Point Cloud 제작을 위한 딥러닝 기반 카메라 파라미터 추정

□ **원종수, 한종기** / 세종대학교

요 약

카메라 파라미터를 추정하는 것은 3D 복원, VR, 로봇 공학, 자율 주행 등에서 핵심적인 기술이다. 기존에는 두 개의 영상을 이용하여 Fundamental matrix를 추정하고 이를 이용하여 카메라 파라미터를 추정하는 방법을 사용하였다. Fundamental matrix를 추정하기 위해서는 두 개의 영상에서 특징점을 찾고 매칭을 하는 과정이 필요하며, RANSAC 알고리즘을 사용해야 하므로 시간이 오래 걸리는 단점이 있다. 하지만 딥러닝을 활용하여 카메라 파라미터를 추정할 경우 한 개의 영상으로도 카메라 파라미터를 얻을 수 있고, 빠른 속도로 꽤 정확한 값을 얻을 수 있다. 최근에는 딥러닝을 활용하여 카메라 파라미터를 추정하는 연구가 진행되고 있다. 본 기고문에서는 최근 딥러닝을 활용하여 카메라 파라미터를 추정하는 연구들을 설명하고 특징을 서술한다.

Ⅰ. 서 론

카메라 파라미터를 추정하는 것은 로봇 공학, 자율 주행, 3D 복원, VR 등의 분야에서 필수적인 과정이다. 로봇 공학 및 자율 주행 자동차는 카메라로 얻은 시각적인 정보를 이용하여 물리적인 행동을 수행하기 때문에 정확한 카메라 파라미터를 추정하는 것이 중요하고, 정밀한 3D 복원 및 VR 영상을 생성하기 위해서는 정확한 카메라 파라미터를 추정하는 것이 중요하다.

카메라 파라미터는 카메라로 얻은 2D 영상과 3D 월드 좌표간의 상관관계를 나타낸다. 핀홀 카메라 모델로 가정할 경우 다음 식과 같은 변환 관계를 얻을 수 있다.

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (1)$$

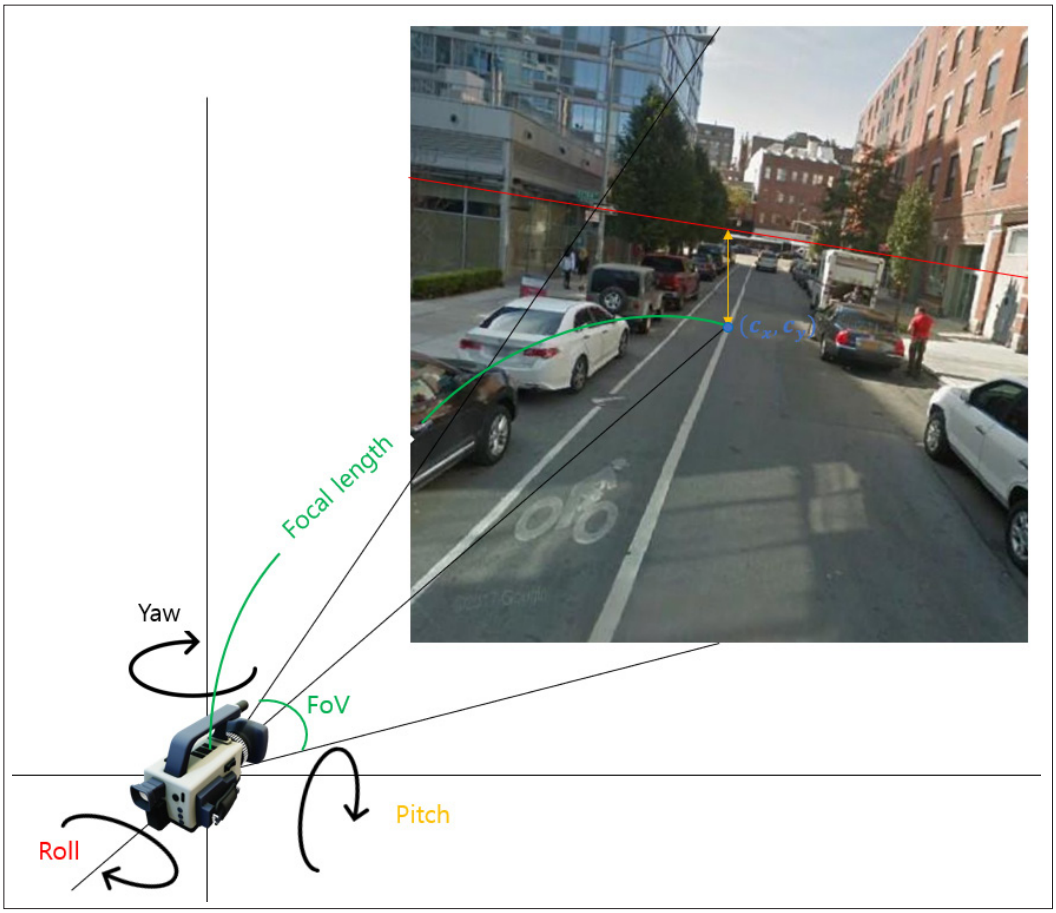
(X,Y,Z)는 월드 좌표계상의 3D 좌표이며, (x,y)는 2D 영상의 좌표이다. s는 스케일 펙터, f는 초점 거리, c는 주점, r은 회전 행렬, t는 평행이동 벡터를 나타낸다. 식 (1)에서 카메라 내부 파라미터와 외부 파라미터로 분류할 수

있는데, 내부 파라미터에는 초점 거리 f와 주점 c로 구성되어 있고, 외부 파라미터에는 회전 행렬 R 및 평행이동 벡터 t로 구성되어 있다. 〈그림 1〉에서와 같이 초점 거리는 시야각 FoV로 변환할 수 있으며, 회전 행렬의 경우, roll, tilt, yaw 혹은 horizon line으로 변환할 수 있다.

카메라 파라미터를 추정하기 위한 전통적인 방법은 Fundamental matrix를 활용하는 것이다. 비슷한 장면을 촬영한 두 개의 2D 영상에서 특징점을 추출하고 매칭쌍을 만든 후, 매칭쌍 사이의 관계를 Fundamental matrix로 표현한다. Fundamental matrix는 초점 거리를 활용하여 Essential matrix로 변환하고 Essential matrix를 이용하여 회전 행렬 R과 평행이동 벡터 t를 구한다. Fundamental matrix를 활용하는 기존의 방법은 정확하지만 RANSAC 알고리즘을 사용하기 때문에 시간이 오래 걸리는 단점이 있다.

최근 딥러닝을 활용한 카메라 파라미터 추정 방법이 연구되고 있다[1][2][4][5]. 딥러닝을 사용할 경우 아직까지는 알고리즘보다 정확한 파라미터를 추정하기 어려우며, 데이터 셋이 달라질 때마다 새로 학습을 해야 하는 단점이 존재한다. 하지만 알고리즘을 사용하는 것보다 빠르게 카메라 파라미터를 얻을 수 있으며, 여러 장이 아닌 한 장의 영상으로도 카메라 파라미터를 구할 수 있는 장점이 존재한다.

본 기고문에서는 최근 카메라 파라미터를 추정하는데



<그림 1> 카메라 내부 및 외부 파라미터

주로 연구되고 있는 딥러닝 기법에 대해 설명하려 한다. 2장에서 Fundamental matrix를 이용한 카메라 파라미터 추정 기법을 설명하고, 3장에서 딥러닝을 활용한 카메라 파라미터 추정 기법들을 설명한다. 4장에서는 딥러닝 실험 결과를 보이고 5장에서 결론을 짓는다.

Ⅱ. 컴퓨터 비전 이론에 기반한 카메라 파라미터 추정 기법

Depth 정보가 없는 2D 영상으로 3D 좌표를 추정하기는 복잡한 과정이다. 기존에는 두 장의 영상을 이용해서 카메라 파라미터를 추정하였는데, Fundamental matrix를 이용하는 것이다.

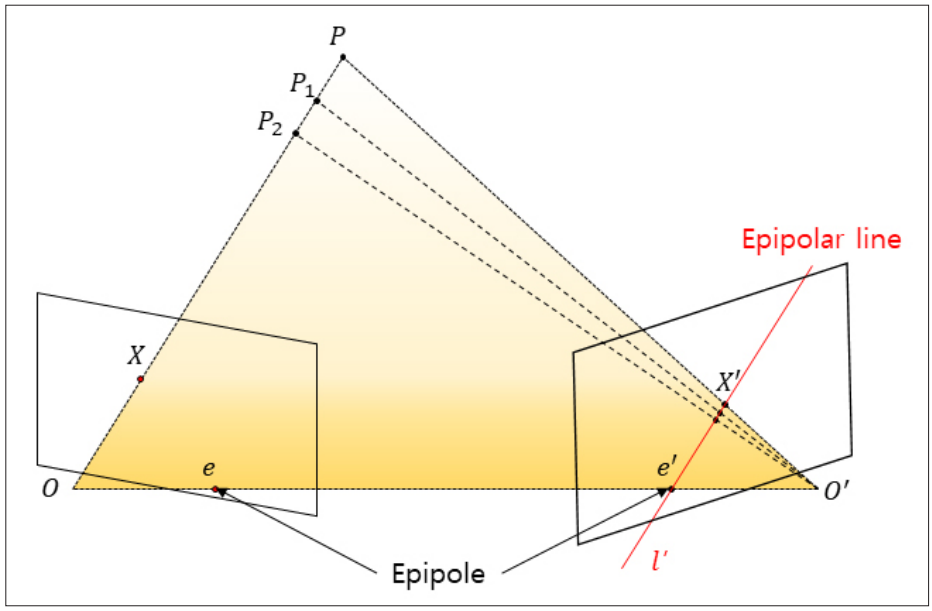
Fundamental matrix는 점에서 선으로의 변환 관계를 나타내는 행렬이며, 회전 행렬 및 평행이동 행렬로 구성되어 있다. 〈그림 2〉는 3D 좌표 P와 두 개의 영상에 투영된 점 X, X′을 기하학적으로 표현한 것이다. 3D 점 P를 카메라로 투영시켜 2D 점 X를 얻기 때문에, 2D 영상만을 이용해서는 3D 좌표가 P_1인지 P_2인지 P인지 특정할 수는 없다. 그렇지만 점 P가 직선 OX 위에 존재하는 것을 알 수 있다. 카메라 원점 O, O′을 잇는 직선이 영상과 만나는 점을 epipole이라 하며, epipole과 투영된 점 X′을 잇는 직선을 epipolar line이라 한다. 정규 좌표계 점 X와 epipolar line의 관계를 나타내는 행렬이 Essential matrix이고 다음 식과 같이 표현한다.

$$l' = EX \tag{2}$$

X′은 epipolar line 위에 존재하므로 다음과 같은 식을 만족한다.

$$X'^{\top} EX = 0 \tag{3}$$

Essential matrix가 정규 좌표계에서 점과 epipolar line의 관계를 나타냈다면, Fundamental matrix는 이미지 좌표계에서 점과 epipolar line의 관계를 나타낸다. 따라서 카메라 내부 파라미터 행렬을 K라고 할 때



<그림 2> Epipolar Geometry

Fundamental matrix F와 Essential matrix E는 다음과 같은 관계식을 만족한다.

$$F=(K^{-1})EK^{-1} \tag{4}$$

만약 카메라로 영상을 찍었을 경우, 초점 거리 정보가 함께 저장되기 때문에 카메라 내부 파라미터 행렬 K를 얻을 수 있다.

Fundamental matrix는 3×3의 크기를 가지고 있는데, Fundamental matrix의 rank가 2이기 때문에 역행렬이 존재하지 않으며, 동시에 Det(F)=0의 특징을 가진다. 즉 Det(F)=0을 만족하므로, 자유도가 1 감소하고, 동차좌표계를 사용하고 있기 때문에 자유도가 1 감소하여 7의 자유도를 가진다.

즉 다음 식과 같이 7개의 매칭쌍 ($X_1(x_1, y_1)$, $X_1'(x_1', y_1')$), (X_2, X_2'), (X_3, X_3'), (X_4, X_4'), (X_5, X_5'), (X_6, X_6'), (X_7, X_7')을 이용하여 Fundamental matrix를 구할 수 있다.

$$\begin{bmatrix} x_1x_1' & x_1y_1' & x_1 & y_1x_1' & y_1y_1' & y_1 & x_1' & y_1' & 1 \\ x_2x_2' & x_2y_2' & x_2 & y_2x_2' & y_2y_2' & y_2 & x_2' & y_2' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_7x_7' & x_7y_7' & x_7 & y_7x_7' & y_7y_7' & y_7 & x_7' & y_7' & 1 \end{bmatrix} \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \\ f_6 \\ f_7 \\ f_8 \\ f_9 \end{bmatrix} = 0 \tag{5}$$

식 (5)에서 Af=0을 만족시키는 해 f_1, f_2는 행렬 A를 특이값 분해를 사용하여 구한다. 행렬 A를 $A=U\Sigma V^T$로 특이값 분해 했을 때, f_1은 V^T의 마지막 열벡터, f_2는 V^T의 마지막에서 두 번째 열벡터이다. 그 후 Af=0을 만족시키는 $f=f_1+af_2$를 만들고 Det(f)=0 식을 이용하여 a를 구한다. a는 3개 이하로 구해지는데, RANSAC을 이용하여 inlier의 수가 가장 많은 f를 구한다.

위의 방법으로 Fundamental matrix를 구하면, 식 (4)를 이용하여 Essential matrix를 계산한다. Essential matrix는 평행이동 벡터 t와 회전 행렬 R의 외적으로 이루어져 있는데, 이를 행렬 곱 형태로 변형할 경우 다음 식을 만족한다.

$$E = t \times R = TR,\ T = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \tag{6}$$

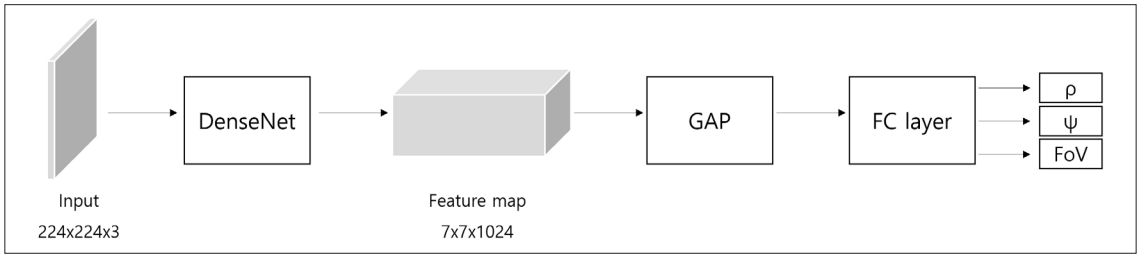
T는 반대칭 행렬로 이루어져 있기 때문에, Essential matrix를 특이값 분해하여 반대칭 행렬이 되도록 유도하는 과정을 통해 R과 t를 구할 수 있다.

$$E = U\Sigma V^T = (UZU^T)(UW^{-1}V^T) = (T)(R),$$
$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix},\ Z = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{7}$$

식 (7)은 동차좌표계를 사용했기 때문에 scale에 무관한 특징을 가지게 된다. 따라서 결과로 얻은 R, t는 scale에 무관하기 때문에 부호가 달라질 수 있어, 총 4가지의 경우가 나오게 된다. 4가지 경우 중 올바른 것을 선택하는 과정은 triangulation 과정에서 구한다.

Ⅲ. 딥러닝을 활용한 카메라 파라미터 추정 기법

딥러닝을 이용하여 카메라 파라미터를 추정하는 연구들은 입력으로 단일 영상만 들어가는지, 추가적인 정보가 들어가는지에 따라 두 가지로 분류할 수 있다. 첫 번째는 입력으로 단일 영상이 들어가는 연구로, 논문으로는 DeepFocal[8], DeepHorizon[2], Perceptual[1] 등이 있다. 이 세 논문 중 카메라 파라미터인 초점 거리, roll, pitch를 모두 추정하며, 실험 결과가 가장 뛰어난 Perceptual[1] 논문에 대해 Ⅲ-1에서 설명한다. 두 번째는 입력으로 단일 영상뿐만 아니라 기하학적인 정보가 추가로 들어가는 연구로, 논문으로는 GPNet[4], CTRL-C[5] 등이 있다. 이 중 가장 좋은 결과를 보여주는 CTRL-C[5] 논문에 대해 Ⅲ-2에서 설명한다.



<그림 3> Perceptual[1] 딥러닝 구조

1. 단일 영상을 입력으로 하는 연구

Perceptual[1] 논문에서 제안하는 딥러닝 모델은 입력으로 단일 영상이 들어가고, 출력으로 roll ψ, pitch θ, 초점 거리 f가 나온다. 회전 행렬 R은 roll, pitch, yaw로 표현이 가능한데, 논문에서 yaw φ는 추정하지 않는다. 기존의 Fundamental matrix로 카메라 파라미터를 추정할 때 두 개의 영상이 필요하며 기준이 되는 영상 혹은 좌표계가 존재하기 때문에, 이를 기준으로 yaw φ를 추정한다. 하지만 딥러닝을 이용하여 카메라 파라미터를 추정할 경우 한 개의 영상으로 카메라 파라미터를 추정하기 때문에 기준이 되는 영상 및 좌표계가 존재하지 않는다. 평행이동 벡터 또한 위와 같은 이유로 단일 영상에서는 구하기 어렵다. 따라서 단일 영상을 입력으로 하는 딥러닝 모델로 카메라 파라미터를 추정할 경우 yaw φ와 평행이동 벡터를 추정하지 않는다. 회전 행렬 R은 roll ψ, pitch θ만을 이용하여 다음과 같이 간소화된 식으로 나타낸다.

$$R=R(\psi)R(\theta) \tag{8}$$

〈그림 1〉에서 알 수 있듯이, roll ψ의 경우 수평선의 회전한 각도를 나타낸다. pitch θ의 경우 주점과 수평선 사이의 수직 거리 ρ로 변환하여 다음과 같은 관계식을 통해 나타낸다.

$$\rho = f \times tan\theta \times sin\psi \tag{9}$$

이때 영상의 상단과 하단의 좌표는 각각 1과 -1의 값을 가지게 된다.

딥러닝 모델이 영상에서 초점 거리보다 FoV를 더욱 정확하게 추정하기 때문에 FoV로 대체한다. 초점 거리 f는 영상의 높이를 h라고 할 때, 핀홀 카메라 모델에서 다음과 같은 식을 통해 FoV로 변환할 수 있다.

$$FoV = 2\tan^{-1}(\frac{h}{2f}) \tag{10}$$

즉 Perceptual[1] 논문에서 FoV, roll ψ, 주점과 수평선 사이의 수직 거리 ρ를 추정하고, 이를 변환하여 카메라 파라미터인 초점 거리 f, roll ψ, pitch θ를 추정한다.

Perceptual[1]에서는 〈그림 2〉와 같이 DenseNet[11]을 전이학습을 사용하여 카메라 파라미터를 추정하는데, 전이학습이란 이미지넷에서 이미지 분류 문제를 해결하기 위해 사용한 딥러닝 네트워크를 사용하여 다른 문제를 해결하는데 적용하여 학습을 진행하는 것이다. DenseNet에서 추출한 특징맵을 이용하여 추정하고자 하는 파라미터를 학습한다.

전이학습을 이용하면 학습이 빠르게 수행되며, 학습 데이터 셋이 적은 경우에도 과적합을 방지할 수 있는 장점이 존재한다. DeepFocal[8]에서는 AlexNet[9]을, DeepHorizon[2]에서는 GoogLeNet[10]을 전이학습을 진행하여 파라미터를 예측하였다. 초기에 딥러닝을 활용하여 카메라 파라미터를 구하는 방법으로 주로 전이학습이 사용되었다. 다음은 Perceptual[1]의 딥러닝 모델에 대

한 설명이다.

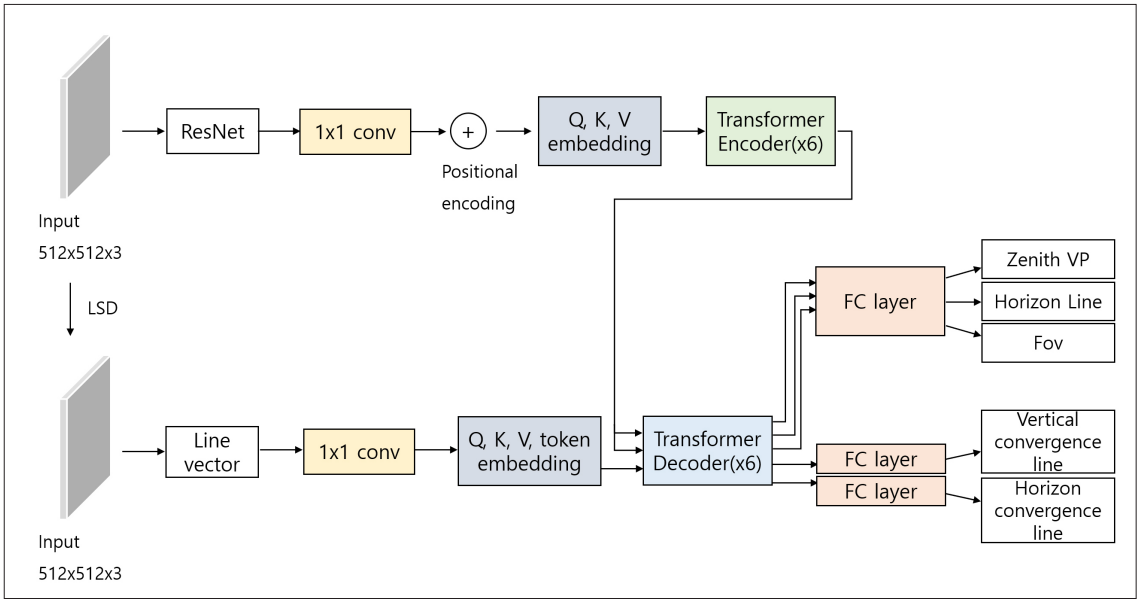
입력의 크기는 224×224×3이며, DenseNet[11]을 통과해 얻은 7×7×1024 크기의 특징맵을 global average pooling을 통해 1차원 벡터로 만들고 Fully connected layer를 거쳐 파라미터를 예측한다. Fully connected layer에서 소프트맥스 활성화 함수를 사용하여 세 개의 파라미터를 각각 256개의 빈으로 이산화하여 확률 분포를 출력한다. pitch를 나타내는 ρ와 roll ψ의 경우, 대부분의 학습 데이터에서 0 근처의 값을 가지고 있기 때문에, 0 근처에서 촘촘한 범위를 가지는 bin을 사용한다.

손실 함수로는 KL divergence를 사용한다. KL divergence는 GT와 딥러닝 모델이 예측한 값, 두 확률 분포 사이의 거리가 멀면 값이 커지고, 가까우면 값이 줄어드는 특징을 가진다. 즉 딥러닝 모델이 정답 값과 비슷하게 예측하는 방향으로 학습시킨다.

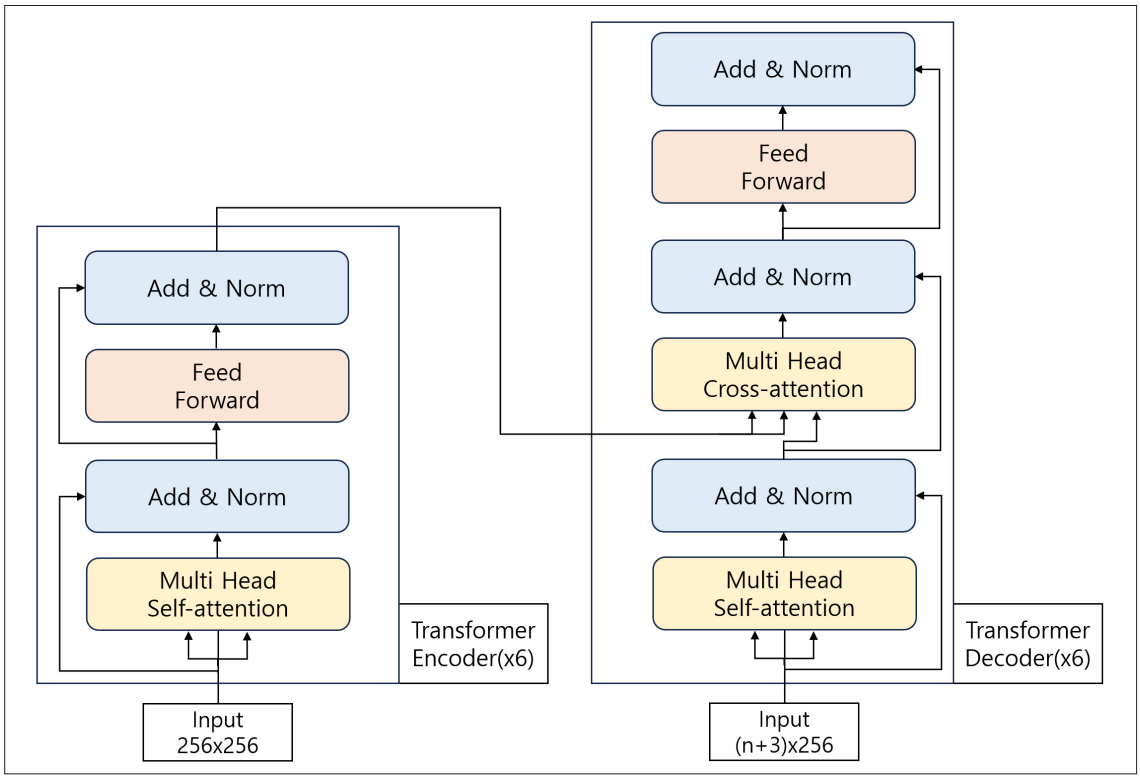
Perceptual[1]과 같이 전이학습을 이용하여 카메라 파라미터를 추정할 경우, 적은 수의 데이터로도 높은 성능을 낼 수 있다. 또한 알고리즘보다 카메라 파라미터를 빠르게 추정할 수 있으며, 실시간 적용 가능하다.

2. 단일 영상 및 기하학 정보를 입력으로 하는 연구

CTRL-C[5]는 Camera calibration TRansformer with Line-Classification의 줄임말로, 트랜스포머 딥러닝 구조를 사용하여 카메라 파라미터를 추정하는 연구이다. 추가로 라인 분류 작업을 수행하여 학습이 잘 되는지를 실험하였다. 논문에서 제안하는 딥러닝 모델은 입력으로 단일 영상과 LSD 알고리즘을 적용한 영상 두 개가 들어가며, 출력으로 Zenith VP, Horizon line, FoV가 나온다. Perceptual과 마찬가지로 하나의 영상이 입력으로 들어가므로 yaw φ와 평행이동 벡터를 제외한 나머지 파라미터 roll, pitch를 추정한다. 여기서 Zenith VP를 추정하는 이유는 up vector를 얻기 위함이다. up vector는 카메라의 윗 방향을 나타내는데, up vector를 통해 roll과 pitch를 얻을 수 있다. 추가로 Zenith VP와 Horizon line을 추정하는데 보조 역할을 하는 vertical and horizontal convergence line을 추정한다. CTRL-C[5] 논문에서 이러한 과정을 추가로 수행하였을 때 Zenith VP와 Horizon



<그림 4> CTRL-C[5] 딥러닝 구조



<그림 5> CTRL-C[5] 트랜스포머 인코더 및 디코더 구조

line을 더 잘 추정하는 결과를 얻음을 확인하였다.

LSD는 line segment detector의 약자로, 영상에서 지역적으로 생기는 직선들을 검출하는 알고리즘이다. 입력 영상에 LSD 알고리즘을 적용하여 추출한 직선, 즉 기하학적인 정보를 추가로 이용하여 카메라 파라미터를 추정한다.

CTRL-C에서 제안하는 딥러닝 네트워크의 구조는 비전 트랜스포머[3][12][13]에서 제안하는 네트워크 구조와 유사하다. 인코더와 디코더로 구성되어 있는 비전 트랜스포머 구조이며, 인코더에는 입력 영상의 특징이 들어가고 디코더에는 LSD 알고리즘에서 추출한 직선의 특징이 들어간다. 다음 내용은 인코더에 대한 설명이다.

입력 영상은 512×512×3의 크기를 가지며, ResNet 전이 학습을 통해 16×16×2048 크기의 특징맵을 얻는다. 1×1 컨벌루션을 통해 16×16×256 크기로 채널의 수를 줄여준 다음, flatten 하여 채널당 256개의 원소로 이루어진 벡터를 256개 생성한다. 256×256 크기의 행렬이 트랜스포머 인코더의 입력이 된다. 트랜스포머 인코더는 셀프 어텐션, skip connection, 정규화, FC layer로 이루어져 있는데 입력과 출력의 크기가 동일한 특징이 있다. CNN의 경우 컨벌루션 연산을 수행하기 때문에 지역적인 특징을 추출하게 되지만, 트랜스포머의 경우 셀프 어텐션 연산을 수행하기 때문에 순서 정보가 사라지며, 전역적인 특성을 추출하게 된다. 따라서 셀프 어텐션 연산을 수행하기 앞서 순서 정보를 추가해주기 위해 입력과 동일한 크기를 가지는 positional encoding을 디코더의 입력 행렬에 더해주는 과정을 거친다. 그 후 Query, Key, Value로 임베딩을 진행하는데, 논문에서 제안한 모델은 헤드가 8개인 멀티 헤드 어텐션이기 때문에 Query, Key, Value의 크기는 256×64를 가진다. 멀티 헤드 어텐션은 CNN의 필터가 늘어나면 여러 특징을 가지는 것과 같이 각 헤드마

다 특징을 가지기 때문에 여러 개의 헤드를 사용하며, 병렬로 연산할 수 있는 장점이 있다. Query, Key, Value는 초기에는 임의의 값으로 초기화되며, 학습을 진행하면서 값을 찾아나가게 된다. Query와 Key의 내적 연산을 통해 어텐션 스코어를 계산하게 되고, 소프트맥스 활성화 함수를 거쳐 나온 값을 Value와 곱하여 출력하는 셀프 어텐션 연산을 수행하게 된다. 이러한 과정을 거치게 되면, 각 이미지의 특징들은 거리에 상관없이 서로 얼마나 연관되어 있는지를 계산할 수 있다. 셀프 어텐션의 출력은 skip connection을 통해 학습할 때 기울기 값이 사라지는 문제를 해결하며, 정규화 과정을 통해 값이 커지는 것을 방지한다. 그 후 FC 레이어를 통과하여 비선형 특징을 가지게 한다. 이러한 인코더 레이어를 6번 반복하고 6번째 인코더 레이어에서 나온 Key와 Value를 디코더 레이어의 입력으로 한다. 다음 내용은 디코더에 대해 설명이다.

LSD 알고리즘을 적용하여 나온 n(n〈512)개의 직선을 디코더에 집어넣기 위해서는 숫자로 표현해야 하는데, 이를 위해 직선의 방정식을 활용한다. ax+by+c=0을 만족하는 직선 방정식 l은 직선의 양 끝점을 p_0, p_1이라 할 때 다음 식과 같이 두 점의 방향벡터를 외적하여 구할 수 있다.

$$l = \overrightarrow{p_0} \times \overrightarrow{p_1} \tag{11}$$

따라서 l은 (a,b,c) 3개의 값을 통해 나타낼 수 있다. 하지만 l과 -l은 같은 직선을 나타내므로 $l^{*}l^{T}$의 상삼각행렬인 $(a^2, ab, ac, b^2, bc, c^2)^T$ 를 s로 정의한다. n개의 s를 1×1 컨벌루션을 통해 채널을 늘려 n×256 크기의 행렬로 만든다. 위 방법을 통해 LSD 알고리즘으로 생성한 직선들의 특징을 추출하는데, 직선들은 임의의 순서로 설정되기 때문에 순서 정보는 필요하지 않아 positional encoding을 하지 않는다. 트랜스포머 디코더와 인코더에서 하는 연산은 비슷하다. 하지만 디코더에서는 파라미터를 추정하기 위한 토큰을 추가로 생성한다. 추가로 생성한 토큰은 추정하고자 하는 파라미터의 개수만큼 설정한다. 또한 크로스 어텐션 연산을 수행하는데, 셀프 어텐션에서는 입력을 임베딩하여 Query, Key, Value를 생성하고 내적 연산을 수행하는 것에 반해, 크로스 어텐션에서는 인코더에서 Key와 Value를, 디코더에서 Query를 가져와서 내적 연산을 수행하는 것에서 차이가 있다. 디코더에서 파라미터를 추정하기 위해 만든 토큰을 FC 레이어에 통과시켜 파라미터를 추정한다.

분류 문제로 파라미터를 추정하는 Perceptual[1]과 달리 회귀 문제로 파라미터를 추정하며, 3개 파라미터 각각 다른 손실 함수를 사용한다. FoV의 경우 식(10)을 이용하여 변환한 초점 거리의 MAE 손실 함수를 적용하였다. Zenith VP의 경우 실제값(GT)의 방향벡터를 z, 예측값의 방향벡터를 $\tilde{z}$라고 할 때, 다음 식과 같이 손실 함수를 계산한다.

$$1 - \left| \frac{z^T \tilde{z}}{\| z \| \, \| \tilde{z} \|} \right| \tag{12}$$

Horizon line의 경우 실제 수평선(GT)이 영상의 왼쪽 경계에서 만나는 교점의 y축 좌표를 b_l, 오른쪽 경계에서 만나는 교점의 y축 좌표를 b_r, 예측한 수평선이 영상의 왼쪽 경계에서 만나는 교점의 y축 좌표를 $\tilde{b_l}$, 오른쪽 경계에서 만나는 교점의 y축 좌표를 $\tilde{b_r}$이라 할 때, 다음 식과 같이 손실 함수를 계산한다.

$$\max(\| b_l - \tilde{b_l} \| , \| b_r - \tilde{b_r} \|) \tag{13}$$

Vertical and horizontal convergence line의 경우 LSD 알고리즘을 통해 추출한 n개의 선에 대해 각각 binary cross entropy 손실 함수를 사용한다.

CTRL-C[5]의 경우, 입력 영상으로 LSD 알고리즘을 적용하고 직선 방정식을 추출해야 하기 때문에, Perceptual[1]과 같은 전이학습을 이용하는 딥러닝 모델보다 학습이 오래 걸리는 단점이 존재하며, 트랜스포머의 특성상 전역적인 특징을 뽑기 때문에 데이터 셋이 많이 필요하다. 하지만 기존의 Fundamental matrix를 활용한 방법보다는 더 빠르게 처리할 수 있으며, 전이학습을 사용했을 때보다 일

<표 1> CTRL-C[5]에서 제공한 딥러닝 모델 성능 비교

Method	FoV(°)		Roll(°)		Pitch(°)		Up(°)		AUC(%)
	Mean	Median	Mean	Median	Mean	Median	Mean	Median	
DeepHorizon	x	x	1.78	1.67	2.76	2.12	3.58	3.01	80.29
Perceptual	4.61	3.89	0.96	0.66	2.39	1.78	2.73	2.13	80.40
GPNet	6.01	3.72	0.75	0.47	1.92	1.38	2.12	1.61	83.12
CTRL-C	3.59	2.72	0.66	0.53	1.58	1.31	1.80	1.52	87.29

반화를 잘하고, 더 높은 성능을 보이는 장점이 있다.

Ⅳ. 성능 비교 분석

〈표 1〉은 CTRL-C[5]에서 Google Street View 데이터셋으로 학습하고 테스트하여 결과를 표로 정리한 것이다. 딥러닝 모델이 예측한 카메라 파라미터인 FoV, Horizon line, Zenith VP를 이용하여 FoV, Roll, Pitch, Up vector를 추정하고 GT와의 차이를 표로 정리하였다. AUC는 horizon line이 정확히 추정됐는지를 나타내는 지표이다. 전이학습을 사용하여 학습을 진행한 DeepHorizon[2]과 Perceptual[1]보다 기하학적 정보를 사용한 GPNet[4]과 CTRL-C[5]의 성능이 더 높은 것을 알 수 있다. 하지만 LSD 알고리즘을 적용한 입력을 추가로 사용하며, 직선 방정식을 추출하는 과정에서 시간이 많이 들기 때문에, 비용이 많이 드는 단점이 존재한다.

Ⅴ. 결 론

본 기고문에서는 딥러닝을 이용하여 카메라 파라미터를 추정하는 논문들을 설명하고 특징을 분석하였다. 기존의 Fundamental matrix를 이용하여 카메라 파라미터를 추정할 경우, 제약되는 사항이 있기 때문에, 딥러닝을 통해 해결하는 방법이 계속해서 개발되어야 한다. 초기에는 CNN 특징맵을 기반으로 카메라 파라미터를 추정하는 방법을 사용하였다. 입력으로 단일 영상을 사용하였는데, 성능 향상의 한계가 존재하였고, 단일 영상뿐만 아니라 기하학적인 정보, LSD 알고리즘으로 추출한 선 정보를 입력으로 추가하여 성능을 상승시키는 추가 연구들이 진행되고 있다. LSD 알고리즘과 같은 카메라 파라미터를 추정하는데 도움이 되는 입력을 추가로 넣을 경우 더욱 학습이 잘 될 것이라 기대한다.

참 고 문 헌

[1] Y. Hold-Geoffroy et al, "A Perceptual Measure for Deep Single Image Camera Calibration," *Proceedings of Conference on Computer Vision and Pattern Recognition*, pp. 2354-2363, 2018.
[2] Scott Workman, Menghua Zhai, Nathan Jacobs, "Horizon Lines in the Wild," *Proceedings of British Machine Vision Conference (BMVC),* pp. 20.1-20.12, 2016.
[3] Xu, Yifan, et al. "Line segment detection using transformers without edges." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4257-4266, 2021.
[4] Lee, Jinwoo, et al, "Neural Geometric Parser for Single Image Camera Calibration," *Proceedings of ECCV,* pp. 541-557, 2020.
[5] Lee, Jinwoo, et al, "Ctrl-c: Camera calibration transformer with line-classification." *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16228-16237, 2021.
[6] SUN360 dataset, https://3dvision.princeton.edu/projects/2012/SUN360/
[7] Google Street View dataset, https://developers.google.com/maps/
[8] Workman, Scott, et al, "Deepfocal: A method for direct focal length estimation," *Proceedings of the 2015 IEEE International Conference on Image Processing (ICIP)*, pp. 1369-1373, 2015.
[9] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton, "Imagenet classification with deep convolutional neural networks," *Proceedings of the Advances in neural information processing systems*, pp. 1097-1105, 2012
[10] Szegedy, Christian, et al. "Going deeper with convolutions," *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 1-9, 2015.
[11] Huang, Gao, et al, "Densely connected convolutional networks," *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition,* pp. 4700-4708, 2017.
[12] Carion, Nicolas, et al, "End-toEnd Object Detection with Transformers," *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 213-229, 2020.
[13] Dosovitskiy, Alexey, et al, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *Proceedings of the 9th International Conference on Learning Representations*, 2021.

저 자 소 개

원 종 수

- 2018년 ~ 현재 : 세종대학교 전자정보통신공학과 학사과정
- ORCID : https://orcid.org/0009-0005-9398-9085
- 주관심분야 : 영상 신호처리, VR

한 종 기

- 1992년 : KAIST 전기및전자공학과 공학사
- 1994년 : KAIST 전기및전자공학과 공학석사
- 1999년 : KAIST 전기및전자공학과 공학박사
- 1999년 3월 ~ 2001년 8월 : 삼성전자 DM연구소 책임연구원
- 2001년 9월 ~ 현재 : 세종대학교 전자정보통신공학과 교수
- 2008년 9월 ~ 2009년 8월 : University California San Diego (UCSD) Visiting Scholar
- ORCID : https://orcid.org/0000-0002-5036-7199
- 주관심분야 : VR, 3D 영상신호 구현, 비디오 코덱, 영상 신호처리, 정보 압축, 방송 시스템