공학박사 학위논문

적층형 3D 라이트 필드를 기반 자유 시점 변환을 지원하는 가상현실 시스템

A Stackable 3D Light Field System for Free Viewpoint Virtual Reality

2020 년 8 월

서울대학교 대학원

전기 정보 공학부

정 현 민

적층형 3D 라이트 필드를 기반 자유 시점 변환을 지원하는 가상현실 시스템

A Stackable 3D Light Field System

for Free Viewpoint Virtual Reality

지도 교수 이 혁 재

이 논문을 공학박사 학위논문으로 제출함

2020 년 8 월

서울대학교 대학원

전기 정보 공학부

정 현 민

정현민의 공학박사 학위논문을 인준함

2020 년 8 월

위 원 장 조 남 익 (인)

부위원장 이 혁 재 (인)

위 원 고 형 석 (인)

위 원 김 영 민 (인)

위 원 이 채 은 (인)

초 록

기존의 촬영된 이미지를 기반으로 하는 가상현실 (VR: Virtual Reality) 시스템은 3-degree-of-freedom (3-DoF)을 만족하며, roll, yaw, pitch의 회전 변환만을 지원하였다. DoF는 사용자가 움직일 수 있는 방향을 의미하며, 최대 6-DoF가 있다. roll, yaw, pitch의 세 가지 회전 변환과 더불어 x, y, z 축을 따라 이동하는 수평 변환을 포함한다. 즉, 기존의 촬영된 이미지 기반의 가상현실 기술은 제한적인 움직임만을 지원하고 있으며, 이는 마치 머리가 고정된 상태에서 고개만 움직이는 것과 같다. 결과적으로 사용자의 몰입도를 크게 낮추는 요인으로 작용한다.

라이트 필드(LF: Light Field)는 자유 공간을 통과하는 빛의 조합을 통해 새로운 시점에서의 뷰를 구성하는 기법으로, 이를 바탕으로 촬영된 이미지 기반의 가상현실보다 DoF를 향상하기 위한 연구가 진행되어 왔다. LF는 가정하는 면을 따라 획득한 light ray의 조합을 통해 임의의 viewpoint에서의 새로운 뷰(view)를 구성한다. 2D 평면 또는 구면을 가정하는 LF가 있으며, 특히 구면을 가정하는 LF 시스템은 360도 방향에서 입사되는 light ray의 조합을 통해 360도 뷰를 만든다. 또한 구 내부에서 viewpoint는 자유롭게 선택될 수 있으며 6-DoF를 만족한다. 하지만 위와 같이 평면이나 구면을 가정하는 LF 구조는 light ray의 획득이 어렵다는 문제가 있다. 특히 구면을 가정한 시스템의 경우 구면의 따라 light ray을 획득하기 위해 아치 모양으로 배치된 다중 카메라를 원으로 회전시키는 특수 장비를 별도로 개발하여 사용한다. 보다 넓은 공간을 커버하기 위해서는 더 큰 평면, 구면을 가정해야 하는데, 큰 면을 가정할수록 light ray 획득 난이도는 더욱 증가한다.

3D LF는 기존의 LF가 면을 가정하는 것과 달리 선을 따라 획득하는 light ray로 구성된다. 평면 대신 직선, 구면 대신 원 구조가 가정되어 LF를 구성한다. 면 대신 선을 가정함으로써 하나의 변수가 고정되고, 4개 변수 대신 3개 변수로 light ray를 표현된다. 이에 따라

획득할 수 있는 light ray의 수가 제한적이며, 특히 vertically 하나의 점에서 획득한 light ray를 사용하기 때문에 vertical parallax를 표현하지 못하는 문제가 있다. 반면, 선을 따라 light ray를 획득하는 과정은 슬라이더나 달리(dolly) 장비와 같은 접하기 쉬운 장비로도 가능하며, 슬라이더, dolly 장비에 장착된 카메라를 이동하면서 간편하게 light ray를 획득할 수 있다. 특히 더 넓은 범위를 움직이기 위해 더 큰 구조를 가정하더라도 획득의 난이도가 크게 증가하지 않는다. 하지만 구조가 커질 경우 vertical parallax를 표현하지 못하는 문제로 인한 뷰가 왜곡되는 에러가 크게 증가한다.

본 논문은 3D LF를 기반으로 보다 넓은 공간을 자유롭게 움직일 수 있는 가상현실 시스템의 개발을 목표로 한다. 기존의 방법에서 넓은 구조를 커버하기 위해 구조 자체를 확장하는 것이 아니라 3D LF를 여러 개 쌓은 형태인 3D LF Stack 구조를 가정하고 이를 통해 커버할 수 있는 범위를 넓힌다. 위의 제안 방안을 바탕으로 여전히 light ray 획득 방식을 간편하게 하면서 동시에 3D LF의 vertical parallax로 인한 에러를 일정 수준으로 제한한다. 또한 제안하는 시스템에서는 3D LF Stack 구조를 수직 방향으로 두 개 배치함으로써 임의의 viewpoint에 대한 360도 뷰를 구성한다. 제안 시스템은 여전히 3D LF의 vertical parallax를 표현하지 못하는 에러를 포함하고 있으며, 이는 특히, viewpoint가 이동하고, 3D LF Stack 상에서 다른 3D LF를 넘어갈 때 두드러지게 나타난다. 이를 개선하기 위해 앞, 뒤로 배치된 두 개의 3D LF를 사용한 뷰 구성 방안을 제안한다. 그리고 제안된 시스템에서 임의의 viewpoint의 뷰는 기존의 LF 기반의 접근 방법과 달리 다수의 LF를 동시에 사용해서 하나의 뷰를 구성한다. 따라서 서로 다른 3D LF의 연결 방안을 제안하고, 다양한 시스템 구현 환경에 따른 적절한 적용 방안을 소개한다.

주요어 : Light Field, Virtual Reality, Free viewpoint, View navigation, View exploration
학 번 : 2016-30216

목 차

표 목차

그림 목차

제 1 장 서 론

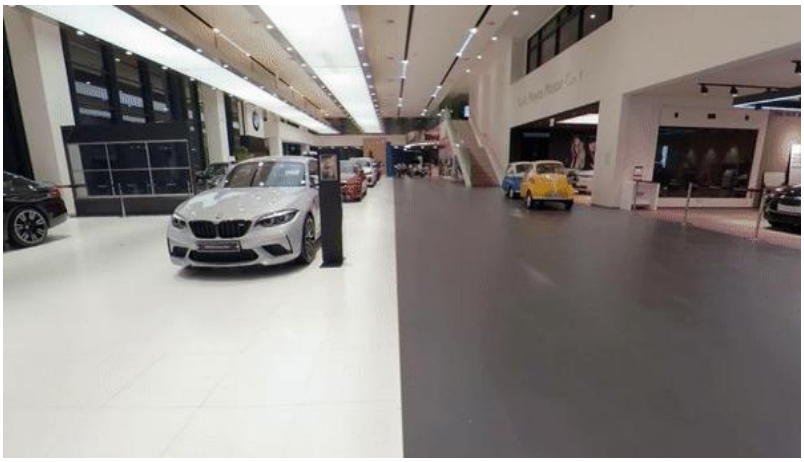
1.1 연구 배경

가상현실 기술은 최근 영화 등을 통해 많은 사람들에게 익숙해지고 있다. Head mounted display (HMD)를 착용한 사용자는 가상의 공간을 자유롭게 이동하면서 다른 사람들과 interaction 하기도 하고, 현실 공간에서 접하기 어려운 우주, 심해와 같은 공간을 체험하기도 한다. 하지만 사람들의 기대와 달리 현재 가상현실 기술의 수준은 제한적이다. 특히, 고정된 viewpoint는 가상현실의 실감도를 낮추는 요인으로 작용하고 있다.

가상현실은 크게 두 가지로 분류된다. 하나는 컴퓨터 그래픽 (CG: Computer Graphics) 기반의 가상현실 기술이다. 가상의 공간을 인위적으로 생성하고, 해당 공간에서 사용자들은 자유롭게 움직인다. 인공적으로 제작된 공간에 대한 3차원 정보를 모두 보유하고 있으며, 이를 바탕으로 자유로운 시점 변화가 가능하다. 실제 환경이 아닌 인공의 공간을 대상으로 하기 때문에 게임, 애니메이션 등의 분야에 적합하다.

다른 하나는 촬영된 이미지 기반의 가상현실 기술이다. 최근 다수의 IT기업에서 360도 카메라를 출시하고 있다. 촬영된 이미지 기반의 가상현실 기술은 360도 이미지를 기반으로 한다. 360도 방향으로 촬영된 이미지에서 사용자가 바라보는 방향의 뷰를 보여준다. 사용자는 고개를 돌려가면서 원하는 시점의 뷰를 바라본다. 한 장의 이미지만으로 가상현실을 구현할 수 있다는 점에 사용자들은 쉽게 접할 수 있다. 촬영된 이미지 기반의 가상현실은 별도의 고성능 HMD 기기 없이 스마트폰을 통해 접할 수 있다. 하지만 한 점에서 촬영된 360도 이미지만을 사용하기 때문에 촬영된 이미지 기반의 가상현실 기술에서 사용자는 움직이는 것이 불가능하다. 이는 마치 고개가 고정된 상태에서 고개만 돌리는 것과 같으며 사용자의 실감도를 크게 떨어뜨린다.

(a)



(b)

그림 1.1 (a) 컴퓨터 그래픽 기반의 가상현실, (b) 촬영된 이미지 기반의 가상현실

실제로 촬영된 이미지를 사용하기 때문에 사용자가 바라보는 뷰는 사실적이며, 관광 분야에서 적극적으로 사용되고 있으나, 실감도 문제의 해결이 여전히 중요하다.

본 논문은 촬영된 이미지 기반의 가상현실 기술을 대상으로 한다. 특히 사용자의 보다 자유로운 움직임을 지원하기 위한 방안을 제시한다. 이를 위해 먼저 degree-of-freedom (DoF)를 통해 사용자의 움직임을 정의한다. DoF는 움직일 수 있는 방향을 의미하는데, 최대 6-DoF가 있다. 아래 그림은 6-DoF로 표현할 수 있는 자유도를 보여준다. x, y, z

축을 따라 수평으로 움직이는 수평 변환 세 가지와 각 축을 따라 회전하는 세 가지 회전 변환을 표현한다. 각 회전 변환은 roll, yaw, pitch로 정의된다.

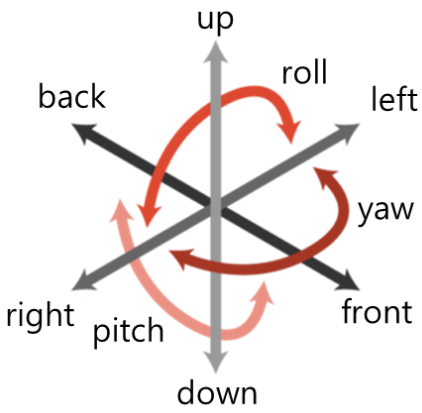


그림 1.2. 6-Degree-of-Freedom (6-DoF)

촬영된 이미지 기반의 가상현실 기술은 세 가지 회전 변환만을 지원할 수 있으며, 결과적으로 3-DoF를 지원한다고 할 수 있다. 본 논문은 촬영된 이미지 기반의 가상현실 기술에서 3-DoF 이상의 자유도 지원으로의 개선을 목표로 한다.

3-DoF 이상의 자유도를 지원하기 위한 다양한 접근의 연구가 있었다 [1-4]. 대표적으로 3D 스캐닝 및 렌더링 방법이 있다 [5-8]. 전통적인 3D 이미지 처리 기술을 통해 다시점에서 촬영된 2D 이미지들로부터 3차원 정보를 획득하고, 이를 바탕으로 공간을 재구성한다. 이는 마치 CG 기반의 가상현실 기술에서 인위적으로 제작하는 가상의 공간을 2D 이미지로 재구성하는 것과 같다. 가상 공간을 재구성한 뒤, CG 기반의 가상현실과 마찬가지로 사용자는 가상의 공간을 자유롭게 돌아다니게 된다. 6-DoF를 지원한다는 점에서 촬영된 이미지 기반의 가상현실 기술의 제한적인 시점 변환 문제를 해결할 수 있다. 하지만 3D 스캐닝 및 렌더링 방법은 계산 복잡도가

높고 에러의 확률이 크다는 단점이 있다. 2D 이미지로부터 3차원 정보를 추정하는 연구는 오래 전부터 진행되어 왔으나 여전히 높은 신뢰도를 보장하지는 못하는 문제가 있다.

또 다른 접근 방법으로 라이트 필드 (LF: Light Field) 기반의 방법이 이용되기도 한다 [9-11]. LF는 자유 공간을 통과하는 light ray를 정의하는 방법으로, 다양한 light ray의 조합을 통해 새로운 뷰를 구성한다. LF는 기존의 3D 이미지 처리 기법에 비해 상대적으로 많은 데이터 량을 바탕으로 복잡한 계산을 줄여주며, 동시에 3D 이미지 처리 기법이 적용되는 depth 추정과 같은 다양한 분야에서 새로운 접근 방법을 제시하고 있다. 특히 임의의 viewpoint에서의 뷰를 구성하기 위해서 해당 viewpoint를 통과하는 모든 light ray를 조합하는 방식은 3차원 정보의 계산을 포함하지 않아 간편하다. 또한 light ray들은 실제 구현에서 촬영된 이미지의 픽셀로 대체되는데, LF를 기반으로 만든 뷰는 visual artifact가 적어 마치 실제로 촬영한 것과 같다는 장점이 있다.

LF를 이용한 뷰 구성을 위해서는 먼저 LF를 구성하는 구조를 정의하고 해당 구조를 따라 light ray를 획득하는 과정이 선행되어야 한다. 2D 평면이 대표적이며, 2D 평면을 통과하는 모든 light ray를 획득하고, 해당 light ray의 조합을 사용한다. Light ray를 획득하는 과정은 카메라를 이용해서 이미지를 촬영하는 것으로 대체된다. 2D 평면을 통과하는 모든 light ray를 획득하기 위해 2D 평면을 따라 다중 카메라를 배치하거나 2D 평면을 따라 카메라를 이동하면서 이미지를 촬영하기도 한다. 구 모양의 구조를 가정하는 경우, 구면을 따라 카메라를 이동하면서 light ray를 획득하게 된다. 다중 카메라를 사용한 light ray 획득 방법은 카메라 사이의 calibration과 rectification 과정 등의 전처리를 필요로 하며, 한 대 카메라를 움직이는 경우에도 가정하는 구조를 따라 정교하게 움직이며 촬영하는 것은 쉽지 않다.

2D 평면을 가정하는 LF 구조는 x, y, z축의 수평 이동이 가능한 시점 변환을 지원한다. 하지만 한 방향으로만 획득된 light ray를 사용하기 때문에 view의 회전 변환은 제한된다. 구 모양의 면을

가정하는 LF 시스템은 360도 방향으로 획득한 light ray를 바탕으로 view를 구성하게 된다. 구 내부의 다양한 위치에서 시점 변환이 가능함과 동시에 360도 뷰를 바탕으로 시점의 회전 변환이 가능하며, 6-DoF를 모두 지원한다.

3D LF는 2D 평면, 구면과 같이 면을 가정하지 않고, 선을 통과하는 light ray로 구성된다. 따라서 획득할 수 있는 light ray는 제한적이지만, 데이터 관리가 간편하고, 특히 light ray 획득이 간편한 장점이 있다. 슬라이더, dolly 장비와 같은 대중적인 장비에 카메라를 장착하여 가정한 선을 따라 움직이면서 light ray를 획득하게 된다. 2D 평면 대신 선을 가정하거나, 구 대신 원을 가정한다. 3D LF는 vertically 오직 하나의 viewpoint에서만 light ray를 획득하기 때문에 vertical parallax를 표현하지 못하는 문제가 있다.

LF 구조는 보다 넓은 범위를 대상으로 시스템을 확장하기 위해서는 가정하는 구조 자체를 크게 가정해야 한다. 면을 대상으로 하는 구조에서 가정하는 구조의 확장은 데이터 량의 방대한 증가를 초래하며, light ray 획득의 난이도를 급격히 증가시킨다. 3D LF의 경우 구조가 커지더라도 슬라이더 장비를 통해 움직이는 거리와 시간이 증가하기 때문에 상대적으로 light ray 획득 난이도는 크게 증가하지 않으며, 증가하는 데이터 량 또한 상대적으로 적다. 하지만 구조가 커짐에 따라 임의의 viewpoint와 3D LF에서 가정하는 선 간의 거리가 멀어지면서 3D LF의 구조적인 한계로 인해 vertical parallax를 표현하지 못함에 따른 에러가 커지는 문제가 있으며, 결과적으로 뷰에 왜곡이 포함되는 문제가 있다.

1.2 연구 내용

본 논문은 LF 기반의 시스템을 바탕으로 넓은 범위를 자유롭게 움직일 수 있는 자유 시점 변환이 가능한 가상현실 시스템의 개발을 목표로 한다. 특히 3D LF를 기반으로 하여, 표현할 수 있는 viewpoint의 범위를 자유롭게 넓히더라도 light ray의 획득을 간편하게 하고, 데이터 량의 부담을 줄인다.

기존의 LF 구조에서 범위를 넓히기 위해 구조 자체의 크기를 키우는 것과 달리, 본 논문에서 제안하는 구조는 여러 개의 3D LF를 일정한 거리를 두고 배치하는 구조이며, 3D LF Stack 으로 정의한다. 이 구조에서 임의의 viewpoint에 따라 viewpoint의 앞에 위치한 3D LF를 사용해서 뷰를 구성하게 된다. 이는 viewpoint를 자유롭게 움직이더라도 viewpoint와 3D LF를 구성하는 직 선 사이의 거리를 일정 범위로 제한하게 되며, 그 결과 vertical parallax를 표현하지 못함에 따른 에러를 일정 수준 이내로 제한할 수 있다. 또한 roll, yaw, pitch의 회전 변환을 모두 지원하기 위해 360도 뷰 구성이 필요하며, 이를 위해 3D LF Stack을 네 방향으로 배치한다. 격자로 배치된 3D LF에서 사용자의 임의의 viewpoint가 정해지면, 해당 viewpoint를 둘러싸는 네 개의 3D LF를 사용해서 뷰를 구성함으로써 최종적으로 360도 뷰를 구성한다.

위의 제안 구조를 위해서는 추가적인 두 가지 해결 과제가 있는데, 하나는 네 개의 3D LF를 이용해서 하나의 뷰를 만드는 방법이 필요하다. 본 논문의 제안 구조는 임의의 viewpoint에서의 360도 뷰를 구성하기 위해 하나의 단일 LF 구조를 가정하지 않고 해당 viewpoint를 둘러싸는 네 개의 3D LF를 사용한다. 독립적으로 배치된 3D LF를 서로 연결하여 하나의 뷰를 구성하기 위한 방안이 필요하다.

또 다른 하나는 3D LF Stack에서 뷰를 구성할 때 사용하는 3D LF가 바뀜에 따른 뷰의 부자연스러운 변화를 해결하는 문제이다. 3D LF Stack은 임의의 viewpoint의 위치에 따라 뷰를 구성하기 위해 사용하는 3D LF가 바뀌게 되는데, 이 때 뷰가 급격히 바뀌는 문제가 발생한다.

본 논문에서는 이를 해결하기 위해 하나의 3D LF를 사용해서 뷰를 구성하지 않고, 앞, 뒤로 배치된 두 3D LF를 사용하는 방법을 제안한다. 이는 3D LF Stack 상에서 3D LF를 넘나 들더라도 하나의 3D LF를 공유함으로써 뷰가 급격히 변하는 문제를 완화할 수 있으며, vertical parallax를 표현하지 못함에 따라 실제 뷰와 차이가 나는 에러를 줄일 수 있다.

1.3 논문 구성

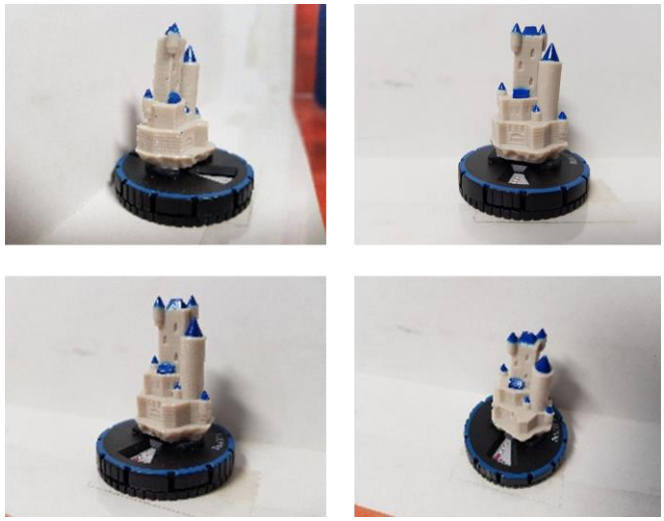
본 논문은 다음의 구성으로 이루어져 있다. 2장에서는 본 논문과 관련된 이전 연구에 대해서 설명한다. 3장에서는 본 논문에서 제시하는 전체 구조를 설명하고, 전반적인 동작 과정을 설명한다. 4장에서는 네 개의 3D LF를 사용해서 하나의 뷰를 구성하기 위한 3D LF 연결 방안을 제시하며, 두 가지 시스템 구성 환경에서 이를 적용한다. 5장에서는 3D LF Stack 구조에서 뷰를 구성하기 위한 방법을 제시한다. 6장에서는 4, 5장에서 설명한 두 가지 제시 방법을 모두 포함한 전체 시스템의 동작 결과를 정리하며, 마지막으로 7장에서는 본 논문의 결론을 다룬다.

제 2 장 관련 연구

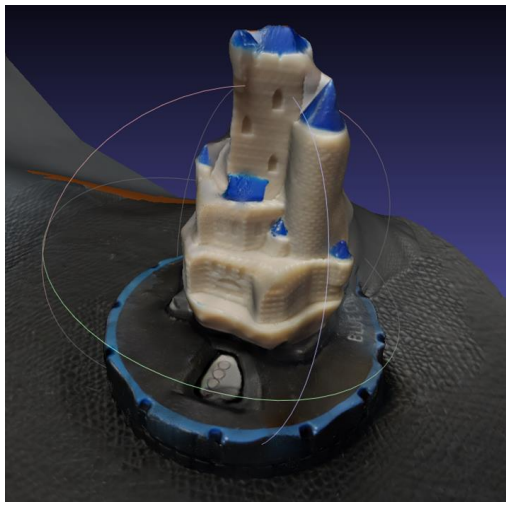
2.1 3D 스캐닝 및 렌더링

높은 DoF를 지원하는 자유 시점 변환이 가능한 가상현실 시스템을 구성하기 위한 기존 연구 중 가장 대표적으로 3D 스캐닝 및 렌더링 기법이 있다. 3D 스캐닝 및 렌더링 기법은 표현하고자 하는 공간에 있는 모든 물체와 배경 등의 3D 정보를 취득하고, 가상 공간에 이를 재구성한다. 이는 마치 컴퓨터 그래픽 기반의 가상현실 기술과 마찬가지로 가상 공간에 대한 모든 3차원 정보를 사용해서 자유로운 시점 변환을 가능하게 한다.

3D 모델링 [5-8]은 다양한 시점에서 촬영된 다수의 이미지를 바탕으로 대상의 point cloud를 구성하고, mesh를 구성하는 과정을 거친다. 그림 2.1는 물체를 대상으로 3D 모델링을 통해 해당 물체를 재구성하는 과정에서 촬영한 이미지와 결과 이미지를 보여준다. 다시점에서 촬영된 이미지를 통해 3차원 정보를 추정하고, 재구성한다. 물체를 대상으로 하는 3D 모델링은 비교적 좋은 결과를 보여준다. 공간을 대상으로 하는 3D 모델링에 대한 연구 또한 지속적으로 진행되고 있다. 실제 공간을 가상의 공간에 모델링하고 해당 공간을 자유롭게 이동하거나, 변형하는 등의 다양한 적용이 가능하다. 대표적으로 집의 다양한 공간을 스캐닝하고 새로운 가구를 배치해 보는 적용이 대표적이다. 하지만 3D 모델링 기법은 비교적 작은 물체를 대상으로 할 경우 좋은 결과를 보이지만, 모델링 대상이 커지고, 계산이 필요한 대상이 급격히 증가함에 따라 계산 복잡도가 크게 증가하는 반면, 정확도는 현저히 떨어진다. 특히, indoor를 대상으로 하는 경우 특징점을 검출하기 어려운 벽, 천장 등은 특징점 기반의 매칭이 어렵고, 3D 정보 추정이 어려워진다. 넓은 공간의 경우 촬영된 이미지 간의 상대적 위치 정보를 추가적으로 고려함으로써 정확도를 높인다 [12-14].



(a)



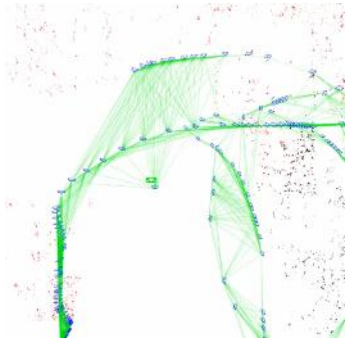
(b)

그림 2.1. (a) 다시점에서 촬영한 이미지, (b) 3D 모델링을 통해 재구성한 결과

대표적으로 simultaneous localization and mapping (SLAM)은 임의의 공간을 이동하면서 촬영된 이미지를 바탕으로 현재 카메라의 위치를 계산하는 localization 과 공간의 맵을 구성하는 mapping 과정을 동시에 수행한다. 공간을 대상으로 하는 3D 모델링에서는 공간을 이동하면서 SLAM을 통해 획득한 정보를 활용하여 대상 공간을 모델링

한다. 주기적으로 구성한 맵을 최적화하고 카메라가 이전의 경로를 통과하여 루프가 구성될 때 경로를 최적화하는 등의 과정을 통해서 정확도를 높인다 [15]. 그림 2.2 (a), 그림 2.2 (b)는 SLAM의 동작 과정을 보여준다. 그림 2.2 (a)는 촬영된 이미지에서 특징점을 검출하는 과정을 보여주며, 그림 2.2 (b) 은 해당 특징점을 사용해서 맵을 구성함과 동시에 현재 위치를 추정한 결과를 보인다.

(a)



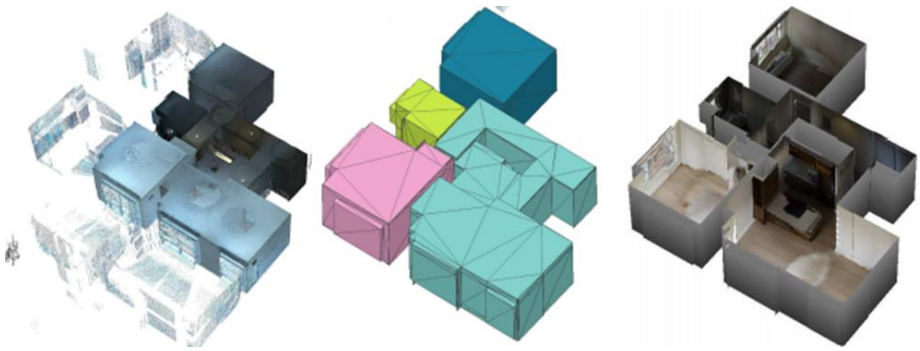
(b)

그림 2.2. SLAM의 동작 과정 [15], (a) 촬영된 이미지에서 특징점 검출, (b) 맵 구성 및 현재 위치 추정

RGB 정보 이외의 정보를 추가적으로 사용함으로써 3D 모델링 성능을 개선하기도 한다. 대표적으로 depth 정보가 사용된다. Depth는 3D 이미지 처리에서 두루 사용되며 픽셀이 해당하는 물체의 거리 정보를 담고 있다. 3D 모델링에서 depth는 특징점의 3D 위치 정보를 계산하는 추가적인 정보로 사용된다. Kinect는 대표적인 RGB-D 카메라로 RGB와 더불어 depth 정보를 동시에 취득한다 [16, 17].

자율주행에서 주로 사용되는 LiDAR 장비는 Kinect에 비해 더욱 정교한 거리 정보를 취득하며 3D 렌더링 분야에서 사용되기도 한다 [19]. 구조광(Structured light)은 사전에 설정된 패턴을 공간에 투영하고, 해당 패턴이 포함된 공간 이미지를 획득함으로써 3D 정보를 취득한다 [20]. 구조광은 텍스쳐를 포함하지는 않는 벽면에서도 임의의 패턴을 만들어 줌으로써 특징점을 구성할 수 있도록 한다.

Indoor 환경을 대상으로 3D 모델링을 적용하는 경우, 건물 내부의 벽, 천장, 바닥 등이 평면이라는 특성을 사용하기도 한다 [21-24]. 위와 같은 전제를 바탕으로 다양한 outlier와 에러를 피할 수 있다.



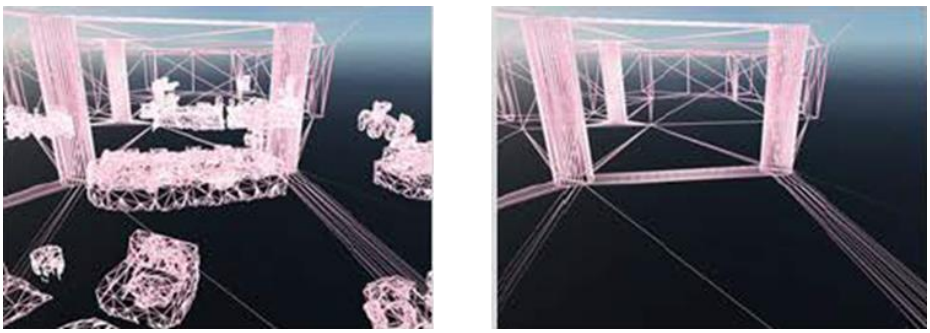
(a) (b) (c)

그림 2.3. Indoor 3D 모델링 [24], (a) point-cloud, (b) mesh 구성, (c) 3D 모델링 결과

그림 2.3는 indoor 환경을 고려한 3D 모델링의 예를 보여준다. 그림 2.3 (a)는 대상 구조에 대해서 3D 정보를 취득하고 최초 구성한 point-cloud를 보여준다. 그림 2.3 (b)는 구조를 방 단위로 나누고, 평면을 고려하여 mesh를 구성한 결과를 보여주며, 마지막으로 그림 2.3 (c)는 texture를 입힘으로써 최종적으로 구성한 3D 모델링 결과를 보여준다. 평면이라는 가정을 포함함으로써 반듯한 3D 모델링 결과를 만들어낸다.

하지만 위 방법에서는 해당 공간에 존재하는 많은 물체에 대한 고려가 필요하다. 두 가지 방법이 가능한데, 하나는 물체가 평면에

붙어있다고 가정하는 것이고, 다른 하나는 물체를 지워주는 과정을 포함하는 것이다 [25]. 그림 2.4은 공간에서 물체를 제거해줌으로써 오롯이 평면만을 가정하도록 mesh를 구성한 결과를 보여준다. 최초 구성한 point cloud에는 물체를 포함하게 되고 그대로 mesh를 구성할 경우 그림 2.4 (a)와 같이 물체가 포함된 결과가 만들어지며, 해당 결과에서 물체를 제거해줌으로써 그림 2.4 (b)에서 보는 바와 같이 평면만을 고려한 렌더링을 구성할 수 있게 된다.



(a) (b)

그림 2.4. Indoor 환경을 고려한 3D 모델링을 위해 물체를 제거, (a) 물체 제거 전, (b) 물체 제거 후

3D 모델링은 과정이 복잡하고 에러의 가능성이 높다는 점에서 실제 적용하는데 어려움이 있으며, 다양한 연구를 통해서 이를 개선하고자 한다. 하지만 3D 모델링은 대상을 정확하게 구축할 수 있다면 자유로운 viewpoint 이동에 따른 뷰 전환과 더불어, 빛의 소스를 자유롭게 이동하거나 조명의 색상을 바꿔줌으로써 대상 공간을 자유롭게 바꾸는 것 또한 가능하다. 나아가 새로운 3D 오브젝트를 배치할 수 있다는 점에서도 다양한 활용이 가능한 장점이 있다.

2.2 라이트 필드 (Light Field)

자유시점 변환을 지원하는 또 다른 접근 방법으로 LF를 이용하는 접근 방법이 있다. LF는 자유 공간을 통과하는 다양한 light ray의 집합을 의미한다. LF는 자유 공간을 통과하는 light ray를 정의하고, light ray를 조합함으로써 다양한 3D 이미지 처리 알고리즘에서 새로운 접근 방법을 제시한다 [26-31].

2.2.1 Light Field Representation

LF가 처음 제안된 plenoptic function에서는 light ray를 7개의 변수를 사용해서 정의한다 [9-11]. Light ray가 통과하는 자유공간의 3차원 상에서의 점 (x, y, z), 해당 점을 통과하는 수직, 수평 각도(θ, φ), 파동(λ), 그리고 시간(t)으로 표현된다. 이후 light ray는 5개 변수를 사용하는 representation으로 간소화하게 되는데, 이 때 시간을 일정 시점으로 고정시키고, 이미지 상의 RGB값으로 light ray의 색을 대체함으로써 파동을 변수에서 제외시킨다. 그리고 이는 다시 4개 변수로 간소화하게 된다. Light ray의 constant radiance의 특징을 기반으로 한다. Constant radiance of ray는 장애물이 없는 환경에서 하나의 light ray는 해당 light ray가 통과하는 다양한 위치에서 획득될 수 있음을 의미한다. 그림 2.5은 주어진 object에서 light ray가 나오는 그림을 나타낸다. 그림에서 검은색 점선 화살표가 light ray에 해당된다.

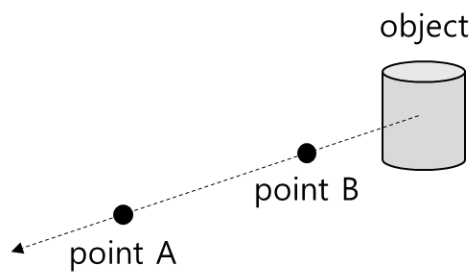


그림 2.5. Light ray의 constant radiance 특징

해당 light ray는 점 A와 점 B를 통과하게 된다. 점 A와 점 B는 3차원 상의 서로 다른 점에 해당된다. 따라서 5개 변수를 사용한 representation 방법에서 다르게 표현될 수 있지만 사실상 동일한 light ray에 해당된다. 이를 제거하기 위해 4개 변수를 사용하는 LF representation 방법에서는 평면을 가정한다.

4개 변수를 이용한 LF representation은 두 가지 방법이 있다. 하나는 하나의 평면을 가정하고 해당 평면을 통과하는 2차원 평면 상의 점 (x, y)와 수직, 수평 각도로 light ray를 표현하는 방법이다. 그림 2.6 (a) 는 하나의 평면을 가정하여 4개 변수를 통해 light ray를 표현하는 방법을 보여준다.

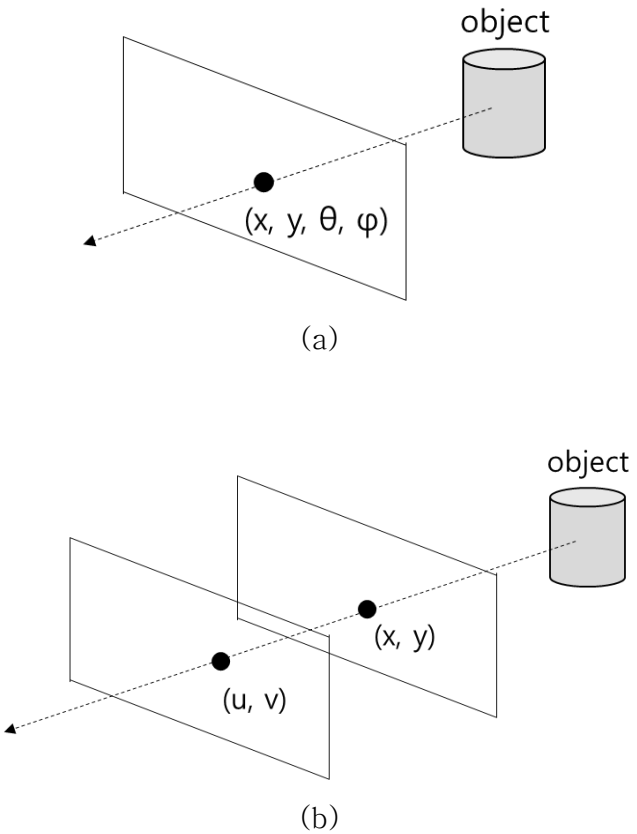


그림 2.6. 두 가지 4D LF representation 비교 (a) 하나의 평면을 가정, (b) 두 개 평면을 가정

그림에서 물체로부터 나오는 light ray는 평면을 통과한다. 평면을 통과하는 점과 각도를 통해 light ray를 표현하게 된다. 또 다른 방법은 두 개의 평면을 가정하는 방법이다. Light ray가 통과하는 두 평면 상의 두 점으로 light ray를 표현하게 된다. 그림 2.6 (b)는 두 개 평면을 이용한 표현 방법을 보여준다. 두 표현 방법은 서로 전환이 가능하다. 위와 같이 네 개 변수를 사용한 LF representation을 4D LF라고 한다.

2.2.2 Light Ray Acquisition

LF 접근 방법의 challenge 중 하나는 light ray를 획득하는 문제이다. 앞서 설명한 바와 같이 2D 평면을 통과하는 모든 light ray를 획득해야 한다. 실제 구현에서 light ray는 카메라로 촬영된 이미지 상의 한 점으로 대체된다. 그림 2.7는 이미지의 픽셀과 light ray의 관계를 보여준다. 그림에서 보는 바와 같이 가정한 평면 상에 카메라가 배치되어 이미지가 촬영된다. 그림의 경우 카메라는 평면 상에 (x_0, y_0)의 점에 배치되어 있다.

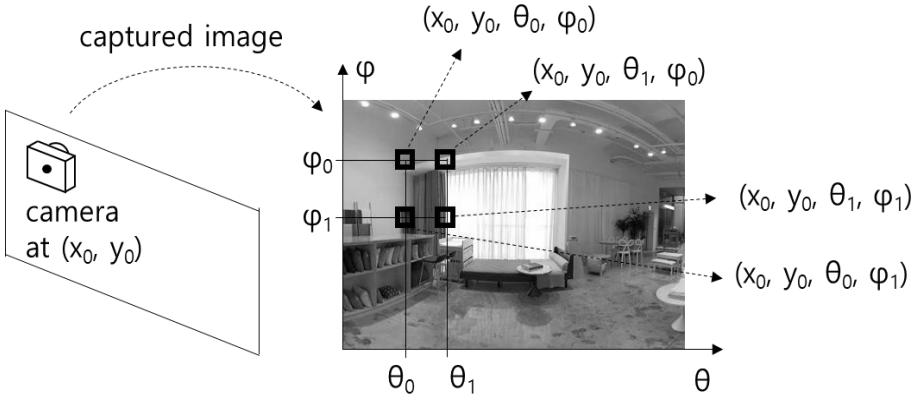


그림 2.7. 이미지 픽셀과 light ray와의 관계

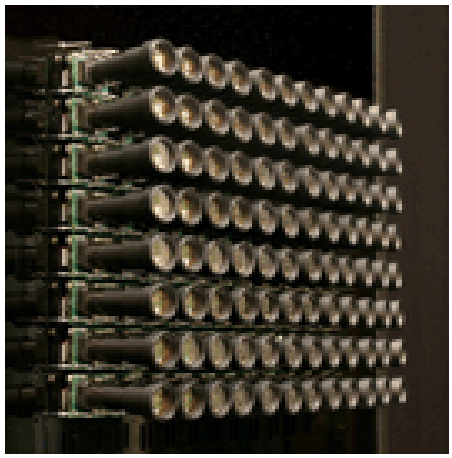
그리고 해당 카메라에서 촬영된 이미지에서 검은 색 네모로 표시된 네 개의 픽셀은 각각 이미지 상에서 (θ_0, φ_0), (θ_1, φ_0), (θ_0, φ_1), (θ_1, φ_1)에 위치한다. 결과적으로 네 개의 픽셀은 (x_0, y_0, θ_0, φ_0), (x_0, y_0, θ_1, φ_0), (x_0, y_0, θ_0, φ_1), (x_0, y_0, θ_1, φ_1)로 표현되는 light ray에 해당되게 된다.

위와 같은 LF를 구성하기 위해서는 가정한 평면을 따라 이미지를 촬영하는 방안이 필요하다. 4D LF를 구성하기 위한 세 가지 방법이 주로 사용된다. 첫 번째 방법은 2D 평면을 따라 다중 카메라를 구성하는 방법이다. 그림 2.8 (a)는 각각 스탠포드 연구팀에서 개발한 4D LF를 구성하기 위한 다중 카메라 구조이다.

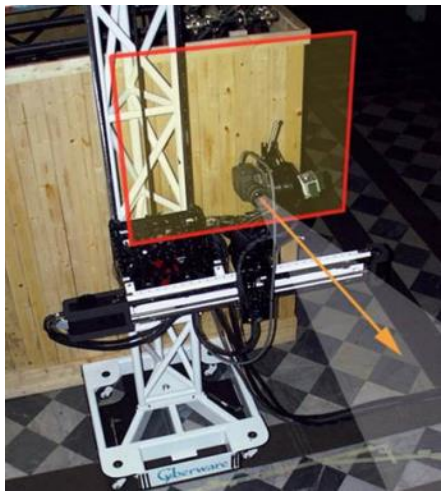
다중 카메라를 사용하는 방법은 동시에 촬영할 수 있기 때문에 움직이는 영상을 촬영할 수 있다는 장점이 있다. 하지만 다중 카메라를 사용할 경우 카메라 사이의 calibration 및 rectification 과 같은 전처리 과정이 필요하다. 또한 카메라의 특성에 따라 컬러의 차이가 발생할 수도 있다는 점에서 color correction이 포함되기도 한다. 추가적으로 카메라의 물리적인 크기로 인해서 매우 조밀하게 배치할 수 없는 문제도 있어서 보다 dense한 light ray를 사용하기 위해서는 이미지 사이의 가상의 이미지를 구성하는 view synthesis 등이 필요하기도 하다.

두 번째 방법은 한 대의 카메라를 움직이면서 촬영하는 방법이다. 그림 2.8 (b)는 gantry 장비를 이용해서 4D LF를 구성하는 카메라 시스템을 보여준다. 그림에서 gantry 는 빨간색 네모로 표시된 2D 평면을 따라 움직이며, gantry에 mount된 카메라를 통해서 2D 평면을 따라 이미지를 획득하게 된다.

한 대의 카메라를 이동하면서 촬영하기 때문에 비디오 형태의 데이터는 획득할 수 없다. 하지만 다중 카메라 구조에서 필요한 카메라 사이의 calibration 과 같은 전처리가 필요하지 않다는 점에서 매우 간편하다. 또한 매우 조밀하게 이미지를 촬영할 수 있다는 점에서 매우 유리하다.



(a)



(b)

(c)

그림 2.8. 4D LF 구성 시스템, (a) 다중 카메라 [32, 33], (b) Gantry 장비 [34], (c) lenslet LF 카메라 [35]

마지막으로 lenslet LF 카메라를 사용하는 방법이 있다. 카메라의 lens와 이미지 센서 사이에 micro lens를 배치하여 다양한 시점에서 촬영된 이미지를 획득한다. 하나의 이미지 센서를 사용해서 촬영하기 때문에 다중 카메라 구조에서 필요한 calibration과 같은 전처리를 포함하지 않는다. 또한 하나의 카메라로 이미지를 획득한다는 점에서 비디오 형태의 데이터도 확보할 수 있다. 앞선 두 획득 방법의 단점을 모두 보완할 수 있다. 하지만 micro lens 크기 내의 다양한 시점에서 획득된 데이터는 사실상 너무 좁은 baseline을 가질 수 밖에 없다. 따라서 넓은 시점을 움직이는 자유 시점 변환을 지원하는 가상 현실 시스템에서는 적합하지 않다. 그리고 정해져 있는 카메라 렌즈 안에 다양한 시점에서 촬영한 이미지를 모두 담기 위해서는 시점의 개수와 이미지 resolution 사이에 trade-off 문제가 있다. Lenslet LF 카메라에서 취득한 LF 정보를 바탕으로 high-resolution 데이터를 확보하는 연구도 활발히 진행되고 있다 [28, 29].

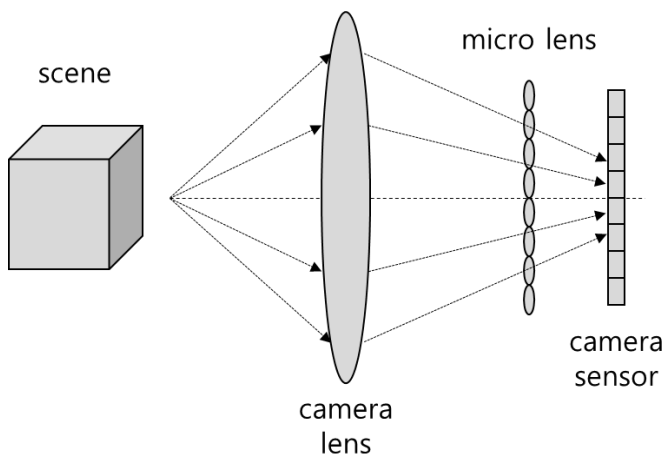


그림 2.9. Lenslet 카메라 내부 구조

2.2.3 View Generation in LF

앞선 light ray 획득 과정을 통해 LF를 구성하게 되면, 해당 LF에 포함된 light ray의 조합을 통해서 임의의 viewpoint에 대한 뷰를 구성할 수 있게 된다. 그림 2.10은 가정한 평면 앞에 임의의 viewpoint가 위치하는 경우, 해당 viewpoint에 대한 뷰를 구성하는 과정의 예를 보여준다. 이 때, 평면을 통과하는 모든 light ray는 이미 획득되어 있다고 가정한다. 임의의 viewpoint를 기준으로 평면을 통과하는 모든 light ray를 선택한다. 해당 light ray 중 일부를 그림 2.10에서 회색 점선 화살표로 표시한다. 각각의 light ray는 평면의 (x_0, y_0), (x_1, y_0), (x_2, y_0), (x_0, y_1), (x_1, y_1), (x_2, y_1)을 통과하는 light ray에 해당된다. 실제 뷰를 구성하는 과정에서는 그림 2.10에서 표시된 일부 light ray 뿐만 아니라 평면을 통과하는 모든 light ray를 사용해서 뷰를 구성하게 된다.

이처럼 LF를 기반으로 임의의 viewpoint에서의 뷰를 구성하는 과정은 LF에 포함되어 있는 light ray를 선택적으로 가져오는 매우 단순한 작업으로 가능하다. 만약 다른 viewpoint가 선택된다면, 평면과의 관계를 바탕으로 또 다른 light ray의 조합이 사용된다.

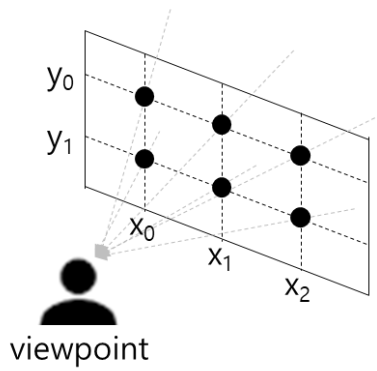


그림 2.10. 평면을 따라 구성된 LF를 이용한 뷰 구성

2.2.4 평면을 가정하는 LF 기반의 자유 시점 변환 시스템

LF는 보유하고 있는 light ray의 조합을 통해서 임의의 viewpoint에서의 뷰를 간편하게 구성할 수 있다는 점에서 시점을 자유롭게 변환할 수 있는 가상현실 시스템에 적합하다. Freeview TV (FTV)는 대표적인 평면을 기반으로 하는 자유 시점 변환 시스템을 제안한다 [33]. 평면을 따라 배치된 다중 카메라를 사용해서 LF를 구성한다. 주어진 평면의 범위 내에서 viewpoint는 x, y, z 축을 따라 이동할 수 있다.

그림 2.11은 가정한 평면 구조에서 viewpoint가 이동하는 예를 보여준다. 임의의 viewpoint에서의 뷰를 특정 FOV 크기로 제한한다고 할 때 임의의 viewpoint에서의 뷰는 그림 2.11의 점선 네모처럼 평면보다 작은 범위로 정해지고 해당 범위 내의 light ray만 사용해서 뷰를 구성하게 된다. Viewpoint가 x축을 따라 좌, 우로 이동함에 따라 점선 네모 박스는 좌, 우로 움직이게 되고, y축을 따라 위, 아래로 이동함에 따라 점선 네모 박스는 위, 아래로 이동하면서 해당 범위의 light ray로 뷰를 구성하게 된다.

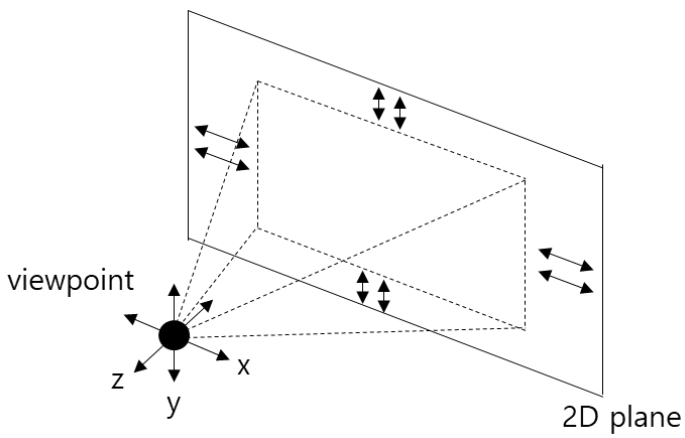


그림 2.11. 평면을 가정한 LF 구조에서 viewpoint의 이동 범위

z축을 따라 이동하는 경우 네모 박스의 크기가 커지거나 작아지게 된다. 가정하는 평면의 크기에 따라서 움직일 수 있는 범위로 제한된다. 평면을 가정한 FTV와 같은 구조에서는 회전 변환은 제한적으로만 지원할 수 있다. 평면이 커버할 수 있는 범위 내에서, roll, yaw, pitch의 회전 변환을 지원할 수 있지만, 한 방향으로 배치된 카메라 구조이기 때문에 넓은 회전 변환은 지원하지 못한다.

2.2.5 구면을 가정하는 LF 기반의 자유 시점 변환 시스템

평면이 아닌 구면을 가정한 LF가 구성되기도 한다 [36]. 그림 2.12는 구 면을 가정하는 LF 구조의 representation을 보여준다. 검은색 점선 화살표가 light ray를 나타낸다고 할 때, 해당 light ray가 통과하는 구면 상의 점과 수직, 수평 입사 각도의 네 개 변수를 사용해서 light ray를 표현하게 된다. 이는 2D 평면을 가정하는 representation과 동일하며, 구면을 가정하는 구조 역시 4D LF로 볼 수 있다.

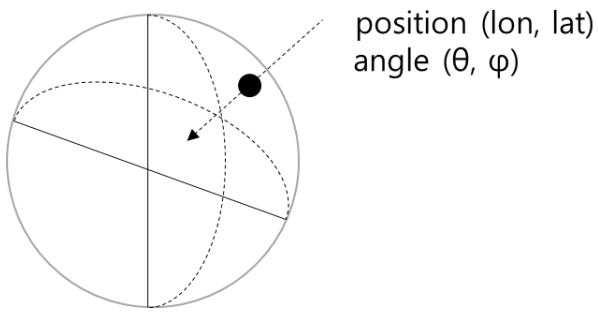


그림 2.12. 구면을 가정하는 LF 구조

하지만 2D 평면을 가정하는 구조와 달리 구면을 가정하는 구조는 360도 방향으로 입사하는 모든 light ray를 사용할 수 있다는 장점이 있다. 결과적으로 360도 방향에서 입사하는 light ray를 사용해서

360도 방향의 뷰를 구성할 수 있으며, 그 결과 roll, yaw, pitch의 세 가지 회전 변환을 완전히 표현해 낼 수 있다. 구 면의 구조에서 움직일 수 있는 viewpoint는 구 내부로 제한되지만, 구 내부에서 x, y, z 축을 따라 자유롭게 이동할 수 있기 때문에 결과적으로 구면을 가정한 LF 시스템은 6-DoF를 모두 지원한다.

구 면을 가정한 LF는 구면을 따라 이동하면서 light ray를 획득해야 한다. 그림 2.13은 구면을 가정하는 LF를 구성하기 위해 사용된 두 가지 장비이다 [37]. 구글에서 개발한 장비이며, 그림 2.13 (a)는 아치형으로 배치된 다중 카메라를 원으로 회전시키면서 이미지를 촬영하고, 그림 2.13 (b)는 반대방향으로 배치된 두 카메라를 구 모양으로 이동시키면서 이미지를 촬영한다.

위의 장비를 이용한 자유 시점 변환 시스템은 약 60cm의 지름을 가지는 구의 범위를 이동하면서 시점 변환을 경험할 수 있다. 60cm 지름의 구는 매우 넓은 공간을 자유롭게 이동하기에는 상당히 제한적인 범위이다.

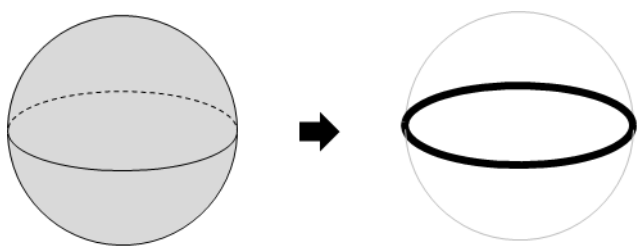
(a) (b)

그림 2.13. 구면을 가정하는 LF 구성 장비 [37], (a) 다중 카메라가 아치형으로 배치, (b) 두 카메라를 반대방향으로 배치

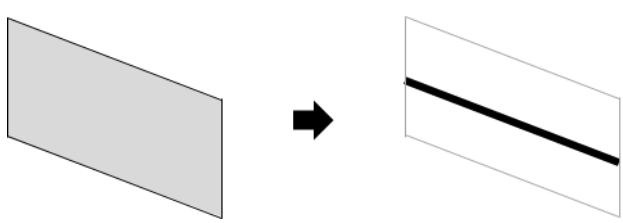
2.3 3D 라이트 필드 (3D LF)

3D LF는 앞선 4D LF와 달리 면 대신 선을 통과하는 light ray로 구성된다 [38-42]. 4D LF에서 light ray representation에 사용되는 네 개 변수 중 y가 한 점으로 고정된다. 결과적으로 직선 위에서의 점 x와 수직, 수평 입사각도의 세 개 변수로 하나의 light ray를 표현하여 3D LF로 정의한다. 그림 2.14 (a)에서 보는 바와 같이 구 면을 가정하는 대신 원을 가정하고, 그림 2.14 (b)에서 보는 바와 같이 평면을 가정하는 대신 직선을 통과하는 light ray를 사용하게 된다.

앞서서 평면, 구면의 경우 평면을 통과하는 모든 light ray를 획득하기 위한 다양한 방법을 제시한다. 반면, 3D LF의 경우 상대적으로 light ray 획득이 간편하다. 2D 평면을 따라 다중 카메라를 것이 아닌, 1D 배열을 따라 다중 카메라를 배치하여 light ray 데이터를 획득 한다.



(a)



(b)

그림 2.14. 4D LF와 3D LF에서 가정하는 구조의 예, (a) 구와 원 구조, (b) 평면과 직선 구조

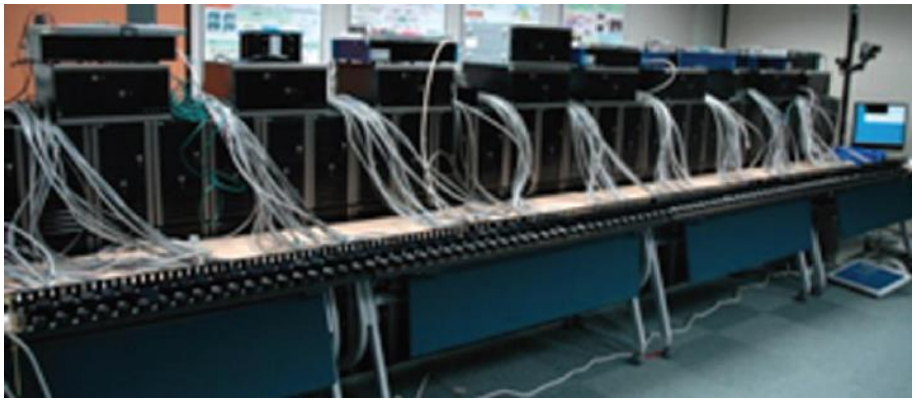
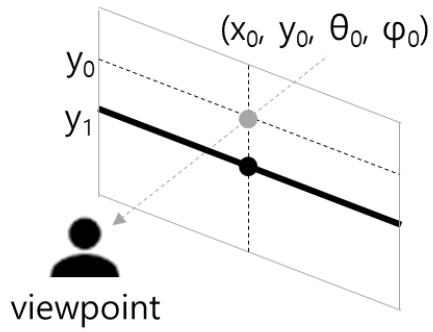


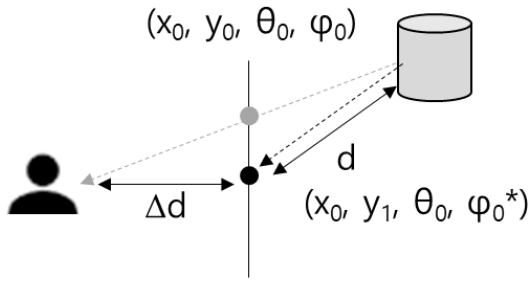
그림 2.15. 100대의 카메라를 1D 배열로 배치하여 3D LF 구성 [33]

그림 2.15는 100대의 카메라를 1D 배열로 배치하여 동시에 이미지를 취득하는 시스템 구성이며, 이를 통해 3D LF를 구성한다 [33]. 또는 2D 평면을 따라 이동하는 gantry를 이용한 것처럼 카메라 슬라이더나 dolly 장비가 이용될 수 있다. 앞선 gantry나 구글의 구면을 따라 이동하는 장비와 다르게 슬라이더나 dolly 장비는 대중적이고, 비용적 부담이 적다.

물론 3D LF를 사용하는 방법은 4D LF와 비교해서 사용할 수 있는 light ray가 매우 제한적인 문제가 있다. 3D LF는 구조적으로 수직 방향으로 오직 하나의 점에서만 light ray를 취득하기 때문에 vertical parallax를 표현하지 못하는 문제가 있다. 그림 2.16 (a)는 임의의 viewpoint에서의 뷰를 구성하는 예를 보여준다. 그리고 해당 뷰를 구성하기 위해 필요한 light ray 중 일부는 가정한 직선을 통과하지 않기 때문에 3D LF에 포함되지 않는다. 이처럼 3D LF의 구조적인 제한으로 인해 확보하지 못한 light ray는 3D LF 내의 다른 light ray로 대체하여 사용한다. 만약 실제로 필요한 light ray가 발산하는 물체의 거리를 안다면 추정이 가능하다. 그림 2.16 (b)는 그림 2.16 (a)를 옆에서 바라본 그림이다. 회색 점선 화살표는 그림 2.16 (a)의 회색 점선 화살표와 동일하다.



(a)



(b)

그림 2.16. (a) 3D LF에서 사용하지 못하는 light ray의 예, (b) 포함하지 않는 light ray 대체

해당 light ray는 회색 원통 물체에서 나오는데, 동일한 점에서 나오는 또 다른 light ray는 3D LF 구성을 위해 가정된 직선을 통과한다. 그림 2.16 (b)의 검은 점은 그림 2.16 (a)의 굵은 검은 직선을 나타내며, 원통 물체의 동일한 점으로부터 검은 점으로 향하는 light ray는 3D LF에서 포함하는 light ray 중 회색 점선 화살표의 light ray와 유사하다고 할 수 있다. 두 light ray의 x와 θ 는 각각 x_0, θ_0로 동일하며, y는 y_0 대신 직선 위치에 해당되는 y_1, φ는 φ_0 대신 φ_0*로 바뀐다.

φ_0 와 φ_0*의 관계는 (2.1)과 같다. 식 (2.1)에서 d는 3D LF를 구성하는 직선과 물체와의 거리를 의미한다. Δd는 3D LF를 구성하는

직선과 임의의 viewpoint 사이의 거리를 의미한다.

$$\varphi_0 = \tan\left(\frac{d \cdot \sin \varphi_0 *}{d \cdot \cos \varphi_0 * + \Delta d}\right) \tag{2.1}$$

식 (2.1)의 관계를 통해서 실제로 필요한 light ray 대신 3D LF에 속한 light ray 중 대체할 수 있는 light ray를 선택할 수 있다. 그리고 4D LF를 사용해서 구성한 뷰에 가까운 뷰를 구성할 수 있게 된다. 하지만 이 방법은 모든 light ray에 대한 depth 정보를 추정해야 한다는 부담이 있다. Depth 추정은 높은 계산 복잡도를 요구하며, 에러의 가능성 또한 높다. Depth 추정은 3D 모델링에 포함되는 주요 과정 중 하나로 depth를 이용한 light ray 대체는 LF가 light ray의 조합으로 뷰를 간편하게 구성할 수 있다는 장점을 줄여준다.

모든 light ray의 depth를 하나로 고정시키는 방법은 부정확한 depth를 사용하더라도 3D LF의 구조적 한계로 인해 표현할 수 없는 수직적 뷰 변환을 자연스럽게 반영할 수 있으며, 또한 depth 추정의 높은 부담을 줄여준다 [41, 42]. 하나의 고정된 depth를 이용한 접근 방법은 constant depth 로 정의한다. Constant depth는 모든 light ray에 대한 개별적인 depth 추정 없이 평균적인 depth를 사용함으로써 3D LF에서 표현할 수 없는 수직적 뷰 변환을 반영시킬 수 있으면서 동시에 depth 추정의 부담스러운 과정을 피할 수 있다.

또한 constant depth를 사용해서 뷰를 구성할 경우 동일한 픽셀 line에 위치하는 light ray는 모두 동일한 vertical angle을 가지는 light ray로 대체되게 된다. 그림 2.17은 constant depth를 이용해서 light ray를 대체하는 예를 보여준다. 그림에서 회색 점을 통과하는 회색 점선 화살표로 표시된 세 개의 light ray가 실제 뷰를 구성하기 위해 필요한 light ray에 해당되며, 각각 (x_0, y_0, θ_0, φ_0), (x_1, y_0, θ_1, φ_0), (x_2, y_0, θ_2, φ_0)로 표현된다. 동일한 depth를 고려해서 각 light ray가

방출되는 물체의 거리가 동일한 거리에 있다고 가정하게 되면, 식 (2.1)에 의해서 대체되는 light ray의 vertical angle은 φ_1로 모두 동일하다. 대체 light ray의 y 값은 3D LF를 구성하는 직선에 해당되는 y_1가 되고, 결과적으로 constant depth를 통해서 대체되는 light ray는 각각 (x_0, y_1, θ_0, φ_1), (x_1, y_1, θ_1, φ_1), (x_2, y_1, θ_2, φ_1)가 된다. 주어진 임의의 viewpoint에서의 뷰는 보정된 3D LF 내의 light ray의 조합을 통해 구성되게 된다.

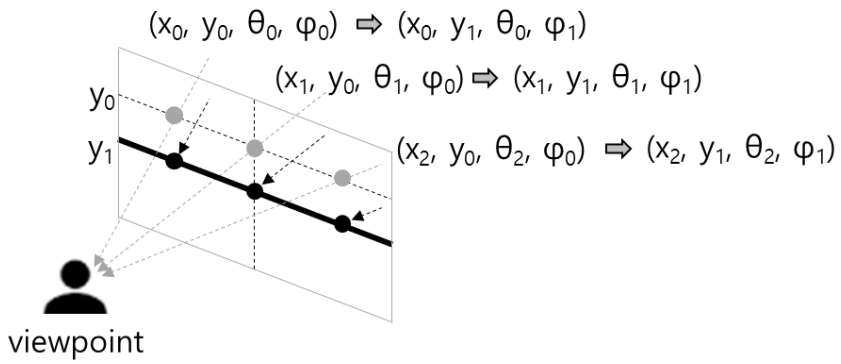


그림 2.17. Constant depth를 이용한 light ray 대체

3D LF를 이용한 환경에서 더 넓은 공간을 대상으로 시스템을 확장하기 위해서는 직선의 길이를 늘리거나 더 큰 원 형태의 3D LF를 가정할 수 있다. 앞선 4D LF 구조와 달리 카메라 슬라이더나 달리 장비가 이동하는 거리는 길어지지만 상대적으로 쉬운 light ray 취득이 가능하다는 점에서 용이하다. 하지만 크게 가정한 3D LF 구조에서 임의의 viewpoint와 3D LF의 거리가 멀어짐에 따라 constant depth로 가정함에 따른 에러가 점차적으로 증가하게 되며, 뷰가 크게 왜곡되는 문제가 발생하게 된다.

제 3 장 Stackable 3D LF 기반의 자유시점 변환 가상현실 시스템

본 논문은 간단한 데이터 획득 과정만으로 넓은 공간을 자유롭게 움직일 수 있는 자유시점 변환 가상현실 시스템의 개발을 목표로 한다. 기존에 제시된 자유시점 변환 시스템은 넓은 공간에 적용되기에 어려운 점들이 있었다.

먼저 3D 모델링의 경우 정확한 모델링을 위해 매우 높은 계산 복잡도를 요구하고 모델링 과정에서 에러를 포함할 가능성이 크다. 크기가 작은 물체에 대해서는 비교적 빠른 모델링이 가능하며, 실제와 유사한 결과를 만들 수 있지만, 모델링 대상을 공간으로 확장함에 따라 계산 복잡도가 급격히 증가하게 된다. 정확한 모델링만 가능하다면 다양한 시점 변환이 가능한 시스템을 완벽하게 구현할 수 있지만, 정확한 모델링을 위해서는 상당히 많은 시간과 노력이 필요하다. 필요에 따라 3D MAX 등과 같은 모델링 툴을 이용한 후처리 과정이 포함되기도 한다.

LF 기반의 시스템은 3차원 정보를 계산할 필요가 없다는 점에서 비교적 적은 복잡도로 뷰를 구성할 수 있다. 그리고 이는 LF 기반의 접근이 자유시점 변환을 지원하는 시스템에 적합한 이유이기도 하다. 별도의 geometry 계산 없이 light ray의 조합만으로 임의의 viewpoint에서의 뷰를 쉽게 생성할 수 있다. 하지만 기존의 LF 기반의 시스템은 넓은 공간으로 대상을 확장하기에는 다음과 같은 문제점을 포함한다. 먼저, 4D LF 기반의 시스템의 경우 넓은 공간을 대상으로 하기 위해서는 가정하는 구조 자체를 키워주어야 하는데, 큰 구조를 가정할 경우, 해당 구조를 통과하는 light ray의 획득이 어렵다. 앞서 살펴본 2D 평면, 구면을 고려해볼 때, 해당 구조를 키우면 키울수록 다중 카메라를 사용하는 방법과 gantry를 이용한 방법 모두 적용이 쉽지 않다. 반면, 3D LF 기반의 시스템은 넓은 공간을 대상으로 가정하고 구조를 크게 가정하더라도 선을 따라 light ray를 획득하는

과정은 매우 간편하고, 현실적이다. 길게 가정한 직선을 따라 슬라이더, dolly 등의 장비를 이동하면 된다. 하지만 3D LF 기반의 시스템은 vertical parallax를 표현하지 못하는 3D LF의 구조적 한계로 인해 발생하는 에러가 viewpoint와 3D LF 사이의 거리가 점차적으로 멀어짐에 따라 점점 커지게 되고, 뷰 자체가 왜곡되는 문제가 발생한다. Constant depth를 이용한 light ray 대체는 결과적으로 잘못된 depth를 사용한다. 따라서 좁은 viewpoint 범위의 이동에서는 비교적 에러가 눈에 띄지 않으며, 동시에 사용자의 뷰 변환은 자연스럽게 반영될 수 있다. 하지만 viewpoint의 이동 범위가 점점 넓어짐에 따라 잘못된 depth를 사용함에 따른 에러는 점차적으로 쌓이게 된다. 에러가 커지고 왜곡이 점점 쌓이게 되면 구성된 뷰는 실제 뷰와 많은 차이를 포함하게 된다.

본 논문에서 제안하는 시스템은 무엇보다 넓은 공간을 대상으로 하는 것에 초점을 맞춘다. 실제 가장현실에서 사용자가 요구하는 것은 가상의 공간을 자유롭게 움직이는 것이며, 이를 충족시키고자 한다. 이번 장에서는 본 논문에서 제안하는 Stackable 3D LF 구조의 주요 특징을 설명하고, 제안된 시스템의 동작 과정을 간략히 소개한다.

3.1 Stackable 3D LF의 구조적 특징

3.1.1 다중 3D LF의 적층형 배치

기존의 3D LF와 달리 제안 구조에서는 다수의 3D LF를 앞, 뒤로 차례로 배치하는 구조를 가정한다. 본 논문에서 우리는 이 구조를 3D LF Stack으로 정의한다. 기존의 4D LF 그리고 3D LF는 하나의 면 또는 하나의 선을 가정하여 시스템을 구성하였다. 시점의 위치에 관계없이 항상 가정한 하나의 면 또는 선을 통과하는 light ray의 조합을 바탕으로 해당 시점의 뷰를 구성한다.

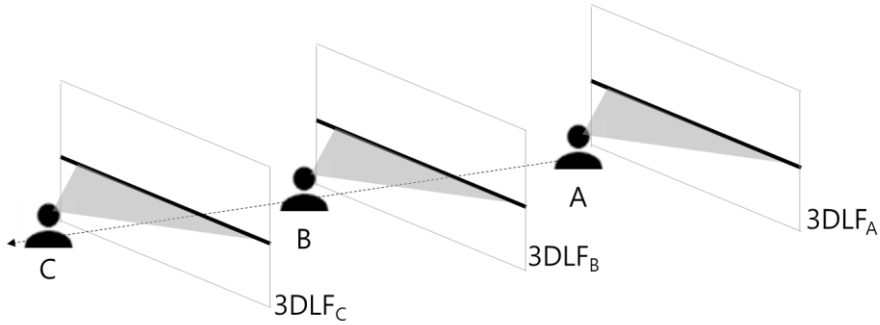


그림 3.1. 세 개의 3D LF가 배치된 구조

반면 제안 하는 3D LF Stack은 그림 3.1에서 보는 바와 같이 3D LF가 앞, 뒤로 배치된다. 그림 3.1의 예에서는 총 세 개의 3D LF가 앞, 뒤로 배치되어 있다. 그리고 임의의 viewpoint 위치에 따라서 해당되는 3D LF를 사용해서 해당 viewpoint에서의 뷰를 구성한다. 예를 들어 그림 3.1에서 viewpoint가 A에서 B, C로 이동한다고 할 때, viewpoint가 A에 위치한 경우 전방에 위치한 3DLF$_A$를 사용해서 뷰를 구성하고, viewpoint가 B에 위치하는 경우 3DLF$_B$를 사용, 그리고 마지막으로 viewpoint가 C에 위치하는 경우 3DLF$_C$를 통해서 뷰를 구성하게 된다.

3D LF Stack 구조를 통해 3D LF와 임의의 viewpoint 사이의 거리를 일정 수준 이내로 제한할 수 있다. 이는 vertical parallax를 표현하지 못하는 3D LF의 구조적 한계로 인해 발생하는 뷰의 에러를 일정 범위로 제한할 수 있음을 의미한다.

3.1.2 양 방향으로 통과하는 light ray를 이용한 3D LF 구성

기존의 4D LF와 3D LF는 한 방향으로 입사하는 light ray 만을 사용해서 구성되었다. 그림 3.2 (a), 그림 3.2 (b)는 기존의 3D LF 시스템의 예를 보여준다. 그림 3.2 (a)는 기존 3D LF를 구성하기 위해

사용하는 light ray를 보여준다. 기존 방법의 경우 바깥쪽에서 안쪽 방향으로 향하는 light ray만을 사용해서 3D LF를 구성하게 된다. 이처럼 구성된 3D LF는 그림 3.2 (b)에서 보는 바와 같이 안쪽에서 바깥쪽으로 향하는 뷰에 대해서만 구성이 가능하다.

본 논문에서 제안하는 시스템에서는 기존의 방법과 달리 안쪽에서 바깥쪽으로 향하는 light ray 또한 동시에 사용해서 3D LF를 구성하도록 한다. 그림 3.2 (c), 그림 3.2 (d)는 새롭게 제안하는 3D LF를 구성을 보여준다. 그림 3.2 (c)는 새롭게 제안하는 3D LF를 구성하기 위해 사용하는 light ray를 보여주는데, 그림 3.2 (a)와 달리 양 방향으로 통과하는 light ray를 모두 포함한다.

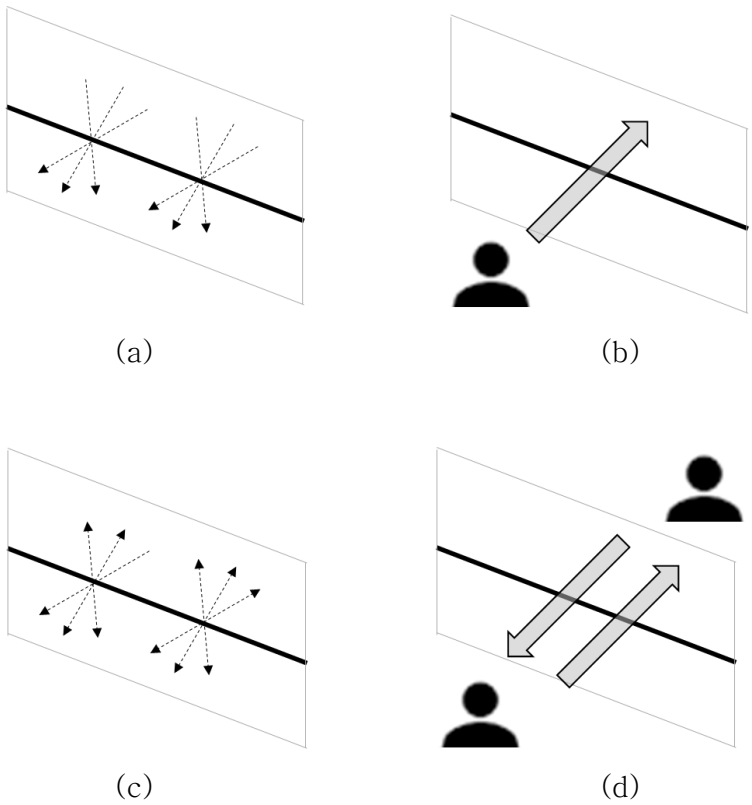


그림 3.2. 3D LF에서 포함하는 light ray에 따른 뷰 구성, (a) 단 방향 light ray, (b) 한 방향의 뷰만 구성 가능, (c) 양 방향 light ray, (d) 양 방향의 뷰 구성 가능

즉, 바깥쪽에서 안쪽으로 입사하는 light ray만 포함하는 것이 아니라 안에서 바깥쪽으로 향하는 light ray도 모두 포함한다. 그 결과 그림 3.2 (d)에서 보는 것처럼 사용자는 양 방향에 모두 위치할 수 있으며, 각각의 viewpoint에서 평면을 바라보는 방향의 뷰를 구성할 수 있게 된다.

실제 구현 과정에 양방향 light ray는 두 가지 방법으로 통해 획득될 수 있다. 하나는 두 번의 스캐닝 과정을 거치는 것이다. 첫 번째 스캐닝에서는 안쪽에서 바깥쪽으로 향하는 light ray를 획득하도록 카메라를 배치하고 light ray를 획득하고, 두 번째 스캐닝 과정에서는 바깥쪽에서 안쪽으로 향하는 light ray를 획득할 수 있도록 카메라를 배치하여 양 방향의 light ray를 획득한다. 또 다른 방법은 360도 카메라를 사용하는 것이다. 360도 카메라는 360도 방향에서 입사하는 light ray를 한번에 획득할 수 있다. 따라서 3D LF를 구성하는 선을 따라 360도 카메라를 움직이면서 한 번에 양 방향으로 입사하는 light ray를 획득한다.

3.1.3 두 개의 3D LF Stack을 수직 방향으로 배치

마지막 특징은 앞서 설명한 3D LF Stack을 수직 방향으로 두 개 배치한다. 그림 3.3는 두 개의 3D LF Stack을 수직 방향으로 배치한 구조를 보여준다.

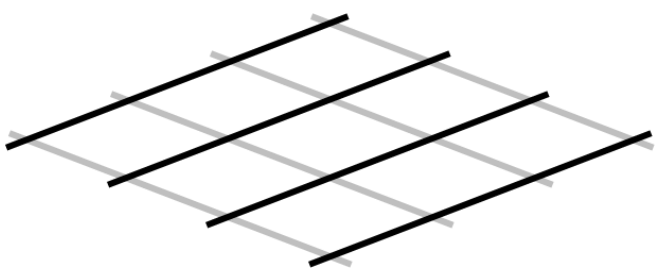


그림 3.3. 두 3D LF Stack의 수직 배치

검은색으로 표시된 하나의 3D LF Stack과 회색으로 배치된 다른 하나의 3D LF Stack을 확인할 수 있다. 결과적으로 이 구조는 격자 구조와 같으며, 임의의 viewpoint가 해당 구조 내에 있다고 할 때, viewpoint를 기준으로 네 개의 3D LF가 viewpoint를 둘러싸게 된다. 이를 통해 viewpoint로 향하는 360도 방향의 light ray를 사용해서 해당 viewpoint에서의 뷰를 구성하게 된다.

3.2 Stackable 3D LF 시스템에서의 자유 시점 변환

앞선 세 가지 특징을 바탕으로 구성된 제안된 시스템의 형태는 그림 3.3과 같은 격자 형태가 된다. 임의의 viewpoint는 격자가 구성된 공간을 자유롭게 이동할 수 있다. 임의의 viewpoint가 격자 형태의 구조 중 선 위에 위치하는 경우 별도의 뷰 구성 과정 없이 해당 위치에서 촬영된 이미지를 해당 viewpoint에서의 뷰로 사용하게 된다. 임의의 viewpoint가 격자 위가 아닌 공간에 배치된 경우 해당 viewpoint에서의 뷰를 구성하기 위한 과정이 필요하다. 제안된 구조에서 임의의 viewpoint가 정해지면 그림 3.4에서 보이는 것처럼 해당 viewpoint를 둘러싸는 네 개의 3D LF가 정해진다. 임의의 viewpoint와 각 방향의 3D LF와의 관계를 바탕으로 네 방향의 뷰를 구성할 수 있으며, 이를 연결함으로써 360도 방향의 뷰를 구성하게 된다.

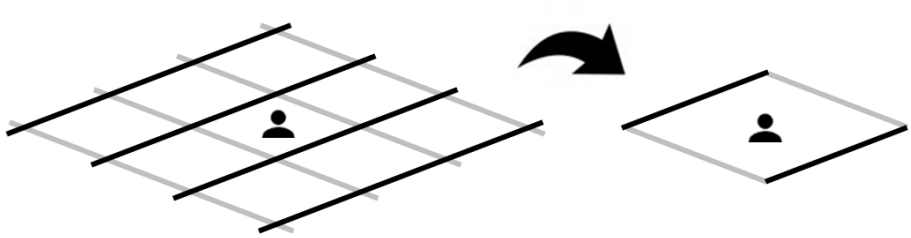


그림 3.4. 임의의 viewpoint를 둘러싸는 네 개의 3D LF

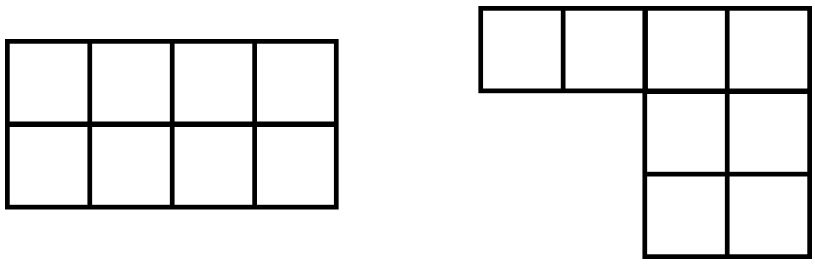
본 논문의 구조는 격자가 배치된 평면을 움직일 수 있다. 즉, x-z 평면을 이동하면서 동시에 360도 뷰를 이용해 roll, yaw, pitch의 세 가지 회전 변환을 지원함으로써 총 5-DoF를 지원한다. y축 방향으로의 시점의 수평 이동은 지원하지 않는다. y축 방향의 시점의 수평 이동을 지원하기 위해서는 앞서 정의한 격자구조를 y축 방향으로 쌓아주어야 한다. 즉 필요한 데이터 량, 데이터 획득 시간, 전처리 시간 등이 모두 배가 되고 큰 부담으로 작용한다. 하지만 넓은 공간을 대상으로 자유롭게 이동하는 시스템에서 이와 같은 y축으로의 시점 변환은 필요한 비용 대비 제공할 수 있는 뷰 변환의 가치가 작다고 판단하여 지원 대상에서 제외한다.

3.3 Light Field Unit (LFU)와 다중 LFU 구조

앞서 격자 구조에서 임의의 viewpoint를 둘러싸는 네 개의 3D LF로 구성된 구조를 살펴보았다. 이 구조는 360도 뷰를 구성하기 위한 기본 단위에 해당되며, 본 논문에서는 light field unit (LFU)로 정의한다.

앞선 설명에서는 top-bottom의 순서로 LFU를 정의하였다. 하지만 시스템을 구성하는 과정에서 LFU는 시스템을 구성하는 최소 기본 단위로써 다수의 LFU를 쌓아가는 방식에 따라 다양한 공간에 본 논문의 구조를 적용할 수 있다. 예를 들어 그림 3.4의 경우 LFU는 가로 3개, 세로 3개씩 총 9개의 LFU가 정사각형의 형태로 연결된 구조로 볼 수 있다. LFU는 반드시 정사각형의 형태로 구조가 연결되어야 하는 것은 아니다.

그림 3.5 (a)는 LFU를 직사각형의 형태로 연결한 구조이다. 이 경우 가로 4 개의 LFU, 세로 2 개의 LFU가 배치되며 총 8개의 LFU로 구성된다. 그림 3.5 (b) 는 ‘ㄱ’ 형태로 다수의 LFU를 연결한 구조이다.



(a) (b)

그림 3.5. 다중 LFU 구조의 예 (a) 직사각형 구조, (b) ‘ㄱ’자 구조

일반적인 건물의 복도는 그림 3.5 (b) 처럼 ‘ㄱ’ 형태의 모양을 하고 있다. 해당 공간 역시 LFU를 다수 연결함으로써 본 논문에서 제안하는 구조를 적용해볼 수 있다. LFU의 연결은 모양과 개수에 제한이 없으며, 다양한 공간에 적용할 수 있을 것으로 기대된다.

3.4 제안 시스템의 간략한 동작 과정 설명

본 논문의 제안 구조의 전체 동작 과정을 간략하게 소개한다. 먼저 제안 구조는 크게 두 과정으로 분류된다. 하나는 3D LF를 구성하기 위한 과정으로 light ray 획득 과정에 해당된다. 이후 viewpoint에 대한 뷰 구성 과정으로 획득한 3D LF를 사용해서 임의의 viewpoint에 대한 뷰를 구성한다.

Light ray 획득 과정에서는 그림 3.6 처럼 사전에 정의해둔 격자를 따라 카메라를 이동하면서 이미지를 획득함으로써 light ray를 획득하고 3D LF를 구성하게 된다. 그림의 경우 LFU를 3 × 3으로 배치한 구조를 나타낸다. 이 경우 총 4번의 카메라 이동 과정을 가로, 세로 방향으로 진행하게 된다.

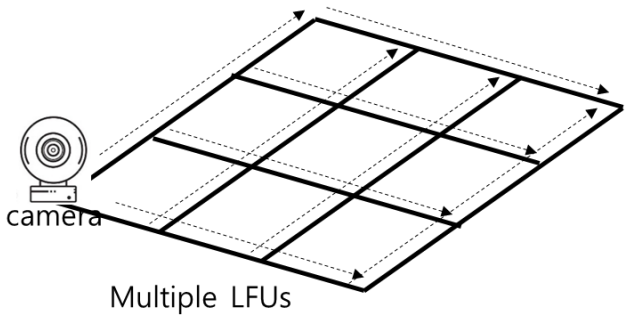


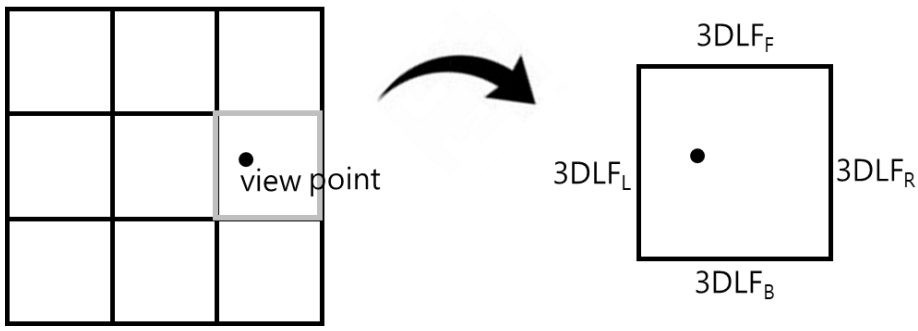
그림 3.6. 3 × 3 격자 구조를 따라 light ray 획득

앞서 언급한 바와 같이 본 논문의 제안 구조는 양 방향으로 통과하는 light ray를 모두 사용한다. 360도 카메라를 사용할 경우 360도 방향으로 통과하는 모든 light ray를 획득할 수 있기 때문에 경로를 따라 한 번 이동으로 모두 획득할 수 있다. 반면, 일반 카메라를 사용하는 경우 해당 경로를 두 차례 이동함으로써 light ray를 획득하고 격자 구조의 3D LF를 구성한다.

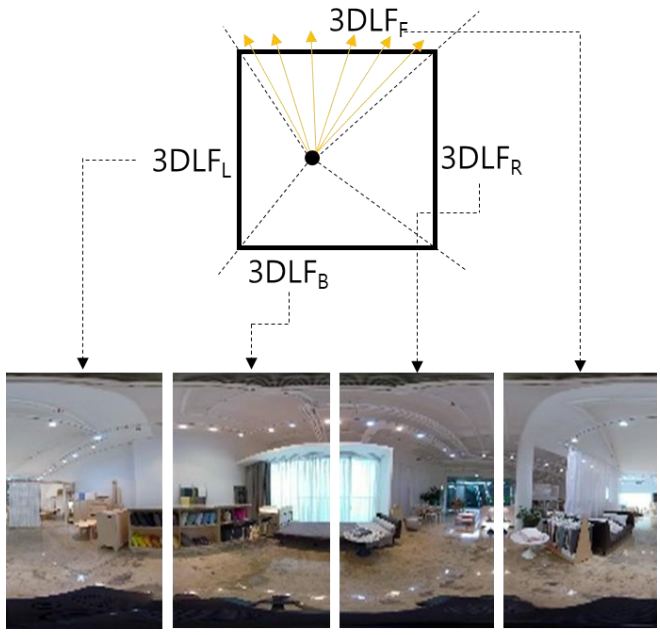
이론적으로는 획득한 이미지를 통해 곧바로 3D LF를 구성하는 것이 가능하다. 하지만 실제 구현 과정에서는 두 가지 전처리 과정이 포함되는데, 하나는 이미지가 균일한 거리를 가지도록 조정하는 작업이다. Dolly 장비의 경우, 장비가 출발하고 멈출 때 등속도로 움직이지 못하는 문제가 있다. 그 결과 획득한 이미지의 프레임이 균일한 거리만큼 떨어져서 추출되는 것이 아니라 처음과 끝에는 조밀하게 프레임이 추출되고, 중간 부분에서는 상대적으로 드물게 프레임이 추출되는 문제가 있다. 따라서 전체적으로 균일한 거리로 프레임이 추출되도록 조정하는 과정이 필요하다.

다른 하나는 격자에서 서로 다른 3D LF가 교차하는 점에서의 이미지를 조정하는 작업이다. 카메라가 정확한 평행으로 장착되지 못하거나 dolly 장비가 사전에 정의해둔 격자를 정확하게 지나가지 못하는 등의 다양한 원인으로 인해 이론적으로 정확하게 동일한

viewpoint에서 촬영된 이미지더라도 실제 촬영한 이미지에는 차이가 발생한다. 이를 보정하기 위해 교차하는 점에서의 두 이미지를 유사하게 만들어 주고, 주변 이미지들에 대해서 동일한 보정 값을 적용해주는 과정이 필요하다. 위 두 과정을 통해 보다 정교한 격자 구조의 3D LF가 만들어질 수 있으며, 외부적인 요인으로 인한 에러를 줄일 수 있다.



(a)



(b)

그림 3.7. (a)임의의 viewpoint에 따른 LFU 선택, (b) 각 방향의 3D LF를 이용한 뷰 구성 및 360도 뷰 구성

Light ray 획득하고, 전처리를 통해 보정된 LF를 구성한 뒤 실제로 임의의 viewpoint에 대한 뷰를 구성한다. 먼저 임의의 viewpoint 위치가 정해지면 해당 viewpoint를 둘러싸는 사각형 모양의 네 개의 3D LF가 선택된다. 즉, 정해진 임의의 viewpoint에 해당되는 LFU를 선택하는 과정이라 할 수 있다.

그림 3.7 (a)의 예에서 임의의 viewpoint는 3 × 3의 격자에서 맨 오른쪽 두 번째의 사각형에 위치한다. 그리고 해당 viewpoint를 둘러싸는 네 개의 3D LF는 해당 viewpoint의 뷰를 구성하기 위한 LFU가 된다. LFU에 포함된 3D LF는 각각 $3DLF_F$, $3DLF_R$, $3DLF_B$, $3DLF_L$이며, 아랫첨자는 front, right, back, left를 의미한다. 임의의 viewpoint와 $3DLF_F$, $3DLF_R$, $3DLF_B$, $3DLF_L$의 관계는 기존의 viewpoint와 단일 3D LF 사이의 관계와 동일하며, 기존과 동일한 방법으로 뷰를 구성하게 된다. 네 개의 3D LF를 통해 구성된 뷰를 연결해 줌으로써 360도 방향의 뷰를 구성하게 된다.

3.5 제안 시스템의 두 가지 해결 과제

본 논문의 제안 구조는 LFU를 다수 개 쌓음으로써 넓은 공간을 커버할 수 있으며 다양한 모양의 공간에 적용될 수 있다. 하지만 이를 위해서는 해결해야 하는 두 가지 과제가 남아있다.

하나는 LFU 구조 내의 임의의 viewpoint에서의 360 뷰를 구성함에 있어서 네 개의 3D LF를 사용해서 하나의 뷰를 구성하는 방법에 대한 고민이다. 기존의 4D LF와 3D LF 구조는 하나의 면, 선을 통과하는 light ray를 통해서 LF를 구성하고 해당 light ray의 조합을 바탕으로 뷰를 구성하였다. 하지만 본 논문에서 제안하는 구조는 네 개의 3D LF를 이용해서 하나의 뷰를 구성해야 한다. 본 논문에서는 서로 다른 두 3D LF가 공통의 light ray를 공유함을 사용해서 뷰를

연결하는 방안을 제시한다. 이 때 서로 다른 두 3D LF가 공유하는 공통의 light ray를 shared light로 정의한다. 이웃한 두 3D LF가 shared light를 공유하는 방식에 따라 두 가지 3D LF 연결 방법을 제안하고, 다양한 환경에서 적용 및 결과를 비교한다.

다른 하나는 3D LF Stack에서 뷰를 구성하는 문제이다. 3D LF Stack 구조를 통해 우리는 3D LF의 vertical parallax를 표현하지 못하는 문제로 인해 뷰의 왜곡을 일정 수준 이내로 제한하였다. 하지만 에러는 여전히 존재하며 특히 viewpoint가 이동하고 사용하는 대상 3D LF가 바뀌는 순간 뷰가 급격하기 바뀌는 문제가 발생한다. 이와 같은 현상이 발생하는 이유는 viewpoint가 3D LF에서 멀어지면서 constant depth를 사용함에 따른 에러가 점차적으로 쌓이고 실제 뷰와 점점 차이가 발생하는데, 대상이 되는 3D LF가 바뀌는 순간 다시 에러가 거의 없어 원본 이미지와 유사한 뷰가 만들어지면서 뷰가 급격히 바뀌게 된다.

본 논문에서는 앞서 언급 한 두 해결 과제에 대한 해결 방안을 4장, 5장에서 차례로 설명한다.

제 4 장 3D LF Connection

이전의 LF를 기반으로 하는 다양한 연구에서는 하나의 평면 또는 구면의 단일 4D LF, 또는 하나의 직선, 원 구조의 단일의 3D LF를 사용해서 뷰를 구성하였다. 반면 본 논문의 구조에서는 네 개의 3D LF를 사각형의 형태로 배치하여 기본 구조를 구성하며, 이 네 개의 3D LF를 사용해서 하나의 360도 이미지를 구성하게 된다. 이를 위해서는 각각의 독립적인 3D LF를 연결하여 하나의 뷰를 구성하는 연결 방안에 대한 고민이 필요하다. 이번 장에서는 서로 다른 두 3D LF를 연결하는 두 가지 방안을 제안하고 LFU 구조에서 이를 어떻게 활용하는지 확인한다. 나아가 다양한 시스템 환경에서 3D LF의 연결 방안을 적용하고 적절한 적용 방법을 제시한다.

4.1 Physical Connection

서로 다른 두 3D LF를 연결하기 위한 첫 번째 방법으로 물리적인 교차점을 이용하는 방법을 먼저 소개한다. 서로 다른 두 3D LF가 물리적으로 교차하는 점을 통과하는 light ray는 두 3D LF에 모두 속함을 바탕으로 두 3D LF를 연결해준다. 물리적인 교차점을 통과하는 light ray를 이용한다는 점에서 이하 physical connection으로 명칭을 정의한다.

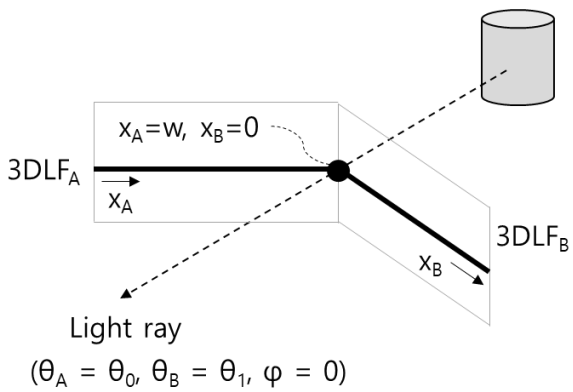


그림 4.1. Physical connection을 이용한 두 3D LF 연결의 예

그림 4.1은 physical connection을 이용해서 서로 다른 두 3D LF가 물리적으로 교차하는 예를 보여준다. 그림에서 두 개의 굵은 검은 선은 각각 $3DLF_A$와 $3DLF_B$를 구성하기 위한 직선 구조를 나타낸다. 그리고 두 직선은 검은 점으로 표시된 한 점에서 만나는 것을 확인할 수 있다. 회색 원통 모양의 물체가 있으며 해당 물체로부터 나오는 검은 점선 화살표로 표시된 light ray는 해당 교차점을 통과한다. 결과적으로 해당 light ray는 두 3D LF에 모두 속한다고 할 수 있다. 다만, 두 3D LF에서 사용하는 representation은 각각 (4.1)와 (4.2)으로 차이가 있다. 비록 representation은 다르게 표현되지만 두 light ray는 동일한 light ray임을 알 수 있다.

$$\left(x_A = w, \theta_A = \theta_0, \varphi_A = 0\right) \tag{4.1}$$

$$\left(x_B = 0, \theta_B = \theta_1, \varphi_B = 0\right) \tag{4.2}$$

이와 같이 이웃한 두 3D LF가 동일하게 획득하는 light ray를 shared light로 정의한다. 그림 4.1의 shared light 처럼 해당 교차점을 통과하는 모든 light ray는 모두 $3DLF_A$와 $3DLF_B$의 shared light라고 할 수 있다. (4.3)는 그림 4.1의 viewpoint 위치의 경우 모든 shared light를 나타낸다. 두 3D LF의 shared light는 $3DLF_A$와 $3DLF_B$가 보유한 light ray 중 공통으로 보유한 것을 의미하며 교집합으로 표현하였다.

$$\begin{aligned} 3DLF_A \cap 3DLF_B &= \left\{(x_A, \theta_A, \varphi_A) \middle| x_A = w, \theta_A = \theta_0\right\} \\ &= \left\{(x_B, \theta_B, \varphi_B) \middle| x_B = 0, \theta_B = \theta_1\right\} \end{aligned} \tag{4.3}$$

(4.3)는 $3DLF_A$에 속한 light ray 중 (x_A=w, $\theta_A = \theta_0$) 변수를 가지는 모든 φ_A를 가지는 light ray는 3DLFB에 속한 light ray 중 (x_B=0, $\theta_B = \theta_1$) 변수를 가지는 모든 φ_B를 가지는 모든 light ray와

동일한 light ray 임을 표현한다. 식 (4.3)의 변수는 임의의 viewpoint 위치에 따라 달라질 수 있다.

4.2 Physical Connection in LFU

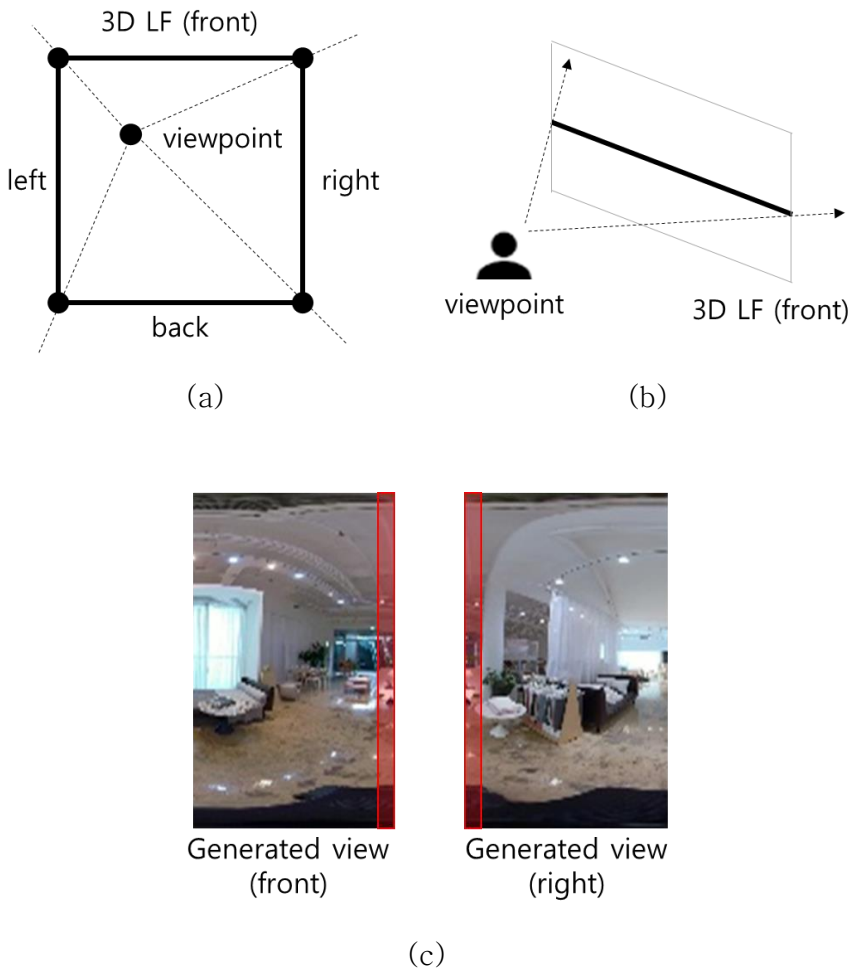
앞서 제안한 physical connection 방법을 LFU 구조에 적용해본다. 그림 4.2 (a)는 주어진 LFU 내의 임의의 viewpoint가 정해져 있는 상황을 보여준다. 각 방향으로 front, right, back, left 방향에 별도의 3D LF가 존재한다. Physical connection은 이웃한 3D LF를 물리적인 교차점에서 연결하는 방법이다. 그리고 LFU에서 볼 수 있듯이 각각의 3D LF는 양 옆으로 이웃한 두 개의 3D LF와 교차점을 가지고 있다. 즉, 하나의 3D LF는 두 개의 이웃한 3D LF와 두 개의 교차점을 공유하고 있다고 할 수 있다.

먼저 임의의 viewpoint가 정해지면 네 개의 교차점과의 관계를 정의하게 된다. 그림 4.2 (a)에서 검은색 점선은 임의의 viewpoint와 각각의 교차점을 연결하는 선이다. 해당 선으로 정의되는 범위만큼 각각의 3D LF에서 뷰를 구성하게 된다. 그림 4.2 (b)는 임의의 viewpoint와 front 방향의 3D LF와의 관계를 별도로 살펴본 그림이다. Front 방향의 3D LF의 양 끝 점을 통과하는 light ray는 그림 4.2 (a)에서 정의한 교차점을 통과하는 light ray이다. 그리고 viewpoint와 front 방향의 3D LF와의 관계를 통해 기존의 3D LF 기반의 뷰 구성 방법을 사용해서 해당 viewpoint에서의 뷰를 구성할 수 있다.

동일한 방법으로 네 방향의 3D LF에 대한 뷰를 구성하게 된다. 이 때 각각의 구성된 뷰에서 양 쪽 끝 pixel column은 이웃한 3D LF와의 shared light로 구성된다. 그림 4.2 (c)는 front 방향의 3D LF와 right 방향의 3D LF를 통해서 구성된 두 개의 뷰를 보여준다. Front 방향의 3D LF에서 구성된 뷰의 가장 오른쪽 pixel column과 right 방향의 3D LF에서 구성된 뷰의 가장 왼쪽 pixel column은 두 3D LF의 shared

light로 구성된 pixel column을 의미한다. 따라서 두 pixel column은 동일하며, 이 두 이미지를 연결해줌으로써 두 3D LF를 이용해 하나의 뷰를 구성하게 된다.

네 방향의 뷰는 동일한 방법을 통해 연결될 수 있으며, 그림 4.2 (a)의 임의의 viewpoint에 대한 360도 뷰를 구성할 수 있게 된다.



(a) (b)

(c)

그림 4.2. LFU 환경에서 physical connection을 이용한 3D LF 연결, (a) LFU 구조에서 임의의 viewpoint와 각 교차점의 관계, (b) 하나의 3D LF와 viewpoint와의 관계 (c) 이웃한 두 3D LF로 구성된 뷰의 shared light에 해당되는 pixel column의 관계

4.3 Non-physical Connection

서로 다른 두 3D LF를 연결하는 두 번째 방법은 light ray의 constant radiance of ray의 특성을 이용하는 방법이다. Constant radiance of ray의 특성은 앞서 살펴본 바 있는데, 장애물이 없는 자유 공간을 통과하는 light ray는 해당 light ray가 통과하는 경로의 모든 점에서 획득이 가능하다는 특성을 의미하였다. 두 번째 방법에서는 이런 light ray의 특성을 바탕으로 서로 다른 두 3D LF가 물리적으로 교차하지 않더라도 light ray를 공유할 수 있는 방안을 제시하고, 이를 이용한 3D LF 연결 방안을 제시한다. 두 번째 방법은 물리적으로 교차하지 않는 점을 이용해서 3D LF를 연결하기 때문에 이하 non-physical connection으로 정의한다.

그림 4.3은 두 개의 3D LF가 배치된 그림이다. 그림에서 두 개의 3D LF는 앞선 그림 4.1의 physical connection의 경우와 달리 물리적인 교차점을 두지 않는다. 하지만 동일한 회색 원통 물체에서 나온 light ray는 두 3D LF를 구성하는 두 직선 위를 통과하며, 각각은 검은 점으로 표시되어 있다.

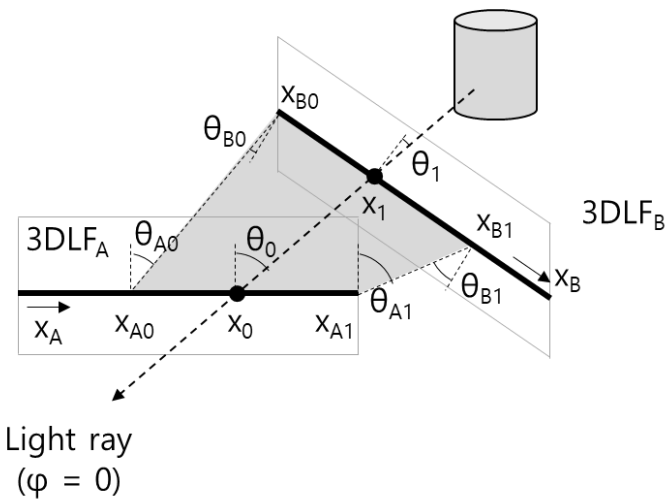


그림 4.3. 물리적으로 교차하지 않는 점을 이용한 3D LF 연결 방법

해당 light ray는 $3DLF_A$와 $3DLF_B$에서 (4.4), (4.5)의 representation을 가진다. Physical connection에서 살펴본 바와 같이 서로 다른 두 3D LF상에서 서로 다른 representation을 가지지만 두 light ray는 동일한 light ray에 해당되며, shared light로써 두 3D LF 연결에 사용된다.

$$\left(x_A = x_0, \theta_A = \theta_0, \varphi_A = 0\right) \tag{4.4}$$

$$\left(x_B = x_{\backslash 1}, \theta_B = \theta_1, \varphi_B = 0\right) \tag{4.5}$$

Non-physical connection의 경우 서로 다른 두 3D LF를 구성하는 직선이 겹치는 정도에 따라 physical connection에 비해 더 많은 shared light를 공유한다. 그림 4.3에서 회색으로 표시된 범위를 통과하는 light ray는 두 3D LF를 모두 통과한다. 해당 light ray는 (4.6)과 같이 표현된다.

$$\begin{aligned} & 3DLF_A \cap 3DLF_B \\ &= \left\{(x_A, \theta_A, \varphi_A \middle| x_{A0} \le x_A \le x_{A1}, \theta_{A0} \le \theta_A \le \theta_{A1}\right\} \\ &= \left\{(x_B, \theta_B, \varphi_B \middle| x_{B0} \le x_B \le x_{B1}, \theta_{B0} \le \theta_B \le \theta_{B1}\right\} \end{aligned} \tag{4.6}$$

Physical connection에서 정의한 shared light와 달리 non-physical connection의 경우 한 점이 아닌 범위의 형태로 shared light를 정의하게 된다.

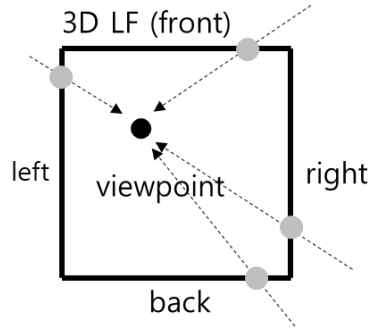
Non-physical connection 기반의 3D LF 연결은 physical connection에 비해 더 많은 양의 shared light를 기반으로 다양한 위치에서 3D LF를 연결할 수 있다는 장점이 있다.

4.4 Non-physical Connection in LFU

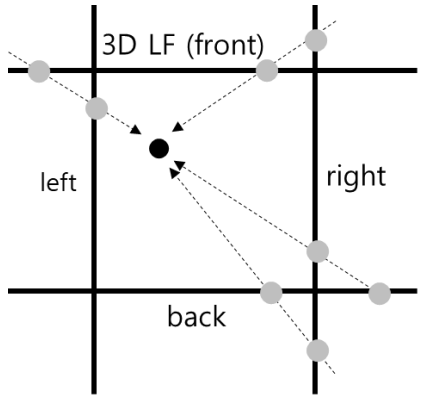
LFU 구조에서 non-physical connection을 이용한 3D LF 연결 방법을 적용해본다. 하지만 non-physical connection을 이용해서 서로 다른 3D LF를 연결하기 위해서는 서로 다른 3D LF를 구성하는 직선이 서로 겹쳐서 light ray를 획득하기 위한 구조가 필요하다.

그림 4.4 (a)는 기본적인 사각형 모양의 LFU에서 임의의 viewpoint가 정해진 그림이다. 사각형 모양의 LFU에서는 viewpoint로 향하는 어떠한 light ray도 서로 다른 3D LF가 동시에 획득할 수 없다. 교차점에서만 가능하다. 그 결과 non-physical connection을 이용하기 위해서는 그림 4.4 (b)와 같이 확장된 사각형 모양의 LFU가 필요하게 된다. 그림 4.4 (a)에서 constant radiance of ray의 특성으로 물리적으로 교차하지 않는 점에서 공유할 수 없었던 light ray를 그림 4.4 (b)의 확장된 LFU 구조에서는 공유할 수 있게 된다. 그림 4.4 (b)에서 볼 수 있듯이 검은색 점선 화살표로 표시된 네 개의 light ray를 이웃한 두 개의 3D LF에서 공통으로 획득하는 것을 확인할 수 있다.

확장된 LFU에서 필요한 확장 길이는 LF를 구성하기 위해 사용하는 카메라 FOV와 관련을 가진다. 카메라 FOV가 큰 경우 수평 각도가 큰 light ray를 획득할 수 있는 반면에 카메라 FOV가 작은 경우 획득할 수 있는 light ray는 수평 각도가 작은 light ray로 제한된다. 그림 4.5 확장된 사각형 구조의 LFU에서 light ray와 확장 길이의 관계를 보여준다. 그림은 LFU 중 front, right, left 방향의 3D LF 부분만 확대한 그림이다. 그림에서 W는 확장되지 않는 사각형 구조의 LFU에서 사각 형의 한 변의 길이를 나타낸다. ΔW는 확장된 사각형 구조의 LFU에서 확장되는 3D LF의 길이를 의미한다. 임의의 점 P0에서 나오는 검은색 점선 화살표로 표시된 light ray는 front 방향의 3D LF와 right 방향의 확장된 3D LF에 끝 점을 통과한다. 해당 light ray는 두 3D LF를 모두 통과하기 때문에 shared light로 정의된다.



(a)



(b)

그림 4.4. Non-physical connection을 이용한 3D LF 연결, (a) 사각형 구조, (b) 확장된 사각형 구조

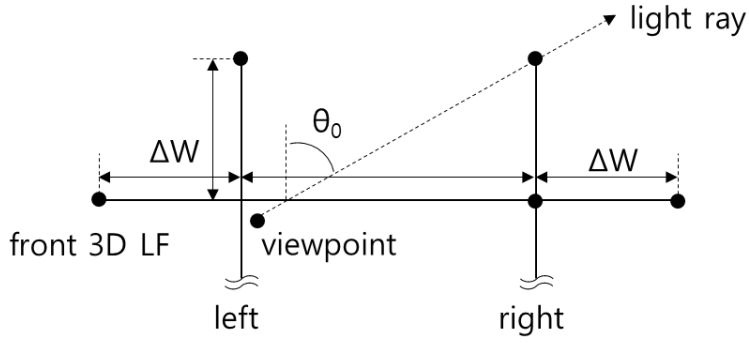


그림 4.5. 확장된 사각형 모양의 LFU에서 확장 길이의 관계

Front 방향의 3D LF를 기준으로 해당 shared light의 수평 각도는 θ_0에 해당되는데, 만약 그림 4.5의 3D LF를 구성하기 위해 사용된 카메라의 FOV가 $(2\times\theta_0)^\circ$ 보다 작은 경우, 해당 light ray는 3D LF에 포함되지 않으며, 이를 사용해서 뷰를 연결하는 것이 불가능해진다. 이처럼 FOV가 제한적인 경우 ΔW를 키워줌으로써 3D LF 내에 속한 light ray로 3D LF를 연결할 수 있도록 조정해 주어야 한다. 식 (4.7)은 3D LF 구성에 사용된 카메라의 FOV에 따른 필요 확장 길이의 관계를 보여준다.

$$\frac{\theta_{in}}{2}=\cot^{-1}\left(\frac{\Delta W}{W}\right) \tag{4.7}$$

식 (4.7)에서 θ_{in}은 3D LF 구성에 사용되는 카메라의 FOV를 의미한다. 표 4.1는 식 (4.7)에서 실제 카메라 FOV에 따른 ΔW의 변화를 비교한 결과이다. 표에서 ΔW/W는 W 대비 ΔW의 길이의 비율을 의미한다.

표 4.1. 카메라 FOV에 따른 확장 길이의 변화

θ_{in}	ΔW/W	θ_{in}	ΔW/W
180°	0.0000	130°	0.4663
170°	0.0875	120°	0.5774
160°	0.1763	110°	0.7002
150°	0.2679	100°	0.8391
140°	0.3640	90°	1.0000

카메라 FOV가 최대 180도를 만족하는 경우 ΔW는 0으로 추가 확장 길이가 필요 없음을 의미한다. 이 경우 physical connection을 이용한 연결을 사용할 수 있다. 카메라 FOV가 감소함에 따라서 ΔW/W 값은 점차적으로 증가하게 된다. 카메라 FOV가 120° 가 되면 약 W/2에 해당되는 길이의 확장이 필요하다. 카메라 FOV가 더 작아져서 90° 가 되면 W만큼의 확장된 길이가 필요가 된다.

확장된 사각형 모양의 LFU는 확장이 없는 사각형 구조에 비해서 추가적인 비용이 필요한 구조이다. 예를 들어 카메라 FOV가 90° 인 경우 확장이 없는 구조에서 LFU를 구성하는데 필요한 3D LF의 길이는 총 4W인 반면, 확장된 구조는 12W로 무려 3배가 증가하게 된다. 하지만 다수의 LFU가 쌓여있는 구조를 고려한다면 그 비용은 감춰지는 효과를 얻을 수 있다. 그림 4.6은 다수의 LFU가 쌓여있는 구조의 예를 보여준다. 그림에서 사각형 형태의 LFU는 5×4의 크기로 연결되어 있다. 그리고 검은색 선 네 개로 구성된 LFU는 확장된 사각형 구조로 구성된 LFU이다. 이 경우 확장이 필요한 부분의 3D LF는 다중 LFU가 연결된 구조를 고려할 때 이미지 데이터가 확보된 영역에 해당되며, 이 때의 추가 비용은 hiding 될 수 있다.

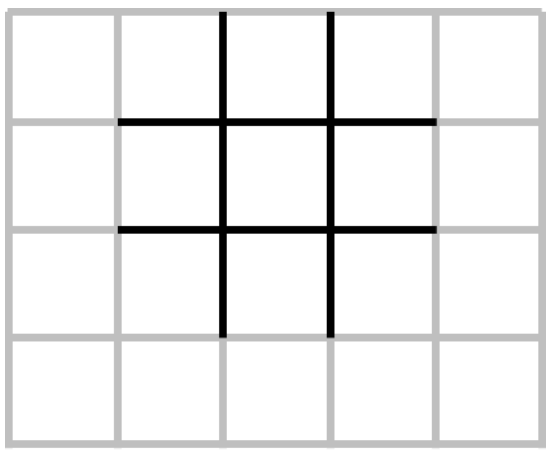


그림 4.6. 다중 LFU 구조에서의 확장된 LFU의 비용 감소

앞서 설명한 것처럼 non-physical connection 기반의 연결을 위해서는 확장된 사각형 모양의 LFU가 필요하다. 확장된 범위를 통해 다양한 shared light를 사용해서 네 개의 3D LF를 연결하고 최종적으로 360도 이미지를 구성할 수 있다. 본 연구에서는 네 방향에 동일한 범위의 뷰 구성 범위를 할당하고 연결하는 방법을 사용한다. 각 방향의 3D LF에 동일하게 90° 만큼의 뷰 구성 범위를 할당한다. 할당된 범위에 대해서 네 개의 3D LF는 뷰를 구성한다. Physical connection과 마찬가지로 각 구성된 뷰의 양 끝 pixel column은 이웃한 3D LF 사이의 shared light로 구성된 pixel column에 해당되며 이를 연결해줌으로써 360도 뷰를 구성하게 된다.

4.5 일반 카메라를 사용하는 3D LF 구성 환경에서의 3D LF Connection

이어서 앞서 제안한 두 가지 3D LF 연결 방법을 다양한 3D LF 구성 환경에 적용해보고 적절한 연결 방안을 제시한다. 본 연구에서는 일반 카메라를 사용해서 3D LF를 구성하는 환경과 360 카메라를 사용해서 3D LF를 구성하는 두 환경에 대해 다룬다. 먼저 일반 카메라를 사용하여 3D LF를 구성하는 환경에 대해서 살펴본다. 일반 카메라는 360도 카메라와 비교하여 획득할 수 있는 light ray가 일반 카메라의 FOV로 제한된다. FOV를 초과하는 수직, 수평 입사 각도를 가지는 light ray는 획득하지 못한다. 식 (4.8)은 일반 카메라를 사용하는 환경에서 획득할 수 있는 light ray를 보여준다.

$$L=\left\{(x,\theta,\varphi)\middle|\frac{-FOV_H}{2}\le\theta\le\frac{FOV_H}{2},\frac{-FOV_V}{2}\le\varphi\le\frac{FOV_V}{2}\right\} \quad (4.8)$$

식 (4.8)에서 FOV_H와 FOV_V는 각각 3D LF를 구성하기 위해 사용하는 카메라의 수평 방향의 FOV와 수직 방향의 FOV를 의미한다. 식 (4.8)은 일반 카메라를 통해 획득할 수 있는 light ray가 수평 입사 각도가 $\pm FOV_H/2$ 이내이면서 동시에 수직 방향의 입사 각도가 $\pm FOV_V/2$ 이내인 모든 light ray 임을 표현하고 있다.

일반 카메라를 사용한 3D LF 구성 환경에서 physical connection을 이용한 3D LF 연결 방안을 살펴본다. 앞서 설명한 바와 같이 LFU 구조에서 physical connection을 이용한 뷰를 구성할 때 임의의 viewpoint와 LFU의 네 개의 교차점 사이의 관계를 통해서 각 3D LF가 구성해야 하는 뷰의 범위을 할당함을 확인하였다. 또한 viewpoint와 각 교차점을 잇는 선은 이웃한 두 3D LF를 연결하는 shared light로 사용된다. 즉, 임의의 viewpoint의 위치에 따라서 shared light의 입사각도가 바뀐다.

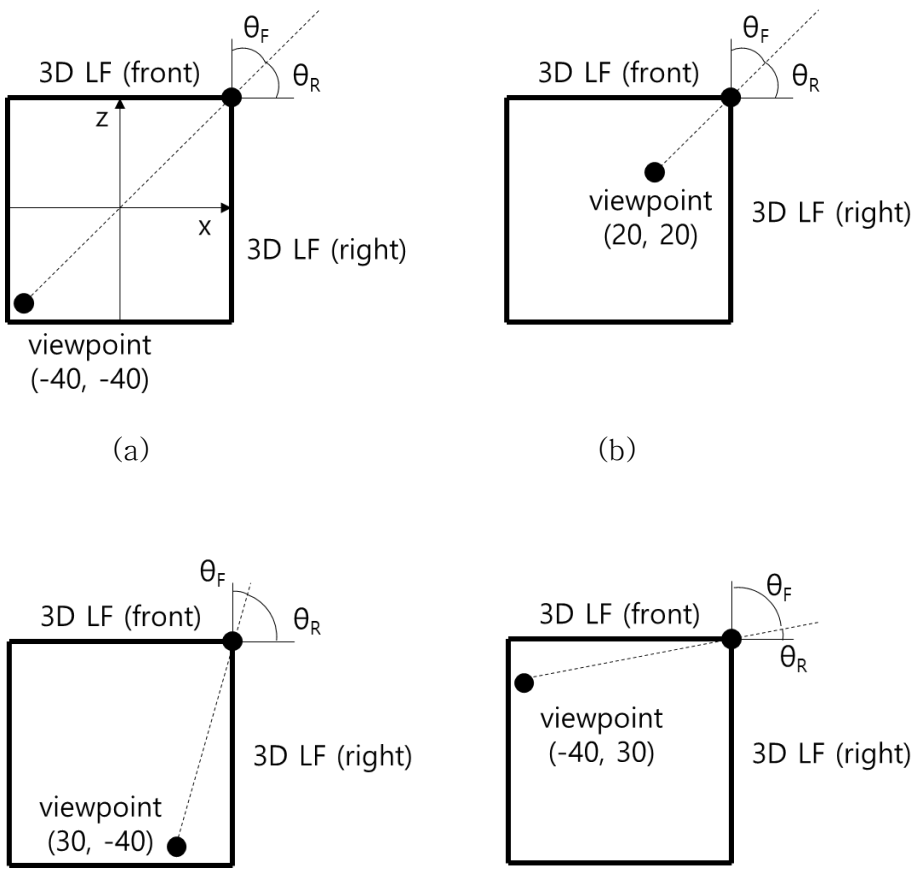


그림 4.7. Viewpoint에 따른 shared light의 수평 입사 각도

그림 4.7은 LFU 내의 다양한 viewpoint 위치에 따른 front 방향의 3D LF와 right 방향의 3D LF의 shared light의 수평 입사 각도를 비교하는 그림이다. 그림의 LFU는 한 변의 길이가 101로 가정한다. x, z 축으로 각각 −50~+50의 범위를 값을 갖는다. 먼저 그림 4.7 (a)에서 viewpoint는 (x=−40, z=−40)에 위치한다. 이 viewpoint와 front 방향의 3D LF와 right 방향의 3D LF의 교차점을 잇는 선을 통해 shared light 가 정해지며, 그림의 검은색 점선과 같다. 이 때 shared light의 front 방향의 3D LF 기준 수평 입사 각도는

θ_F이며, right 방향의 3D LF 기준 θ_R이고, 각각의 값은 식 (4.9), (4.10)의 관계를 통해 구해진다.

$$\theta_F = \tan^{-1}\left(\frac{x_C - x_P}{z_C - z_P}\right) \tag{4.9}$$

$$\theta_R = \tan^{-1}\left(\frac{z_C - z_P}{x_C - x_P}\right) \tag{4.10}$$

식 (4.9), (4.10)에서 x_C와 z_C는 각각 front 방향의 3D LF와 right 방향의 3D LF가 만나는 교차점의 x, z 좌표를 의미하며, 그림 4.7에서 가정한 LFU의 경우 (50, 50)이다. 또한 x_P와 z_P는 각각 임의의 viewpoint의 x, z 좌표를 의미한다. 그림 4.7 (a)의 경우 shared light의 입사 각도는 두 3D LF에서 모두 45° 이다. 그림 4.7 (b)의 경우 viewpoint는 그림 4.7 (a)와 비교하여 x축 방향으로 60, z축 방향으로 60만큼 이동한다. 그림 4.7 (b)의 viewpoint의 경우 여전히 shared light의 입사 각도는 45° 와 45° 로 유지된다.

그림 4.7 (c)의 viewpoint는 그림 4.7 (a)와 비교하여 x축 방향으로 70만큼 이동하였고, z축 방향으로의 이동은 없었다. 이 경우 shared light의 입사 각도가 달라지는데, θ_F는 12.5° 로 작아지고, θ_R은 77.5° 로 증가하게 된다. 수평 입사각도가 77.5° 인 light ray를 획득하기 위해서는 일반 카메라의 수평 입사 각도가 155° 를 만족해야 하는데, 이는 상당히 큰 각도이다. 만약 카메라 FOV가 155° 를 만족하지 못하는 경우 이렇게 FOV를 초과하는 light ray는 구성된 3D LF에 포함되지 못하며 view 구성이 불가능하다. 반면 front 방향의 3D LF에서 필요한 light ray의 입사각도는 12.5° 로 작으며, 어렵지 않게 획득할 수 있다.

그림 4.7 (d)의 viewpoint는 그림 4.7 (c)의 viewpoint와 반대로 (−40, 30)에 위치한다. 그 결과 입사 각도 역시 반대로 θ_F는 77.5° ,

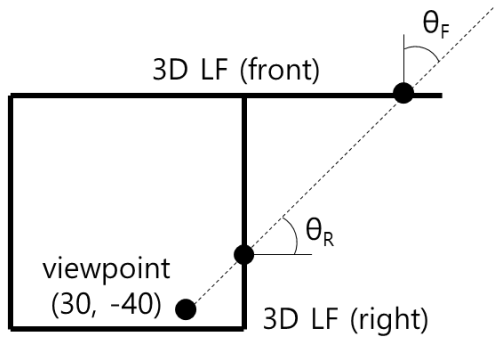
θ_R은 12.5° 가 된다. 그림 4.7 (d)의 viewpoint에서는 front 방향의 3D LF에서 shared light를 포함하지 못할 가능성이 크며, 뷰를 구성하기 어려울 수 있다.

이처럼 physical connection을 이용한 뷰 구성에서는 viewpoint의 위치에 따라 요구되는 shared light의 입사 각도가 달라지며, 최대 ±90° 의 입사 각도를 가지는 light ray가 필요하다. 이를 위해서는 180도의 FOV를 만족하는 장비가 필요하며, 일반 카메라를 이용한 획득은 현실적으로 어렵다.

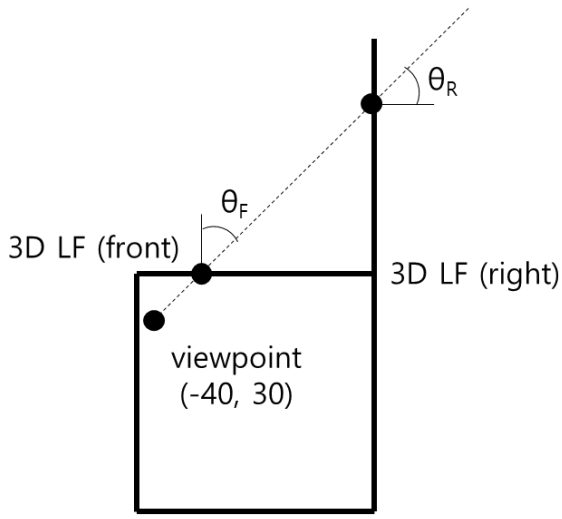
Non-physical connection을 이용하는 방법을 살펴본다. Non-physical connection의 경우 physical connection과 달리 LFU 내의 임의의 viewpoint에 관계 없이 네 개의 3D LF에 항상 90° 만큼 뷰 구성 범위를 할당한다. 따라서 shared light의 수평 입사 각도는 항상 ±45° 로 고정된다. 그림 4.8은 그림 4.7 (c)와 (d)의 viewpoint의 위치와 같은 경우 non-physical connection 방법에서 정해진 shared light를 보여준다. 그림 4.7 (a)와 (b)의 viewpoint 위치의 경우 동일하게 ±45° 의 입사각도를 가지는 light ray를 shared light로 정의한다. 그림 4.8 (a)의 경우 임의의 viewpoint는 (30, −40)에 위치한다. Non-physical connection을 이용한 경우 여전히 ±45° 의 입사 각도를 가지는 light ray를 shared light로 정한다. 이는 확장된 사각형 모양의 LFU를 통해 가능하다. 그림에서 shared light가 통과하는 front 방향의 3D LF 상에서의 x_F 값과 right 방향의 3D LF 상의 x_R 값은 식 (4.11), (4.12)와 같다.

$$x_F = x_P + \left(\frac{W_{LFU}}{2} - z_P \right) \cdot \tan(45°) \tag{4.11}$$

$$x_R = -z_P - \left(\frac{W_{LFU}}{2} - x_P \right) \cdot \tan(45°) \tag{4.12}$$



(a)



(b)

그림 4.8. Non-physical connection 환경에서 임의의 viewpoint에 따른 shared light 수평 입사 각도

식 (4.11), (4.12)에서 W_{LFU}는 가정하는 사각형 모양의 LFU 구조의 한 변의 길이를 의미한다. x_P, z_P는 임의의 viewpoint의 위치를 의미한다. x_F는 front 방향의 3D LF를 구성하는 직선 위에서의 위치를 의미하며, x_R는 right 방향의 3D LF를 구성하는 직선 상에서의 위치를 의미한다. 그림 4.8 (a)의 경우를 살펴보면, 해당 viewpoint에서 non-

physical connection을 이용할 때 선택되는 shared light는 front 방향의 3D LF의 직선 상의 위치 x_F = 120인 점을 통과고, right 방향의 3D LF의 직선 상의 위치 x_R = −20인 점을 통과한다. 동일한 방식으로 그림 4.8 (a)의 경우 x_F와 x_R은 각각 −20, −120이 된다.

그림 4.9는 앞서 설명한 physical connection과 non−physical connection을 사용해서 뷰를 구성한 결과를 비교한다. 일반 카메라를 사용하는 환경에서의 결과이며, 두 개의 viewpoint에서의 뷰를 두 연결 방법을 사용해서 구성한 결과이다. 구성된 뷰는 front 방향의 뷰와 left, right 방향 뷰의 일부를 통해서 구성된 결과이다. 그림 4.9 (a)와 그림 4.9 (b)는 임의의 viewpoint가 (0, 10)에 위치하는 경우를 보여주며, 그림 4.9 (a)는 확장된 사각형 모양의 LFU에서 non−physical connection을 이용해 뷰를 구성하는 과정을, 그림 4.9 (b)는 사각형 모양의 LFU에서 physical connection을 이용해서 구성한 뷰를 보여준다. 그림 4.9 (a)에서 보는 것처럼 non−physical connection을 이용하는 방법에서는 임의의 viewpoint를 기준으로 네 개의 3D LF에 각각 90° 에 해당되는 뷰 구성 범위를 할당한다. Non−physical connection을 통개 구성된 뷰는 빈 공간 없이 모두 구성되는 것을 확인할 수 있다.

그림 4.9 (b)의 경우 임의의 viewpoint와 네 개의 교차점을 잇는 검은색 점선 화살표를 통해서 shared light가 정해지고, 각각의 3D LF에서 구성 해야 하는 범위가 정해진다. 그림 4.9 (b)의 경우 역시 해당 viewpoint의 대한 뷰는 빈 공간 없이 모두 구성된 것을 확인할 수 있다. 단, 구성된 뷰에서 흰색 점선 사각형에 해당되는 부분을 어느 3D LF를 사용해서 구성했는가의 차이가 있다. 그림 4.9 (a)의 경우 해당 범위를 각각 right 방향의 3D LF와 left 방향의 3D LF에 할당하여 뷰를 구성한다. 반면, physical connection 기반의 3D LF 연결을 사용하는 그림 4.9 (b)의 경우 해당 범위를 front 방향의 3D LF에 할당하여 뷰를 구성한다. 해당 범위는 front 방향의 3D LF 기준 ±45° 의 수평 입사각도를 초과하는 영역에 해당된다.

(a)

(b)

(c)

(d)

그림 4.9. 두 개의 viewpoint에서 physical connection과 non-physical connection을 이용한 뷰 구성 결과 비교

할당 방식은 다르지만 결과적으로 (0, 10)의 viewpoint에서의 뷰 구성은 두 방법이 모두 잘 동작하는 것을 확인할 수 있다. 그림 4.9 (c)와 그림 4.9 (d)는 (0, 30)의 viewpoint에서의 뷰를 non-physical 기반의 연결과 physical 기반의 연결을 통해 구성한 결과를 보여준다. 해당 viewpoint는 앞선 그림 4.9 (a)와 그림 4.9 (b)에 비해 front 방향으로 다소 이동한 위치이다. Non-physical 기반의 연결에서는 viewpoint에 관계 없이 네 개의 3D LF에 90° 의 동일한 범위를 할당하며, 그림 4.9 (c)의 경우 역시 네 개의 3D LF에 90° 의 범위를 할당하여 뷰를 구성한다. 구성된 뷰는 빈 공간 없이 모두 채워진 것을 확인할 수 있다.

그림 4.9 (d)는 동일한 viewpoint에서 physical connection을 이용하여 뷰를 구성한 결과이다. 그림 4.9 (c)와 달리 그림 4.9 (d)에서는 viewpoint와 네 개의 교차점의 관계를 통해 구성 범위를 할당한다. 그림 4.9 (b)의 할당 범위와 비교하여 viewpoint가 앞으로 이동함에 따라 front 방향의 3D LF에 더 많은 양의 범위를 할당하는 것을 확인할 수 있다. 뷰의 가장자리 부분은 수평 입사각도가 큰 light ray가 필요한 부분에 해당되며, 해당 light ray는 카메라의 FOV 제한으로 인해 획득하지 못한 부분에 해당된다. 그 결과 non-physical connection을 이용한 방법과 달리 구성한 뷰의 일부 채우지 못하는 부분이 발생하게 된다.

4.6 360도 카메라를 사용하는 3D LF 구성 환경에서의 3D LF Connection

일반 카메라를 사용하는 환경과 달리 360도 카메라는 full FOV를 지원하며 360도 방향의 모든 light ray를 획득할 수 있다. 따라서 일반 카메라를 사용하는 환경에서 불가피하게 획득하지 못한 light ray로 인해 physical connection의 사용이 제한된 것과 달리 360도 카메라를 사용하는 환경에서는 physical connection 방법과 non-physical connection 방법을 모두 사용할 수 있다. 하지만 수직, 수평의 입사각도가 큰 light ray의 경우 3D LF의 제한적 환경에서 에러를 포함하게 된다. 이번 섹션에서는 수평, 수직 입사각도가 큰 light ray를 3D LF 구조에서 사용할 때 발생할 수 있는 두 가지 에러를 살펴보고, 이를 보완한 새로운 연결 방법을 제시한다.

4.6.1 수평, 수직 입사 각도가 큰 light ray를 이용한 3D LF 뷰 구성에서 발생하는 에러

360도 카메라를 이용함으로써 보다 많은 양의 light ray를 사용할 수 있게 되며, 360도 방향으로 입사하는 모든 light ray를 사용할 수 있게 된다. 하지만 360도 방향으로 입사하는 모든 light ray 중에서 수평, 수직 입사 각도가 큰 light ray는 3D LF 기반의 뷰 구성에서 에러를 포함할 수 있다. 두 가지 종류의 에러가 발생하는데, 하나는 뷰 구성에서 물체가 휘어지는 것처럼 뷰가 구성되는 에러이다. 이는 3D LF의 뷰 구성은 기본적으로 스테레오 constraint [43]를 기반으로 하는데, 수평, 수직 입사 각도가 큰 light ray는 이를 위반하기 때문이다. 좌, 우로 배치된 스테레오 카메라에서 동일한 오브젝트는 동일한 픽셀 라인에 맺힌다. 마찬가지로 직선 상을 이동하면서 획득한 light ray를 고려할 때, 임의의 물체로부터 입사하는 light ray는 이미지 상에 동일한 수직 입사 각도를 가지는 것을 고려한다. 하지만 ±90° 방향에 위치한

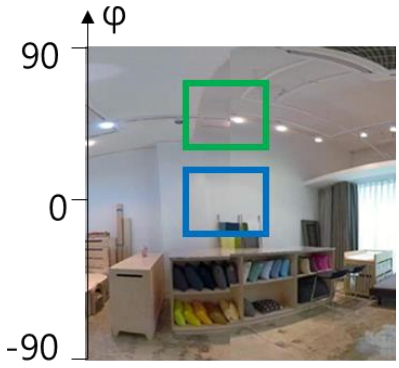
물체로부터 입사하는 light ray는 3D LF의 직선 상의 서로 다른 x에 서로 다른 수직 입사 각도를 가지고 입사하게 된다. 그림 4.10은 수평 입사 각도가 큰 light ray를 이용한 3D LF의 뷰 구성에서 발생하는 banding 에러를 보여준다. 그림 4.10 (a)는 3D LF를 이용해 구성한 이미지이며, 그림 4.10 (b)는 해당 공간을 실제로 촬영한 이미지이다. 그림 4.10 (a)의 입사 각도가 −90° 에 위치하는 탁자에서 그림 4.10 (b)의 실제 촬영된 이미지의 탁자 모양과 달리 휘어지는 듯 한 결과가 만들어지는 것을 확인할 수 있다.

(a) (b)

그림 4.10. 수평 입사 각도가 큰 light ray를 이용한 뷰 구성에서 발생할 수 있는 banding error, (a) 구성된 뷰, (b) 촬영한 이미지

수직, 수평 각도가 큰 light ray를 이용한 3D LF 뷰 구성에서 발생하는 또 다른 에러는 두 3D LF를 연결하는 부분에서 발생하며, 특히 non−physical connection을 이용한 3D LF 연결에서 발생한다. 앞서 설명한 것처럼 non−physical connection은 constant radiance of ray의 특성을 바탕으로 서로 다른 두 3D LF를 물리적으로 교차하지 않는 점에서 연결하는 방법을 제시한다. 하지만 실제로 물리적인 서로 다른 점에서 획득하는 light ray는 정확히 같은 light ray를 공유하지는 않는다. 정확히 동일한 물체, 동일한 배경으로부터 발산되는 light ray임은 맞지만, 각 점에 입사하는 수직 각도가 다르기 때문에 이를

고려하지 않고 non-physical connection을 이용해서 3D LF를 연결하여 구성된 뷰는 두 뷰가 연결되는 부분에서 mismatching 에러가 발생한다. 이 에러는 특히 수직 입사 각도가 클수록 커지며, 수직 입사 각도가 ±90° 일 때 최대가 된다. 수직 입사 각도가 0° , ±180° 인 경우에는 정확하게 동일한 light ray를 공유하며, mismatching 에러는 없다. 그림 4.11는 앞서 설명한 mismatching 에러가 발생한 경우를 보여준다. 그림 4.11 (a)에서 녹색 네모 박스와 파란색 네모 박스는 각각 수직 입사 각도가 0° 에 가까운 light ray에 해당하는 부분과 약 50° 에 가까운 light ray로 구성된 부분을 보여준다. 그리고 해당 부분을 확대한 이미지는 그림 4.11 (b), 그림 4.11 (c)와 같다.



(a)

(b)

(c)

그림 4.11. Non-physical connection의 mismatching 에러

그림 4.11 (b)의 뷰는 수직 입사 각도가 0° 에 해당하는 부분이며, 두 뷰 사이에서 뷰가 자연스럽게 연결되지 않는 부분이 눈에 띄지 않는다. 반면, 그림 4.11 (c)의 경우 수직 입사 각도가 약 50° 에 가까운 부분이며, 이 경우 두 뷰가 자연스럽게 연결되지 않는 부분이 명확하게 드러난다.

일반 카메라를 고려하는 환경에서는 banding 에러와 mismatching 에러를 고려하지 않았다. 먼저 banding 에러의 경우 수평 입사 각도가 ±90° 에 가까운 light ray에서 발생하는데 일반 카메라의 경우 FOV 제한으로 인해 수평 입사 각도가 ±90° 만큼 큰 light ray는 애초에 획득 자체가 불가능했기 때문이다. 그리고 일반 카메라를 고려한 환경에서는 non-physical connection을 기반으로 ±45° 이내의 light ray만 사용하기 때문에 banding 에러가 발생하지 않았다. Mismatching 에러의 경우 수직 입사 각도가 큰 light ray에서 발생한다. 하지만 이 역시 일반 카메라에서는 수직 FOV의 제한으로 인해 mismatching 에러가 눈에 띌 정도로 수직 입사 각도가 큰 light ray가 사용되지 않았다. 360도 카메라를 이용한 환경에서 위 두 에러를 고려한 새로운 3D LF 연결 방안이 필요하다.

4.6.2 Hybrid 3D LF Connection

360도 카메라를 사용하는 환경을 위해 새롭게 제안하는 3D LF 연결 방식은 hybrid 3D LF connection 이다. Hybrid 3D LF connection은 앞서 설명한 non-physical connection과 physical connection 둘 중 하나를 선택해서 사용하는 방식이 아니라 LFU 내의 임의의 viewpoint의 위치에 따라서 non-physical connection과 physical connection을 선택적으로 사용하기 위한 방법이다.

LFU의 중심부에 viewpoint가 위치하는 경우 physical connection을 이용해서 3D LF를 연결한다. Viewpoint가 LFU의 가운데에 위치한 경우, viewpoint와 각 교차점과의 연결을 통해서 결정되는 shared light의 수평 입사 각도는 대체로 작다. 따라서 banding 에러는 눈에 띄지 않는 작은 수준으로 유지될 수 있다. 또한 physical connection은 물리적은 교차점을 통과하는 shared light를 사용하기 때문에 mismatching 에러가 전혀 발생하지 않는다.

Viewpoint가 이동하고 LFU 내의 가장자리로 이동함에 따라 viewpoint와 네 개의 교차점을 잇는 선 중 $\pm K^{\circ}$ 를 초과하는 shared light가 발생하는 경우 해당 shared light는 $\pm K^{\circ}$ 의 수평 입사 각도를 가지는 light ray를 shared light로 non-physical connection을 이용해서 연결한다. 사용하는 light ray를 $\pm K^{\circ}$ 이내로 조정함으로써 banding error를 일정 수준으로 제한시킬 수 있다. 또한 가능한 물리적인 교차점에서 멀지 않은 지점에서 두 3D LF를 연결해줌으로써 mismatching 에러 역시 줄여주는 효과가 있다.

그림 4.12는 임의의 viewpoint에 대한 뷰를 구성함에 있어서 hybrid 3D LF connection 방식이 적용되는 예를 보여준다. 그림에서 viewpoint는 비교적으로 앞으로 이동한 상태이다. 해당 viewpoint와 front 방향의 3D LF와 left 방향의 3D LF의 교차점을 통과하는 shared light의 수평 입사 각도는 θ_0으로 이는 $\pm K^{\circ}$ 보다 작다. 이 경우 θ_0의 입사 각도를 가지는 light ray를 그대로 shared light로 사용한다.

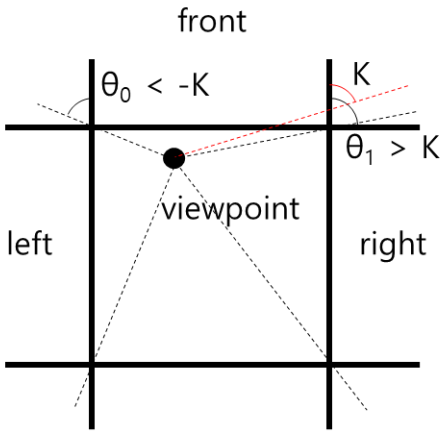


그림 4.12. Hybrid 3D LF connection의 예

반면, viewpoint와 front 방향의 3D LF와 right 방향의 3D LF의 교차점을 통과하는 shared light의 수평 입사 각도는 θ_1로 이는 ±K° 보다 크다. 이 경우 θ_0의 입사 각도를 가지는 light ray 대신 빨간색 점선으로 표시된 K° 의 입사 각도를 가지는 light ray를 shared light로 사용한다. 해당 light ray는 front 방향의 3D LF와 right 방향의 3D LF의 교차점을 통과하지는 않지만 확장된 구조를 통해 물리적으로 교차하지 않는 점에서 공유할 수 있다. 두 3D LF는 non-physical connection을 사용해서 연결한다. Hybrid 3D LF connection을 위해서는 banding 에러와 mismatching 에러를 적절히 줄여주기 위한 최적의 K° 선정이 중요하다. 본 논문에서는 다음의 과정을 바탕으로 K° 를 정한다.

먼저 (4.13)는 banding 에러와 mismatching 에러를 모두 포함하는 total 에러를 정의한다.

$$E_t(K,d)=\frac{E_b(K,d)}{\max(E_b)}+\frac{E_m(K,d)}{\max(E_m)} \tag{4.13}$$

식 (4.13)에서 E_b와 E_m은 각각 banding 에러와 mismatching 에러를 의하며, E_b와 E_m은 K, d에 따라 변한다. d는 3D LF를 구성하기 위해 이미지를 촬영하는 점과 light ray가 나오는 물체와의 거리를 의미한다. 식 (4.13)의 전체 에러는 banding 에러와 mismatching 에러의 합으로 정의되며, 이 때, 두 에러는 최대 값을 기준으로 정규화된 값이다.

식 (4.14)는 주어진 K, d에 대한 mismatching 에러를 정의한다.

$$E_m(K,d)=\sum_{x,z}\sum_{j=-90^\circ}^{90^\circ}\left|\varphi_a(K,j,d)-\varphi_b(K,j,d)\right| \quad (4.14)$$

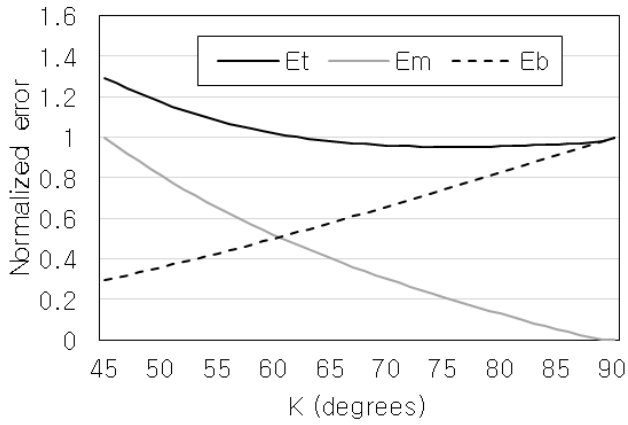
Mismatching 에러는 서로 다른 두 3D LF가 만나는 하나의 픽셀 라인에서 발생한다. (4.14)에서 x와 z는 주어진 viewpoint의 위치를 의미한다. φ_a(K, j, d)와 φ_b(K, j, d)는 light ray가 나오는 물체의 위치에 대한 정보 longitude, latitude, 그리고 거리가 각각 K, j, d일 때, 해당 물체로부터 나오는 light ray가 서로 다른 두 3D LF에 입사하는 점에서의 수직 입사 각도를 의미한다. 이 때 서로 다른 두 3D LF는 물리적으로 교차하지 않는 점에 해당되며, non-physical connection을 통해 연결되는 점을 의미한다. Hybrid 3D LF connection에서 non-physical connection은 항상 K° 의 입사 각도를 가지는 light ray를 사용한다. 따라서 K° 에 위치한 물체로부터의 light ray가 에러를 정의하는 기준이 된다. φ_a(K, j, d)와 φ_b(K, j, d)의 차이만큼 mismatching 에러가 발생한다. Mismatching 에러는 픽셀 라인 단위로 발생하기 때문에 90° ~ −90° 의 longitude에 위치한 모든 물체를 대상으로 에러를 모두 더한다. K° 의 고정된 입사 각도에 대한 에러를 정의하더라도 임의의 viewpoint에 따라 shared light가 통과하는 두 3D LF 상의 위치가 달라진다. 따라서 viewpoint에 따른 에러 또한 모두 더하여 최종적으로 mismatching 에러를 정의한다.

마지막으로 (4.15)은 banding 에러를 정의한다.

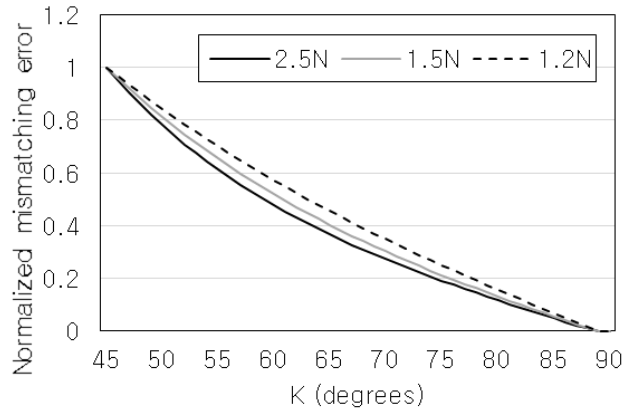
$$E_b(k) = \sum_{i=0}^{k^\circ} \sum_{j=-90^\circ}^{90^\circ} |\varphi_n(i, j, d) - \varphi_{n+1}(i, j, d)| \qquad (4.15)$$

식 (4.15)에서 φ_n(i, j, d)와 φ_b(i, j, d)는 light ray가 나오는 물체의 위치에 대한 정보 longitude, latitude, 그리고 거리가 각각 i, j, d일 때, 해당 물체로부터 나오는 light ray가 동일한 3D LF의 직선 상의 이웃한 두 점으로 입사하는 수직 입사 각도를 의미한다. 이웃한 점에서 수직 입사 각도의 차이가 점차적으로 쌓이게 되며, 수평 입사 각도가 ±90° 에 가까워짐에 따라 banding 에러로 드러난다. 따라서 사용하는 light ray를 K° 로 제한할 경우 K° 까지의 에러의 합이 전체 banding 에러에 포함되며, 90° ~ −90° 의 longitude에 위치한 모든 물체를 대상으로 에러를 모두 더해줌으로써 banding 에러를 정의한다. 단, banding 에러는 viewpoint에 dependent 하지 않기 때문에 viewpoint 위치는 포함되지 않는다.

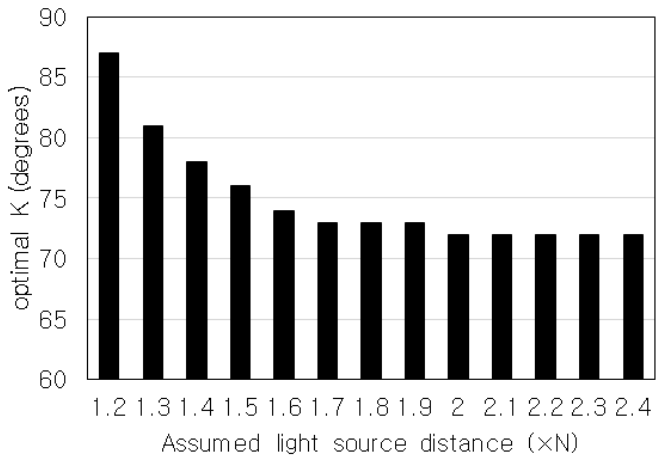
그림 4.13은 K와 d에 따른 에러를 그래프로 살펴본다. K는 45° ~ 90° 의 범위에 대해서 살펴본다. d의 경우 절대적인 거리 값은 의미가 없으며, LFU의 크기 대비 비율로 정의한다. 그림 4.13 그래프에서 LFU는 사각형 모양 기준 한 변의 길이가 N 값을 갖는다고 가정했다. 그리고 이 때 d 값은 1.2N ~ 2.5N으로 가정한다. 즉, 한 변의 길이 대비 1.2 배의 거리에 물체가 위치하는 경우, 2.5 배의 거리에 물체가 위치하는 경우를 고려하는 것이다. 먼저 그림 4.13 (a)는 (4.13) ~ (4.15)에서 정의한 Et, Eb, Em의 관계를 비교한 그래프이며, x축은 K, 그리고 이 때 d 값은 1.5N이다. 그림 4.13 (a)에서 볼 수 있듯이 Em은 K가 증가함에 따라 감소한다. 이는 K가 커짐에 따라서 non-physical connection을 통해서 연결되는 두 3D LF의 점 사이의 거리가 가까워지기 때문이다.



(a)



(b)



(c)

그림 4.13. Banding 에러와 mismatching 에러 그래프

반면, Eb는 K가 증가함에 따라 같이 증가하는데, 이는 큰 K를 사용하는 경우 ±90° 에 가까운 light ray의 사용 빈도가 증가하기 때문이다. 두 에러를 합친 Et는 오목한 그래프의 형태를 나타내며, K가 76° 일 때 가장 작다. 즉, d가 1.5N인 경우 최적의 K는 76° 라고 할 수 있다. (4.15)에서 Eb를 정의할 때 가정한 동일한 3D LF 내의 이웃한 두 점 사이의 거리는 N/100 정도로 매우 작다. 따라서 d 변화에 따른 Eb의 변화는 크지 않다. 반면, Em의 경우 수직 입사 각도를 비교하는 두 점의 거리가 viewpoint에 따라 N보다 커지는 경우도 발생할 수 있다. 그림 4.13 (b)는 d를 2.5N, 1.5N, 1.2N으로 바꿔줌에 따른 Em의 변화를 비교한 그래프이다. K에 따른 Em의 변화를 살펴보면 물체의 거리가 멀어질수록 그래프는 더 오목한 결과를 나타낸다. 이는 전체 에러에 반영되어 d가 큰 물체일수록 최적의 K 값을 더 작게 만들어준다.

마지막으로 그림 4.13 (c)는 d에 따라 정해진 K를 비교한 결과이다. 작은 d가 1.2N인 경우 최적의 K는 87° 였으며, d가 증가함에 따라 K값은 급격히 감소하다가 d가 1.7N이 되면서 감소량이 급격히 줄어들고, D가 2N 이상이 되면 최적 K는 72° 로 유지가 되었다. 전반적으로 물체의 거리가 가까운 환경에서는 큰 K를 사용해 주고, 물체의 거리가 먼 환경에서는 작은 K가 적합하다는 결론이 나왔다. 하지만 실제 구조를 구성하는 환경에 따라 이는 조정될 필요가 있다. 실제 환경에 모든 물체가 동일한 거리에 있을 수는 없으며 다양한 거리에 배치될 것이기 때문에 그 때 환경을 고려할 필요가 있다.

제 5 장 View generation in 3D LF Stack

이번 장에서는 3장에서 소개한 두 가지 해결 과제 중 두 번째 과제인 3D LF stack 구조에서 뷰를 구성하는 문제를 해결하기 위한 제안 방안을 소개한다. 3D LF stack은 단일 3D LF 구조를 가정하는 것에 비해 일정 단위로 3D LF를 배치함으로써 vertical parallax를 표현하지 못하는 3D LF의 구조적 한계로 인한 뷰의 왜곡 현상을 일정 수준으로 제한할 수 있다. 하지만 3D LF 기반의 뷰 구성에서 에러의 포함은 불가피하며, 이는 viewpoint가 이동하고 뷰를 구성하는 사용하는 대상 3D LF가 바뀌는 순간 드러난다. 3D LF가 바뀌는 순간 뷰가 급격히 바뀌는 문제가 생긴다. 이번 장에서는 이처럼 3D LF stack 구조에서 뷰가 급격히 바뀌는 문제의 이유를 정리하고, 이를 해결하기 위해 제시한 방법을 소개한다.

3D LF stack은 단일의 3D LF 구조에서 넓은 공간을 대상으로 움직이는 경우, 3D LF와 viewpoint 사이의 거리가 점차적으로 멀어지게 되고, vertical parallax를 표현하지 못하는 3D LF의 구조적인 한계로 인해 에러가 증가하는 문제가 커지는 문제

5.1 3D LF Stack 구조에서 뷰가 급격히 바뀌는 문제

3D LF stack에서 viewpoint가 이동하고 뷰 구성에 사용하는 3D LF가 바뀌면 뷰가 급격히 바뀐다. 이는 3D LF에서 constant depth를 고려한 뷰 구성이 불가피한 에러를 포함하고 결과적으로 구성된 뷰와 실제 뷰의 차이가 발생하기 때문이다.

그림 5.1은 3D LF Stack의 뷰의 급변 문제에 대해 설명한다. 그림 5.1에는 두 개의 3D LF, $3DLF_A$와 $3DLF_B$가 앞, 뒤로 배치되어 있다 그리고 두 개의 viewpoint A, B가 있는데, viewpoint A는 $3DLF_A$와 $3DLF_B$ 사이에 배치되어 있으며, $3DLF_B$에 가까운 위치에 있다.

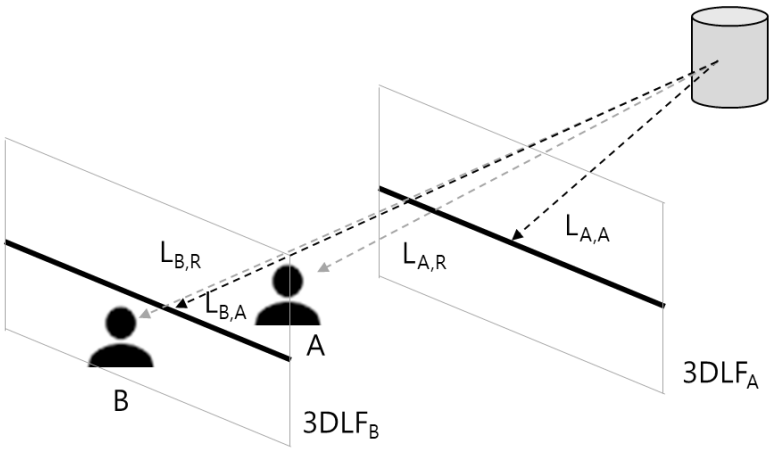


그림 5.1. 3D LF stack 환경에서 viewpoint A와 viewpoint B 가 배치된 예

그리고 Viewpoint B는 $3DLF_B$ 뒤에 위치한다. 이 경우 viewpoint A는 전방에 위치한 $3DLF_A$를 사용해서 뷰를 구성하게 되고, viewpoint B는 $3DLF_B$를 사용해서 뷰를 구성하게 된다. Viewpoint A에서 뷰를 구성함에 있어서 원통 모양의 물체를 구성하기 위해 실제 필요한 light ray는 물체로부터 viewpoint A로 향하는 회색 점선 화살표로 표시된 $L_{A,R}$이다. 하지만 해당 light ray는 $3DLF_A$에 포함되어 있지 않으며, $3DLF_A$에 속한 light ray 중 $L_{A,A}$를 대체 light ray로 사용하여 뷰를 구성하게 된다. 물체의 정확한 depth를 알고 이를 적용한다면 light ray는 정확하게 대체될 수 있다. 하지만 constant depth를 고려할 경우 뷰를 구성하기 위해 필요한 light ray 중 일부는 정확한 depth를 사용할 수 있지만 대부분의 경우 실제와 다른 depth를 사용해서 대체 light ray를 선택하게 된다. Viewpoint B의 경우, 해당 viewpoint의 뷰를 구성하기 위해 필요한 light ray, $L_{B,R}$은 $3DLF_B$에 속하지 않으며, 이를 동일한 물체로부터 입사한 light ray, $L_{B,A}$로 대체하여 사용한다.

그림 5.1의 구조에서 뷰를 구성하기 위해 실제 사용한 $L_{A,A}$, $L_{B,B}$는

정확한 depth가 고려 되었을 가능성도 있지만 대부분의 경우 그렇지 못하며, 실제 뷰와 다른 결과를 만든다. 두 viewpoint에서 구성된 뷰의 차이가 있다면 뷰 구성에 사용된 3D LF와의 거리가 viewpoint A의 경우 비교적 먼 반면, viewpoint B의 경우 $3DLF_B$와의 거리가 매우 가깝다. 이는 두 viewpoint에서 구성된 뷰에 반영된 에러의 정도가 다름을 의미한다. Viewpoint A의 뷰는 비교적 에러가 많이 포함되는 반면, viewpoint B의 뷰는 거의 실제 뷰와 동일한 뷰가 구성된다. 결과적으로 viewpoint A에서 viewpoint B로 이동하는 경우 에러가 많이 포함되어 실제 뷰와 차이가 큰 뷰에서, 에러가 많이 포함되지 않아 실제와 거의 유사한 뷰로 바뀌게 되고, 사용자의 입장에서는 뷰가 급격히 바뀌었다는 느낌을 받게 된다.

그림 5.1의 경우로 한정해서 viewpoint A에서의 뷰 또한 $3DLF_B$를 사용해서 구성하는 방법이 사용될 수 있다. 그리고 이 방법을 바탕으로 viewpoint A에서 viewpoint B로 이동할 때 뷰의 급변 문제를 해결할 수 있다. 하지만 결국 다양한 viewpoint의 이동에서 3D LF와 viewpoint 사이의 거리를 일정 수준으로 제한 시키기 위해 대상 3D LF가 바뀌는 순간이 필요하고 그 때의 뷰의 급변 문제는 불가피 하다. 따라서 이를 개선하기 위해 다수의 3D LF를 배치한 3D LF stack 구조를 위한 뷰 구성 방안이 필요하다.

5.2 앞, 뒤로 배치된 3D LF 사이의 light ray 공유

본 논문은 3D LF를 기반으로 구조를 구성한다. 3D LF는 직선을 따라 통과하는 light ray의 집합으로 구성이 되며, 앞서 설명한 본 논문의 구조의 특성처럼, 한 방향으로 입사하는 light ray가 아니라 양 방향으로 통과하는 모든 light ray를 포함한다. 그림 5.2은 본 논문에서 가정하는 3D LF의 구조와 해당 3D LF가 포함하는 light ray의 예를 보여준다.

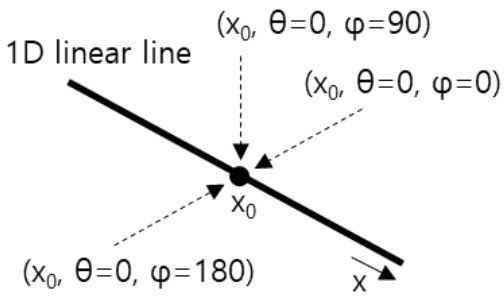


그림 5.2. 3D LF 구성에 사용되는 직선 구조와 light ray

굵은 검은 선은 3D LF를 구성하기 위해 가정한 직선을 나타낸다. 가정하는 3D LF는 해당 직선을 통과하는 모든 light ray를 포함한다고 가정한다. 직선 위의 임의의 점 x_0가 있다고 할 때 해당 점을 통과하는 모든 light ray는 (5.1)과 같다.

$$\left((x,\theta,\varphi)\middle| x = x_0, -90 \le \theta \le 90, -180 \le \varphi \le 180\right) \quad (5.1)$$

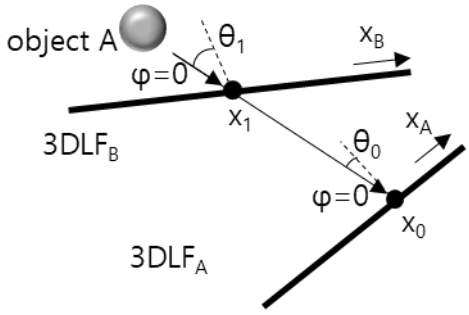
점 x_0를 통과하면서 수평 방향의 입사 각도가 −90° ~ 90° 의 범위를 가지며, 수직 입사 각도가 −180° ~ 180° 의 범위를 가진다. 360도 방향으로 입사하는 light ray는 수평 방향의 입사 각도를 −180° ~ 180° 의 범위로 수직 입사 각도를 −90° ~ 90° 의 범위로 둘 수 있지만, 본 논문의 경우 수직 방향으로 −180° ~ 180° 의 입사 각도를 가지는 모든 light ray를 하나의 세트로 고려하기 때문에 (5.1)과 같은 범위로 정의한다. 그림 5.2의 세 개의 점선 화살표는 x_0로 입사하는 light ray의 예를 보여주며, 각각 (x_0, 0, 180), (x_0, 0, 90), (x_0, 0, 0)의 representation으로 표현된다.

그림 5.3 (a)는 서로 다른 두 3D LF, $3DLF_A$와 $3DLF_B$를 구성하는

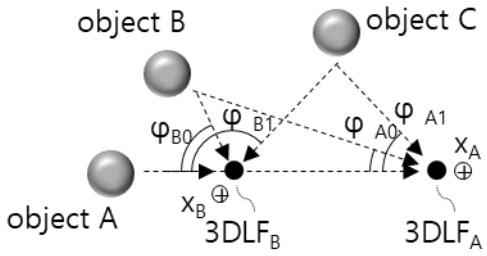
두 개의 직선이 배치된 그림을 보여준다 그림에서 굵은 검은 선은 3D LF를 구성하기 위해 가정한 직선을 의미한다. 두 3D LF는 앞서 가정한 3D LF stack의 구조처럼 정렬되어 있는 상황을 고려하지는 않는다. 임의의 물체 A를 가정하였으며, 해당 물체로부터 검은색 직선으로 표시된 light ray가 $3DLF_A$와 $3DLF_B$를 구성하기 위한 두 직선에 입사하며, 해당 light ray는 두 3D LF에 속한다. 물체 A에서 $3DLF_B$로 입사하는 light ray는 $3DLF_B$의 x_B 상의 x_1에 θ_1의 수평 입사 각도로 입사하였고, $3DLF_A$에는 X_A 상의 x_0에 θ_0의 수평 입사 각도로 입사하였다. 두 경우 모두 수직 방향으로는 0° 의 입사 각도를 갖는다.

그림 5.3 (b)는 (그림) (a)를 옆에서 바라본 그림이며, $3DLF_B$에서 ($x_B=x_1$, $\theta_B=\theta_1$)와 $3DLF_A$에서 ($x_A=x_0$, $\theta_A=\theta_0$)의 조건을 만족하는 light ray 중 일부를 비교한다. 그림 5.3 (b)의 세 개의 물체로부터 두 3D LF로 향하는 light ray는 위 조건을 만족하며, 각각 서로 다른 위치에 있다. 물체 A는 그림 5.3 (a)의 물체 A와 동일하며, 두 직선과 동일한 높이에 위치한다. 물체 A에서 두 3D LF로 입사하는 light ray는 모두 수직 방향으로 0° 의 입사 각도로 입사하였다. 물체 B는 물체 A와 비교하여 높은 위치에 있다. 이 경우 물체 B에서 $3DLF_B$로 입사하는 light ray는 φ_{B0}의 수직 입사 각도로 입사한 반면, $3DLF_A$로 입사하는 light ray는 φ_{A0}의 수직 입사 각도로 서로 다른 수직 입사 각도로 입사하였다. 동일한 물체에서 서로 다른 점으로 향하는 방향이 다르기 때문에 이처럼 다른 수직 입사 각도를 가진다. 마찬가지로 물체 C는 높이는 물체 B와 유사하지만 두 3D LF 사이에 위치하였으며, 두 3D LF에 입사하는 light ray의 수직 입사 각도는 각각 φ_{B1}, φ_{A1}로 달랐다. 두 수직 입사 각도의 차이는 물체 B의 경우보다 커진다.

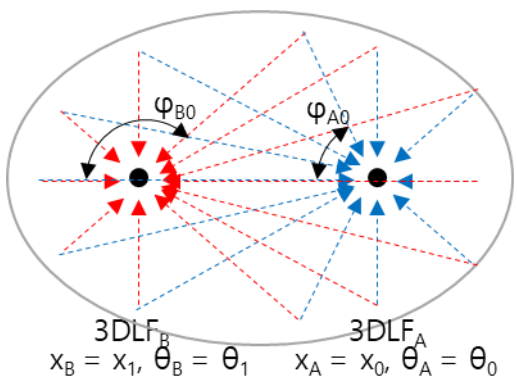
그림 5.3 (c)는 그림 5.3 (b)를 전체 light ray로 확장한다. 회색은 두 3D LF를 둘러쌓는 배경을 의미한다. 해당 장소에서 $3DLF_B$와 $3DLF_A$로 향하는 light ray는 빨간색 점선 화살표, 파란색 점선 화살표와 같다.



(a)



(b)



(c)

그림 5.3. (a) 두 개의 3D LF가 배치된 구조. (b) 서로 다른 물체로부터 입사하는 light ray. (b) 두 3D LF에서 epipolar 관계를 가지는 두 light ray 세트 쌍의 관계

$3DLF_B$에서 ($x_B=x_1$, $\theta_B=\theta_1$)와 $3DLF_A$에서 ($x_A=x_0$, $\theta_A=\theta_0$)의

조건을 만족하는 모든 light ray는 두 3D LF의 위치로 인해서 특정한 규칙을 가진다. 물체로부터 두 3D LF로 입사하는 light ray의 수직 입사 각도의 절대 값은 항상 $3DLF_B$로 향하는 light ray의 수직 입사 각도의 절대 값이 $3DLF_A$로 향하는 light ray의 수직 입사 각도의 절대 값보다 크다. 그리고 물체가 두 3D LF와 동일한 높이에 위치하는 경우에는 수직 입사 각도가 0° , ±180° 로 동일하다.

이처럼 그림 5.3 (a)의 $3DLF_B$ 상에서 $(x_B=x_1,\ \theta_B=\theta_1)$의 조건을 만족하는 모든 light ray와 $3DLF_A$ 상에서 $(x_A=x_0,\ \theta_A=\theta_0)$의 조건을 만족하는 모든 light ray는 epipolar geometry 관계를 가진다. 정확하게 동일한 물체, 배경으로부터 나오는 light ray를 공유하지만 수직 방향의 입사 각도가 다르게 된다.

그림 5.4은 epipolar geometry 관계를 가지는 light ray 세트를 픽셀 관점에서 살펴본 그림이다. 두 3D LF 내에서 각각의 light ray 세트는 x 축 상의 동일한 점에서 동일한 수평 입사 각도를 가진다. 따라서 해당 light ray 세트는 하나의 픽셀 column을 구성하게 된다. 그림 5.4의 왼쪽 픽셀 column은 $3DLF_A$에서 $(x_A=x_0,\ \theta_A=\theta_0)$의 조건을 만족하는 light ray 세트로 구성된 픽셀 column을 의미하며, 오른쪽 픽셀 column은 $3DLF_B$에서 $(x_B=x_1,\ \theta_B=\theta_1)$의 조건을 만족하는 light ray 세트로 구성된 픽셀 column을 나타낸다. 그림 5.4에서 볼 수 있듯이 두 픽셀 column은 유사한 픽셀 값을 가지고 있는 것을 알 수 있다. 하지만 유사한 픽셀의 위치가 다소 차이가 있는 것을 확인할 수 있다. 이는 앞서 그림 5.3에서 확인한 규칙과 유사하다. 동일한 물체로부터 입사한 light ray중 물체가 3D LF와 동일한 높이에 있음으로 인해서 수직 입사 각도가 입사 각도가 0° , ±180° 인 light ray는 두 3D LF의 픽셀 column에 동일한 높이에 위치한 것을 확인할 수 있다. 그 외의 물체로부터 입사하는 light ray는 항상 $3DLF_B$로 향하는 light ray의 수직 입사 각도의 절대 값이 $3DLF_A$로 향하는 light ray의 수직 입사 각도의 절대 값보다 컸으며, 그림 5.4에서 볼 수 있듯이 픽셀 column 상에서도 항상 $3DLF_B$의 픽셀 column의 위치는

절대값 측면에서 더 크다. φ가 0보다 큰 경우, 동일한 물체로부터 입사한 픽셀은 $3DLF_A$의 픽셀 column에서보다 $3DLF_B$의 픽셀 column에서 더 높은 위치에 있으며, φ가 0보다 작은 경우, 더 낮은 높이에 위치한다.

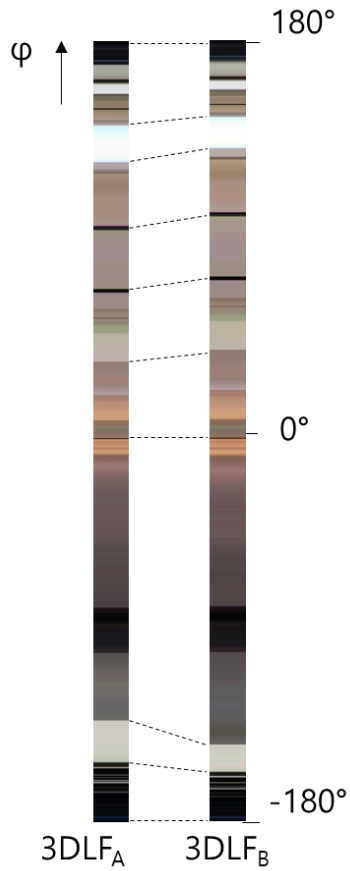
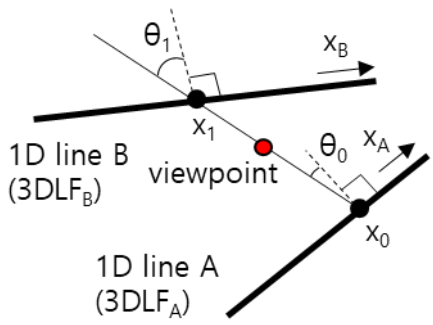


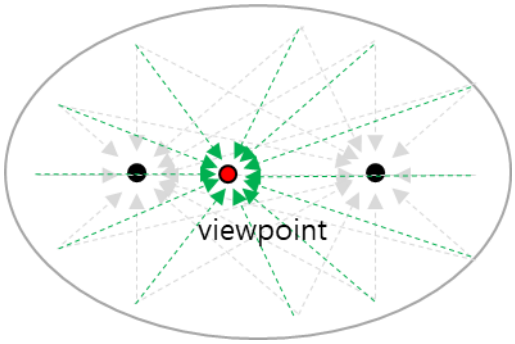
그림 5.4. Epipolar geometry 관계를 가지는 light ray 세트로 구성된 픽셀 column과 두 픽셀 column 간의 관계

5.3 Epipolar geometry 관계를 가지는 light ray 세트를 이용한 View Generation

본 논문은 3D LF stack 구조에서 3D LF의 vertical parallax를 표현하지 못하는 문제를 개선하기 위해 앞서 설명한 epipolar geometry 관계를 가지는 두 light ray 세트를 이용한다. 그림 5.5 (a)는 그림 5.3 (a)와 동일하게 가정한 구조에서 $x_B=x_1$인 점과 $x_A=x_0$인 점을 잇는 선 위에 임의의 viewpoint가 위치한 상황을 가정한다. 앞서 설명한 것처럼 $x_B=x_1$인 점과 $x_A=x_0$인 점을 각각 $\theta_B=\theta_1$의 수평 입사 각도와 $\theta_A=\theta_0$의 수평 입사 각도로 입사하는 light ray 세트가 epipolar geometry 관계를 가진다.



(a)



(b)

그림 5.5. 임의의 viewpoint에서의 light ray 세트 구성

임의의 viewpoint가 두 점을 잇는 선 상에 있게 됨으로써 해당 viewpoint에 동일한 방향으로 입사하는 light ray 세트는 앞서 설명한 epipolar geometry의 관계를 가지는 두 light ray 세트와 동일하게 epipolar geometry 관계를 가지게 된다. 그림 5.5 (b)는 그림 5.3 (b)에 임의의 viewpoint로 입사하는 light ray 세트를 나타낸 그림이며, 해당 light ray는 녹색 점선 화살표와 같다. 그림 5.3 (b)에서 확인할 것처럼 동일한 물체, 배경으로부터 나오는 light ray가 다른 수직 입사 각도로 입사하게 된다. 그림 5.6은 이를 픽셀 column의 관점에서 비교한다.

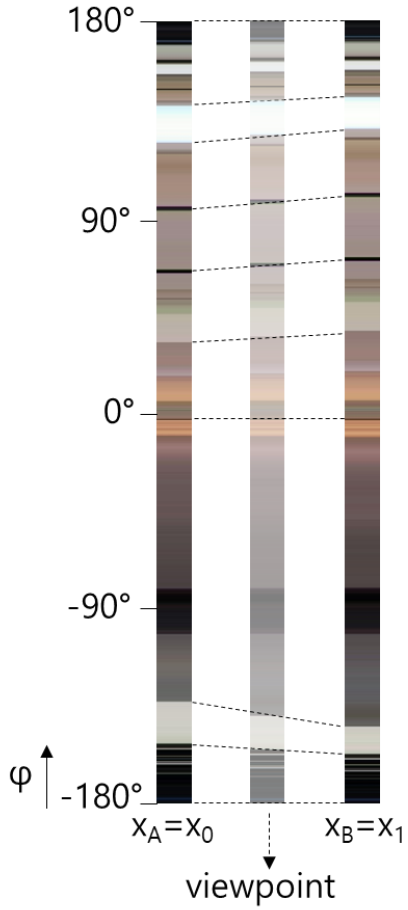
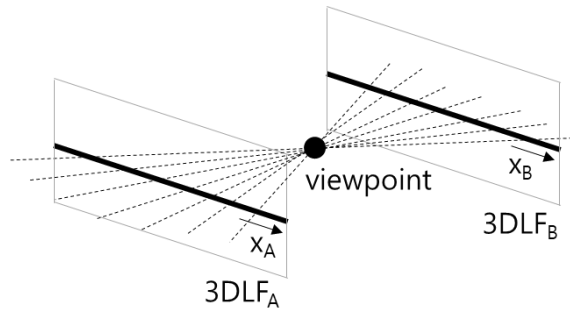


그림 5.6. Epipolar geometry 관계를 가지는 light ray 세트와 임의의 viewpoint에서의 light ray 세트로 구성된 픽셀 column

그림 5.6에서 좌, 우에 위치한 픽셀 column은 각각 3DLFA와 3DLFB에 속한 light ray를 픽셀 column의 관점에서 살펴본 그림이며, 이는 그림 5.4와 동일하다. 가운데 위치한 픽셀 column은 임의의 viewpoint로 향하는 light ray 세트로 구성된 픽셀 column을 의미한다. 수직 입사 각도가 0° , ±180° 인 픽셀은 모두 동일한 높이에 위치한다. 그 외 물체로부터 입사하는 light ray의 픽셀은 앞선 그림 5.4에서 살펴본 두 픽셀 column의 관계로 추정된 disparity의 중간에 위치하는 것을 확인할 수 있다. 그림 5.5 (a)의 viewpoint 위치에 따라, viewpoint가 $3DLF_A$에 가까운 방향으로 이동할 경우 그림 5.6의 임의의 viewpoint에 대한 픽셀 column은 $3DLF_A$의 픽셀 column에 가까운 방향으로, 반대로 viewpoint가 $3DLF_B$에 가까운 방향으로 이동할 경우 $3DLF_B$의 픽셀 column에 가까운 방향으로 바뀌게 된다.

이처럼 epipolar geometry 관계를 가지는 light ray 세트를 바탕으로 임의의 viewpoint에서의 light ray 세트를 구성할 수 있음을 확인하였다. 하지만 앞서 살펴본 과정은 하나의 light ray 세트를 구성하는 과정에 해당된다. 임의의 viewpoint에 대한 뷰를 구성하기 위해서는 하나의 light ray 세트가 아니라 뷰를 구성하는 모든 light ray 세트를 고려해야 한다. 즉, 다수의 픽셀 column이 쌓여 하나의 뷰를 구성하게 된다. 그림 5.7 (a)는 임의의 viewpoint에서의 뷰를 구성하는 과정의 예를 보여준다. 그림에는 두 개의 3D LF, $3DLF_A$, $3DLF_B$가 있으며, 임의의 viewpoint는 두 3D LF 사이에 위치한다. 이때 해당 viewpoint에서의 뷰를 구성하기 위해 필요한 light ray 세트를 정한다. 그림의 검은색 점선은 뷰를 구성하기 위한 light ray 중 일부를 예로 보여준다. 각각의 light ray 세트는 앞, 뒤로 배치된 3D LF를 통과하는 두 점과 각 점을 통과하는 수평 입사 각도로 정의될 수 있으며, 이를 통해 epipolar geometry 관계를 가지는 light ray 세트를 정의하고, 임의의 viewpoint에서의 light ray 세트를 추정하 수 있게 된다. 그리고 추정된 light ray 세트를 모두 이어줌으로써 최종 뷰를 구성한다.



(a)

(b)

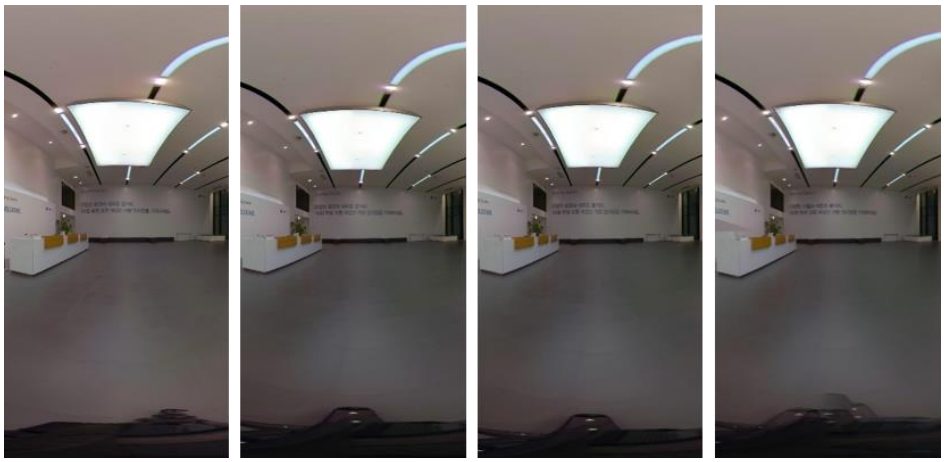
그림 5.7. Epipolar geometry 관계를 가지는 light ray 세트를 이용한 임의의 viewpoint에서의 뷰 구성

그림 5.7 (b)는 추정한 light ray 세트의 픽셀 column을 모두 연결함으로써 구성한 뷰 이다. 해당 뷰는 수직 각도로 −180° ~ 180° 의 범위를 가진다. 따라서 정면의 뷰와 더불어 후면에 해당되는 뷰를 모두 포함한다. 결과적으로 그림 5.7 (a)의 임의의 viewpoint를 기준으로 앞, 뒤 방향의 뷰를 구성할 수 있게 된다. 임의의 viewpoint의 위치에 따라 그림 5.7 (a)에서 필요한 light ray 세트는 달라진다.

5.4 Epipolar geometry 관계의 light ray 세트 기반의 뷰 구성 결과 비교

이번 섹션에서는 앞서 설명한 epipolar geometry 관계를 가지는 light ray 세트 기반의 뷰 구성 결과를 기존의 단일 3D LF 기반의 뷰 구성 방법과 비교한다. 비교 대상은 3D LF stack 구조에서 임의의 viewpoint가 정해졌을 때 전방에 위치한 하나의 3D LF를 사용한 방법이며, constant depth를 사용한 보정을 포함한 결과와 포함하지 않는 결과를 모두 비교 대상으로 한다.

그림 5.8과 그림 5.9는 두 개의 샘플을 대상으로 3D LF 기반의 뷰 구성 결과를 비교한다. 첫 번째 샘플 그림 5.8의 결과에서 그림 5.8 (a)는 원본 이미지에 해당되며, 그림 5.8 (b)는 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려한 보정을 포함하지 않은 경우, 그림 5.8 (c)는 단일 3D LF를 사용한 뷰 구성에서 constant depth 기반의 뷰 보정을 포함한 결과, 그리고 그림 5.8 (d)는 본 논문에서 제안하는 epipolar geometry 관계의 light ray 세트를 고려한 뷰 구성 결과를 보여준다. 그림 5.8 (b), 그림 5.8 (c), 그림 5.8 (d)의 3D LF 기반의 뷰 구성 결과를 살펴보면 전반적으로 깔끔한 뷰 구성 결과를 확인할 수 있다. 이는 LF 기반의 접근 방법의 장점으로 많은 양의 데이터를 기반으로 별도의 픽셀 합성 없이 뷰를 구성하기 때문에 가능하다.



(a) (b) (c) (d)

그림 5.8. 3D LF 기반의 뷰 구성 결과 비교 (샘플 1) (a) 원본, (b) 단일 3D LF 사용 (without constant depth 보정), (c) 단일 3D LF 사용 (with constant depth 보정), (d) Epipolar 관계를 고려한 뷰 구성

(a) (b) (c) (d)

그림 5.9. 3D LF 기반의 뷰 구성 결과 비교 (샘플 2) (a) 원본, (b) 단일 3D LF 사용 (without constant depth 보정), (c) 단일 3D LF 사용 (with constant depth 보정), (d) Epipolar 관계를 고려한 뷰 구성

결과적으로 그림 5.8 (b), 그림 5.8 (c), 그림 5.8 (d)의 3D LF 기반의 뷰 구성 결과는 그림 5.8 (a)의 원본 이미지와 유사한 것을 확인할 수 있다. 이는 또 다른 샘플에서도 동일하다. 그림 5.9 (b), 그림 5.9 (c), 그림 5.9 (d)는 각각 단일 3D LF 기반의 뷰 구성 결과에서 constant depth를 고려한 뷰 보정을 포함하지 않은 결과, 단일 3D LF 기반의 뷰 구성에서 constant depth를 고려하여 뷰를 보정한 결과, 그리고 epipolar geometry 관계를 고려하여 뷰를 구성한 결과를 보여주며, 샘플 1의 결과와 마찬가지로 visual artifact가 눈에 띄지 않으면서 그림 5.9 (a)의 원본 결과와 유사한 것을 확인할 수 있다.

실질적으로 실험 결과에서 살펴봐야 할 부분은 임의의 viewpoint 이동에 따른 뷰 변화가 잘 반영되는가와 임의의 viewpoint가 3D LF를 통과하고 사용하는 3D LF가 바뀌는 순간의 뷰 변화를 살펴보는 것이며, 이를 살펴볼 필요가 있다.

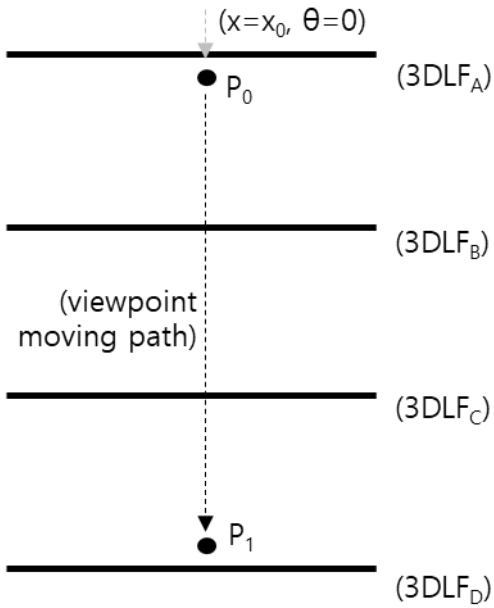
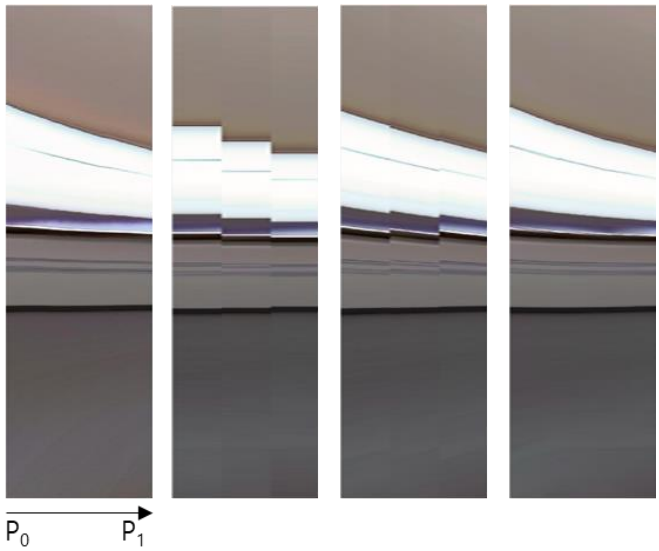


그림 5.10. 임의의 viewpoint가 이동하는 경로

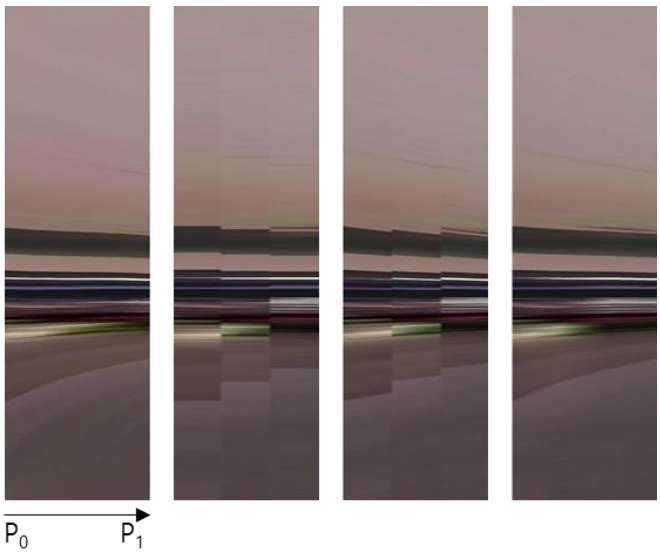
그림 5.10 는 viewpoint 이동에 따른 뷰 변화를 살펴보기 위해 가정한 뷰의 이동 경로를 보여준다. 가정하는 구조는 총 네 개의 3D LF로 구성된다. 가장 앞에서부터 $3DLF_A$, $3DLF_B$, $3DLF_C$, $3DLF_D$가 차례로 배치되어 있다. 임의의 viewpoint는 $3DLF_A$의 뒤에 P_0의 점에서 검은색 점선 화살표를 따라 뒤로 이동하여 $3DLF_D$ 앞에 인접한 P_1까지 이동하게 된다. 이 때 뷰 전체를 대상으로 비교하는 것이 아닌 픽셀 column의 변화를 통해 뷰 변화의 자연스러움을 비교하며, 이 때 대상이 되는 픽셀 column은 임의의 viewpoint에 ($x=x_0$, $\theta=0$)의 조건으로 입사하는 light ray 세트로 구성되는 픽셀 column에 해당된다. $x=x_0$ 인 점은 각각의 3D LF의 가운데 점을 의미하며, light ray가 $\theta=0$임은 3D LF에 수직으로 입사하는 light ray를 의미한다. 그림의 회색 점선 화살표와 같다.

그림 5.11과 그림 5.12 임의의 viewpoint가 그림 5.10 에서 가정한 경로를 따라 움직일 때, ($x=x_0$, $\theta=0$)의 조건으로 입사하는 light ray 세트로 구성되는 픽셀 column의 변화를 비교한다. 그림 5.11과 그림 5.12은 각각 그림 5.8과 그림 5.9의 샘플 1, 샘플 2에 대한 결과를 보여준다. 샘플 1에 대한 결과를 보여주는 그림 5.11에서 그림 5.11 (a)는 원본 이미지, 그림 5.11 (b)는 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려한 뷰 보정을 포함하지 않는 결과, 그림 5.11 (c)는 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려하여 뷰를 보정한 결과, 그리고 그림 5.11 (d)는 epipolar geometry를 고려하여 뷰를 구성한 제안 방법에 대한 결과를 보여준다. 각 이미지는 픽셀 viewpoint가 이동함에 따른 픽셀 column의 변화를 보여주고 있으며, 가장 왼쪽 픽셀 column은 viewpoint가 P_0에 위치한 경우이고, 가장 오른쪽 픽셀 column은 viewpoint가 P_1에 위치한 경우를 의미한다. 그림 5.11 (a)의 원본 이미지를 보면 viewpoint가 P_0에서 P_1으로 이동함에 따라 픽셀 column이 자연스럽게 변하는 것을 확인할 수 있다.



(a) (b) (c) (d)

그림 5.11. Viewpoint 이동에 따른 픽셀 column 변화 비교 (샘플 1)
(a) 원본, (b) 단일 3D LF 사용 (without constant depth 보정),
(c) 단일 3D LF 사용 (with constant depth 보정), (d) Epipolar
관계를 고려한 뷰 구성



(a) (b) (c) (d)

그림 5.12. Viewpoint 이동에 따른 픽셀 column 변화 비교 (샘플 2)
(a) 원본, (b) 단일 3D LF 사용 (without constant depth 보정),
(c) 단일 3D LF 사용 (with constant depth 보정), (d) Epipolar
관계를 고려한 뷰 구성

그림 5.11 (b)의 결과를 보면 픽셀 column의 변화가 크게 세 개의 파트로 분리되는 것을 확인할 수 있다. 이는 viewpoint가 이동함에 따라 처음 1/3 지점에서는 $3DLF_A$를 사용해서 뷰를 구성하고, 가운데 1/3 지점에서는 $3DLF_B$를 사용, 마지막 1/3 지점에서는 $3DLF_C$를 사용하기 때문이며, 경계가 두드러지게 나타남은 3D LF가 바뀌는 점에서 큰 뷰의 변화를 확인할 수 있음을 의미한다. 그리고 그림 5.11 (b)의 결과는 constant depth를 고려하여 보정을 포함하지 않기 때문에 동일한 3D LF를 사용하는 구간 내에서도 수직 방향의 뷰 변화가 전혀 반영되지 못한다. 그림 5.10의 viewpoint의 이동 경로에서 보는 것처럼 viewpoint가 뒤로 이동하기 때문에 물체 또는 배경은 점점 작아지는 형태로 뷰가 구성되어야 하는데, 이와 같은 변화가 전혀 드러나지 않는다. 이는 그림 5.11 (c)와 대조적이다.

그림 5.11 (c)는 그림 5.11 (b)의 경우와 동일하게, viewpoint의 처음 1/3 지점에서는 $3DLF_A$를 사용, 가운데 1/3 지점에서는 $3DLF_B$를 사용, 마지막 1/3 지점에서는 $3DLF_C$를 사용한다. 하지만 동일한 3D LF를 사용하는 구간 내에서 constant depth를 고려해서 뷰를 보정하기 때문에 viewpoint가 뒤로 이동함에 따른 부의 수직 방향으로의 변화가 반영된 것을 확인할 수 있다. 하지만 3D LF가 바뀌는 구간에서는 여전히 부자연스러운 변화를 확인할 수 있다.

마지막으로 그림 5.10 (d)의 경우를 살펴보면 epipolar geometry를 고려한 뷰 구성에서는 viewpoint 이동에 따른 수직 방향의 뷰 변화와 더불어 viewpoint가 3D LF를 넘어가는 구간에서도 부자연스러운 변화 없이 뷰가 구성되는 것을 확인할 수 있다. Epipolar geometry 관계를 고려하는 방법에서는 viewpoint가 이동함에 따라 처음 1/3 지점에서는 $3DLF_A$와 $3DLF_B$를 사용, 가운데 1/3 지점에서는 $3DLF_B$와 $3DLF_C$를 사용, 마지막 1/3 지점에서는 $3DLF_C$와 $3DLF_D$를 사용해서 뷰를 구성한다. 단일 3D LF를 사용하는 방법과 달리 viewpoint 가 이동하고 3D LF를 넘어가는 순간에도 하나의 공통의 3D LF를 공유하기 때문에 뷰의 급격히 바뀌는 문제를 해결할 수 있었을

것으로 예상할 수 있다.

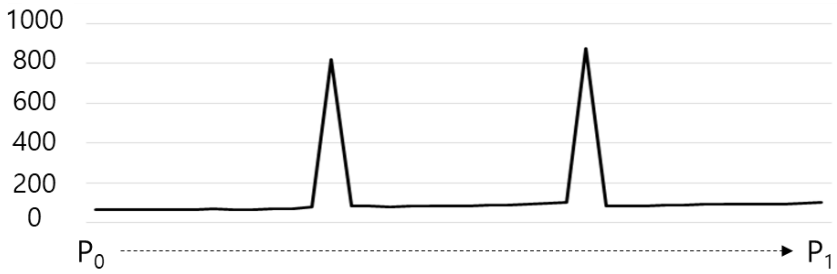
그림 5.12는 샘플 2에서 viewpoint 이동에 따른 픽셀 column의 변화를 비교한다. 그림 5.12 (a), 그림 5.12 (b), 그림 5.12 (c), 그림 5.12 (d)는 각각 원본에서의 픽셀 column 변화, 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려한 뷰 보정을 포함하지 않은 결과, 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려하여 뷰를 보정한 결과, 그리고 앞, 뒤로 배치된 두 3D LF의 epipolar geometry 관계를 고려하여 뷰를 구성한 제안 방법의 결과를 보여준다. 샘플 1의 결과와 동일하게 각각의 이미지에서 가장 왼쪽 픽셀 column은 viewpoint가 P_0에 위치한 경우를 나타내고, 가장 오른쪽 픽셀 column은 viewpoint가 P_1에 위치한 경우를 나타낸다.

그림 5.12 (b)의 경우 viewpoint가 이동하고 사용하는 3D LF가 바뀜에 따른 뷰의 급격한 변화 문제가 두드러짐과 동시에 viewpoint 이동에 따른 뷰의 수직 방향으로의 변화가 반영되지 못한 결과를 확인할 수 있다. 그림 5.12 (c)의 경우 constant depth를 바탕으로 viewpoint 이동에 따른 수직 방향으로의 뷰 변화가 잘 반영된 것을 확인할 수 있지만 3D LF가 바뀌는 순간의 뷰의 급격한 변화 문제는 여전히 남아있음을 확인할 수 있으며, 그림 5.12 (d)의 epipolar geometry 관계를 고려한 제안 방법의 결과에서는 viewpoint 이동에 따른 뷰의 수직 방향의 변화와 더불어 3D LF를 통과하는 순간의 뷰의 부자연스러운 문제를 모두 제거한 것을 확인할 수 있다. 그림 5.12 (d)의 결과는 그림 5.12 (a)의 원본과 유사한 것을 확인할 수 있다.

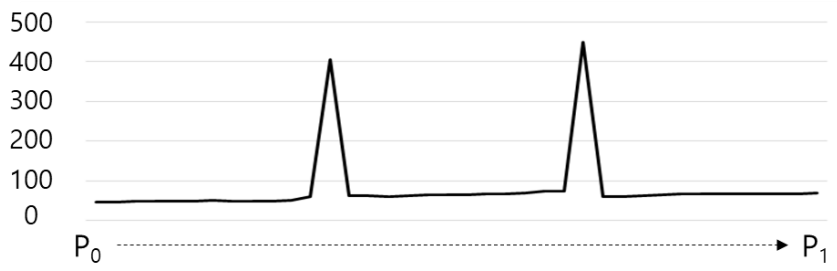
제안한 epipolar geometry 관계를 고려한 뷰 구성 방법을 객관적 지표를 통해 비교한다. 이를 비교하기 위해서 식 (5.2)와 같이 프레임 단위 픽셀 difference를 사용한다.

$$diff(p) = mse(frame(p), frame(p-1)) \qquad (5.2)$$

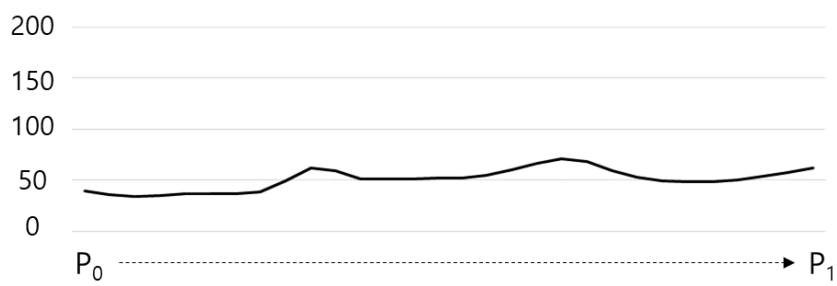
(5.2)에서 frame(p)는 현재 viewpoint에서 구성된 뷰, frame(p-1)는 이전 viewpoint에서 구성된 뷰를 의미한다. 프레임 단위의 픽셀 difference는 구성된 두 뷰의 차이를 mse로 계산한다. 임의의 viewpoint 이동에 따라 뷰가 자연스럽게 바뀌는 경우 mse 값은 큰 변화 없이 자연스럽게 바뀔 것으로 예상할 수 있다.



(a)



(b)



(c)

그림 5.13. 임의의 viewpoint 이동에 따른 프레임 단위 difference 비교

그림 5.13은 임의의 viewpoint가 이동함에 따른 프레임 단위 픽셀 difference를 비교한 그래프이다. 그림 5.13 (a)는 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려한 뷰 보정을 포함하지 않은 결과이다. 그림 5.13 (b)는 단일 3D LF를 사용한 뷰 구성에서 constant depth를 고려하여 뷰를 보정한 결과이고, 그림 5.13 (c)는 논문에서 제안한 epipolar geometry 관계를 고려하여 뷰를 구성한 결과의 프레임 단위 difference이다. 그래프는 왼쪽에서 오른쪽으로 이동함에 따라 viewpoint는 그림 5.10의 P_0의 위치에서 P_1의 위치로 이동하는 상황을 의미한다.

먼저 그림 5.13 (a), 그림 5.13 (b)의 그래프에서 두 점에서 프레임 단위 difference가 크게 증가하는 상황을 확인할 수 있는데, 이는 viewpoint가 이동하면서 뷰 구성에 사용하는 3D LF가 바뀌는 수간으로 예측할 수 있다. 해당 부분은 그림 5.11 (b), 그림 5.11 (c)와 그림 5.12 (b), 그림 5.12 (c)에서 픽셀 column이 부자연스럽게 연결된 부분에 해당된다. 단, 프레임 단위 픽셀 difference의 수치를 보면 constant depth를 사용해서 뷰를 보정한 그림 5.13 (b)의 그래프가 1/2 정도로 작다. 이는 정확하지는 않지만 constant depth를 통해 뷰를 실제에 가깝도록 보정을 해줌으로써 3D LF가 바뀌는 순간 바뀐 3D LF로 구성한 실제와 거의 유사한 뷰와 차이를 좁혀 놓았기 때문이다.

그림 5.13 (c)의 그래프는 프레임 단위 픽셀 difference는 그림 5.13 (a), 그림 5.13 (b)의 그래프와 달리 크게 프레임 단위 픽셀 difference가 증가하는 부분 없는 것을 확인할 수 있다. 즉, viewpoint가 이동함에 따라 뷰가 급격히 변하는 부분 없이 자연스럽게 바뀌었다는 것을 확인할 수 있다. Epipolar geometry 관계를 이용한 뷰 구성에서 프레임 단위 픽셀 difference는 약 50 정도의 값으로 유지된 것을 확인할 수 있다.

Epipolar geometry 관계를 이용한 뷰 구성한 결과의 그래프에서 확인할 수 있는 또 다른 정보는 3D LF가 유지되는 구간에서 프레임

단위의 픽셀 difference가 오목한 그래프를 나타내는 특정 패턴을 보이는 것이다. 그림 5.13 (c)의 그래프에서 총 세 개의 오목한 그래프를 확인할 수 있다. 그래프의 처음 1/3 지점은 viewpoint가 그림 5.10에서 $3DLF_A$와 $3DLF_B$ 사이에 위치하는 경우, 가운데 1/3 지점은 $3DLF_B$와 $3DLF_C$ 사이에 위치하는 경우, 마지막 1/3 지점에서는 $3DLF_C$와 $3DLF_D$ 사이에 위치하는 경우에 해당된다. 각각의 그래프에서 양쪽 끝 점의 프레임 단위 픽셀 difference가 다소 크고, 가운데 점의 값은 다소 작다. 이는 epipolar geometry 관계를 고려하여 뷰를 구성하는 방법에서 viewpoint의 위치에 따라 앞, 뒤의 3D LF 정보를 가중치를 고려하여 합하게 되는데, viewpoint가 두 3D LF 가운데에 위치하는 양 쪽의 데이터를 전반적으로 고려하기 때문에 viewpoint 이동에 따른 뷰 변화가 크지 않다. 반면, viewpoint가 두 3D LF 중 하나에 가까워질수록 해당 3D LF의 비중을 많이 포함하게 되고, 이에 따른 뷰 변화가 비교적 커지게 된다. 하지만 이 정도의 뷰 변화는 사용자의 입장에서 두드러지지 않아, 자연스러운 뷰 변화를 경험할 수 있다.

제 6 장 제안 시스템 구현

앞서 본 논문에서는 넓은 공간을 자유롭게 이동하기 위한 새로운 LF 기반의 시스템을 제시하였다. 기존의 하나의 3D LF 또는 4D LF를 구성하고 이를 바탕으로 뷰를 구성하는 방식이 아니라 다수의 3D LF를 차례로 배치하는 3D LF stack의 구조이다. 또한 이 3D LF stack의 구조를 수직 방향의 두 방향으로 배치시키는 방식으로 구조를 구성한다.

단, 이 시스템을 위해서는 두 가지 해결 과제가 남아 있었으며, 본 논문에서는 physical connection과 non-physical connection 그리고 hybrid connection 방법을 제시하여 서로 다른 3D LF를 연결하는 문제의 해결 방안을 제시하였으며, epipolar geometry 관계를 가지는 light ray 세트의 관계를 바탕으로 3D LF stack 구조를 위한 뷰 구성 방법 제시하였다.

6장에서는 앞서 제시한 두 방안을 포함한 전체 시스템의 구현과 구현된 시스템에서 임의의 viewpoint에서 구성된 뷰를 살펴봄으로써 본 논문의 제안 시스템의 결과를 살펴본다. 총 두 가지 형태의 방법으로 시스템을 구성한다. 하나는 일반적인 카메라를 사용하는 환경에서 카메라 슬라이더를 사용한 시스템 구성, 다른 하나는 360도 카메라와 dolly 장비를 사용한 시스템 구성이다. 각 시스템의 구성과 결과를 순서대로 다룬다.

6.1 일반 카메라 + 슬라이더를 이용한 구성

본 논문에서 제안하는 격자 모양의 구조에 대한 시스템을 구성하기 위해 일반 카메라와 슬라이더를 이용한다. 일반 카메라는 Gorpo Hero 액션 카메라 [44]를 사용하며, 카메라 슬라이더 장비는 코노바 K3 카메라 슬라이더 [45]를 사용한다. 코노바 K3 슬라이더는 총 1m의 거리를 움직일 수 있다.

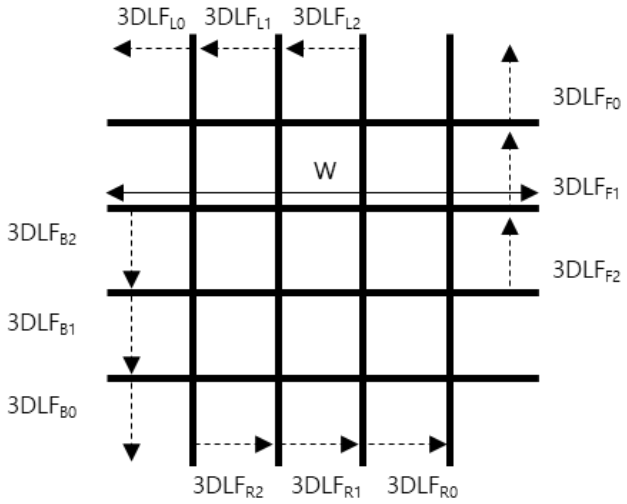


그림 6.1. 일반 카메라와 슬라이더를 이용한 구성

그림 6.1는 일반 카메라와 슬라이더를 이용한 환경 구성에 있어서 구성한 3D LF의 구조를 나타낸다. 그림에서 굵은 검은 선은 3D LF를 나타낸다. 총 네 개의 방향의 세 개씩의 3D LF가 있으며, 각각의 3D LF를 구성하는 길이 W는 슬라이더가 움직일 수 있는 제한으로 인해 95cm로 정한다. 슬라이더의 제원상 1m의 이동이 가능하지만 슬라이더에 장착된 카메라 렌즈가 움직일 수 있는 거리가 제한된다.

일반 카메라는 한 방향으로 입사하는 light ray만 획득할 수 있다. 따라서 양 방향의 light ray가 모두 필요한 경우 두 차례 light ray 획득 과정을 진행한다. 예를 들어 3DLF$_{F1}$에 해당되는 직선과 3DLF$_{B2}$에 해당되는 직선은 동일한 직선 상에 위치하지만 슬라이더를 통해 두 차례 스캐닝 과정을 거친다. 동일하게 3DLF$_{F2}$와 3DLF$_{B1}$, 3DLF$_{L2}$와 3DLF$_{R1}$, 3DLF$_{L2}$와 3DLF$_{R1}$은 동일한 직선을 공유하며 light ray를 획득하게 된다. 일반 카메라를 사용하는 경우는 앞선 3D LF 연결 방법에서 살펴본 바와 같이 확장된 사각형 구조의 LFU를 이용한 non-physical connection이 불가피하며, 이를 위해 확장된 사각형 LFU를 구성할 수

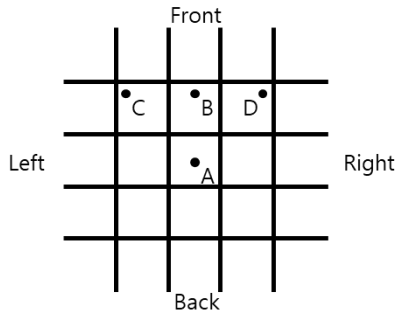
있도록 구조를 구성한다. 그 결과 그림 6.1의 구성 구조에는 총 9개의 확장된 사각형 모양의 LFU가 생긴다. 그리고 해당 범위에 대해서 자유로운 시점 변화를 지원하며, 임의의 viewpoint에 대한 수평 방향의 360도 뷰를 구성하게 된다. 단, 일반 카메라의 수직 방향의 FOV 한계로 인해 수직 방향의 완전한 360도 뷰를 지원하지는 못한다.

6.2 일반 카메라 + 슬라이더를 이용한 구조 결과

6.2.1 자유시점 변화에 대한 뷰 구성 결과 비교

앞선 시스템 구성 환경을 바탕으로 구성한 결과를 살펴본다. 그림 6.2는 A 장소에 대해 제안한 시스템을 구성하고 임의의 viewpoint에서의 뷰를 구성한 결과이다. 그림 6.2 (a)는 해당 장소에 구성된 구조와 임의의 viewpoint 위치를 나타낸다. 그림 6.2 (b)는 임의의 viewpoint가 A에 위치한 경우 구성된 뷰를 나타낸다. 그림 6.2 (c), 그림 6.2 (d), 그림 6.2 (e)는 각각 임의의 viewpoint가 B, C, D에 위치한 경우 구성된 뷰를 나타낸다.

그림 6.2 (b) ~ 그림 6.2 (e)의 구성된 뷰를 보면 수평 방향으로는 360도 방향의 모든 뷰를 구성하는 것을 확인할 수 있다. 반면, 위, 아래로는 충분히 넓은 뷰를 구성하지 못하며 완전한 360도 뷰를 구성하지는 못한다. 각각의 구성된 뷰는 네 방향의 뷰로 나눌 수 있다. 가장 왼쪽부터 그림 6.2 (a)의 left, front, right, back 방향의 뷰를 의미한다. 그림 6.2 (b)와 그림 6.2 (c)를 비교하면 임의의 viewpoint는 A에서 B로 이동한 것을 확인할 수 있다. 이는 front 방향으로 이동한 것에 해당되며, 그림 6.2 (b)의 front 방향에 뷰에 비해 그림 6.2 (c)의 front 방향의 뷰에서 물체가 가까워지는 것으르 확인할 수 있다. 나머지 방향의 뷰에서도 해당 viewpoint 이동에 따른 뷰 변화를 확인할 수 있다.



(a)

(b)

(c)

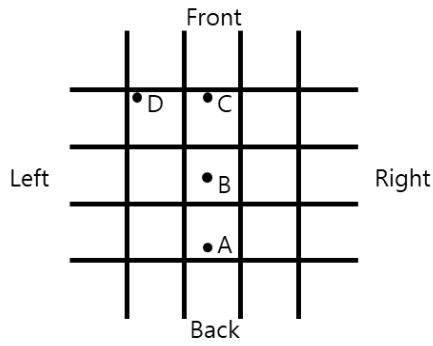
(d)

(e)

그림 6.2. 임의의 viewpoint에서의 뷰 구성 결과 (장소 A), (a) 임의의 viewpoint 위치, (b) Viewpoint A, (c) Viewpoint B, (d) Viewpoint C, (e) Viewpoint D

그림 6.2 (d)는 그림 6.2 (c)에 비해 viewpoint가 B에서 C로 이동하였으며 이는 left 방향에 가깝게 이동한 것이며, 그림 6.2 (e)는 그림 6.2 (c)에 비해 viewpoint가 D로 이동함으로써 right 방향으로 이동하였다. 그림 6.2 (c), 그림 6.2 (d), 그림 6.2 (e)의 결과를 살펴보면, front 방향의 뷰에서 녹색 테이블과 의자 사이의 parallax가 자연스럽게 반영되는 것을 확인할 수 있다. 세 구성 결과의 right 방향의 뷰에서는 그림 6.2 (d), 그림 6.2 (c), 그림 6.2 (e)의 순서도 점차적으로 다가가는 것과 같은 뷰 변화를 확인할 수 있으며, 반대로 left 방향의 뷰에서는 점차적으로 멀어지는 듯한 뷰 변화를 확인할 수 있다. 즉, 임의의 viewpoint의 이동에 따른 뷰 변화가 자연스럽게 반영되고 있음을 의미한다.

그림 6.3은 장소 B에 대해서 시스템을 구성하고 임의의 viewpoint에서의 뷰를 구성한 결과를 보여준다. 그림 6.3 (a)는 임의의 viewpoint의 위치를 보여준다. 그림 6.3 (b), 그림 6.3 (c), 그림 6.3 (d), 그림 6.3 (e)는 각각 viewpoint가 A, B, C, D의 위치에 있는 경우의 구성된 뷰를 보여준다. 각각의 뷰는 네 개의 뷰로 구성되며, 가장 왼쪽부터 left, front, right, back 방향의 뷰를 나타낸다. 그림 6.3 (b), 그림 6.3 (c), 그림 6.3 (d)의 결과는 임의의 viewpoint가 A의 위치에서 B를 거쳐 C로 이동하는 과정을 보여준다. 즉, viewpoint가 back 방향에서 front 방향으로 점차적으로 이동한다. 구성된 뷰에서 front 방향의 뷰를 보면 viewpoint가 뒤에서 앞으로 이동함에 따라 물체가 점점 다가가는 뷰 변화를 관찰할 수 있다. Right 방향의 뷰에서 물체는 점점 왼쪽에서 오른쪽으로 이동하고, left 방향의 뷰에서 물체는 오른쪽에서 왼쪽으로 이동하는 것을 확인할 수 있다. 그림 6.3 (e)의 viewpoint D에서의 뷰는 그림 6.3 (d)의 viewpoint C에서의 구성된 뷰에 비해 왼쪽 방향으로 이동한 위치이며, 그림 6.3 (e)의 왼쪽 방향에 해당되는 뷰는 그림 6.3 (d)의 왼쪽 방향에 해당되는 뷰에 비해서 물체에 다가간 듯한 뷰 변화를 관찰할 수 있다.



(a)

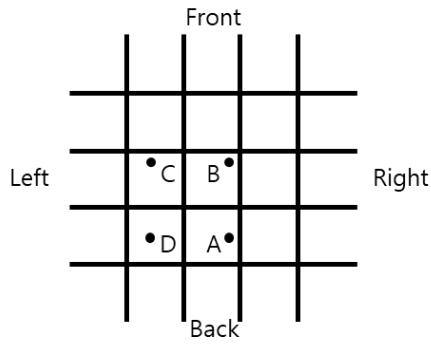
(b)

(c)

(d)

(e)

그림 6.3. 임의의 viewpoint에서의 뷰 구성 결과 (장소 B), (a) 임의의 viewpoint 위치, (b) Viewpoint A, (c) Viewpoint B, (d) Viewpoint C, (e) Viewpoint D



(a)

(b)

(c)

(d)

(e)

그림 6.4. 임의의 viewpoint에서의 뷰 구성 결과 (장소 C), (a) 임의의 viewpoint 위치, (b) Viewpoint A, (c) Viewpoint B, (d) Viewpoint C, (e) Viewpoint D

그림 6.4는 장소 C에 대한 시스템 구성 및 임의의 viewpoint에 대한 뷰 구성 결과이다. 그림 6.4 (a)는 임의의 viewpoint의 위치를 보여준다. 그림 6.4 (b), 그림 6.4 (c), 그림 6.4 (d), 그림 6.4 (e)는 각각 viewpoint가 A, B, C, D에 위치했을 때의 구성된 뷰를 보여준다. 앞선 다른 결과와 마찬가지로 각 뷰는 네 방향의 뷰로 구성되어 있으며, left, front, right back 방향의 뷰에 해당된다. 그림 6.4 (b) 그림 6.4 (c)는 각각 viewpoint가 A, B에 위치한 경우이며, viewpoint A에서 viewpoint B로 left 방향을 기준으로 오른쪽으로 이동함에 따라 left 방향의 뷰의 책상이 왼쪽으로 이동하는 것을 확인할 수 있으며, 반면 right 방향의 책상은 오른쪽으로 이동하는 뷰 변환이 잘 반영되는 것을 확인할 수 있다. 그림 6.4 (d)는 viewpoint가 C에 위치한 경우이며, 그림 6.4 (c)의 결과에 비해 left 방향으로 이동하였다. 그리고 left 방향의 책상이 가까워지고, right 방향의 책상이 멀어지는 뷰 변화를 확인할 수 있다. 그림 6.4 (e)는 view point가 D에 위치한 경우이며, 그림 6.4 (d)의 viewpoint가 C에 위치한 경우와 비교하여 back 방향으로 이동한 결과이다.

세 장소를 대상으로 시스템을 구성하고 임의의 viewpoint의 위치에 따른 뷰 구성 결과를 통해 임의의 viewpoint의 변화에 따른 뷰 변화가 자연스럽게 반영되는 것을 확인할 수 있다. 확장된 사각형 모양의 LFU 구조에서 non-physical connection 기반의 3D LF 연결을 통해 360도 방향의 뷰는 빈 공간 없이 모두 잘 채워지는 것을 확인할 수 있었으며, 격자 구조를 자유롭게 이동하면서 자유 시점 변환 시스템을 체험할 수 있었다.

6.2.2 Blending을 이용한 3D LF 연결 부분 보정

제안 구조는 단일 3D LF를 사용하는 구조와 달리 서로 다른 3D LF를 연결하여 뷰를 구성하는데, 이론적으로 두 3D LF가 자연스럽게

연결될 수 있다고 하더라도 구현의 관점에서 뷰가 다소 자연스럽게 연결되지 못하는 문제가 생기는 문제가 있었다. 이를 개선하기 위해 두 뷰를 연결함에 있어서 뷰가 오버랩 될 수 있도록 shared light로 정의되는 영역보다 넓게 범위를 할당하고 뷰를 구성한 뒤, 오버랩되는 부분을 blending을 통해서 개선하는 방법을 적용한다. 그림 6.5은 blending을 적용하기 전, 후의 3D LF 연결을 통해 뷰를 구성한 결과를 비교한다. 그림 6.5 (a)는 blending을 포함하지 않는 뷰 구성 결과, 그림 6.5 (b)는 blending을 포함한 뷰 구성 결과를 보여준다. 그림 6.5 (a)에서 두 이미지가 연결되는 부분에서 뚜렷한 경계선이 확인된다. 특히, 이 경우 카메라가 배치된 방향에 따라 빛 노출 정도의 차이가 달라져 동일한 물체의 색감에서 차이가 발생하고, 이로 인해 경계가 더욱 뚜렷해 보인다. 그림 6.5 (b)에서는 그림 6.5 (a)에 비해서 두 3D LF가 연결되는 부분의 경계가 흐릿해진다. 그림 6.5 (a)에 비해 비교적 두 뷰는 자연스럽게 연결되는 것을 확인할 수 있지만 ghost, blur 등의 에러가 포함되는 문제가 있다.

(a) (b)

그림 6.5. Blending을 포함하는 3D LF 연결을 통해 뷰를 자연스럽게 연결, (a) blending 반영 전, (b) 반영 후

6.2.3 제안 구조와 원 구조 3D LF의 구성 뷰 비교

일반 카메라와 슬라이더를 통해서 구성한 결과를 원 구조의 3D LF를 사용한 결과와 비교한다. 원 구조의 3D LF는 단일 구조를 바탕으로 뷰를 구성한다. 따라서 실제와 가까운 뷰를 구성하고 특히 서로 다른 3D LF를 연결함에 따른 뷰의 연결 문제가 전혀 발생하지 않는다.

그림 6.6은 동일한 viewpoint에 대해 원 구조의 3D LF를 이용해서 구성한 뷰와 제안하는 방법으로 구성한 뷰를 비교한다. 그리고 해당 뷰를 객관적 평가 지표로 비교한다. 총 세 가지 평가 지표를 사용하며, 각각 PSNR, SSIM [46], FSIM [47]이다. PSNR은 픽셀 단위로 이미지를 비교한다. SSIM은 이미지의 구조적 유사성을 비교하며, FSIM은 이미지에서 계산한 phased congruency와 gradient magnitude 정보를 비교한다. 세 가지 평가 지표는 모두 reference 이미지를 사용한다. 원 구조의 3D LF를 구성하기 위해 획득하는 이미지의 위치와 LFU 구조의 구성을 위해 획득하는 이미지의 위치가 다르고, 해당 구조에서 임의의 viewpoint를 가능한 동일한 위치로 정한다고 하더라도 정확하게 동일한 viewpoint를 구성하는 것은 어려움이 있다. 또한 각각의 구조에서 가정한 임의의 viewpoint에 대해서 정확하게 동일한 원본 이미지를 획득하는 것은 어려운 문제이다. 가능한 유사한 viewpoint에 대한 뷰를 기준으로 두 구조의 결과를 비교한다.

표 6.1는 그림 6.6의 3개 샘플 이미지에 대해서 각 구조의 결과에 대한 PSNR, SSIM, FSIM을 평가한다. 세 가지 지표는 모두 값이 클수록 reference 이미지와 유사함을 의미한다. PSNR 결과를 먼저 비교하면 세 가지 샘플 이미지에서 모두 원 구조를 가정한 3D LF의 PSNR이 더 큰 것을 확인할 수 있다. 원 구조의 3D LF 결과가 더 실제 이미지와 유사함을 의미한다. 하지만 0.5dB ~ 1.0dB 정도의 차이로 큰 차이를 보이지는 않는다. SSIM 결과 또한 원 구조의 3D LF에서 더 높은 것을 확인할 수 있다.

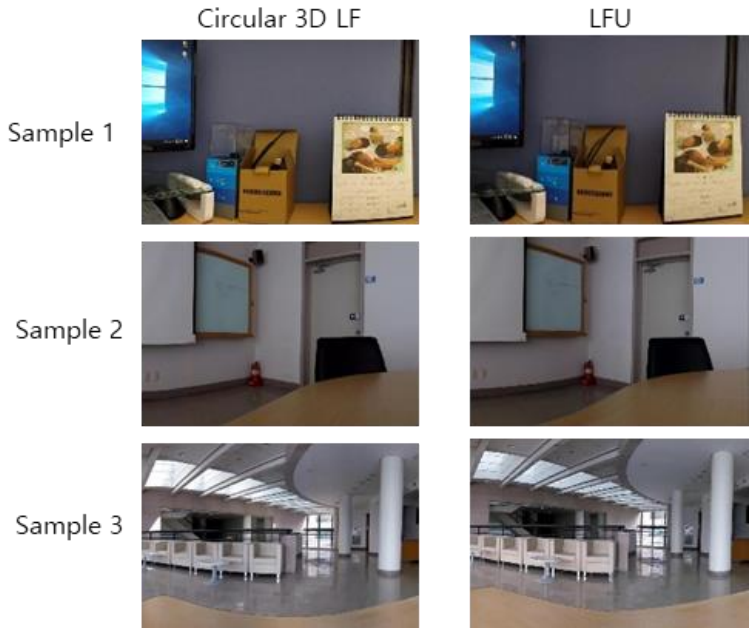


그림 6.6. 원 구조의 3D LF와 LFU를 이용해서 구성한 뷰 비교

표 6.1. 원 구조의 3D LF와 LFU를 이용해서 구성한 뷰 화질 비교

		PSNR(dB)	SSIM	FSIM
Sample 1	Circular 3D LF	**19.30**	**0.812**	0.727
	LFU	18.42	0.785	0.727
Sample 2	Circular 3D LF	**20.77**	**0.807**	0.777
	LFU	20.24	0.792	**0.778**
Sample 3	Circular 3D LF	**17.82**	**0.692**	**0.787**
	LFU	17.35	0.664	0.762

FSIM의 경우 샘플 1에서는 두 가지 뷰 구성 방법의 수치가 동일했으며, 샘플 2의 경우 원 구조의 3D LF가 샘플 3의 경우 LFU에서 더 높은 수치가 측정되었다. 전반적으로 값의 차이는 크지 않았다. 구조적으로 3D LF의 연결 에러가 없는 원 구조의 3D LF의 결과가 좋은 것은 예상된 결과와 같다. LFU의 구조의 결과는 대부분의 경우 원 구조의 3D LF보다 작았지만 그 차이가 크지 않았으며, 따라서 비교적 실제와 유사한 뷰를 만들고 있음을 알 수 있다.

6.2.4 Physical connection과 Non-physical connection의 효율성 비교

그림 4.9 의 결과를 통해 일반 카메라를 사용하는 환경에서 FOV의 제한으로 인해 physical connection의 사용이 제한되는 것을 확인하였다. FOV가 제한될 경우 표현할 수 있는 viewpoint의 범위는 사각형의 구조의 가운데 부분에 해당된다. 물론 180도의 FOV를 만족할 수 있다면 physical connection을 이용한 뷰 구성이 가능하지만, 일반 카메라에서 180도 FOV를 지원하는 것은 현실적으로 어렵다. 하지만 non-physical connection을 이용하는 방법은 확장된 사각형 모양의 LFU를 사용한다는 사실을 고려해야 한다. 따라서 실질적으로 사용하는 데이터 량 대비 표현할 수 있는 viewpoint의 범위를 비교할 필요가 있다. 이를 비교하기 위해 사용하는 데이터 대비 표현할 수 있는 viewpoint의 범위를 (6.1)과 같이 C/L로 정의한다.

$$\frac{C}{L} = \frac{(coverage)\cdot(area)}{(length)} \tag{6.1}$$

식 (6.1)에서 coverage는 전체 범위 중, 표현할 수 있는 viewpoint 범위의 비율을 의미한다. area는 가정하는 구조가 커버하는 전체 범위를 의미하고, length는 해당 구조를 구성하기 위해 필요한 3D LF를 구성하는 직선의 길이를 의미한다. 그림 6.7은 C/L를 비교하기 위해 가정한 사각형 모양의 LFU와 확장된 사각형 모양의 LFU를 의미한다. 사각형 모양의 LFU의 한 변의 길이는 W이다. 확장된 사각형 모양의 LFU에서 내부의 사각형은 마찬가지로 한 변의 길이를 W로 가정하고 확장되는 길이를 각각 W로 가정한다. 이처럼 가정한 구조의 area와 length는 표 6.2와 같다.

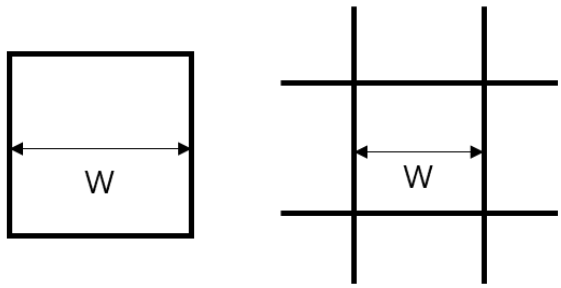


그림 6.7. C/L 비교를 위해 가정한 LFU 구조

두 LFU 구조로 둘러 쌓이는 공간은 W^2로 동일하다. 단, 사각형 모양의 LFU는 해당 구조를 구성하기 위해 4W 만큼의 3D LF 구성을 위한 직선이 필요한 반면, 확장된 사각형 LFU 구조에서는 총 12W의 길이가 필요하다. 단순히 (area)/(length)를 통해 필요한 데이터 량 대비 표현할 수 있는 공간의 범위를 비교하면 확장된 사각형 구조는 매우 비효율적인 것을 알 수 있다.

사각형 모양의 LFU에서 physical connection을 이용해서 뷰를 구성하는 방법은 FOV에 따라 표현할 수 있는 viewpoint의 범위가 사각형의 중심부로 제한된다. 표 6.3는 physical connection과 non-physical connection을 이용하는 방식에서 FOV에 따른 coverage와 C/L 값을 비교한다.

표 6.2. 가정한 LFU 구조의 area, length 비교

	area	length	area/length
사각형 LFU	W^2	4W	W/4
확장된 사각형 LFU	W^2	12W	W/12

표 6.3. Physical connection과 non-physical connection의 FOV에 따른 C/L 비교

	FOV	coverage	C/L
Physical connection	90°	0.00	0.00W
	120°	0.11	0.03W
	150°	0.42	0.11W
	180°	1.00	0.25W
Non-physical connection	90°	2.00	0.16W

사각형 모양의 LFU에서 physical connection을 이용해 뷰를 구성하는 방식은 FOV에 따라 coverage가 바뀐다. 먼저 90° 의 FOV를 만족하는 경우 사각형의 정 가운데 한 점에서만 수평 방향으로 360도 방향의 모든 뷰를 구성할 수 있다. 표현할 수 있는 점은 $(1/W^2)$로 매우 작다. FOV가 120° , 150° 로 증가함에 따라 coverage는 0.11, 0.42로 증가하게 되고, 180° 가 되면 coverage는 1.00이 되어 사각형 내부의 모든 viewpoint에서의 뷰를 구성할 수 있게 된다. 이에 따른 C/L은 0.03W, 0.11W, 0.25W이다. 0.25W가 최대 C/L 이다. 반면 확장된

사각형 모양의 LFU에서 non-physical connection을 이용하는 방법은 항상 90° 만큼의 범위를 할당하기 때문에 카메라 FOV는 최소 90° 만 만족하면 되며, 따라서 90° FOV에 대해서만 살펴본다. Non-physical connection을 이용한 경우 coverage는 2.00인데, 이는 확장된 사각형 구조에서 실제로 표현할 수 있는 범위는 그림 6.8의 회색 범위에 해당된다. 이는 사각형으로 둘러 쌓인 공간의 2배에 해당되는 범위이다. 결과적으로 non-physical connection 방식의 C/L은 0.16W가 된다. 이 값은 physical connection 방식에서 150° 의 FOV를 만족하는 경우의 C/L 값보다 크다.

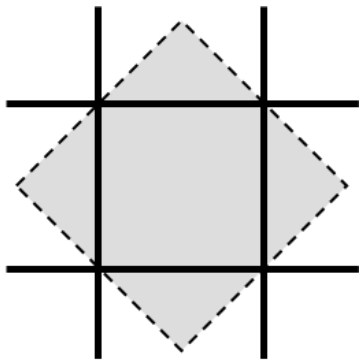


그림 6.8. 확장된 사각형 구조에서 표현할 수 있는 범위

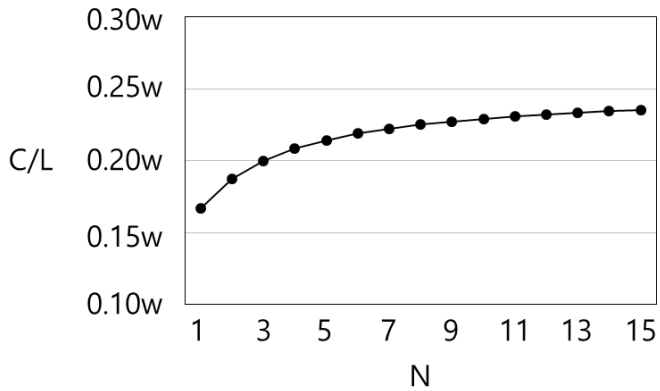


그림 6.9. N × N 크기의 구조를 가정할 때 C/L의 변화 그래프

단, 최대 C/L인, 0.25W에는 미치지 못하며, 이는 확장된 부분으로 인한 비효율은 불가피 하기 때문이다. 그림 4.6에서 설명한 것처럼 이와 같은 비효율은 다수의 LFU가 연결된 구조에서 감춰질 수 있다. 그림 6.9은 LFU가 N × N 으로 다수 개 연결되는 경우 C/L의 변화를 나타내는 그림이다.

N이 1인 경우 C/L은 0.16W이다. N이 증가함에 따라 C/L이 점차적으로 증가하며, 확장된 3D LF로 인한 부담이 점차적으로 감소하게 된다. N이 증가함에 따라 C/L은 점차적으로 0.25W에 점차적으로 수렴하게 되며, N이 무한히 커질 때 C/L은 0.25W가 된다.

6.3 360도 카메라 + dolly를 이용한 구성

일반 카메라를 사용하는 환경은 수평 방향으로 360 방향의 뷰를 구성할 수 있다. 하지만 카메라의 수직 방향의 FOV 제한으로 인해 equi-rectangular 360도 이미지와 같은 완전한 360도 이미지를 구성하지 못하는 한계가 있다. 이를 위해 두 번째 구성 환경에서는 360도 카메라를 사용해서 완전한 360도 이미지 구성을 포함한다. 360도 이미지를 구성함으로써 roll, yaw, pitch의 세 가지 회전 변환을 모두 지원할 수 있다. 환경을 구성하기 위해 사용한 360도 카메라는 Gopro Fusion [48]이다.

또한 앞서 사용한 카메라 슬라이더는 장비의 제원에 의해 이동할 수 있는 범위가 1m도 제한적이었다. 따라서 두 번째 구성 환경에서는 보다 넓은 범위를 대상으로 LF 시스템을 구성하기 위해 dolly 장비를 사용하며, 에델크론사의 dolly plus [49]를 사용한다.

이론적으로 본 연구에서 제안하는 구조는 가정한 직선을 따라 카메라를 이동시키면서 촬영된 이미지로 3D LF를 구성함으로써 뷰를 구성하기 위한 데이터를 준비하는 과정이 모두 끝난다. 그리고 임의의 viewpoint가 정해지면 해당 데이터를 바탕으로 뷰를 구성하게 된다.

하지만 실제 시스템을 구현하는 과정에서 필요한 전처리 과정이 필요하다. 크게 두 가지 과정을 포함하였다. 하나는 3D LF를 구성하기 위한 균일한 이미지 샘플링 작업이다. 슬라이더 장비의 경우 비교적 짧은 거리를 대상으로 정교한 움직임이 가능하기 때문에 카메라의 움직이는 속도를 등속도로 유지할 수 있다. 하지만 dolly 장비는 넓은 공간으로 대상하는데 시작 점부터 도착 점까지 등속도를 유지하지 못하는 문제가 있다. 시작하는 점과 도착하는 점에서 등가속도로 이동을 하고 중간에는 등속도로 이동을 한다. 그 결과 3D LF를 구성하는 직선상에서 균일한 거리만큼 떨어져서 이미지를 추출하는 것이 아니라 처음과 끝 점에서는 더 조밀한 이미지를 추출하게 되는 문제가 있다. 이를 위해서 dolly 장비를 통해 촬영된 비디오에서 이미지를 균일하게 추출하는 과정이 필요하다.

또 다른 전처리 과정은 격자의 교차점에서 이미지를 보정하는 과정이다. 격자 구조를 따라 3D LF를 구성할 경우 왼쪽에서 오른쪽으로 이동하면서 확보한 3D LF와 위, 아래로 이동하면서 확보한 3D LF가 있다. 교차점에서는 두 3D LF에 모두 속하는 이미지가 존재하게 되며, 이론적으로 두 이미지는 완전히 동일해야 한다. 하지만 실제로 카메라 슬라이더, dolly 장비에 장착된 카메라의 alignment 등의 문제로 인해 두 이미지는 대부분의 경우 정확하게 일치하지 않는다. 이는 이후 뷰 구성에 있어서 두 3D LF를 연결하는 과정에서 뷰가 자연스럽게 연결되지 않는 문제를 야기한다. 이를 위해서 교차점에서 두 이미지의 차이를 최대한 줄일 수 있도록 360도 이미지를 회전시키면서 보정하는 과정을 포함한다. 그림 6.10는 교차점에서 두 방향으로 이동하면서 촬영된 두 이미지를 겹쳐놓은 그림이며, 전처리를 통해 보정하기 전, 후의 결과를 보여준다. 그림 6.10 (a)는 360도 이미지 회전을 통한 이미지 보정 전의 두 이미지를 겹쳐놓은 그림이다. 그림에서 두 이미지가 잘 맞지 않음으로 인해 blur와 ghost가 크게 확인된다. 참고로 이미지의 아래에 위치한 검은색 물체는 dolly 장비가 촬영된 부분이며, 이미지의 해당 부분이 잘 맞지 않는 문제는 불가피하다.

(a)

(b)

(c)

그림 6.10. 교차점의 두 이미지가 동일하도록 보정하기 전, 후의 겹쳐진 이미지 비교, (a) 보정 전, (b) 보정 후, (c) 보정 후 결과에서 일부 영역 확대

그림 6.10 (b)는 360도 이미지 회전으로 두 이미지가 잘 맞춰지도록 보정한 결과이다. 그림 6.10 (a)에 비해 ghost가 많이 사라진다. 하지만 보정을 적용하더라도 두 이미지를 정확하게 일치시키기는 어려우며, 그림 6.10 (c)에서 보는 것과 같이 일치하지 않는 부분이 남게 된다.

6.4 360도 카메라 + dolly를 이용한 구조 결과

6.4.1 Hybrid 3D LF Connection Result

360도 이미지를 사용한 환경에서 서로 다른 3D LF를 연결하기 위한 새로운 방안으로 hybrid 3D LF connection을 제시하였다. 이번에는 360도 이미지를 사용하는 환경에서 실제로 hybrid 3D LF connection의 결과가 이전에 제시된 두 방법, physical connection과 non-physical connection과 비교하여 실제로 우수한 성능을 보이는 지 비교한다.

그림 6.11와 그림 6.12는 non-physical connection, physical connection, 그리고 hybrid 3D LF connection을 이용해서 구성한 뷰를 원 형태의 3D LF를 통해 구성한 뷰와 비교한다. 원 형태의 3D LF는 하나의 3D LF 구조를 통해서 뷰를 구성하기 때문에 두 개서의 서로 다른 3D LF의 연결 과정이 필요하지 않다. 따라서 연결 과정에서의 에러가 없으며, 원본 이미지와 유사하다고 할 수 있다. 단, 원 구조는 사각형 구조와 별도의 light acquisition 과정을 통해 획득되었으며, 정확하게 동일한 지점을 찾는 과정에 어려움이 있어 구성된 뷰의 차이가 다소 있다.

먼저 그림 6.11의 샘플 1에 대한 뷰 구성 결과를 비교한다. 그림 6.11 (a)는 샘플 1의 환경에서 3D LF를 구성하는데 사용된 샘플 이미지이다.

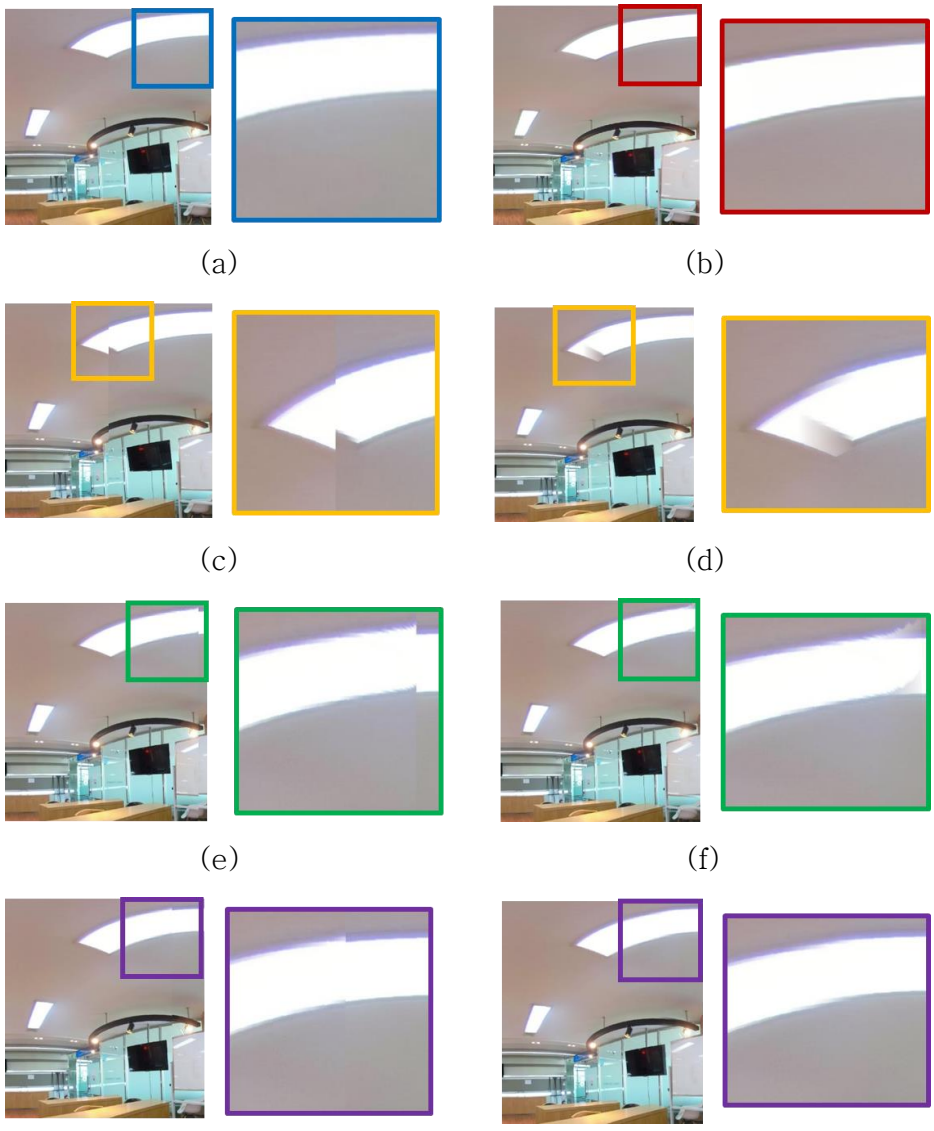


그림 6.11. 구성된 뷰 비교 (샘플 1) (a) Source 이미지, (b) 원 구조, (c) Non-physical connection (w/o blending), (d) Non-physical connection (w/ blending), (e) Physical connection (w/o blending), (f) Physical connection (w/ blending), (g) Hybrid connection (w/o blending), (d) Hybrid connection (w/ blending)

즉, 3D LF의 직선 위의 임의의 점에서 실제로 촬영된 이미지에 해당된다. 그림 6.11 (b)는 원 구조의 3D LF로 구성된 뷰를 보여준다. 각각의 뷰는 특별한 visual artifact 없이 구성된 것을 확인할 수 있다.

원 구조에서 구성된 뷰는 실제 촬영된 것과 같다. 그림 6.11 (c)와 그림 6.11 (d)는 non-physical connection을 이용해서 구성된 뷰를 나타내며, 그림 6.11 (c)는 blending을 포함하지 않은 결과, 그림 6.11 (d)는 blending을 포함한 결과를 나타낸다. Blending은 이미지를 합성하는 알고리즘에서 자주 사용되는 방법으로 본 연구의 경우 두 3D LF를 연결할 때 shared light로 정해진 만큼만 뷰를 할당하는 것이 아니라 추가적으로 범위를 할당하여 오버 랩되는 부분을 만들어 주고 이를 weighted 합을 통해 blending을 적용한다.

그림 6.11 (c)의 blending을 포함하지 않는 non-physical connection은 mismatching이 분명하게 드러나는 것을 확인할 수 있다. 이와 같이 mismatching이 분명한 결과에서는 그림 6.11 (d)에서 보는 바와 같이 blending을 포함하더라도 연결 부분이 자연스럽게 보완되지 않는다.

그림 6.11 (e)와 그림 6.11 (f)는 physical connection을 이용해서 구성된 뷰를 나타내며, 마찬가지로 blending을 포함하지 않을 때의 결과와 포함하는 경우의 과를 보여준다. 그림 6.11 (e)의 physical connection 결과도 마찬가지로 두 3D LF가 연결되는 부분에서 부자연스러운 결과를 확인할 수 있다. 왼쪽 천장의 등이 렌더링 부분에서 휘어지는 듯한 결과가 미세하게 드러난다. 이는 그림 6.11 (f)에서 더욱 극명하게 드러난다. Blending을 위해 추가 범위를 할당하면서 수평 입사 각도가 더 큰 light ray가 사용되고, 그 결과 banding 에러가 눈에 띄게 된다. Blending 과정에서는 banding 현상이 일부 흐려질 수 있지만, 그래도 남으면서 눈에 띄는 것을 확인할 수 있다.

그림 6.11 (g)와 그림 6.11 (h)는 hybrid 3D LF connection을 통해서 구성된 결과를 보여준다. Non-physical connection에 비해 3D LF가 연결되는 지점이 다소 오른쪽으로 이동하였고, physical connection에 비해서는 왼쪽에 위치한다. 해당 위치에서 3D LF를 연결함으로써 mismatching 에러와 banding 에러를 조정한다. 그림

6.11 (g)의 결과에서는 그림 6.11 (c)의 mismatching 에러가 눈에 띄지 않으며, 그림 6.11 (e)의 banding 에러 또한 나타나지 않는 것을 확인할 수 있다. 하지만 미세하게 두 3D LF의 연결이 부자연스러운 것을 확인할 수 있다. 그림 6.11 (h)의 blending을 포함한 결과에서 부자연스럽게 연결된 부분이 다소 개선되는 것을 확인할 수 있으며, 그 결과는 그림 6.11 (a)와 그림 6.11 (b)의 source 이미지와 원 구조에서 구성된 뷰와 유사한 것을 확인할 수 있다.

그림 6.12는 또 다른 샘플에 대해서 구성된 뷰를 비교한 결과이다. 그림 6.12 (a)와 그림 6.12 (b)는 각각 3D LF를 구성하기 위해 사용된 source 이미지와 원 구조의 3D LF를 통해서 구성된 뷰를 나타내며, 역시 원 구조를 기반으로 구성된 뷰는 원본과 유사한 결과를 나타냄을 확인할 수 있다. 그림 6.12 (c), 그림 6.12 (d)의 non-physical connection을 사용한 결과에서는 두 이미지가 연결되는 부분에서 mismatching 에러가 확인되고, blending을 포함한 결과에서는 두 3D LF가 연결되는 부분에 명확한 경계가 두드러지지는 않지만 부자연스럽게 연결됨을 확인할 수 있다. 그림 6.12 (e), 그림 6.12 (f)의 physical connection 을 기반으로 구성된 결과에서는 banding 에러가 확인된다. 그림 6.12 (e)에서 은색 봉 부분이 휘어지는 것처럼 구성된 것을 확인할 수 있다. Blending을 포함한 결과에서도 마찬가지로 banding 에러가 여전히 남아있는 것을 확인할 수 있다. 마지막으로 그림 6.12 (g)와 그림 6.12 (h)의 hybrid 3D LF connection을 적용된 결과에서는 눈에 띄는 mismatching 에러와 banding 에러 없이 뷰가 자연스럽게 구성된 것을 확인할 수 있다.

그림 6.11과 그림 6.12의 결과에서 360도 이미지를 이용한 3D LF 환경을 위해 새롭게 제안된 hybrid 3D LF connection의 결과를 visually 비교하였다. Hybrid 3D LF connection은 mismatching 에러와 banding 에러를 적절하게 줄여주기 위한 방법으로 visual 비교에서 이를 확인할 수 있었다.

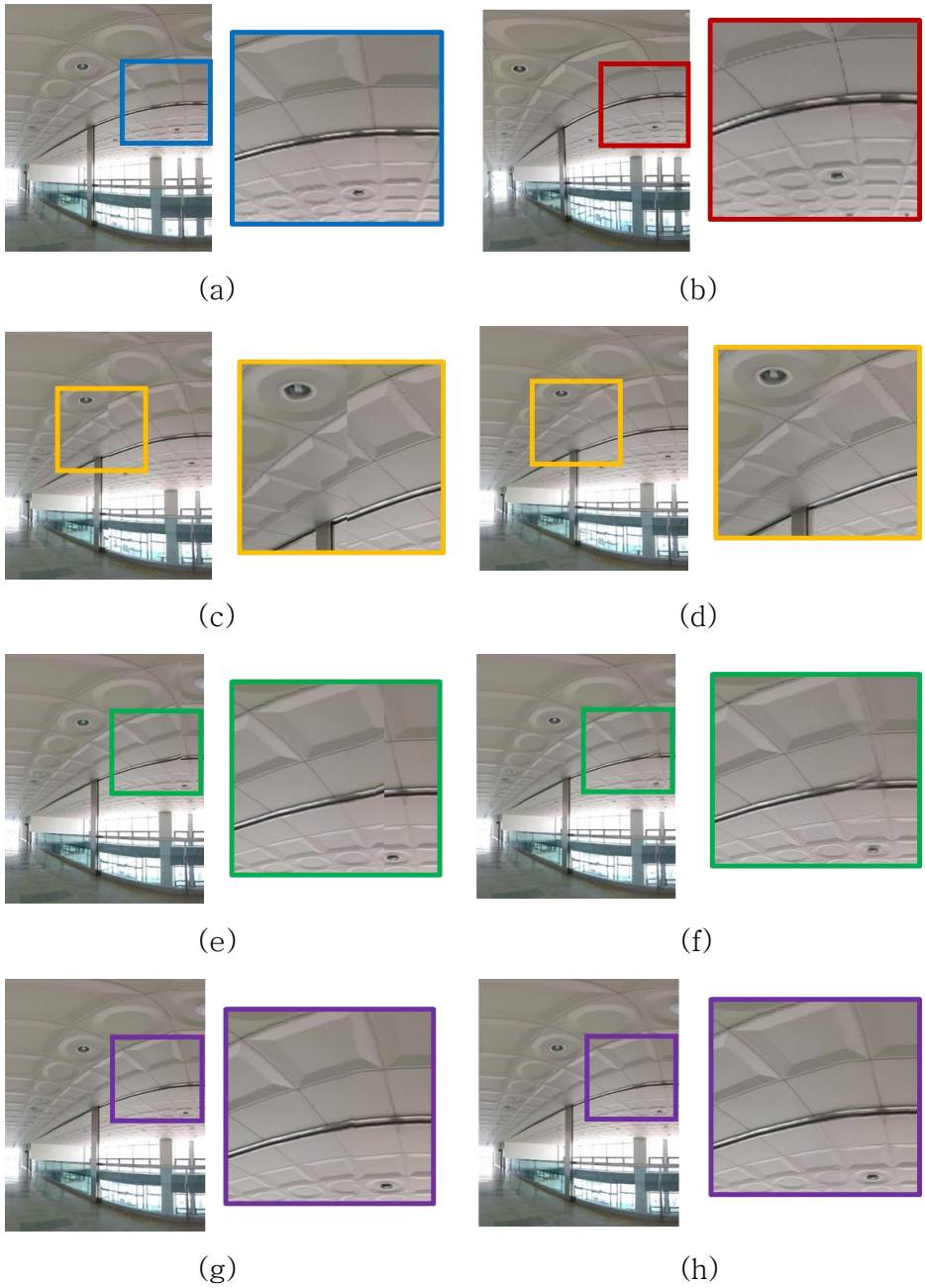


그림 6.12. 구성된 뷰 비교 (샘플 2) (a) Source 이미지, (b) 원 구조, (c) Non-physical connection (w/o blending), (d) Non-physical connection (w/ blending), (e) Physical connection (w/o blending), (f) Physical connection (w/ blending), (g) Hybrid connection (w/o blending), (d) Hybrid connection (w/ blending)

이어서 hybrid 3D LF connection의 성능을 객관적 지표를 사용해서 비교한다. 본 연구에서는 3D LF 연결의 에러를 평가하기 위해 connection 에러를 (6.2)과 같이 정의한다.

$$E_{connect} = \frac{1}{N}\sum_{(x,z)}^{N} mse\left(p_1(x,z), p_2(x,z)\right) \tag{6.2}$$

(6.2)에서 x, z는 임의의 viewpoint의 위치를 의미한다. p_1(x, z)는 임의의 viewpoint에서 하나의 3D LF를 통해 구성된 뷰를 나타내고, p_2(x, z)는 이웃한 또 다른 하나의 3D LF를 통해 구성된 뷰를 나타낸다. 여기서 두 뷰는 동일한 범위의 뷰를 의미한다. 앞선 blending에 대한 설명과 같이 shared light 보다 큰 범위에 대해서 뷰를 구성함으로써 이웃한 두 3D LF에서 구성된 뷰를 오버랩 되도록 하였다.

그림 6.13. 객관적 평가 지표의 평가 대상 샘플

(6.2)의 p1(x, z), p2(x, z)는 이웃한 서로 다른 두 3D LF에서 구성된 뷰 중 오버랩 된 영역을 의미한다. Connection 에러는 이 두 뷰의 픽셀의 mean square error (mse)를 통해 정의된다. 이 때 viewpoint의 위치에 따라 비교 대상이 되는 뷰의 영역이 바뀌기 때문에 LFU 내의 모든 viewpoint에서의 평균 값을 최종 connection error로 정의한다. 그림 6.13는 평가 지표를 비교하는 평가 대상 샘플이다.

표 6.4. K 변화에 따른 Hybrid 3D LF connection의 connection error 비교

K	Sample No.					
	1	2	3	4	5	6
70	650.1	411.0	276.7	1251.8	638.4	506.7
72	646.0	389.1	266.1	1261.0	625.4	503.7
74	637.9	379.0	263.4	1260.0	612.9	494.1
76	630.1	**377.8**	267.8	1256.8	604.9	**486.7**
78	621.5	397.2	262.5	1251.8	600.0	490.0
80	609.8	410.3	**261.3**	**1245.5**	594.1	489.4
82	**601.5**	424.7	264.1	1255.5	**592.6**	495.6
84	602.1	440.4	268.3	1278.5	600.2	516.8
86	621.8	477.9	285.7	1301.8	620.2	541.8

표 6.4은 Hybrid 3D LF connection으로 구성된 뷰의 connection error를 K의 변화에 따라 비교한 결과이다. 실험에서 K는 70° ~ 86° 의 범위를 두었다. 그리고 각각의 샘플에서 최소의 connection error에 해당되는 부분은 굵은 글씨로 표시된 부분이다. 샘플 1의 경우 K가 82일 때 최소 connection 에러를 확인하였다. 6개의 샘플에서 최적의 K는 각각 76° , 80° , 82° 의 세 가지 경우로 선택되었다. 샘플 1, 샘플 5의 경우 82° 로 다른 샘플에 비해서 다소 큰 K가 선택되었다. 이는 주변의 물체, 또는 배경이 상대적으로 가까웠다는 것으로 해석될 수 있다. 샘플 2, 샘플 6의 경우 76° 의 K 값이 선택되었으며, 이는 반대로 물체 또는 배경의 위치가 평균적으로 멀었다고 해석될 수 있다. 전반적으로 K의 선택이 큰 차이는 없었다.

표 6.5는 Hybrid 3D LF connection을 이용한 비교에서 최적으로 선택된 K에 대한 connection error와 non-physical connection, physical connection 기반의 뷰 구성의 connection error를 비교한다.

표 6.5. 각 3D LF 연결 방법의 Connection error 비교

No.	NonPhysical	Physical	Hybrid
1	812.6	621.8	601.5
2	968.0	479.6	377.8
3	543.2	286.6	261.3
4	1447.0	1303.5	1245.5
5	1664.9	620.6	592.6
6	1252.9	542.0	486.7

표 6.5의 결과에서 모든 샘플에서 Hybrid 3D LF connection의 connection 에러가 가장 작은 것을 확인할 수 있었다.

눈에 띄는 결과는 physical connection의 connection error를 살펴보면, 표 6.4의 K가 86° 인 경우 Hybrid 3D LF connection의 connection error와 유사하다는 점이다. 이는 K가 커짐에 따라서 Hybrid 3D LF에서 physical connection을 사용하는 비중이 크기 때문이며, 실질적으로 K가 90° 인 경우의 Hybrid 3D LF connection은 physical connection과 동일하다고 할 수 있다. 반대로 K가 45° 인 경우의 Hybrid 3D LF connection은 non-physical connection과 동일하다고 할 수 있다. 실험 범위에서 45° 만큼 작은 K는 포함되지 않았으나 K가 45° 에 가까워짐에 따라 표 6.5의 non-physical connection의 결과와 가까워졌을 것이라고 생각할 수 있다.

표 6.5에서 전반적으로 non-physical connection의 connection error가 컸다. 이는 non-physical connection으로 인한 mismatching 에러는 이미지 전반에 걸쳐서 mse 값을 크게 높인다면, banding 에러는 ±90° 에 가까운 light ray가 사용되는 이미지의 일부에서 발생하기 때문으로 생각될 수 있다. 그 결과 작은 K를 사용하는 Hybrid 3D LF connection 에서는 mismatching 에러로 인한 connection error의 상승으로 physical connection 보다 connection error가 커지는 경우가 샘플 1, 샘플 4에서 나타난다.

6.4.2 구성한 구조 및 임의의 viewpoint에 따른 뷰 구성 결과

그림 6.14은 360도 카메라와 dolly 장비를 사용해서 구성한 LFU 구조를 나타낸다. 세로 방향으로 28m, 가로 방향으로 3m의 공간을 대상으로 LFU 구조를 구성하였다. 사각형 모양의 LFU를 기준으로 한 변은 50cm에 해당된다. 따라서 세로 방향으로 56개의 LFU가 쌓이고,

가로 방향으로 6개의 LFU가 쌓여 있는 구조이다. 그림 6.14의 구조를 통해 이동할 수 있는 면적은 총 84m^{2}에 해당된다.

360도 이미지를 기반으로 하기 때문에 non-physical connection 대신 hybrid 3D LF connection을 사용해서 뷰를 구성하고 연결한다. Hybrid 3D LF에서 사용하는 K는 75° 로 제한을 두었다. 또한 임의의 viewpoint에 대한 뷰를 구성함에 있어서 3D LF stack 구조를 고려한 뷰 구성을 같이 사용한다.

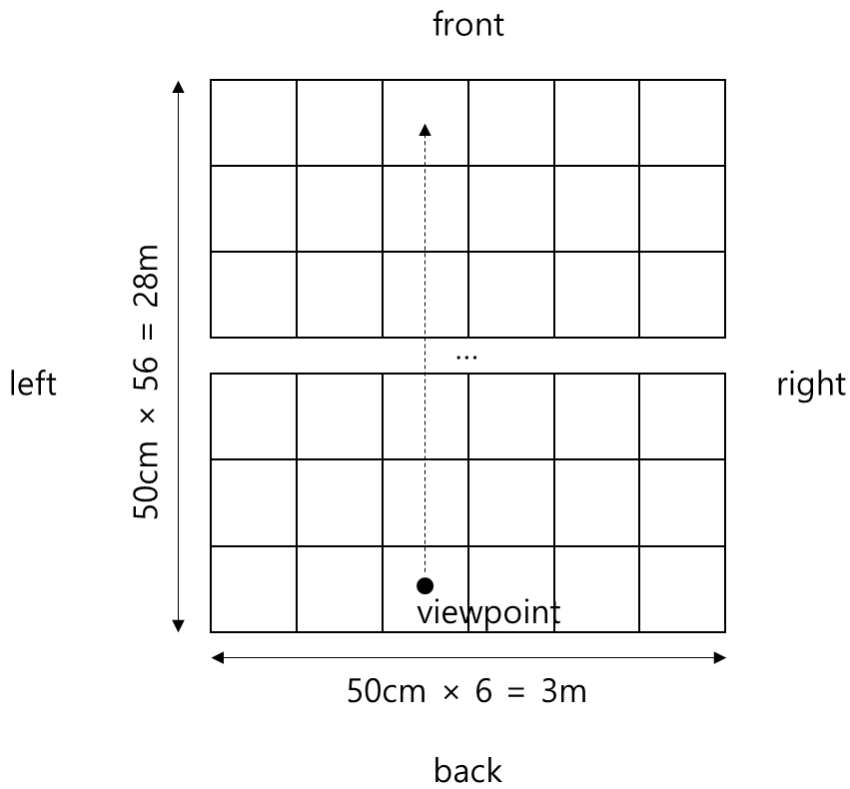


그림 6.14. 360도 카메라 + dolly 장비 사용 환경으로 구성한 LFU 구조

임의의 viewpoint는 그림 6.14의 viewpoint와 같이 왼쪽에서 세

번째 LFU에 위치하며 검은 색 점선 화살표로 표시된 경로를 따라 움직이는 상황을 고려한다. 그림 6.15는 해당 viewpoint의 이동에 따른 뷰 변화를 보여준다. 각각의 이미지는 5 개의 LFU 단위로 떨어진 viewpoint에서의 뷰를 나타낸다.

그림 6.15 (a) ~ 그림 6.15 (i)의 결과 이미지는 앞선 일반 카메라를 이용한 환경 구성과 달리 수직, 수평 방향으로 모두 만족하는 360도 이미지를 만들고 있는 것을 확인할 수 있다. 그림 6.15 (a) ~ 그림 6.15 (i)의 이미지에서 흰색 바닥으로 표시된 부분은 그림 6.14에서 left 방향에 해당된다. 즉, viewpoint가 이동함에 따라 회색 바닥 부분에 전시되어 있는 차들은 360도 뷰에서 왼쪽 방향으로 이동해야 하며, 그림 6.15 (a) ~ 그림 6.15 (i)의 결과 이미지에서 이는 잘 반영되고 있음을 확인할 수 있다. 맞은 편에 해당되는 right 방향의 자동차 및 배경은 오른쪽 방향으로 이동하는 것을 확인할 수 있다.

360도 카메라와 dolly 장비를 이용한 뷰 구성은 이전의 일반 카메라를 사용한 결과에 비해서 더 넓은 공간을 커버할 수 있음을 보였으며, 완전한 360도 이미지를 구성함으로써 roll, yaw, pitch의 세 가지 회전 변환을 모두 지원할 수 있다. 구성된 뷰에는 여전히 남아 있는 visual artifact가 있다. 하나는 이웃 3D LF가 연결되는 부분에서 여전히 mismatching 에러를 포함하고 있는 문제이다. 그림 6.16 (a)는 그림 6.15 (i)의 일부분을 확대한 그림이며, 두 3D LF가 연결되는 부분에서 발생하는 mismatching 에러를 보여준다. 해당 부분에서 여전히 mismatching 에러가 남아있음을 확인할 수 있는데, 이는 hybrid 3D LF connection에서 non-physical connection을 이용하는 경우 mismatching 에러가 불가피 하게 포함되는 문제도 있지만, 구현 과정에서 카메라의 alignment의 문제도 포함된다. 앞서 전 처리를 통해 alignment를 보정해주는 과정을 포함하지만 완전히 동일한 이미지로 보정하지 못하는 문제가 3D LF의 연결에서 드러나기도 한다.

(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

(i)

그림 6.15. 임의의 viewpoint 이동에 따른 뷰 변화

제안 시스템에 남아 있는 또 다른 visual 에러는 ghost 에러이다. 3D LF stack 구조에서 앞, 뒤 두 개의 3D LF를 동시에 사용해서 뷰를 구성하는데, 두 3D LF를 동시에 사용하는 과정에서 정확하게 동일한 픽셀을 찾지 못하는 경우, 그림 6.16 (b)와 같은 ghost 에러가 남게 된다.

(a) (b)

그림 6.16. 제안 시스템에서 구성된 뷰의 에러, (a) mismatching 에러, (b) ghost error

6.4.3 다른 자유 시점 변화 시스템과의 비교

앞서 본 연구에서는 LF 기반의 새로운 자유 시점 변환 시스템을 제안하였다. 기존의 단일 3D LF 또는 4D LF를 가정하고 해당 LF를 통해 뷰를 구성하는 방식이 아니라 다수의 3D LF를 다수 쌓는 3D LF stack 기반의 자유 시점 변환 시스템을 제안하였다.

이번 장에서는 마지막으로 본 논문에서 제안한 시스템을 기존의 다른 자유시점 변환 시스템 및 가상현실 시스템과 비교함으로써 제안 시스템의 실험 결과를 마무리 한다. 표 6.6의 첫 번째 column은 3-DoF VR 시스템이다. 이는 기존의 대중화된 가상현실 시스템을 의미한다. 이 시스템은 한 점에서 촬영된 360도 이미지를 사용한다. 360도 이미지를 구성하기 위해 다양한 수의 카메라가 사용된다. 180도 이미지를 촬영할 수 있는 fisheye 렌즈를 장착한 두 개의 카메라를 사용하는 경우가 많다. 대표적으로 삼성 Gear360, Gopro Fusion 등이 그러하다. 다수 개의 카메라를 통해 동시에 360도 방향의 이미지를 촬영하고 이를 스티칭을 통해 연결함으로써 360도 뷰를 구성한다. 다중 카메라로 동시에 이미지를 촬영하기 때문에 비디오 데이터를 획득할 수 있다. 하지만 표현할 수 있는 범위는 이미지가 촬영된 한 점으로만 제한이 된다. 따라서 기존의 가상현실 시스템에서 표현할 수 있는 DoF는 roll, yaw pitch의 세 가지 회전 변환뿐이다.

표 6.6의 두 번째 column은 3D 모델링이다. 3D 모델링은 실제 공간의 3차원 정보를 추정하고 가상의 공간에 재구성함으로써 해당 공간의 다양한 viewpoint에서의 뷰를 자유롭게 구성할 수 있다. 일반적으로 한 대의 카메라를 이동하면서 주변 정보를 스캐닝하는 방식을 사용한다. RGB 카메라뿐 아니라 깊이 정보를 추출하는 카메라나 구조광, 라이다 등 다양한 전문 장비가 동원되기도 한다. 스캐닝 방식의 데이터 취득을 사용하기 때문에 비디오 데이터를 획득할 수 없다. 얼마나 넓은 공간의 데이터를 획득하느냐에 따라서 넓은 공간을 커버할 수 있지만 그만큼 계산 복잡도가 급격히 증가한다.

표 6. 6. 다양한 자유 시점 변환 시스템의 비교

	3-DoF VR	3D Modeling	LF 기반의 접근 방법			
			FTV	Spherical LF	Proposed (일반 카메라)	Proposed (360 카메라)
구조 모양	One point	Hand-held	평면	구면	확장된 사각형	확장된 사각형
카메라 개수	2~30	1	~100	2 or 16	1	1
LF 획득 방식	다중 카메라	하나의 카메라로 스캐닝 (전문 장비)	다중 카메라	다중 카메라로 스캐닝	하나의 카메라로 스캐닝	하나의 카메라로 스캐닝
비디오 획득 가능 여부	가능	불가능	가능	불가능	불가능	불가능
표현 범위의 확장 가능 여부	불가능	가능	불가능	불가능	가능	가능
지원 DoF	3DoF: yaw, pitch, roll	6DoF: x, y, z, yaw, pitch, roll	3DoF: x, y, z	6DoF: x, y, z, yaw, pitch, roll	4DoF: x, z, yaw, roll	5DoF: x, z, yaw, pitch, roll

정교한 3D 모델링을 해놓기만 한다면 해당 공간에 대해서 자유로운 시점 변환이 가능하다. 따라서 x, y, z 축을 따라 수평 이동하거나 roll, yaw, pitch의 세 가지 회전 변환을 모두 표현함으로써 6-DoF를 지원할 수 있다. 3D 모델링의 또 다른 단점은 최종적으로 구성된 뷰가 photo-realistic 하지 않다는 점이다.

세 번째부터 여섯 번째 column은 LF 기반의 시스템에 해당된다. 세 번째 column은 FTV이다. FTV는 2D 평면을 가정하고 해당 평면을 통과하는 light ray를 바탕으로 뷰를 구성한다. 다양한 시스템 환경을 사용하며, 최대 100대의 다중 카메라를 이용해서 2D 평면을 통과하는 light ray를 획득한다. 다중 카메라를 사용해서 동시에 이미지를 취득하기 때문에 비디오 형태의 데이터 취득이 가능하다. FTV의 구조는 확장 가능한 구조는 아니며, 넓은 공간을 대상으로 하기 위해서는 구조 자체를 크게 가정해야 한다. 가정하는 2D 평면은 한 방향으로 배치가 되어 있어 viewpoint는 가정한 2D 평면을 지원하는 범위 내에서 x, y, z 축을 따라 수평 이동할 수 있으며, 3-DoF를 지원한다.

네 번째 column의 spherical LF는 구글에서 개발한 장비를 통해서 실제로 구현되었다. 구면을 가정하고 해당 면을 통과하는 light ray를 통해 뷰를 구성한다. 구면을 통과하는 light ray를 취득하기 위해 반대 방향을 바라보는 두 대의 카메라를 구 형태로 회전시키는 장비 또는 16대의 카메라를 아치형으로 배치하고 이를 원으로 회전시키는 장비가 사용된다. 두 장비 모두 스캐닝 방식으로 light ray를 획득하기 때문에 비디오 형태의 데이터 취득은 불가능하다. Spherical LF 또한 구조 자체의 크기를 크게 가정함으로써 표현할 수 있는 viewpoint의 범위를 넓힌다. 하지만 매우 큰 구를 가정하더라도, 해당 구 면을 통과하는 light ray를 취득하는 장비의 개발이 현실적으로 어렵기 때문에 표현할 수 있는 viewpoint의 범위는 제한될 수 밖에 없다. 단, Spherical LF는 제한된 구 범위 내에서는 x, y, z 축의 수평 이동과 roll, yaw, pitch의 회전 변환을 모두 지원하는, 6-DoF 지원 시스템이다. 현재 구글에서 개발한 장비는 60cm 지름의 구 공간에 대해서 viewpoint 이동이

가능하다.

다섯 번째 column과 여섯 번째 column은 본 논문의 제안 시스템이며, 다섯 번째 column은 일반 카메라를 사용하여 구성한 시스템, 여섯 번째 column은 360도 카메라를 사용해서 구성한 시스템이다. 두 경우 모두 LFU 라는 이름의 확장된 사각형 구조이다. LFU를 여러 개 쌓아줌으로써 표현할 수 있는 범위를 자유롭게 확장할 수 있다. 다수의 LFU가 연결된 구조는 결과적으로 격자 형태가 된다. 격자를 따라 한 대의 카메라를 이동하면서 데이터를 취득하게 된다. 두 가지 구성 시스템 모두 스캐닝 방식으로 데이터를 취득하기 때문에 비디오 데이터는 구성할 수 없다. 두 가지 구성 시스템에서 viewpoint는 격자가 배치된 평면을 따라 x, z축의 수평 이동이 가능하다. 일반 카메라를 사용하여 구성된 시스템은 수직 방향의 FOV 제한으로 인해 pitch의 회전 변환을 지원하지 못하여, roll, yaw만 지원하는 4-DoF 시스템에 해당된다. 360도 카메라를 사용하고 수직 방향의 전체 FOV에 해당하는 light ray를 사용함으로써 pitch를 추가적으로 지원하는 5-DoF 시스템이 된다.

제안하는 시스템은 결과적으로 6-DoF 보다 한 단계 낮은 DoF를 지원한다. y축으로의 시점 변환을 지원하기 위해서는 구성한 격자를 지원하고자 하는 y축 만큼 쌓아줘야 하는데, 이는 상당히 부담스러운 비용 증가이다. 약 $80m^2$에 해당하는 넓은 공간을 자유롭게 이동할 수 있는 시스템에서 y축의 지원은 요구되는 비용 대비 필요성이 낮다는 점에서 지원에서 제외한다.

제 7 장 결 론

본 논문은 넓은 공간을 자유롭게 이동하기 위한 새로운 3D LF 기반의 가상현실 시스템을 제시한다. 기존의 LF 기반의 자유시점 변환 시스템은 light ray 획득에 어려움이 있었으며, 특히 구 면을 가정하는 LF 시스템은 전문 장비를 개발하는 등의 많은 비용이 필요하다. 또한 보다 넓은 범위로 대상을 넓히기 위해서는 가정하는 구조 자체를 크게 가정해야 하는데, 이로 인한 light ray 획득 난이도는 더욱 증가하게 된다. 3D LF는 면 대신 선을 통과하는 light ray를 구성되며, 선을 따라 카메라를 이동하면서 light ray를 획득하는 과정이 상대적으로 매우 간편하다. 하지만 기존의 LF에 비해 획득할 수 있는 light ray가 제한적인 문제가 있다. 즉, vertically 한 점에서만 light ray를 획득하기 때문에 3D LF 기반의 view generation은 vertical parallax를 표현하지 못하는 문제가 있다. 특히 이는 넓은 공간을 대상으로 viewpoint를 이동할 때, viewpoint와 3D LF간의 거리가 멀어짐에 따라 더욱 증가하게 되고, 구성된 뷰에 많은 왜곡을 포함하게 된다.

본 논문은 3D LF Stack을 이용한 자유 시점 변환 시스템을 제안하였다. 3D LF Stack은 다수의 3D LF를 동일한 방향을 바라보도록 순서대로 배치한 구조이며, 각각의 3D LF는 일정 거리를 두고 배치된다. 이를 통해 임의의 viewpoint의 위치에 따라 앞에 위치한 3D LF를 이용해서 뷰를 구성하게 된다. 이는 viewpoint가 넓은 공간을 자유롭게 움직이더라도 viewpoint와 3D LF 사이의 거리를 일정 수준으로 제한할 수 있으며, 그 결과 vertical parallax를 표현하지 못함으로 발생하는 뷰의 왜곡 또한 일정 수준으로 제한할 수 있다. 또한 본 논문의 제안 구조는 두 개의 3D LF Stack을 수직 방향으로 배치하고, 각 3D LF를 구성함에 있어서 단 방향으로 통과하는 light ray 뿐 아니라 양 방향으로 통과하는 모든 light ray를 포함하도록 함으로써 임의의 viewpoint에 대한 360도 뷰를 구성할 수 있도록 하였다. 이는 마치 3D LF가 격자로 배치된 구조와 같으며, 해당 구조에서 임의의 viewpoint는

자유롭게 이동하게 된다. 임의의 viewpoint에서의 뷰는 해당 viewpoint를 둘러쌓는 네 개의 3D LF를 사용해서 구성하게 되며, 네 개의 3D LF로 구성된 사각형 형태의 구조를 LFU로 정의한다.

본 논문의 제안 구조를 위해서는 두 가지 해결 과제가 있다. 하나는 이웃한 3D LF 간의 연결이고, 또 다른 하나는 LFU를 이동함에 있어서 발생하는 뷰의 급변 문제이다. 먼저 이웃 3D LF 간의 연결은, 기존 LF 또는 3D LF 기반 시스템은 하나의 LF 구조를 바탕으로 하나의 뷰를 구성하였는데, 본 논문의 구조는 네 개의 3D LF를 연결하여 하나의 뷰를 구성하기 때문에 이를 연결하기 위한 방안이 필요하다. 본 논문은 이웃한 3D LF를 연결하기 위한 두 가지 연결 방안, physical connection과 non-physical connection을 이용한 방안을 제시한다. 각각의 연결 방안은 장, 단점이 있으며, 주어진 환경에 따른 적정 방안을 소개한다.

LFU를 이동함에 따른 뷰의 급변 문제는 3D LF Stack 구조에서 발생하는 문제이다. 임의의 viewpoint가 움직이면서 뷰를 구성하는 3D LF가 바뀌게 되는데, 3D LF 가 바뀌는 순간의 뷰가 급격하게 바뀌는 문제가 발생하게 된다. 이는 서로 다른 3D LF로 만들어진 뷰가 서로 다른 왜곡을 포함하기 때문에 자연스럽게 연결되지 못하기 때문이다. 본 논문은 3D LF Stack 구조에서 epioplar geometry 관계를 가지는 light ray 세트를 이용한 뷰 구성 방안을 제시함으로써 viewpoint가 이동함에 따른 뷰 변환이 자연스럽도록 한다.

본 논문의 제안 구조를 실제 적용하고 임의의 viewpoint를 이동하는 뷰를 구성한 결과 기존 연구에 비해 상당히 넓은 공간을 자유롭게 움직일 수 있었다.

참고 문헌

[1] R. Ma, T. Maugey, and P. Frossard, “Optimized data representation for interactive multiview navigation,” *IEEE Trans. Multimedia*, vol. 20, no. 7, pp. 1595−1609, July 2018.

[2] O. Stankewics et al., “A free−viewpoint television system for horizontal virtual navigation,” *IEEE Trans. Multimedia*, vol. 20, no. 8, pp. 2182−2195, Aug. 2018.

[3] T. Maugey, L. Guillo, and C. L. Cam, “FTV360: a multiview 360° video dataset with calibration parameters,” in *Proc. 10th ACM Multimedia Syst. Conf*. Jun. 2019, pp. 291−295.

[4] T. Maugey and P. Frossard, “Interactive multiview video system with low complexity 2d look around at decoder,” *IEEE Trans. Multimedia*, vol. 15, no. 5, pp. 1070−1082, Aug. 2013.

[5] Y. Furukawa, and J. Ponce, “Accurate, dense, and robust multiview stereopsis.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 32(8), p. 1362−1376, 2010.

[6] H−H. Vu, P. Labatut, and JP. Pons, “High accuracy and visibility−consistent dense multiview stereo.” *IEEE transactions on pattern analysis and machine intelligence*, vol. 34(5) no. 5, p. 889−901, 2012.

[7] M. Kazhdan, and H. Hoppe, “Screened poisson surface reconstruction.” *ACM Transactions on Graphics (ToG)*, vol. 32(3), no. 29, 2013.

[8] F. Remondino, “From point cloud to surface: the modeling and visualization problem.” *International Archives of photogrammetry, Remote Sensing and spatial information sciences*, vol. 34, 2003.

[9] E. H. Adelson and J. R. Bergen, “The plenoptic function and the elements of early vision,” in *Computation Models of Visual Processing*, M. S. Landy and J. A. Movshon, Eds., Cambridge, MA, USA: MIT Press, 1991, pp. 3−20.

[10] M. Levoy and P. Hanrahan, “Light field rendering,” in *Proc. Computer Graphics, Ann. Conf. Series*, SIGGRAPH, Aug. 1996, pp. 31−42.

[11] T.−T. Wong, C.−W. Fu, P.−A. Heng, and C.−S. Leung, “The plenoptic illumination function,” *IEEE Trans. Multimedia*, vol. 4, no. 3, pp. 361−371, Sep. 2002.

[12] S. Choi, Q-Y. Zhou, and V. Koltun, "Robust reconstruction of indoor scenes." *in proc of the IEEE Conference on Computer Vision and Pattern Recognition*. p. 5556-5565, 2015.

[13] A. Dai, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration." *ACM Transactions on Graphics (TOG)*, vol. 36(3), no. 24, 2017.

[14] C. Kerl, J. Sturm, and D. Cremers. "Dense visual SLAM for RGB-D cameras." *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on. IEEE*, 2013.

[15] R. Mur-Artal, and J. M. M. Montiel, J. D. Tardós, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System." *IEEE transactions on Robotics*, vol. 31(5), p. 1147-1163, 2015.

[16] RA. Newcombe, "KinectFusion: Real-time dense surface mapping and tracking." *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE, p. 127-136, 2011.

[17] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald, "Kintinuous: Spatially extended KinectFusion," *in Proc. Workshop RGB-D, Adv. Reason. Depth Cameras*, 2012.

[18] Kinect, Retrieved Jun. 20. 2020., from https://developer.microsoft.com/ko-kr/windows/kinect/

[19] R. Wang, "3D building modeling using images and LiDAR: A review." *International Journal of Image and Data Fusion*, vol. 4(4), p. 273-292, 2013.

[20] J. Geng, "Structured-light 3D surface imaging: a tutorial." *Advances in Optics and Photonics*, vol. 3(2), p. 128-160, 2011.

[21] P. Jenke, B. Huhle, and W. Straßer. "Statistical reconstruction of indoor scenes." *In Proc. of 17. Int. Conf. in Central Europe on Comp. Graphics, Vis. and Comp. Vision (WSCG '09)*, February 2 - 5, p. 17-24.

[22] J. Xiao, and Y. Furukawa. "Reconstructing the world's museums." *International journal of computer vision* vol. 110(3), p. 243-258, 2014.

[23] A. Flint, D. Murray, and I. Reid, "Manhattan scene understanding using monocular, stereo, and 3d features." *In Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2228-2235, 2011

[24] S. Ikehata, H. Yang, and Y. Furukawa, “Structured indoor modeling.” in *Proc. IEEE International Conference on Computer Vision (ICCV),* pp. 1323–1331, 2015.

[25] N. Doh, H. Choi, B. Jang, S. Ahn, H. Jung, and S. Lee, “TeeVR: Spatial Template–Based Acquisition, Modeling, and Rendering of Large–Scale Indoor Spaces,” in *Proc, ACM SIGGRAPH Emerging Technologies*, 2019, pp. 1

[26] S.Wanner and B. Goldl¨uecke, “Variational light field analysis for disparity estimation and super–resolution,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 3, pp. 606–619, Mar. 2014.

[27] Y. Wang, Y. Liu, W. Heidrich, and Q. Dai, “The light field attachment: Turning a DSLR into a light field camera using a low budget camera ring,” *IEEE Trans. Vis. Comput. Graph.*, vol. 23, no. 10, pp. 2357–2364, 2016.

[28] P. Didyk, P. Sitthi–Amorn, W. Freeman, F. Durand, and W. Matusik, “Joint view expansion and filtering for automultiscopic 3D displays,” in *Proc. ACM SIGGRAPH*, 2015, pp. 3906–3913.

[29] S. Vagharshakyan, R. Bregovic, and A. Gotchev, “Light field reconstruction using Shearlet transform,” *IEEE Trans. Pattern Anal. Mach. Intell.*, http://ieeexplore.ieee.org/document/7817742/.

[30] T. Basha, S. Avidan, A. Sorkine–Hornung, and W. Matusik, “Structure and motion from scene registration,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 1426–1433.

[31] S. Zhang, H. Sheng, C. Li, J. Zhang, and Z. Xiong, “Robust depth estimation for light field via spinning parallelogram operator,” *Comput. Vis. Image Understanding*, vol. 145, pp. 148–159, 2016.

[32] B. Wilburn et al., “High performance imaging using large camera arrays,” in *Proc. ACM SIGGRAPH*, 2005, pp. 765–776.

[33] M. Tanimoto, M. P. Tehrani, T. Fujii, and T. Yendo, “Free–viewpoint TV,” *IEEE Signal Process. Mag.*, vol. 28, no. 1, pp. 67–76, Dec. 2010.

[34] C. Galdi, V. Chiesa, C. Busch, P. L. Correia, J.–L. Dugelay, and C. Guillemot, “Light Fields for Face Analysis,” *Sensors*, vol. 19, no. 12, pp. 1–27, Jun. 2019.

[35] Lytro illum light field camera, Retrieved Jun. 20. 2020., from https://raytrix.de/

[36] M. Tanimoto and H. Kurokawa, “Ray-space processing for omnidirectional FTV,” *Three-Dimensional Imaging, Visualization, and Display*, vol. 10666, May 2018.

[37] R. S. Overbeck, D. Erickson, and D. Evangelakos, “The making of welcome to light fields VR,” in *Proc. ACM SIGGRAPH talks*, Aug. 2018, pp. 63.

[38] D. Donatsch, S. A. Bigdeli, P. Robert, and M. Zwicker, “Hand-held 3D light field photography and applications,” *The Visual Comp.*, vol. 30, no. 6-8, pp. 897-907, Jun. 2014.

[39] C. Kim, H. Zimmer, Y. Pritch, A. Sorkine-Hornung, and M. Gross, “Scene reconstruction from high spario-angular resolution light field,” *ACM Trans. Graph.*, vol. 32, no. 4, July 2013.

[40] C. Kim, A. Hornung, S. Heinzle, W. Matusik, and M. Gross, “Multi-Perspective stereoscopy from light fields.” *ACM Trans. Graph.*, vol. 30(6), no. 190, Dec. 2011.

[41] H.-Y. Shum, K.-T. Ng, and S.-C. Chan, “A virtual reality system using the concentric mosaic: construction, rendering, and data compression,” *IEEE Trans. Multimedia*, vol. 7, no. 1, pp. 85-95, Feb. 2005.

[42] S. Maesen, P. Goorts, and P. Bekaert, “Omnidirectional free viewpoint video using panoramic light fields,” in *Proc. 3DTV-Conf.: The True Vision-Capture, Transmission and Display 3D Video*, July 2016, pp. 1-4.

[43] S. M. Seitz and J. Kim, “The space of all stereo images,” *International Journal of Computer Vision*, vol. 48, pp. 21-38, Jun. 2002.

[44] Gopro Hero action camera, Retrieved Jun. 20. 2020., from https://gopro.com/ko/kr/shop/cameras/hero8-black/CHDHX-801-master.html

[45] Konova K3 camera slider, Retrieved Jun. 20. 2020., from http://bbosasi.com/goods/goods_view.php?goodsNo=1000003446

[46] Z. Wang, AC. Bovik, HR. Sheikh, and EP. Simoncelli, “Image Quality Assessment: From Error Visibility to Structural Similarity,” *IEEE Trans. Image Processing*, vol. 13, no. 4, pp. 600-612, Apr. 2004.

[47] L. Zhang, L. Zhang, X. Mou, and D. Zhang, “FSIM: A Feature Similarity Index for Image Quality Assessment,” *IEEE Trans. Image*

Processing, vol. 20, no. 8, pp. 2378-2386, Jan. 2011.

[48] Gopro Fusion 360-degree camera, Retrieved Jun. 20. 2020., from https://gopro.com/en/us/fusion

[49] Edelkrone dolly plus, Retrieved Jun. 20. 2020., from https://edelkrone.com/

Abstract

A Stackable 3D Light Field System for Free Viewpoint Virtual Reality

Hyunmin Jung
Electrical and Computer Engineering
The Graduate School
Seoul National University

Conventional captured-image-based virtual reality (VR) systems only support rotational view direction changes, roll, yaw, and pitch. It is 3-degree-of-freedom (3-DoF). DoF represents the user's movements, and the highest DoF is 6, which includes three rotational view direction changes, roll, yaw, and pitch, as well as three translational viewpoint movements along the x, y, and z axes. The limited DoF of the conventional captured-image-based VR lowers user's sense of reality.

Light field (LF), which can generate a view at a free viewpoint through a combination of light rays, is a suitable approach to support the freedom to change viewpoints. LF assumes a planar or spherical surface, and generates a view by combing light rays passing through the surface. In particular, the spherical LF system creates a 360-degree view through light rays incident from 360-degree direction. In the spherical LF, a viewpoint freely moves along the x, y, and z axes inside the sphere and changes view direction, and thus 6-DoF is supported. However, it is difficult to

acquire light rays for LF assuming a planar or spherical surface. In the case of spherical LF, a special equipment for rotating multiple cameras arranged in an arch shape is used to acquire light rays along a spherical surface. In order to cover a larger space, it is necessary to assume a larger plane or spherical surface. The larger the surface, the more difficult the light ray acquisition is.

3D LF consists of light rays acquired along the line, unlike conventional LF that assumes surfaces. A line instead of a plane and a circular structure instead of a spherical surface are used to construct 3D LF. It is easy to acquire light rays, which is acquired by moving the camera mounted on a camera slider and a dolly along the line. However, 3D LF cannot acquire vertical parallax because it obtains light rays at only one vertical point, and it causes distortion of the generated views. Assuming a larger structure does not significantly increase the difficulty of acquisition, but it increases the distortion of 3D LF view generation.

This paper aims to develop a free viewpoint VR system for large space based on 3D LF. In contrast to extending the structure in the existing method, it assumes a 3D LF Stack in which multiple 3D LFs are stacked in front and back. The proposed system is simple to obtain light rays and limits the distortion to a certain range. In addition, two 3D LF Stacks are arranged orthogonally to generate a 360-degree view at a free viewpoint. There are two challenges for the proposed system. First is the need to connect independent 3D LFs. The existing LF-based approach creates a view using a single LF, while the proposed system generates a view using four 3D LFs. This paper proposes two 3D LF connection methods and introduces appropriate usage methods according to various implementation environments.

Another is that 3D LF Stack still contains distortion, and the error is particularly noticeable as the viewpoint moves and the 3D LF that generates a view changes. This paper proposes a view generation method using a light ray set with epipolar geometry relationship in 3D LF stack.

Keywords : Light Field, Virtual Reality, Free viewpoint, View navigation, View exploration
Student Number : 2016-30216