12장. FOLLOW-BOT

동의대학교 컴퓨터소프트웨어공학과
이종민 교수

목차

- Python용 OpenCV 설치
- 영상 획득
- 선 감지
- 선 따라가기

Python용 OpenCV 설치

$ pip2 install [--upgrade] opencv-python

$ pip2 install [--upgrade] opencv-contrib-python

$ pip2 install imutils

$ pip2 search opencv

...
opencv-python (4.1.0.25) - Wrapper package for OpenCV python bindings.
...
opencv-python-headless (4.1.0.25) - Wrapper package for OpenCV python bindings.
...
imutils (0.5.2) - A series of convenience function
...

OpenCV-Python Tutorial

- OpenCV Documentation
  - https://docs.opencv.org/

- OpenCV: OpenCV-Python Tutorials
  - https://docs.opencv.org/4.1.0/d6/d00/tutorial_py_root.html
  - https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_tutorials.html

follower_opencv2.py

목적: ROS CvBridge를 이용하여 ROS 이미지와 OpenCV 이미지 변환 방법 학습

문제점: Ubuntu 16.04 + ROS Kinetic에서 cv2.imshow(), cv2.waitKey() 등의 메소드가 블록되는 현상 발생
→ 해결 방법: 처리된 이미지를 ROS Image 메시지로 발행하여 rviz에서 확인

```
1  #!/usr/bin/env python
2
3  import cv2
4  import cv_bridge
5  import rospy
6  from sensor_msgs.msg import Image
7
8
9  class Follower:
10     def __init__(self):
11         self.bridge = cv_bridge.CvBridge()
12         # cv2.namedWindow("window", 1)
13         self.image_sub = rospy.Subscriber('/camera/rgb/image_raw',
14                                           Image, self.image_callback)
15         self.image_pub = rospy.Publisher('camera/rgb/image_raw/cv2_to_imgmsg', Image,
           queue_size=1)  # LJM: added
```

CvBridge 객체 생성

Image 유형의 /camera/rgb/image_raw 토픽 구독자 생성

Image 유형의 /camera/rgb/image_raw/cv2_to_imgmsg 토픽 발행자 생성

```
17      def image_callback(self, msg):
18          image = self.bridge.imgmsg_to_cv2(msg, desired_encoding='bgr8')
19          # cv2.imshow("window", image)  # blocked in Ubuntu 16.04 + ROS Kinetic
20          cv2.imwrite('test.jpg', image)  # LJM: added for test
21          # cv2.waitKey(1)  # blocked in Ubuntu 16.04 + ROS Kinetic
22          # LJM: added - begin
23          image_msg = self.bridge.cv2_to_imgmsg(image, 'bgr8')
24          self.image_pub.publish(image_msg)
25          # LJM: added - end
26          return
27
28
29  rospy.init_node('follower')
30  follower = Follower()
31  rospy.spin()
```

sensor_msgs/Image 메시지를 numpy.ndarray 객체로 변환

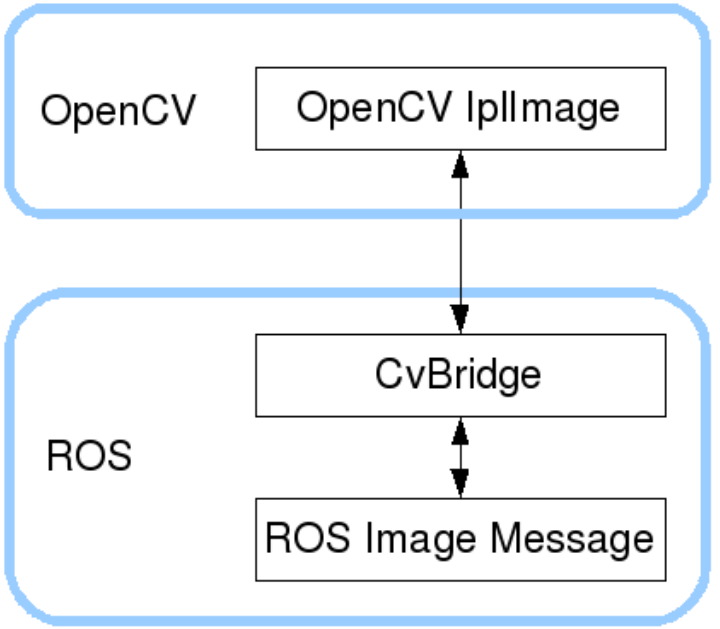
numpy.ndarray 객체를 sensor_msgs/Image 메시지로 변환

ROS 이미지 메시지를 발행

cv_bridge 패키지

- ROS 이미지 메시지 ⇔ OpenCV 이미지
  : sensor_msgs::Image ⇔ cv::Mat (numpy.ndarray)
- cv_bridge 튜토리얼
  - http://wiki.ros.org/cv_bridge/Tutorials

- 주요 클래스
  - 파일: /opt/ros/kinetic/lib/python2.7/dist-packages/cv_bridge/core.py
  - CvBridge
    - imgmsg_to_cv2(img_msg, desired_encoding = "passthrough")
    - cv2_to_compressed_imgmsg(cvim, dst_format = "jpg")
    - cv2_to_imgmsg(cvim, encoding = "passthrough")



이미지 인코딩

- 파일: /opt/ros/kinetic/include/sensor_msgs/image_encodings.h
- 이미지 인코딩 유형

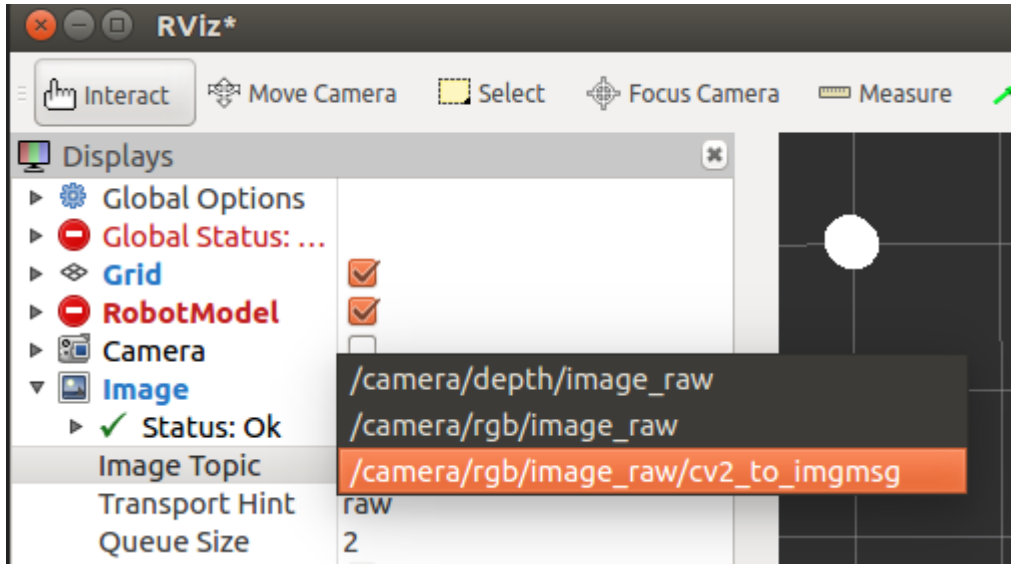
```
43 namespace sensor_msgs
44 {
45   namespace image_encodings
46   {
47     const std::string RGB8 = "rgb8";
48     const std::string RGBA8 = "rgba8";
49     const std::string RGB16 = "rgb16";
50     const std::string RGBA16 = "rgba16";
51     const std::string BGR8 = "bgr8";
52     const std::string BGRA8 = "bgra8";
53     const std::string BGR16 = "bgr16";
54     const std::string BGRA16 = "bgra16";
55     const std::string MONO8="mono8";
56     const std::string MONO16="mono16";
57
58     // OpenCV CvMat types
59     const std::string TYPE_8UC1="8UC1";
60     const std::string TYPE_8UC2="8UC2";
61     const std::string TYPE_8UC3="8UC3";
62     const std::string TYPE_8UC4="8UC4";
63     const std::string TYPE_8SC1="8SC1";
64     const std::string TYPE_8SC2="8SC2";
65     const std::string TYPE_8SC3="8SC3";
66     const std::string TYPE_8SC4="8SC4";
67     const std::string TYPE_16UC1="16UC1";
68     const std::string TYPE_16UC2="16UC2";
69     const std::string TYPE_16UC3="16UC3";
70     const std::string TYPE_16UC4="16UC4";
```

컬러 이미지

깊이 이미지

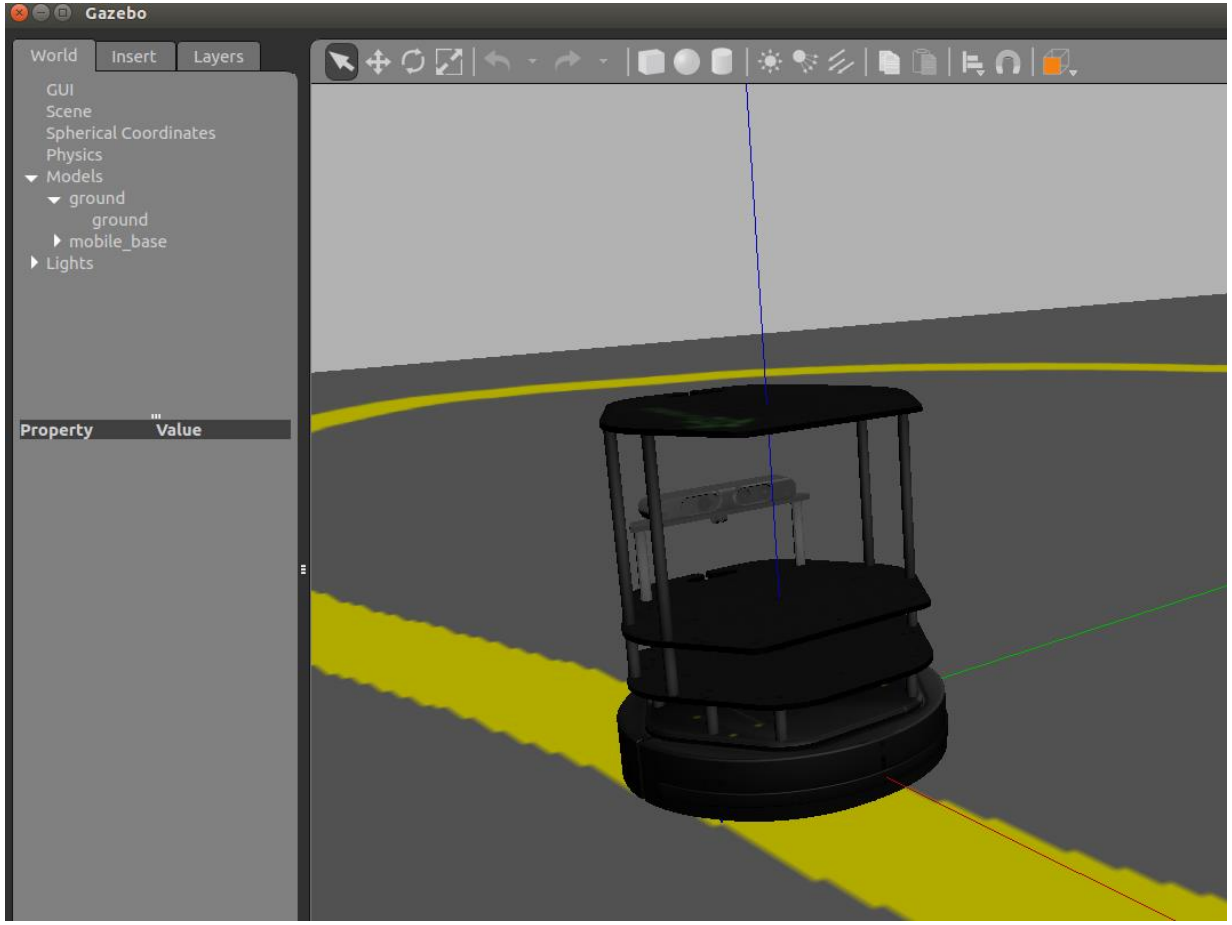
실행 결과

- rviz로 확인
  - 입력 메시지: /camera/rgb/image_raw
  - 출력 메시지: /camera/rgb/image_raw/cv2_to_imgmsg

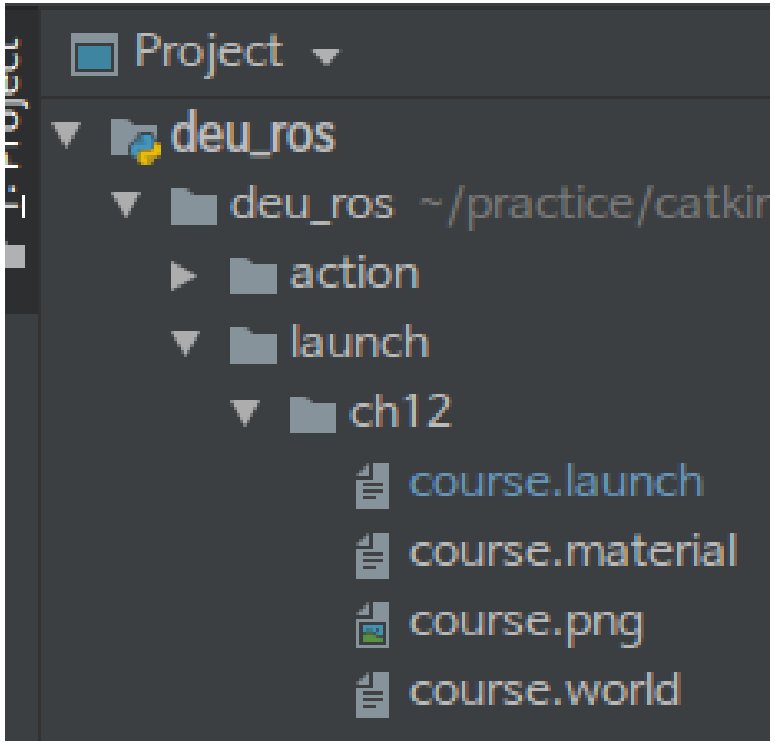


가제보 가상 세계 구현

- 노란 색 선 따라가기를 구현할 가제보 가상 세계



launch 파일 관련 구조



course.launch

```
1  <launch>
2    <env name="GAZEBO_RESOURCE_PATH" value="$(find deu_ros)/launch" />  <!-- followbot -->
3
4    <include file="$(find gazebo_ros)/launch/empty_world.launch">
5      <arg name="use_sim_time" value="true"/>
6      <arg name="debug" value="false"/>
7      <arg name="world_name" value="$(find deu_ros)/launch/ch12/course.world"/>  <!-- followbot -->
8    </include>
9
10   <include file="$(find turtlebot_gazebo)/launch/includes/kobuki.launch.xml">
11     <arg name="base" value="kobuki"/>
12     <arg name="stacks" value="hexagons"/>
13     <arg name="3d_sensor" value="asus_xtion_pro"/> <!-- kinect -->
14   </include>
```

```
16   <node pkg="robot_state_publisher" type="robot_state_publisher" name="robot_state_publisher">
17     <param name="publish_frequency" type="double" value="30.0" />
18   </node>
19
20   <!-- Fake laser -->
21   <node pkg="nodelet" type="nodelet" name="laserscan_nodelet_manager" args="manager"/>
22   <node pkg="nodelet" type="nodelet" name="depthimage_to_laserscan"
23         args="load depthimage_to_laserscan/DepthImageToLaserScanNodelet laserscan_nodelet_manager">
24     <param name="scan_height" value="10"/>
25     <param name="output_frame_id" value="/camera_depth_frame"/>
26     <param name="range_min" value="0.45"/>
27     <remap from="image" to="/camera/depth/image_raw"/>
28     <remap from="scan" to="/scan"/>
29   </node>
30 </launch>
```

empty_world.launch 일부 코드

```
33  <arg unless="$(arg debug)" name="script_type" value="gzserver"/>
34  <arg     if="$(arg debug)" name="script_type" value="debug"/>
35
36  <!-- start gazebo server-->
37  <group if="$(arg use_clock_frequency)">
38    <param name="gazebo/pub_clock_frequency" value="$(arg pub_clock_frequency)" />
39  </group>
40  <node name="gazebo" pkg="gazebo_ros" type="$(arg script_type)" respawn="$(arg respawn_gazebo)"
    output="$(arg output)"
41  args="$(arg command_arg1) $(arg command_arg2) $(arg command_arg3) -e $(arg physics) $(arg
    extra_gazebo_args) $(arg world_name)" />
42
43  <!-- start gazebo client -->
44  <group if="$(arg gui)">
45    <node name="gazebo_gui" pkg="gazebo_ros" type="gzclient" respawn="false" output="$(arg output)"
    args="$(arg command_arg3)"/>
46  </group>
```

course.world

```
<?xml version="1.0"?>
<sdf version="1.4">
  <world name="default">
    <scene>
      <ambient>0 0 0 1</ambient>
      <shadows>0</shadows>
      <grid>0</grid>
      <background>0.7 0.7 0.7 1</background>
    </scene>
    <!--
    <physics type="ode">
      <gravity>0 0 -9.8</gravity>
      <ode>
        <solver>
          <type>quick</type>
          <iters>10</iters>
          <sor>1.3</sor>
        </solver>
```

```
      <constraints>
        <cfm>0</cfm>
        <erp>0.1</erp>
        <contact_max_correcting_vel>10</contact_max_correcting_vel>
        <contact_surface_layer>0.001</contact_surface_layer>
      </constraints>
    </ode>
    <real_time_update_rate>1000</real_time_update_rate>
    <max_step_size>0.001</max_step_size>
    <real_time_factor>1</real_time_factor>
  </physics>
  -->
  <include>
    <uri>model://sun</uri>
  </include>
```

```
34    <model name="ground">
35      <pose>1 2.3 -.1 0 0 0</pose>
36      <static>1</static>
37      <link name="ground">
38        <collision name="ground_coll">
39          <geometry>
40            <box>
41              <size>10 10 .1</size>
42            </box>
43          </geometry>
44          <surface>
45            <contact>
46              <ode/>
47            </contact>
48          </surface>
49        </collision>
```

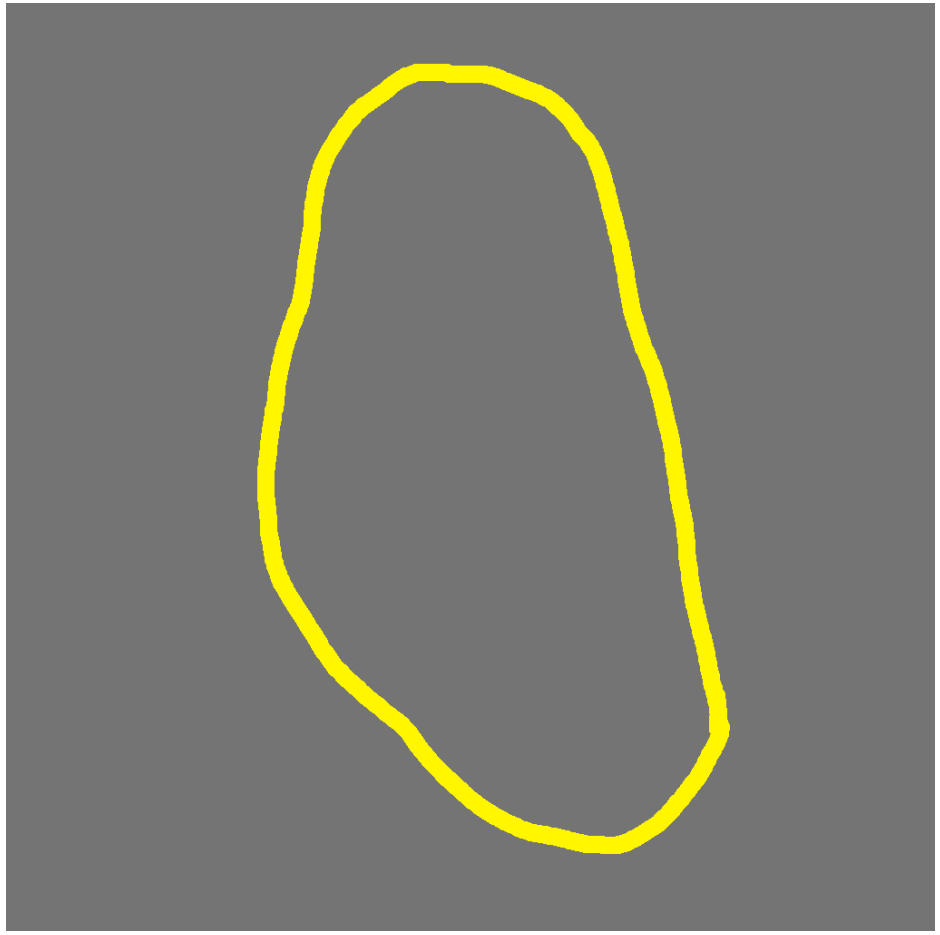
```
50        <visual name="ground_vis">
51          <geometry>
52            <box>
53              <size>10 10 .1</size>
54            </box>
55          </geometry>
56          <material>
57            <script>
58              <uri>file://course.material</uri>
59              <name>course</name>
60            </script>
61          </material>
62        </visual>
63      </link>
64    </model>
65  </world>
66 </sdf>
```

course.material

```
material course
{
  receive_shadows on
  technique
  {
    pass
    {
      ambient 0.5 0.5 0.5 1.0
      texture_unit
      {
        texture course.png
      }
    }
  }
}
```

course.png

- 해상도: 1024 x 1024

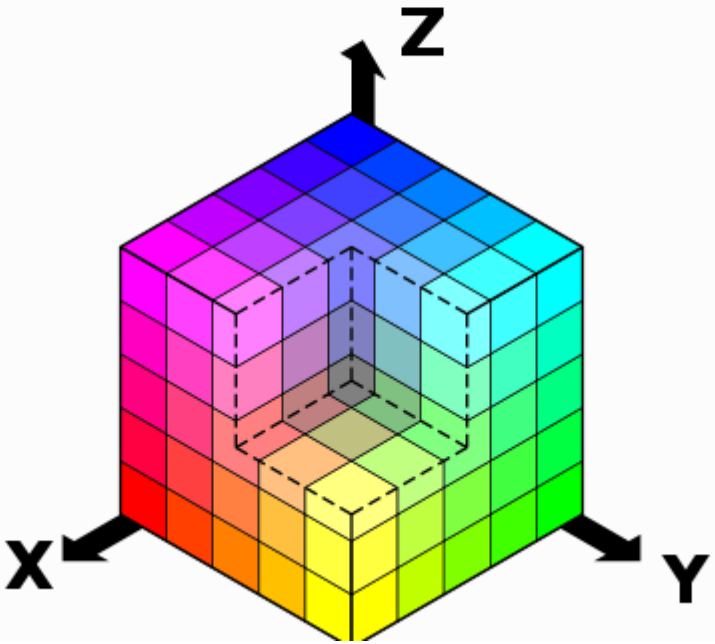


RGB vs. HSV

- URL: https://opencv-python.readthedocs.io/en/latest/doc/08.imageProcessing/imageProcessing.html

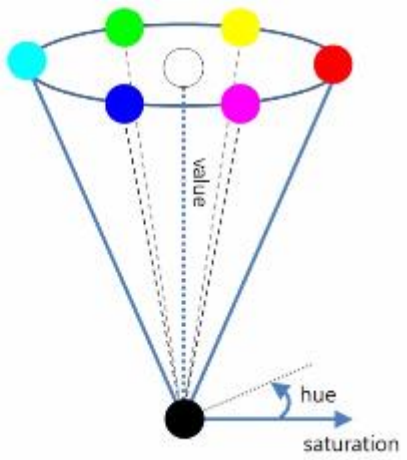
RGB Color-space

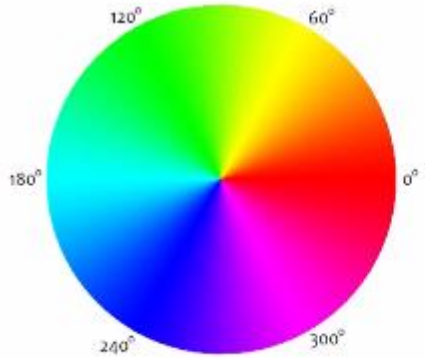
RGB 모델은 빛의 삼원색인 빨간색, 초록색, 파란색을 기본 색으로 사용을 합니다. 정육면체 모델 형태로 표현할 수 있습니다.



HSV Color-space

이미지 처리에서 가장 많이 사용되는 형태의 Color 모델입니다. 하나의 모델에서 색과 채도, 명도를 모두 알 수 있습니다. 원뿔 형태의 모델로 표현이 됩니다.





cv2.cvtColor()

- URL: https://docs.opencv.org/4.1.0/d8/d01/group__imgproc__color__conversions.html#ga397ae87e1288a81d2363b61574eb8cab
- dst = cv2.cvtColor(src, code[, dst[, dstCn]])

Parameters

src	input image: 8-bit unsigned, 16-bit unsigned (CV_16UC...), or single-precision floating-point.
dst	output image of the same size and depth as src.
code	color space conversion code (see **ColorConversionCodes**).
dstCn	number of channels in the destination image; if the parameter is 0, the number of the channels is derived automatically from src and code.

- Color Conversion Codes
  - 헤더: #include <**opencv2/imgproc.hpp**> → /usr/include/opencv2/imgproc 폴더 참조
  - cv2.BGR2HSV, cv2.BGR2GRAY 등

```
>>> cv2.COLOR_BG
COLOR_BGR2BGR555
COLOR_BGR2BGR565
COLOR_BGR2BGRA
COLOR_BGR2GRAY
COLOR_BGR2HLS
COLOR_BGR2HLS_FULL
COLOR_BGR2HSV
COLOR_BGR2HSV_FULL
COLOR_BGR2LAB
COLOR_BGR2LUV
```

cv2.inRange()

- URL: https://docs.opencv.org/4.1.0/d2/de8/group__core__array.html#ga48af0ab51e36436c5d04340e036ce981
- dst = cv.inRange(src, lowerb, upperb[, dst])

Parameters

src	first input array.
lowerb	inclusive lower boundary array or a scalar.
upperb	inclusive upper boundary array or a scalar.
dst	output array of the same size as src and CV_8U type.

Checks if array elements lie between the elements of two other arrays.

The function checks the range as follows:

- For every element of a single-channel input array:

$$\texttt{dst}(I) = \texttt{lowerb}(I)_0 \leq \texttt{src}(I)_0 \leq \texttt{upperb}(I)_0$$

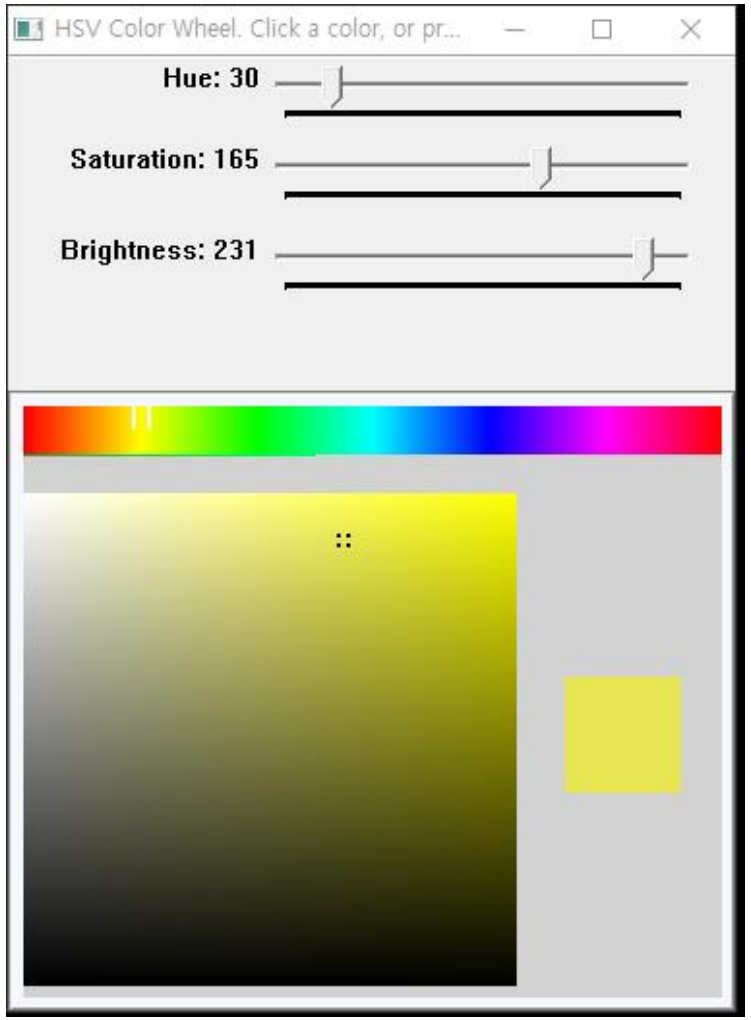
- For two-channel arrays:

$$\texttt{dst}(I) = \texttt{lowerb}(I)_0 \leq \texttt{src}(I)_0 \leq \texttt{upperb}(I)_0 \land \texttt{lowerb}(I)_1 \leq \texttt{src}(I)_1 \leq \texttt{upperb}(I)_1$$

- and so forth.

HSV 영상의 노란 색 정보 추출

- HSV에서 Hue는 색상 정보로 [0, 179] 구간의 값을 사용
- Saturation은 채도로 [0, 255] 구간의 값을 사용
- Value는 명도로 [0, 255]구간의 값을 사용



follower_color_filter2.py

```
1  #!/usr/bin/env python
2
3  import rospy
4  import cv_bridge
5  from sensor_msgs.msg import Image
6  import cv2
7  import numpy
8
9
10 class Follower:
11     def __init__(self):
12         self.bridge = cv_bridge.CvBridge()
13         # cv2.namedWindow("window", 1)
14         self.image_sub = rospy.Subscriber('camera/rgb/image_raw', Image, self.image_callback)
15         self.hsv_pub = rospy.Publisher('camera/rgb/image_raw/hsv', Image, queue_size=1)  # LJM: added
16         self.mask_pub = rospy.Publisher('camera/rgb/image_raw/mask', Image, queue_size=1)  # LJM: added
```

목적: OpenCV 함수를 사용하여 처리를 하기 위해
RGB 영상을 HSV 영상으로 변환하고,
노란 색 부분만 추출한 영상 생성

변환된 hsv 영상과 mask 영상을 /camera/rgb/image_raw/{hsv, mask} 토픽으로 발행하기 위한 발행자 추가

```
18      def image_callback(self, msg):
19          image = self.bridge.imgmsg_to_cv2(msg)
20
21          hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
22          lower_yellow = numpy.array([25, 20, 50])   # H: [25,35], S: [20,255], V: [50,255]
23          upper_yellow = numpy.array([35, 255, 255])
24
25          mask = cv2.inRange(hsv, lower_yellow, upper_yellow)
26          # cv2.imshow("window", mask)
27          # cv2.imshow("hsv", hsv)
28          # cv2.waitKey(3)
29
30          mask_image = self.bridge.cv2_to_imgmsg(mask)
31          hsv_image = self.bridge.cv2_to_imgmsg(hsv)
32          self.mask_pub.publish(mask_image)
33          self.hsv_pub.publish(hsv_image)
34
35
36  rospy.init_node('follower_color_filter2')
37  follower = Follower()
38  rospy.spin()
```

BGR 형식을 HSV 형식으로 변환

교재와 다른 부분

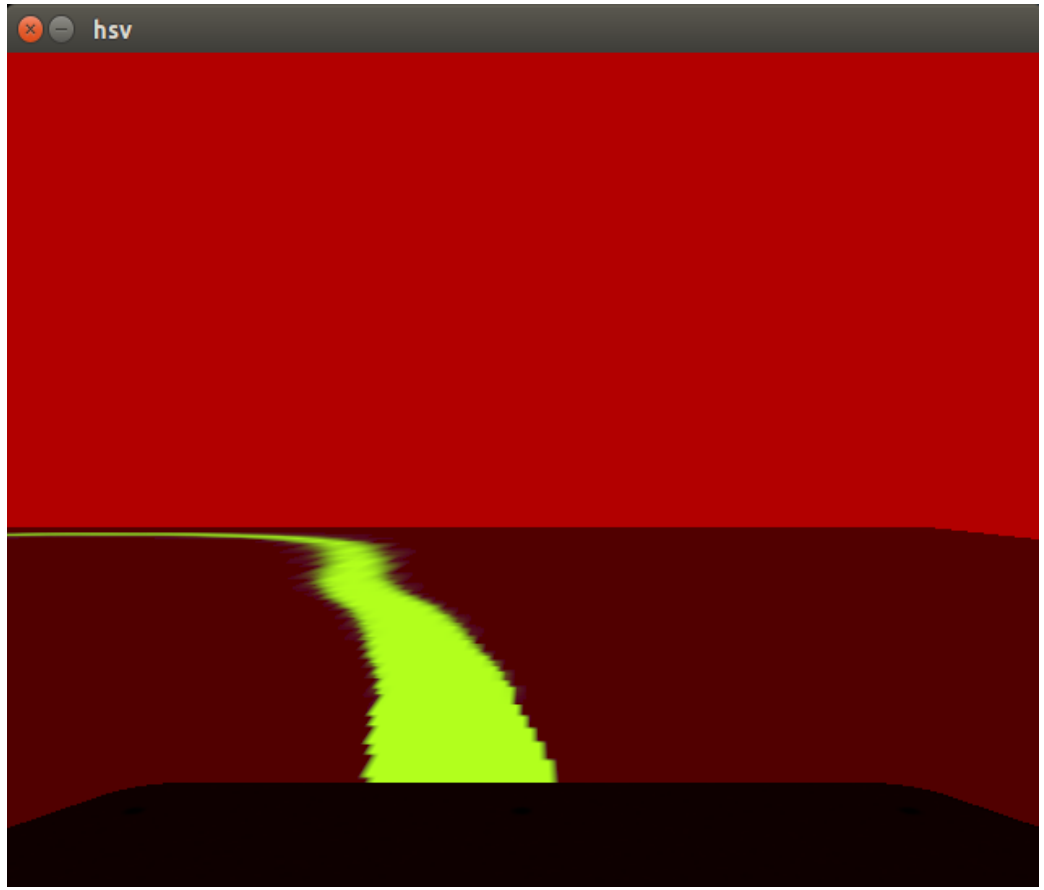
hsv 영상에서 노란 색만 추출한 mask 영상 생성

mask와 hsv 영상을 ROS 영상 형식으로 변환

변환된 maks와 hsv ROS 영상을 발행

실행 결과

- 실행 환경: Ubuntu 14.04 + ROS Indigo



- 실행 환경: Ubuntu 16.04 + ROS Kinetic

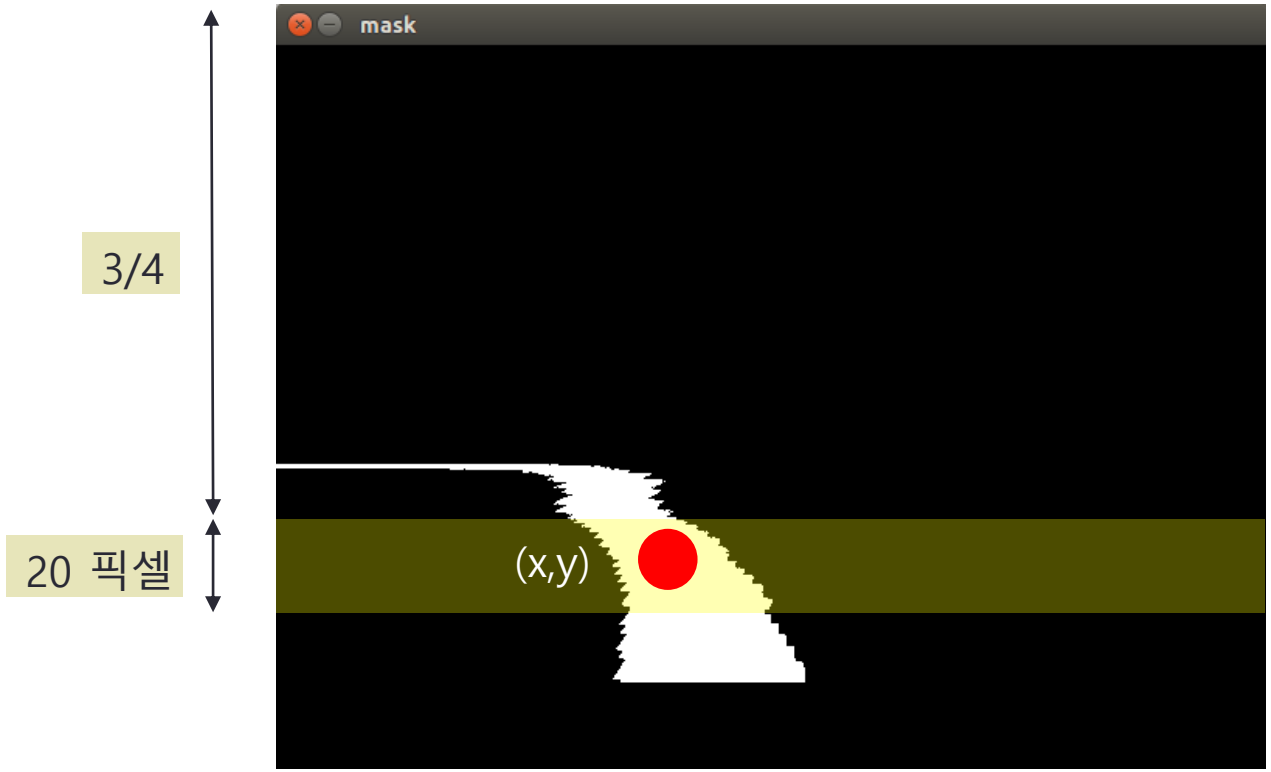
/camera/rgb/image_raw

/camera/rgb/image_raw/hsv

/camera/rgb/image_raw/mask

추적할 선의 중심 찾기 개념

1) mask에서 노란 색 영역만 남겨 두고 모두 데이터 지운다.
2) 노란 색 영역 중 흰 색 부분의 중심점을 구한다.



cv2.moments()

- URL: https://docs.opencv.org/3.0-beta/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html, https://docs.opencv.org/4.1.0/d8/d23/classcv_1_1Moments.html
- retval = cv2.moments(array[, binaryImage])

Parameters:
- **array** – Raster image (single-channel, 8-bit or floating-point 2D array) or an array ($1 \times N$ or $N \times 1$) of 2D points (`Point` or `Point2f`).
- **binaryImage** – If it is true, all non-zero image pixels are treated as 1's. The parameter is used for images only.
- **moments** – Output moments.

```
// spatial moments
double  m00, m10, m01, m20, m11, m02, m30, m21, m12, m03;
// central moments
double  mu20, mu11, mu02, mu30, mu21, mu12, mu03;
// central normalized moments
double  nu20, nu11, nu02, nu30, nu21, nu12, nu03;
```

In case of a raster image, the spatial moments $\texttt{Moments::m}_{ji}$ are computed as:

$$\texttt{m}_{ji} = \sum_{x,y} \left(\texttt{array}(x,y) \cdot x^j \cdot y^i \right)$$

The central moments $\texttt{Moments::mu}_{ji}$ are computed as:

$$\texttt{mu}_{ji} = \sum_{x,y} \left(\texttt{array}(x,y) \cdot (x - \bar{x})^j \cdot (y - \bar{y})^i \right)$$

where $(\bar{x}, \bar{y})$ is the mass center:

$$\bar{x} = \frac{\texttt{m}_{10}}{\texttt{m}_{00}}, \; \bar{y} = \frac{\texttt{m}_{01}}{\texttt{m}_{00}}$$

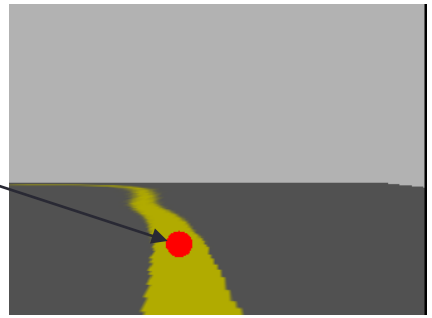
```
M = cv2.moments(mask)
cx = int( M['m10'] / M['m00'] )
cy = int( M['m01'] / M['m00'] )
```

follower_line_finder2.py

```
1  #!/usr/bin/env python
2
3  import cv2
4  import cv_bridge
5  import numpy
6  import rospy
7  from sensor_msgs.msg import Image
8
9
10 class Follower:
11     def __init__(self):
12         self.bridge = cv_bridge.CvBridge()
13         # cv2.namedWindow("window", 1)
14         self.image_sub = rospy.Subscriber('camera/rgb/image_raw', Image, self.image_callback)
15         self.image_pub = rospy.Publisher(S'camera/rgb/image_raw/line_finder', Image, queue_size=1)  # LJM: added
16
17     def image_callback(self, msg):
18         image = self.bridge.imgmsg_to_cv2(msg, desired_encoding='bgr8')  # 'pass-through'
19         hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
20         lower_yellow = numpy.array([25, 20, 50])
21         upper_yellow = numpy.array([35, 255, 255])
22         mask = cv2.inRange(hsv, lower_yellow, upper_yellow)
```

```
24      h, w, d = image.shape
25      search_top = 3 * h / 4
26      search_bot = search_top + 20
27      mask[0:search_top, 0:w] = 0
28      mask[search_bot:h, 0:w] = 0
29
30      M = cv2.moments(mask)
31      if M['m00'] > 0:
32          cx = int(M['m10'] / M['m00'])
33          cy = int(M['m01'] / M['m00'])
34          cv2.circle(image, (cx, cy), 20, (0, 0, 255), -1)
35
36      # cv2.imshow("window", image)  # BLOCKED
37      # cv2.imwrite('line_finder.jpg', image)  # OK
38      line_image = self.bridge.cv2_to_imgmsg(image)
39      self.image_pub.publish(line_image)
40      # cv2.waitKey(3)
41
42
43  rospy.init_node('follower')
44  follower = Follower()
45  rospy.spin()
```

영상의 height, width, depth를 구한다.
탐색 창의 height 시작 지점: ¾ * h
탐색 창의 height 종료 지점: ¾ * h + 20 (픽셀)



실습 과제: 이 mask를 rviz에서 볼 수 있도록 코드를 수정하시오.

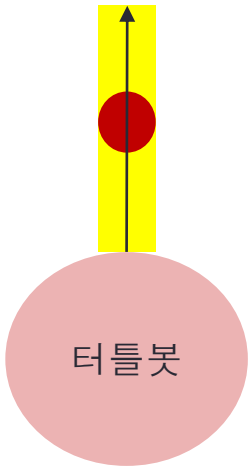
선 따라가기

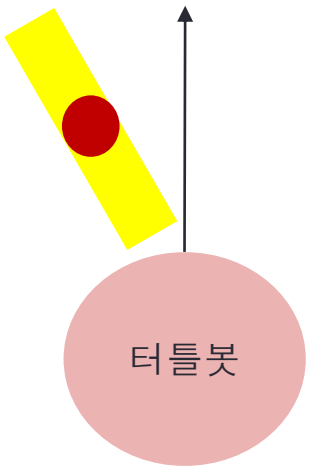
By the right hand rule, the yaw component of orientation increases as the child frame rotates counter-clockwise, and for geographic poses, yaw is zero when pointing east.

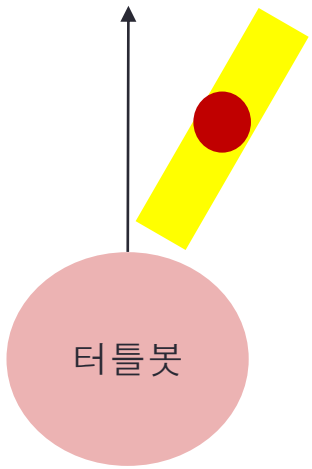
- 선속도 0.2
- x 좌표 w/2

cx w/2

w/2 cx







err = cx – w/2	0	음수	양수
각속도	0	적은 양수	적은 음수

참고: P-제어기: 이동 방향으로 오차가 있을 경우 그 오차보다 적은 값을 주어 점차 오차가 0으로 수렴하게 제어
(P: proportional)

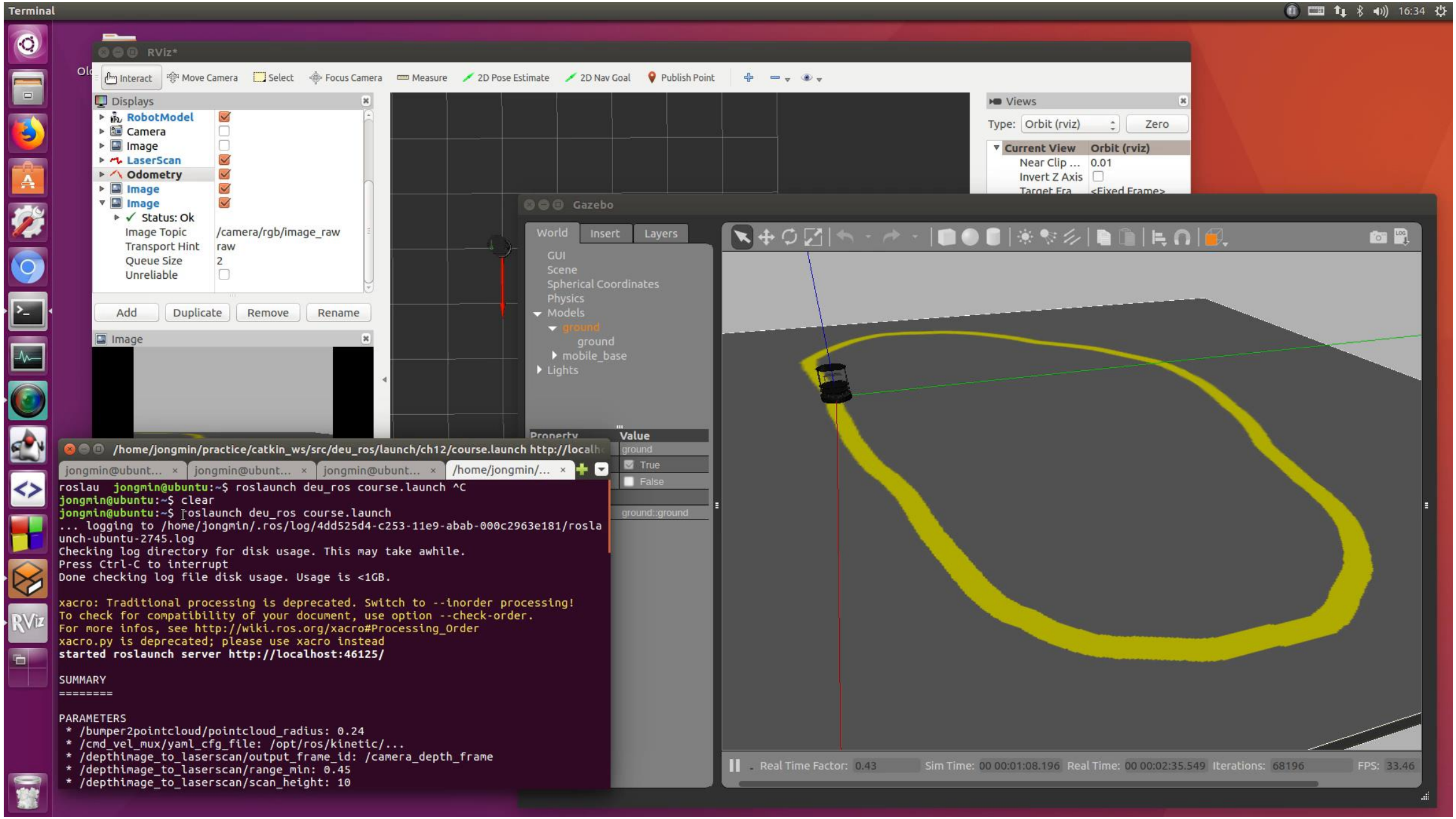
follower_p2.py

```
1  #!/usr/bin/env python
2
3  import cv2
4  import numpy
5
6  import cv_bridge
7  import rospy
8  from geometry_msgs.msg import Twist
9  from sensor_msgs.msg import Image
10
11
12 class Follower:
13     def __init__(self):
14         self.bridge = cv_bridge.CvBridge()
15         # cv2.namedWindow("window", 1)
16         self.image_sub = rospy.Subscriber('camera/rgb/image_raw', Image, self.image_callback)
17         self.cmd_vel_pub = rospy.Publisher('cmd_vel_mux/input/teleop', Twist, queue_size=1)
18         self.p2_image_pub = rospy.Publisher('camera/rgb/image_raw/p2', Image, queue_size=1)
19         self.twist = Twist()
20         self.count = 0
```

```
22  def image_callback(self, msg):
23      image = self.bridge.imgmsg_to_cv2(msg, desired_encoding='bgr8')
24      hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
25      lower_yellow = numpy.array([25, 20, 50])
26      upper_yellow = numpy.array([35, 255, 255])
27
28      mask = cv2.inRange(hsv, lower_yellow, upper_yellow)
29
30      h, w, d = image.shape
31      search_top = 3 * h / 4
32      search_bot = 3 * h / 4 + 20
33      mask[0:search_top, 0:w] = 0
34      mask[search_bot:h, 0:w] = 0
35      M = cv2.moments(mask)
36      if M['m00'] > 0:
37          cx = int(M['m10'] / M['m00'])
38          cy = int(M['m01'] / M['m00'])
39          cv2.circle(image, (cx, cy), 20, (0, 0, 255), -1)
```

P-제어기

```
41              err = cx - w / 2
42              self.twist.linear.x = 1.0  # default: 0.2, OK: 0.7
43              self.twist.angular.z = -float(err) / 300  # 400: 0.1, 300: 0.15, 250, 0.2
44              self.cmd_vel_pub.publish(self.twist)
45
46          # cv2.imshow("window", image)
47          # cv2.imshow("mask", mask)
48          # cv2.imwrite('image.jpg', image)  # LJM - for test
49          # cv2.imwrite('mask.jpg', mask)  # LJM - for test
50
51          image_msg = self.bridge.cv2_to_imgmsg(image, 'bgr8')
52          self.p2_image_pub.publish(image_msg)
53          rospy.loginfo('count = %d', self.count)  # LJM - for test
54          self.count += 1
55          # cv2.waitKey(3)
56
57
58  rospy.init_node('follower')
59  follower = Follower()
60  rospy.spin()
```


Terminal
RViz*
Interact
Move Camera
Select
Focus Camera
Measure
2D Pose Estimate
2D Nav Goal
Publish Point
Displays
RobotModel
Camera
Image
LaserScan
Odometry
Image
Image
Status: Ok
Image Topic /camera/rgb/image_raw
Transport Hint raw
Queue Size 2
Unreliable
Add
Duplicate
Remove
Rename
Views
Type: Orbit (rviz)
Zero
Current View Orbit (rviz)
Near Clip ... 0.01
Invert Z Axis
Gazebo
World
Insert
Layers
GUI
Scene
Spherical Coordinates
Physics
Models
ground
ground
mobile_base
Lights
Property
Value
ground
True
False
ground::ground
/home/jongmin/practice/catkin_ws/src/deu_ros/launch/ch12/course.launch http://localh
jongmin@ubunt...
jongmin@ubunt...
jongmin@ubunt...
/home/jongmin/...
roslau jongmin@ubuntu:~$ roslaunch deu_ros course.launch ^C
jongmin@ubuntu:~$ clear
jongmin@ubuntu:~$ roslaunch deu_ros course.launch
... logging to /home/jongmin/.ros/log/4dd525d4-c253-11e9-abab-000c2963e181/roslaunch-ubuntu-2745.log
Checking log directory for disk usage. This may take awhile.
Press Ctrl-C to interrupt
Done checking log file disk usage. Usage is <1GB.
xacro: Traditional processing is deprecated. Switch to --inorder processing!
To check for compatibility of your document, use option --check-order.
For more infos, see http://wiki.ros.org/xacro#Processing_Order
xacro.py is deprecated; please use xacro instead
started roslaunch server http://localhost:46125/
SUMMARY
========
PARAMETERS
* /bumper2pointcloud/pointcloud_radius: 0.24
* /cmd_vel_mux/yaml_cfg_file: /opt/ros/kinetic/...
* /depthimage_to_laserscan/output_frame_id: /camera_depth_frame
* /depthimage_to_laserscan/range_min: 0.45
* /depthimage_to_laserscan/scan_height: 10
Real Time Factor: 0.43
Sim Time: 00 00:01:08.196
Real Time: 00 00:02:35.549
Iterations: 68196
FPS: 33.46

Aruco Markers 예