Sperical Camera system setup

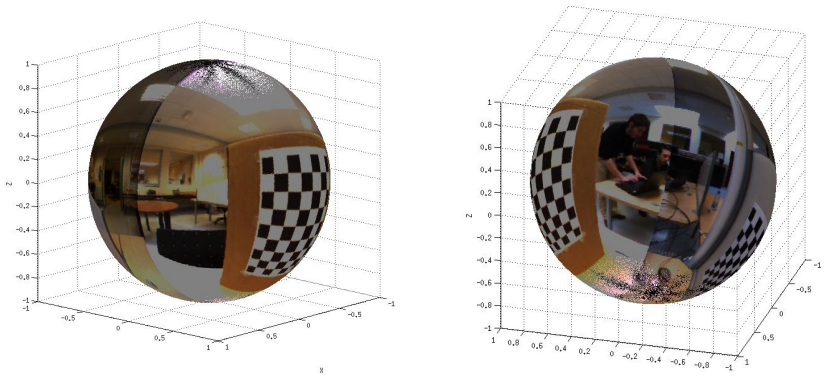
Jeremie Deray

October 8, 2014

1 Introduction

1.1 Objectives

To install and configure the spherical camera designed during Jeremie Deray MSc Thesis. The system is composed of two (2) fisheye cameras and produces a complete spherical image.



(a) One view (b) Another view

Figure 1: Two hemispherical images stitched together forming one single image lying on the full unit sphere. This alignment uses ΔR

1.2 Prerequisites

Robot Operating System (ROS) - version Hydro
Information and installation tutorial can be found at :

<http://wiki.ros.org/hydro/Installation> 1

Catkin - A low-level build system macros and infrastructure for ROS. A tutorial on how to set-up a catkin environment can be found at :

<http://wiki.ros.org/catkin/Tutorials/create_a_workspace> 2

2 Camera & Driver

2.1 Material

- 2 * IDS-Imaging uEye camera model UI-3240CP 3
- 2 * fisheye lens FUJINON FE185C046HA-1 4

It is better (main option is the literature) to neglect the distance between both optical centers. Assuming this, it is necessary to physically minimize it as much as possible. It means that the cameras have to be as close as possible one to the other. Different configurations have been tested and it appeared that the best one is when the cameras are back-to-back. Despite the fact that the distance could be reduced if cameras where side-by-side, the misalignment in the final spherical image is more important and this is actually what have to be minimized.
The cameras have to be back-to-back (relative rotation 180 over y axis) as showed below.

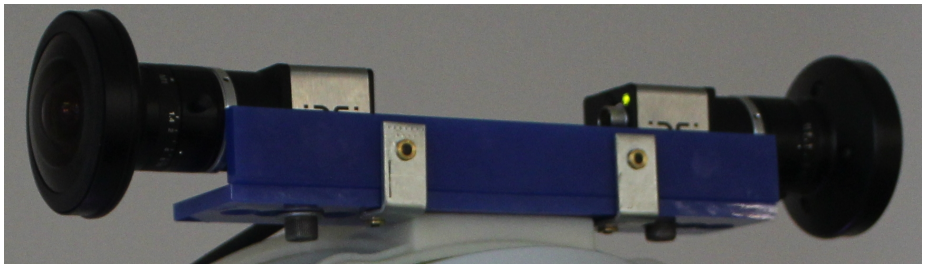


Figure 2: The proposed omnidirectional camera prototype

The use of angled cables should allow for a minimum distance between the camera 5.

2.2 Drivers

The drivers can be downloaded at the following address

<http://en.ids-imaging.com/download-ueye.html> 6

Of course it depends on your system, make sure to download the correct one.

The drivers installation on an Linux machine is as follows :

Copy in a folder of your choice the driver zip file you just downloaded and unzip it. Open a terminal within the uncompressed folder :

- Run the installation script
  ~/myfolder $ sh ./ueyesdk-setup-4.*-*-*.gz.run
  (replace asterisk by the correct values, depend on your system)
- Launch the deamon
  ~/myfolder $ sudo /etc/init.d/ueyeusbdrc start
- Verify that everything works
  ~/myfolder $ ueyedemo
- If you want to stop / restart the deamon
  ~/myfolder $ sudo /etc/init.d/ueyeusbdrc stop
  ~/myfolder $ sudo /etc/init.d/ueyeusbdrc restart

In addition to the driver and ueye demo, two (2) other tools have been installed:

- A camera's ID manager - allows to manually define camera's ID
  ~/myfolder $ ueyesetid
- A camera's parameters manager - allows to set different parameters of the camera
  ~/myfolder $ ueyecameramanager

3 ROS interface

Now that the driver is installed it requires a wrapper with the ROS environment. The wrapper is of the form of a ROS package and so requires a working catkin environment.

3.1 Camera node

The package must be downloaded within the catkin workspace.
The download and installation on a Linux machine is as follows:

- Download the package within catkin workspace
  $ cd ~/catkin_ws
  ~/catkin_ws/src $ clone https://bitbucket.org/kmhallen/ueye 7

- Compile it
  ~/catkin_ws $ catkin_make
- Verify that it works launching the camera node
  ~/catkin_ws $ rosrun ueye camera

  in another terminal

  $ rosrun image_view image_view image:='???'

The same manner for the stereo configuration :

- Launch the stereo camera node
  ~/catkin_ws $ rosrun ueye stereo

  in another terminal

  $ rosrun image_view image_view image:=left/image_raw

  in another terminal

  $ rosrun image_view image_view image:=right/image_raw

Some of the camera's parameters can be changed live using dynamic reconfigure:

$ rosrun rqt_reconfigure rqt_reconfigure
go to section 'camera'

3.2 Rosbag

Rosbag is a very useful tool provided by ROS allowing to record any data from an experiment so that they can be 'replay' anytime later. It is particularly interesting when for instance an algorithm has to be tuned, refined or debugged whereas no robot are immediately available.
To record images from the stereo node in a rosbag on a Linux machine is as follows:

~/mybagfolder $ rosbag record left/image_raw right/image_raw

The rosbag have a max size of 2Go. In often happens that the limit is quickly reached due to the size of the raw image and a high camera frame rate. It is possible to automatically split the recording in several rosbags as follows:

~/mybagfolder $ rosbag record –split –size=1024 /left/image_raw /right/image_raw

where '–size=1024' stands for the size in Mo of the rosbags.

3.3 Image extraction

One may wants to extract images from a recorded rosbag (or from a live experiment) in order to exploit them later.
An important feature for the camera setup which has not been mentioned so far is to provide synchronized images. Indeed considering the setup embedded on a mobile robot, images from each camera must be taken at the same 'moment' in order to do not introduce more displacement (distance) from an optical center to the other and keep the calibration valid. ROS provides a tool to extract images from a given topic, however using it twice for two different image topics do not ensure the synchronization of their extraction. Moreover, the ueye package uses a 'soft' synchronization (software synchronization) which induces a slight time gap between both images.
In order to extract images in a synchronous manner a package has been developed for this purpose.

Its download and installation on a Linux machine is as follows:

- Download within catkin workspace
  ~/catkin_ws/src $ git clone https://github.com/artivis/ros_img_extractor.git
  8
- Compile
  ~/catkin_ws $ catkin_make
- Launch the extraction node
  ~/myimagefolder $ roslaunch ros_img_extractor extract_img_sync.launch topic1:=left/image topic2:=right/image

The above command will extract images in a synchronous manner and save them within the path where the roslaunch command has been executed (in above example '~/myimagefolder/'). It is possible to define a custom path for each topic (see launch files for options details).

4 Calibration

The spherical camera setup requires a double calibration. Indeed it is necessary to calibrate both the intrinsic parameters (camera matrix) and the extrinsic parameters (relative cameras pose). Both calibration can be done in a single process.
Once the cameras have been mounted on a rigid support (e.g. Fig.2), the system has to be placed between two (2) calibration patterns (chessboards). The relative pose of a pattern to another is not important, however it must NOT CHANGE during the whole image acquisition.
The process is then to acquire images of the pattern (with both cameras) for different poses of the system.

To do so:

- Launch image_view from a given path
  ~/imgcam1 $ rosrun image_view image_view image:=left/image_raw
- Launch a second image_view from a different path
  ~/imgcam2 $ rosrun image_view image_view image:=right/image_raw
- Position the system properly so that one pattern appears in each image
- Take a snapshot by right-clicking on each image_view windows
- Repeat the two (2) previous step until enough images have been acquired
- Save the calibration

For the calibration to be successful and accurate enough, it is necessary to have around twelve (12) image pairs. However it is preferable to take up to sixteen (16) image pairs.

4.1 Intrinsic calibration

The intrinsic calibration of both fisheye cameras is handle by the Mei Toolbox. It can be download at:

<http://www-sop.inria.fr/icare/personnel/Christopher.Mei/ChristopherMeiPhDStudentToolbox.html> 9

Moreover, the calibration process is well detailed in the above link.

Tips:

* When selecting '3 : dioptric (fisheye)' for the mirror type, it often happens that the distortion parameters estimation mess up the whole calibration. To deactivate it you have to enter the following command in Matlab right before executing the calibration (before clicking 'Calibration'):

paramEst.est_dist = [0;0;0;0;0]

* Image names have to follow a sequence, for instance for four (4) images:
image1.jpg, image2.jpg, image3.jpg, image4.jpg

The following sequence will fail:
image1.jpg, image2.jpg, image4.jpg, image5.jpg

* If an image has to be discarded (poor quality, pattern too much distorted etc) the rest of the image sequence has to be renamed. !!! If an image has to be discarded in one set (one camera) do not forget to discard it in the second set as well! Images pairs need a one-one correspondence.

4.2 Extrinsic calibration

The extrinsic calibration is achieved directly from Mei toolbox calibration results. It is done using another Matlab package which can be download at:

<https://github.com/artivis/SpheriCam> 10

An example script shows how to use this above toolbox:

SpheriCam_Calib.m

4.3 Images stitching

Once the full calibration is know, images from both cameras can be stitched together as a unique panoramic image.
An example of the stitching is showed in the Matlab script:

SpheriCam_image.m

5 Links

1. ROS install
   <http://wiki.ros.org/hydro/Installation>
2. Catkin install
   <http://wiki.ros.org/catkin/Tutorials/create_a_workspace>
3. IDS-Imaging Camera
   <http://en.ids-imaging.com/store/ui-3240cp.html?___store=sv_gmbh_en&_
   __from_store=sv_gmbh_fr>
4. Fujifilm Fisheye Lens
   <http://www.fujifilmusa.com/products/optical_devices/security/fish-eye/
   5-mp/fe185c046ha-1/>
5. Angled cables
   <http://www.angledcables.com/a-to-microb-cables.html>
6. uEye driver
   <http://en.ids-imaging.com/download-ueye.html>
7. uEye ROS package
   <https://bitbucket.org/kmhallen/ueye>
8. Image extraction node
   <https://github.com/artivis/ros_img_extractor.git>
9. Mei Toolbox
   <http://www-sop.inria.fr/icare/personnel/Christopher.Mei/ChristopherMeiPhDStudentToolbox.
   html>
10. SpheriCam Toolbox
    <https://github.com/artivis/SpheriCam>