PickNPlaceHLib

[]

목차

[개요 4](#_Toc72159275)

[이슈 트래킹(TODO?) 5](#_Toc72159276)

[PicknPlaceHLIB ReleaseNote 8](#_Toc72159277)

[Create Surface Model (모델 생성) 13](#_Toc72159278)

[1. 3D 모델 생성 방법 13](#_Toc72159279)

[2. Simple Model 생성 파라메터 일람 14](#_Toc72159280)

[1) Box 14](#_Toc72159281)

[2) Sphere 14](#_Toc72159282)

[3) Cylinder 14](#_Toc72159283)

[3. ImageModel 생성 파라메터 일람 14](#_Toc72159284)

[1) BackgoundFeature 14](#_Toc72159285)

[2) 자동 지면 삭제 14](#_Toc72159286)

[3) 지면(Z축)에서 일정 z축 범위내의 피사체의 PointCloud를 획득: 0 14](#_Toc72159287)

[4. Simple, Image Model 공통 전처리 Params 일람 15](#_Toc72159288)

[1) Sampling 3D Object Model Params 15](#_Toc72159289)

[2) ModelForm Params 16](#_Toc72159290)

[3) Triangle Model Params 16](#_Toc72159291)

[4) CreateSurfaceModel Params 17](#_Toc72159292)

[5. SurfaceModel Picking 위치 생성 19](#_Toc72159293)

[1) 피킹 위치 pose값 의미 19](#_Toc72159294)

[2) 피킹 위치 pose값 중 회전 속성의 의미 19](#_Toc72159295)

[3) pose 회전에 관해 20](#_Toc72159296)

[6. SurfaceModel, ObjectModel3D 파일 생성 20](#_Toc72159297)

[Surface based Matching (모델-씬 매칭) 21](#_Toc72159298)

[1. Scene 전처리 21](#_Toc72159299)

[1) Sampling 3D Object Model Params 21](#_Toc72159300)

[2) RoiForm Params 22](#_Toc72159301)

[2. SurfaceMatching 방법 25](#_Toc72159302)

[1) SurfaceMatching Parameters 25](#_Toc72159303)

[3. Surface based Matching 단계 26](#_Toc72159304)

[1) Approximate matching 대략적인 매칭 26](#_Toc72159305)

[2) Sparse pose refinement 희소 포즈 조정 26](#_Toc72159306)

[3) Dense pose refinement 밀집 포즈 조정 26](#_Toc72159307)

[4. Surface based Match 구동 방법 27](#_Toc72159308)

[5. 일부 매칭 사례 27](#_Toc72159309)

[6. 좌표 정렬 28](#_Toc72159310)

[1) 피사체 중심 원점을 기준 Tz’ 90, 180’ 위치할 경우 28](#_Toc72159311)

[2) Pointcloud Bounding box의 중심 Trans X, Y, Z활용 28](#_Toc72159312)

[UML Diagram 32](#_Toc72159313)

[AG\_Vision Protocol 35](#_Toc72159314)

[Test Result Summary 38](#_Toc72159315)

[부 록 44](#_Toc72159316)

[1. Halcon회전영향이 있는(pose\_compose)에 관하여 44](#_Toc72159317)

[2. pose\_compose 내부 변환 방식에 관하여 46](#_Toc72159318)

# 개요

PickNPlaceHLib는 Halcon 3D모델(3D Obj) 생성을 하고, Scene을 찍어 이 모델을 씬에서 찾아 6자유도 축의 좌표를 획득하는 기능을 한다.

# 이슈 트래킹(TODO?)

\*\*notice : 추가해야할 것 \*\*more : 테스트미비, 구현미비 등 채워야 할 부분 \*\*bug : 기능 수행에 버그 있음

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **발생일** | **Todo** | **ISSUE** | **Desc** | **Note** | **반영 예정일 / 반영일** |
| 2021.04.28 | \*\*bug | 배경 분리에 관계적 알고리즘 사용 | 지면 PointCloud와 피사체 PointCloud가 붙어 있을 경우 Halcon Lib로 분리가 되지 않는다  원인: triangulate이후 PointCloud에 Mesh가 그려지고 Connection\_object\_model3d를 이용해 mesh를 구분 짓는데, 지면과 피사체의 PointCloud는 음영, 빛 반사등으로 인해 얼마든지 달라 붙을 수있고 이걸 Mesh, Distance3D로 구분 짓기엔 붙어있는 것이 지면과 피사체라고 알고리즘 적으로 나눌 수 있는 부분이 아닌 것으로 생각된다. | 차라리 AutoROI Code 활용해 z깊이 단위로 pointcloud를 잘라 point개수가 적어지는 지점(지면이 있는 깊이 단위에는 x,y의 point가 넓게 펼쳐져 있으니 당연히 point 개수가 많다.) 다음부터 z깊이 최상단 까지 자르는 것이 낫지 않을까?  AutoROI 방식을 이용할 경우 Connection\_object\_model3d 관련 파라메터 설정하는 부분 제거 필요. | 2021.05 / |
| 2021.04.28 | \*\*bug | Moving ROI 동작 | 원인: ROI, ModelForm, ROIBOX, ObjectArrow 로직 추가 후 정상동작 불가  *bug 외, 특정 영역을 이동하며 매칭하는 부분이 시간이 걸림.*  *Static, Auto ROI가 있고, 미리 각각 잘라놓은 ROI Image를 모아서 각 각 매칭을 하려고 해도 결국은 시간은 걸린다.*  *Status: 본 기능의 효율성에 의문.* | 수치에 의한 것이 아닌 X,Y에 대해 마우스로 영역을 지정하여 매칭 하는 기능으로 변경하는 것이 어떨까?... | 2021.05 / |
| 2021.04.28 | \*\*notice | 변수명이 원래 속성값 의미X | find\_sfm\_FindMethod->‘find\_sfm\_scene\_normal\_computation’ | 0.1.5 신규 버전에서 수정 예정 | 2021.05 / |
| 2021.04.28 | \*\*notice : | Triangle Model Params | Auto와 Default로 두는 것이 PointCloud에서 Mesh로 변환이 더 간단함… 아래 파라메터들을 통해 Auto보다 더 잘 만드는 것이 되려 어렵다  생성, 매칭 단계에서train view based 역할 조사 및 구현추가 필요 |  | 2021.05 / |
| 2021.04.28 | \*\*more: | CreateSurfaceModel Params | 1. symmertric을 ObjectModel3D단계에서 할 경우 유저나 개발자가 x,y,z에 대해 대칭을 기입해야하는데 이는 피킹위치 지정에서 또 달라짐… ObjectModel3D 전처리 과정을 다 거치고 피킹 위치 지정 후 train\_self\_similar\_pose로 Halcon이 자동으로 셋팅 가능하니 검토  2. train\_Edge 모델 생성시 ObjectModel3D의 전처리는 피사체의 평면 상태일 때 정상적으로 생성되는데 왜 그런지 확인  3. train\_view\_based를 이용해 view\_base score를 활용 가능하며 기존 모델에 이 속성을 활용시 매칭 결과의 점수는 100~80%에 달한다.  옵션 역할조사 및 구현해도 괜찮다면 옵션 추가. | 1. train\_self\_similar\_pose: 하기 항목에서 테스트 참조  *4. Simple, Image Model 공통 전처리 Params 일람*  *4) CreateSurfaceModel Params*  *c) train\_self\_similar\_pose:*  2. 최초 생성한 Pointcloud에 Triangulate\_Obj3d를 통해 Mesh를 생성해야 EdgeModel이 생성된다. 문제는 이전에 SmoothObj3d를 통해 Mesh를 단순화 하는 부분을 추가하였는데, 이 SmoothObj3D는 PointCloud를 단순화 하여 재배치 하는데, 여기서 Mesh가 왜곡 되어 Model이 단순화 되고 이로 인해 정상적인 EdgeModel이 생성되지 못한 Case를 경험하여 \*\*more에 추가  3. Train\_view\_based: 하기 항목에서 테스트 참조  *4. Simple, Image Model 공통 전처리 Params 일람*  *4) CreateSurfaceModel Params*  *d) create\_sfm\_train\_view\_based* | 2021.05 / |
| 2021.04.28 | \*\*more | num\_scene\_point  점수 정의 | SurfaceModel이 피사체의 360’ 모두를 표현하는 PointCloud를 가지고 있고, Scene에서의 타겟은 일부만 보이기 때문에 model\_point\_fraction 디폴트 점수 비율이 20~50% 수준. 이 대안으로 num\_scene\_point를 사용하였으나, 점수가 거리가중방식이라 500% 1000%에 달한다. 점수를 어떻게 정의 할 것인지 결정 필요. ex)train\_view\_based 점수를 기존 모델에서 테스트할 경우 90%~100%의 점수가 나오는데 어떻게 나오는 것인지 확인 필요 | 3. Train\_view\_based: 하기 항목에서 테스트 참조  *4. Simple, Image Model 공통 전처리 Params 일람*  *4) CreateSurfaceModel Params*  *d) create\_sfm\_train\_view\_based* | 2021.05 / |
| 2021.04.28 | \*\*more | find\_sfm\_pose\_ref\_use\_scene\_normals\_value | Scene에서 피사체의 Normal은 카메라가 바라보는 Z축을 따라 형성되기 때문에 생성한 SurfaceModel과는 방향 유형, 보이는 부분이 분명 다르다 정확도에 영향을 주는지 확인 필요 | 포즈미세조정의 정확도를 높이는 기능이라고 되어있다.  Sample Scene 테스트로는 점수는 동일. 테스트 씬에 노이즈가 거의 없어서 인걸로 추측. | 2021.05 / |
| 2021.04.28 | \*\*more | Rp+T gba 회전 순서 | Green? 축 rotation? 축과의 각도? 정확한 회전의 의미, 효과. 저 수치의 발생 요인… Z 축을 보면 X축을 중심으로 rotation한 것 같은데… 등등 고찰과 결과 추가 필요. | 회전 순서  5. SurfaceModel Picking 위치 생성 정리 및 부록 추가 | 2021.05 / |
| 2021.05.17 | \*\*more | 매칭 후 회전 좌표 정렬  (= Symmetric 옵션) | 피킹위치 지정 사진하고 rX, rY 90’ 무작위회전하는 사진, 결과 예시 사진 필요 |  |  |
| 2021.05.17 | \*\*more: | Center, Horizon을 지정하는 범위 값을 30mm | 화살표가 대각선일 경우 up과 down의 중간을 의미하는, 30mm (= 화살표의 얇은 지름 Pointcloud영역)이 모호한 부분이 있음. 다른 접근 방법 필요.. |  | 2021.06 |
| 2021.05.17 | \*\*notice | 직육면체, 정육면체, 구, 실린더 등 형태에 따른 회전 옵션 추가 검토 | 90, 180’등 어떻게 정렬할 것인가에 대한 옵션 필요 Pickit3D how to pick - youtube영상 추후 참조 하여 기능 추가 |  |  |

# PicknPlaceHLIB ReleaseNote

|  |  |  |  |
| --- | --- | --- | --- |
| 버전 | 변경사항 | | 비고 |
| C# | HALCON |
| 0.0.1 | 1. 모델 생성 프로시저 연결  2. 모델 합성 프로시저 연결  3. 모델 찾기 프로시저 연결 | 1. 모델 생성 프로시저  2. 모델 합성 프로시저  3. 모델 찾기 프로시저 | 최초 릴리즈 |
| 0.0.2 | 1. 모델 생성 프로시저 연결부  - 프로시저 삭제 기능 추가  - om3, sfm3 생성기능 추가  - 엣지기능 파라메터 추가  - 스무싱 파라메터 추가  - stl파일 인입기능 추가  2. 모델 합성 프로시저 연결부  - 프로시저 삭제기능 추가  - om3, sfm 생성기능 추가  3. 모델 찾기 프로시저 연결부  - 프로시저 삭제기능 추가  - surfacematching with edge 파라메터 추가  - moving roi 파라메터 추가  - 스무싱 파라메터 추가  - 모델 찾기 스코어 유형 변경으로 인한 파라메터 추가 | 1. 모델 생성 프로시저  - om3, sfm 파일 생성 기능 삭제  - 엣지기능 추가  - [] 값으로 sample, triangulate 동작기능 추가  - triangulate 파라메터 확장  - 싱글 모델로 매칭을 위한 방향성주입 기능 추가  - stl파일로 모델 생성기능 추가  - 폴리곤, 포인트가 없을 경우 리턴 예외처리 추가  2. 모델 합성 프로시저  - om3, sfm생성기능 삭제  - [] 값으로 sample triangulate 동작 기능 추가  3. 모델 찾기 프로시저  - surfacematching with edge기능 추가  - moving roi 기능 추가  - find\_surface\_model 스코어 유형 변경 |  |
| 0.0.3 | 1. 전체 lib에 로그기능 저장 기능 추가  - Halcon 예외 C#으로 받아 기록  - 로그저장위치는 3Dvision 파라미터 정리 문서 참조  2. Assembly Version 이력 추가.  3. 프로시저, 3D Model 준비, 완료, 에러 등의 이벤트 콜백 및 확인가능한 Property 변수 추가  4. 모델생성, 매칭, 3DVis 프로시저 업데이트에 따른 파라메터 증가  5. libSurfaceMatching.cs  - saveSurfmatchParam() 인입 파라메터 분리  saveSceneParam() 메서드로 씬에 관한 인입 파라메터 이양  - findSurfaceMatching() 메서드 오버로딩 추가  SimpleObj생성 기능 추가 | 1. CreateModelSurfModel0.0.3 업데이트  - 지면 기준 배경 분리 기능 변경  (기존) 카메라기준 DEPTH  (변경) 지면을 0으로 기준  2. Halcon3DVis 0.0.1 업데이트  - 동일 모델 다량 매칭 시 색 표현 수정  3. SurfMatch\_FileHandle 0.0.4 업데이트  - 심플모델 생성 기능 추가(박스, 구체, 원통)  - ROI\_USE 튜플명 변경  - MatchForm 튜플 추가  Point매칭, 엣지매칭, Triangle(폴리곤)매칭  - 점수 반환 20.3.26 PM6:10 메일의 형식으로 변경 | 프로시져완료시  PROCEDUREDONE  콜백 발생  모델생성가능시  MODELREADY  콜백 발생  모델파일로 생성후  MODELCREATE  콜백 발생 세부 내용은  PicknPlaceHLib\_0.0.3.chm 도움말 문서 참조 |
| 0.0.4 | libModelCreate    libModelUnion  1. 파일명 리스트 1개만 로깅 되는 버그 수정  libSurfaceMatching  1. saveSurfmatchParam() 피사체 매칭 좌표 정렬 on, off 파라메터 추가  2. getResult3DDisp() 화살표 Obj3D 변수 및 파라메터 추가 | createSurfModel\_0\_0\_3  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_5  1. 피사체 매칭 좌표 정렬 기능 On추가  2. PointCloud View 피사체 좌표 표현 3DObj 추가  Halcon3DVis\_0\_0\_3  1. PointCloud View 피사체 좌표 표현 3DObj UI구현  2. 3D View Static View 구현 및 Movable View와 구분 | libSurfaceMatching  - saveSurfmatch()  인입 변수 1개 추가  - getResult3DDisp()  인입 변수 1개 추가  Halcon Procedure 디버깅 및 분석용 Halcon 프로젝트 툴 릴리즈  Procedure와 동일 기능 수행 |
| 0.0.5 | libModelCreate  1. 사용자 의도에 따른 모델 생성 종료 시 Procedure에러 이벤트 추가  libModelUnion  1. 파일명 리스트 1개만 로깅 되는 버그 수정  libSurfaceMatching  1. 메모리 누수 버그 수정  - Halcon 엔진, 프로시저 객체 생성, 해제를 Halcon procedure 호출하는 findSurfaceMatching(), Halcon3DVis() 메서드 내에서 1회성으로 할 수 있도록 변경  2. SurfMatch\_FileHandle\_0\_0\_6 점수 산정 방식 파라메터 추가 | createSurfModel\_0\_0\_4  1. 배경 분리과정, Surface Model 생성 전 사용자의 의도에 따라 종료후 에러처리용 OUT Param추가  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_6  1. SurfaceMatching시 점수 산정 방식 파라메터 추가  Halcon3DVis\_0\_0\_3 | libSurfaceMatching  - saveSurfmatchParam()  인입 변수 1개 추가 |
| 0.0.6 | libModelCreate  1. saveSegmentParam Summary의 디폴트 값 내용 변경  libModelUnion  libSurfaceMatching  libDisp (신규 클래스)  1. 생성자를 이용하여 객체 생성  2. getDisp(string, int, 2dhwin, 3dhwin) 호출  1) ply file 인입: 2d, 3d 디스플레이, hwin handle 2개 필요)  2) om3 file 인입: 3d만 디스플레이, hwin handle 1개 필요)  3) om3 file 인입: 3d만 디스플레이, hwin handle 1개 필요)  3. Dispose() 호출하여 객체 해제  1) getDisp() 호출시 int값이 1일 경우:  상호작용이 가능한 halcon 3d disp를 사용 (hwindow의 continue버튼 클릭 후) Dispose()를 해야함 | createSurfModel\_0\_0\_4  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_6  Halcon3DVis\_0\_0\_3  HalconVis\_0\_0\_1 (신규 프로시저)  - ply, om3, sfm의 경로를 받아 halconwindow의 용도(2d, 3d)에 따라 window에 이미지 데이터를 인입시키는 프로시저  *본 프로시저와 달리 Halcon3DVis는 SurfaceMatching후 매칭된 피사체의 표기, ROI영역 표기 등의 내용이 들어있음* | libDisp class 추가  - form에서 사용할 수 있는 2D, 3D Display Class  - API Reference  Enumerations에 Summary 추가 |
| 0.0.7 | libModelCreate  1. Summary Parameter 내용 추가  2. libModelCreate() 인자값 ply 파일 입력 삭제  3. saveImageCreateModelParam() 메서드 추가  4. saveSimpleCreateModelParam() 메서드 추가  5. 사용하지 않는 메서드, 변수 정리  6. 메서드 입력변수 명칭 정리  libModelUnion  libSurfaceMatching  1. get3DDisp() - 버튼 있는 3D View 사용시 Halcon의 Continue버튼으로 Procedure 종료 처리하지 않아도 Dispose 호출 가능  2. Summary Parameter 내용 추가  3. findSurfacematch(SimpleObj) 메서드 삭제  libDisp  1. getDisp() - 버튼 있는 3D View 사용시 Halcon의 Continue버튼으로 Procedure 종료 처리하지 않아도 Dispose 호출 가능 | createSurfModel\_0\_0\_5  1. SimpleObjModel 생성 기능 추가  2. 거리 알고리즘을 이용한 배경분리 기능 삭제  3. 스무싱 -> ModelForm변경 및 모델 종류 정리  (ModelForm(생성) - MatchForm(매칭) 명칭 연동  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_7  2. SimpleObjModel 생성 기능 삭제  Halcon3DVis\_0\_0\_3  1. visualize\_object\_model\_3d 쓰레드 적용(C#단 Procedure 강제 종료 용도)  HalconVis\_0\_0\_1  1. visualize\_object\_model\_3d 쓰레드 적용(C#단 Procedure 강제 종료 용도) | - API 사용법 확인용  *PicknPlaceHLib\_DLL\_Test\_Proj\_0.0.7.zip 배포* |
| 0.0.7a | libModelCreate  - saveImageCreateModelParam()  거리 알고리즘을 이용한 배경분리 기능 복원 | createSurfModel\_0\_0\_5  거리 알고리즘을 이용한 배경분리 기능 복원  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_7  Halcon3DVis\_0\_0\_3  HalconVis\_0\_0\_1 | saveImageCreateModelParam()  거리 알고리즘 관련 파라메터 복원 |
| 0.0.8 | libModelCreate  1. Keypoint 반전 기능 파라메터 추가  saveImageCreateModelParam()  saveSimpleCreateModelParam() | createSurfModel\_0\_0\_6  1. 버튼 뷰어 C#단 강제 종료를 위한 쓰레드 제거  2. SurfaceModel생성 API에서 Surface Model의 Keypoint 반전 기능 유무 추가  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_7  1. ROI 박스 생성 버그수정  지면 씬의 Z중심값 -> 지면 우선속성의 z축 좌표값  Halcon3DVis\_0\_0\_3  HalconVis\_0\_0\_1 | saveImageCreateModelParam()  saveSimpleCreateModelParam()  상기 두 메서드 입력 파라메터 1개 추가 |
| 0.0.8a | libModelCreate  libModelUnion  libSurfaceMatching  3개 클래스 내 halcon in 파라메터 명칭 변경  우측 비고 참조 | createSurfModel\_0\_0\_6  ModelUnion\_0\_0\_2  SurfMatch\_FileHandle\_0\_0\_7  3개 프로시저 입력 파라메터 명칭 변경  우측 비고 참조 | 파라메터 명칭 규칙  halcon operator 명칭 유형  - *blah\_*blah\_blah(*param*, *paramval*)  c#, halcon 파라메터 명칭 유형  *1. blah*  *2. blah\_param*  *3. blah\_paramval* |
| 0.0.9 | libSurfaceMatching.saveSurfmatchParam()  입력 파라메터 4종 추가  string *find\_sfm\_max\_overlap\_dist\_type*,  int *find\_sfm\_pose\_ref\_use\_scene\_normals\_value*,  int *find\_sfm\_pose\_ref\_num\_steps\_value*,  int *find\_sfm\_pose\_ref\_sub\_sampling\_value* | SurfMatch\_FileHandle\_0\_0\_8  find\_surface\_model()입력 파라메터 추가  *max\_overlap\_dist\_type(rel, abs)*  *pose\_ref\_use\_scene\_normals*,  *pose\_ref\_num\_steps*,  *pose\_ref\_sub\_sampling* | libSurfaceMatching.saveSurfmatchParam()  입력 파라메터 4종 추가 |
| 0.1.0 | libModelCreate()  입력 파라메터 1종 추가  int Smooth\_Feature | CreateSurfModel\_0\_0\_7  SmoothFeatrue 입력파라메터 추가 | SurfaceModel 생성전 모델 스무싱  TestSummury6/25 모델의 내부, 외부면 매칭 참조 |
| 0.1.2 | libModelCreate()  입력 파라메터 1종 추가  int Simple\_HalfCut | CreateSurfModel\_0\_0\_8  Simple\_HalfCut 입력파라메터 추가 | OBJ3D -> Surface Model로 만들기 전 z축 탑뷰 중심에서 뒷면의 PointCloud삭제  사각형 모델 6면으로 씬의 일부 단면에 매칭하러 갈 경우 로테이션 좌표가 6면에 걸치는 문제 방지 |
| 0.1.3 |  | SurfMatch\_FileHandle\_0\_1\_0  - 회전좌표 정렬 기능 버그 수정  (피사체 좌표에서 RX or RY180’ 회전 정렬-> 피사체 중심 좌표 추출 RX or RY 90, 180’ 복합 회전 정렬)  Halcon3DVis\_0\_0\_3  HalconVis\_0\_0\_1  CreateSurfModel\_0\_0\_8  - (공통) Normals 시각화 기능 추가 |  |
| 0.1.4 | libSurfaceMatching()  입력 파라메터 3종 추가  int PickLimitDegree  int FindSurfModelTimeoutSec  int RZAlignOrder | SurfMatch\_FileHandle\_0\_1\_2  입력 파라메터 3종 추가  PickLimitDegree  FindSurfModelTimeoutSec  RZAlignOrder | PickLimitDegree 값 활용 방법  Rx, Ry를 회전 시킬 때 315 ~ 45 각도 값 안이면 3축이 정렬되었다고 판단함.  FindSurfModelTimeoutSec  무한매칭에 대한 타임아웃값  RZAlignOrder  RZ(뷰어상의 X축) 방향을 4구역(0~90, 90~180, 180~270, 270~360)으로 나누어 매칭된 피사체를 90’씩 돌려 Rz 정렬  현상적으로 정렬시킨 최종 Rz회전 값이 4구역 값 45, 135, 225, 315 값 이상일 경우 90’ 역회전하여 로봇의 필요이상의 회전 방지 |
| 0.1.5 | libSurfaceMatching()  saveSceneParam()  int RoiForm (3: Auto ROI) 추가 | SurfMatch\_FileHandle\_0\_1\_3  int RoiForm (3: Auto ROI) 추가 | Auto ROI  Scene PointCloud를 감싸는 Boundbox속성(point가 존재하는 위치 정보 min xyz, max x y, z)을 응용한 AutoRoi  위 속성값을 이용해 x,y, max z roi 범위를 설정하고 min z는 지면 0 ~ 7 범위 씬 포인트 개수를 1,2,3차 까지 반씩 잘라 비교하여 지면이 아니라고 판단한 값(지면이 있는 깊이 단위에는 x,y의 point가 넓게 펼쳐져 있으니 당연히 point 개수가 많으니 이 지점의 z깊이값 제외)을 min z roi값으로 자동 입력 |

# Create Surface Model (모델 생성)

## 1. 3D 모델 생성 방법

Table 1. Simple Model

|  |  |  |  |
| --- | --- | --- | --- |
| Simple Model | | | |
| 모델 유형 | Box | Sphere | Cylinder |
| 생성 방식 |  |  |  |
| 생성 방법 | 실제 사이즈 입력 X, Y, Z  mm단위 | Radius 실제 사이즈 입력  mm 단위 | Radius 실제 사이즈 입력  Z축 높이 중앙에서 상/하단 길이 mm입력 |

*SimpleModel은 생성 후 매칭 테스트 진행 시 실제 사이즈에서 +-1~2mm 차이나는 모델을 추가 테스트 하여 매칭 결과 점수가 더 높은 모델을 선택하는 방법이 추천된다. (피사체의 빛 반사, PointCloud의 노이즈 등으로 인해 꼭 실제 대상의 사이즈를 입력한 SurfaceModel이 더 우수하다고 할 수는 없다)*

Table 2. Image Model

|  |  |
| --- | --- |
| 모델 유형 | Image Model |
| 생성 방식 |  |
| 생성 방법 | 취득한 Scene 2D Image에서 사각형 영역을 그려 지면을 제거하고 피사체만을 ObjectModel3D화 |

\**ImageModel은 많은 PointCloud를 생성하기 위해 피사체의 많은 부분이 보이는 위치와 방향에서 놓고 Scene을 취득하는 것이 중요하다.*

*Box영역을 지정할 때, 피사체 + 주변의 지면이 포함되어야 지면을 인식해 자르는 기능이 정상 동작한다.*

*\*모델 생성시 지면을 제거하는 것은 지면으로 인한 오매칭 확률을 낮추고 연산 속도를 증대 시키기 위함.*

*\*비율은 피사체 주위의 지면을 소폭 추가하는 정도 8~9(피사체) : 1~2(지면)를 추천한다.*

\*\* more : 이 박스영역이 어느 정도의 피사체의 Point와 지면의 Point의 비율이 Halcon내에서 지면을 인식해 자르는 기능에 영향이 있다.

지면을 인식하기 위해선 어느 정도의 지면 PointCloud가 박스영역안에 들어가야 하는지?

**MHK: 수치로 나타내기 어렵고, 피사체의 복잡도, 원하는 정밀도 등에 따라서 케바케임. 지면을 포함해라 정도와 그 이유를 기술하는 것으로. 지면도, 피사체도 정보가 많으면 의도에 부합하는데 지면은 일정 기준 +/- 정도로 간주할 테니… 피사체 9, 지면 1 정도의 영역 지정…**

## 2. Simple Model 생성 파라메터 일람

### 1) Box

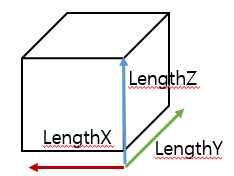
a) LengthX = X축 실제 길이(mm)

Figure 1. Mean of Box Length

b) LengthY = Y축 실제 길이(mm)

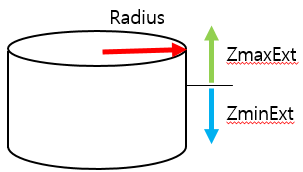
c) LengthZ = Z축 실제 길이(mm)

### 2) Sphere

a) Radius = 반지름 실제 길이(mm)

### 3) Cylinder

Figure 2. Mean of Sphere Radius

a) Radius = 기둥 상단부 원형의 반지름 실제 길이(mm)

b) ZminExt = 기둥 높이 중앙에서 하단까지의 길이(mm)

c) ZminExt = 기둥 높이 중앙에서 상단까지의 길이(mm)

Figure 3. Mean of Radius, ZminExt and ZmaxExt

## 3. ImageModel 생성 파라메터 일람

### 1) ObjectModel3DCrop

Image는 Scene에서 박스영역을 통해 PointCloud를 잘라냈을 때

다음과 같이 지면이 섞여있어 지면 PointCloud로 인해 매칭오류

NG 등의 악영향을 제거하기 위해 지면을 잘라내고 피사체만을

의미하는 PointCloud로 ObjectModel3D를 만들어야 한다

Figure 4. Image Model Pointcloud

#### a) 지면(Z축)에서 일정 z축 범위내의 피사체의 PointCloud를 획득: 0

지면 기준으로 피사체와 배경 분리 도구 연관 파라메터

minZDepth: 지면의깊이를 0으로 만든다. Ex)0~5 mm까지 지면 PointCloud 영역, 이후 피사체의 포인트 클라우드를 형성(default:0.0) -N ~ N (카메라 센서 위치)

maxZDepth: 피사체의 PCL이 형성될 높이이며, 0.0 ~ N(카메라 센서 위치)

#### b) 자동 지면 삭제

***자동으로 자를 지면의 margin값(parameter) == (1mm씩 자른 씬의 point 개수의 abs(현재-이전) 차 값)***

구동 방식 참조: [[@AutoROI](#_d)_AutoRoi)]

#### c) Image 미세 Crop

Image의 Max, Min X, Y, Z를 조절하여 ObjectModel3D를 미세조정

****

Figure 5. Specifying a target in a 2d image

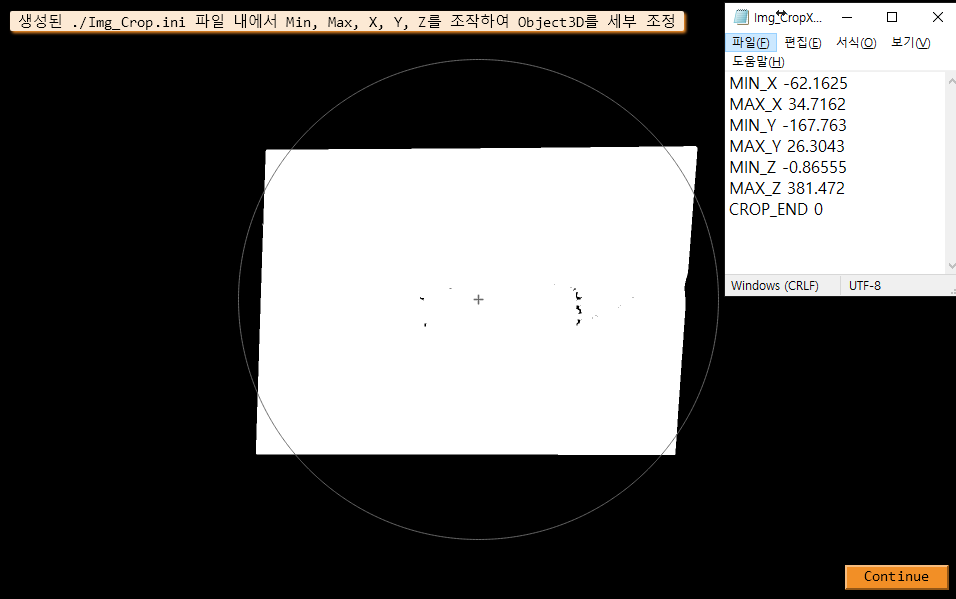


Figure 6. ObjectModel3D cropped from 2d image

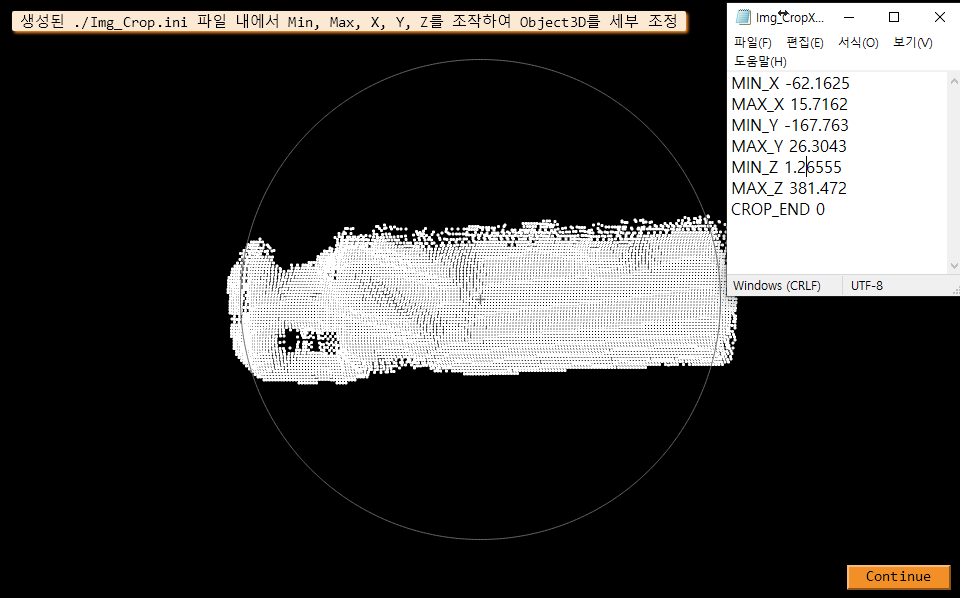


Figure 7. Fine-tune using Max, Min X,Y,Z in Ini file

## 4. Simple, Image Model 공통 전처리 Params 일람

### 1) Sampling 3D Object Model Params

SimpleModel은 최초 생성시 Mesh(Polygon)만으로 형성되어 매칭 단계에서 사용할 Point가 없어 Keypoint, Normal을 비교할 수 없기 때문에 Sampling을 거쳐 Mesh를 기반으로 Point, Keypoint를 생성해주어야한다

*매칭 단계에서 사용할 Model에는 normal이 필요하며, SimpleModel에서 fast, accurate는 Normal을 생성하지 않기 때문에 사용불가*

#### a) Sampling\_method

accurate: Model Point 간의 Samping 거리 간격 값 사이에 Point를 생성하지만, 그 사이 다른 Point들이 있다면 무게 중심을 계산하여 점을 생성한다. \*연산량이 많기에 Sampling에 시간이 필요

accurate\_use\_normals: ObjectModel3D에 Normal이 있어야하며 이 Normal이 유사한 Point만을 보간한다. \*선택할 수 있는 기능 중 가장 많은 연산량을 가짐

fast: Model의 Point간의 Samping 거리값에 따라 Point를 생성 \*가장 빠름

fast\_compute\_normals: Model을 fast로 Samping을 하고 Normal을 생성

#### b) Samping\_distance

- Point간의 샘플링 거리를 설정하며 0.0 ~N 사이의 값을 가질 수 있음

값이 0에 가까울수록 Model을 구성하는 Point의 개수가 증가하며 정교한 ObjectModel3D이 생성됨 \*연산량 증가

값이 클수록 모델을 구성하는 Point의 개수가 적어지며, 단순한 PointCloud가 형성된 모델이 생성됨 \*연산량 감소

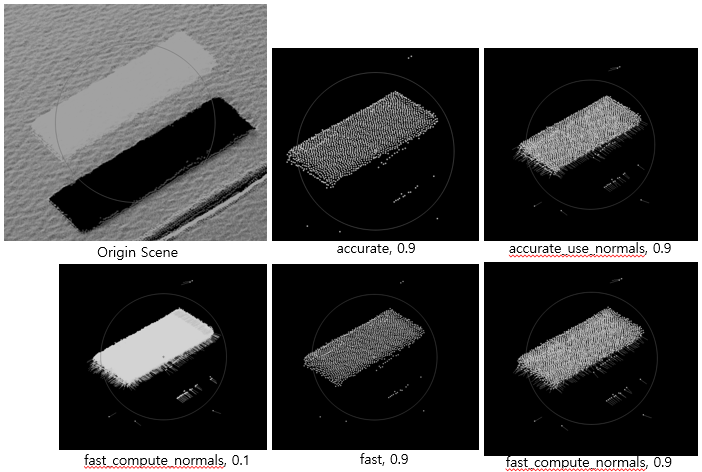


Figure 8. Pointcloud comparison of each sampling parameter

### 2) ModelForm Params

Model의 형상은 매칭 시 특정 속성을 이용하여 매칭 결과 Pose의 정확도를 높혀주는 역할을 한다.

반대 급부로 피사체의 복잡도를 고려하지 않는 ObjectModel3D 형상의 선택은 매칭NG를 야기한다.

Table 3. ModelForm

|  |  |  |  |
| --- | --- | --- | --- |
| ModelForm | | | |
| 형상 유형 | Point | Triangle(Polygon, Mesh) | Edge |
|  |  |  |
| 특징 | PointCloud만을 가진 모델  Point의 Keypoint, Normal의 방향만을 이용하여 매칭  모델 생성이 빠름  피사체의 형상이 복잡할수록 추천 | 모델생성이 Point형상 보다는 느림  Train View Based속성을 활성화 할 수 있음  *장점: TrainViewBased 활성화 가능,*  *매칭 후 결과 Pose가 모델 생성시 Picking위치 지정에 좀 더 가까움*  *단점: 생성시간 증대*  \*\*notice : 생성, 매칭 단계에서train view based 역할 조사 및 구현추가 필요 | 모델생성이 가장 느림  Triangle Model을 기반으로 Edge Train속성 활성화 가능  *장점*  *피사체의 특징이 없을수록 유리*  *Edge Based 활성화 가능* |

### 3) Triangle Model Params

\*\*notice : Auto와 Default로 두는 것이 PointCloud에서 Mesh로 변환이 더 간단함… 아래 파라메터들을 통해 Auto보다 더 잘 만드는 것이 되려 어렵다…

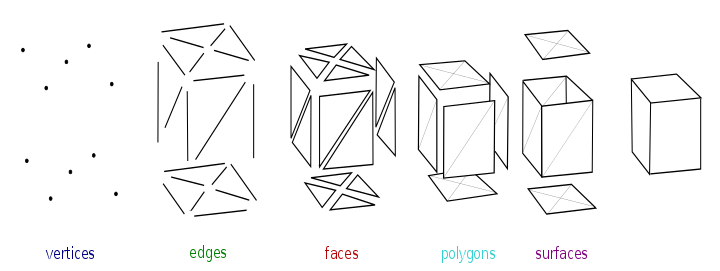
[](https://commons.wikimedia.org/wiki/File:Mesh_overview.svg)

Figure 9. expression type of pointcloud

a) triangulate\_greedyKnnCnt : Triangle의 경계가 될 이웃의 거리값

b) triangulate\_greedyKnnRadiusParam:

Fixed: Selective Neighbor Caching 반경 고정(Radius meter 단위)

z\_factor: PointCloud 좌표값에 밀도가 기록되는 거리센서, TOF 카메라를 사용시 설정

auto: halcon 알고리즘 내부적으로 위 두 기능 중 한 기능 활성화(대부분은 fixed)

c) triangulate\_greedyKnnRadiusValue: Selective Neighbor Caching 반경 거리 값으로 클수록 Model의 형상이 단순해지고 연산량이 적어짐

d) triangulate\_smallsurfaceRemoveValue: smallsurfaceRemove를 하기 위한 제어값 0.0~ 1.0 일 경우 작은 surfaces를 비율에 따라 제거한다

e) triangulate\_greedy\_mesh\_dilationValue: ObjectModel3D의 한 면의 Point간의 Mesh의 촘촘한 연결이 큰 연결로 한 면의 Mesh로 구성되는 주기

### 4) CreateSurfaceModel Params

\*\*more:

1. symmertric을 ObjectModel3D단계에서 할 경우 유저나 개발자가 x,y,z에 대해 대칭을 기입해야하는데 이는 피킹위치 지정에서 또 달라짐… ObjectModel3D 전처리 과정을 다 거치고 피킹 위치 지정 후 train\_self\_similar\_pose로 Halcon이 자동으로 셋팅 가능하니 검토

2. train\_Edge 모델 생성시 ObjectModel3D의 전처리는 피사체의 평면 상태일 때 정상적으로 생성되는데 왜 그런지 확인

3. train\_view\_based를 이용해 view\_base score를 활용 가능하며 기존 모델에 이 속성을 활용시 매칭 결과의 점수는 100~80%에 달한다.

옵션 역할조사 및 구현해도 괜찮다면 옵션 추가.

이전까지의 ObjectModel3D에 대한 후처리가 끝나면 이 모델을 SurfaceModel로 변경하여 매칭에 사용될 수 있도록 한다

매칭시 Model, Scene 간의 KeyPoint들의 Normal의 방향 일치는 매우 중요하다.

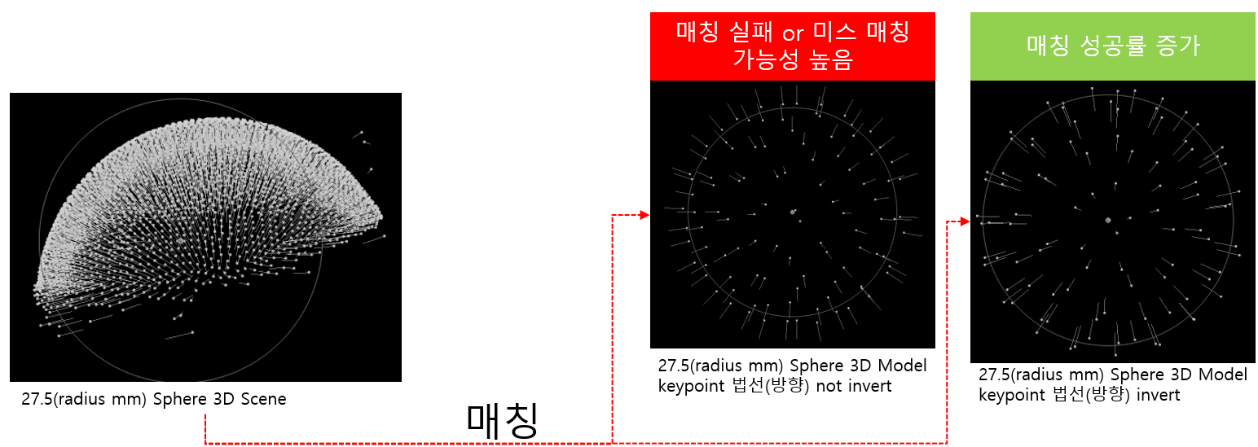


Figure 10. Influence of matching result to normal direction

#### a) create\_sfm\_RelSampleDistance

SurfaceModel 내 KeyPoint간의 배치 거리 값 (=Model의 mm단위 직경 \* RelSampleDistance)

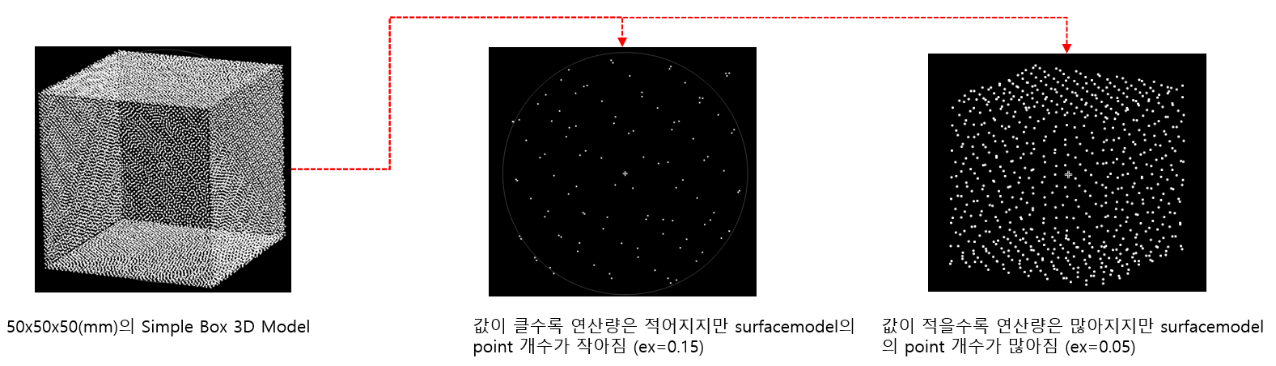


Figure 11. Pointcloud difference of surfacemodel according to Relsampledistance

#### b) CreateSurfModelTimeoutSec

SurfaceModel 연산 타임아웃 값(Sec)

#### c) create\_sfm\_train\_self\_similar\_poses

생성하려는 SurfaceModel의 형상이 ex) Box, Cylinder, Sphere 형과 같이 X, Y, Z의 방향으로 symmetric할 경우 Halcon이 스스로 Model의 Pose를 최적화 한다. 매칭 시 연산시간에 영향이 크다.

추가로 이 속성이 활성화 되어 있을 경우 매칭시 자동으로 train\_self\_similar\_poses를 반영한다.

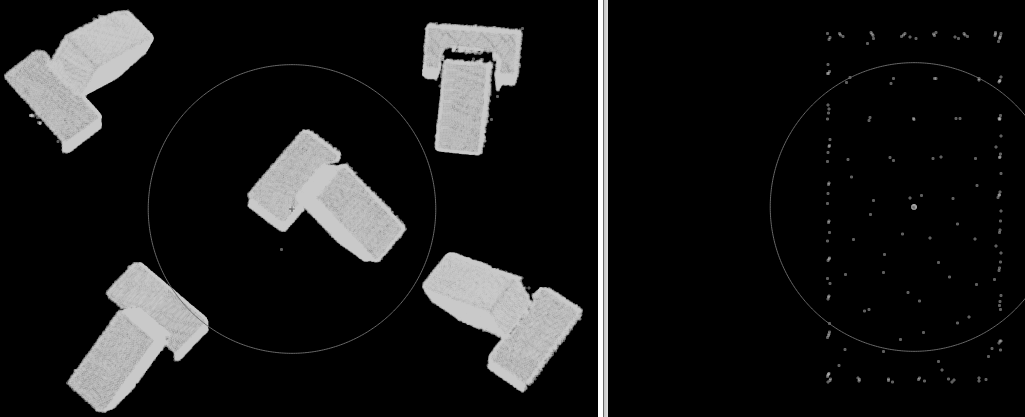


Figure 12. SurfaceModel and Scene used in the train\_self\_similar\_poses test

속성 On. : 47.267 ms 속성 Off : 96.082 ms

#### d) create\_sfm\_train\_view\_based

이 옵션은 SurfaceMatching시 View지점마다 모델이 일부만 보이는 비율을 고려하여 Score 산출

하기 매칭결과 Figure 10. 과 같이 이 옵션이 없다면 Surface Score가 리턴된다.

장점: SurfaceMatching Parameter인 MinScore로 오매칭을 억제하는데 다음과 같은 케이스의 방법이 될 수 있다.

*오매칭된 피사체 위치가 0.2이며, 하기 Scene에서의 피사체도 0.2일 경우 MinScore 0.19일 경우 오매칭 억제 불가*

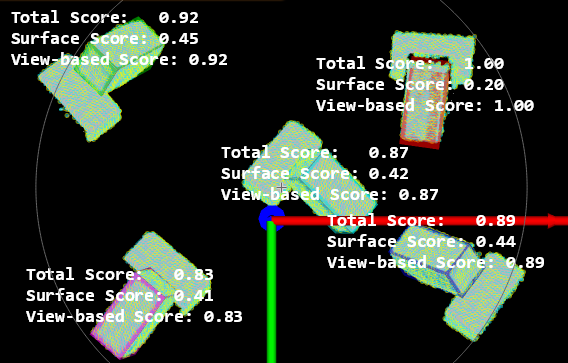
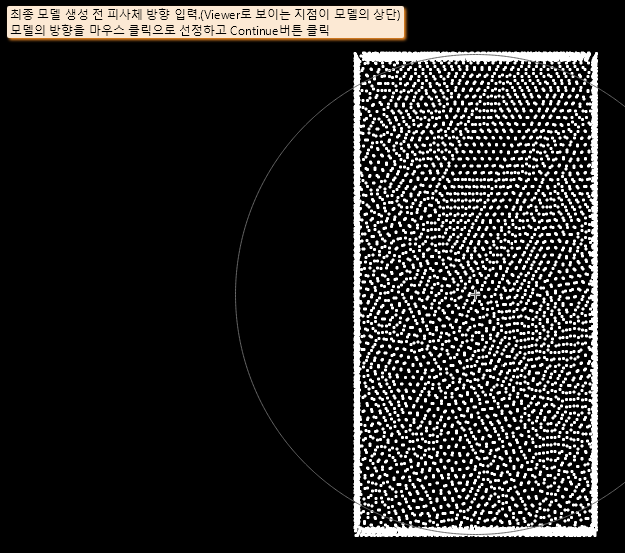
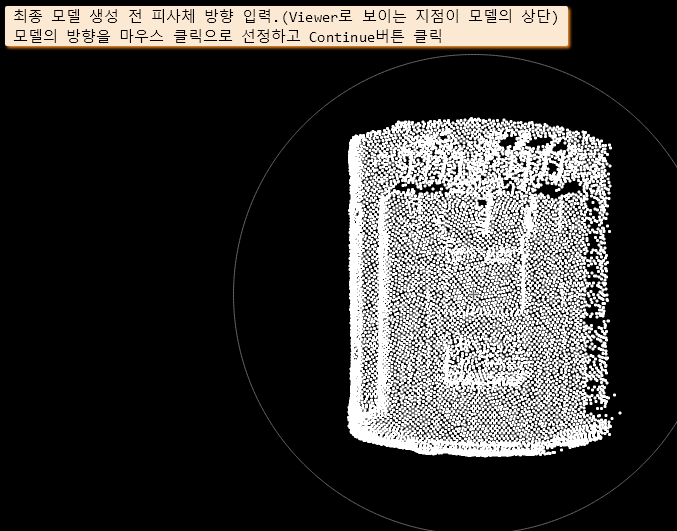


Figure 13. Matching result score using SurfaceModel with Trainviewbased enabled

## 5. SurfaceModel Picking 위치 생성

**Figure 14. Simple Model Figure 15. Image Model**

Image, Simple(자체), Cad 등 ObjectModel3D를 생성하고 샘플링, 엣지, 스무싱 등의 전처리 이후 이 ObjectModel3D는 Visualize\_object\_model3d에서 Picking 위치 지정을 위해 ObjectModel3D를 탑뷰 시점으로 Picking 위치를 지정하며, 여기서 반환되는 Pose(Picking 위치)는 아래와 같으며, 이 Pose를 생성한 ObjectModel3D에 rigid 3d transformation(강체변환: 3DObj의 형상을 보존하며, Trans, Rotation 변형)을 하여 Picking 위치를 부여한다.

**Table 4. Mean of Halcon Pose**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Visualize\_object\_model3d() -> Pose | | | | | | |
| Tx | Ty | Tz | Rx | Ry | Rz | 회전방향, 순서에 대한 속성값 |
| 0 | 0 | Visualize\_object\_model3d  추측 거리 | 0 | 0 | 0 | 0 |

### 1) 피킹 위치 pose값 의미

탑뷰를 조작하지 않을 경우: Visualize\_object\_model3d Operator가 추측하는 Top시점으로 ObjectModel3D 모델이 다 보이는 Tz거리를 갖는 View가 바라보는 피사체의 Pose가 생성

탑뷰를 조작하였을 경우: 사용자가 의도한 Viewr가 Top시점으로 ObjectModel3D 모델을 바라보는 Tz거리를 갖는 피사체의 Pose가 생성

### 2) 피킹 위치 pose값 중 회전 속성의 의미

위 좌표의 7번째 자리 0은 Rp+T gba라는 속성값을 의미하며 회전은 시계방향으로, 회전 순서는 Yaw-Pitch-Roll를 의미한다. Halcon내에서의 변환순서, 회전순서를 따로 설정하지 않는 이상 오퍼레이터에서 반환되는 대부분의 Pose 기본값은 Rp+T, gba 를 갖는다.

**Table 5. Halcon Rotation Property**

|  |  |  |  |
| --- | --- | --- | --- |
| [***OrderOfTransform***](https://www.mvtec.com/doc/halcon/2011/en/create_pose.html#OrderOfTransform) | [***OrderOfRotation***](https://www.mvtec.com/doc/halcon/2011/en/create_pose.html#OrderOfRotation) | [***ViewOfTransform***](https://www.mvtec.com/doc/halcon/2011/en/create_pose.html#ViewOfTransform) | **Code** |
| *'Rp+T' (Transfomation –N 🡪 +N 이동)* | *'gba' (Yaw-Pitch-Roll)* | *'point' (시계방향 회전)* | 0 |
| *'Rp+T'* | *'abg'(Roll-Pitch-Yaw)* | *'point'* | 2 |
| *'Rp+T'* | *'rodriguez'(회전순서 없음)* | *'point'* | 4 |
| *'Rp+T'* | *'gba'* | *'coordinate\_system' (반시계방향 회전)* | 1 |
| *'Rp+T'* | *'abg'* | *'coordinate\_system'* | 3 |
| *'Rp+T'* | *'rodriguez'* | *'coordinate\_system'* | 5 |
| *'R(p-T)'((Transfomation +N 🡪 -N이동)* | *'gba'* | *'point'* | 8 |
| *'R(p-T)* | *'abg'* | *'point'* | 10 |
| *'R(p-T)'* | *'rodriguez'* | *'point'* | 12 |
| *'R(p-T)'* | *'gba'* | *'coordinate\_system'* | 9 |
| *'R(p-T)'* | *'abg'* | *'coordinate\_system'* | 11 |
| *'R(p-T)'* | *'rodriguez'* | *'coordinate\_system'* | 13 |

***\*부록: 위 속성 값 영향은*** [***[@Halcon회전 영향이 있는(pose\_compose)에 관하여]***](#_1._Halcon회전영향이_있는(pose_compose)에) ***참조***

### 3) pose 회전에 관해

회전은 하기의 순서를 따른다.

a) pose\_to\_hom\_mat3d(Pose)-> homogeneous transformation matrix

b) hom\_mat3d\_rotate\_local(homogeneous transformation matrix, phi(ex: 1.57rad=90deg), ‘x’, hom3d\_rotate)

**\* Table 5. Halcon Rotation Property, Rotation Matrix에 이동 및 회전 순서 방향들이 있지만, hom\_mat3d\_rotate\_local은 phi값에 맞게 회전하기에 속성값에 따라 회전이 달라지진 않는다.**

**\* hom\_mat3d\_rotate\_local은 transformation matrix [0,0,0]을 중심으로 회전한다**

c) hom\_mat3d\_to\_pose(hom3d\_rotate)-> pose\_Rotated

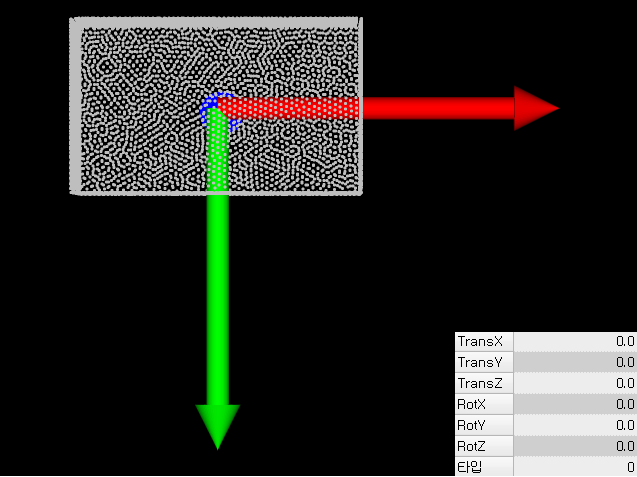
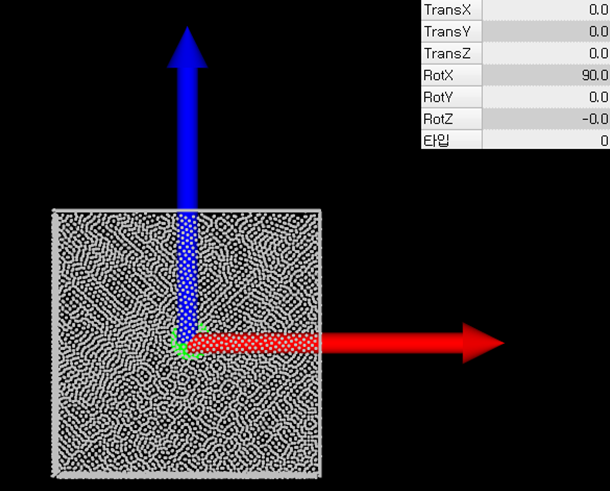
 

Figure 16. Origin Pose Figure 17. X is rotated 90’

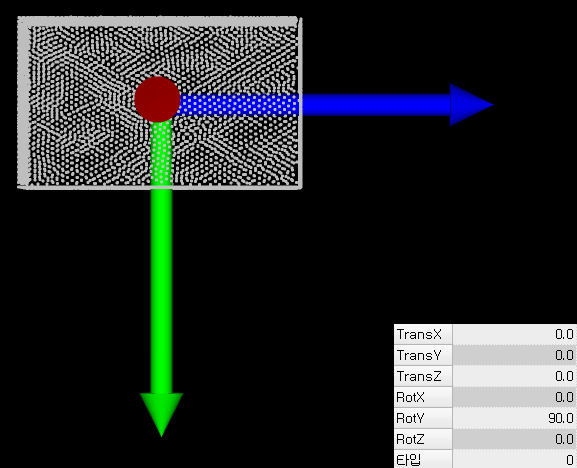
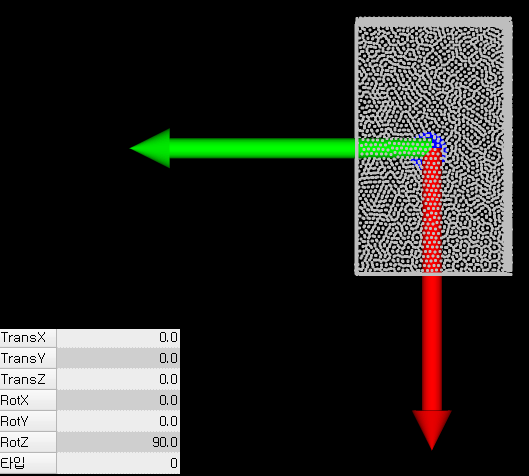
 

Figure 18. Y is rotated 90’ Figure 19. Z is rotated 90’

## 6. SurfaceModel, ObjectModel3D 파일 생성

전, 후처리가 완료된 ObjectModel3D를 SurfaceModel로 만들어 둘 다 파일로 저장한다.

# Surface based Matching (모델-씬 매칭)

## 1. Scene 전처리

### 1) Sampling 3D Object Model Params

Scene은 Camera의 취득 해상도에 영향을 받으며, 매칭용 모델에 비해 많은 PointCloud를 가지고 있으며, 이는 매칭 시 연산속도에 많은 영향을 가져오게 된다. 따라서 Scene을 구성하는 Point개수를 매칭에 필요한 만큼 줄이는 Sampling 과정을 거친다.

*\*매칭 단계에서 Scene의 PointCloud 내의 Keypoint, Normal은 자동생성되며, 하기 fast 옵션을 추천한다.*

#### a) Sampling\_method

accurate: Model Point 간의 Samping 거리 간격 값 사이에 Point를 생성하지만, 그 사이 다른 Point들이 있다면 무게 중심을 계산하여 점을 생성한다. \*연산량이 많기에 Sampling에 시간이 필요

accurate\_use\_normals: ObjectModel3D에 Normal이 있어야하며 이 Normal이 유사한 Point만을 보간한다. \*선택할 수 있는 기능 중 가장 많은 연산량을 가짐

fast: Model의 Point간의 Samping 거리값에 따라 Point를 생성 \*가장 빠름

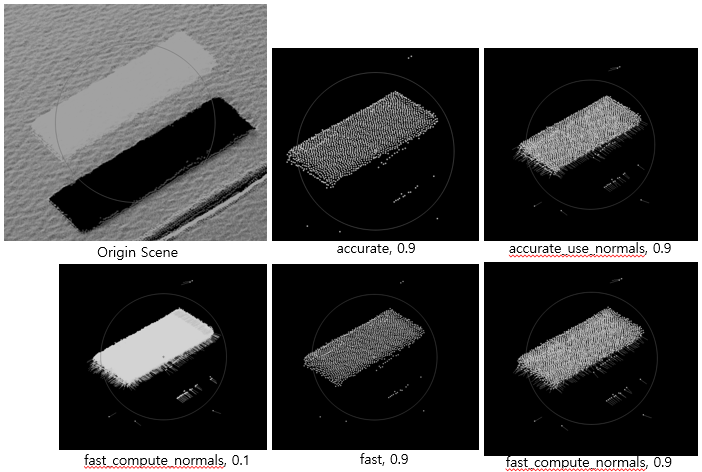
fast\_compute\_normals: Model을 fast로 Samping을 하고 Normal을 생성

#### b) Samping\_distance

- Point간의 샘플링 거리를 설정하며 0.0 ~N 사이의 값을 가질 수 있음

값이 0에 가까울수록 Model을 구성하는 Point의 개수가 증가하며 정교한 ObjectModel3D이 생성됨 \*연산량 증가

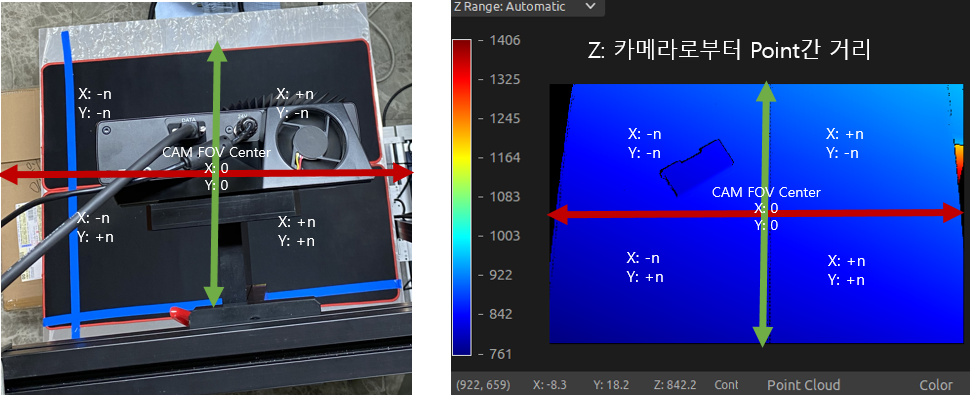
값이 클수록 모델을 구성하는 Point의 개수가 적어지며, 단순한 PointCloud가 형성된 모델이 생성됨 \*연산량 감소



**Figure 20. Pointcloud comparison between each sampling params compare to the Origin Scene**

### 2) RoiForm Params

Sampling 과정을 거친 Scene에 추가로 ROI를 적용할 때의 이점은 불필요한 PointCloud를 삭제함으로, 피사체만을 구성하는 PointCloud로 효율적인 매칭 과정을 거칠 수 있다. Roi영역 지정에 사용되는 해상도는 3D DepthMap의 좌표와 같다.



**Figure 21. Coordinate Presentation of Camera DethMap**

#### a) None

Camera로 취득한 Scene PointCloud 전체를 사용

Scene

Figure 22. None ROI

#### b) Movable Roi

Scene에서 지면을 제거하고 특정 범위의 X영역을 스캔하여 피사체 매칭

Scene

ROI

Figure 23. Movable ROI

- ScanXArea: X축 탐색 범위 값 0 ~ N

- ScanXOverwrap: x축 탐색 이동량 값, ScanXArea보다 작을 경우 탐색시 겹치는 영역이 발생한다

\**하기 Z축에 대한 두 값은* 지면(0값)을 기준으로 범위가 설정됨

- ROIZPlaneMinDepth: 지면(0)을 기준으로 Z축 최소 값

- ROIZPlaneMaxDepth 지면(0)을 기준으로 Z축 최대 값:

\*\*bug ROI, ModelForm, ROIBOX, ObjectArrow 로직 추가 후 정상동작 불가

\*\*more 매칭과정이 시간이 걸리기 때문에 잘라놓고 매칭 가능한지 검토 필요

#### c) Static Roi

Scene PointCloud에서 사용자가 지정한 X, Y, Z 특정 영역만을 잘라내어 Scene을 구성

Scene

ROI

Figure 24. Static ROI

- ROIXAreaMin: Depth X축 좌측 좌표 범위 (-N ~ 0)

- ROIXAreaMax: Depth X축 우측 좌표 범위 (0 ~ +N)

- ROIYAreaMin: Depth Y축 상단 좌표 범위 (-N ~ 0)

- ROIYAreaMax: Depth Y축 하단 좌표 범위 (0 ~ +N)

- ROIZPlaneMinDepth: 지면(0)을 기준으로 Z축 최소 값

- ROIZPlaneMaxDepth 지면(0)을 기준으로 Z축 최대 값:

\* Static ROI 접근 방법

- Sampling처리를 거친 Scene\_object\_model3d에 fit primitive object model3d(plane)을 이용해 object\_model3d로 변환한다.

🡪 fit primitive object model3d()에서는 halcon이 plane으로 변환한, ①*카메라가 바라보는 plane object\_model3d(지면)*와 ②*plane 위치의 “pose”*가 나온다.

- Sampling된 Scene\_object\_model3d와 ③*plane 위치의 pose를 Invert한 “pose”*의 homogenous\_transformation matrix를 affine\_trans를 하여 ④*plane object\_model3d(지면)이 카메라를 바라보는 affine\_object\_model3d*가 생성된다.

**②plane 위치의 pose 의미**

ex) tX:-13.4, tY: -6.5, tZ: 635.628, rx, ry, rz

tX, tY : 카메라 중심으로부터 plane의 중심은 -13.4, -6.5 위치

tZ: 거리는 635.6mm에 있음

③**plane 위치의 pose를 invert한 pose의 의미**

ex) tX:-25.4 , tY: -229.4 , tZ: 592.4, rx, ry, rz

tX, tY : plane 중심으로부터 카메라의 중심은 -25.4, -229.4 위치

tZ: 거리는 592.4mm에 있음

**①카메라가 바라보는 plane object\_model3d**

**cam**

**scene plane**

tX

tY

tZ: 0~N\_inf

tX:-13.4, tY: -6.5, tZ: 635.628, rx, ry, rz

**④plane object\_model3d(지면)이 카메라를 바라보는 Affine\_object\_model3d**

**cam**

**scene plane**

tX

tY

tZ: -N ~ N\_inf

tX:-13.4, tY: -6.5, tZ: 635.628, rx, ry, rz

Figure 25. plane object model3d and affine\_ object model3d with different viewpoints, and the meaning of each pose

- affine\_object\_model3d활용

지면이 카메라를 바라보는 pointcloud에서 tZ는 자기 자신(-N~N\_inf mm)을 기준으로 카메라 이후까지 거리를 가지게 된다.

**즉, -n~5mm 정도의 tZ범위를 무시하고 이후 범위의 pointcloud만을 선택하여 다시 object\_model3d를 만들게 된다면 타겟이 되는 피사체만의 pointcloud를 얻을 수 있다.**

#### d) AutoRoi

Scene PointCloud에서 피사체를 표현하는 PointCloud를 자동으로 잘라내어 Scene을 구성

Scene

ROI

Figure 26. Auto ROI

\* AutoROI 접근 방법

*- StaticROI 접근방법 - affine\_object\_model3d활용*을 기초로, 지면 pointcloud를 의미하는 -n~5mm 정도의 tZ범위를 의미하는 pointcloud의 개수에 대한 가정이다.

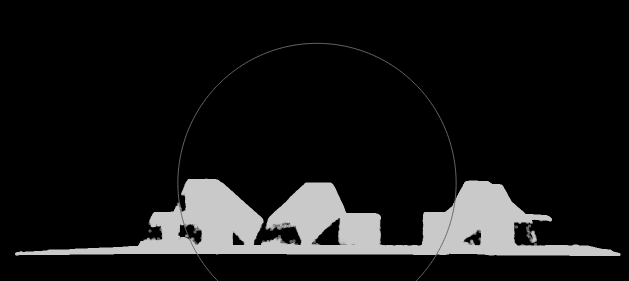


Figure 27. tZ -7.0 ~ 592.245 mm range of affine3d scene and

number of pointcloud’s affine3d scene cut in 1mm table

-7.0 mm

-6.0 mm

-5.0 mm

-4.0 mm

-3.0 mm

-2.0 mm

-1.0 mm

+0.0 mm

+1.0 mm

+2.0 mm

+3.0 mm

Figure24에서 파란색 선은 1mm으로 자른 scene들을 의미하며, 우측의 표를 보면 각 scene의 pointcloud의 개수의 특징을 보자면 지면에 해당하는 지점의 pointcloud가 가장 많고 피사체에 근접할수록 그 개수가 적어진다.

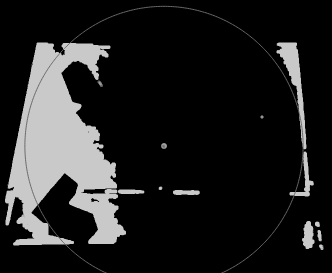
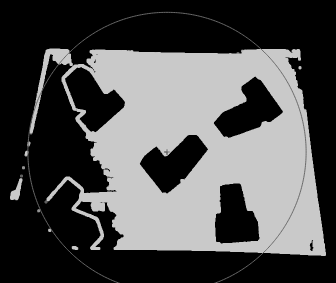
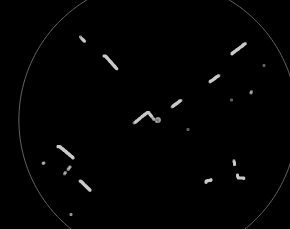
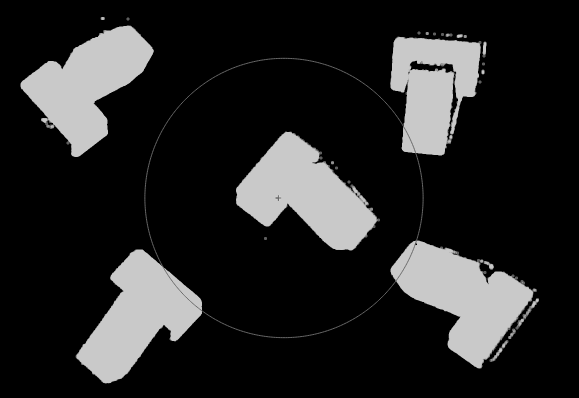
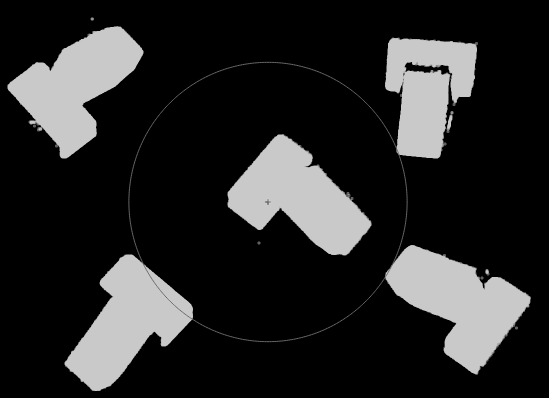
   

Figure 28. tZ -7.0mm sliced affine scene Figure 29. tZ -6.0mm sliced affine scene Figure 30. tZ 0.0mm sliced affine scene Figure 31. tZ +3.0mm sliced affine scene

**Figure 32. tZ 0.0mm ~ +592mm range of affine3d scene Figure 33. tZ 3.0mm ~ +592mm range of affine3d scene**

Figure 29는 0.0mm부터 이후까지의 pointcloud를 재구성한 것으로 우측 상단, 하단의 피사체 경계에 지면에 해당하는 point가 붙어있다.

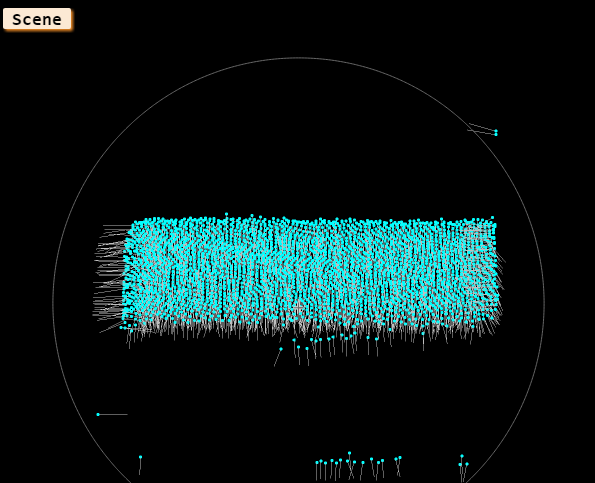
반면, Figure 30은 3.0mm부터 이후까지의 pointcloud를 재구성한 것으로 Figure 29와 달리 우측 상단, 하단의 피사체 경계의 지면이 제거되어 있다. 즉, 지면을 제거해도 어느 정도의 margin을 더해 pointcloud를 재구성하는 것이 피사체만을 획득하는 좋은 방법이 된다.

유저가 잘린 각각의 pointcloud의 개수를 알 수 없으며, scene의 구성마다 point의 개수는 차이가 너무 크다. 따라서 현재 자른 씬의 point갯수와 이전 point개수의 차 값을 유저가 설정하게 하였으며, **이 값은 피사체를 자르기 시작한 구간(Figure24 -1.0 ~ +3.0mm)부터는 현재와 이전의 pointcloud 개수의 차 값이 비슷해진다는 가정에서 접근하였다.**

***자동으로 자를 지면의 margin값(parameter) == (1mm씩 자른 씬의 point 개수의 abs(현재-이전) 차 값)***

## 2. SurfaceMatching 방법

### 1) SurfaceMatching Parameters

a) find\_sfm\_RelSamplingDistance : Scene PointCloud에서 생성하는 KeyPoint간 거리

**Figure 34. RelSamplingDistance: 0.01 Figure 35. RelSamplingDistance : 0.1**

이 값이 작을 경우: Scene내의 KeyPoint 개수 많아짐

장점: SurfaceModel과 Scene 간의 정교한 매칭이 가능

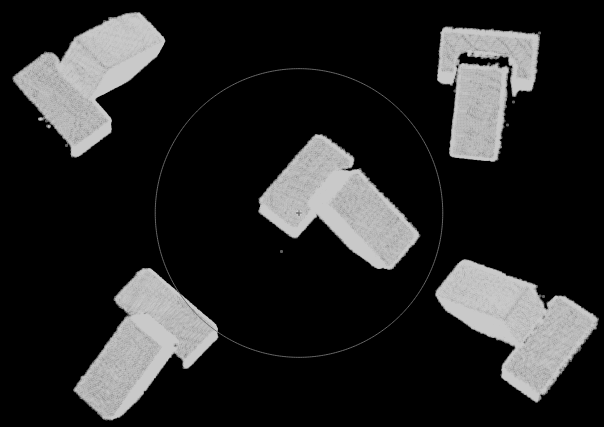
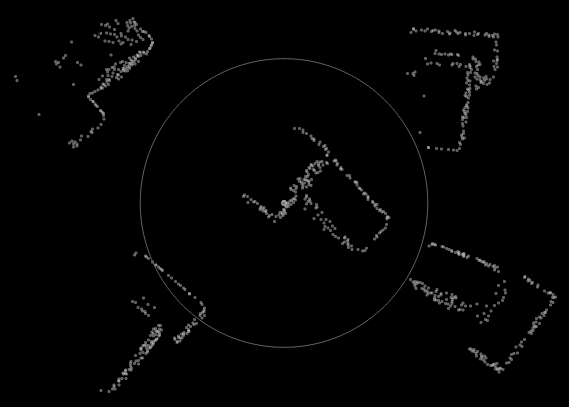
단점: 많은 연산량으로 인해 매칭연산이 오래 걸림

이 값이 높을 경우: Scene내의 KeyPoint 개수 적어짐

장점: 연산량이 낮아 빠른 매칭가능

단점: 적은 KeyPoint개수로 인해 오 매칭 확률이 높음

b) find\_sfm\_KeyPointFraction: Sampling된 Scene에서 선택되는 KeyPoint 개수의 비율을 의미한다. 정밀한 Matching을 필요로 한다면 SurfaceModel과 Scene의 Sampling 거리를 최소화하고 이 값을 높은 비율로 설정이 가능하다. 이 값이 높을수록 연산량은 증대된다

**Figure 36. Sampling Scene Figure 37. Scene KeyPoint**

c) find\_sfm\_MinScore: SufaceModel과 일치한다고 판정하는 최소 점수

d) find\_sfm\_NumMatch: Scene에서 찾아야하는 SufaceModel의 개수

e) find\_sfm\_FindMethod: ‘Scene\_normal\_computation’

fast: z축을 따라 카메라가 바라보는 방향 그리고 small neighborhood 방식으로 normal이 생성된다

mls: z축을 따라 카메라가 바라보는 방향 그리고 small neighborhood 방식으로 normal이 계산되고 larger neighborhood로 재계산하여 normal을 생성한다. 이 옵션의 경우 잡음이 심한 scene일수록 유리하다

\*\*notice 변수명 교체 필요 find\_sfm\_FindMethod -> ‘find\_sfm\_scene\_normal\_computation’

f) find\_sfm\_ScoreType:

g) model\_point\_fraction: SurfaceModel의 Point갯수 대비 Scene에서 찾아낸 SurfaceModel의 Point의 비율 점수

h) num\_scene\_point: Scene에서 찾아낸 SurfaceModel의 Point개수 가중치를 계산 후, 각 Point에서 찾아낸 SurfaceModel까지의 거리를 기준으로 하는 가중 점수

\*\*more SurfaceModel이 피사체의 360’ 모두를 표현하는 PointCloud를 가지고 있고, Scene에서의 타겟은 일부만 보이기 때문에 model\_point\_fraction 디폴트 점수 비율이 20~50% 수준. 이 대안으로 num\_scene\_point를 사용하였으나, 점수가 거리가중방식이라 500% 1000%에 달한다. 점수를 어떻게 정의 할 것인지 결정 필요. ex)train\_view\_based 점수를 기존 모델에서 테스트할 경우 90%~100%의 점수가 나오는데 어떻게 나오는 것인지 확인 필요

i) find\_sfm\_max\_overlap\_dist\_value: Scene에서 찾아낸 SurfaceModel이 중첩인지 아닌지 거리를 측정하는 방식

max\_overlap\_dist\_rel: 연관거리 사용

max\_overlap\_dist\_abs: 절대값 거리 사용

j) find\_sfm\_max\_overlap\_dist\_value: Scene에서 찾아낸 SurfaceModel이 중첩될 경우 그 거리에 따라 중첩을 할지 말지 정하는 거리비율

max\_overlap\_dist\_rel: 1일 경우 중첩 불가 0.0~1 (def:0.5)

max\_overlap\_dist\_abs: 1일 경우 중첩 불가 1~n

k) find\_sfm\_pose\_ref\_use\_scene\_normals\_value: Scene의 Normal을 매칭에 사용 on, off값

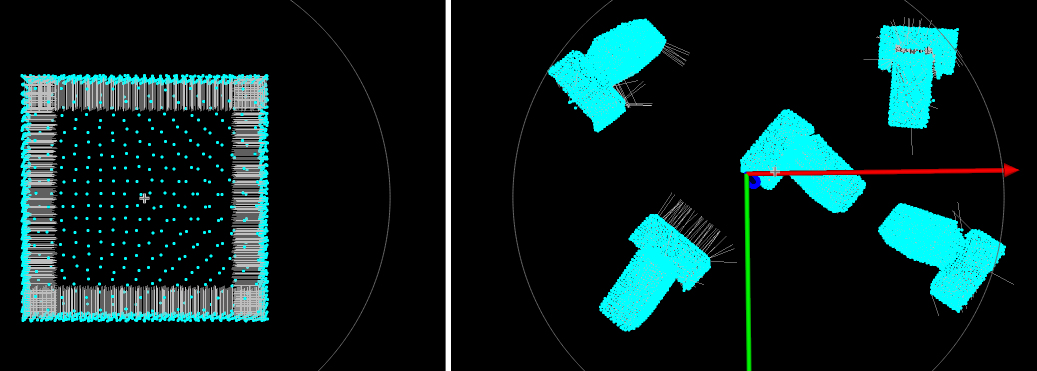


Figure 38. Normal direction comparison between Scene and SurfaceModel

\*\*more Scene에서 피사체의 Normal은 카메라가 바라보는 Z축을 따라 형성되기 때문에 생성한 SurfaceModel과는 방향 유형, 보이는 부분이 분명 다르다 정확도에 영향을 주는지 확인 필요

l) find\_sfm\_pose\_ref\_num\_steps\_value: dense pose refinement 반복 횟수 지정

값이 높을수록 정확도 향상 연산시간 증대, 특정횟수에서 최적화되면 그 이상 지정해도 무의미.

단순한 SurfaceModel의 경우 default 값 5회가 많을 수 있음

m) find\_sfm\_pose\_ref\_sub\_sampling\_value: dense pose refinement에 사용되는 Scene point 비율

값이 높다면 사용되는 Point는 낮고, 정확도, 연산시간 감소

## 3. Surface based Matching 단계

### 1) Approximate matching 대략적인 매칭

Scene이 들어왔을 때 Point는 입력된 RelSamplingDistance 값에 따라 샘플링되며, 이 샘플링 단계에서 Keypoint들이 생성된다.

CreateSurfaceModel에서 생성된 SurfaceModel과 Scene의 Keypoint를 얼마나 사용할 것인지 KeyPointFraction 값에 따라 비율이 지정된다.

Scene에서 SurfaceModel과 비슷한 거리와 상대적인 방향을 가진 Keypoint를 모든 샘플링 장면마다 비교하며 찾아낸다.

여기서 거리와 상대적인 방향이 가장 일치한 높은 점수를 가진 Pose를 반환한다.

### 2) Sparse pose refinement 희소 포즈 조정

이 단계에서는 Approximate Matching 단계에서 샘플링 된 Scene Point를 사용하여 Pose를 최적화한다.

샘플링된 Scene Point에서 가장 가까운 SurfaceModel Point의 평면까지 거리를 최소화한다.(=Scene Point와 SurfaceModel의 형상이 가장 일치하는 것을 본다는것으로 생각?)

여기서 각 SurfaceModel Point의 평면은 Normal에 수직인 평면으로 정의(=모델의 형상을 말하는 것으로 보인다?).

### 3) Dense pose refinement 밀집 포즈 조정

이 단계는 Sparse Pose Refinement와 유사하게 동작하며, Scene Point와 가장 가까운 SurfaceModel 점의 평면 사이의 거리를 최소화 한다.

위 단계와의 차이점은 이전 단계에서 가장 높은 점수를 가진 Pose만 조정되며, Scene의 샘플링 Point를 전부 사용할 것인지, 몇 개를 건너 뛰며 사용할 것인지 정할 수 있다.

## 4. Surface based Match 구동 방법

1) Create\_Surface\_Model에서 저장한 모델(3D Obj, Surface Model)을 불러온다

2) 취득한 Scene에 대해 2D image분리, Sampling, 및 ROI 적용(지면 PointCloud제거 등)을 한 다음 Scene에서 불러온 SurfaceModel과 KeyPoint가 가장 유사한 것을 찾아 그 Pose를 결과로 반환한다.

3) Create\_Surface\_Model에서 생성된 SurfaceModel로 찾아낸 결과로 반환된 Pose는 Trans축은 카메라가 바라본 Scene기준의 DepthMap Coordinate의 위치이며, Rotate축은 SurfaceModel이 가진 Rotate기준으로 카메라가 바라본 Scene에서 Rp+T gba방법으로 얼마나 회전했는지를 표현한다

## 5. 일부 매칭 사례

Pose가 CreateSurfaceModel에서 지정한 피킹 위치대로 피사체가 매칭이 되지 않는 이유

1) Sparse, Dense Pose Refinement단계에서 Normal의 평면 사이의 거리를 최소화 하는데, 여기서 Scene의 Normal의 방향, 그리고 튀는 Normal의 방향으로 인해 SurfaceModel의 Normal과 일치하지 않게 되어 Scene에서 SurfaceModel의 평면이 다르게 생성되어 발생하는 문제

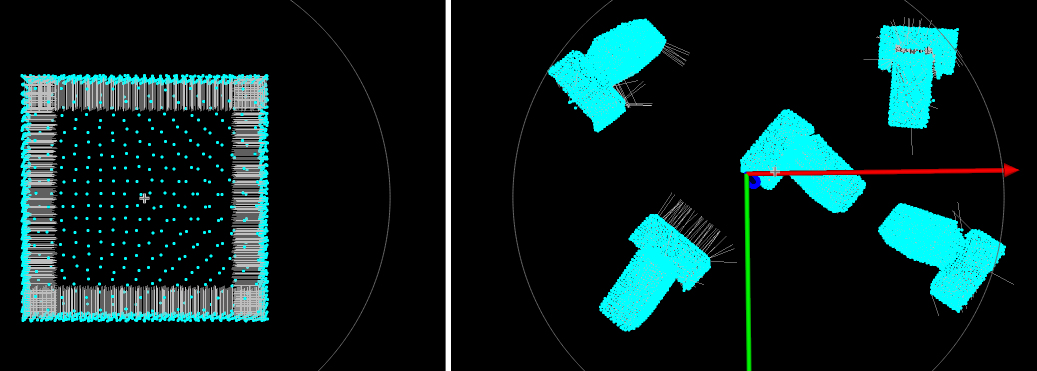


Figure 39. The direction of the scene and some other normals that are only partially visible compared to the SurfaceModel



Figure 40. Left: Specified picking position of SurfaceModel, Right: SurfaceMatching returns the result of the subject's orientation reversed

## 6. 좌표 정렬

### 1) 피사체 중심 원점을 기준 Tz’ 90, 180’ 위치할 경우

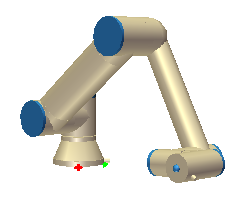
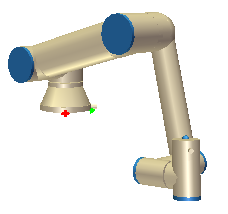
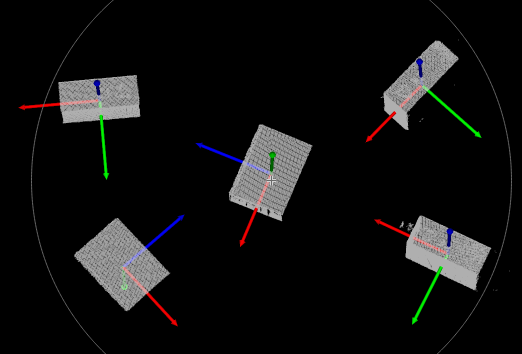


Figure 41. Rotation Pose 180'(White Box), 90’(Yellow Box) flipped and Picking Poses (Red: tX, Green: tY, Blue: tZ)

Figure 38’의 좌측 매칭 결과에서 tZ(청색화살표)가 수평에 위치한 노란색 박스의 매칭 결과들은 UR Robot이 수평으로 Picking

Figure 38’의 좌측 매칭 결과에서 tZ(청색화살표)가 수직 상단에 위치한 흰색 박스의 매칭 결과들은 UR Robot이 수평으로 Picking

### 2) Pointcloud Bounding box의 중심 Trans X, Y, Z활용

Figure 42. Matching result for box and Bounding Box shape

Bounding box는 피사체를 감싸는 가장 작은 사각형 box로 피사체의 중심, 피사체의 길이, 좌표 내의 위치 등의 정보를 알아낼 수 있으며, ROI, 매칭, Indexing에 활용되는 메타데이터의 성격을 지니고 있다.

매칭하고자 하는 Object가 사각형 상자이고 여기에 Bounding box를 씌우게 되면 Figure 39와 같이 피사체와 동일한 위치, 크기, 좌표 위치를 가진 Bounding box가 형성된다.

Halcon에서 Bounding Box의 정보는 매칭된 피사체의 Point가 위치한 Trans X, Y, Z 최소, 최대 위치 값이며 다음과 같다.

Min x ~ Max x, Min y ~ Max y, Min z ~ Max z (Depth Map coordinate) *참고* [*2) RoiForm Params Figure18*](#_2)_RoiForm_Params)

여기서 Figure 38과 같이 로봇이 집을 수 없는 매칭 결과가 발생한다면 tZ방향을 피사체의 중심축을 기준으로, 형상을 고려하여 90’ 혹은 180’를 회전 시키거나 특정 축을 0’로 처리하는 등의 좌표정렬을 매칭결과Pose에 대해 후처리 하여 Robot의 원활한 Picking을 지원한다.

a) Symmetric 정렬 옵션 1

- 정육면체와 같이 어떻게 회전하여 집어도 상관없는 모델일 경우 Tz를 Robot이 위에서 아래로 집는 자세가 나올 수 있도록 rX, rY를 90’ 회전시키면서 tZ를 회전정렬.

\*\*more 피킹위치 지정 사진하고 rX, rY 90’ 무작위회전하는 사진, 결과 예시 사진 필요

b) Symmetric 정렬 옵션 2

- 직육면체와 같은 경우 Picking위치를 보존하면서 피사체의 Tz를 정렬해야한다. 이 경우 rY를 180’처리한다.

Picking위치를 보존하면서 회전하기 위해서는 최초 등록한 모델 대비 피사체의 tX, tY, tZ가 어느 방향으로 향해 있는지를 알아야한다.

가정은 다음과 같다.

Figure 43. Matching result for box and Bounding Box shape

- 매칭된 피사체의 좌표에 tX, tY, tZ를 의미하는 화살표 arrow\_object\_model3d를 생성한다.

- 회색상자는 매칭된 피사체의 Bounding Box를 나타낸다.

- 여기서 매칭 결과가 어떻게 놓여있는지 확인하기 위한 arrow\_object\_model3d를 위치시킬 Pose를 만든다.

- Point가 위치한 tX, tY, tZ의 Min, Max의 중간 값을 TransPose로, 매칭결과좌표 의 rx, ry, rz를 이용하여 하나의 Pose를 만들어 이 위치

에 arrow\_object\_model3d를 입힌다. 이 형상은 Figure 40과 같다.

- 세가지 arrow\_object\_model3d은 각각의 Bounding box를 가지며, 이 bounding box의 tX, tY, tZ의 Min Max를 지니고 있다. 원기둥 형태의 pointcloud기 때문에 tx, ty, tz 한 방향으로 가장 높은 min, max편차를 가지고 있으며, 이 값으로 화살표가 어느 방향으로 향하는지 1차로 확인할 수 있다.

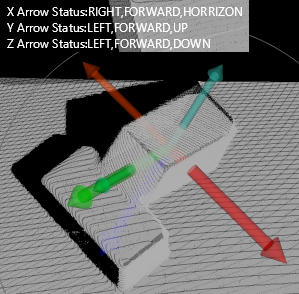
Figure 44. arrow\_object\_model3d and Bounding Box shape

- 위 Bounding box tX, tY, tZ의 abs(Max-Min)을 하여 arrow\_object\_model3d\_X, arrow\_object\_model3d\_Y, arrow\_object\_model3d\_Z의 tX, tY, tZ 길이를 구한다. 이 길이의 예시 테이블은 Table 6과 같다.

**Table 6. example of arrow\_object\_model3d\_X, arrow\_object\_model3d\_Y, arrow\_object\_model3d\_Z Length Table**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Tx Arrow BoundBox** | | | **Ty Arrow BoundBox** | | | **Tz Arrow BoundBox** | | |
| XLength | YLength | ZLength | XLength | YLength | ZLength | XLength | YLength | ZLength |
| 11.9 | 31.4 | 76.3 | 38.9 | 68.9 | 28.6 | 72.6 | 36.8 | 20.0 |
|  |  | Tz방향 |  | Ty방향 |  | Tx방향 |  |  |

- 여기서 각 축 arrow\_object\_model3D의 XLength, YLength, ZLength 중 제일 큰 값이 화살표가 향하는 방향을 의미하며, 이 값들과 피사체의 중심 tX, tY, tZ 좌표와 비교를 하여 각 화살표가 어느 방향으로 향해 있는지 알 수 있다.

- 피사체의 중심 tX, tY, tZ기준으로 arrow\_object\_model3D\_X,Y,Z의 중심 tX, tY, tZ가 어느 방향으로 향하고 있는 지에 대한 정의와 테스트결과는 Figure 42와 같다.

Forward

Backward

Right

Left

Up

Down

X,Y = Center

Z = Horizon

Figure 45. example of X, Y, Z arrow definitions and result

- 화살표가 어떻게 놓여있는 지 DepthCoodinate 기준으로 3차원 상에서 어떻게 판단하는 방법은 다음과 같다.s

arrow\_object\_model3D\_X:

피사체 중심 X축 대비 : X를 나타내는 화살표의 X축은 우측에 위치

피사체 중심 Y축 대비 : X를 나타내는 화살표의 Y축은 전방에 위치

피사체 중심 Z축 대비 : X를 나타내는 화살표의 Z축은 수평에 위치

arrow\_object\_model3D\_Y

피사체 중심 X축 대비 : Y를 나타내는 화살표의 X축은 좌측에 위치

피사체 중심 Y축 대비 : Y를 나타내는 화살표의 Y축은 전방에 위치

피사체 중심 Z축 대비 : Y를 나타내는 화살표의 Z축은 상단에 위치

arrow\_object\_model3D\_Z

피사체 중심 X축 대비 : Z를 나타내는 화살표의 X축은 좌측에 위치

피사체 중심 Y축 대비 : Z를 나타내는 화살표의 Y축은 전방에 위치

피사체 중심 Z축 대비 : Z를 나타내는 화살표의 Z축은 하단에 위치

- Center, Horizon의 표현방안: arrow\_object\_model3D\_X, Y, Z 모델의 tX, tY, tZ 각각의 길이를 구해 그 값이 *\*\*30mm* 이하일 경우 Center, Horizon에 있다고 판단한다. Figure 43이 세가지 arrow\_object\_model3D 중 arrow\_object\_model3D\_X일 경우 Center, Center, Up의 결과가 나올 것이며, View상으로는 tX를 의미하는 적색 화살표가 카메라가 바라보는 시점으로 향하고 있을 것이다.

tXLength : Center

tZLength: Up

tYLength : Center

Figure 46. example of arrow\_object\_model3d X result location

\*\*more: Center, Horizon을 지정하는 범위 값을 30mm으로 지정한 상태이며, 화살표가 대각선일 경우 up과 down이 모호한 부분이 있음. 다른 접근 방법 필요..

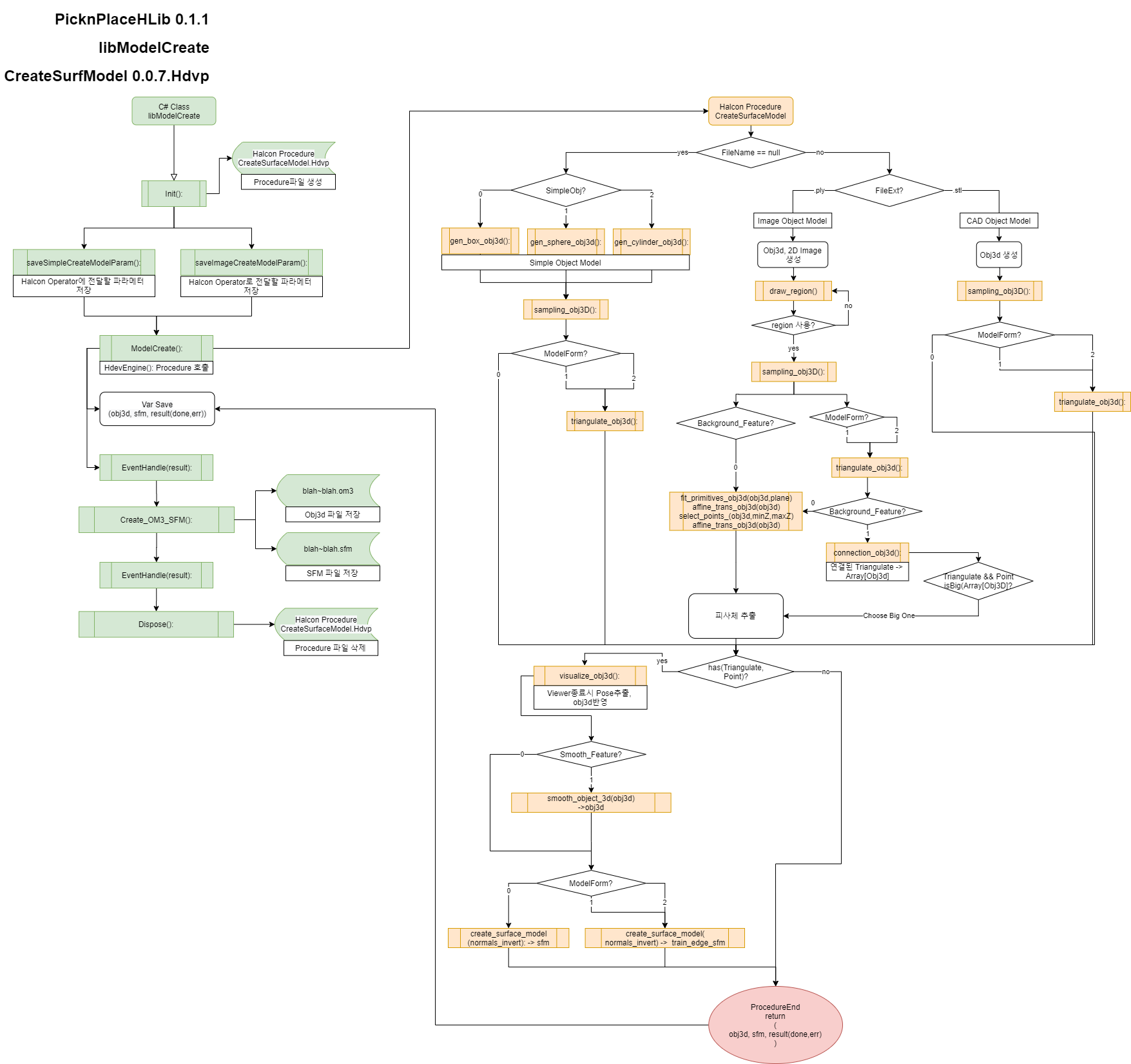
c) Symmetric 정렬 옵션 3

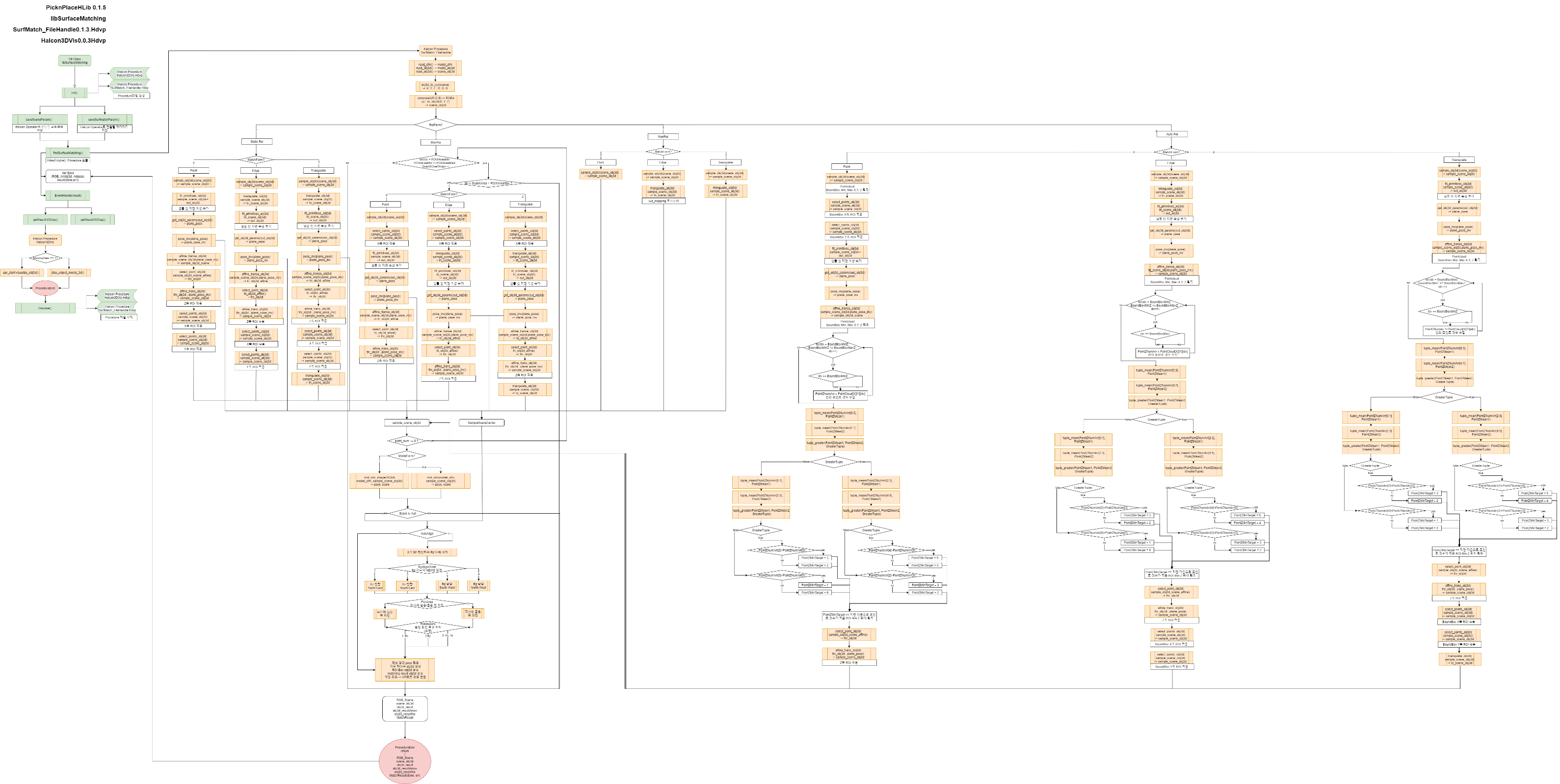
- 구체의 경우 피사체 위에서 수직으로 피킹. 여기에는 Rz를 어느 방향으로 정렬할 것인가에 대한 옵션

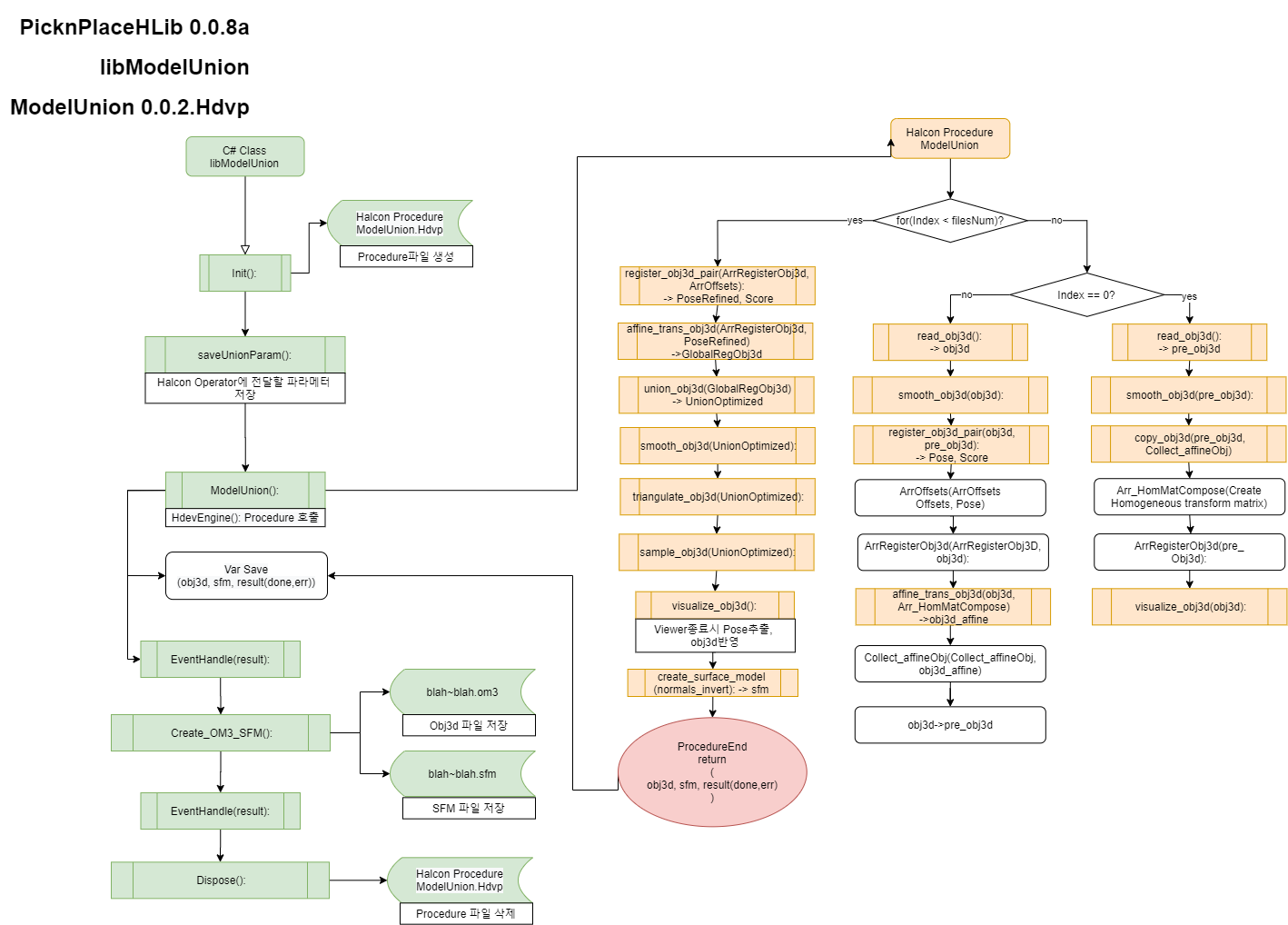
\*\*more 피킹위치 지정 사진 매칭 결과, 그리고 이 옵션을 이용한 결과 예시 사진 필요

\*\*notice 직육면체, 정육면체, 구, 실린더 등에 따라 90, 180’등 어떻게 정렬할 것인가에 대한 옵션 필요 Pickit3D how to pick - youtube영상 추후 참조 하여 기능 추가

# UML Diagram







# AG\_Vision Protocol

**v3.3**

|  |  |  |
| --- | --- | --- |
| Data |  | Version |
| 2020년 3월 10일 | 초기 프로토콜 생성 | 1.0 |
|  | String 통신 : 시작(>),끝(<),구분자(:) |  |
| 2020년 4월 10일 | 로봇 통신 중지, Next Pose 요청 추가 | 2.0 |
| 2020년 6월 16일 | Python 통신내용 삭제 | 3.0 |
| 2020년 9월 9일 | 로봇 Mode 추가 | 3.2 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **현재 위치 요청** | | | |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **MAIN** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **ROBOT** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **CPOSE** |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **ROBOT** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **MAIN** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **CPOSE** |  | | | | | | |
| **N-04** | **동작 확인** | | | **OK** | **OK or NG** | | | | | | |
| **N-05** | **로봇 현재 위치** | | | **p[0.0,0.0,0.0,0.0,0.0,0.0]** | **X,Y,Z,rX,rY,rZ** | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **절대값 이동** | | | |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **MAIN** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **ROBOT** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **MOVE** |  | | | | | | |
| **N-04** | **MOVE OPTION** | | | **CAL or PNP** | **CAL 모드 or Pick & Place 모드** | | | | | | |
| **N-05** | **MODE** | | | **0~N** | **0 : robot pose 1:robot offset** | | | | | | |
| **N-06~10** | **로봇 타겟 위치** | | | **0.0:0.0:0.0:0.0:0.0:0.0** | **X,Y,Z,rX,rY,rZ** | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **ROBOT** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **MAIN** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **MOVE** |  | | | | | | |
| **N-04** | **동작 확인** | | | **OK** | **OK or NG** | | | | | | |
|  |  | | |  |  | | | | | | |
|  |  | | |  |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **Next pose 요청** | | | | **비전 프로그램 다음 매칭 진행** |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **ROBOT** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **MAIN** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **NEXTPOSE** |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **티칭 모드** | | | | **Freedrive Mode** |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **MAIN** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **ROBOT** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **TEACH** |  | | | | | | |
| **N-04** | **스위치** | | | **ON or OFF** | **ON:활성 OFF:비활성** | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **ROBOT** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **MAIN** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **TEACH** |  | | | | | | |
| **N-04** | **동작 확인** | | | **OK** | **OK or NG** | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **~~통신 유지~~** | | | |  |  |  |  |  |  |  |  |
| **~~순번~~** | **~~데이터 종류~~** | | | **~~Data~~** | **~~상세 설명~~** | | | | | | |
| **~~N-01~~** | **~~Sender~~** | | | **~~MAIN~~** | **~~전달자~~** | | | | | | |
| **~~N-02~~** | **~~Reciever~~** | | | **~~ROBOT~~** | **~~수신자~~** | | | | | | |
| **~~N-03~~** | **~~Command~~** | | | **~~ACK~~** |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **~~순번~~** | **~~데이터 종류~~** | | | **~~Data~~** | **~~상세 설명~~** | | | | | | |
| **~~N-01~~** | **~~Sender~~** | | | **~~ROBOT~~** | **~~전달자~~** | | | | | | |
| **~~N-02~~** | **~~Reciever~~** | | | **~~MAIN~~** | **~~수신자~~** | | | | | | |
| **~~N-03~~** | **~~Command~~** | | | **~~ACK~~** |  | | | | | | |
|  |  | | |  |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **통신 정지** | | | | **프로그램 정지** |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **MAIN** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **ROBOT** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **STOP** |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **ROBOT** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **MAIN** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **STOP** |  | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **로봇 모델 매칭 요청** | | | |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **ROBOT** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **MAIN** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **DETECT** |  | | | | | | |
| **N-04** | **모델번호** | | | **0~N** | **MODEL IDX** | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |
| **순번** | **데이터 종류** | | | **Data** | **상세 설명** | | | | | | |
| **N-01** | **Sender** | | | **MAIN** | **전달자** | | | | | | |
| **N-02** | **Reciever** | | | **ROBOT** | **수신자** | | | | | | |
| **N-03** | **Command** | | | **DETECT** |  | | | | | | |
| **N-04** | **모델번호** | | | **0~N** | **MODEL IDX** | | | | | | |
| **N-05** | **매칭결과** | | | **OK** | **OK or NG** | | | | | | |
| **N-06** | **MODE** | | | **0~N** | **0 : robot pose 1:robot offset** | | | | | | |
| **N-07~12** | **로봇좌표** | | | **0.0,0.0,0.0,0.0,0.0,0.0** | **X,Y,Z,rX,rY,rZ** | | | | | | |

# Test Result Summary

Accuracy

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **모델생성 방법** | **테스트 객체** | **축 편차**  **[mm]** | **회전 편차**  **[rad]** | **처리 시간**  **[sec]** |
| **SIMPLE**  (객체 수치 입력) | **구** | **0.51** | **-** | **1.83170** |
| **원통** | **3.398** | **0.223** | **4.5707** |
| **육면체** | **0.639** | **0.037** | **3.4618** |
| **CAD**  (파일 importing) | **센서 브라켓** | **0.199** | **0.002** | **3.4572** |
| **실린더** | **3.19** | **0.393** | **3.0584** |
| **IMAGE**  (이미지에서 선택) | **M7 볼트** | **0.437** | **0.128** | **2.7255** |
| **M20 볼트** | **1.948** | **0.356** | **4.0892** |
| **M4 볼트** | **0.387** | **0.034** | **5.5564** |

TradeOff (Accuracy vs. Cycle time)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **모델생성 방법** | **테스트 객체** | **축 편차**  **[mm]** | **회전 편차**  **[deg]** | **처리 시간**  **[sec]** |
| **CAD**  (파일 importing) | **센서 브라켓** | **1.008** | **2.758** | **2.8441** |
| **0.793** | **2.760** | **4.7985** |
| **IMAGE**  (이미지에서 선택) | **M7 볼트** | **1.125** | **0.084** | **2.5291** |
| **0.788** | **0.101** | **4.1142** |

**매칭 유형별 성능표**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **매칭 종류** | |  | **CycleTime(sec)** | **Trans X(mm)** | **Trans Y(mm)** | **Trans Z(mm)** | **Rotation X(rad)** | **Rotation Y(rad)** | **Rotation Z(rad)** | **횟수** |
| **SIMPLE** | **구체** | **최대** | **1.8317** | **-28.278** | **-49.585** | **1212.410** | **2.460** | **2.740** | **2.653** | **100** |
| **최소** | **1.6534** | **-28.685** | **-49.951** | **1211.900** | **-2.422** | **-2.574** | **-2.620** |
| **원통** | **최대** | **4.5707** | **-36.203** | **28.415** | **758.281** | **1.17** | **2.555** | **1.713** | **100** |
| **최소** | **2.7929** | **-37.936** | **25.732** | **754.883** | **-0.83** | **-2.556** | **-1.716** |
| **박스** | **최대** | **3.4618** | **39.459** | **27.567** | **766.734** | **-1.947** | **0.379** | **-0.444** | **500** |
| **최소** | **3.1304** | **38.899** | **26.928** | **766.303** | **-1.952** | **0.355** | **-0.481** |
| **IMAGE(Allen M7x25mm)** | | **최대** | **2.5291** | **-59.968** | **36.41** | **769.769** | **0.19** | **0.136** | **0.985** | **500** |
| **최소** | **2.3862** | **-61.093** | **35.563** | **769.539** | **0.143** | **0.052** | **0.968** |
| **IMAGE(Hex M20X60mm)** | | **최대** | **4.0892** | **75.234** | **-85.437** | **812.67** | **0.887** | **0.673** | **0.533** | **500** |
| **최소** | **4.0247** | **74.852** | **-85.755** | **810.722** | **0.781** | **0.317** | **0.196** |
| **IMAGE(Allen M4x20mm)** | | **최대** | **5.5564** | **69.293** | **4.378** | **779.969** | **0.073** | **0.107** | **-0.008** | **500** |
| **최소** | **5.3775** | **69.143** | **4.225** | **779.582** | **0.063** | **0.073** | **-0.029** |
| **CAD(Sensor BRK)** | | **최대** | **3.4572** | **-101.891** | **22.984** | **758.783** | **-0.46** | **1.913** | **0.741** | **500** |
| **최소** | **3.2336** | **-102.09** | **22.843** | **758.669** | **-0.462** | **1.911** | **0.74** |
| **CAD(Complex Cylinder)** | | **최대** | **3.0584** | **64.143** | **-26.499** | **777.889** | **2.86** | **0.702** | **0.512** | **500** |
| **최소** | **2.8584** | **60.936** | **-27.787** | **774.708** | **2.467** | **0.368** | **-1.14** |

**셋팅: 시간 vs 정확도**

**1. 산포된 PointCloud 구성의 Scene(속도)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **매칭 종류** |  | **CycleTime(sec)** | **Trans X(mm)** | **Trans Y(mm)** | **Trans Z(mm)** | **Rotation X(rad)** | **Rotation Y(rad)** | **Rotation Z(rad)** | **횟수** |
| **IMAGE(Allen M7x25mm)** | **최대** | **2.5291** | **-59.968** | **36.41** | **769.769** | **0.19** | **0.136** | **0.985** | **500** |
| **최소** | **2.3862** | **-61.093** | **35.563** | **769.539** | **0.143** | **0.052** | **0.968** |
| **Sensor\_BRK** | **최대** | **2.8441** | **71.541** | **35.266** | **749.845** | **-0.443** | **0.307** | **2.392** | **500** |
| **최소** | **2.6418** | **70.533** | **34.645** | **749.628** | **-1.96** | **-1.555** | **-0.368** |

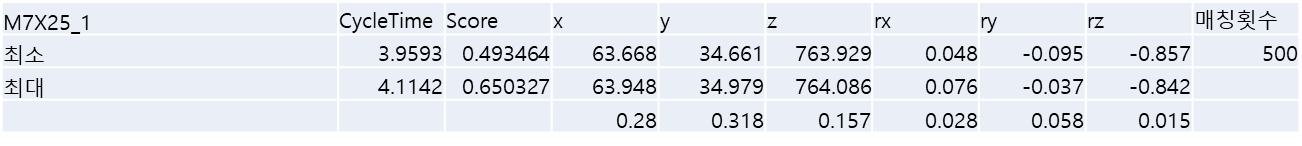
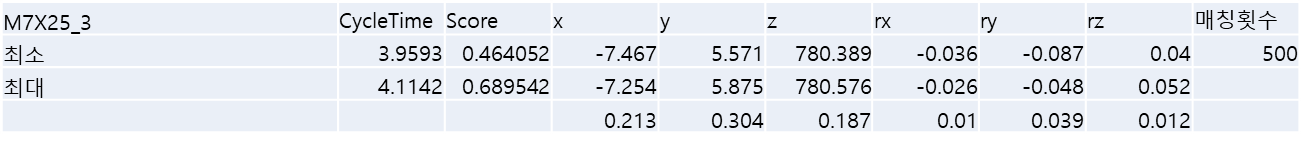
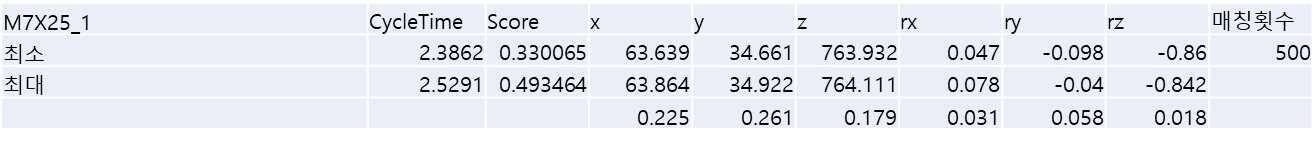
**2. 조밀한 PointCloud 구성의 Scene(정확도)**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **매칭 종류** |  | **CycleTime(sec)** | **Trans X(mm)** | **Trans Y(mm)** | **Trans Z(mm)** | **Rotation X(rad)** | **Rotation Y(rad)** | **Rotation Z(rad)** | **횟수** |
| **IMAGE(Allen M7x25mm)** | **최대** | **4.1142** | **-60.28** | **36.31** | **769.785** | **0.201** | **0.127** | **0.992** | **500** |
| **최소** | **3.9593** | **-61.068** | **35.701** | **769.529** | **0.148** | **0.026** | **0.97** |
| **Sensor\_BRK** | **최대** | **4.7985** | **71.447** | **35.217** | **749.853** | **-0.443** | **0.305** | **2.392** | **500** |
| **최소** | **4.6464** | **70.654** | **34.68** | **749.569** | **-1.959** | **-1.555** | **-0.366** |

**셋팅: 정확도 VS 시간**

F

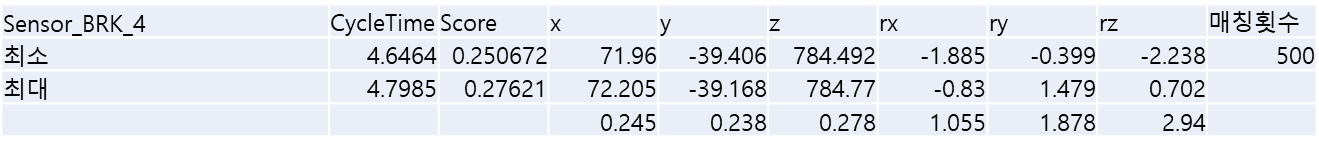
F -> Scene Point less

****

**셋팅: 정확도 vs 시간**

F

F -> Scene Point less

****

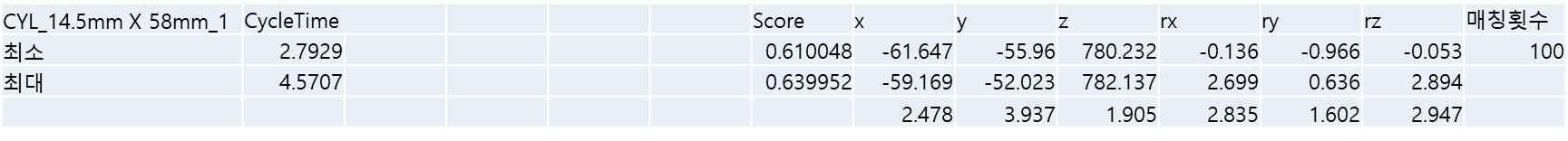
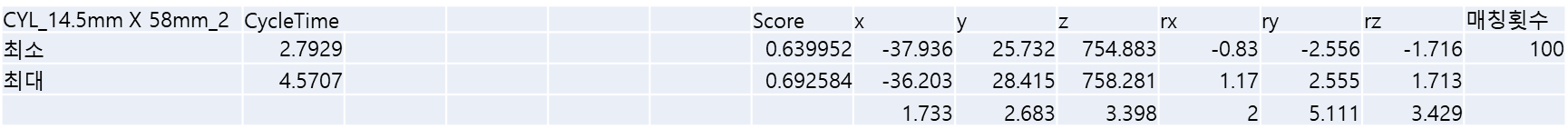
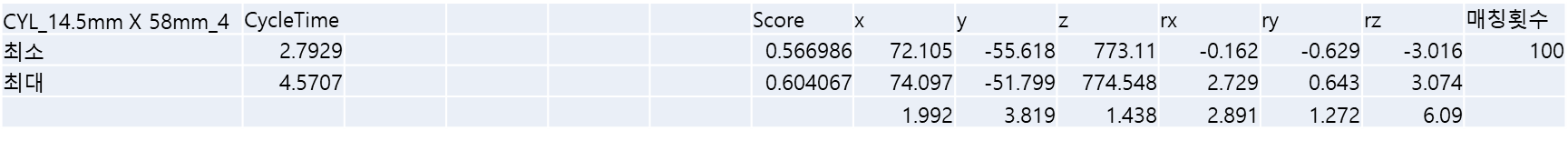
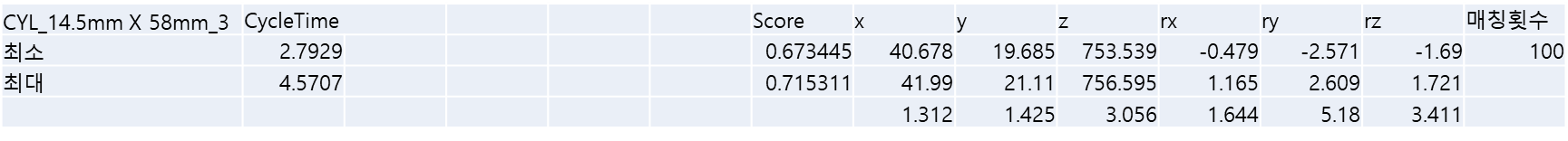
****

****

****

****

**Cylinder Test**

****

# 부 록

## 1. Halcon회전영향이 있는(pose\_compose)에 관하여

회전은 pose\_compose(poseorigin, posechange) -> pose\_changed를 이용한다.

posechange에 회전을 시키기 위한 ex) pose[0,0,0,90,0,0,0]를 넣어 halcon내부적으로 pose를 homogeneous transformation matrix로 변환하여 행렬곱을 한다.

pose\_compose의 절차는 하기와 같다.

hom\_mat3d\_identity (HomMat3DIdent) – *[0,0,0,0,0,0,0]을 의미하는 homogeneous transformation matrix의 단위행렬 =*

hom\_mat3d\_rotate (HomMat3DIdent, RotZ, 'z', 0, 0, 0, HomMat3DRotZ)

hom\_mat3d\_rotate (HomMat3DRotZ, RotY, 'y', 0, 0, 0, HomMat3DRotYZ)

hom\_mat3d\_rotate (HomMat3DRotYZ, RotX, 'x', 0, 0, 0, HomMat3DXYZ)

1) Rp+T gba 회전 영향

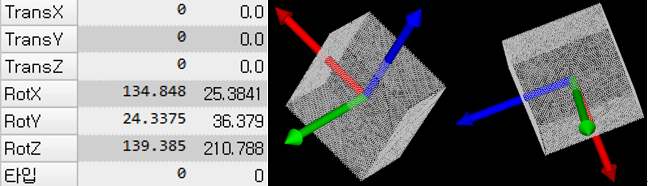


Figure 1. X is rotated 90’, other rotation value changed RED:X, GREEN:Y BLUE:Z

Figure 11.의 좌표 테이블은 pose\_compose([0,0,0,134.848, 24.3375, 139.385, 0], [0,0,0,90,0,0,0]) (*pose\_compose(nowpose, targetpose) -> Transform Pose*)을 이용해 X축을 90’ 돌려 [0,0,0,25.3841,36.379,210.788,0] 변환된 결과를 의미한다. 우측의 사진은 왼쪽 변환 전, 우측 변환 후의 결과를 나타낸 그림이다.

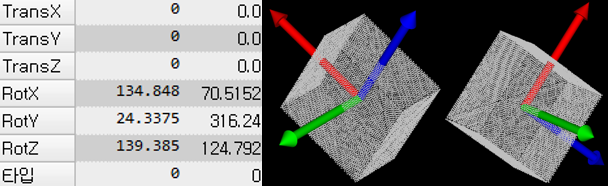


Figure 2. Y is rotated 90’, other rotation value changed RED:X, GREEN:Y BLUE:Z

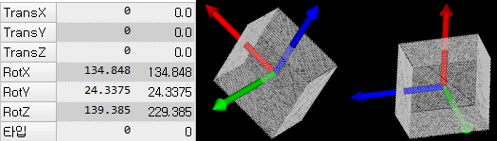


Figure 3. Z is rotated 90’, other rotation value NOT changed RED:X, GREEN:Y BLUE:Z

2) Rp+T abg 회전 영향

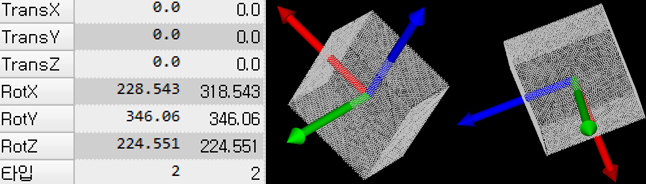


Figure 4. X is rotated 90’, other rotation value NOT changed RED:X, GREEN:Y BLUE:Z



Figure 5. Y is rotated 90’, other rotation value changed RED:X, GREEN:Y BLUE:Z

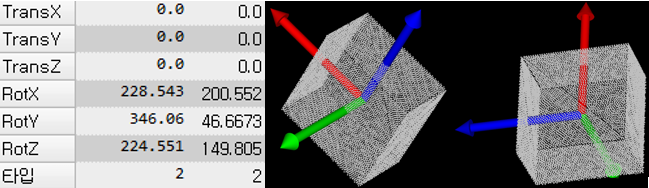


Figure 6. Z is rotated 90’, other rotation value changed RED:X, GREEN:Y BLUE:Z

3) ‘point’, ‘coordinate’ 회전 방향 비교

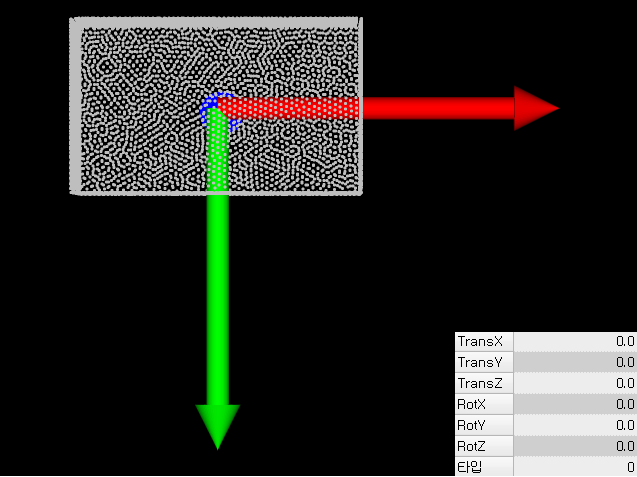


Figure 7. Origin Pose with point or coordinate property

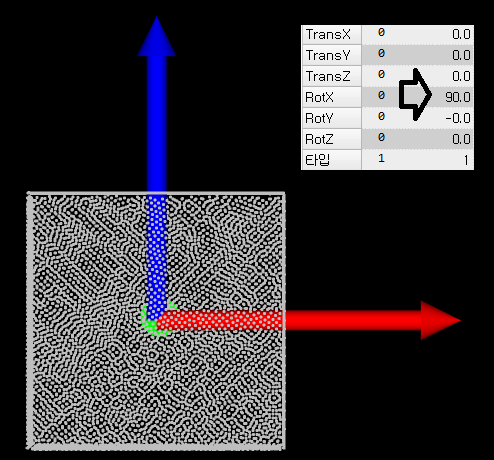
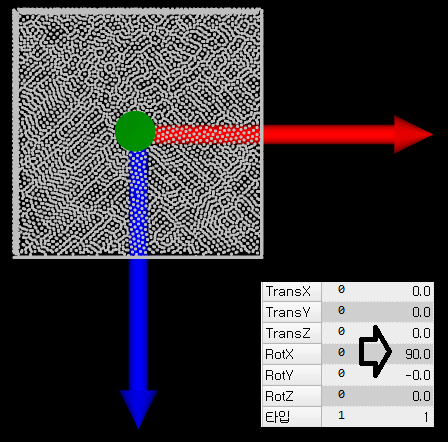
 

Figure 8. Rotate X Axis 90’ with point property Figure 9. Rotate X Axis 90’ with coordinate property

## 2. pose\_compose 내부 변환 방식에 관하여

*pose\_compose(nowpose, targetpose) -> Transform Pose* = =

***1) pose to homogeneous transformation matrix***

pose\_compose([0,0,0,134.848, 24.3375, 139.385, 0], [0,0,0,90,0,0,0])

origin pose(이하 **op)** = [0,0,0,134.848, 24.3375, 139.385, 0] 🡺

rotate x 90’ pose(이하 **rx90p)** = [0,0,0,90, 0, 0, 0] 🡺

***2)*** ***two homogeneous transformation matrix multiply***

=

=

=

***3. homogeneous transformation matrix to pose***