Intelligent Robotics

Assignment 1

Implementing a multi-scale saliency model to simulate visual attention

Kevin Klein

H00233447

MSc Software Engineering

1. Software Description

The project was realised using the iCub Simulator as a visual feedback environment. I used C++ and OpenCV for processing the images while using the YARP network as a middleware.

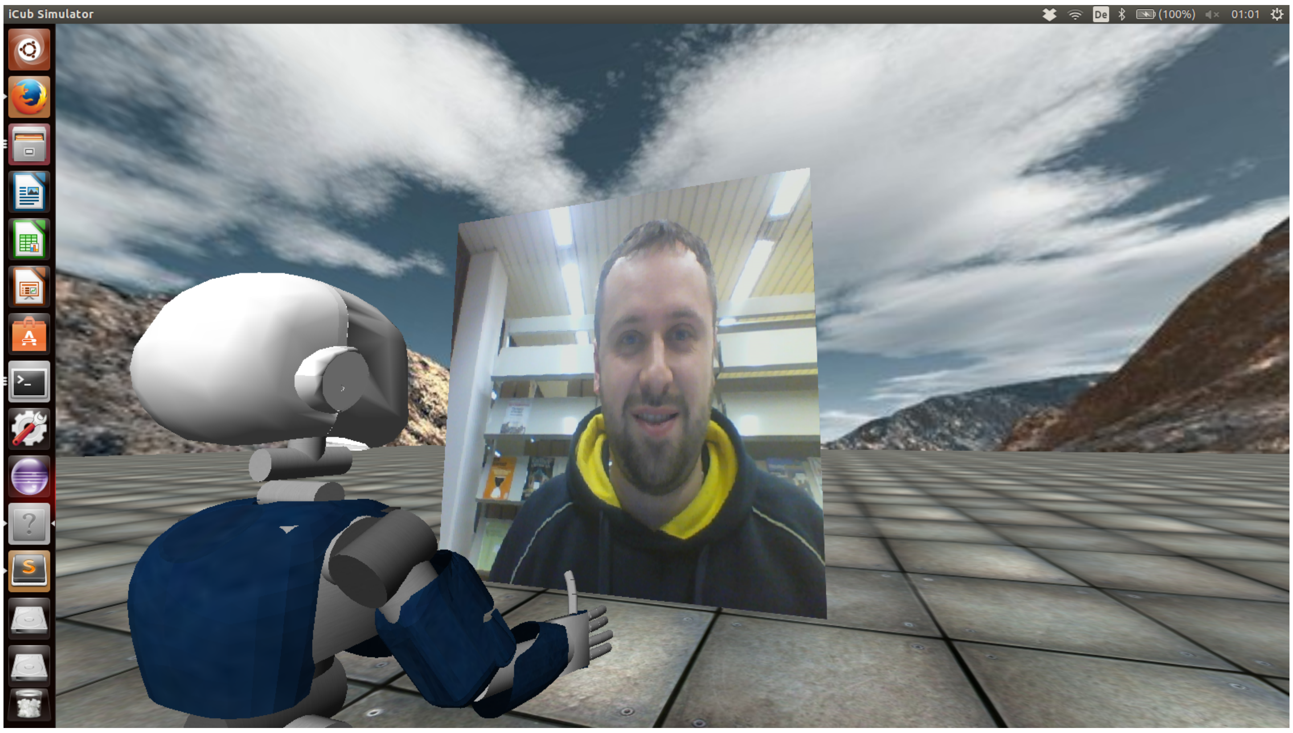
The yarpserver and the iCub Simulator have to be executed in two different terminals:

yarpserver

iCub_SIM

Then, the C++ program can be compiled using the makefiles provided. It is important that the *CMakeLists.txt* file is included in the same location, as it enables compilation of YARP and openCV libraries, with the file being specifically created for compiling in Linux. Also note that the 'screen' object has to be set to 'on' in the *iCub_parts_activation.ini* file.

Once the program is run, you should see yourself on the screen of the iCub Simulator, with the robot staring right at you.



Img 1: starting the program

The Software starts with initiating YARP ports to connect the iCub Simulator with the local web camera. From this point on, various linear filters are processed inside a loop which will display additional webcam windows with some fancy real time visual effects.

The robot should be able to follow circles and colours you show him on the screen and move his head accordingly.

2. Rationale of linear filters

2.1. Hough Circle Transform

This linear filter takes parameters such as centre position and radius. The function applies a Gaussian Blur to the image in order to reduce the noise, then applies the Hough Circle transformation to the blurred image.

2.2. Sobel Derivatives

Sobel Derivatives are used to detect the edges of an object in order to better identify the specific shape that distinguishes the object from the rest of the image. A change in gradient is often a strong indication for a change in the actual image. To compute this, both horizontal and vertical changes are calculated by convolving the image with the kernel of the 2D-array/matrix. The results are then summed up to approximate the current gradient point.

The Sobel and Sharr functions OpenCV provides are then used, with Sharr seeming to deliver more accuracy than Sobel. As a result, the edges of objects appear white while the rest of the imgae, apart from rather thin lines in between appear very dark.

3. Problems and issues

While implementing the knowledge build up in the previous exercises of this course and playing around with OpenCV for image processing was quite straightforward, there were some obstacles and pitfalls involved in the development of the project.

To carry out the implementation, I installed YARP on my Linux distribution Ubuntu 14.04, with cmake and OpenCV already being included in the factory settings of the OS. Setting up the CmakeLists.txt file in order to compile YARP and OpenCV with cmake was a larger obstacle than initially expected.

After testing streaming with some static image files (.jpg, jpeg) at first, the biggest challenge was to get the content of my webcam to be streamed to the iCub simulator in order to carry out real time image processing. The online documentation for realising this with YARP however is fairly thin.

What could be improved is the circle detection, which sometimes loses focus and strongly depends on the manual configuration in the code in terms of size and distance to the webcam. Of course, the natural environment surrounding the PC and user in terms of brightness in the rooms and background as well as the webcam itself influence the performance of the filters.

In the end, I could implement all required tasks as well as go beyond the call of duty with webcam streaming and real time image processing.

4. Paper summary

The paper “A Model of Saliency-Based Visual Attention for Rapid Scene Analysis” by Laurent Itti, Christof Koch and Ernst Niebur describes a biologically inspired saliency model for computational efficient image analysis and target detection.

It's architecture provides 5 different levels for bottom-up, feed-forward processing for visual search, starting with a visual input which is passed on to various linear filters, feature maps and conspicuity maps, to then finally be passed on to a so called saliency master map.

The input is determined as a digital, static colour image with a 640x480 resolution. An early set of visual features is provided by three different linear filters which result in producing a number of 42 feature maps all together with 6 intensity maps for creating Gaussian pyramids, 12 colour maps using red, green and blue channels and 24 orientation maps. After normalising the values in the map to a fixed range and finding the map's local maxima of activity, the most active location is compared with the average value, with it's deviation distinguishing how strong the most active location stands out of the map. The architecture represents the theory that filtering methods, which differ from each other, contribute independently to a optimum result while methods with similar attributes rather compete strongly for saliency. These feature maps are then later combined to 3 conspicuity maps, which are then normalised and summed up for the final input value to the saliency map. The saliency map itself has the propose of providing saliency at every level in the visual field.

The most salient image location, represented by the maximum of the saliency map, should attract the so called focus of attention. At this point the implementation of the biologically inspired artificial neural network comes into play in order to specify the most active location in the image. A artificial neural network usually consists of neurons to model the human brain, in this case the brain of a primate, whereas in this model the weights (inputs) are not used to cause the neurons to 'fire' like it would usually be the case. Instead, these neurons are intended to be used as pure integrators and eventually trigger a corresponding threshold to fire.

Due to the feed-forward architecture and rapidly selected combination of various different maps, results in terms of optimisation as well as performance seem to be promising, in particular in regard to visual noise when tested with artificial images and real images. It's main strength lies in good recognition of various shapes, textures and colours as well as its computational performance, which is mainly owed to the parallel implementation of the model. On the other hand, conjunctions of feature without undertaking further modifications.

Overall, the model represents a simple but efficient solution for visual attention based on saliency, although the results strongly depends on the implemented feature types.