AI Robotics

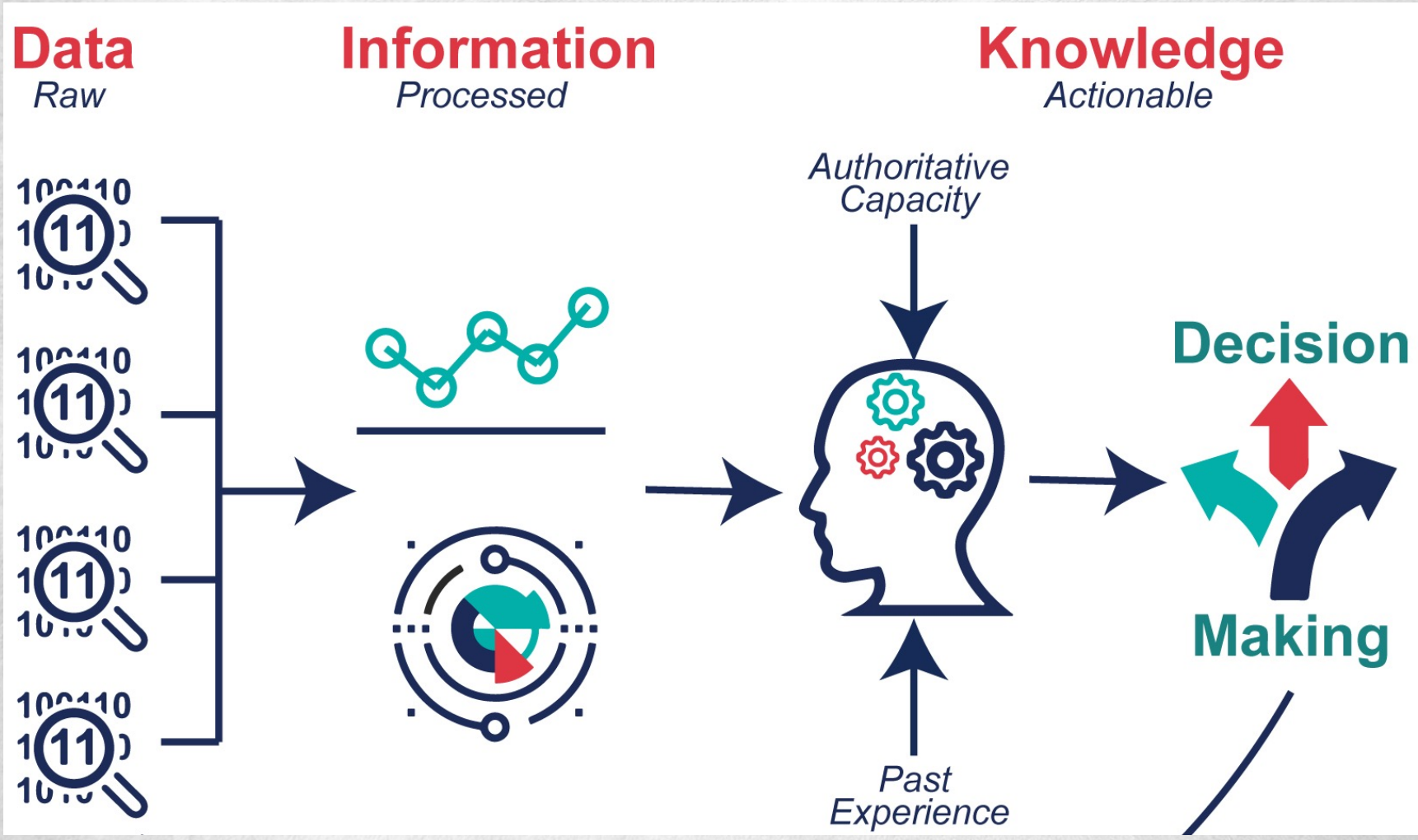
Perception

Overview

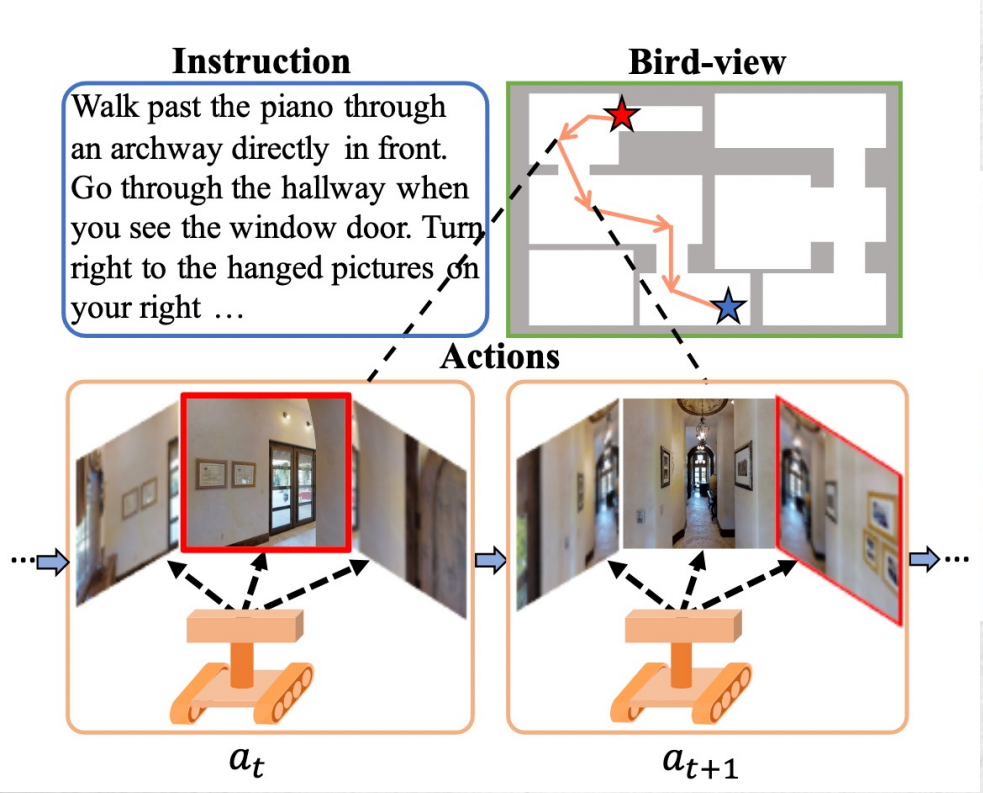
Perception

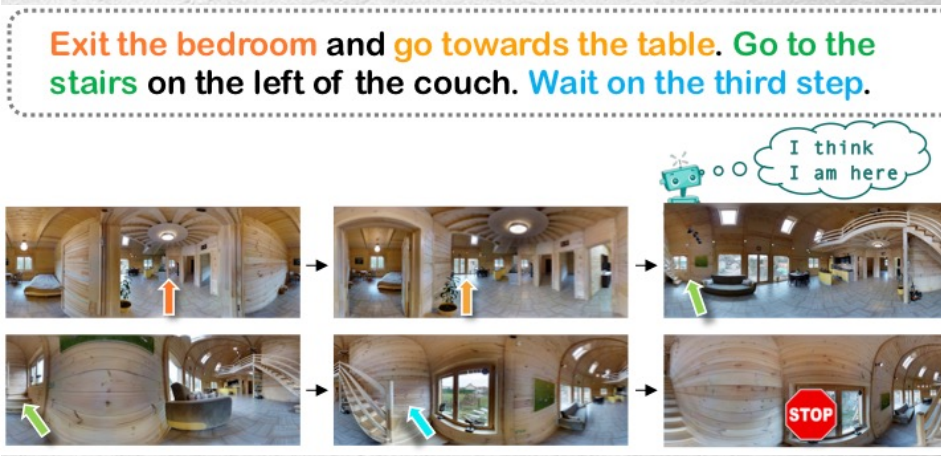
For robots to act and react more richly with the world around them, they require a deeper perception and understanding of the environment, often referred to as semantic understanding, of the world in which they operate

Perception


Data
Raw
Information
Processed
Knowledge
Actionable
Authoritative Capacity
Decision Making
Past Experience

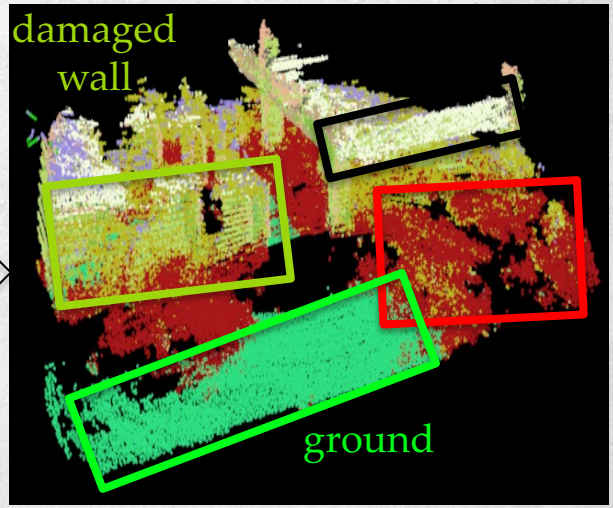
Perception in Indoor Navigation





Perception in Disaster Relief

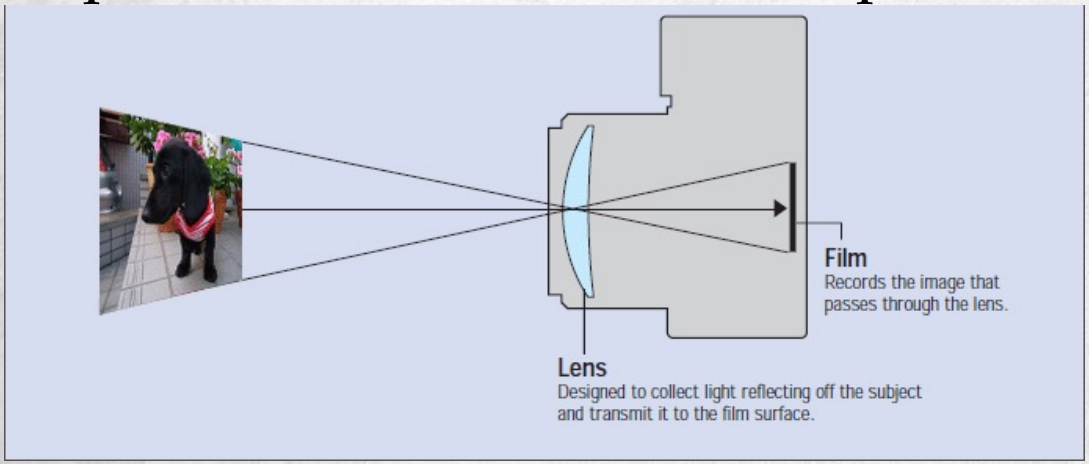
Improve the ability of mobile robots to carry out post-disaster reconnaissance tasks through automated segmentation and damage detection from acquired point cloud data



2D Perception

Camera

- Uses Charge-Coupled Device (CCD) sensors to measure light intensity from the environment
- Lenses focus the light entering the camera and the size of the aperture can be widened or narrowed to let more or less light into the camera
- Shutter mechanism determines the amount of time the photosensitive surface is exposed to the light.



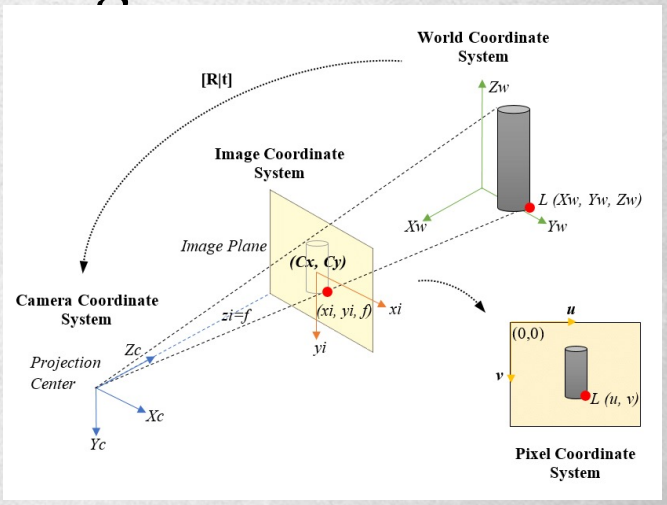
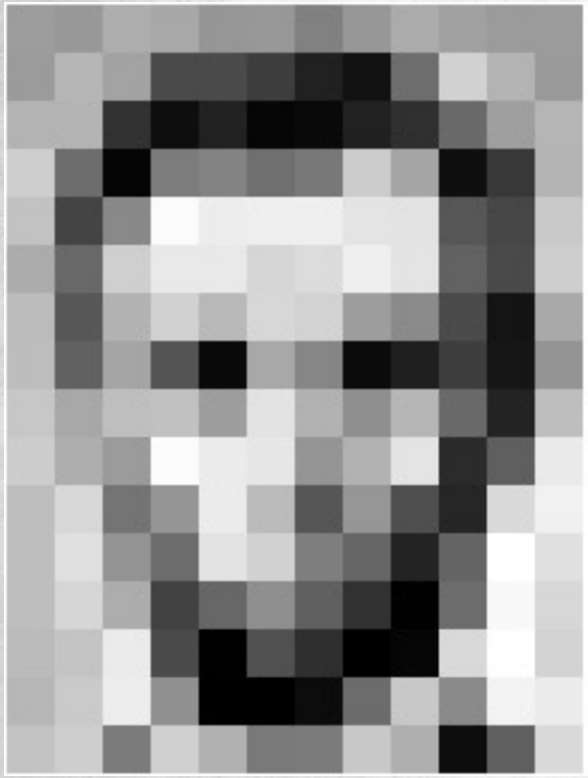


Image Data Representation



157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

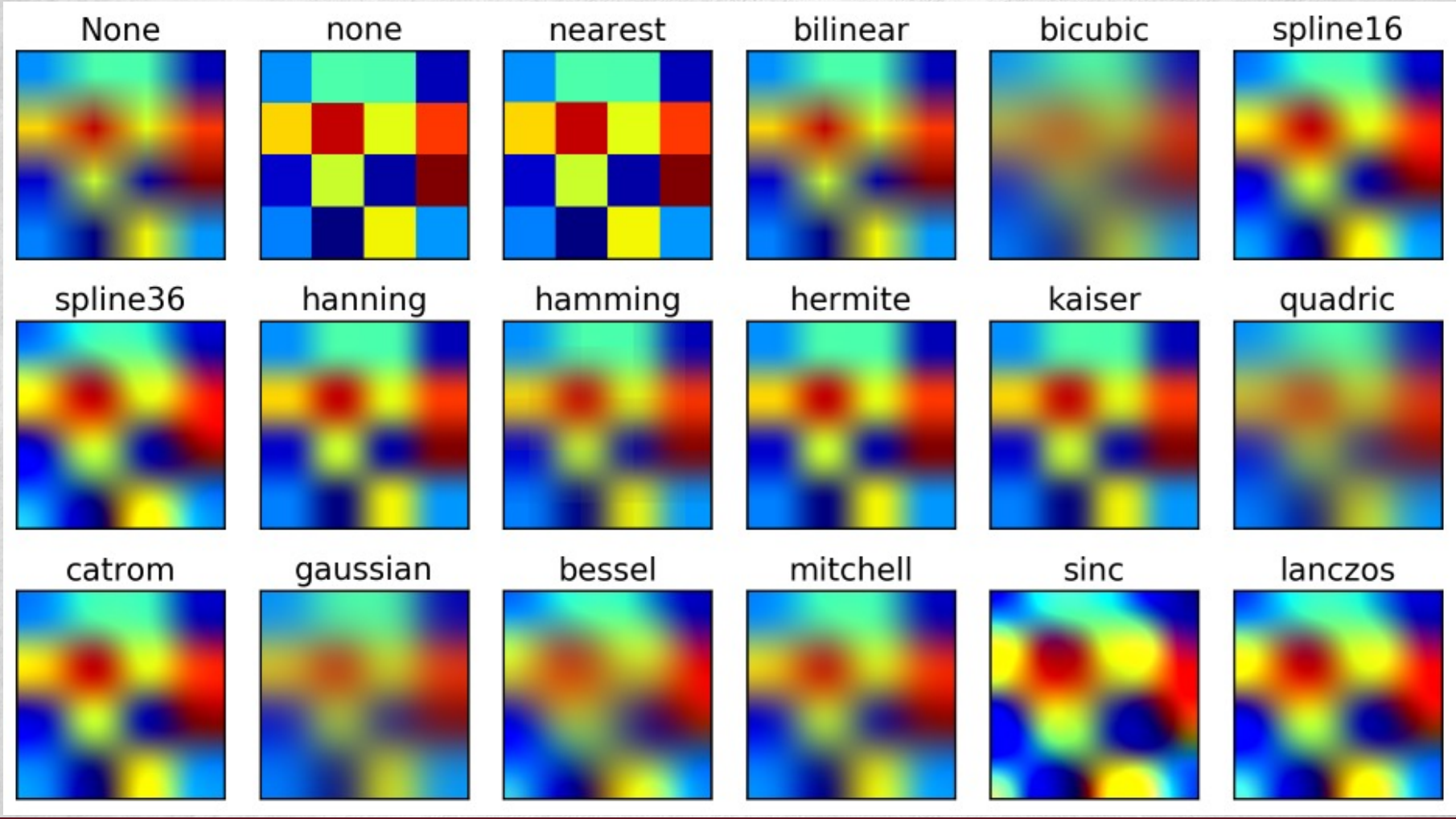
Image Scaling

Scaled down
to 30%

Original
raster image

Scaled up
to 300%

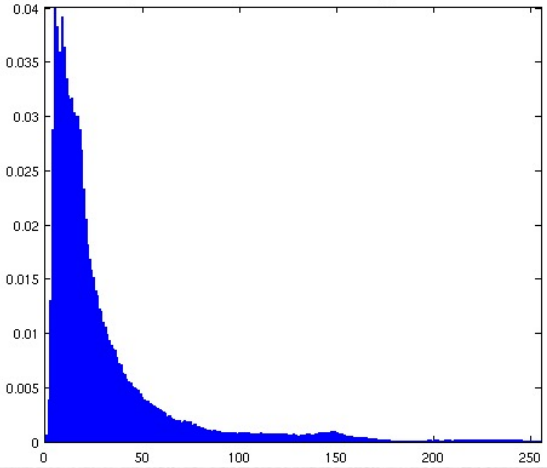
Image Filtering



Brightness Normalization

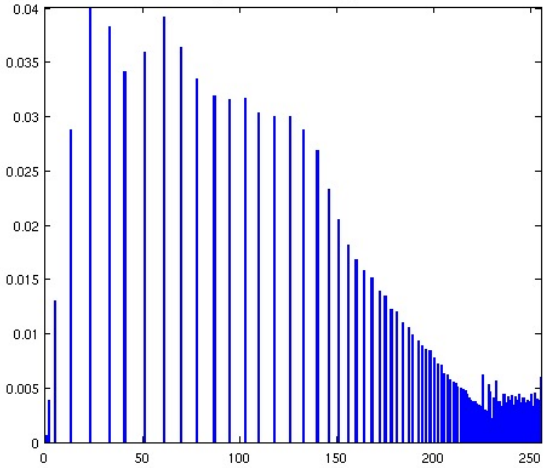
Input Image

Original Histogram before equalization



Output Image

Equalized histogram



Binary Thresholding

Connected Components

Orginal Image

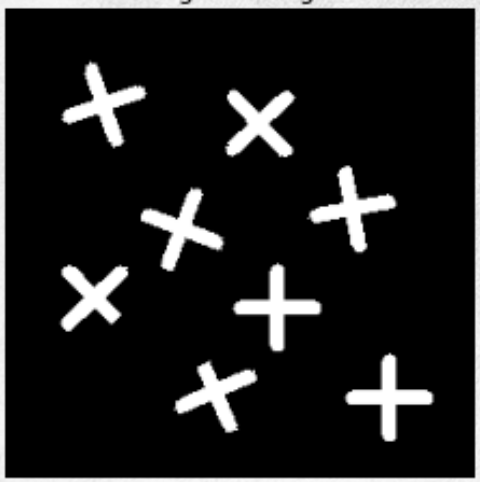
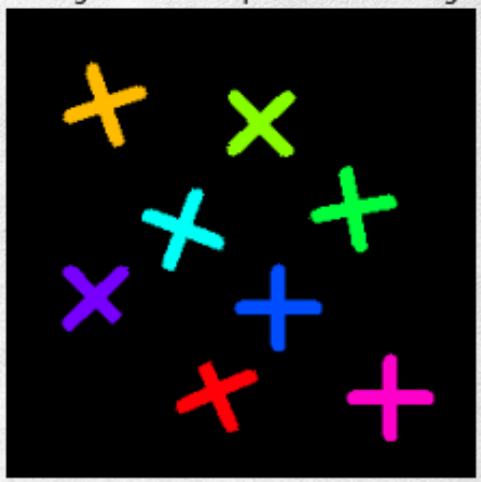


Image after Component Labelling

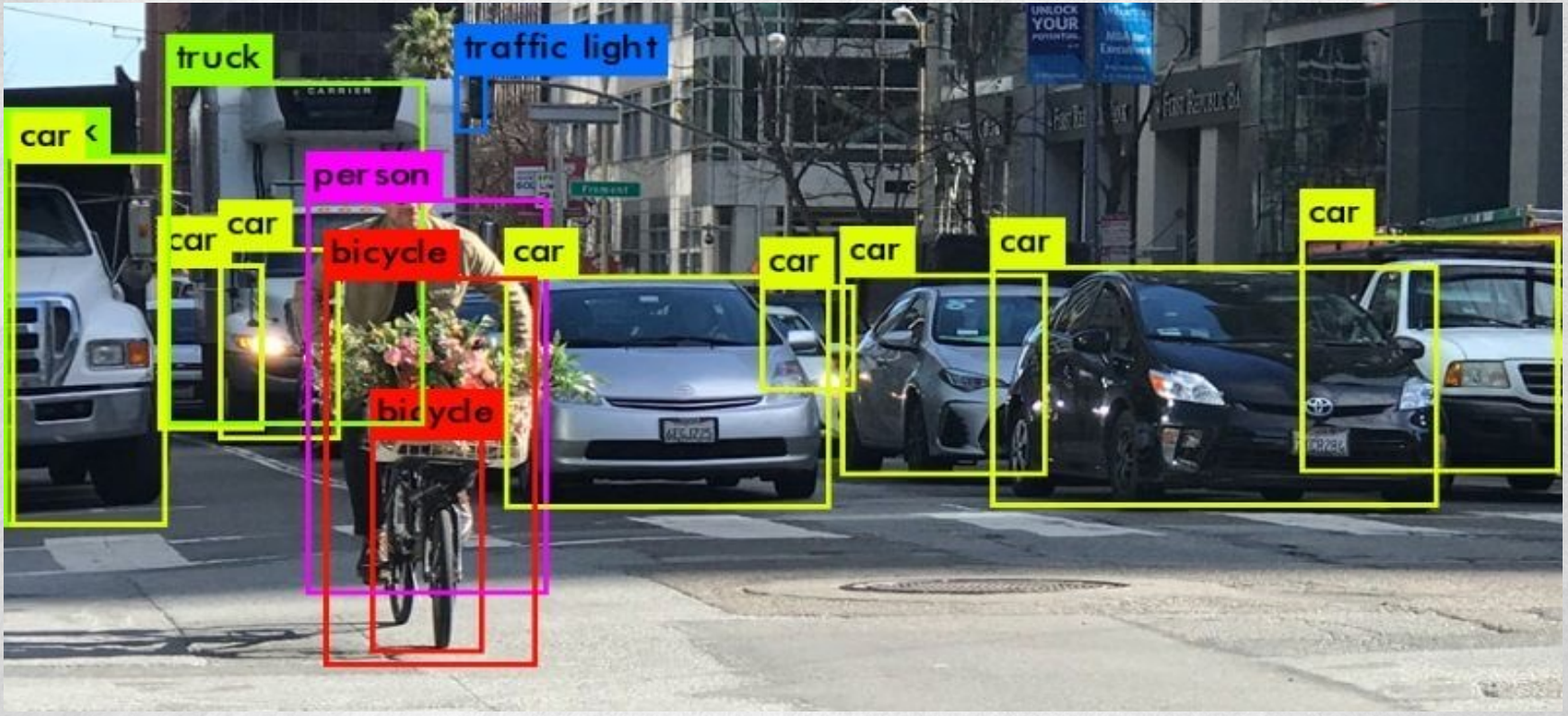


1	1	0	0	1
0	1	0	1	0
0	1	0	0	0
0	0	1	1	1
0	0	0	0	1

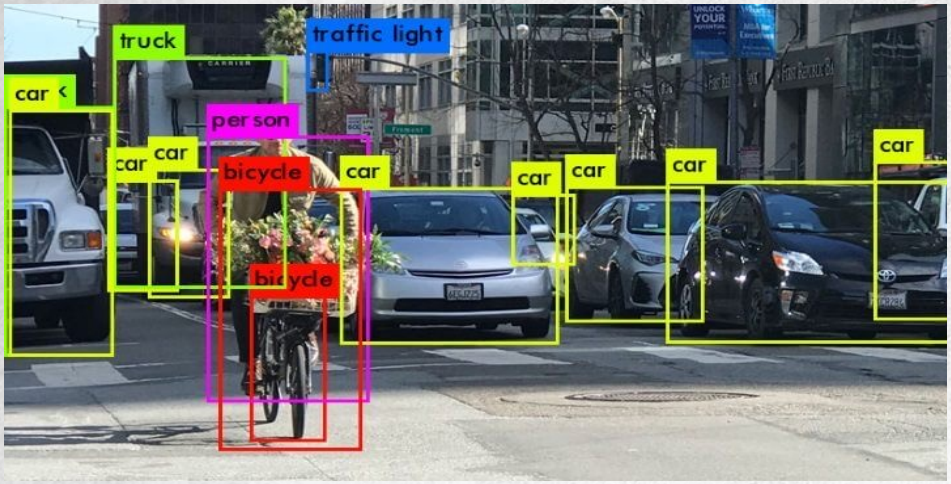
1	1	0	0	2
0	1	0	3	0
0	1	0	0	0
0	0	4	4	4
0	0	0	0	4

Object Recognition

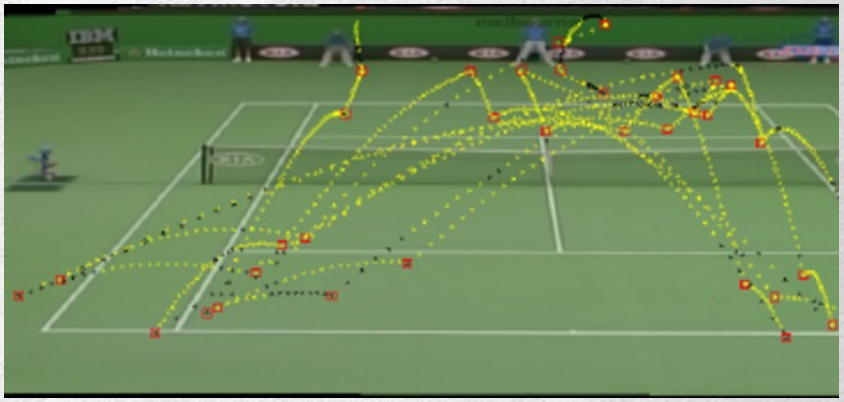
Computer vision technique that deals with detecting semantic objects of a certain class (e.g. humans, buildings, or cars) present in digital images and videos



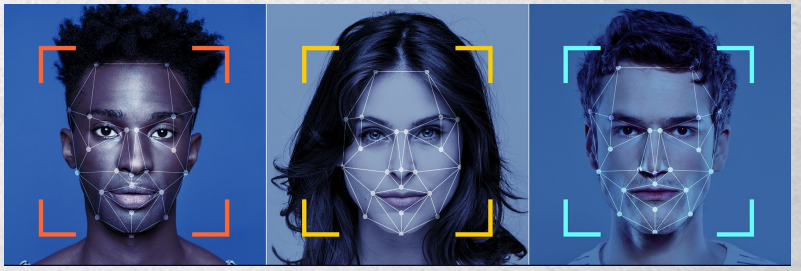
Autonomous Driving



Tracking

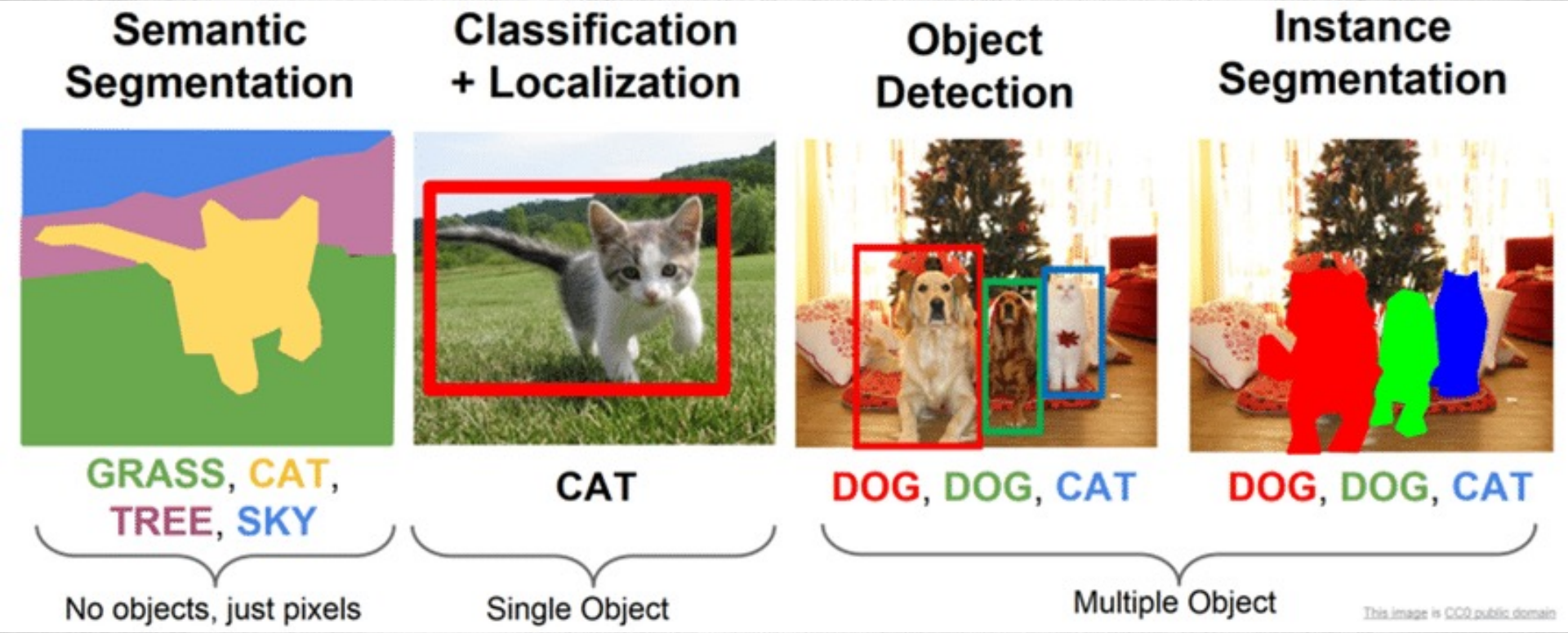


Face Recognition

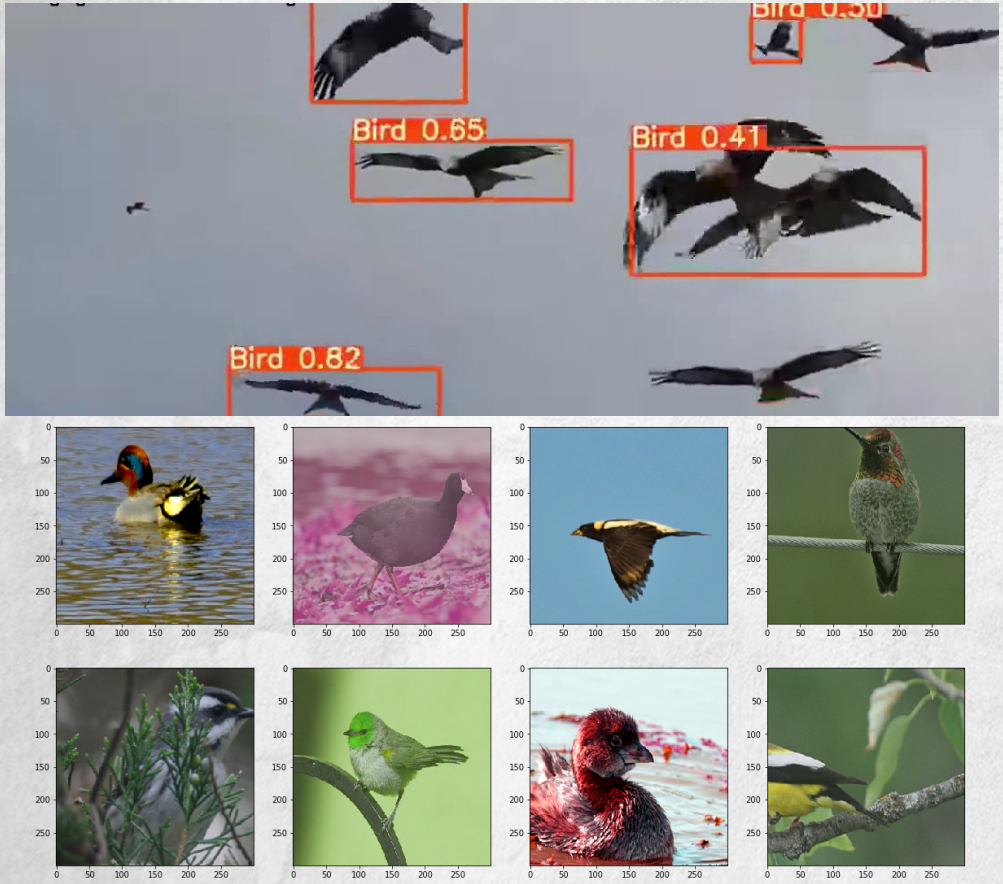


Activity Recognition

Different flavors of object recognition

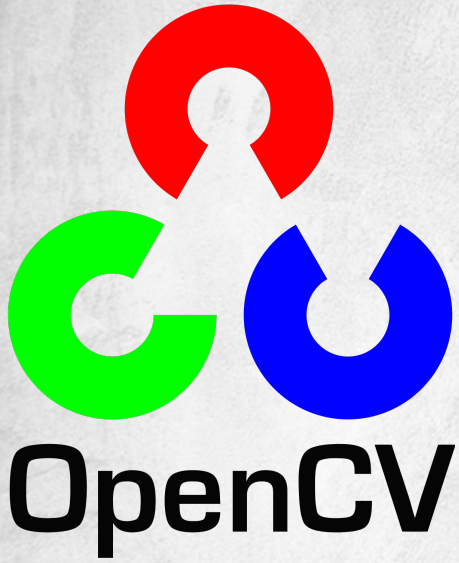


Real-world object recognition is difficult



Perception in Autonomous Driving

Image Processing Software



OpenCV

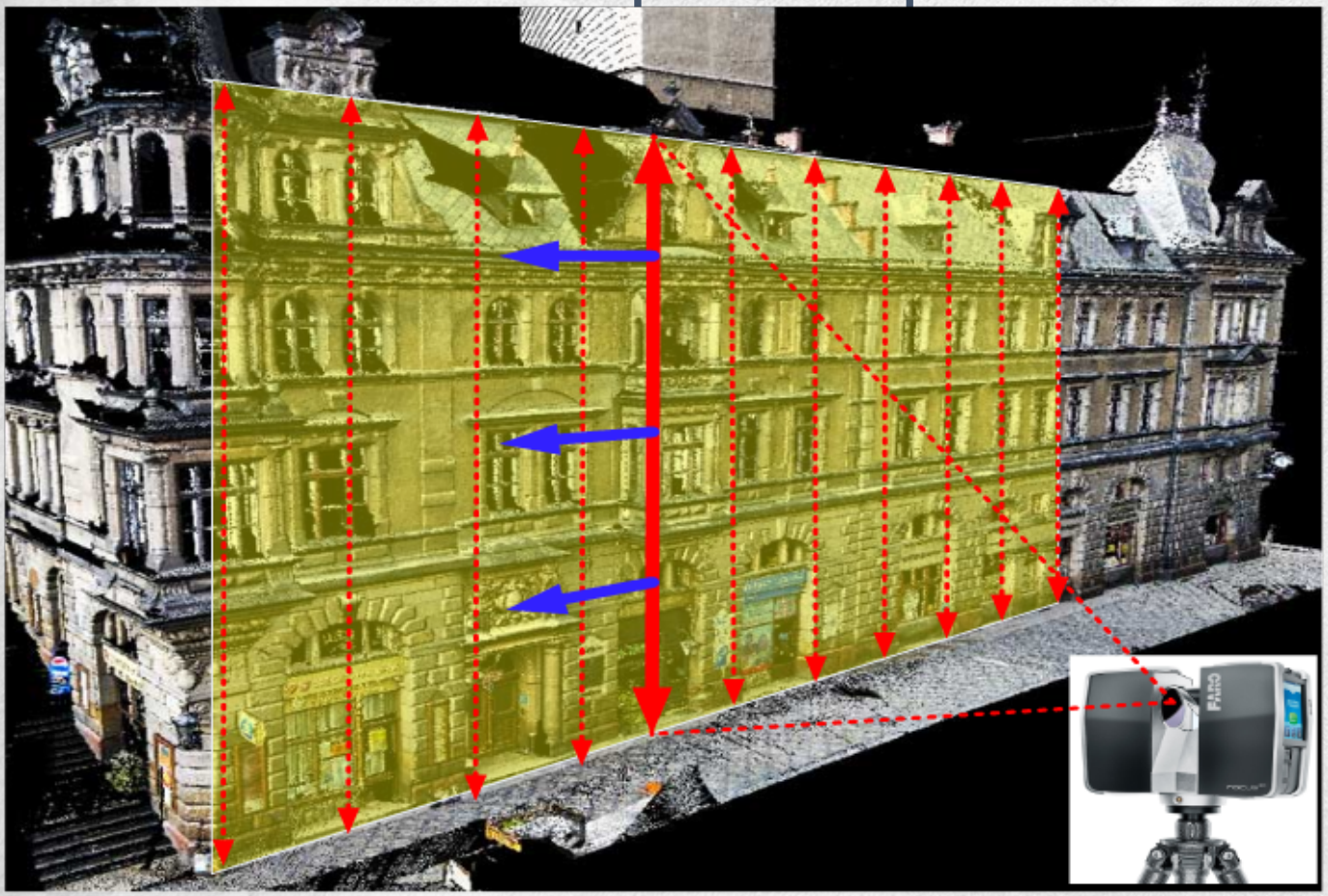
https://docs.opencv.org/4.5.0/d6/d00/tutorial_py_root.html

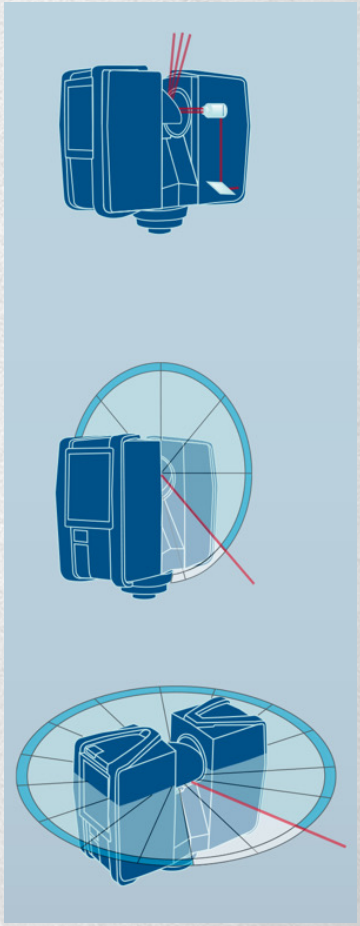
scikit-image

https://scikit-image.org/docs/dev/auto_examples/

3D Perception

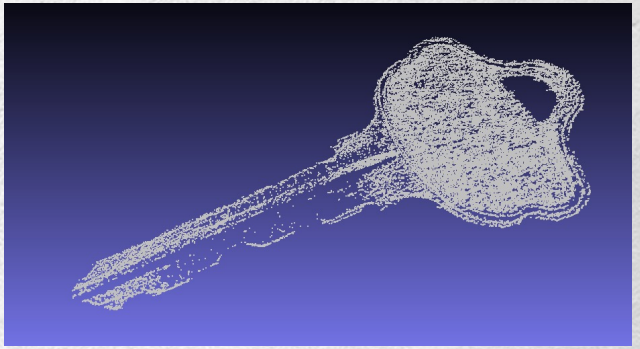
LiDAR Scanning – working principles

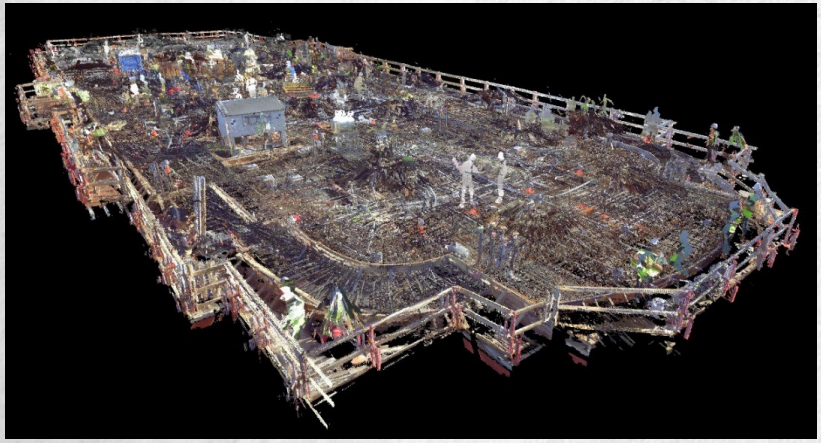




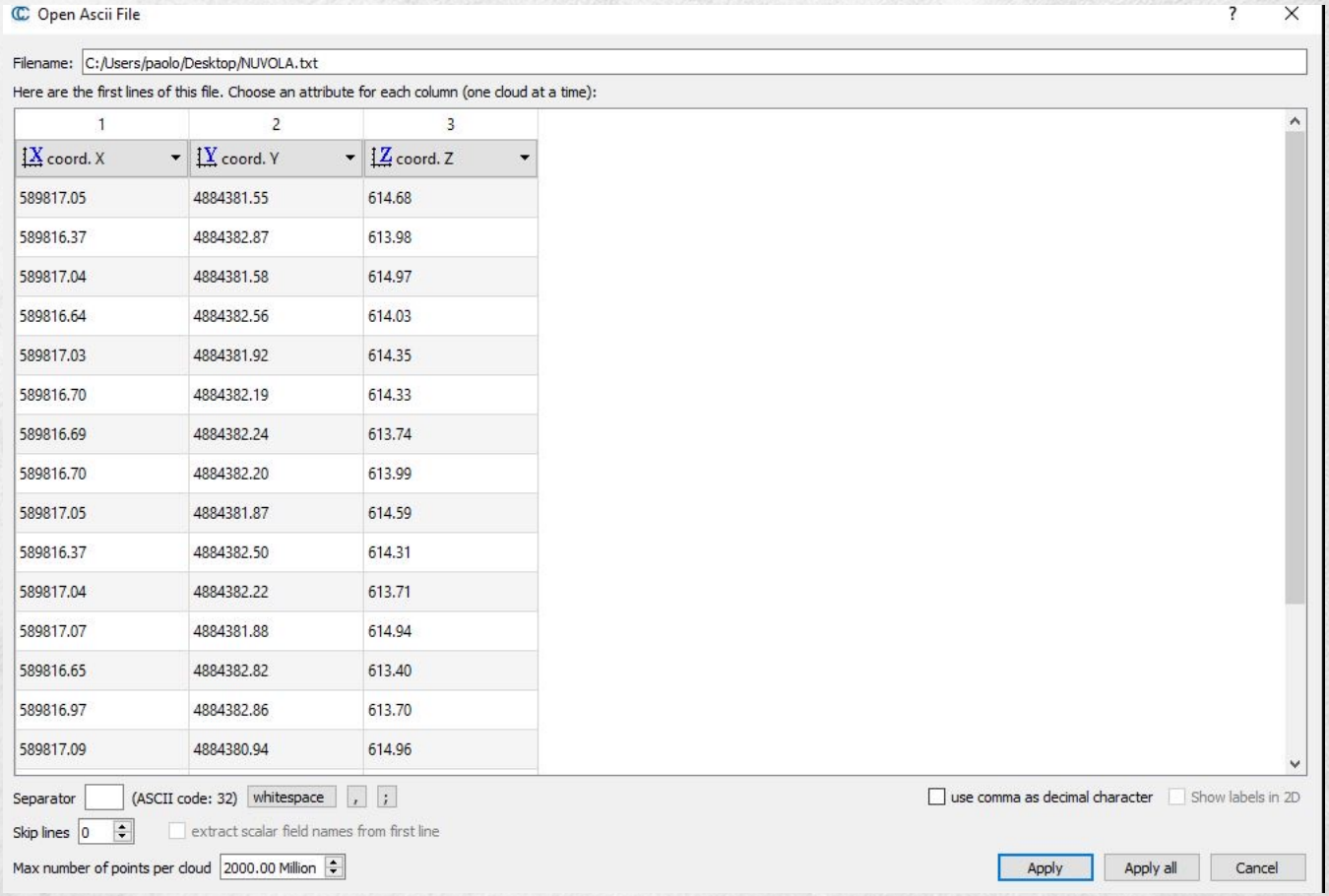
LiDAR scanning– point cloud

- Point cloud
  - product of a laser scanner
  - a set of points (millions)
  - point info: X, Y, Z, (I, R, G, B)
  - formats: fls, pts, xyz, las, txt…

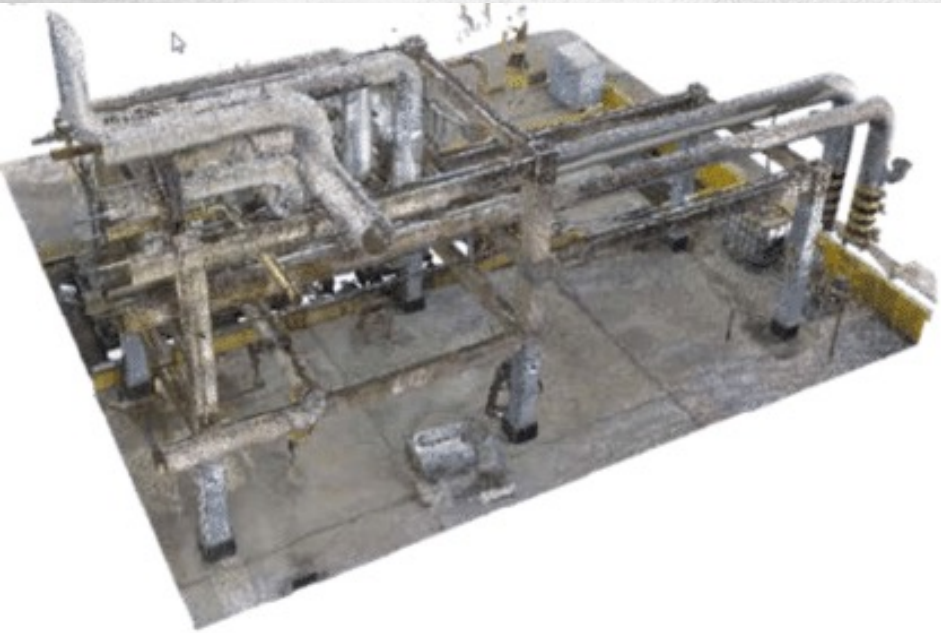




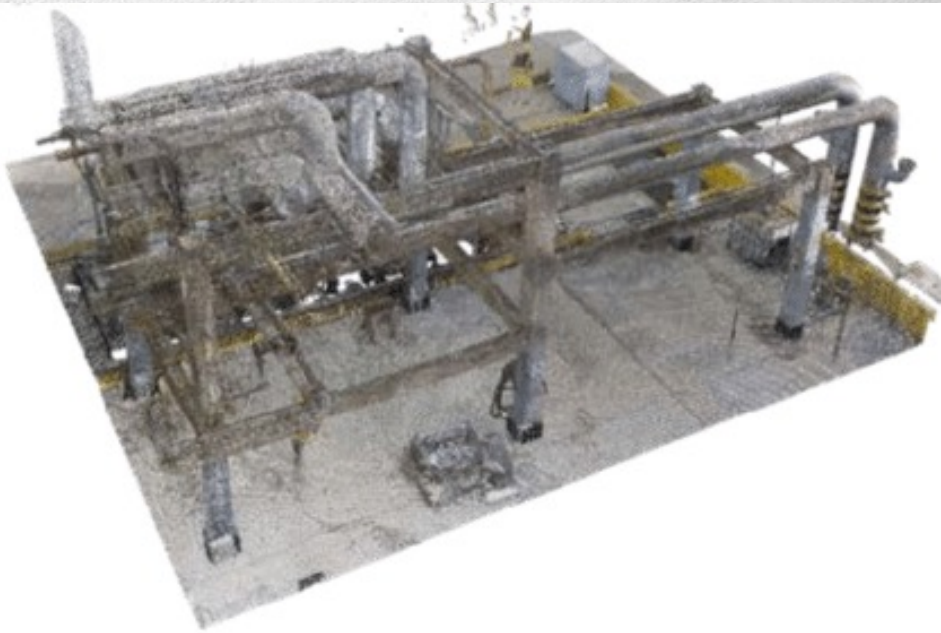
Point Cloud Data Representation



Point Cloud Downsampling

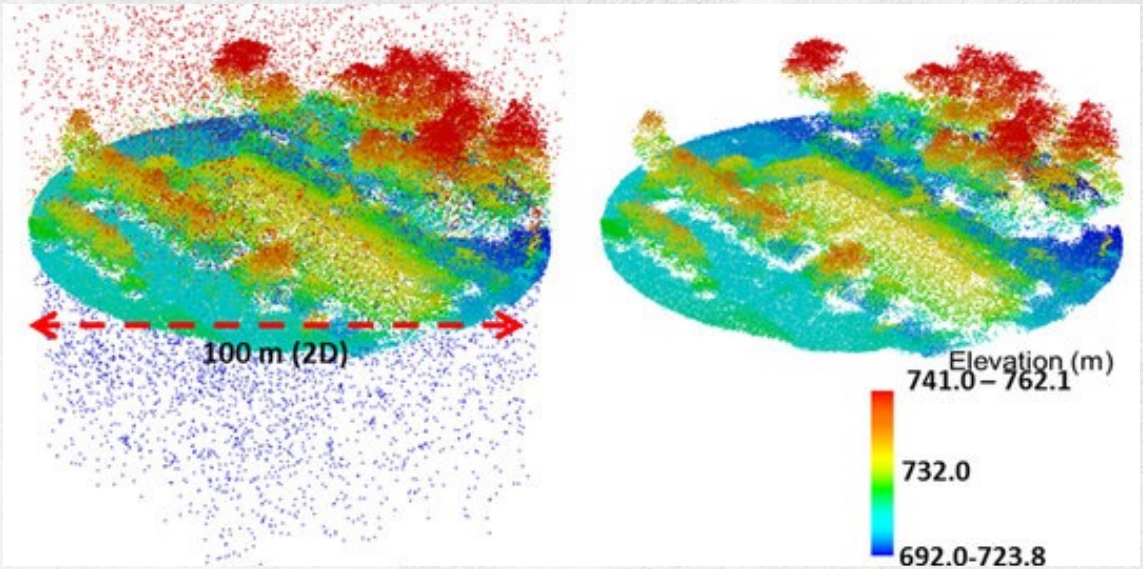


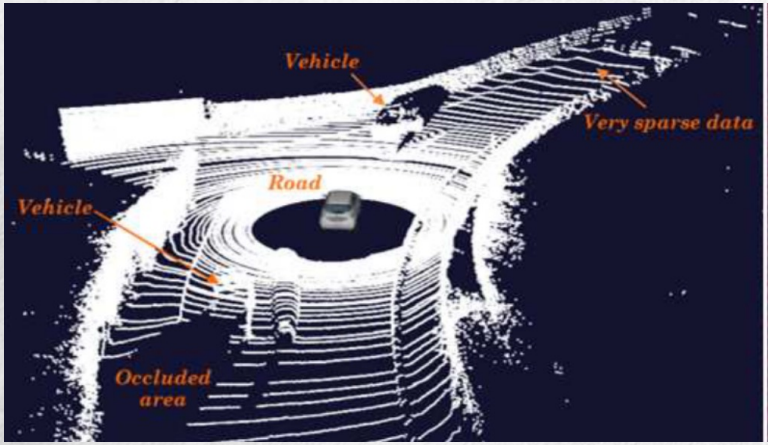
(a) Original [3,467,072 points]

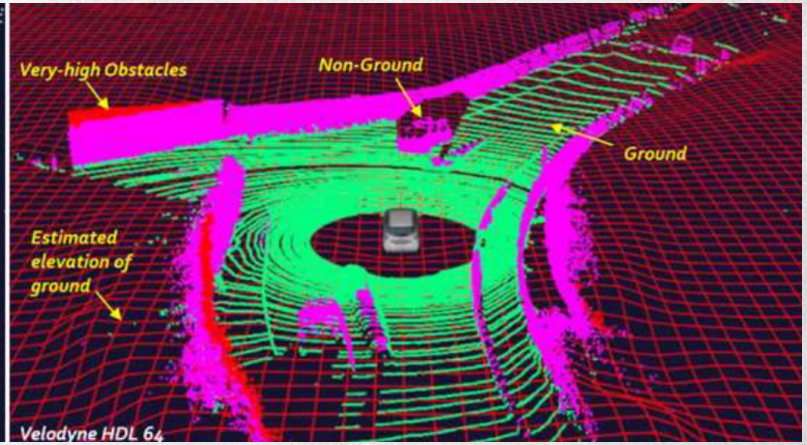


(b) Downsampled [322,797 points]

Spatial Filtering



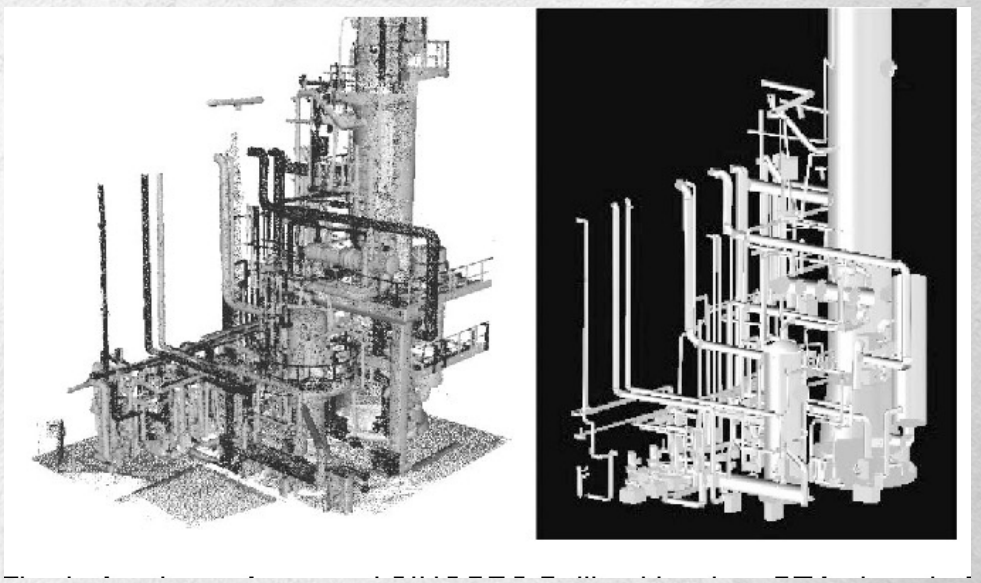




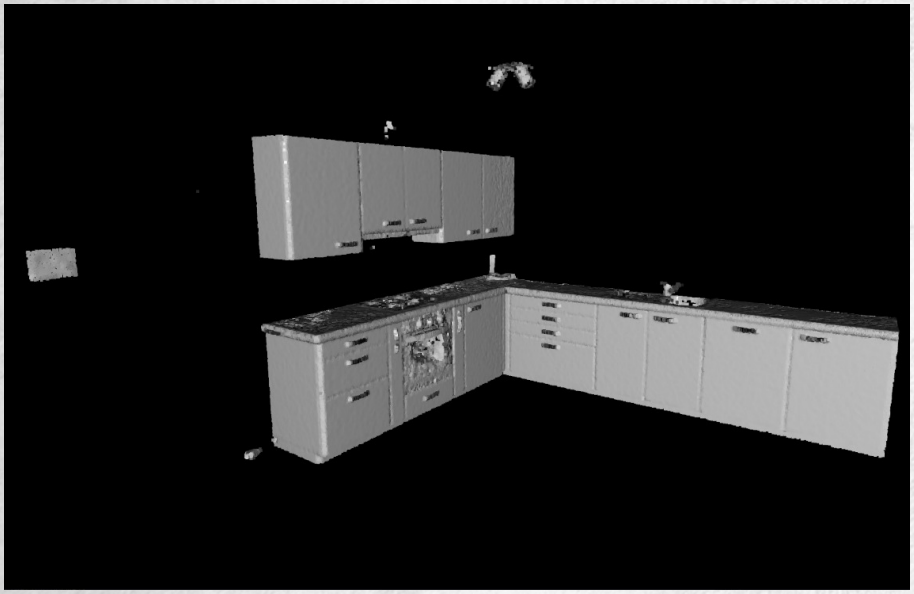
Model-based Segmentation

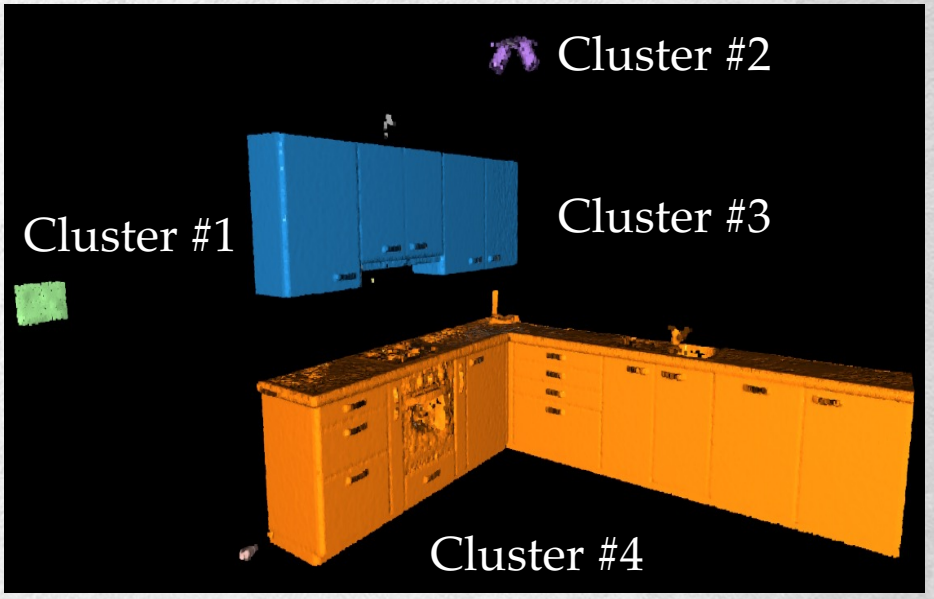
Plane segmentation

Cylinder segmentation



Euclidean Clustering / Connected Components

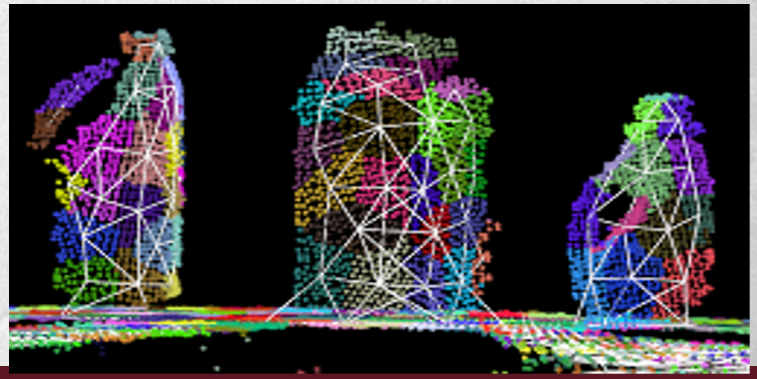




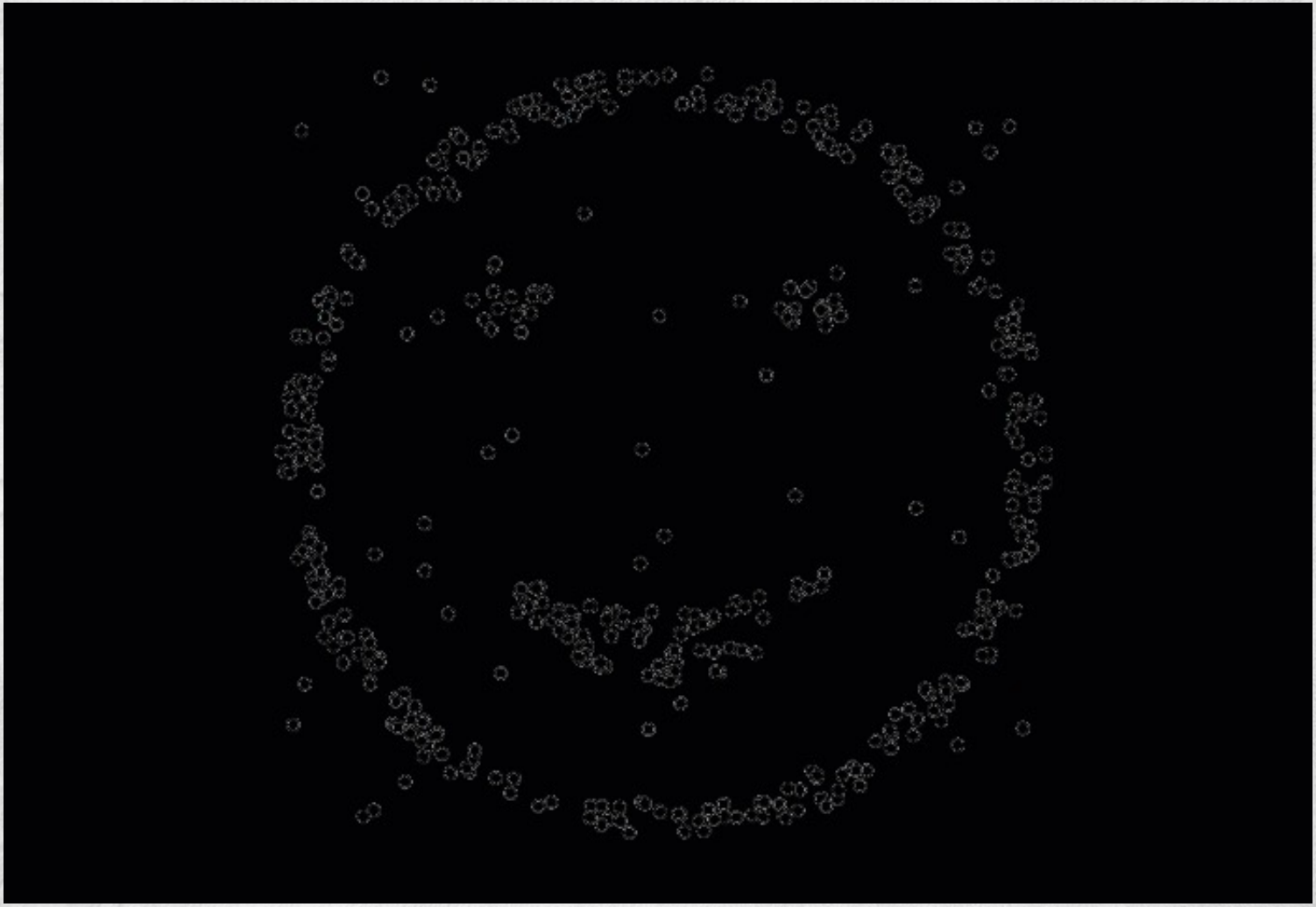
Euclidean Clustering

Point cloud clustering (Euclidean)

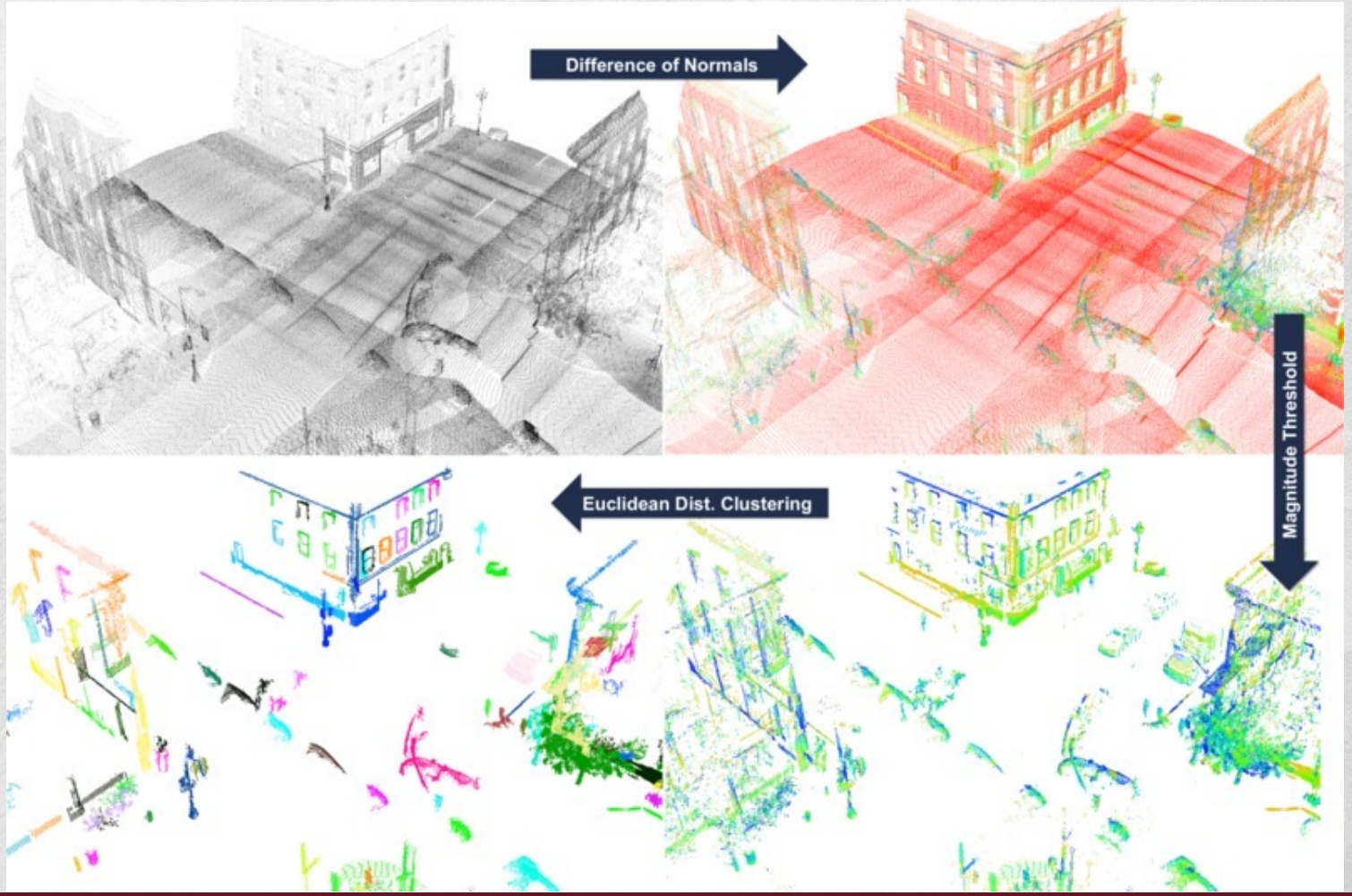
- Organize the point cloud into separate clusters corresponding to different objects
- Iterative process with the following steps:
  - Pick a seed point and initialize a cluster
  - Add all the neighboring points within a fixed distance threshold to the same cluster
  - Repeat for each adjacent point until no more neighboring points are found



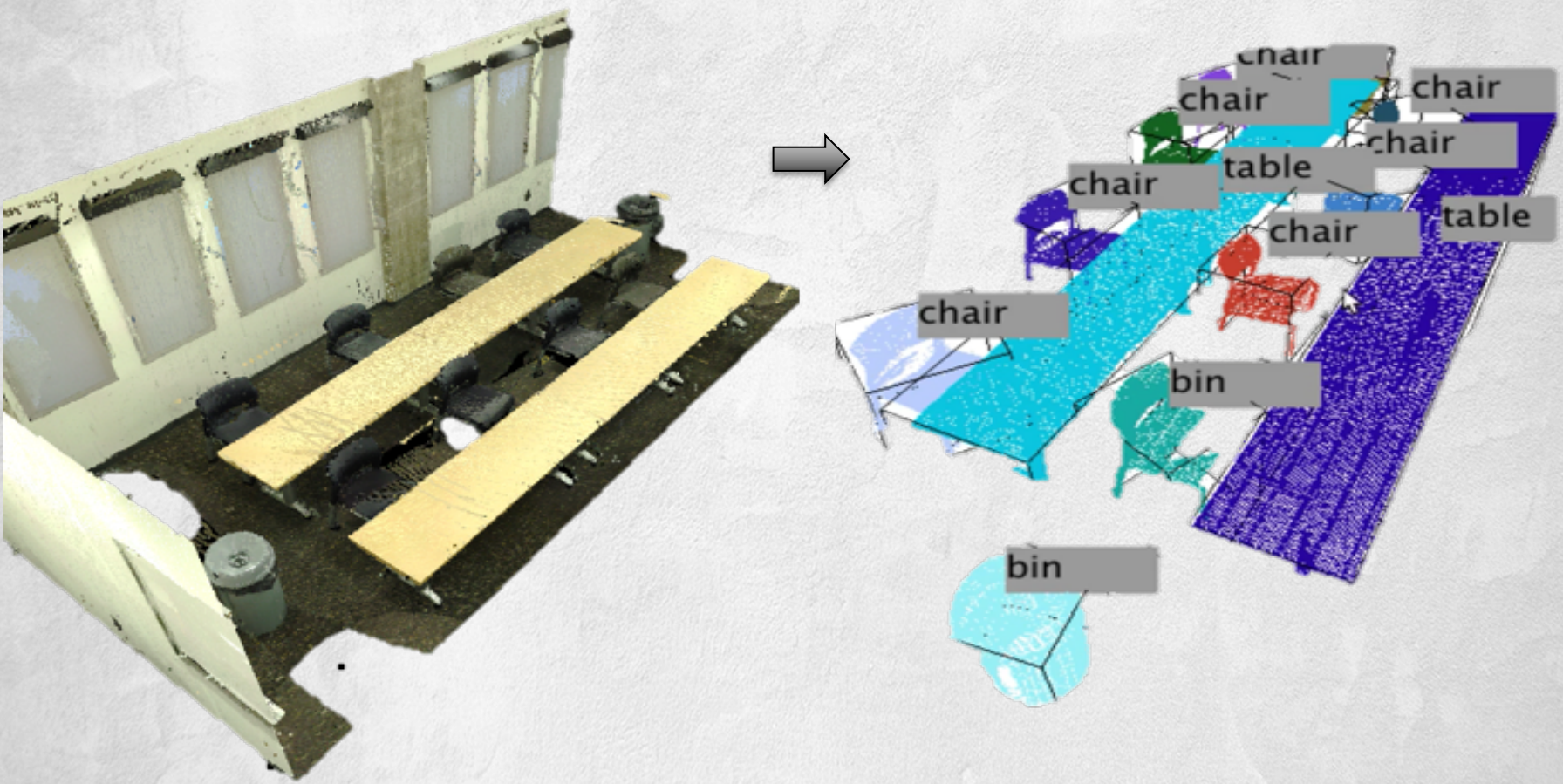
Euclidean Clustering



Normal-based Segmentation



Object Recognition



Different flavors of object recognition

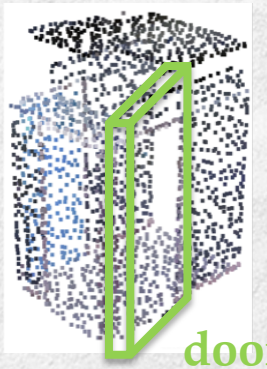
Classification

Input: Object
Output: Object Label

Part Segmentation

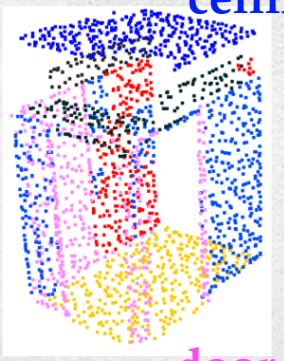
Input: Object
Output: Point Labels

Object Detection



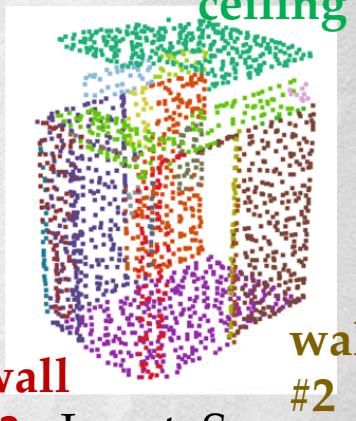
Input: Scene
Output: Bounding Boxes

Semantic Segmentation



Input: Scene
Output: Point Labels

Instance Segmentation

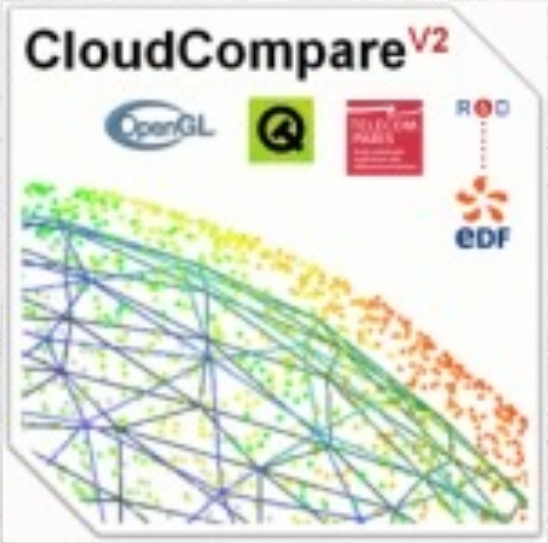


Input: Scene
Output: Instance ID + Object Labels

Point Cloud Processing Software

Point Cloud Library

https://pointclouds.org/

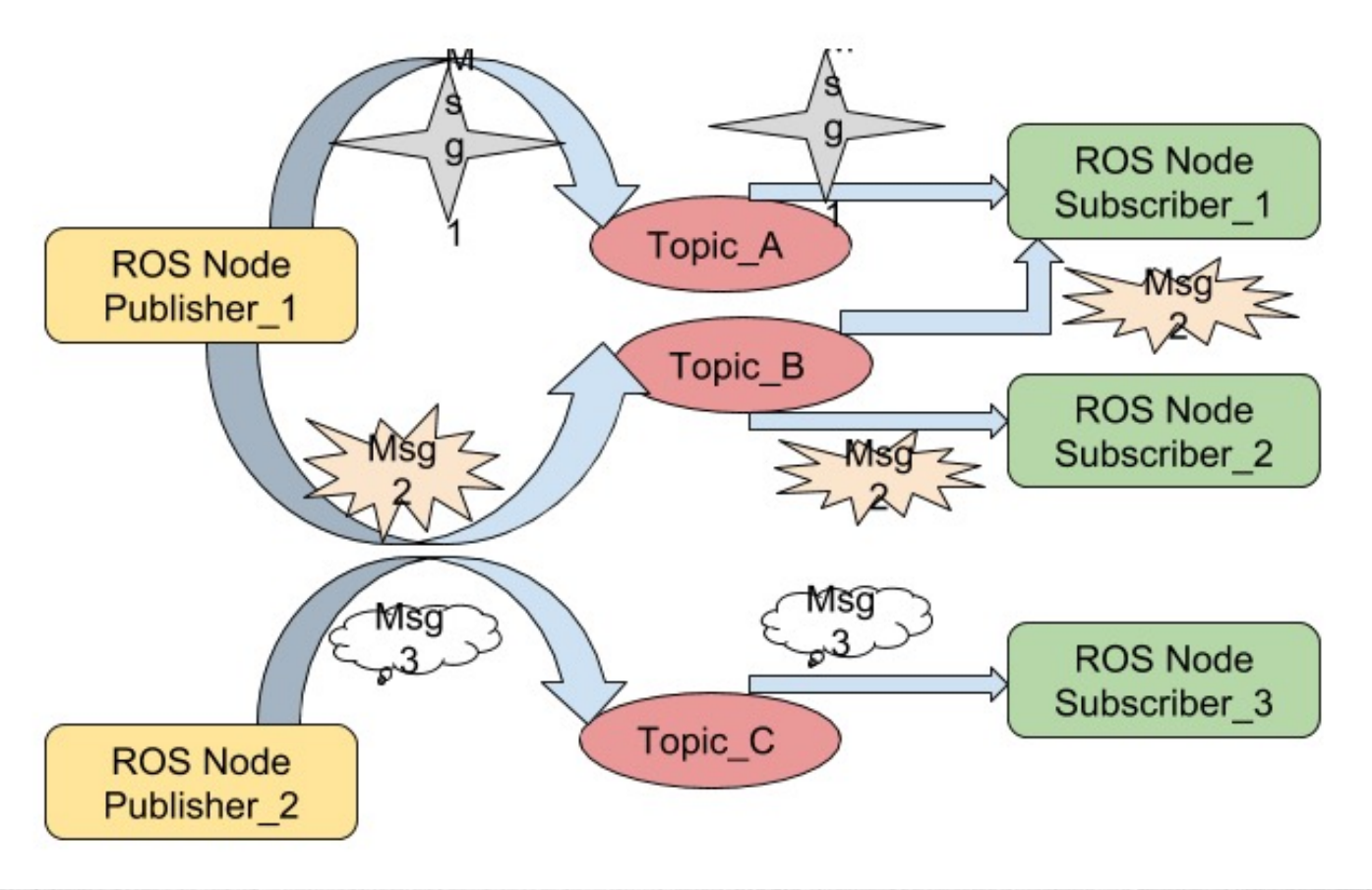


CloudCompare

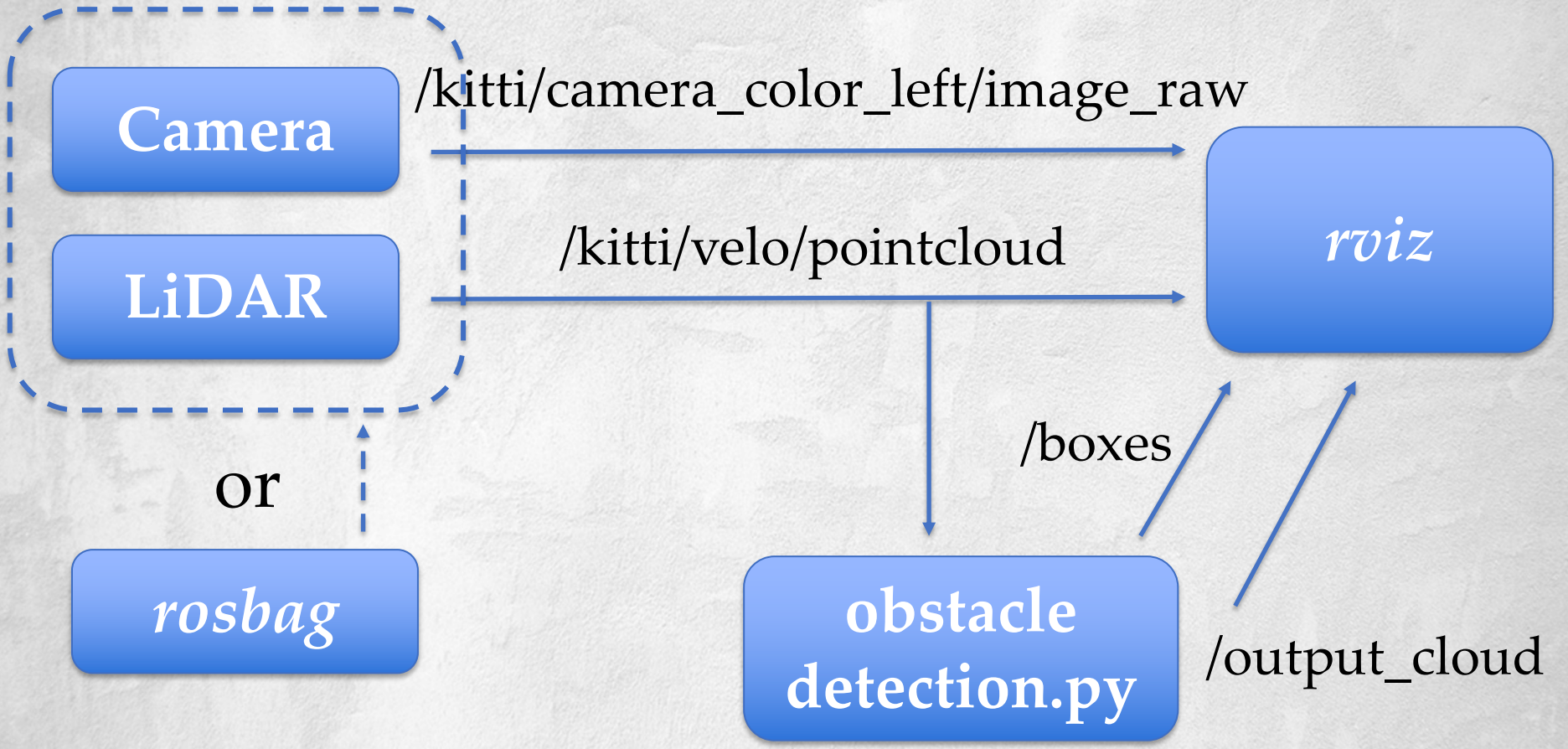
https://www.danielgm.net/cc/

Sensor Messages in ROS

sensor_msgs API



Example


Camera
LiDAR
/kitti/camera_color_left/image_raw
/kitti/velo/pointcloud
rviz
/boxes
or
rosbag
obstacle detection.py
/output_cloud

sensor_msgs/Image Message

File: sensor_msgs/Image.msg

Raw Message Definition

```
# This message contains an uncompressed image
# (0, 0) is at top-left corner of image
#

Header header        # Header timestamp should be acquisition time of image
                     # Header frame_id should be optical frame of camera
                     # origin of frame should be optical center of camera
                     # +x should point to the right in the image
                     # +y should point down in the image
                     # +z should point into to plane of the image
                     # If the frame_id here and the frame_id of the CameraInfo
                     # message associated with the image conflict
                     # the behavior is undefined

uint32 height         # image height, that is, number of rows
uint32 width          # image width, that is, number of columns

# The legal values for encoding are in file src/image_encodings.cpp
# If you want to standardize a new string format, join
# ros-users@lists.sourceforge.net and send an email proposing a new encoding.

string encoding       # Encoding of pixels -- channel meaning, ordering, size
                      # taken from the list of strings in include/sensor_msgs/image_encodings.h

uint8 is_bigendian    # is this data bigendian?
uint32 step           # Full row length in bytes
uint8[] data          # actual matrix data, size is (step * rows)
```

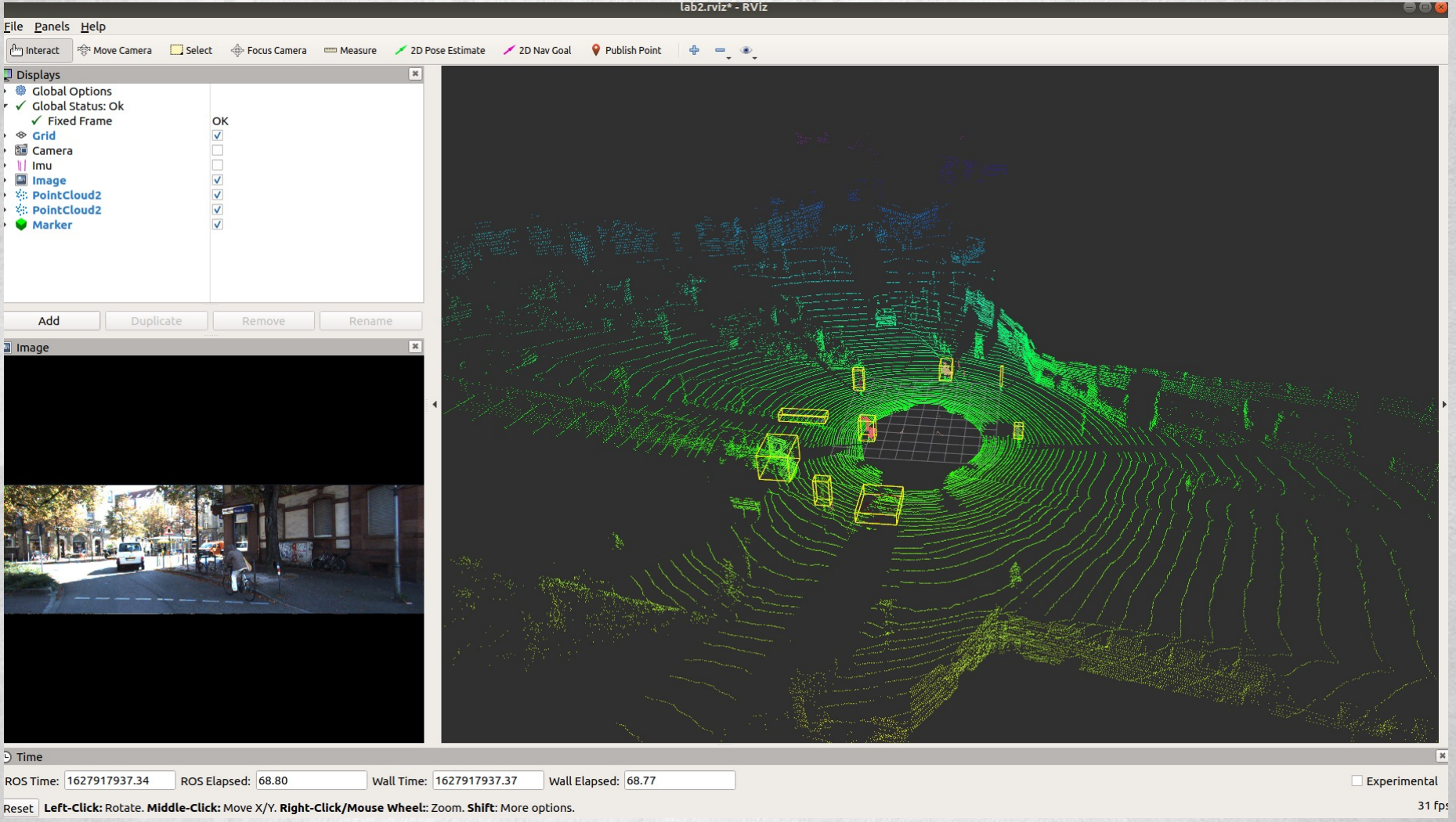
sensor_msgs/PointCloud2 Message

File: sensor_msgs/PointCloud2.msg

Raw Message Definition

```
# This message holds a collection of N-dimensional points, which may
# contain additional information such as normals, intensity, etc. The
# point data is stored as a binary blob, its layout described by the
# contents of the "fields" array.

# The point cloud data may be organized 2d (image-like) or 1d
# (unordered). Point clouds organized as 2d images may be produced by
# camera depth sensors such as stereo or time-of-flight.

# Time of sensor data acquisition, and the coordinate frame ID (for 3d
# points).
Header header

# 2D structure of the point cloud. If the cloud is unordered, height is
# 1 and width is the length of the point cloud.
uint32 height
uint32 width

# Describes the channels and their layout in the binary data blob.
PointField[] fields

bool    is_bigendian # Is this data bigendian?
uint32  point_step   # Length of a point in bytes
uint32  row_step     # Length of a row in bytes
uint8[] data         # Actual point data, size is (row_step*height)

bool is_dense        # True if there are no invalid points
```

RVIZ


lab2.rviz* - RViz
File Panels Help
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
Global Options
Global Status: Ok
Fixed Frame
OK
Grid
Camera
Imu
Image
PointCloud2
PointCloud2
Marker
Add
Duplicate
Remove
Rename
Image
Time
ROS Time: 1627917937.34
ROS Elapsed: 68.80
Wall Time: 1627917937.37
Wall Elapsed: 68.77
Experimental
Reset
Left-Click: Rotate. Middle-Click: Move X/Y. Right-Click/Mouse Wheel:: Zoom. Shift: More options.
31 fps

Challenges in Perception

References

1. http://wiki.ros.org/sensor_msgs
2. https://docs.opencv.org/4.5.0/d6/d00/tutorial_py_root.html
3. https://scikit-image.org/docs/dev/auto_examples/
4. https://pointclouds.org/
5. https://www.danielgm.net/cc/
6. https://towardsdatascience.com/how-to-automate-3d-point-cloud-segmentation-and-clustering-with-python-343c9039e4f5