jsk_recognition

Caution: This document only covers implement official tutorial, bag of features, and partially cover the deep learning with own dataset and ssb_object_detector. For more functionality, visit the official jsk documentation site.

Last update: 04/04/2019

Created by: Kate. Student number: s2861506

https://jsk-docs.readthedocs.io/en/latest/jsk_recognition/doc/index.html

Setups, installation before going to the official tutorial

Install pip

sudo apt-get install python-pip

update pip to 9.0.3 or above

sudo pip install pip==9.0.3

Install jsk_recognition

Two ways to install, recommend use apt-get install instead of build from source. Because build from source may take 4-6 hours to complete.

Install from apt-get

sudo apt-get install ros-kinetic-jsk-pcl-ros

sudo apt-get install ros-kinetic-jsk-pcl-ros-utils

sudo apt-get install ros-kinetic-jsk-perception

sudo apt-get install ros-kinetic-jsk-recognition

sudo apt-get install ros-kinetic-jsk-recognition-msgs

sudo apt-get install ros-kinetic-jsk-recognition-utils

Build from source, only if the installed packages not working

cd ~/catkin_ws/src

git clone https://github.com/jsk-ros-pkg/jsk_recognition.git

Install dependency for all packages in the workspace

cd ~/catkin_ws

rosdep install --from-paths src --ignore-src -r -y

Notice: this can take some time depends on how many dependency needs to be installed

Build workspace

catkin build –continue-on-failure

If build from source, and cannot compile due to memory error, use following command

catkin build --continue-on-failure -p1 -j1

Buld from source can take up to 4-6 hours to complete

cd ~/catkin_ws

catkin build --continue-on-failure

. ~/catkin_ws/devel/setup.bash

Create new package

Create new package with any name you would like to have in the work space

catkin_create_pkg replace_here_with_the_name_you_prefer std_msgs rospy roscpp

. ~/catkin_ws/devel/setup.bash

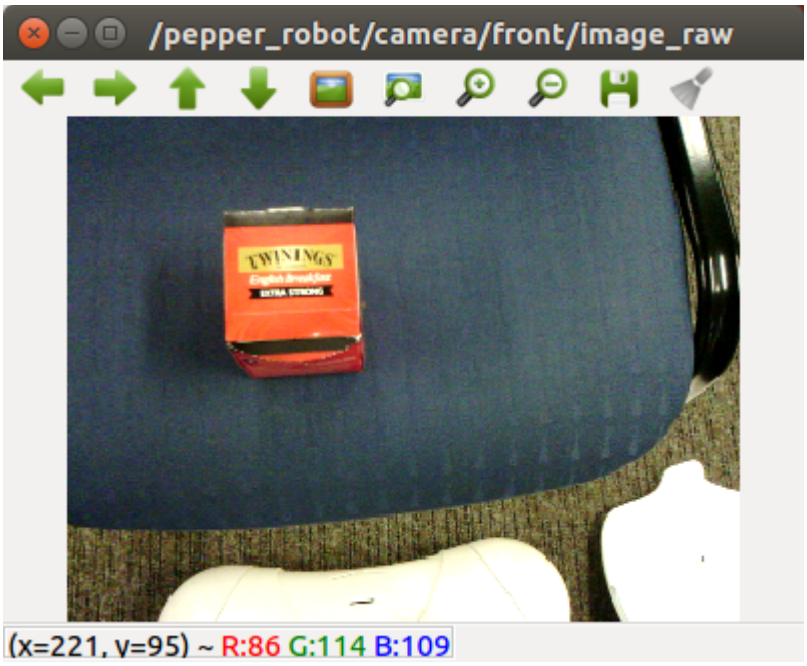
Launch pepper_bringup

roslaunch pepper_bringup pepper_full.launch networ_interface:=enp2s0 roscore_ip:=kate-iMac.local

JSK – Official Tutorial

https://jsk-docs.readthedocs.io/en/latest/jsk_recognition/doc/tutorials/index.html
Run the tutorial to see if all the required node can be run with rosrun command. If any missing, build from source. Following is the implementation with the tea boxes

run image_view from pepper's camera
rosrun image_view image_view image:=/pepper_robot/camera/front/image_raw



Step 1. Apply color filter

run hsv_color_filter
rosrun opencv_apps hsv_color_filter image:=/pepper_robot/camera/front/image_raw __name:=hsv_color_filter
rosrun image_view image_view image:=/hsv_color_filter/image
run rqt_reconfigure
rosrun rqt_reconfigure rqt_reconfigure
set values for hsv_color_filter to
h_limit_max 30
h_limit_min 341
s_limit_max 128
s_limit_min 256
v_limit_max 113
v_limit_min 256
Above value should be adjust depends on what type of background is using. The main purpose is to get object that you want become white, and surrounding are black
Optional: Create launch file in the package
roscd package_name_you_created
mkdir launch
touch apply_color_filter.launch
copy and paste following, implemented with pepper's camera from the original file https://jsk-docs.readthedocs.io/en/latest/jsk_recognition/doc/tutorials/find_object_with_color_filtering.html

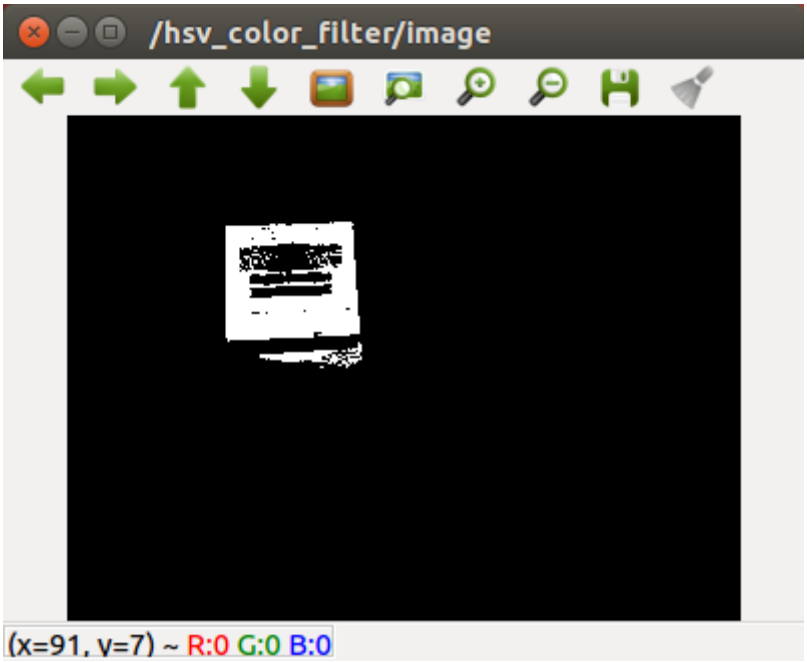
```
<launch>
```

```
<node name="hsv_color_filter"
      pkg="opencv_apps" type="hsv_color_filter">
  <remap from="image" to="/pepper_robot/camera/front/image_raw" />
  <rosparam>
    use_camera_info: false
    h_limit_max: 164
    h_limit_min: 360
    s_limit_max: 152
    s_limit_min: 256
    v_limit_max: 158
    v_limit_min: 256
  </rosparam>
</node>

<node name="image_view_color_filtering"
      pkg="image_view" type="image_view">
  <remap from="image" to="hsv_color_filter/image" />
</node>

<node name="rqt_reconfigure"
      pkg="rqt_reconfigure" type="rqt_reconfigure"
      args="hsv_color_filter">
</node>
```

Run the launch file

roslaunch package_name_you_created apply_color_filter.launch

Result



Step 2. Get masked image

see the stamp

rostopic echo /pepper_robot/camera/front/image_raw/header/stamp

rostopic echo /hsv_color_filter/image/header/stamp

apply mask image

rosrun jsk_perception apply_mask_image _clip:=false _approximate_sync:=false \~input:=/pepper_robot/camera/front/image_raw \~input/mask:=hsv_color_filter/image
rosrun image_view image_view image:=/JSK_NODELET_jsk_perception_apply_mask_image/output (Attention: if use launch file, the path name will be different! Check with rostopic list to see the correct path. Here is without using the launch file)

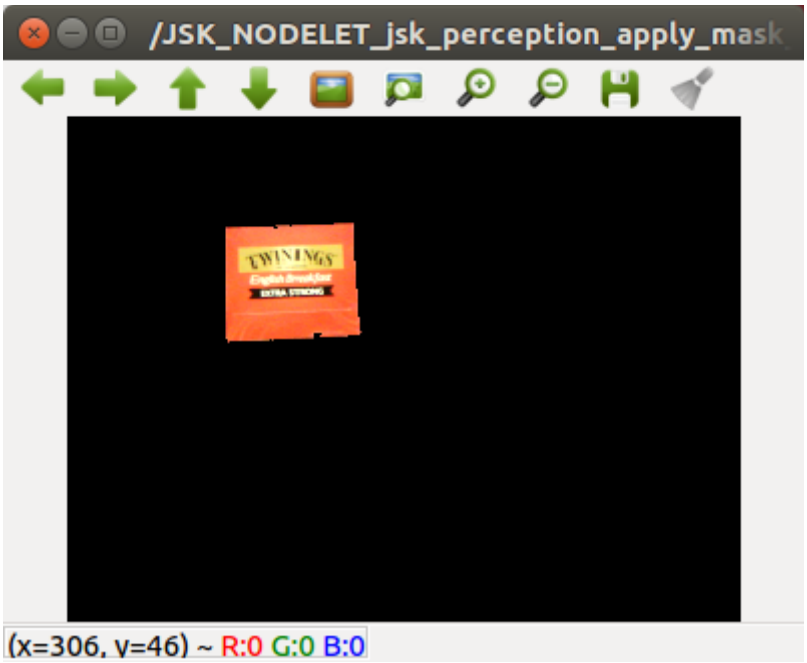
bound object mask image

rosrun jsk_perception bounding_object_mask_image \~input:=hsv_color_filter/image

apply mask image better

rosrun jsk_perception apply_mask_image \~input:=/pepper_robot/camera/front/image_raw \~input/mask:=/JSK_NODELET_jsk_perception_bounding_object_mask_image/(Attention: if use launch file, the path name will be different! Check with rostopic list to see the correct path. Here is without using the launch file) output _clip:=false

Result



Step3. Save image

roscd package_you_created
mkdir data
cd data
mkdir collected_images
cd collected_images
mkdir object_name
cd object_name

Save the image

rosrun image_view image_saver image:=/JSK_NODELET_jsk_perception_apply_mask_image/ (Attention: if use launch file, the path name will be different! Check with rostopic list to see the correct path) output _save_all_image:=false _filename_format:=side(Attention: need to change name according to side, back, front view)%04i.%s __name:=image_saver
open a new terminal
rosservice call /image_saver/save
When change object view, just need to adjust the rqt_reconfigure values.

JSK - Bag of features(Bof) for object recognition

Bag of features for object recognition
https://jsk-docs.readthedocs.io/en/latest/jsk_recognition/doc/jsk_perception/bof_object_recognition.html

Step 1. create descriptors dataset

Make sure have more than one objects image data under the collected_images folder
rosrun jsk_perception create_sift_dataset.py $(rospack find package_you_created)/data/collected_images

Step 2. create bag of features

rosrun jsk_perception create_bof_dataset.py extract_bof collected_images_sift_feature.pkl.gz -O collected_images_bof.pkl.gz

Step 3. create bag of features histogram

rosrun jsk_perception create_bof_dataset.py extract_bof_hist collected_images_sift_feature.pkl.gz collected_images_bof.pkl.gz -O collected_images_bof_hist.pkl.gz

Step 4. train classifier use sklearn_classifier_train see

rosrun jsk_perception sklearn_classifier_trainer.py collected_images_bof_hist.pkl.gz -O collected_images_clf.pkl.gz

mask image to label

https://jsk-docs.readthedocs.io/en/latest/jsk_recognition/doc/jsk_perception/nodes/mask_image_to_label.html
roscd package_you_created
cd launch
touch get_masked_imageto_label.launch
copy and paste following, implement the original launch file from - https://github.com/jsk-ros-pkg/jsk_recognition/blob/master/jsk_perception/sample/sample_mask_image_to_label.launch

```
<launch>
  <arg name="gui" default="true" />

  <node name="apply_mask_image"
        pkg="jsk_perception" type="apply_mask_image">
    <remap from="~input" to="/pepper_robot/camera/front/image_raw" />
    <remap from="~input/mask" to="hsv_color_filter/image" />
    <rosparam>
      clip: false
    </rosparam>
  </node>

  <node name="mask_image_to_label"
        pkg="jsk_perception" type="mask_image_to_label.py">
    <remap from="~input" to="/apply_mask_image/output/mask" />
  </node>

  <group if="$(arg gui)">
    <node name="image_view0"
          pkg="image_view" type="image_view">
      <remap from="image" to="apply_mask_image/output" />
    </node>
    <node name="image_view1"
```

```
        pkg="image_view" type="image_view">
    <remap from="image" to="mask_image_to_label/output" />
   </node>
  </group>
</launch>
```

Bof object recognition

touch bof_object_recognition.launch

copy and paste following, implment the orginal launch file from - https://github.com/jsk-ros-pkg/jsk_recognition/blob/master/jsk_perception/sample/sample_bof_object_recognition.launch

```
<launch>

  <arg name="gui" default="true" />

  <node name="colorize_labels"
        pkg="jsk_perception" type="colorize_labels">
    <remap from="~input" to="/mask_image_to_label/output" />
  </node>

  <node name="imagesift"
        pkg="imagesift" type="imagesift">
    <remap from="image" to="/pepper_robot/camera/front/image_raw" />
    <remap from="Feature0D" to="~output" />
  </node>

  <node name="feature0d_to_image"
        pkg="posedetection_msgs" type="feature0d_to_image">
    <remap from="image" to="/pepper_robot/camera/front/image_raw" />
    <remap from="Feature0D" to="imagesift/output" />
  </node>

  <node name="bof_histogram_extractor"
        pkg="jsk_perception" type="bof_histogram_extractor.py">
    <remap from="~input" to="imagesift/output" />
    <remap from="~input/label" to="/mask_image_to_label/output" />
    <param name="~bof_data" value="$(find
pepper_jsk_image_recognition)/data/collected_images_bof.pkl.gz" />
    <rosparam>
      approximate_sync: true
      slop: 1.0
    </rosparam>
  </node>

  <node name="sklearn_classifier"
        pkg="jsk_perception" type="sklearn_classifier.py">
    <remap from="~input" to="bof_histogram_extractor/output" />
    <param name="~clf_path" value="$(find
pepper_jsk_image_recognition)/data/collected_images_clf.pkl.gz" />
  </node>

  <group if="$(arg gui)">
    <node name="rqt_gui"
```

```
        pkg="rqt_gui" type="rqt_gui"
        args="--perspective-file $(find
jsk_perception)/sample/config/sample_bof_object_recognition.perspective" />
  </group>

</launch>
```

Launch the launch files

roslaunch package_you_created get_masked_imageto_label.launch
roslaunch package_you_created bof_object_recognition.launch
The ClassificationResult will be /sklearn_classifier/output

Draw classfication result using the Bof

https://github.com/jsk-ros-pkg/jsk_recognition/blob/master/jsk_perception/sample/sample_draw_classification_result.launch
rosrun jsk_perception draw_classification_result.py ~input:=/sklearn_classifier/output ~input/image:=/pepper_robot/camera/front/image_raw
rosrun image_view image_view image:=/draw_classification_result/output

This only display the probability of the objects. Beacuase under the collected_images only have two objects data, therefore the probalibity is 50 ~ 60%

JSK - Deep learning with your own dataset

https://jsk-recognition.readthedocs.io/en/latest/deep_learning_with_image_dataset/overview.html

Annotate images
https://jsk-recognition.readthedocs.io/en/latest/deep_learning_with_image_dataset/annotate_images_with_labelme.html
training: folder
- images: folder
- labels.txt
create dataset for semantic segmentation
Before followintg the tutorial, first you can choose to
1. git clone https://github.com/wkentaro/labelme.git whole repo to one of the folder, and copy the labelme2voc.py file from labelme/example/semantic_segmentation to the training folder
or
2. create a python file under the training folder and copy the code from https://github.com/wkentaro/labelme/blob/master/examples/semantic_segmentation/labelme2voc.py
follow the tutorial to create dataset
https://github.com/wkentaro/labelme/tree/master/examples/semantic_segmentation
python labelme2voc.py inputdata_folder(exsits) outputdata_folder(not exist) --labels labels.txt

Train neural network
https://jsk-recognition.readthedocs.io/en/latest/deep_learning_with_image_dataset/overview.html
sudo pip install opencv-python
sudo pip install chainer-mask-rcnn
https://pypi.org/project/chainer-mask-rcnn/
https://github.com/wkentaro/chainer-mask-rcnn
sudo pip install --upgrade cryptography
sudo python -m easy_install --upgrade pyOpenSSL

semantic segmentation
--gou, -1 cpu mode, 0 gpu mode
rosrun jsk_perception train_fcn.py --train_dataset_dir $(rospack find pepper_jsk_image_recognition)/data/teabox_dataset(folder for the image file)/train --val_dataset_dir $(rospack find package_you_created)/data/teabox_dataset/test --out_dir $(rospack find package_you_created)/data/teabox_dataset/trainned_data --gpu -1
No reaction called on the mac
instance segmentation
rosrun jsk_perception train_mask_rcnn.py --train_dataset_dir $(rospack find pepper_jsk_image_recognition)/data/teabox_dataset_instance/train --val_dataset_dir $(rospack find pepper_jsk_image_recognition)/data/teabox_dataset_instance/test --out_dir $(rospack find pepper_jsk_image_recognition)/data/teabox_dataset_instance/trainned_data --gpu -1
Getting error
/usr/local/lib/python2.7/dist-packages/chainercv/utils/bbox/non_maximum_suppression.py:81: RuntimeWarning: invalid value encountered in true_divide
  iou = area / (bbox_area[i] + bbox_area[selec] - area)

/usr/local/lib/python2.7/dist-packages/chainercv/utils/bbox/non_maximum_suppression.py:82:
RuntimeWarning: invalid value encountered in greater_equal
  if (iou >= thresh).any():

ssb_object_detector (GPU mode)
https://jsk-docs.readthedocs.io/en/latest/jsk_recognition/doc/jsk_perception/nodes/ssd_object_detector.html
create yml file contains labeling name
vim label_name.yml
the format of yml should be:
- lable name
- lable name
git clone https://github.com/yuyu2172/image-labelling-tool
cd image-labelling-tool
sudo pip install -e .
python ../image-labelling-tool/flask_app.py --image_dir $(rospack find pepper_jsk_image_recognition)/data/experiment/train/twinings_english_breakfast_teabox_extra_strong/ --label_names $(rospack find pepper_jsk_image_recognition)/data/experiment/train/twinings_english_breakfast_teabox_extra_strong/label_names.yml --file_ext jpg