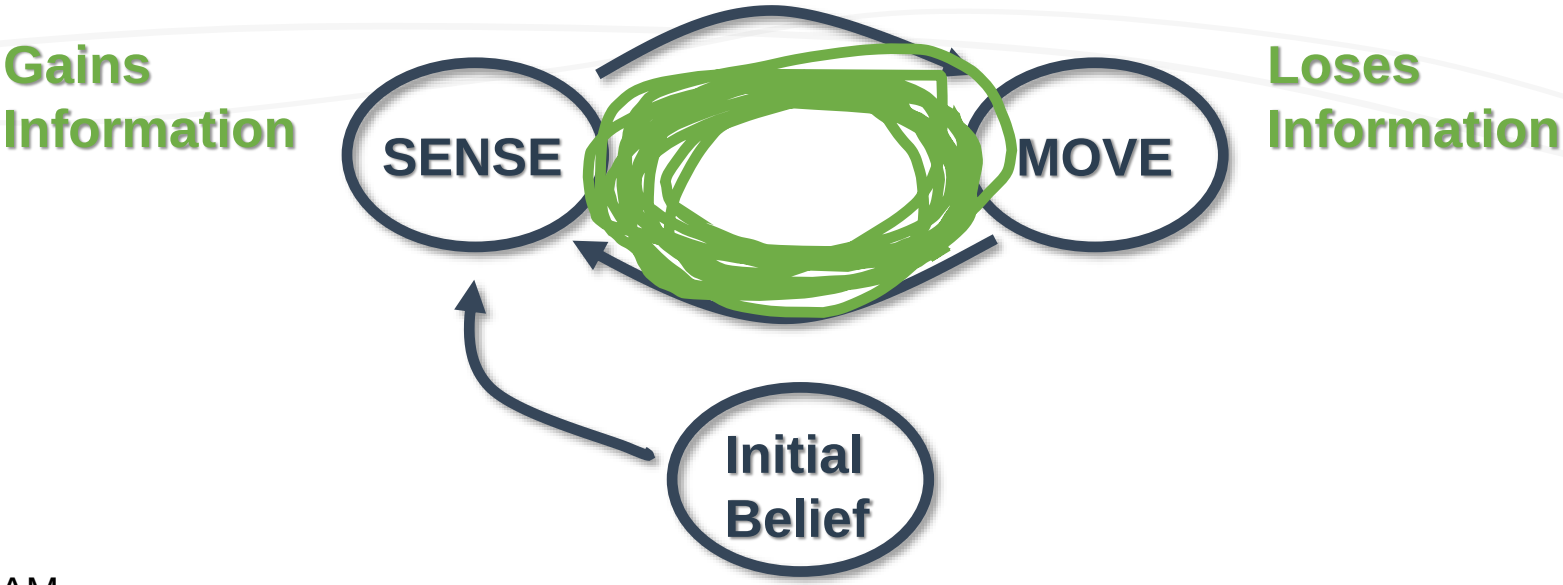
RAFSet Motion Estimation

전현호

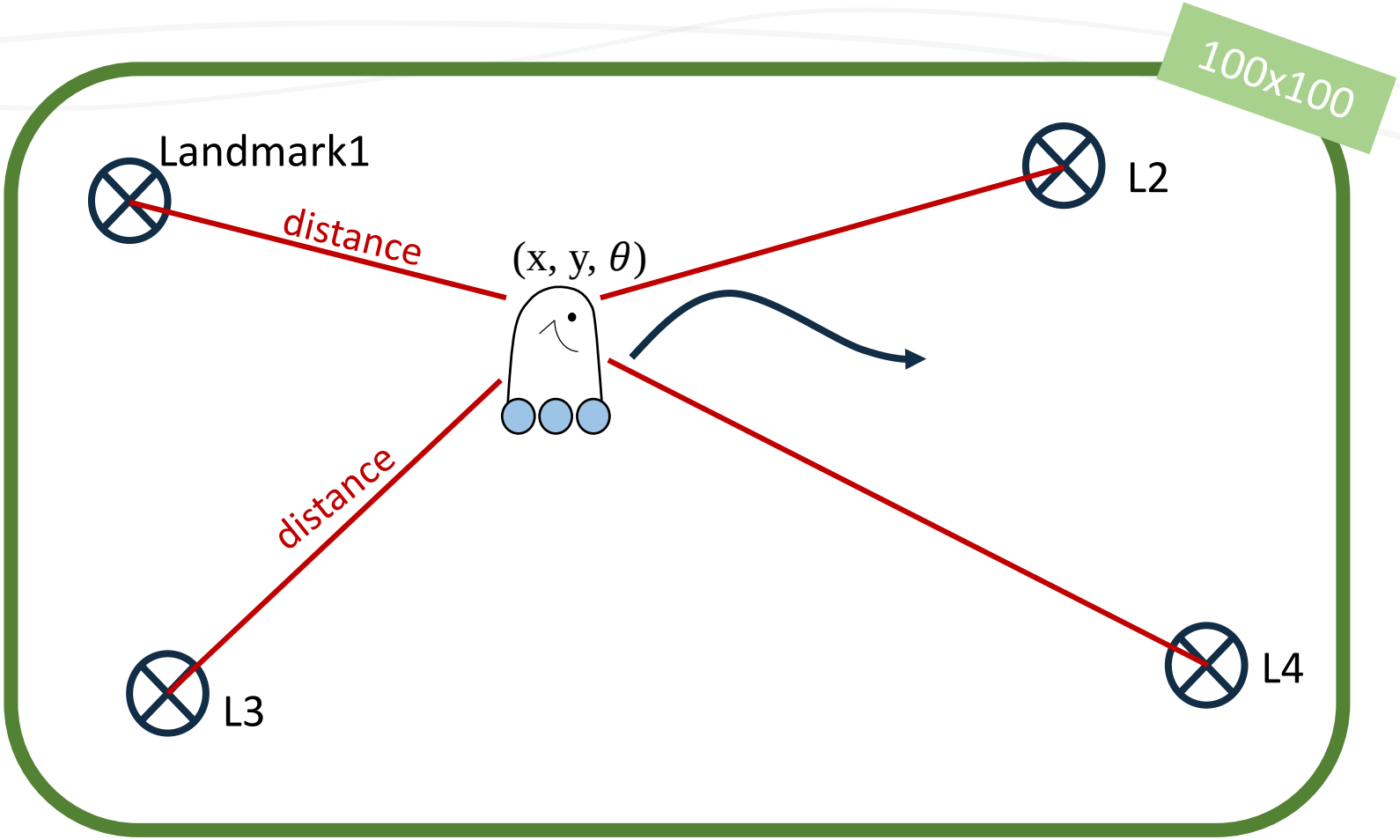
Contents

Intro : SLAM

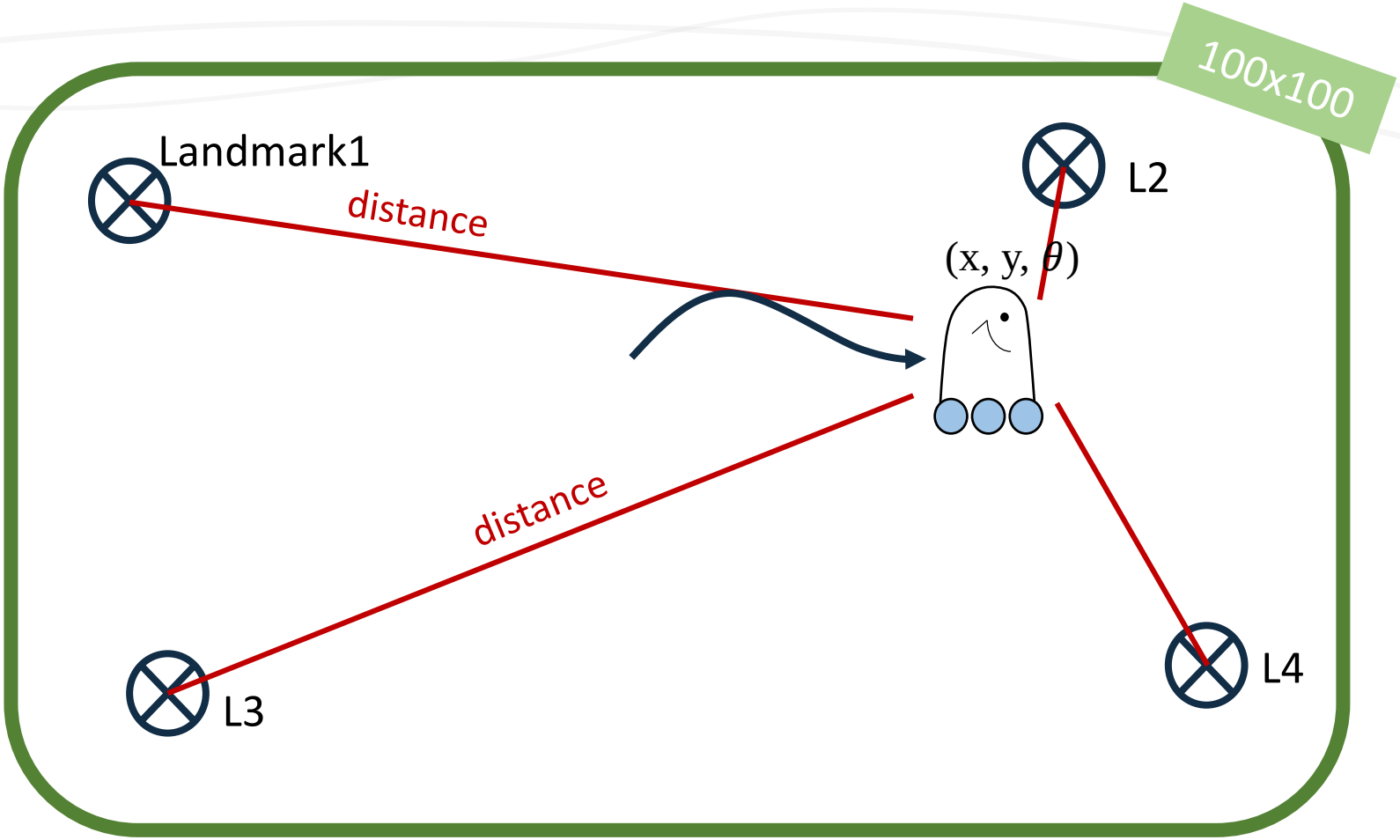


- SLAM
  - 이동 로봇이 자율 주행하기 위해 필수적으로 필요한 기술로서 자신의 위치 추정(Localization) 및 주변의 지도 작성(Mapping)을 동시에 수행한다.
  - Applications : 실내, 우주, 바다 속, 지하, 자율 주행
- 요구되는 센서
  - 거리 센서, 비전 센서(라이다, 레이더, 초음파, 스테레오카메라, 모노카메라 등등)
  - 모션 추정 센서 (엔코더, IMU, GPU 등등)

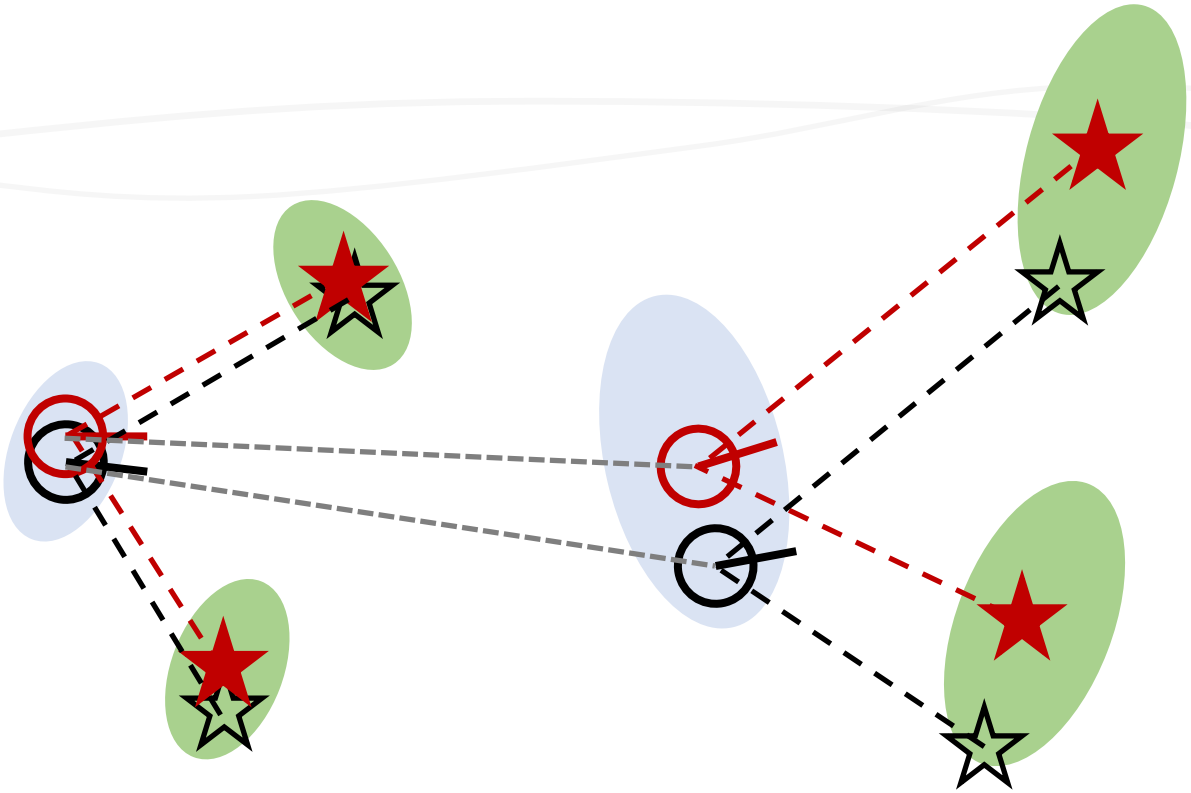
Intro : Localization



Intro : Localization



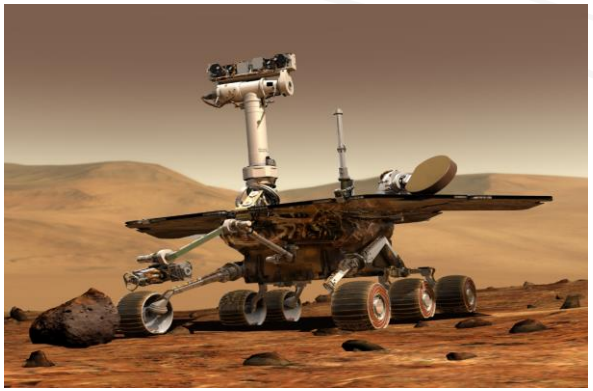
Intro : SLAM



	Robot	Feature
True		
Estimate		

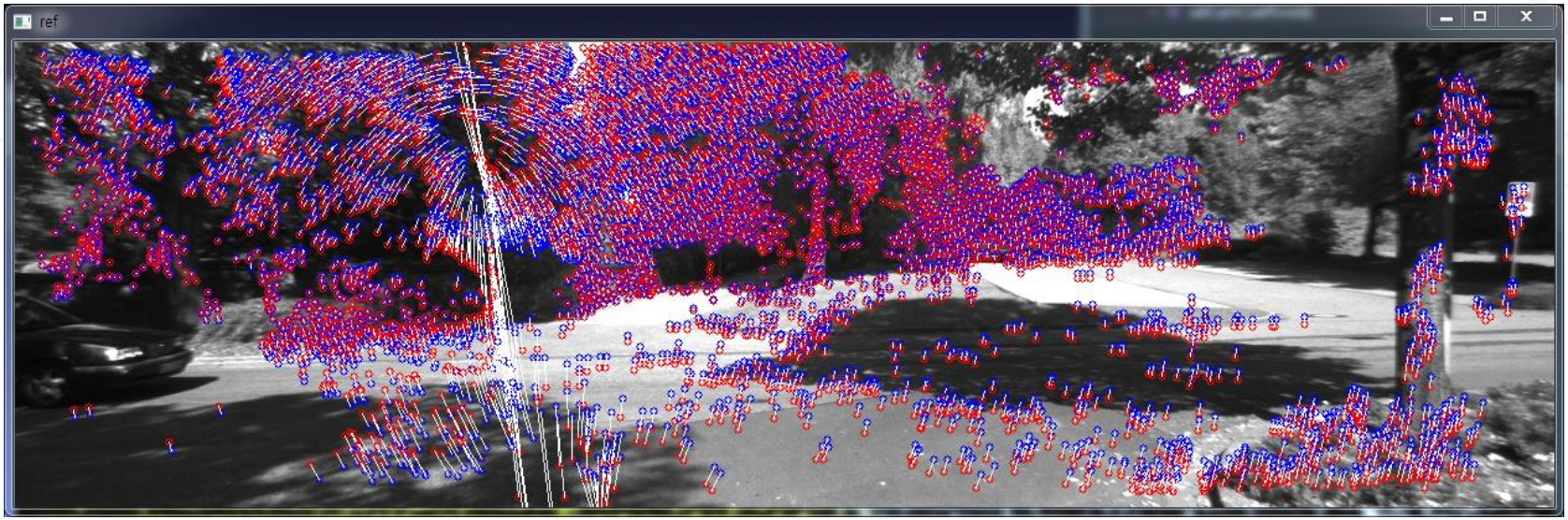
- 문제점
  - 맵이 없는 상태에서 시작함. 따라서 랜드마크를 스스로 결정하며 자신의 모션과 관측 정보를 통해 위치 추정 및 지도 작성을 수행해야 하는 문제점 발생 (chicken and egg problem)
  - 정확한 관측과 정확한 모션 정보가 요구됨

Intro : SLAM



- 문제점
  - 엔코더 정보의 경우 미끄러짐, 마찰, 체결 문제 발생하는 오차에 영향을 받음
  - 거리 센서나 비전 센서의 경우 이러한 영향을 받지 않음
  - 또한 대부분의 로봇의 경우 동작을 위해 영상 정보를 획득하는 것이 일반적임
  - 따라서 영상 기반의 모션 추정을 흔히 사용함

Intro : SLAM



- 영상 기반의 모션 추정
  - 2D-2D : 영상 정보만을 이용하여 모션 추정, 스케일 정보를 얻을 수 없음
  - 3D-2D : LiDAR, RGB-D 등 깊이, 거리 정보를 얻을 수 있는 센서를 함께 사용
  - 방법 : 두 프레임 사이의 특징 쌍과 해당 특징의 3차원 좌표를 가지고 있는 경우 수식적으로 두 프레임 사이의 [R|t]를 구할 수 있음
  - 문제점 : 영상 내 잡음, 폐색, 조명 변화 등으로 인한 Optical flow(OF), Matching의 실패로 인하여 부정확한 특징 쌍이 구해질 수 있음 -> RAF 고안

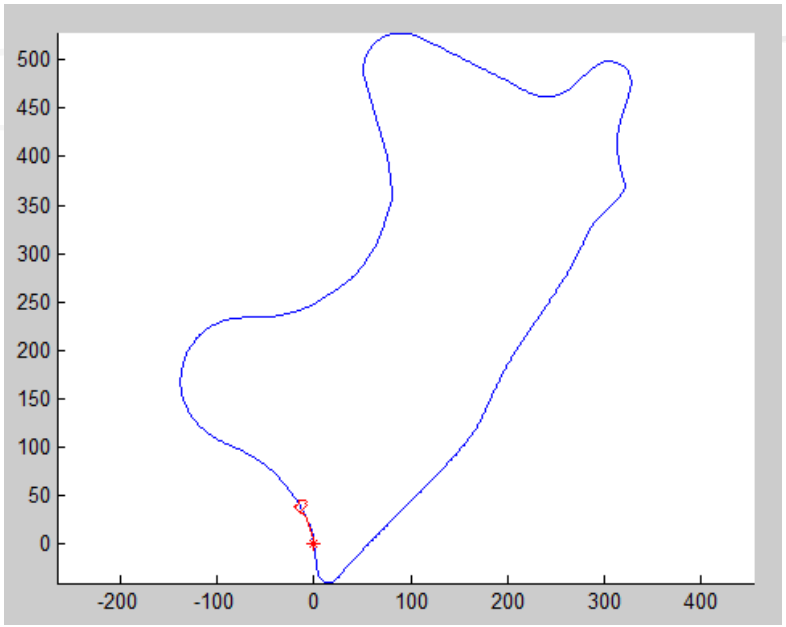
RAFSet Motion Estimation

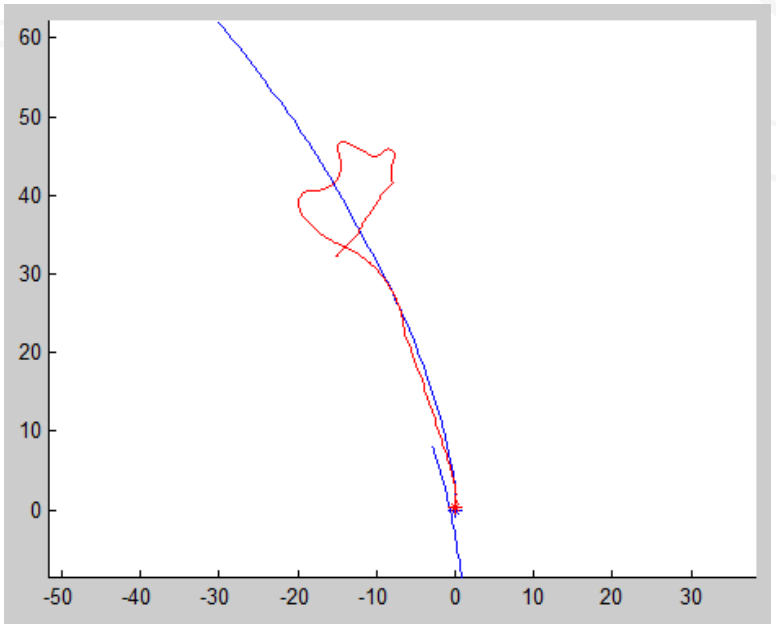
```
struct RAFeature
{
    int                                 numFailed;
    double                              age;
    DescriptorRAF                       descriptor;
    bool                                isActive;
    bool                                isUsed;
    std::vector<FeaturePosition>        vecPoint;
    bool                                isNew;
    bool                                has3Dinfo;
    int                                 method;
```

```
struct FeaturePosition
{
    unsigned long int   numFrame;
    PT2lf               posPoint;
    int                 matchCost;
    cv::Point3d         Pt3d;
```

- RAF : Robust Aged Feature
  - ➢ 지속적으로 특징 검출 및 추적에 성공하며(특징성이 좋음) 이상치를 제거하는 필터들을 통과한 점들(적합한 3차원 좌표 정보를 가짐)을 이용해 모션을 추정함
  - ➢ ver 1.0
    a. 검출 성공 시 모두 RAFSet에 기록되며 초기 age(특징성을 판단하는 지표)가 설정됨
    b. 특징성이 좋은 경우 age 값이 증가(aging) : 검출, 추적, 중복 성공
    c. 특징성이 나쁜 경우 age 값이 감소(de-aging) : 검출, 추적 실패
  - ➢ age 기준 값보다 높은 age를 가지는 RAF만을 이용하여 모션을 추정하여 정확도를 높임

RAFSet Motion Estimation

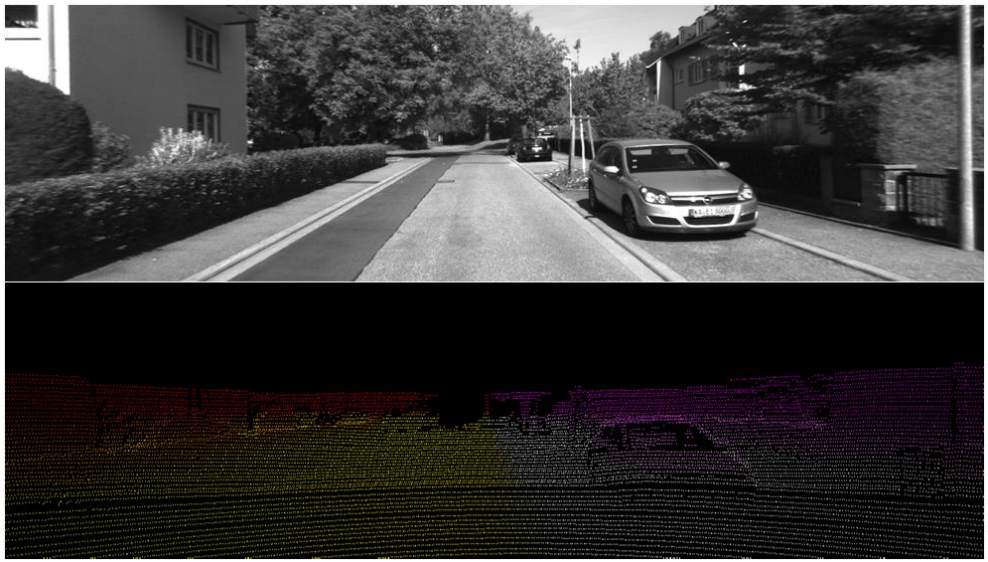


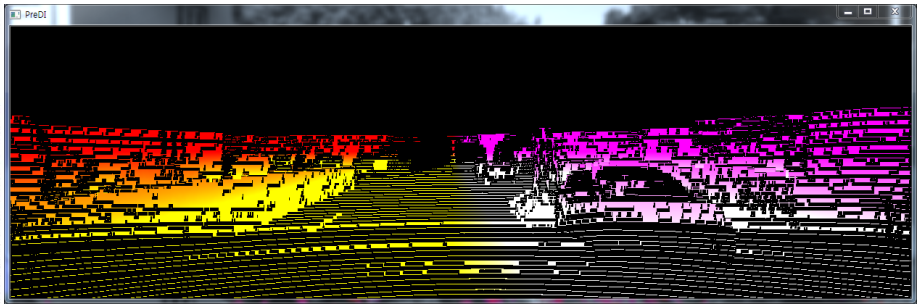


- 구조적 문제 수정 (ver 1.1)
  - 스케일이 점점 작아지는 문제
  - 구조적 문제로 인하여 스케일이 작아지는 것을 확인 -> 구조 수정
  - 글로벌 좌표계 기준 모션 추정에서 로컬 좌표계 기준 모션 추정으로 수정

- 수정사항 : 추적과 검출 및 3차원 정보 획득 실패 시 -> 이전 프레임 정보 사용

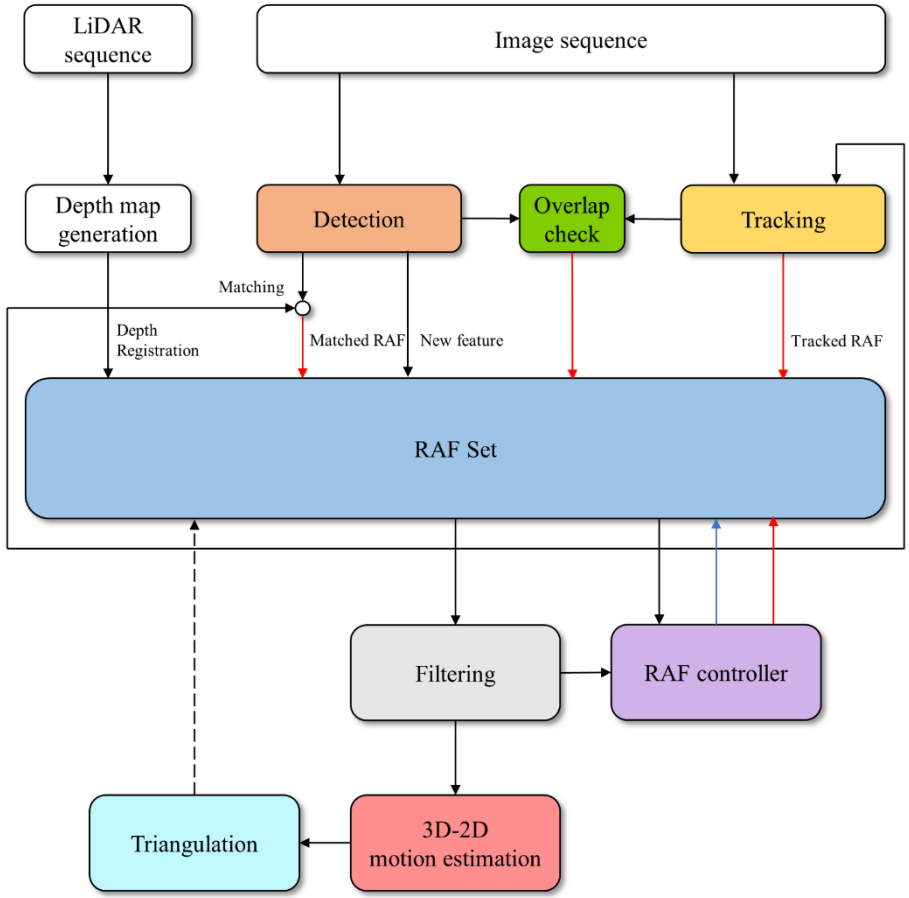
RAFSet Motion Estimation





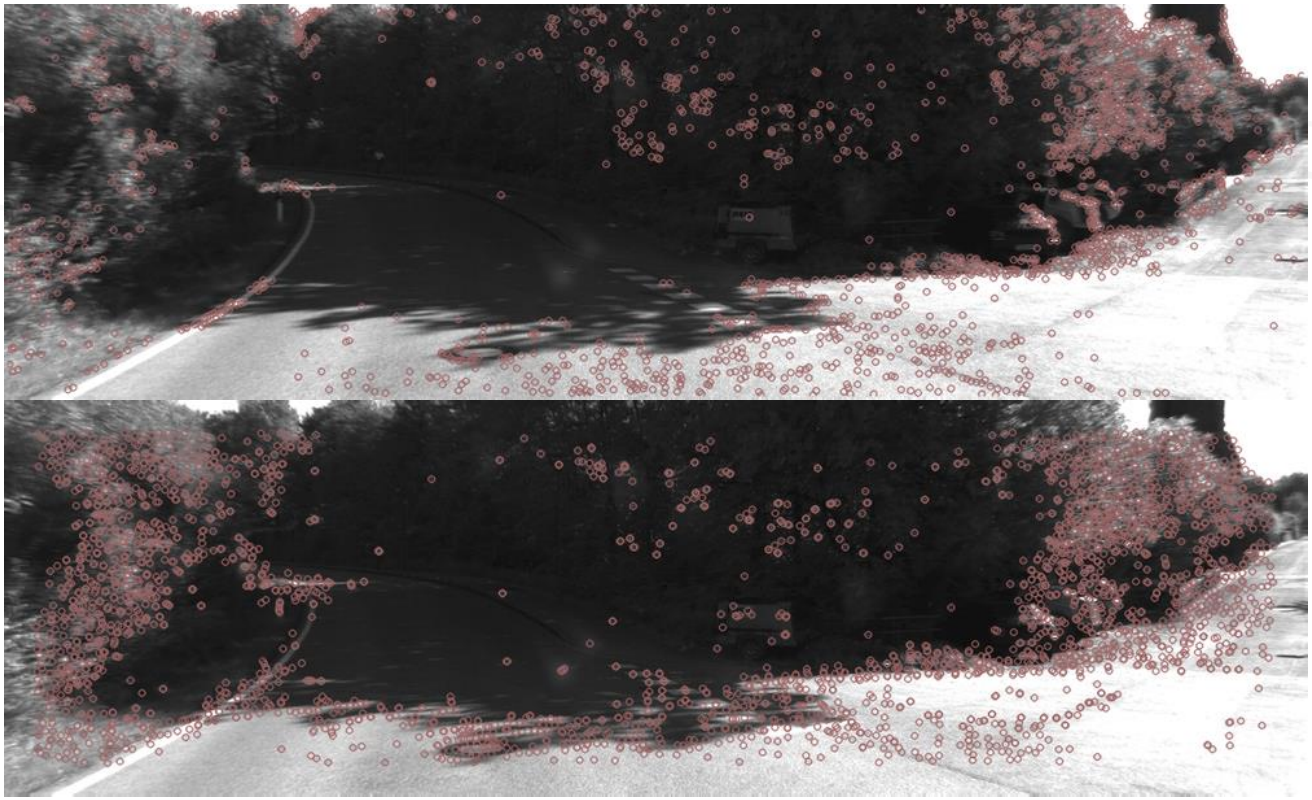
- LiDAR interpolation (ver 2.x)
  - ver 1.1의 결과로 RAFSet 내에 사용 가능한 3차원 좌표의 수가 적어짐
  - 3D-2D 매칭 쌍이 적어 모션 추정이 실패하는 문제 발생
  - 매칭 쌍을 더 얻기 위해 LiDAR interpolation 적용 (구현 시기는 다름)
  - 팽창 ver 2.0, **선택적 쌍선형 ver 2.1**, 평면 ver 2.2

RAFSet Motion Estimation



- 모션 추정 과정 및 de-aging 추가 (ver 2.1.1)
  - 기존 de-aging의 경우는 검출 및 추적 실패의 경우 뿐
  - 에피폴라 기하 이용 : 에피폴라 라인기준 5픽셀 이상 거리 차이 있을 경우 이상치로 판별

RAFSet Motion Estimation

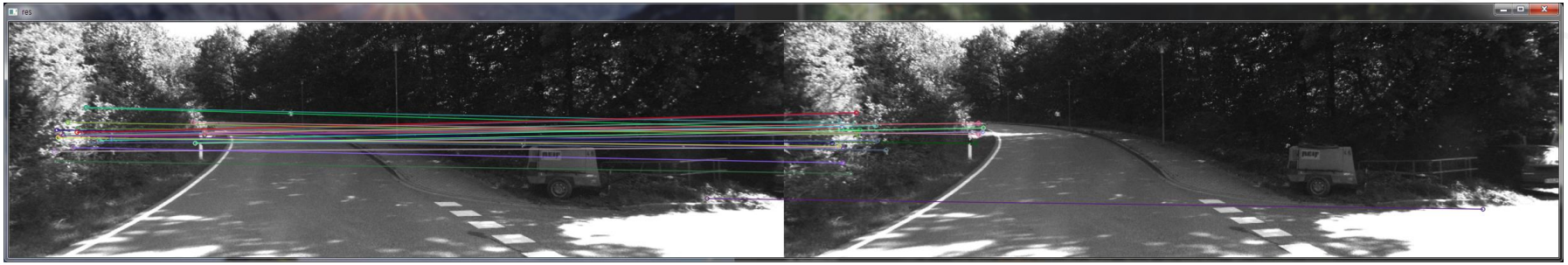


FAST (Gradient)

AKAZE (Gray)

- 특징 검출 방법 수정 (ver 2.1.2)
  - 기존 특징 추출은 Gradient 이미지에서 수행 -> Gray 이미지에서 추출하도록 수정 (특징 수 증가)
  - FAST는 빠른 처리 속도를 가지는 장점이 있으나 코너 특징에 취약하다는 단점이 있음
  - FAST 기반 코너 특징을 Harris로 고려하는 ORB를 적용 -> 상위 호환 AKAZE로 수정
  - 현재 AKAZE와 FAST를 바꿔가며 사용

RAFSet Motion Estimation



- 매칭 수정 (ver 2.1.3)
  - 기존 매칭 방법 : HOG을 기술자로 하여 매칭
  - AKAZE 기술자를 이용한 매칭을 테스트함
  - 연산량 문제로 인하여 HOG로 재수정
  - 매칭 부정확함을 반영하기 위해 Aging 수치 감소
- 에러를 줄이기 위한 구조 수정 (ver 2.1.4)
  - Triangulation을 통해 얻은 3차원 정보는 현재 모션 추정에 사용하지 않음
  - 생성은 하며 에러 확인에 사용

RAFSet Motion Estimation

- 이동 객체 검출 (ver 2.1.5)
  - ➢ 이동 객체는 모션 추정에 있어 다수의 이상치를 발생 시킴
  - ➢ 이를 해결하기 위해 알고리즘을 추가하였으나 성능이 좋지 않아 제거됨

1) OF벡터를 K-nearest neighbor 방법으로 분류, OF 크기 값 히스토그램 생성

2) 분류된 군의 영상내 좌표가 유사(분산이 작음) && 최빈 OF 크기 값보다 a배 이상 큰 경우
   모션 추정 적합성 플래그 OFF – 글로벌 모션과 불일치하는 모션을 가지는 객체
   (Age를 수정하지 않는 이유 : 특징성과는 상관이 없음)

3) 영상내 좌표가 유사하며 OF의 크기가 기준치보다 작은 경우
   모션 추정 적합성 플래그 OFF – 글로벌 모션과 동일한 모션을 가지는 객체
   ex) 같은 방향을 달리는 도로의 차량들

  - ➢ Deep learning 기반으로 객체를 찾아 내는 방법을 적용 고려

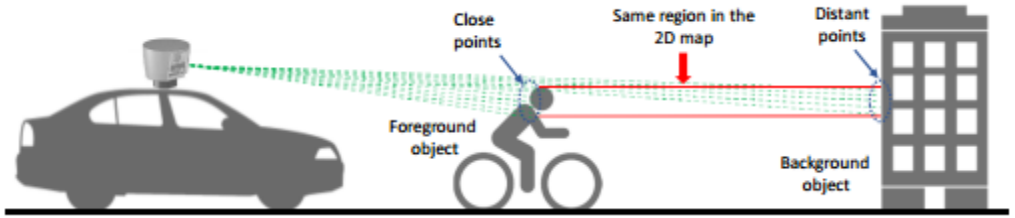
RAFSet Motion Estimation

```
struct RAFeature
{
    unsigned long int               id;
    unsigned long int               first;
    int                             numFailed;
//  PT2lf                           initialPoint;
//  PT2lf                           prevPoint;
//  PT2lf                           curPoint;
    double                          age;
//  MatcherName::Enum               Matcher;
    DescriptorRAF                   descriptor;
    bool                            isActive;
    bool                            isUsed;
    std::vector<FeaturePosition>    vecPoint;
    bool                            isNew;
    //UINT                          state;
    bool                            has3Dinfo;
    int                             method;
    bool                            flag_pre;
    bool                            flag_cur;
```

```
struct FeaturePosition
{
    unsigned long int   numFrame;
    PT2lf               posPoint;
    int                 matchCost;
    cv::Point3d         Pt3d;
    int                 method;
    double              age;
```

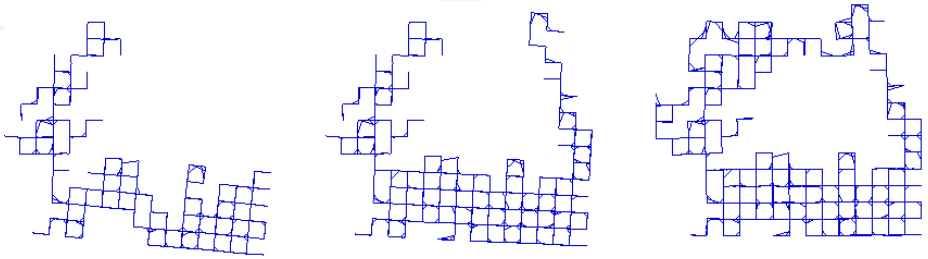
- 성능 향상을 위한 구조 수정 (ver 2.1.6)
  - 매 프레임에 대한 RAF의 age와 3차원 정보 획득 방법
  - 검출 및 추적 여부는 획득 프레임의 벡터를 확인하는 방식으로 처리, 방법의 중요성은 없음
  - 이를 통해 단순히 age값으로 걸러지지 못한 남은 이상치들을 제거하여 위치 추정이 가능해짐
  - 약간의 성능 향상 효과를 얻음

RAFSet Motion Estimation

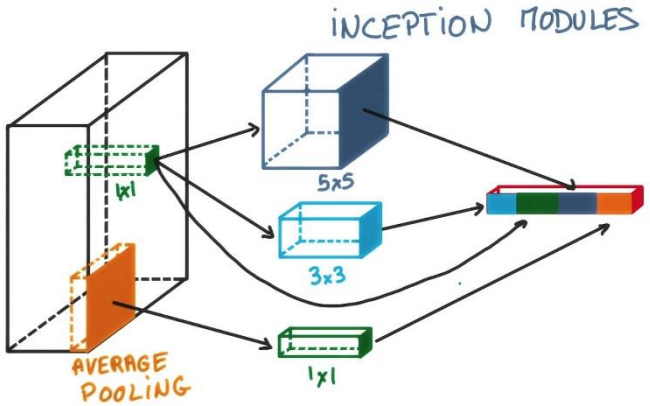


- LiDAR 데이터에 존재하는 문제점을 해결하기 위한 알고리즘 (ver 2.1.7)
  - 카메라와 LiDAR 사이의 위치 차이로 인한 투영 과정의 문제점을 발견
  - 해결하기 위해 평균과 분산을 고려하는 형태의 알고리즘을 적용
  - 현재는 사용하지 않음

RAFSet Motion Estimation



iSAM : incremental smoothing and mapping



GoogLeNet : Inception module

- 최적화 적용 (ver 2.1.7)
  - isam 라이브러리를 통한 최적화 결과를 사용
  - 성능 향상
- 가칭 iRAF (ver 3.0) 개념 도입
  - Solvepnp의 iteration 수, threshold 값, solve 방법 등 파라미터에 따른 성능 차가 심한 것을 확인
  - 여러 방법으로 solvepnp를 수행한 후 reprojection error를 가장 적게 하는 [R|t]를 선정
  - 성능 향상

RAFSet Motion Estimation

- 진행중 (ver 3.x)

1) 최적화

◆ 최적화 과정에서 Age를 고려하는 최적화 수행

◆ 일반적인 경우 신뢰도를 따로 신경 쓰지 않거나(동일하게), reprojection에러를 반영하여 설정

◆ 고려중인 방법 -> reprojection error와 age를 함께 고려하는 방식 ex. 비례 반비례 상수

2) 이동 객체 검출

◆ 딥러닝을 이용한 이동 객체 검출 (Tensorflow, Inception v2)

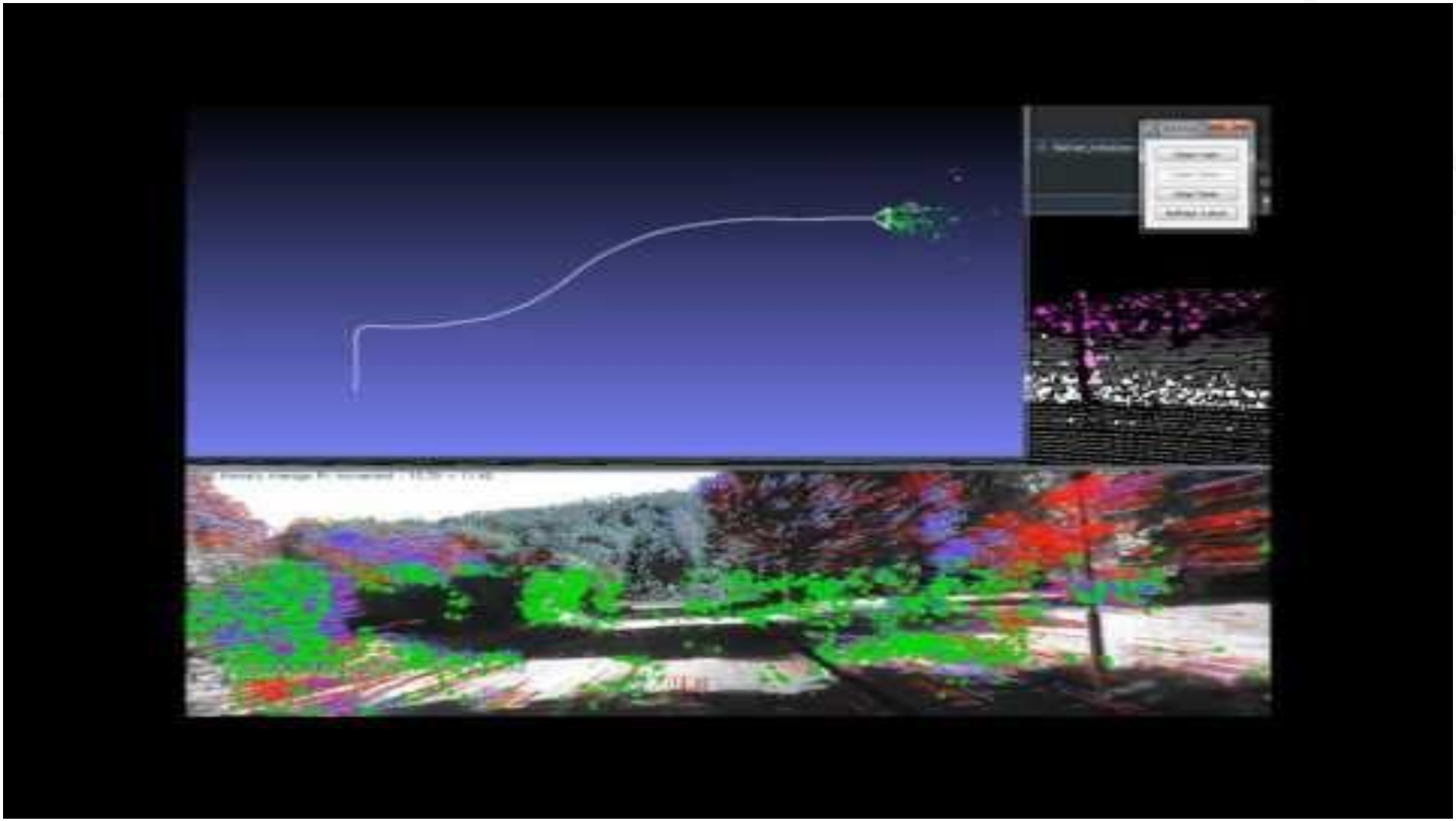
3) 모션 추정 정확도 향상 아이디어

◆ 딥러닝을 통한 모션 추정 정확도 상승 방법

◆ 수많은 데이터들(프레임별)과 참값이 있으므로 이를 학습하여 최적화 혹은 solvepnp의 초기값으로 설정하여 성능 향상

◆ 입력 값 : 단순 영상 or OF and [R|t]을 통해 학습(?)

Result



Q&A