

3. État de l'art

3.1. Introduction

La modélisation tridimensionnelle de fragments urbains fait l'objet de nombreuses recherches. Les techniques d'acquisition et les méthodes de traitements sont multiples et variées. Nous nous efforcerons, dans ce chapitre, d'exposer les plus représentatives d'entre elles. Nos recherches se limiteront aux différentes techniques d'acquisition en milieu urbain recensées dans la littérature, ainsi qu'aux moyens qui sont proposés à l'utilisateur pour traiter ses données. Nous disposerons ainsi d'une bonne base sur laquelle nous pourrons fonder notre travail futur en terme de techniques d'acquisition et de développement de composants de logiciel.

3.2. Les techniques d'acquisition en milieu urbain

Dans le cadre de l'acquisition terrestre de données en trois dimensions en milieu urbain, deux techniques principales sont utilisées. La première est basée sur l'analyse de photographies en exploitant, ou pas, les principes de la stéréoscopie. L'autre technique utilise les récentes avancées techniques dans le domaine des mesures laser. À l'origine mise au point pour la reconstruction tridimensionnelle d'objets de petite taille, le développement des scanners laser a permis de généraliser leurs utilisations. Nous les retrouvons maintenant dans d'autres applications, notamment les levés topographiques de fragments urbains.

3.2.1. Mesures laser

Les techniques de mesures laser offrent un avantage majeur par rapport aux mesures réalisées à partir de photographies. Il s'agit de l'obtention directe de mesures dans un référentiel tridimensionnel, ce qui ferait penser qu'il s'agit d'une méthode plus rapide et plus simple (Figure 2). Cependant, il ne faut pas négliger la part de post traitement consacrée au nettoyage et à l'analyse du nuage de points obtenu. De plus, lorsque le levé est composé de plusieurs nuages de points, l'ensemble des coordonnées des points doit être recalculé et remis dans le même référentiel.

Ces difficultés sont rencontrées dans la plupart des travaux de reconstruction tridimensionnelle de fragments urbains : c'est par exemple le cas dans BRYER (2003). Leur travail met en avant les difficultés qui apparaissent lors du traitement des nuages de points. Cependant, ils insistent aussi sur le fait que la procédure d'acquisition est relativement simple, notamment au niveau de l'utilisation du matériel.

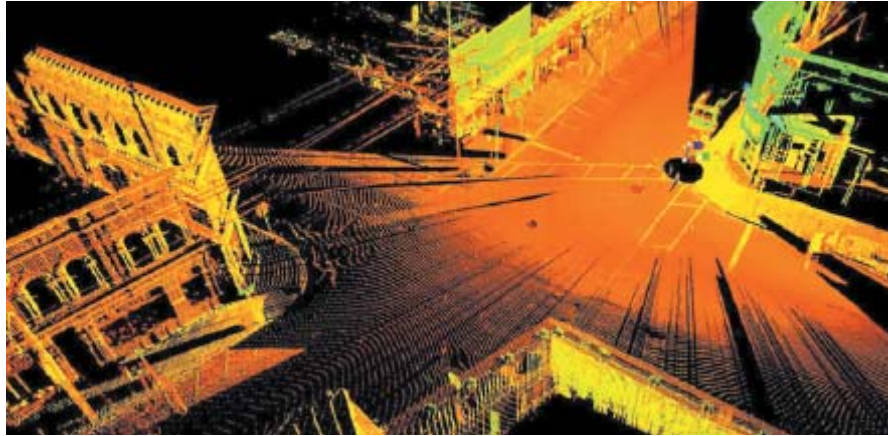


Figure 2. Exemple de visualisation d'un nuage de points en milieu urbain
(D'après JACOBS, 2006, p. 1)

3.2.2. Mesures laser autoportées

Dans ZHAO & SHIBASAKI (2001), est développée une application inédite de reconstruction tridimensionnelle laser de fragments urbains.

Le système embarqué dans le véhicule se compose d'un GPS (Global Positioning System), d'un odomètre et de deux lasers mesurant respectivement, un profil horizontal et l'autre, un vertical. La méthode suppose que le véhicule se déplace sur une surface lisse et plane. Les scans horizontaux représentent le contour des bâtiments, ils permettent d'assurer la cohérence entre les différents scans verticaux et de le replacer dans le même système de référence.

3.2.3. Mesures photogrammétriques

Les méthodes de photogrammétrie peuvent être divisées en deux catégories. La première, basée sur les principes classiques de la photogrammétrie, nécessite d'utiliser une chambre métrique ou un appareil calibré. La seconde, quant à elle, permet l'utilisation d'appareils photographiques conventionnels. Cette dernière manière d'aborder la

photogrammétrie terrestre présente un avantage indéniable par rapport à la première, il s'agit de son coût et de sa facilité de mise en œuvre sur le terrain. Cependant, elle peut, dans certains, cas se révéler moins précise que la méthode classique. On choisira donc, en fonction de l'application la méthode qui paraît la mieux adaptée.

Dans le cas de levé en milieu urbain, la précision n'est pas le critère primordial. Par contre, le temps passé sur le terrain et le fait que ce genre de levé soit souvent effectué par des non professionnels (architectes, urbanistes,...) plaident en faveur d'une méthode photogrammétrique réalisée à partir d'appareils conventionnels.

Nous allons donc nous efforcer, dans la suite, de décrire les moyens et techniques disponibles qui sont consacrées à l'utilisation de ces méthodes photogrammétriques.

3.2.3.1. Applications

Un grand nombre de méthodes d'acquisition et de traitements sont utilisées lors de campagnes de levés photogrammétriques terrestres en milieu urbain. Néanmoins, la solution logicielle dans le cas de traitements photogrammétriques réalisés à partir de clichés conventionnels est souvent identique. Un grand nombre d'auteurs (DRAP 2002 ; JOSÉ MARTINS GOMES *et al.* 1999 ; PERRIN & CHEVRIER 2002) utilisent pour leurs traitements le logiciel de la société canadienne Eos Systems : PhotoModeler (cf. § 3.4.2).

3.2.3.2. La Direct Linear Transformation 3D

Dans LAPLANCHE (2003) la DLT, méthode de photogrammétrie digitale développée par ABDEL-AZIZ & KARARA (1971), a été testée dans un site urbain. Elle nécessite l'utilisation minimum de deux clichés d'une même séquence de bâtiments avec 6 points de calage. Les clichés ont été pris avec un appareil digital compact Nikon Coolpix 4500 dont la résolution du capteur est de $4,5 \cdot 10^6$ pixels. L'appareil était placé sur un pied pour une meilleure stabilité et de façon telle que l'angle entre l'axe optique et l'alignement de la façade soit de minimum 30 degrés. Les points de contrôle ont été mesurés à l'aide d'une station totale laser (Leica TC 1103) dont la portée est de 50 mètres.

Le travail sur site a été réduit à un minimum de deux visites. La première pour prendre les clichés. Ensuite, après avoir choisi et repéré les points de calage sur les photos, une deuxième visite est programmée pour mesurer les coordonnées des points dans un repère tridimensionnel local arbitraire.

Deux modules de calcul sont développés en Visual Basic pour déterminer les coordonnées tridimensionnelles des points se trouvant dans la zone de recouvrement. Le premier permet de déterminer la valeur des paramètres d'orientation L1 – L16 à l'aide des 10 points de calage. Le second fournit, à partir des paramètres calculés dans le premier module, la position tridimensionnelle de n'importe quel point dans la zone de recouvrement.

Nous retrouvons aussi dans la littérature, de nombreuses applications qui utilisent la DLT. La plupart d'entre elles insistent sur le fait que la DLT présente les avantages du laser scanner lors de l'acquisition des données, tout en évitant la grande quantité de traitements induits par le laser. Nous développerons plus en détail les avantages et les potentialités qu'offre la DLT en matière d'acquisition de données (cf. § 3.2).

3.2.4. Mesures photogrammétriques autoportées

Il nous semble intéressant de citer les applications autoportées telles que celle qui a été développée par le Ministère des Équipements et des Transports (MET). Elle permet d'acquérir les données d'une manière très simple. Il suffit de parcourir les rues qui composent la zone d'étude avec un engin mobile équipé de quatre cameras pour recréer son modèle tridimensionnel

Certaines campagnes de levés photogrammétriques classiques utilisent des moyens techniques plus importants. Nous retrouvons, par exemple, dans KERSTEN (2005) des méthodes de restitution utilisant des moyens tels que des élévateurs, ou encore des hélicoptères. Ces moyens permettent de modifier la position des points de vues selon l'axe Z (altitude) et ainsi de diminuer les surfaces cachées et d'augmenter la qualité de la restitution.

3.3. Caractéristiques de la DLT

Dans cette première partie de notre état de l'art, nous avons montré et expliqué plusieurs techniques de levé en milieu urbain ainsi qu'une série de logiciels et applications photogrammétriques. Nous allons maintenant justifier le choix de la méthode qui sera développée dans la suite.

Comme nous l'avons vu plus haut, la Direct Linear Transformation est une méthode de photogrammétrie digitale permettant l'utilisation d'appareils de prises de vues

conventionnels et une simplification maximale du travail de terrain (cf. § 3.2.3.2). Elle répond donc très bien aux exigences que nous avons fixées (cf. § 2.2.2). Rappelons que la méthode choisie doit être utilisée par des Architectes et des Urbanistes et que son objectif est de permettre une mise en œuvre sur le terrain rapide et simple. Nous présentons ici les caractéristiques positives et négatives de la DLT.

3.3.1. Positif

- La distance principale (longueur de la focale) peut être inconnue. De plus, elle peut être différente pour les clichés qui sont réalisés lors de la campagne de levé ;
- le système de coordonnées image peut être arbitraire. Les axes ne doivent pas être orthogonaux ni de même échelle ;
- la position du point principal de l'image, et celle du point de vue peuvent ne pas être connues;
- les paramètres de modélisation des erreurs optiques peuvent être introduits dans les inconnues de la DLT.

Il n'y a donc apparemment aucune contrainte sur le type d'appareil photographique utilisé. Bien entendu, il semble évident que l'utilisation d'une optique et d'un capteur CCD de bonne qualité est un gage de précision pour l'ajustement des paramètres de la DLT. La position de l'appareil est aussi une inconnue des équations de la DLT. Cette propriété permettrait par exemple de travailler à partir d'anciens clichés numérisés. Cependant, nous veillerons à suivre les recommandations énoncées dans LAPLANCHE (2005) sur le positionnement et la stabilisation de l'appareil photographique (cf. § 3.2.3.2).

3.3.2. Négatif

- Pour chaque cliché, nous devons déterminer un minimum de onze paramètres (seize au maximum si l'on tient compte des erreurs optiques). Ce grand nombre de paramètres impose l'utilisation de points de calage abondants et peut induire des problèmes de convergence lors de la résolution par moindres carrés du système.
- Si tous les points de contrôle se situent sur un même plan, cela rend les onze inconnues de la DLT dépendantes et le système à résoudre devient singulier. Des

instabilités numériques apparaissent si la configuration des points de contrôle est proche du plan.

3.3.3. Analyse des caractéristiques

Nous déduisons donc de cette énumération de caractéristiques concernant la DLT que cette technique possède un grand avantage, c'est le peu de contraintes qui encadrent l'acquisition des données. En effet, l'appareil, sa focale, sa position,... peuvent être totalement inconnus lors des traitements à appliquer aux clichés. Cela en fait une technique dont la mise en œuvre est extrêmement souple. La seule recommandation à faire est de mettre l'appareil sur un pied pour le stabiliser.

Par contre, des instabilités numériques peuvent apparaître dans certains cas. Mais mis à part les problèmes liés à la planéarité des points de calage qui sont du ressort de l'utilisateur, les autres problèmes numériques peuvent être évités avec l'utilisation d'un algorithme de calcul stable et bien construit.

L'ensemble de ces raisons nous conforte dans le choix de la technique photogrammétrique. Nous allons développer une méthodologie de levé en milieu urbain utilisant la Direct Linear Transformation 3D (DLT).

3.4. Les outils de traitement

Nous avons choisi la méthode photogrammétrique, il nous faut maintenant déterminer l'application à utiliser pour les traitements des images et l'orientation de celles-ci.

C'est dans cette optique que nous allons de décrire dans cette partie les outils disponibles. Par outils, nous entendons les applications prenant en compte toute la démarche photogrammétrique ou à celles s'attachant à une étape plus particulière, comme par exemple la calibration des appareils de photographie.

3.4.1. Arpenteur

3.4.1.1. Aperçu général

ARPENTEUR (pour ARchitectural PhotogrammEtry Network Tool for EdUcation and Research) est un ensemble d'outils logiciels développés en collaboration avec le laboratoire

LERGEC de l'ENSAIS Strasbourg. Les principales raisons d'être du projet sont les suivantes :

- proposé un logiciel pour l'éducation et la recherche, le langage de développement JAVA fournit un outil et une technologie offrant à des équipes travaillant sur des sites et des systèmes distincts et distants, un moyen commode d'analyser et d'échanger leurs données ;
- un outil dédié à l'architecture, ARPENTEUR bénéficie de l'expertise des deux équipes dans les domaines de la photogrammétrie rapprochée et de la représentation des connaissances architecturales ;
- un outil dédié à la photogrammétrie, ARPENTEUR est un système simple et doit être classé parmi les systèmes légers de photogrammétrie, léger tant pour sa simplicité d'utilisation que pour le matériel nécessaire à son exploitation.

L'intégration de ces objectifs dans un même ensemble s'appuie sur des choix techniques et conceptuels.

Le premier de ces choix consiste à utiliser des images digitales. Celles-ci peuvent être obtenues à l'aide d'appareils de photographie numérique que l'on trouve aujourd'hui dans le commerce et dont certains présentent des caractéristiques suffisantes de qualité des images produites. Ces images numériques permettent également d'offrir à l'utilisateur des outils de traitement qui automatisent certaines tâches habituellement réalisées par un opérateur humain. Enfin, elles permettent l'intégration totale de la chaîne de traitement depuis les photographies jusqu'à certains résultats finaux comme la visualisation en 3 dimensions dans des logiciels de CAO-DAO 3D.

Cette intégration est mise à profit pour servir un autre choix, conceptuel celui-ci, il consiste à rendre utilisable par tous et selon son domaine d'application les techniques proposées par la photogrammétrie et le logiciel ARPENTEUR. Concernant l'architecture aussi bien que l'archéologie, il s'agit de permettre à l'expert du domaine d'exploiter ses connaissances pour produire au mieux un résultat conforme à ses desiderata. Ce résultat pouvant consister en documents de relevés, en fichiers destinés à la visualisation ou en corpus destiné à une base de données. Le moyen utilisé dans ce but consiste à fournir à cet expert un ensemble d'outils lui permettant d'exprimer des hypothèses relatives à son champ d'investigation, hypothèses dont l'utilisation conduit à un allègement du processus de

mesurage. Parmi celles-ci, citons par exemple, la création d'un corpus représentant les objets présents dans le champ d'investigation.

Comme bénéfice de ces choix ARPENTEUR apparaît comme un outil destiné à être utilisé par des professionnels de l'architecture ou de l'archéologie avec une intervention réduite de l'expert photogrammètre.

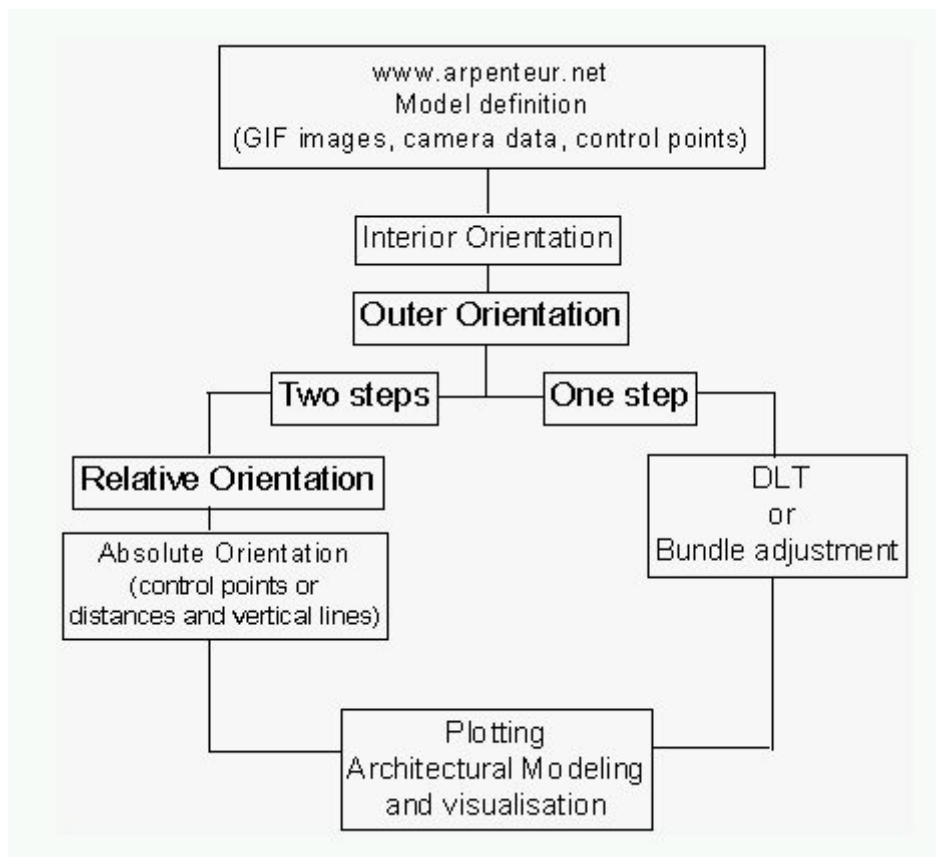


Figure 3. Organigramme présentant la structure de l'application ARPENTEUR

(Source : <http://www.arpenteur.net>, consultation le 18 novembre 2005)

3.4.1.2. Orientation interne

Le logiciel ne propose pas de méthode pour réaliser cette orientation. Les paramètres de l'appareil sont supposés être connus, c'est le cas de l'utilisation de matériel de photogrammétrie terrestre calibré en usine. Sinon, ces paramètres doivent être calculés à l'aide d'un autre logiciel (par exemple avec le module de calibration de PhotoModeler (cf. § 3.4.2.1)). Les paramètres nécessaires pour réaliser l'orientation sont les suivants :

- le nombre de marques fiduciales ;

- les coordonnées des marques fiduciales ;
- les coordonnées du point principal ;
- la longueur de la focale calibrée ;
- une table des déformations radiales.

La photo est alors redressée à l'aide d'une transformation affine à six paramètres. On peut ensuite vérifier la qualité de l'ajustement à l'aide des résidus entre les coordonnées mesurées et théoriques ainsi qu'avec d'autres indicateurs statistiques.

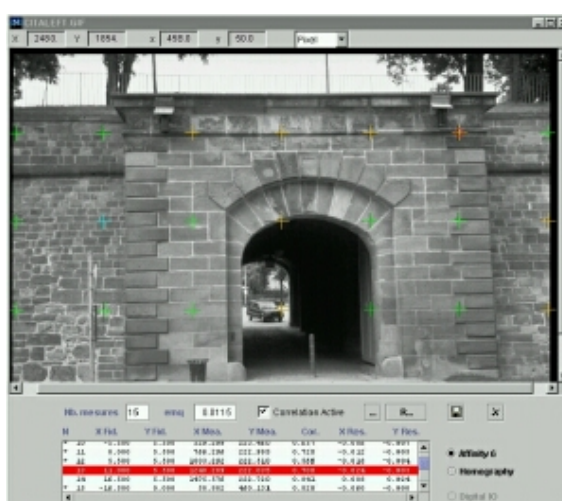


Figure 4. Mesures des points de calage pour l'orientation interne
 (Source : <http://www.arpeniteur.net>, consultation le 18 novembre 2005)

3.4.1.3. Orientation externe

ARPENITEUR propose deux méthodes d'orientation externe. La première exploite les procédés conventionnels utilisés dans les restituteurs analogiques ou digitaux, et l'autre, plus « analytique », utilise notamment la DLT (Direct Linear Transformation).

La première méthode (Two steps) combine une orientation relative et une orientation absolue. Elle se réalise en identifiant des points homologues dans les deux photos (orientation relative). La détermination de la position d'au moins trois points de contrôle peut permettre la mise à l'échelle et l'orientation correcte du modèle dans le système de référence (orientation absolue). Pour permettre l'utilisation d'un appareil digital classique pour une campagne de mesures photogrammétriques, le logiciel propose de déterminer

l'orientation absolue en mesurant, sur le modèle, la longueur d'une verticale comme un bord de mur (attention à la précision de la mesure et à la verticalité du mur). Cela permet d'avoir une direction et une longueur de référence pour déterminer l'orientation absolue.

La seconde méthode (One step) réalise l'orientation relative et absolue en une étape, nous parlons alors d'orientation externe. Elle doit aussi être calculée à l'aide des points de contrôle mesurés sur le terrain. Si au moins six points de calage sont mesurés dans l'image, le logiciel permet d'obtenir une approximation de la position du centre optique ainsi que de la direction de l'axe principal par un calcul de DLT (Direct Linear Transformation). Ces valeurs pourront servir par la suite à des méthodes plus classiques d'ajustement global (*bundle block ajustement*).



Figure 5. Détermination des paramètres de la DLT par ARPENTEUR

(Source : <http://www.arpenteur.net>, consultation le 18 novembre 2005)

3.4.2. PhotoModeler

PhotoModeler est un logiciel de modélisation 3D qui permet la mesure et la construction de modèles en trois dimensions à partir de simples photographies dans des domaines aussi vastes que l'archéologie, l'architecture, le cinéma, l'animation vidéo,... PhotoModeler peut accélérer ou rendre possible la mise en œuvre de certains projets de mesures photogrammétriques.

3.4.2.1. Orientation interne

La calibration de l'appareil photographique peut se faire dans un module spécifique. Celle-ci consiste à photographier une grille imprimée de 100 points (disponible avec le logiciel en format Acrobat .pdf). Elle permettra, après une analyse des déformations, de déterminer les 4 ou 5 paramètres de distorsions radiales de la lentille. Ces paramètres seront alors utilisés pour l'orientation de l'image qui a été prise par cet appareil. On peut remarquer que les déformations introduites lors de l'impression de la grille de points ne sont pas prises en compte par le logiciel.

Si l'appareil utilisé n'a pas été calibré, le logiciel utilise une technique appelée « Inverse Camera ». Cette technique est présentée comme une méthode de résolution de l'orientation interne d'un appareil inconnu à partir d'un certain nombre de points de contrôle (le nombre minimal n'est pas spécifié). En fait, elle est principalement utilisée pour déterminer la longueur de la focale. Elle peut aussi permettre la résolution du point principal, mais il est conseillé d'avoir une très bonne répartition des points de contrôle en plan et en profondeur pour réaliser cette deuxième correction. Il est aussi conseillé de d'abord déterminer la longueur de la focale et puis selon la précision obtenue, de faire le choix de déterminer ou non la position du point principal. Le type de méthode utilisée n'est pas mentionné, mais on peut croire qu'il ne s'agit d'une technique exploitant les même principe que la DLT, puisque celle-ci consiste en une résolution de paramètres inconnus (L1 – L11) qui ne représentent pas directement les inconnues physiques (focale, position du point principal,...) permettant l'orientation interne d'un appareil photographique. De plus, les distorsions de la lentille ne sont pas prises en compte par ce module « Inverse Camera » (L12 – L16).

3.4.2.2. Orientation externe

Tout comme pour l'orientation interne, les informations sur les techniques d'orientation externe sont très vaguement abordées dans les références consultées : « À partir du moment ou PhotoModeler connaît la position des appareils et les angles de prise de vue dans l'espace à 3 dimensions, le processus de modélisation peut se faire avec une plus grande précision. ».

La résolution de l'orientation absolue du modèle, c'est-à-dire l'échelle et l'orientation d'un modèle, est expliquée dans le cas simple du calcul de la modélisation 3D d'un parallélépipède à base rectangle. Cette orientation absolue nécessite la connaissance d'au moins 3 points pour définir un repère droit en deux dimensions. L'axe qui détermine la

troisième dimension est calculé en fonction du plan formé par les deux précédents. Une autre technique existe, mais elle n'est applicable que dans le seul cas où on désire recréer un simple modèle en trois dimensions. Elle consiste à mettre assez de segments de droite en correspondance dans les deux photos homologues. On peut ainsi obtenir directement un modèle en 3D formé de ces segments dans un système arbitraire.

Dans les différents rapports de campagnes photogrammétriques que nous avons collectés, l'utilisation de PhotoModeler pour la modélisation en 3 dimensions est toujours précédée par une calibration de l'appareil en suivant la méthode décrite ci-avant (bien sûr, celle-ci est adaptée en fonction des conditions de prise de vue). Une autre étude utilise directement un appareil calibré pour lequel tous les paramètres d'orientation interne sont connus. L'orientation absolue des photographies est réalisée en plaçant des mires verticales graduées dans le champ de la photographie. Dans d'autres cas, cette orientation est calculée en fonction de points de contrôle connus dans le repère de référence.

Nous énonçons ci-contre différents travaux utilisant le logiciel PhotoModeler :

- Un système de gestion de documents hétérogènes dédiés au patrimoine archéologique de l'épave Etrusque du Grand Ribaud (DRAP 2002) ;
- Projet REVCAP : reconstruction 3D interactive de zones urbaines (PERRIN & CHEVRIER 2002) ;
- A photogrammetric project in Brazil: the use of the PhotoModeler Software (JOSÉ MARTINS GOMES *et al.* 1999).

PhotoModeler est un logiciel très complet qui permet d'utiliser plusieurs techniques pour orienter les photographies. Même si les traitements du logiciel peuvent être qualifiés de « boîte noire », il permet une utilisation assez simple qui peut convenir à bon nombre d'utilisateurs qui ne cherchent pas à savoir quelle technique est réellement utilisée. Le seul reproche que nous pouvons lui attribuer est que, lorsqu'on ne connaît pas les paramètres d'un appareil, le module « inverse camera » ne permet pas de résoudre avec rigueur l'entièreté de l'orientation interne.

3.4.3. FotoG

Le programme de photogrammétrie terrestre de la société Vexcel, FotoG, permet des mesures précises sur des clichés digitaux, de les convertir en 3 dimensions, d'importer les données dans un programme de type CAD et de le faire facilement et rapidement. FotoG

est plutôt orienté vers la modélisation en 3 dimensions dans un logiciel CAD. La calibration de l'appareil n'est pas abordée dans les données consultées.

L'orientation externe nécessite la connaissance d'au moins trois points de calage dans l'image. Ces points peuvent être levés selon différentes manières : soit en plaçant des marques dans le champ de la caméra ou en référençant dans l'image des points ou des objets connus. Ensuite, il est proposé à l'utilisateur d'identifier des points homologues qui permettront de mettre en relation les différents clichés, et ainsi déterminer la position des appareils. Enfin, le logiciel permet un transfert des données dans un logiciel de type CAD pour pouvoir procéder aux mesures. En fait, FotoG réalise directement un modèle 3D à partir d'un couple stéréoscopique qui peut ensuite être importé dans un logiciel approprié.

3.4.4. Elcovision

Elcovision est un logiciel qui permet de corriger les déformations des images engendrées par les distorsions de la lentille.

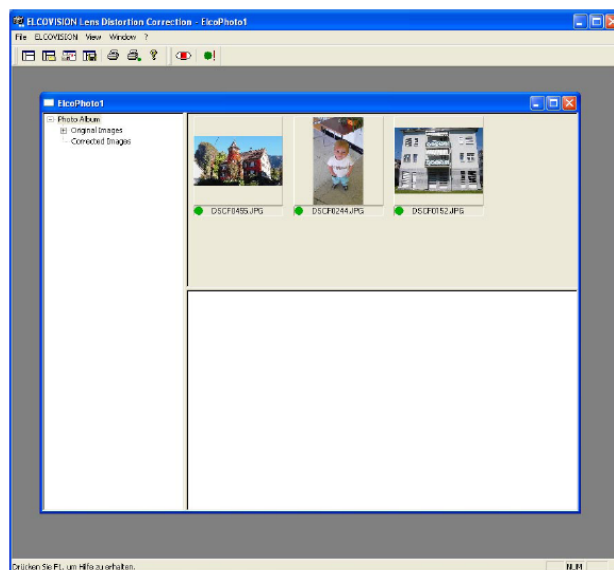


Figure 6. Interface graphique : Elcovision

(Source : Photo Mess System AG, 2004, p. 6)

Il n'est cependant pas capable de recalculer des photos prises avec des appareils dont il ne connaît pas les caractéristiques. Son système de correction est basé sur les données Exif qui sont attachées à chaque image (il faut, dès lors, prendre garde à la compression éventuelle des photographies). Les métadonnées définies dans le format Exif standard sont:

- l'information de la date et de l'heure. Les appareils numériques enregistrent la date et l'heure de la prise de vue et les insèrent dans les métadonnées ;
- les réglages de l'appareil. Cela comprend des informations constantes telles que la marque et le modèle de l'appareil ainsi que des informations variables telles que l'orientation, l'ouverture, la vitesse d'obturation, la longueur de focale, la sensibilité... ;
- des informations géographiques provenant d'un éventuel système GPS connecté à l'appareil. En 2005, seuls quelques rares appareils supportent cette fonction. Certains photographes utilisent un système classique pour localiser leurs déplacements puis ajoutent manuellement aux métadonnées, les coordonnées géographiques ;
- la description des droits d'auteur. Encore une fois, c'est surtout une opération effectuée après traitement des images car seulement très peu d'appareils permettent aux photographes de renseigner ces champs.

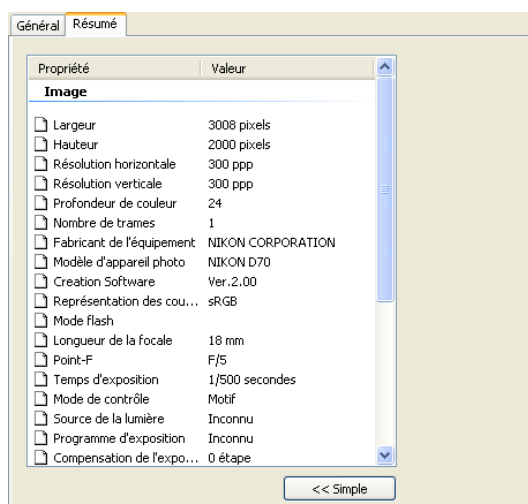


Figure 7. Exemple de données Exif
Canon D70, focale 18 mm

3.5. Conclusion

En guise de conclusion, nous présentons un résumé (Tableau 1) des méthodes utilisées par les logiciels que nous avons étudiés (cf. § 3.4). Nous développerons ensuite les potentialités qu'offrirait l'implémentation d'un applicatif uniquement basée sur la DLT.

Tableau 1. Résumé des techniques utilisées par les logiciels étudiés
Orientation interne et externe

	<i>Quels types d'appareils</i>	<i>Orientation interne</i>	<i>Orientation externe</i>
PhotoModeler	· Tous	· Grille · « inverse camera »	· Mire verticale · Points de contrôles
ARPENTEUR	· Tous	· Utilise les paramètres connus	· Relative et absolue (3 pts) · Ajustement + DLT (6 pts)
FotoG	· Tous	· ?	· Points de contrôle (3 pts)
Elcovision	· Appareils digitaux connus	· Oui	· Non

Quelque soit l'application, il faut toujours connaître la position d'un minimum de trois points en trois dimensions dans le référentiel terrain afin de pouvoir résoudre l'orientation interne et externe du modèle. Dans le cas de PhotoModeler, le calcul de l'orientation interne est surtout basé sur des distorsions subies par une grille après qu'elle ait été photographiée. Même si aucun test n'a été effectué sur cette méthode, la photographie de la grille ne tient pas compte des conditions réelles d'utilisation sur le terrain.

Même si les informations acquises sur PhotoModeler ne permettent pas de déterminer avec certitude les procédés utilisés pour l'orientation des photographies, nous pouvons toutefois affirmer que ni lui, ni Arpenteur ne permet d'utiliser exclusivement la DLT, que ce soit pour l'orientation externe et pour l'orientation interne.

L'implémentation d'une application photogrammétrique exclusivement dédiée à la DLT (orientation interne et externe) nous paraît donc intéressante après l'analyse des moyens actuellement disponibles. Non seulement parce qu'aucun d'entre eux ne réalise le calcul complet de l'orientation des clichés par la méthode de la DLT, mais aussi parce que l'implémentation de la méthode nous permettra d'avoir un contrôle total sur l'ensemble du processus.

4. Théorie de la DLT

4.1. Introduction

Dans ce chapitre sont développés et expliqués les équations de la méthode de la Direct Linear Transformation (DLT). L'exposé commence par une brève introduction historique rapportant l'ensemble des intervenants qui ont contribué au développement de la méthode. Ensuite est abordée la méthode d'orientation des images selon la méthode de la DLT en vue de la restitution, qui est le troisième point de ce chapitre. Nous trouvons, finalement, un paragraphe traitant de la problématique de la compensation globale de plusieurs modèles lors de l'utilisation de la DLT.

4.1.1. Historique

La méthode de la Direct Linear Transformation (DLT) a été proposée par ABDEL-AZIZ & KARARA (1971). Elle fait partie des techniques les plus utilisées pour la reconstruction tridimensionnelle d'objets. Surtout employée dans les domaines de la médecine, son utilisation peut être généralisée à tout autre type de mesure de points dans un espace tridimensionnel.

Ce n'est que quelques années plus tard que MARZAN & KARARA (1975) introduisent dans les équations de la DLT des paramètres de corrections qui modélisent les erreurs introduites par les distorsions dues à la lentille. On retrouve le même développement dans KWON (1998). Ils ont utilisé un modèle classique (GHOSH in CHALLIS & KERWIN 1992) mais avec seulement trois paramètres pour les distorsions radiales alors que GHOSH recommande d'en utiliser au moins cinq et deux pour le décentrement.

Par la suite, plusieurs auteurs ont étudié les effets de certains facteurs sur la reconstruction tridimensionnelle de points avec la méthode de la DLT.

WOOD & MARSHALL (1986) ont analysé l'influence de la distribution des points de calage sur la précision de la restitution. Ils en ont conclu que la perte de précision sur la restitution est importante si le point se situe hors de la zone couverte par le point de calage (détermination de la position par extrapolation).

En ce qui concerne la modélisation des erreurs optiques, MILLER *et al.* (1980), WALTON (1981), WOOD & MARSHALL (1986) ont tous démontré que la modélisation des distorsions dues aux lentilles n'augmentait pas de manière significative l'exactitude de position lors de la reconstruction de points.

4.2. La Direct Linear Transformation 3D (DLT)

4.2.1. Principes de base

La Direct Linear Transformation 3D (DLT) est une méthode de photogrammétrie numérique. Elle permet la reconstruction tridimensionnelle d'objets à partir d'un minimum de six points de calage et de deux clichés en format numérique. Afin d'augmenter sa précision théorique, la DLT permet à l'utilisateur d'augmenter le nombre de clichés ou le nombre de points de calage. La modélisation des erreurs optiques est aussi possible, mais elle implique d'augmenter le nombre minimum de points de calage. Il passe de six à huit.

La différence essentielle avec les principes de la photogrammétrie classique est l'absence d'orientation relative durant le processus de reconstitution des gerbes perspectives. Les paramètres d'orientation de la DLT peuvent modéliser l'orientation intérieure, relative et absolue, et ils sont résolus en une étape par un seul système d'équations.

4.2.2. Théorie

Les équations de la Direct Linear Transformation (DLT) peuvent être obtenues à partir des équations de base de la projection centrale [équation (1') et (1'')].

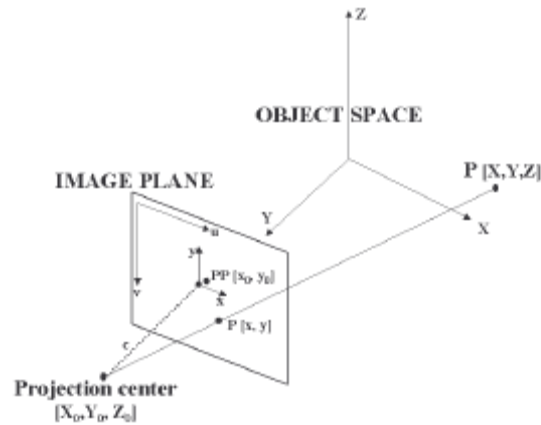


Figure 8. Projection centrale
(D'après REMONDINO, 2002, p. 149)

Ces équations sont basées sur la colinéarité du vecteur liant le centre de projection à la représentation du point dans le plan de l'image et de celui qui lie le centre de projection et le point dans l'espace objet. Cette propriété est aussi dénommée condition de colinéarité :

$$u - u_0 = -c \frac{r_{11}(X - X_0) + r_{21}(Y - Y_0) + r_{31}(Z - Z_0)}{r_{13}(X - X_0) + r_{23}(Y - Y_0) + r_{33}(Z - Z_0)} \quad (1')$$

$$v - v_0 = -c \frac{r_{12}(X - X_0) + r_{22}(Y - Y_0) + r_{32}(Z - Z_0)}{r_{13}(X - X_0) + r_{23}(Y - Y_0) + r_{33}(Z - Z_0)} \quad (1'')$$

En faisant passer les coordonnées du point principal dans le membre de droite, respectivement u_0 et v_0 dans la première et la seconde équation, en réduisant l'expression au même dénominateur, on obtient l'expression suivante :

$$u = \frac{L'_1 X + L'_2 Y + L'_3 Z + L'_4}{L'_9 X + L'_{10} Y + L'_{11} Z + L'_{12}} \quad (2')$$

$$v = \frac{L'_5 X + L'_6 Y + L'_7 Z + L'_8}{L'_9 X + L'_{10} Y + L'_{11} Z + L'_{12}} \quad (2'')$$

Si enfin, on divise les deux équations par L'_{12} on obtient une relation simple entre les coordonnées bidimensionnelles image et les coordonnées tridimensionnelles terrain.

$$u = \frac{L_1 X + L_2 Y + L_3 Z + L_4}{L_9 X + L_{10} Y + L_{11} Z + 1} \quad (3')$$

$$v = \frac{L_5 X + L_6 Y + L_7 Z + L_8}{L_9 X + L_{10} Y + L_{11} Z + 1} \quad (3'')$$

Nous avons obtenu une nouvelle relation entre le système de coordonnées image et le référentiel terrain : ce sont les équations de base de la DLT. Elles contiennent onze inconnues indépendantes L_i pour $0 < i \leq 11$. Les différentes inconnues de la DLT sont fonction des neuf paramètres indépendants, eux aussi, de la projection centrale (trois pour l'orientation interne et six pour l'orientation externe). Or, dans tous les cas, nous devons avoir le même nombre d'inconnues indépendantes.

La solution à cette incohérence mathématique est d'ajouter deux paramètres aux équations de la projection centrale. Ces deux paramètres supplémentaires caractérisent le référentiel des clichés. Ils décrivent respectivement la non orthogonalité des axes et la différence d'échelle entre les deux axes du système de coordonnées image. Cette opération équivaut à appliquer une transformation affine au système de coordonnées image avant de calculer la projection centrale. Cette propriété nous autorise l'utilisation de n'importe quel système de coordonnées image. Plus particulièrement, cela nous permettra, dans nos applications futures, de digitaliser les points de calage dans un logiciel sans se soucier de son système de coordonnées.

Un développement analogue à celui qui a permis de développer l'équation (1' et 1''), nous permet de déterminer les onze coefficients de la DLT en fonction des onze paramètres de la projection centrale y compris les deux paramètres additionnels.

4.2.3. Modélisation des erreurs optiques

Nous avons dit plus haut que certains auteurs avaient démontré le peu d'influence de la modélisation des erreurs optiques sur les résultats de la restitution tridimensionnelle. Cependant, pour le travail qui nous occupe, nous allons utiliser différents appareils de prise de vue tels que des compacts numériques ou encore des appareils reflex numériques avec

grand angle. Il semble intéressant de vérifier, avec un tel matériel, si la modélisation des erreurs optiques apporte un gain de précision significatif.

Les distorsions introduites par les lentilles peuvent être scindées en deux catégories. La première correspond aux distorsions radiales propres au système de lentilles de l'appareil. Cette erreur est une fonction de la distance du point au point principal du cliché. La seconde catégorie correspond au décentrement du point principal par rapport au centre géométrique du cliché. Nous veillerons toutefois à ne pas augmenter excessivement le nombre de paramètres, ce qui complique la résolution du système.

Nous trouvons principalement deux types de modélisations des erreurs optiques dans la littérature concernant la DLT. Elles diffèrent par la fonction qui définit les distorsions radiales et par le nombre de paramètres qui lui est associé.

4.2.3.1. Modélisation à 3 paramètres

Dans les travaux de WONG (1975) les erreurs optiques sont modélisées à l'aide de trois paramètres. Les expressions suivantes donnent la valeur des corrections Δu et Δv à apporter aux coordonnées u et v mesurées sur le cliché :

$$\Delta u = \xi r^2 K_1 + 2\xi\eta P_1 + (r^2 + 2\xi^2) P_2 \quad (4')$$

$$\Delta v = \eta r^2 K_1 + 2\xi\eta P_1 + (r^2 + 2\eta^2) P_2 \quad (4'')$$

Où $\xi = u - u_0$, $\eta = v - v_0$ et $r^2 = \xi^2 + \eta^2$ le carré de la distance entre le point (u, v) du cliché et le point principal (u_0, v_0) de l'image.

Nous voyons que les erreurs de distorsions radiales dues à la lentille sont modélisées par une cubique (un coefficient K_1). Tandis que le décentrement du point principal est une fonction des coordonnées et de deux coefficients P_1 et P_2 .

Si nous suivons les recommandations définies par GHOSH en terme de modélisation des distorsions radiales (cf. § 4.1.1), l'unique paramètre modélisant ces distorsions est insuffisant. Pour augmenter l'efficacité du modèle, nous utiliserons une autre méthode.

4.2.3.2. Modélisation à 5 paramètres

KWON (1998) utilise une expression à cinq paramètres. Elle a la même forme que celle qui est décrite dans MARZAN & KARARA (1975). Le modèle de distorsions radiales est inspiré du modèle de BROWN (in GÓMEZ *et al.* 2003 ; in COLLIGNON 2005) et diffère de celui développé dans WONG (1975) qui ne comporte qu'un seul paramètre (cf. § 4.2.3.1). L'expression de ce modèle se retrouve ci-après [équation (5') et équation (5'')]. Les équations donnent une valeur des corrections Δu et Δv à apporter aux coordonnées u et v mesurées sur le cliché.

$$\Delta u = \xi \left(r^2 L_{12} + r^4 L_{13} + r^6 L_{14} \right) + \left(r^2 + 2\xi^2 \right) L_{15} + \xi \eta L_{16} \quad (5')$$

$$\Delta v = \eta \left(r^2 L_{12} + r^4 L_{13} + r^6 L_{14} \right) + \xi \eta L_{15} + \left(r^2 + 2\eta^2 \right) L_{16} \quad (5'')$$

Où $\xi = u - u_0$, $\eta = v - v_0$ et $r^2 = \xi^2 + \eta^2$ le carré de la distance entre le point (u, v) du cliché et le point principal (u_0, v_0) de l'image.

L'expression du modèle peut aussi être scindée en deux parties. La première modélise les distorsions radiales et comporte trois paramètres. Et la seconde modélise le décentrement du point principal. Nous avons donc dans l'équation (5') et l'équation (5''),

L_{12} , L_{13} et L_{14} les trois paramètres modélisant les distorsions radiales, et

L_{15} et L_{16} les deux paramètres modélisant le décentrement du point principal.

Nous pouvons introduire ce modèle d'erreurs optiques dans les équations standard de la DLT [équation (3) et équation (3')]. Nous écrivons cette nouvelle relation qui tient compte de erreurs optiques :

$$u - \Delta u = \frac{L_1 X + L_2 Y + L_3 Z + L_4}{L_9 X + L_{10} Y + L_{11} Z + 1} \quad (6')$$

$$v - \Delta v = \frac{L_5 X + L_6 Y + L_7 Z + L_8}{L_9 X + L_{10} Y + L_{11} Z + 1} \quad (6'')$$

4.2.3.3. Correction d'une image par le modèle à 5 paramètres

Afin d'illustrer la théorie de la modélisation des erreurs optiques (cf. § 4.2.3.2), nous proposons dans ce paragraphe de visualiser les corrections apportées à une image par le modèle à cinq paramètres. Nous avons utilisé une photographie de l'ancien commissariat de police, la longueur de la focale est de 18 millimètres.

Nous avons tout d'abord calculé les valeurs des Δu et Δv en 25 points bien répartis sur l'ensemble de l'image (Annexe 1). Nous obtenons ainsi une série de points corrigés des erreurs optiques $(u_{\text{mod}}, v_{\text{mod}})$. Nous construisons, à partir de ces points corrigés, un diagramme représentant les erreurs calculées pour chacun d'eux (Figure 9).

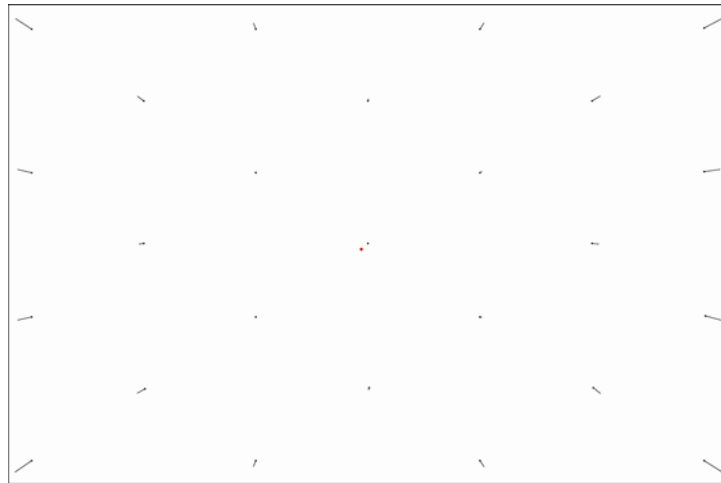


Figure 9. Visualisation graphique des erreurs dues à la lentille

Le point rouge est la représentation de point principal du cliché

Ensuite, à l'aide de la fonction *RESAMPLE* du logiciel IDIRSI (Menu : *Reformat* → *RESAMPLE*), nous ajustons l'image de départ dans un nouveau système défini par les points de coordonnées $u_{\text{mod}}, v_{\text{mod}}$. Nous choisissons une interpolation cubique pour approcher le mieux possible la fonction qui modélise les erreurs optiques [équations (5') et (5'')]. Le résultat obtenu (image initiale et image corrigée) est présenté dans la figure ci-après (Figure 10).

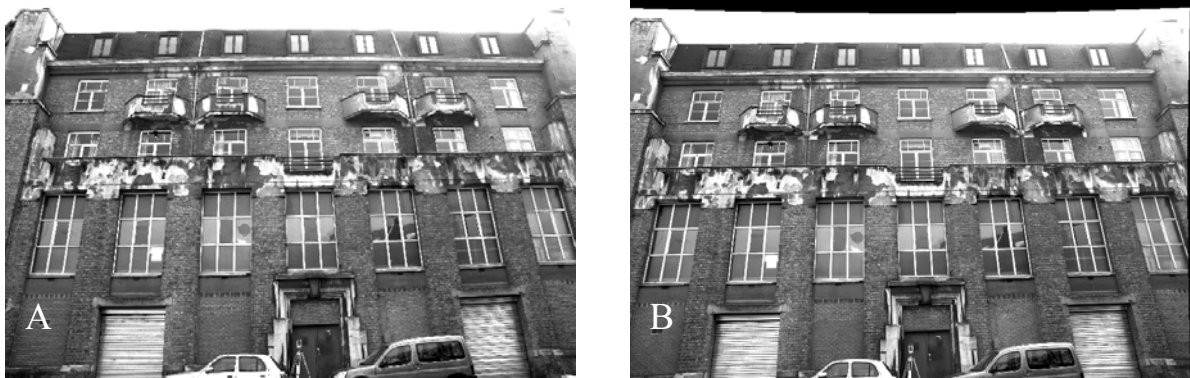


Figure 10. Correction d'une image à partir du modèle de distorsion à 5 paramètres

Figure A représente l'image avant correction et la figure B après.

(Ancien commissariat de police de Verviers)

Les effets de la correction de l'image sont visibles. Nous pouvons par exemple, comparer la forme de la corniche blanche sous les barbacanes. Elle est courbe sur la figure (10-A) et rectiligne sur la figure (10-B).

4.3. Résolution

4.3.1. Détermination des paramètres

Pour déterminer les paramètres de la DLT, en tenant compte de la modélisation des erreurs optiques, nous devons résoudre un système de $2n$ équations à 16 inconnues, et à 11 inconnues dans le cas contraire (où n vaut le du nombre de points de calage utilisés dans l'ajustement). Le nombre de points de calage minimum nécessaires à la résolution des inconnues varie donc de 6 à 8 suivant que nous tenions compte ou non des erreurs optiques.

En développant les équations (6') et (6''), on trouve :

$$\frac{1}{R}u = \frac{1}{R}(L_1x + L_2y + L_3z + L_4 - L_9ux - L_{10}uy - L_{11}uz) + \Delta u \quad (7')$$

$$\frac{1}{R}v = \frac{1}{R}(L_5x + L_6y + L_7z + L_8 - L_9vx - L_{10}vy - L_{11}vz) + \Delta v \quad (7'')$$

Avec $R = L_9x + L_{10}y + L_{11}z + 1$

Si maintenant, nous généralisons ces expressions à l'utilisation simultanée de n points de calage et que l'on écrit le système sous la forme de matrice, nous déterminons les équations de bases qu'il faut résoudre pour un calcul de DLT :

$$\begin{bmatrix} x_1 & y_1 & z_1 & 1 & 0 & 0 & 0 & 0 & -u_1 x_1 & -u_1 y_1 & -u_1 z_1 \\ 0 & 0 & 0 & 0 & x_1 & y_1 & z_1 & 1 & -v_1 x_1 & -v_1 y_1 & -v_1 z_1 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x_n & y_n & z_n & 1 & 0 & 0 & 0 & 0 & -u_n x_n & -u_n y_n & -u_n z_n \\ 0 & 0 & 0 & 0 & x_n & y_n & z_n & 1 & -v_n x_n & -v_n y_n & -v_n z_n \end{bmatrix} \quad (8)$$

$$\begin{bmatrix} \xi_1 r_1^2 R_1 & \xi_1 r_1^4 R_1 & \xi_1 r_1^6 R_1 & (r_1^2 + 2\xi_1^2) R_1 & \xi_1 \eta_1 R_1 \\ \eta_1 r_1^2 R_1 & \eta_1 r_1^4 R_1 & \eta_1 r_1^6 R_1 & \eta_1 \xi_1 R_1 & (r_1^2 + 2\eta_1^2) R_1 \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \xi_n r_n^2 R_n & \xi_n r_n^4 R_n & \xi_n r_n^6 R_n & (r_n^2 + 2\xi_n^2) R_n & \xi_n \eta_n R_n \\ \eta_n r_n^2 R_n & \eta_n r_n^4 R_n & \eta_n r_n^6 R_n & \eta_n \xi_n R_n & (r_n^2 + 2\eta_n^2) R_n \end{bmatrix} \begin{bmatrix} L_1 \\ L_2 \\ \cdot \\ L_{15} \\ L_{16} \end{bmatrix} = \begin{bmatrix} u_1 \\ v_1 \\ \cdot \\ u_1 \\ v_1 \end{bmatrix}$$

Avec $\xi = u - u_0$, $\eta = v - v_0$ et $r^2 = \xi^2 + \eta^2$

$$\text{Où } u_0 = \frac{L_1 L_9 + L_2 L_{10} + L_3 L_{11}}{L_9^2 + L_{10}^2 + L_{11}^2} \text{ et } v_0 = \frac{L_5 L_9 + L_6 L_{10} + L_7 L_{11}}{L_9^2 + L_{10}^2 + L_{11}^2}$$

4.3.1.1. Résolution itérative

Lors de l'introduction des cinq paramètres additionnels, nous avons rendu le système non linéaire. Ceci implique une résolution par itérations successives du système.

En effet, nous voyons que certains éléments de la matrice des coefficients sont fonction d'éléments du vecteur des inconnues. Il s'agit des R_j et des coordonnées du point principal. Pour la première itération, nous choisirons d'initialiser l'ensemble du vecteur des inconnues à une valeur nulle (L'ensemble des R_j a une valeur unitaire). Quant au point principal, nous assimilerons ses coordonnées à celles du centre géométrique de l'image.

Lors des itérations suivantes, nous utilisons la valeur des paramètres qui viennent d’être calculés pour déterminer la valeur des R_j et des coordonnées du point principal. À l’itération i , pour le point de calage j , nous calculons donc :

$$R_j^i = L_9^{i-1} x_j + L_{10}^{i-1} y_j + L_{11}^{i-1} z_j + 1 \quad (9)$$

Et la valeur des coordonnées du point principal à l’itération i :

$$u_0^i = \frac{L_1^{i-1} L_9^{i-1} + L_2^{i-1} L_{10}^{i-1} + L_3^{i-1} L_{11}^{i-1}}{(L_9^{i-1})^2 + (L_{10}^{i-1})^2 + (L_{11}^{i-1})^2} \quad (10)$$

$$v_0^i = \frac{L_5^{i-1} L_9^{i-1} + L_6^{i-1} L_{10}^{i-1} + L_7^{i-1} L_{11}^{i-1}}{(L_9^{i-1})^2 + (L_{10}^{i-1})^2 + (L_{11}^{i-1})^2} \quad (11)$$

Le processus se poursuit jusqu’à ce que la différence entre les valeurs du vecteur des inconnues entre deux itérations successives soit inférieure à un certain seuil défini α

$$\max |L_i^j - L_i^{j-1}| \leq \alpha \quad (12)$$

Il s’agit d’un critère d’arrêt heuristique qui peut bien entendu montrer ses limites. Car même si la différence est proche de zéro, le calcul peut pour autant ne pas converger. Cependant l’expérience a montré que si α est assez petit, ce critère d’arrêt permet d’obtenir des résultats plus que satisfaisants.

4.3.1.2. Nombre d’observations surabondantes

Dans le cas où le nombre d’équations est supérieur au nombre d’inconnues, le système est sur déterminé. Cela correspond à 6 points de calage lors de la détermination des onze paramètres standard et 9 si nous prenons en compte les erreurs optiques. Ce système est résolu par la méthode dite des moindres carrés (ARNOULD 2003).

Si on considère de n mesures d’un phénomène, x_1, x_2, \dots, x_n entachées d’erreurs accidentelles qui valent respectivement $\Delta_1 = (x_1 - \mu), \Delta_2 = (x_2 - \mu), \dots, \Delta_n = (x_n - \mu)$ avec μ la moyenne qui obéit à une distribution gaussienne. La probabilité de commettre cette erreur Δ_i sur un mesure est donnée par :

$$\varphi(\Delta_i) = \frac{h_i}{\sqrt{\pi}} e^{-h_i^2 \Delta_i^2} d\Delta_i \quad (13)$$

Avec $h_i = \frac{1}{\sqrt{2}\sigma}$, le coefficient de précision qui caractérise la forme de la courbe de Gauss (Figure 11).

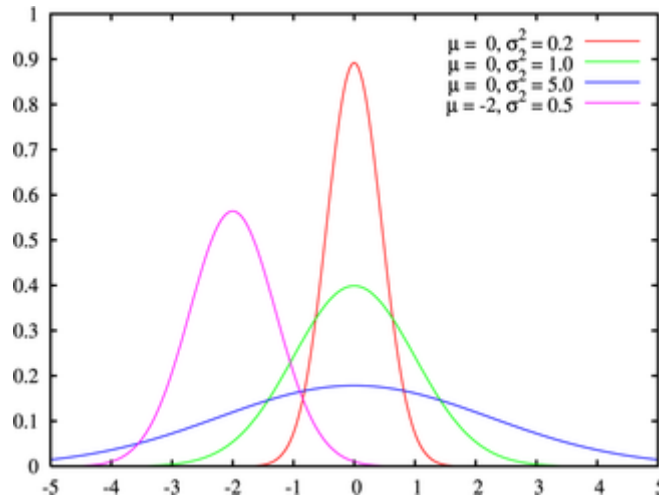


Figure 11. Représentation de différentes courbes de Gauss en fonction de la moyenne μ et la variance σ^2

(Source : www.wikipedia.com, consultation le 13 juillet 2006)

La méthode des moindres carrés consiste à déterminer parmi tous les groupements d'erreurs possibles $\Delta_1, \Delta_2, \dots, \Delta_n$ celui qui est le plus probable, dont la probabilité est maximale. Cette probabilité vaut :

$$\prod_{i=1}^n \varphi(\Delta_i) = \frac{h^n}{\sqrt{\pi}} e^{(-h_1^2 \Delta_1^2 - h_2^2 \Delta_2^2 - \dots - h_n^2 \Delta_n^2)} d\Delta_1 . d\Delta_2 \dots d\Delta_n \quad (14)$$

Dans les cas qui nous occupent, toutes les mesures sont d'égale précision, donc $h_i = \text{constante } \forall i$. Et la fonction [équation (14)] possède un maximum lorsque

$$\Delta_1^2 + \Delta_2^2 + \dots + \Delta_n^2 = \text{minimum} \quad (15)$$

On trouve en conséquence que la valeur la plus probable de la grandeur recherchée est celle qui minimise le somme des carrés des erreurs.

Appliquons maintenant ce qui vient d'être exposé à la résolution de l'équation (8). Au sens des moindres carrés, pour trouver la solution la plus probable pour le vecteur des inconnues, il faut minimiser la somme des erreurs Δ dont sont entachées les observations (vecteur B) :

$$\begin{aligned} AL &= B + \Delta \\ \Leftrightarrow \Delta &= AL - B \end{aligned} \quad (16)$$

Où A est la matrice des coefficients

B est le vecteur des termes indépendants

L est le vecteur des inconnues

Δ est le vecteur des erreurs accidentelles

Et les valeurs du vecteur des inconnues L qui rendent la somme des carrés des erreurs (Δ) minimale est :

$$L = (A \cdot A')^{-1} \cdot A' \cdot B \quad (17)$$

C'est la solution par moindres carrés des équations [équation (8)] de la DLT.

4.3.2. Reconstruction tridimensionnelle

Maintenant que les paramètres sont déterminés pour chaque cliché, nous pouvons passer à l'étape suivante, la restitution. Une des caractéristiques de la DLT est de ne pas restreindre à deux le nombre de clichés qui peuvent être utilisés lors de la restitution. En redeveloppant une nouvelle fois des équations (6') et (6'') et en posant $v = u - \Delta u$ et $\omega = v - \Delta v$, où u et v sont les coordonnées du point à calculer, nous trouvons :

$$\frac{1}{R}(vL_9 - L_1)x + \frac{1}{R}(vL_{10} - L_2)y + \frac{1}{R}(vL_{11} - L_3)z = \frac{1}{R}(L_4 - v) \quad (18')$$

$$\frac{1}{R}(\omega L_9 - L_5)x + \frac{1}{R}(\omega L_{10} - L_6)y + \frac{1}{R}(\omega L_{11} - L_7)z = \frac{1}{R}(L_8 - \omega) \quad (18'')$$

Avec $R = L_9x + L_{10}y + L_{11}z + 1$

Si on généralise à l'utilisation de m clichés de références et que nous écrivons le système sous la forme de matrice :

$$\begin{bmatrix} \frac{\omega^1 L_9^1 - L_5^1}{R^1} & \frac{\upsilon^1 L_{10}^1 - L_2^1}{R^1} & \frac{\upsilon^1 L_{11}^1 - L_3^1}{R^1} \\ \frac{\omega^1 L_9^1 - L_5^1}{R^1} & \frac{\omega^1 L_{10}^1 - L_6^1}{R^1} & \frac{\omega^1 L_{11}^1 - L_7^1}{R^1} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \frac{\omega^m L_9^m - L_5^m}{R^m} & \frac{\upsilon^m L_{10}^m - L_2^m}{R^m} & \frac{\upsilon^m L_{11}^m - L_3^m}{R^m} \\ \frac{\omega^m L_9^m - L_5^m}{R^m} & \frac{\omega^m L_{10}^m - L_6^m}{R^m} & \frac{\omega^m L_{11}^m - L_7^m}{R^m} \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{L_4^1 - \upsilon^1}{R^1} \\ \frac{L_8^1 - \omega^1}{R^1} \\ \cdot \\ \cdot \\ \frac{L_4^m - \upsilon^m}{R^m} \\ \frac{L_8^m - \omega^m}{R^m} \end{bmatrix} \quad (19)$$

Avec $R^j = L_9^j x + L_{10}^j y + L_{11}^j z + 1$ propre à chaque cliché j .

Cette procédure doit être effectuée autant de fois qu'il y a de points à restituer.

4.3.2.1. Résolution itérative

Nous nous retrouvons, ici, dans la même situation que pour la résolution de l'équation (8). L'élément R^j de la matrice des inconnues est fonction du vecteur des inconnues. Nous initialisons leurs valeurs à : $x = 1000$, $y = 1000$, $z = 1000$. Pour les itérations suivantes, nous nous servons des valeurs du vecteur des inconnues qui viennent d'être déterminées. Dès lors, nous déterminons la valeur de R , pour le cliché j à l'itération i ($i > 1$) de la façon suivante :

$$R_i^j = L_9^j x_{i-1} + L_{10}^j y_{i-1} + L_{11}^j z_{i-1} + 1 \quad (20)$$

Une nouvelle fois le processus se poursuit jusqu'à ce que la différence entre deux itérations successives des valeurs du vecteur des inconnues soit inférieure à un certain seuil (cf. § 4.3.1.1).

4.3.2.2. Résolution d'observations surabondantes

Nous appliquons exactement le même raisonnement que celui qui a été démontré pour la résolution des paramètres de la DLT (cf. § 4.3.1.2).

4.4. Compensation globale

La méthode de restitution par voie photogrammétrique développée dans ce chapitre est destinée à des travaux de levé dans les domaines de l'architecture et plus généralement de l'urbanisme. Un tel levé ne peut, en général, être effectué à partir d'un seul couple (triplet ou plus) de clichés photographiques. L'exemple type est celui d'un levé complet des quatre façades d'un bâtiment, il nécessitera un minimum de quatre couples.

Cette constatation introduit un nouveau problème, la compensation globale de l'ensemble du modèle.

4.4.1. Compensation à partir du nuage de points

De nombreuses méthodes existent si l'on aborde le problème de la compensation globale par rapport aux nuages de points obtenus. Nous considérons alors que chaque nuage associé à un couple stéréoscopique est dans un système de coordonnées différent. À partir de points homologues, nous reformons un nuage complet composé par les autres nuages. Cette technique est souvent appliquée lorsque ce sont des semis de points non géo référencés qui sont traités (cas du scannage laser).

Une méthode récente de compensation globale est la Least Square 3D (LS3D) (AKCA & GRUEN 2005). Elle consiste à minimiser l'écart entre les deux surfaces définies par les semis de points. Elle est souvent utilisée pour la reconstruction tridimensionnelle d'objets obtenus à partir de semis de points qui ne sont pas géo référencés (pas de mise en station de l'appareil)

Une autre méthode plus classique nécessite de s'appuyer sur des points communs à plusieurs nuages et d'appliquer une transformation (type affine) à un des deux nuages. Dans le cas de l'utilisation de scanner laser, ces points homologues sont souvent matérialisés par des prismes ou des cibles de réflectance particulière reconnues par le scanner.

4.4.2. Cas de la DLT

En ce qui concerne les méthodes photogrammétriques, il est préférable d'appliquer la compensation avant d'obtenir le semi de points et d'agir directement sur les paramètres d'orientation des clichés.

Au premier abord, en suivant les fondements de la théorie de la DLT, si les points qui servent aux calages des couples photogrammétriques sont dans le même référentiel, il n'y a pas de raison que les points restitués ne la soient pas. Après quelques recherches, Nous trouvons plusieurs références bibliographiques traite de ce sujet.

Dans KRAUS (1997), il est spécifié dans la théorie relative à la DLT que les paramètres défini par cet d'ajustement peuvent servir de base pour réaliser une triangulation globale. Nous trouvons aussi les mêmes considérations pour les paramètres de la DLT dans REMONDINO (2002) ainsi que dans les références dédiées au logiciel ARPENTEUR (cf. § 3.4.1.3).

Dans le cadre de notre application de la DLT pour des travaux urbanistiques qui requièrent des précisions moins importantes, nous considérerons la précision des paramètres de la DLT comme étant assez importante pour ne pas appliquer d'autres traitements supplémentaires aux points qui seront restitués. Nous réaliserons donc une polygonale autour de chacun des bâtiments étudiés afin de garantir à l'ensemble des points de calage d'être dans le même système de coordonnées.

5. Mise en oeuvre de la DLT

5.1. Introduction

Ce chapitre expose les différentes étapes du processus qui ont permis d'aboutir au développement de notre application (cf. § 5.6). Il est composé premièrement par une mise en œuvre sur le terrain et le développement d'une méthodologie spécifique. Ensuite, nous proposerons une optimisation de l'algorithme de calcul des paramètres de la DLT et plus particulièrement de l'inversion de matrice lors du processus par moindres carrés. Et finalement, nous développerons les principales caractéristiques de l'application que nous avons programmée.

5.2. Technique de validation

Il nous semble opportun de définir à ce niveau, avant toutes applications ultérieures, nos choix en matière de validation de notre travail de levé ainsi que ceux que nous avons fait pour obtenir une référence. Cette référence sera considérée comme vérité terrain et nous supposerons qu'il n'y a aucune erreur associée à ces mesures ou du moins que les erreurs sont négligeables par rapport à notre technique de levé.

5.2.1. Mesures laser

La mise en œuvre de la DLT nécessite la mesure de points de calage dans un référentiel terrain. De plus, pour permettre de valider la reconstruction 3D dans ce référentiel, une série de points de contrôle doit aussi être mesurée. L'écart qui existe entre cette mesure et les coordonnées restituées par la DLT nous permet d'obtenir de bonnes indications sur la précision de notre levé.

Comme le nombre de points à mesurer est assez conséquent, nous utiliserons une station totale équipée d'un distancemètre laser. Une méthode par intersection pourrait aussi être appliquée, mais elle serait plus fastidieuse à mettre en œuvre. En effet, le temps utile pour réaliser les mesures serait au moins multiplié par deux. Toutefois, dans le cadre d'un travail de recherche ultérieur, il serait certainement très intéressant de vérifier la

précision de mesures réalisées à l'aide d'un distancemètre laser, afin de déterminer si le gain de précision sur les paramètres de la DLT est significatif.

5.2.2. Méthode de mesure

Pratiquement, nous utiliserons une station totale Leica TCRA 1103 plus, robotisée et équipée d'un distancemètre laser. Son utilisation induit un certain nombre d'automatismes que l'opérateur topographe doit acquérir. Nous profiterons de la première campagne de mesure menée sur le bâtiment d'Astrophysique de l'Université de Liège pour nous familiariser avec l'utilisation du laser.



Figure 12. Leica TCRA 1103 +

Parmi les techniques utiles, on retrouve le *rabattement*. Si, par exemple, nous devons mesurer un point situé exactement sur l'arête d'un mur, nous veillerons à mesurer la distance sur un autre point contenu dans le même plan horizontal puis *rabattre* sur le point à mesurer en agissant exclusivement sur le mouvement horizontal (Figure 13).

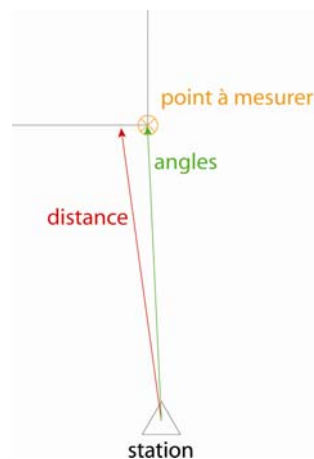


Figure 13. Schéma explication de la méthode du rabattement

5.3. Méthodologie

Pour se familiariser avec l'application d'un calcul de DLT, une première campagne de mesures est réalisée sur le bâtiment d'astrophysique (B5c) de l'Université de Liège. Ce site a tout d'abord été choisi pour sa proximité, pour son accessibilité et, enfin, parce qu'il permet de répondre facilement à la condition de non coplanéarité des points de contrôle (Figure 14). C'est en effet une condition sine qua non pour permettre la résolution des paramètres de la DLT (cf. § 3.3).



Figure 14. Site de mise en œuvre de la DLT
Bâtiment B5c de l'Université de Liège

Nous souhaitons déterminer l'entière des paramètres de la DLT (les 11 paramètres standard, les 3 paramètres de distorsions de la lentille et les 2 paramètres de décentrement du point principal) à l'aide de trois clichés. La résolution de ces 16 inconnues, pour chacun des clichés, nécessite la connaissance d'au moins 8 points de calage. Un tableur programmé sous Excel est développé dans ce but. Il permet de prendre en compte trois clichés et 15 points de calage. La résolution de ce système se fait par moindres carrés puisque nous disposons de 30 équations d'observation et de 16 inconnues (un autre tableur est programmé, il permet de résoudre le système de manière stricte et donc de n'utiliser que 8 points de calage).

En suivant la méthodologie décrite dans LAPLANCHE (2005), nous avons prévu deux sorties distinctes. La première se déroule le lundi 7 novembre 2005 entre 13h15 et 13h30. Sa faible durée s'explique par une très bonne connaissance de la zone étudiée. En effet, elle a consisté à choisir la meilleure position de l'appareil photographique pour prendre les

clichés. Les conditions climatiques étaient moyennes, le temps était gris et nuageux, mais la lumière était assez forte pour permettre une bonne exploitation des clichés.

5.3.1. Prises de vues

Trois clichés de la façade seront pris sous des angles différents pour pouvoir analyser la méthode de la DLT en utilisant deux ou trois images (Annexe 3). On respectera un angle minimum d'environ 30 degrés entre l'alignement de la façade et l'axe optique de l'appareil, et on veillera aussi à placer l'appareil sur un pied pour augmenter au maximum sa stabilité lors de la prise de vue (LAPLANCHE 2005). L'appareil utilisé est un compact digital SONY DSC-W12, sa résolution est de $5,04 \cdot 10^6$ pixels (Tableau 3). La longueur approximative de focale est de 8 millimètres (donnée Exif). Les clichés sont automatiquement compressés en format jpeg. Notons qu'il n'est pas possible de sélectionner un autre mode de stockage qui permettrait de diminuer les pertes de qualité.

Pour faciliter l'identification des points homologues, nous avons imprimés les trois clichés en format A3. L'impression en grand format nous a permis de sélectionner 20 points bien identifiables sur chacune des photographies, en respectant la condition de non planéarité des points de calage.

Une fois tous les points identifiés, nous les avons recensés dans un inventaire. Chaque point y est décrit à l'aide d'un cliché rapproché et d'une phrase courte et précise aidant à sa localisation sur la façade. Il y est aussi reconnu, de manière univoque, par un identifiant alphanumérique (Annexe 3).

La mesure des coordonnées image a été réalisée à l'aide du logiciel *DDPS*. Cependant, un traitement préalable des photos a dû être envisagé, puisque ce logiciel ne permet de travailler qu'avec des images en niveaux de gris codées en Bitmap sur 8 bits par pixel. Les images ont été recodées en format Bitmap avec le logiciel *Adobe Photoshop CS*.

5.3.2. Mesure des points de calage

Nous avons levé les points de calage le 18 novembre à 14h15. Nous avons mis 2 heures pour nous familiariser avec les règles d'utilisation du laser et procéder au levé de 20 points sur la façade.

Un total de 20 points de calage a été mesuré dans un système de coordonnées local arbitraire (Annexe 4 et Annexe 14). Une seule station a été nécessaire pour réaliser le levé de l'ensemble des points. Ces coordonnées exprimées en mètres ont été fixées à :

$$X=1000, Y=1000, Z=100$$

À partir de cette station, quatre autres points sont mesurés. Ils seront très utiles si d'autres points doivent être mesurés ultérieurement sur la façade dans le même système de coordonnées local. Nous pourrions alors déterminer les coordonnées de la nouvelle station comme étant fonction des mesures angulaires et des coordonnées des quatre points.

Ce problème est généralement connu sous le nom de problème de *Pothenot*. Il consiste à déterminer les coordonnées de la nouvelle station à partir des observations faites sur au moins trois autres points. Dans notre cas, nous disposerions en plus des coordonnées approchées de la station, nous pouvons appliquer la méthode par variation de coordonnées. Elle se résout simplement et présente l'avantage de pouvoir utiliser un nombre d'observations surabondantes (si nous visons plus de trois points connus). Le rattachement sur plus de trois points est recommandé dans ARNOULD (2000) afin d'augmenter la précision.

5.4. Calcul

5.4.1. Traitement des données

Pour déterminer les paramètres de la DLT, nous avons préalablement programmé un tableur sous le logiciel Excel. La programmation de ce tableur est une étape préalable indispensable au développement d'une application plus conséquente. Nous pourrions y identifier les éventuels problèmes, ainsi que les subtilités de la DLT avant de nous lancer dans un travail de programmation plus approfondi.

En réalité ce sont trois tableurs qui sont programmés (Annexe 5 et Annexe 14). Ils permettent de prendre en compte plusieurs cas d'utilisation de la DLT. Leurs caractéristiques sont décrites dans le tableau (Tableau 2) ci-dessous :

Tableau 2. Caractéristiques des tableurs permettant la résolution des paramètres de la DLT et la restitution

	<i>Nombre points de calage</i>	<i>Modélisation des erreurs optiques</i>	<i>Restitution</i>	<i>Nombre de clichés</i>
<i>dlt_8pt.xls</i>	8	non	oui	2 ou 3
<i>dlt_disto_8pt.xls</i>	8	oui	oui	2 ou 3
<i>dlt_disto_15pt.xls</i>	16	oui	oui	2 ou 3

Notons que dans chaque tableur, une *macro* est programmée en Visual Basic. Elle permet d’automatiser un certain nombre de tâches assez fastidieuses inhérentes à l’utilisation du logiciel (cf. § 5.4.2).

5.4.2. Exploitation des données

Comme nous l’avons déjà expliqué, la résolution de l’équation (8) doit se faire par itérations successives (les coefficients sont, en effet, fonction des inconnues). Dans Excel, nous activons dans l’onglet *Calcul* du menu *Options*, la case à cocher *Itération*, afin de pouvoir résoudre les équations. Le calcul est déclenché lorsque l’utilisateur introduit une *référence circulaire* dans la feuille Excel. Pratiquement, elle apparaît lorsque nous égalons la valeur des paramètres calculés aux paramètres initiaux.

La construction de la matrice initiale des coefficients nécessite de connaître une valeur approchée des paramètres de la DLT (L_1, \dots, L_{11} ou L_{16}) ainsi que les coordonnées du point principal de l’image (u_0, v_0) . Nous choisissons initialement la valeur nulle pour les paramètres de la DLT et nous assimilons les coordonnées du point principal aux coordonnées du centre géométrique de l’image (LAPLANCHE, 2004). Soit :

$$L_i = 0 \quad i \in \{1, \dots, 11\} \quad \text{ou} \quad i \in \{1, \dots, 16\} \quad (21)$$

Où les L_i sont les paramètres de la DLT.

$$(u_0, v_0) = \left(\frac{u_{\max} - u_{\min}}{2}, \frac{v_{\max} - v_{\min}}{2} \right) \quad (22)$$

Où u_{\max}, v_{\max} sont respectivement l'abscisse et l'ordonnée maximales, et u_{\min}, v_{\min} les minimales.

Comme nous le disions plus haut, une *macro* est programmée dans chaque tableur. Une fois que les coordonnées des points de calage (image et terrain) des deux ou trois clichés sont introduites et que les calculs sont initialisés (référence circulaire), la combinaison des touches *ctrl+d* permet de calculer les paramètres et de les copier dans la feuille consacrée à la restitution.

Les différents tableurs sont testés sur base des données acquises sur le bâtiment B5c de l'Université de Liège. Malheureusement, les résultats sont loin d'être satisfaisants, le calcul itératif des paramètres ne semble pas converger correctement.

5.4.3. Identification du problème

Les problèmes rencontrés pendant le test sur les données acquises sur le bâtiment d'astrophysique de l'Université de Liège (B5c) se sont à nouveau présentés lors de l'introduction des valeurs utilisées dans le travail de LAPLANCHE (2004). Ce ne sont donc pas nos données qui sont la cause de ce problème numérique. Dès lors, nous avons entrepris une large analyse de tout le processus de la détermination des paramètres de la DLT.

Après avoir pris connaissance des principales origines d'instabilités numériques dans le calcul matriciel, nous avons identifié la cause probable des problèmes rencontrés. Il s'agirait l'inversion de matrice lors de la détermination des paramètres de la DLT par moindres carrés. En effet, il s'agit d'une difficulté récurrente dans le domaine de l'analyse numérique et source de nombreuses instabilités numériques.

5.5. Optimisation de calcul

5.5.1. Introduction

La mise au point et l'analyse de résultats obtenus avec le tableur programmé sous Excel nous a permis d'identifier un problème important. Il apparaît lors de la détermination des paramètres de la DLT [équation (6)] par moindres carrés. En effet, nous avons remarqué lors de l'inversion du produit de la transposée matrice des coefficients avec elle-même, qu'il apparaissait une instabilité numérique [équation (17)]. C'est à la suite de

l'identification du problème rencontré que nous avons entrepris des recherches dans le domaine du calcul matriciel et plus précisément dans l'inversion de matrice.

Nos recherches nous ont mené à identifier trois méthodes d'inversion. Elles sont toutes les trois basées sur des concepts mathématiques différents. La première, la méthode de *Gauss Jordan*, apparaît comme étant une méthode classique utilisée couramment en analyse numérique. Son principe est simple et est basé sur l'algorithme de *Gauss* qui permet de résoudre les systèmes d'équations. La deuxième construit par une série d'itérations la pseudo inverse d'une matrice. Elle permet donc l'inversion, au sens de *Moore Penrose*, de matrices non carrées. Et la dernière consiste à construire l'inverse d'une matrice, à partir de sa décomposition en deux autres matrices. Ces deux matrices présentent, en effet, des propriétés qui les rendent facilement inversibles.

Nous allons présenter dans ce chapitre, les différentes méthodes d'inversion de matrice qui vont être implémentées, testées et ajoutées à la librairie de calcul matriciel qui a été créée pour le logiciel *DDPS*. Chacune des méthodes sera expliquée et commentée. Un algorithme accompagnera cette description, il permettra de mieux comprendre le raisonnement mathématique qui amène à l'inversion.

Remarque : pour une meilleure lisibilité, toutes les équations de ce chapitre sont notées en notation matricielle.

5.5.2. Méthode de Gauss Jordan

5.5.2.1. Présentation

Nous avons utilisé l'algorithme qui était déjà implémenté dans la librairie de calcul matriciel du logiciel *DDPS*. C'est pourquoi nous ne nous attarderons pas trop longuement sur les détails de cette méthode.

Il s'agit d'un algorithme facilement implémentable. Toutefois certaines informations tendent à montrer que l'algorithme est numériquement instable, notamment à cause des nombreuses opérations élémentaires qu'il engendre sur des nombres arrondis par la machine. Leur nombre, pour l'inversion d'une matrice carrée de dimension n vaut :

$$\frac{16n^3 - 9n^2 - n}{6} \sim \frac{8}{3}n^3 \text{ opérations élémentaires.}$$

5.5.2.2. Algorithme

Pour inverser la matrice carrée A par la méthode de *Gauss Jordan* nous suivons l'algorithme suivant :

```

For i=1 to n
  If  $a_{kk}^{k-1} \neq 0$ 
     $I_k^k = \frac{1}{a_{kk}^{k-1}} I_k^{k-1}$  ;
    For j=1 to n and  $i \neq j$ 
       $I_i^k = I_i^{k-1} - a_{ik}^{k-1} I_k^k$  ;
    Else matrice  $A$  n'est pas inversible
  
```

5.5.2.3. Convergence

Malheureusement la méthode ne s'est pas montrée efficace dans l'inversion de la matrice $A^t \cdot A$ pour le calcul des paramètres de la DLT.

Pour illustrer le propos, nous avons calculé une fonction montrant la convergence d'un processus itératif [équation (23)]. Nous allons l'appliquer pour du calcul des paramètres de la DLT par moindres carrés utilisant l'inversion de matrice par la méthode de *Gauss Jordan*. Nous avons défini un indice de convergence à la j^{eme} itération χ_j par l'équation suivante :

$$\chi_j = \alpha \sqrt{\sum_{i=1}^{i \leq n} |L_i^{j-1} - L_i^j|} \quad \text{avec } \alpha = \begin{cases} 1 & \text{si } j \text{ est impair} \\ -1 & \text{si } j \text{ est pair} \end{cases} \quad (23)$$

Où j est le nombre d'itérations, n le nombre de paramètres calculés.

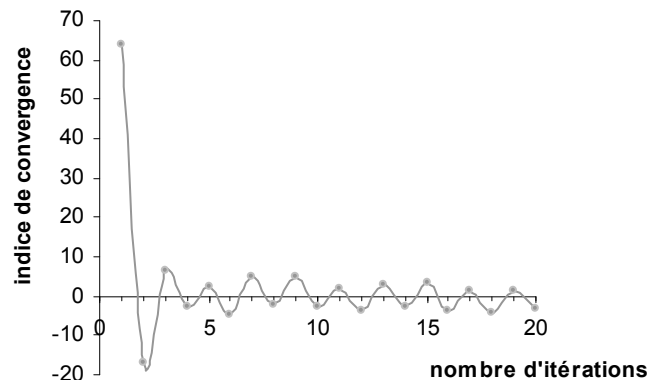


Figure 15. Représentation graphique de l'indice de convergence : Gauss Jordan

Inversion de matrice durant le processus de détermination des paramètres de la DLT

L'analyse de la représentation graphique de notre indice de convergence montre bien l'impossibilité, pour l'algorithme de *Gauss Jordan*, de faire converger le calcul de paramètres de la DLT. La cause est, sans doute, les petites imprécisions lors de la construction de la matrice inverse.

Remarque : le calcul de la fonction de convergence est présenté en annexe (Annexe 6).

5.5.3. Méthode de Gréville

5.5.3.1. Présentation

Les méthodes classiques d'inversion de matrice se ramènent, soit à résoudre un ensemble de systèmes linéaires, soit à généraliser la méthode du pivot de *Gauss Jordan* (cf. § 5.5.2), soit à utiliser des identités algébriques (Cayley-Hamilton ou série de Neumann). L'algorithme conçu par *Gréville* construit la matrice A^{-1} , ligne par ligne, à partir des colonnes de A . Son principal intérêt vient du fait qu'il peut aussi fonctionner pour des matrices non inversibles, conduisant ainsi à la notion de pseudo inverse ou inverse généralisée de *Moore Penrose*.

La définition de la matrice généralisée de *Moore Penrose* :

Soit A une matrice, alors A^+ est la matrice inverse généralisée de *Moore Penrose* de A si :

$$\begin{aligned}
 AA^+A &= A \\
 A^+AA^+ &= A^+ \\
 (A^+A)^t &= A^+A \\
 (AA^+)^t &= AA^+
 \end{aligned}
 \tag{24}$$

5.5.3.2. Algorithme

L'algorithme d'inversion de la matrice A par la méthode de *Gréville* présenté ici, suit les notations suivantes :

- a_m la colonne m de A ;
- A_m la matrice formée par les m premières colonnes de A ;
- A_m^+ les m premières lignes de la pseudo inverse de A ;
- n le nombre de lignes.

$$A_1 = a_1$$

If $a_1 \neq 0$

$$A_1^+ = a_1^t (a_1^t a_1)^{-1} ;$$

El se 0^t

For $i=2$ to n

$$r_i = a_i - A_{i-1} A_{i-1}^+ a_i ;$$

$$p_i = \frac{1}{r_i^t r_i} r_i^t ;$$

$$A_i^+ = \begin{pmatrix} A_{i-1}^+ - A_{i-1}^+ a_i p_i \\ p_i \end{pmatrix} ;$$

End;

5.5.3.3. Convergence

Encore une fois, l'algorithme ne s'est pas révélé assez efficace pour permettre de faire converger correctement le calcul. Pour l'illustrer, nous proposons une représentation graphique de notre indice de convergence (cf. § 5.5.2.3). Nous l'avons déterminé à partir des données qui ont servi pour la méthode de *Gauss Jordan*.

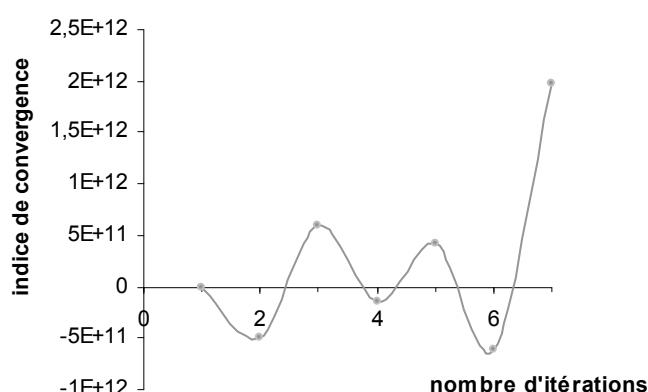


Figure 16. Représentation graphique de l'indice de convergence : Gréville

Inversion de matrice durant le processus de détermination des paramètres de la DLT

Bien que cette représentation graphique de l'indice de convergence semble fort peu probable, elle est tout de même correcte. Nous avons aussi été surpris par ces résultats, et nos expérimentations ont été poursuivies. Un tableur est programmé sous Excel (Annexe 14) et une application en langage C est spécifiquement développée dans le but de valider notre algorithme.

Ils ont, tous les deux, révélé des résultats très satisfaisants uniquement dans le cadre d'inversion de matrices carrées de plus petites tailles. Nous en avons conclu à une incapacité de la méthode de *Gréville* à inverser des matrices de grandes tailles et donc à permettre la détermination des paramètres de la DLT.

Remarque : le calcul de la fonction de convergence est présenté en annexe (Annexe 6).

5.5.4. Décomposition QR

5.5.4.1. Présentation

Le principe d'une décomposition de matrice, est de construire à partir d'une matrice que nous appellerons A , deux matrices dont le produit vaut la matrice A . Il existe de nombreuses décompositions (LU , *Choleski*,...), et parmi elles, on retrouve la décomposition QR . Il s'avère que cette dernière possède des caractéristiques particulièrement intéressantes pour notre application. Au niveau de son implémentation, elle présente l'avantage d'être rapide, numériquement très stable et permet aussi de gérer aisément des problèmes de grande taille ($n \gg m$) (GOVEARTS 2001). Mais elle est aussi intéressante pour les propriétés des matrices qui sont construites (la matrice Q et la matrice R).

L'inverse de la matrice A est alors calculé à partir des deux matrices, en effet on a [équation (25)] :

$$A^{-1} = Q^{-1} \cdot R^{-1} \quad (25)$$

Nous voyons que si on peut déterminer facilement l'inverse des matrices Q et R , il est facile de calculer l'inverse de la matrice A . Or, nous allons montrer que le calcul de Q^{-1} et R^{-1} est effectivement très simple. Cela découle directement de propriétés de ces deux matrices.

5.5.4.2. Propriétés des matrices Q et R

Si A est une matrice de dimension $n \times m$ (avec $n \geq m$) alors,

- La **matrice Q** est orthogonale de dimension $n \times m$. Une matrice est dite orthogonale si toutes les lignes ont une norme qui vaut 1, et de même pour les colonnes.

Si A est une matrice orthogonale, $A \cdot A^t = I$ où I est la matrice identité.

Cette propriété est intéressante dans le domaine de l'inversion de matrice, puisque l'inverse d'une matrice orthogonale est sa transposée. Et la transposée d'une matrice est construite très facilement : les lignes de la matrice transposée sont les colonnes de la matrice de départ, et les colonnes de la transposée sont les lignes de la matrice de départ.

- La **matrice** R est triangulaire supérieure $m \times m$. Une matrice est triangulaire supérieure si tous les éléments de la sous matrice triangulaire inférieure sont nuls. Soit, $a_{i,j} = 0 \quad \forall i > j$.

Il s'agit aussi d'une propriété intéressante pour inverser ce type de matrice, conformément au raisonnement suivant (inspiré de SCHNEIDERS 2005)

Posons $S = R^{-1}$. Le produit de R et S vaut la matrice identité, ce qui correspond à l'égalité suivante :

$$\sum_{p=j}^m r_{jp} s_{pk} = \delta_{jk} \quad (j, k \in \{1, \dots, m\}) \quad (26)$$

On en tire que :

$$s_{jk} = \left(\delta_{jk} - \sum_{p=j+1}^n r_{jp} s_{pk} \right) / r_{jj} \quad (27)$$

Ce qui permet de calculer les lignes de S (inverse de R) par récurrence descendante sur leur indice. Comme S est aussi triangulaire supérieure, son calcul complet demande

$$\frac{4n^3 + 3n^2 - n}{6} \sim \frac{2}{3} n^3 \text{ opérations élémentaires.}$$

5.5.4.3. Algorithme

Plusieurs méthodes amènent à la décomposition QR d'une matrice. On trouve notamment la méthode de *Householder*, celle des *Givens rotation* ainsi que celle de *Gram-Schmit*. Sachant que la première est souvent plus stable que les autres, notre choix s'est naturellement porté sur elle. Nous présentons ci-après l'algorithme utilisé dans l'application.

X= première colonne de A_i
e= vecteur 1, 0, 0, ..., 0 ;

```
For i=1 to n-1
  normX= norme de X;
  vecteur U = X- e*norme;
  normU= norme de X;
  vecteur V=U/normU;

   $Q_i = I - 2 * V * V^t$ ;
   $A_i = Q_i * A$ ;
   $Q_f = Q_i$  ;
End;
```

Une fois la décomposition réalisée, nous devons encore calculer les inverses des matrices Q et R . Pour Q c'est très simple, puisqu'il s'agit d'une matrice orthogonale, son inverse est égal à sa transposée. Nous utiliserons dès lors la fonction disponible dans la librairie de calcul matriciel de *DDPS*. Quant à l'inversion de R , la méthode utilisée est celle qui est présentée plus haut. Elle ne présente pas de difficulté particulière pour la programmation de l'algorithme (cf. § 5.5.4.2).

5.5.4.4. Convergence

La représentation graphique de l'indice qui a été introduit plus haut (cf. § 5.5.2.3) montre, une fois de plus, l'impossibilité des méthodes d'inversion qui sont présentées dans ce chapitre.

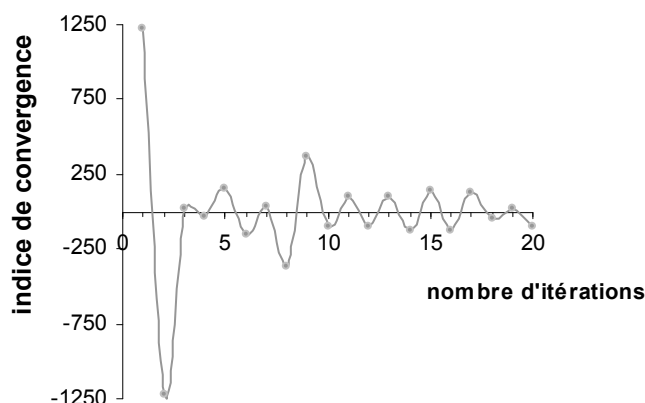


Figure 17. Représentation graphique de l'indice de convergence : QR

Inversion de matrice durant le processus de détermination des paramètres de la DLT

Remarque : le calcul de la fonction de convergence est présenté en annexe (Annexe 6).

5.5.5. Conclusion

Les méthodes d'inversion de matrice (cf. § 5.5.2 ; cf. § 5.5.3 et cf. § 5.5.4) ont été testées dans une application implémentée en langage C. Les données utilisées sont celles qui ont été acquises lors de la campagne de mesures sur le bâtiment B5c (cf. § 5.3). Malheureusement, aucune d'entre elles ne s'est avérée efficace dans la résolution d'un calcul de DLT. Il nous a été impossible d'assurer la convergence du processus itératif.

La portée des recherches a donc été élargie pour pouvoir trouver une issue favorable à ce problème. Le but n'est plus maintenant de trouver une méthode d'inversion de matrice compatible avec les exigences numériques demandées lors de la détermination des paramètres de la DLT. Nous allons contourner le problème qui s'est posé et montrer une résolution du système par moindres carrés ne faisant pas intervenir d'inversion de matrice.

5.5.6. La décomposition QR et les moindres carrés

5.5.6.1. Présentation

Lors de nos recherches portant sur la décomposition QR , les propriétés intéressantes des matrices Q et R ont été mises en avant (cf. § 5.5.4.2). La décomposition QR d'une matrice A construit deux matrices :

- Q une matrice orthogonale ;
- R une matrice triangulaire supérieure.

Si on remplace dans l'équation des moindres carrés [équation (17)] la matrice A par sa décomposition QR et si on utilise les propriétés des deux matrices substituées à A , on trouve

$$L = \left((QR)^t \cdot QR \right)^{-1} \cdot (QR)^t \cdot B \quad (28)$$

La transposée d'un produit de deux matrices est le produit des transposées, donc :

$$L = \left(Q^t R^t \cdot QR \right)^{-1} \cdot Q^t R^t \cdot B \quad (29)$$

Le produit d'une matrice orthogonale par sa transposée donne la matrice identité :

$$L = \left(R^t \cdot R \right)^{-1} \cdot Q^t R^t \cdot B \quad (30)$$

Le produit d'une matrice par son inverse donne la matrice identité :

$$L = R^{-1} \cdot Q^t \cdot B \quad (31)$$

Par cette démonstration, on montre que la résolution par moindres carrés du système suivant où le nombre d'équations est surabondant :

$$Q \cdot R \cdot L = B \quad (32)$$

Où $Q \cdot R$ est la décomposition QR d'une matrice A

Equivalut à résoudre le système d'équation triangulaire suivant [(équation (33)) :

$$R \cdot L = Q^t \cdot B \quad (33)$$

Où R est une matrice carrée triangulaire supérieure de dimension m x m ;

L est le vecteur des inconnues de dimension m x 1 ;

Q est une matrice de dimension n x m ;

B est le vecteur des termes indépendant de dimension n x 1.

La méthode de résolution de ce type de système est simple. Nous la retrouvons dans la littérature sous la dénomination de *back substitution*. Elle consiste à déterminer les valeurs des inconnues L, une par une, de manière ascendante (en commençant de L_{16} jusqu'à L_1).

5.5.6.2. Algorithme *back substitution*

Après avoir décomposé la matrice A (cf. § 5.5.4.3), il nous reste à résoudre le système [équation (33)]. Si on respecte la dénomination des matrices telle qu'utilisée précédemment, on peut écrire l'algorithme de la façon suivante :

```

For i=m to 1
  Xi =0;
  For j=m to i
    Xi =Xi -Ri,j *Lj;
  End;
  Li=(QtB)i -Xi /Ri,i;
End;
```

Nous pouvons noter que par rapport à la résolution par moindres par la méthode traditionnelle [équation (17)], nous gagnons le calcul de l'inverse de R et de trois produits matriciels en utilisant la décomposition QR (cf. § 5.5.6).

5.5.6.3. Convergence

Nous avons, une nouvelle fois, construit le graphe de notre indice de convergence (cf. § 5.5.2.3). Les calculs ont, évidemment, été menés sur base de données identiques.

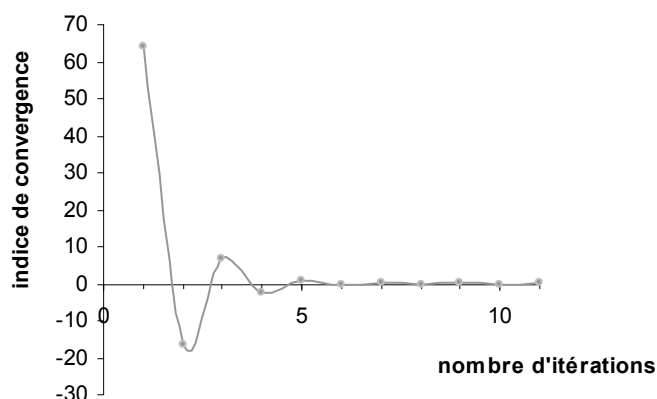


Figure 18. Représentation graphique de l'indice de convergence : décomposition *QR*
Détermination des paramètres de la DLT par décomposition *QR*

La convergence est, dans ce cas, très nette et rapide. Elle s'explique par une très bonne stabilité numérique de l'algorithme lors de la détermination des paramètres par moindres carrés.

Remarque : le calcul de la fonction de convergence est présenté en annexe (Annexe 6).

5.5.6.4. Validation

Cette méthode a permis une convergence rapide et stable du calcul des paramètres de la DLT à partir de données déjà utilisées dans les tableurs Excel (cf. § 5.4.1). Cependant pour s'assurer que la convergence de l'algorithme n'est pas biaisée, nous l'avons soumise à un second test. Un exemple de résolution des paramètres de la DLT est présenté dans KRAUS K. (1997). Nous avons introduit les données dans notre application et analysé les résultats.

Les résultats ont été très satisfaisants. Nous arrivons aux mêmes valeurs numériques après un nombre d'itérations identiques. Se référant à la renommée de la source dans le domaine de la photogrammétrie, nous avons validé notre algorithme.

5.6. Développement d'une application

5.6.1. Introduction

Nous avons maintenant identifié et testé une méthode d'inversion de matrice qui est applicable pour le calcul des paramètres de la DLT. La prochaine étape de notre développement est de construire, autour de cet algorithme, une application capable de gérer ce calcul, ainsi que la restitution à partir des paramètres calculés.

La structure de l'application se doit d'être assez simple. Il est en effet important, dans l'optique d'une intégration future dans un autre logiciel, de minimiser au maximum les lignes de code qui ne sont pas indispensables. Nous avons donc choisi d'écrire une application simple, exécutable dans la console (Annexe 7).

Le logiciel dans lequel pourrait être intégré ce module photogrammétrique est le logiciel créé dans l'Unité de Géomatique de l'Université de Liège, *DDPS* (cf. § 2.2.2.2). Le choix de notre langage de programmation a été contraint par celui qui est utilisé pour programmer *DDPS*, le C. De plus, en choisissant ce langage, nous pouvons bénéficier de la librairie de calcul matriciel développée pour ce logiciel. Nous avons exploité l'inversion de matrice de *Gauss Jordan* (cf. § 5.5.2.) ainsi que les fonctions standard de création et de gestion des matrices.

5.6.2. Contrôle de la qualité de l'ajustement

Dans l'application, nous avons développé deux fonctions qui permettent d'avoir un indice de la qualité de l'ajustement réalisé. La première utilise les lois de la statistique et l'erreur quadratique moyenne, tandis que la seconde se base sur les propriétés de la photogrammétrie.

5.6.2.1. Indicateur statistique

Nous proposons d'abord, et à chaque fois qu'un ajustement est réalisé, d'afficher la valeur des résidus en unité pixels sur tous les points de calage. Ils sont calculés automatiquement à partir des paramètres qui ont été résolus. En introduisant dans l'équation (6) les coordonnées image et terrain des points de calage ainsi que les paramètres calculés, on obtient une valeur pour les coordonnées de ces points en utilisant la gerbe reconstruite. L'écart entre cette valeur et celle qui a été digitalisée représente le résidu

selon un axe du référentiel image, ainsi que l'écart quadratique moyen total [équation (34)]. Dans la littérature, on retrouve souvent l'acronyme RMS pour *Root Mean Square* :

$$\sigma_{\text{total}} = \sqrt{\frac{\sum_{i=1}^{i \leq n} v_x^i + \sum_{i=1}^{i \leq n} v_y^i}{n-1}} \quad (34)$$

À partir de cet indicateur statistique, nous pouvons calculer l'erreur probable sur le pointé des points de calage dans l'image. L'expression de cette erreur en fonction de l'écart quadratique moyen total [équation (35)] est :

$$e_{\text{probable}} = 0,67 \sigma_{\text{total}} \quad (35)$$

5.6.2.2. Indicateur de la qualité de la reconstruction

Un second contrôle peut être effectué. Si nous suivons les travaux réalisés, notamment en photogrammétrie aérienne analytique, la comparaison de la position du centre optique calculée à partir des paramètres de la DLT et celle mesurée sur le terrain est un bon indicateur de la qualité de la reconstruction de la gerbe perspective par la DLT. Si la position du centre optique mesurée est connue, notre application permet d'utiliser ce paramètre de qualité.

Si nous repartons des équations de base de la DLT [équations (3') et (3'')] et de l'expression des paramètres L_1 à L_{11} d'un cliché (KWON 1998 ; KRAUS 1997), nous pouvons déduire les trois équations suivantes :

$$\begin{aligned} L_1 x_0 + L_2 y_0 + L_3 z_0 &= -L_4 \\ L_5 x_0 + L_6 y_0 + L_7 z_0 &= -L_8 \\ L_9 x_0 + L_{10} y_0 + L_{11} z_0 &= -1 \end{aligned} \quad (36)$$

Ce système de trois équations à trois inconnues se résout de manière stricte [équation (37)] et nous calculons facilement la valeur des coordonnées du point de vue (x_0, y_0, z_0) , à partir de la valeur des paramètres relatifs à l'image :

$$\begin{bmatrix} x_0 \\ y_0 \\ z_0 \end{bmatrix} = \begin{bmatrix} L_1 & L_2 & L_3 \\ L_5 & L_6 & L_7 \\ L_9 & L_{10} & L_{11} \end{bmatrix}^{-1} \cdot \begin{bmatrix} L_4 \\ L_8 \\ -1 \end{bmatrix} \quad (37)$$

5.6.3. Structure de l'application

Nous présentons ici la structure de notre programme. C'est ce cheminement que l'utilisateur suivra tout au long du processus de calcul des paramètres de la DLT (cf. § 5.6.3.1) et de la reconstruction tridimensionnelle des objets (cf. § 5.6.3.2). Chaque étape de ce cheminement est numérotée pour

5.6.3.1. Détermination des paramètres

1. Entrée du répertoire de travail.

Test :

- Si il ne se termine pas par « \ » le programme renvoie « répertoire non valide » et retour au point 1.

2. Tous les paramètres des clichés qui vont être utilisés sont-ils déjà calculés ?

Oui : le programme envoie au chargement des fichiers de paramètres « .Lparam ». Envoi de l'utilisateur au point 2 de la partie coordonnées tridimensionnelles.

Non : le programme envoie l'utilisateur au point 3.

Test :

- Comme pour toutes les questions où la réponse est oui ou non, le programme teste si la réponse est valide. Si ce n'est pas le cas, un message d'erreur apparaît et la question est reposée.

3. De combien de clichés l'utilisateur veut-il calculer les paramètres ?

4. Chargement des fichiers des paramètres qui ont été calculés auparavant. L'utilisateur entre le nom du fichier sans l'extension et le numéro du cliché (tous les clichés possèdent un identifiant numérique unique (cf. § 5.6.4.1)). Ces deux informations doivent être séparées par un espace.

Test :

- Si le fichier n'a pas pu être ouvert, retour au point 4.
 - Si le numéro du cliché diffère de celui spécifié par l'utilisateur, retour au point 4 et message d'erreur.
 - Si il n'y pas assez de points de calage « ctrl » (moins de 6), retour au point 4 et message d'erreur.
 - Si le nombre de points contrôle « check » est différent dans les fichiers chargés, retour au point 4 et message d'erreur.
 - Si un même numéro de cliché apparaît dans 2 fichiers différents, retour au point 4 et message d'erreur.
5. Calcul du nombre maximal de paramètres qui peut être réalisé en fonction du nombre de points « ctrl » identifiés dans le fichier de points de calage. Nous retenons le plus petit pour la restitution.
6. L'application procède au calcul des paramètres.
7. Fonction : `calcul_param ()`
- La boucle s'arrête lorsque la plus grande différence entre deux itérations successives de la valeur des paramètres, est inférieure à 10^{-5} .

Test :

- La fonction renvoie un message d'erreur si le calcul converge en une seule itération (se produit si il y a un retour à la ligne de trop à la fin du fichier).
- La fonction renvoie un message d'erreur si le nombre d'itérations est égal à 200 (le calcul ne converge pas).
- Dans les deux cas, retour au point 4 et message d'erreur.

8. L'application procède au calcul des paramètres.
9. Renvoi à l'écran des résidus sur chaque point de calage, de l'écart quadratique moyen et de l'erreur probable sur une mesure en unité image. Si l'utilisateur est satisfait par la valeur de ces indicateurs de précision, le processus continue, sinon retour au point 4.
10. Calcul de la position du point de vue à partir des paramètres calculés.
11. Ecriture des paramètres et de la position du point de vue dans un fichier « Lparam » et dans un tableau intermédiaire « Lmem » (ce tableau sera ensuite copié dans « Lfinal » avec les paramètres qui seront chargés directement à partir d'un fichier).
12. Calcul des résidus sur la position des points de contrôle « check ».
 - Avec la fonction `calcul_xyz()`, on calcule la position des points de contrôle à l'aide des paramètres calculés et des coordonnées image de ceux-ci dans les différents clichés.
 - On soustrait la position mesurée (fichier des points de calage) et la position calculée.
 - Ces valeurs sont stockées, avec l'identifiant du point, dans un tableau.

5.6.3.2. Restitution

1. L'utilisateur veut-il utiliser les paramètres qui viennent d'être calculés ?
 - Oui :

L'utilisateur veut-il utiliser un (ou plusieurs) fichier(s) d'autres paramètres?

 - Oui : le programme demande combien de clichés l'utilisateur veut charger. Les paramètres, les coordonnées du point principal et le numéro de tous les clichés sont stockés dans un tableau « Lfinal ».

Remarque : Si la position du point de vue ne figure pas dans le fichier de paramètres, l'utilisateur peut la calculer et l'inscrire dans le fichier.

- Non : seules les données qui viennent d'être calculées sont stockées dans le tableau « Lfinal ».
- Non :
L'utilisateur veut-il utiliser un (ou plusieurs) fichier(s) d'autres paramètres?
 - Oui : le programme demande combien de clichés l'utilisateur veut charger. Les données chargées sont stockées dans le tableau « Lfinal ».

Remarque :

Si la position du point de vue ne figure pas dans le fichier de paramètres, l'utilisateur peut la calculer et l'inscrire dans le fichier.

- Non : le programme informe l'utilisateur qu'il doit utiliser au moins deux clichés.

Test :

- Dans tous les cas, l'utilisateur doit au moins écrire les paramètres de deux clichés dans le tableau « Lfinal ». Si ce n'est pas le cas, un message d'erreur apparaît invitant l'utilisateur à charger le bon nombre de fichiers de paramètres et retour au point 1.

2. Si aucun paramètre n'a été calculé, on peut les charger ici.

- On charge les noms des fichiers de paramètres. Si la position du point de vue ne figure pas dans le fichier, l'utilisateur peut la calculer et l'inscrire dans le fichier.

Test :

- Au moins deux fichiers doivent être introduits. Sinon message d'erreur et retour au point 2.

3. Le programme indique le nombre de paramètres qui va permettre la restitution des points. Il s'agit du minimum de paramètres calculés ou chargés jusqu'ici.
4. On entre le nom du fichier qui contient les coordonnées image des points à restituer.

Test :

- Le fichier doit au moins contenir 1 point à restituer.
 - Les numéros des clichés doivent correspondre à ceux des paramètres qui ont été calculés ou chargés, mais pas nécessairement dans le même ordre. Le programme identifie à l'aide du numéro du cliché quels paramètres doivent être associés aux coordonnées.
5. Calcul des coordonnées tridimensionnelles des points à restituer.
 - Les coordonnées sont calculées par la fonction : `calcul_xyz ()`.
 - Le programme crée une table de correspondance entre le tableau « Lfinal » et le tableau contenant les points à calculer afin de s'assurer que les coordonnées des points mesurés sur un cliché sont mises en relation avec les paramètres relatifs à celui-ci.
 - L'appel de la fonction ne se fait qu'une fois. Dès qu'elle est lancée, la fonction calcule les coordonnées tridimensionnelles de tous les points à restituer.
 6. Ecriture des résultats dans les fichiers «.dlt ».
 - Un fichier de résultats contient les coordonnées terrain des points à restituer et la position des points de vue.
 - Un fichier de résidus. Il n'est créé que si la position d'au moins un point de vue est connue ou si au moins un point est déclaré comme « *check* » dans le fichier des points de calage (ceci implique que nous avons calculé les paramètres des clichés).

5.6.4. Fichiers types

L'application que nous avons programmée nécessite d'utiliser des fichiers d'entrée (*input*) écrit dans un format défini. Aucune erreur ne peut s'introduire dans les fichiers sous peine d'erreur grave sur le processus. Un certain nombre de vérifications sont effectués par l'application sur les *inputs*. Cependant, mais elles ne permettent pas de contrôle infaillible. Nous apporterons donc une attention particulière lors de la construction de ces fichiers.

Dans la suite, nous adopterons la symbolique suivante pour décrire les fichiers types :

- = tabulation ;
- = espace.

5.6.4.1. Points de calage

Il s'agit d'un fichier exclusivement *input*, il contient :

- *l'identifiant du cliché*. Cet identifiant est indispensable à l'exécution du programme. Il permet de faire le lien entre les points de calage, les paramètres et les points à reconstruire ;
- *la taille de l'image*. Elle permet de calculer la première approximation des coordonnées du point principal du cliché ;
- *la position mesurée de la camera*. Elle n'est pas indispensable à l'exécution de l'application. Si elle n'est pas connue, la ligne doit rester entièrement vide ;
- *la liste des points de calage et de contrôle*.

Structure du fichier :

```
Poi nt_de_contrô le_cl i che_ i d cl i ché
Tai lle_ i mageu0  vo
Posi ti on_cameraX0  Yo  Zo
check_ ' i d pt 1'   U1  V1  X1  Y1  Z1
ctrl _ ' i d pt 2'   U2  V2  X2  Y2  Z2
```

5.6.4.2. Paramètres

Ce fichier avec l'extension « *.Lparam* » est un *output* si les paramètres sont calculés. Si les paramètres sont déjà calculés que ce soit ou non avec notre application, le fichier devient un *input*. Dans les deux cas, ce fichier doit avoir la même forme :

- *l'identifiant du cliché*. Cet identifiant est indispensable à l'exécution du programme, il permet de faire le lien entre les points de calage, les paramètres et les points à reconstruire ;
- *les coordonnées du point principal*. Elles sont indispensables dans le processus de reconstruction tridimensionnelle d'objets ;
- *la position calculée de la camera*. Elle n'est pas indispensable à l'exécution de l'application, et est calculée à partir des paramètres. Si le fichier est un *input* et que la position de la camera n'y figure pas (ligne complètement vide), l'application peut calculer la position ;
- *la liste des paramètres*. Les paramètres y sont inscrits l'un après l'autre. Il y a 11, 14 ou 16 paramètres en fonction du nombre de points de calage introduits.

Structure du fichier :

Paramètres_cliché_id_cliché
Point_principal _xx. xxxx_xx. xxxx
Position_camera __xx. xxxxxx__xxxxxx. xxxx
' L1' __xx. xxxx
' L2' __xx. xxxx
' L3' __xx. xxxx
' L4' __xx. xxxx
' L5' __xx. xxxx
' L6' __xx. xxxx
' L7' __xx. xxxx
' L8' __xx. xxxx
' L9' __xx. xxxx
' L10' __xx. xxxx
' L11' __xx. xxxx

5.6.4.3. Restitution

Il s'agit du deuxième fichier exclusivement *input*. Il contient, pour chaque cliché dont les paramètres ont été calculés ou chargés depuis un fichier « .Lparam », l'ensemble des coordonnées image des points à calculer. Ce fichier doit impérativement contenir les informations suivantes :

- *l'identifiant du cliché*. Cet identifiant est indispensable à l'exécution du programme, il permet de faire le lien entre les points de calage, les paramètres et les points à reconstruire. L'ordre dans lequel sont inscrit les clichés n'a aucune importance;
- *la liste des coordonnées des points à restituer* ;
- *répétition des informations*. Les informations décrites dans les deux points précédents, sont reproduites autant de fois qu'il y a de clichés participants à la reconstruction.

Structure du fichier :

```
Poi nt_à_ cal cul er  
Cl i ché_ id cl i ché 1  
' id pt 1' __U11__v11  
' id pt 2' __U12v12  
Cl i ché_ id cl i ché 2  
' id pt 1' __U21__v21  
' id pt 2' __U22__v22
```

5.6.4.4. Résultats

Le fichier de résultats est un fichier strictement *output*. C'est pour cette raison que nous n'allons pas développer les informations qu'il doit contenir :

- *la liste des points restitués*. Chaque ligne contient l'identifiant du points et ses coordonnées (X, Y, Z).
- *la liste de la position des points de vue*. Chaque ligne contient l'identifiant du point de vue et ses coordonnées (X, Y, Z).

Structure du fichier :

```
Résul tats  
Resti tuti on_(X, Y, Z)  
ID_' id pt 1' __xx. xxxx__xx. xxxx__xx. xxxx  
ID_' id pt 2' __xx. xxxx__xx. xxxx__xx. xxxx  
  
Posi ti on_des_poi nts_de_vues_(X, Y, Z)  
Cl i ché_' id cl i ché 1' __xx. xxxx__xx. xxxx__xx. xxxx  
Cl i ché_' id cl i ché 2' __xx. xxxx__xx. xxxx__xx. xxxx
```


5.6.4.5. Résidus

Le fichier de résidus est, également, strictement *input*. Cependant, il n'est construit que lorsqu'une des deux conditions suivantes est remplie :

- soit il existe des points de contrôle communs dans les différents fichiers de points de calage (cf. § 5.6.4.1) ;
- soit la position calculée et mesurée des points de vue est connue pour au moins un cliché.

Si le fichier est créé, il contient :

- *la liste des résidus sur les points de contrôle*. Chaque ligne contient l'identifiant du point de contrôle et ses résidus (dX, dY, dZ).
- *la liste de résidus sur la position des points de vue*. Chaque ligne contient l'identifiant du point de vue et ses résidus (dX, dY, dZ).

Structure du fichier :

Résidus

Position_des_points_de_contrôle_(dX, dY, dZ)

ID_'id pt 1' _xx. xxxx_xx. xxxx_xx. xxxx

ID_'id pt 2' _xx. xxxx_xx. xxxx_xx. xxxx

Position_des_points_de_vues_(dX, dY, dZ)

Cliché_'id cliché 1' _xx. xxxx_xx. xxxx_xx. xxxx

Cliché_'id cliché 1' _xx. xxxx_xx. xxxx_xx. xxxx

5.6.5. Les principales fonctions

Dans cette partie du travail, nous présenterons les deux algorithmes principaux de l'application programmée (le code source complet est présenté en annexe (Annexe 7)). Il s'agit des fonctions qui permettent d'une part le calcul des paramètres des clichés, et d'autre part la restitution tridimensionnelle des objets :

- *Calcul_param ()* : pour une description théorique complète de la fonction, nous renvoyons au chapitre consacré à la théorie de la DLT (cf. § 4.3.1) ;

- *Calcul_xyz ()* : pour une description théorique complète de la fonction, nous renvoyons au chapitre consacré à la théorie de la DLT (cf. § 4.3.2).

Remarque : pour permettre une meilleure lisibilité, la couleur de la police des lignes de codes est affichée en grisé.

5.6.5.1. Fonction : *Calcul_param ()*

Les arguments de la fonction sont, dans l'ordre, la matrice *D* contenant les coordonnées image et terrain des points de calage, les coordonnées *u* et *v* maximales, le nombre de paramètres qui doit être calculés, et le nombre de points de calage qui va servir à l'ajustement.

```
matrix* calcul_param(matrix *D,long double uo,long double vo,int nbparam, int
nbpt){
int i,p,o,nb=0;
```

Le réel *difi* (critère d'arrêt) (cf. § 4.3.1.1) est initialisé à 10^{10} pour permettre au calcul itératif de démarrer.

```
long double difi=10E10,diff;
```

Lors de la première itération, les coordonnées du point principal sont assimilées à celles du centre géométrique du cliché et les paramètres sont tous initialisés à une valeur nulle.

```
uo=uo/2;
vo=vo/2;
for (i=0; i<nbparam; i++) ValAssign(L,i,0,0);
```

Début de la boucle, elle est stoppée lorsque la valeur du critère d'arrêt est inférieure à 10^{-5}

```
while (difi>1E-5){
difi=0;
```

On recopie dans *Linitial* les valeurs de *L* dans lequel vont être stockés les nouveaux paramètres (*Linitial* = *L_{i-1}* et *L* = *L_i*).

```
Linitial=CloneMat(L);
```

Incrémentation du compteur d'itérations.

```
nb++;
```

Remplissage des matrices. À chaque passage dans la boucle, deux lignes de la matrice des coefficients et de celle des termes indépendants sont écrites. Il y a autant de passages qu'il y a de points de calage.

```
for(i=0; i<nbpt; i++){
o=2*i;
p=2*i+1;

//vecetur des termes indépendant

//1ere*i ligne
ValAssign(I,o,0,ValRead(D,i,0));
//2eme*i ligne
ValAssign(I,p,0,ValRead(D,i,1));

//paramètres standards

//L1
ValAssign(C,o,0,ValRead(D,i,2));
ValAssign(C,p,0,0);

//L2
ValAssign(C,o,1,ValRead(D,i,3));
ValAssign(C,p,1,0);

//L3
ValAssign(C,o,2,ValRead(D,i,4));
ValAssign(C,p,2,0);

//L4
ValAssign(C,o,3,1);
ValAssign(C,p,3,0);

//L5
ValAssign(C,o,4,0);
ValAssign(C,p,4,ValRead(D,i,2));

//L6
ValAssign(C,o,5,0);
ValAssign(C,p,5,ValRead(D,i,3));

//L7
ValAssign(C,o,6,0);
ValAssign(C,p,6,ValRead(D,i,4));

//L8
ValAssign(C,o,7,0);
ValAssign(C,p,7,1);

//L9
ValAssign(C,o,8,-(ValRead(D,i,0)*ValRead(D,i,2)));
ValAssign(C,p,8,-(ValRead(D,i,1)*ValRead(D,i,2)));

//L10
ValAssign(C,o,9,-(ValRead(D,i,0)*ValRead(D,i,3)));
ValAssign(C,p,9,-(ValRead(D,i,1)*ValRead(D,i,3)));

//L11
ValAssign(C,o,10,-(ValRead(D,i,0)*ValRead(D,i,4)));
ValAssign(C,p,10,-(ValRead(D,i,1)*ValRead(D,i,4)));
```

Si le nombre de paramètres à calculer est supérieur à onze, il faut ajouter trois lignes à la matrice des coefficients, elles correspondent aux distortions radiales. Pour ce faire, nous devons calculer les termes R, $\text{zeta}=\text{u}-\text{u}_0$, $\text{nu}=\text{v}-\text{v}_0$, $\text{r2}=\text{zeta}^2+\text{nu}^2$.

```
//distortions radiales
if(nbparam>11){
//vecteur R
ValAssign(R,i,0,(ValRead(L,8,0)*ValRead(D,i,2)+ValRead(L,9,0)*ValRead(D,i,3) &lt;
+ValRead(L,10,0)*ValRead(D,i,4)+1));
//vecteur zeta
ValAssign(zeta,i,0,ValRead(D,i,0)-u0);
//vecteur nu
ValAssign(nu,i,0,ValRead(D,i,1)-v0);
//vecteur r2
ValAssign(r2,i,0,pow(ValRead(zeta,i,0),2)+pow(ValRead(nu,i,0),2));

//L12
ValAssign(C,o,11,(ValRead(zeta,i,0)*ValRead(r2,i,0)*ValRead(R,i,0)));
ValAssign(C,p,11,(ValRead(nu,i,0)*ValRead(r2,i,0)*ValRead(R,i,0)));

//L13
ValAssign(C,o,12,(ValRead(zeta,i,0)*pow(ValRead(r2,i,0),2)*ValRead(R,i,0)));
ValAssign(C,p,12,(ValRead(nu,i,0)*pow(ValRead(r2,i,0),2)*ValRead(R,i,0)));

//L14
ValAssign(C,o,13,(ValRead(zeta,i,0)*pow(ValRead(r2,i,0),3)*ValRead(R,i,0)));
ValAssign(C,p,13,(ValRead(nu,i,0)*pow(ValRead(r2,i,0),3)*ValRead(R,i,0)));
```

Si le nombre de paramètres à calculer est supérieur à 14, il faut encore ajouter deux lignes à la matrice des coefficients, elles correspondent au décentrement du point principal de la lentille.

```
//decentrement lentille
if(nbparam>14){
//L15
ValAssign(C,o,14,(ValRead(r2,i,0)+2*pow(ValRead(zeta,i,0),2))*ValRead(R,i,0));
ValAssign(C,p,14,(ValRead(zeta,i,0)*ValRead(nu,i,0)*ValRead(R,i,0)));

//L16
ValAssign(C,o,15,(ValRead(nu,i,0)*ValRead(zeta,i,0)*ValRead(R,i,0)));
ValAssign(C,p,15,(ValRead(r2,i,0)+2*pow(ValRead(nu,i,0),2))*ValRead(R,i,0));
}
}
}
```

Détermination des paramètres par la fonction *LeastSquareQR* qui détermine, par moindres carrés, le vecteur des inconnues à partir de la décomposition QR (cf. § 5.5.4).

```
L= LeastSquareQR(C,I);
```

À partir des nouveaux paramètres du tableaux L qui sont calculés, nous pouvons déterminer les coordonnées approchées du point principal.

```
//calcul du point principal

uo=((ValRead(L,0,0)*ValRead(L,8,0))+(ValRead(L,1,0)*ValRead(L,9,0))+
ValRead(L,2,0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+
pow(ValRead(L,9,0),2)+pow(ValRead(L,10,0),2));

vo=((ValRead(L,4,0)*ValRead(L,8,0))+(ValRead(L,5,0)*ValRead(L,9,0))+
ValRead(L,6,0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+
pow(ValRead(L,9,0),2)+pow(ValRead(L,10,0),2));
```

La valeur du critère d'arrêt *difi* est égale à la plus grande différence, en valeur absolue, entre les éléments *Linitial* et *L* respectivement.

```
//critière d'arrêt

difL=SubMat(Linitial,L);
for(i=0; i<nbparam; i++){
diff=fabs(ValRead(difL,i,0));
if(diff>difi) difi=diff;
}
```

La boucle est stoppée si le nombre d'itération dépasse 200 ou bien si le calcul converge en une seule itération (ce qui signifie que tous les paramètres sont nuls). Si la boucle est arrêtée, l'application renvoie un message d'erreur à l'écran et permet à l'utilisateur de charger un autre fichier.

```
if (nb==200) break;
}
if(nb==1 || nb==200){
for(i=0; i<nbparam; i++)ValAssign(L,i,0,0);
printf("\n>>LE CALCUL N'A PAS CONVERGER...veuillez verifier le fichier<<\n");
}
```

La fonction retourne la valeur finale du vecteur des inconnues.

```
return(L);
}
```

5.6.5.2. Fonction : Calcul_xyz()

Les arguments de la fonction sont, dans l'ordre, la matrice *Lfinal*, contenant l'ensemble des paramètres qui ont été calculés ou chargés; la matrice *CAL*, elle contient les coordonnées image de l'objet à restituer, le nombre de points à calculer, le nombre de clichés utilisés pour la restitution, le nombre de paramètres utiles à la restitution.

```
matrix* calcul_XYZ(matrix *Lfinal, matrix *CAL, int cptc, int ntotal, int
minparam){

int nb;
int i,j,k,o,p,q,r;
int num, colonne;
long double difi,diff,valX;
```

```
//fonction
matrix* least_square(matrix *C, matrix*I);
```

Création d'un table (*TABLE*) de correspondance sur base du numéro de cliché entre la matrice des paramètres (*Lfinal*) et qui contient les points à calculer (*CAL*).

```
//recherche des paramètres correspondants au point a restitué
//(les paramètres ne sont pas entrés dans le meme ordre)
//on crée une table de correspondance

for(i=0; i<ntotal; i++){
num=int (ValRead(CAL,0,i)); //on obtient le numero du cliche de la 1ere colonne
dans CAL
for(j=0; j<ntotal; j++){
if (ValRead(Lfinal,0,j)==num){
ValAssign (TABLE,i,0,num); //numero du cliche
ValAssign (TABLE,i,1,i); //numero colonne dans CAL
ValAssign (TABLE,i,2,j); //numero colonne dans Lfinal
}
}
}

for(i=0; i<ntotal; i++){ //pour tous les clichés
for(j=0; j<cptc; j++){ //pour tous les points
o=2*j;
p=2*j+1;
colonne= int (ValRead(TABLE,i,2)); //identification de la colonne des

//matrice zeta u-u0
ValAssign(zeta,j,i,(ValRead(CAL,o+1,i)-ValRead(Lfinal,1,colonne)));
//matrice nu v-v0
ValAssign(nu,j,i,(ValRead(CAL,p+1,i)-ValRead(Lfinal,2,colonne)));
//matrice r2
ValAssign(r2,j,i,(pow(ValRead(zeta,j,i),2)+pow(ValRead(nu,j,i),2)));
```

L'application détermine les erreurs optiques ($\Delta u, \Delta v$) pour chaque point à calculer. Le nombre de paramètres utilisés dépend du nombre minimum de paramètres. Si les fichiers de paramètres contient les 16 paramètres.

```
//matrice erreur de la lentille
if (minparam==16){
ValAssign(deltaUV,o,i,((ValRead(zeta,j,i)*((ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i))+ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i),2))+ValRead(Lfinal,16,colonne)*pow(ValRead
(r2,j,i),3)))+(ValRead(Lfinal,17,colonne)*(ValRead (r2,j,i)+(2*pow(ValRead
(zeta,j,i),2)))+(ValRead(Lfinal,18,colonne)*ValRead (zeta,j,i)*ValRead
(nu,j,i)))));
ValAssign(deltaUV,p,i,((ValRead(nu,j,i)*((ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i))+ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i),2))+ValRead(Lfinal,16,colonne)*pow(ValRead
(r2,j,i),3)))+(ValRead(Lfinal,18,colonne)*(ValRead (r2,j,i)+(2*pow(ValRead
(nu,j,i),2)))+(ValRead(Lfinal,17,colonne)*ValRead (zeta,j,i)*ValRead
(nu,j,i)))));
}
}
```

Si les fichiers contiennent 14 paramètres, le calcul des erreurs optiques est le suivant.

```
if(minparam==14){
```

```

ValAssign(deltaUV,o,i,(ValRead(zeta,j,i)*ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i)+(ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i),2)))+(ValRead(Lfinal,16,colonne)*pow(ValRead (r2,j,i),3)));
ValAssign(deltaUV,p,i,(ValRead(nu,j,i)*ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i)+(ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i),2)))+(ValRead(Lfinal,16,colonne)*pow(ValRead (r2,j,i),3)));
}

```

Sinon, la valeur des corrections ($\Delta u, \Delta v$) dues aux erreurs optiques est nulle.

```

else{
ValAssign(deltaUV,o,i,0);
ValAssign(deltaUV,p,i,0);
}
}
}

```

Application des correction calculées aux coordonnées image.

```

for(i=0; i<ntotal; i++){//pour tous les clichés
for(j=0; j<2*cptc; j++){//pour tous les points (2X pour u et v)
//matrice upsilon omega
ValAssign(Upsilon,j,i,ValRead(CAL,j+1,i)-ValRead(deltaUV,j,i));
}
}

//*****
//*Résolution*
//*****

```

Résolution itérative des coordonnées tridimensionnelles des points contenus dans la matrice (*CAL*). Pour chaque point, tout le processus itératif doit être relancé. Début de la boucle 1.

```

for(i=0; i<cptc; i++){//pour tout les points
q=2*i;
r=2*i+1;
nb=0;

```

Le réel *difi* (critère d'arrêt) (cf. § 4.3.1.1) est initialisé à 10^{10} pour permettre au calcul itératif de démarrer.

```
difi=10E10;
```

La valeur initiale des inconnues est égale à 1000 (matrice *X*).

```
for(k=0; k<3; k++) ValAssign(X,k,0,1000);
```

Début de la boucle 2, elle s'arrête lorsque la valeur du critère d'arrêt est inférieure à 10^{-5} .

```
while (difi>1E-5){
difi=0;
```

La matrice *X* est copiée dans *Xinitial*.

```
Xinitial=CloneMat(X);
nb++;
```

Remplissage des matrices intervenant dans le calcul.

```
for(j=0; j<ntotal; j++){//remplissage des elements d'un colonne pour le ntotal
clichés
o=2*j;
p=2*j+1;
colonne= int (ValRead(TABLE,j,2));//numero col(Lfinal) du cliché A(col i(CAL et
Upsilon)

//matrice des R: 1 par point et par camera

ValAssign(R,i,j,((ValRead(Lfinal,11,colonne)*ValRead(X,0,0))+(ValRead(Lfinal,12,c
olonne)*ValRead(X,1,0))+(ValRead(Lfinal,13,colonne)*ValRead(X,2,0))+1));

//terme independants
ValAssign(I,o,0,(ValRead(Lfinal,6,colonne)-ValRead(Upsilon,q,j))/ValRead(R,i,j));
ValAssign(I,p,0,(ValRead(Lfinal,10,colonne)-
ValRead(Upsilon,r,j))/ValRead(R,i,j));

//coefficients
for(k=0; k<3; k++){//remplissage des trois elements d'une ligne des coefficients
ValAssign(C,o,k,(((ValRead(Lfinal,11+k,colonne)*ValRead(Upsilon,q,j))-
ValRead(Lfinal,3+k,colonne))/ValRead(R,i,j)));
ValAssign(C,p,k,(((ValRead(Lfinal,11+k,colonne)*ValRead(Upsilon,r,j))-
ValRead(Lfinal,7+k,colonne))/ValRead(R,i,j)));
}
}
```

Résolution par moindres carrés par la méthode de la décomposition *QR*.

```
X= LeastSquareQR (C,I);
```

La valeur du critère d'arrêt *difi* est égale à la plus grande différence, en valeur absolue, entre les éléments *Xinitial* et *X* respectivement.

```
difX=SubMat(Xinitial,X);
for(k=0; k<3; k++){
diff=fabs(ValRead(difX,k,0));
if(diff>difi) difi=diff;
}
```

Fin de la boucle 1.

```
}
```

Écriture de la valeur des inconnues dans la matrice *RESULT*.

```
for(k=0; k<3; k++){
valX=ValRead(X,k,0);
ValAssign(RESULT,k,i,valX);
}
```

Fin de la boucle 2.

```
}
return(RESULT);
}
```


6. Terrain et application

6.1. Introduction

Ce chapitre peut être divisé en trois parties. La première, s'attache à décrire l'ensemble du travail qui a été réalisé sur le terrain et les traitements préliminaires à la mise en œuvre d'un calcul de DLT (cf. § 6.2 et § 6.3). La seconde, expose les différents moyens que nous avons utilisés pour déterminer la précision de la DLT en fonction de l'utilisation de différents appareils de prise de vue (cf. § 6.4 et § 6.5). Et enfin, la dernière décrit un exemple de restitution complète d'un édifice (cf. § 6.6).

En guise d'introduction, nous proposons quelques conditions et règles qu'il semble indispensable à respecter afin d'optimiser l'utilisation d'appareils photographiques conventionnels dans le cadre de campagnes de levés photogrammétriques. Ces règles appelées aussi *3x3 rules* ont été testées et publiées par WALDHAUESL & OGLEBY (1994 in HANKE & GRUSSENMYER, 2002). Nous les proposons ici dans leur version originale :

- 3 règles géométriques:

- *preparation of control information,*
- *multiple photographic all-around coverage,*
- *taking stereopartners for stereo-restitution.*

- 3 règles photographiques:

- *the inner geometry of the camera has to be kept constant,*
- *select homogenous illumination,*
- *select most stable and largest format camera available.*

- 3 règles organisationnelles :

- *make proper sketches,*
- *write proper protocols,*
- *don't forget the final check.*

6.2. Travail de terrain

6.2.1. Visite préliminaire

Le quartier dans lequel doivent se dérouler les tests de la DLT avait été défini par le laboratoire du *LEMA*. Comme nous ne connaissions pas la configuration des lieux, une visite préliminaire nous a paru indispensable. Nous nous sommes rendus une première fois dans les anciens quartiers de Verviers (Figure 1). Nous avons pu y identifier les potentialités et les contraintes du site.

6.2.1.1. Choix des sites de test

Suite à cette première visite, nous avons identifié deux sites qui présentent un intérêt certain pour tester la DLT en milieu urbain. Le premier est situé place du marché. Il s'agit de l'hôtel de ville de Verviers très caractéristique de style Louis XVI de la fin du XVIII^e siècle, inscrit au Patrimoine exceptionnel de Wallonie. Nous pouvons remarquer sous le fronton courbe l'inscription « Publicité, Sauvegarde du Peuple », allusion à l'ouverture aux citoyens des débats et séances du Conseil communal.

Ce bâtiment devait, à l'origine, servir de site de test mais suite aux problèmes rencontrés (cf. § 6.2.3), nous avons décidé d'utiliser les clichés de ce bâtiment réalisés lors de la visite préliminaire pour restituer l'ensemble du bâtiment (cf. § 6.6).



Figure 19. Hôtel de ville de Verviers
Façade coté opposé à la place du marché

Le second étant l'ancien commissariat de police, il constituera notre second choix pour tester la DLT en utilisant plusieurs types d'appareils photographiques. Le test sera réalisé sur la façade présentée ci-dessous (Figure 20).



Figure 20. Ancien commissariat de police de la ville de Verviers
Façade utilisée pour le test de la DLT

6.2.1.2. Identification des points sur les façades

Comme nous l'avons déjà dit plus haut (cf. § 4.3.1), l'application de la DLT nécessite un minimum de 6 points de calage par cliché si nous ne prenons pas en compte les erreurs optiques, et 8 dans le cas contraire. Dans le cas d'un levé d'une certaine ampleur, ce nombre peut être relativement élevé. Cela implique d'identifier préalablement l'ensemble des points utiles. Notons qu'il faut encore ajouter les points de contrôle à la liste des points à lever (Annexe 14).

Pour rappel :

- *points de calage* : points servant à l'ajustement, ils interviennent dans les équations de résolution des paramètres de la DLT [équation (8)] ;
- *points de contrôle* : points servant au contrôle de l'ajustement mais n'intervenant pas dans celui-ci. On peut comparer la qualité de l'ajustement réalisé à partir des points de calage en effectuant la différence entre les coordonnées mesurées et calculées des points de contrôle.

Nous avons photographié, sous tous les angles, ces bâtiments de manière à ce que leur ensemble soit visible (Annexe 14) avec un Canon IXUS 750 (Tableau 8). À partir de ces clichés, nous avons identifié et nommé, de manière univoque, l'entièreté des points qui nous seront utiles par la suite. Des fichiers en format A3 identifiant exactement les points sont créés. Ils nous seront très utiles lors des différentes campagnes de mesures prévues plus ultérieurement.

6.2.2. Première campagne de mesures

Durant cette première journée de travail sur le terrain, nous avons implanté et mesuré la polygonale autour de l'hôtel de ville. Nous avons aussi mesuré l'ensemble des points de calage et de contrôle qui nous seront utiles sur les façades du bâtiment. Les coordonnées de ces points sont reprises dans un fichier texte sous format numérique (Annexe 14).

Pour être rigoureux, nous devons admettre qu'une petite erreur a été commise lors du cheminement de précision sur les points de la polygonale. Nous n'avons pas enregistré dans la mémoire de l'instrument la dernière mesure, c'est-à-dire la visée de la dernière station (station d'ouverture) vers la première. De ce fait, nous n'avons pas d'autre contrôle sur la précision de la polygonale que l'écart entre la position de la dernière station lors de l'ouverture et après le cheminement. Cet écart est de l'ordre du millimètre.

6.2.3. Deuxième campagne de mesures

Nous avons divisé cette campagne en deux visites distinctes sur le terrain. La première, a été mise à profit pour implanter les stations de autour de l'ancien commissariat. Nous avons mesuré précisément la polygonale et la position des points de vue des appareils photographiques testés (cf. § 6.2.3.1). En effet, il nous paraissait intéressant de mesurer la position des points de vue pour donner un indice de la qualité de la reconstruction de la gerbe perspective par la DLT.

Nous avons présenté en annexe (Annexe 9) un extrait du calcul de la compensation de la polygonale, les ellipses d'erreurs sur chaque des stations y figure aussi. Les coordonnées de points figure aussi en format numérique (Annexe 14).

Lors de cette visite, nous avons eu la mauvaise surprise de découvrir que l'hôtel de ville était couvert sur l'entièreté de ses façades par un filet de protection. Des morceaux de briques s'étaient détachés et les travaux de rénovation étaient déjà prévus. Nous avons décidé d'annuler l'entièreté des tests prévus et de les reporter sur l'autre édifice de notre

étude, l'ancien commissariat de police, tout en sachant que les conditions ne sont pas idéales, à savoir :



- le manque de recul ;
- l'exiguïté des rues ;
- la mauvaise orientation par rapport au soleil.

6.2.3.1. *Appareils utilisés*

Le test de la méthode de la DLT que nous allons mettre en œuvre, permettra de comparer l'influence de l'utilisation de différents types d'appareils de prise de vue sur la précision de l'ajustement des paramètres de la DLT. C'est un total de trois appareils photographiques qui sont utilisés. Les caractéristiques de chacun d'entre eux sont reprises dans le tableau ci-dessous (Tableau 3). Une petite précision doit cependant être apportée aux informations que l'on y retrouve. Le format de compression NEF (Nikon Electronic Format) est un format propriétaire de Nikon. Il est dérivé du format standard de stockage de photographie sans compression, le RAW.

Tableau 3. Caractéristiques techniques des appareils de prises de vue.

Remarque : informations complémentaires sur les données Exif (cf. § 3.4.4)

Type	Illustration	Référence boîtier	Référence objectif	Taille du capteur (pixels)	Stockage	Focale approchée (millimètres)
Compact numérique		Sony DSC W12	Carl Zeiss 2,8-5,2/7,9-23,7	2592 x 1944	JPEG (Memory Stick, 128 MB)	8 (données Exif)
Reflex numérique		Nikon D70	Nikon 18-70 mm 3,5-4,5	3008 x 2000	NEF (Sandisk ultra 2, 1GB)	35 (données Exif)
Reflex numérique (grand angle)		Nikon D70	Nikon 18-70 mm 3,5-4,5	3008 x 2000	NEF (Sandisk ultra 2, 1GB)	18 (données Exif)

6.2.3.2. Mesure de la position des points de vue

L'élaboration de la procédure permettant de mesurer la position de vue, nous a demandé quelques recherches. En effet, nous utilisons des appareils photographiques standard, il n'est pas possible de les fixer sur une embase classique ou encore de les mettre en station au droit d'un point connu.

Nos recherches nous ont menés vers une méthode de mesure simple et applicable aux appareils utilisés dans notre travail (CLARKE *et al.* 1998). Dans cette référence, la position des points de vue est mesurée directement avec un système laser. Comme nous disposons de cette technologie sur notre appareil, nous avons pu appliquer cette technique à la mesure de nos points de vue.

De manière pratique, nous avons positionné la station totale laser (Leica TC1103 plus) au pied de la façade, ce qui nous permet de viser directement le centre du système de lentille de l'appareil quelque soit sa position. La configuration est présentée ci-dessous (Figure 21) :

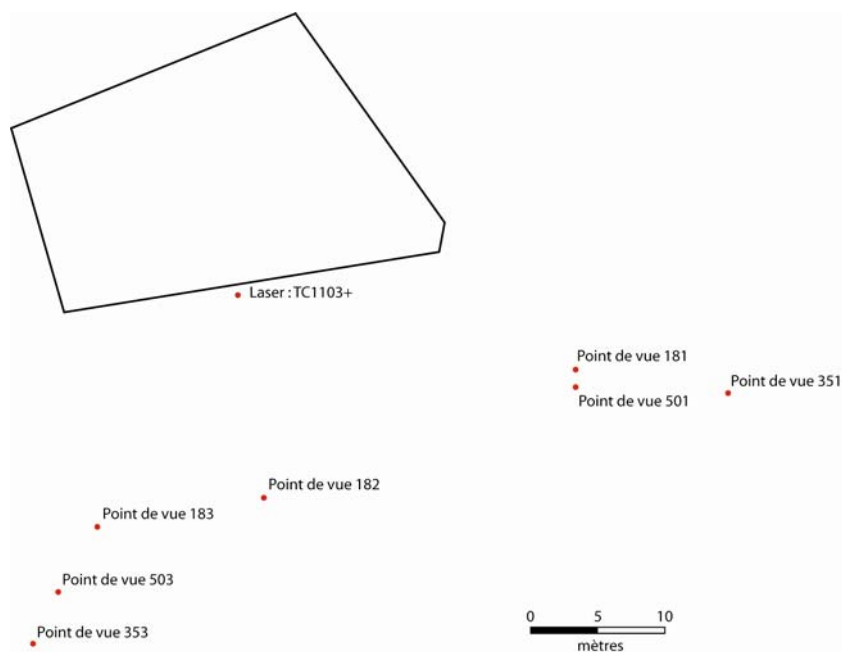


Figure 21. Position des points de vue pour le test de la DLT
Ancien commissariat de police de la ville de Verviers

Le code affecté aux points de vue et aux images (Annexe 8) associées est construit de la manière suivante :

- Les deux premiers chiffres identifient l'appareil qui est utilisé (18=Nikon D70 18mm, 35=Nikon D70 35mm, 50=Sony DSC W12) ;
- Le dernier représente la position du point de vue par rapport à la façade (1=droite, 2=milieu, 3=gauche).

6.2.4. Troisième campagne de mesures

Cette nouvelle journée de mesures a dû être ajoutée au programme initialement prévu. Ceci est la conséquence de la pose de filets de protection sur les façades de l'hôtel de ville et de notre décision d'effectuer les tests complets de la méthode de la DLT sur un autre bâtiment : l'ancien commissariat de police.

Elle a été menée le vendredi 4 août et a consisté à mesurer les coordonnées des points de calage nécessaires à l'orientation des clichés ainsi que les points de contrôle qui nous serviront à vérifier la qualité de l'ajustement.

Nous nous sommes mis en station en utilisant le principe de la station libre (cf. § 5.3.2). Les coordonnées des points levés lors de cette campagne de mesure figure en annexe (Annexe 14).

6.3. Traitement des images

Chacun des clichés utilisés dans notre travail a subi la même séquence de traitements avant de pouvoir être exploités. Cette procédure va nous permettre d'augmenter sensiblement la précision de la digitalisation des points, lors de l'orientation des images et la reconstruction tridimensionnelle des objets.

6.3.1. Conversion

Lors de notre première mise en œuvre de la DLT sur la façade du bâtiment d'astrophysique de l'Université de Liège (B5c), nous avons déjà dû appliquer un certain nombre de traitements aux images, pour qu'elles puissent être utilisées dans le logiciel *DDPS* (cf. § 2.2.2.2). Nous avons, cependant, rencontré quelques problèmes avec *DDPS*. En effet, de nombreuses fermetures inopinées du logiciel ne nous ont pas permis de travailler sereinement. Ces problèmes sont certainement dus à la conversion des images vers le format fichier lisible par ce logiciel (BMP bits/pixel). Comme le travail à réaliser est de plus grande ampleur, nous ne pouvons pas nous permettre ce genre de contretemps. Nous avons donc choisi de digitaliser les coordonnées des points de calage avec le logiciel *IDRIS Kilimanjaro*. Ce dernier dispose d'un autre système de coordonnées image, mais cela n'a pas d'influence sur l'utilisation d'une méthode photogrammétrique telle que la DLT (cf. § 4.2.2).

Avant d'être importées dans *IDRISI*, les images doivent subir une suite de traitements spécifiques :

1. ouverture du cliché dans *Adobe Photoshop CS*. S'il s'agit de cliché enregistré au format RAW par l'appareil (cas du Nikon D70), l'image doit être importée par l'intermédiaire du *plug in* Camera Raw ;
2. conversion des images en niveaux de gris (Menu : *Image* → *Mode* → *Niveaux de gris*);
3. amélioration du contraste de l'image en réalisant un étirement de l'histogramme (Menu : *Image* → *Réglages* → *Niveaux*). Il s'agit d'une fonction linéaire

permettant de reclasser les 256 niveaux de gris en sortie. Ce traitement est indispensable pour permettre une bonne exploitation de certains clichés (en particulier pour ceux de l'ancien commissariat de police) pour lesquels les conditions d'éclairage lors des prises de vues n'étaient pas optimales;

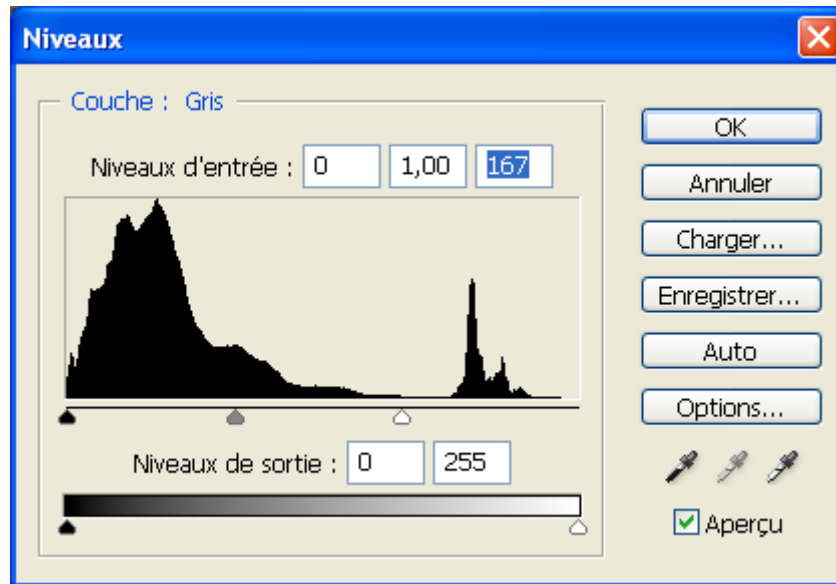


Figure 22. Amélioration interactive du contraste

Adobe Photoshop CS

4. enregistrement de l'image au format Bitmap. Bien qu'il existe un module d'importation d'image au format Jpeg dans *IDRISI*, celui-ci ne peut pas fonctionner correctement ;
5. importation de l'image avec le module de conversion d'image d'*IDRISI* (Menu : *File* → *Import* → *Desktop Publishing Format* → *BMPIDRIS*). Le logiciel crée deux fichiers : un fichier *.rdc contenant les métadonnées de l'image et un autre *.rst qui contient l'image enregistrée au format *IDRISI*.

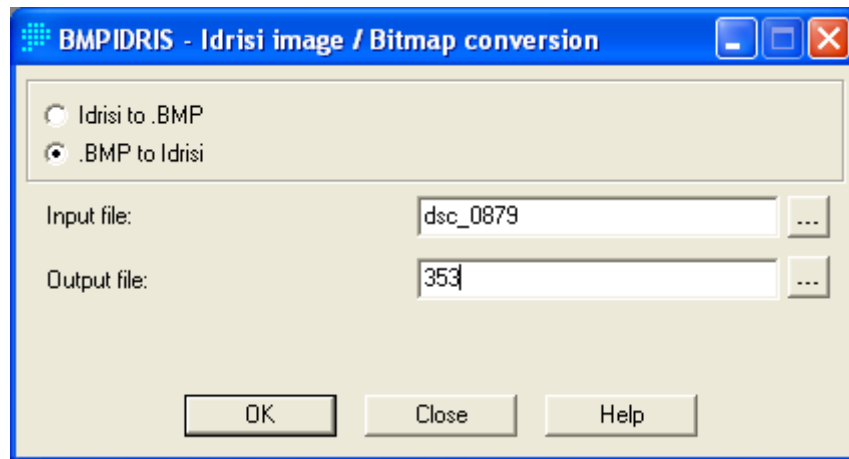


Figure 23. Importation des images Bitmap dans *IDRISI*
IDRISI Kilimanjaro

6.3.2. Digitalisation des points de calage

Les images peuvent, maintenant, être exploitées. Nous utilisons un petit outil pour visualiser les coordonnées des points de l'image (*Windows list* → *FeatureProperties*). Nous tenons compte des valeurs *x coord* et *y coord* permettant d'atteindre une précision inférieure au pixel. Nous pouvons indiquer que l'origine de ce système de coordonnées se situe en bas à gauche de l'image.

6.4. Comparatif des précisions obtenues

Dans cette partie du travail, nous allons étudier la précision des différentes méthodes et techniques énoncées plus haut. La comparaison est réalisée à l'aide d'une analyse de la variance à un critère de classification (*ANOVA 1*). Les différentes techniques qui sont comparées sont les quatre applications de la DLT.

Nous avons donc choisi d'utiliser un ensemble de 9 points, bien répartis sur la façade. La valeur absolue de l'écart entre les mesures effectuées sur les modèles restitués et la vérité terrain, est considérée comme un bon critère de précision, elle servira de base pour l'analyse de la variance. Rappelons que nous avons assimilé les mesures laser à la vérité terrain (cf. § 3.4).

À la suite de cette analyse, nous pourrions conclure s'il existe une différence significative entre les méthodes et ainsi établir une classification sur base de la précision obtenue avec les mesures de distances.

En résumé, l'analyse de la variance va nous permettre de comparer quatre applications distinctes de la DLT :

1. trois clichés (n° 181, 182, 183), appareil Nikon D70 avec un objectif 18 mm (code : *DLT-3 18 mm*);
2. un couple stéréoscopique (n° 181, 183), appareil Nikon D70 avec un objectif 18 mm (code : *DLT 18 mm*);
3. un couple stéréoscopique (n° 351, 353), appareil Nikon D70 avec un objectif 35 mm (code : *DLT 35 mm*);
4. un couple stéréoscopique (n° 501, 503), avec un appareil compact numérique Sony DSC W12 (code : *DLT Sony*).

6.4.1. Les données

Comme nous l'avons expliqué plus haut, nous n'avons pas pu effectuer le test de la DLT sur le site idéal (cf. § 6.2.3). Ceci à occasionner quelques difficultés pour faire converger le processus itératif lors de la détermination des paramètres des différents clichés.

Nous avons utilisé notre application pour déterminer les paramètres et résidus sur les points *check* nous ont permis de calculer la distance entre la vérité terrain et ses points. Nous présentons ci-contre un exemple des captures d'écrans des fichiers utiles pour réaliser ce travail. Ces fichiers sont relatifs à la détermination de la précision dans le cas de l'utilisation d'un couple stéréoscopique, l'appareil utilisé est le Canon D70 avec une focale approchée de 18 mm (données Exif).

Le fichier des points de calage « 181.txt » (cf. § 5.6.4.1) associés au cliché 181 qui sert à l'orientation de cette image et la détermination des résidus sur les points de contrôle *check* (Figure 24) :

```

Point de contrôle cliché 181
Taille image      3008 2000
Position camera   1013.645 963.705 98.368
ctrl '28'        582.525 822.645 977.1108 968.1883 109.4803
ctrl '67'        587.834 174.458 978.7300 968.9980 99.7186
ctrl '47'        631.603 572.666 979.7623 968.9271 105.1254
ctrl '25'        720.647 969.167 979.4111 969.6911 111.3805
ctrl '5'         831.741 1212.401 982.5204 970.3728 114.2647
ctrl '21'        965.861 1033.024 987.9257 971.0400 109.4663
ctrl '71'        816.113 208.529 987.3331 970.0513 99.8069
ctrl '35'        1049.238 1098.402 991.3575 970.9373 108.9162
ctrl '17'        1375.009 1478.981 996.1725 972.4813 111.3845
ctrl '54'        1414.897 950.629 997.9147 971.9434 105.1426
ctrl '56'        1519.316 504.927 998.7691 972.3256 101.4032
ctrl '51'        956.712 728.818 989.8436 970.6120 105.1423
check '26'       685.867 942.176 977.9280 969.4580 111.3910
check '97'       1017.638 1379.758 989.2044 971.4927 114.2673
check '16'       1496.339 1571.471 998.0512 972.7847 111.3903
check '33'       967.945 955.756 987.9903 971.1205 108.4587
check '94'       666.685 589.429 981.1210 969.1643 105.1354
check '50'       877.143 690.792 987.8658 970.2796 105.1441
check '53'       1318.297 903.909 996.5699 971.7240 105.1419
check '77'       1336.857 338.451 996.8432 971.7664 100.3748
check '74'       945.441 222.126 990.4273 970.5971 99.8149
check '65'       596.932 295.692 978.6353 968.9770 101.3851
    
```

Figure 24. Fichier des points de calage du cliché 181

Le fichier des 16 paramètres « 181.Lparam » (cf. § 5.6.4.2) associés au cliché 183. (Figure 25):

```

Paramètres clichés 181
Point principal 1140.398863 1154.003866
Position camera 1015.146165 963.329876 97.857084
'L1' -1.131016e-001
'L2' 5.243762e+000
'L3' 1.082316e+000
'L4' -5.042571e+003
'L5' -8.016916e-003
'L6' -1.660183e-001
'L7' 5.257836e+000
'L8' -3.464477e+002
'L9' -1.620246e-003
'L10' 5.867636e-004
'L11' 8.128123e-004
'L12' -1.828920e-008
'L13' -3.422097e-014
'L14' 1.843484e-020
'L15' 3.177058e-006
'L16' -3.259369e-006
    
```

Figure 25. Fichier des paramètres du cliché 181

Le fichier des points de calage « 183.txt » (cf. § 5.6.4.1) associé au cliché 183 qui sert à l'orientation de cette image et la détermination des résidus sur les points de contrôle *check* (Figure 26) :

```

Point de contrôle cliché 183
Taille image      3008 2000
Position camera   977.283   951.532   98.096
ctrl '28'        421.636   1786.004   977.1108   968.1883   109.4803
ctrl '67'        562.708   196.244    978.7300   968.9980   99.7186
ctrl '47'        769.773   1041.188   979.7623   968.9271   105.1254
ctrl '25'        725.33    1895.625   979.4111   969.6911   111.3805
ctrl '21'        1713.35   1389.337   987.9257   971.0400   109.4663
ctrl '71'        1719.132   164.046    987.3331   970.0513   99.8069
ctrl '35'        2086.946   1269.331   991.3575   970.9373   108.9162
ctrl '17'        2427.29   1410.527   996.1725   972.4813   111.3845
ctrl '54'        2659.744   731.064    997.9147   971.9434   105.1426
ctrl '56'        2735.672   296.643    998.7691   972.3256   101.4032
ctrl '51'        1959.997   844.743    989.8436   970.6120   105.1423
check '26'       520.814   1956.667   977.9280   969.4580   111.3910
check '97'       1720.137   1925.274   989.2044   971.4927   114.2673
check '16'       2569.311   1370.644   998.0512   972.7847   111.3903
check '33'       1716.095   1265.179   987.9903   971.1205   108.4587
check '94'       958.323    1009.564   981.1210   969.1643   105.1354
check '50'       1760.046   877.873    987.8658   970.2796   105.1441
check '53'       2554.959   747.947    996.5699   971.7240   105.1419
check '77'       2611.776   188.411    996.8432   971.7664   100.3748
check '74'       2045.844   149.982    990.4273   970.5971   99.8149
check '65'       563.607    472.058    978.6353   968.9770   101.3851
    
```

Figure 26. Fichier des points de calage du cliché 183

Le fichier des 16 paramètres « 183.Lparam » (cf. § 5.6.4.2) associés au cliché 183 (Figure 27):

```

Paramètres clichés 183
Point principal 1408.307962 944.865642
Position camera 972.363048 943.184743 98.203245
'L1' -3.509892e+000
'L2' 4.309887e-001
'L3' -2.879203e-001
'L4' 3.034662e+003
'L5' 4.284837e-002
'L6' 3.221927e-002
'L7' -3.440851e+000
'L8' 2.658499e+002
'L9' -3.267708e-004
'L10' -7.040387e-004
'L11' -1.855499e-004
'L12' 1.266155e-008
'L13' -9.791705e-015
'L14' 2.754942e-021
'L15' -3.213935e-007
'L16' 6.243327e-007
    
```

Figure 27. Fichier des paramètres du cliché 183

Le fichier des résidus « resid180.dlt » (cf. § 5.6.4.5). Il contient les résidus sur la position des points de contrôle et celle des points de vue selon les trois axes du système de coordonnées. Ce sont ces données que nous allons exploiter pour déterminer la précision de cette application de la DLT (Figure 28) :

Résidus			
Position des points de contrôle (dX,dY,dZ)			
ID '26'	-0.039	0.063	0.093
ID '97'	-1.010	-0.176	0.030
ID '16'	0.021	-0.023	-0.016
ID '33'	-0.002	-0.011	0.004
ID '94'	0.037	0.132	0.050
ID '50'	0.022	0.022	0.003
ID '53'	0.038	-0.026	-0.015
ID '77'	0.080	-0.041	0.025
ID '74'	0.089	0.055	0.064
ID '65'	0.048	0.294	0.076
Position des points de vues (dX,dY,dZ)			
Cliché '181'	1.501	-0.375	-0.511
Cliché '183'	-4.920	-8.347	0.107

Figure 28. Fichier des résidus associés aux clichés 181 et 183

Remarque : le point de contrôle 97 n'a pas été utilisé dans l'analyse de précision car il présente une erreur trop importante selon X et cela pour l'ensemble des techniques testées. Cette erreur est certainement due à mauvaise manipulation lors de la mesure du point avec la station totale.

6.4.2. Quid de la méthode du *LEMA*

L'objectif initial était de comparer notre méthode de levé avec celle utilisée par le laboratoire du *LEMA*. Malheureusement, nous n'avons pas pu décrire rigoureusement la précision de leurs données. La cause en est l'inexactitude des éléments représentés sur le modèle tridimensionnel. Un bref comparatif du cliché (Figure 20) et du modèle tridimensionnel (Figure 29) permet de se rendre compte des difficultés que nous avons rencontrées.

Toutefois, nous avons simplement comparé les trois distances (Figure 29). Les écarts avec la réalité sont de 4,353 m sur la distance représentée en bleu, 4,313 m sur celle en rouge et 1,948 m sur la dernière, en vert (Annexe 14). Notons, que lorsque nous utilisons la DLT, les écarts sont du même ordre que ceux obtenus pour les tests des appareils de prises de vue (cf. § 6.4.5).



Figure 29. Représentation de la façade sur l'ancien commissariat de police qui a servi au test de la DLT

Extrait de la restitution effectuée par le laboratoire du *LEMA*

Dans ces conditions, une comparaison approfondie n'a pas d'utilité car il est, en effet, incontestable que la DLT permet d'obtenir une précision supérieure à la méthode utilisée par le *LEMA*. Il pourrait par contre être intéressant d'effectuer nous-même le levé afin de déterminer la précision réelle de cette méthode. Malheureusement, nous n'avons pas pu mettre en oeuvre cette expérimentation. Les raisons principales étant le manque de temps disponible et l'incertitude sur l'obtention de résultats exploitables.

6.4.3. L'analyse de la variance

L'analyse de la variance consiste à comparer la distribution d'une variable quantitative d'une, ou plusieurs, populations. Cela revient en statistique à comparer la moyenne de chacune des populations. Bien que cela paraisse étonnant, c'est bien la moyenne qui exprime de mieux la variabilité entre les différentes populations. Par contre, la variabilité d'un échantillon d'une population est, elle, quantifiée par son écart type ou sa variance.

Dans le cas qui nous occupe, la variable quantitative est la valeur absolue de la distance les coordonnées mesurées et la les coordonnées de référence. Les populations étudiées sont les différentes techniques de levé, elles sont au nombre de quatre.

Nous nous trouvons dans le cas de la comparaison de plus de deux moyennes. Si nous concluons à une différence significative entre les moyennes, nous devons encore déterminer celles qui diffèrent les unes des autres, ce qui permettra d'établir un classement des différentes techniques, en fonction de leur précision. La méthode que nous allons utiliser est appelée comparaisons multiples de *Scheffé*.

6.4.3.1. Énoncé du test

Nous allons tester l'égalité des moyennes des quatre populations, soit une même distribution de la variable pour chacune d'entre elles. Nous testons l'hypothèse nulle, l'homogénéité des moyennes :

$$H_0 : \mu_1 = \mu_2 = \dots = \mu_k \quad (38)$$

contre l'hypothèse alternative, l'hétérogénéité des moyennes :

$$\exists i, j \in \{1, 2, 3, 4\}, i \neq j : \mu_i \neq \mu_j \quad (39)$$

L'hypothèse de base H_0 suit une distribution de F de Snedecor à 3 (nombre de populations moins 1) et 36 (nombre total d'effectif moins le nombre de populations) degrés de liberté. Si la valeur de F de Snedecor observée, est supérieure au quantile $1-\alpha$, l'hypothèse de base sera rejetée. La décision s'énonce de la manière suivante :

« On rejette H_0 si $F_{obs} \geq Q_F(1-\alpha; k-1; n-k)$, sinon on ne rejette pas H_0 . »

6.4.3.2. Table des données

Nous disposons de quatre populations et de 9 individus par échantillon, soit un total de 36 individus. Nous notons respectivement la moyenne et l'écart type de chaque échantillon \bar{x}_i et s_i . Ces informations sont reprises dans le tableau ci-dessous (Tableau 4).

Tableau 4. Distance entre le point reconstruit avec le DLT et la vérité terrain
Valeurs exprimées en mètres

<i>DLT-3 18 mm</i> n ₁ =10	<i>DLT 18mm</i> n ₂ =10	<i>DLT 35mm</i> n ₃ =10	<i>DLT Sony</i> n ₄ =10
0,370289076	0,118907527	0,165520391	0,200937801
0,163131849	0,035014283	0,190223553	0,183668179
0,007141428	0,011874342	0,039268308	0,197468985
0,174112033	0,145921212	0,026172505	0,179513231
0,015362291	0,031256999	0,031160873	0,227890324
0,13876599	0,0484252	0,206642203	0,193762742
0,21988406	0,093305948	0,375821766	0,180280337
0,079063266	0,122645832	0,156476835	0,214464449
0,405705558	0,307434546	0,085070559	0,171341764
$\bar{x}_1 = 0,174828395$ $s_1 = 0,14039956$	$\bar{x}_2 = 0,101642877$ $s_2 = 0,090418147$	$\bar{x}_3 = 0,141817444$ $s_3 = 0,112602905$	$\bar{x}_4 = 0,194369757$ $s_4 = 0,018181647$

6.4.3.3. Calculs

À partir des données résumées dans le tableau précédent (Tableau 4), nous pouvons tester l’hypothèse d’homogénéité des moyennes des quatre échantillons. Nous ne présentons, ici, que la valeur des résultats (Tableau 5). Cependant, l’ensemble des calculs réalisés est présenté en annexe (Annexe 10).

Tableau 5. Table récapitulative de l’analyse de la variance

<i>Source de variabilité</i>	<i>Somme des carrés</i>	<i>Degrés de liberté</i>	<i>Carré moyen</i>	<i>Valeur $F_{calculé}$</i>
Inter population	0,044553878	3	0,014851293	1,452539216
Intra population	0,327179715	32	0,010224366	
Total	0,371733593	35		

Vu ce qui précède, nous ne rejetons pas H_0 car $F_{obs} = 0,139 \leq Q_F(0,95; 3; 36) = 2,87$ et nous concluons à une différence non significative des distributions des quatre populations étudiées. D’un point de vue statistique, il n’y a donc pas de différence entre les précisions obtenues avec les quatre appareils de prise de vue testés.

6.4.4. Comparaisons multiples de Scheffé

Nous avons montré par l'analyse de la variance et le test F de Snedecor qu'il n'existe pas de différence significative entre les moyennes des populations étudiées (cf. § 0). Le travail est donc terminé et nous ne pouvons pas établir un classement des populations sur base statistique selon la méthode dite des *intervalles de confiances simultanés de Scheffé*. Cependant, nous allons quand même expliciter son raisonnement car il est la suite logique de ce qui vient d'être fait.

Les *intervalles de confiance simultanés de Scheffé* consistent à comparer les différences entre les moyennes des populations deux à deux à une différence critique, soit :

$$d_{i,j} = \mu_i - \mu_j \quad (40)$$

La différence entre les moyennes des populations i et j . Et :

$$d_{i,j}^*(\alpha) = \sqrt{(k-1)Q_F(1-\alpha; k-1; n-k)s_p^2 \left(\frac{1}{n_i} + \frac{1}{n_j} \right)} \quad (41)$$

La différence critique pour les populations i et j . Ici, comme le nombre d'effectifs de chaque population est égal, la différence critique est la même pour tous les couples de populations.

6.4.4.1. Énoncé du test

Pour chaque couple de population, on teste l'hypothèse nulle

$$H_0^{(i,j)} : \mu_i = \mu_j \quad (42)$$

contre l'hypothèse alternative

$$H_1^{(i,j)} : \mu_i \neq \mu_j \quad (43)$$

À partir des valeurs de $d_{i,j}$ et $d_{i,j}^*(\alpha)$ [équation (40) et équation (41)], la décision s'énonce de la manière suivante :

« On rejette $H_0^{(i,j)} : \mu_i = \mu_j$ si $|d_{i,j}| \geq d_{i,j}^*(\alpha)$, sinon on ne rejette pas $H_0^{(i,j)}$. »

6.4.5. Analyse des résultats

Tout d’abord, nous devons insister sur le fait que le choix du site de test n’a pas pu être sélectionné en fonction de critères objectifs tels que la configuration des lieux, l’exposition des façades, ... Nous avons été contraints de modifier notre premier site de test (Figure 19) qui nous paraissait pourtant idéal au niveau de ces critères à cause d’éléments indépendants de notre volonté.

Cette mauvaise configuration des lieux s’est directement répercutée sur la détermination des paramètres. En effet, nous avons eu beaucoup de difficultés pour déterminer les paramètres des clichés. Ces problèmes révèlent les limites de la DLT en milieu urbain et, surtout, dans des conditions difficiles, lorsque l’angle horizontal ou vertical entre le plan de la façade et l’axe optique devient faible.

Malgré tout, nous avons quand même obtenu des précisions acceptables. Et nous entrons tout de même dans les tolérances d’une vingtaine de centimètres que nous nous étions fixés (cf. § 2.2.2.1). En analysant les moyennes et les écarts type de différents échantillons (Tableau 4) sans se baser sur les conclusions statistiques (cf. § 6.4.3) nous pouvons affirmer que :

- L’utilisation d’un appareil reflex numérique apporte un gain de précision appréciable ;
- Travailler sur base d’un triplet stéréoscopique n’augmente pas la précision de la restitution.

6.4.6. Position des points de vues

Nous avons vu qu’il était possible d’exprimer la position du point de vue en fonction des onze paramètres standard de la DLT [équation (37)] (nous l’appelons la *position calculée*). La comparaison de cette valeur avec les coordonnées tridimensionnelles mesurées avec la station totale laser (cf. § 5.6.2.2) nous donne un bon indicateur de la qualité de l’ajustement réalisé (nous l’appelons *position mesurée*).

Les valeurs de la position des coordonnées mesurées ont été collectées lors de notre seconde campagne de mesures (cf. § 6.2.3) et nous avons déterminé la position calculée des points de vue à partir des paramètres des clichés (cf. § 5.6.2.2).

Les résultats obtenus sont présentés dans le tableau ci-dessous (Tableau 6). Les Δx , Δy , Δz représentent selon les trois axes de notre système terrain, la différence entre la position mesurée et calculée des points de vue.

Tableau 6. Écart entre la position mesurée et calculée des points de vue
Données : ensemble des clichés utilisés pour le test de la DLT (ancien commissariat)

<i>Id cliché</i>	<i>dx (m)</i>	<i>dy (m)</i>	<i>dz (m)</i>	<i>distance (m)</i>
181	1,501	-0,375	-0,511	1,629
182	-0,174	0,259	0,171	0,356
183	-4,92	-8,347	0,107	9,69
351	5,298	-0,81	-1,337	5,524
353	5,304	9,461	0,179	10,848
501	4,336	-0,514	-1,292	4,553
503	0,356	0,421	0,281	0,619

Ces résultats sont surprenants, ils ne reflètent absolument pas la précision obtenue lors de la restitution. Ils s'expliqueraient peut-être par une erreur dans notre développement théorique (cf. § 5.6.2.2). Mais, nous n'irons pas plus loin dans l'analyse de ces résultats, cependant il serait très intéressant de trouver la raison exacte qui conduit à ce problème.

6.5. Validation des choix de compensation globale

Au cours des développements théoriques précédents (cf. § 4.4), nous avons expliqué nos choix en matière de compensation globale. Pour rappel, nous avons choisi de n'utiliser aucune méthode de compensation lorsque plus de deux couples stéréoscopiques sont nécessaires à la restitution d'un édifice architectural. Nous admettons que la précision de l'ajustement effectué avec la DLT est suffisante.

Nous allons maintenant tenter de démontrer par un exemple que la technique de restitution par la méthode de la DLT permet d'obtenir directement des coordonnées terrain dans le référentiel tridimensionnel global. En tout cas, la précision obtenue par ce procédé est suffisamment grande pour des levés d'architectes ou d'urbanistes.

Notre expérimentation consiste à comparer les coordonnées des points restitués à partir de deux couples stéréoscopiques différents. Nous utiliserons les données acquises pour le test de la DLT sur l'ancien commissariat de police. Le premier couple est composé des

clichés 181 et 182, et le second du 182 et du 183 (Annexe 11). Les écarts entre 8 points communs aux deux couples sont présentés dans le tableau ci contre (Tableau 7).

Tableau 7. Écart sur les coordonnées restituées à partir de deux couples stéréoscopiques
Données : couples 181-182 vs 182-183 (ancien commissariat)

<i>Id point</i>	<i>dx (m)</i>	<i>dy (m)</i>	<i>dz (m)</i>	<i>distance (m)</i>
26	0,358	-0,49	-0,479	0,773
97	-0,02	-0,057	-0,122	0,136
33	-0,001	-0,005	-0,011	0,012
94	0,033	-0,06	-0,003	0,069
50	-0,005	0,046	0,023	0,052
53	0,23	0,672	0,243	0,751
74	0,015	0,23	0,099	0,251
65	0,122	-0,194	0,088	0,245
Moyenne	0,092	0,018	-0,02	0,286
Écart type	0,137	0,336	0,213	0,306

Nous remarquons, évidemment, qu’il existe des écarts entre les points restitués selon l’un ou l’autre couple stéréoscopique. Nous expliquons les plus grands écarts observés par une calibration des images qui n’a pas pu être optimale suite aux difficultés que nous avons rencontrées.

Cependant, si nous tenons compte des mauvaises conditions dans lesquelles ont été effectuées les mesures, ces écarts peuvent être considérés comme étant du même ordre que ceux que nous avons rencontrés lors du test de la DLT (DLT 18 mm : écart avec le vérité terrain (Tableau 4)).

Pour des travaux de photogrammétrie réclamant une précision moyenne nous validons, par l’expérimentation, nos choix en matière de compensation globale effectués lors des développements théoriques antérieurs (cf. § 4.4).

6.6. Cas concret de restitution

Pour mettre en pratique l’ensemble des concepts théoriques exposés dans ce mémoire, nous avons choisi d’appliquer notre méthode de restitution par voie photogrammétrique sur l’entièreté d’un bâtiment de la zone d’étude. Afin de pouvoir restituer l’ensemble de

l'immeuble, et pour des raisons esthétiques, nous avons choisi l'hôtel de ville de Verviers (Figure 19).

Dans cette section, nous allons présenter la séquence d'action qui compose le levé complet d'un bâtiment, par voie photogrammétrique, en utilisant la méthode de la DLT. Nous les détaillerons une à une et nous donnerons une approximation du temps nécessaire pour leur réalisation. Cette dernière partie de ce mémoire peut aussi être perçue comme une méthodologie spécifique à l'application de la DLT en milieu urbain, et servir de base à tout travail de restitution ultérieur.


6.6.1. Visite du site de mesure

Cette première visite est indispensable pour ce rendre compte du travail qui va devoir être réalisé. Elle nous sert à choisir la configuration idéale pour la position des différents points de vue ainsi que celle des stations à partir desquelles seront levés les points de calage (cf. § 6.6.2). Elle permet en outre d'avoir une bonne vision globale du site de mesure.

Pour minimiser la durée totale de la campagne de mesure, tant au niveau de l'acquisition de données qu'au niveau des traitements en laboratoire, nous avons choisi de réaliser une couverture stéréoscopique minimale pour chaque point du bâtiment (Figure 30). Une fois que la position de tous les points de vue est choisie, nous pouvons entreprendre l'acquisition des images (Annexe 12). L'appareil utilisé pour la restitution, est un Canon IXUS 750 (Tableau 8), nous n'avons pas pu utiliser l'un des appareils que nous avons testés (cf. § 6.2.3.1) car seuls les clichés de la visite préliminaire (cf. § 6.2.1) sont antérieurs à la pose des filets de protection sur les façades de l'hôtel de ville. Suite à ce que nous avons conclu du test de la DLT (cf. § 6.4.5), l'utilisation d'un appareil reflex numérique aurait pu augmenter la précision de la restitution.

Tableau 8. Caractéristiques techniques de l'appareil.

Canon IXUS 750

Type	Illustration	Référence boîtier	Référence objectif	Taille du capteur (pixels)	Stockage	Focale approchée (millimètres)
Compact numérique		Canon IXUS 750	Canon 7,7-23,1 mm 2,8-4,9	3072 x 2304	JPEG (SanDisk SD, 256 MB)	9 (données Exif)

Afin d'augmenter, le plus possible, la qualité des images, nous choisissons la résolution maximale et le taux de compression minimal possible pour l'appareil. De plus, il est positionné sur un pied pour lui donner le plus de stabilité possible (un pied photo classique est amplement suffisant).

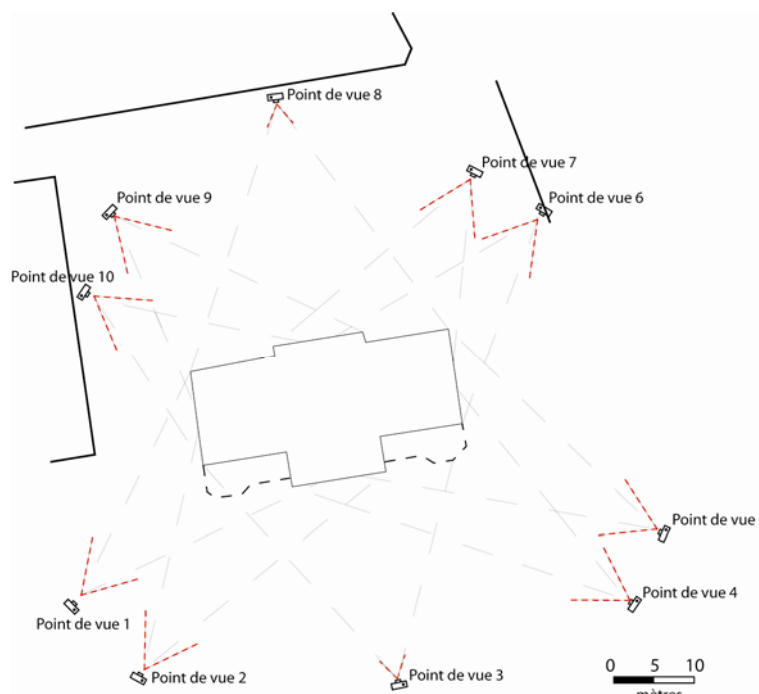


Figure 30. Position (approchée) des points de vue autour de l'hôtel de ville
Configuration permettant la couverture totale du bâtiment

Nous pouvons estimer la durée totale de cette première visite à une ou deux heures, tout au plus. Cette valeur, assez faible, s'explique par le fait qu'il y a très peu de contraintes quant à la manière de réaliser les prises de vue.

6.6.2. Mesure des points de calage

L'étape préalable à la mesure des points, est l'identification des points de calage sur les images. Toujours dans le but de minimiser la durée totale de traitement, nous avons choisi d'utiliser dix points de calage par cliché, au minimum. Nous avons ensuite utilisé un logiciel de dessin vectoriel (*Adobe Illustrator CS*) pour ajouter un à un les différents points sur les images et imprimer ces dernières sur un grand format (A3). Cette démarche permet d'éviter beaucoup de confusions lors du levé (Annexe 14).

Pour une description plus exhaustive de la mesure des points et des calculs réalisés, nous renvoyons aux notes sur la première campagne de mesure (cf. § 6.2.2).

Avec un peu d'entraînement et d'automatismes, nous avons atteint un rendement de dix points mesurés en quatre minutes. En comptant qu'il a fallu mesurer la position d'environ 120 points de calage, nous arrivons à une durée totale, approximative, d'une heure à laquelle nous devons ajouter le temps d'implantation de mesure de la polygonale. Elle est composée de quatre points, le temps nécessaire est de deux heures à raison d'une demie heure par point : soit une addition totale de trois heures.

Il est évident que cette étape peut être effectuée par un organisme extérieur si l'opérateur ne possède pas la technologie nécessaire. Dans ce cas, on soignera particulièrement l'identification des points de calage et leur visualisation sur les supports A3.

6.6.3. Digitalisation

Pour la méthode de digitalisation et les traitements préalables appliqués aux images, nous renvoyons à la section qui y est consacrée (cf. § 6.3). En effet, nous avons appliqué une même séquence de traitements identiques à celle qui a été mise en oeuvre pour les tests de précision de la DLT.

Nous avons donc digitalisé, dans le logiciel *IDRISI*, l'entièreté des points de calage ainsi que les points à restituer : ce qui fait un total de plus de 500 points homologues (1000 points à digitaliser).

Une géocodification simple a également été développée afin de diminuer le volume de post traitement (Tableau 9). Comme notre application ne supporte pas la géocodification, nous avons utilisé un petit artifice permettant d'intégrer le code dans l'identifiant. Le *pseudo identifiant* construit se présente de la manière suivante :

id999code où *id* est l'identifiant du point et *code*, le code qui lui est associé

Par une simple manipulation dans un éditeur de texte, nous pouvons, à la suite des calculs, séparer l'identifiant et le code du *pseudo identifiant* qui a été construit.

Tableau 9. Géocodification utilisée pour la restitution de l'hôtel de ville de Verviers
Suffixes, liaisons et blocs

<i>Suffixes</i>	
0	début liaison droite
3	point dans une liaison droite
<i>Liaison</i>	
11	escalier
31	limite pierre
41	
81	axe vertical façade
<i>Blocs</i>	
10	point escalier
20	centre horloge
30	point mur
50	point corniche
60	

En ce qui concerne la durée de cette étape consacrée à la mesure des coordonnées des points dans le système bidimensionnel image, nous avons eu besoin de deux jours complets. Cependant, la procédure pourrait être considérablement accélérée. En automatisant, notamment, l'écriture des coordonnées image dans les fichiers qui sont introduits dans l'application.

6.6.4. Détermination des paramètres et restitution

Pour chaque couple stéréoscopique, nous avons utilisé notre application pour calculer les 16 paramètres de chaque cliché et ensuite, restituer les points visibles sur ce couple

(Annexe 14). Les paramètres sont acceptés lorsque l'erreur quadratique moyenne est inférieure à 1,5. Nous avons obtenu des précisions sur les points de contrôle qui restent dans les tolérances fixées au début de ce travail. À titre d'exemple nous avons montré quelques résidus dans le tableau ci-dessous (Tableau 10).

Tableau 10. Exemple de résidus sur 4 points de contrôle
Restitution de l'hotel de ville : couple stéréoscopique 2-4

<i>Id point</i>	<i>dx (m)</i>	<i>dy (m)</i>	<i>dz (m)</i>	<i>distance (m)</i>
109	-0,099	-0,169	0,036	0,199
110	-0,036	-0,161	0,02	0,166
111	-0,35	-0,183	0,035	0,189
351	0,034	-0,14	0,01	0,145

Notons que nous avons eu quelques problèmes pour déterminer les paramètres du cliché numéro 5. Ces difficultés se sont ressenties lors de la restitution et ont été traduites par une grande imprécision sur les points restitués à partir des paramètres de ce cliché.

La durée de traitement est négligeable par rapport au temps mis pour la digitalisation des points. Nous ne la comptabilisons donc pas dans la durée totale.

6.6.5. Traitement des points restitués

À partir d'un fichier texte contenant, pour chaque point, dans l'ordre : l'identifiant, X, Y, Z, et le code, le logiciel de topographie *COVADIS* crée une *géodatabase*. Cette base de données est interprétée et le dessin est automatiquement créé dans *AutoCAD*.

Ensuite, nous devons reconstruire manuellement toutes les faces 3D qui composent ce bâtiment. Elles pourront supporter une texture et permettre un meilleur rendu graphique du plan (Figure 31).



Figure 31. Exemple de modèle 3D texturé pouvant être obtenu
Reproduction grand format du plan en annexe (Annexe 13)

Cette étape, assez fastidieuse, nous a demandé beaucoup de travail, notamment parce que nous avons dû nous familiariser avec le dessin en trois dimensions dans *AutoCAD* et les différents principes du texturage. Nous pouvons chiffrer sa durée approximativement à une journée et demi de travail.

6.6.6. Analyse

La restitution tridimensionnelle d'un bâtiment d'importance est réalisable avec la méthode de la DLT. Nous avons montré qu'il est possible, avec une méthode de levé photogrammétrique telle que la DLT d'obtenir un très bon résultat dans un délai d'une petite semaine (une demi journée sur le terrain et, entre trois jours et demi et quatre jours de travail en laboratoire).

Malgré l'utilisation de nombreux couples stéréoscopiques, aucune méthode de compensation globale n'a été appliquée (cf. § 4.4). Nous voyons que cela n'occasionne aucun artefact visible, la représentation graphique des points restitués est fluide et ne présente pas de problème apparent au niveau des liaisons entre les modèles.

7. Conclusion

7.1. Conclusion générale

La modélisation tridimensionnelle de fragments urbains fait l'objet de nombreuses recherches scientifiques. Celles-ci sont très intéressantes pour les Architectes et les Urbanistes, qui utilisent très largement ces techniques. Cependant, Il existe un grand nombre de techniques permettant ce type de levé et chacune d'entre elles possède des caractéristiques spécifiques.

Le nombre élevé de techniques qui peuvent être mise en œuvre ne facilite pas le choix d'un d'entre elles. Nous en avons sélectionné une en fonction de critères que nous a décrit le Laboratoire d'Etude Méthodologique Architectural (*LEMA*) telles que la facilité et la rapidité de mise en œuvre, la précision, le type de matériel requis,...

Afin de prendre les meilleures décisions, notre sélection a été effectuée après avoir exposé et analysé, par rapport aux recommandations émises par le *LEMA*, les différentes méthodes existantes. À la suite de quoi, notre choix s'est porté sur une méthode photogrammétrique digitale permettant l'utilisation d'appareils conventionnels : la Direct Linear Transformation 3D ou DLT.

Dans ce travail, nous avons développé les concepts théoriques de la méthode photogrammétrique et conçu un applicatif permettant le calcul de cette méthode sans imposer de limites sur l'utilisation du nombre de points de calage et de clichés. Nous avons aussi développé d'autres fonctions connexes telles que le calcul des résidus sur les points servant à l'orientation des images, la création d'un fichier contenant les résidus sur les points de contrôle, le calcul de la position des points de vue à partir des paramètres de la DLT,...

À l'aide notre application, nous avons ensuite étudié l'influence de l'utilisation de différents appareils photographiques sur la précision du modèle tridimensionnel. Bien que les conditions d'expérimentation n'étaient pas idéales, nous avons pu démontrer que le modèle formé respecte les conditions initiales de matière de précision. En outre, nous avons montré que l'utilisation d'un appareil photographique de type reflex numérique permet d'augmenter de la qualité de la restitution.

Nous avons ensuite décrit une méthodologie spécifique et illustré chacune des étapes qui la composent. Elle peut servir de base à tout travail de restitution tridimensionnelle en milieu urbain utilisant la méthode de la Direct Linear Transformation.

Ce mémoire est donc une étape préalable indispensable à la généralisation de l'utilisation de la Direct Linear Transformation au levé de fragments urbains. Et nous avons montré par nos travaux théoriques et les illustrations pratiques qui en découlent que cette méthode permet une modélisation simple, rapide et précise de fragments urbains.

7.2. Perspectives

Enfin, pour terminer cette conclusion, il nous paraît intéressant de citer quelques perspectives qui ont été mises à jour durant l'accomplissement de ce mémoire. Nous en avons relevé trois, les deux premières d'entre elles s'attachent à préciser certains concepts théoriques, tandis que la troisième constitue une évolution logique du développement de notre applicatif.

Dans ce travail, nous avons fait l'hypothèse qu'aucune méthode de compensation globale n'était nécessaire pour la formation de modèles nécessitant l'utilisation de plusieurs couples stéréoscopiques. Toutefois, nous n'avons pas prouvé l'influence positive ou non sur la précision de la restitution. Il serait intéressant et constructif d'appliquer quelques unes des méthodes de compensation globale aux paramètres de la DLT et d'étudier les résultats obtenus.

Un travail complémentaire à ce mémoire, consisterait à développer une nouvelle fois la théorie que nous avons appliquée pour la détermination de la position du point de vue en fonction des paramètres de la DLT. Car, suite aux mauvais résultats que nous avons obtenus dans cette section, il serait intéressant d'en identifier les causes.

Finalement, une autre perspective intéressante, serait l'intégration de l'algorithme de la DLT dans un logiciel permettant l'obtention des coordonnées dans une image. Dans le texte, nous avons déjà cité le logiciel de photogrammétrie aérienne *DDPS*, il s'agirait en effet d'une très bonne solution. Cela compléterait les fonctions déjà existantes dans le logiciel et permettrait de combiner dans une même application la photogrammétrie terrestre et aérienne.

Bibliographie

- ABDEL-AZIZ Y. I. & KARARA H. M. 1971. Direct linear transformation from comparator coordinates into object space coordinates in close range photogrammetry. American Society of Photogrammetry (éditeur): *ASP Symposium of Close Range Photogrammetry*, actes du symposium, Fall Church, p. 1-18.
- ARNOULD R. E. 2000. *Topographie : instruments et méthodes*, collection SURFACES .Liège : Les Éditions de l'Université de Liège, 209 p.
- ARNOULD R. E. 2003. *Topographie : erreurs et ajustements*, notes de cours, Université de Liège, Faculté de Sciences, inédit, 85 p.
- BRYER A. 2003. *Technologie pour le levé de monument historiques : la photogrammétrie digitale comparée au laser scanner 3D*, mémoire de fin d'étude, Conservatoire de arts et métiers, Paris, inédit, 81p.
- CHALLIS J. H. & KERWIN D. G. 1992. Accuracy assessment and control point configuration when using the DLT for photogrammetry, *Journal of Biomechanics*, **25** (9), p. 1053-1058.
- CLARKE T.A. FRYER J.F. & WANG X. 1998. The principal point and CCD cameras. *Photogrammetric Record*, **16** (92), p. 293-312.
- COLLIGNON A. 2005. *Notion de photogrammétrie*, notes de cours, Université de Liège, Faculté des Sciences, inédit, 174 p.
- DA COL A. & SCHUMACKER B. 2002. *Digital & Didactic Photogrammetric Software: software help*. OSTC, Université de Liège, Laboratoire SURFACES, inédit 42 p.
- DONNAY J.-P. 2005. *S.I.G.*, notes de cours, Université de Liège, Faculté des Sciences, inédit, 466 p.
- DRAP P. 2002. *Un système de gestion de documents hétérogènes dédiées au patrimoine archéologique : l'épave Étrusque du Grand Ribaud*. Site de L'UMR 694 MAP (Modèles et simulations pour l'Architecture, l'urbanisme et le Paysage) (<http://www.map.archi.fr/theme1/axe/axe1/SourcePdf/RIBAUDrecto.pdf>), consultation le 20 octobre 2005.

- Eos Systems Inc. 2005. *Photodeler Pro single photo road skid tutorial for Photodeler Pro* 5. Manuel d'utilisation, site dédié au logiciel Photodeler de la société Eos Systems Inc. (<http://www.photodeler.com/downloads2/skiddtutor.pdf>), consultation le 8 juillet 2006.
- FAURE F. 2002. *Outils mathématiques : position, orientation et mouvement*. Site du Master en Recherche Imagerie, Vision, Robotique de l'Université Joseph Fourier de Grenoble. (http://www-evasion.imag.fr/Membres/Francois.Faure/enseignement/dea_ivr/OM/main.pdf), consultation le 5 juillet 2006.
- GÓMEZ LAHOZ J., CUADRADO MÉNDEZ O. & MARTINEZ RUBIO J. 2003. *Lens distortion simulation. An application for understanding geometric distortion*. Site du comité international de photogrammétrie appliquée à l'architecture (<http://cipa.icomos.org/fileadmin/papers/antalya/94.pdf>), consultation le 4 juillet 2006.
- GOVEARTS B. 2001. *Méthodes numériques pour la régression linéaire*. Site de l'institut de statistique de l'Université Libre de Louvain (<http://www.stat.ucl.ac.be/cours/stat2430/documents/c5reglin.pdf>), consultation le 11 juillet 2006.
- GRUSSENMEYER P., HANKE K. & STREILEIN A. 2002. *Architectural photogrammetry: basic theory, procedures, tools*. chapitre dans *Digital Photogrammetry*, édition M. KASSER and Y. EGELS, Taylor & Francis (2002), p. 300-339.
- JACOBS G. 2006 Profitable Uses for Marketing Proposals, *Professional Surveyor magazine*, January 2006. Site de la société Leica Geosystems (http://www.leica-geosystems.com/hds/en/lgs_50581.htm?id=6113?id=6113), consultation le 5 août 2006.
- JOSÉ MARTINS GOMES C., DA SILVA PRADO W., ERWES H. & GILSON DIMENSTEIN K. 1999. *A photogrammetric project in Brazil: the use of Photodeler software*. Site du comité international de photogrammétrie appliquée à l'architecture. (<http://cipa.icomos.org/fileadmin/papers/olinda/99c313.pdf>), consultation le 31 octobre 2005.
- KERSTEN TH., ACEVEDO PARDO C., LINDSTAEDT M. 2004. 3D acquisition, modelling and visualization of North German castles by digital architectural photogrammetry. *XXth ISPRS Congress*, Istanbul, Turkey, July 2004. Commission 5, part B5, p. 126-132.
- KRAUS K. 1993. *Photogrammetry, Fundamentals and Standard Processes*. Volume 1. Dümmler Verlag, Bonn, 377 p.
- KRAUS K. 1997. *Photogrammetry, Advanced Methods and Applications*. Volume 2. Dümmler Verlag, Bonn, 446 p.
- KWON Y.-H., 1998. *The DLT Method*. Site Internet de Kwon3D. (<http://kwon3d.com/theory/dlt/dlt.html>), consultation le 12 octobre 2005.

- LAPLANCHE F. *Spécification d'une méthodologie de relevés adaptés aux enquêtes environnementales stratégiques en milieu urbain*, mémoire de troisième cycle, Université de Liège, Faculté des sciences appliquées, inédit, 95 p.
- MILLER N. R., SHAPIRO R. & McLAUGHIN T. M. 1980. A technique for obtaining spatial kinematic parameters of segment of biomechanical systems from cinematographic data. *Journal of Biomechanics*, **13**, p. 535-547.
- PERRIN J.-P. & CHEVRIER C. 2002. *Projet REVCAP: reconstruction 3D interactive de zones urbaines*. Site de L'UMR 694 MAP (Modèles et simulations pour l'Architecture, l'urbanisme et le Paysage). (<http://www.map.archi.fr/theme1/axe/axe2/SourcePdf/REVCAP.pdf>), consultation le 20 novembre 2005.
- Photo Mess System AG. 2004. *ELCOVISION Lens Distortion Correction*. Manuel d'utilisation, site dédié au logiciel Elcovision de la société PMS AG (http://www.elcovision.com/Software/ElcoPhoto_ENG.pdf), consultation le 29 octobre 2005.
- REMONDINO F. 2002. *3D reconstruction of articulated objects from uncalibrated images*, San Jose (California), USA, January 2002. Three-Dimensional Image Capture and Application V, Corner, Pargas, Nurre (éd.), SPIE Electronic Imaging, 4661, p. 148-154.
- RIBOT M. 2005. *Algorithme pour le problème des moindres carrés. Décomposition QR*. Site du laboratoire Dieudonné J. A. Université de Nice-Sophia Antipolis. (http://math1.unice.fr/~ribot/moindres_carres.pdf), consultation le 23 février 2006.
- SCHNEIDERS J.-P. 2005. *Introduction à l'analyse numérique*, notes de cours, Université de Liège, Faculté des Sciences, inédit, 146 p.
- STEVENS W. P. 1997. Reconstruction of three-dimensional anatomical landmark coordinates using video-based stereophotogrammetry. *Journal of Anatomy*, **191**, p. 277-284
- WONG K. W. 1975. Mathematical formulation and digital analysis in close range photogrammetry. *Photogrammetric Engineering and Remote Sensing*, **41**, p. 1355-1373
- WOOD G.A & MARSHALL R.N. 1986. The accuracy of DLT extrapolation in three-dimensional film analysis. *Journal of Biomechanics*, **19**, p. 781-785.

Liste des annexes

Annexe 1. Tableur Excel : calcul des erreurs optiques d'une image en vue de la redresser	114
Annexe 2. Liste des images utilisées pour la mise au point des tableurs Excel (Bâtiment B5c)	116
Annexe 3. Inventaire des points de calage : bâtiment B5	117
Annexe 4. Tableur Excel : coordonnées (image et terrain) de points de calage sur la façade du B5c.....	119
Annexe 5. Exemple de fichier Excel programmé pour effectuer l'orientation des clichés et la reconstruction tridimensionnelle d'objets par la méthode de la DLT (dlt_8pt.xls)	120
Annexe 6. Tableur Excel : calcul et représentation graphique des différents indices de convergence.....	122
Annexe 7. Code source commenté de l'application dlt3D.exe	124
Annexe 8. Liste des images utilisées pour le test de la DLT (ancien commissariat de police). 157	
Annexe 9. Extrait du rapport d'erreur sur la compensation de la polygonale autour de l'ancien commissariat de police	160
Annexe 10. Tableur Excel : analyse de la variance des 4 échantillons qui interviennent dans le test de la DLT	161
Annexe 11. Tableur Excel : calcul des écarts entre la restitution de mêmes points à partir de 2 couples stéréoscopiques différents	163
Annexe 12. Liste des images utilisées pour la restitution de l'hôtel de ville	164
Annexe 13. Plan : vue perspective de l'hôtel de ville de Verviers.....	168
Annexe 14. CD de données.....	169

Annexe 1. Tableur Excel : calcul des erreurs optiques d'une image en vue de la redresser

Correction géométrique d'une image (image n°182, ancien commissariat de police)

Fonction resample d'IDRISI

Données

u0	1474,14655
v0	978,04796
L12	-3,62E-08
L13	9,08E-15
L14	-9,84E-22
L15	1,71E-07
L16	1,34E-06
umax	3072
vmax	2304

u	v	ξ	η	r^2
100	100	-1374,14655	-878,04796	2659246,95
100	700	-1374,14655	-278,04796	1965589,4
100	1300	-1374,14655	321,95204	1991931,84
100	1900	-1374,14655	921,95204	2738274,29
1036	100	-438,146545	-878,04796	962940,615
1036	700	-438,146545	-278,04796	269283,063
1036	1300	-438,146545	321,95204	295625,511
1036	1900	-438,146545	921,95204	1041967,96
1972	100	497,853455	-878,04796	1018826,28
1972	700	497,853455	-278,04796	325168,731
1972	1300	497,853455	321,95204	351511,179
1972	1900	497,853455	921,95204	1097853,63
2908	100	1433,85346	-878,04796	2826903,95
2908	700	1433,85346	-278,04796	2133246,4
2908	1300	1433,85346	321,95204	2159588,85
2908	1900	1433,85346	921,95204	2905931,29
100,00	100,00	-1374,14655	-878,04796	2659246,95
100,00	700,00	-1374,14655	-278,04796	1965589,4
100,00	1300,00	-1374,14655	321,95204	1991931,84
1036,00	400,00	-438,146545	-578,04796	526111,839
1036,00	1000,00	-438,146545	21,95204	192454,287
1036,00	1600,00	-438,146545	621,95204	578796,735
1972,00	400,00	497,853455	-578,04796	581997,507
1972,00	1000,00	497,853455	21,95204	248339,955

Δu	Δv	U_{mod}	V_{mod}
72,2460981	50,2757408	27,7539019	49,7242592
61,3643815	15,0279172	38,6356185	684,972083
60,6899291	-11,2488423	39,3100709	1311,24884
69,8832111	-41,5363013	30,1167889	1941,5363
12,7198691	27,4249643	1023,28013	72,5750357
4,26714692	3,12358221	1031,73285	696,876418
4,27991796	-2,54736507	1031,72008	1302,54737
12,400157	-23,1077256	1023,59984	1923,10773
-14,5210332	28,3959581	1986,52103	71,6040419
-5,4463795	3,63707593	1977,44638	696,362924
-5,43932171	-2,97219633	1977,43932	1302,9722
-14,102396	-23,9242117	1986,1024	1923,92421
-75,1288956	51,3503146	2983,1289	48,6496854
-64,6841106	15,6516045	2972,68411	684,348396
-63,9333254	-11,4764941	2971,93333	1311,47649
-72,6030437	-42,1818401	2980,60304	1942,18184
72,2460981	50,2757408	27,7539019	49,7242592
61,3643815	15,0279172	38,6356185	684,972083
60,6899291	-11,2488423	39,3100709	1311,24884
7,80380175	11,2895306	1028,1962	388,710469
2,99437413	0,11245086	1033,00563	999,887549
7,73155523	-9,49105203	1028,26844	1609,49105
-9,25955234	12,1462659	1981,25955	387,853734
-4,06372255	0,15134624	1976,06372	999,848654
-9,13665984	-10,2300311	1981,13666	1610,23003

Annexe 2. Liste des images utilisées pour la mise au point des tableurs Excel (Bâtiment B5c)

Sony DSC W12 image de gauche : B5_gauche.bmp



Sony DSC W12 image du milieu : B5_milieu.bmp



Sony DSC W12 image de droite : B5_droite.bmp



Annexe 3. Inventaire des points de calage : bâtiment B5

Répertoire des points de contrôle

Une aide pour la localisation des points de contrôle mesurés sur la façade du bâtiment d'astrophysique et géophysique de l'Université de Liège est présentée ci-dessous. Elle reprend un inventaire de tous les points ainsi que leur identifiants et une petite description pour aider à la localisation du point.



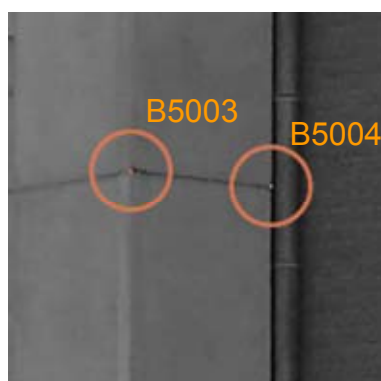
B5001

Situation : extrémité de la corniche au dessus de l'entrée du B5



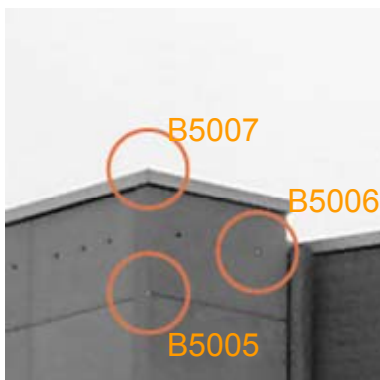
B5002

Situation : pierre à mi-hauteur du mur de l'entrée du B5.



B5003 et B5004

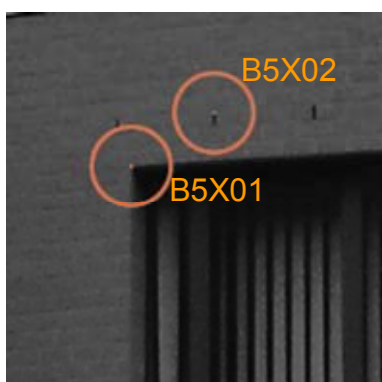
Situation : milieu de la tranche de la paroi en béton.



B5005, B5006 et B5007

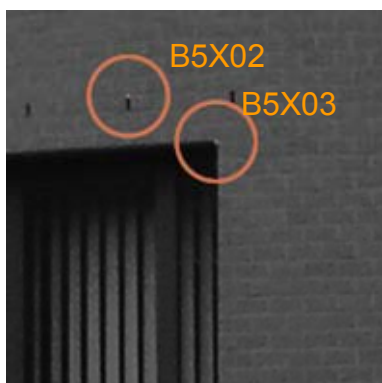
Situation : partie supérieure de la tranche de la paroi en béton.

Les points de contrôle situés dans l'encadrement des fenêtres de la façade du bâtiment B5c suivent une numérotation différente. En effet, pour faciliter leur localisation, les points correspondant à chaque fenêtre ont reçu un numéro allant de 1 à 5. Cette numérotation apparaît en troisième position dans la nomenclature de l'identifiant du point. On a par exemple B5303 qui signifie qu'il s'agit du point 3 de la fenêtre 3 du bâtiment B5. En suivant cette logique, on a donc :



B5X01 (X=1, 2, 3, 4), B5102 et B5402

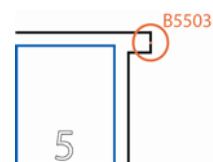
Situation : coin supérieur gauche de chaque fenêtre.



B5202 et B5302, B5X03 (X=1, 2, 3, 4, 5)

Situation : coin supérieur droit de chaque fenêtre.

Précision pour B5503



Annexe 4. Tableur Excel : coordonnées (image et terrain) de points de calage sur la façade du B5c

	U (pixels)	V (pixels)	U (pixels)	V (pixels)	U (pixels)	V (pixels)	X (m)	Y (m)	Z (m)
B5001	874,75	903,75	712	1009,75	701,5	991,5	1002,68	1016,009	101,452
B5002	1657,25	1259,5	824,5	1290,5	365,75	1256	1000,171	1010,457	99,083
B5003	1016,75	657,75	1061,5	781	1147,25	748,25	1007,252	1016,513	104,048
B5004	1052,75	681	1137	790,25	1245	751,75	1008,3	1016,523	104,052
B5005	1021,5	312,75	1065,25	500,5	1149,75	466,25	1007,264	1016,512	107,068
B5006	1050,25	299,5	1123,25	479	1226	434,75	1008,087	1016,52	107,437
B5007	1024,5	230,75	1064	434,25	1148	400	1007,248	1016,494	107,774
B5101	1224,25	419,5	1271	554	1335,25	495,75	1009,015	1015,137	106,516
B5102	1214,75	388	1315,5	526	1365,25	461	1009,224	1014,671	106,808
B5103	1394,75	434,75	1417,75	545	1441,75	469,5	1009,796	1013,716	106,501
B5201	1660	482	1714,25	542,5	1727,25	427	1011,982	1011,482	106,507
B5202	1726,25	474,25	1812,75	521,5	1848	386,75	1012,932	1010,903	106,835
B5203	1753	511,25	1857,5	549,75	1906,5	413,5	1013,411	1010,681	106,522
B5301	1223	747,5	1271,5	832	1335,25	787,25	1009,023	1015,146	103,524
B5302	1341,75	722	1371,25	801,5	1405,5	750,5	1009,528	1014,15	103,834
B5303	1394,25	755,75	1419,25	826,5	1443,25	773,5	1009,795	1013,711	103,514
B5401	1662,25	785	1718,25	825	1732	751	1011,994	1011,483	103,517
B5402	1695	763	1764,5	800	1786,75	718	1012,426	1011,207	103,811
B5403	1756,75	803	1862,5	828,25	1911,5	742,75	1013,407	1010,675	103,528
B5503	1760,75	1100,25	1868,5	1112,75	1917,75	1079,5	1013,434	1010,674	100,52

Annexe 5. Exemple de fichier Excel programmé pour effectuer l'orientation des clichés et la reconstruction tridimensionnelle d'objets par la méthode de la DLT (dlt_8pt.xls)

Microsoft Excel - dlt_8pt.xls

1 Calcul des 11 paramètres de la DLT pour le cliché 1
 2 Les paramètres en rouge doivent être entrés par l'utilisateur

paramètres de la DLT

1	1248,27822	1002,668	1016,003	874,75	300,75	-375,5282	-39,08018	149340,214
2	1002,63019	1000,171	1010,437	1657,25	1829,5	408,3716	256,6688	233197,309
3	-0,012796	1007,252	1016,513	1016,75	657,75	-231,5282	-345,0802	172685,654
4	-0,0500145	1006,3	1016,523	1052,75	1052,75	-195,5282	-321,8302	141805,856
5	-0,0208667	1007,254	1016,512	1021,75	312,75	-226,7782	-690,0802	527639,103
6	-0,0202086	1008,087	1016,52	1050,25	289,5	-195,0882	-703,3302	533888,533
7	-0,0202353	1007,248	1016,434	1084,5	290,75	-223,7782	-772,0802	646184,152
8	-0,0219231	1003,105	1015,137	1224,25	419,5	-84,02822	-583,3302	340851,467

matrices des coefficients

1	-52472,74	-53309,21	-6321,044	-62,20076	0	0	0	54648009,4	55374486,1	55939303
2	-78162,96	-78966,8	-7743,236	-78,14953	0	-62472,74	-63309,21	54453758,9	57210793,3	5712844
3	-50326,11	-50788,83	-5198,631	-43,96377	0	-78162,96	-78966,8	123533558	130863733	12532377
4	-48320,3	-48714,37	-4926,498	-47,32314	0	-50326,11	-50788,83	33101939,3	334063437	3419339
5	-43843,39	-50301,02	-5236,147	-43,48394	0	-48320,3	-48714,37	32906533,4	33174896,5	3395805
6	-48233,29	-48636,78	-5140,463	-47,64636	0	-43843,39	-43843,39	15589215	15731644,5	1656395
7	-49716,89	-50233,81	-5326,051	-43,4187	0	-48233,29	-50301,02	50671013,8	51080771,5	5396778
8	-47306,86	-47593,88	-4939,317	-46,8842	0	-49716,89	-47306,86	14445870,6	1456619,4	1538710

vecteur des termes inconnus

1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	0	0

position de la caméra

1	-1,55233093	2,031853	-0,091324054
2	0,3788137	-0,2402888	2,280534343
3	-0,0002811	-0,0001931	-6,11658E-05
4	511,607651		
5	-379,63657		
6	-1		
7	384,7024		
8	1003,081		
9	39,385328		

DLT

1	-1,55233093	2,031853	-0,091324054
2	0,3788137	-0,2402888	2,280534343
3	-0,0002811	-0,0001931	-6,11658E-05
4	511,607651		
5	-379,63657		
6	-1		
7	384,7024		
8	1003,081		
9	39,385328		

Formes automatiques

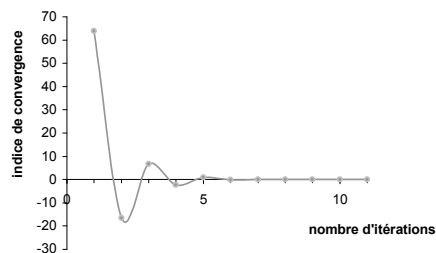
LI-L1II / LI-L1II / LI-L1IIII / XYZ /

Annexe 6. Tableau Excel : calcul et représentation graphique des différents indices de convergence

Représentation graphique de l'indice de convergence

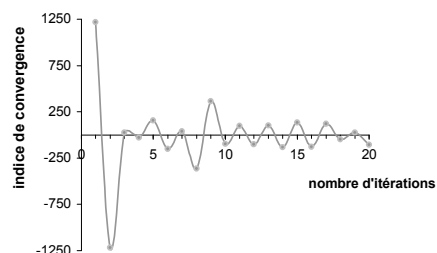
Décomposition QR

itération		diff L(i, i-1)		indice
1	1	4083,46499	63,9019952	63,9019952
2	-1	273,808259	16,5471526	-16,5471526
3	1	43,951442	6,62958837	6,62958837
4	-1	5,189329	2,27800988	-2,27800988
5	1	0,78839	0,88791328	0,88791328
6	-1	0,049493	0,22247022	-0,22247022
7	1	0,008724	0,09340236	0,09340236
8	-1	0,00207	0,04549725	-0,04549725
9	1	0,000332	0,01822087	0,01822087
10	-1	0,000051	0,00714143	-0,00714143
11	1	0,000009	0,003	0,003



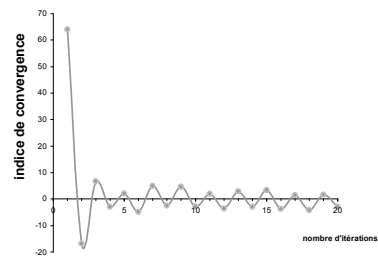
Inverse QR

itération		diff L(i, i-1)		indice
1	1	1483962,39	1218,17995	1218,17995
2	-1	1485292,87	1218,72592	-1218,72592
3	1	597,748371	24,448893	24,448893
4	-1	979,021481	31,289319	-31,289319
5	1	24252,0701	155,730762	155,730762
6	-1	23156,2643	152,171825	-152,171825
7	1	1484,16185	38,5248212	38,5248212
8	-1	133280,082	365,075447	-365,075447
9	1	132069,014	363,413007	363,413007
10	-1	9773,67435	98,8618953	-98,8618953
11	1	9453,33099	97,2282417	97,2282417
12	-1	10250,579	101,245143	-101,245143
13	1	10306,7019	101,521928	101,521928
14	-1	17698,2932	133,034932	-133,034932
15	1	17892,0946	133,761334	133,761334
16	-1	16268,6658	127,54868	-127,54868
17	1	14799,5364	121,653345	121,653345
18	-1	2073,0952	45,5312551	-45,5312551
19	1	648,411561	25,4639267	25,4639267
20	-1	11249,2587	106,062522	-106,062522



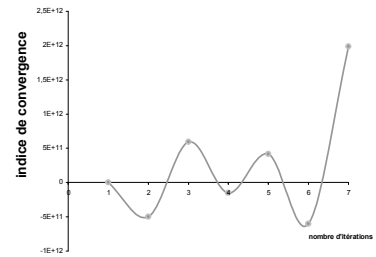
Inverse Jordan

itération		diff L(i, i-1)		indice
1	1	4093,29898	63,9788948	63,9788948
2	-1	283,638872	16,8415816	-16,8415816
3	1	44,576247	6,67654454	6,67654454
4	-1	8,753213	2,95858294	-2,95858294
5	1	4,730883	2,17505931	2,17505931
6	-1	23,741451	4,87251998	-4,87251998
7	1	24,530941	4,952872	4,952872
8	-1	6,024208	2,45442621	-2,45442621
9	1	21,959232	4,68606786	4,68606786
10	-1	7,544649	2,74675245	-2,74675245
11	1	4,095875	2,02382682	2,02382682
12	-1	12,821042	3,58064827	-3,58064827
13	1	8,090466	2,84437445	2,84437445
14	-1	8,307113	2,88220627	-2,88220627
15	1	11,004383	3,31728549	3,31728549
16	-1	13,509042	3,67546487	-3,67546487
17	1	2,073475	1,4399566	1,4399566
18	-1	17,772279	4,21571809	-4,21571809
19	1	2,581018	1,6065547	1,6065547
20	-1	9,082523	3,01372245	-3,01372245



Inverse Gréville

itération		diff L(i, i-1)		indice
1	1	1,6543E+12	1286178,56	1286178,56
2	-1	2,5124E+23	5,0124E+11	-5,0124E+11
3	1	3,5157E+23	5,9293E+11	5,9293E+11
4	-1	2,3668E+22	1,5385E+11	-1,5385E+11
5	1	1,7189E+23	4,146E+11	4,146E+11
6	-1	3,6434E+23	6,0361E+11	-6,0361E+11
7	1	3,9375E+24	1,9843E+12	1,9843E+12



Annexe 7. Code source commenté de l'application dlt3D.exe

CODE PRINCIPAL

```
//Code source de l'application dlt3D.exe programmée en C
//par Julien ERNST dans le cadre du mémoire de fin d'étude
//pour l'obtention du titre de licencié en géographie option géométrie et
géomatique

//L'application permet de déterminer les paramètres de la méthode DLT et
//de calculer à partir d'un certain nombre de clichés les coordonnées
tridimensionnelles
//de points homologues visibles sur chacun d'entre eux

#include "stdafx.h"
#include "MatrixLib.h"
#include "UtilLib.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>

#pragma warning( disable : 4996 )

void main(){

//variables
FILE * entree;
FILE * sortie;
int i, j, k, n,nbrautereparam ,ntotal=0,o,p,nb,nbpos=0,idrest, nbrparam,
nbparam=0,nb_check,total_check=0,nb_ctrl;
int cpti=4, minparam=20,
cptx=0, cptp=0, cptc=0,reponse,longueur,test,testcal,test2,test3;
int cliché ,id ,numcliche, pcliche;
long double u, v, x, y, z, l;
long double uo, vo, xo, yo, zo, xp, yp, zp;
long double valL,valP;
int X01,X02;
char *reptest, lire[100], str1[15], str2[15], str3[15], str4[15];
char nomfichcalage[20], nomfichparam[20], clichécrire[20],debut[5],cam[5],
aucun[5],param[5],autreparam[5],resid[5],avert[5],
disto[5],nomfichcal[20],rep[1000],path[100],residu[20],resultat[20];

//fonctions
int compte_lignes(FILE *fichier);
int test_oui_non(char reponse[5]);
matrix* position_camera(matrix *MATRICE,int k);
matrix* calcul_param(matrix *D,long double uo,long double vo,int nbcol, int
nbpt);
matrix* calcul_XYZ(matrix *Lfinal, matrix *CAL, int cptc, int ntotal, int
minparam);
matrix* calcul_uovo(matrix *L);
matrix* least_square(matrix *C, matrix*I);
int ecrire_param(FILE * sortie, matrix* L, matrix* POSmem, matrix* UoVo, int
nbparam,int cliché,int i);
int ecrire_param2(FILE* sortie, matrix* L, matrix* POSmem, int nbparam, int i);
matrix* calcul_resid(matrix* L, matrix* D, int nbparam, int nb_ctrl);
double calcul_rms(matrix* Resid, int nb_ctrl);

//tableaux
ptr_to_mat D;
ptr_to_mat CHECK;
ptr_to_mat L;
```

```

ptr_to_mat Resid;
ptr_to_mat Lmem;
ptr_to_mat Lfinal;
ptr_to_mat LP;
ptr_to_mat UoVo;

ptr_to_mat POSmem;
ptr_to_mat POS;
ptr_to_mat POSo;
ptr_to_mat POSdelta;
ptr_to_mat POSfinal;
ptr_to_mat POSp;

ptr_to_mat CAL;
ptr_to_mat CAL_check;
ptr_to_mat RESULTAT;
ptr_to_mat RESULTAT_check;
ptr_to_mat RESIDU_check;
ptr_to_mat ID;

LP=CreateMat(19,1);
POSp=CreateMat(4,1);

printf("\n\t\t\t\t\t****BIENVENUE: methode de la DLT 3D****\n\n");//bienvenue!!
repertoire:printf(">>>Veuillez entrer repertoire de travail: ");//rep de
travail
gets(rep);
longueur=int(strlen(rep));
reptest=strchr(rep,'\\');//test
if(longueur!=(reptest-rep+1)){//message d'erreur
printf("\n>>>VOTRE REPERTOIRE N'EST PAS VALIDE! se termine-t-il par '\\\\'<<\n\n");
goto repertoire;
}
//Si les param ne sont pas encore calcule
para:printf("\nAvez-vous deja calcule les parametres des cliches a
traites?(oui/non) ");
scanf("%s",debut);

reponse=test_oui_non(debut);
if(reponse==1)goto para;
if(strcmp(debut,"oui")==0){
n=0;
goto newparam;
}

////////////////////////////////////
//*****
//* module de calcul des paramètres des clichés *//
//*****
////////////////////////////////////

debut:printf("\n\t\t\t\t\t****BIENVENUE: Calcul des parametres de la DLT****\n");
printf("\nCombien de cliches voulez vous traiter: ");
scanf("%d",&n);

//ouverture des fichiers coordonnées image (u,v,x,y,z)

Lmem=CreateMat(19, n);//création du tableau des tous paramètres qui vont etre
calculés
POSo=CreateMat(4,n);
POSmem=CreateMat(4,n);

for(i=0; i<n; i++){
//entrée du nom des fcheir de points de calage

```

```

uncalage:if(i==0)printf("\n>>>Nom du %der fichier de point de calage et numero du
cliche: ",(i+1));
else printf("\n>>>Nom du %deme fichier de point de calage et numero de cliche:
", (i+1));
strcpy(path,rep);
scanf("%20s %d",nomfichcalage, &numcliche);
strcpy(clichecrire,nomfichcalage);
strcat(nomfichcalage, ".txt");
strcat(path,nomfichcalage);
entree = fopen (path, "r");//ouverture du fichier
if (entree == NULL){
perror ("\n>>LE FICHIER N'A PAS PU ETRE OUVERT<<");
goto uncalage;
}
else {
cpti=compte_lignes(entree)-3;
D=CreateMat(cpti,5);
if (i==0) CHECK=CreateMat(n*cpti,7);

fseek(entree, 0, SEEK_SET);

//parcours le fichier texte contenant les données

fgets(lire,100,entree);
sscanf(lire,"%s %s %s %s %d", str1, str2, str3, str4, &cliche);
itoa(cliche,path,20);
if(numcliche!=cliche){
printf("\n>>LE NUMERO DU CLICHE NE CORRESPOND PAS, veuillez verifier...<<");
goto uncalage;
}
if(i>0){
for(j=0; j<n-1; j++){
test=int(ValRead(POSmem,0,j));
if(test==cliche){
printf("\n>>LES PARAMETRES DE CE CLICHE ONT DEJA ETE CALCULES, veuillez
verifier...<<");
goto uncalage;
}
}
}
fgets(lire,100,entree);
sscanf(lire,"%s %s %lf %lf",str1, str2, &uo, &vo);
fgets(lire,100,entree);
sscanf(lire,"%s %s %lf %lf %lf",str1, str2, &xo, &yo, &zo);

//test ligne vide=>si non=> on calcule la position camera, et delta avec mesuré

if(strcmp(str1,"Taille")!=0){
ValAssign(POSo,0,i,0);
ValAssign(POSo,1,i,xo);
ValAssign(POSo,2,i,yo);
ValAssign(POSo,3,i,zo);
X01=0;
nbpos++;
}
else X01=1;

nb_ctrl=0;
nb_check=0;
while((fgets(lire,100,entree))!=NULL){
sscanf(lire,"%s '%d' %lf %lf %lf %lf %lf",str1, &id, &u, &v, &x, &y, &z);
if(strcmp(str1,"ctrl")==0){//copie des points ctrl dans un tableau
ValAssign(D,nb_ctrl,0,u);//u
ValAssign(D,nb_ctrl,1,v)//v
ValAssign(D,nb_ctrl,2,x)//x

```

```

ValAssign(D,nb_ctrl,3,y); //y
ValAssign(D,nb_ctrl,4,z); //z
nb_ctrl++;
} //copie des points check dans un tableau
if(strcmp(str1,"check")==0){
nb=total_check+nb_check;
ValAssign(CHECK,nb,0,cliche); //id cliché
ValAssign(CHECK,nb,1,id); //id pt
ValAssign(CHECK,nb,2,u); //u
ValAssign(CHECK,nb,3,v); //v
ValAssign(CHECK,nb,4,x); //x
ValAssign(CHECK,nb,5,y); //y
ValAssign(CHECK,nb,6,z); //z
nb_check++;
} //test du check ou ctrl qui doit être présent
if(strcmp(str1,"check")!=0 && strcmp(str1,"ctrl")!=0){
printf("\n>>ERREUR : DENOMINATION DU TYPE DE POINT, veuillez verifier...<<");
goto uncalage;
}
}
fclose(entree);
total_check=total_check+nb_check;
if(nb_check != total_check/(i+1)){
avert:printf("\n>>AVERTISSEMENT : NOMBRE DE POINTS 'check' ENTRE LES FICHIERS,
veuillez verifier...<<\nIgnorer l'avertissement?(oui/non) ");
scanf("%s",avert);
reponse=test_oui_non(avert);
if(reponse==1)goto avert;
if(strcmp(avert,"non")==0){
total_check=total_check-nb_check;
goto uncalage;
}
}
} //identification du nombre de param qui peuvent être calculés
if (nb_ctrl<6){
printf("Le fichier ne contient que %d pts de controle, aucun parametre ne peut
être calculé\n",nb_ctrl);
goto uncalage;
}
if (nb_ctrl<7){
printf("Le fichier contient %d pts de controle, les 11 parametres standards
peuvent être calculés\n",nb_ctrl);
nbparam=11;
}
if (nb_ctrl==7){
printf("Le fichier contient %d pts de controle, les 11 parametres standards et
les distorsions radiales\npeuvent être calculés\n",nb_ctrl);
nbparam=14;
}
if (nb_ctrl>=8){
printf("Le fichier contient %d pts de controle, les 11 parametres standards,\nles
distorsions radiales et le decentrement peuvent être calculés\n",nb_ctrl);
nbparam=16;
}
if (nbparam>=14){ //l'utilisateur peut choisir ou on de calculer les param de disto
disto:printf("\nVoulez vous calculer les parametres des erreurs
optiques?(oui/non) ");
scanf("%s",disto);
reponse=test_oui_non(disto);
if(reponse==1)goto disto;
if(strcmp(disto,"non")==0)nbparam=11;
}

//calcul du plus petit nombre de paramètres

```

```

if (nbparam < minparam){
minparam=nbparam;
}
if (nb_check!=0){
if(i==0) CAL_check=CreateMat((2*nb_check)+1,n);
ValAssign(CAL_check,0,i,ValRead(CHECK,(i*nb_check),0));
for(j=0; j<nb_check; j++){
o=2*j;
p=2*j+1;
ValAssign(CAL_check,o+1,i,ValRead(CHECK,j+(i*nb_check),2));
ValAssign(CAL_check,p+1,i,ValRead(CHECK,j+(i*nb_check),3));
}
}

//calcul des paramètres à faire L1-L11 ou L16...

L=calcul_param(D,uo,vo,nbparam,nb_ctrl);
if(ValRead(L,1,2)==0){
total_check=total_check-nb_check;
goto uncalage;
}
UoVo=calcul_uovo(L);
//retour à l'écran des residus sur chaque points ctrl
Resid=calcul_resid(L,D,nbparam,nb_ctrl);
printf("\n residus U\t residus V (unties image)\n -----\t -----\n");
ViewMat(Resid);
//calcul de la RMS total et de l'erreur probable sur 1 mesure
double rms = calcul_rms(Resid, nb_ctrl);
printf("\n\n**** RMS = %lf ****\n\n",rms);
printf("**** Erreur probable = %lf ****\n\n",0.67*rms);

resid:printf("\nVoulez vous modifier vos points de calage ('ctrl')?(oui/non) ");
scanf("%s",resid);
reponse=test_oui_non(resid);
if(reponse==1)goto resid;
if(strcmp(resid,"oui")==0){
total_check=total_check-nb_check;
goto uncalage;
}

//calcul de la position de la camera

POS=position_camera(L,0);
for(j=0;j<3;j++){
ValAssign(POSmem,0,i,cliche);
valP=ValRead(POS,j,0);
ValAssign(POSmem,j+1,i,valP);
}

//écriture dans le fichier texte paramètres L + uo et vo

strcpy(path, rep);
strcat(clichecrire, ".Lparam");
strcat(path, clichecrire);
sortie = fopen (path, "w");//creation du fichier *.Lparam
if (sortie == NULL){
perror ("\n>>LE FICHER N'A PAS PU ETRE OUVERT<<");
goto uncalage;
}
else{//écriture des paramètres dans un fichier
int ecrire= ecrire_param(sortie ,L ,POSmem ,UoVo ,nbparam,cliche, i);
fclose(sortie);
}

```



```

//écriture des paramètres dans un tableau commun

ValAssign(Lmem,0,i,cliche);//ligne numero cliche
ValAssign(Lmem,1,i,ValRead(UoVo,0,0));//ligne uo
ValAssign(Lmem,2,i, ValRead(UoVo,1,0));//ligne vo
for(j=0; j<nbparam; j++){//écriture des paramètres calculés
valL=ValRead(L,j,0);
ValAssign(Lmem,j+3,i,valL);
}

};//ferme le for du cliché dont les paramètres viennent d'etre calculés

if (XO1==0) POSdelta=SubMat(POSmem,POSo);
if (n>1 && nb_check!=0 && strcmp(avert,"non")!=0) {//calcul des résidu sur les
points check
RESULTAT_check=calcul_XYZ(Lmem, CAL_check, nb_check, n, minparam);
RESIDU_check=CreateMat(4,nb_check);
for(i=0; i<nb_check; i++){
ValAssign(RESIDU_check,0,i,ValRead(CHECK,i,1));
ValAssign(RESIDU_check,1,i,ValRead(RESULTAT_check,0,i)-ValRead(CHECK,i,4));
ValAssign(RESIDU_check,2,i,ValRead(RESULTAT_check,1,i)-ValRead(CHECK,i,5));
ValAssign(RESIDU_check,3,i,ValRead(RESULTAT_check,2,i)-ValRead(CHECK,i,6));
}
}

////////////////////////////////////
//*****
//* module de calcul de coordonnées tridimensionnelles */
//*****
////////////////////////////////////

printf("\n ****BIENVENUE: Calcul des coordonnees tridimensionnelles****\n\n");
if (n!=0){//le calcul de paramètres a été effectué n fois (=/ 0)
tridim:printf("\nVoulez vous travaillez avec les parametres calculés?(oui/non)
");
scanf("%s",paramm);
reponse=test_oui_non(paramm);
if(reponse==1) goto tridim;
if(strcmp(paramm,"oui")==0) ntotal=n;
}

else{//pas de calcul de paramètres n=0
tridim2:printf("Aucun parametres n'ont ete calculés, voulez vous le
faire?(oui/non) ");
scanf("%s",aucun);
reponse=test_oui_non(aucun);
if(reponse==1) goto tridim2;
if(strcmp(aucun,"oui")==0) goto debut;
}

//ouverture de fichiers paramètres supplémentaires (on en a déjà calculer)

tridim3:printf("\nVoulez vous travaillez avec d'autres clichés déjà
traités?(oui/non) ");
scanf("%s",autreparam);
reponse=test_oui_non(autreparam);
if(reponse==1)goto tridim3;
if(strcmp(autreparam,"oui")==0){
erreur:printf("\nAvec combien d'autres clichés voulez vous travailler: ");
scanf("%d",&nbrautreparam);

//on sait avec combein de cliché on va travailler=> on les écrit dans le tableau
final

```

```

if(strcmp(paramm,"non")==0){//travaillez avec les param calculé :non...
n=0;
goto newparam;
}
ntotal=n+nbrautreparam;

if ((ntotal)<=1){//pas assez de param charger
printf("\n>>CHOISISSEZ AU MOINS %d AUTRES CLICHES!<<\n", (2-(ntotal)));
goto erreur;
}
else{
printf("\n**Vous avez choisi de travailler avec %d images**\n ",(ntotal));
goto param;
}
}
else {
if(strcmp(paramm,"non")==0) n=0;

else{
if(n<2){
printf("\n>>CHOISISSEZ AU MOINS %d AUTRES CLICHES!<<\n", (2-n));
goto erreur;
}
//si on ne travaille pas avec d'autres fichiers param et qu'on a calculé au moins
2 x.Lparam
else goto uncalcul;
}
}

//si on n'a calculé aucun paramètres de la DLT, on arrive ici (oui a la première
question)

newparam:printf("\n   ****BIENVENUE: Calcul des coordonnees
tridimensionnelles****\n\n");
newparam2:printf("Combien de fichier de parametres voulez vous chargez : ");
scanf("%d",&nbrparam);

if (nbrparam<=1){
printf("\n>>CHOISISSEZ AU MOINS 2 CLICHES!<<\n\n");
goto newparam2;
}
else {
printf("\n**Vous avez choisi de travailler avec %d images**\n",nbrparam);
nbrautreparam=nbrparam;
ntotal=nbrautreparam;
goto param;
}
}

//ouverture des (nouveaux) fichiers paramètres (on en a pas calculé)

param:for(int i=0; i<nbrautreparam; i++){
//demande de nom du fichier et numero de cliché
unparam:if(i==0)printf("\n>>>nom du %der fichier de parametres et le numero du
cliche: ",(i+1));
else printf("\n>>>nom du %deme fichier de parametres et le numero du cliche:
", (i+1));
strcpy(path,rep);
scanf("%20s %d",nomfichparam,&numcliche);
strcat(nomfichparam, ".Lparam");
strcat(path,nomfichparam);
entree = fopen (path, "r");//ouverture du fichier
if (entree == NULL){
perror ("\n>>>LE FICHIER N'A PAS PU ETRE OUVERT<<");
goto unparam;
}
}

```

```

else {
cptp=compte_lignes(entree)-3;
fseek(entree, 0, SEEK_SET);

//parcours le fichier contenant les paramètres

fgets(lire,100,entree);
sscanf(lire,"%s %s %d", str1, str2, &pcliche);
if(numcliche!=pcliche){//test sur le numero du cliché dans le fichier et
introduit par l'utilisateur
printf("\n>>LE NUMERO DU CLICHE NE CORRESPOND PAS, veuillez verifier...<<\n");
goto unparam;
}
if(n!=0 && i==0){//les param sont déjà utilisés?
for(j=0; j<n; j++){
if (pcliche == int(ValRead(Lmem,0,j))){
printf("\n>>LES PARAMETRES DE CE CLICHE SONT DEJA UTILISES, veuillez
verifier...<<\n");
goto unparam;
}
}
}
if(i!=0){
for(j=0; j<n+nbrautreparam; j++){
if (pcliche == int(ValRead(Lfinal,0,j))){
printf("\n>>LES PARAMETRES DE CE CLICHE SONT DEJA UTILISES, veuillez
verifier...<<\n");
goto unparam;
}
}
}
ValAssign(LP,0,0,pcliche);
fgets(lire,100,entree);
sscanf(lire,"%s %s %lf %lf", str1, str2, &uo, &vo);
ValAssign(LP,1,0,uo);
ValAssign(LP,2,0,vo);
fgets(lire,100,entree);
sscanf(lire,"%s %s %lf %lf %lf", str1, str2, &xp, &yp, &zp);

if(strcmp(str1,"Point")==0){//la position de la camera est elle dans le fichier
XO2=1;
}
else{//Si oui copie dans un tableau de X Y Z
ValAssign(POSp,0,0,pcliche);
ValAssign(POSp,1,0,xp);
ValAssign(POSp,2,0,yp);
ValAssign(POSp,3,0,zp);
}

nb=3;
while((fgets(lire,100,entree))!=NULL){//copie des param dans un tableau
sscanf(lire,"%s %lf",str1, &l);
ValAssign(LP,nb,0,l);
nb++;
}
fclose(entree);
}
//vérification du nombre de param dans le fichier charger
if (cptp<11){
printf(">>ATTENTION VOTRE FICHIER NE CONTIENT PAS ASSEZ DE PARAMETRES<<\n\n");
printf(">>veuillez le verifier et le charger a nouveau<<\n");
goto unparam;
}
if (cptp>16){

```

```

printf(">>ATTENTION VOTRE FICHER CONTIENT TROP DE PARAMETRES<<\n\n");
printf(">>Veuillez le vérifier et le charger a nouveau<<\n");
goto unparam;
}

//calcul ou ecriture de position camera dans le fichier de param si il n'y est pas

if (XO2==1){
POS=position_camera(LP,3);//calcul de la position camera
ValAssign(POSp,0,0,pcliche);
for(j=0;j<3;j++){
valP=ValRead(POS,j,0);
ValAssign(POSp,j+1,0,valP);
}

paramdlt:printf("\nVoulez vous inscrire la position de la camera dans le
fichier?(oui/non) ");
scanf("%s",cam);//choix si l'utilisateur veut inscrire la postion dans le fichier

reponse=test_oui_non(cam);
if(reponse==1)goto paramdlt;

if(strcmp(cam,"oui")==0){
sortie = fopen (path,"w");
if (sortie == NULL){
perror ("\n>>LE FICHER N'A PAS PU ETRE OUVERT<<");
goto paramdlt;
}
else{//écriture de la positon dans le fichier de parmaètres
int ecriture= ecrire_param2(sortie ,LP ,POSp ,cctp, i);
fclose(sortie);
}
}
}

//inscription à l'ecran du nombre de param dans le fichier
if (cctp==11)printf("Le fichier contient les 11 parametres standards\n");
if (cctp<16 && cctp>11)printf("Le fichier contient les 11 parametres standards et
une partie des erreurs de la lentille\n");
if (cctp==16)printf("Le fichier contient les 16 parmetres standards et
distorsions radiales peuvent etre calcules\n");

//calcul du plus petit nombre de paramètres

if (cctp < minparam) minparam=cctp;

// ecriture des paramètres et de la position des points de vue dans les tableaux
finaux

uncalcul:if(i==0 || strcmp(autreparam,"non")==0){//param charger d'un fichier=non
Lffinal=CreateMat(19,ntotal);
POSfinal=CreateMat(4,ntotal);
}

if(strcmp(paramm,"oui")==0){//travaillez avec les param calculé :oui...

for (j=0; j<n; j++){//écriture des paramètre dans le tableau finale des param
for(k=0; k<19; k++){
valL=ValRead(Lmem,k,j);
ValAssign(Lffinal,k,j,valL);
valP=ValRead(POSmem,k,j);
ValAssign(POSfinal,k,j,valP);
}
}
}
}

```

```

if (strcmp(autreparam,"oui")==0 || strcmp(debut,"oui")==0){//param charger d'un
fichier=oui
for(j=0; j<19; j++){//écriture des paramètre dans le tableau finale des param
valL=ValRead(LP,j,0);
ValAssign(Lfinal,j,i+n,valL);
valP=ValRead(POSp,j,0);
ValAssign(POSfinal,j,i+n,valP);
}
}
} //ferme le for nbrautreapram

printf("\nL'ensemble des fichiers qui vous avez charger ou calculer
permettent\nde travailler avec %d parametres\n",minparam);

//on a: -ntotal: nbr de cliché
//-minparam: nbr de paramètres
//-Lfinal: contient num cliché, Uo Vo et les paramètres
//-POSfinal: contient num cliché et position des points vue

//ouverture du fichier des points en calculer (u,v)

uv:printf("\n>>>Nom du fichier de coordonnees image (U, V): "); //ouverture du
fichier contenant les points à calculer
scanf("%20s", nomfichcal);
strcpy(path, rep);
strcat(nomfichcal, ".txt");
strcat(path, nomfichcal);
entree = fopen (path, "r");
if (entree == NULL){
perror ("LE FICHIER N'A PAS PU ETRE OUVERT");
goto uv;
}
else{
cptc=(compte_lignes(entree)/ntotal)-1;
if (cptc<1){ //test: nombre point a calculer > 1
printf(">>ATTENTION VOTRE FICHIER CONTIENT MOINS D'UNE COORDONNEE<<\n\n");
printf(">>Veuillez le verifier et le charger a nouveau<<\n");
goto uncalcul;
}

CAL=CreateMat(2*cptc+1,ntotal);
ID=CreateMat(cptc,1);
fseek(entree, 0, SEEK_SET);

//parcours le fichier texte

fgets(lire,100,entree);
sscanf(lire,"%s %s %s", str1, str2, str3);

nb=0;
while(fgets(lire,100,entree)!=NULL){
o=2*nb;
p=2*nb+1;
sscanf(lire,"%s %d ", str1, &numcliche); //écriture du numéro du cliché dans le
tableau
ValAssign(CAL,0,nb,numcliche);
for(i=0; i<cptc; i++){ //remplissage du tableau contenant les points à calculer
o=2*i;
p=2*i+1;
fgets(lire,100,entree);
sscanf(lire,"%d' %lf %lf", &idrest, &u, &v); // id_point ui vi
ValAssign(CAL,o+1,nb,u);
ValAssign(CAL,p+1,nb,v);
if(nb==0) ValAssign(ID,i,0,idrest);
}
}

```

```

nb++;
}
fclose(entree);
for(i=0; i<ntotal; i++){//vérification que tous les points à restituer
correspondent à 1 set de param
test2=0;
testcal=int(ValRead(Lfinal,0,i));
for(j=0; j<ntotal; j++){
if(testcal==int(ValRead(CAL,0,j)))test2=1;
}

if(test2==0){//si problème:message d'erreur à l'écran
printf("IL Y A UNE ERREUR DANS LA NUMEROTATION DES CLICHE\n");
printf("\nTapez 1:pour retourner au debut de l'application\nTapez 2:pour charger
le fichier de points a calculer : ");
scanf("%d",&test3);
if(test3==1)goto para;
else goto uv;
}
}

////////////////////////////////////
//*****
//* calcul des coordonnées X, Y, Z *//
//*****
////////////////////////////////////

RESULTAT=calcul_XYZ(Lfinal, CAL, cptc, ntotal, minparam);
printf("\n**Vous avez calcule les coordonnees de %d points**\n", cptc);
resultat:printf("\n>>>Nom du fichier des resultats: ");//choix du nom du fichier
de résultats
scanf("%20s", resultat);
strcpy(path, rep);
strcat(resultat, ".dlt");
strcat(path, resultat);
sortie = fopen(path, "w");//création du fichier de résultats
if (sortie == NULL){
perror ("LE FICHER N'A PAS PU ETRE OUVERT");
goto resultat;
}
else{

//écriture dans le fichier texte

fprintf(sortie,"Résultats\n\nRestitution (X,Y,Z)");//écriture des résultats de la
restitution
for(i=0; i<cptc; i++){
fprintf(sortie,"\nID '%d'\t%8.3lf %8.3lf
%8.3lf",int(ValRead(ID,i,0)),ValRead(RESULTAT,0,i),ValRead(RESULTAT,1,i),ValRead(
RESULTAT,2,i));
}

fprintf(sortie,"\n\nPosition des points de vues (X,Y,Z)");
for(i=0; i<ntotal; i++){
fprintf(sortie,"\nCliché '%d'\t%8.3lf %8.3lf
%8.3lf",int(ValRead(POSfinal,0,i)),ValRead(POSfinal,1,i),ValRead(POSfinal,2,i),Va
lRead(POSfinal,3,i));
}
}
fclose(sortie);
if((n>1 && nb_check!=0 && strcmp(avert,"non")==0)||X01==0){
residu:printf("\n>>>Nom du fichier des residus : ");//choix du nom du fichier de
résidus

```

```

scanf("%20s", residu);
strcpy(path, rep);
strcat(residu, ".dlt");
strcat(path, residu);
sortie = fopen (path, "w");//création du fichier de résidu
if (sortie == NULL){
    perror ("LE FICHIER N'A PAS PU ETRE OUVERT");
    goto residu;
}
else{

//écriture dans le fichier texte des résidus

fprintf(sortie,"Résidus");
if (n>1 && nb_check!=0 && strcmp(avert,"non")!=0){//résidu sur les points de
contrôle
fprintf(sortie,"\n\nPosition des points de contrôle (dX,dY,dZ)");
for(i=0; i<nb_check; i++) fprintf(sortie,"\nID '%d'\t%8.3lf %8.3lf
%8.3lf",int(ValRead(RESIDU_check,0,i)),ValRead(RESIDU_check,1,i),ValRead(RESIDU_c
heck,2,i),ValRead(RESIDU_check,3,i));
}
if (XO1==0){//résidu sur la position des points de vue
fprintf(sortie,"\n\nPosition des points de vues (dX,dY,dZ)");
for(i=0; i<nbpos; i++) fprintf(sortie,"\nCliché '%d'\t%8.3lf %8.3lf
%8.3lf",int(ValRead(POSdelta,0,i)),ValRead(POSdelta,1,i),ValRead(POSdelta,2,i),Va
lRead(POSdelta,3,i));
}
}
fclose(sortie);
}
}

```

LIBRAIRIE DE CALCUL MATRICIELLE

```

// *****
// ***** INCLUDE LIB TYPEDEF *****
// *****

#include "stdafx.h"

// système
#include <stdio.h>
#include <iostream>
#include <stdlib.h>
#include <math.h>
#include <iomanip>
#include <fstream>
#include <string>
using namespace std;

// persos
#include "UtilLib.h"
#include "MatrixLib.h"

// *****
// ***** MATRIX LIB *****
// *****

```

```

// *****                                CREATEMAT                                *****

ptr_to_mat CreateMat(int l, int c) {

matrix *m;
long double *d;

m=(matrix*)malloc(sizeof(matrix));          //vérifier si le malloc a réussi !
if (m==0) {cout<<"***manque de memoire***\n";
exit(1);
}
if (m!=0) {
//cout<<"***adresse de m***\n"<<m<<"\n";

}

m->nbl=l;
m->ncb=c;
//cout<<"matrice de l: "<<m->nbl<< "et de c:"<<m->ncb<<"\n";

d=(long double*)malloc(sizeof(long double)*l*c);      //vérifier si le malloc a
réussi !
m->data=d;
return(m); //on rend l'adresse du bloc de données
//réservé pour la matrice
}

// *****                                FREEMAT                                *****

void FreeMat(matrix *m) {
free(m->data); //libérer m en premier conduirait à perdre
free(m); //l'adresse des données (data)
}

// *****                                VALASSIGN                                *****

int ValAssign(matrix *m, int l, int c, long double val) {
//quelques tests
if (m==0) {return(-1);}
if ((l<0)|| (l>=m->nbl)) {return(-1);}
if ((c<0)|| (c>=m->ncb)) {return(-1);}
if (m->data==0) {return(-1);}
m->data[l*(m->ncb)+c]=val;
return(0);
}

// *****                                VALREAD                                *****

long double ValRead(matrix *m, int l, int c) {
//tests à ajouter
return(m->data[l*(m->ncb)+c]);
}

// *****                                VIEWMAT                                *****

void ViewMat(matrix *m) {

//on va essayer d'afficher la matrice en récupérant ses arguments dimx et dimy

```



```

int l, c;
for(l=0;l<m->nbl;l++)
{
for (c=0;c<m->nbc;c++)
{
cout << setw(10) << ValRead(m,l,c)<<"\t";
}
cout <<"\n";
}
}

// *****                                SUMMAT                                *****

matrix* SumMat(matrix *a, matrix *b) {

int i,j;
matrix *result;

//tests
if ((a==0)||(b==0)) {return((matrix*)0);} //convertit 0 en un type matrix
if ((a->data==0)||(b->data==0)) {return((matrix*)0);}
if ((a->nbl!=b->nbl)||(a->nbc!=b->nbc)) {return((matrix*)0);}

// create result matrix

result=CreateMat(a->nbl,a->nbc);

if(result==0) return ((matrix*)0);

// do the sum
for(i=0;i<a->nbl;i++)
{
for(j=0;j<a->nbc;j++)
{//le résultat vaut : result(i,j)=a(i,j)+b(i,j)
ValAssign(result, i, j, (ValRead(a,i,j)+ValRead(b,i,j)));
}
}
//return the result
return(result);
}

// *****                                SUBMAT                                *****

matrix* SubMat(matrix *a, matrix *b) {

int i,j;
matrix *result;

//tests
if ((a==0)||(b==0)) {return((matrix*)0);} //convertit 0 en un type matrix
if ((a->data==0)||(b->data==0)) {return((matrix*)0);}
if ((a->nbl!=b->nbl)||(a->nbc!=b->nbc)) {return((matrix*)0);}

// create result matrix

result=CreateMat(a->nbl,a->nbc);

```

```

if(result==0) return ((matrix*)0);

// do the sum
for(i=0;i<a->nbl;i++)
{
for(j=0;j<a->nbc;j++)
{//le résultat vaut : result(i,j)=a(i,j)+b(i,j)
ValAssign(result, i, j, (ValRead(a,i,j)-ValRead(b,i,j)));
}
}
//return the result
return(result);
}

// *****                                MULTMAT                                *****

matrix* MultMat(matrix *a, matrix *b) {

int L, C, K;
long double valeur;
matrix *result;

//tests
if ((a==0)|| (b==0)) {return((matrix*)0);} //convertit 0 en un type matrix
if ((a->data==0)|| (b->data==0)) {return((matrix*)0);}
if (a->nbc!=b->nbl) {return((matrix*)0);} //on vérifie que le ncl = nl2

// create result matrix

result=CreateMat(a->nbl,b->nbc);

if(result==0) return ((matrix*)0);

// do the product
// boucles générales
for(L=0;L<a->nbl;L++)
{
for(C=0;C<b->nbc;C++)
{//le résultat vaut :
//boucle locale - calcul d'un élément de la mat résultat
valeur=0;
for(K=0;K<b->nbl;K++) //ou K < a->nbc c'est pareil
{
valeur=valeur+ValRead(a,L,K)*ValRead(b,K,C);
}
ValAssign(result, L, C, valeur);
}
}
//return the result
return(result);
}

// *****                                TRANSPPOSEMAT                                *****

matrix* TransposeMat(matrix *a) {

int i,j;
matrix *result;

//tests

```

```

    if (a==0) {return((matrix*)0);} //convertit 0 en un type matrix
    if (a->data==0) {return((matrix*)0);}

// create result transposed matrix

result=CreateMat(a->nbc,a->nbl); //on inverse le nbr de ligne et de col de la
matrice orig

if(result==0) return ((matrix*)0);

// do the transposition
for(i=0;i<a->nbl;i++)
{
for(j=0;j<a->nbc;j++)
{//le résultat vaut :
ValAssign(result, j, i, ValRead(a,i,j));
}
}
//return the result
return(result);
}

// ***** SCALEMAT *****

matrix* ScaleMat (matrix *a, long double scal) {

int i,j;
matrix *result;

//tests
if (a==0) {return((matrix*)0);} //convertit 0 en un type matrix
if (a->data==0) {return((matrix*)0);}

// create result matrix

result=CreateMat(a->nbl,a->nbc);

if(result==0) return ((matrix*)0);

// do the scalproduct
for(i=0;i<a->nbl;i++)
{
for(j=0;j<a->nbc;j++)
{//le résultat vaut : result(i,j)=a(i,j)+b(i,j)
ValAssign(result, i, j, ValRead(a,i,j)*scal);
}
}
//return the result
return(result);
}

// ***** INVERSEJORDANMAT *****

matrix* InverseJordanMat(matrix* a)
{
int n=a->nbc;
int i,j,k,l,err;
long double max,pivot,coef;
matrix *temp;

```

```

matrix *result;

//float t[NMAX][N2MAX]; remplacé par
temp = CreateMat(n,2*n);//on crée une matrice avec le long double de lignes
result = CreateMat(n,n);//et la matrice résultat inversée

for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
{
//t[i][j]=a[i][j]; on balance le contenu de a dans temp
ValAssign(temp,i-1,j-1,ValRead(a,i-1,j-1)); //-1 pour respecter les boucles
//if (i==j) t[i][j+n]=1.0;else t[i][j+n]=0.0;
if (i==j) ValAssign(temp,i-1,j-1+n,1.0);
else ValAssign(temp,i-1,j-1+n,0.0);

}
err=1;
k=1;
while (err==1 && k<=n)
{
//max=fabs(t[k][k]);
max=fabs(ValRead(temp,k-1,k-1));
l=k;
for(i=k+1;i<=n;i++)
//if(max<fabs(t[i][k]))
if (max<fabs(ValRead(temp,i-1,k-1)))
{
max=fabs(ValRead(temp,i-1,k-1));
l=i;
}
if(max!=0)
{
for(j=k;j<=2*n;j++)
{
max=ValRead(temp,k-1,j-1);
//t[k][j]=t[l][j];
ValAssign(temp,k-1,j-1,ValRead(temp,l-1,j-1));
//t[l][j]=max;
ValAssign(temp,l-1,j-1,max);
}
pivot=ValRead(temp,k-1,k-1);

for(j=k+1;j<=2*n;j++)
//t[k][j] /= pivot; on divise l'élément par le pivot et on affecte cette
valeur
ValAssign(temp,k-1,j-1,ValRead(temp,k-1,j-1)/pivot);
for(i=1;i<=n;i++)
if(i!=k)
{
//coef=t[i][k];
coef=ValRead(temp,i-1,k-1);
for(j=k+1;j<=2*n;j++)
//t[i][j] -= coef*t[k][j];
ValAssign(temp,i-1,j-1,ValRead(temp,i-1,j-1)-coef*ValRead(temp,k-1,j-1));
}
}
else err=0;
k++;
}
for(i=1;i<=n;i++)
for(j=1;j<=n;j++)
//b[i][j]=t[i][j+n];
ValAssign(result,i-1,j-1,ValRead(temp,i-1,j-1+n));
return(result);
}

```

```
// copie d'une matrice dans une autre existante

void CopyMat(matrix* a, matrix* b) {
int l,c;
for (l=0;l<a->nbl;l++) {
for (c=0;c<a->nc;c++) {
ValAssign(b,l,c,ValRead(a,l,c));
}
}
}

//
// *****                               INVERSEQRMAT                               *****
//

matrix* InverseQRMat(matrix* MAT){//voir leastSquareQR

int nbrc=MAT->nc;
int nbrl=MAT->nbl;
int i,j,k;
long double norm;

ptr_to_mat A;
ptr_to_mat E1;
ptr_to_mat I;
ptr_to_mat Identity;
ptr_to_mat U;
ptr_to_mat U1;
ptr_to_mat V;
ptr_to_mat Q;
ptr_to_mat Qinitial;
ptr_to_mat Q1;
ptr_to_mat QA;
ptr_to_mat QAinitial;
ptr_to_mat Qfinal;
ptr_to_mat invR;
ptr_to_mat Result;
invR=CreateMat(nbrl,nbrc);

for(i=0;i<nbrl-1;i++){
I=CreateMat(nbrl-i,nbrl-i);
E1=CreateMat(nbrl-i,1);
U1=CreateMat(nbrl-i,1);
A=CreateMat(nbrl-i,nbrl-i);
U=CreateMat(nbrl-i,1);
V=CreateMat(nbrl-i,1);

for(j=0;j<nbrl-i;j++){
for(k=0;k<nbrl-i;k++){
if (j==k) ValAssign(I,k,j,1);
else ValAssign(I,k,j,0);
}
}
if(i==0){
Identity=CloneMat(I);
A=CloneMat(MAT);
}
else{
for(j=0;j<nbrl-1;j++){
for(k=0;k<nbrc-1;k++) ValAssign(A,j,k,ValRead(QA,j+i,k+i));
}
}
ValAssign(E1,0,0,1);
for(j=0;j<nbrl-i;j++){
```

```

if (j!=0) ValAssign(E1,j,0,0);
ValAssign(U1,j,0,ValRead(A,j,0));
}
norm=0;
for(j=0;j<nbrl-i;j++) norm=norm+pow(ValRead(A,j,0),2);
norm=sqrt(norm);
U=SubMat(U1,ScaleMat(E1,norm));

norm=0;
for(j=0;j<nbrl-i;j++) norm=norm+pow(ValRead(U,j,0),2);
norm=sqrt(norm);
V=ScaleMat(U,1/norm);

Q1=ScaleMat(MultMat(V,TransposeMat(V)),2);
if(i==0) Q=SubMat(Identity,Q1);
else{
Q=CloneMat(Identity);
for(j=0+i;j<nbrl;j++){
for(k=0+i;k<nbrl;k++){
if (j==k) ValAssign(Q,j,k,1-ValRead(Q1,j-i,k-i));
else ValAssign(Q,j,k,-ValRead(Q1,j-i,k-i));
}
}
}
if (i==0) {
Qinitial=CloneMat(Q);
QA=MultMat(Qinitial,MAT);
QAinitial=CloneMat(QA);
}
else{
Qfinal=MultMat(Qinitial,Q);
Qinitial=CloneMat(Qfinal);
QA=MultMat(Q,QAinitial);
QAinitial=CloneMat(QA);
}
}

//oooooooooooooooooooooooooooo//
// inversion de R (QA) //
//oooooooooooooooooooooooooooo//

invR=CreateMat(nbrl,nbrl);
invR=ScaleMat(invR,0);
long double somme;
//construction de la mat inverse
for(i=nbrl-1;i>=0;i--){
for(j=nbrl-1;j>=i;j--){
if(i==j)ValAssign(invR,i,j,1/ValRead(QA,i,i)); //diagonale

else {
somme=0;
for(k=i+1;k<nbrl;k++)somme=somme+(ValRead(QA,i,k)*ValRead(invR,k,j));
ValAssign(invR,i,j,-somme/ValRead(QA,i,i));
}

}
}
Result=MultMat(invR,TransposeMat(Qfinal)); //produit des deux matrices inversées
return(Result); //retourne la matrice inversée
}

// //
// ***** INVERSEGREVILLEMAT *****
// *****//
// //

```

```

matrix* InverseGrevilleMat(matrix* a){
int n=a->nbc;
matrix *al;
matrix *Aplus;
matrix *Afplus;
matrix *DENO;
matrix *Ak;
matrix *ak;
matrix *Bk;
matrix *Dk;
matrix *Ck;
//creation des matrice
al=CreateMat(n,1);
ak=CreateMat(n,1);

int i,j,k;
double deno;

//*****//
// k = 1 //
//*****//
for(i=0;i<n;i++) ValAssign(al,i,0,ValRead(a,i,0));
DENO=MultMat(TransposeMat(al),al);
deno=1/ValRead(DENO,0,0);
Aplus=ScaleMat(TransposeMat(al),deno);

//*****//
// k >= 2 //
//*****//
for(i=1;i<n;i++){
Afplus=CreateMat(i+1,n);
Ak=CreateMat(n,i);
for(j=0;j<n;j++){
for(k=0;k<i;k++) ValAssign(Ak,j,k,ValRead(a,j,k));
ValAssign(ak,j,0,ValRead(a,j,i));
}
//calcul de param
Dk=MultMat(Aplus,ak);
Ck=SubMat(ak,MultMat(Ak,Dk));
DENO=MultMat(TransposeMat(Ck),Ck);
deno=1/ValRead(DENO,0,0);
Bk=ScaleMat(TransposeMat(Ck),deno);

for(j=0;j<n;j++){//creation de la mat inverse etape i
for(k=0;k<i;k++) ValAssign(Afplus,k,j,ValRead(Aplus,k,j)-
ValRead(MultMat(Dk,Bk),k,j));
ValAssign(Afplus,i,j,ValRead(Bk,0,j));
} //on copie la matrice final dans initial
Aplus=CloneMat(Afplus);
}
return(Aplus); //retourne le matrice inversée
}

//
// ***** LeastSquareQR ***** //
//
matrix* LeastSquareQR(matrix* COEF, matrix* B){

int nbrc=COEF->nbc;
int nbl=COEF->nbl;
int i,j,k,nbiter;
long double b;

```

```

long double norm;
//pointeur vers les matrices
ptr_to_mat A;
ptr_to_mat E1;
ptr_to_mat I;
ptr_to_mat Identity;
ptr_to_mat U;
ptr_to_mat U1;
ptr_to_mat V;
ptr_to_mat Q;
ptr_to_mat Qinitial;
ptr_to_mat Q1;
ptr_to_mat QA;
ptr_to_mat QAinitial;
ptr_to_mat Qfinal;
ptr_to_mat invR;
ptr_to_mat Result;
ptr_to_mat C;

invR=CreateMat(nbrl,nbrc);
//determination du nbr d'iteration
if (nbrc==nbrl) nbiter=nbrc-1;
else nbiter=nbrc;

for(i=0;i<nbiter;i++){//debut de process iteratif
I=CreateMat(nbrl-i,nbrl-i);//creation des matrices
E1=CreateMat(nbrl-i,1);
U1=CreateMat(nbrl-i,1);
A=CreateMat(nbrl-i,nbrl-i);
U=CreateMat(nbrl-i,1);
V=CreateMat(nbrl-i,1);
for(j=0;j<nbrl-i;j++){//creation matrice identité
for(k=0;k<nbrl-i;k++){
if (j==k) ValAssign(I,k,j,1);
else ValAssign(I,k,j,0);
}
}

if(i==0){//a l'étape 1
Identity=CloneMat(I);
A=CloneMat(COEF);
}
else{//autre étape
for(j=0;j<nbrl-1;j++){
for(k=0;k<nbrc-1;k++) ValAssign(A,j,k,ValRead(QA,j+1,k+i));
}
}
//remplissage du vecteur E1 (1,0,0,0,...)t
ValAssign(E1,0,0,1);
for(j=0;j<nbrl-i;j++){
if (j!=0) ValAssign(E1,j,0,0);
ValAssign(U1,j,0,ValRead(A,j,0));
}
norm=0;//calcul de la norme 1 ligne de QA
for(j=0;j<nbrl-i;j++) norm=norm+pow(ValRead(A,j,0),2);
norm=sqrt(norm);//norme de la
U=SubMat(U1,ScaleMat(E1,norm));

norm=0;//norme de U
for(j=0;j<nbrl-i;j++) norm=norm+pow(ValRead(U,j,0),2);
norm=sqrt(norm);
V=ScaleMat(U,1/norm);

Q1=ScaleMat(MultMat(V,TransposeMat(V)),2);
if(i==0) Q=SubMat(Identity,Q1);//a l'etape 1

```



```
matrix* calcul_param(matrix *D,long double uo,long double vo,int nbparam, int
nbpt){

ptr_to_mat C;//coefficients
ptr_to_mat L;
ptr_to_mat Linitial;//paramètres dlt
ptr_to_mat I;//terme independant
ptr_to_mat R;//valeur de R
ptr_to_mat zeta;//valeur de zeta
ptr_to_mat nu;//valeur de nu
ptr_to_mat r2;//valeur de r2
ptr_to_mat difL;

//matrix* least_square(matrix *C, matrix*I);

C=CreateMat((2*nbpt),nbparam);
L=CreateMat(nbparam, 1);
Linitial=CreateMat(nbparam, 1);
I=CreateMat((2*nbpt), 1);
R=CreateMat(nbpt, 1);
zeta=CreateMat(nbpt, 1);
nu=CreateMat(nbpt, 1);
r2=CreateMat(nbpt, 1);
difL=CreateMat(nbparam, 1);

int i,p,o,nb=0;
long double difi=10E10,diff;

uo=uo/2;
vo=vo/2;

for (i=0; i<nbparam; i++) ValAssign(L,i,0,0);
while (difi>1E-5){
difi=0;
Linitial=CloneMat(L);
nb++;
for(i=0; i<nbpt; i++){
o=2*i;
p=2*i+1;
//vecetur des termes indépendant

//1ere*i ligne
ValAssign(I,o,0,ValRead(D,i,0));
//2eme*i ligne
ValAssign(I,p,0,ValRead(D,i,1));

//parmètres standards

//L1
ValAssign(C,o,0,ValRead(D,i,2));
ValAssign(C,p,0,0);

//L2
ValAssign(C,o,1,ValRead(D,i,3));
ValAssign(C,p,1,0);

//L3
ValAssign(C,o,2,ValRead(D,i,4));
ValAssign(C,p,2,0);

//L4
ValAssign(C,o,3,1);
ValAssign(C,p,3,0);

//L5
```

```

ValAssign(C,o,4,0);
ValAssign(C,p,4,ValRead(D,i,2));

//L6
ValAssign(C,o,5,0);
ValAssign(C,p,5,ValRead(D,i,3));

//L7
ValAssign(C,o,6,0);
ValAssign(C,p,6,ValRead(D,i,4));

//L8
ValAssign(C,o,7,0);
ValAssign(C,p,7,1);

//L9
ValAssign(C,o,8,-(ValRead(D,i,0)*ValRead(D,i,2)));
ValAssign(C,p,8,-(ValRead(D,i,1)*ValRead(D,i,2)));

//L10
ValAssign(C,o,9,-(ValRead(D,i,0)*ValRead(D,i,3)));
ValAssign(C,p,9,-(ValRead(D,i,1)*ValRead(D,i,3)));

//L11
ValAssign(C,o,10,-(ValRead(D,i,0)*ValRead(D,i,4)));
ValAssign(C,p,10,-(ValRead(D,i,1)*ValRead(D,i,4)));

//distortions radiales

if(nbparam>11){

//vecteur R

ValAssign(R,i,0,(ValRead(L,8,0)*ValRead(D,i,2)+ValRead(L,9,0)*ValRead(D,i,3)+ValR
ead(L,10,0)*ValRead(D,i,4)+1));
//vecteur zeta
ValAssign(zeta,i,0,ValRead(D,i,0)-uo);
//vecteur nu
ValAssign(nu,i,0,ValRead(D,i,1)-vo);
//vecteur r2
ValAssign(r2,i,0,pow(ValRead(zeta,i,0),2)+pow(ValRead(nu,i,0),2));

//L12
ValAssign(C,o,11,(ValRead(zeta,i,0)*ValRead(r2,i,0)*ValRead(R,i,0)));
ValAssign(C,p,11,(ValRead(nu,i,0)*ValRead(r2,i,0)*ValRead(R,i,0)));

//L13
ValAssign(C,o,12,(ValRead(zeta,i,0)*pow(ValRead(r2,i,0),2)*ValRead(R,i,0)));
ValAssign(C,p,12,(ValRead(nu,i,0)*pow(ValRead(r2,i,0),2)*ValRead(R,i,0)));

//L14
ValAssign(C,o,13,(ValRead(zeta,i,0)*pow(ValRead(r2,i,0),3)*ValRead(R,i,0)));
ValAssign(C,p,13,(ValRead(nu,i,0)*pow(ValRead(r2,i,0),3)*ValRead(R,i,0)));

//decentrement lentille

if(nbparam>14){

//L15
ValAssign(C,o,14,(ValRead(r2,i,0)+2*pow(ValRead(zeta,i,0),2))*ValRead(R,i,0));
ValAssign(C,p,14,(ValRead(zeta,i,0)*ValRead(nu,i,0)*ValRead(R,i,0)));

//L16
ValAssign(C,o,15,(ValRead(nu,i,0)*ValRead(zeta,i,0)*ValRead(R,i,0)));
ValAssign(C,p,15,(ValRead(r2,i,0)+2*pow(ValRead(nu,i,0),2))*ValRead(R,i,0));

```

```

}
}
}

L=LeastSquareQR(C,I);

//calcul du point principal

uo=((ValRead(L,0,0)*ValRead(L,8,0))+ValRead(L,1,0)*ValRead(L,9,0))+ValRead(L,2,
0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+pow(ValRead(L,9,0),2)+pow(ValRead(L,1
0,0),2));

vo=((ValRead(L,4,0)*ValRead(L,8,0))+ValRead(L,5,0)*ValRead(L,9,0))+ValRead(L,6,
0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+pow(ValRead(L,9,0),2)+pow(ValRead(L,1
0,0),2));

difL=SubMat(Linitial,L);
for(i=0; i<nbparam; i++){
diff=fabs(ValRead(difL,i,0));
if(diff>difi) difi=diff;
}
ViewMat(L);
printf("\n-->%lf",difi);
if (nb==200) break;
}
if(nb==1 || nb==200){
for(i=0; i<nbparam; i++)ValAssign(L,i,0,0);
printf("\n>>LE CALCUL N'A PAS CONVERGER...veuillez verifier le fichier<<\n");
}
return(L);
}

```

RECONSTRUCTION TRIDIMENSIONNELLE

```

#include "stdafx.h"
#include "MatrixLib.h"
#include "MatrixLibInt.h"
#include "UtilLib.h"
#include <math.h>

matrix* calcul_XYZ(matrix *Lfinal, matrix *CAL, int cptc, int ntotal, int
minparam){

ptr_to_mat TABLE;// table de correspondance

ptr_to_mat R;//valeur de R
ptr_to_mat zeta;//valeur de zeta
ptr_to_mat nu;//valeur de nu
ptr_to_mat r2;//valeur de r2
ptr_to_mat deltaUV;
ptr_to_mat Upsilon;

ptr_to_mat C;//matrice des coefficients
ptr_to_mat X;//matrice des inconnues
ptr_to_mat I;//matrice des termes indépendants
ptr_to_mat Xinitial;//matrice des inconnues
ptr_to_mat difX;//matrice des inconnues

ptr_to_mat RESULT;

```

```

//creation des tableaux
TABLE=CreateMat(ntotal, 3);
R=CreateMat(cptc, ntotal);
zeta=CreateMat(cptc, ntotal);
nu=CreateMat(cptc, ntotal);
r2=CreateMat(cptc, ntotal);
deltaUV=CreateMat((2*cptc),ntotal);
Upsilon=CreateMat((2*cptc),ntotal);

C=CreateMat(2*ntotal,3);
X=CreateMat(3,1);
I=CreateMat(2*ntotal,1);

RESULT=CreateMat(3,cptc);

int nb;
int i,j,k,o,p,q,r;
int num, colonne;
long double difi,diff, valX;

//fonction
matrix* least_square(matrix *C, matrix*I);

//recherche des paramètres correspondants au point a restitué
//(les paramètres ne sont pas entrés dans le meme ordre)
//on crée une table de correspondance

for(i=0; i<ntotal; i++){
num=int (ValRead(CAL,0,i)); //on obtient le numero du cliche de la lere colonne
dans CAL
for(j=0; j<ntotal; j++){
if (ValRead(Lfinal,0,j)==num){
ValAssign (TABLE,i,0,num); //numero du cliche
ValAssign (TABLE,i,1,i); //numero colonne dans CAL
ValAssign (TABLE,i,2,j); //numero colonne dans Lfinal
}
}
}

for(i=0; i<ntotal; i++){ //pour tous les clichés
for(j=0; j<cptc; j++){ //pour tous les points
o=2*j;
p=2*j+1;
colonne= int (ValRead(TABLE,i,2)); //identification de la colonne des

//matrice zeta u-uo
ValAssign(zeta,j,i,(ValRead(CAL,o+1,i)-ValRead(Lfinal,1,colonne)));
//matrice nu v-vo
ValAssign(nu,j,i,(ValRead(CAL,p+1,i)-ValRead(Lfinal,2,colonne)));
//matrice r2
ValAssign(r2,j,i,(pow(ValRead(zeta,j,i),2)+pow(ValRead(nu,j,i),2)));
//matrice erreur de la lentille
//calcul des correction delta u et delta v
if (minparam==16){
ValAssign(deltaUV,o,i,((ValRead(zeta,j,i))*((ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i))+ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i),2))+ValRead(Lfinal,16,colonne)*pow(ValRead
(r2,j,i),3)))+(ValRead(Lfinal,17,colonne)*(ValRead (r2,j,i)+(2*pow(ValRead
(zeta,j,i),2)))+(ValRead(Lfinal,18,colonne)*ValRead (zeta,j,i)*ValRead
(nu,j,i)))));
ValAssign(deltaUV,p,i,((ValRead(nu,j,i))*((ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i))+ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i),2))+ValRead(Lfinal,16,colonne)*pow(ValRead
(r2,j,i),3)))+(ValRead(Lfinal,18,colonne)*(ValRead (r2,j,i)+(2*pow(ValRead

```

```

(nu,j,i,2)))+(ValRead(Lfinal,17,colonne)*ValRead (zeta,j,i)*ValRead
(nu,j,i)));
}
if(minparam==14){
ValAssign(deltaUV,o,i,(ValRead(zeta,j,i)*ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i)+(ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i,2)+(ValRead(Lfinal,16,colonne)*pow(ValRead (r2,j,i,3))));
ValAssign(deltaUV,p,i,(ValRead(nu,j,i)*ValRead(Lfinal,14,colonne)*ValRead
(r2,j,i)+(ValRead(Lfinal,15,colonne)*pow(ValRead
(r2,j,i,2)+(ValRead(Lfinal,16,colonne)*pow(ValRead (r2,j,i,3))));
}
}
else{
ValAssign(deltaUV,o,i,0);
ValAssign(deltaUV,p,i,0);
}
}
}
for(i=0; i<ntotal; i++){//pour tous les clichés
for(j=0; j<2*cptc; j++){//pour tous les points (2X pour u et v)
//matrice upsilon omega
ValAssign(Upsilon,j,i,ValRead(CAL,j+1,i)-ValRead(deltaUV,j,i));
}
}

//*****
//*Résolution*
//*****

for(i=0; i<cptc; i++){//pour tout les points
q=2*i;
r=2*i+1;
nb=0;
difi=10E10;
//valeur initial des coordonnees=1000
for(k=0; k<3; k++) ValAssign(X,k,0,1000);

while (difi>1E-5){//verif du critère d'arret et debut boucle
difi=0;
Xinitial=CloneMat(X);
nb++;

for(j=0; j<ntotal; j++){//remplissage des elements d'un colonne pour le ntotal
clichés
o=2*j;
p=2*j+1;
colonne= int (ValRead(TABLE,j,2));//numero col(Lfinal) du cliche A(col i(CAL et
Upsilon)

//matrice des R: 1 par point et par camera

ValAssign(R,i,j,((ValRead(Lfinal,11,colonne)*ValRead(X,0,0)+(ValRead(Lfinal,12,c
olonne)*ValRead(X,1,0)+(ValRead(Lfinal,13,colonne)*ValRead(X,2,0))+1));

//terme independants
ValAssign(I,o,0,(ValRead(Lfinal,6,colonne)-ValRead(Upsilon,q,j))/ValRead(R,i,j));
ValAssign(I,p,0,(ValRead(Lfinal,10,colonne)-
ValRead(Upsilon,r,j))/ValRead(R,i,j));

//coefficients
for(k=0; k<3; k++){//remplissage des trois elements d'une ligne des coefficients
ValAssign(C,o,k,(((ValRead(Lfinal,11+k,colonne)*ValRead(Upsilon,q,j))-
ValRead(Lfinal,3+k,colonne))/ValRead(R,i,j)));
ValAssign(C,p,k,(((ValRead(Lfinal,11+k,colonne)*ValRead(Upsilon,r,j))-
ValRead(Lfinal,7+k,colonne))/ValRead(R,i,j)));
}
}

```

```

}
X=least_square(C,I);//resolution par moindres carrés
difX=SubMat(Xinitial,X);
for(k=0; k<3; k++){//calcul du critère d'arret
diff=fabs(ValRead(difX,k,0));
if(diff>difi) difi=diff;
}
} //fin de la boucle pour un point
for(k=0; k<3; k++){//copie des coordonnées dans un tableau commmun
valX=ValRead(X,k,0);
ValAssign(RESULT,k,i,valX);
}
}
return(RESULT);
}

```

FONCTIONS DE CALCULS DIVERS

```

#include "stdafx.h"
#include "MatrixLib.h"
#include "UtilLib.h"
#include <math.h>

//*****
//*Moindres carrés*
//*****

matrix* least_square(matrix *C, matrix*I){//fonction des moindres carrés

ptr_to_mat Ct;
ptr_to_mat CtC;
ptr_to_mat invCtC;
ptr_to_mat CtI;
ptr_to_mat result;

Ct=TransposeMat(C);
CtC=MultMat(Ct,C);
invCtC=InverseJordanMat(CtC);
CtI=MultMat(Ct,I);
result=MultMat(invCtC,CtI);

return(result);
}

//*****
//*Calcul Uo Vo*
//*****

matrix* calcul_uovo(matrix *L){//fonction de calcul du point principal

ptr_to_mat UoVo;
UoVo=CreateMat(2,1);

//point principal

ValAssign(UoVo,0,0,((ValRead(L,0,0)*ValRead(L,8,0))+(ValRead(L,1,0)*ValRead(L,9,0)
))+(ValRead(L,2,0)*ValRead(L,10,0)))/(pow(ValRead(L,8,0),2)+pow(ValRead(L,9,0),2)
+pow(ValRead(L,10,0),2)));

ValAssign(UoVo,1,0,((ValRead(L,4,0)*ValRead(L,8,0)+ValRead(L,5,0)*ValRead(L,9,0)+

```

```

ValRead(L,6,0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+pow(ValRead(L,9,0),2)+pow
(ValRead(L,10,0),2));

return(UoVo);
}

//*****
//*nombre de ligne*
//*****

int compte_lignes(FILE *fichier){//calcul du nombre de lignes dans un fichier
int nb_lignes=0;
char c;
if (fichier==NULL) return 0; //si ps de fichier
else{
c=fgetc(fichier);
while (c!=EOF) {
c=fgetc(fichier);
if (c=='\n') nb_lignes++;
} // tant qu'on est pas a la fin du fichier
}
return nb_lignes+1;
}

//*****
//*Position camera*
//*****

matrix* position_camera(matrix *MATRICE,int k){//fonction calcul position camera
int i,j,l;
ptr_to_mat POS;
ptr_to_mat COEF;
ptr_to_mat invCOEF;
ptr_to_mat IND;
POS=CreateMat(3,1);
COEF=CreateMat(3,3);
IND=CreateMat(3,1);

//remplissage des matrices
l=k;
for(i=0;i<3;i++){
for(j=0;j<3;j++){
ValAssign(COEF,i,j,ValRead(MATRICE,j+1,0));
}
l+=4;
}
ValAssign(IND,0,0,-ValRead(MATRICE,3+k,0));
ValAssign(IND,1,0,-ValRead(MATRICE,7+k,0));
ValAssign(IND,2,0,-1);

//calcul

invCOEF=InverseJordanMat(COEF);
POS=MultMat(invCOEF,IND);

return (POS);
}

//*****
//*Résidu point de calage*
//*****

matrix* calcul_resid(matrix* L, matrix* D, int nbparam, int nb_ctrl){
//calcul des résidu sur les points de calage

```



```

ptr_to_mat C;//coefficients
ptr_to_mat R;//valeur de R
ptr_to_mat zeta;//valeur de zeta
ptr_to_mat nu;//valeur de nu
ptr_to_mat r2;//valeur de r2

ptr_to_mat CALCULE;
ptr_to_mat RESIDU;

C=CreateMat(2*nb_ctrl,nbparam);
R=CreateMat(nb_ctrl, 1);
zeta=CreateMat(nb_ctrl, 1);
nu=CreateMat(nb_ctrl, 1);
r2=CreateMat(nb_ctrl, 1);

int i,p,o,nb=0;
long double uo,vo;

for(i=0; i<nb_ctrl; i++){//meme fonction que calcul param

o=2*i;
p=2*i+1;

uo=((ValRead(L,0,0)*ValRead(L,8,0))+ValRead(L,1,0)*ValRead(L,9,0))+ValRead(L,2,
0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+pow(ValRead(L,9,0),2)+pow(ValRead(L,1
0,0),2));

vo=((ValRead(L,4,0)*ValRead(L,8,0))+ValRead(L,5,0)*ValRead(L,9,0))+ValRead(L,6,
0)*ValRead(L,10,0))/(pow(ValRead(L,8,0),2)+pow(ValRead(L,9,0),2)+pow(ValRead(L,1
0,0),2));

//vecteur R

ValAssign(R,i,0,(ValRead(L,8,0)*ValRead(D,i,2)+ValRead(L,9,0)*ValRead(D,i,3)+ValR
ead(L,10,0)*ValRead(D,i,4)+1));
//vecteur zeta
ValAssign(zeta,i,0,ValRead(D,i,0)-uo);
//vecteur nu
ValAssign(nu,i,0,ValRead(D,i,1)-vo);
//vecteur r2
ValAssign(r2,i,0,pow(ValRead(zeta,i,0),2)+pow(ValRead(nu,i,0),2));

//paramètres standards

//L1
ValAssign(C,o,0,(ValRead(D,i,2)/ValRead(R,i,0)));
ValAssign(C,p,0,0);

//L2
ValAssign(C,o,1,(ValRead(D,i,3)/ValRead(R,i,0)));
ValAssign(C,p,1,0);

//L3
ValAssign(C,o,2,(ValRead(D,i,4)/ValRead(R,i,0)));
ValAssign(C,p,2,0);

//L4
ValAssign(C,o,3,(1/ValRead(R,i,0)));
ValAssign(C,p,3,0);

//L5
ValAssign(C,o,4,0);
ValAssign(C,p,4,(ValRead(D,i,2)/ValRead(R,i,0)));

```

```

//L6
ValAssign(C,o,5,0);
ValAssign(C,p,5,(ValRead(D,i,3)/ValRead(R,i,0)));

//L7
ValAssign(C,o,6,0);
ValAssign(C,p,6,(ValRead(D,i,4)/ValRead(R,i,0)));

//L8
ValAssign(C,o,7,0);
ValAssign(C,p,7,(1/ValRead(R,i,0)));

//L9
ValAssign(C,o,8,(-(ValRead(D,i,0)*ValRead(D,i,2))/ValRead(R,i,0)));
ValAssign(C,p,8,(-(ValRead(D,i,1)*ValRead(D,i,2))/ValRead(R,i,0)));

//L10
ValAssign(C,o,9,(-(ValRead(D,i,0)*ValRead(D,i,3))/ValRead(R,i,0)));
ValAssign(C,p,9,(-(ValRead(D,i,1)*ValRead(D,i,3))/ValRead(R,i,0)));

//L11
ValAssign(C,o,10,(-(ValRead(D,i,0)*ValRead(D,i,4))/ValRead(R,i,0)));
ValAssign(C,p,10,(-(ValRead(D,i,1)*ValRead(D,i,4))/ValRead(R,i,0)));

//distortions radiales
if(nbparam>11){
//L12
ValAssign(C,o,11,(ValRead(zeta,i,0)*ValRead(r2,i,0)));
ValAssign(C,p,11,(ValRead(nu,i,0)*ValRead(r2,i,0)));

//L13
ValAssign(C,o,12,(ValRead(zeta,i,0)*pow(ValRead(r2,i,0),2)));
ValAssign(C,p,12,(ValRead(nu,i,0)*pow(ValRead(r2,i,0),2)));

//L14
ValAssign(C,o,13,(ValRead(zeta,i,0)*pow(ValRead(r2,i,0),3)));
ValAssign(C,p,13,(ValRead(nu,i,0)*pow(ValRead(r2,i,0),3)));

//decentrement lentille
if(nbparam>14){
//L15
ValAssign(C,o,14,(ValRead(r2,i,0)+2*pow(ValRead(zeta,i,0),2)));
ValAssign(C,p,14,(ValRead(zeta,i,0)*ValRead(nu,i,0)));

//L16
ValAssign(C,o,15,(ValRead(nu,i,0)*ValRead(zeta,i,0)));
ValAssign(C,p,15,(ValRead(r2,i,0)+(2*pow(ValRead(nu,i,0),2))));
}
}
}
CALCULE=MultMat(C,L);//terme indépendant selon le modèle
RESIDU=CreateMat(nb_ctrl,2);
for(i=0; i<nb_ctrl; i++){//différence entre les deux terme indépendant(modèle
réel)
ValAssign(RESIDU,i,0,ValRead(CALCULE,2*i,0)-ValRead(D,i,0)/ValRead(R,i,0));
ValAssign(RESIDU,i,1,ValRead(CALCULE,2*i+1,0)-ValRead(D,i,1)/ValRead(R,i,0));
}
return(RESIDU);
}

//*****
//*      RMS      *

```

```

//*****

double calcul_rms(matrix* Resid, int nb_ctrl){//calcul des résidus

int i;
double somme=0;

for(i=0;i<nb_ctrl;i++)
somme=somme+pow(ValRead(Resid,i,0),2)+pow(ValRead(Resid,i,1),2);
somme=sqrt(somme/(nb_ctrl-1));
return (somme);
}

```

AUTRES FONCTIONS

```

#include "stdafx.h"
#include <string.h>
#include "MatrixLib.h"
#include "UtilLib.h"
#include <math.h>

//*****
//*test oui non*
//*****

int test_oui_non(char reponse[5]){//test de la reponse si ->?(oui/non)

if((strcmp(reponse,"oui")!=0)&&(strcmp(reponse,"non")!=0)){
printf("\n>>VOTRE REPONSE N'EST PAS VALIDE!<<\n");//renvoi d'un message d'erreur
return 1;
}
else{
return 0;
}
}

//*****
//*Ecrire paramètre type 1*
//*****

int ecrire_param(FILE* sortie, matrix* L, matrix* POSmem, matrix* UoVo, int
nbparam, int cliche, int i){
long double valL;//fonction d'écriture des param dans un fichier après calcul
int j;

fprintf(sortie,"Paramètres clichés %d\n",cliche);
fprintf(sortie,"Point principal %lf %lf \n", ValRead(UoVo,0,0),
ValRead(UoVo,1,0));
fprintf(sortie,"Position camera %lf %lf %lf",
ValRead(POSmem,1,i),ValRead(POSmem,2,i),ValRead(POSmem,3,i));
for(j=0; j<nbparam; j++){
valL=ValRead(L,j,0);
fprintf(sortie,"\n'L%d'\t%e",j+1,valL);
}

return(0);
}

//*****
//*Ecrire paramètre type 2*

```

```
//*****  
  
int ecrire_param2(FILE* sortie, matrix* L, matrix* POSmem, int nbparam, int i){  
    long double valL;//fonction d'écriture des param dans un fichier si pas calculé  
    int j;  
  
    fprintf(sortie,"Paramètres clichés %d\n",ValRead(POSmem,0,i));  
    fprintf(sortie,"Point principal %lf %lf \n", ValRead(L,1,0), ValRead(L,2,0));  
    fprintf(sortie,"Position camera %lf %lf %lf",  
    ValRead(POSmem,1,i),ValRead(POSmem,2,i),ValRead(POSmem,3,i));  
    for(j=0; j<nbparam; j++){  
        valL=ValRead(L,j+3,0);  
        fprintf(sortie, "\n'L%d'\t%e", j+1, valL);  
    }  
  
    return(0);  
}
```

Annexe 8. Liste des images utilisées pour le test de la DLT (ancien commissariat de police)

Canon D70 18 mm image de droite : 181.bmp



Canon D70 18 mm image du milieu : 182.bmp



Canon D70 18 mm image de gauche : 183.bmp



Canon D70 35 mm image de droite : 351.bmp



Canon D70 35 mm image de gauche : 353.bmp



Sony DSC W12 image de droite : 501.bmp



Sony DSC W12 image de droite : 503.bmp



Annexe 9. Extrait du rapport d'erreur sur la compensation de la polygonale autour de l'ancien commissariat de police

OBSERVATIONS BRUTES REDUITES						
Station	Rep.	Point	AH	AV	Di (m)	Dh (m)
ST1	1	ST2	274.3570	101.4100	51.239	51.226
ST1	1	ST2	274.3575	101.4045	51.239	51.227
ST2	1	ST1	74.3555	98.5975	51.239	51.227
ST2	1	ST1	74.3560	98.5910	51.239	51.226
ST2	1	ST3	138.9350	103.4580	34.213	34.163
ST2	1	ST3	138.9355	103.4510	34.213	34.163
ST3	1	ST2	338.9365	96.5510	34.213	34.163
ST3	1	ST2	338.9370	96.5450	34.213	34.163
ST3	1	STM	43.4725	98.7235	11.822	11.820
ST3	1	STM	43.4745	98.7185	11.821	11.819
ST3	1	ST4	79.8280	97.5485	30.567	30.544
ST3	1	ST4	79.8280	97.5405	30.567	30.544
ST4	1	ST3	279.8260	102.4605	30.567	30.544
ST4	1	ST3	279.8275	102.4530	30.567	30.544
ST4	1	ST1	379.8790	96.3955	31.803	31.752
ST4	1	ST1	379.8760	96.3875	31.803	31.752
ST1	2	ST4	179.8835	103.6130	31.803	31.752
ST1	2	ST4	179.8815	103.6080	31.803	31.752
ST1	2	STC	279.6710	103.4995	23.406	23.371
ST1	2	STC	279.6715	103.4920	23.406	23.371
STM	1	ST4	98.9720	97.7995	21.575	21.562

ALTITUDES COMPENSEES (m.) ET EMQ (m.)			
Erreur moyenne quadratique générale 0.00630 Gr.			
Point	Z	EmqZ	fz
ST1	100.000		O
ST2	98.845	0.002	N
ST3	97.004	0.002	N
STM	97.246	0.002	N
ST4	98.179	0.001	N
STC	98.696	0.002	N

ELLIPSES DE CONFIANCE (Probabilité = 0.63)				
Point	X	Y	AxeX	AxeY
ST1	1000.000	1000.000		
ST2	952.873	979.921	0.002	0.001
ST3	980.844	960.304	0.002	0.001
STM	988.303	969.473	0.002	0.001
ST4	1009.867	969.820	0.002	0.001
STC	977.81	992.665	0.002	0.001

Annexe 10. Tableur Excel : analyse de la variance des 4 échantillons qui interviennent dans le test de la DLT

Analyse de la variance

Test de la DLT

DLT 3-18mm

<i>id</i>	<i>dx</i>	<i>dy</i>	<i>dz</i>	<i>distance</i>
26'	0,048	0,329	0,163	0,37028908
16'	-0,152	0,058	-0,012	0,16313185
33'	-0,001	-0,007	-0,001	0,00714143
94'	0,053	0,155	0,059	0,17411203
50'	0,006	0,014	0,002	0,01536229
53'	-0,138	-0,004	0,014	0,13876599
77'	-0,192	-0,006	0,107	0,21988406
74'	0,009	0,029	0,073	0,07906327
65'	0,094	0,369	0,14	0,40570556

DLT 35mm

<i>id</i>	<i>dx</i>	<i>dy</i>	<i>dz</i>	<i>distance</i>
99'	0,12	0,026	-0,111	0,16552039
16'	-0,19	0,006	0,007	0,19022355
33'	-0,011	-0,035	0,014	0,03926831
94'	-0,003	-0,01	0,024	0,0261725
50'	-0,001	-0,023	0,021	0,03116087
53'	-0,198	-0,004	0,059	0,2066422
77'	-0,339	-0,02	0,161	0,37582177
74'	-0,084	-0,073	0,11	0,15647684
65'	0,054	-0,036	0,055	0,08507056

DLT 18mm

<i>id</i>	<i>dx</i>	<i>dy</i>	<i>dz</i>	<i>distance</i>
26'	-0,039	0,063	0,093	0,11890753
16'	0,021	-0,023	-0,016	0,03501428
33'	-0,002	-0,011	0,004	0,01187434
94'	0,037	0,132	0,05	0,14592121
50'	0,022	0,022	0,003	0,031257
53'	0,038	-0,026	-0,015	0,0484252
77'	0,08	-0,041	0,025	0,09330595
74'	0,089	0,055	0,064	0,12264583
65'	0,048	0,294	0,076	0,30743455

DLT Sony

<i>id</i>	<i>dx</i>	<i>dy</i>	<i>dz</i>	<i>distance</i>
26'	0,054	0,188	0,046	0,2009378
16'	0,023	0,138	-0,119	0,18366818
33'	0,097	0,169	0,032	0,19746898
94'	0,04	0,168	0,049	0,17951323
50'	0,126	0,187	0,033	0,22789032
53'	0,138	0,136	-0,002	0,19376274
77'	0,081	0,076	0,142	0,18028034
74'	0,155	0,117	0,091	0,21446445
65'	0,01	0,167	0,037	0,17134176

Tables des données

n=	DLT 3-18mm	DLT 18mm	DLT 35 mm	DLT Sony
	9	9	9	9
	0,37028908	0,11890753	0,16552039	0,2009378
	0,16313185	0,03501428	0,19022355	0,18366818
	0,00714143	0,01187434	0,03926831	0,19746898
	0,17411203	0,14592121	0,0261725	0,17951323
	0,01536229	0,031257	0,03116087	0,22789032
	0,13876599	0,0484252	0,2066422	0,19376274
	0,21988406	0,09330595	0,37582177	0,18028034
	0,07906327	0,12264583	0,15647684	0,21446445
	0,40570556	0,30743455	0,08507056	0,17134176
moyenne	0,17482839	0,10164288	0,14181744	0,19436976
ecart type	0,14039956	0,09041815	0,11260291	0,01818165

n=	DLT 3-18mm	DLT 18mm	DLT 35 mm	DLT Sony	total
	9	9	9	9	36
Σobservations	1,57345555	0,91478589	1,27635699	1,74932781	5,51392625
Σobservations ²	0,432781	0,158385	0,282445	0,342661	1,216272

Sommes des carrés

totale	0,37173359
intra échantillons	0,32717972
inter échantillons	0,04455388

Carrés moyens

intra échantillons	0,01022437
inter échantillons	0,01485129

Rapport des carrés moyens

1,45253922

Tables d'analyse de la variance

Source de variabilité	Somme des carrés	Degrés de liberté	Carré moyen	F _{observé}
inter échantillons	0,04455388	3	0,01485129	1,45253922
intra échantillons	0,32717972	32	0,01022437	
total	0,37173359	35		

Annexe 11. Tableur Excel : calcul des écarts entre la restitution de mêmes points à partir de 2 couples stéréoscopiques différents

Validation compensation globale

Données : clichés 182, 182, 183

Couple 181-182

<i>id</i>	<i>x</i>	<i>z</i>	<i>z</i>
'26'	978,33	969,509	111,202
'97'	988,197	971,383	114,182
'33'	987,99	971,114	108,45
'94'	981,207	969,284	105,193
'50'	987,859	970,31	105,156
'53'	996,35	971,821	105,207
'74'	990,398	970,683	99,939
'65'	978,845	969,216	101,601

Couple 182-183

<i>id</i>	<i>x</i>	<i>z</i>	<i>z</i>
'26'	977,972	969,999	111,681
'97'	988,217	971,44	114,304
'33'	987,991	971,119	108,461
'94'	981,174	969,344	105,196
'50'	987,864	970,264	105,133
'53'	996,12	971,149	104,964
'74'	990,383	970,453	99,84
'65'	978,723	969,41	101,513

Ecart entre les deux restitution

<i>id</i>	<i>dx</i>	<i>dz</i>	<i>dz</i>	<i>distance</i>
'26'	0,358	-0,49	-0,479	0,77311383
'97'	-0,02	-0,057	-0,122	0,13613596
'33'	-0,001	-0,005	-0,011	0,01212436
'94'	0,033	-0,06	-0,003	0,06854196
'50'	-0,005	0,046	0,023	0,05167204
'53'	0,23	0,672	0,243	0,75068835
'74'	0,015	0,23	0,099	0,25085055
'65'	0,122	-0,194	0,088	0,24548727

Moyenne 0,0915 0,01775 -0,02025 0,28607679

Ecart type 0,13664866 0,33566342 0,21323947 0,30607108

Annexe 12. Liste des images utilisées pour la restitution de l'hôtel de ville

Canon IXUS 750 image 001 : hotel_ville 001.bmp



Canon IXUS 750 image 002 : hotel_ville 002.bmp



Canon IXUS 750 image 003 : hotel_ville 003.bmp



Canon IXUS 750 image 004 : hotel_ville 004.bmp



Canon IXUS 750 image 005 : hotel_ville 005.bmp



Canon IXUS 750 image 006 : hotel_ville 006.bmp



Canon IXUS 750 image 007 : hotel_ville 007.bmp



Canon IXUS 750 image 008 : hotel_ville 008.bmp



Canon IXUS 750 image 009 : hotel_ville 009.bmp



Canon IXUS 750 image 010 : hotel_ville 010.bmp



Annexe 13. Plan : vue perspective de l'hôtel de ville de Verviers

Annexe hors texte.

Annexe 14. CD de données