MSFT-Visual Servoing Practical 2
Intensity Based Visual Servoing

Masters in Computer Vision

UNIVERSITE DE BOURGOGNE

Centre Universitaire Condorcet - UB, Le Creusot

21 December 2017

Team Members:
Mohit Kumar Ahuja
GopiKrishna Erabati
Dousai Nayee Muddin Khan

Supervisor:
Dr. Nathan Crombez

Task: Intensity Based Visual Servoing

What we do:
We tried to compute the velocities required to merge the current image to a desired image using control law.

How we do:
There was an error using the function "vpImageio". So, we used OpenCV function "Imread" to read the image. For Intensity-Based Visual Servoing (InBVS), we followed the following sequence:

1. We gave the current pose of the camera (cMo) as well as the desired pose of the camera (cdMo). The only difference in current and desired pose is the rotation. The translation has not been changed.

2. We have to set the Robot Position using robot.setPosition(cMo).

3. We loaded the image and started acquiring the images using getImage() function of "vpImageSimulator". But before getting image, we have to set the position of the camera by giving the homogeneous matrix (cMo) setCameraPosition function.

4. We acquired images from both current and desired pose and then computed the difference between both using imageDifference function of "VpImageTools" object. And then displaying all the three images as shown in figure 1.

Figure 1: (a) Desired Image, (b) Current Image and (c) Difference of both images before VS.

5. We created two objects named vpFeatLumCur for current image and vpFeatLumDes for desired image, these objects are made from class "vpFeatureLuminance". We initialized both using init() function and once giving the parameters of current image and secondly giving the parameters of the desired image.

6. Then we set the camera parameters and built two features, one for current image and one for desired image using buildfrom(). The visual feature for current image is computed every time but the visual feature of desired image is only computed once.

The initialization is done till now, Now in the while loop, we followed the following sequence:

1. set.CameraPosition(cMo) -- Setting the camera position.
2. getImage(I,cam) -- Getting the image

3. display -- Display frames of current image
4. flush -- Flush the display of current image
5. imageDifference -- Computing the difference
6. display -- Display frames for differenced image
7. flush -- Flush the display of differenced image
8. buildFrom -- Build visual feature from current image everytime
9. error -- compute the error for control Law
10. interaction -- Computed interaction matrix for visual features
11. robovel -- Robot velocities are computed by control law

{The control law is used to compute the velocities of the robot to go to a desired pose. Velocities = -Lambda*$(\text{InteractionMatrix})^T * \text{error}$}

12. setvelocities -- velocities are sent to the robot.
13. Getposition -- getting the current position of robot
14. Setposition -- setting the position of robot
15. euclideanNorm -- To compute L2 Norm of error.

We made an if statement in the while loop to terminate as soon as the robot reaches the desired location. The result can be seen in figure 2.

Figure 2: (a) Desired Image, (b) Current Image and (c) Difference of both images after VS.

Why we do:
This task was to do Intensity Based Visual Servoing by computing velocities for the robot using Control Law and how VISP library can be used to compute velocities using control law which can be used as a basis for visual servoing. The results can be seen in Fig. 2.

Conclusion:

We have done Intensity Based Visual Servoing by computing velocities for the robot using Control Law by VISP library. We had seen great advantages of this library as we can track and detect using a single line of code and the data structures defined for holding data are well defined and the class and methods used in the library are well documented with many examples. We found it's easy to use this library for visual servoing in the context of our tasks.

The code is found here.
The video can be found here.