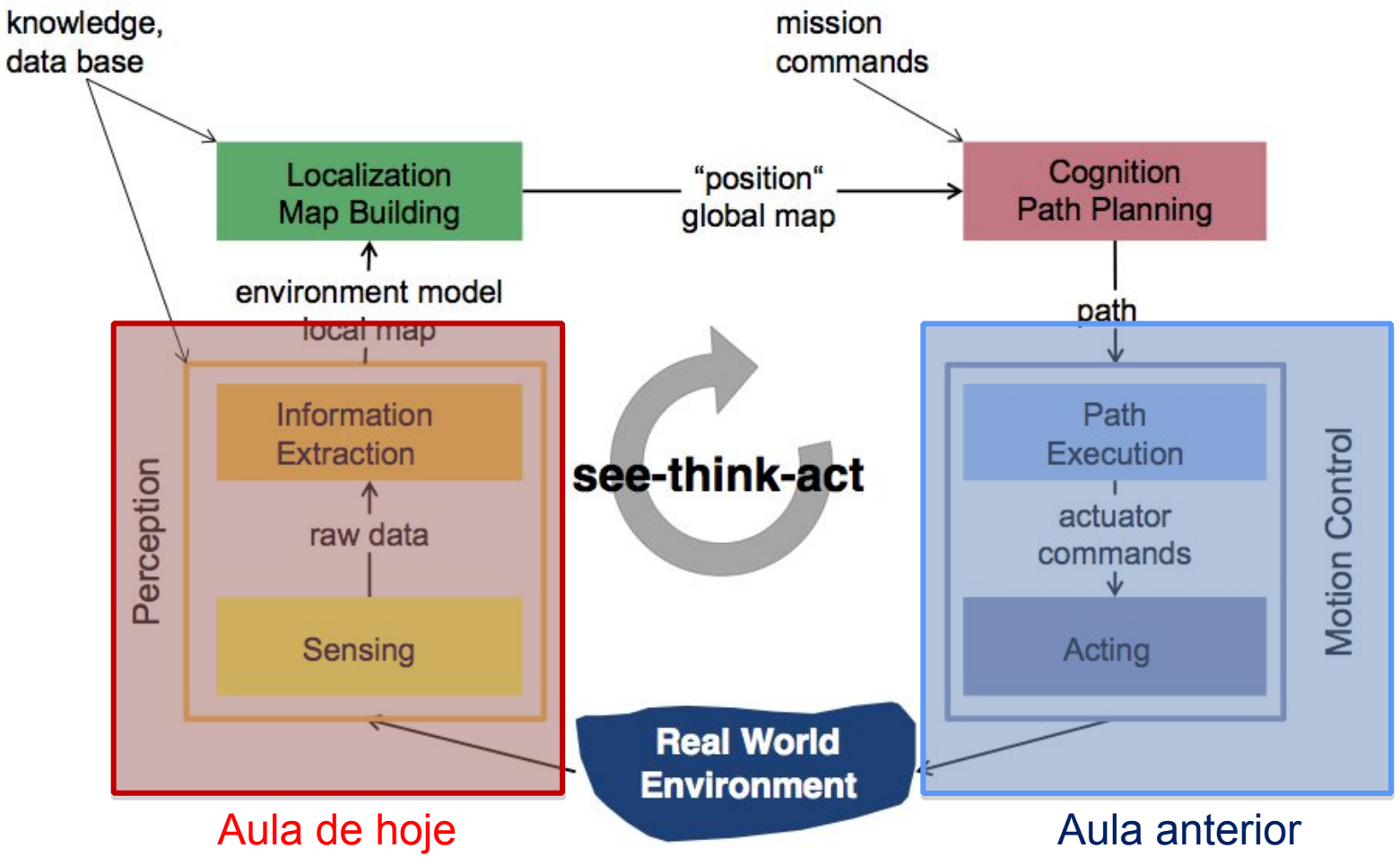
Percepção

Prof. André Schneider de Oliveira
Prof. João Alberto Fabro

Universidade Técnológica Federal do Paraná (UTFPR)

Controle de Robôs Móveis



Percepção

- Capacidade do robô sentir (ou perceber) as características do ambiente em que está inserido
- Podem ser detectadas diferentes informações
  - distância, cores, temperatura, força, altura ...
- A percepção é utilizada para que o robô interaja com o ambiente (movimente-se)

Classificação dos Sensores

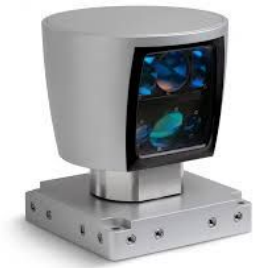
- Quanto a medição:
  - **Sensores Proprioceptivos**: realizam a medição de grandezas internas ao robô
  - **Sensores Exteroceptivos**: medem informações do ambiente

- Quanto ao funcionamento:
  - **Sensores Passivos**: medem a energia do ambiente e sofrem muita influência deste
  - **Sensores Ativos**: emitem sua própria energia e medem a reação.

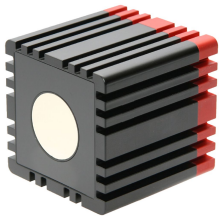
Principais sensores de percepção

- Sonar

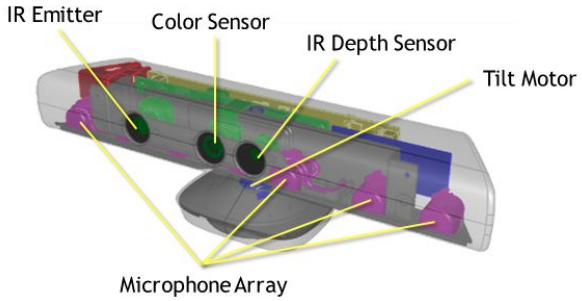
- Laser Scanner ou LIDAR(Light Detection and Ranging)



- Time of Flight (TOF) Camera



- Luz Estruturada



LIDAR

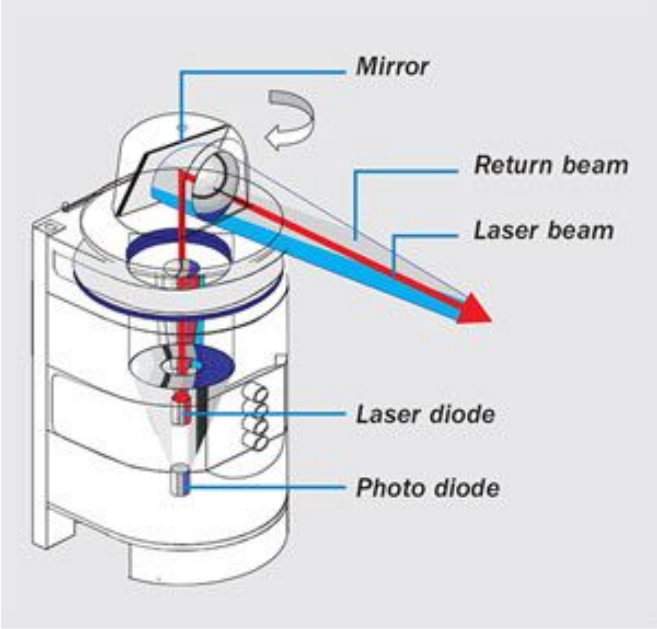
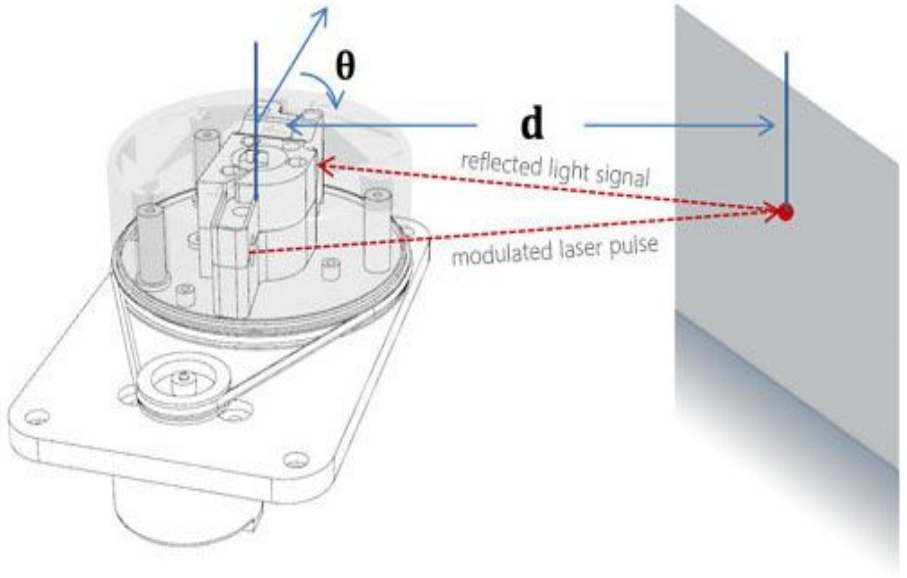
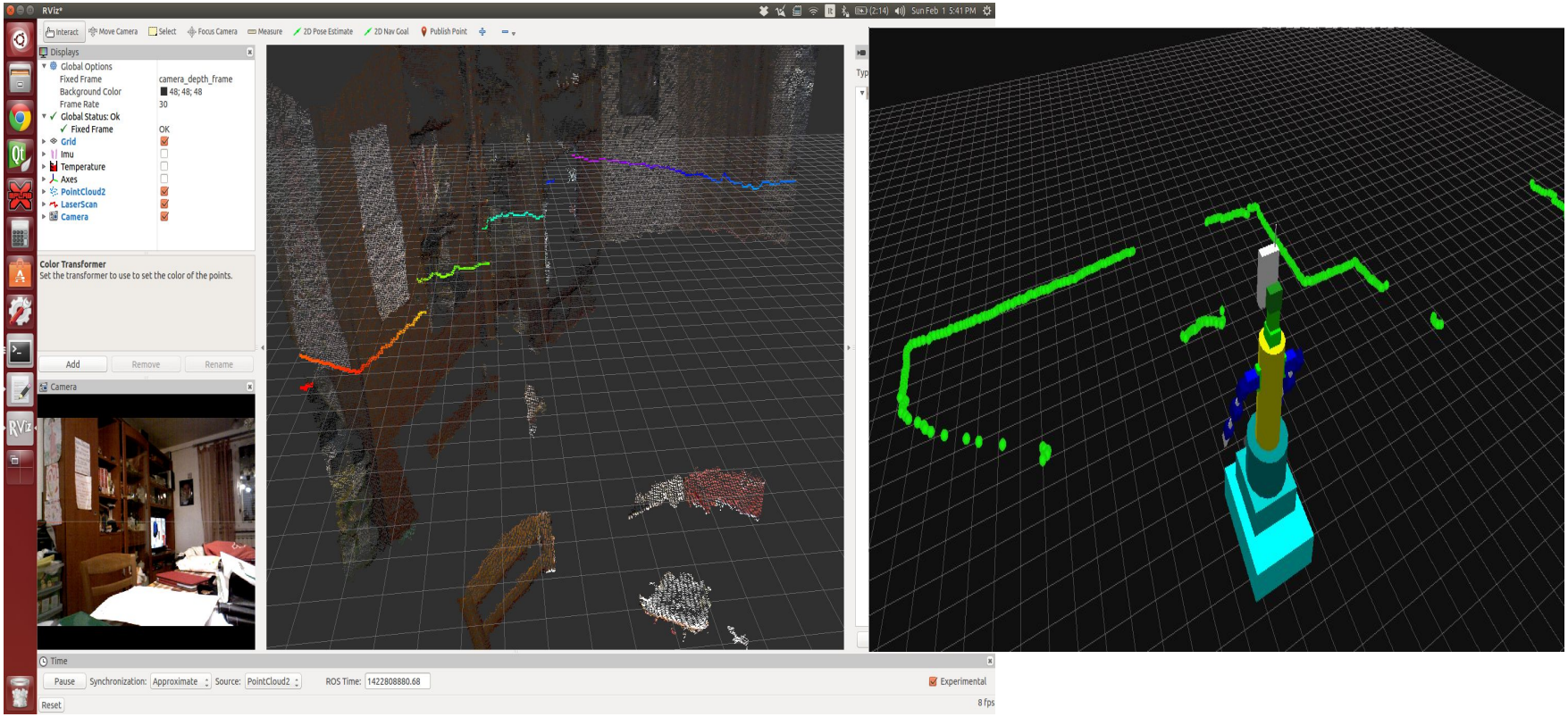


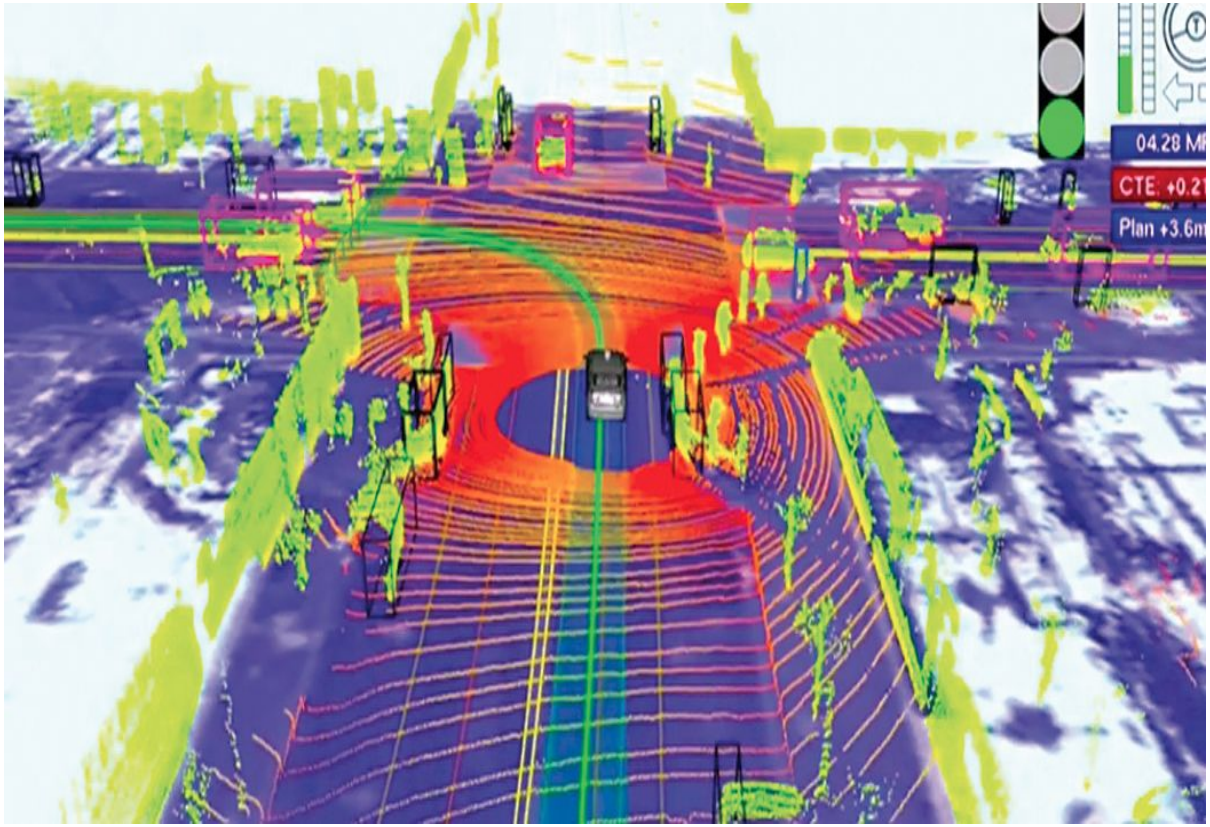
Illustration of LIDAR sensor demonstrating the time of flight principle. *(Courtesy of SICK, Inc.)*



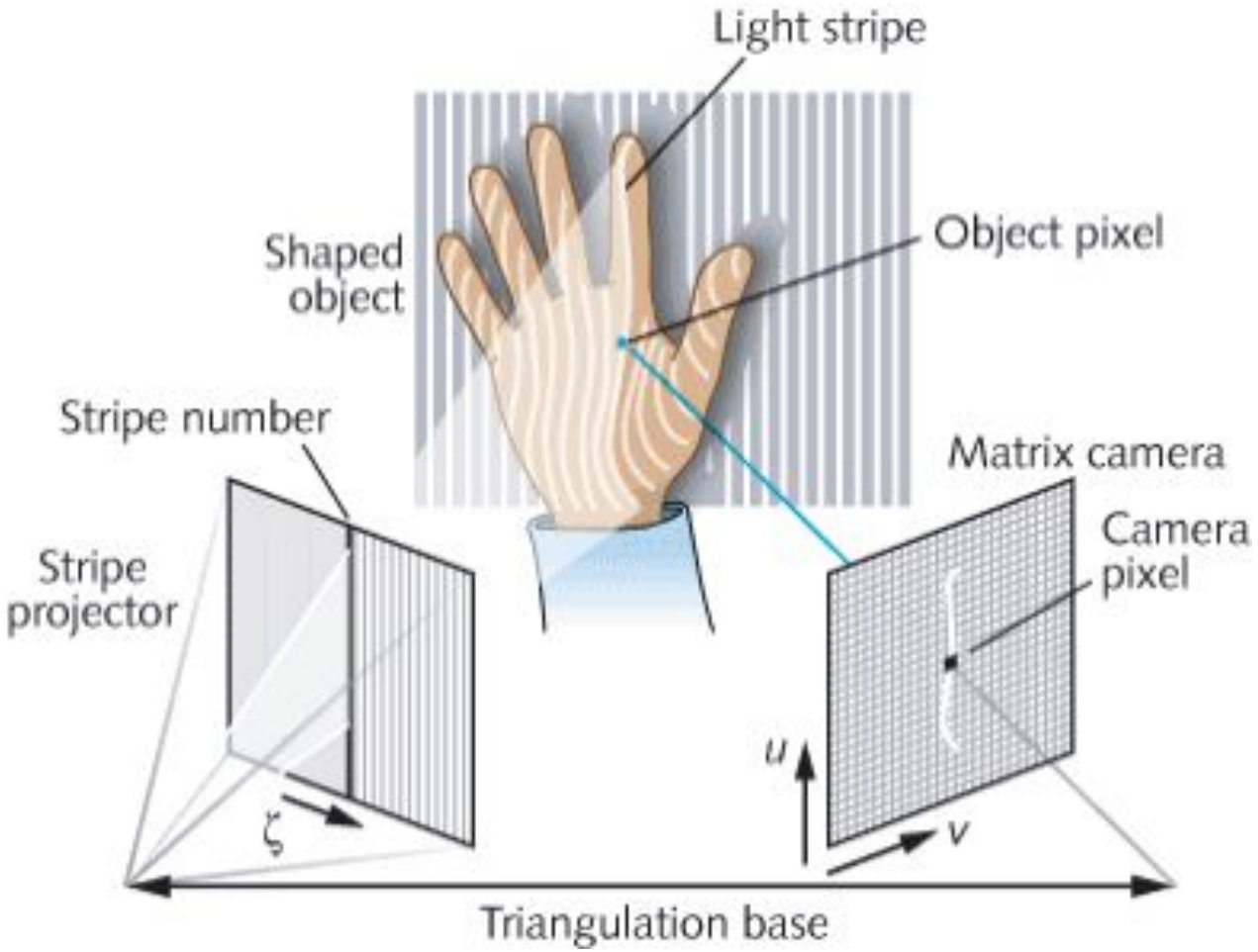
Informações Laserscan 2D



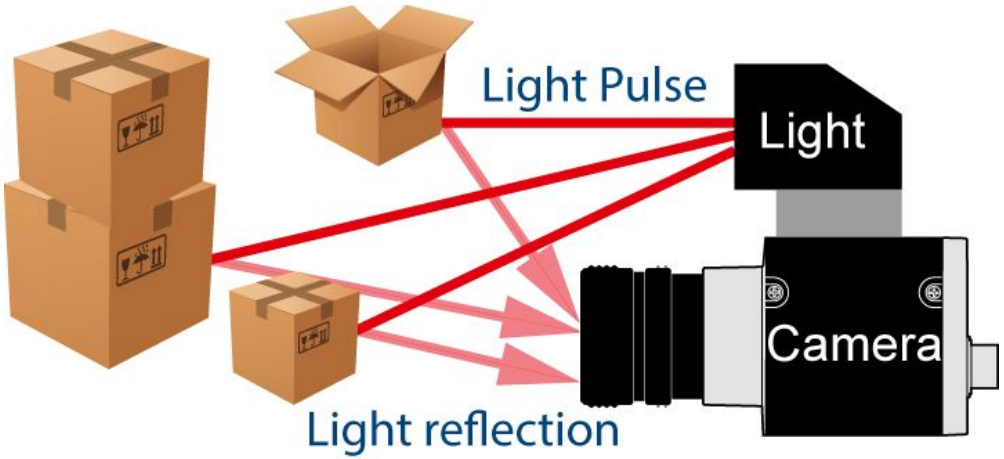
Informações Laserscan 3D

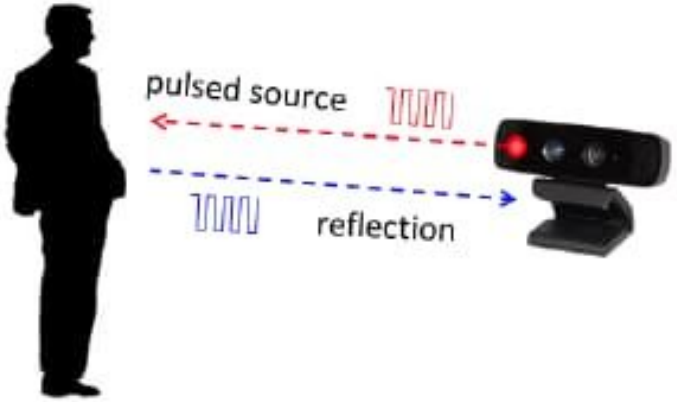


Luz Estruturada

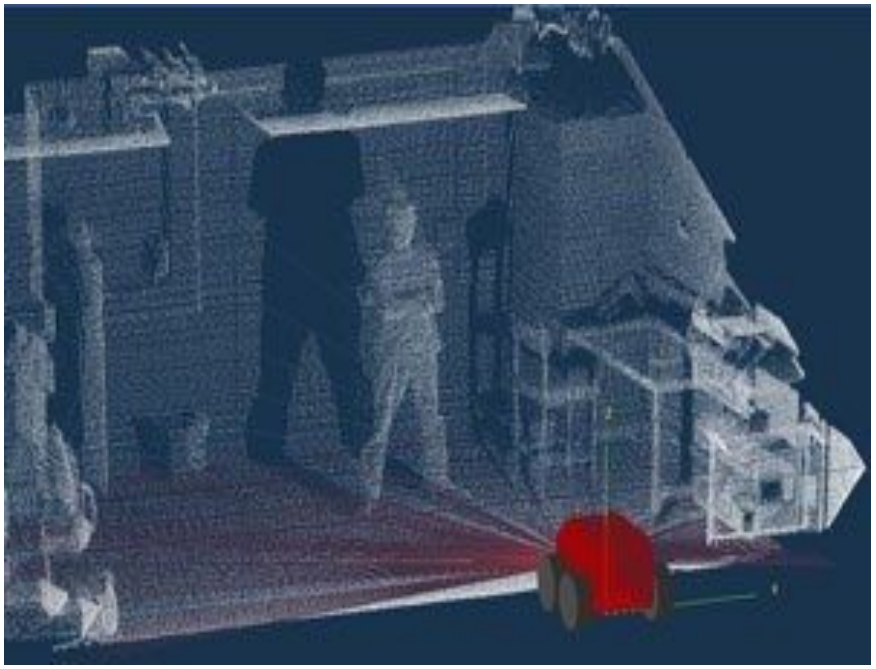


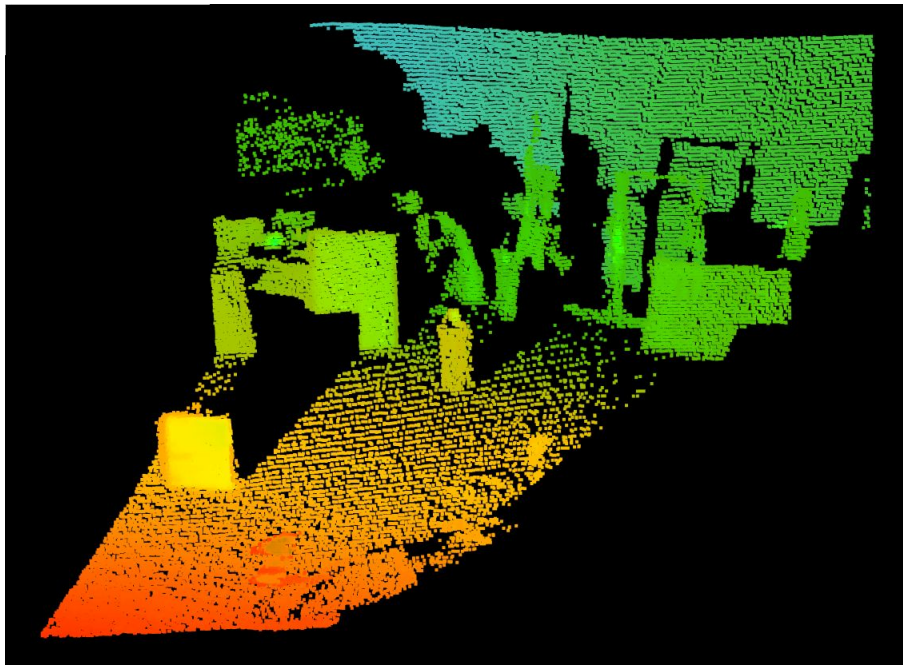
Time-of-flight



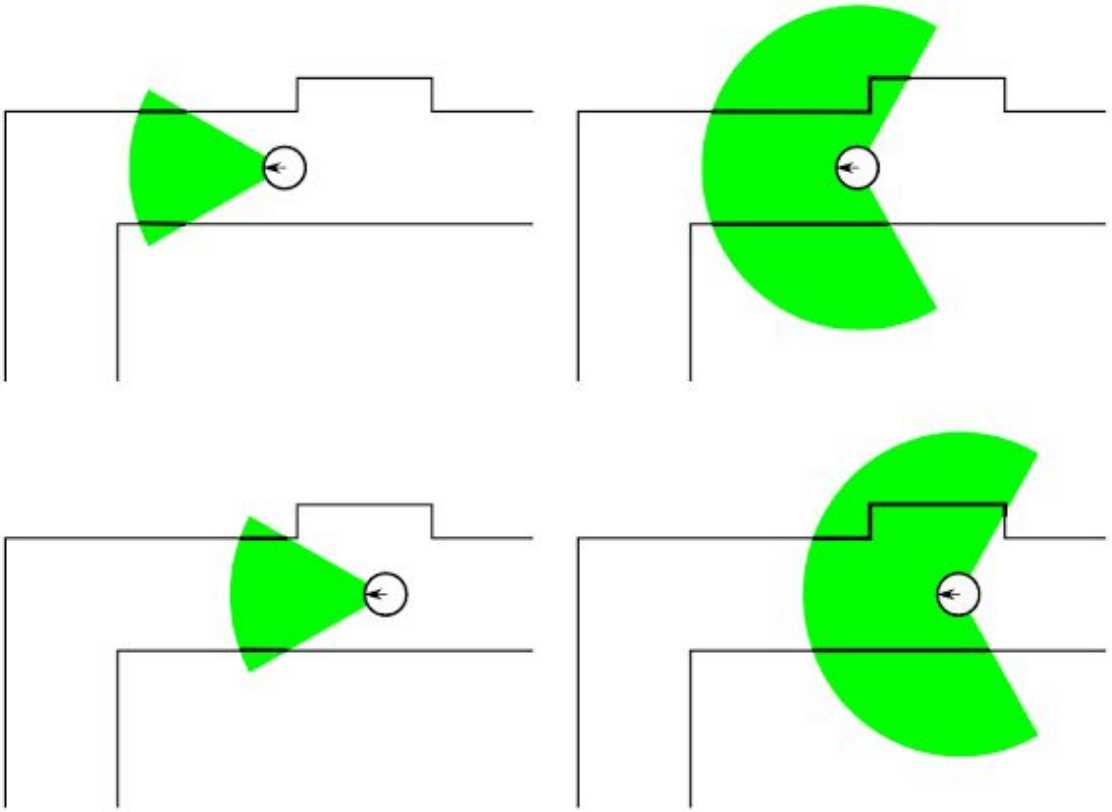


Informações Pointcloud

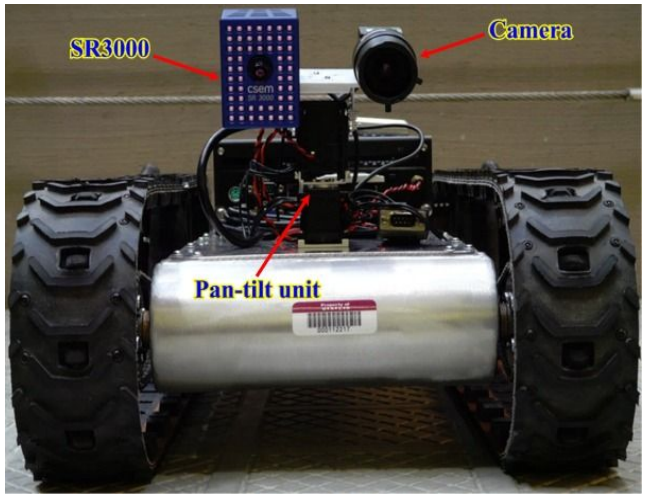




Pointcloud vs Laserscan



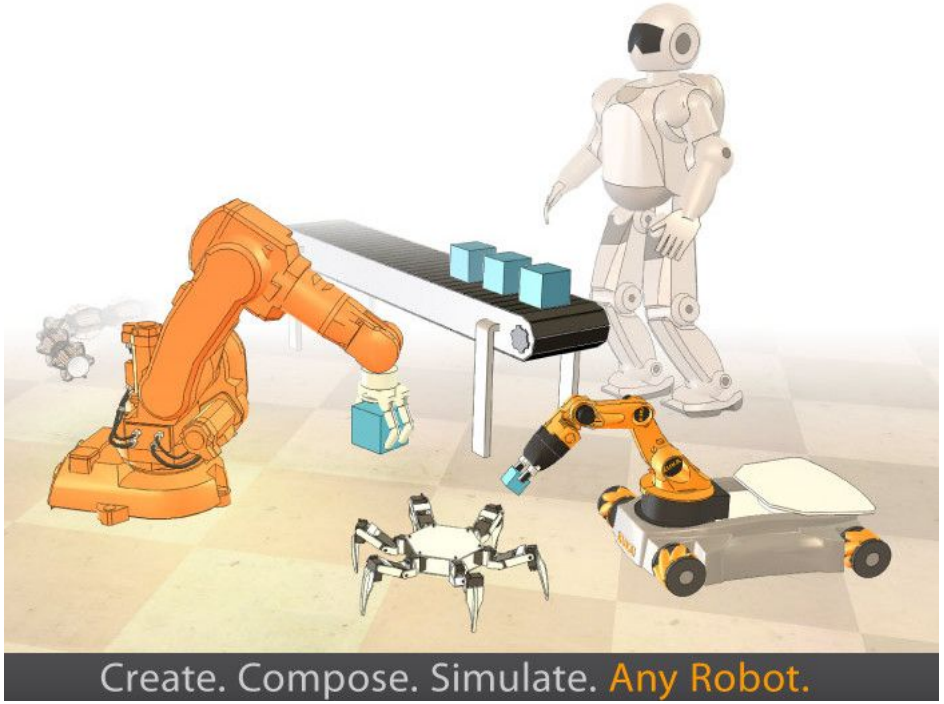
Exemplos de aplicação dos sensores



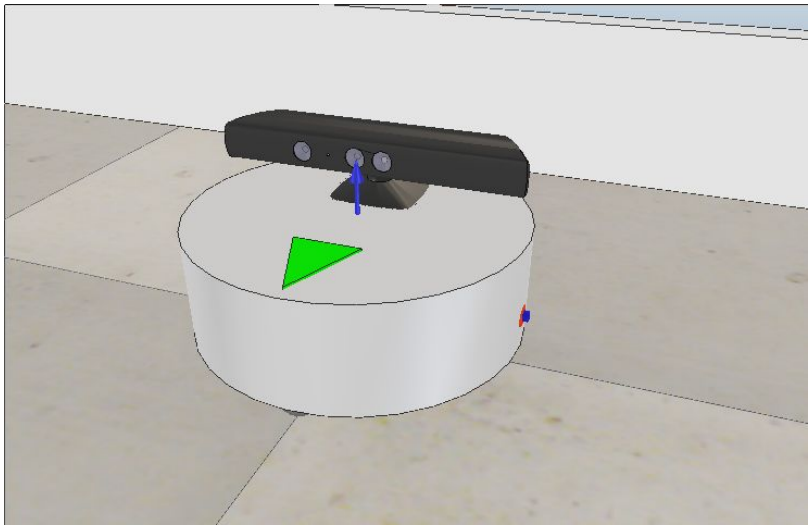
Experimentação Virtual

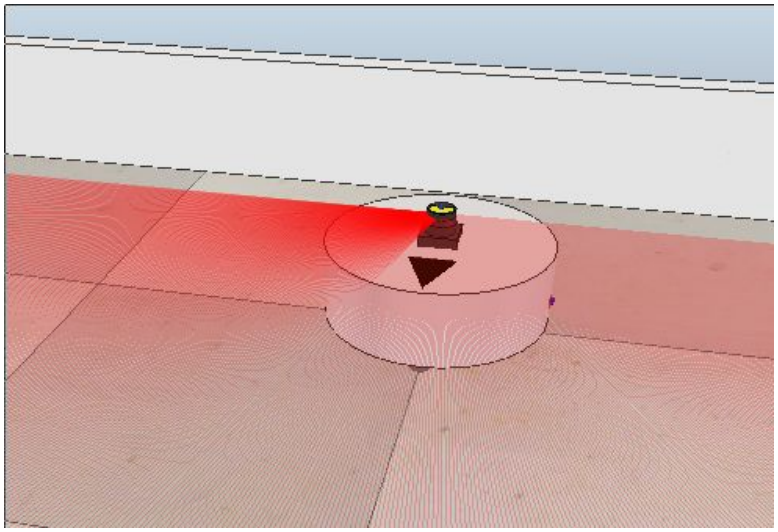
- Coppelia Robotics V-REP: Create. Compose. Simulate. Any Robot

www.coppeliarobotics.com/



Robo Bob





Kinect

tipo: PointCloud2

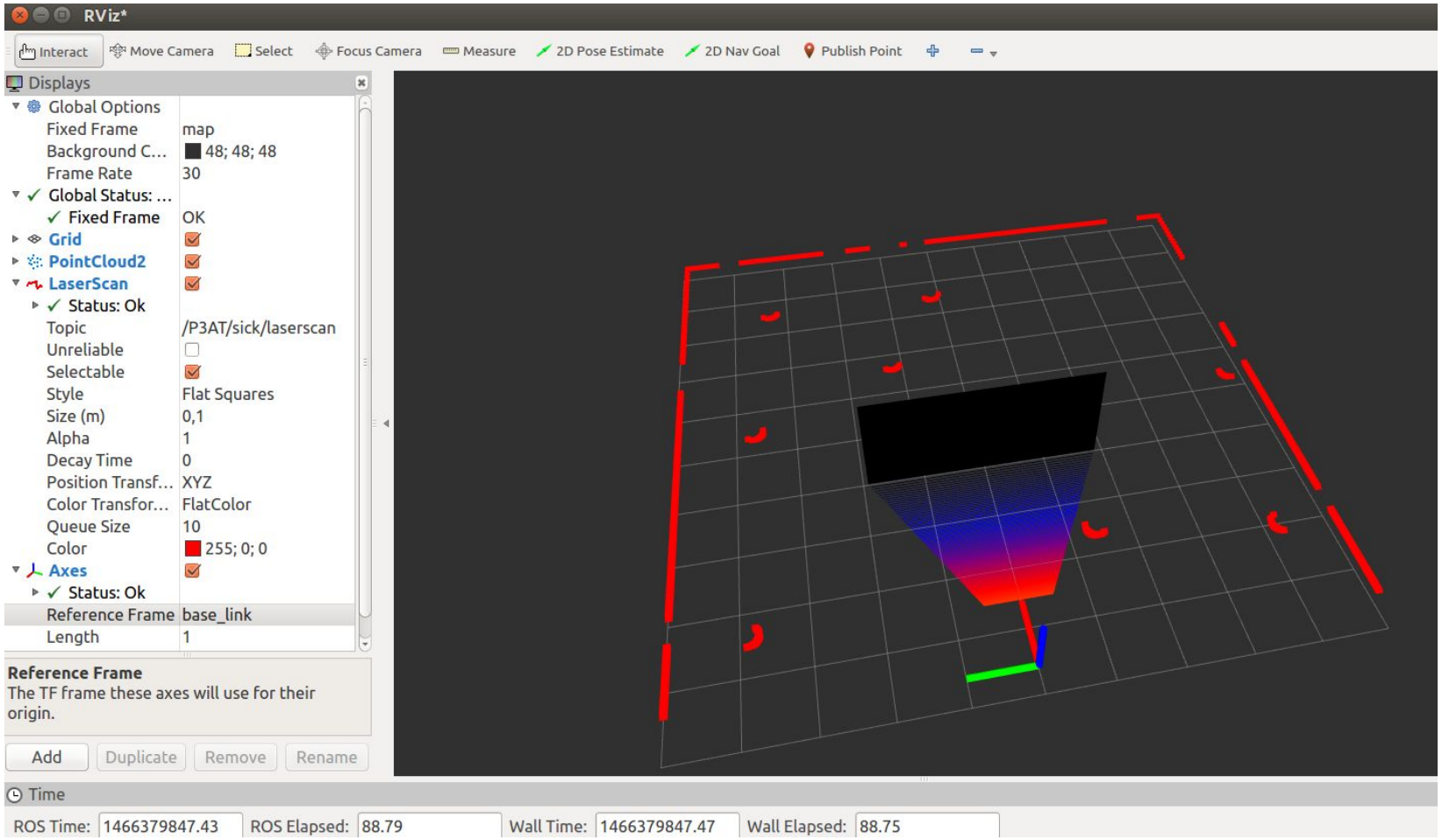
tópico: /pointcloud

Hokuyo

tipo: Laserscan

topico: /scan

Visualizador ROS - Rviz



(no terminal)
rviz

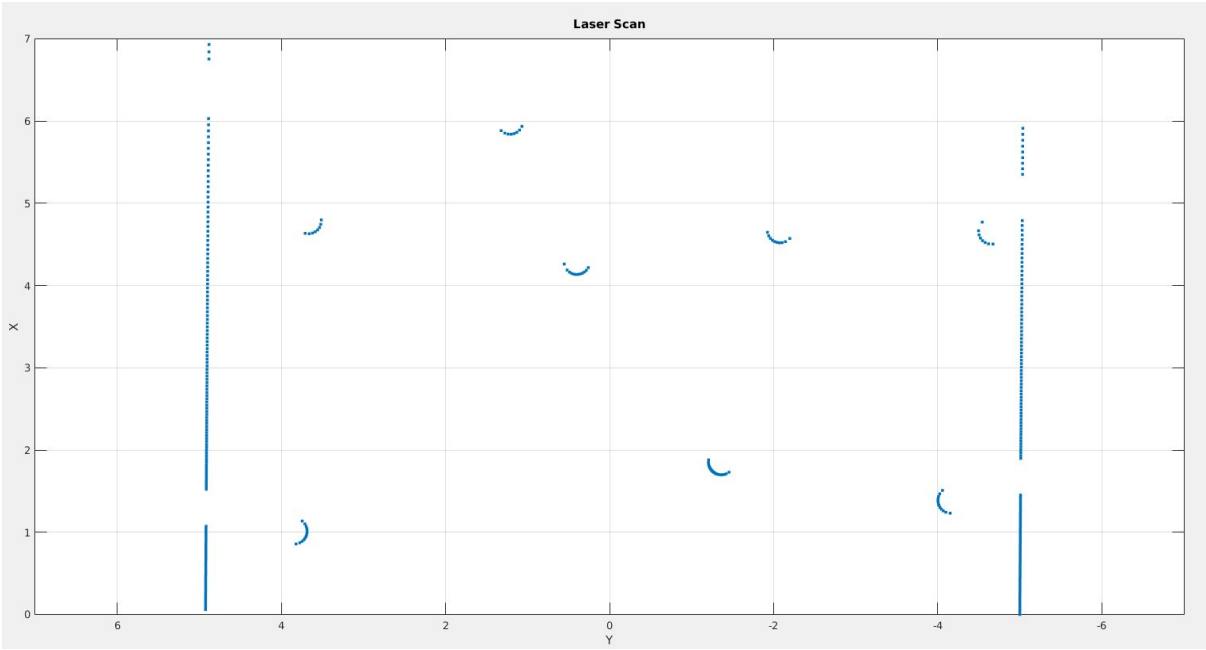
Laserscan

```
# Single scan from a planar laser range-finder
#
# If you have another ranging device with different behavior (e.g. a sonar
# array), please find or create a different message, since applications
# will make fairly laser-specific assumptions about this data

Header header            # timestamp in the header is the acquisition time of
                         # the first ray in the scan.
                         #
                         # in frame frame_id, angles are measured around
                         # the positive Z axis (counterclockwise, if Z is up)
                         # with zero angle being forward along the x axis

float32 angle_min        # start angle of the scan [rad]
float32 angle_max        # end angle of the scan [rad]
float32 angle_increment  # angular distance between measurements [rad]

float32 time_increment   # time between measurements [seconds] - if your scanner
                         # is moving, this will be used in interpolating position
                         # of 3d points
float32 scan_time        # time between scans [seconds]

float32 range_min        # minimum range value [m]
float32 range_max        # maximum range value [m]

float32[] ranges         # range data [m] (Note: values < range_min or > range_max should be discarded)
float32[] intensities    # intensity data [device-specific units].  If your
                         # device does not provide intensities, please leave
                         # the array empty.
```

rosmsg show sensor_msgs/LaserScan

Matlab: LaserScan

```
%cria msg
scandata = rosmessage('sensor_msgs/LaserScan')

%cria subscriber
sublaser = rossubscriber('/scan')

%Le as informacoes
scandata = receive(sublaser,10)

%Plota os dados
figure
plot(scandata,'MaximumRange',7)

%Converte em coordenadas XY
xy = readCartesian(scandata)
```



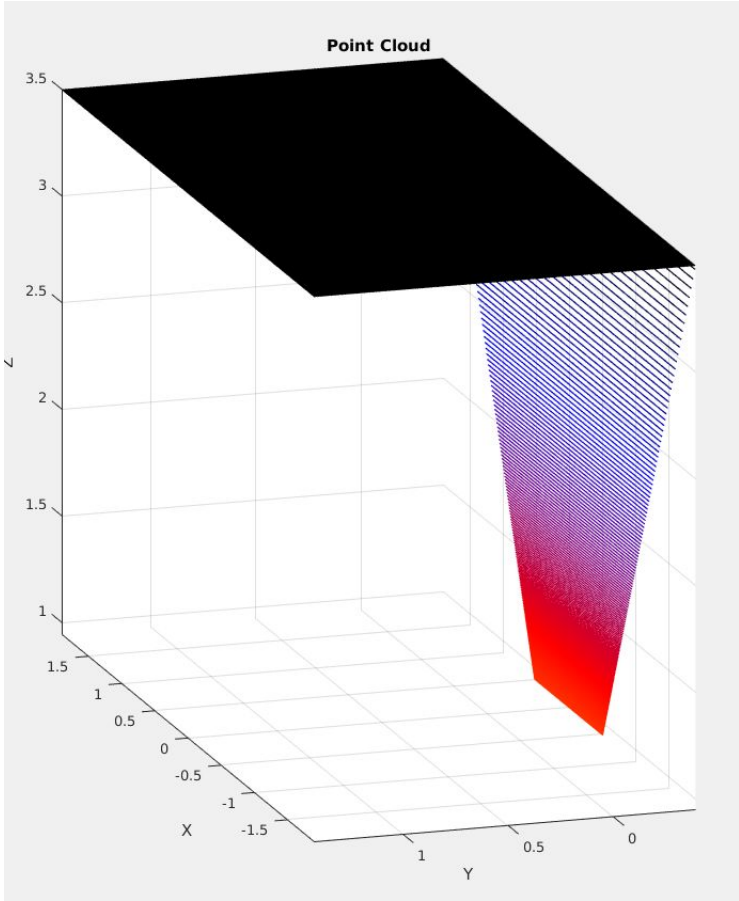
PointCloud2

```
# 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
```

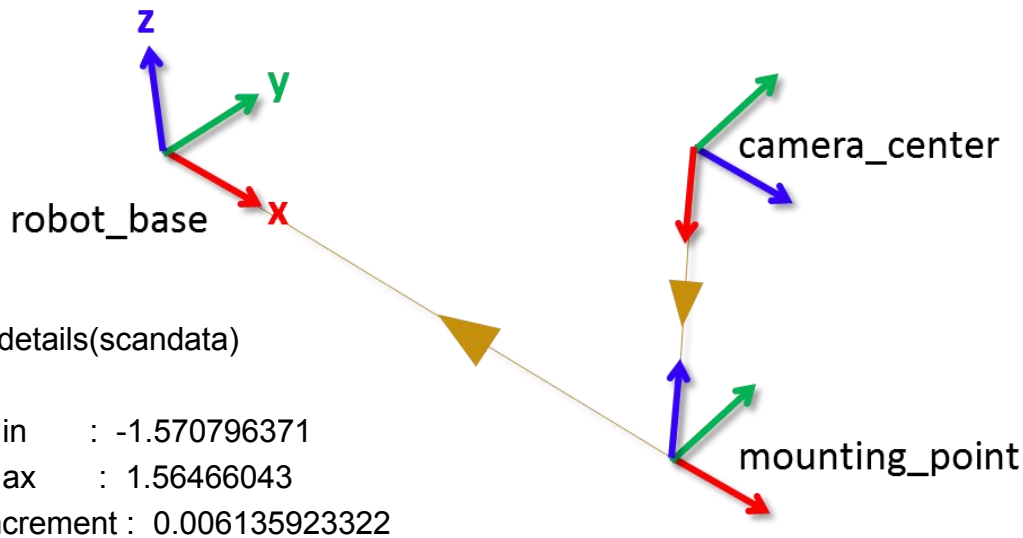
rosmsg show sensor_msgs/PointCloud2

Matlab: PointCloud2

```
%cria msg
pc2data = rosmessage('sensor_msgs/PointCloud2')

%cria subscriber
subpc2 = rossubscriber('/pointcloud')

%Le as informacoes
pc2data = receive(subpc2,10)

%define que a leitura sera de 640x480x3 ao inves de 307200x3
pc2data.PreserveStructureOnRead=1;

%Converte em coordenadas XYZ
xyz = readXYZ(pc2data)

%Remove dados invalidos
xyzvalid = xyz(~isnan(xyz(:,1)),:)

%Plota os dados
figure
scatter3(pc2data)
```



Ponto de vista das msgs



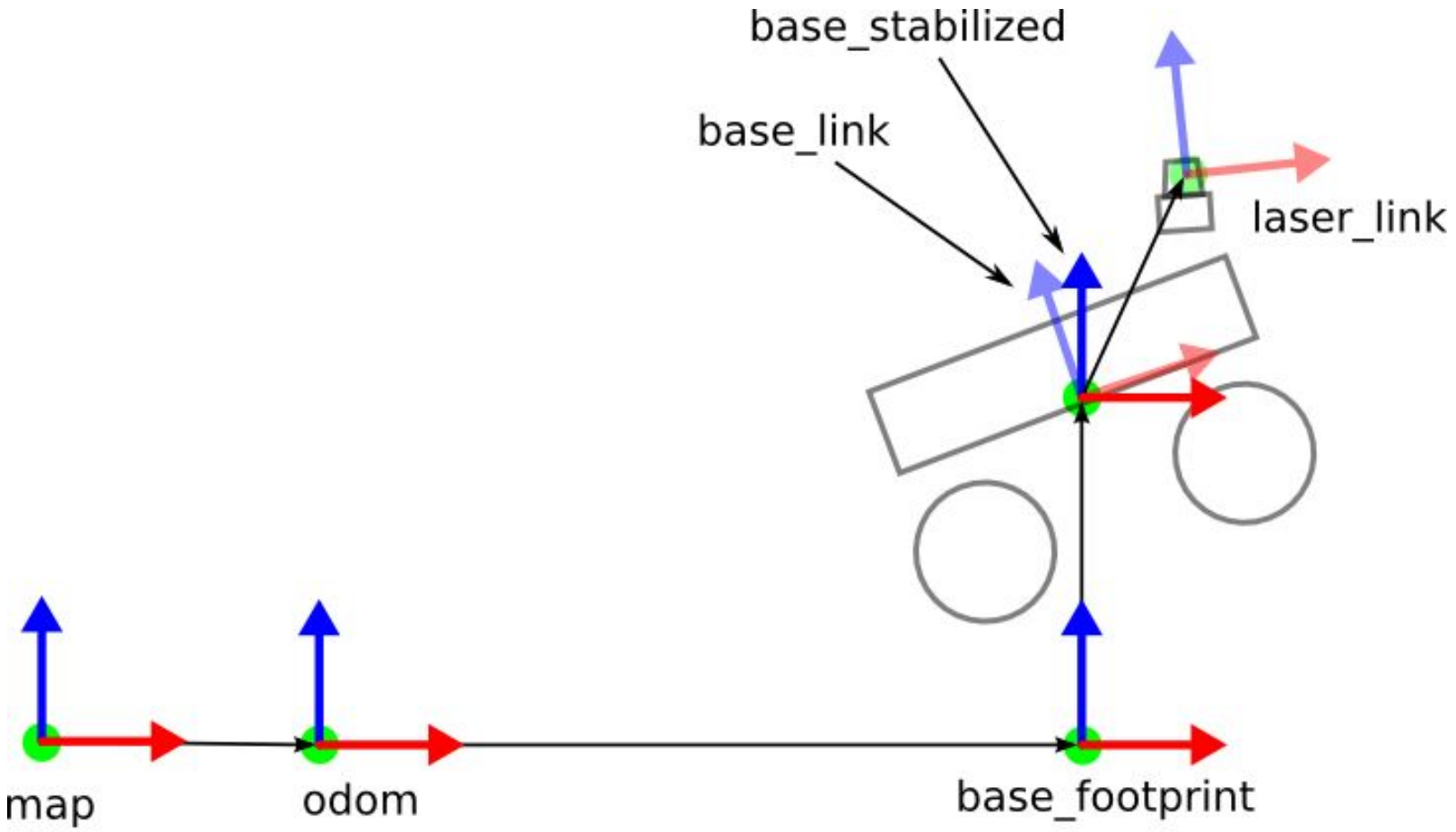
>> showdetails(pc2data)

```
  Height          : 480
  Width           : 640
  IsBigendian     : 0
  PointStep       : 16
  RowStep         : 10240
  IsDense         : 1
  Data            : **********

  Header
    Seq     : 4718
    FrameId :  /kinect_visionSensor
    Stamp
      Sec  :  1466378052
      Nsec :  640330820
```

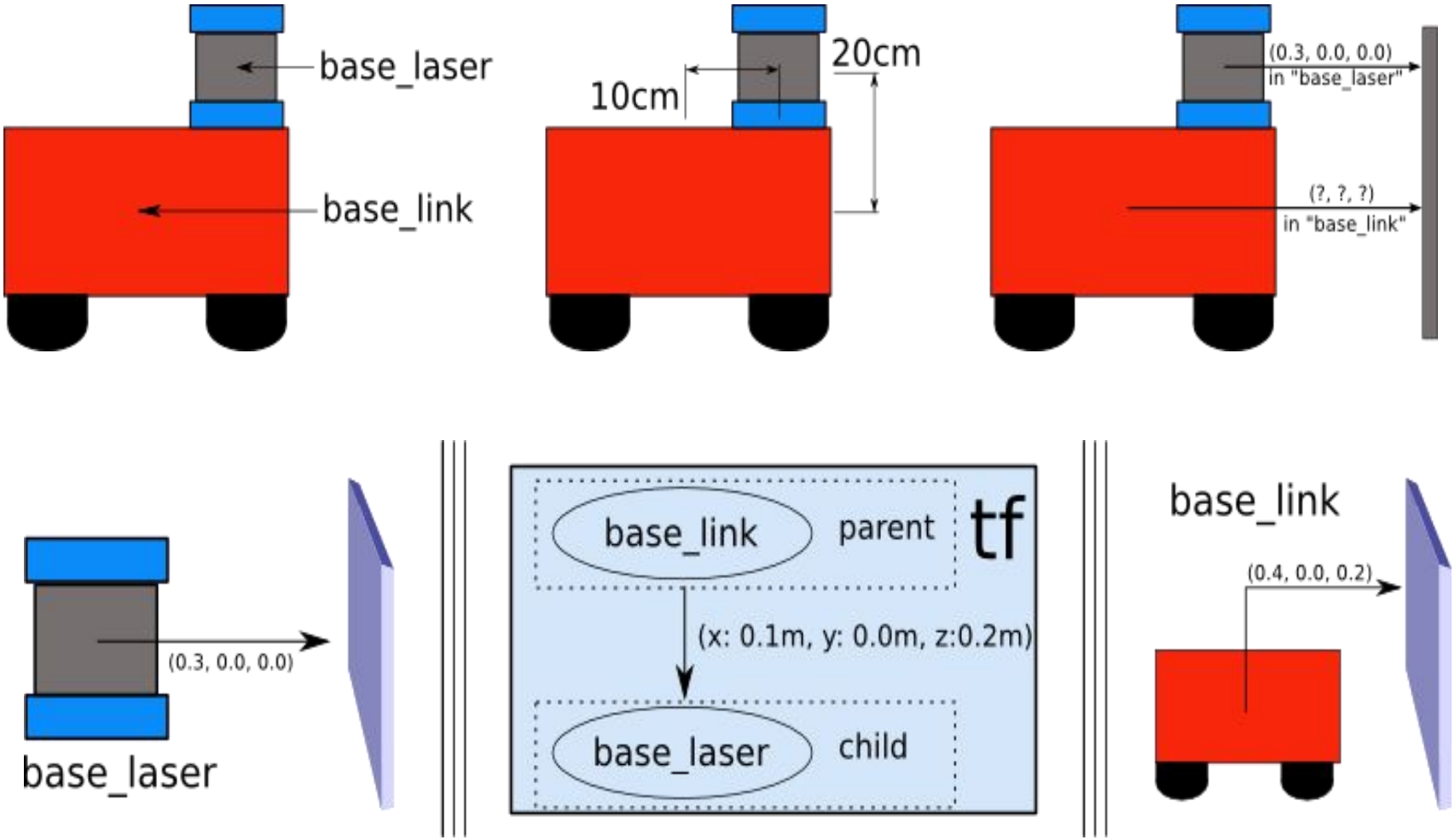
>> showdetails(scandata)

```
  AngleMin       : -1.570796371
  AngleMax       : 1.56466043
  AngleIncrement :  0.006135923322
  TimeIncrement  :  1.736111153e-05
  ScanTime       : 0.02500000037
  RangeMin       : 0.02300000004
  RangeMax       : 60
  Ranges         : ***********
  Intensities    : []
  Header
    Seq     : 7246
    FrameId : /Hokuyo_URG_04LX_UG01_ROS
    Stamp
      Sec  :  1466378360
      Nsec :  24563560
```

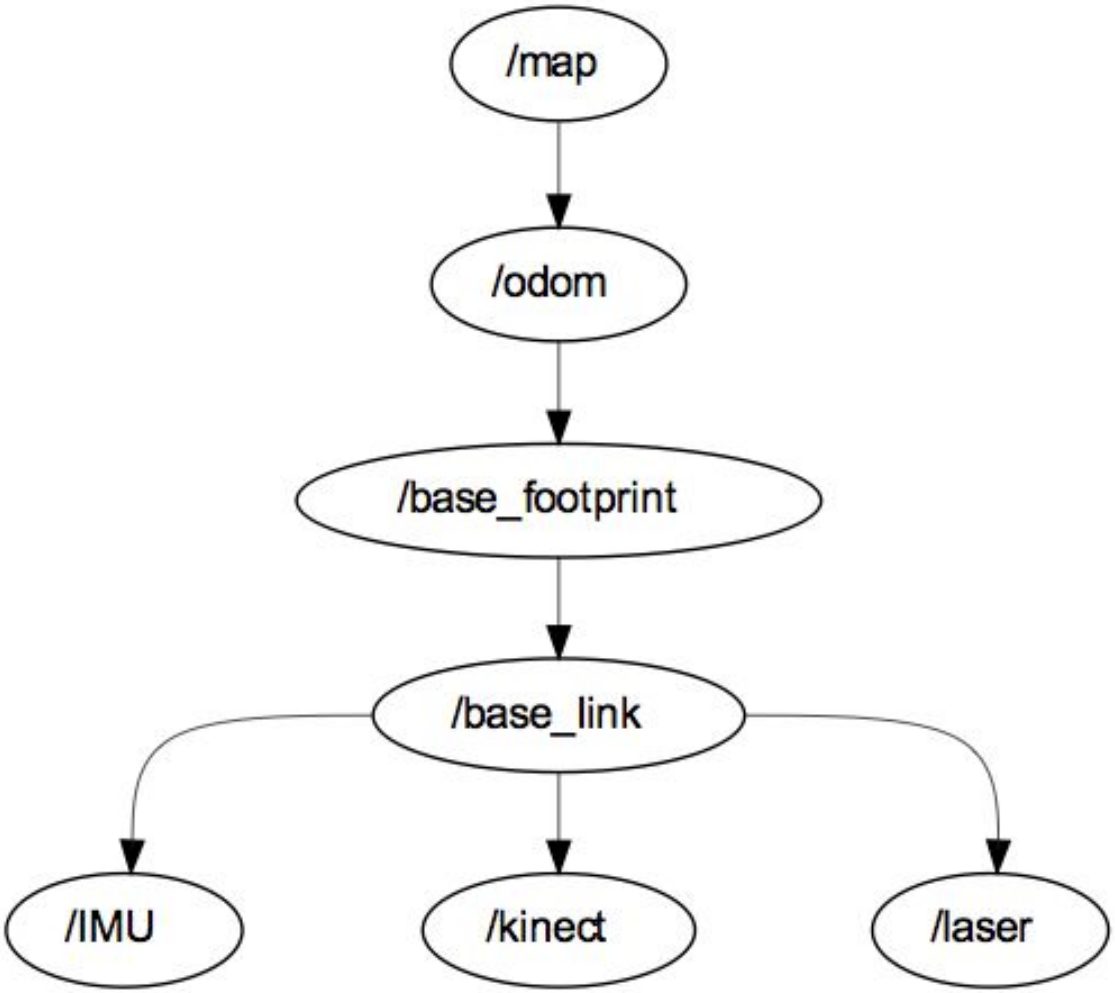
Sistemas de coordenadas



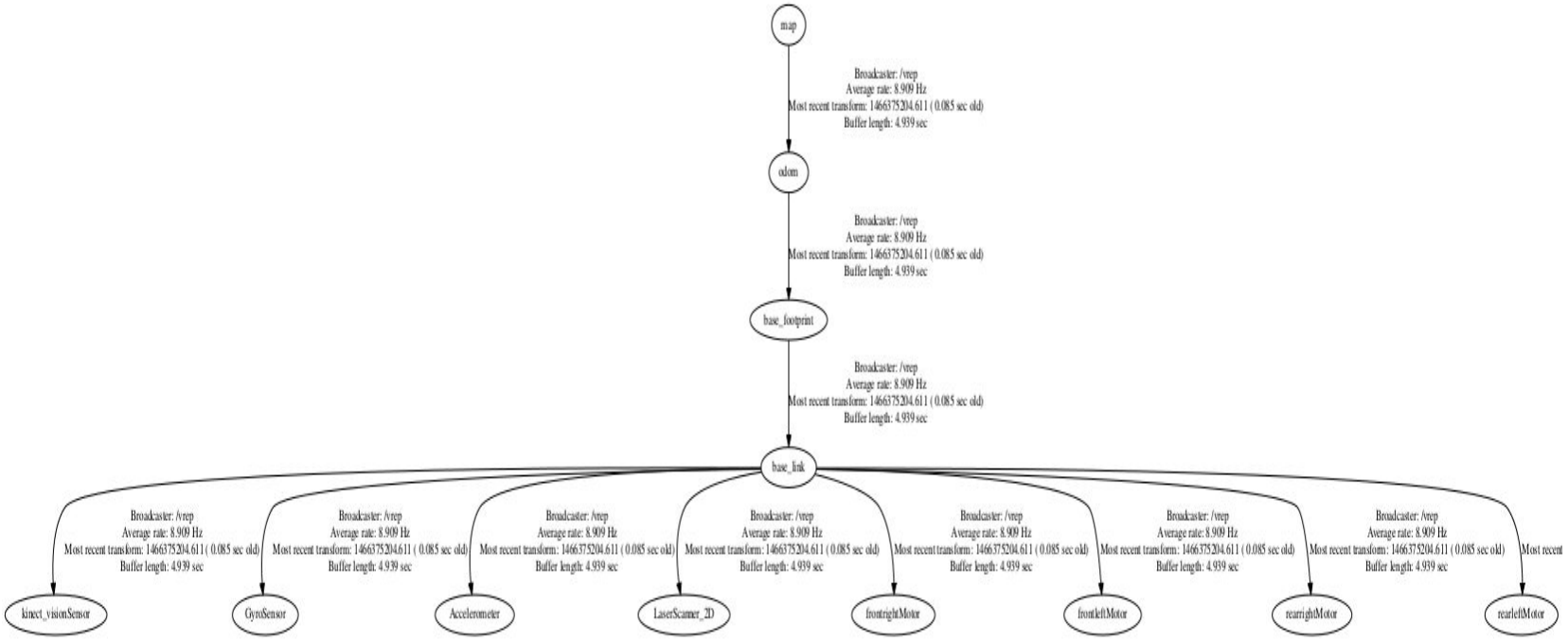
Sistemas de coordenadas



Árvore de transformações

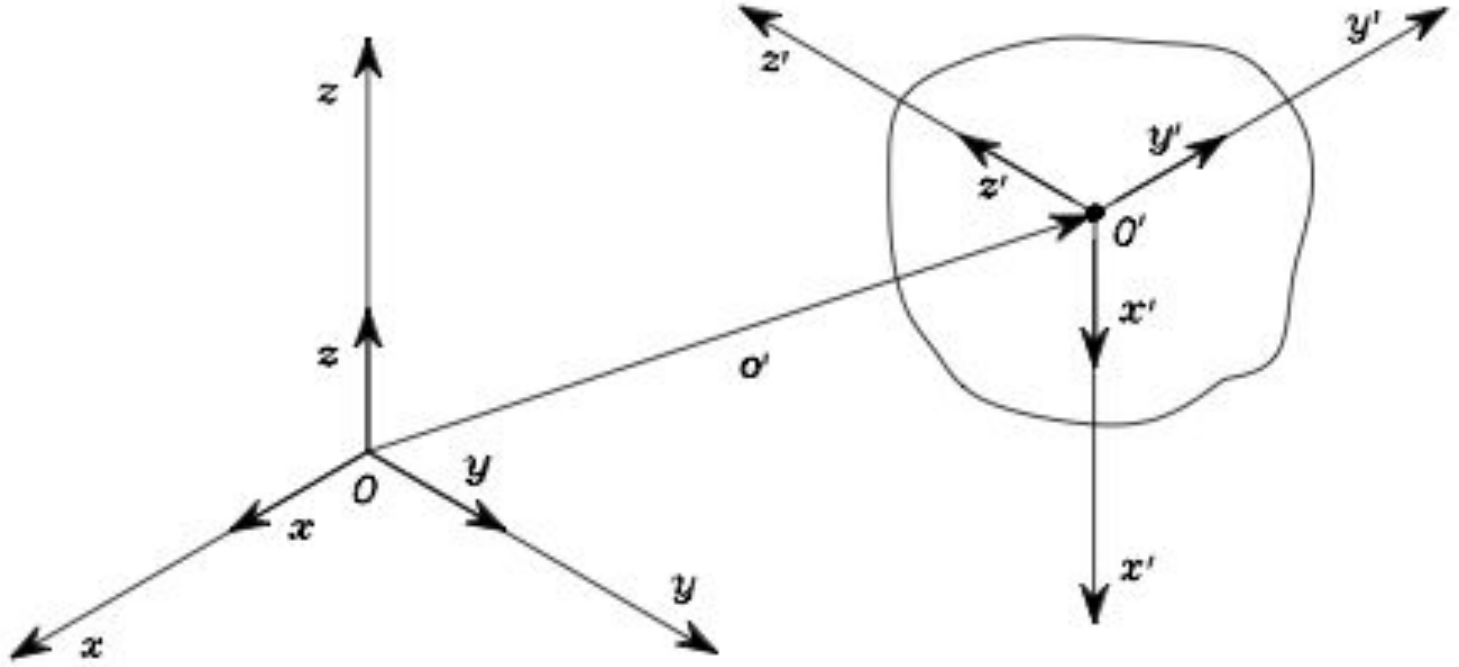


Árvore de transformações



(no terminal)
rosrun tf view_frames
evince frames.pdf

Transformações



Matlab: Transformações

LaserScan

```
%cria a arvore de TF no matlab
tftree = rostf

%espera 1seg para receber e mostra as tf disponiveis
pause(1);
tftree.AvailableFrames

%transformar pointcloud2 para o base_link
tfpc2 = transform(tftree, 'base_link', pc2data)

%confere a transformacao
showdetails(tfpc2data)

%a transformacao de laserscan para o base_link
(precisa ser ponto a ponto)
%cria a msg do tipo ponto
pt = rosmessage('geometry_msgs/PointStamped');

for i=1:size(xy,1)
    pt.Header.FrameId = 'LaserScanner_2D';
    pt.Point.X = xy(i,1);
    pt.Point.Y = xy(i,2);
    pt.Point.Z = 0;
    tfpt = transform(tftree, 'base_link', pt);
    tfxy(i,:) = [tfpt.Point.X tfpt.Point.Y];
end
```

PointCloud

```
%cria a arvore de TF no matlab
tftree = rostf

%espera 1seg para receber e mostra as tf disponiveis
pause(1);
tftree.AvailableFrames

%transformar pointcloud2 para o base_link
tfpc2 = transform(tftree, 'base_link', pc2data)

%confere a transformacao
showdetails(tfpc2data)
```