auviitkgp / **datasheets**

Image Transport with ROS Messages and OpenCV

groverjeenu edited this page on 20 Jul 2014 · 3 revisions

Image Transport with ROS Messages and OpenCV

Overview

ROS passes around images in its own sensor_msgs/Image message format, but many users will want to use images in association with OpenCV. This can be done with help of a ROS library CVBridge. CVBridge provides the required interface between ROS and OpenCV.

CvBridge can be found in the cv_bridge package in the vision_opencv stack.

OpenCv uses cv::Mat format for storing images. A node which listens to a ROS topic will get image in a sensor_msgs/Image format,then it will be converted to Opencv format cv :: Mat and now we can use all sorts of functions in OpenCV to draw on image,threshold it etc.

Similarly while publishing a image via a ROS message ,the OpenCV format is changed to sensor_msgs/Image format.

Details

Now, a package that subscribes image from a topic ,converts it to OpenCv,again converts it to sensor_msgs/Image an publishes it, must have the following dependencies..

sensor_msgs
cv_bridge
std_msgs
image_transport

Structure of CVBridge library

Now to be able to use the funtions of library CVbridge,we should know its structure.. Under namespace CVBridege there is a class named CVImage which has a public variable cv :: Mat image which can be accessed.

It has two pointers for the same class CvImagePtr and CvImageConstPtr

```
namespace cv_bridge {

class CvImage  {  public:  std_msgs::Header header;

std::string encoding;

cv::Mat image;

};

typedef boost::shared_ptr<CvImage> CvImagePtr;

typedef boost::shared_ptr<CvImage const> CvImageConstPtr;

}
```

CONVERTING ROS Image msg to OpenCV Image

There are 2 ways for doing this..

1. if want to modify data obtained from ROS message, we need to make a copy of it.We make use of toCvCopy() here.

Clone this wiki locally

https://github.com/auviit

```
CvImagePtr toCvCopy(const sensor_msgs::ImageConstPtr& source,
const std::string& encoding = std::string());
```

```
CvImagePtr toCvCopy(const sensor_msgs::Image& source,
const std::string& encoding = std::string());
```

Here the second argument in toCvCopy() is image encoding type.It has been declared by default,so even if we miss it,no need to worry.
But generally we use **"bgr8"** encoding in our work.It solves most of our problems.

bgr8: CV_8UC3, color image with blue-green-red color order

Now if want to process the image obtained ... it goes like this..

```
cv_bridge::CvImageConstPtr cv_ptr;   cv_ptr = cv::toCvCopy(msg,CV_8UC3);
```

Now access as **cv_ptr->image**

2. If we donot want to change data, then we need not copy it,we can just share it.

But here we must use **Const** pointers to avoid modifying the original data.

here we use **toCvShare()** ...

Syntax goes like this..

```
CvImageConstPtr toCvShare(const sensor_msgs::ImageConstPtr& source,
const std::string& encoding = std::string());
```

Rest everything is same as above.

**Converting OpenCV images to ROS image messages **

To convert a CvImage into a ROS image message, use the toImageMsg() member function

```
sensor_msgs::ImagePtr toImageMsg() const;
```

Now this image can be published aa shown below

```
img_publisher_variable.publish(cv_ptr->toImageMsg);
```

Note

Here we must note that if the cv_img ptr is not the one that has been taken from any other ROS messages i.e it has been created by us,
then before publishing we must specify its Header and encoding as these are two member variables of class CVImage.

For more details go to site
http://wiki.ros.org/cv_bridge/Tutorials/UsingCvBridgeToConvertBetweenROSImagesAndOpenCVImages

To see how it is used and implemented in AUV's work, go to...
https://github.com/iit-kgp-auv-team/kraken_3.0/blob/master/vision_stack/camera/videoread/src/videoread.cpp