텍스트, 표지판이(가) 표시된 사진

자동 생성된 설명

**Vehicle Vision System**

Mid-Term Assignment: Stereo Matching

2022년 11월 4일

|  |  |
| --- | --- |
| 학과 | 전기컴퓨터공학 |
| 전공 | 미래형자동차 |
| 성명 | 장경준 |
| 학번 | 22221280 |

목차

[Introduction 3](#_Toc118473810)

[Stereo Matching 3](#_Toc118473811)

[Undistort Image 3](#_Toc118473812)

[Get Correspondence Points 3](#_Toc118473813)

[Get Fundamental Matrix 3](#_Toc118473814)

[Get Essential Matrix 3](#_Toc118473815)

[Essential Matrix Decomposition 3](#_Toc118473816)

[Rectify Image 3](#_Toc118473817)

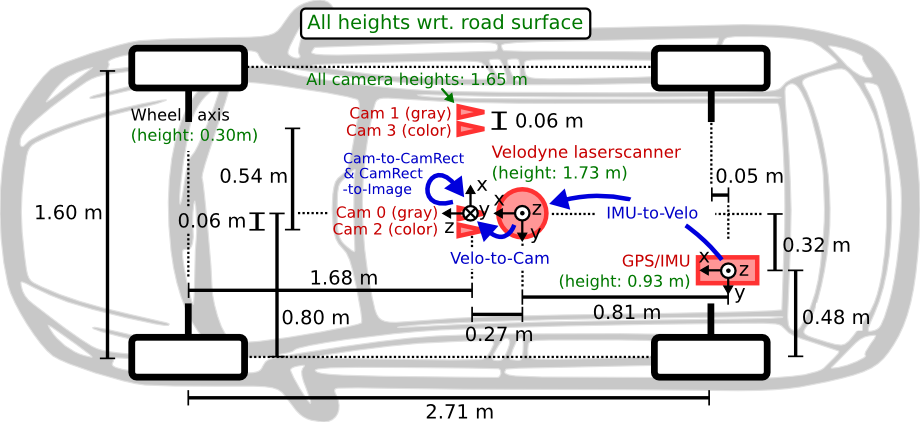
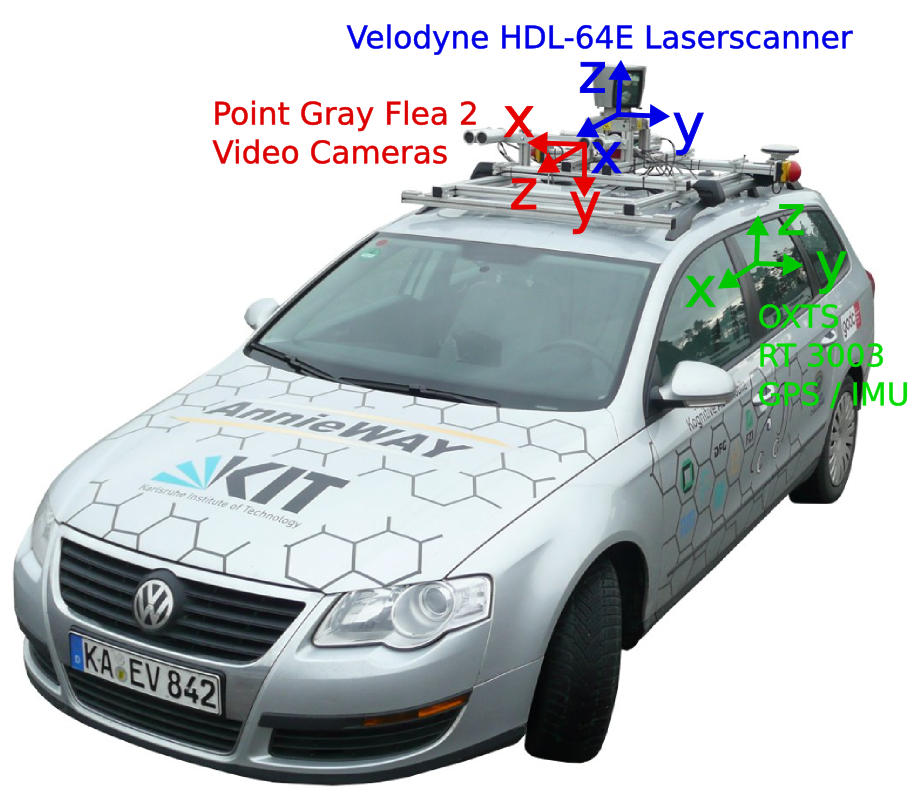
[Get Disparity Map 3](#_Toc118473818)

[Conclusion 3](#_Toc118473819)

# Introduction

스테레오 카메라가 설치된 차량인 KITTI 데이터셋의 raw data를 가지고 내부 파라미터를 이용하여 왜곡 보정부터 시작하여 Disparty를 구하는 과정까지 진행해본다. 중간 과정으로는 특징점 매칭과 이를 이용한 8Point Algorithm을 이용한 Fundamental Matrix 구하기, 이를 통해 Essential Matrix를 얻을 수 있고 분해하여 R, t를 구해낼 수 있다. 이 Rotation Matrix로 Image Recification을 하여 Epipolar Line의 평행을 맞춰 Disparity를 구하는 과정을 거쳐본다.

# Stereo Matching



보고서에 첨부되는 이미지는 KITTI Dataset CITY category의 raw data로 2011\_09\_26\_0048의 CAM 2, CAM 3 의 0000000000.png를 기준으로 한다.

카메라는 그림의 CAM 2와 CAM3로 RGB camera인 Point Grey Flea 2 2대를 baseline 0.54m 로 설치한 스테레오 카메라이다.

## [Undistort Image](http://www.ptgrey.com/products/flea2/flea2_firewire_camera.asp)





왜곡 보정 단계에서는 KITTI Dataset에서 주어진 파라미터를 이용하여 카메라의 내부 파라미터와 왜곡계수를 가지고 이미지를 왜곡 보정하는 과정을 거친다.

Forward warping 과정에서 발생하는 구멍이 생기는 문제를 해결하기 위해 backward warping으로 왜곡 보정을 수행한다.

렌즈 왜곡의 수학적 모델은 카메라 내부 파라미터의 영향이 제거된 normalized image plane에서 정의 되기 때문에

월드 좌표계의 한 점을 예로 들었을 때

스몰 x 노말라이즈 언디스토션된거 = 라지 X 월드좌표계 / Z

렌즈의 왜곡계수를 고려하여 렌즈계 왜곡 모델은 다음과 같다.

x normalized distortion = 식

ru = xnu2 + ynu2

우변의 첫번째항은 radial distortion 두번째항은 tangential distortion

다시 x pixel\_distor = f x

## Get Correspondence Points

피쳐는 키포인트와 디스크립터로 이루어져있다.

피쳐를 추출하는 방법으로 Harris Corner 방법을 이용하여 피쳐의 위치인 키포인트를 추출하고 이를 매칭하기 위해서는 피쳐의 특성을 알아야 하기에 sift를 이용하여 디스크립터를 구해 이미지 쌍에서의 피쳐들을 매칭할 수 있게 된다. 매칭한 결과를 토대로 인라이어와 아웃라이어를 나눠야하니 RANSAC을 통해 인라이어의 피쳐 매칭 쌍들만 남기게

한쌍의 이미지로만 추출하기엔 펀더멘탈 매트릭스를 구할때 부족하여 28장의 이미지를 모두 사용하여 매칭쌍들을 많이 만들어서 피쳐들을 이미지의 다양한 위치에 분포하도록 하였다.

Calibration 이미지는 이미지의 변화가 적고 체커보드 위주로 피쳐쌍이 뽑혀 이미지의 일부 위치에만 매칭쌍이 생겨 좋은 Fundamental Matrix를 얻을 수 없었고, 도시 이미지를 사용하였다.

## Get Fundamental Matrix

8 point algorithm을 이용하여 Fundamental Matrix를 구할 수 있다.

## Get Essential Matrix

위에서 얻은 Fundamental Matrix에 CAM 2와 CAM 3의 내부 파라미터를 이용하면 쉽게 Essential Matrix를 얻을 수 있다.

## Essential Matrix Decomposition

Essential Matrix를 SVD 분해하여 늘려서 translation의 skew matrix와 Rotation matrix로 분해할 수 있다.

Rotation matrix의 determinant는 1이 되어야 한다. -1이 나올 시 rotation 혹은 reflection이 된 것으로 -1을 곱하여 t와 R을 구할 수 있다. 적절한 t값으로 예상되는 것은 차량의 설치된 카메라의 베이스라인이 x축 방향으로 54cm 떨어져서 설치되어 있기에 x축의 값이 양수로(CAM 2 → CAM 3)나와야할 것으로 예상되어 이를 기준으로 t와 R을 결정하였다.

Estimate R\_rectification

강의노트 식 작성.

## Rectify Image

왜곡 보정 때와 마찬가지로 backward warping을 통해 Image rectification 해야 한다. Epipolar line을 평행하게 하기 위한 이미지 변환으로 CAM 2의 이미지와 CAM 3의 이미지에 대해 각각 수행하여야 한다. 각 Rectification을 위한 Rotation matrix를 구하고, backward warping 직접 구현에서 해결하지 못한 오류로 내장함수를 이용하여 Fundamental Matrix를 이용하여 rectification을 수행하였다.

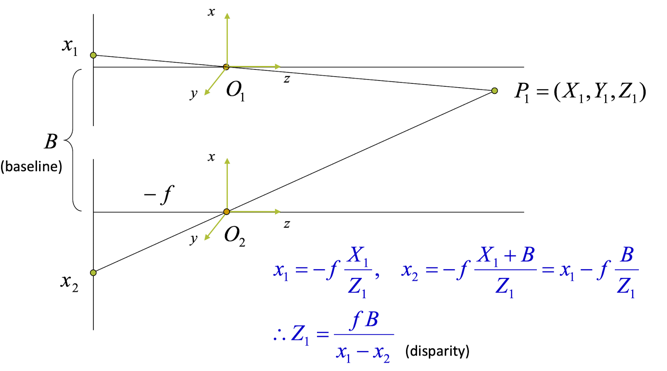
## Get Disparity Map

Disparity Map을 얻는 방법으로 Sum of Squared Differences(SSD)를 이용하였다.

# Conclusion

이론적으로 공부한 이미지 피쳐와 카메라 좌표계의 변환 내부 파라미터의 활용 등 직접 구현하면서 이해가 더 깊이 되는 경험이 되었다.

sift를 통한 디스크립터 익스트랙트를 정확히 이해하지 못하고 넘어가는 점이 아쉬웠고, 수업 시간에 배운 Harris Corner를 Feature Keypoint Detector로 사용하였는데 FAST 등과 같이 Feature Keypoint Detector를 교체해보며, 혹은 BRIEF와 같이 Feature Descriptor Extractor를 교체해가며 매칭쌍 뽑는 것을 바꿔가며 실험을 진행하여보면 좋을 것으로 생각된다.

* Stereo Matching
* 두 개의 영상을 가지고 어떻게 depth를 추정할 지.
* Basic Principle = Triangulation : 캘리브레이트 된 카메라를 통해서 대응점을 구하고, 두 레이어를 구한다. 그 다음 교점을 구한다.
* Point correspondence를 구하는 것이 핵심이고, 대응점을 잘 찾는 것이 중요하다.
* Disparity ( 시차 ) : 원본 영상과 타겟 영상 사이의 대응점의 픽셀 간 차이를 말하며, 이는 곧 깊이값으로 사용된다.
  + - 객체가 카메라와 가까이 있을수록, 카메라가 움직이면 대응점간 거리가 길어지고, 객체가 카메라와 멀수록 대응점간 거리가 짧다.
    - 즉 멀리 있는 물체의 시차는 작고, 가까이 있는 물체의 시차는 커지게 된다.
* Basic Stereo Derivation
* Stereo Reconstruction Process
  + Camera calibration

: 카메라 내,외부 파라미터를 구하여 regualize.

* + Image Rectification

: 이미지를 common plane에 맞추기. ( middlebury dataset은 이미 recify된 데이터셋을 공유한다. )

* + Disparity Computation
    - Matching Cost Computation

: d ( disparity ) 만큼 차이나는 픽셀 ( src, target ) 간의 cost ( intensity, gradient, census, SSD, SAD, NCC … )

* + - Cost Aggregation

: 이웃하는 영역간에는 같은 cost를 가질 수 있게끔, raw-level에서 구한 cost에 가중치 부여.

* + - Disparity Selection & Optimization

: 그 중 가장 해당 pixel과 잘 대응하는 disparity를 해당 픽셀의 disparity로 채택. ( Minimize cost )

* + - Disparity Refinement

: 깊이 영상에 맞게 화소 밝기값 조절 및 노이즈 제거.

* + Depth Estimation

: 최종적으로 구해진 disparity를 깊이로서 해당 영상의 깊이를 추정.

* Correspondence Problem

: Hard한 제약조건 대신에 Soft 하게 바꾸어서 corresponding points 를 식별하는데 사용한다.

* + Similarity, Uniqueness, Ordering, Disparity Gradient
  + World는 natural pattern이기 때문에, 이웃한 영역끼리 비슷한 cost를 가진다.
    - 이를 이용하여 image의 view에서 가장 잘 맞는 대응점을 찾는데,
    - 이때 region , 즉 window filtering을 통해 similar apperance를 찾는다.
  + Correlation : 각 픽셀에 대한 scanline에 대해서 ( epipolar line ) 각 픽셀 혹은 픽셀을 중심으로한 윈도우 내에서 가장 대응점간 가장 잘 매칭되는, 즉 cost가 minimum인 position을 선택하여 disparity로써 취한다.
  + 이때 SAD, SSD, NCC 등 다양한 kernel을 사용하여 convolution하여 값을 얻는다.
* Stereo Matching Algorithm
  + Local Method

: 영상의 픽셀에 대한 finite한 윈도우 내에서의 값들을 이용하여 대응점을 찾는다.

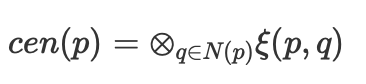
* + - 종류 : SAD, SSD, NCC, Census Transform, ….
  + Global Method

: 에너지 함수 ( Energy Function )을 기반으로, 깊이 영상을 추출한다.

* + - 에너지 함수를 최소화 하는, smoothness를 충족하는 경우를 찾음으로써 깊이 영상을 만들어낸다.
    - Local method와 달리 영상의 일정 부분 ( window ) 만 이용하는 것이 아닌 영상 전체에 대해서 비용을 구하고, 해당 값을 게속 사용하기 때문에 알고리즘이 복잡하지만, local method에 비해 매우 정확함.
    - 종류 : Belief Propagation, Graph Cut , …
  + Semi-Global Method

: 1D의 제약조건을 결합하여 global 2D smoothness term을 픽셀단위로 일치시키고, 근사화 한다는 아이디어.

* + - 종류 : Dynamic Programing, …
* Local stereo Matching
  + 각 픽셀에 위치에 대한 disparity를 구하기 위해, scanline에 위치한 모든 픽셀에 대해서 finite window 영역 내에서 raw-level cost를 구하고, 해당 cost에 대해 신뢰도를 높이기 위해 cost aggregation 방법으로 cost에 가중치를 부여하여, scane line중에 가장 cost 가 적을때의 disparity를 해당 픽셀의 깊이로 추정한다.
  + Matching Cost computation
    - Intensity ( no filtering ) : 각 픽셀에 대해서 RGB 각 채널에 대해서 Intensity를 구한다. 논문에서는 40 값이랑 비교하여 min값을 취하라고 기재되어있다.
    - Census Transform : 각 픽셀에 대해서 RGB 각 채널에 대해서 해당 위치의 Intensity와 해당 픽세을 중심으로 Census window 내의 값들을 비교하여 중심 픽셀의 값보다 작은 윈도우 내의 픽셀의 값은 1로, 크다면 0으로 vector를 구성하여 해당 vector와 target image의 vector사이의 hamming distance ( xor gate , 다르면 1 ) 을 계산하여 cost를 구한다. 정확도가 가장 높은 방법이였다.

텍스트이(가) 표시된 사진

자동 생성된 설명텍스트이(가) 표시된 사진

자동 생성된 설명

* + - Gradient : 각 픽셀에 대해서 RGB각 채널에 대해서 sobelX ( dx/d ) 인 x방향 미분 커널을 취하여 해당 픽셀의 x 미분값을 해당 픽셀의 cost로써 사용하였으나, 결과가 제대로 나오지 않았다.

텍스트, 시계이(가) 표시된 사진

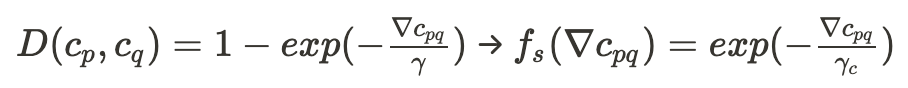
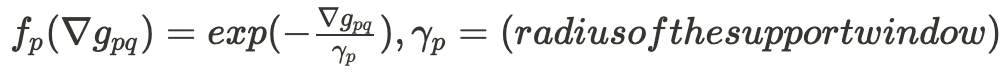
자동 생성된 설명

* + Cost Aggregation – Adaptive Support Weight

: Bilateral Filter와 유사하게 해당 픽셀에 대해서 위치 유사도와 밝기( 컬러 ) 유사도를 통해서 해당 cost에 가중치를 부여하여 통합화 하는 것.

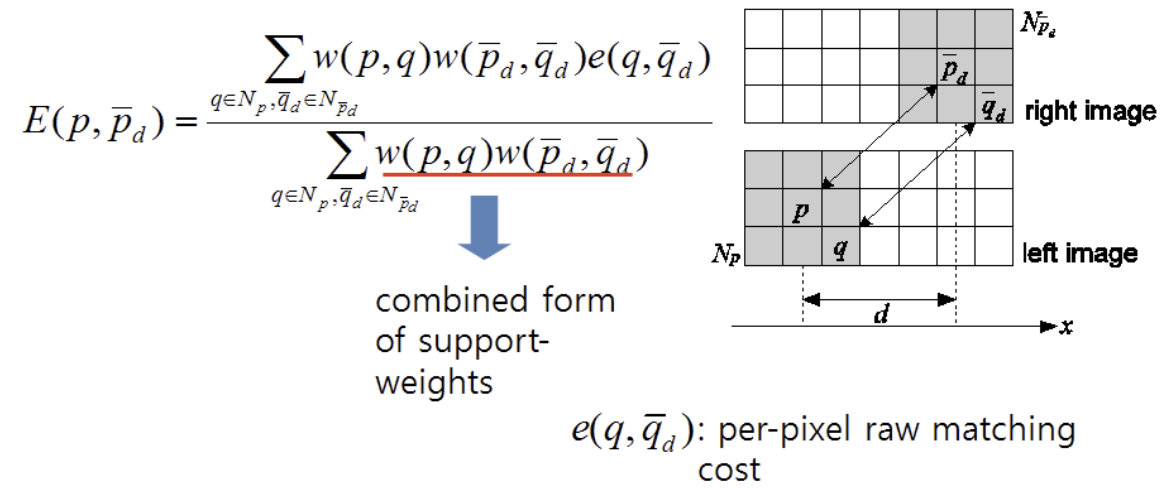
* + - Support weight of pixel : 텍스트이(가) 표시된 사진

      자동 생성된 설명 텍스트이(가) 표시된 사진

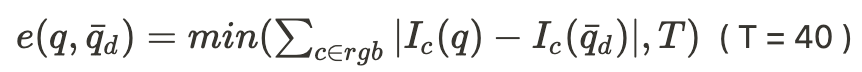
      자동 생성된 설명
    - Grouping by Similarity : 
    - Grouping by Proximity : 
    - Final From of support window : 텍스트이(가) 표시된 사진

      자동 생성된 설명
    - Dissimilarity Computation ( Matching cost )

Using the support-weights in both support window



* + - Pixel-based raw matching cost



* + Disparity Selection & Optimization
    - Winner Takes All method 텍스트, 클립아트이(가) 표시된 사진

      자동 생성된 설명
* Global Stereo Matching – Belief Propagation

: BP 알고리즘은 목표 함수를 MRF기반의 에너지 함수로 모델링 하는 방식으로, 에너지 함수가 최소화 하는 변위 값을 찾아 스테레오 정합을 수행.

: 모든 픽셀에 대해서 diaprity level에 따른 모든 cost값을 계산 하고 시작, iteration을 통해 해당 값을 mimize 하는 disparity를 찾는것.

* + 에너지 기반의 matching은 잡음에 취약한 local matching 방법의 단점을 보완.

: 영상의 위치에서 뿐만아니라 어느 레벨에서 가장 좋은 리스폰스가 나오는지 찾자.

* + Level q의 신뢰를 가지는 이웃한테, p가 그 이웃에 신뢰도에 얼마나 동의하는지에대한 message를 전파.

시계, 손목시계, 시간, 지붕이(가) 표시된 사진

자동 생성된 설명- 해당픽셀(p)에 대해서 메시지 전파시작전에계산된 cost와 diaparity level 차이 ( x<->x’ 거리 차이)에 대한 smoothness cost, 그리고 이전 iteration의 상태, 과거의 신뢰도 ( 이때까지 쌓아온 신뢰도 )를 q가 나에대해서 신뢰한 값 말고 나머지 신뢰도를 모두 더한 값이 가장 최소화되는 disparity level lp를 취한다.

- 해당 단계에서 계산한 메시지는 다음 노드로 전파되며, 왼쪽에서 오른쪽으로, 오른쪽에서 왼쪽으로, 위에서 아래로, 아래에서 위로 전파하는 과정이 iteration수 만큼 반복되면, 각 픽셀들은 matches disparity level ( min cost ) 를 알게되고, 해당 값을 disparity로 취하게 된다.

- Disparity 레벨을 함수 시작전에 정하고 시작하기 때문에, 초기에 cost initialize 하는데 시간이소모된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

* + - 각 노드들은 왼쪽에서, 오른쪽에서, 위에서, 밑에서 오는 메시지들을 알고있어야 하므로, 왜냐하면 다음 iteration term에 해당 값을 기억하고 사용해야함. 한 픽셀은 최대 5개의 값을 저장하고있어야한다. ( 내 자신의 raw-level cost 포함. )

1. **주요 구현 코드 설명**

주어진 과제는 다음과 같다.

OpenCV 내장 함수를 사용하지 않고,

1. Local Stereo Matching ( various matching cost computation ) .
2. Global Stereo Matching ( Belief Propagation, Graph cut )

텍스트이(가) 표시된 사진

자동 생성된 설명

먼저 주어진 2문제를 각각 확인할 수 있도록 메인 함수에서 cin으로 문제 번호를 받고, while문을 통해서 계속 문제를 확인할 수 있도록 하였다. 문제 번호에 해당하는 문제의 결과 이미지가 imshow함수를 통해서 보여지게된다.

이미지는 2003 dataset teddy, cones와 2005 dataset raindeer와 art를 사용하였다.

사용자가 문제 번호 입력 후 이미지 번호를 입력하여 해당하는 이미지에 대한결과를 확인할 수 있다.

Display 및 Stereo Matching 결과를 수행하는 간단한 함수는 main 함수에서 작성하였고, local stereo matching, global stereo matching task별로 header를 만들어서 include하여 사용하였다.

(1) Local Stereo Matching

텍스트이(가) 표시된 사진

자동 생성된 설명

Main함수에서 1번입력시에 실행되는 local stereo matching부분이다.

Local stereo match 클래스 선언시에 파라미터 4개를 입력받으며 각각 scan line ragne, adaptive support window sieze, filtering type, fitering dimension 이다.

Type에는 0,1,2가 들어갈 수 있으며 순서대로 intensity, gradient, census 방법이다.

프로세싱에 시간이 다소 소모되기 때문에 clock 함수를 이용하여 sec 단위로 시간을 측정하였다.

결과는 파라미터 계수들과 이미지 이름으로 저장된다.

Local stereo matching은 아래와 같은 순서대로 진행되며, adaptive support weight로 cost aggregation시에는 코스트를 그때마다 계산해줘야하므로, 3-a와 3-b는 동시에 진행된다.

텍스트이(가) 표시된 사진

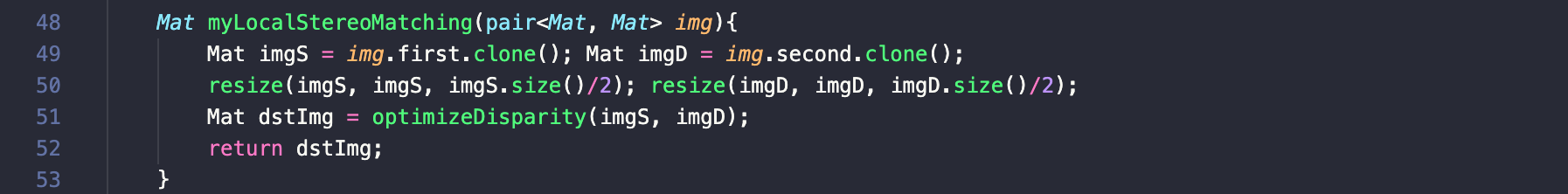
자동 생성된 설명

LocalstereoMatch class의 선언 부분이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

동장시마다 필요한 마라미터는 global 변수로써 선언을 하였고, main함수에서 클래스 선언시에 파라미터들을 인자로 넣어준다.



Main 함수에서 불려지는 호출함수이며, 해당 결과로 optimize 된 depth map을 결과로 반환한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

추정된 disparity map을 밝기값 normalize와 median filter를 이용하여 salt and pepper noise와 같이 픽셀 중간중간에 튀는 값을 보정해준다. makeDisparityMap 함수를 통해서 raw-level의 disparity 결과와 해당 disparity mat을 완성하면서 가장 큰 값을 Mat, int pair로써 리턴한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Local stereo matching 알고리즘 수행 순서중에 compute disparity의 부분이며, 메인으로 돌아가는 함수이기 때문에, 각 열에 대해 disparity를 추정할 때마다 raw열을 출력하여 진행중임을 알려준다.

함수내에서 disparity를 구하는 computeDisparity는 ASW와 여러가지 cost compute 방법을 이용해서 해당 픽셀에 대해 최적의 disparity를 추정하는 함수 ( cost가 가장 작은 disparity ) 이다. 각 픽셀별로 best match disparity를 찾으며, 해당 영상 전체에 대해서 가장 큰 disparity값을 Mat와 함께 pair쌍으로 반환한다. ( normalize 하기 위함 )

텍스트이(가) 표시된 사진

자동 생성된 설명

makeDisparity 함수에서 영상내의 각 픽셀에 대해 접근 하였다면, 이제 computeDisparity 함수에서는 해당 픽셀에 대해서 scan line range내의 이웃한 모든 픽셀들을 검수하여 최적의 disparity 값을 도출한다. 이때 dissimilarity를 ASW와 cost compute로 구하고, 해당 dissimilarity를 가장 최소화하는 disparity 값 ( 대응점간 픽셀 거리 )를 해당 픽셀의 최적의 disaprity로써 리턴한다.

이 함수에서는 다방향 시차를 계산하기 위해서 scan line의 range를 -부터 +까지 지정하였는데, 이 부분에서 오류가 생길 수 있음을 감안하지 않고 작성하였다. 실제 영상은 오른쪽으로 이동했음에도 불구하고, 왼쪽에도 비슷한 요소가 있었다면, 해당 값을 min으로 오판하여 disparity를 잘못 추정했을 가능성이 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명  
calcDissimilaritys는 compute diaparity 단계에서 cost compute와 cost aggregation을 동시에 수행하는 함수이다.

Adaptive Support weight의 window 사이즈 내에서, 각 src image와 dst image에 대해서 similarity ( 밝기 유사도 )와 proximity ( 거리 유사도 ) 를 구하여 각각 src와 dst에 대해서 weight를 구하고, dissimilarity compute 공식에 따라 각 픽셀에 대한 cost를 구하여 weighted sum 을 통해 energy를 구하여 해당 값을 dissimialrity로써 return 한다.

텍스트이(가) 표시된 사진

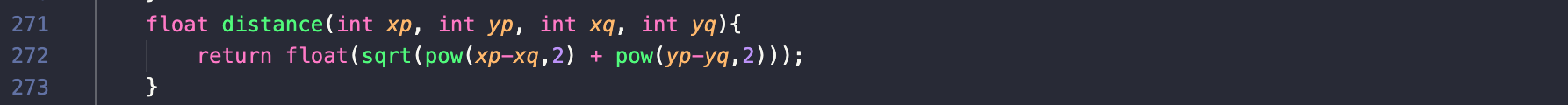
자동 생성된 설명

Similarity를 구하기 위해 dirrenceIntensity함수에 인자로 color vector를 넣어서 차이의 절대값을 반환 받았고,

텍스트이(가) 표시된 사진

자동 생성된 설명

Proximity를 구하기 위해 distance 함수로 픽셀 position간의 unclidean distance를 구하였다.



해당 weight들은 weight 함수를 통해서 하나의 weight로 통합되며, gamma c와 gamma r은 논문에서 얘기하는 값을 넣었다.



calcdissimilarity 함수에서는 3가지 cost compute 방법을 global 변수 filter type에 따라 각각 동작하게 하였다.

Intensity와 gradient는 각각 cost가 int형으로 반환되게 하여, cost vector에는 src와 target의 차이의 절대값을 cost로써 push하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

census 일 경우, 각 픽셀에 대해서 윈도우 영역내의 값들을 0,1로 바꾸고 해당 값들을 벡터화 하여 hamming window를 통해 cost를 계산해야한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Census cost를 구하기 위해 각 픽셀에대해서 census vector를 구하였으며, 해당 벡터를 구한 후 light ssd 방법을 통해서 최종 cost를 결정하였다.

RGB이미지 이기 때문에 먼저 중심 픽셀의 intensity 값을 구한다 ( 주변 픽셀과 비교하기 위해서 )

텍스트이(가) 표시된 사진

자동 생성된 설명

그 후 census filtering 영역내에의 이웃한 픽셀들의 화소를 모두 비교하여 0,1 로 구성된 벡터를 반환한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

그후 src vector와 dst vector를 input으로 받아 각 인덱스에 대한 값이 다를 때 ++ 연산을 통해서 최종 cost를 산출한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Disparity optimize 단계에서 사용하는 median filter이다. 윈도우 내의 값들을 sorting하여 중간값을 해당 픽셀의 값으로 대체하는 함수이다.

(2) Global Stereo Match

텍스트이(가) 표시된 사진

자동 생성된 설명

Global Stereo match도 local 방법과 마찬가지로 사용자가 파라미터를 인자로 입력하여 클래스를 선언한다. Iteration 수와 windo dimension 그리고 global matching algorithm type 값을 넣는다.

0은 belief propagation 알고리즘이고, 1 은 graph cut 알고리즘이다. Graph cut은 과제에 주어진 open library를 사용해 보려고 하였으나, 코드 분석에 어려움이 있어 구현하지 못하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Global Stereo Match 함수도 local stereo match 와 동일하게 선언부에서 사용자로부터 사용할 parameter를 입력받고 global 변수를 설정한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Local stereo match 방법과 마찬가지로 raw-level disparity map을 optimize를 거친 뒤에 최종 결과로 도출한다.

Belief Propagation 구현에는 10번 레퍼런스 블로그를 참고하여 작성하였다. 알고리즘 구성에 어려움이 있어 해당 블로그를 보고 BP 알고리즘의 동작방법을 구성할 수 있었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

먼저 BP는 iteration 수 만큼 message를 상, 하, 좌, 우로 전달하는 방식으로 신뢰도를 수렴해나가는 방식이다. 따라서 iteration을 돌기전, 각 픽셀에대해서 사전에 지정한 disparity level에 따라 모든 cost를 cache영역에 저장을 하고 시작해야한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

모든 픽셀에대한 cost를 저장하기 위해서, 각 픽셀에 대해서 상하좌우, 그리고 원래의 픽셀의 고유한 cost ( disparity level에 따른 ) 를 저장해야하므로, 2중 배열을 가지는 struct를 선언하여, 해당 struct를 가지는 vector를 영상으로써 모든 영상을 구성하는 픽셀에 대해 총 5 \* disparity level 의 데이터를 가지고 있게끔 initialize하였다.

상하좌우에서 온 메시지는 아직 생성된 것이 없으므로, 초기에는 고유의 disparity level에 따른 cost를 계산하여 0번 index에 저장하였다. 이때 cost계산에는 SSD방식을 사용하였다.

데이터 고유의 cost는 0번, 오른쪽에서 온 메시지는 1, 왼쪽에서 온 메시지는 2, 밑에서 올라온 메시지는 3, 위에서 내려온 메시지는 4 인덱스에 각각 disparity level 별로 cost를 저장할 수 있게끔 하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

getSSD함수는 초기 cost 계산시에 딱 한번 사용하는 함수이다. 초기 cost 설정을 잘해야지 오류가 수렴하지 않는 것을 알 수 있다. SSD는 각 src, dst 픽셀의 윈도우 내의 픽셀에 대해서 Intensity 값의 차의 제곱을 weighted sum 한 결과이다. RGB이미지를 사용했기 때문에 각 채널에 대해서 구한 뒤 3으로 나눈 결과를 cost로써 return 하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Iteration시에 각 상하좌우 방향으로 메시지를 전파할때 돌아가는 함수이다. Direction에 따라서 움직여야 할 방향이 다르므로 switch case문을 통해서 방향을 설정해주었다, 왼쪽에서 오른쪽으로, 위에서 밑으로 가능 방향은 순방향이라서 어려운 부분이 없지만, 오른쪽에서 왼쪽으로, 밑에서 위로는 역방향이기때문에 인덱스에 신경을 써야한다. 또한 좌우 방향은 opencv pixel access 방식과 동일하게 각 row마다 각 column마다 검사를 시행하지만, 상하 방향은 각 colum마다 각 row마다 검사를 해야하므로, 인덱스가 혼동되지 않게 조심해야한다.

각 방향에서 수행되는 minimumBelief 함수는 BP 알고리즘의 message를 전달하는 식을 구하는 과정이다.

이때 p->q로 가는데 q한테 p가 q의 판단에 얼만큼 동의하는지를 알려주는 과정이므로, 인덱스와 방향을 잘 고려해야 값이 제대로 잘 도출된다.

각 q의 disparity label에 대해서 수행해야하므로 diaprity levelㅇㅔ 대한 for 문이 돌아가고, p가진 신뢰도를 알려줘야하므로, p의 dispriaty label에 대한 for문이 한 번 더 돌아간다.

텍스트이(가) 표시된 사진

자동 생성된 설명

해당 lp for 반복문을 통해서 min 값을 도출해야 q->p로 보내는 신뢰도를 구할 수 있다.

텍스트, 화이트보드이(가) 표시된 사진

자동 생성된 설명

Data matching cost는 Initial 단계에서 구한 disparity level 별 cost이고, smoothness cost는 disparity level 차이간의 cost를 주기 위함으로 disparity level 차이의 절대값과, filtering window dimension의 크기를 비교하여 더 작은 값에 lamda값을 구하여 사용하였다 ( 논문에서 제시하는 lamda 값 20 ). 그리고 가장 마지막 term이 중요한데, 이는 과거의 상태에 대한 정보를 더함으로써 현재 나의 동의정도에 신뢰성을 높이는 방식이다. 이때 과거에 q가 p( 나 )한테 보여줬던 신뢰도는 배제되어야하므로, 해당 픽셀에 저장된 해당 direction의 값은 빼고 summation 해줘야한다.

또한 새로운 minimum belief 가 계산되었다면, 현재 cache에 저장된 영상 전체의 방향별, disparity level 별 cost를 갱싱해줘야한다. 이때 해당 메시지 전파가 어디서 어디로 이어지고 있는지를 잘 고려하여 인덱스를 지정해야한다.

왼쪽에서 오른쪽, 위에서 밑으로 가능 방향은 해당 direction에서 +1인 ( 오른쪽에서 왼쪽, 밑에서 위 ) 방향의 데이터 위치에 값을 갱신시켜야하고 ( q입장에서 봤을때, 왼쪽에서 오른쪽으로 전파되는 메시지는 왼쪽에서 들어온 메시지 이므로 ), 또한 새로 갱신될 데이터의 위치도 vector 공간상에서 인덱스를 조절해야하기 때문에 좌우로 들어온 메시지는 x position에 값을 더하고 빼고, 상하로 들어온 메시지는 y position에 값을 더하고 뺀다.

이런식으로 영상전체에 대해 신뢰도를 업데이트 하면서 각 iteration이 돌 때마다 best disaparity를 찾아줘야한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

각 disparity level 별로, 상하좌우, raw-level cost를 모두 summation 계산하여 cost가 가장 적은 disparity label 을 해당 픽셀의 disparity로 추정한다.

이때 해당 값은 normalize하여 저장해준다.

Iteration이 모두 반복되면 자동적으로 가장 잘 찾아진 disparity가 disparMat에 저장되어 해당 결과를 optimzie하면최종 deptmap을 얻을 수 있다.

1. **실행 결과 및 분석**

사용한 데이터 셋

|  |  |  |  |
| --- | --- | --- | --- |
| Teddy ( 2003 ) | | Cones ( 2003 ) | |
|  |  |  |  |
| Art ( 2005 ) | | Reindeer ( 2005 ) | |
|  |  |  |  |

컴파일 후 실행하면 다음과 같은 문구들이 출력되며, 문제번호를 입력하면 해당 문제를 해결하게 된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. Local Stereo Matching ( various matching cost computation ) .

|  |  |  |  |
| --- | --- | --- | --- |
| **Teddy (original size), Cost compute == Intensity** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 200 | 11 | “ | 6 min |
| **Cones (original size), Cost compute == gradient ( x )** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 200 | 11 | 7 | ? sec |
| **Teddy (original size), costcompute == census** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 200 | 11 | 7 | 1.5 hour |
| **Teddy (original size), costcompute == census** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 200 | 15 | 7 | 2 hour |

먼저 Teddy dataset으로 local stereo matching 방법에서 compute cost 방법을 3가지로 바꿔가며 테스트 해보았다. 몇번의 테스트를 통해 scane line rage를 적절하게 설정해야함을 알았다. 해당 값을 너무 작게 설정해 버리면, 멀리 있는 물체에 대한 disparity는 잘 추정 되지만, 가까이있는 물체는 그러지 않았다.

이유는 탐색 길이가너무 짧기 때문에, 우측 target 영상에서 source pixel 에 대응하는 점을 제대로 찾지 못햇기 때문이다. 해당 값은 몇번의 반복적인 테스트를 통해서 실험적으로 최상의 값을 설정해야함을 알았다.

또한 Intensity 방법으로 cost를 구하는 방법이 제일 빨리 끝났는데, 이는 다른 Filtering 방법과 달리 해당 픽셀에 대한 값만을 이용하기 때문에, filtering 시간이 단축되어 빨리 진행할 수 있었다.

이는 evaluation 방법을 통해서 정확한 지표로써 결과를 분석해야 제대로 구현한 것인지 확일할 수 있을것 같은데, 시간이 부족하여 evaluation process를 구현하지 못했다.

|  |  |  |
| --- | --- | --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 |  | 텍스트, 하얀색, 이미지, 군중이(가) 표시된 사진  자동 생성된 설명 |
| Intensity cost | Ground Truth | Census cost |

눈으로만 확인했을때는, census cost를 구하여 disparity를 추정한 결과가 ground truth와 더욱 가까운것을알 수 있다. 하지만 개구리 인형 머리 위에 화분 이나,곰인형 뒤의 배경등이 제대로 추정되지 않은 것을 확인할 수 있다. Intensiy로 구한 결과는 census에 반해 정확도가 객체 경계 부분에서 정확도가 더욱 떨어지는 것을 확인할 수 있다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Cones (original size), Cost compute == Intensity** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 80 | 11 | “ | 1576sec |
| **Cones (original size), Cost compute == intensity** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 160 | 11 | “ | 1092 sec |
| **Cones (resize), costcompute == census** | | | |
|  | | | |
| Scan line range | ASW window size | Filter Dimension | Processing Time |
| 80 | 11 | 7 | 1068 sec |

Census transform 방식으로 raw level cost를 구하여 disparity를 추정하는 것은, census transform에도 또 filtering 비용이 들어가기 때문에, 세개의 cost copute 방법 중에서 가장 오래걸리며, 또 이미지 사이즈가 클수록 더 오래걸린다. 따라서 census transform 의경우에만 size를 반으로 줄여서 사용하였는데, 이로인해 경게 부분이 불명확하고, 픽셀이 줄어들엇기 때문에, 정확도가 떨어지는 결과를 얻은 것을 볼 수 있다.

|  |  |  |
| --- | --- | --- |
| 자연, 비이(가) 표시된 사진  자동 생성된 설명 |  | 텍스트이(가) 표시된 사진  자동 생성된 설명 |
| Intensity cost | Ground Truth | Census cost |

눈으로만 확인했을때, census transform으로 얻은 결과는 resize를 햇기 때문에 오브젝트의 경계가 뭉뚝하지만, depth는 대체적으로 잘 추정된 것을 볼 수 있다. 단순히 intensity만 cost로 고려한 결과는 census의 결과에 비해 다소 노이즈가 많이 생성되었으며, median filtering을 한 후에도 튀는 값들이 제거되지 않은 결과를 볼 수 있다. 이는 이웃 픽셀을 고려하지 않고, cost를 계산하여 smoothness가 없는 결과를 도출한 것이라 볼 수도 있다. 이러한 결과로 census transform은 노이즈에 강인하다는 특징을 확인할 수 있다.

ASW cost aggregation으로 이러한 단점을 최대한 보완하지만, asw window 를 벗어나는 범위에 해당하는 smooth 영역 혹은 noisy 영역은 깊이 추정이 잘 되지 않는 것을 확인하였다.

이러한 단점을 보완하기 위해, 영상에서의 한 부분만이 아닌 전체 영상의 cost를 고려하여 streo matching을 수행하는 global stereo matching 중 belief propagation 방법을 구현한 결과는 아래에 있다.

1. Global Stereo Matching ( Belief Propagation, Graph cut )

|  |  |  |  |
| --- | --- | --- | --- |
| **Cones (original size)** | | | |
|  | | | |
| Disparity Level | Iteration | Filter Dimension | Processing Time |
| 50 | 20 | 7 | 459 sec |
| **Cones (original size)** | | | |
|  | | | |
| Disparity Level | Iteration | Filter Dimension | Processing Time |
| 50 | 40 | 7 | 919 sec |
| **Cones (original size)** | | | |
|  | | | |
| Disparity Level | Iteration | Filter Dimension | Processing Time |
| 50 | 60 | 9 | 1402 sec |

BP algorithm을 이용하여 stereo matching을 수행한 결과이다. Local method에서의 scan line range와 비슷한 역할을 하는 disparity level을 잘 조절해줘야 가까운 object 에 대해서도 깊이 추정을 원할하게 할 수 있다.

BP method의 경우 iteration 횟수를 늘릴수록, 초기 cost값을 계산하는 filtering winodw의 영역을 늘릴 수록 정확도가 향상된 이미지를 얻을 수 있었다. 하지만 이러한 결과도 정량적인 evaluation process를 거친 것이 아니기에 추가적인 구현이 필요하다.

Bp method는 local method에 비해서 시간대비, computation cost대비 훨씬 뛰어난 성능을 보였다.

하지만 global method는 초기에 모든 disparity level에 대해서 cost를 계산을 하고,belief propagation을 수행하기 때문에, 초기 cost initialize 단계가 가장 중요함을 알았다.

강의노트에는 data matching cost를 어떤 것을 써야할지에 대한 언급이없어 SSD를 이용하여 data cost를 계산하였다.

초기 값이 잘 생성되지 않으면, 토기 옆 삼각뿔 검정색 부분 처럼 계속 의미없는 min 값을 가지고 있음을 알 수 있다. 해당영역을 포함하여 영상전역에 검정색으로 채워지지 않는 부분은 iteration 회수가 늘어날 수록 점점 채워져가는 것을 볼 수 있다.

Iteration을 반복할 때마다 중간 과정을 imshow 함수로 계속 확인하였는데, 처음에는 부정확하고 많이 비어있던 cost들이 점점 정답에 수렴하는 것을 확인할 수 있었다.

|  |  |  |
| --- | --- | --- |
| 자연, 떼, 비, 그룹이(가) 표시된 사진  자동 생성된 설명 | 티피이(가) 표시된 사진  자동 생성된 설명 | 떼, 자연, 그룹, 이미지이(가) 표시된 사진  자동 생성된 설명 |
| Iteration = 20 | Ground Truth | Iteration = 60 |

Ground truth와 비교하였을 때, 컵에꽂힌 빨대 부분과 삼각 뿔들이 겹쳐지는 영역은 제대로 깊이가 추정되지 않은 것을 확인할 수 있다. 이는 이웃과비슷한 영역이 매우 적고, 빨대가 위쪽 아래쪽, 왼쪽 오른쪽 정방향으로 꽂혀있는게아니고, 대각선으로 꽂혀있기때문에, 상하좌우에서 메시지를 전달받는 BP 알고리즘으로는제대로 추정하지 못했을 것이라고 예상된다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Teddy (original size)** | | | |
|  | | | |
| Disparity Level | Iteration | Filter Dimension | Processing Time |
| 70 | 20 | 9 | 934 sec |
| **Teddy (original size)** | | | |
|  | | | |
| Disparity Level | Iteration | Filter Dimension | Processing Time |
| 70 | 40 | 9 | 919 sec |
| **Teddy (original size)** | | | |
|  | | | |
| Disparity Level | Iteration | Filter Dimension | Processing Time |
| 70 | 80 | 9 | 1402 sec |

결과 영상 좌측에 보이는 흰색 띄는 disparity 차이라고 예측된다. Src 이미지와 target 이미지에서 왼쪽부분은잘리 기때문에 저런 현상이 나타난것으로 보인다. 여러 논문 혹은 참조한 코드에서는저런 부분을 그냥 0으로 padding 시켰지만, 정확히 어던 부분이 matching 되지 않았는지를 확인하고 싶어서 그대로 display될 수 있게 하였다. 이와 마찬가지로 추정된 depth map의 우측부분도 disparity가 제대로 추정되지 않은 것을 볼 수 있는데, 이는 source 이미지에서 우측 부부에 대한 정보가 없기 때문에, 제대로 추정하지 못한 것으로 보인다. 이러한 문제는 pixel 단위에서가 아닌, 인공지능 모델을 활용하여 데이터를 학습시켜 모델을 사용해야 할 것으로 보인다.

|  |  |  |
| --- | --- | --- |
| 텍스트, 하얀색, 이미지, 군중이(가) 표시된 사진  자동 생성된 설명 |  | 떼, 비행, 자연, 실외이(가) 표시된 사진  자동 생성된 설명 |
| Local ( ASW, Census ) | Ground Truth | Global ( BP, iteration = 80 ) |

마지막으로 local method로 얻은 결과중 가장 잘 나온 census transform으로 cost를 계산한 방법과, BP algorithm으로 iteration 을 80번 수행하여 얻은 영상을 비교해 보았다.

물론 시간적 측면이나 정확도 면에서는 global method가 훨씬 뛰어나지만, 구현 측면에서는 아무래도 pixel 단위로 filtering을 수행하는 방법이기 때문에, 구현이 덜 복잡하였다.

Stereo matching 의 여러가지 방법을 픽셀단위로 구현하여 분석해보았다. 실제로 픽셀단위로 구현하는데 사용하는 방법들이나, 요령들은 원래 영상처리 방법과 크게 다른 것이 없었지만, 접근하는 방법이 생각보다 복잡하여 처음에 algorithm을 구성하는데에 시간이 많이 걸렸다.

또한 과제의 나머지 부분인 global method graph cut을 이용한 방법과 stereo matched된 결과와 grount truth를 비교하는 Evaluation process를 구현하지 못하여 각 방법들에 대해 정량적인 평가를 하지 못하였다.

1. **참고 문헌**
2. **intensity and gradient -based stereo matching using hierarchical Gaussian basis functions**

<https://ieeexplore.ieee.org/document/730551>

1. **Census Filtering Based Stereomatching under Varying Radiometric Conditions**

<https://www.sciencedirect.com/science/article/pii/S1877050915021377/pdf?md5=c6ce4bc9766a9adc463e196f9320e58c&pid=1-s2.0-S1877050915021377-main.pdf>

1. **Belief Propagation를 적용한 스테레오 정합과 영역 기반 정합 알고리즘의 정확성 비교**

<https://www.koreascience.or.kr/article/CFKO201114258942422.pdf>

1. **Stereo Matching Using Belief Propagation**

<http://www.jiansun.org/papers/Stereo_PAMI.pdf>

1. **Local stereo matching with adaptive shape support window based cost aggregation**

<https://www.osapublishing.org/ao/fulltext.cfm?uri=ao-53-29-6885&id=302045>

1. **Adaptive support-weight approach for correspondence search**

[https://ieeexplore.ieee.org/abstract/document/1597121?figureId=fig8#fig8](https://ieeexplore.ieee.org/abstract/document/1597121?figureId=fig8" \l "fig8)

1. **Stereo Matching With Fusing Adaptive Support Weights**

<https://ieeexplore.ieee.org/document/8712528>

1. **Development of stereo matching algorithm based on sum of absolute RGB color differences and gradient matching**

[Development\_of\_stereo\_matching\_algorithm\_based\_on\_.pdf](https://s3-us-west-2.amazonaws.com/secure.notion-static.com/150ecec2-848f-42e2-b8be-f68e0fb33d27/Development_of_stereo_matching_algorithm_based_on_.pdf)

1. **Stereo Matching 16-385 Computer Vision (Kris Kitani) Carnegie Mellon University**

<https://www.cs.cmu.edu/~16385/s17/Slides/13.2_Stereo_Matching.pdf>

1. **LOOPY BELIEF PROPAGATION, MARKOV RANDOM FIELD, STEREO VISION**

[http://nghiaho.com/?page\_id=1366#LBP](http://nghiaho.com/?page_id=1366" \l "LBP)