

國立交通大學

電子工程學系 電子研究所

碩士論文

使用景深感應器與RGB相機對用於影像合成的景



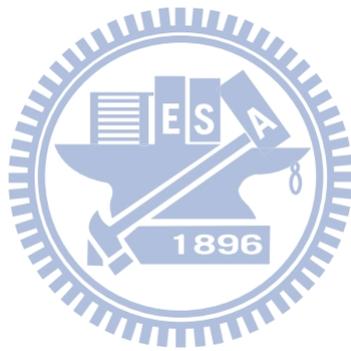
深改善

Depth Refinement for View Synthesis using Depth
Sensor and RGB Camera

研究生：邱義文

指導教授：杭學鳴 教授、蔡彰哲 博士

中華民國一〇一年七月



使用景深感應器與RGB相機對用於影像合成的景
深改善

Depth Refinement for View Synthesis using Depth
Sensor and RGB Camera

研究生：邱義文

Student：Yi-Wen Chiou

指導教授：杭學鳴

Advisor：Dr. Hsueh-Ming Hang

蔡彰哲

Dr. Jang-Jer Tsai



A Thesis

Submitted to Department of Electronics Engineering and
Institute of Electronics

College of Electrical and Computer Engineering
National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

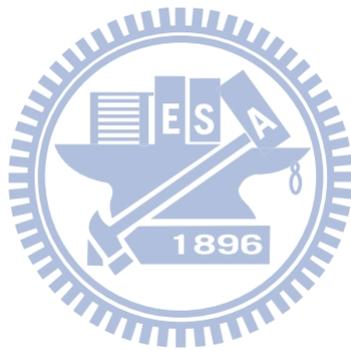
in

Electronics Engineering

July 2012

Hsinchu, Taiwan, Republic of China

中華民國一〇一年七月



使用景深感應器與 RGB 相機對用於影像合成的 景深改善

研究生：邱義文

指導教授：杭學鳴 博士、蔡彰哲 博士

國立交通大學

電子工程學系 電子研究所碩士班

摘要

近年來，自從電影阿凡達(Avatar)在2009年上映後，立體視訊就成為一股新的風潮，不管是電視、電影甚至是手機螢幕都有立體效果的產品出現。立體影像顯影技術中，以視點合成技術最為關鍵。現今視點合成的技術大多以既有的2維影像加上對應的景深資訊，來產生虛擬視點的影像。所以景深資訊在影像合成中扮演重要的角色。

一般常見取得景深資訊的方法，是以兩張二維不同視角的二維影像，用立體匹配(Stereo Matching)的方法，找出兩張影像中對應的方塊，再設法估測出景深影像，我們稱其為「被動式景深估測」。此方法在特徵點較少的區域，常估測出錯誤的景深資訊。並且因為視角的不同，造成遮蔽(Occlusion)區域的出現，而遮蔽區域在合成過程中會有破洞(Hole)現象的產生。

本論文中，為改善被動式景深估測的缺點，我們使用主動式景深感應器-Kinect取得景深資訊。主動式景深影像在缺少特徵點的區域，可以估測出較準確的景深資訊，且其執行速度較快，可以做即時(Real-Time)的處理。我們使用兩台Kinect感應器來做為左右相機，利用其實現影像合成。考慮感應器在3維空間的資訊，及景深影像的邊緣雜訊與缺陷(Defects)，我們提出一套演算法來改進，最後利用改善後的景深資訊與彩色影像來做合成，利用合成後的影像檢測其結果。

在「校正處理」的步驟，利用彩色影像之間的資訊，來估測出兩個感應器之間的3維空間幾何關係，並且考慮兩感應器接收到的光源特性不同，會針對其色彩差異做調整。考慮景深感應器與彩色相機位於不同視角，為得到互相匹配的景深影像與彩色影像，在「校準處理」會考慮兩種影像間的不匹配，除利用Kinect SDK的函數外，我們會針對其不足，做水平方向與垂直方向的誤差考慮，最後讓這兩種相機投影在相同的視角上。考慮Kinect擷取到景深影像上的雜訊及缺陷問題，利用聯合雙向濾波器(Joint Bilateral Filter)濾除雜訊以及補「洞」，及參考彩色影像的資訊來修復景深影像上的缺陷。最後的實驗結果中，除主動式景深資訊會較被動式景深資訊來的正確外，改善後的景深影像，也較原始的景深影像佳。以及合成結果中，可以看出改善前的合成影像，與改善後的合成影像相比較，可以看到改善後的合成影像可以得到較好的表現。



Depth Refinement for View Synthesis using Depth Sensor and RGB Camera

Student: Yi-Wen Chiou

Advisor: Dr. Hsueh-Ming Hang

Dr. Dr. Jang-Jer Tsai

Department of Electrical Engineering &

Institute of Electronics

National Chiao Tung University

Abstract

In recent years, three-dimension (3D) video has been a trend after the very popular science fiction movie – Avatar produced in the 2009. Many 3D movies, TV sets and even mobile phone sets have been developed. The view synthesis technology is an essential element in a 3D video system. The current technology adopted by the international MPEG committee on 3DVC standard is to generate a virtual viewpoint scene by using the received 2D views and their associated depth information. Therefore, the depth information plays an important role in 3D view synthesis.

In general, we can estimate the depth information by using two 2D texture images of different viewpoints using various depth estimation methods, and this approach is called Passive Depth Estimation. Very often, this approach fails to provide accurate depth information on the textureless regions. Furthermore, the occlusion regions which always exist due to 2 cameras, often lead to the “hole” defects on the synthesized views.

In this study, we adopt an active sensor – Kinect to capture the depth information. This active sensor provides pretty accurate depth information on the textureless regions, and it can operate in real-time. To generate a new view or virtual view, we use a pair of Kinect sensors as the left and right cameras. The Kinect depth sensor can be treated as another camera and thus we employ and improve some conventional techniques to estimate its camera parameters, calibrate its images (depth maps) and reduce artifacts.

In the calibration step, we use the information between two texture images to estimate the 3D geometry relationship between two Kinect sensors. Furthermore, because the depth sensor and color camera are located at two different positions, we propose an “Alignment” procedure to match the coordinates of the depth image and the texture image. In designing and implementing our alignment procedure, we use a disparity model and the Kinect SDK functions. Finally, we use the Joint Bilateral Filter and color information to reduce noises and defects on the sensor-acquired depth map. Comparing to the depth map estimated by using the MPEG Depth Estimation Reference Software (DERS), the captured and processed depth map clearly provide more accurate depth information. At the end, we examine and compare the synthesized images using the original and the refined depth maps. The quality of the refined synthesized image is noticed improved.

誌謝

這段不長也不短的兩年研究所生涯中，學習到許多專業知識，在挫折與壓力的磨練下，讓自己成長許多。這段充實的人生經歷，在我人生道路上絕對是一段重要的里程碑。

這段研究生生涯中，首先要感謝我的指導教授-杭學鳴老師，除在專業領域的指導以外，每次的討論中，老師也教導我們做研究應有的態度以及做學問的方法。老師也提供充足的研究資源，讓我可以恣意的從事我的研究而不用有任何顧慮。且老師謙遜的態度與和善地待人處事是我學習的目標。

研究的過程中，首先要感謝蔡彰哲學長的指導。在每個繁瑣的研究討論中，學長總是不厭其煩聆聽我的解釋，並且在仔細思考後，總是會給我一些很好的建議。雖然每次討論不一定得到很好的結果，但學長總是會激勵我繼續努力，並且適時的建議我一些方向。從學長身上學到研究生應具備的行為與態度，以及工程師解決問題的步驟與方法。

然後我必須感謝 Commlab 一起奮鬥的夥伴們，感謝朝雄學長辛苦的管理實驗室，感謝峻利學長及郁婷學姊的幫忙，感謝立恆以及介甫學弟在研究上的幫忙，還有維哲、讀修、士傑、基峰、政憲、柔綾、建志、建宏、晉源以及學弟們的陪伴，讓我碩班生活豐富且精彩。

最後要感謝我的家人以及珮茹，在我背後一直鼓勵我，給我進步的能量。在此將這邊論文獻給所有愛我的人以及我愛的人，你們的溫暖是我前進的動力。

目錄

摘要.....	i
Abstract	iii
誌謝.....	v
目錄.....	vi
圖目錄.....	x
表目錄.....	xviii
第 1 章 緒論(Introduction)	1
1.1 研究背景(Background)	1
1.2 研究動機及貢獻(Motivation and Contribution)	2
1.3 論文大綱(Organization of this Thesis)	3
第 2 章 校正處理演算法與 MPEG 參考軟體概述(Overview of Calibration	
Algorithm and MPEG Reference Software)	4
2.1 相機校正(Camera Calibration)	4
2.2 影像校正(Image Rectification)	9
2.2.1 立體影像幾何概述(Overview of Stereo Geometry)	9
2.2.2 立體像對校正(Rectification of Stereo Pairs)	11
2.3 色彩校正(Color Calibration).....	13

2.4	被動式景深估測演算法介紹(Introduction to Passive Depth Estimation Algorithm)	15
2.4.1	景深值與視差值之間的關係(Relation between Depth and Disparity)	15
2.4.2	景深估測(Depth Estimation)	17
2.4.3	景深參考軟體修改(DERS Modification)	18
2.5	景深映射(Depth Map Warping)	20
2.6	1D 視點合成演算法介紹(Introduction to 1D View Synthesis Algorithm)	22
2.6.1	VSRS 1 維模式假設(Assumption of VSRS 1D Mode)	23
2.6.2	1 維自由視點合成(1D Mode View Synthesis)	24
第 3 章	Microsoft Kinect 介紹 (Introduction to Microsoft kinect).....	27
3.1	Microsoft Kinect 硬體架構與景深估測原理介紹(Microsoft Kinect: Basic Operations and Software)	27
3.2	Kinect SDK 介紹(Introduction for Kinect SDK)	29
3.3	OPENNI 介紹及景深感應器的相機參數估測(Introduction to OPENNI and Camera Calibration of Depth Camera)	32
第 4 章	主動式景深攝影機的景深圖改善演算法 (Depth Refinement Algorithm for Active Depth Camera)	35
4.1	彩色影像校正(Color Image Calibration)	37
4.2	影像校準(Alignment of Depth Image and Texture Image)	39

4.2.1	使用 Kinect SDK 函數的初步校正(Initial Calibration by Using Kinect SDK Function)	41
4.2.2	景深影像垂直校正(Vertical Calibration on the Depth Image)	45
4.2.3	景深影像的水平校正(Horizontal Calibration on the Depth Image).	49
4.3	景深影像改進(Depth Image Improvement)	52
4.3.1	聯合雙向濾波器濾除景深影像雜訊(Noise Filtering by Using Joint Bilateral Filter on the Depth Image).	53
4.3.2	使用聯合雙向濾波器消除景深影像的缺陷問題(Defects Reduction Using Joint Bilateral Filter on the Depth Image)	56
4.3.3	基於彩色影像改善景深影像(Depth Image Refinement based on Color Image).	59
4.3.4	中值濾波(Median-Filtering)	62
第 5 章	實驗結果與比較 (Experiment Results and Comparison)	64
5.1	第 1 組測試影像模擬與結果 (Simulation of First Test Sequence)	64
5.1.1	校正處理	65
5.1.2	校準處理	70
5.1.3	景深影像改善	73
5.1.4	影像合成	81
5.1.5	被動式景深影像與合成結果比較	83
5.2	第 2 組測試影像模擬與結果 (Simulation of Second Test Sequence)	87
5.2.1	校正處理	88
5.2.2	校準處理.	90

5.2.3	景深影像改善	91
5.2.4	影像合成	97
5.2.5	被動式景深影像與合成結果比較	99
5.3	第 3 組測試影像模擬與結果 (Simulation of Third Test Sequence).....	102
5.3.1	校正處理	103
5.3.2	校準處理	105
5.3.3	景深影像改善	106
5.3.4	影像合成	113
5.3.5	被動式景深影像與合成結果比較	115
5.4	第 4 組測試影像模擬與結果 (Simulation of Fourth Test Sequence) ..	117
5.4.1	校正處理	118
5.4.2	校準處理	120
5.4.3	景深影像改善	121
5.4.4	影像合成	127
5.4.5	被動式景深影像與合成結果比較	129
第 6 章	結論與未來工作(Conclusion and Future Work)	131
6.1	結論(Conclusion).....	131
6.2	未來工作(Future Work).....	133
參考文獻.....		134
自傳.....		139

圖目錄

圖 1	針孔模型示意圖	5
圖 2	相機外部參數轉換關係示意圖	7
圖 3	多張影像用於單一相機作相機校正示意圖	9
圖 4	立體影像幾何關係示意圖	10
圖 5	校正後影像的極線幾何關係	11
圖 6	A: 原始座標系統 B: 影像校正後座標系統	12
圖 7	左右相同型號相機色調差異示意圖	13
圖 8	色彩校正流程圖	14
圖 9	以 CDF 做匹配示意圖	14
圖 10	Z_{far} 、 Z_{near} 與 Z 關係示意圖[21].....	15
圖 11	景深與視差關係圖	16
圖 12	DERS Segmentation Mode [11]	17
圖 13	產生右邊景深示意圖	18
圖 14	產生左邊景深示意圖	19
圖 15	原始 DERS 產生中間影像	19
圖 16	修改後 DERS 產生中間影像	20
圖 17	景深示意圖	21
圖 18	視角改變造成景深改變示意圖	21
圖 19	景深影像校正流程圖	22
圖 20	1 維自由視點合成流程圖	24
圖 21	前映射後的彩色影像與景深影像	25
圖 22	A: 結合後的彩色影像 B: 結合後的景深影像	26
圖 23	補洞完的圖	26

圖 24	Kinect for Xbox 360[23]	27
圖 25	Kinect 與軟體程式庫連結關係示意圖	30
圖 26	紅外線影像	32
圖 27	鹵素燈照射的影像	33
圖 28	紅外線測試影像	33
圖 29	特徵點擷取示意圖	33
圖 30	演算法流程圖	35
圖 31	兩台 Kinect 感應器擺放示意圖	35
圖 32	景深影像	36
圖 33	校正處理流程圖	37
圖 34	Kinect 在相同光源下影像色彩差異示意圖	38
圖 35	校準處理流程圖	39
圖 36	A:彩色影像 B:對應的景深影像	39
圖 37	由圖 36 兩張影像作疊合	40
圖 38	A:校正版在景深影像 B:校正板在彩色影像	40
圖 39	A:挖洞校正版在彩色影像 B:挖洞校正版在景深影像	41
圖 40	利用 Kinect SDK 函式做初步校正	42
圖 41	圖 40 的一角在 MATLAB 的數值圖	43
圖 42	填補後的影像	43
圖 43	彩色影像與景深影像疊合圖	43
圖 44	Kinect SDK 函式預期的映射示意圖	44
圖 45	Kinect SDK 函式映射至虛擬相機示意圖	44
圖 46	相機關係造成各自遮蔽現象的示意圖	45
圖 47	A:景深影像 B:初始邊緣影像	46
圖 48	初始邊緣影像參考彩色影像產生改善後邊緣影像	47

圖 49	景深影像疊合彩色影像	47
圖 50	垂直校正後的相機示意圖	49
圖 51	基線估測測試影像	50
圖 52	相機投影示意圖	50
圖 53	考慮視差模型做水平位移後的影像	51
圖 54	影像改善與合成流程圖	52
圖 55	影像改善須改善的問題示意圖	52
圖 56	A: 原始影像 , B: 聯合雙向濾波後影像	54
圖 57	聯合雙向濾波使用 1 個通道的灰階影像	55
圖 58	聯合雙向濾波使用 3 個通道的彩色影像	55
圖 59	圖 57、圖 58 細部比較(1).....	56
圖 60	圖 57、圖 58 細部比較(2).....	56
圖 61	遞迴式瑕疵現象消除流程圖	57
圖 62	原始輸入影像	58
圖 63	第 1 次遞迴	58
圖 64	第 5 次遞迴	58
圖 65	第 9 次遞迴	59
圖 66	輸出影像	59
圖 67	圖 66 與對應的彩色影像疊合.....	60
圖 68	基於彩色影像改善景深影像演算法流程圖	60
圖 69	A: 景深影像的參考點 B: 相對於彩色影像取出參考方塊	61
圖 70	像素值統計表	62
圖 71	輸出影像	62
圖 72	中值濾波輸出影像	63
圖 73	兩台 Kinect 感應器擺放示意圖.....	64

圖 74	A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像 .	64
圖 75	A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像 .	65
圖 76	校正函式- calib_gui	66
圖 77	標準模式	66
圖 78	校正軟體-影像讀取	66
圖 79	特徵點擷取示意圖	67
圖 80	反投影後影像	68
圖 81	A: 原始左邊 Kinect 影像 B: 原始右邊 Kinect 影像	68
圖 82	A: 左邊 Kinect 參考影像 B: 右邊 Kinect 校正影像	69
圖 83	A: 原始影像 B: 校正完影像	70
圖 84	A: 切割後左邊影像 B: 切割後右邊影像	71
圖 85	A: 映射切割後左邊景深影像 B: 映射切割後右邊景深影像 ...	71
圖 86	原始彩色影像以及 B1 輸出的景深影像	72
圖 87	圖 86 投影至新視角的影像.....	72
圖 88	原始彩色影像以及 B3 輸出的景深影像	72
圖 89	圖 88 投影至新視角的影像.....	73
圖 90	A: C1 - 左邊景深影像 B: C1 - 右邊景深影像	74
圖 91	C1 - 細部比較示意圖(1)	74
圖 92	C1 - 細部比較示意圖(2)	75
圖 93	A: C2 - 左邊景深影像 B: C2 - 右邊景深影像	76
圖 94	C2 - 細部比較示意圖(1)	76
圖 95	C2 - 細部比較示意圖(2)	77
圖 96	A: C3 - 左圖影像 B: C3 - 右圖影像	78
圖 97	C3 - 細部比較示意圖(1)	78
圖 98	C3 - 細部比較示意圖(2)	79

圖 99	A: C4 - 左圖影像 B: C4 - 的右圖影像	80
圖 100	C4 - 細部比較示意圖(1)	80
圖 101	C4 - 細部比較示意圖(2)	80
圖 102	原始合成影像(圖 84 與圖 85 所合成)	81
圖 103	影像改善後(C)合成影像(圖 84 與圖 95 所合成)	81
圖 104	圖 98 與圖 99 細部比較示意圖(1)	82
圖 105	圖 98 與圖 99 細部比較示意圖(2)	82
圖 106	圖 98 與圖 99 細部比較示意圖(3)	82
圖 107	A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像	83
圖 108	被動式景深影像合成結果	84
圖 109	A: VSRS 結合步驟的彩色影像 B: VSRS 結合步驟的景深影像 ..	84
圖 110	圖 105 A 細部區域分析	85
圖 111	A: VSRS 前映射後的左邊影像 B: VSRS 前映射後的右邊影像 ..	86
圖 112	A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像 .	87
圖 113	A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像 .	87
圖 114	A: 左邊參考影像 B: 右邊校正影像	88
圖 115	A: 影像校正後左邊影像 B: 影像校正後右邊影像	89
圖 116	A: 原始左右影像 校正後左右影像	89
圖 117	A: 映射切割後左邊景深影像 B: 映射切割後右邊景深影像	90
圖 118	A: 切割後左邊彩色影像 B: 切割後右邊彩色影像	90
圖 119	A: C1 - 左邊景深影像 B: C1 - 右邊景深影像	91
圖 120	C1 - 細部比較示意圖(1)	91
圖 121	C1 - 細部比較示意圖(2)	92
圖 122	A: C2 - 左邊景深影像 B: C2 - 右邊景深影像	92
圖 123	C2 - 細部比較示意圖(1)	93

圖 124	C2 - 細部比較示意圖(2)	93
圖 125	A: C3 - 左邊景深影像 B: C3 - 右邊景深影像	94
圖 126	C3 - 細部比較示意圖(1)	95
圖 127	C3 - 細部比較示意圖(2)	95
圖 128	圖 123 左邊景深影像與彩色影像疊合圖	95
圖 129	A: C4 - 左邊景深影像 B: C4- 右邊景深影像	96
圖 130	C4 - 細部比較示意圖(1)	96
圖 131	C4 - 細部比較示意圖(2)	97
圖 132	原始合成影像(圖 113 與圖 114 所合成)	97
圖 133	影像改善後(C)合成影像(圖 114 與圖 125 所合成)	98
圖 134	合成影像細部比較示意圖(1)	98
圖 135	合成影像細部比較示意圖(2)	98
圖 136	合成影像細部比較示意圖(3)	99
圖 137	A:DERS 估測左邊景深影像 B:DERS 估測右邊景深影像	100
圖 138	被動式景深影像合成結果	100
圖 139	圖 134 細部區域示意圖	101
圖 140	A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像	102
圖 141	A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像	102
圖 142	A: 左邊參考影像 B: 右邊校正影像	103
圖 143	A: 影像校正後左邊影像 B: 影像校正後右邊影像	104
圖 144	A: 原始左右影像 B: 校正後左右影像	104
圖 145	A: 切割映射後左邊景深影像 B: 切割映射後右邊景深影像 ..	105
圖 146	A: 切割後左邊彩色影像 B: 切割後右邊彩色影像	106
圖 147	A: C1 - 左邊景深影像 B: C1 - 右邊景深影像	106
圖 148	C1 - 細部比較示意圖(1)	107

圖 149	C1 - 細部比較示意圖(2)	107
圖 150	A: C2 - 左邊景深影像 B: C2 - 右邊景深影像	108
圖 151	C2 - 細部比較示意圖(1)	109
圖 152	C2 - 細部比較示意圖(2)	109
圖 153	A: C3 - 左邊景深影像 B: C3 - 右邊景深影像	110
圖 154	C3 - 細部比較示意圖(1)	110
圖 155	C3 - 細部比較示意圖(2)	111
圖 156	A: C4 - 左邊景深影像 B: C4 - 右邊景深影像	111
圖 157	C4 - 細部比較示意圖(1)	112
圖 158	C4 - 細部比較示意圖(2)	112
圖 159	原始合成影像(圖 142 與圖 141 所合成)	113
圖 160	影像改善後(C)合成影像(圖 142 與圖 152 所合成)	113
圖 161	合成影像細部比較示意圖(1)	114
圖 162	合成影像細部比較示意圖(2)	114
圖 163	合成影像細部比較示意圖(3)	114
圖 164	A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像	115
圖 165	被動式景深影像合成結果	116
圖 166	圖 161 細部區域示意圖	116
圖 167	A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像	117
圖 168	A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像	117
圖 169	A: 左邊參考影像 B: 右邊校正影像	118
圖 170	A: 影像校正後左邊影像 B: 影像校正後右邊影像	119
圖 171	A: 原始左右影像 B: 校正後左右影像	119
圖 172	A: 切割映射後左邊景深影像 B: 切割映射後右邊景深影像 ..	120
圖 173	A: 切割後左邊彩色影像 B: 切割後右邊彩色影像	120

圖 174	A: C1 - 左邊景深影像 B: C1 - 右邊景深影像	121
圖 175	C1 - 細部比較示意圖(1)	121
圖 176	C1 - 細部比較示意圖(2)	122
圖 177	A: C2 - 左邊景深影像 B: C2 - 右邊景深影像	122
圖 178	C2 - 細部比較示意圖	123
圖 179	A: C3 - 左邊景深影像 B: C3 - 右邊景深影像	124
圖 180	C3 - 細部比較示意圖(1)	124
圖 181	C3 - 細部比較示意圖(2)	125
圖 182	A: C4 - 左邊景深影像 B: C4 - 右邊景深影像	125
圖 183	C4 - 細部比較示意圖(1)	126
圖 184	C4 - 細部比較示意圖(2)	126
圖 185	原始合成影像(圖 168 與圖 169 所合成)	127
圖 186	影像改善後(C)合成影像(圖 169 與圖 178 所合成)	127
圖 187	合成影像細部比較示意圖(1)	128
圖 188	合成影像細部比較示意圖(2)	128
圖 189	合成影像細部比較示意圖(3)	129
圖 190	A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像	129
圖 191	被動式景深影像合成結果	130
圖 192	圖 187 細部區域示意圖	130

表目錄

表 1	Kinect 規格表.....	28
表 2	景深感應器相機參數結果.....	34
表 3	垂直位移量與精準度統計表.....	48
表 4	基線估測表.....	50
表 5	濾除雜訊-聯合雙向濾波器參數表.....	73
表 6	抑制瑕疵現象-聯合雙向濾波器參數表.....	75



第1章 緒論(Introduction)

1.1 研究背景(Background)

近年來，隨著立體顯影演算法與 IC 製程上的進步，多重顯像(Multiview Imaging)已經有越來越多應用與研究[1]。MPEG 國際標準會議(ISO/IEC Moving Picture Expert Group)近年來致力於訂定 3DAV(3D Audio - Video)標準文件[2]，其中自由視點視頻(Free Viewpoint Video 簡稱 FVV) [3]的主要目的是希望利用有限的影像設備，合成三維空間中任意視角的場景。目前最常使用的合成方式是景深圖式合成(Depth Image-Based Rendering，簡稱 DIBR[4])，此合成系統主要利用 2 維彩色影像加上對應的景深資訊，藉此投影到虛擬視角。其優點是資訊量較少，且標準格式(如 MPEG-4)已把景深圖式合成系統列入其中[2]。

2010 年，微軟(Microsoft)發行一組結合彩色攝影機與景深感應器的機器-Kinect，這台機器初始設計主要是應用在 XBOX360 上，藉由此感應器來操作遊戲的進行[5]。Kinect 景深感應器跟以往的景深感應器(如:SR4000)不完全相同，Kinect 的價格較為低廉，景深影像的解析度較高。且 Kinect 使用光學編碼(Light Coding)的方式，來偵測空間中物體的景深資訊。

現在常見偵測景深資訊的方式有兩種，分別為主動式方法(Active Method)和被動式方法(Passive Method)[6]。前者主要是用感應器的方式，利用光學的方式對空間中的物體估測景深資訊(可參考 3.1 節);後者主要是透過兩台左右相機拍攝到的影像，利用立體匹配(Stereo Matching)的方式，計算出視差(Disparity)資訊，再轉換成景深資訊(可參考 2.4 節)。但被動式方法的計算複雜度高，如 480×640 大小的影像，使用參考軟體-DERS(2.4.2 節)估測一張景深影像，約需花費兩分鐘。儘管擁有較高的複雜度，被動式方法估出的景深影像也不保證完全

準確。主動式方法可即時(Real-Time)產生景深資訊(如:Kinect 可達到 30fps)，且大多數像素點的景深資訊也較為正確[6]。

主動式感應器也存在一些缺點，如邊緣雜訊問題、缺陷的發生(紅外線照射到的區域，但 CMOS 影像感應器接受不到;光滑的物體表面;發熱的物體等等)。本篇論文會著重於如何解決這些雜訊與缺陷。

1.2 研究動機及貢獻(Motivation and Contribution)

本篇論文中，我們使用兩台 Kinect 的彩色資訊以及景深資訊，來做影像合成。對於 Kinect 做問題分析，提出 4 個 Kinect 所擁有的問題，並針對某些問題再切割成數個子問題。利用校正處理、校準處理和景深影像改善對每個問題一一擊破。

目前為止，鮮少文獻論文使用 Kinect 的彩色資訊以及景深資訊，來實現影像合成。以[17]來說，使用一台 Kinect 做即時的影像合成，從結果來看其影像的遮蔽區域並不會貼緊物體邊緣，原因是該系統的景深影像跟彩色影像不完全匹配(Matching)，所以考慮視差模型時，產生的遮蔽區域對應到彩色影像時，並不會完全貼緊物體邊緣。所以我們在校準處理時，會針對 Kinect 的景深影像與彩色影像的匹配問題，來加以討論。相較於[17]的單相機系統，我們使用兩台 Kinect 感應器，因為遮蔽區域的出現機率會較少，相較於單純使用背景來做影像修補(Inpainting)遮蔽區域，會較為準確。

[24][25]中，對於景深影像的邊緣雜訊以及缺陷問題，考慮聯合雙向濾波器來濾除。但對於景深資訊估算錯誤的區域(如圖 93 的桌腳)，並無法使用濾波器來徹底修補，所以我們會參考彩色影像來做修補。在彩色影像差異鮮明的地方，我們利用周圍的景深資訊，對這些估算錯誤的地方做合理的修補。但聯合雙向濾波器的確減少邊緣雜訊問題，所以我們在參考彩色影像做修補景深影像前，會先使用聯合雙向濾波器來去除雜訊，以增加景深資訊的正確性。

1.3 論文大綱(Organization of this Thesis)

本篇論文中，第二章會簡單針對三維空間與二維影像之間的關係做討論，並且對於 MPEG 所發布的參考軟體-DERS 與 VSRS 做簡單的介紹。在第三章會對於所使用的軟體-Kinect SDK 做詳細的介紹。第四章主要針對整體的演算法流程做介紹，分別對彩色影像作校正處理，及景深影像作校準處理，最後在影像改善的部分對景深影像做濾波及改善加強。第五章是實驗結果的呈現，針對每一步處理做詳細討論，並且對於細部區域做放大比較。最後，在第六章對本論文未來可能的研究方向做簡單的說明，並且對本論文做個總結。



第2章 校正處理演算法與 MPEG 參考軟體概述 (Overview of Calibration Algorithm and MPEG Reference Software)

本章節主要介紹關於「相機校正」、「影像校正」、「色彩校正」以及「景深映射」等校正演算法;並對於 MPEG 標準會議,所提供的景深估測參考軟體(Depth Estimation Reference Software, 簡稱 DERS)與影像合成參考軟體(View Synthesis Reference Software, 簡稱 VSRS)做簡單的介紹。

現實中 3 維座標系統與 2 維影像座標系統之間存在某種轉換關係,這個轉換關係可以透過相機參數來描述,在 2.1 節的「相機校正」,我們估測出相機參數來描述出兩座標系統之間的關係。雙相機系統之間的幾何關係,及如何降低雙相機系統間的問題複雜度,都會影響合成後虛擬視點的影像品質,這些問題在 2.2 節的「影像校正」的處理會被考慮到。考慮雙相機系統中,兩台相機位於不同位置,所以接收到的光源特性不同,造成拍攝出的影像之間有色差關係,且最後合成出的虛擬視角影像看起來不自然,2.3 節中考慮「色彩校正」來平衡兩張影像間顏色的不一致性。2.4 節利用 MPEG 的景深估測軟體(DERS)來實現被動式景深估測演算法,藉此與主動式景深估測得到的景深影像做比較。彩色影像做完影像校正(2.2 節)後,其對應的景深影像也必須映射到與彩色影像相同的視角上,所以在 2.5 節使用「景深映射」將景深影像的視角投影到彩色相機的視角。最後,2.6 節利用 MPEG 合成參考軟體(VSRS),來合成出虛擬視點的影像。

2.1 相機校正(Camera Calibration)

3D 幾何或電腦視覺中,為得知 2D 影像座標點與 3D 空間座標點之間對應關係,我們必須定義相機參數來表示它們之間的轉換關係[14]。相機參數主要分成內部參數(Intrinsic Parameter)和外部參數(Extrinsic Parameter),分別代表座標系統之間的轉換關係。

內部參數與相機內部構造有關，代表相機座標與影像座標之間的轉換關係。這些參數一般可視為針孔相機模型(Pinhole Camera Model)[18]，針孔相機模型主要利用針孔來做針孔成像的轉換。內部參數主要包含焦距長(Focal Length)、中心點(Principal Point)和歪斜係數(Skew Factor)，其中歪斜係數代表著影像平面 X 軸與 Y 軸之間夾角的歪斜程度，其發生的可能原因之一是 CCD (charge-coupled device) 的感光晶元歪斜所導致。圖 1 為針孔模型示意圖。

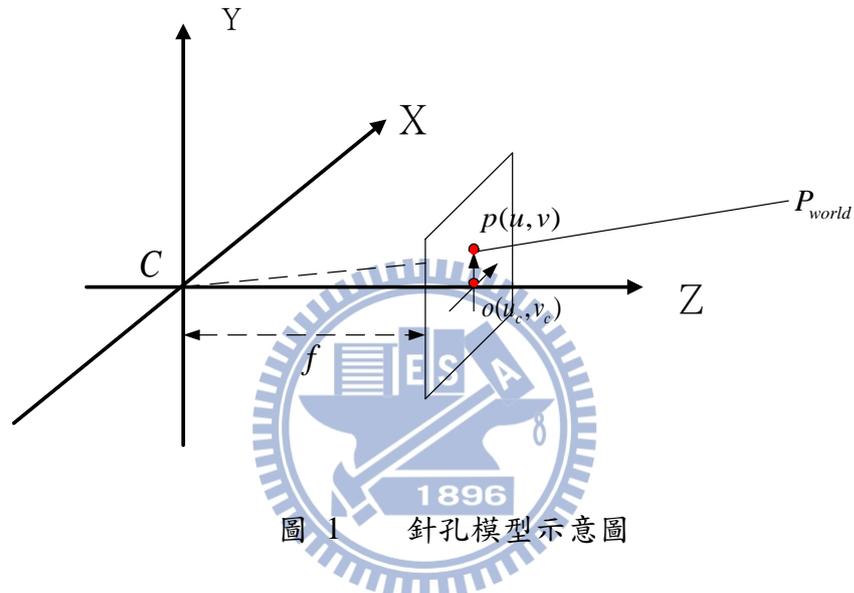


圖 1 針孔模型示意圖

圖 1 中， C 代表相機中心點、 f 為焦距長。假設一點 P_{world} 為三維空間中的一點，其座標為 (X_w, Y_w, Z_w) ，對應 P_{world} 點投影到二維影像座標上的點為 p ，其座標為 (u, v) 。考慮影像中心點 (u_c, v_c) ，利用相似三角形的幾何關係可獲得關係式如下式(1)。

$$(u, v) = \left(\frac{Xf}{Z} + u_c, \frac{Yf}{Z} + v_c \right) \quad (1)$$

以齊次(Homogeneous)座標系統來表示影像座標點與三維空間上的座標點轉換關係式，可表示成式(2)。

$$s \cdot p = s \cdot \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c & 0 \\ 0 & f_v & v_c & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = M \cdot P \quad (2)$$

其中， P 為三維空間中的座標點， p 為影像平面中所對應的座標點， s 為點 p 所對應的景深值(在這裡景深值指的是相機與物體在空間中的實際距離，單位為公厘(Millimeter))， M 是一個 3×4 的轉換矩陣，我們稱之為投影矩陣(Projective Matrix)。

式(2)可以再簡化成 3×3 的矩陣 K 並且考慮歪斜係數 γ 。

$$p = K[I_{3 \times 3} | 0]P_{cam} \quad (3)$$

$$K = \begin{bmatrix} f_u & \gamma & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \quad (4)$$

式(2)的投影矩陣中，因為現在 CCD 的品質越來越好，歪斜程度幾乎很小，所以 γ 通常都設為 0，。 f_u 和 f_v 分別代表 u 方向和 v 方向不同的焦距長，原因之一是晶元長度會導致影像平面 x 軸和 y 軸比例不一致。完整的內部參數矩陣可表示成式(4)。

外部參數主要跟相機的擺放位置與定義的世界座標有關，代表世界座標與相機座標之間的轉換關係。由於兩種座標系統都是三維的座標系統，所以它們之間可以用一個旋轉矩陣與位移矩陣(Translation Matrix)去做轉換。如圖 2 所示。

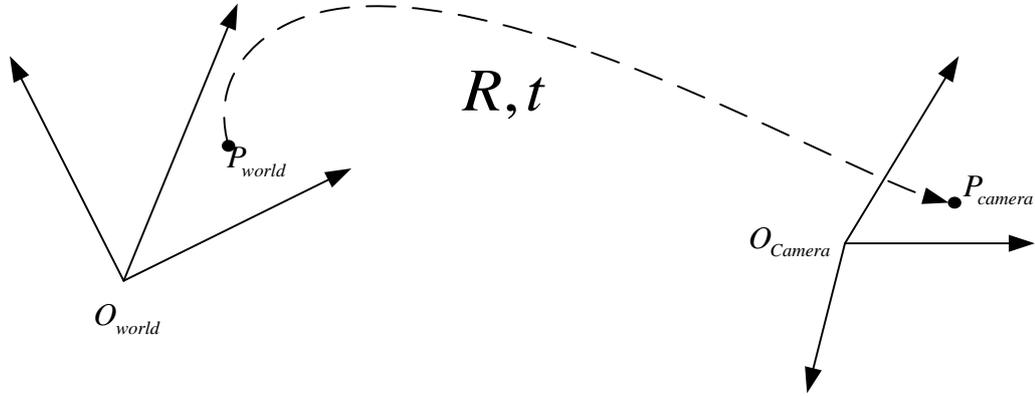


圖 2 相機外部參數轉換關係示意圖

其中 O_{world} 為世界座標系統原點， O_{camera} 為相機座標系統原點。

圖 2 中，有一點 P_{camera} 為相機座標系統上的一點且座標為 $(X_{cam}, Y_{cam}, Z_{cam})$ ，其對應於三維世界座標上的點 P_{world} 且座標為 (X_w, Y_w, Z_w) ，其關係式可寫成式 (5)。

$$\begin{bmatrix} X_{cam} \\ Y_{cam} \\ Z_{cam} \end{bmatrix} = R_{cw} \begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} + t_{cw} = [R_{cw} \quad | \quad t_{cw}] \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (5)$$

其中， R_{cw} 為旋轉矩陣、 t_{cw} 為位移向量。若同時考慮式(3)與式(5)，則空間中三維的座標點與二維影像平面的對應座標點關係，則可以表示成式(6)。

$$s \cdot \tilde{p} = K \cdot P_{cam} = K \cdot [R_{cw} \quad | \quad t_{cw}] \cdot \tilde{P} = M \cdot \tilde{P} \quad (6)$$

$$\tilde{p} = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \quad \tilde{P} = \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (7)$$

K 是內部參數，外部參數由 3×3 旋轉矩陣 R_w 和 3×1 位移向量 t_w 組成， M 為 3×4 的投影矩陣， \tilde{p} 為影像平面上的齊次座標點， \tilde{P} 為三維世界座標上的齊次座標點。

現實情況中，針孔模型只針對完美鏡頭可適用，但因為完美鏡頭幾乎不可能獲得，所以基於針孔模型考慮不完美鏡頭的影響，必須加入「鏡頭扭曲」(Lens Distortion)的係數針對投影點做轉換。假設 (u, v) 及 (\hat{u}, \hat{v}) 分別為理想(無扭曲)和實際情況(扭曲)的影像座標點。假設 k_1 及 k_2 分別為扭曲係數(Distortion Coefficient)，理想影像座標點與實際情況影像座標點可表示為式(8)、(9)。

$$\hat{u} = u + [k_1(u^2 + v^2) + k_2(u^2 - v^2)^2] \quad (8)$$

$$\hat{v} = v + [k_1(u^2 + v^2) + k_2(u^2 - v^2)^2] \quad (9)$$

更詳細的鏡頭扭曲介紹可參考[14]。

在了解空間座標點與影像平面座標的關係式後，利用特徵點在三維空間中與二維影像平面中的關係可估測內、外部參數矩陣。如何利用特徵點來求出相機參數，可參考[14]。實作上，我們利用校正板(如圖 3 所示)來擷取板子上的特徵點座標。定義世界座標原點及量測校正板上格子長度，利用這些資料來計算相機參數。為縮小估測出來的相機參數與實際相機參數之間的誤差，我們會利用多張不同角度、不同位移的照片，來獲得更多不同的特徵點位置，讓估測出來的相機參數更準確。

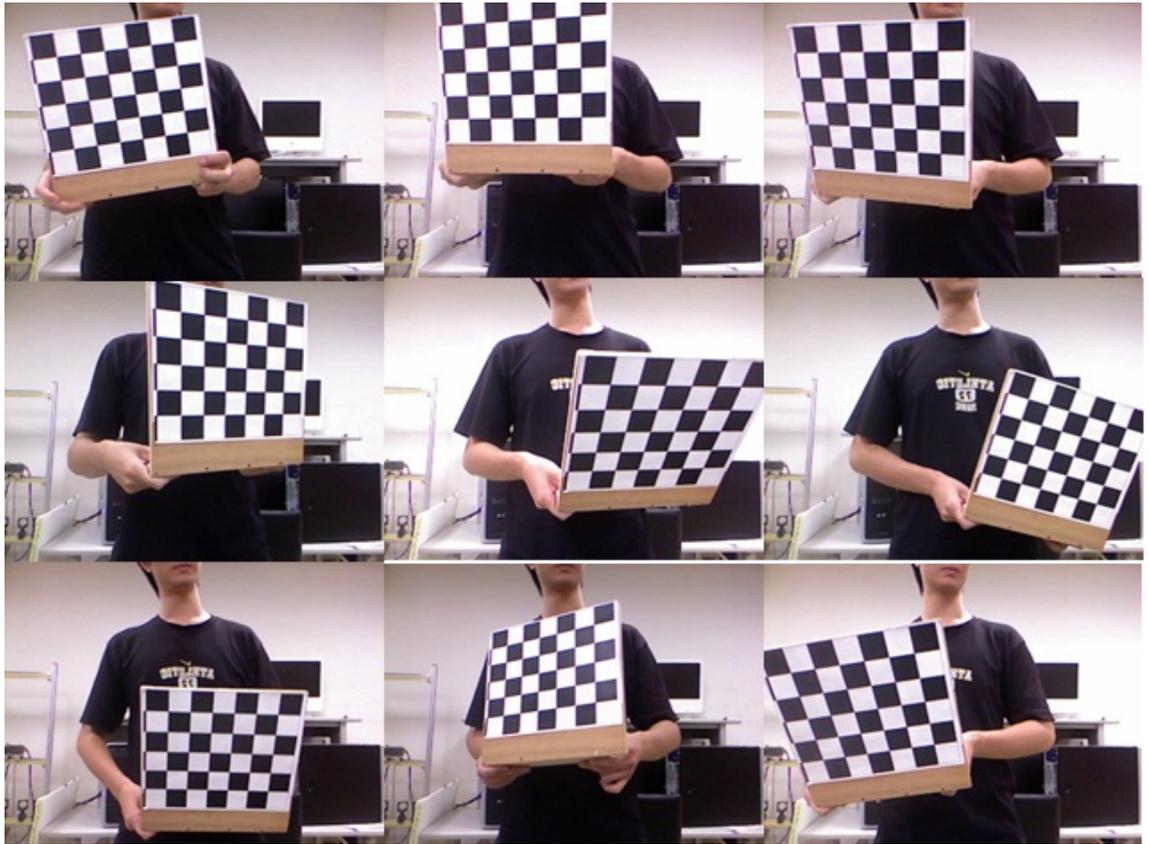


圖 3 多張影像用於單一相機作相機校正示意圖

2.2 影像校正(Image Rectification)

一般的情況下，兩台相同型號的相機擺在同一張桌子上，我們並不能保證這兩台相機是水平等高，拍攝出來的影像通常都有垂直位移差，且兩台相機的內部參數與旋轉矩陣也不會相同。電腦視覺領域中，常常利用「影像校正」的方式，使得影像之間沒有垂直的位移差，並讓兩台相機的內部參數、旋轉矩陣相同。立體影像之間的幾何關係可用「極幾何」來解釋，可以透過它來調整影像之間的關係，讓兩台相機投影到適當的虛擬位置，藉此達到影像校正的目的。

2.2.1 立體影像幾何概述(Overview of Stereo Geometry)

對於雙相機系統的幾何關係，我們可以用「極幾何」[18]來解釋。如圖 4 所示，利用兩台針孔成像相機(C_L 和 C_R)來擷取影像資料，並且將成像平面(I_L 和 I_R)置於鏡頭前面，則兩個鏡頭之間的連線($\overline{C_L C_R}$)為基線(Baseline)，空間中任一點

P 和兩台相機的基線組成的平面 $\triangle PC_L C_R$ 為極平面(Epipolar Plane)，極平面與成像平面相交的線稱為極線(Epipolar Line)(l_L 和 l_R)，基線與極線相交的點為極點(Epipole)(e_L 和 e_R)。所以空間中不同的點都會對應到不同的極線，且所有的極線會交於同一極點。

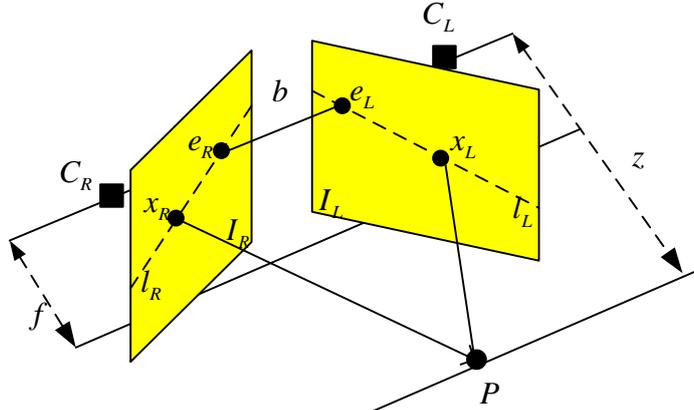


圖 4 立體影像幾何關係示意圖

圖 4 中，假設空間中點 P 投影到兩個成像平面的點分別為 x_L 和 x_R ，則 x_L 必定在 l_L 上，而 x_R 必定在 l_R 上。從代數的觀點來看，兩個點轉換關係存在一個轉換關係，這個轉換矩陣我們稱為基本矩陣(Fundamental Matrix)，可得到 x_R (需表示成齊次座標系統)與 l_L 之間的關係可表達為式(10)。

$$l_L = Fx_R \quad (10)$$

其中 F 為 3×3 的基本矩陣(Fundamental Matrix)，且 x_L (需表示成齊次座標系統)在 l_L 上，其內積必為 0，此關係可表示成式(11)。

$$x_L \cdot l_L = x_L^T Fx_R = 0 \quad (11)$$

從幾何上的觀點，線上各點都存在幾何對應關係。在兩張影像上要搜尋互相對應的點時，可利用此特性將原本的搜尋範圍，從二維的搜尋轉換成一維的搜尋，可以大量減少搜尋的時間複雜度。我們稱之為極線限制(Epipolar Constraint)。

2.2.2 立體像對校正(Rectification of Stereo Pairs)

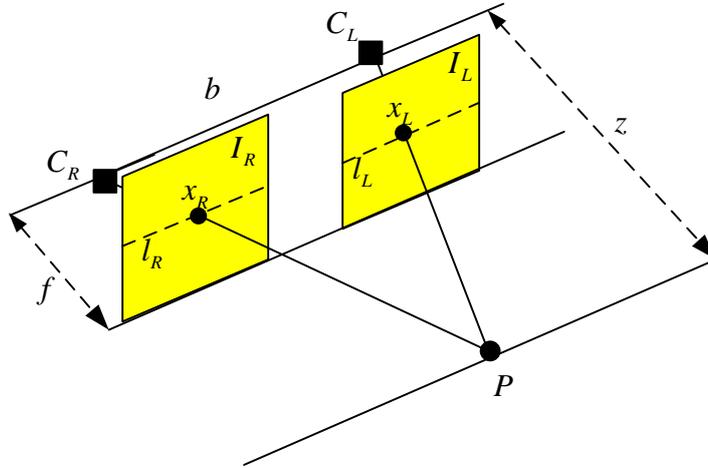


圖 5 校正後影像的極線幾何關係

圖 4 中，兩個影像平面並不平行的。從極線幾何來看，空間中各點對應到同一個影像平面上的極線彼此並不會互相平行，所以這些極線通通會交於極點(e_l 和 e_r)。當做完影像校正後(如圖 5 所示)，兩個影像平面會平行且共面，且兩影像平面對應的極線(l_l 和 l_r)會共線，同一影像平面的所有極線會互相平行，且交於無窮遠處的極點。

影像校正的想法主要透過簡單的相機座標系統旋轉(如圖 6 所示)來做到校正的工作[15]。一開始利用兩台相機各自的相機座標系統(如圖 6 A 所示)套用式 (12)，可得到鏡心 c 在空間中的三維座標。

$$c = -inv(R) \cdot t \quad (12)$$

其中 K 為內部參數， R 、 t 分別為外部參數的旋轉矩陣與位移向量。已知鏡心位置(c_L 和 c_R)後，必須定義新的座標系統(如圖 6 B 所示)，讓原始的座標系統旋轉至新位置。

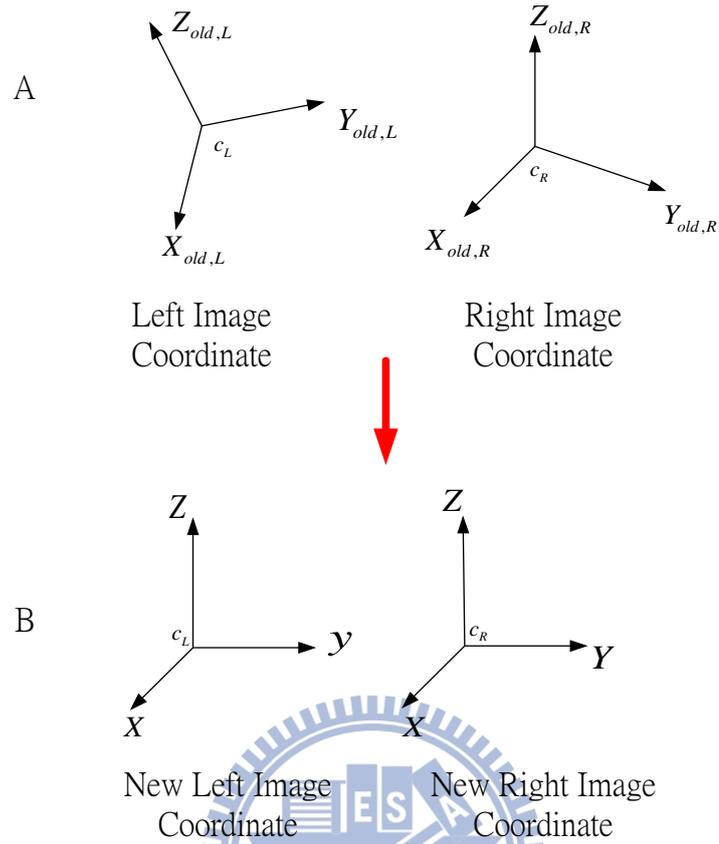


圖 6 A: 原始座標系統 B: 影像校正後座標系統

原始的旋轉矩陣 R ，分別為原左右座標系統的 X 軸、 Y 軸、 Z 軸三個方向向量(如圖 6 A)，新的座標系統定義在圖 6 B，新的旋轉矩陣就是由新的 x 軸、 y 軸、 z 軸所組成。新的 x 方向向量定義成兩個鏡心座標做相減，如式(13)所示。

$$\vec{X} = c_L - c_R \quad (13)$$

其中 c_L 、 c_R 分別為左右的鏡心座標。而 y 方向的向量由新的 x 軸向量跟左邊舊的 z 軸向量做外積，如式(14)所示。

$$\vec{Y} = \vec{X} \times \vec{Z}_{old,L} \quad (14)$$

最後的 z 軸向量就由新的 x 軸向量跟新的 y 軸向量去做外積，如式(15)所示。

$$\vec{Z} = \vec{X} \times \vec{Y} \quad (15)$$

新的旋轉矩陣 R' 新的座標系統所組成，新的內部參數 K' 因為可以任意選擇，所以由舊的內部參數 K_1 、 K_2 直接平均取得。所以新的相機參數 P_{n1} 和 P_{n2} 表示如式(16)與式(17)。

$$P_{n1} = K' [R' \quad -R'c_1] \quad (16)$$

$$P_{n2} = K' [R' \quad -R'c_2] \quad (17)$$

更詳盡的推導過程可以參考[15]。

2.3 色彩校正(Color Calibration)

一般來說，相同型號的相機，也不能保證內部的結構完全一模一樣。擺在相同的場景中，可能因為兩台相機位置的不同，導致收到的光源特性不同，所以拍攝出來的影像色調會有所差異(如圖 7)。對於立體影像來說，在後續的處理色調的相似度會有很大的影響，為後續處理工作的精確度，我們必須去做「色彩校正」。



圖 7 左右相同型號相機色調差異示意圖

基本想法是調整被校正影像的色調，使被校正影像的色調接近參考影像的色調。所以我們利用「直方圖匹配」(Histogram Matching)[26]，設法去調整色調。參考圖 8 為色彩校正流程圖。

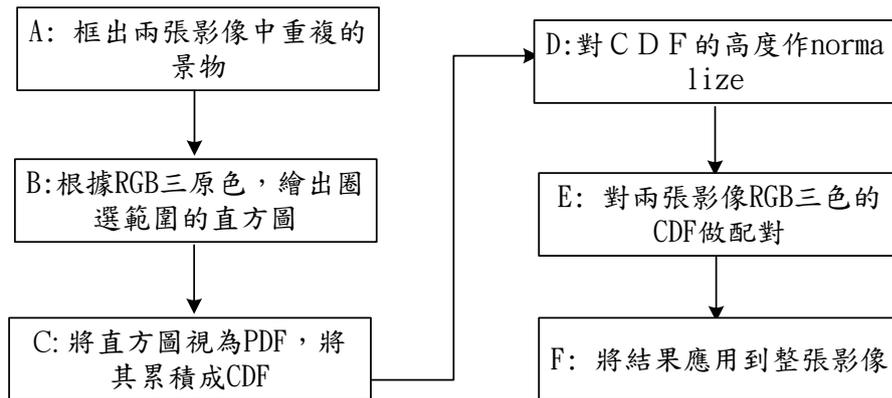


圖 8 色彩校正流程圖

圖 8 中的 A 步驟，對於兩張影像中我們選擇有重疊的地方做統計(圖 7 中選擇中間的色板來做統計)。圖 8 中的 B 步驟，我們分別去計算兩張圖 R、G、B 三色的直方圖，這些直方圖可近似看成不同的機率分布函數(Probability Distribution Function)。圖 8 中的 C 步驟，再把這些機率分布函數做累積，讓它們成為累積分布函數(Cumulative distribution function)。圖 8 中的 D 步驟，因為在圖 8 中的 A 步驟，圈選色板的區塊時不會完全一樣大，所以必須做正規化的動作。所以得到兩張圖各自 R、G、B 的累積分布函數，圖 8 中的 E 步驟讓被調整影像的每個顏色領域(Color Domain)去匹配目標影像的顏色領域，如圖 9 所示。

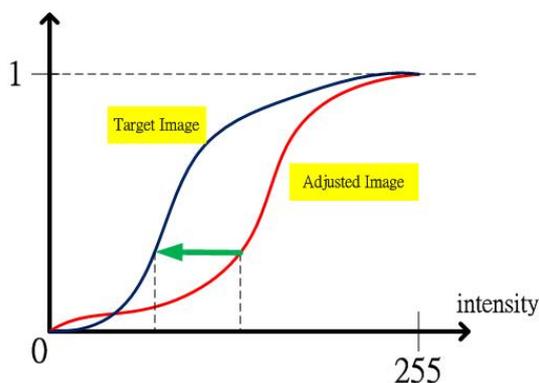


圖 9 以 CDF 做匹配示意圖

最後將匹配的結果在圖 8 中的 F 步驟，應用到被校正影像。

2.4 被動式景深估測演算法介紹(Introduction to Passive Depth Estimation Algorithm)

傳統的景深演算法中，利用兩張彩色影像之間的視差關係去估測景深值。我們使用 MPEG 標準會議所提供的景深估測參考軟體(Depth Estimation Reference Software)[19]來實現被動式景深估測演算法。

2.4.1 景深值與視差值之間的關係(Relation between Depth and Disparity)

景深資訊可用來解釋 3 維場景中，物體與相機之間的距離關係，假設空間中有一點的景深值為 Z ，參考[29]利用 Z_{far} 和 Z_{near} 針對 z (如圖 10 所示)做最好的非均勻(Non-Uniform)景深量化(Quantization)，對應在景深圖上的點 v 之間的關係可表達成式(18)。

$$Z = \frac{1}{\frac{v}{255} \cdot \left(\frac{1}{Z_{near}} - \frac{1}{Z_{far}} \right) + \frac{1}{Z_{far}}} \quad (18)$$

其中 v 是 8 位元強度的灰階值， Z_{near} 是 3 維空間中離相機最近的平面， Z_{far} 是 3 維空間中離相機最近的平面。

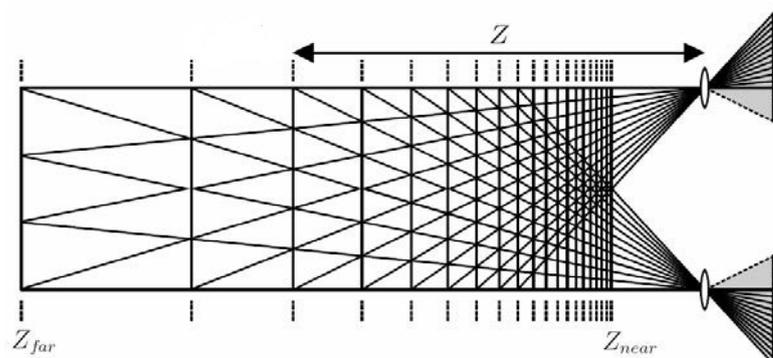


圖 10 Z_{far} 、 Z_{near} 與 Z 關係示意圖[29]

假設影像經過 2.2 節的影像校正，相機之間是 1 維平行的。參考圖 11 的景深與視差之間的關係圖。

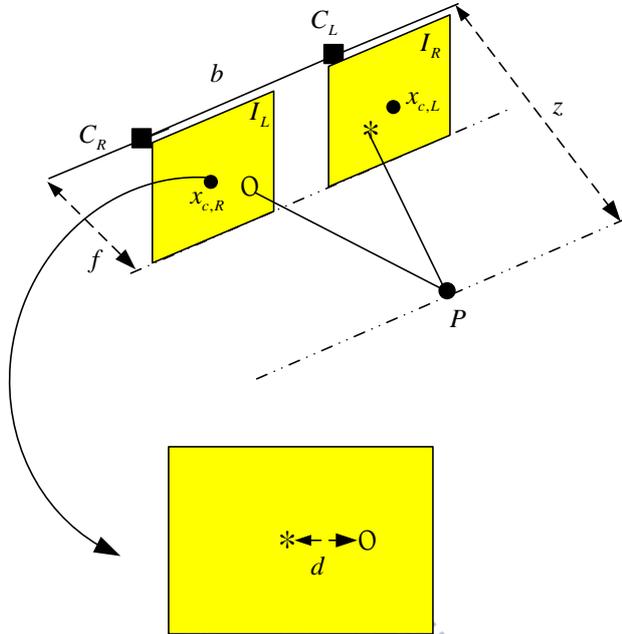


圖 11 景深與視差關係圖

其中， C_L 與 C_R 分別為左右相機鏡頭， $x_{c,L}$ 和 $x_{c,R}$ 分別為 C_L 與 C_R 在影像平面上的投影點， P 為 3 維空間中的一點， O 與 $*$ 分別為 P 點投影在左 (I_L)、右 (I_R) 影像平面上的對應點， f 是焦距長， b 是基線長， z 是景深值， d 為 O 與 $*$ 之間的視差值。利用相似三角形可求得式(19)。

$$z = \frac{f \cdot b}{d} \quad (19)$$

2.4.2 景深估測(Depth Estimation)

DERS 內有三種模式，分別為 Segmentation Mode、Automatic Mode 和 Semi-automatic Mode [19]。在此針對我們使用 Segmentation Mode 做介紹，詳細流程如圖 12 所示。

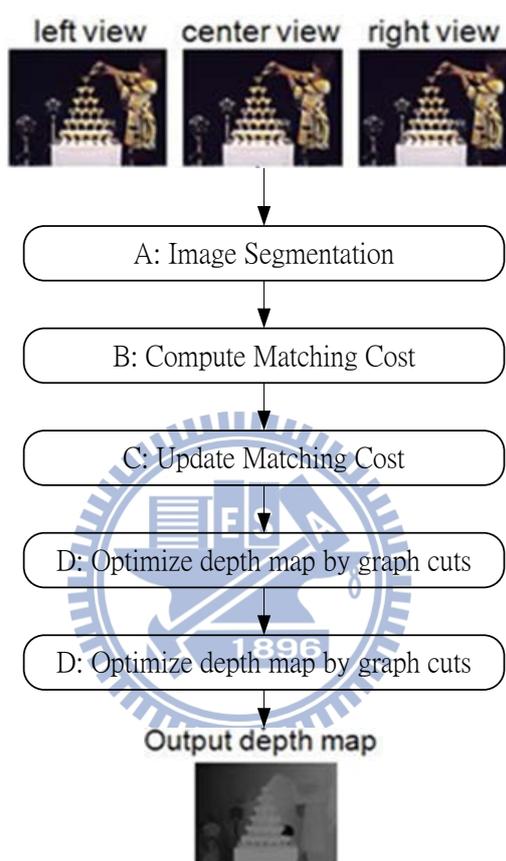


圖 12 DERS Segmentation Mode [19]

圖 12 的 A 步驟中，在一個任意被選擇的視窗大小(如:11×11 像素)內，比較中心像素與其他像素間的色彩差異與幾何距離，給予一個權重來表示這些像素與中心像素為同一個分割碎片的機率。這步驟對於後面計算相配成本會有較好的表現，但會增加較多的複雜度。

圖 12 的 B 步驟中，比較中間影像與左邊影像和中間影像與右邊影像之間每個碎片的配合成本函數。

圖 12 的 C 步驟中，比較左右的成本函數，且對於成本函數在時域空間作調整，最後選擇較小的成本函數當作中間影像在該像素的成本函數。

圖 12 的 D 步驟中，目的是決定所有像素的視差值。方法是使用 α 擴張移動 (α -expansion move)，來對於碎片做最佳化(Optimization)，使得能量函數最小。

圖 12 的 E 步驟中，對剩餘的雜訊做濾除，主要的方法包含基於 RANSAC(RANdom SAMple Consensus)和基於 x 和 y 歪斜值。

每個步驟更詳細的推導可參考[27]。

2.4.3 景深參考軟體修改(DERS Modification)

原始的 DERS 必須使用左、中、右三張影像當作輸入，最後輸出為中間影像所對應的景深圖，目的主要是為減少中間影像遮蔽區域的部分。實際上，使用兩張影像就可估出各自的景深圖，雖然與使用三張影像比較品質會較差，但是對雙相機系統這是不可避免的缺點。我們針對 DERS 做適當的修改，讓雙相機系統也可以利用 DERS 估測景深。

為達到我們需求，我們修改第三步的「Update Matching Cost」，讓參考軟體只選擇特定的成本函數。因為第三張影像完全不會被考慮，所以我們可以用左右的圖在加上一張任意影像(但大小須與左右圖相同)來估測左右圖景深，如圖 13 和圖 14 所示。

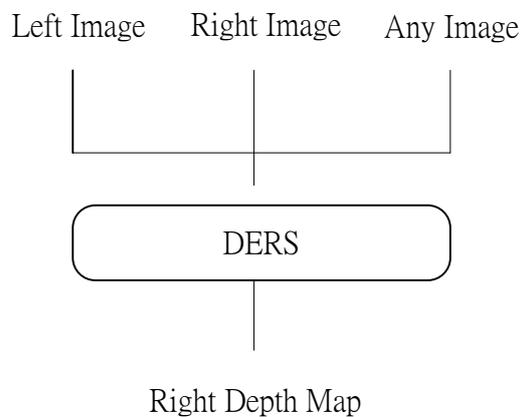


圖 13 產生右邊景深示意圖

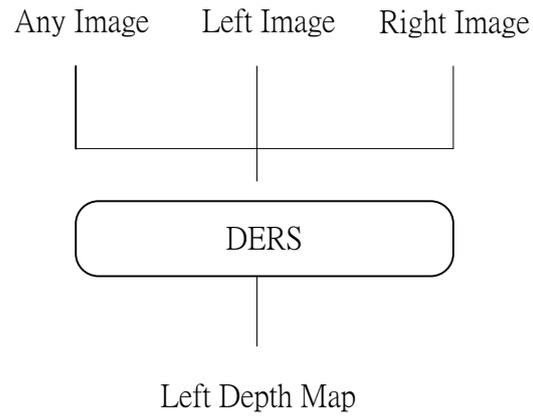


圖 14 產生左邊景深示意圖

而使用 MPEG 標準會議提供的測試影像-BOOK 的 07、05 和 03 視角的影像，來比較修改前跟修改後的影像品質差異。

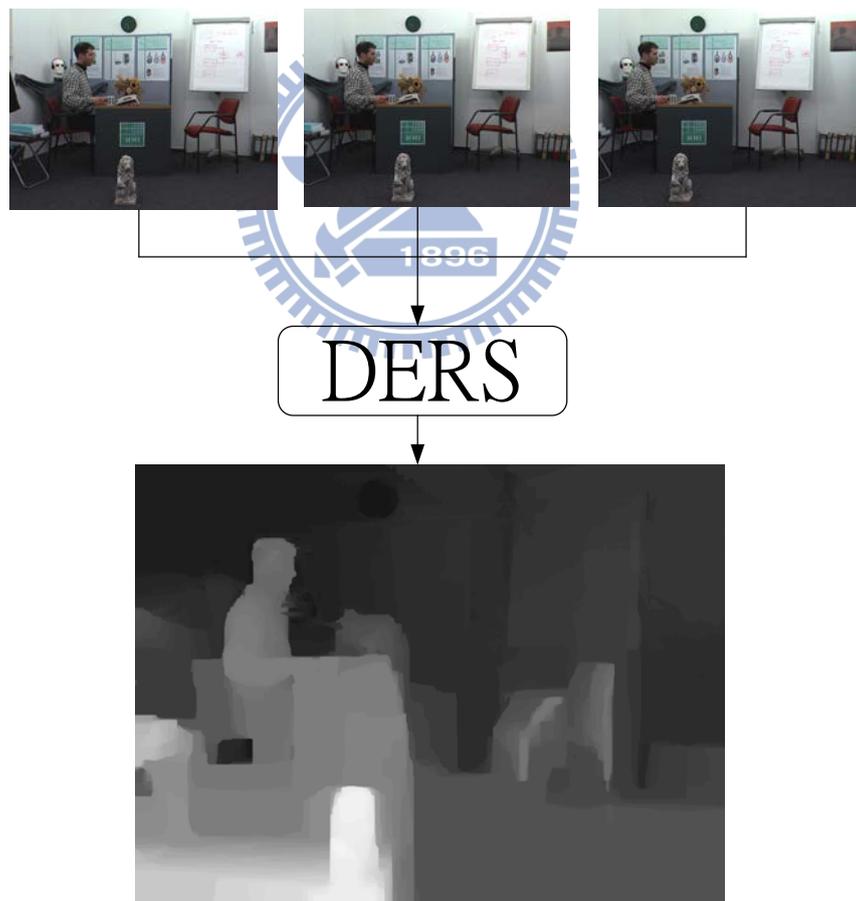


圖 15 原始 DERS 產生中間影像

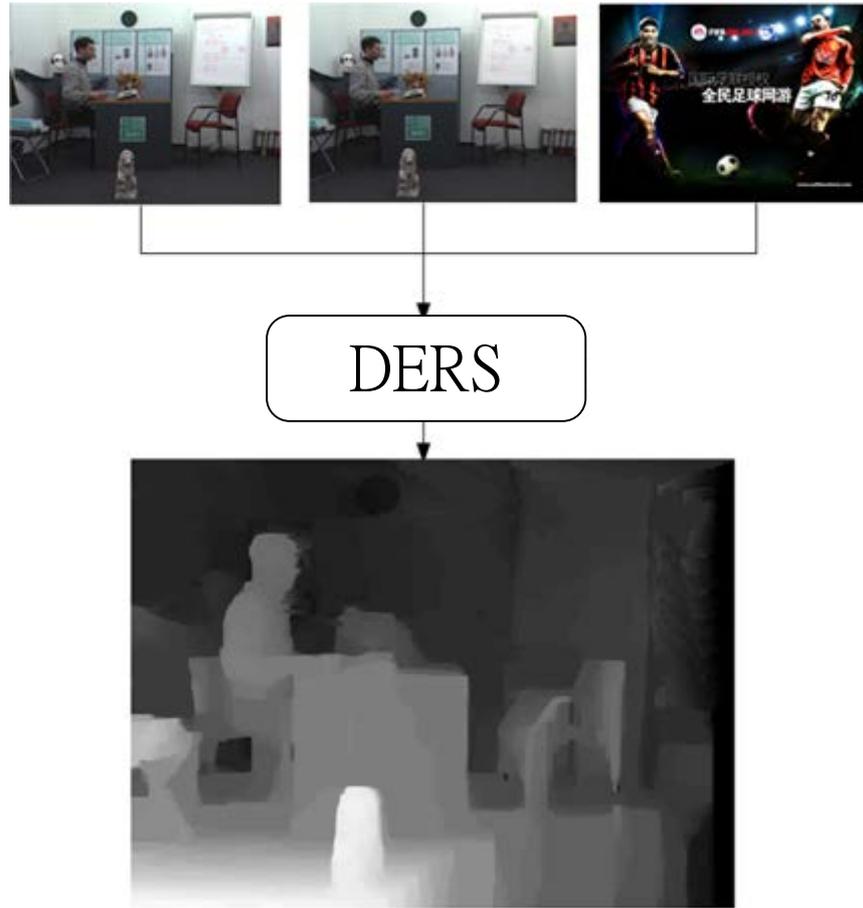


圖 16 修改後 DERS 產生中間影像

由圖 15 和圖 16 產生的景深影像來看，修改後的 DERS 所產生的景深圖，因為缺少右邊輸入影像的資訊，輸出影像右邊會有一些遮蔽區域(Occlusion Region)。比較圖 15 和圖 16，景深影像上的物體大多相同，部份物體的景深資訊會不太一樣，原因是缺少第三右邊影像的資訊，所以最後的輸出影像大致符合我們的預測。

2.5 景深映射(Depth Map Warping)

當彩色影像做完影像校正後，對應的主動式景深影像也必須配合彩色影像做調整，所以我們必須將景深影像映射至彩色影像校正後的相機位置，最後視點合成時，才能利用彩色影像與其對應的景深影像做合成。特別注意的是，這裡輸入的景深影像必須是做完 4.2 節「影像校準」的景深影像。影像校正比較直接的觀

點可以看成「換個角度來看場景」，因此可參考圖 17 所表示的景深定義，來考慮影像校正對於景深的變化。

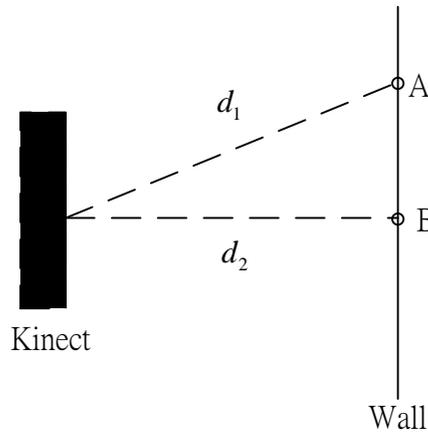


圖 17 景深示意圖

圖 17 中，對於一道與 Kinect 平行的牆(Wall)，上面有 A、B 兩點，其中 A 點距離 Kinect 為 d_1 ，B 點距離 Kinect 為 d_2 。因為景深圖是對於場景中與相機的垂直距離做紀錄，所以在圖 17 中，A、B 兩點的景深皆為 d_2 。此時針對景深影像做影像校正的時候會有問題，因為當我們換個角度看場景時，這時候場景跟鏡頭的垂直距離也會改變，如圖 18 所示。

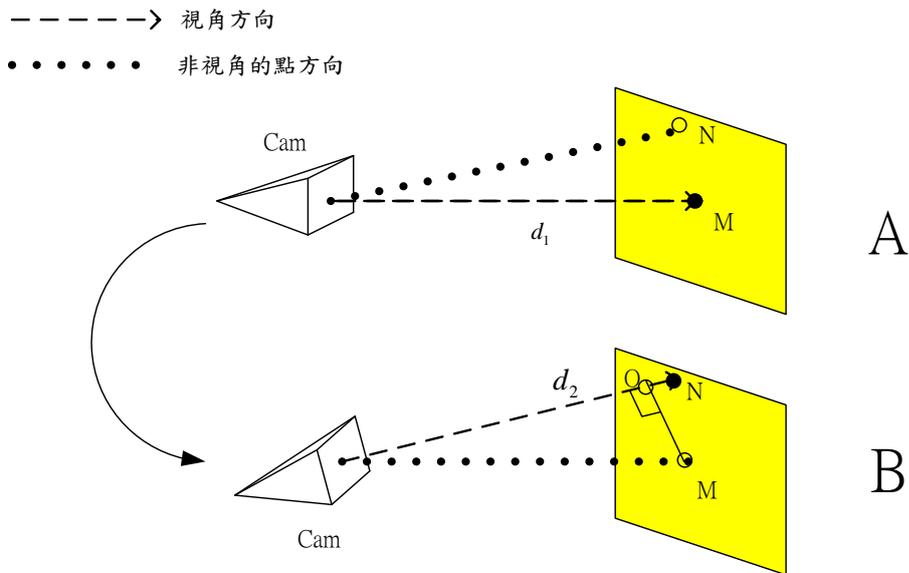


圖 18 視角改變造成景深改變示意圖

其中，白色三角錐為相機，黃色平面可想像是一道牆，虛線為相機視角的方向，點線代表非視角的點方向。圖 18 A 中，相機與牆是平行，所以相機的視角延伸垂直於牆上的點 M，此時景深距離為 d_1 。圖 18 B 中，相機旋轉一個角度，且視角延伸交於牆上在點 N，此時 M 點與相機並不垂直，所以 M 點的景深距離，必須投影在相機與 N 點的方向向量上且交於 O 點，O 點與相機距離 d_2 則為景深值。所以在景深影像做影像校正時，必須考慮到景深變化的問題。

參考 3 維映射投影的精神，詳細的流程如圖 19 所示。

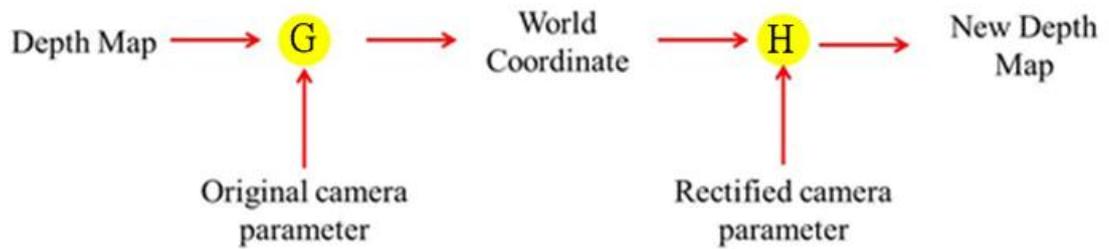


圖 19 景深影像校正流程圖

圖 19 的 G 步驟中，參考原始的相機參數，將原始的 2 維景深影像的像素點投影到 3 維空間座標上。圖 19 的 H 步驟則參考彩色影像校正後的相機參數，投影到新的 2 維影像平面上(如式(6)所示)，投影點的像素值則利用影像上最遠的景深平面與最近的景深平面去做量化的工作，如式(20)所示。

$$v = \frac{Z - Z_{\min}}{Z_{\max} - Z_{\min}} \times 255 \quad (20)$$

v 為量化後的像素值， Z 為該點的景深， Z_{far} 與 Z_{near} 為該影像上最遠與最近的景深平面。最後得到的影像為影像校正後的景深影像。

2.6 1D 視點合成演算法介紹(Introduction to 1D View Synthesis Algorithm)

對於虛擬視點的影像合成，我們基於 Depth Image Based Rendering (DIBR) 的方式，利用 2 維影像與對應的景深影像資訊來做合成。其優點是影像合成中，景

深影像的資料量較 2 維影像的資料量小，對於傳輸與儲存來說較有效率[30]。我們使用 MPEG 標準會議所提供的「影像合成參考軟體(View Synthesis Reference Software (VSRS)」3.0 版本[19]來合成虛擬視點的影像。

2.6.1 VSRS 1 維模式假設(Assumption of VSRS 1D Mode)

在 VSRS 1 維模式中，假設相機間的光軸(Optical Axes)平行(影像必須做過影像校正(2.2 節)的狀況下，我們可以利用幾個公式化(Formulation)的表達來陳述 1 維模式的假設。

- a. 每個相機參數的旋轉矩陣(R)都相同
- b. 每個相機參數位移向量的 T_y 、 T_z 皆相同
- c. $z_v = z_r$ (下標 v 代表是 Virtual Viewpoint，下標 r 代表是 Reference Viewpoint)
- d. 影像必須都做過校正，讓相機間的光軸沒有垂直位移差

其中，影像座標的點與世界座標的對應點可寫成式(21)，對照上面假設的參數，可了解世界座標點與影像座標點之間，哪些參數在 1 維模式下需要被特別假設。

$$z \times \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & 0 & u_c \\ 0 & f_v & v_c \\ 0 & 0 & 1 \end{bmatrix} \times R \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \times \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \quad (21)$$

式(21)中， z 是景深值， $[u \ v \ 1]^T$ 是影像座標系統的齊次(Homogeneous)座標， f_u 、 f_v 分別為 u 方向和 v 方向不同的焦距長， u_c 、 v_c 為中心點(Principal Point)的座標， R 是旋轉矩陣， $[T_x \ T_y \ T_z]^T$ 為位移向量， $[X_w \ Y_w \ Z_w \ 1]^T$ 為世界座標系統的齊次座標。

2.6.2 1 維自由視點合成(1D Mode View Synthesis)

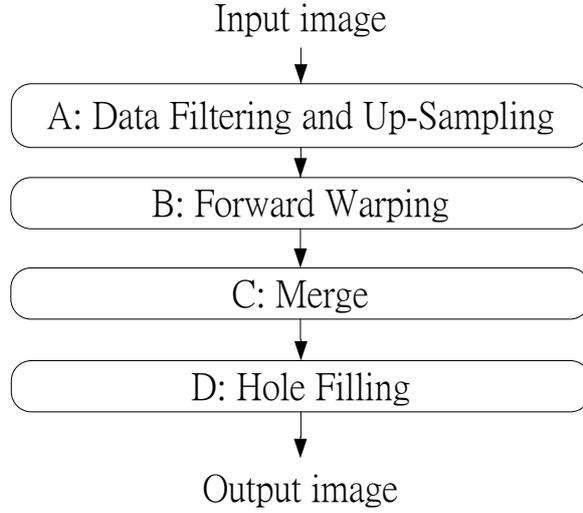


圖 20 1 維自由視點合成流程圖

輸入影像必須經過 2.2 節和 2.5 節的處理，使得影像沒有垂直位移差。

圖 20 的 A 步驟中，對於輸入影像的格式升取樣到 YUV444，並且會根據彩色影像的資訊先對景深影像做濾波。如果像素精準度(Precision of Pixel)到 1/4 個像素(Quarter Pixel)或 1/2 個像素(Half Pixel)的話，影像需要做升取樣(Upsample)的動作。

圖 20 的 B 步驟，可參考式(22)作為映射公式。

$$u_v = u_r + \frac{f_u \cdot (T_{x,v} - T_{x,r})}{z_r} + d_{u,v} - d_{u,r} \quad (22)$$

其中 u_r 指參考圖(Reference View)的某一點，而 u_v 為其在虛擬視點上的對應點，

f_u 是 u 方向的焦距長， $T_{x,v}$ 與 $T_{x,r}$ 分別為虛擬視點與參考圖在位移向量的 x 方向值，

$d_{u,v}$ 與 $d_{u,r}$ 為虛擬視點與參考圖在中心點在 u 方向值。

此步驟的結果可參考圖 21。

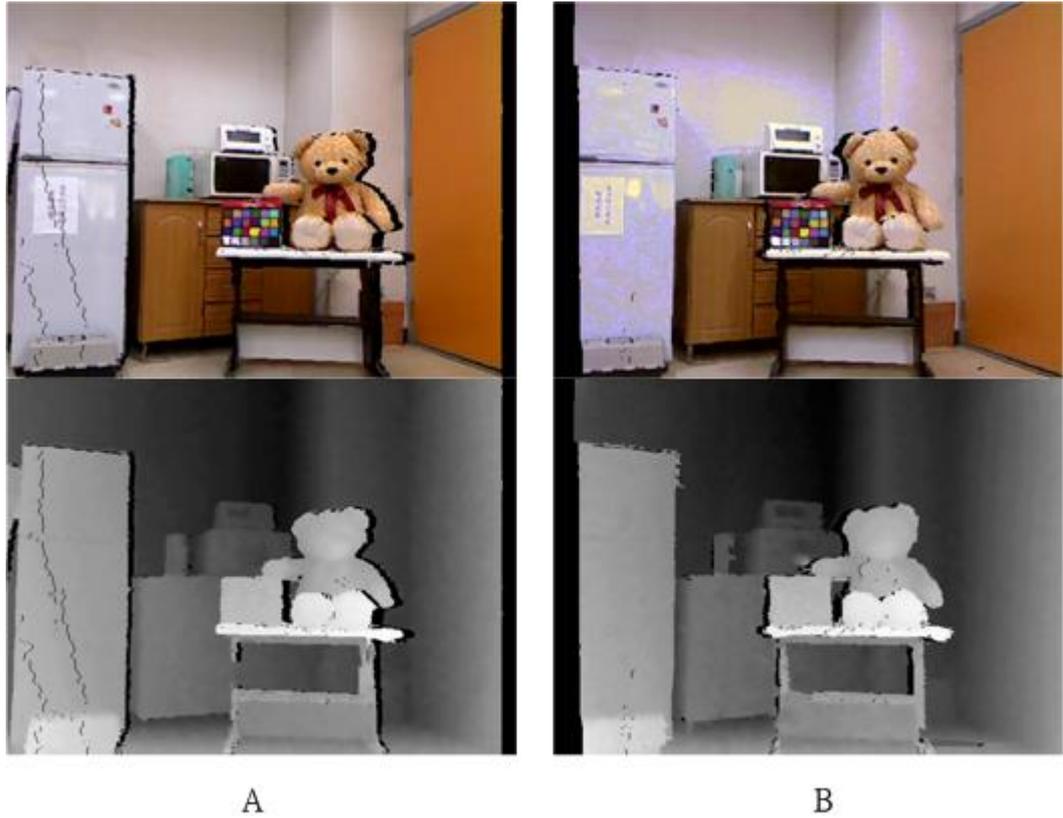


圖 21 前映射後的彩色影像與景深影像

其中，A 圖為虛擬視點左邊的影像，B 圖為虛擬視點右邊的影像。

圖 20 的 C 步驟，結合(Merge)步驟為把兩張前映射的影像設法結合成一張，決定虛擬視點影像中每個點的方法有三種。

- a. Z-緩衝器結合(Z-buffer Merge):選擇離虛擬相機最近的點。
- b. 平均結合(Average Merge):給予兩台相機一個權重在相加，而這個權重跟虛擬相機與其他相機的距離成反比。
- c. 適應結合(Adaptive Merge):當兩個相機與虛擬相機各自的距離差超過一個臨界值(Threshold)時，則選用 b 方法，否則選用 a 方法。

參考圖 22 是使用 c 方法做結合後的彩色影像與景深影像。

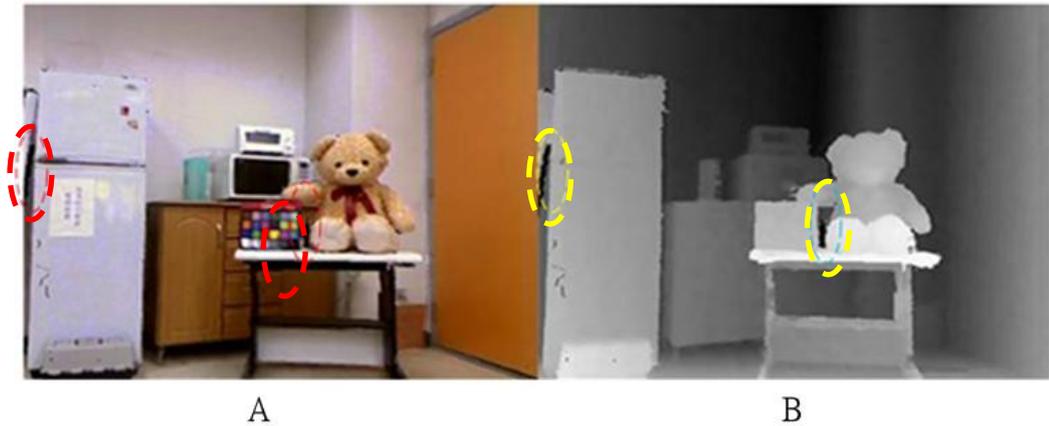


圖 22 A: 結合後的彩色影像 B: 結合後的景深影像

最後，圖 20 的 D 步驟是做補洞(Hole Filling)的動作。做完結合(Merge)後，有時左右影像在某些區域會同時都是遮蔽區域，會導致結合後的影像有遮蔽區域(如圖 22 彩色影像與景深影像虛線圈圈的部分)，則使用傳播補洞(Propagating Hole Filling)的方式來做填補。最後必須把 YUV444 轉成 YUV420，且如果影像一開始有做升取樣的話，此步驟也須要做降取樣(Down-Sample)的動作。

最後，參考圖 23 為最後補完洞的影像。從圖 23 可以明顯看到對應在圖 22 有破洞的地方(虛線圈圈部分)，已經完全被修補完成。



圖 23 補洞完的圖

第3章 Microsoft Kinect 介紹 (Introduction to Microsoft kinect)

微軟(Microsoft)在 2010 年 11 月發行一款體感裝置，其正式名稱為「Kinect for Xbox 360」(簡稱為 Kinect)，不同於任天堂所發行的 Wii，Kinect 透過攝影機及紅外線鏡頭，經由內部晶片計算，可以判斷使用者位置的遠近和整體動作。

2011 年 6 月，微軟釋出一款 Beta 版的 SDK，可使用的程式語言有 C++與 C#，其安裝過程簡單，且官方持續在更新版本(目前已經更新到 1.5 版，並且支援 Windows 8)，讓 Kinect 可以有更大的發揮空間。

微軟官方在 2012 年 2 月 1 日也發行最新版的 Kinect，名為「Kinect for Windows」，專為連接在電腦上設計的 Kinect，功能也比 Kinect for Xbox 360 較為更多，目前已經有多種基於 Kinect 的應用，相信未來 Kinect 的應用會越來越多。

3.1 Microsoft Kinect 硬體架構與景深估測原理介紹 (Microsoft Kinect: Basic Operations and Software)



圖 24 Kinect for Xbox 360[13]

Kinect 的運作主要是透過 Kinect 的鏡頭以及旁邊的麥克風陣列(如圖 24 所示)。中間的鏡頭是 RGB 彩色攝影機，可以擷取出彩色影像。左邊跟右邊分別為紅外線發射器(IR projector)及紅外線 CMOS 影像感應器，兩個鏡頭擷取到資訊，

再由內部晶片計算出 3D 深度影像。旁邊麥克風陣列(Microphone array)，可擷取玩家的聲音訊號，藉由多組麥克風收音可以達到降噪的功能。底座有馬達系統，搭配 Kinect 的追焦功能可隨著對焦物體轉動。Kinect 詳細的規格如表 1 [1] [13]。

表 1 Kinect 規格表

感應器	彩色和深度感應鏡頭
	陣列式麥克風
	輔助感應傾斜驅動馬達
	完全相容所有的 Xbox 360 裝置
視野角度	水平視野:57 度
	垂直視野:43 度
	實體傾斜範圍:±28 度
	彩色與深度感應器範圍:1.2m - 4m
資料串流	深度感應器: 320×240 @ 30 fps
	彩色感應器: 640×480 @ 30 fps
	聲音規格:16 bit @ 16kHz
骨架追蹤系統	可同時辨識 6 人，包含 2 人的動作追蹤
	每人可追蹤 20 個點
	能讓 Xbox LIVE Avatars 虛擬人物與使用者動作同步
聲音系統	支援遊戲語音交談

	具回音消除功能
	支援多國語言

目前捕捉深度影像的方法，大致上可分為「接觸式」與「非接觸式」的方法 [12]，接觸式指透過實際接觸來獲得深度資訊，如精密工業常使用的三次元量床 (coordinate measuring machine)，非接觸式是利用透過光線、音波等方式去測量，現在常使用的如三角測距(triangulation)、結構光掃描(structured light scanning)、時差測距(TOF, time of flight)。Kinect 使用的偵測方法為「光學編碼」(light coding) 的技術，光學編碼主要利用雷射光散斑(laser speckle)的方式對空間的物體做編碼。當雷射光通過鏡頭表面時，透過鏡頭前的光柵產生光學上所謂的「繞射」(diffraction)，產生隨機的反射斑點，稱之為散斑。散斑具有高度的隨機性，也會隨著移動物體的距離做變化，空間中任兩處的散斑都會不同，等於將整個空間標上記號。最後經由紅外線 CMOS 影像感應器去記錄下空間中的斑點，由晶片計算後產生深度影像。

3.2 Kinect SDK 介紹(Introduction for Kinect SDK)

Kinect for Xbox 360 主要是應用於 Xbox 遊戲機上面，若要把 Kinect 連接到電腦上做應用，需要透過一些平台來驅動 Kinect，在這裡我們使用「Kinect for Windows SDK」。其他也有類似 Kinect SDK，但非官方的驅動平台，如 OPENNI。

利用 Kinect 搭配 Kinect SDK，主要可以實行下面幾種研究。

- a. 針對位於 Kinect 感應器前方的人做骨骼追蹤
- b. 彩色照相機與描述物體與相機距離的景深資訊
- c. 音頻資訊的處理

我們的研究主要是針對 b 的資訊做處理。

Kinect SDK 的運作環境主要是利用 Visual Studio 2010 搭配 C++(或者是 C#)，且必須安裝於 Window 7 的環境，主要原因是為搭配 Windows 7 內建的多媒體

API(Application Programming Interface)來使用。其他詳細的硬體需求與安裝過程可參考[11][13]。

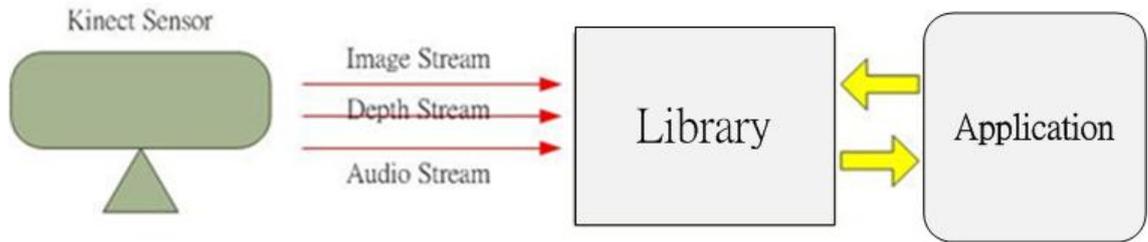


圖 25 Kinect 與軟體程式庫連結關係示意圖

圖 25 為 Kinect 與軟體程式庫之間的關係圖，包含 Kinect 接收到的資料串流可傳送給 NUI(Natural User Interface)，我們可在應用程式的階段針對這些串流資料做應用處理。

為使用 Kinect API 接收感應器所以收集到的資料，必須對 NUI 做初始化的動作(在 C++內為 `NuiInitialize`)，這個動作主要是指定要接收哪些類型資料，例如：彩色影像串流(Stream)、景深影像串流、景深與玩家編號 (Player Index)串流和人體骨骼資料。最後應用程式結束後則會呼叫函式關閉 Kinect(在 C++內為 `NuiShutdown`)。

接下來利用 Kinect SDK 內的函數開啟彩色影像串流。在 C++中是使用 `NuiImageStreamOpen` 這個函式，裡面的參數包含影像串流類型(在這裡選擇彩色影像)、影像串流解析度(一般都選擇 640×480 大小)、緩衝區數量(一般設定為 2 個，如果電腦處理速度不夠快的話，最多可以使用 4 個)等等。

彩色影像的品質可分為兩種。

a. 一般品質

由 Kinect 取得 1280×1024 大小的影像串流，在傳送給 NUI 之前會先轉成 RGB 並且壓縮，在 NUI 接收到後再傳給應用程式。因為被傳送的影像串流有被壓縮過，所以整體的畫格(Frame)的速度可以達到 30 FPS(Frame Per Second)，但影像品質會降低。

b. 高品質

Kinect 的得到的影像串流不做壓縮，直接傳給 NUI。影像品質較好，但畫格速度只有 15 FPS，且需要較大的緩衝空間。

一般來說，我們所使用的彩色影像品質多為 640×480 解析度，也就是經過壓縮後的一般品質的影像。而影像取得方式也分為兩種。

a. 輪詢模型(Polling Model)

在應用程式端一直呼叫 `NuiImageGetNextFrame`，可設定對影像物件的等待時間。如果有新的畫格就讀取並返回等待下次呼叫，或者等待時間到就返回，等待下次呼叫。

b. 事件模型(Event Model)

事件模型對於應用程是有較大的靈活性。只有當 `NuiImageStreamOpen` 得到新的畫格後，才會產生一個事件，並且呼叫 `NuiImageGetNextFrame` 來取得畫格。

這兩種模型以事件模型是較好的選擇，只有事件發生時才會中斷程序去讀取畫格。我們所使用的模型也以事件模型為主。

景深影像同樣使用 `NuiImageStreamOpen` 這個函式，但參數內的影像串流類型必須選擇景深影像串流。每個像素是 16 位元的資料串流，如果景深影像串流是選擇 `NUI_IMAGE_TYPE_DEPTH` 的話，景深資料位於較低的 12 個位元(bits 0-11)，其他 4 個位元是沒有被使用的，且此種型態的影像大小可為 640×480、320×240 80×60 三種大小(一般最常使用的為 320×240);如果景深影像資料型態是 `NUI_IMAGE_TYPE_DEPTH_AND_PLAYER_INDEX`，則較低的 3 位元(bits 0-2)是玩家編號，剩餘的位元是景深資料，而這種資料型態的影像大小只限制於 320×240 和 80×60 兩種大小。

最後，骨骼追蹤不屬於我們的討論範圍，需要更多的參考資料請參考 [11][13][28]。

3.3 OPENNI 介紹及景深感應器的相機參數(Introduction to OPENNI and Camera Calibration of Depth Camera)

OPENNI 是由 PrimeSense 主導的開放平台，但 OPENNI 本身並沒有支援 Kinect，所以必須透過第三方驅動程式 (SensorKinect、NITE) 才能使用 Kinect，所以安裝過程較 Kinect SDK 複雜。與 Kinect SDK 之間也有些微的不同，如：OPENNI 可輸出紅外線影像 (如圖 26 所示) 以及 OPENNI 可用於較多種的作業系統 (如：Windows、Mac OS X、Android 等) 等等，詳細的介紹以及安裝方式可參考 [9]。

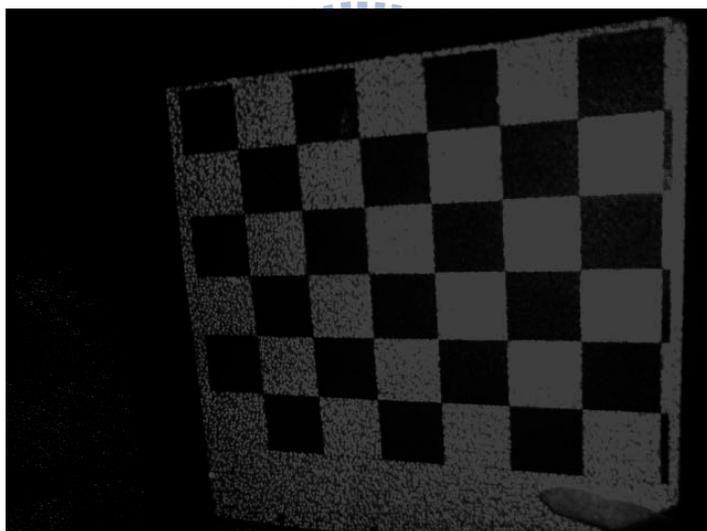


圖 26 紅外線影像

Kinect SDK 並無法獲得景深感應器的相機參數 (參考 4.2 節)。但由於 OPENNI 可輸出紅外線影像，所以我們可以在 OPENNI 的環境下，使用紅外線影像來估測相機參數。首先我們必須遮蔽紅外線發射器，並且利用發熱的光源 (如：鹵素燈泡) 照射校正板，則輸出影像為圖 27。參考 2.1 節的相機校正，我們利用多組測試照片 (如圖 28 所示)，並擷取特徵點 (如圖 29 所示) 來估測相機參數。

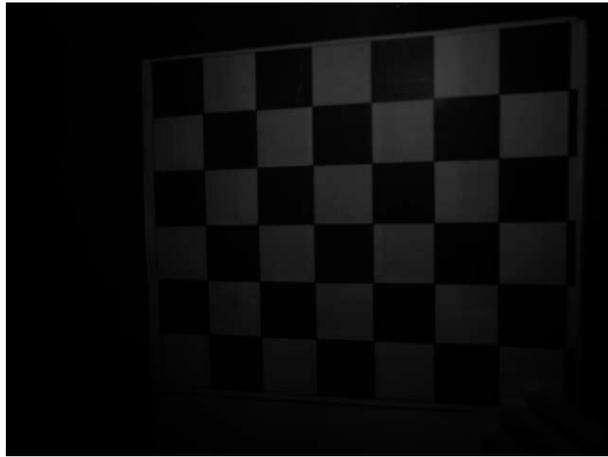


圖 27 鹵素燈照射的影像

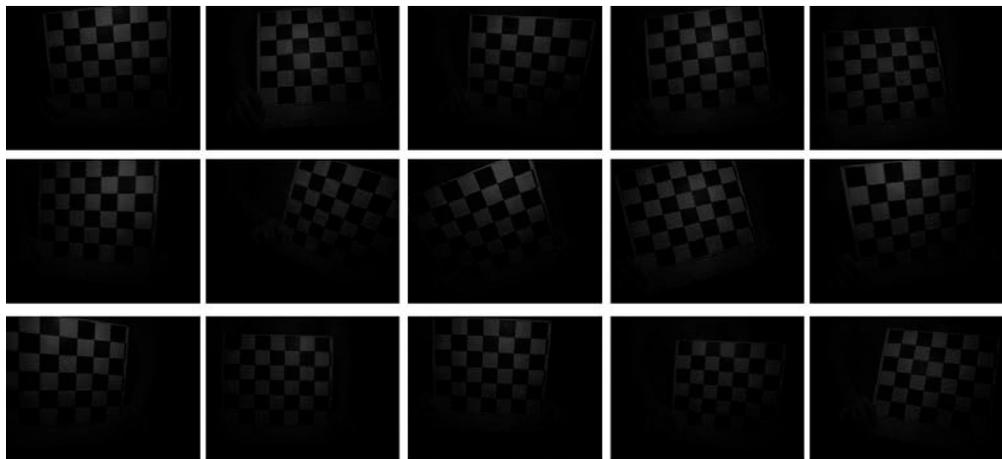


圖 28 紅外線測試影像

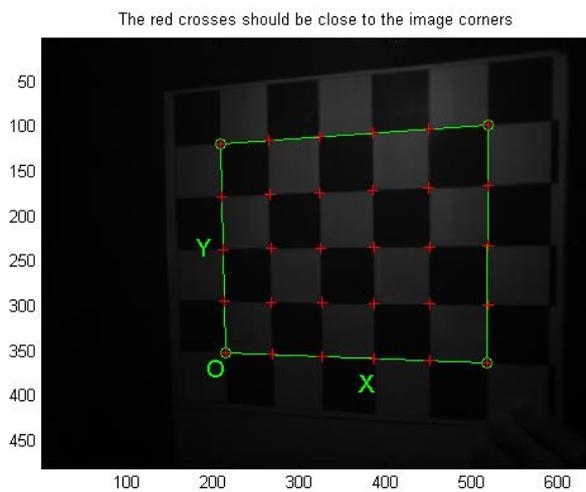
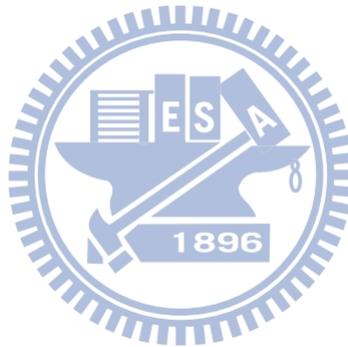


圖 29 特徵點擷取示意圖

表 2 景深感應器相機參數結果

Focal length		Principal point	
f_u (pixel)	f_v (pixel)	u_c (pixel)	v_c (pixel)
595.45	594.59	314.85	240.26
Distortion coefficients			
k_{c1}	k_{c2}	k_{c3}	k_{c4}
-0.0812	0.23488	-0.0038	-0.0005

其結果可參考表 2。且與[8][10]的結果比較，我們的參數結果大致上是合理的範圍內。



第4章 主動式景深攝影機的景深圖改善演算法 (Depth Refinement Algorithm for Active Depth Sensor)

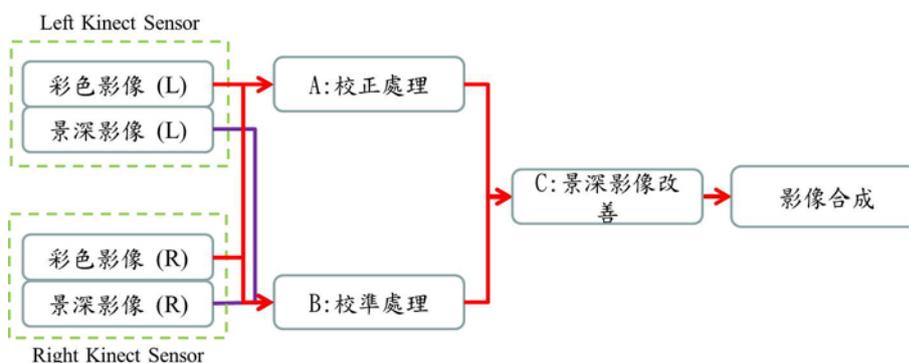


圖 30 演算法流程圖

如圖 30 所示，兩台 Kinect 感應器產生出各自的彩色資訊與景深資訊，所以兩台 Kinect 就有兩組彩色影像與兩組景深影像，分別以左右兩台相機來表示。對於 Kinect 所擁有的問題解析，主要可分成四個部分。

1. 兩台 Kinect 之間的關係連結：參考圖 31，兩台感應器分別放在兩端拍攝場景，所以兩者拍攝出的影像必存在某種幾何關係，且兩台感應器位於不同位置，接收到的光源特性必定有差異，則拍攝出來的影像色彩必定有差異。



圖 31 兩台 Kinect 感應器擺放示意圖

2. Kinect 上彩色相機與景深相機的校準(Alignment)問題：參考圖 31，對於一台 Kinect 感應器上，彩色相機與景深相機位於不同位置上，但我們

目的是得到相同視角的彩色照片與景深照片，所以必須考慮到校準問題。

3. Kinect 無法偵測的破洞(Hole)區域: 參考圖 32 的景深影像，實線方框的部分為 Kinect 感應器計算出合理景深資訊的部分，且這些破洞也可稱為缺陷(Defect)。

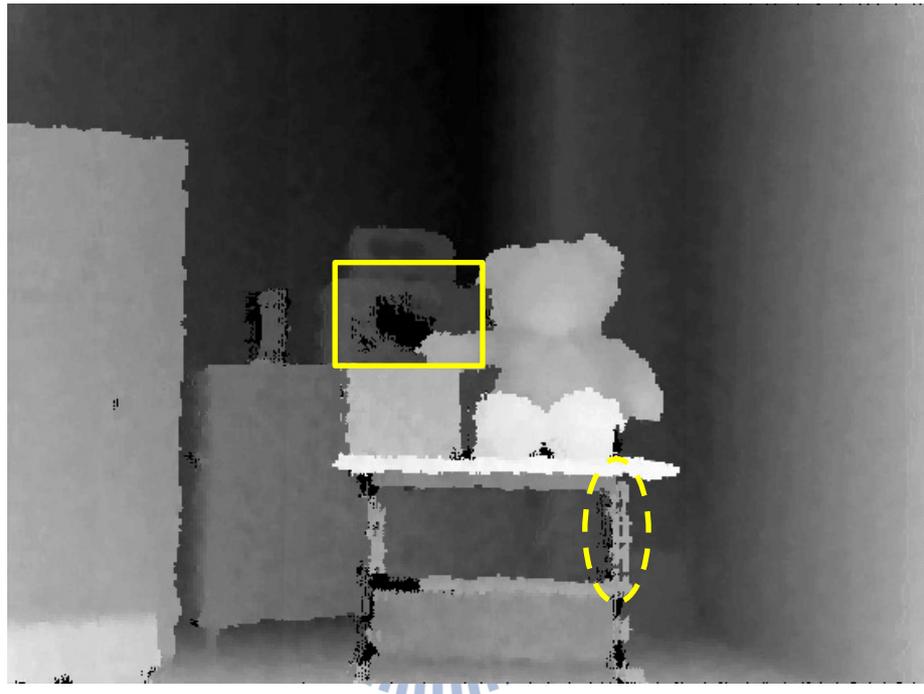


圖 32 景深影像

4. Kinect 偵測錯誤與雜訊問題: 參考圖 32 的景深影像虛線圓圈的部分，可看到 Kinect 內部晶片計算錯誤，導致桌腳部分景深不一致;並且在桌角的邊緣也被雜訊影響，導致該地方凹凸不平。

這些問題可在圖 30 的流程圖中的各種處理獲得改善。

問題 1 在「校正處理」(Calibration Processing)經過色彩校正、相機校正、影像校正等處理後，可以獲得兩台 Kinect 的 3D 幾何關係，並且可以調整兩台 Kinect 影像上的差異。這部分與彩色影像較相關，在圖 30 中我們以紅色實線表示校正處理的輸入影像為兩張彩色影像。

問題 2 在 Kinect 的感應器位置架構上是顯而易見的問題，但又不易解決。在「校準處理」(Alignment Processing)的過程，可以解決彩色影像與景深影像不匹配(Mismatch)的狀況。

問題 3 與問題 4 透過「景深影像改善」(Depth Image Refinement)的處理，可以做到消除雜訊、補洞、修補邊界的效果，使得最後的景深影像可以更好。

4.1 彩色影像校正(Color Image Calibration)

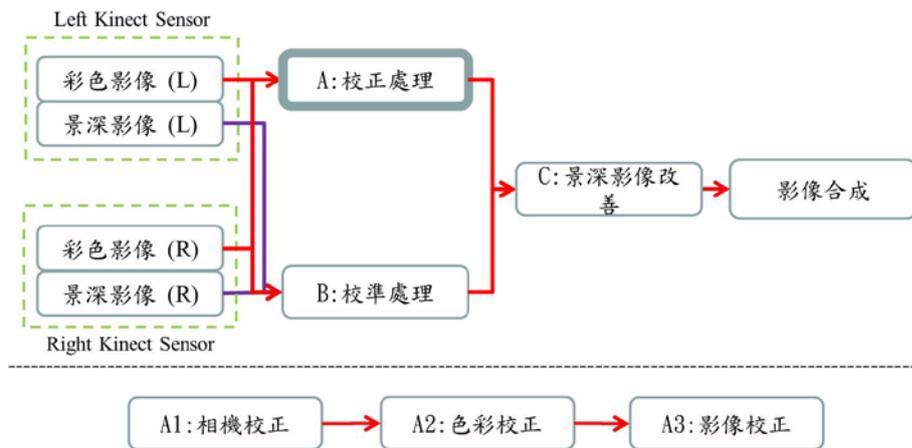


圖 33 校正處理流程圖

如圖 33 所示，在校正處理的部分主要可解決「兩台 Kinect 之間關係連結」的問題，而這個問題又可以細分成兩個子問題(Sub-Problem):

- a. 兩台 Kinect 之間的 3 維空間幾何關係
- b. 兩台 Kinect 顏色一致性的問題

在 2.1 節提到透過 A1 步驟的相機校正，可以得到該台相機的內外部參數。透過各自的相機參數我們知道每一台相機當時在空間中的位置關係，除此之外，在 2.2 節中 A3 步驟的影像校正，可以讓兩台相機在 3 維空間中為平行，且讓兩台相機的內部參數與旋轉矩陣皆相同。這兩步驟可以得到兩台 Kinect 在 3 維空間中的關係，進而解決子問題 a。

在 2.3 節中，A2 步驟的色彩校正，主要是針對相機接收光源特性問題，導致影像亮度差異所做的校正。



圖 34 Kinect 在相同光源下影像色彩差異示意圖

圖 34 可明顯看到，使用同一台 Kinect 在相同的光源下，不同視角會得到亮度明顯不同的影像，所以使用兩台 Kinect 來做自由視點的合成，子問題 b「兩台 Kinect 顏色一致性的問題」是必須被考慮的，否則在虛擬視點的合成影像會顯得不自然。

4.2 影像校準 (Alignment of Depth Image and Texture Image)

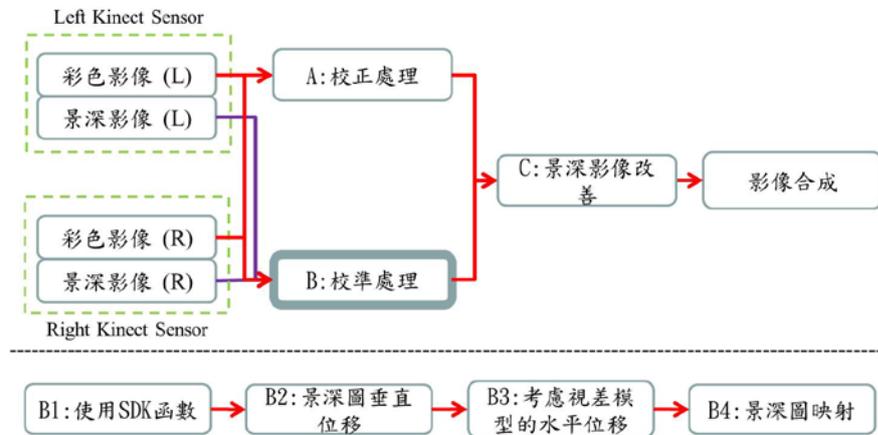


圖 35 校準處理流程圖

圖 35 的校準流程圖，主要是要解決景深相機跟彩色相機位於不同位置的問題。此外因為彩色相機與景深相機之間的焦距也不相同(如圖 36 與圖 37 所示)，所以整個「Kinect 上 RGB 相機與景深相機的校準(Alignment)問題」可以再細分成三個子問題：

- 彩色相機與景深相機的焦距不同
- 彩色相機與景深相機的水平位移差
- 彩色相機與景深相機的垂直位移差



圖 36 A:彩色影像 B:對應的景深影像



圖 37 由圖 36 兩張影像作疊合

a、b、c 三個問題最直接的解決辦法是利用 3 維轉換映射(3D Warping)。利用景深相機的相機參數跟彩色相機的相機參數，兩者間的幾何關係，把景深影像投影到彩色相機的位置。兩種影像皆可用 2.1 節的方式去求得參數。取得彩色相機的相機參數並不是太大的問題，比較困難的是景深感應器的相機參數，如圖 38 所示。



圖 38 A:校正版在景深影像 B:校正板在彩色影像

可以明顯看出在景深圖內，因為校正版是一塊平板，所以擁有相似的景深值。而作相機校正必須去取得校正板上的特徵點才能去做估測，很明顯地，在圖 38 A 並無法取得適合的特徵點，所以無法做相機校正。



圖 39 A:挖洞校正版在彩色影像 B:挖洞校正版在景深影像

圖 39 中利用挖洞的校正版來做相機校正，對應到景深圖，可觀察到洞的出現，然而左邊彩色影像的洞是近乎矩形的洞，可是在景深圖內卻變成不規則的方形洞，角落的部分是不完全的直角，雖然比圖 38 更容易找到特徵點的所在，不過因為無法準確抓的特徵點位置，得到的相機參數仍不準確。所以我們利用多個步驟來解決這個問題，包含利用 SDK 的函式(B1)來解決兩台相機焦距長不同的問題(子問題 a)、調整垂直位移差(B2)來解決垂直位移差(子問題 b)、考慮視差模型(B3)來解決水平位移差(子問題 c)。

4.2.1 使用 Kinect SDK 函數的初步校正(Initial Calibration by Using Kinect SDK Function)

首先考慮景深圖物體與彩色圖物體大小不一的問題，這牽涉到相機內部參數的焦距長，在無法得知景深相機的相機參數時，我們利用 Kinect SDK 內建的一個函式” NuiImageGetColorPixelCoordinatesFromDepthPixel” (C++)，這函數的用

處可從它字面上得知，也就是對於景深圖的每個像素點可得知他對應在彩色圖上的座標位置。

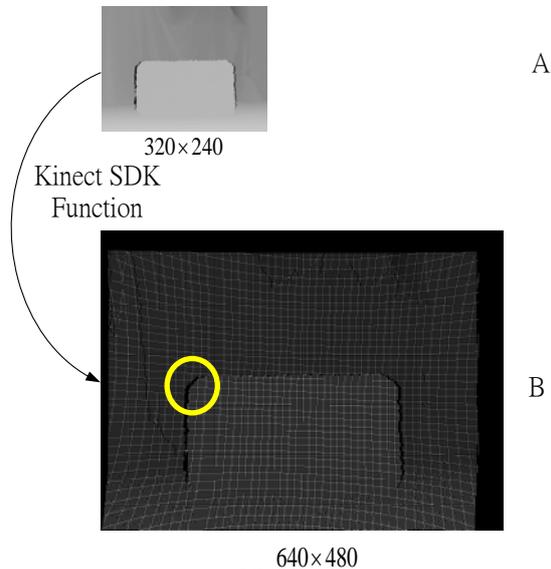


圖 40 利用 Kinect SDK 函式做初步校正

參考圖 40 A，原始輸入景深影像為 320×240 大小，透過 Kinect SDK 函式，我們可以把這張圖投影到跟彩色影像(640×480)一樣大小的。圖 40 B 為投影後的圖，其中裡面一條條白色的線代表是原始景深圖投影過去的景深資訊，相反的黑色部分就是沒有景深資訊的地方。

最後，我們必須針對缺少景深資訊的地方做填值的動作，參考圖 40 B 實線圈圈在 MATLAB 內的數值(如圖 41 所示)。

可以明顯看到大部分沒有景深資訊的點，周圍都有已知景深資訊點，所以我們直接選取周圍已知景深資訊點的最大值去做填補動作。在圖 41 中間有一大塊沒有景深資訊的洞，這就是遮蔽區域，不用去做特別的填補，而填補完的影像則如圖 42 所示。

最後參考圖 43 把景深影像與彩色影像疊合，其中紅色的區塊為遮蔽區域。

130	131	132	133	134	135	136	137	138	139	140	141	142	143	144
250	143	0	143	143	0	143	0	144	0	0	0	0	0	0
251	143	0	143	0	0	0	0	0	0	0	0	0	0	0
252	0	0	143	0	143	0	143	0	0	0	0	0	0	0
253	143	0	143	143	0	0	0	0	0	0	0	0	0	0
254	0	0	0	0	0	0	0	0	0	0	0	0	0	0
255	143	0	143	143	0	0	0	0	0	0	0	180	0	0
256	0	0	0	0	0	0	0	0	0	0	0	0	0	0
257	143	0	143	0	0	0	0	0	0	0	0	180	0	0
258	0	0	0	0	0	0	0	0	0	0	0	0	0	0
259	143	0	143	143	0	0	0	0	0	0	180	180	0	0
260	0	0	0	0	0	0	0	0	0	0	0	0	0	0
261	143	0	143	143	0	0	0	0	0	0	0	180	0	0
262	0	0	0	0	0	0	0	0	0	0	0	0	0	0
263	143	0	143	0	0	0	0	0	0	0	0	181	0	0
264	143	0	143	0	0	0	0	0	0	0	181	181	0	0
265	0	0	0	0	0	0	0	0	0	0	0	0	0	0
266	143	0	144	0	0	0	0	0	0	0	181	181	0	0
267	0	0	0	0	0	0	0	0	0	0	0	0	0	0
268	144	0	144	0	0	0	0	0	0	0	180	180	0	0
269	0	0	0	0	0	0	0	0	0	0	0	0	0	0
270	144	0	143	0	0	0	0	0	0	0	180	180	0	0
271	0	0	0	0	0	0	0	0	0	0	0	0	0	0
272	144	0	143	0	0	0	0	0	0	0	180	180	0	0
273	0	0	0	0	0	0	0	0	0	0	0	0	0	0
274	144	0	143	0	0	0	0	0	0	0	0	180	0	0
275	144	0	144	0	0	0	0	0	0	0	0	180	0	0
276	0	0	0	0	0	0	0	0	0	0	0	0	0	0
277	144	144	0	0	0	0	0	0	0	0	180	180	0	0

圖 41 圖 40 的一角在 MATLAB 的數值圖

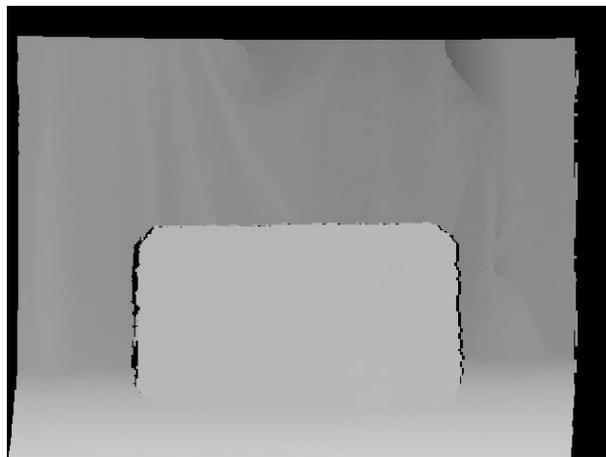


圖 42 填補後的影像



圖 43 彩色影像與景深影像疊合圖

圖 43 可看出，不管是水平或者垂直距離，景深圖對於彩色圖而言還是有水平跟垂直的位移差。在[21]中提到這個函式已經考慮如內部參數、兩種相機間的視差和鏡面扭曲(Lens Distortion)等因素去做映射。可是從影像的結果並不符合我們預期。

圖 44 中， C_{cmos} 指紅外線 CMOS 影像感應器、 C_{rgb} 指彩色相機、 f 指焦距長、 I_{depth} 虛線表示景深影像的平面但不匹配彩色影像、 I_{rgb} 表示彩色影像平面、 b 是基線長。我們預期 Kinect SDK 函式可以把景深影像從 C_{cmos} 投影到 C_{rgb} ，參考圖 43 的結果，Kinect SDK 函式做的投影結果並不如我們預期。

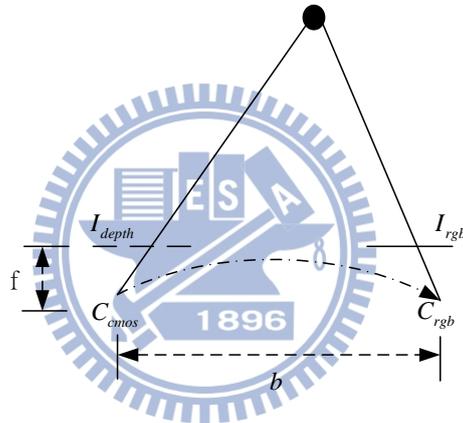


圖 44 Kinect SDK 函式預期的映射示意圖

我們可以想像 Kinect SDK 函式的作用，是讓 C_{cmos} 投影到虛擬相機的位置，如圖 45 所示。

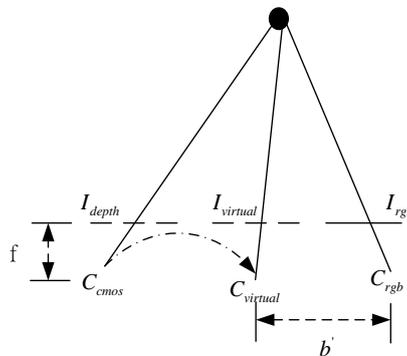


圖 45 Kinect SDK 函式映射至虛擬相機示意圖

其中 $C_{virtual}$ 是虛擬相機， b' 是虛擬相機與彩色相機之間的基線長， $I_{virtual}$ 虛線是指彩色影像平面的物體與虛擬影像平面的物體不匹配。

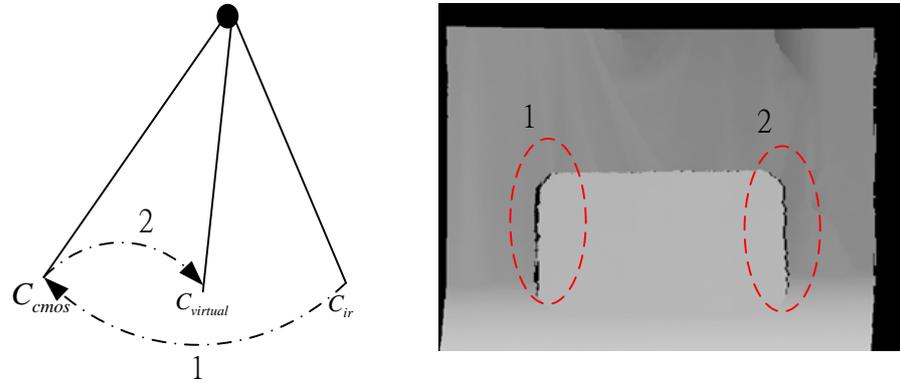


圖 46 相機關係造成各自遮蔽現象的示意圖

最後討論景深影像遮蔽區域與相機位置之間的關係。在圖 46 中，編號 1 的遮蔽區域可視為，景深影像從紅外線發射器投影到紅外線 CMOS 影像感應器，所以在影像左邊產生遮蔽效應；編號 2 的遮蔽區域可視為，紅外線 CMOS 影像感應器投影到虛擬相機所產生的遮蔽效應。

4.2.2 景深影像垂直校正(Vertical Calibration on the Depth Image)

圖 43 中，物體在垂直方向有不匹配的情況發生，這垂直誤差經由觀察多組影像後，可判定這個誤差是幾乎為一個常數值，所以我們利用兩種方法求出最佳的位移常數並且做驗證。這兩種方法分別為基於邊緣估測(Edge-Based Estimation)和基於陰影估測(Shadow-Based Estimation)。

A. 基於邊緣估測

參考[22]的方式來求得邊緣。其做法利用原始的景深圖套用式(23)來求得初始的邊緣影像。

$$IsEdge(x, y) = \begin{cases} 1 & |d(x, y) - d(x \pm 1, y \pm 1)| \geq d_m \\ 0 & |d(x, y) - d(x \pm 1, y \pm 1)| < d_m \end{cases} \quad (23)$$

其中， $d(x, y)$ 為該像素點對應的景深值， d_m 為一個臨界值(我們的實驗 d_m 取 100 公厘)。圖 47 為利用式(23)所得到的初始邊緣影像。

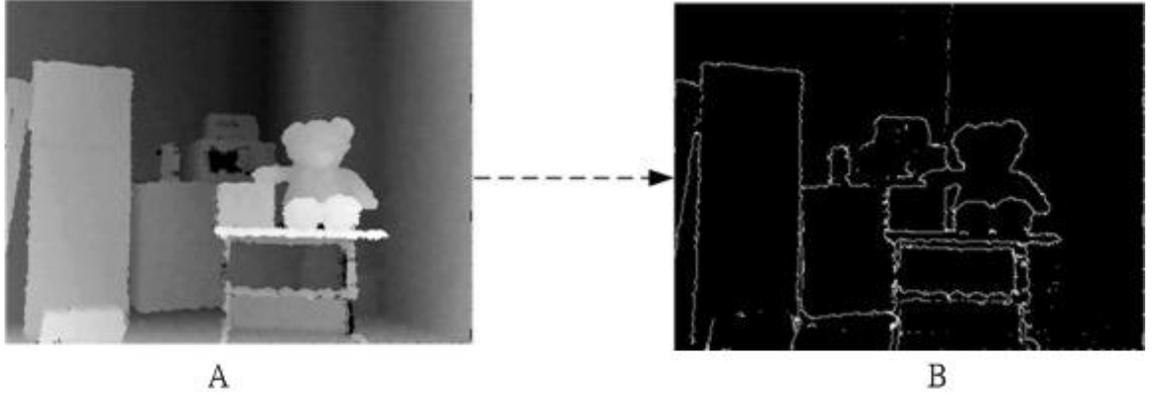


圖 47 A:景深影像 B:初始邊緣影像

我們觀察到初始邊緣影像跟彩色影像的邊緣實際上還是有差異的，所以我們根據每個邊緣點去做垂直位移的調整。對於每個邊緣點 $x_d(x, y)$ 我們假設其 $x_d(x, y+k)$ ($k = -6, -5, \dots, 5, 6$) 都是可能的候選者(Candidate)，對應於彩色點 $x_c(x, y)$ 在上下兩個方向各選擇 12 點來計算成本函數。參考式(24)。

$$\begin{aligned}
 CV(k) &= UD(k) \cdot UW(k) + DD(k) \cdot DW(k) \\
 UD(k) &= \left(\frac{\sum_{m=k+1}^{12+k} C(x, y+12+m)}{12+k+1} - \frac{\sum_{m=-12}^{12} C(x, y+m)}{25} \right)^2 \\
 DD(k) &= \left(\frac{\sum_{m=k+1}^{12} C(x, y+m)}{12-k} - \frac{\sum_{m=-12}^{12} C(x, y+m)}{25} \right)^2 \\
 UW(k) &= \frac{12+k}{25} \\
 DW(k) &= 1 - UW(k)
 \end{aligned} \tag{24}$$

其中， $CV(k)$ 是成本函數， $C(x, y)$ 是像素點在 (x, y) 的強度值。 $UD(k)$ 為上方的差異(up-difference)， $DD(k)$ 為下方的差異(down-difference)， $UW(k)$ 為上方的權重(up-weighting)， $DW(k)$ 為下方的權重(down-weighting)。

最後的結果影像如圖 48 所示。

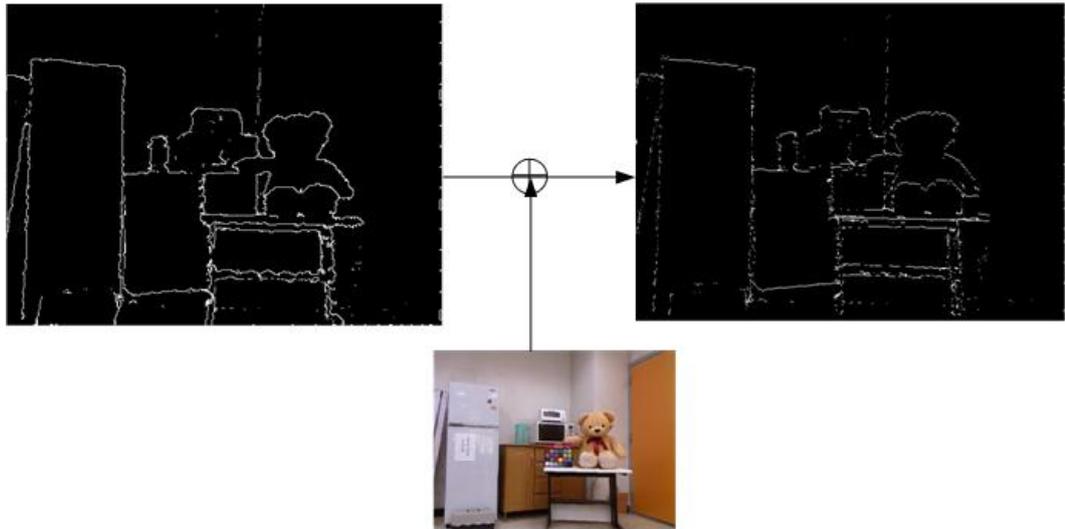


圖 48 初始邊緣影像參考彩色影像產生改善後邊緣影像

我們將初始邊緣影像的每個邊緣點當作一個特徵點，並剔除不合理的成本函數，只取 $k > 0$ 的點來計算平均垂直位移量。最後圖 48 中的影像平均垂直位移量為向上位移 3.16 個像素(約 3 個像素)。

B. 基於陰影估測(Shadow-Based Estimation)

對於陰影估測的方法，我們可以想像影像中同一個物體，應該是要對應到差不多的景深值。



圖 49 景深影像疊合彩色影像

參考圖 49，可以想像我們是要統計彩色影像對應到多少正確的景深，也可看成計算物體上所對應到的陰影。我們稱此方法為基於陰影估測。

在圖 49 中我們選用中間藍色板子來做統計，其結果可參考表 3。

表 3 垂直位移量與精準度統計表

Upward Pixels	Accuracy Rate
0	0.9818
1	0.9866
2	0.9894
3	0.9904
4	0.9887
5	0.9854

其中準確率(Accuracy Rate)定義為式(25):

$$Accuracy\ Rate = \frac{Correct\ Depth\ Pixels}{Total\ Pixels} \quad (25)$$

表 3 中可看出垂直位移差以 3 個像素的位移最適當，相對於一開始的影像，垂直位移 3 個像素多 0.0086 的精準度。最後陰影估測與邊緣估測的結果相符合，以垂直位移 3 個像素是最好的結果。

4.2.3 景深影像的水平校正(Horizontal Calibration on the Depth Image)

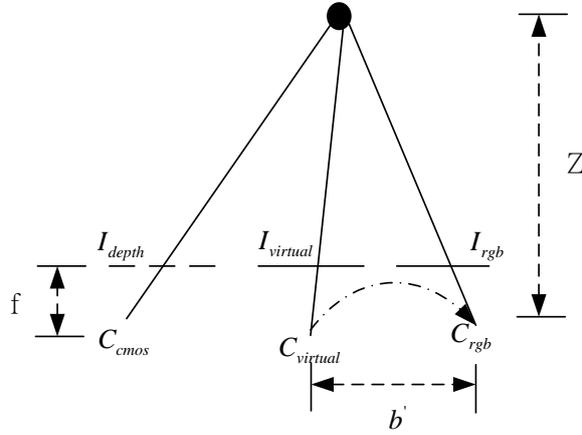


圖 50 垂直校正後的相機示意圖

圖 50 為經過垂直校正後的示意圖，其中 $I_{virtual}$ 代表景深影像與彩色影像物體匹配，在這裡以實線表示之。對於 $I_{virtual}$ 與 I_{rgb} 可利用視差模型來考慮它們之間水平位移的關係，可表達成式(26)。

$$x_{virtual} = x_{rgb} + \frac{f \cdot b'}{Z} \quad (26)$$

其中， $x_{virtual}$ 為 $I_{virtual}$ 上的一點， x_{rgb} 為 $x_{virtual}$ 在 I_{rgb} 上的對應點， f 是焦距長， b' 為虛擬相機跟彩色相機之間的基線長， Z 為景深值。

因為我們並不知道 b' 確切的值是多少，所以我們需利用多張影像來估測 b' 的值。假設 D 為視差值，則可導出式(27)。

$$b' = \frac{D \cdot Z}{f} \quad (27)$$

參考圖 51，考慮校正板右邊虛線的遮蔽區域做為視差值的基準，主要原因可參考圖 52。1 代表 B1 步驟中，CMOS 影像感應器接收紅外線發射器的訊號；2 代表 B2 步驟中，從 CMOS 影像感應器投影到虛擬相機位置；3 代表 B3 步驟中，從虛擬相機位置投影到彩色相機位置。我們現在做的事是從 1 步驟到 2 步驟再到

3 步驟，等效於做第 4 步驟(從紅外線發射器投影到彩色相機位置)，從 4 輸出影像考慮遮蔽效應，只會有左邊的遮蔽區域，所以 2 步驟到 3 步驟，合理來說並不會出現右邊的遮蔽區域。



圖 51 基線估測測試影像

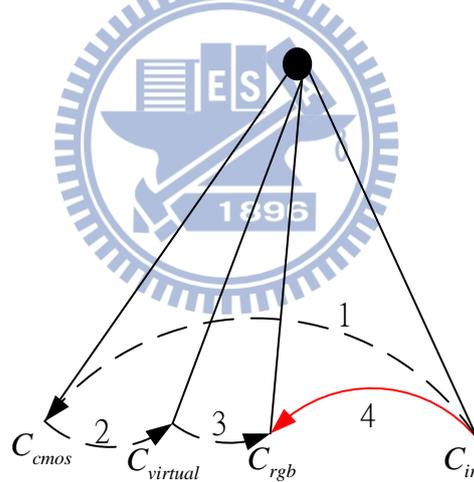


圖 52 相機投影示意圖

所以參考表 4 配合式(27)，算出不同距離的基線，考慮誤差則取其平均值。

表 4 基線估測表

Distance(mm)	Disparity(pixel)	Baseline(mm)
900	7	12.16
1050	6	12.16

1200	5	11.54
1350	4	10.38
1500	4	11.54
Average Baseline Length : 11.56		

最後考慮視差模型，並利用估算出來的基線長套用到圖 51，最後結果如圖 53 所示，利用陰影估測的方式估算此圖，結果為 0.9987，相較於單純做垂直位移（如表 3 所示），會擁有更好的結果。



圖 53 考慮視差模型做水平位移後的影像

4.3 景深影像改進(Depth Image Improvement)

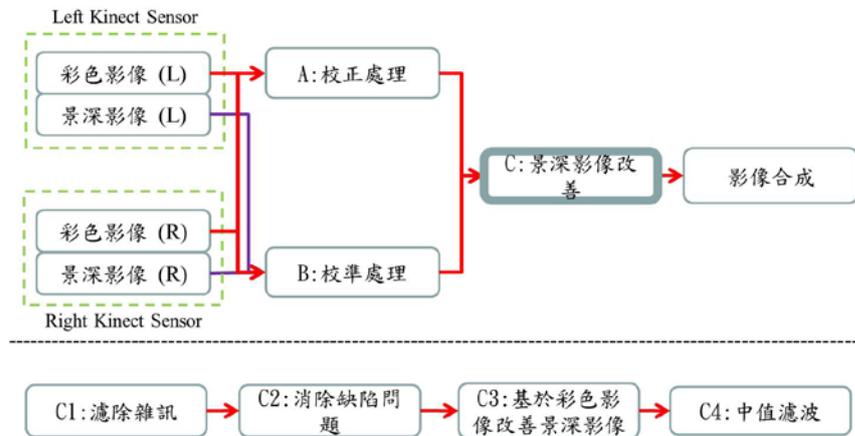


圖 54 影像改善與合成流程圖

參考圖 54，景深影像改善步驟，為改善景深影像的邊緣雜訊及遮蔽區域。這些雜訊以及遮蔽區域在最後的影像合成步驟，都會影響最後合成影像的品質。

我們把校準後的影像問題，再細分成 3 個子問題。

- 景深影像的邊緣雜訊
- 補「洞」的問題
- 景深影像細部區域修補問題

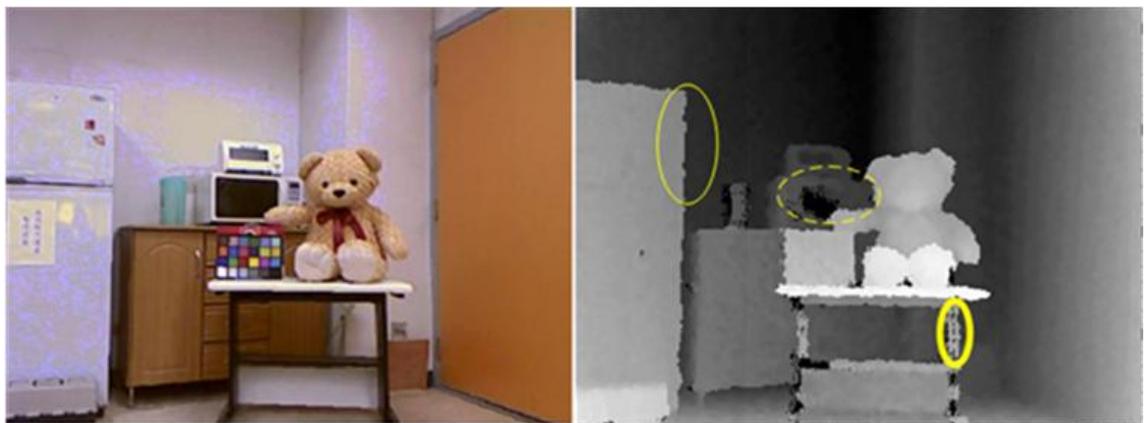


圖 55 影像改善須改善的問題示意圖

如圖 55 所示，問題 a 可參考細實線的部分，在冰箱的邊緣不是直線，會有不規則的凹凸地方，像這種邊緣雜訊我們利用「濾除雜訊」(C1)的步驟來解決；

問題 b 是在虛線部分(彩色影像上烤箱)會有一大塊破洞，這部分因為 Kinect 無法計算出合理的景深值，所以我們使用「消除缺陷問題」(C2)來做填補;問題 c 是粗實線的部分，並且可對照彩色影像看出桌腳有瑕疵，實際上彩色圖內小熊的大小跟景深圖內小熊的大小也不吻合，在這裡我們在「考慮彩色影像改善景深影像」(C3)步驟考慮彩色影像來做加強後;最後針對邊緣的胡椒鹽(Pepper and Salt)雜訊使用「中值濾波」(C4)，可得到明顯的改善。

4.3.1 聯合雙向濾波器濾除景深影像雜訊(Noise Filtering by Using Joint Bilateral Filter on the Depth Image)

對於濾除邊緣雜訊的部分，我們使用聯合雙向濾波器(Joint Bilateral Filter (JBF))來濾除。不同於其他的平滑(Smooth)濾波器，會導致景深影像在不連續的地方模糊掉，雙向濾波器在邊緣的部分有很好的保留性[23]。它利用像素點之間幾何上的接近程度(即域項(Domain Term))和像素點之間的光度(photometric)的相似程度(即範圍項(Range Term))，來做為保持影像邊緣的根據。

我們所使用的聯合雙向濾波器[24][25]，除對於景深影像考慮它們的域項以及範圍項以外，我們額外考慮彩色影像的範圍項，其中彩色影像我們使用 3 個向度影像(即 RGB)。所以聯合雙向濾波器可表達成式(28)。

$$\begin{aligned}\hat{D}^p &= \frac{R(p)}{k^p} \sum_{q \in \Omega^p} D^q f(p, q) h(\|D^p - D^q\|) r(\|I^p - I^q\|) g(\|I^p - I^q\|) b(\|I^p - I^q\|) \\ k^p &= \sum_{q \in \Omega^p} f(p, q) h(\|D^p - D^q\|) r(\|I^p - I^q\|) g(\|I^p - I^q\|) b(\|I^p - I^q\|)\end{aligned}\quad (28)$$

其中， \hat{D}^p 為濾波後新的像素值， k^p 是正規項， q 為在 Ω^p 區間內的參考點(Reference Pixel)， D^p 是原始像素值， $f(p, q)$ 為在高斯模型(Gaussian Model)下的域項， $h(\cdot)$ 為景深影像在高斯模型下的範圍項，而 $r(\cdot)$ 、 $g(\cdot)$ 、 $b(\cdot)$ 分別為彩色影像 3 個向度在高斯模型下的範圍項。而在這裡為考慮未知景深資訊點，我們利用 $R(p)$ 來做決定，其中 $R(p)$ 可用式(29)來表示。

$$R(p) = \begin{cases} 0, & p \in \text{unknown} \\ 1, & \text{otherwise} \end{cases} \quad (29)$$

對於 q 是未知景深資訊點的話，我們則不考慮該點的影響，如式(30)所示。

$$f(p, q) = 0 \quad \text{if } q \in \text{unknown} \quad (30)$$

只要把其中一項改成 0，即可消除該點的影響。最後濾波後的結果如圖 56 所示。

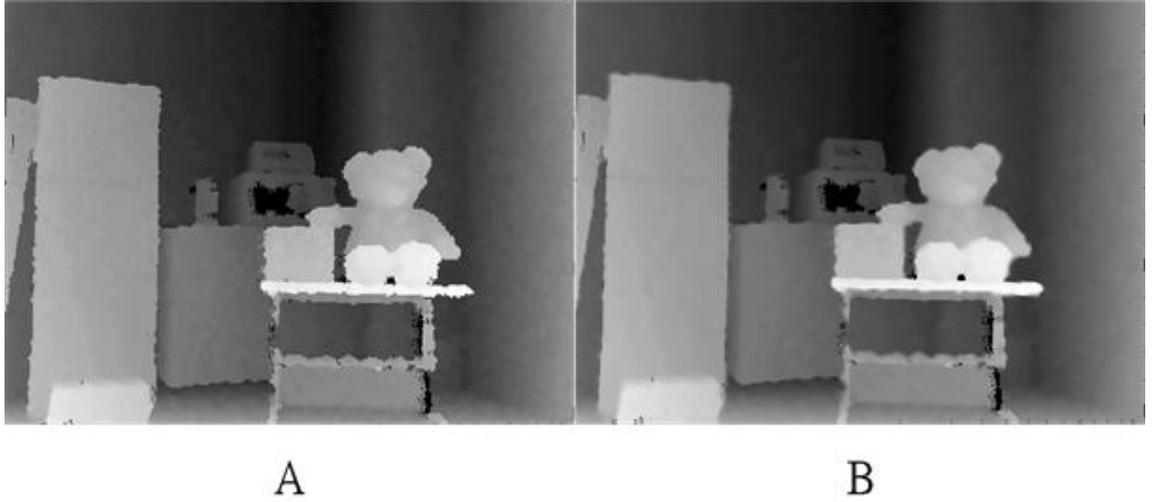


圖 56 A: 原始影像, B: 聯合雙向濾波後影像

最後我們針對 3 個向度的 RGB 彩色影像與 1 個向度的灰階 (Gray Level) 影像，使用聯合雙向濾波器後之間的差異做比較與討論。

我們使用的聯合雙向濾波器在灰階影像的公式如式(31)所示。

$$\begin{aligned} \hat{D}^p &= \frac{R(p)}{k^p} \sum_{q \in \Omega^p} D^p f(p, q) h(\|D^p - D^q\|) \text{gray}(\|I^p - I^q\|) \\ k^p &= \sum_{q \in \Omega^p} f(p, q) h(\|D^p - D^q\|) \text{gray}(\|I^p - I^q\|) \end{aligned} \quad (31)$$

其中， $\text{gray}(\|I^p - I^q\|)$ 為灰階影像在高斯模型下的範圍項。在這裡我們必須在相同標準差影響的情形下來做兩者間的討論。

$$\begin{aligned} h(\cdot) &= r(\cdot)g(\cdot)b(\cdot) \\ \exp\left(-\frac{c^2}{\text{sigma_gray}^2}\right) &= \exp\left(-\frac{c^2}{\text{sigma_r}^2}\right) \exp\left(-\frac{c^2}{\text{sigma_g}^2}\right) \exp\left(-\frac{c^2}{\text{sigma_b}^2}\right) \\ \text{if } \text{sigma} &= \text{sigma_r} = \text{sigma_g} = \text{sigma_b} \\ \rightarrow \text{sigma_gray} &= \frac{\text{sigma}}{\sqrt{3}} \end{aligned} \quad (32)$$

比較兩者差異，如圖 57 與圖 58 所示。在熊頭頂與色板上(紅色虛線圓圈)可看出，彩色影像邊緣效果較清楚而灰階影像的會有點模糊掉。從圖 59、圖 60 的細部比較可以更明顯看到兩者之間的差異。

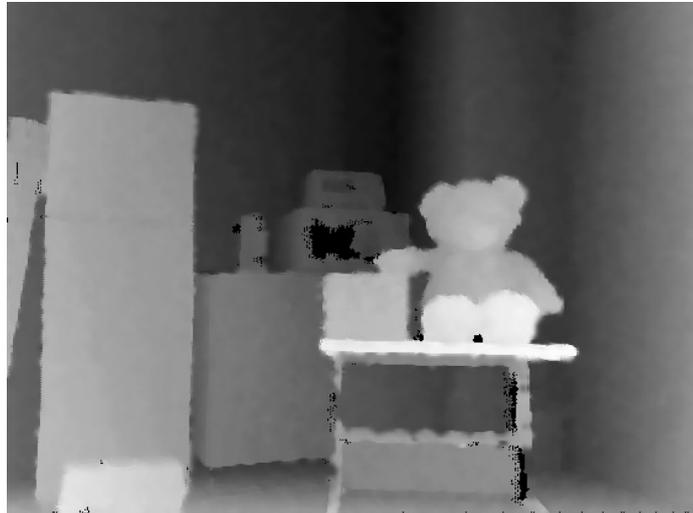


圖 57 聯合雙向濾波使用 1 個通道的灰階影像

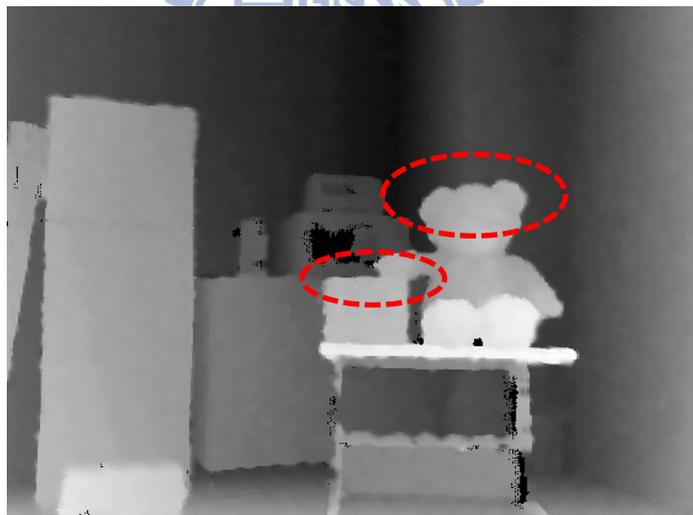


圖 58 聯合雙向濾波使用 3 個通道的彩色影像

<p>Joint Bilateral Filter in gray domain</p>	
--	--

Joint Bilateral Filter in RGB domain	
--	--

圖 59 圖 57、圖 58 細部比較(1)

Joint Bilateral Filter in gray domain	
Joint Bilateral Filter in RGB domain	

圖 60 圖 57、圖 58 細部比較(2)

4.3.2 使用聯合雙向濾波器消除景深影像的缺陷問題(Defects

Reduction Using Joint Bilateral Filter on the Depth Image)

在 4.3.2 節做完濾除雜訊後，我們會針對未知景深資訊點，參考彩色影像來做填補的動作。主要的想法在於如何利用可靠的數值填補未知景深資訊區域。

在這裡使用聯合雙向濾波器來修補未知景深資訊區域[24][25]。初始的想法是利用彩色影像的資訊搭配聯合雙向濾波器，針對可靠性高的點去做填補。所以延用式(28)但除去景深影像的範圍項，則可寫成式(33)。

$$\begin{aligned} \hat{D}^p &= \frac{R(C_{map}, \Omega^p)}{k^p} \sum_{q \in \Omega^p} D^p f(p, q) r(\|I^p - I^q\|) g(\|I^p - I^q\|) b(\|I^p - I^q\|) \\ k^p &= \sum_{q \in \Omega^p} f(p, q) r(\|I^p - I^q\|) g(\|I^p - I^q\|) b(\|I^p - I^q\|) \end{aligned} \quad (33)$$

其中，除 $R(C_{map}, \Omega^p)$ 以外，其他的參數都與式(28)相同，而 $R(C_{map}, \Omega^p)$ 可以定義成式(34)。

$$R(C_{map}, \Omega^p) = \begin{cases} 1, & \text{if } count[\Omega^p \in \text{known-pixel}] / Area(\Omega^p) > th \\ 0, & \text{otherwise} \end{cases} \quad (34)$$

C_{map} 為紀錄每個點是否為可信任點的表， $Area(\Omega^p)$ 為方塊內所有點的數目，最後 th 為臨界值。根據區域內的已知景深點的比例，超過臨界值才判斷 p 點為可信任點，未超過則為不可信任點。

因為只填補可信任點，所以必須用遞迴(iteration)的方式來填補，詳細的流程圖可參考圖 61。圖 61 的 A 步驟中，會針對輸入的景深影像偵測可信任點，並且更新 C_{map} 。圖 61 的 B 步驟中，則參考 C_{map} 針對可信任點做填補。每次填補完檢查是否還有未知景深資訊點，若仍有未知景深資訊點則重複圖 61 的 A、B 步驟直到沒有任何未知景深資訊點。最後每次遞迴的結果與最後得到的輸出影像可參考圖 62、圖 63、圖 64、圖 65 和圖 66，並且每次遞迴填補的部分以實線圈圈表示。

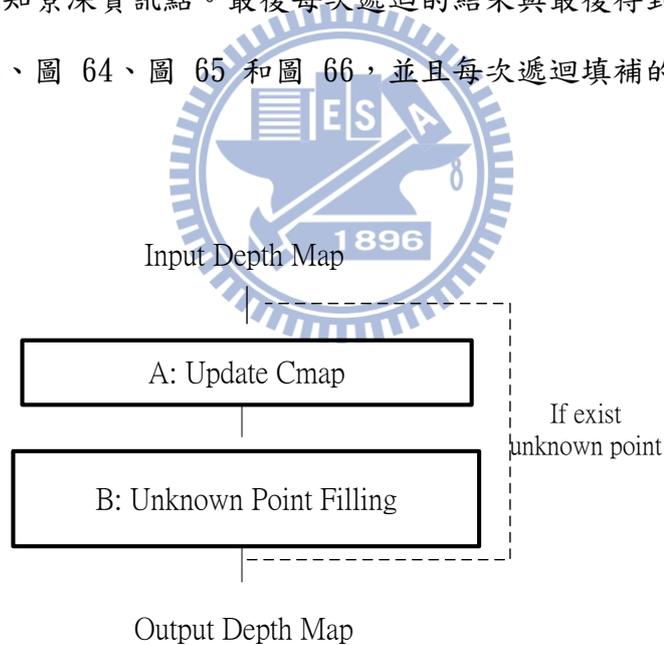


圖 61 遞迴式瑕疵現象消除流程圖

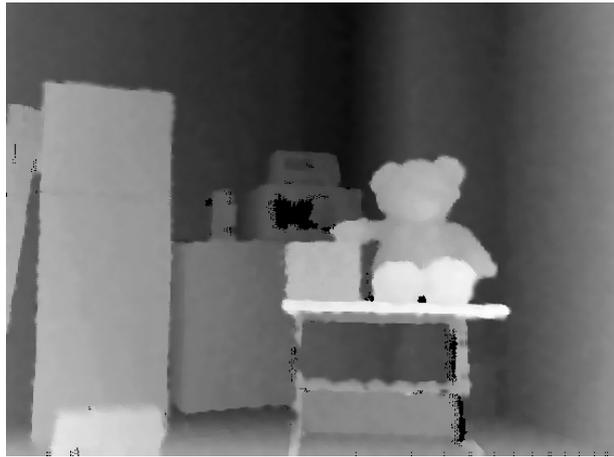


圖 62 原始輸入影像

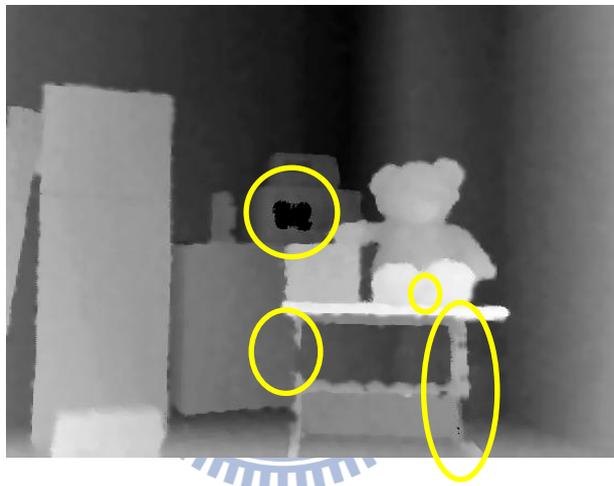


圖 63 第 1 次遞迴

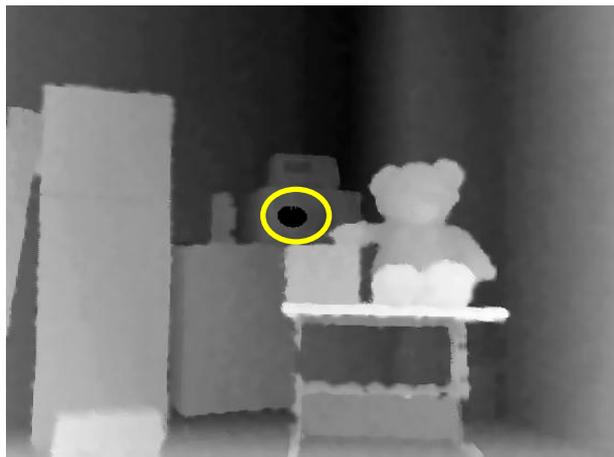


圖 64 第 5 次遞迴

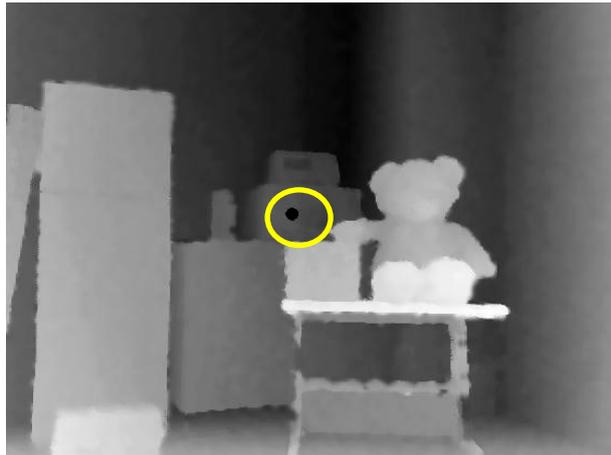


圖 65 第 9 次遞迴

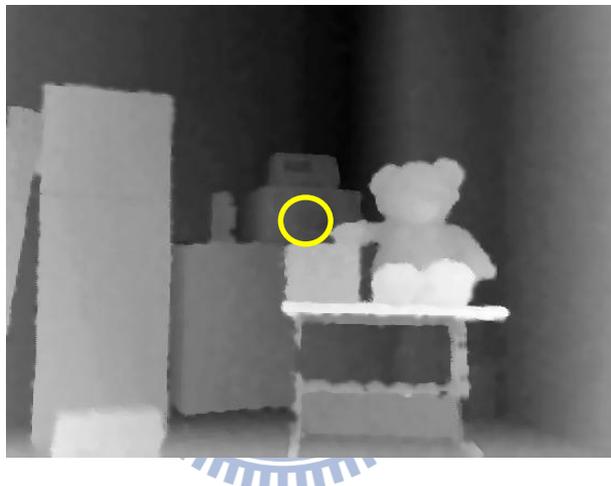


圖 66 輸出影像

4.3.3 基於彩色影像改善景深影像(Depth Image Refinement based on Color Image)

在圖 66 中，大部分邊緣雜訊已經濾除，但少數邊緣凹凸範圍很大的地方仍未獲得完全的改善。參考圖 67 與彩色影像作疊合可更明顯看到，紅色虛線部份的桌腳，景深圖相對於彩色圖少一塊；紅色實線的桌底是因為邊緣凹凸範圍太大，利用濾除雜訊(C1)的方法並無法解決；在紅色點圈的小熊肩膀，因為原始的景深

圖估測不準確導致那邊缺少一塊。所以我們希望參考彩色影像的資訊，對於景深圖內物體景深錯誤的地方再加以改進，藉此最後合成虛擬視點的影像能更完美。



圖 67 圖 66 與對應的彩色影像疊合



圖 68 基於彩色影像改善景深影像演算法流程圖

參考圖 68 的流程圖，基本想法是考慮「多數決」的想法來決定景深值。基於這個想法，我們的演算法步驟如下。

- a. 我們會針對「有問題」的像素點做討論，在這裡「有問題」的像素點是位於景深有明顯變化的邊緣。我們會先對於垂直方向做 1 維向量的搜尋，一旦找到「有問題」的像素點(如式(35))，則記錄該點的位置，並在彩色影像相同的位置框選出一個適當大小的參考方塊，如圖 69 所示。其中在圖 69 A 中，紅色星星代表在景深影像中所找到的參考點(有問題的點)，以其為中心點，框選出一個適當大小

的參考方塊;在圖 69 B 中，根據參考點的位置，對於彩色影像取出一個以參考點為中心的參考方塊。

$$pixel(i, j) = \begin{cases} uncertain, & \text{if } abs(pixel(i, j) - pixel(i-1, j)) > th \\ certain, & \text{if } abs(pixel(i, j) - pixel(i-1, j)) < th \end{cases} \quad (35)$$

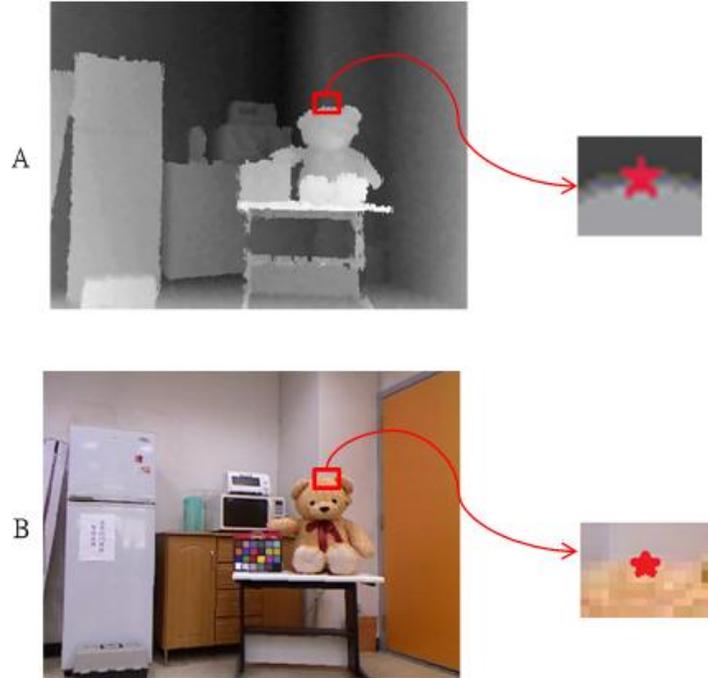


圖 69 A:景深影像的參考點 B:相對於彩色影像取出參考方塊

- b. 針對參考方塊內與參考點(中心點)相似顏色的位置做紀錄(以圖 69 B 為例，參考點位於小熊的頭部為皮膚色，所以我們會取出參考方塊內也是皮膚色的部分做紀錄)。
- c. 最後，返回景深圖的參考方塊，取出 b 步驟所紀錄對應位置的像素點，根據這些點的像素值(0~255)做統計(如圖 70 所示)，根據這些結果，我們會選取出現機率最大(統計個數最多的)的像素值，來做為該參考點的像素值。

做完垂直方向的改善後，必須針對水平方向再做一次改善(a 步驟的垂直搜尋改為水平搜尋，並重複 a、b、c 的步驟)。利用圖 66 作為輸入影像，可得到圖 71 的結果。

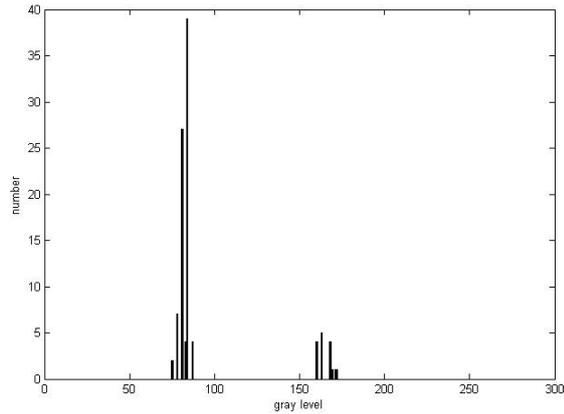


圖 70 像素值統計表



圖 71 輸出影像

c 步驟中是選取出現機率最多的值來填補，在這裡可以選用中值濾波器來做選擇。在大部分的像素點的統計，只會有一個峰值，所以用中值濾波器是另一個合理的選擇。但在圖 70 中會有兩個較明顯的峰值，但因為其值接近，即使選用中值濾波器，還是可以得到合理的數值。

4.3.4 中值濾波(Median-Filtering)

參考圖 71 的結果，雖然大致上整張圖算是有完整恢復。但由於原始輸入的景深圖，某些物體邊緣會超過彩色影像中對應邊緣的位置，所以參考彩色影像修

正時(C3 步驟)，雖然大多數的區域都可以修正到與彩色影像的區域匹配，但少數像素仍會保留，形成胡椒鹽雜訊(Pepper and Salt Noise)。

對於這些雜訊，我們使用中值濾波器(Median Filter)做濾除。參考圖 72 為圖 71 經過中值濾波器後的影像(此圖的濾波方塊大小為 5×5)。



圖 72 中值濾波輸出影像

由圖 72 與圖 71 做比較，明顯看到圖 72 的邊緣部分會較圖 71 的邊緣來的平滑，雜訊的減少更有助於最後的影像合成結果。

第5章 實驗結果與比較 (Experiment Results and Comparison)

本章節主要介紹測試影像的流程設定與模擬結果，並且在 5.1 節至 5.4 節利用 4 組測試影像來呈現。其中在 5.1 節利用一組測試影像針對實驗流程做詳細的介紹，包含參數設定以及使用工具的介紹。

5.1 第 1 組測試影像模擬與結果 (Simulation of First Test Sequence)

我們設置兩台 Kinect 感應器平行擺放，如圖 73 所示。



圖 73 兩台 Kinect 感應器擺放示意圖

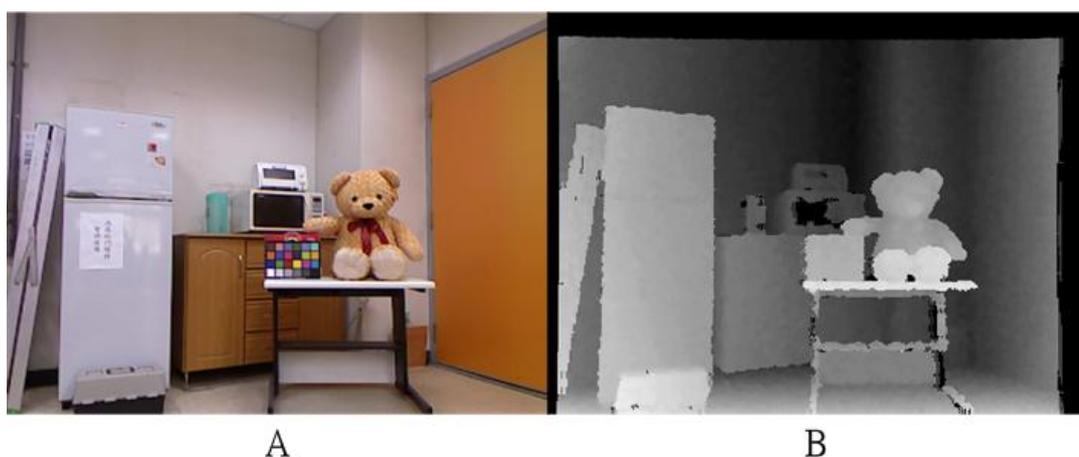
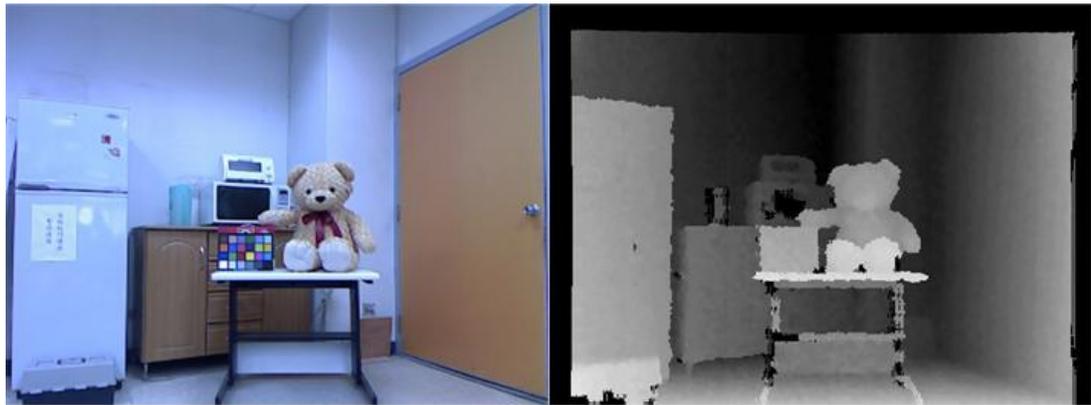


圖 74 A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像



A

B

圖 75 A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像

兩台 Kinect 感應器皆使用 Kinect SDK-v1.0-beta2 做為平台，並且外掛 OPENCV 2.2 來幫助串流資料的存取。此外，在 Kinect SDK 內讀取完資料串流後，我們立即使用「校準處理」(4.2 節)，針對景深影像作校準的動作。最後左右 Kinect 感應器輸出影像分別為圖 74、圖 75 所示。

5.1.1 校正處理

參考圖 30 的演算法流程，針對左右的彩色影像我們進行「校正處理」(4.1 節)。在 A1 步驟的相機校正，我們使用[32]的相機校正軟體，簡單的軟體相機校正使用流程如下。

- a. 把 Matlab 目錄指到相機校正軟體的資料夾底下。
- b. 開啟主要的校正函式-calib_gui(如圖 76 所示)。開啟後有兩種模式，其中，標準模式可用在圖片檔案容量不大或者張數不多的狀況下，把影像存在記憶體中；記憶體節省模式主要是用於影像檔案容量很大或者張數很多的狀況下(在標準模式記憶體會爆滿)，影像一張一張讀入，且永不存入記憶體中。

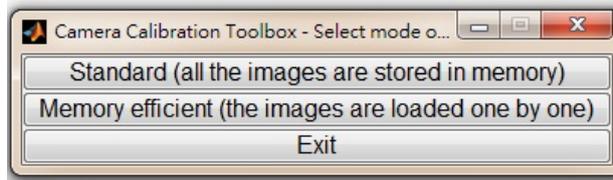


圖 76 校正函式- calib_gui

- c. 參考圖 77 為進入標準模式內的 GUI 介面(在此只介紹至程式估出相機參數完為止，其他的函式功能可參考[32]內更詳細的說明)。一開始會使用「Image names」開始讀檔，在這裡需要特別注意的是檔名命名格式，一般都會命名成:名稱+編號+檔案格式(如: ir1.jpg)。最後讀取到的影像如圖 78 所示。

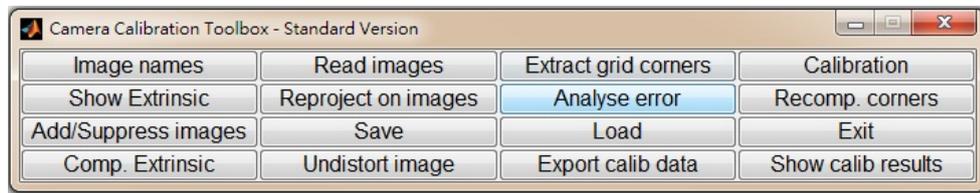


圖 77 標準模式

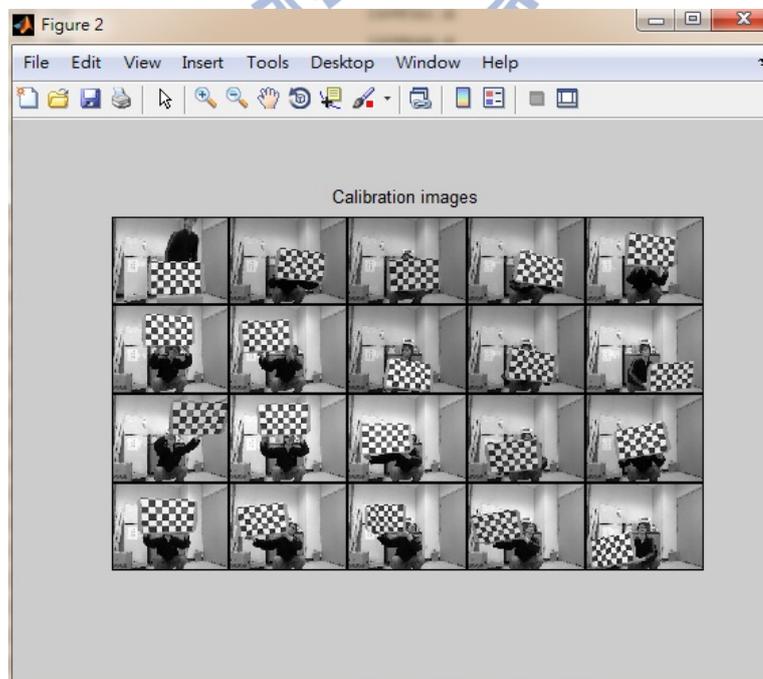


圖 78 校正軟體-影像讀取

- d. 使用「Extract grid corners」擷取特徵點(如圖 79 所示)。在圖 79 中，O 點為世界座標系統定義的原點，X、Y 分別為水平軸方向跟垂直軸方向。(在圖 79 中，校正板格子實際大小為 $8cm \times 8cm$)。

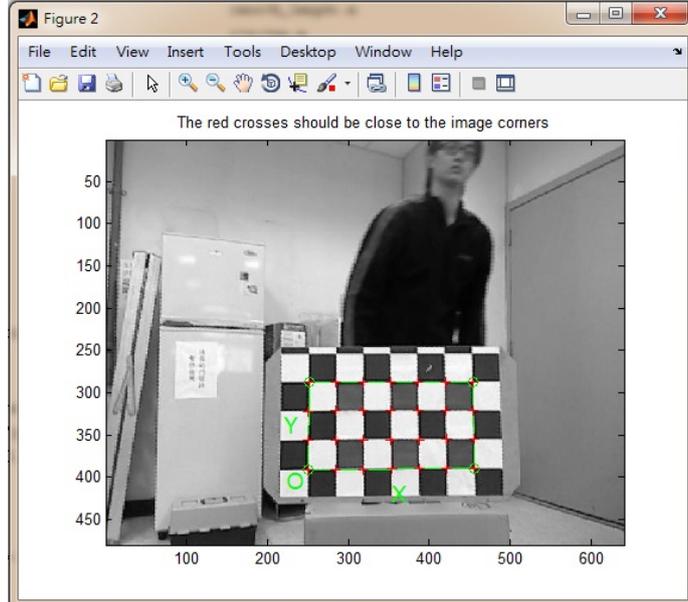


圖 79 特徵點擷取示意圖

- e. 使用「Calibration」估測相機參數。一開始會使用「closed-form solution」的方式來估測初始相機參數，再使用「non-linear optimization」（考慮鏡頭扭曲(lens distortion)）以遞迴的方式，來對相機參數做最佳化。

最後，式(36)和式(37)分別為左邊 Kinect 的內、外部參數；式(38)和(39)分別為左邊 Kinect 的內、外部參數。參考圖 80 反投影後影像(Re-projected Image)，為利用世界座標參考式(36)和式(37)的相機參數作反投影，可看到世界座標的特徵點經過反投影，大致上都可以準確投影到影像上特徵點(紅色圈圈)的位置。

$$K_L = \begin{bmatrix} 530.66 & 0 & 327.07 \\ 0 & 529.57 & 249.9 \\ 0 & 0 & 1 \end{bmatrix} \quad (36)$$

$$[R_L | t_L] = \begin{bmatrix} 0.9992 & -0.0047 & 0.040 & -180.61 \\ -0.002 & -0.9977 & -0.068 & 326.74 \\ 0.0402 & 0.0678 & -0.9969 & 1244.1 \end{bmatrix} \quad (37)$$

$$K_R = \begin{bmatrix} 519.95 & 0 & 334.87 \\ 0 & 516.93 & 264.16 \\ 0 & 0 & 1 \end{bmatrix} \quad (38)$$

$$[R_R | t_R] = \begin{bmatrix} 0.9989 & -0.0004 & 0.0479 & -460.98 \\ 0.0016 & -0.9991 & -0.0416 & 294.49 \\ 0.0479 & 0.0417 & -0.9980 & 1225.2 \end{bmatrix} \quad (39)$$



圖 80 反投影後影像

在 A2 步驟的色彩校正，由圖 81 可看到原始輸入的左右影像。



圖 81 A: 原始左邊 Kinect 影像 B: 原始右邊 Kinect 影像

針對同一時間點的左右影像，可看出兩者有明顯的色差。所以我們套用色彩校正演算法(2.3節)來修正。最後結果如圖 82 所示。



圖 82 A: 左邊 Kinect 參考影像 B: 右邊 Kinect 校正影像

圖 82 中右邊校正後影像，在冰箱表面與牆壁部分地方仍無法獲得百分之百的修改，主要原因是因為原始左右影像色彩差異太大，若從 RGB 色彩轉換到其他色彩空間(如:YCbCr 或 HSI)皆無法獲得更完整的改善。不過相較於原始影像，改善的幅度是明顯可看到的。

考慮「影像校正」(A3)步驟。以圖 82 當作輸入影像，使用影像校正演算法(2.2節)，可得到圖 83 的結果。針對其結果我們可由極線來驗證結果，在圖 83 A 中，原始左右影像的極線並未平行，所以交於一極點(在影像平面外);在圖 83 B 中明顯看到，校正完的左右影像極線是平行的，則會交於無限遠的極點。最後，左邊相機的內外部參數分別為式(40)及式(41)，右邊相機的內外部參數分別為式(42)及式(43)。

$$K_L = \begin{bmatrix} 525.3 & 0 & 204.5 \\ 0 & 523.3 & 143 \\ 0 & 0 & 1 \end{bmatrix} \quad (40)$$

$$[R_L | t_L] = \begin{bmatrix} 0.9989 & 0.0084 & -0.0466 & -73.85 \\ 0.0052 & -0.9977 & -0.0677 & 325.4 \\ -0.0470 & 0.0673 & -0.9966 & 1255.3 \end{bmatrix} \quad (41)$$

$$K_R = \begin{bmatrix} 525.3 & 0 & 204.5 \\ 0 & 523.3 & 143 \\ 0 & 0 & 1 \end{bmatrix} \quad (42)$$

$$[R_R | t_R] = \begin{bmatrix} 0.9989 & 0.0084 & -0.0466 & -344.3 \\ 0.0052 & -0.9977 & -0.0677 & 325.4 \\ -0.0470 & 0.0673 & -0.9966 & 1255.3 \end{bmatrix} \quad (43)$$

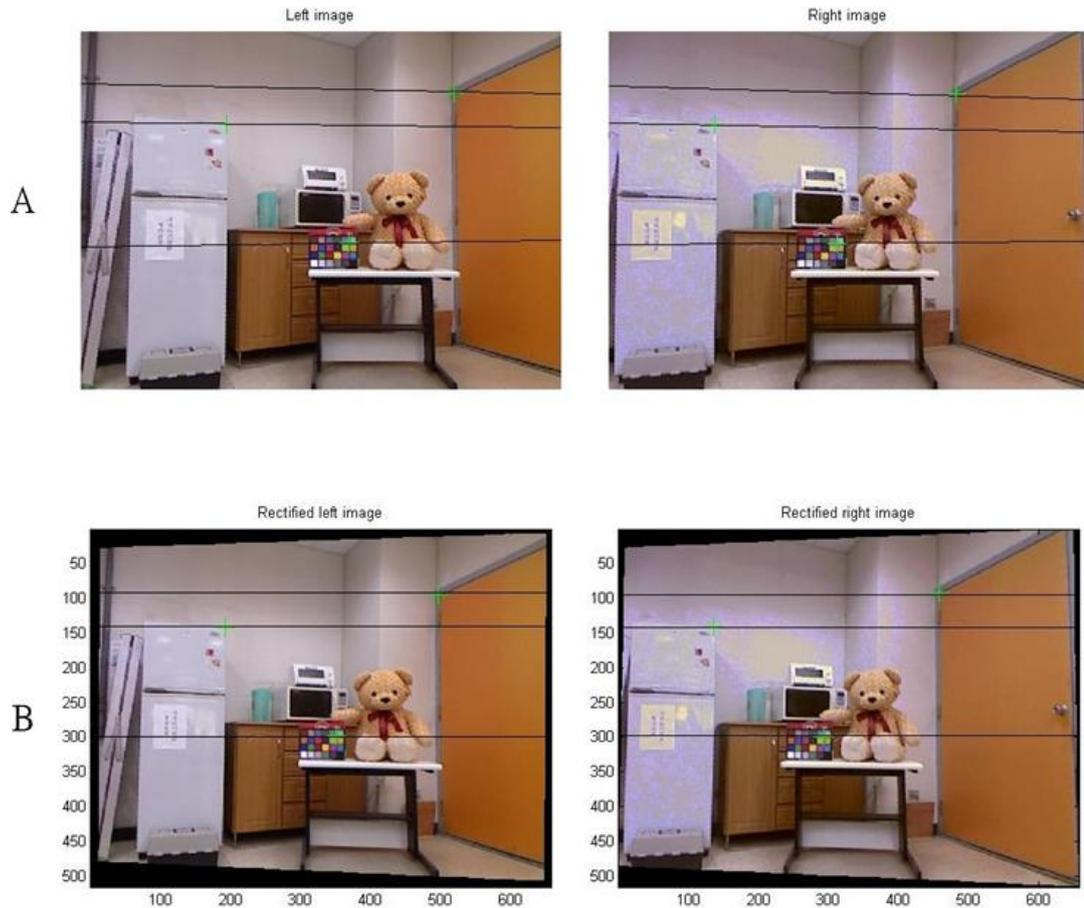


圖 83 A: 原始影像 B: 校正完影像

5.1.2 校準處理

如前面所提到的，Kinect SDK 讀取資料串流的時候，就針對「校準處理」(B) 中 B1 步驟的「使用 Kinect SDK 函數」、B2 步驟的「景深圖垂直位移」和 B3 步驟的「考慮視差模型的水平位移」做過考慮。最後必須考慮到 B4 步驟的「景深圖映射」，因為必須使用到「影像校正」(A3)後的相機參數，所以必須額外再做。

參考 2.5 節的景深映射演算法，且為保留重要的資訊，針對校正後的彩色影像與映射後的景深影像作切割。最後結果可參考圖 84 與圖 85。



圖 84 A: 切割後左邊影像 B: 切割後右邊影像

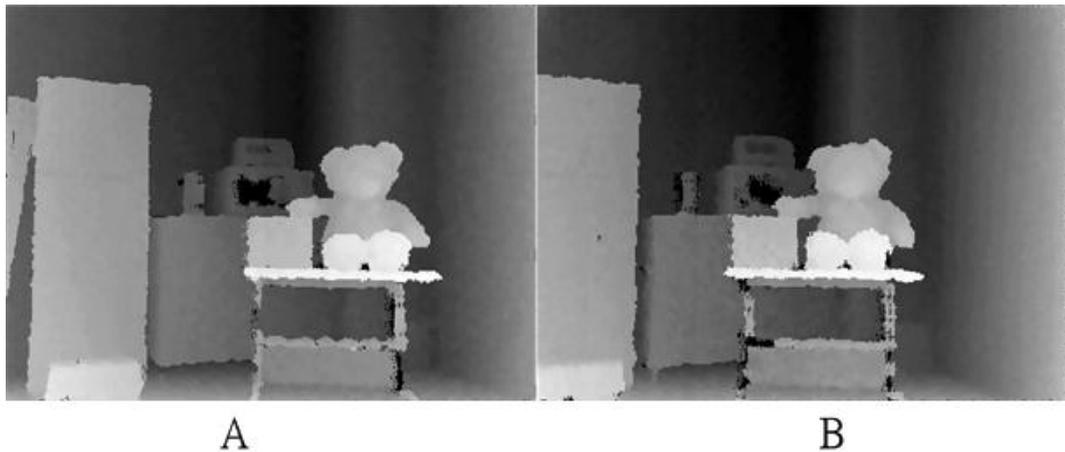


圖 85 A: 映射切割後左邊景深影像 B: 映射切割後右邊景深影像

對於校準處理的好處，我們可以參考[17]的想法，使用 1 維投影的方式來產生新視角的影像。所以我們使用圖 35 的 B1 步驟以及 B3 步驟的輸出景深影像，將彩色影像投影到新視角上。在此特別注意的一點是，做 1 維影像投影前，會先將景深影像「洞」(未知景深資訊區域)利用 OPENCV 的補洞函式(cvInpaint)做填補，且將彩色影像投影到右邊距離彩色鏡頭約 9.6cm 的位置上。B1 步驟後的原始影像及投影後的影像可參考圖 86 及圖 87。

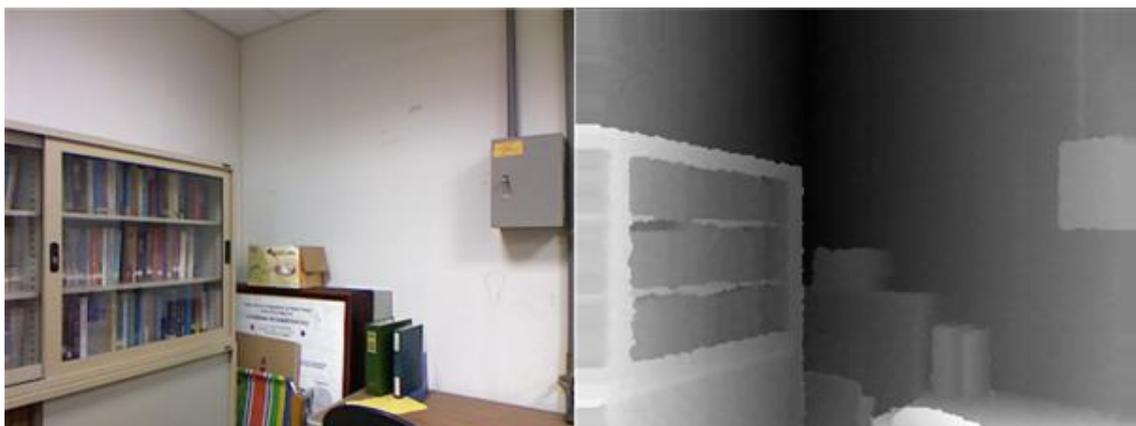


圖 86 原始彩色影像以及 B1 輸出的景深影像



圖 87 圖 86 投影至新視角的影像

圖 87 中，黑色洞的區域(如紅色圈圈的區域)，皆為投影後遮蔽區域的部份。
另外參考圖 88 及圖 89，為 B3 步驟的原始影像以及投影結果。



圖 88 原始彩色影像以及 B3 輸出的景深影像



圖 89 圖 88 投影至新視角的影像

比較圖 87 及圖 89 的結果，兩張影像的遮蔽區域出現的地方是最明顯的不同，圖 87 的景深資訊與彩色影像並未完全校準，投影完遮蔽區域出現的區域，並未貼緊在彩色影像上的物體邊緣。但圖 89 的遮蔽區域對應到彩色影像上，會明顯較圖 87 的結果合理，遮蔽區域大多出現在物體邊緣。從這些結果可以看出校準演算法，對於影像合成是有幫助的。

5.1.3 景深影像改善

進入「景深影像改善與合成」(C)的部分。一開始對於景深影像採用 C1 步驟的「濾除雜訊」，來濾掉景深影像邊界的雜訊。考慮 4.3.1 節的聯合雙向濾波器演算法，以圖 85 當作輸入影像，則可得到聯合雙向濾波後的影像(如圖 90 所示)。各參數值可參考表 5。

表 5 濾除雜訊-聯合雙向濾波器參數表

Filter Size	15×15
σ for filter $f(\cdot)$	1.875
σ for filter $h(\cdot)$	15
σ for filter $r(\cdot)$	15
σ for filter $g(\cdot)$	15

σ for filter $b(\cdot)$	15
--------------------------------	----

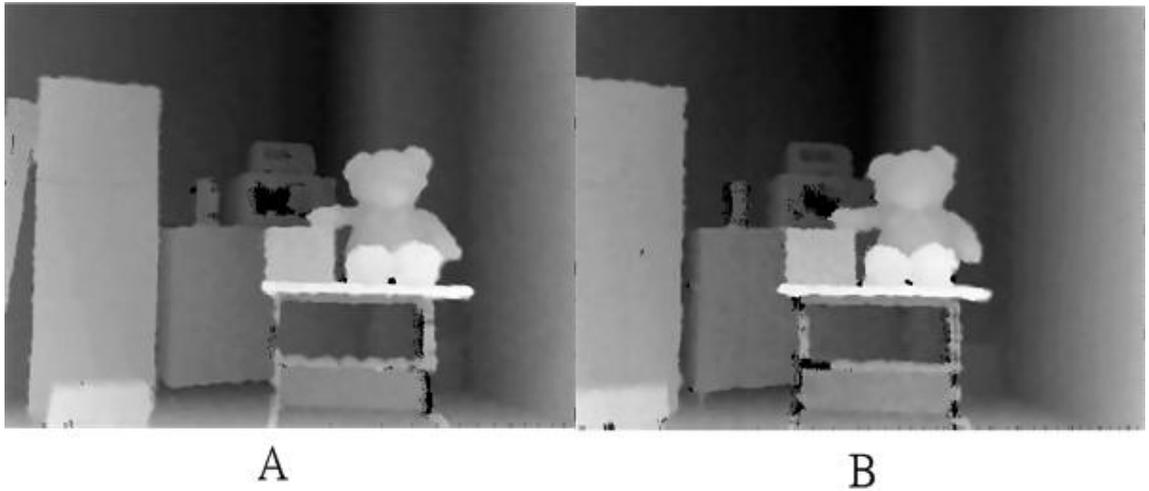


圖 90 A: C1 - 左邊景深影像 B: C1 - 右邊景深影像

	Left Image	Right Image
Original		
C1-JBF		

圖 91 C1 - 細部比較示意圖(1)

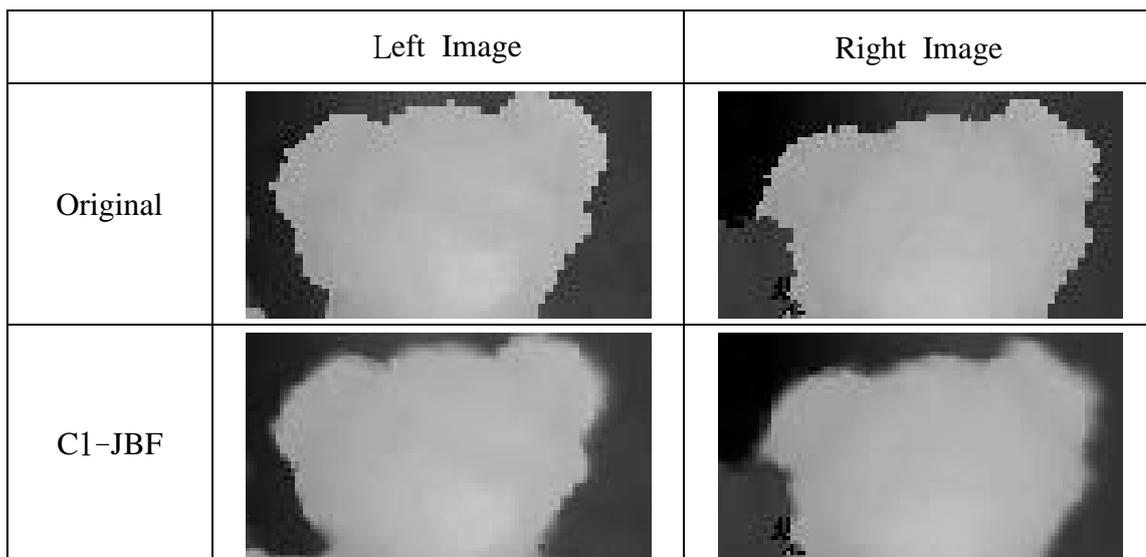


圖 92 C1 - 細部比較示意圖(2)

對照圖 85 與圖 90，在物體邊界的部份會較平滑，因為尖銳的高頻雜訊在參考彩色影像的資訊下被濾除，景深圖的物體會顯得較為柔順。

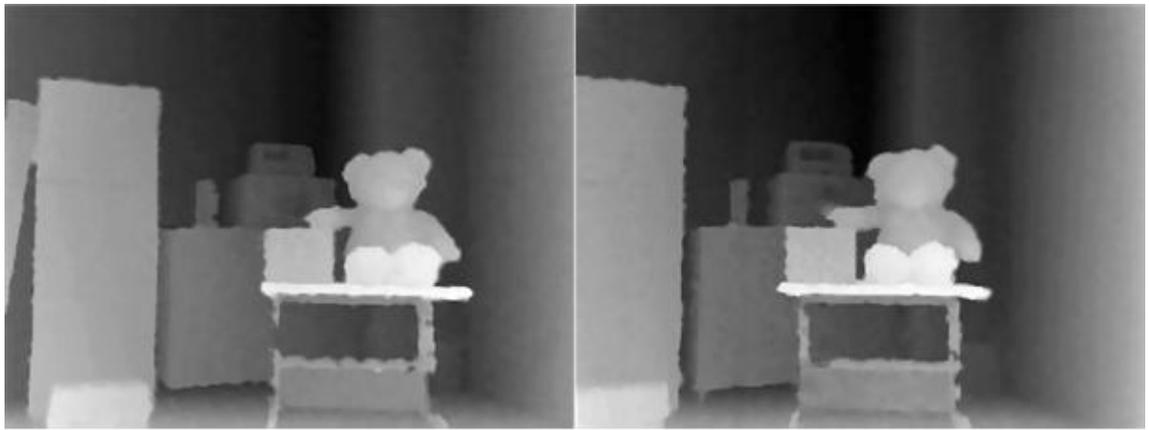
參考圖 91 的桌腳細部放大圖，原始的左右影像在邊緣部分凹凸不平，表示其高頻雜訊明顯。使用雙向濾波器濾除邊緣雜訊，且加入彩色影像的 R、G、B 資訊，讓濾波器在濾波時會額外考慮顏色資訊，讓整張圖會變得更完整。對於圖 92 小熊頭部的細部放大圖，原始影像跟濾波後的影像在邊緣部份，也有明顯的改進。

考慮「消除缺陷問題」(C2)來修補景深影像中未知的景深資訊。參考聯合雙向濾波器(4.3.2 節)演算法，利用遞迴的方式來消除缺陷問題。其結果可參考圖 93。

表 6 消除缺陷問題-聯合雙向濾波器參數表

Filter Size	15×15
σ for filter $f(\cdot)$	5
σ for filter $r(\cdot)$	15
σ for filter $g(\cdot)$	15

σ for filter $b(\cdot)$	15
th	0.5



A

B

圖 93 A: C2 - 左邊景深影像 B: C2 - 右邊景深影像

	Left Image	Right Image
Original		
C2		

圖 94 C2 - 細部比較示意圖(1)

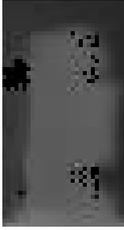
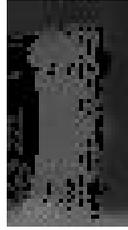
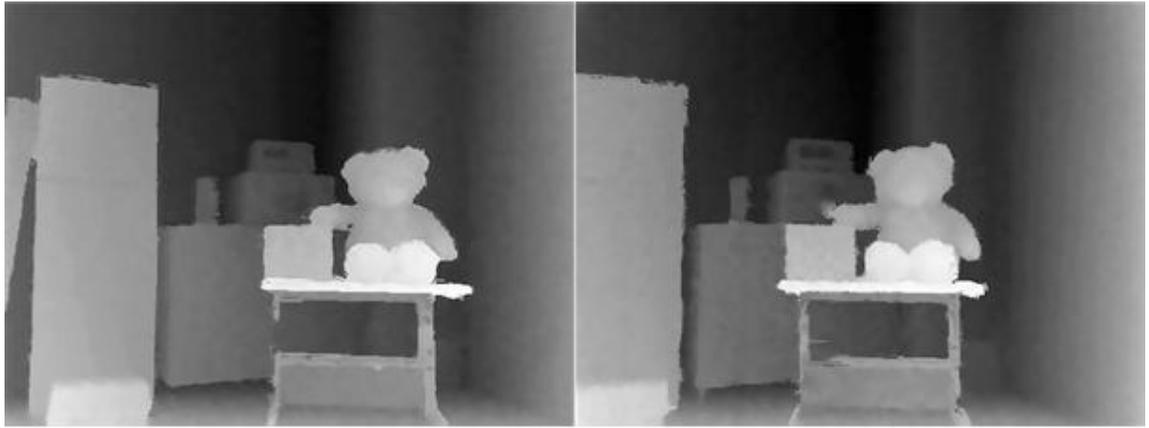
	Left Image	Right Image
Original		
C2		

圖 95 C2 - 細部比較示意圖(2)

在圖 93 中，缺陷問題已完整消除，但仍有些不足的地方。參考圖 94 的左圖影像，微波爐上的「洞」有很完整的改善；但右圖改善完的影像，手部會有殘影，主要原因是因為 Kinect 感應器偵測景深時，微波爐上的部分景深資訊，被手的景深資訊覆蓋，所以參考彩色影像資訊時，會有誤判的情況。相較於圖 95 的右邊影像，杯子被錯誤資訊嚴重干擾。使用 C2 演算法後，杯子的邊緣很明顯的被修補回來，影像看起來更自然。

參考圖 93 相較於圖 85，經過各個步驟的雜訊去除，景深影像已有明顯改善，但部分區域因為缺陷過大，使用經過好幾個步驟的去雜訊仍無法修補完全，所以最後利用 C3 步驟的「基於彩色影像改善景深影像」針對這些區域做改善。

參考 4.3.3 「基於彩色影像改善景深影像」(C3)的演算法(方塊大小為 21×21 ， $th = 15$)，針對圖 93 做加強改善，其結果可參考圖 96。



A

B

圖 96 A: C3 - 左圖影像 B: C3 - 右圖影像

	Left Image	Right Image
Original		
C3		

圖 97 C3 - 細部比較示意圖(1)

	Left Image	Right Image
Original		

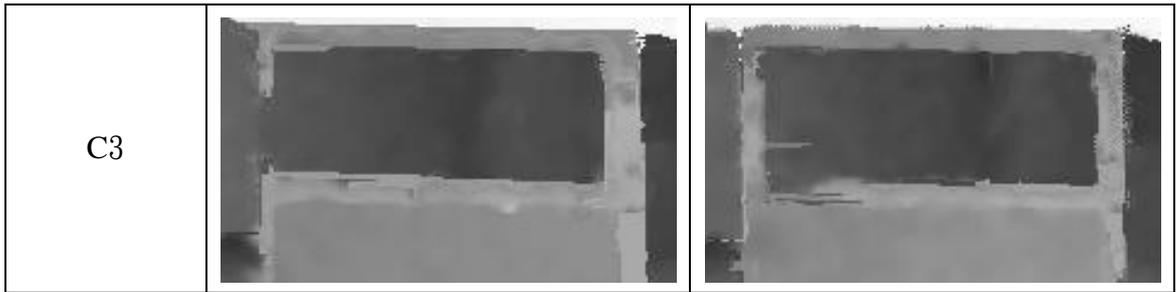
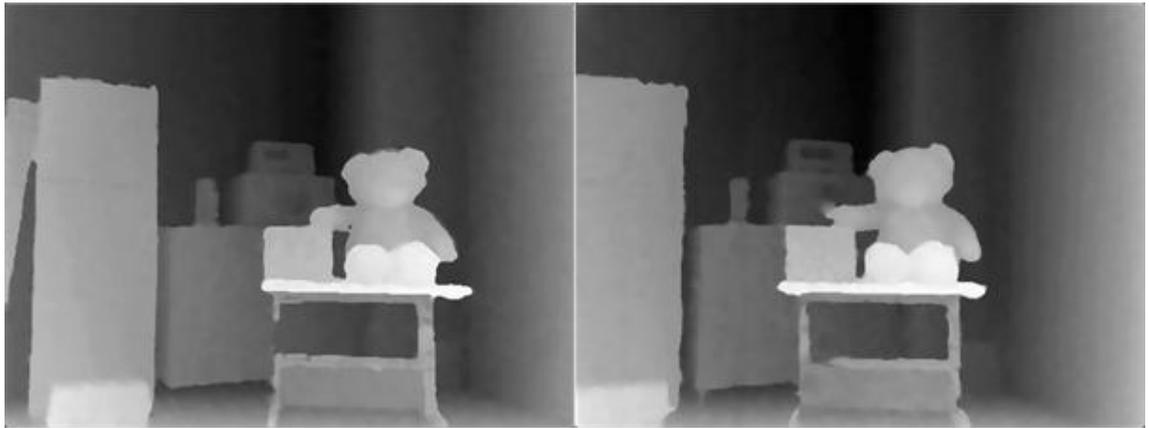


圖 98 C3 - 細部比較示意圖(2)

從圖 96 明顯看到，整個影像相較於圖 93 有明顯改善。就細部影像來看，在圖 97 中不管左右圖的冰箱上都有殘影的出現，主要是因為彩色影像在冰箱上邊緣的部份，跟背景牆壁很像(可參考圖 82)，所以參考彩色影像資訊時，在這個部份會有誤差。對於圖 98 為例，桌腳的景深資訊大部分都有修補起來，小缺陷如左圖的左邊桌腳部分，因為桌腳的資訊量太少，所以無法完全修補；以及右圖左邊靠近桌腳的部分，也因為相同的原因無法修補。但整體看起來已有明顯改善。

參考 4.3.4 節 C4 步驟的「中值濾波」演算法，針對圖 96 做濾波動作(濾波器大小為 5×5)。其結果可參考圖 99。

比較圖 99 與圖 96 之間的差異，明顯看到邊緣的胡椒鹽雜訊有大幅的減少。對於圖 100 的冰箱上緣，中值濾波後的影像在該處凹凸不平，主要原因原始輸入影像在該處本來就有嚴重的雜訊問題，所以使用中值濾波也無法完全還原該處的資訊，但該處相較於輸入影像明顯較為連續。在圖 101 中，中值濾波器明顯發揮其用處，整體的影像變的更連續平滑，邊緣雜訊幾乎都被濾除。



A

B

圖 99 A: C4 - 左圖影像 B: C4 - 的右圖影像

	Left Image	Right Image
Original		
C3		

圖 100 C4 - 細部比較示意圖(1)

	Left Image	Right Image
Original		
C3		

圖 101 C4 - 細部比較示意圖(2)

5.1.4 影像合成

最後是使用 MPEG 的影像合成參考軟體(VSRS)，針對經過「影像改善」(C)每個步驟的合成影像與原始影像做比較。



圖 102 原始合成影像(圖 84 與圖 85 所合成)



圖 103 影像改善後(C)合成影像(圖 84 與圖 99 所合成)

原始合成 影像	
------------	--



圖 104 圖 102 與圖 103 細部比較示意圖(1)

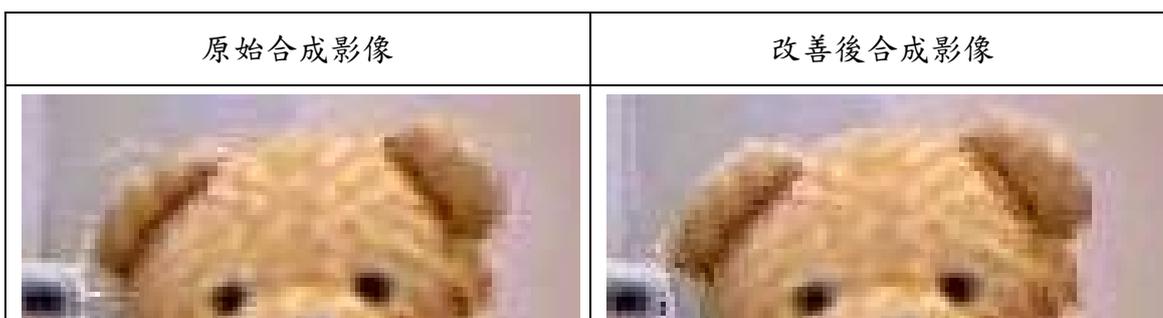


圖 105 圖 102 與圖 103 細部比較示意圖(2)

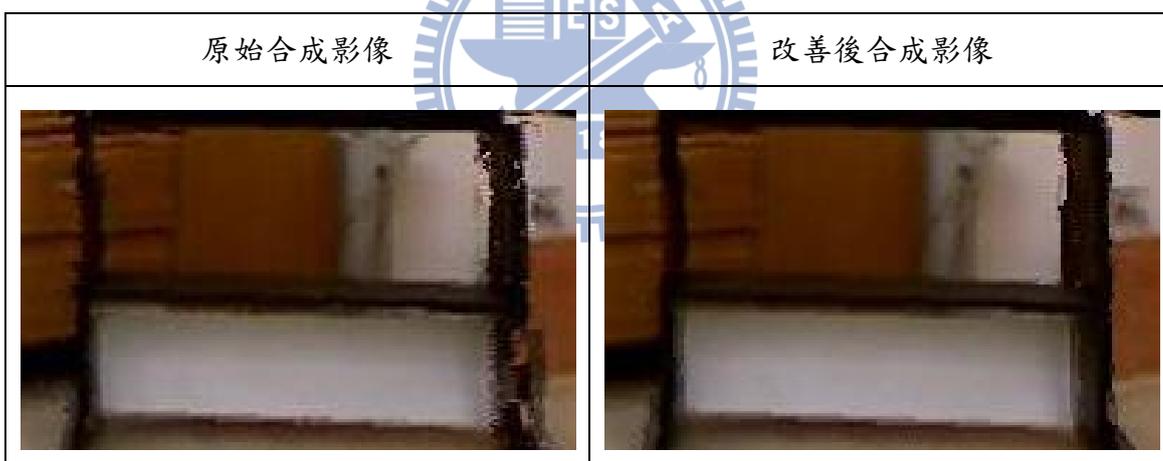


圖 106 圖 102 與圖 103 細部比較示意圖(3)

對照圖 104 的細部比較圖，白色桌子的部分在左右兩端原本有明顯的殘影以及缺角，經過「影像改善」(C)後，桌子左邊的缺角被補足，右邊的殘影被消除。

圖 105 中，原始的左側小熊頭有較多的雜訊，且小熊頭上包含耳朵附近也較模糊不清。經過改善後，小熊頭左側的雜訊大幅減少，小熊頭的邊緣也較為清晰，小熊左邊的耳朵在原始合成影像中像似缺一角，改善後的合成影像這部分也完全補回，看起來較自然。

圖 106 中，這是整張影像改善最大的區塊，對於原始 Kinect 感應器所偵測的景深影像，在這部分有嚴重的誤差(包含邊界凹凸不平、未知區域過多及偵測錯誤的區域)，這些誤差的影響可由原始合成影像看出。經過改善後，其結果大致上完美，但在右邊桌腳的部分仍有微小的缺陷，主要還是歸咎於 C3 步驟無法徹底的改善這些區域。但整體的結果已比原始影像好。

5.1.5 被動式景深影像與合成結果比較

比較使用 MPEG 標準景深估測軟體(DERS)所估測的景深影像。參考圖 107 經由圖 84 所產生的景深影像。



圖 107 A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像

從圖 107 可明顯觀察到 DERS 的景深影像有很大的缺點，包含有大量的遮蔽區域、特徵點缺少(如背景牆壁)的地方景深資訊會雜亂等等。針對圖 107 的景深影像利用 MPEG 的合成軟體(VSRS)產生合成影像。如圖 108 所示。



圖 108 被動式景深影像合成結果

參考圖 108，其結果較使用 Kinect 加上許多處理所得到合成影像來的好。對於較不好的景深影像，卻得到較好的合成結果，主要原因是由於合成軟體使用 1 維模式時，在結合(Merge)步驟時(參考 2.6.2 節)對於遮蔽區域有較好的處理，如圖 109 所示。



圖 109 A: VSRS 結合步驟的彩色影像 B: VSRS 結合步驟的景深影像

圖 109 A 與 B 中，對於遮蔽區域仍會選擇有該資訊的影像做填補。所以觀察其細部區域(如圖 110 所示)。

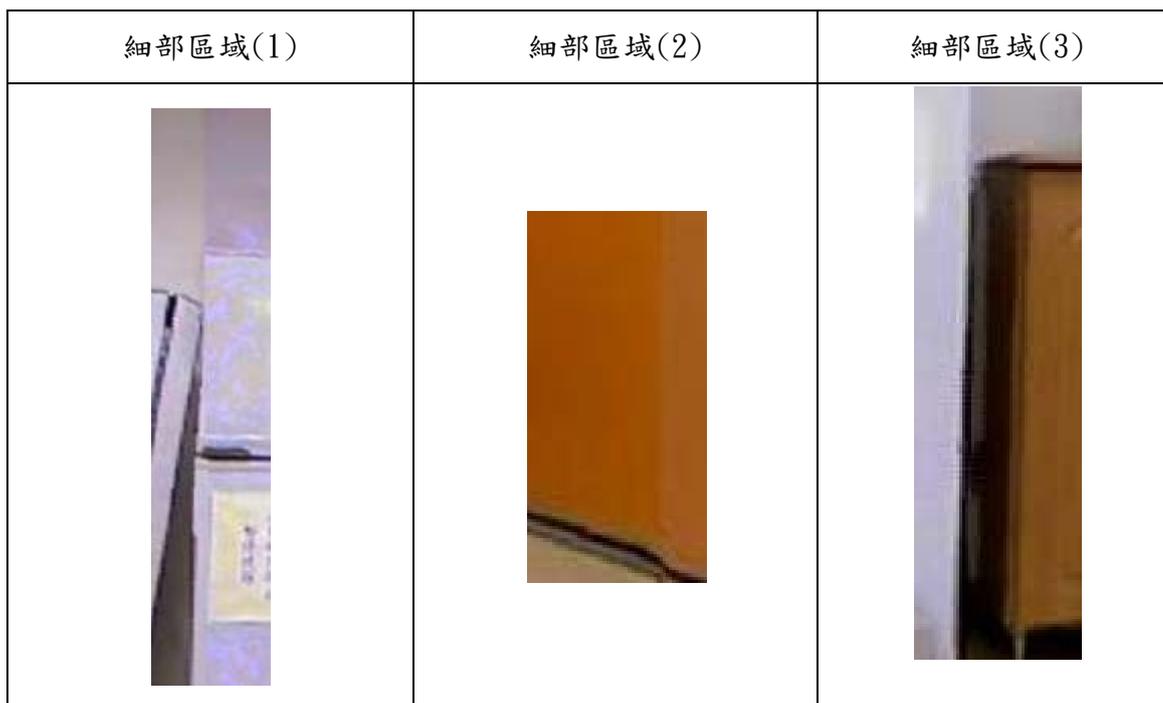


圖 110 圖 109 A 細部區域分析

在圖 110 的細部區域(1)，可以看到梯子跟冰箱之間較不自然，原因是因為在景深影像內該區域是未知景深資訊，且參考圖 111 A 中，梯子部分的景深值錯誤，導致前映射的結果錯誤，所以結合(Merge)步驟後的影像才顯得不自然。在圖 110 的細部區域(2)的門縫，也因為最右邊的區域在景深影像內是未知景深區域，所以在「結合」的步驟會直接選右邊影像來填補。在圖 110 的細部區域(3)內，可以發現櫃子與冰箱之間的縫隙不自然，主要原因可參考圖 111，在圖 111 B 中因為景深錯誤較嚴重，所以在櫃子與冰箱之間前映射的值有錯，剛好在圖 111 A 中該區塊為遮蔽區域，所以最後結合的影像只能參考右邊影像，而導致該區域有不自然的錯誤。

比較使用 Kinect 景深資訊合成出的影像(圖 103)，以及使用參考軟體-DERS 估出的景深資訊所合成出的影像(圖 108)，可以看出在 DERS 明顯錯誤的區域(如背景的牆壁)，卻在合成影像上看不出明顯的錯誤。主要原因是背景區域多是顏色一致，且沒什麼特徵點的區域(這部分是被動式景深估測演算法最易錯的區域)，

即使景深資訊錯誤，彩色影像投影過去的區域還是背景資訊，所以看不出來有差異。

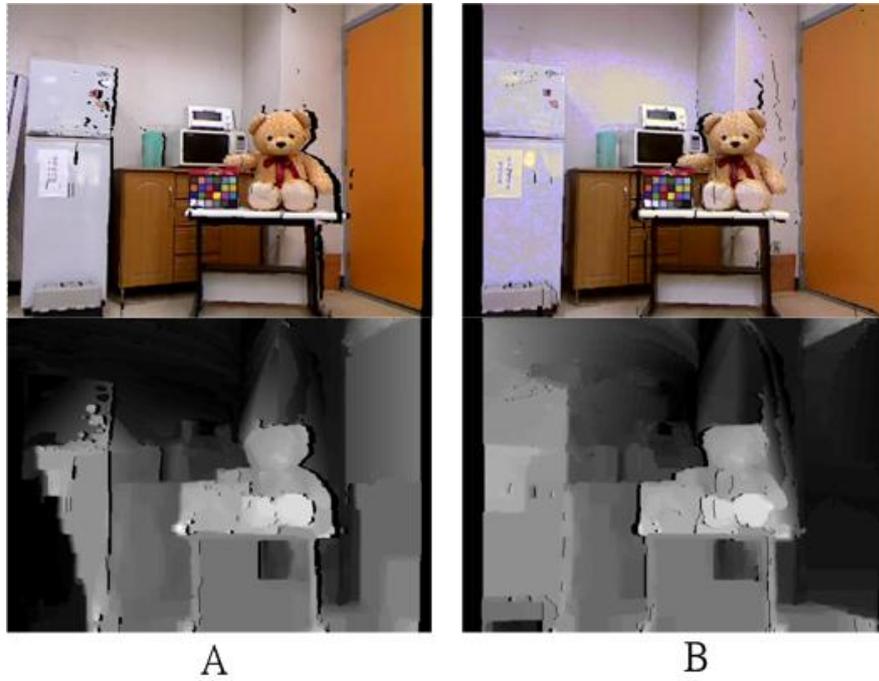


圖 111 A: VSRs 前映射後的左邊影像 B: VSRs 前映射後的右邊影像



5.2 第 2 組測試影像模擬與結果 (Simulation of Second Test Sequence)

在這裡使用第 2 組測試影像來做測試。參考圖 73 擺放示意圖，圖 112、圖 113 分別為左右 Kinect 感應器輸出的影像。



圖 112 A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像

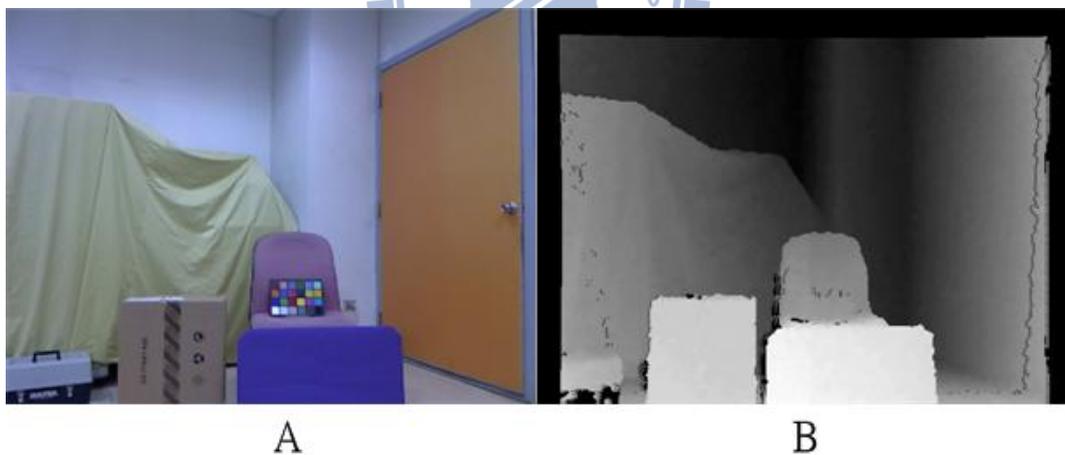


圖 113 A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像

圖 112、圖 113 為考慮「校準處理」(4.2 節)中 B1、B2 和 B3 後的輸出影像。

5.2.1 校正處理

在 A1 步驟中的相機校正，我們仍使用[32]的相機校正軟體。其原始左右相機參數與式(36)、式(37)、式(38)和(39)類似。

在校正處理(A)的第二步是對左右的彩色影像作色彩校正(A2)，參考色彩校正演算法(2.3 節)，其結果如圖 114 所示。



圖 114 A: 左邊參考影像 B: 右邊校正影像

從圖 114 會看到兩張影像的色彩仍無法完全相同，從白色背景牆壁做比較還是可以看到差異。對照圖 113 與圖 114 B 校正前後的影像，在椅子前面的板子在實際生活中是紫色，但在圖 113 中卻呈現深藍色，但是在圖 114 B 中，對照前面的板子顏色跟圖 114 A 的板子顏色，兩者的顏色已十分接近。所以整體的顏色校正雖然不是完美，但已有明顯的改善。

在影像校正(C3)中，利用 2.2 節的影像校正演算法，設法把雙相機投影到虛擬位置上，讓這兩台相機可以達到「極幾何」(Epipolar Geometry)上的平行。最後影像校正完的相機參數與式(40)、式(41)、式(42)和式(43)類似。輸出影像如圖 115 所示。



A

B

圖 115 A: 影像校正後左邊影像 B: 影像校正後右邊影像

參考圖 115，明顯看到影像校正已經完成。參考圖 116 利用極線來做驗證。

圖 116 A 分別是原始左右影像，圖 116 B 為校正完的左右影像，可看到兩者的極線順利變成水平，則表示影像校正順利完成。



A

B

圖 116 A: 原始左右影像 校正後左右影像

5.2.2 校準處理.

Kinect SDK 讀取資料串流的時候，已針對「校準處理」(B)中 B1 步驟的「使用 Kinect SDK 函數」、B2 步驟的「景深圖垂直位移」和 B3 步驟的「考慮視差模型的水平位移」做過考慮。所以這裡只針對最後一步的「景深圖映射」(B4)做考慮。

參考 2.5 節的景深映射演算法。為保留重要的資訊，針對校正後的彩色影像 (圖 115)與映射後的景深影像作切割。最後結果可參考圖 117 與圖 118。

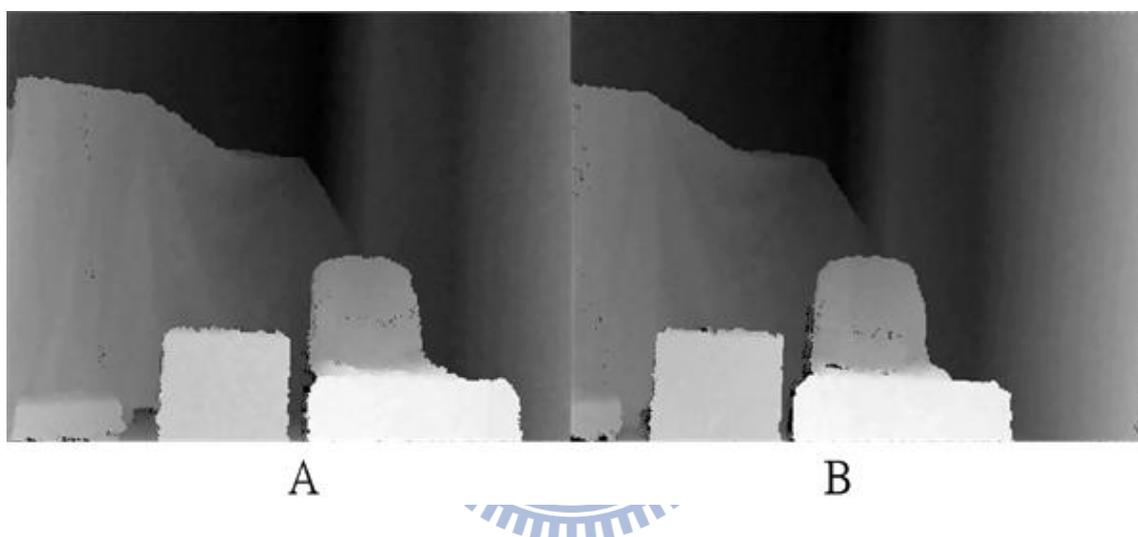


圖 117 A: 映射切割後左邊景深影像 B: 映射切割後右邊景深影像



圖 118 A: 切割後左邊彩色影像 B: 切割後右邊彩色影像

5.2.3 景深影像改善

在「景深影像改善與合成」(C)的部分。首先，考慮「濾除雜訊」(C1)的步驟，來濾掉景深影像邊界的雜訊。考慮 4.3.1 節的聯合雙向濾波器演算法，以圖 117 當作輸入影像，則可得到聯合雙向濾波後的影像(如圖 119 所示)。各參數值可參考表 5。

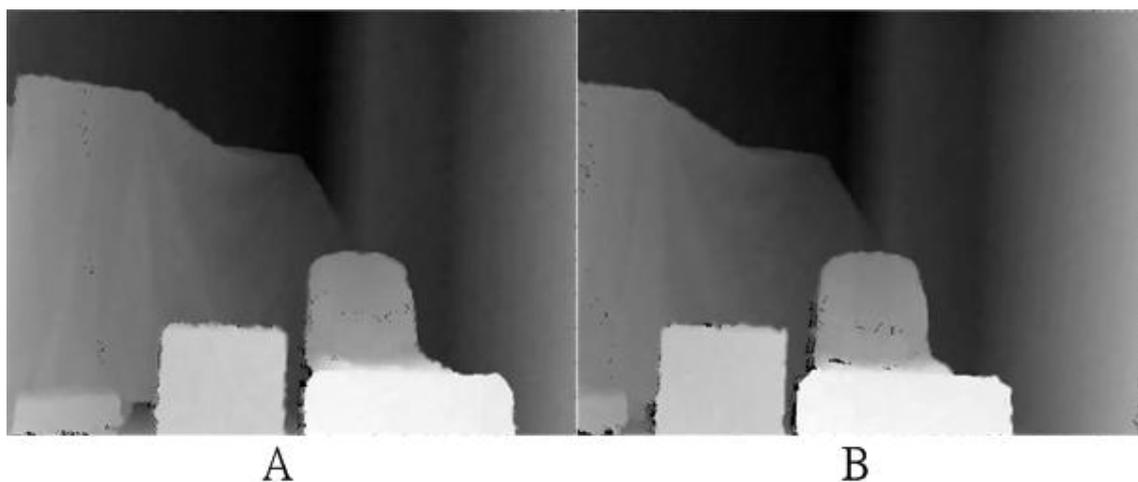


圖 119 A: C1 - 左邊景深影像 B: C1 - 右邊景深影像

	Left Image	Right Image
Original		
C1		

圖 120 C1 - 細部比較示意圖(1)

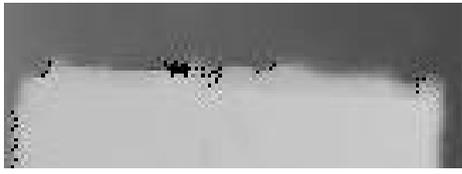
	Left Image	Right Image
Original		
C1		

圖 121 C1 - 細部比較示意圖(2)

參考圖 119 對照圖 117，未知景深資訊點明顯的被保留下來，已知景深點經過聯合雙向濾波後，在邊緣部分明顯有改善。參考圖 120，原始影像的邊緣雜訊(鋸齒狀)在濾波後，明顯得到改善。在圖 121 原始影像雜訊更為明顯，凹凸起伏更為明顯，經過濾波後雖然不能完全濾除雜訊，但結果已比原始影像有明顯改善。

在圖 119 中，「洞」(未知景深資訊點)仍存在，對於缺陷問題我們使用 C2 演算法-「消除缺陷問題」來修補景深影像中未知的景深資訊。參考消除缺陷問題(4.3.2 節)演算法，利用遞迴的方式來消除缺陷問題。其參數設定可參考表 6，結果可參考圖 122。

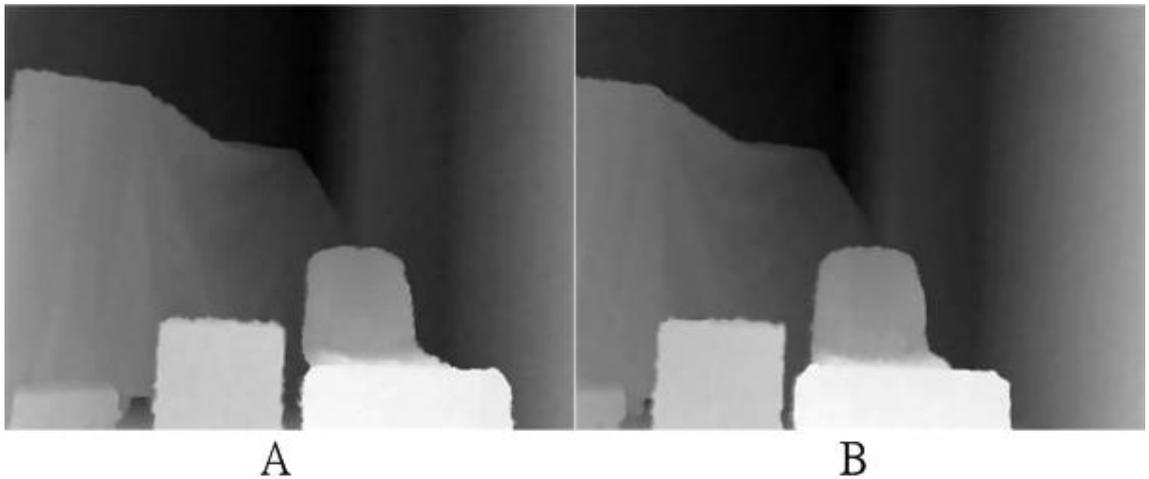


圖 122 A: C2 - 左邊景深影像 B: C2 - 右邊景深影像

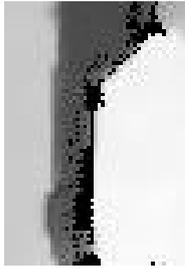
	Left Image	Right Image
Original		
C2		

圖 123 C2 - 細部比較示意圖(1)

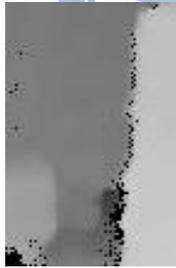
	Left Image	Right Image
Original		
C2		

圖 124 C2 - 細部比較示意圖(2)

參考圖 122 與圖 119，可看出整張影像的缺陷現象已順利被移除。從細部影像(圖 123、圖 124)來看，除缺陷問題明顯被消除外，邊緣部分因為加入彩色影像資訊，所以物體邊緣部分順利保留下來。

參考圖 122 相較於圖 117，經過雜訊濾除與缺陷問題的消除，景深影像已有明顯改善。但部分區域因為缺陷過大，使用經過好幾個去除雜訊的步驟仍無法修補完全。所以利用 C3 步驟的「基於彩色影像改善景深影像」，針對這些區域做改善。

參考 4.3.3 節，C3 演算法-「基於彩色影像改善景深影像」(左邊方塊大小為 19×19 ，右邊方塊大小為 11×11 ， $th=15$)，針對圖 122 做加強改善，其結果可參考圖 125。

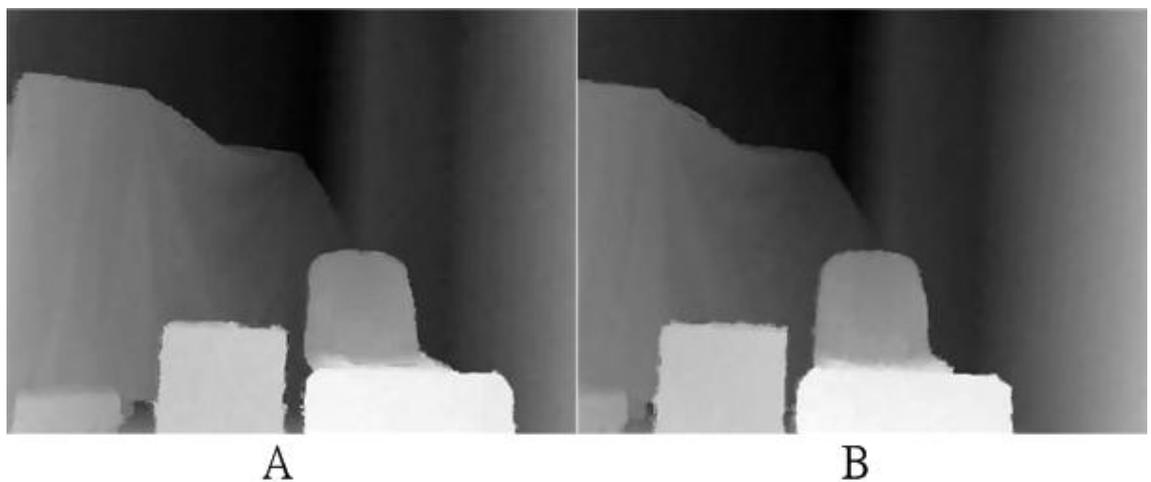


圖 125 A: C3 - 左邊景深影像 B: C3 - 右邊景深影像

	Left Image	Right Image
Original		
C3		

圖 126 C3 - 細部比較示意圖(1)

	Left Image	Right Image
Original		
C3		

圖 127 C3 - 細部比較示意圖(2)

參考圖 125，由於圖 122 並無太大的缺陷，所以 C3 演算法改善的部份不大。參考圖 126 的左圖，原始影像靠上緣的部份缺一角(實線圓圈)，但在改善後也有成功把該缺角補起來。圖 127 的影像改善的部分，主要是讓景深邊緣部分與彩色影像的邊緣更匹配(參考圖 128 實線圈圈部分)。

	Original	C3
Left Image		

圖 128 圖 127 左邊景深影像與彩色影像疊合圖

在圖 125 中，景深影像已大致上跟彩色影像匹配，但物體邊界存在一些胡椒鹽雜訊，所以我們需要再加一層濾波器來濾除。參考 4.3.4 節「中值濾波」(C4) 的演算法，針對圖 125 做濾波動作(濾波器大小為 5×5)。其結果可參考圖 129。

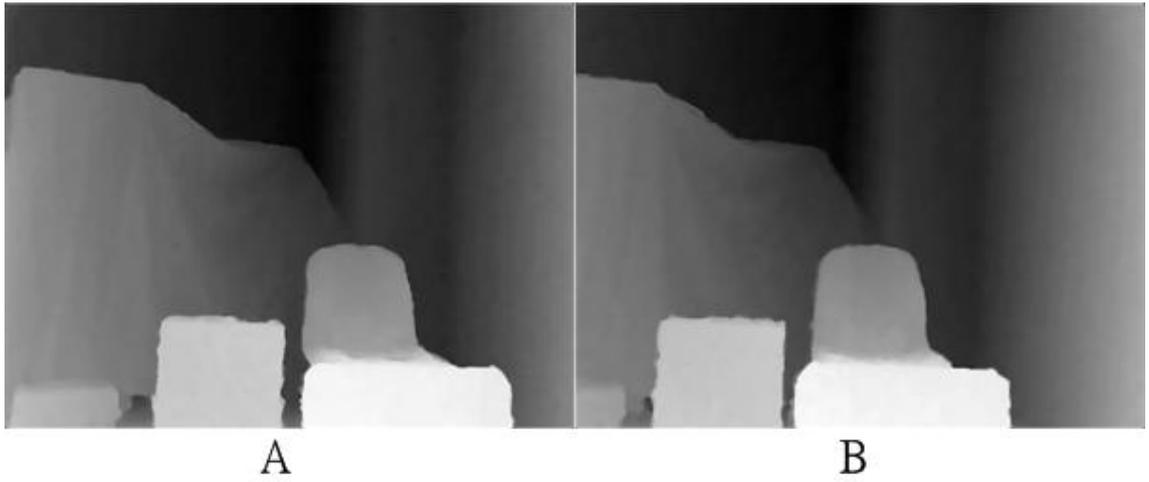


圖 129 A: C4 - 左邊景深影像 B: C4- 右邊景深影像

	Left Image	Right Image
Original		
C4		

圖 130 C4 - 細部比較示意圖(1)

	Left Image	Right Image
Original		

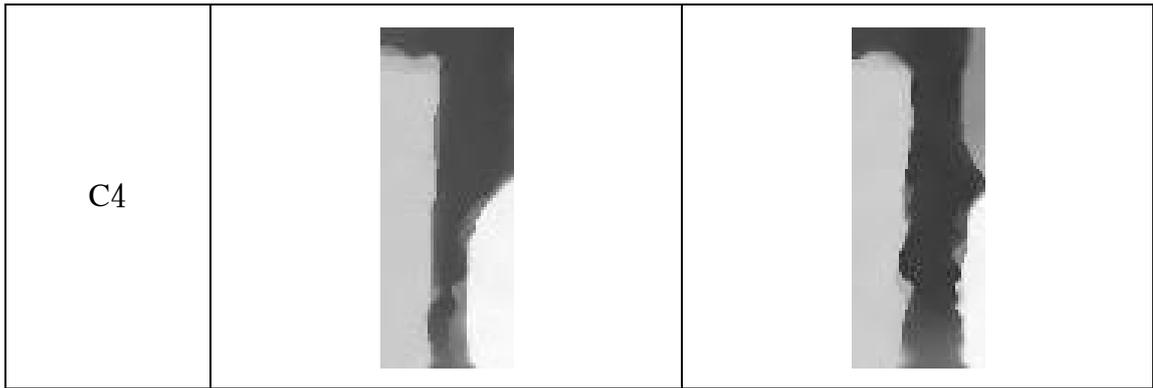


圖 131 C4 - 細部比較示意圖(2)

比較圖 125 與圖 129，最明顯改善的部份為邊緣雜訊幾乎被濾除，這部分會在 5.2.4 節的合成部分會看到明顯的變化。參考圖 130 與圖 131 的細部比較，更可以看到原始影像的邊緣雜訊，經過中值濾波後幾乎都消失了。

5.2.4 影像合成

最後是使用 MPEG 的影像合成參考軟體(VSRS)，針對經過「影像改善」(C) 每個步驟的合成影像與原始影像做比較。參考圖 132 為原始合成影像，圖 133 為景深影像改善後的合成影像。



圖 132 原始合成影像(圖 117 與圖 118 所合成)



圖 133 影像改善後(C)合成影像(圖 118 與圖 129 所合成)

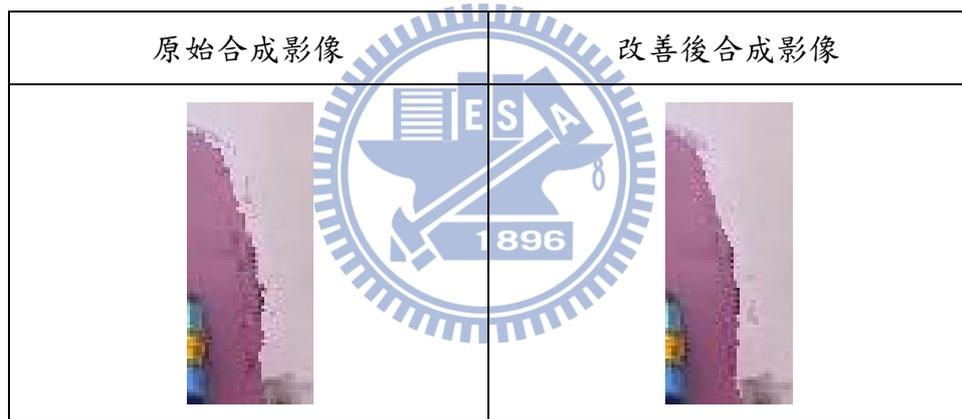


圖 134 合成影像細部比較示意圖(1)

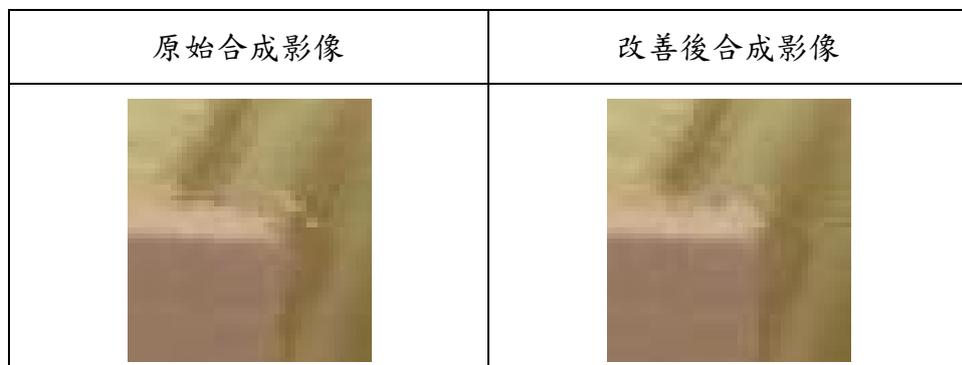


圖 135 合成影像細部比較示意圖(2)

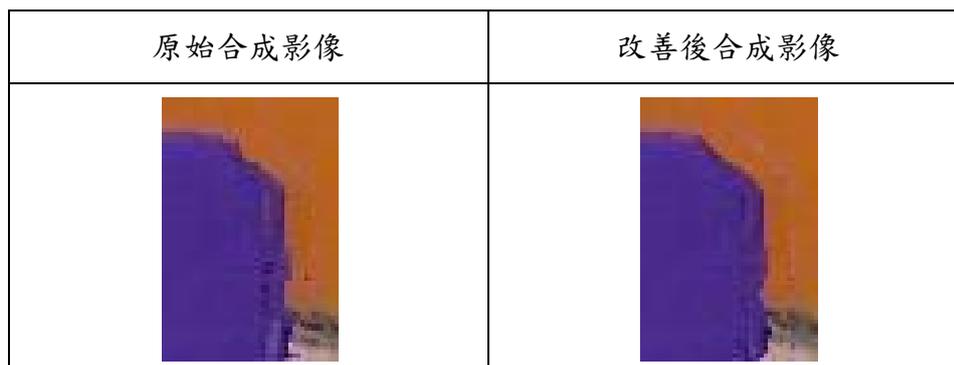


圖 136 合成影像細部比較示意圖(3)

參考圖 132 與圖 133，兩張影像基本上差異不大，主要原因是原始的景深圖瑕疵現象較少，大部份的物體只有在邊界有雜訊的存在，所以改善的幅度沒有 5.1 節的第一組測試影像來的明顯。不過在邊緣部分還是看得出差異。

參考圖 134，明顯看到椅子的邊緣雜訊變少，凹凸不平的邊緣也較為平滑。而在邊緣部分，椅子有黑色的邊條，但因為太細小，所以合成時不易完全貼在椅子上，導致看起來像是雜訊。參考圖 135，原始影像的箱子邊緣會投影到不適當的位置上，所以看起來像是雜訊；相對於改善後的影像，箱子的邊緣大部分投影到適合的位置上，箱子看起來較完整且邊緣雜訊也較少。參考圖 136，對照原始合成影像與改善後的合成影像，邊緣雜訊同樣減少，且原始合成影像較不自然(白色地板部分資訊被認為前景)，改善後得合成影像較為自然。

5.2.5 被動式景深影像與合成結果比較

比較使用 MPEG 標準景深估測軟體(DERS)所估測的景深影像。參考圖 137 由圖 118 藉由 DERS(2.4 節)估測出的景深影像。

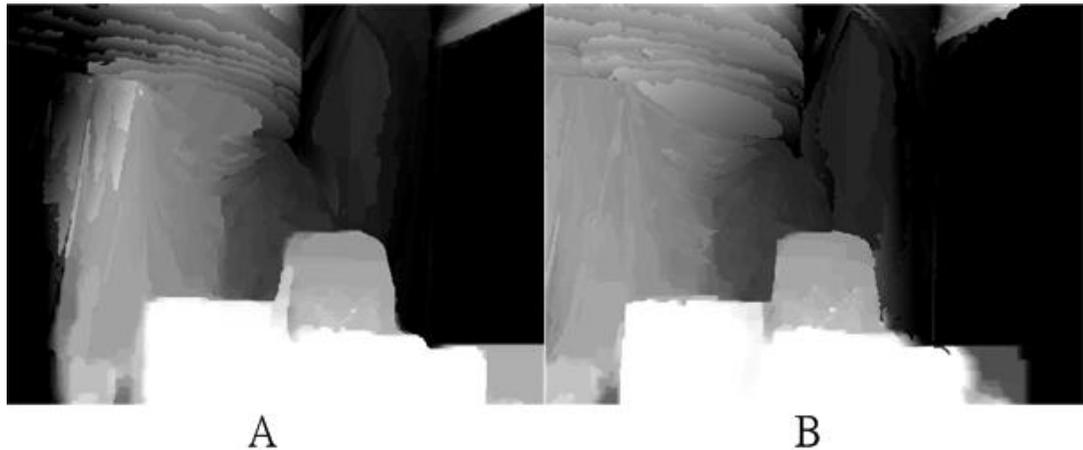


圖 137 A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像

在圖 137 中，明顯可以看到背景資訊十分雜亂(對照圖 118)，景深資訊十分不一致。此外也有大面積的遮蔽區域(如圖 137 A 的最左邊與最右邊，圖 137 B 的最右邊)。所以利用 MPEG 所提供的合成參考軟體-VSRS，基於圖 118 與圖 137 實現影像合成，其結果可參考圖 138。



圖 138 被動式景深影像合成結果

參考圖 138，其結果看似非常理想，但仔細觀察一些細節也可發現其缺點。參考圖 139 的細部區域示意圖。

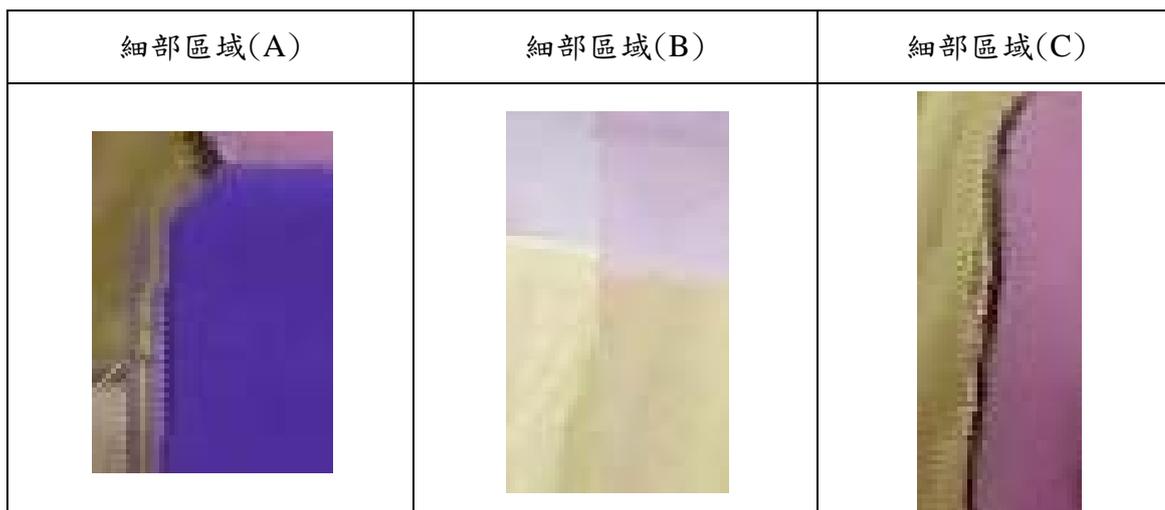


圖 139 圖 138 細部區域示意圖

在圖 139 A 明顯看到板子旁邊有殘影的存在，主要原因是在圖 137 的左右景深影像在該處的景深資訊錯誤，在合成時會誤以為該處為前景，導致映射投影錯誤。在圖 139 B 中，明顯看到同一個物體卻擁有高低落差，主要原因是圖 137 A 在該處為遮蔽區域，所以只能拿右邊的彩色影像直接填補，所以看起來才會有明顯的不自然。圖 139 C 中，儘管黑色邊條相較於圖 133 較為成功，但邊緣雜訊還是看的到。所以雖然圖 138 看似較為理想，但其細節還是看得出不理想的部份。

5.3 第 3 組測試影像模擬與結果 (Simulation of Third Test Sequence)

在這裡使用第 3 組測試影像來做測試。參考圖 73 擺放示意圖，圖 140、圖 141 分別為左右 Kinect 感應器輸出的影像。

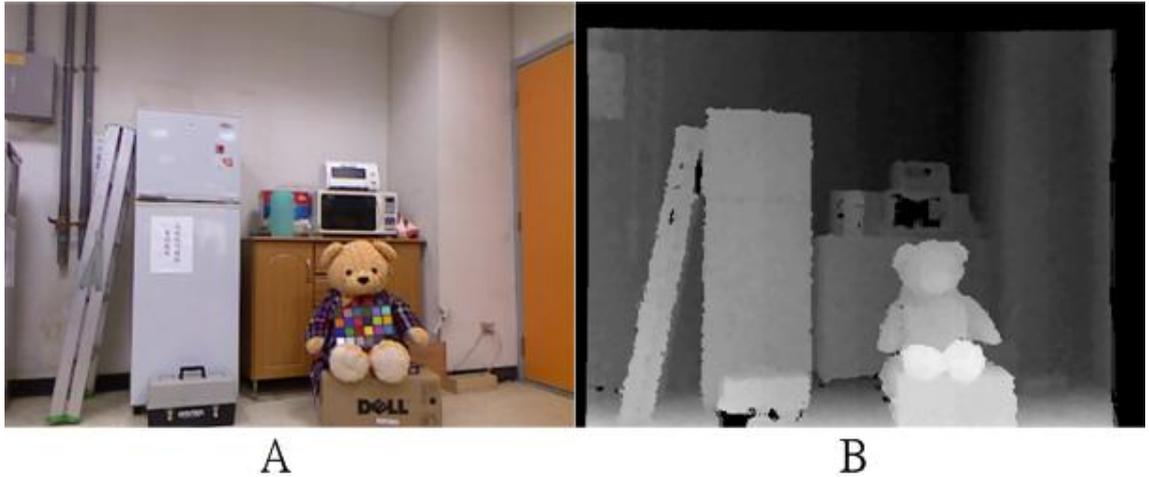


圖 140 A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像



圖 141 A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像

且圖 140 B、圖 141 B 是考慮「校準處理」(4.2 節)中 B1、B2 和 B3 後的輸出影像。

5.3.1 校正處理

相機校正(A1)的步驟，我們使用[32]的相機校正軟體。其原始左右相機參數與式(36)、式(37)、式(38)和(39)類似。

校正處理(A)的第二步是對左右的彩色影像作色彩校正(A2)，參考色彩校正演算法(2.3節)針對圖 140 A 和圖 141 A 做處理，其結果如圖 142 所示。



圖 142 A: 左邊參考影像 B: 右邊校正影像

從圖 142 會看到兩張影像的色彩仍無法完全一樣，圖 142 B 牆壁比較圖 142 A 有點偏紅色，但相較於圖 141 A 未校正前的影像，圖 142 B 影像整體偏暗，整體的色調較接近圖 142 A 的參考影像。此組左右測試影像相較於 5.1 節與 5.2 節的左右測試影像，色調較為接近，所以做完色彩校正以後，左右影像的色調也較前兩組測試影像接近。

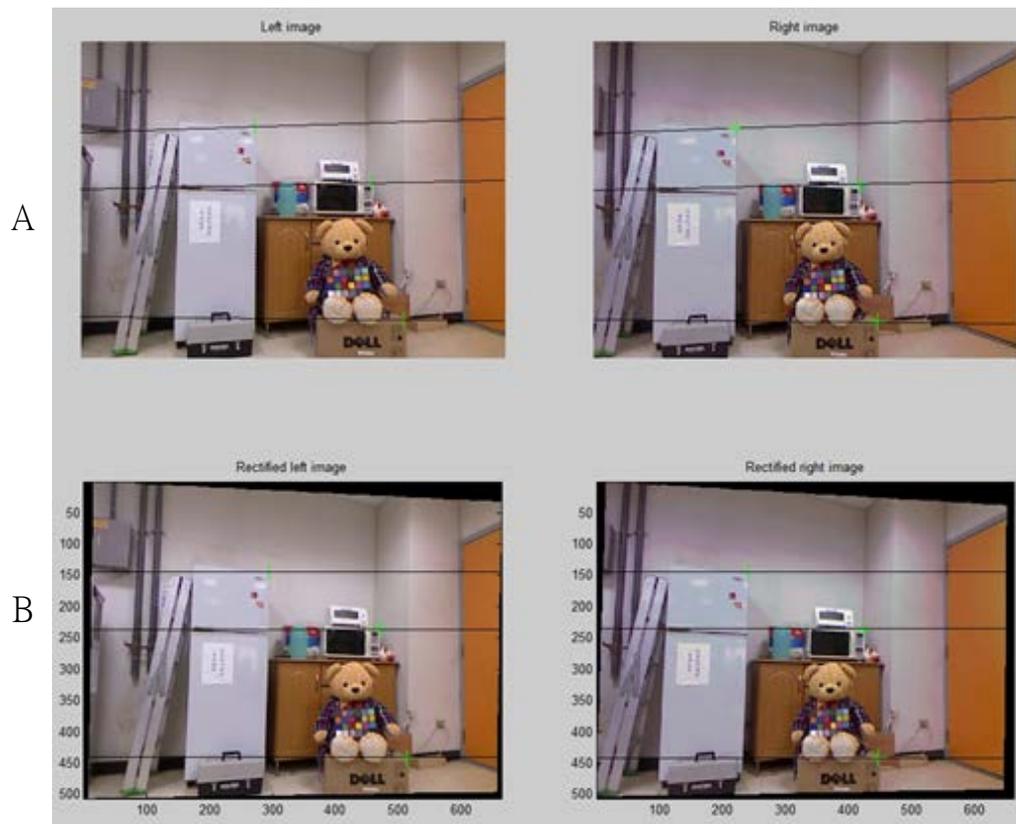
在影像校正(C3)中，利用 2.2 節的影像校正演算法，設法把雙相機投影到虛擬位置上，讓這兩台相機可以達到「極幾何」(Epipolar Geometry)上的平行。最後影像校正完的相機參數與式(40)、式(41)、式(42)和式(43)類似。輸出影像如圖 115 所示。



A

B

圖 143 A: 影像校正後左邊影像 B: 影像校正後右邊影像



A

B

圖 144 A: 原始左右影像 B: 校正後左右影像

參考圖 143，我們可看出影像校正後的結果是合理的。再參考圖 144 我們利用極線幾何來做驗證。圖 144 A 分別是原始左右影像，可看到該原始影像的極線並不水平；圖 144 B 為校正完的左右影像，可看到兩影像的極線順利變成水平，則表示影像校正順利完成。

5.3.2 校準處理

Kinect SDK 讀取資料串流的時候，已考慮「校準處理」(B)中 B1 步驟的「使用 Kinect SDK 函數」、B2 步驟的「景深圖垂直位移」和 B3 步驟的「考慮視差模型的水平位移」做過考慮。所以這裡只針對最後一步 B4 步驟的「景深圖映射」做考慮。

參考 2.5 節的景深映射演算法。為保留重要的資訊，我們針對校正後的彩色影像(圖 143)與映射後的景深影像作切割。最後結果可參考圖 145 與圖 146。

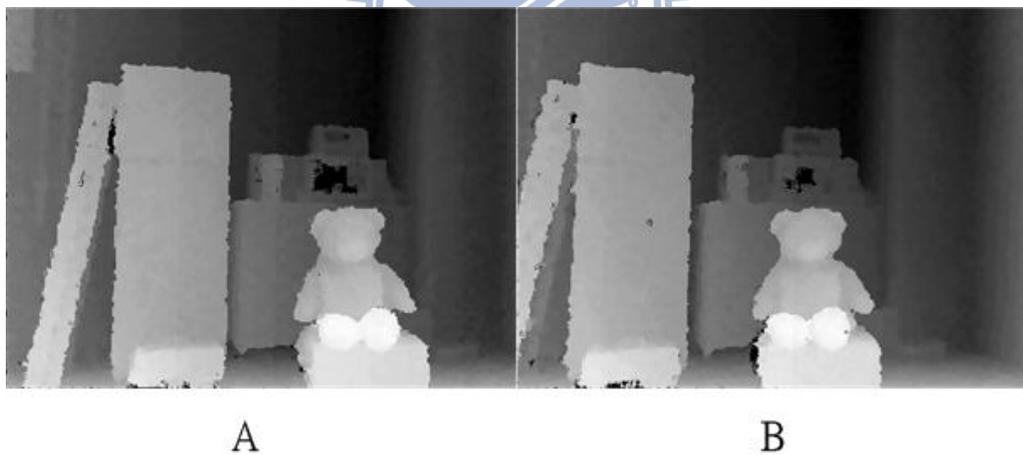


圖 145 A: 切割映射後左邊景深影像 B: 切割映射後右邊景深影像



圖 146 A: 切割後左邊彩色影像 B: 切割後右邊彩色影像

5.3.3 景深影像改善

在「景深影像改善與合成」(C)的部分。第一步考慮 C1 步驟的「濾除雜訊」，來濾掉景深影像邊界的雜訊。考慮 4.3.1 節的聯合雙向濾波器演算法，以圖 145 當作輸入影像，則可得到聯合雙向濾波後的影像(如圖 147 所示)。各參數值可參考表 5。

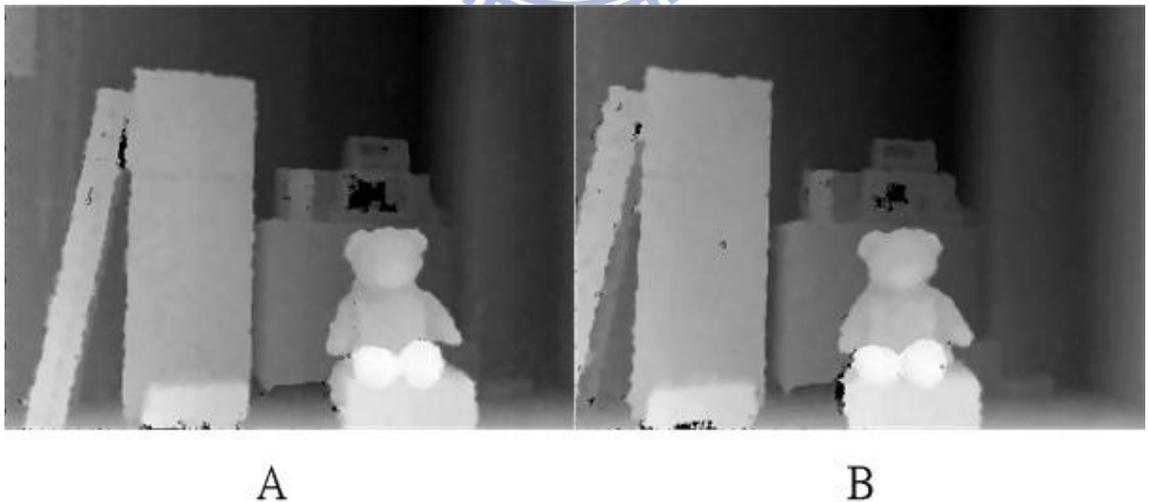


圖 147 A: C1 - 左邊景深影像 B: C1 - 右邊景深影像

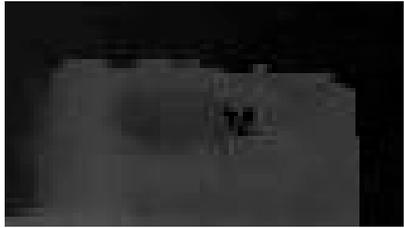
	Left Image	Right Image
Original		
C1		

圖 148 C1 - 細部比較示意圖(1)

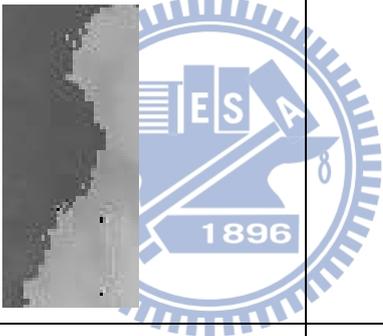
	Left Image	Right Image
Original		
C1		

圖 149 C1 - 細部比較示意圖(2)

比較圖 147 與圖 145，明顯地邊緣變得更平滑，大部分的邊緣雜訊，經由聯合雙向濾波後都被濾除，且藉由彩色影像的資訊讓邊緣保留下來。參考圖 148 的烤箱部分，不管是左右原始影像，邊緣雜訊都十分明顯，經過濾波器以後，邊緣

雜訊消失變得較平滑。參考圖 149，雖然雜訊沒有圖 148 那麼明顯，但經過濾波器做濾波後，同樣也得到很好的結果。

在圖 147 中，洞(未知景深資訊點)的面積雖然不大，但仍然顯而易見。在此我們使用 C2 演算法-「消除缺陷問題」來修補景深影像中未知的景深資訊。參考消除缺陷問題(4.3.2 節)演算法，針對缺陷問題利用遞迴的方式來逐漸消除。其參數設定可參考表 6，結果可參考圖 150。

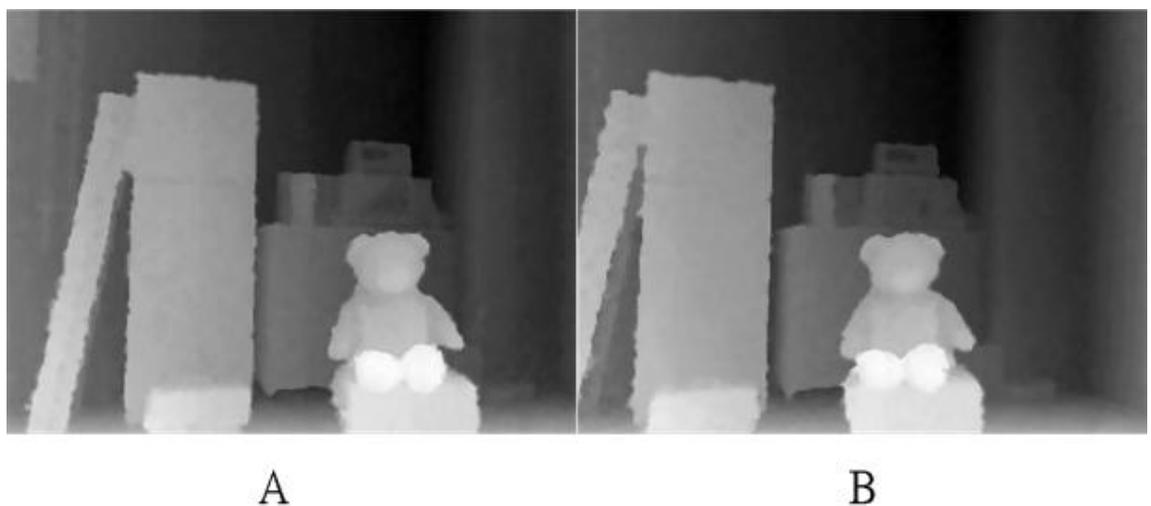


圖 150 A: C2 - 左邊景深影像 B: C2 - 右邊景深影像

	Left Image	Right Image
Original		

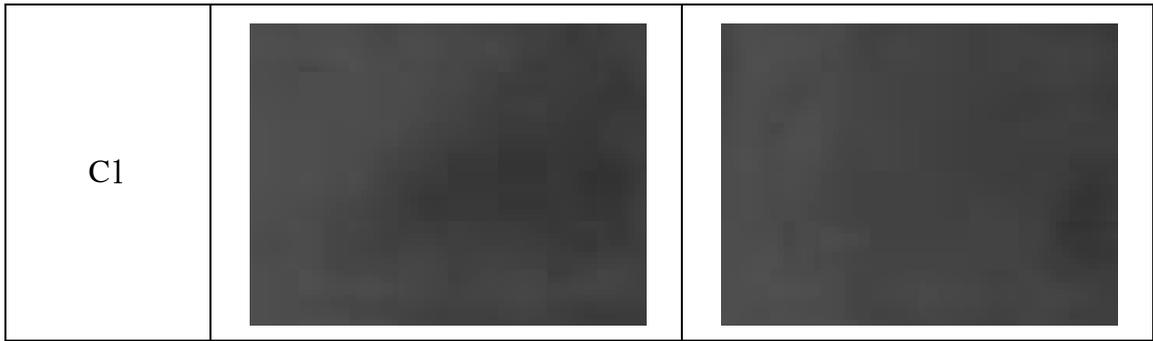


圖 151 C2 - 細部比較示意圖(1)

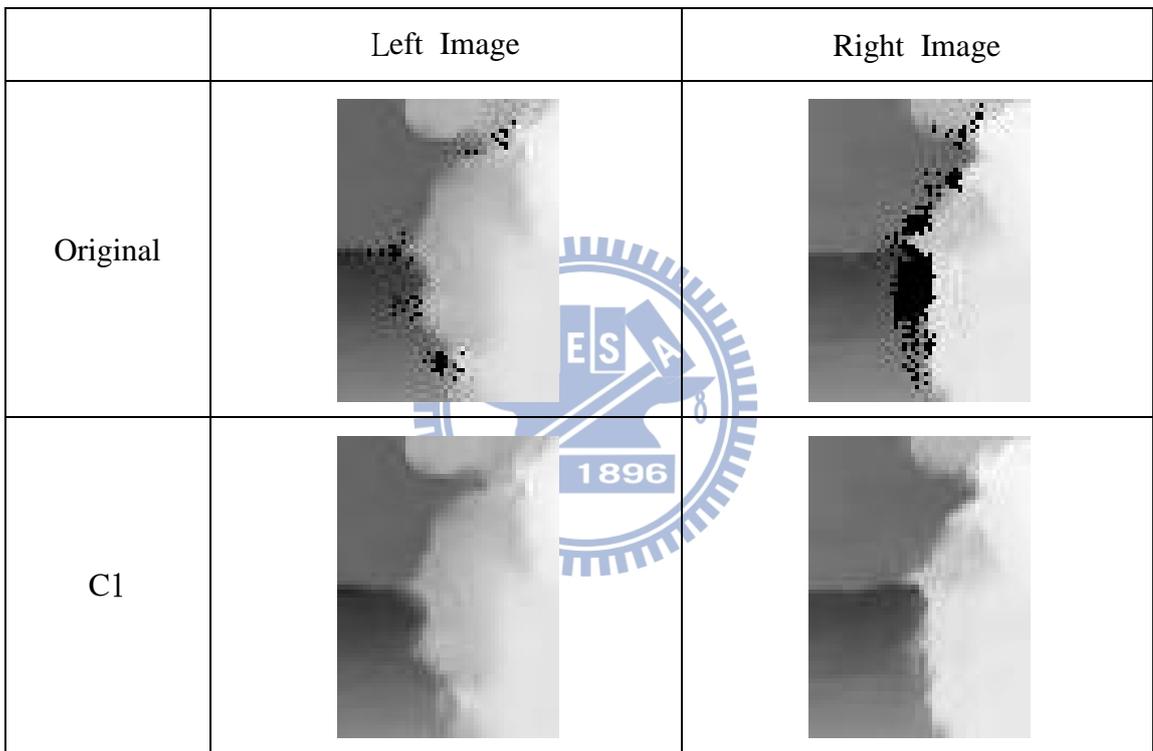


圖 152 C2 - 細部比較示意圖(2)

參考圖 150 對照圖 147，可看出圖 150 的左右影像的瑕疵現象已順利被移除。從細部影像來看，在圖 151 大面積的「洞」明顯消除。在圖 152 B 中，邊緣部分因為聯合雙向濾波器的優點(保留邊緣資訊)，所以物體邊緣的景深資訊完整的被保留下來。

對於圖 150 來說，C1 與 C2 步驟分別改善邊緣雜訊與缺陷問題，但相較於圖 146，部分物體邊緣的景深資訊仍與彩色影像不匹配。所以第三步參考 4.3.3 節

C3 演算法-「基於彩色影像改善景深影像」(左邊方塊大小為 19×19 ，右邊方塊大小為 23×23 ， $th=10$)，針對圖 150 做加強改善，其結果可參考圖 153。

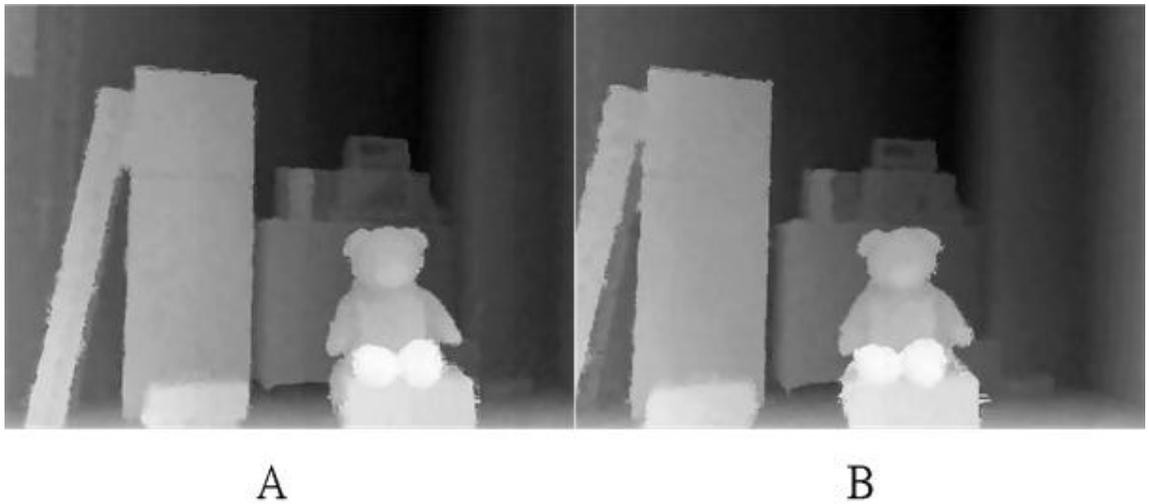


圖 153 A: C3 - 左邊景深影像 B: C3 - 右邊景深影像

	Left Image	Right Image
Original		
C1		

圖 154 C3 - 細部比較示意圖(1)

	Left Image	Right Image
Original		
C1		

圖 155 C3 - 細部比較示意圖(2)

參考圖 153 相較於圖 150，並無明顯的差異。C3 演算法的目的主要是為讓物體景深資訊可以與彩色影像更匹配，所以圖 153 在物體邊緣殘留一些胡椒鹽雜訊，但其實我們的目的已經達到，而這些雜訊可以在 C4 步驟的中值濾波去濾除。參考圖 154 與圖 155 的細部比較，這兩組細部比較的共通點，都是改善後的影像邊緣較原始影像少雜訊，且與原始圖比較也會較匹配。

如上一段所提到的胡椒鹽雜訊問題，在邊緣部份會造成凹凸不平的狀況。在這裡我們使用 C4 步驟的中值濾波(5×5 大小的中值濾波器)，來解決雜訊問題。

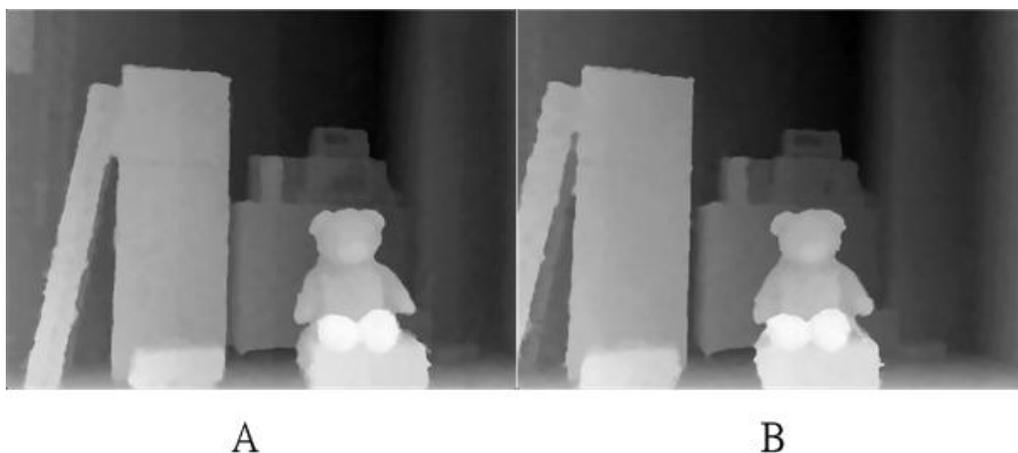


圖 156 A: C4 - 左邊景深影像 B: C4 - 右邊景深影像

	Left Image	Right Image
Original		
C1		

圖 157 C4 - 細部比較示意圖(1)

	Left Image	Right Image
Original		
C1		

圖 158 C4 - 細部比較示意圖(2)

參考圖 153 與圖 156，明顯改善的部份為邊緣雜訊幾乎濾除，尤其是前景(小熊)的部份效果更加明顯。參考圖 157 與圖 158 的細部比較，可以看到原始影像的邊緣雜訊十分明顯，經過中值濾波器後，大部分的邊緣因為雜訊被濾除而變得更平滑。

5.3.4 影像合成

使用 MPEG 的影像合成參考軟體(VSRS)，針對經過「影像改善」(C)每個步驟的合成影像與原始影像做比較。參考圖 159 為原始合成影像，圖 133 為景深影像改善後的合成影像。



圖 159 原始合成影像(圖 146 與圖 145 所合成)

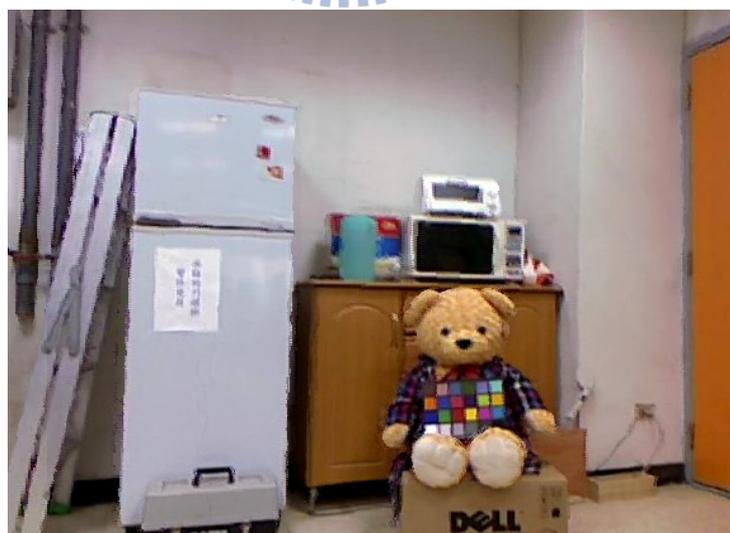


圖 160 影像改善後(C)合成影像(圖 146 與圖 156 所合成)

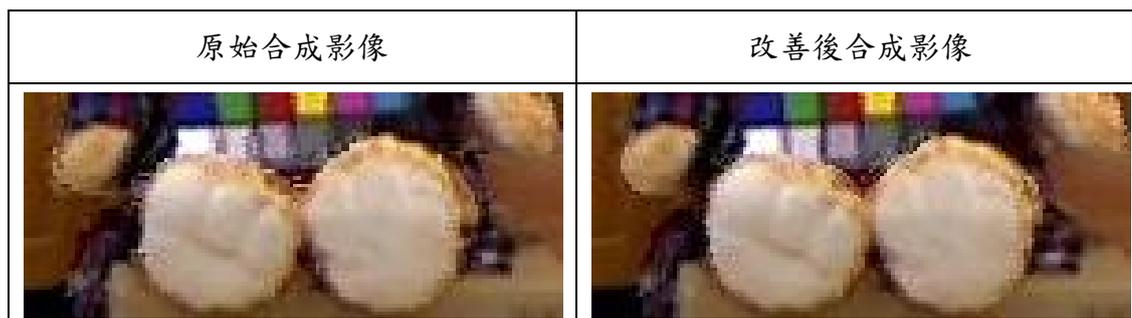


圖 161 合成影像細部比較示意圖(1)



圖 162 合成影像細部比較示意圖(2)

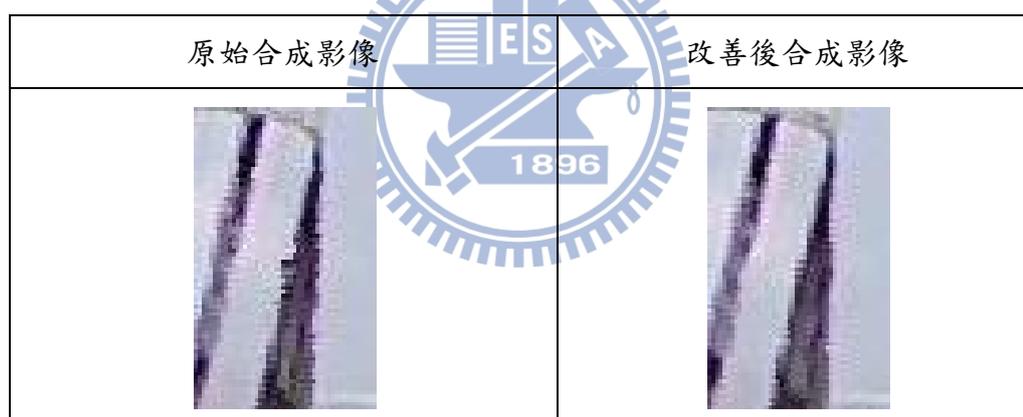


圖 163 合成影像細部比較示意圖(3)

參考圖 159、圖 160，原始合成影像邊緣雜訊明顯，尤其是前景小熊的部份，在視覺上會顯得很不自然；相較於改善後的合成影像，除背景部份(如牆壁、門)外，邊緣雜訊的成分有明顯的減少，整體看起來較自然。

參考圖 161 的細部比較，原始影像的小熊腳雜訊明顯，改善後影像的大部分邊緣雜訊濾除後，也變得較為自然。參考圖 162 的細部比較，原始合成影像在左邊的熊耳明顯的凹凸不平，相較於改善後的影像，在該部分的耳朵較為完整。圖

163 為梯子與冰箱交界的影像，原始影像的梯子有明顯的凹陷處(在圖 159、圖 160 會更明顯)，在改善後的合成影像內，梯子較為平順且凹陷處也消失了。

5.3.5 被動式景深影像與合成結果比較

比較使用 MPEG 標準景深估測軟體(DERS)所估測的景深影像。參考由圖 146 藉由 DERS(2.4 節)估測出的景深影像。

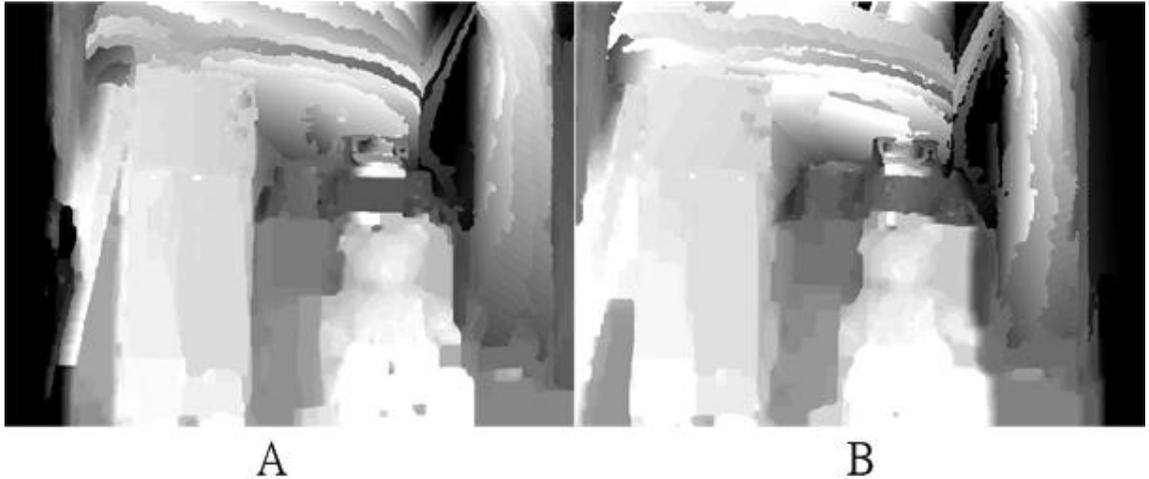


圖 164 A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像

在圖 164 中，明顯可以看到背景資訊十分雜亂(對照圖 146)，在牆壁的景深資訊十分不一致，前景的小熊景深邊緣也與圖 146 不一致。此外大面積的遮蔽區域(如圖 164 A 的最左邊，圖 164 B 的最右邊)。利用 MPEG 所提供的合成參考軟體-VSRS，基於圖 146 與圖 164 實現影像合成，其結果可參考圖 165。

參考圖 165，影像看似很好，觀察小細節可發現一些不理想的區域。參考圖 166 A，在背後牆壁的部份明顯很雜亂，因為用來合成的兩張彩色影像(圖 146)在牆壁部分顏色不一致，且錯誤的景深值也會把彩色像素點映射至錯誤的位置。如果原始兩張彩色影像顏色相當一致的話，這部分景深值的問題會比較不明顯。參考圖 166 B，梯子的形狀與顏色也十分不自然，主要原因是圖 164 A 在最左邊是遮蔽區域，所以結合(Merge)時只能參考右邊影像，當右邊景深資訊不正確

時，則會導致合成影像不自然。參考圖 166 C，烤箱的形狀嚴重變形，原因也是景深資訊造成的嚴重錯誤。



圖 165 被動式景深影像合成結果

細部區域(A)	細部區域(B)	細部區域(C)
		

圖 166 圖 165 細部區域示意圖

5.4 第4組測試影像模擬與結果 (Simulation of Fourth Test Sequence)

使用第4組測試影像來做測試。參考圖 73 擺放示意圖，圖 167、圖 168 分別為左右 Kinect 感應器輸出的影像。



圖 167 A: 左邊 Kinect 輸出彩色影像 B: 左邊 Kinect 輸出景深影像



圖 168 A: 右邊 Kinect 輸出彩色影像 B: 右邊 Kinect 輸出景深影像

且圖 167 B、圖 168 B 是考慮「校準處理」(4.2 節)中 B1、B2 和 B3 後的輸出影像。

5.4.1 校正處理

在相機校正(A1)的步驟，我們使用[32]的相機校正軟體。其原始左右相機參數與式(36)、式(37)、式(38)和(39)類似。

校正處理(A)的第二步是對左右的彩色影像作色彩校正(A2)，參考色彩校正演算法(2.3節)針對圖 167 A 和圖 168 A 做處理，其結果如圖 169 所示。



圖 169 A: 左邊參考影像 B: 右邊校正影像

比較圖 167 A 與圖 168 A，兩張彩色影像之間顏色並無太多差異，所以改善有限，唯有兩者之間的亮度有差(圖 167 A 稍亮，圖 168 A 稍暗)。比較圖 169 B 與圖 168 A 兩圖，校正後的圖 169 B 也較圖 168 A 稍亮，且圖 169 B 也與圖 169 A 亮度接近。

在影像校正(C3)中，利用 2.2 節的影像校正演算法，設法把雙相機投影到虛擬位置上，讓這兩台相機可以達到「極幾何」(Epipolar Geometry)上的平行。最後影像校正完的相機參數與式(40)、式(41)、式(42)和式(43)類似。輸出影像如圖 170 所示。



圖 170 A: 影像校正後左邊影像 B: 影像校正後右邊影像

參考圖 170，我們可看出影像校正後的結果是合理的。再參考圖 171 我們利用極線幾何來做驗證。上面兩張分別是原始左右影像，可看到該原始影像的極線是歪斜不水平的；比較下面兩張為校正完的左右影像，可看到兩影像的極線是水平的，則表示影像校正順利完成。

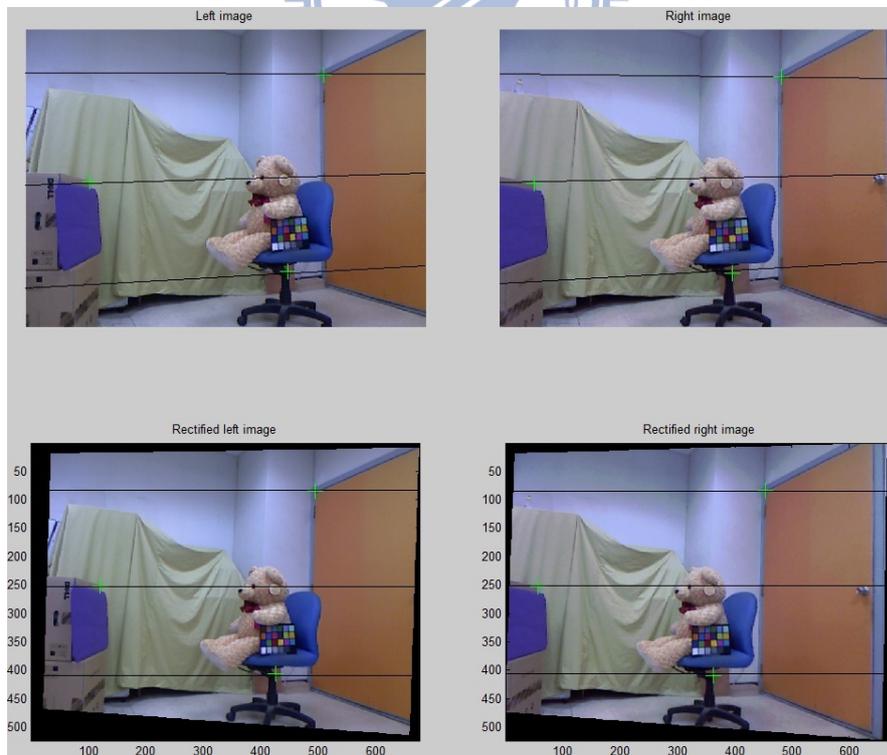


圖 171 A: 原始左右影像 B: 校正後左右影像

5.4.2 校準處理

在 Kinect SDK 讀取資料串流的時候，已考慮「校準處理」(B)中 B1 步驟的「使用 Kinect SDK 函數」、B2 步驟的「景深圖垂直位移」和 B3 步驟的「考慮視差模型的水平位移」做過考慮。所以這裡也只針對最後一步 B4 步驟的「景深圖映射」做考慮。

參考 2.5 節的景深映射演算法，為保留重要的資訊，我們針對校正後的彩色影像(圖 143)與映射後的景深影像作切割。最後結果可參考圖 172 與圖 173。

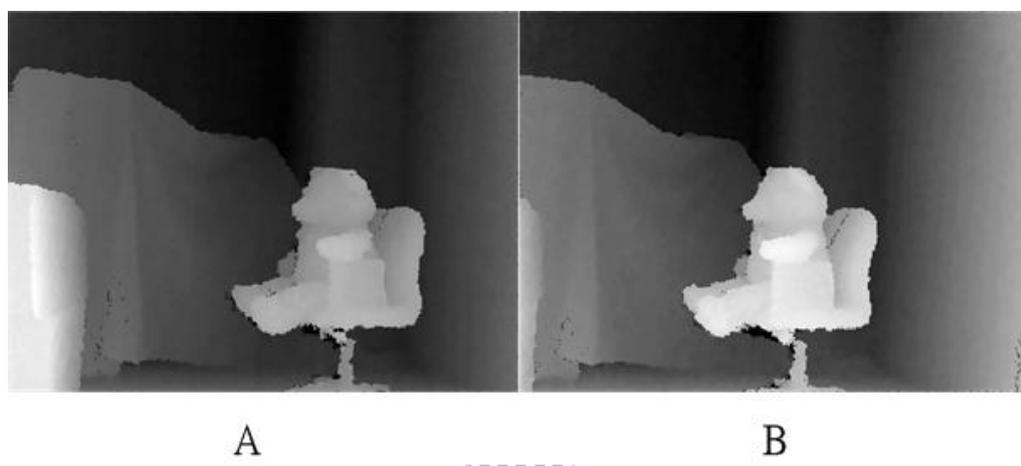


圖 172 A: 切割映射後左邊景深影像 B: 切割映射後右邊景深影像



圖 173 A: 切割後左邊彩色影像 B: 切割後右邊彩色影像

5.4.3 景深影像改善

在「景深改善與合成」(C)的部分。第一步考慮 C1 步驟的「濾除雜訊」，來濾掉景深影像邊界的雜訊。考慮 4.3.1 節的聯合雙向濾波器演算法，以圖 172 當作輸入影像，則可得到聯合雙向濾波後的影像(如圖 174 所示)。各參數值可參考表 5。

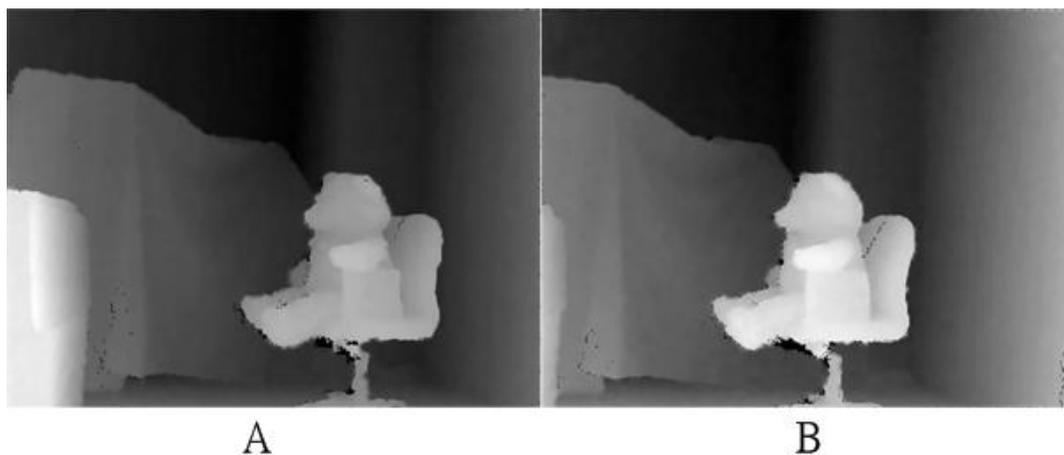


圖 174 A: C1 - 左邊景深影像 B: C1 - 右邊景深影像

	Left Image	Right Image
Original		
C1		

圖 175 C1 - 細部比較示意圖(1)

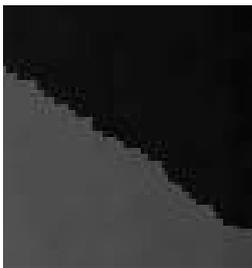
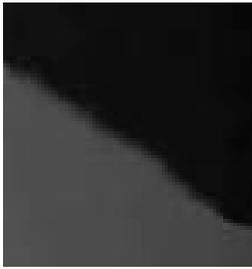
	Left Image	Right Image
Original		
C1		

圖 176 C1 - 細部比較示意圖(2)

參考圖 174，最明顯的是整張影像的邊緣雜訊，經由聯合雙向濾波器濾波，整體變得較平滑。參考圖 175 與圖 176 的細部比較，不管是左右的原始影像，邊緣凹凸不平，代表存在有邊緣雜訊；在濾波後，整個看起來較平滑。

在圖 174 中，「洞」(未知景深資訊點)幾乎位於前景部份，尤其是椅子的下方。在此我們使用 C2 演算法-「消除缺陷問題」來修補景深影像中未知的景深資訊。參考消除缺陷問題(4.3.2 節)演算法，搭配彩色影像資訊，藉由聯合雙向濾波器的優點來保留邊緣且修補瑕疵現象。其參數設定可參考表 6，結果可參考圖 177。

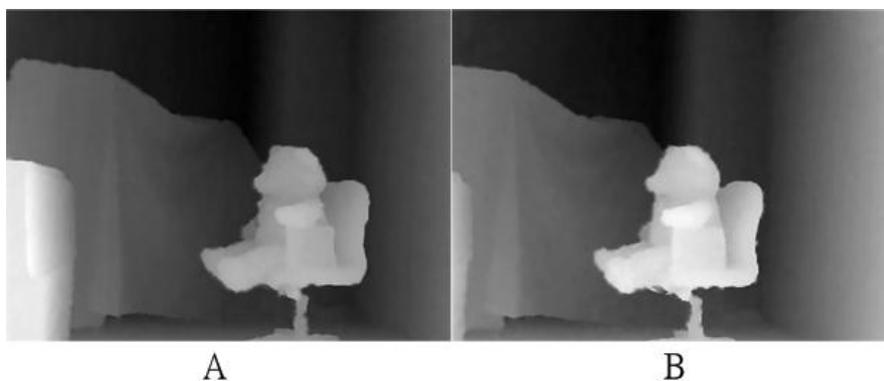


圖 177 A: C2 - 左邊景深影像 B: C2 - 右邊景深影像

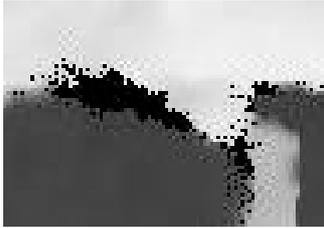
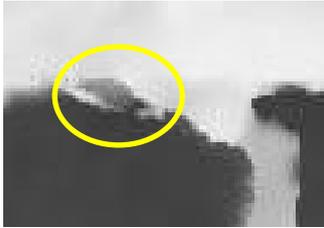
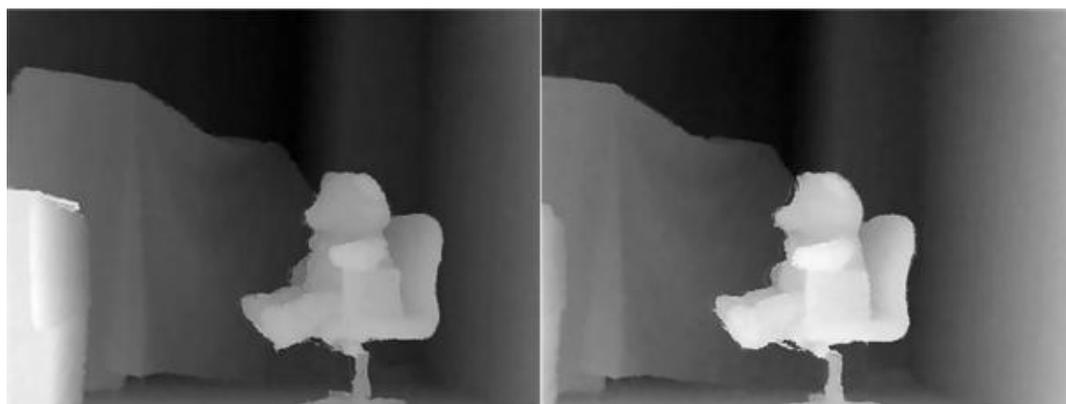
	Left Image	Right Image
Original		
C1		

圖 178 C2 - 細部比較示意圖

在圖 174 的洞的區域並不多，所以在圖 177 也被完整填補。參考圖 178 的細部區域示意圖，在圖 178 右邊影像，黃色區域那細小的座椅高度調整器(參考圖 173 B)也有稍為被填補到，主要原因是聯合雙向濾波器參考彩色影像的資訊；但相對在圖 178 左邊影像，座椅高度調整器就沒有被填補出來。該座椅高度調整器十分細小，在最原始 Kinect 抓取該位置的景深資訊時就十分不容易，在這裡儘管參考彩色影像資訊，能改進的還是很有限。

對於圖 177 來說，C1 與 C2 步驟分別改善邊緣雜訊與缺陷問題，但仍有物體邊緣需要做改善。所以第三步參考 4.3.3 節 C3 演算法-「基於彩色影像改善景深影像」(左邊方塊大小為 23×23 ，右邊方塊大小為 19×19 ， $th=10$)，針對圖 150 做加強改善，其結果可參考圖 179。



A

B

圖 179 A: C3 - 左邊景深影像 B: C3 - 右邊景深影像

	Left Image	Right Image
Original		
C1		

圖 180 C3 - 細部比較示意圖(1)

	Left Image	Right Image
Original		

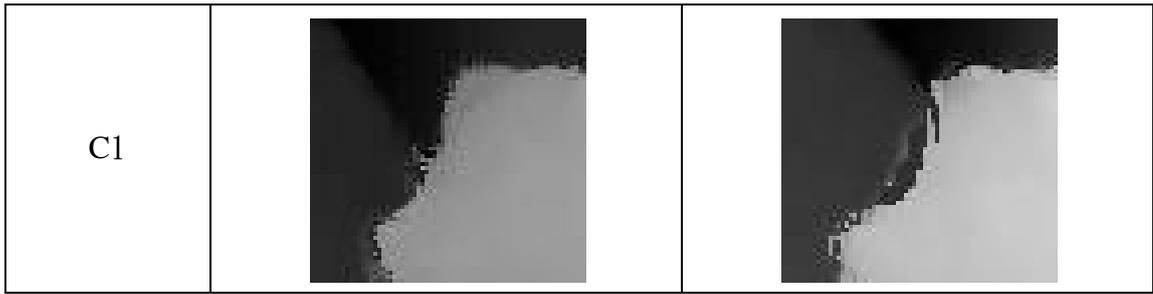
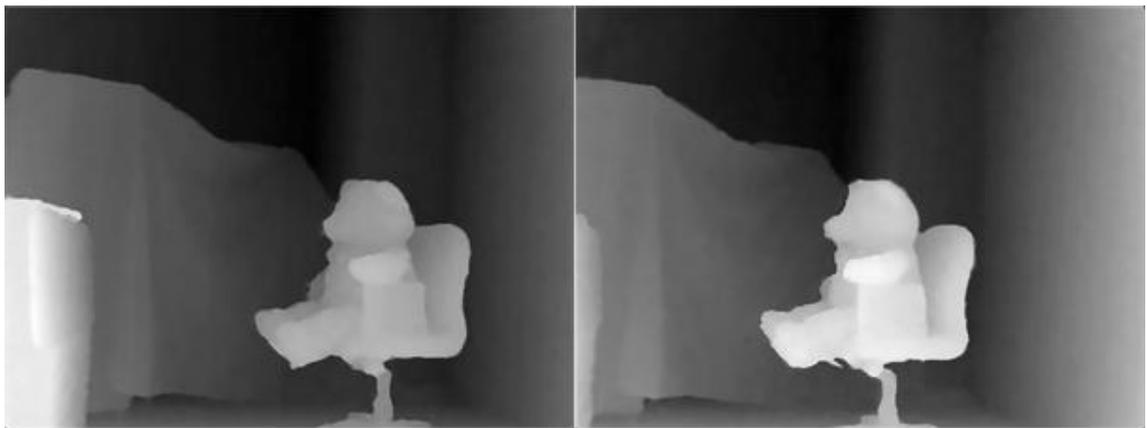


圖 181 C3 - 細部比較示意圖(2)

參考圖 179，改善後的影像與原始影像並無太大差異，C3 演算法主要調整景深影像與彩色影像之間的邊緣匹配問題，所以輸入影像無太大的缺陷時，改善幅度不會太大。但從小地方去看還是看得出差異，參考圖 180 的左邊影像，原始影像的椅腳會缺少一塊，但改善後就有順利填補回來；參考圖 181 左邊影像，原本小熊頭上凸出一塊，改善後也被去除了(仍留有一些邊緣雜訊)，但在右邊影像小熊頭就變得較為不自然，主要原因是圖 173 B 的該區域的顏色與背景顏色相似，所以容易判斷錯誤。

針對圖 179 的胡椒鹽雜訊問題，在邊緣部份會造成凹凸不平的狀況。我們使用 C4 步驟的中值濾波(5×5 大小的中值濾波器)，來解決雜訊問題。



A

B

圖 182 A: C4 - 左邊景深影像 B: C4 - 右邊景深影像

	Left Image	Right Image
Original		
C1		

圖 183 C4 - 細部比較示意圖(1)

	Left Image	Right Image
Original		
C1		

圖 184 C4 - 細部比較示意圖(2)

參考圖 182 與圖 179，兩者之間最明顯的差別是胡椒鹽雜訊的濾除，整張影像較為平滑。參考圖 183，可以更明顯看到小熊頭部的雜訊都被濾除。而圖 184 的左邊影像，在原始圖座椅高度調整器的景深資訊有稍微被填補出來，但因為資訊量還是太少了，所以被濾波器認為雜訊而被濾除；圖 184 的右邊影像，原始影像座椅高度調整器的景深資訊較多，所以濾波後仍有保留大部分的景深資訊。

5.4.4 影像合成

最後是使用 MPEG 的影像合成參考軟體(VSRS)，針對經過「影像改善」(C)每個步驟的合成影像與原始影像做比較。參考圖 185 為原始合成影像，圖 186 為景深影像改善後的合成影像。



圖 185 原始合成影像(圖 172 與圖 173 所合成)



圖 186 影像改善後(C)合成影像(圖 173 與圖 182 所合成)

參考圖 185 的原始合成影像與圖 186 改善後合成影像，最明顯的部份是邊緣雜訊明顯減少，如背景的綠色布最下面與黑色相接的部份以及小熊腳的部份。

至於座椅高度調整器(靠近小熊腳)部份明顯較為失敗，主要原因是該處的景深資訊較微薄，即使步驟 C 內做很多加強，仍無法得到該處合理的景深資訊，所以合成後的影像會有明顯的錯誤。參考圖 187，原始影像在椅背處有明顯的白色條紋，這些白色條紋主要是源自於後面白色的牆壁，因為景深資訊的錯誤，在合成時導致該處的像素值映射錯誤所導致，但在改善後的影像幾乎改善這個問題。在圖 188 的椅腳，原始影像的雜訊十分明顯，原因也是景深資訊錯誤，導致映射到錯誤的位置。相對於原始影像，改善後的影像改善大部分的邊緣雜訊，整個椅腳也較為完整。參考圖 189，小熊的手掌在改善後的合成影像中，會較原始合成影像明顯；此外小熊背部與椅子的交接處在原始影像中有這較多邊緣雜訊，相較於改善後的影像，該處較為平滑，大多數的雜訊都已被濾除。



圖 187 合成影像細部比較示意圖(1)

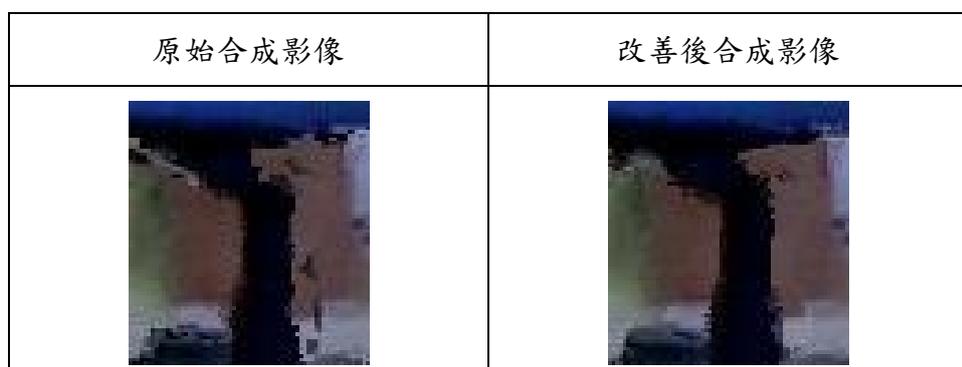


圖 188 合成影像細部比較示意圖(2)

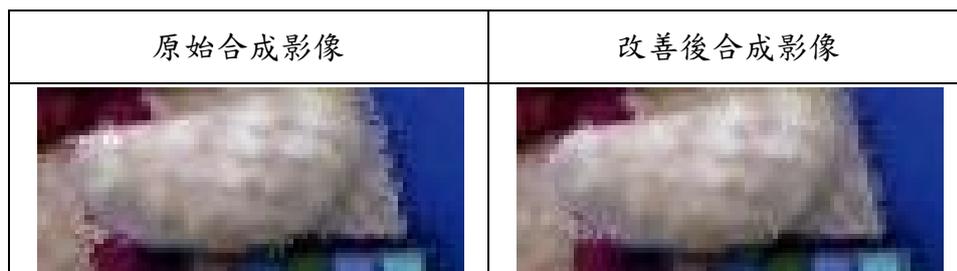


圖 189 合成影像細部比較示意圖(3)

5.4.5 被動式景深影像與合成結果比較

比較使用 MPEG 標準景深估測軟體(DERS)所估測的景深影像。參考由圖 146 藉由 DERS(2.4 節)估測出的景深影像。

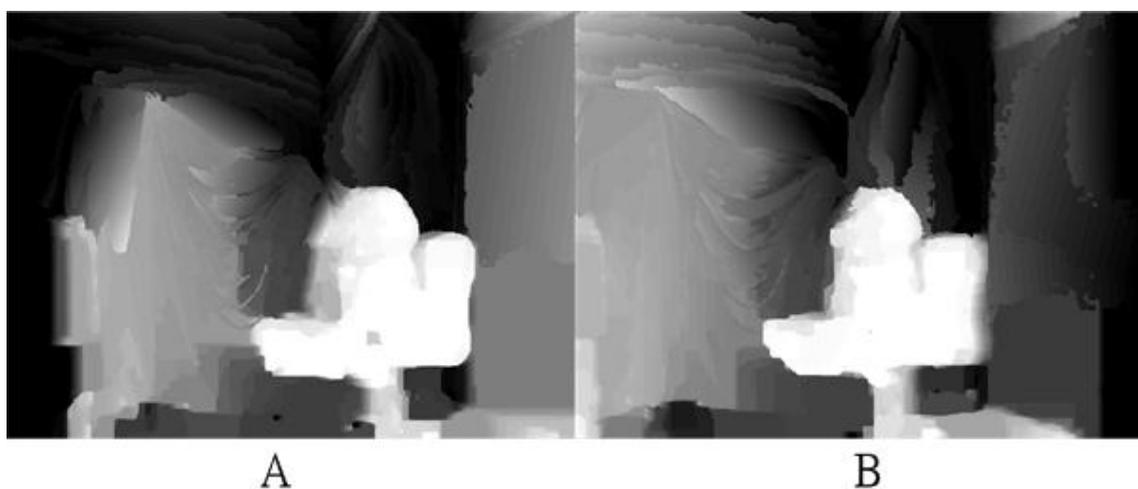


圖 190 A: DERS 估測左邊景深影像 B: DERS 估測右邊景深影像

在圖 190 中，背景資訊(圖 173 的牆壁及綠色布)因為被動式演算法本身的缺點，在特徵點較少的區域估測的景深較易失敗。此外遮蔽區域(如圖 190 A 的最左邊與上面，圖 190 B 的最右邊)在整張影像中也明顯可見。所以利用 MPEG 所提供的合成參考軟體-VSRS，基於圖 173 與圖 190 實現影像合成，其結果可參考圖 191。

參考圖 191 的結果，前景的小熊大致上合成的很好，但是在一些細節的地方還是看得出不合理的地方。參考圖 192 A，椅背的部份跟圖 185 有類似的狀況，因為景深資訊的錯誤，導致白色的牆壁映射到錯誤的位置，尤其在這裡更明顯。

在圖 192 B，也是因為圖 190 A 遮蔽區域的問題，所以只能拿圖 190 B 的資訊當參考，但在這裡由於景深資訊的錯誤，導致有明顯的大小差異，所以影像看起來不自然。在圖 192 C 中，椅腳明顯比圖 186 的結果好，尤其是座椅高度調整器(虛線圓圈)的部份。但在背景棕色的區塊以及椅子最下面(實線圓圈)的部份，跟圖 173 交叉比較後，會發現這部份也不完全正確，其發生主要原因是因為該處的景深資訊是背景的景深資訊，導致映射的結果不完全正確。



圖 191 被動式景深影像合成結果

細部區域(A)	細部區域(B)	細部區域(C)
		

圖 192 圖 191 細部區域示意圖

第6章 結論與未來工作(Conclusion and Future Work)

6.1 結論(Conclusion)

Kinect 感應器從 2010 年上市以來，帶給世界一股新風潮。不管是遊戲方面或者應用方面，常看到的研究多以物體追蹤(Object Tracking)以及物體辨識(Object Reconization)相關。有鑑於多視點視訊近年來在多媒體領域逐漸變成研究主流，我們嘗試將 Kinect 感應器用於多視點視訊的應用上。

Kinect 的優點很多(如:景深資訊大多比被動式演算法估出的準確、可以即時產生景深資訊等等)，但用於多視點合成上仍有很多缺點待改進。本論文針對 Kinect 感應器偵測的景深資訊，以及彩色影像提出多個待解決的問題(第 4 章)。並且把這些問題作分類討論，並且使用三種處理(校正處理、校準處理和影像改善)分別解決這些問題。

1. 校正處理: 有別於使用理想的測試影像，我們考慮真實環境可能發生誤差(如:相機參數的誤差問題、感應器接收的光源特性不同等等)，針對這些誤差，我們提出適當的方法來減少這些誤差的發生，降低這些誤差對於合成影像的影響。
2. 校準處理: Kinect 的彩色攝影機與景深感應器，位於不同的位置上，所以看到的視角會有所差異。對於多視點合成演算法，我們需要一張彩色影像與對應的景深影像來做合成，所以校準處理在這裡是相當重要的一個步驟。對於 Kinect SDK 而言，因為無法順利取得景深感應器的相機參數，所以我們利用多種演算法來解決它們之間不匹配的問題。
3. 影像改善: 對於 Kinect 的景深資訊雖然大致上精準，但在物體的邊緣會有雜訊干擾，以及一些 CMOS 影像感應器無法接收到反射紅外線的位置，所出現的「洞」。我們利用聯合雙向濾波器保留邊界的特性，加上景深改

善演算法以及中值濾波演算法，讓景深影像消除大多數的雜訊，如此更
有利於最後合成影像的品質。

在我們最後的實驗結果，並沒有用實際的數值去量化我們的結果表現，其主要原因可分為三類。

1. 2 台 Kinect 感應器的鏡頭物理距離大約為 30 公分，其距離已遠超於一般的密集相機陣列(Dense Camera Array)，若中間還要擺一台 Kinect 感應器，則左右感應器拍攝出的影像位移會太大。
2. 量化數值的計算常以像素值的相似程度(如 PSNR)來做比較，在圖 34 中明顯可看到，Kinect 感應器的自動白平衡系統並不是很好，即使中間有台感應器當作目標影像，我們合成出來的影像與目標影像之間色彩特性的差異，也會影響量化的數值成效。
3. 由於我們已做過影像校正(2.2 節)，並為保留重要資訊而切除部分影像資訊，所以最後合成影像與目標影像作比較的話，只能針對某些前景資訊作數值量化的比較，而不是對整張影像作比較。所以這並不是一個很好的數值比較方式。

基於我們的演算法，比較改善前後的合成影像，從主觀的感受上會看到明顯的改善變化。比較被動式景深影像所合成出的虛擬視點影像，在某些區域未必會比使用主動式感應器得到的景深資訊差。如何加強主動式景深影像的景深資訊精確度，是一門重要的課題。

6.2 未來工作(Future Work)

我們的演算法對於主動式景深影像有很明顯的改善，尤其是在合成影像的結果可看出改善前與改善後的差異。且我們的演算法大多簡單易懂，但仍有幾個可供參考的未來工作，來做為改善此演算法的目標(1~2);對於環境的架設也有幾點可供參考(4~6)。如下所示。

1. 在 $C1$ (式(28))與 $C2$ (式(33))可以由程式自動的決定最好的標準差數值，讓影像的表現可以達到最大。
2. 在 $C3$ 的參考方塊大小，可以自動的由程式決定最好的大小，讓整體的表現達到最好。
3. 將此演算法從單一影像延伸至影片(Video)上。
4. 將此演算法用於不同種的景深感應器(如:SR4000、Xtion Pro)
5. 使用不同平台(如:OPENNI)產出的景深影像套用至我們的演算法上。
6. 針對主動式景深影像的缺點，開發出適合的合成演算法。



參考文獻

- [1] A. Kubota , A. Smolic , M. Magnor , M. Tanimoto , T. Chen and C. Zhang "Multiview imaging and 3DTV", *IEEE Signal Process. Mag.*, vol. 24, pp.10 - 21, 2007.
- [2] A. Smolic and D. M. Cutchen "3DAV exploration of video-based rendering technology in MPEG", *IEEE Trans. Circuits Syst. Video Technol., Special Issue on Immersive Commun.*, vol. 14, pp.348 - 356, 2004.
- [3] M.Tanimoto, "Overview of free viewpoint Television," *Signal Process. Image Commum*, vol. 21, no. 6, pp. 454 - 461, July 2006.
- [4] C.Fehn, "Depth-Image-Based Rendering (DIBR), compression and transmission for a new approach on 3D-TV," *Proc. SPIE Stereoscopic Displays and Virtual Reality Systems XI*, pages 93 - 105, San Jose, CA, USA, January 2004.
- [5] Microsoft, "XBOX360+Kinect," <http://www.xbox.com/zh-TW/Kinect/>, 2012.
- [6] Y.S. HO and E.K. Lee, "3DTV: Technical challenfes for realistic experiment," in *Interational Workshop on Advanced Image Technology*, Jakarta, Indonesia, pp. K3 1-6, Jan. 2011.
- [7] K. Khoshelham and S.O. Elberink, "Accuracy and resolution of Kinect depth data for indoor mapping application," *Sensors*, vol. 12, no. 2, pp. 1437-1454, 2012.

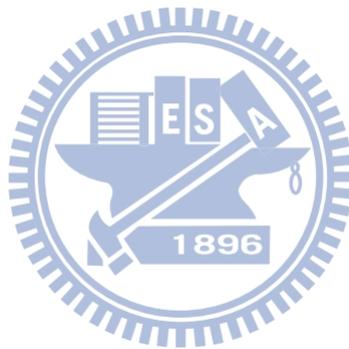
- [8] J. Smisek, M. Jancosek, and T. Pajdla, “3d with Kinect,” in *ICCV Workshop*, Barcelona, Spain, pp. 1154-1160, 2011.
- [9] Heresy, “OPENNI 環境/Kinect 安裝教學,” http://kheresy.wordpress.com/2012/03/15/2012_openni_env_kinect_setup/, 2012.
- [10] D. Herrera, J. Kannala, and J. Heikkila, “Accurate and practical calibration of a depth and color camera pair,” in *International Conference on Computer Analysis of Images and Patterns*, Spain, pp. 437–445, 2011.
- [11] Programming Guide: Getting Started with Kinect for Windows SDK Beta from Microsoft Research, Microsoft Research, July 2011.
- [12] Marduk, “初探 Kinect 背後的 3D 測量技術,” <http://blog.uns.org.tw/node/417>, 2010.
- [13] 曹祖聖, “Kinect for Windows 開發,” <http://msdn.microsoft.com/zh-tw/hh367958.aspx>, 2012.
- [14] Z. Zhang. “A flexible new technique for camera calibration,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330-1334, 2000.
- [15] A. Fusiello, E. Trucco, A. Verri , “A compact algorithm for rectification of stereo pairs,” *Machine Vision and Applications*, vol. 12, pp. 16–22, 2000.
- [16] F. Porikli, Inter-camera color calibration using cross-correlation model function,

in *IEEE Int. Conf. on Image Processing*, vol. 2, pp. II-133-6, 2003.

- [17] S. Lee and Y. Ho, "Real-time Stereo View Generation using Kinect Depth Camera," in *Proc. APIPA Annual Summit and Conference(APSIPA ASC)*, Xi'an, China, pp. RS3.12(1-4), Oct 2011.
- [18] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, Second Edition, Cambridge University Press, March 2004.
- [19] ISO/IEC JTC1/SC29/WG11, N11631, "Report on Experimental Framework for 3D Video Coding," October 2010.
- [20] PrimeSense, "3D Sensing Solution," <http://www.primesense.com/en/technology/115-the-primesense-3d-sensing-solution>, 2010.
- [21] NUI API and Programming – Kinect for Windows SDK, "Image rectification between RGB and depth camera," <http://social.msdn.microsoft.com/Forums/en-US/kinectsdknuiapi/thread/40edbbc6-d0b1-4f14-b6a1-4a173c4dc588/>, 2011
- [22] H. Yu, G. Yang, Y. Zhou, Z. Jin, "Improving Depth Estimation through Fusion of Stereo and TOF," *International Conference Multimedia Technology(ICMT)*, Beijing, China, pp. 26-28, 2011.
- [23] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," *International Conference on Computer Vision(ICCV)*, Bombay, India, vol. 846, pp. 839-846, 1998.

- [24] Lai P., Tian D., and Lopez P., “Depth map processing with iterative joint multilateral filtering,” in *Picture Coding Symposium (PCS)*, Nagoya, Japan, pp. 9-12, dec. 2010.
- [25] M. Camplani, L. Salgado, “Efficient spatio temporal hole filling strategy for Kinect depth maps,” *IS&T/SPIE Int. Conf. on 3D Image Processing (3DIP) and Applications*, CA, USA, pp. 82900E 1-10, Jan. 2012.
- [26] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd Edition Prentice Hall, 2008.
- [27] A. Olofsson, “Modern Stereo Correspondence Algorithms: Investigation and evaluation,” Thesis from Dept. of Electrical Engineering, Linköping Univ., Linköping, Sweden, 2010.
- [28] Microsoft Research, *SkeletalViewer Walkthrough C++ and C#: Capturing Data with the NUI API*, July 2011.
- [29] K. Oh and Y. Ho, “Non-linear Bi-directional Prediction for Depth Coding,” *Advances in Multimedia Information Processing*, Bangkok, Thailand, pp. 522-531, 2009.
- [30] 劉楷哲、吳其霖、黃偉豪、陳信榮、李錕和羅豐祥，基於 3D 顯示器格式之即時 3D 內容合成技術，影像與識別，第十六冊，第二期，第 76-95 頁，2010。
- [31] Jean-Yves Bouguet, “Camera Calibration Toolbox for Matlab,”

http://www.vision.caltech.edu/bouguetj/calib_doc/, 2010.



自傳

邱義文，1988年1月3日出生於高雄市，高雄市人。2010年6月畢業於國立中興大學電機工程學系。2010年7月於國立交通大學電子研究所攻讀碩士學位。承蒙杭學鳴教授的指導，進入通訊電子與訊號處理實驗室(CommLab)。論文題目為「使用景深感應器與RGB相機對用於影像合成的景深改善」

