Xiao Sun

# DEVELOPMENT OF A VISION SYSTEM AND BASIC DRAWING WITH NAO ROBOT

Information Technology
2016

# FOREWORD

This is my final undergraduate thesis in Information Technology at Vaasan Ammattikorkeakoulu, Vaasa University of Applied Sciences.

I would like to express heartfelt thanks to my thesis supervisor, Dr. Yang Liu, for his patient and professional guidance, not only for this thesis but also throughout my years of study. Before he became my supervisor, his supervising list was already full. He had no obligation and no working time to supervise me, but he still agreed to supervise me in his precious rest time. During my thesis time, he helped me both academically and spiritually. When I had no ideas or have a wrong direction of thinking, he always helped me by providing many constructive suggestions. When I was making no progress and feeling disappointed, he told me not to worry and not to be impatient. Nothing worthwhile is not difficult. What I need to do is just keep the right direction and hold on. He always inspires me and give me confidence.

As an Embedded System student, Dr. Yang Liu is my specialized courses teacher. I am thankful for his profound knowledge and effective advice, so I have the chance to study the NAO robot and learned much about scientific research approaches.

Furthermore, I would like thank other professors in the IT department of VAMK: Dr. Menani Smail, Mr. Jukka Matila, Mr. Santiago Chavez, Mr. Jani Ahvonen, Dr. Chao Gao, Mr. Antti Virtanen, Dr. Ghodrat Moghdampour and other staff who have guided me and kindly provided me with help. I am thankful for their wisdom and knowledge, which have given me a lot of help during my three years of studies at VAMK.

Finally, I want to express my appreciation to my parents for their love, encouragement, and support throughout all my life, and all my friends at the Botnia RoboCup laboratory, including Zhou Ziye, Liao Ke, Li Xiuyang, and Zhou Yang. I am thankful for their company, encouragement, and help. I wish you all good luck.

Sun Xiao

Xiangyang, Hubei, China

26.03.2016

VAASAN AMMATTIKORKEAKOULU
UNIVERSITY OF APPLIED SCIENCES
Information Technology

## ABSTRACT

| | |
|---|---|
| Author | Xiao Sun |
| Title | Development of a Vision System and Basic Drawing with NAO Robot |
| Year | 2016 |
| Language | English |
| Pages | 61 |
| Name of Supervisor | Yang Liu |

This thesis introduces the development of the vision system and basic drawing of a NAO robot. In this thesis three kinds of detection systems are discussed in order to test the algorithm for the vision system. Each detection corresponds to a kind of drawing, but all of the drawings use the same method, which is introduced in this thesis in detail.

This thesis can be divided into two main sections, the vision module section and the motion module section. The vision module is used for detection. As mentioned above, three kinds of detection systems are included namely contours detection, character detection, and face detection. In the detection of contours and characters, the robot will recognize the contours and characters that it detects. The motion module is based on a mathematic model that is built by two 2D coordinate systems. The two 2D coordinate systems have the same x-axis, but the unit length in the physical significance are not necessarily the same.

The main carrier of this thesis is the fifth generation NAO robot developed with Python language under the Windows platform. OpenCV and Tessract are used as the libraries of image processing and character recognition. In the character recognition, a technique named optical character recognition (OCR) is also used. Matlab is used to do some mathematical calculations and generate functions.

In this application the NAO robot can detect shapes, characters, and faces successfully and can draw corresponding graphs. However, due to the unstable hardware, the drawing cannot be very accurate, and the robot has to learn how to draw classes of shapes. In further study, the robot is expected to calculate the equations of motion by itself.

Key Words NAO robot, OpenCV, Image Process, OCR

# CONTENTS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| **CV** | **Computer Vision** |
| **3D** | **Three-Dimensional** |
| **AI** | **Artificial Intelligence** |
| **RGB** | **Red, Green and Blue** |
| **2D** | **Two-Dimensional** |
| **CMYK** | **Cyan, Magenta, Yellow and Black** |
| **OpenCV** | **Open Source Computer Vision** |
| **BSD** | **Berkeley Software Distribution license** |
| **PIL** | **Python Imaging Library** |
| **MV** | **Machine Vision** |
| **API** | **Application Programming Interface** |
| **HCI** | **Human – Computer Interface** |
| **HMI** | **Human – Machine Interface** |
| **SfM** | **Structure from Motion** |
| **OCR** | **Optical Character Recognition** |
| **IDLE** | **Intergrated Development and Learning Environment** |

# LIST OF FIGURES AND TABLES

# 1 INTRODUCTION

## 1.1 Purpose

This thesis introduces the development of the vision system and the motion system for the NAO robot. The whole thesis can be divided into three steps. The first step is about detection and recognition. The robot will get the information from its cameras and through a series of analyses and processing in order to recognize what it saw. The second step is about the basic drawing. After the recognition in the first part, the drawing function will be called. Different recognitions correspond to different drawings, but the different drawings have to use the same method. In the third step, which is the theoretical study, a more complex but not as concise a method as in the second step is discussed. The main purpose of this thesis is to find a method that the robot can use to calculate the drawing locus itself by coordinating the image it saves from its camera.

## 1.2 Overview Structure

Eight chapters are included in this thesis. The first chapter introduces the background of the whole thesis, which includes relevant concepts, and knowledge. It starts with the objective of the thesis. The structure of the whole thesis follows. Then comes the background and the relevant concepts. As mentioned above, the first section of the thesis is about the NAO robot’s vision, which is a computer vision. When talking about vision, color must be discussed. Anything that can be shown as an image has colors. However, computer vision is not the same as human vision. Computers process information by digitizing the information, and when they process information, the information should be in a format that can be read by a computer. After the color module, image processing is introduced. After the basic background, the tools and techniques used in this thesis are introduced. OpenCV is used as the computer vision library to process the image for the further process detection or recognition. Optical character recognition (OCR) and Tessract are used to recognize the characters. In this thesis, the main carrier is a NAO robot, and the study is all about a NAO robot, so after introducing the tools, there is a brief introduction to the NAO robot in two main blocks; the hardware and software of the robot. After the introduction, Chapter Two is about the flowchart of the whole thesis and each main section area. Chapter Three introduces the vision module, which includes the hardware of the vision and the process result of what the robot sees. After recognize the contours, characters, or faces, the robot will draw them down, so Chapter Four is about the motion module. It includes the hardware of NAO and the mathematical module which builds a theory and algorithm about how to draw. Chapter Five introduces the results of the drawing and Chapter Six discussed the further reach about how to make NAO calculate the motion itself. Chapter Seven ends the thesis.

## 1.3 Basic Background

### 1.3.1 Computer Vision (CV)

From the late 1950s, CV began to be the tool to realize the human intelligence and human perception. With a computer, humans realized the extension of brain power and perception for the first time.

CV is the science that studies how to make a machine “see”. It uses cameras and computers instead of human eyes to identify, track, and measure a target and conduct further image processing in order to make the image more suitable for transmitting to instruments or human eyes. CV uses computers and related equipment in order to simulate biological vision. Its main task is to obtain the 3D information of the corresponding scene by processing the captured images and videos, similarly to what humans and many other types of biological do every day.

CV is a challenging and important research area both in science and engineering. As a scientific discipline, CV research related theory and technology try to create AI systems that can obtain information from images or multidimensional data. The “information” above refers to the information that Shannon defined as something that can be used to help make a “decision”. Because perception can be referred to in order to extract information from the senses, CV can also be seen as a science that studies how to make artificial systems “sense” from images or multidimensional data.

As an engineering discipline, CV seeks a method that is based on related theories and models to create a CV system. The components of such a system include: process control (e.g., industrial robots and unmanned vehicles), event monitoring (e.g., image monitoring), information organization (e.g., image database and image sequences index creation), objects and environment modeling (e.g., industrial inspection, medical image analysis and topology modeling), sympathetic interaction (e.g., man-machine interactive input device), etc. CV can be referred to as a complement of biological vision as well. In the biological visual area, the vision of human and a various of animals have been studied so that these physical models that used in the process of visual system perceiving information are created. In CV, the AI system that relies on software and hardware to realize has been studied and described. Interdisciplinary communication has brought tremendous value to CV and other disciplines. CV includes for example, the following branches: picture reconstruction, event monitoring, target tracking, target recognition, machine learning, index creation, and image restoration.

CV is an integral part of various intelligent and autonomous systems with various application areas, such as manufacturing, testing, documentation analysis, medical diagnostics, military and other fields.

Figure 1 shows the relationship between CV and other fields.

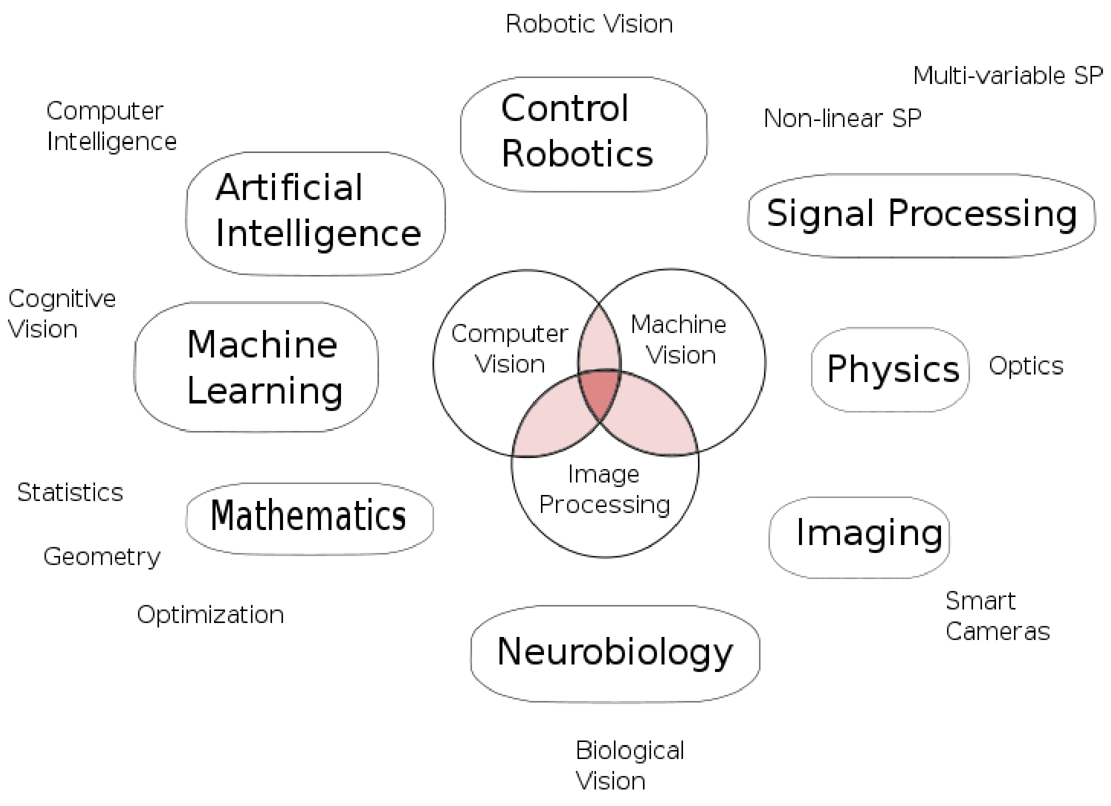


**Figure 1.** Relation between computer vision and various other fields /1/

### 1.3.2 Color Model

A color model is an algorithm that represents color in the digital world. In such a world, in order to describe various colors, each is usually divided into several components. The differences of fineness principle decide the differences of the way to produce color between the color device and the printing equipment. Color device refers to the kind of a device that uses color light to synthesize color directly, like monitors, projectors, and scanners. The printing equipment e.g. printer use pigments.

Color model is a model that expresses a color in a digital form, in other words, it is a way to record the image color. It includes some normal modes:

**RGB** mode: In nature all the colors can be combined by different intensities of the wavelength of red, green and blue. This is commonly known as color theory. Sometimes the three primary colors are called “additive colors”, because when the wavelengths of two different lights are put together, the resulting color will be brighter. It is suitable for monitors, projectors, scanners, and cameras.

**CMYK** mode: It is a printing mode that relies on reflection. It has different principles for producing colors with RGB mode. CMYK mode is also known as the subtractive method. It is suitable for printers.

**HSB** mode: It is the color mode that is based on the human psychological experience of color, which is in-line with human visual perception and makes people feel that it is intuitive.

**Lab** mode: It is converted from RGB three primary colors. “L” refers to the colorless channel, “a” is the yellow-blue channel, and “b” is a red-green channel. It is a color mode that is relatively close to the visual display of  human eyes.

**Bitmap** mode: It uses two colors (black and white) to represent the pixels in an image. Bitmap mode images are also known as black and white images. Because the depth of pixel is 1, it is referred to as a one-bite image as well.

**Grayscale** mode: It use up to 256 shades of gray to represent the image to make the transition smoother and finer. Each pixel of the grayscale image has a brightness value from 0 (black) to 255 (white).

### 1.3.3 Image Processing

Image processing is a technology that by using computer analysis target images to achieve the desired result. Image processing generally refers to digital image processing. Digital image means using industrial cameras, camcorders, scanners, and other equipment through photographs in order to get a large 2D array. The elements of the array are called pixels. They have a value called gray value. Image processing technology typically includes three parts which are image compression, image enhancement and restoration and image matching, description and identification.

Figure 2 shows the conversion between physical image and digital image.

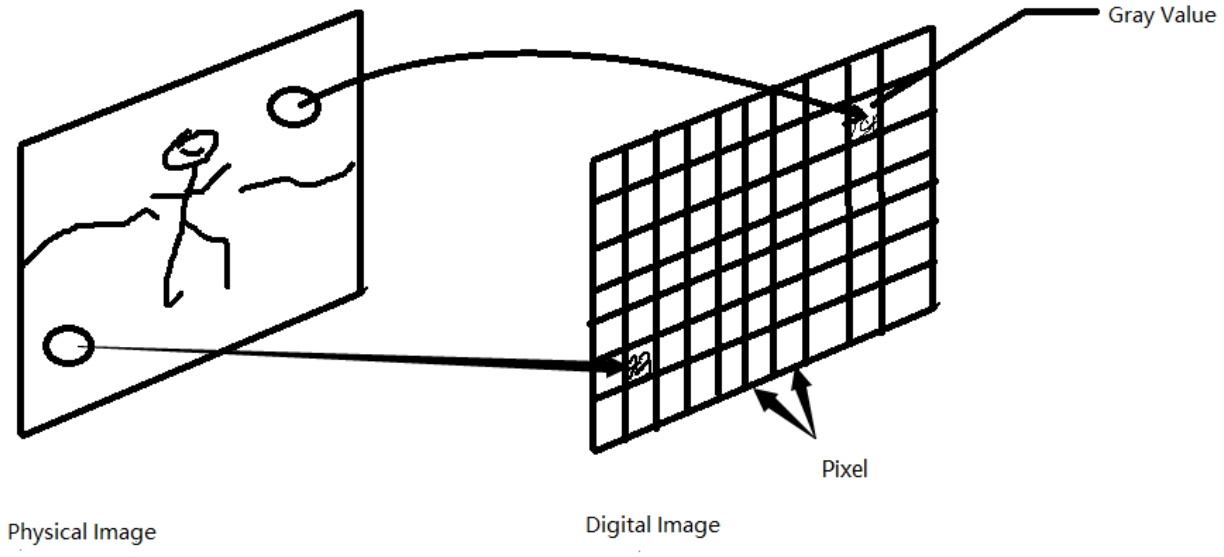


**Figure 2.** The conversion between Physical Image and Digital Image

## 1.4 Tools and Techniques Used in This Thesis

### 1.4.1 Open-Source Computer Vision Library (OpenCV)

The full name of OpenCV is Open Source Computer Vision Library. It is an open-source computer vision library. OpenCV was originally initiated and developed by Intel. Now it is supported by Willow Garage as a major supporter, it is released under a BSD license and can be free to use in commercial and research. It can be used to develop real-time image processing, computer vision, and pattern recognition program. Currently, it is widely used in the field of industry and scientific research.

OpenCV is a cross-platform CV library which is can be run on Windows, Linux and Mac OS. It is lightweight and efficient and combined with a series of C functions and a few C++ classes. OpenCV provides the interfaces for Python, Ruby, MATLAB and other languages which realized many common algorithms for image processing and CV.

The application areas of OpenCV include HCI (or HMI), face perception, image segmentation, object recognition, SfM, and robots, for example. OpenCV includes a statistical machine learning library which contains boosting, decision tree learning, random forest, etc. to support some of the areas above.

### 1.4.2 Optical Character Recognition (OCR)

OCR is a process in which an electronic device (such as a scanner or a digital camera) checks the characters printed on paper and through detecting the model of dark and light determines the shape and then translates the shape into computer text by using a character recognition method. That is the technique for printed characters using optical way to convert the text on paper documents into image files with a black and white dot matrix and convert the text in the image into text format by recognition software for further editing by word processing software.

Due to the popularity and wide use of scanners, OCR simply provides the interface with the scanner and drives the software using a scanner. Thus, OCR software is mainly composed of the following sections:

**Image input:** For different image formats there are different storage formats and different compression methods. Figure 3 is an example of the input image. It is an image that is saved on a local computer**.** Figure 4 is a grayscale image.

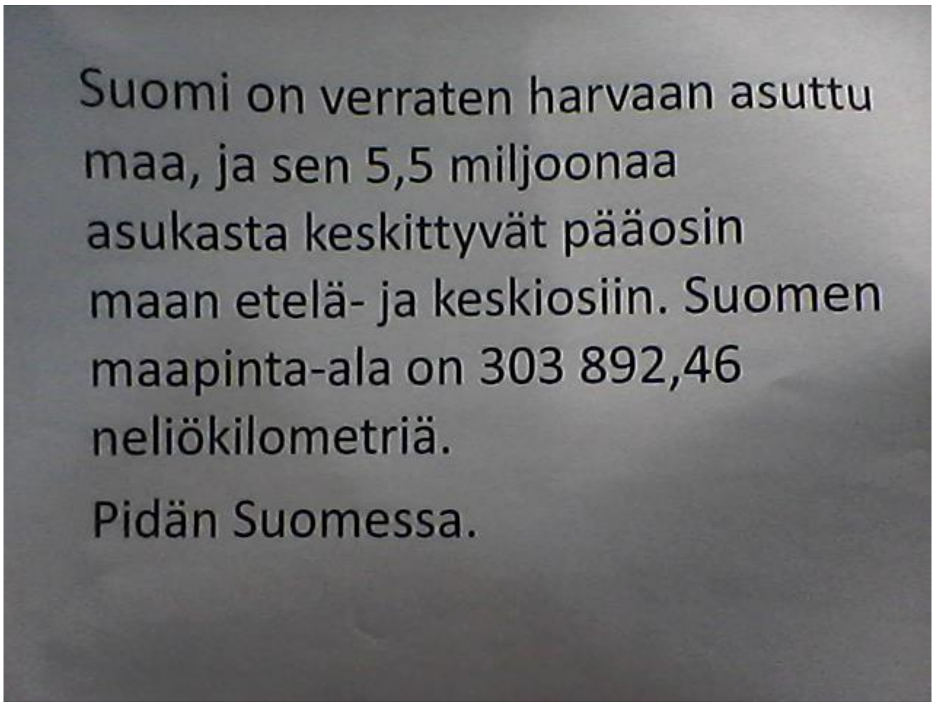
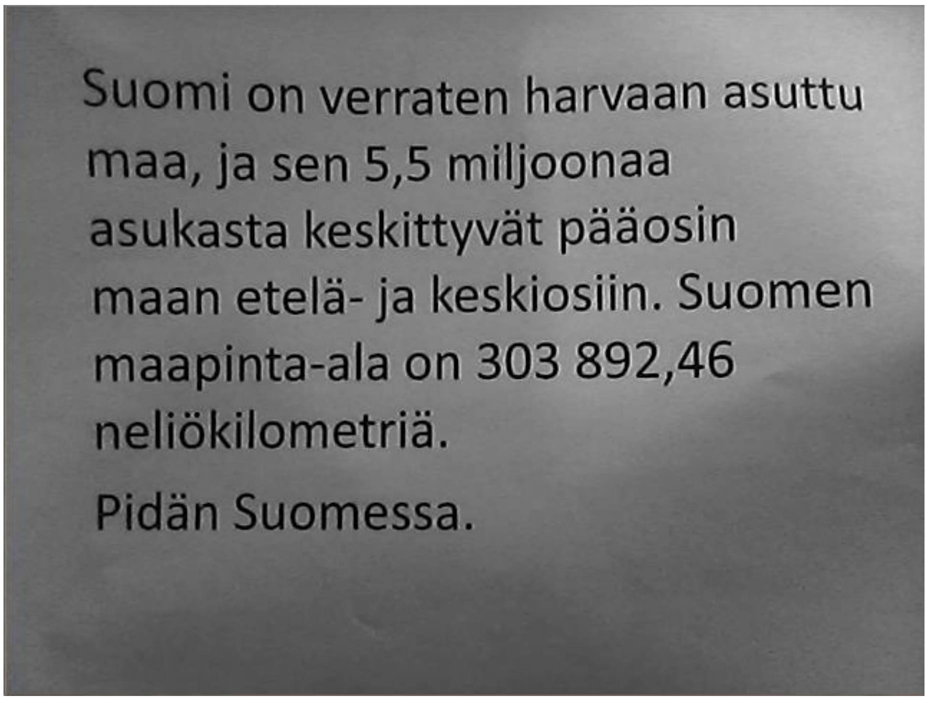

Suomi on verraten harvaan asuttu
maa, ja sen 5,5 miljoonaa
asukasta keskittyvät pääosin
maan etelä- ja keskiosiin. Suomen
maapinta-ala on 303 892,46
neliökilometriä.
Pidän Suomessa.

**Figure 3.** Image saved on a local computer

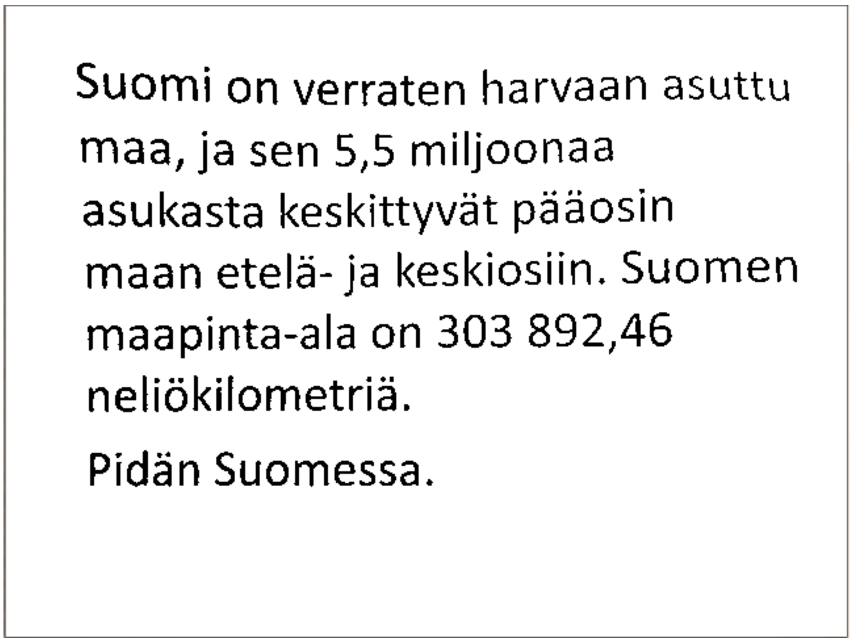


Suomi on verraten harvaan asuttu
maa, ja sen 5,5 miljoonaa
asukasta keskittyvät pääosin
maan etelä- ja keskiosiin. Suomen
maapinta-ala on 303 892,46
neliökilometriä.
Pidän Suomessa.

**Figure 4.** Grayscale image

**Image preprocessing:** It includes binarization, noise removal, tilt correction and so on.

- **Binarization:** Pictures taken by cameras are mostly color images that contain a huge amount of information. The content of the image can be simply divided into foreground and background. In order to make the computer recognize the text faster and better, the color image needs to be processed to make the image only have foreground information and background information. The foreground information can be defined with black and  the background information with white, so this is a binarized image (Figure 5).

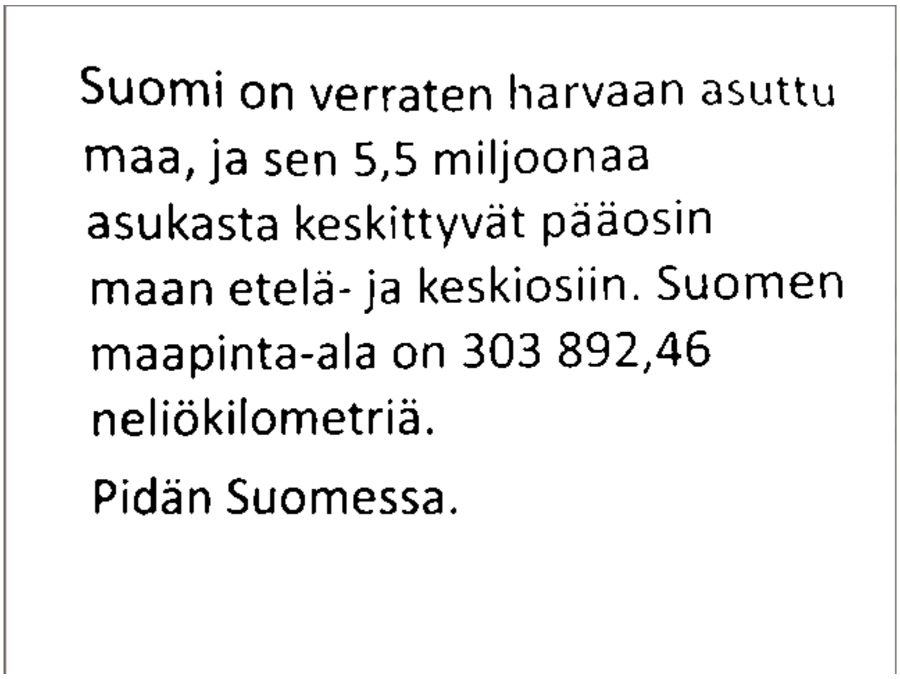


Suomi on verraten harvaan asuttu maa, ja sen 5,5 miljoonaa asukasta keskittyvät pääosin maan etelä- ja keskiosiin. Suomen maapinta-ala on 303 892,46 neliökilometriä.

Pidän Suomessa.

**Figure 5.** Binarized image

- **Noise removal:** For different documents, it is possible to have the different definitions for noise. Denoizing according to the characteristics of the noise is called noise removal (Figure 6).



Suomi on verraten harvaan asuttu maa, ja sen 5,5 miljoonaa asukasta keskittyvät pääosin maan etelä- ja keskiosiin. Suomen maapinta-ala on 303 892,46 neliökilometriä.

Pidän Suomessa.

**Figure 6.** Denoized image

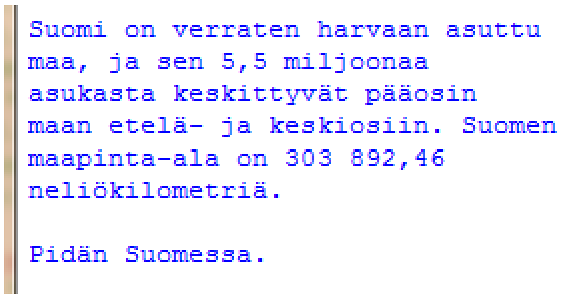
- **Tilt correction:** Most users are very casual when taking photographs for documentation, so the images are inevitably tilted. Thus, character recognition software is requited in order to correct the tilt.

**Layout analysis:** Divides the document picture into paragraphs and branches, this process is called layout analysis.

**Cutting characters:** Due to the limitations of photographs, the performance of the recognition system is greatly limited. It requires the character recognition software to have a character cutting function.

**Character recognition:** This study began was one of the first in the field of image recognition, first there was template matching and then feature extraction was the major.

**Layout recovery:** It is hoped that after character recognition, the text is still arranged as the original image with the same paragraphs, same positions, and same order when output to a word document, PDF document, and documents of other formats. The process is called layout recovery (Figure 7).



```
Suomi on verraten harvaan asuttu
maa, ja sen 5,5 miljoonaa
asukasta keskittyvät pääosin
maan etelä- ja keskiosiin. Suomen
maapinta-ala on 303 892,46
neliökilometriä.

Pidän Suomessa.
```

**Figure 7.** Output of the recognized characters

**Post-processing and proofreading:** Based on the relationship of the context, in the specific locale, the processing to correct the identify results is called post-processing.

### 1.4.3 Tessract

Tessract is an OCR engine and it is not only open-source but also probably the most accurate of the various of OCR engine. It can process a wide variety of image formats and convert them to text in over 60 languages by being combined with Leptonica Image Processing Library. It works on Windows, Linux, and Mac OS. It can also be compiled for other platforms, including Android and iOS.

### 1.4.4 Python IDLE

IDLE, Integrated Development and Learning Environment in full, is for beginners to use to create, run, test and debug Python programs easily.

In this thesis, the programming language used is Python. Therefore, IDLE is used to program and provide the import to NAOqi and OpenCV.

### 1.4.5 MATLAB

MATLAB is mathematical software. It is an advanced technique used for algorithm development, data visualization, data analysis and numerical calculation. MATLAB and Simulink are two main parts that are included in the software.

In this thesis, MATLAB is used to so some mathematical calculations and generate functions.

## 1.5 Brief introduction to the NAO robot

In this thesis, a NAO robot is the main carrier. So the question is: who is NAO?

### 1.5.1 Who is NAO?

Developed by Aldebaran Robotics, the programmable humanoid NAO Evolution Robot has reached the fifth generation and comes with enhanced audio and video capabilities, more than natural motion and more computing power. /2/

The NAO robot is 22-inch-high, or around 58cm, with a cute and small round body and a pleasing appearance. It has a certain degree of artificial intelligence and a certain degree of emotional intelligence and can interact with people in a friendly manner.

The robot has an ability to learn as well, just like a real human baby. It can infer a person’s emotion changes by learning body language and facial expressions and “know” more people with time passing and distinguish the different behaviors and faces of these people. The NAO robot can also exhibit different emotions just like anger, fear, sadness, happiness, excitement, and pride. When it is faced with tension with which it cannot cope, if no one communicates with the robot, it may even be offended. Its “brain” can make it remember a positive or a negative experience in the past.

NAO robot is shown in Figure 8.

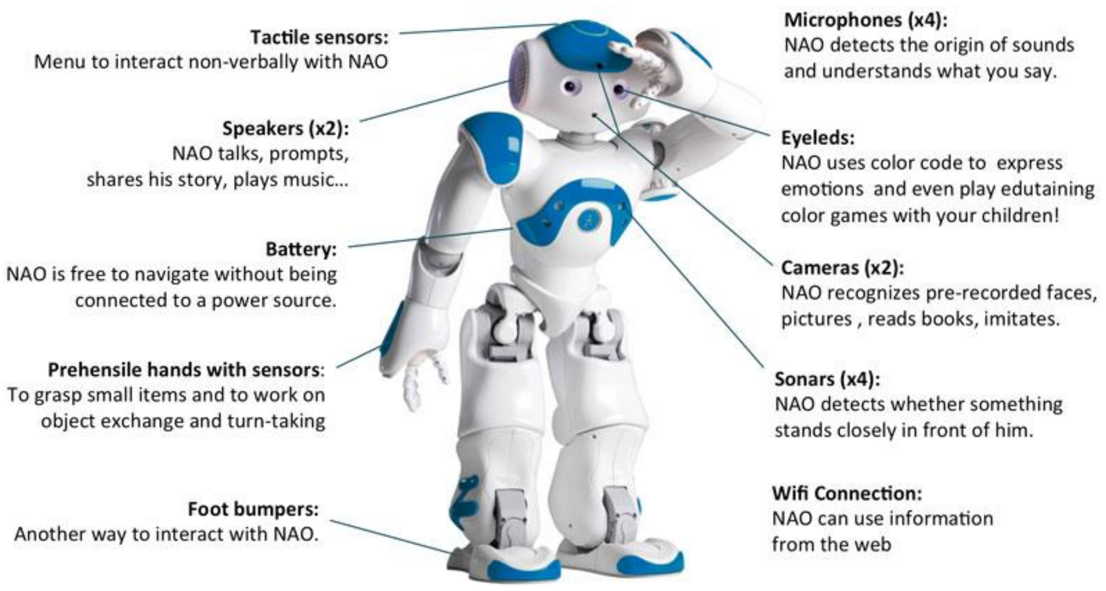
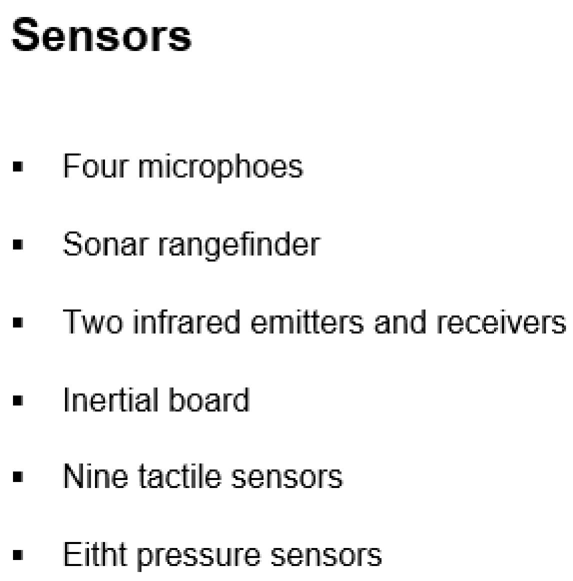

Tactile sensors:
Menu to interact non-verbally with NAO
Speakers (x2):
NAO talks, prompts, shares his story, plays music...
Battery:
NAO is free to navigate without being connected to a power source.
Prehensile hands with sensors:
To grasp small items and to work on object exchange and turn-taking
Foot bumpers:
Another way to interact with NAO.
Microphones (x4):
NAO detects the origin of sounds and understands what you say.
Eyeleds:
NAO uses color code to express emotions and even play edutaining color games with your children!
Cameras (x2):
NAO recognizes pre-recorded faces, pictures , reads books, imitates.
Sonars (x4):
NAO detects whether something stands closely in front of him.
Wifi Connection:
NAO can use information from the web

**Figure 8.** NAO robot and the design/3/

The robot is a unique product that is combined with a variety of hardware and software.

### 1.5.1.1 Hardware

The hardware of a NAO robot includes a large number of motors and sensors. The robot has flexible movements. It has 25 motors to enable it to move freely. Figure 9 shows its sensors.



Sensors

- Four microphoes
- Sonar rangefinder
- Two infrared emitters and receivers
- Inertial board
- Nine tactile sensors
- Eitht pressure sensors

**Figure 9.** The list of sensors/6/

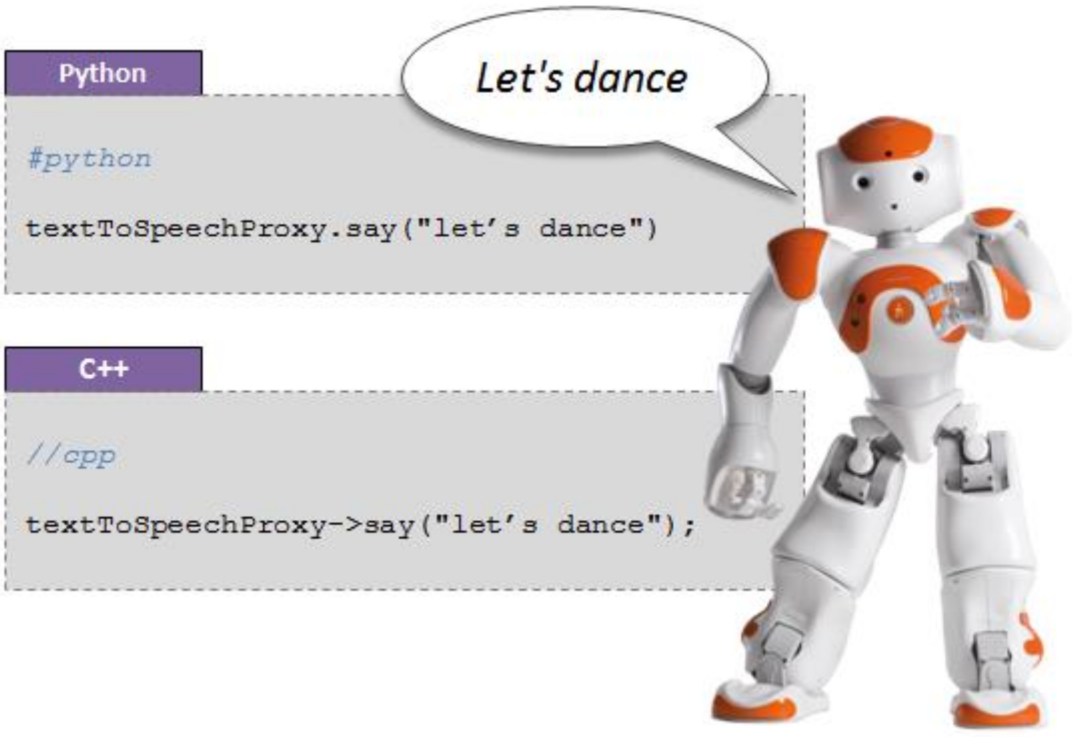
The robot has an inertial navigation system in order to maintain stability in the moving mode. It can also rely on ultrasonic sensors to detect and bypass obstacles, which makes its activity very accurate. Each foot is equipped with four pressure sensors for determining the position of the pressure center of each foot and make some appropriate adjustments to make NAO maintain balance.

#### 1.5.1.2 Software

A major feature of the NAO robot is its embedded software. Through this software, NAO robot can do sound synthesis, sound localization, detection of vision images and colored shapes, obstacles detection (with dual-channel ultrasound system) and though a large number of light-emitting diodes produce visual effects or interaction.

All of the software is managed by a specially designed operating system called NAOqi, which is running on the motherboard located in the head of the robot. NAOqi OS is the operating system of the robot, specifically developed to fit the robot needs. It is a GUN/Linux distribution based on Gentoo, so the robot can be used as a Linux OS computer. It provides and runs a number of programs and libraries. /4/

NAOqi is the name of the main software that runs on the robot and controls it. NAOqi framework is the programming framework used to program the robot. The NAOqi frame work is cross-platform and cross-language which allows people to create distributed applications. It can be used on Windows, Linux and Mac OS and the programming language could be Python and C++. Figure 10 shows the same application in two different languages. /5/



```
#python

textToSpeechProxy.say("let’s dance")
```

```
//cpp

textToSpeechProxy->say("let’s dance");
```

**Figure 10.** The same application in two different languages/5/

Choregraphe and Monitor are desktop software. Choregraphe is a multi-platform desktop application. It allows people to create applications without writing code. The applications include interacting with people, e-mail sending, grasp, tracking red ball, Taichi, dance and so on. It is very convenient for beginners. However, in this thesis Choregraphe is not used. Figure 11 shows the interface of Choregraphe and 12 shows the interface of Monitor.

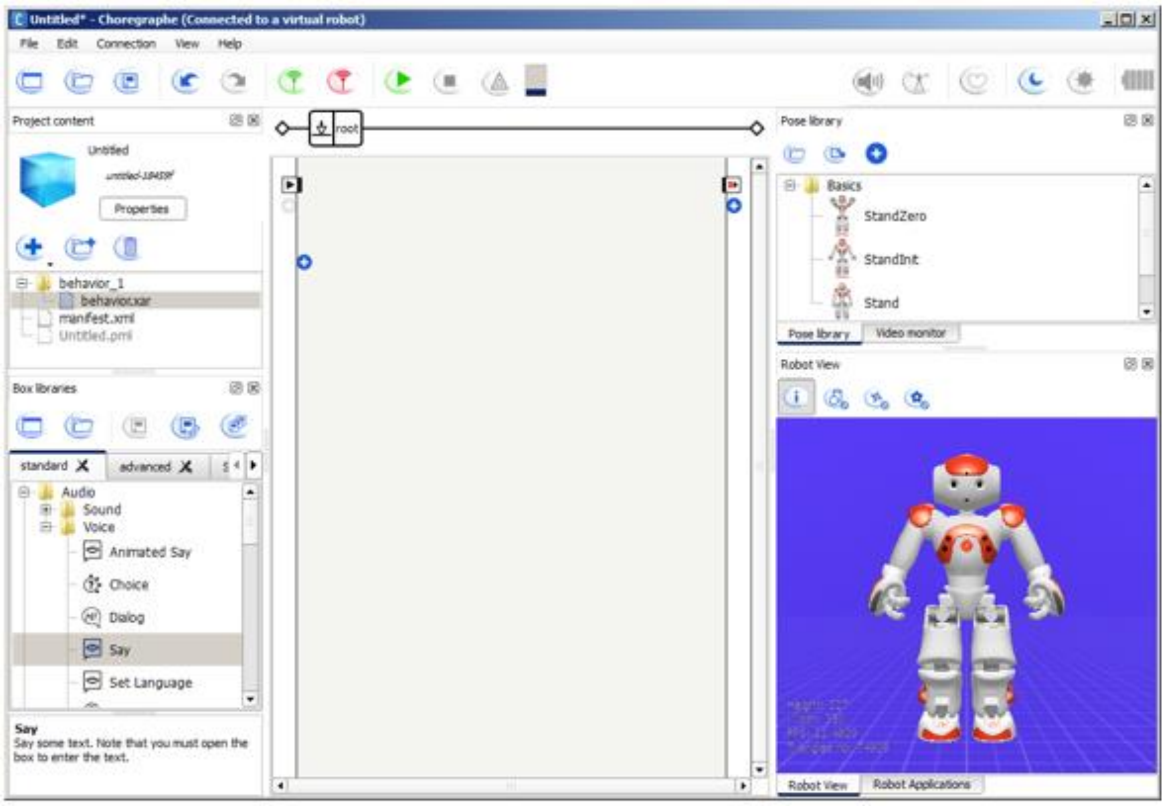
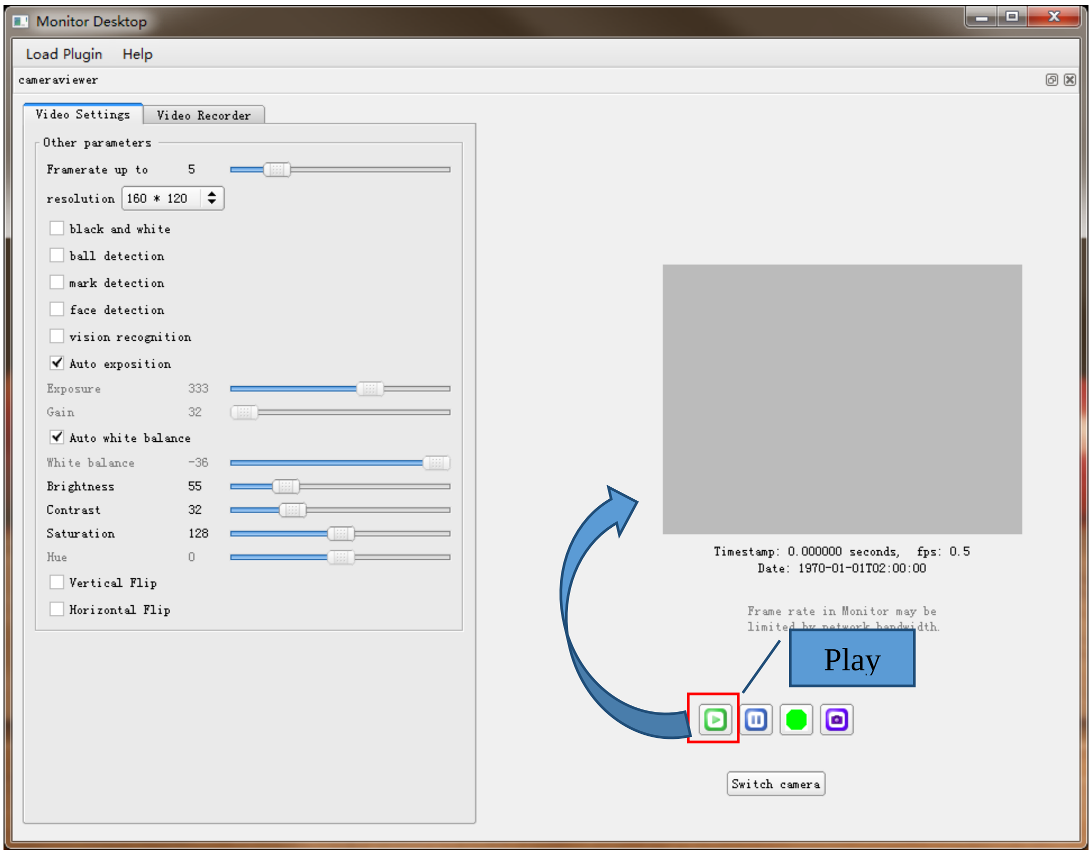


**Figure 11.** The interface of Choregraphe



Other parameters

Framerate up to 5

resolution 160 * 120

☐ black and white

☐ ball detection

☐ mark detection

☐ face detection

☐ vision recognition

☑ Auto exposition

Exposure 333

Gain 32

☑ Auto white balance

White balance -36

Brightness 55

Contrast 32

Saturation 128

Hue 0

☐ Vertical Flip

☐ Horizontal Flip

Timestamp: 0.000000 seconds, fps: 0.5
Date: 1970-01-01T02:00:00

Frame rate in Monitor may be limited by network bandwidth.

**Figure 12.** The interface of Monitor

The monitor is dedicated to use in order to get feedback from the robot and simple access to its camera setting. In this thesis the monitor is used to view what the robot sees by clicking the play button.

### 1.5.2 NAOqi Process

The NAOqi execution which runs on the robot is a broker. When it starts to work, a preferences file named "autoload.ini" will be loaded which defines which libraries should be loaded. One or more modules are contained in each library and may use the broker to advertise their methods.

The following figure (Figure 13) is the NAOqi process.

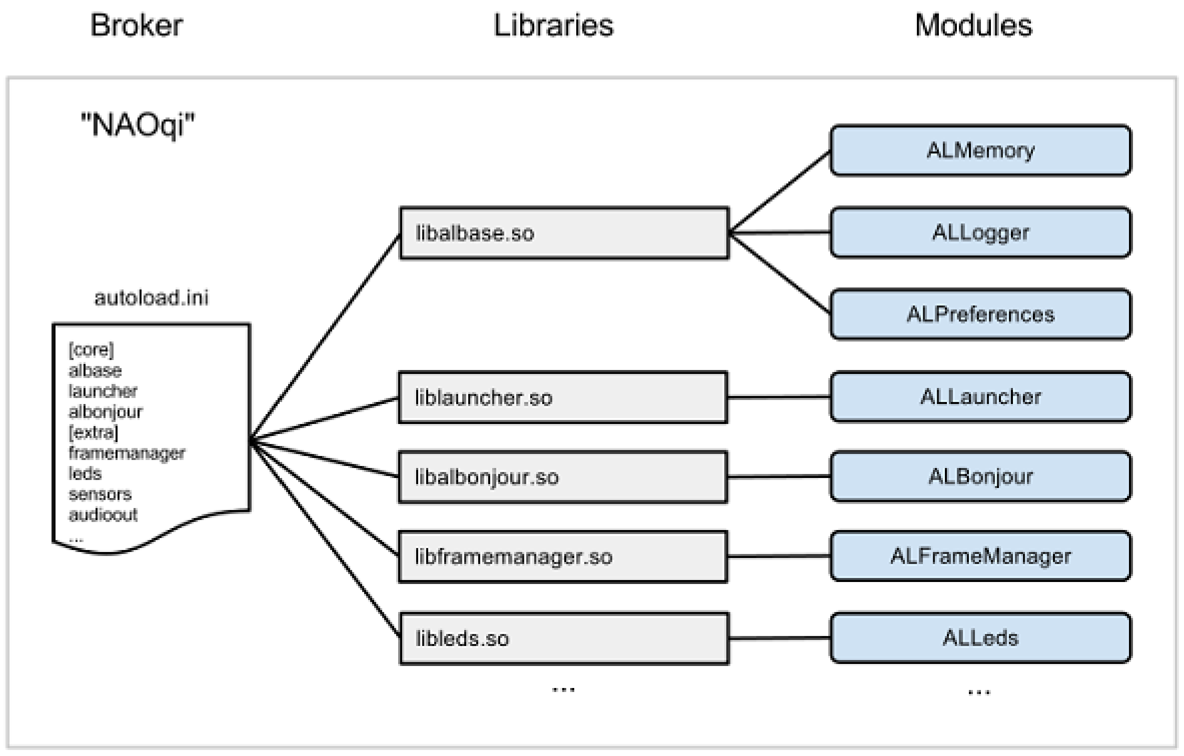


**Figure 13.** The NAOqi process/5/

#### 1.5.2.1 Broker

A broker is an object that provides lookup services, which includes directory services and network access so that any module in the tree or across the network can find any method that has been quoted. Loading modules forms a tree of methods attached to the modules, and the modules are attached to a broker.

Most of the time, the broker does not need to be thought about. /5/

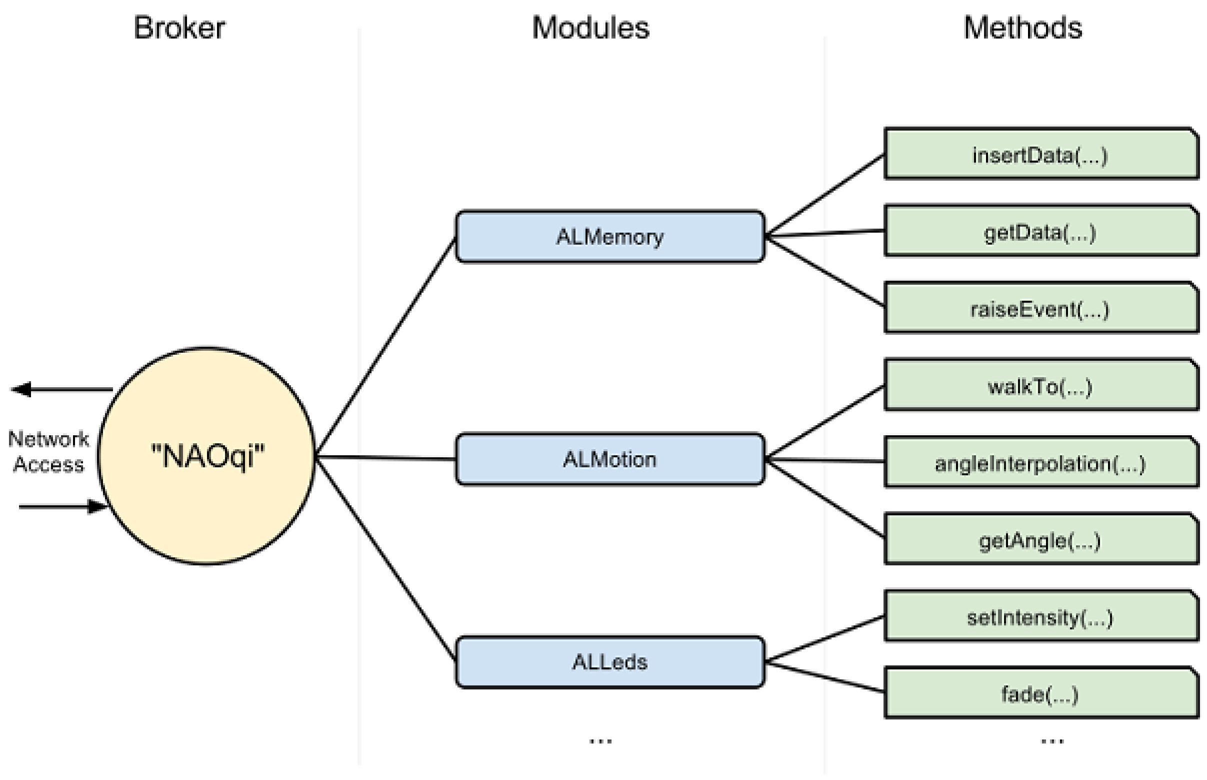


**Figure 14.** Broker

### 1.5.2.2 Proxy

A proxy is also an object that will behave as the module it represents. ALProxy is the Python API which was used in this thesis. This object lets people create a proxy to a module and call all the methods they want from NAOqi in order to make it easy to execute the robot without Chrographe.

The necessary proxy is imported as:

```
>>> from naoqi import ALProxy
```

# 2 OVERALL DESIGN

This thesis can be divided into two main sections: the vision module and the motion module. In the first section, there are three detections, and in the second section, there are three corresponding drawings.

## 2.1 Outline

This is the outline of the whole thesis. Two main sections are included, and in each section there are three parts.

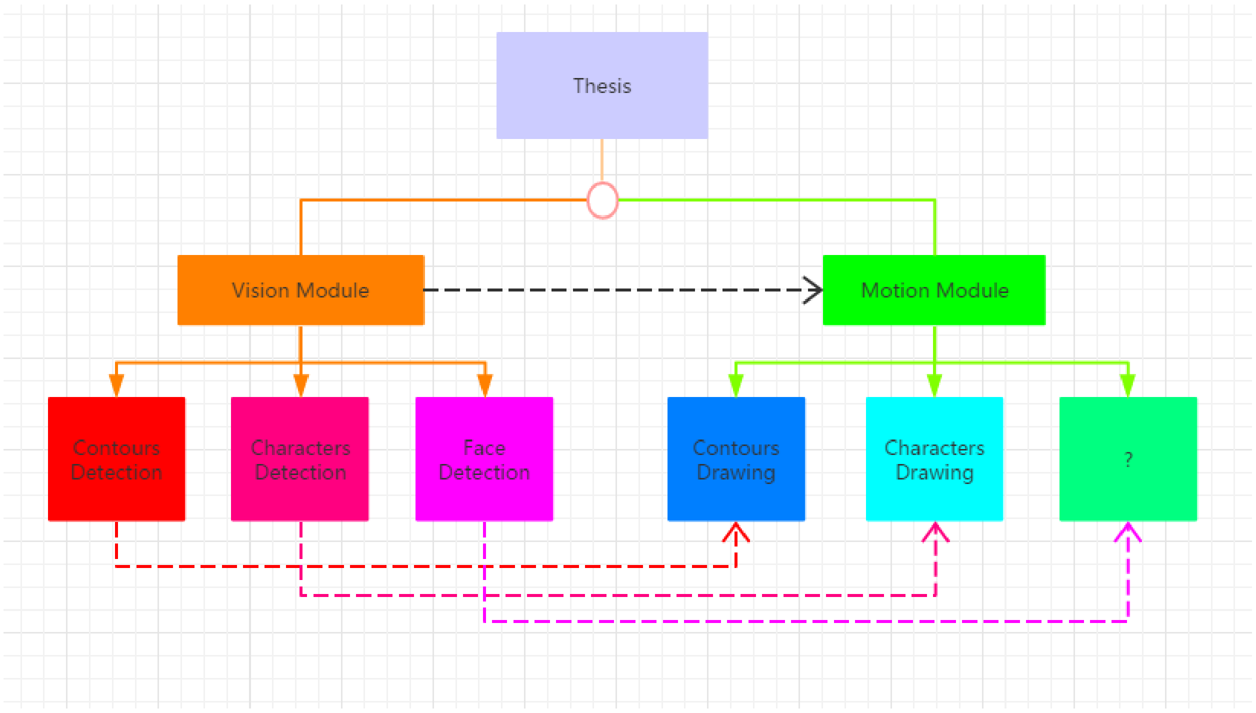


**Figure 15.** Outline of the whole thesis

## 2.1 Flowchart

At first the robot will be given a line drawn on a piece of paper in front of it. It captures one frame by using its camera and saves it locally for further analysis and processing. The computer will preprocess the image through a series of algorithms in order to convert the image into a format that a computer can recognize. After that, the feature of the graph will be extracted for the detection. The robot will get the information on what it saw. Then it will call for the motion module to draw down the graph.

Figure 16 and 17 show the flowchart of the whole process.

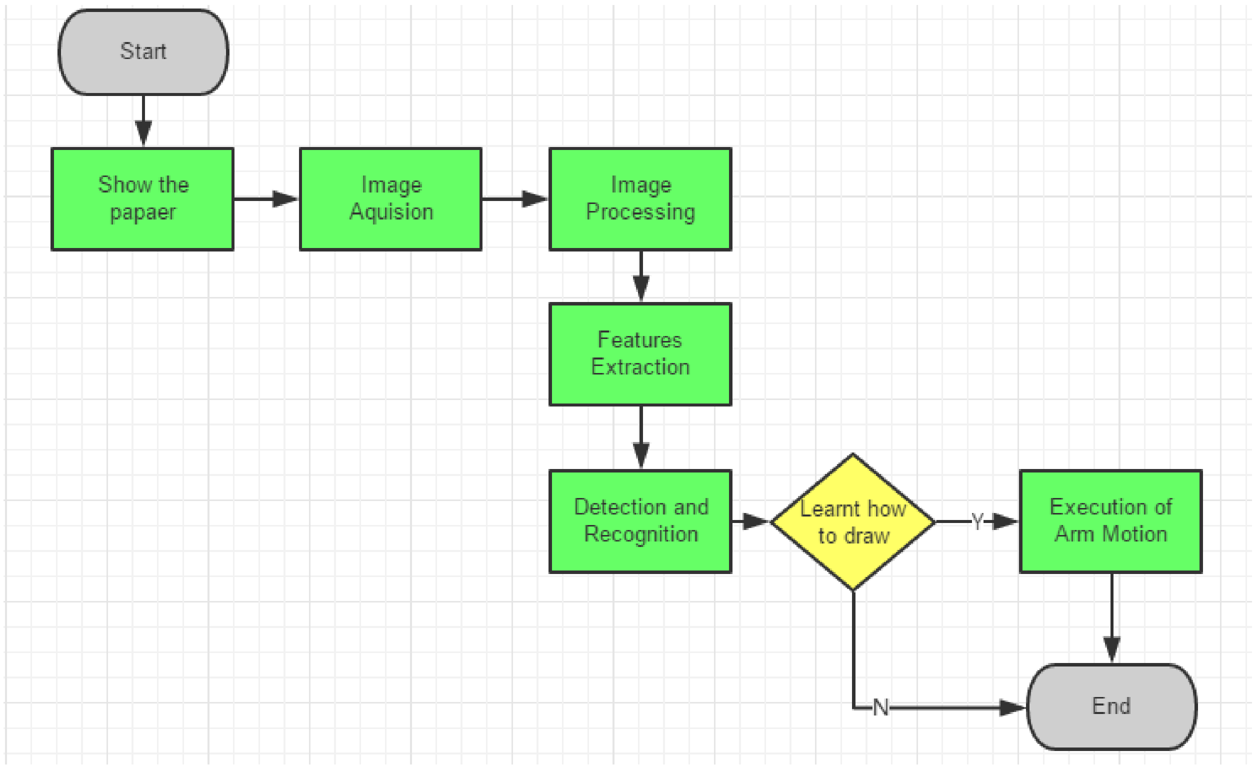


**Figure 16.** The flowchart of the whole process

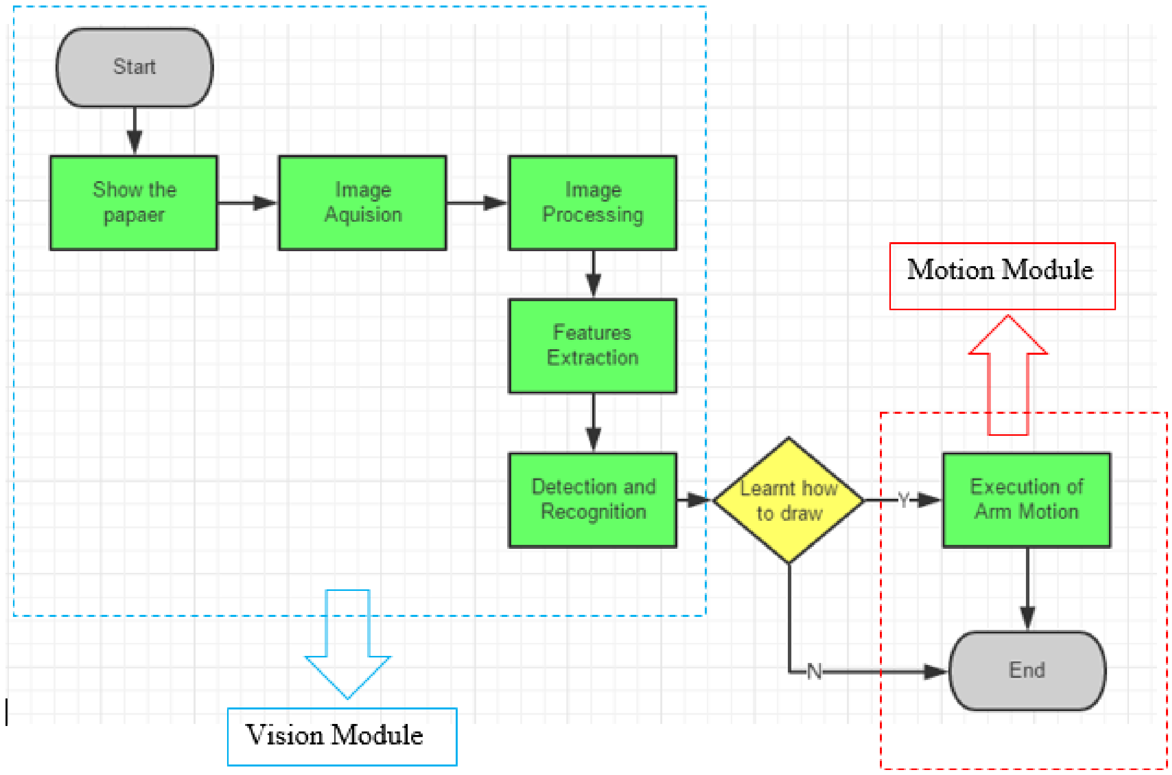
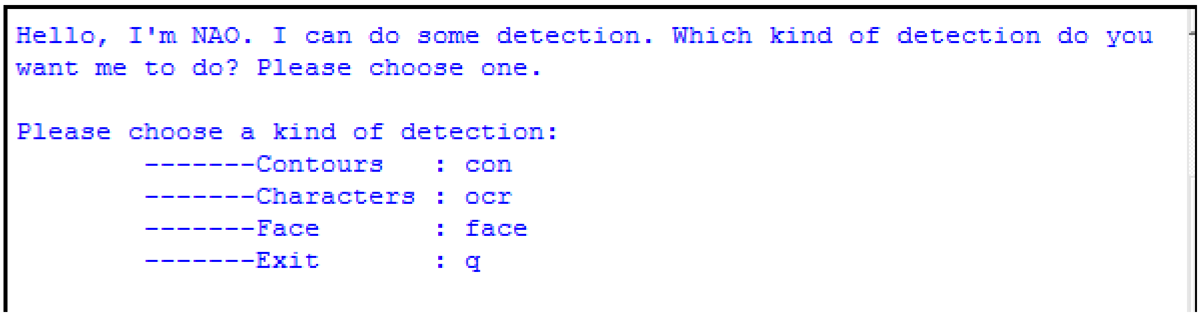


**Figure 17.** Two main parts of the thesis

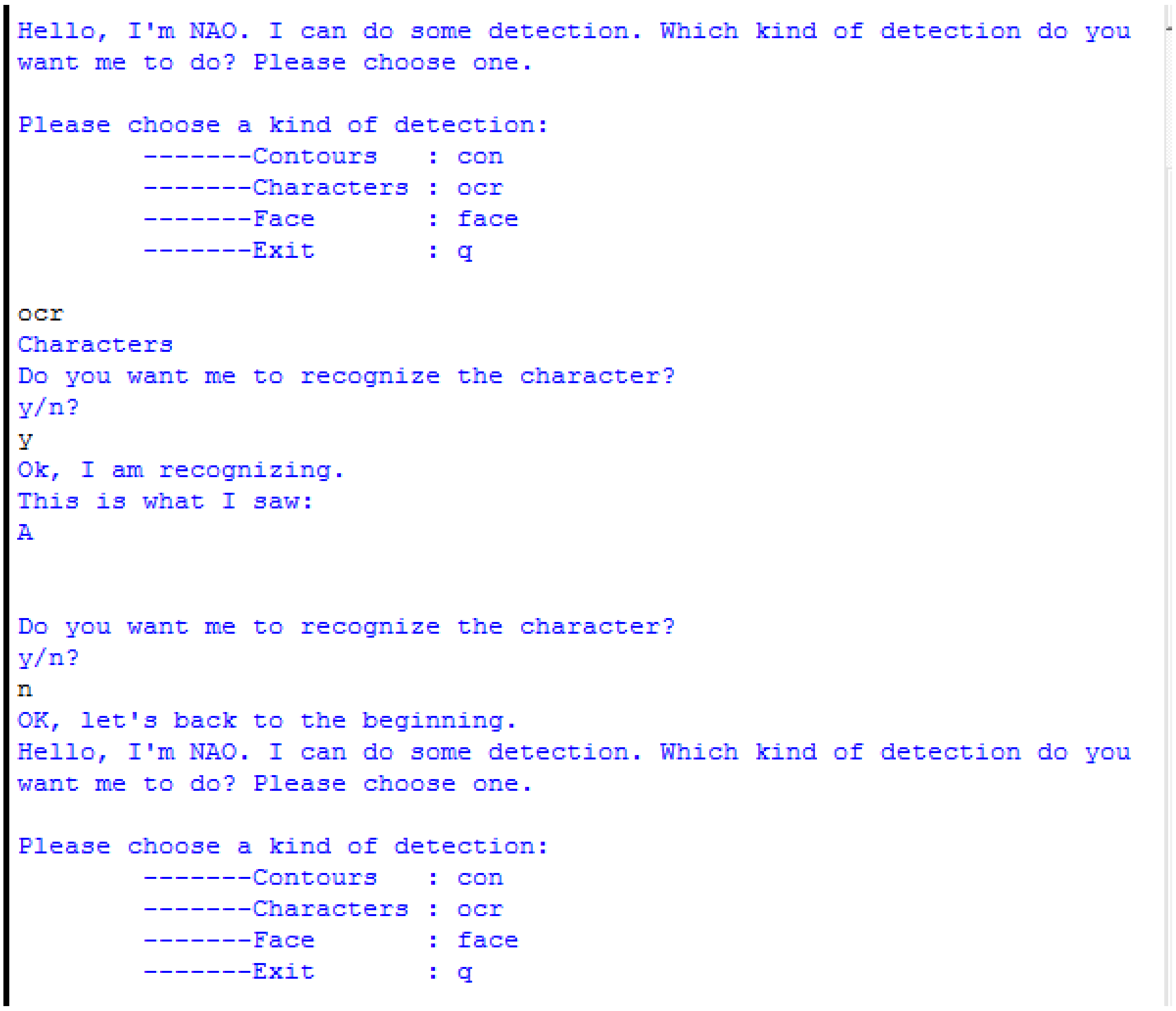
Since the thesis has three kinds of detection, at the beginning of the project the robot needs to choose one kind of detection to do.



```
Hello, I'm NAO. I can do some detection. Which kind of detection do you
want me to do? Please choose one.

Please choose a kind of detection:
        -------Contours   : con
        -------Characters : ocr
        -------Face       : face
        -------Exit       : q
```

**Figure 18.** The beginning of this thesis

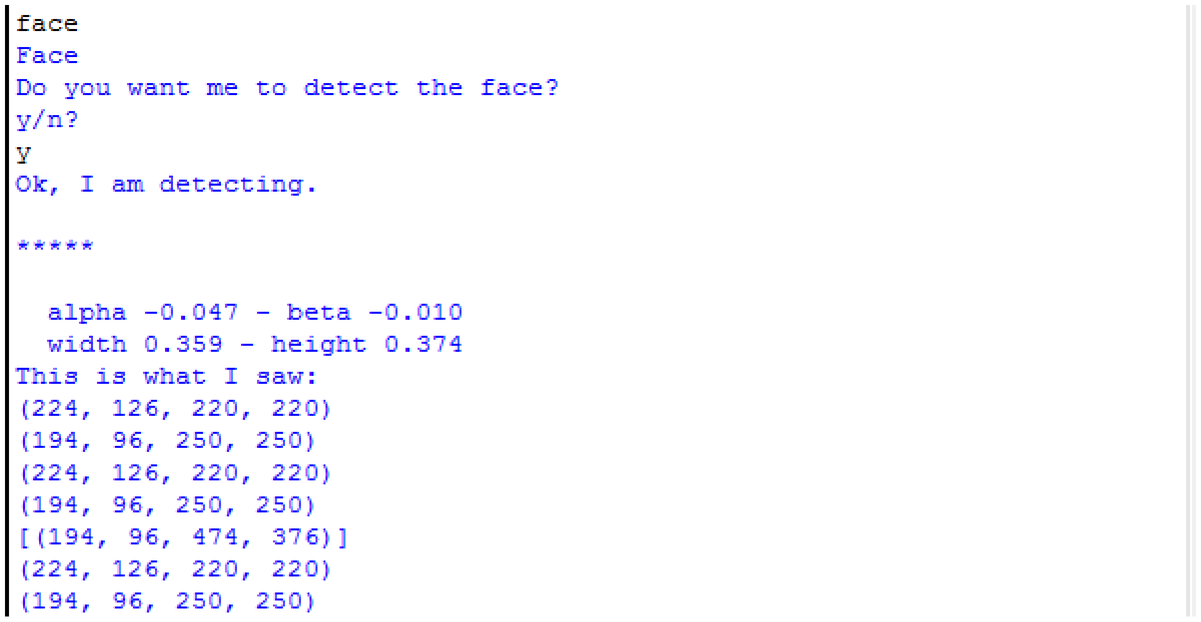
We make the decision for the NAO robot. After choosing one kind of detection, the robot will ask whether it needs to detect the chosen one.



```
Hello, I'm NAO. I can do some detection. Which kind of detection do you
want me to do? Please choose one.

Please choose a kind of detection:
        -------Contours   : con
        -------Characters : ocr
        -------Face       : face
        -------Exit       : q

ocr
Characters
Do you want me to recognize the character?
y/n?
y
Ok, I am recognizing.
This is what I saw:
A


Do you want me to recognize the character?
y/n?
n
OK, let's back to the beginning.
Hello, I'm NAO. I can do some detection. Which kind of detection do you
want me to do? Please choose one.

Please choose a kind of detection:
        -------Contours   : con
        -------Characters : ocr
        -------Face       : face
        -------Exit       : q
```

**Figure 19.** A whole process of one kind of detection

However, in the detection of the face, there was a slight difference. When face detection is chosen, the robot will detect a face before it captures one frame from the video stream. This process is realized by the robot itself, not through any other advanced tools or techniques. When analyzing the image, another face detection by OpenCV is needed.



```
face
Face
Do you want me to detect the face?
y/n?
y
Ok, I am detecting.

*****

  alpha -0.047 - beta -0.010
  width 0.359 - height 0.374
This is what I saw:
(224, 126, 220, 220)
(194, 96, 250, 250)
(224, 126, 220, 220)
(194, 96, 250, 250)
[(194, 96, 474, 376)]
(224, 126, 220, 220)
(194, 96, 250, 250)
```

**Figure 20.** The backstage of face detection

The following figure (Figure 21) is the flowchart of the whole thesis process.

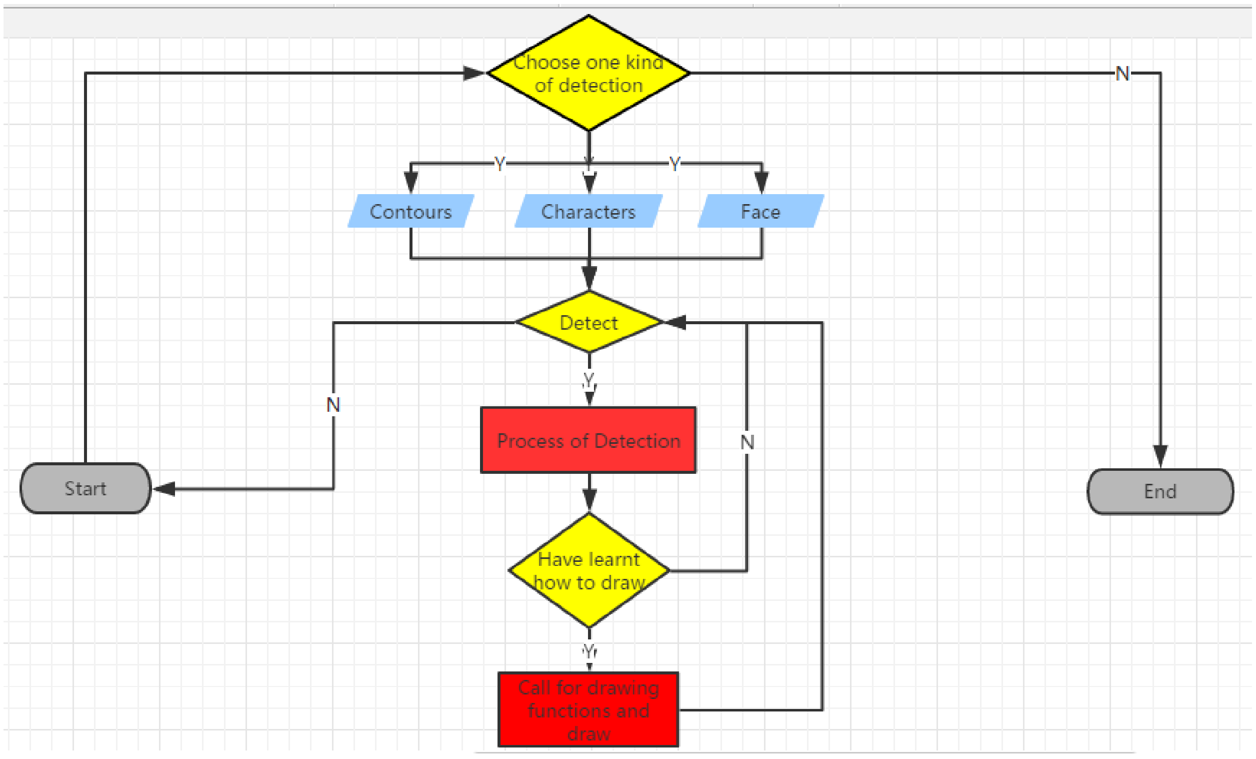


**Figure 21.** Flowchart of whole thesis process

# 3 VISION MODULE

Vision module is the first main section of the whole thesis. In this section, a system about vision will be built, and through the process of the system built above, the robot will get what its sees and by capturing one frame to “remember” what it sees. Then by processing its “memory”, it can recognize the graph.

## 3.1 Hardware of Vision System

The robot has two identical HD video cameras on its face, which are used for looking ahead and downwards. The resolution of the cameras is up to 1280*960 at 30 frames per second (Fps). The one that is located in the forehead is used to look ahead, and the other one, which is located in its mouth, is used to look downwards. Since the cameras a fixed, the angles of the vision are limited. The two cameras are used together to help NAO robot to identify the immediate surrounding by the feedback of the cameras.

In this thesis, the first camera, which is located on the robot’s forehead, is mainly used to look at the given paper in front of it and get the information from it.

The following figures 22 and 23 show the location and visual angles of the two cameras. As mentioned above, the fixed cameras cannot scroll like human eyes, so the visual angles are limited. Figure 22 shows the visual angle in the vertical direction and 23 shows the visual angle in horizontal direction.

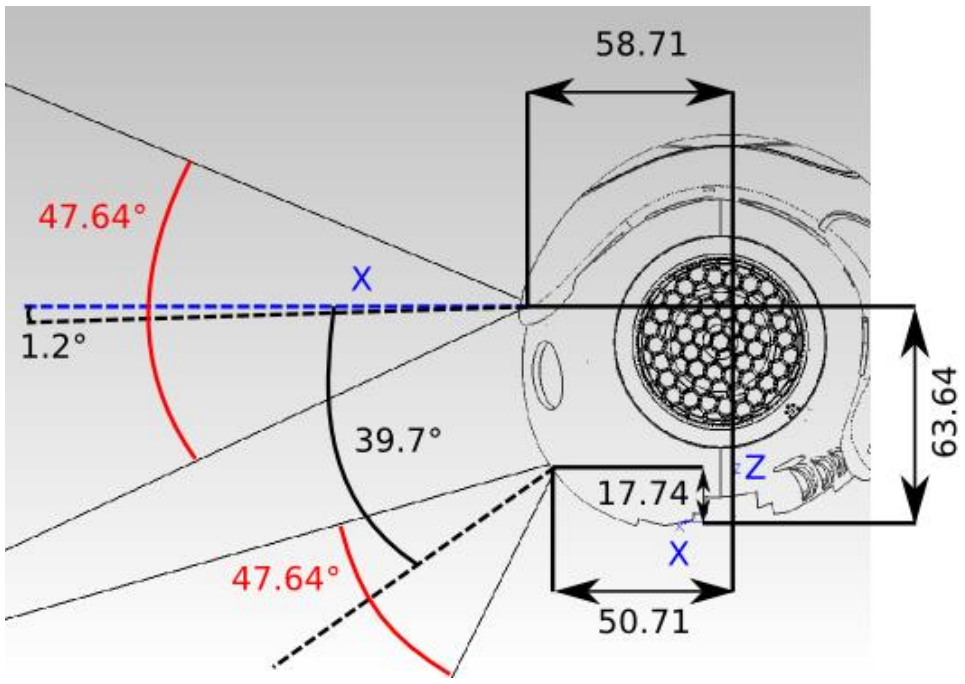


**Figure 22.** The visual angle in vertical /6/

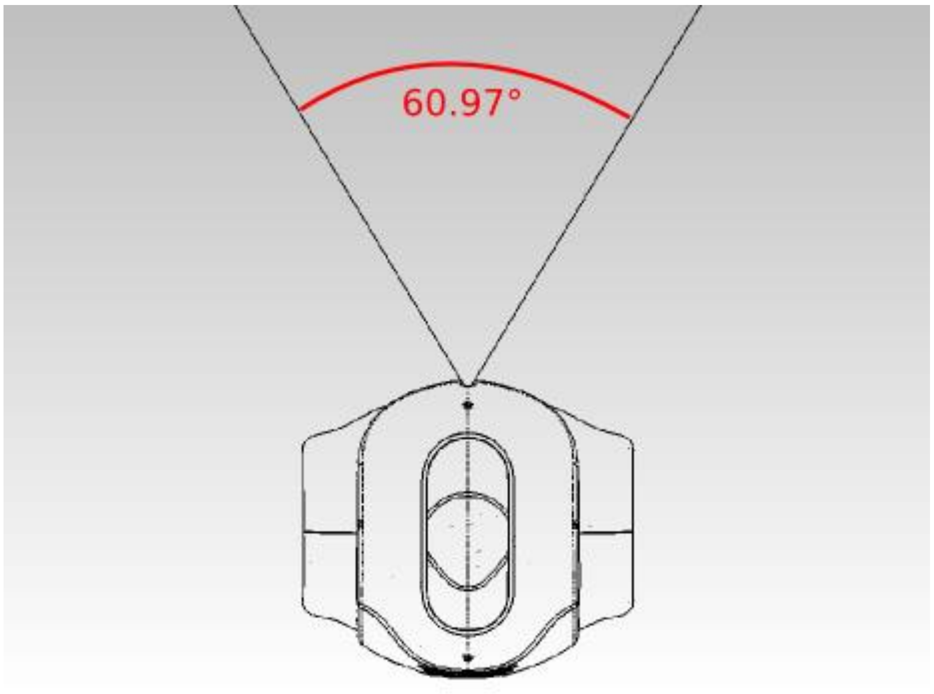
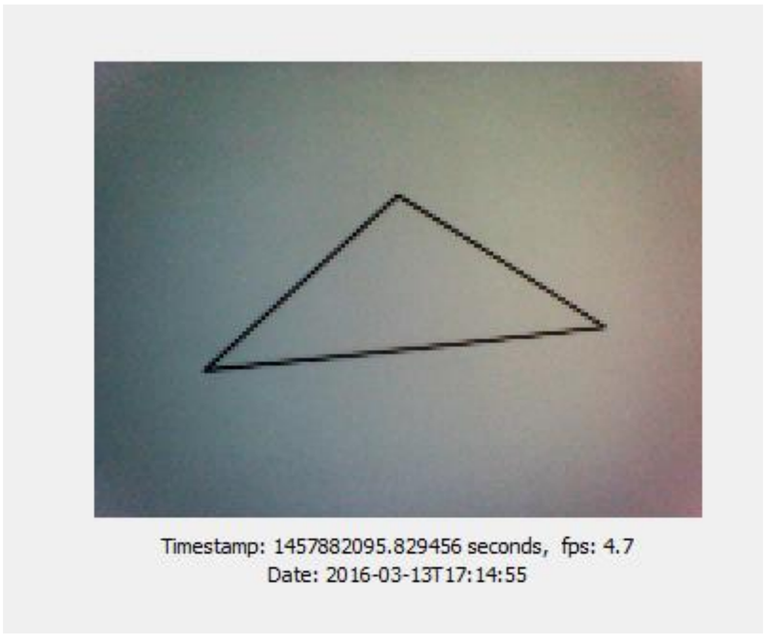


Figure 23. The visual angle in horizontal /6/

## 3.2 Image Acquisition

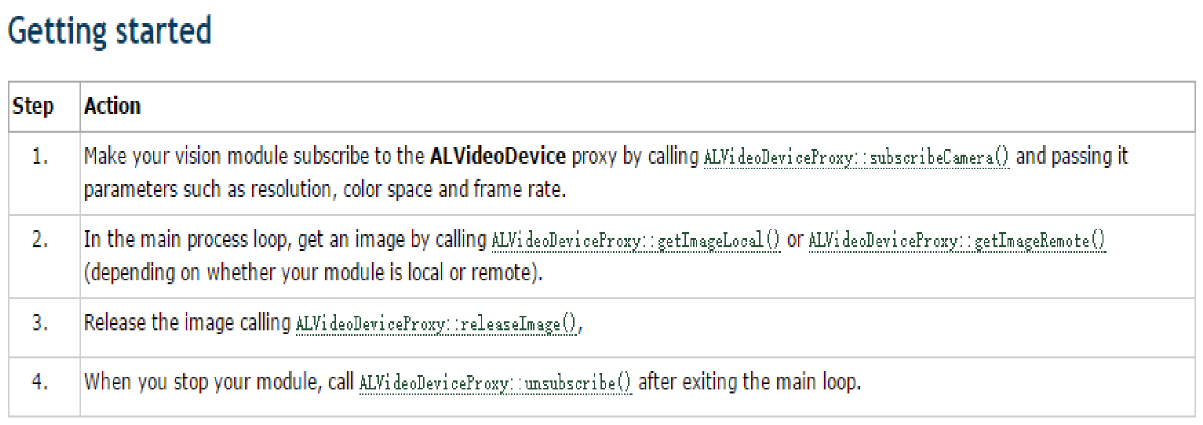
What a NAO robot sees is a stream of real-time video; thus, before analysis, one frame from the robot's video stream must be captured. Figure 24 is a case of triangle:



Timestamp: 1457882095.829456 seconds, fps: 4.7
Date: 2016-03-13T17:14:55

**Figure 24.** Video stream showed in Monitor

The first camera that is located on the forehead is used to get the information from the given paper in front of it. At first, the user should have the access to the camera. Thus, a variable needs to be created for image acquiring. In this thesis the name of the variable is **camProxy**. The module **ALVideoDevice** is used to call the robot's camera with the module name, the IP address, and the port of the robot. Figure 25 shows how to use this module.



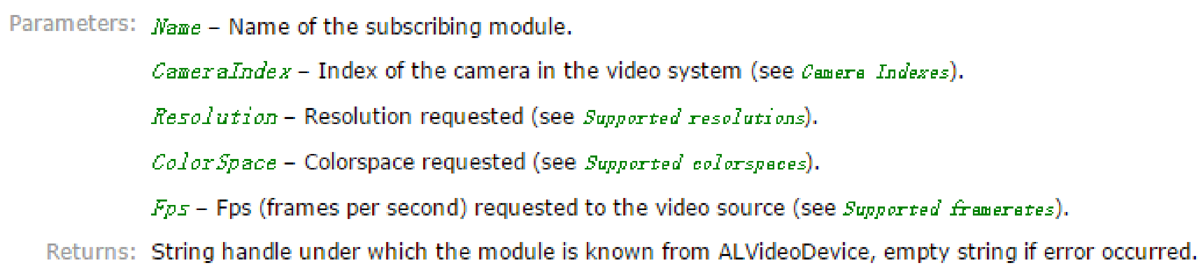
Getting started

| Step | Action |
|---|---|
| 1. | Make your vision module subscribe to the **ALVideoDevice** proxy by calling ALVideoDeviceProxy::subscribeCamera() and passing it parameters such as resolution, color space and frame rate. |
| 2. | In the main process loop, get an image by calling ALVideoDeviceProxy::getImageLocal() or ALVideoDeviceProxy::getImageRemote() (depending on whether your module is local or remote). |
| 3. | Release the image calling ALVideoDeviceProxy::releaseImage(), |
| 4. | When you stop your module, call ALVideoDeviceProxy::unsubscribe() after exiting the main loop. |

**Figure 25.** How to start the ALVideoDevice proxy /7/

In the first step, the module ALVideoDevice needs to be subscribed. When a Video Module registers to ALVideoDevice, a buffer of the requested image format is added to the buffer's list. Returns a handle which identifies the module in ALVideoDevice. (This handle is based on the name parameter, e.g. the 3rd one getting _3 added to its name for instance to build this handle). /8/

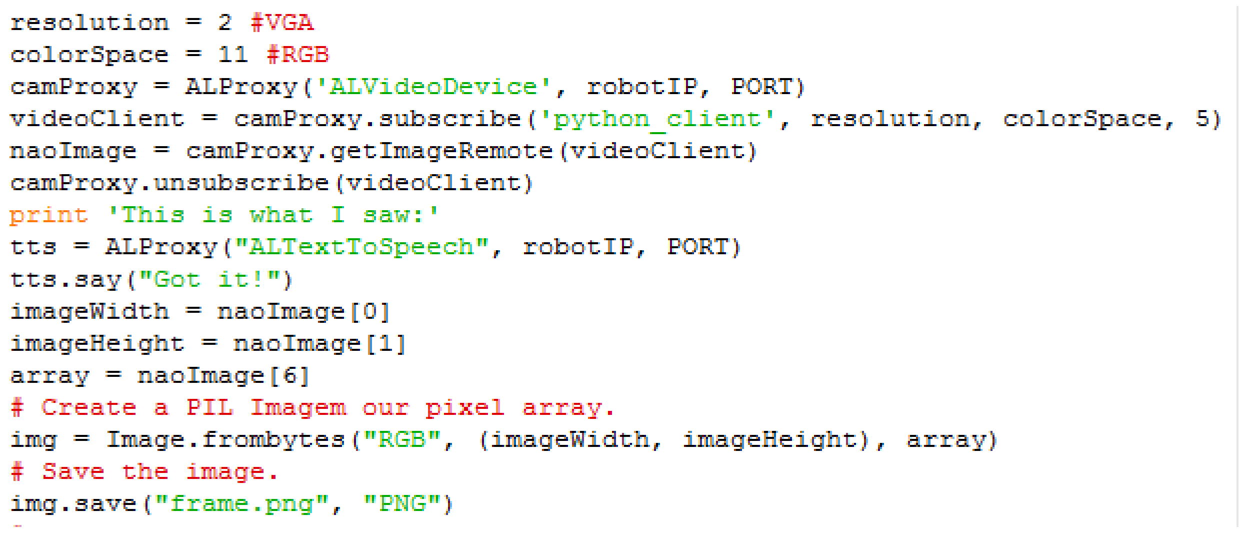
Figure 26 shows the parameters of the subscribing module.



| | |
|---|---|
| Parameters: | *Name* – Name of the subscribing module. |
| | *CameraIndex* – Index of the camera in the video system (see *Camera Indexes*). |
| | *Resolution* – Resolution requested (see *Supported resolutions*). |
| | *ColorSpace* – Colorspace requested (see *Supported colorspaces*). |
| | *Fps* – Fps (frames per second) requested to the video source (see *Supported framerates*). |
| Returns: | String handle under which the module is known from ALVideoDevice, empty string if error occurred. |

**Figure 26.** Parameters of subscribing module /8/

Figure 27 is the code of how to access to the robot's camera and how to capture one frame and save it on a local computer.



```
resolution = 2 #VGA
colorSpace = 11 #RGB
camProxy = ALProxy('ALVideoDevice', robotIP, PORT)
videoClient = camProxy.subscribe('python_client', resolution, colorSpace, 5)
naoImage = camProxy.getImageRemote(videoClient)
camProxy.unsubscribe(videoClient)
print 'This is what I saw:'
tts = ALProxy("ALTextToSpeech", robotIP, PORT)
tts.say("Got it!")
imageWidth = naoImage[0]
imageHeight = naoImage[1]
array = naoImage[6]
# Create a PIL Imagem our pixel array.
img = Image.frombytes("RGB", (imageWidth, imageHeight), array)
# Save the image.
img.save("frame.png", "PNG")
```

**Figure 27.** The code used to get one frame

The code `resolution = 2` means that the resolution of the image is 640*480px, and the framerate is from 1 to 30 Fps.

Figure 28 is the image that was captured by the robot's camera. It was saved on a local computer.

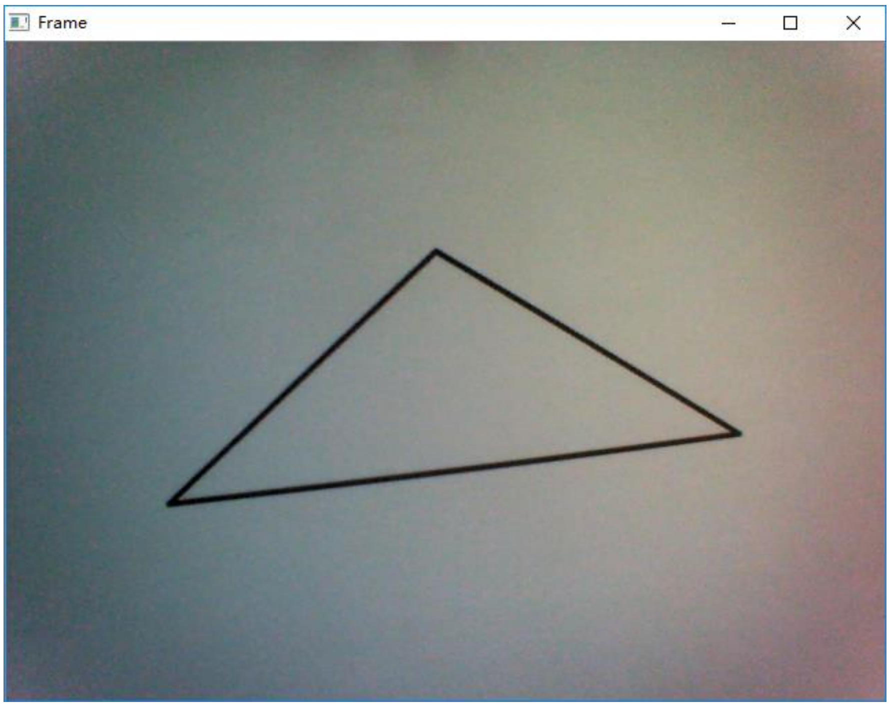


**Figure 28.** The image saved on a local computer

## 3.3 Preprocessing and Results of the Detections

### 3.3.1 Contours Detection

#### 3.3.1.1 Flowchart of Preprocessing

Figure 29 is the flowchart of preprocessing contours.

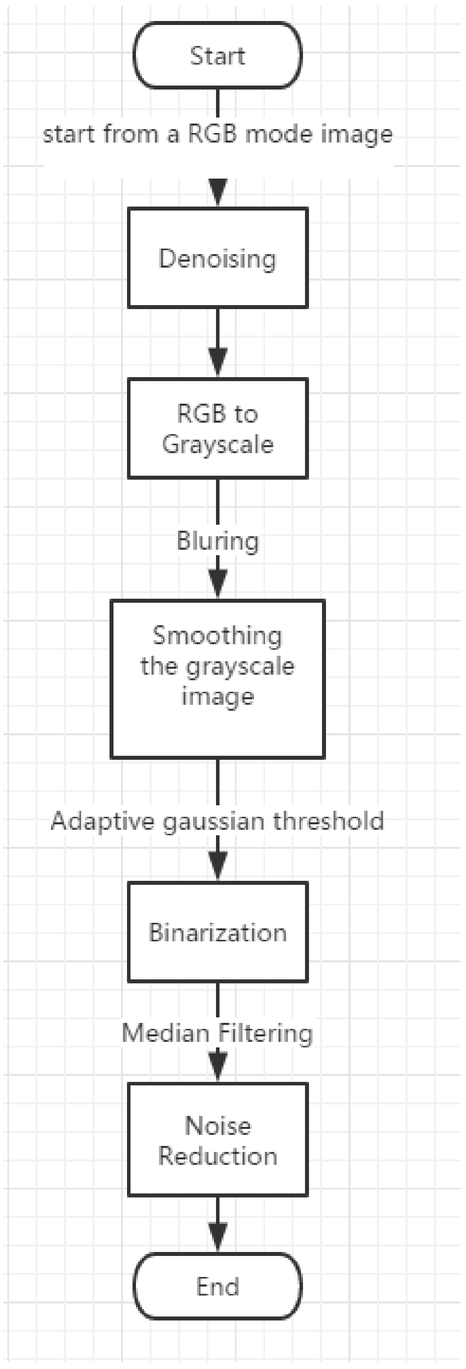

Start
start from a RGB mode image
Denoising
RGB to Grayscale
Bluring
Smoothing the grayscale image
Adaptive gaussian threshold
Binarization
Median Filtering
Noise Reduction
End

**Figure 29.** Flowchart of preprocessing (Contours)

### 3.3.1.2 Details

As showed in the flowchart in Figure 29, the first step is denoizing. Figure 30 shows the denoized image.

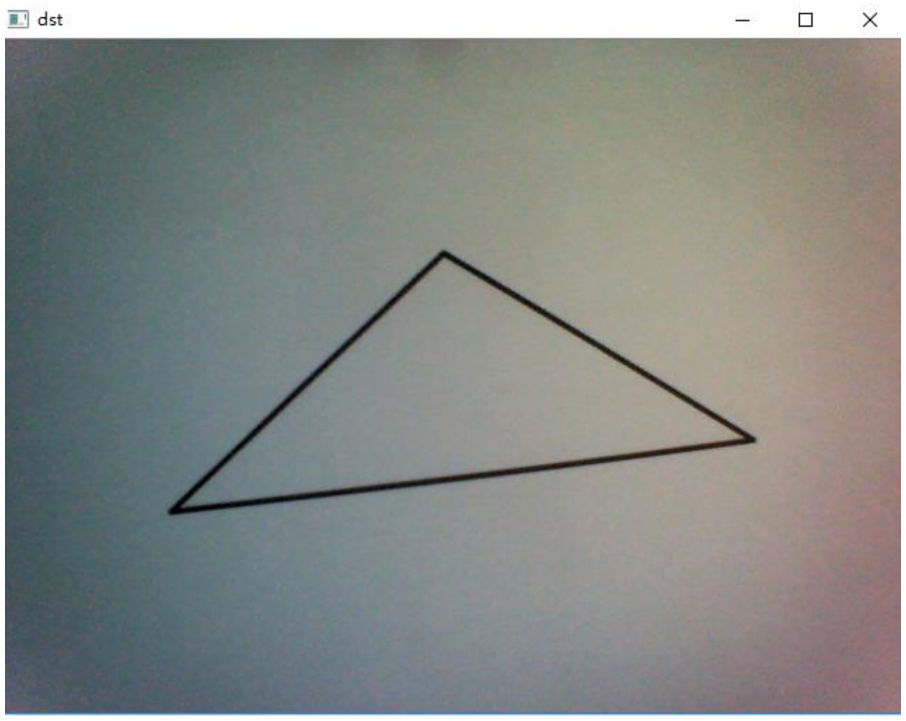


**Figure 30.** Denoized image

The next step is to convert RGB mode to grayscale mode. **Figure 31** is the grayscale image.

Figure 32, Figure 33 and Figure 34 respectively are the smoothed image, the binarized image and the final image that has been reduced in noise.

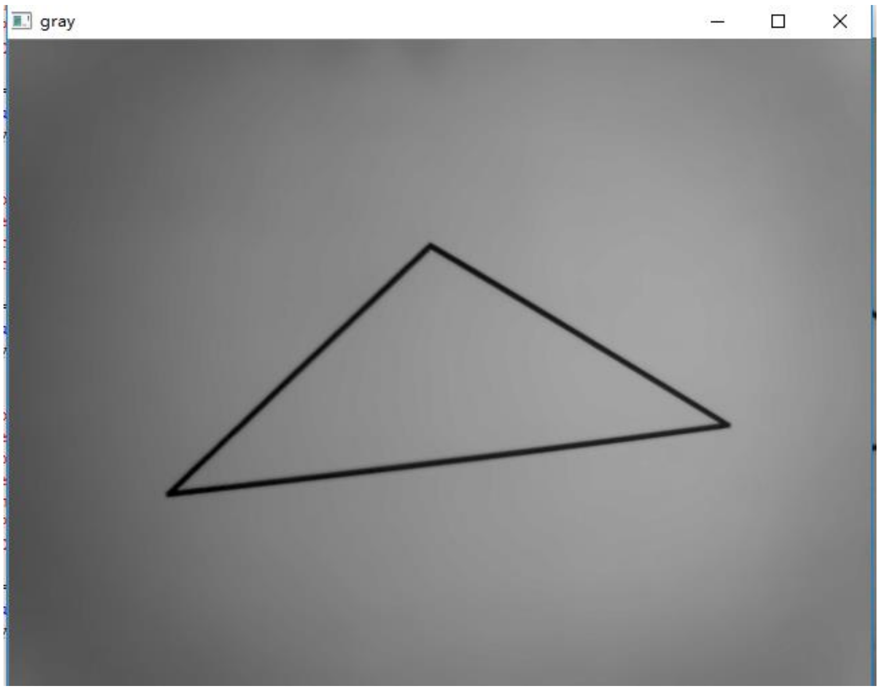


**Figure 31.** Grayscale

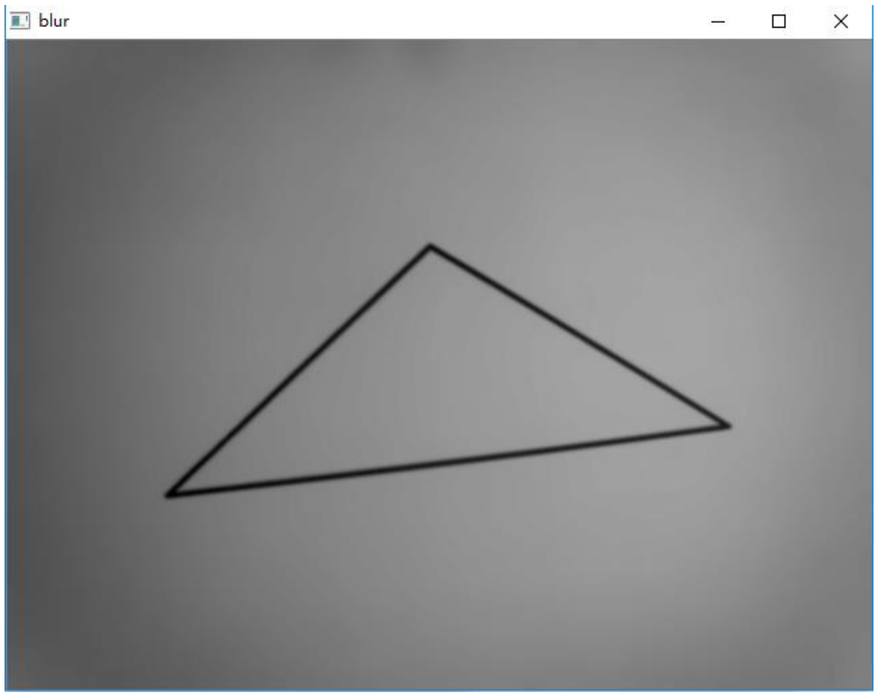


**Figure 32.** Smoothed image

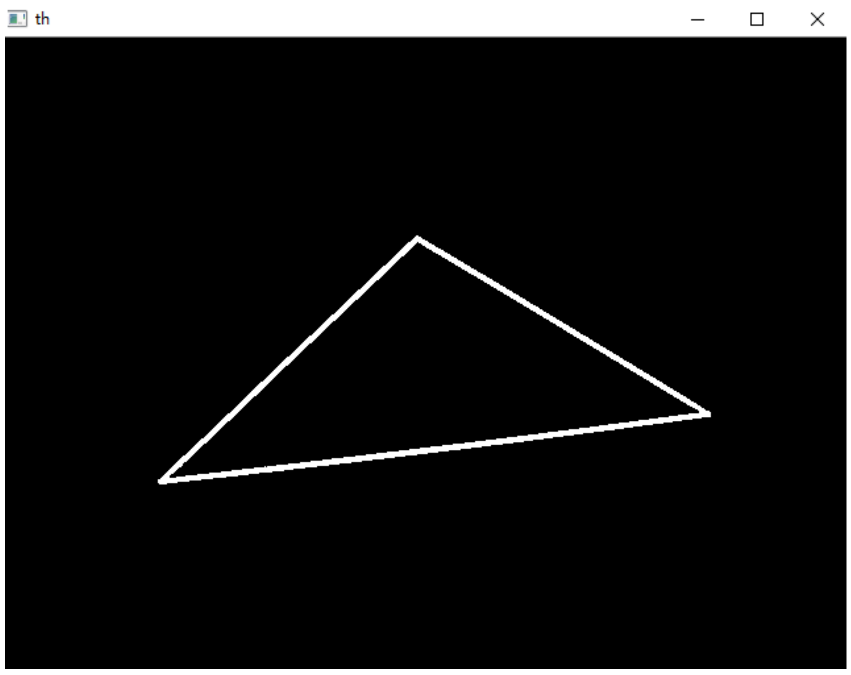


**Figure 33.** Preliminary binarized image

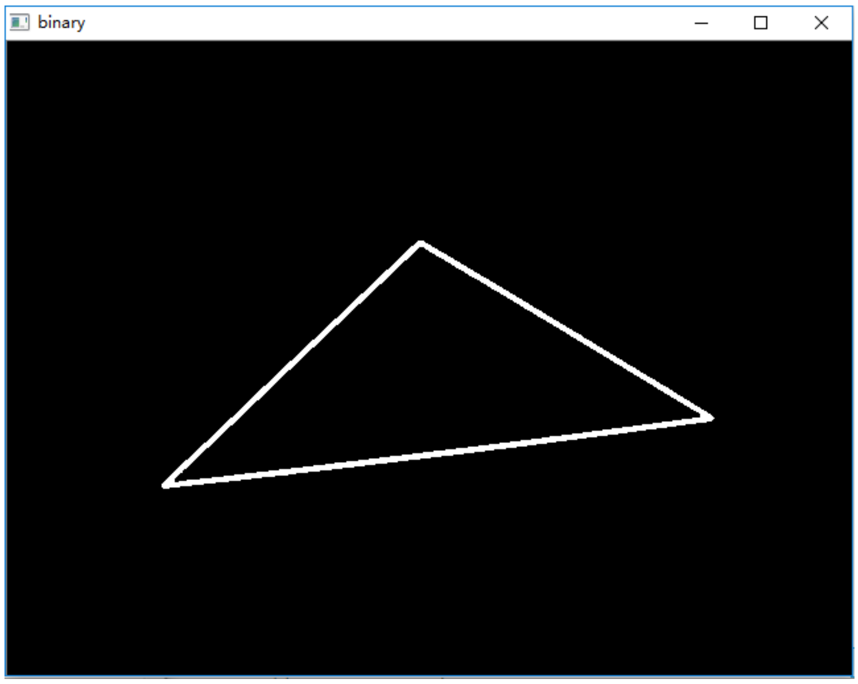


**Figure 34.** Final image that can be recognized

The robot detected other geometries, following are the comparison images of line and rectangle. Figure 35 is the comparison image of the line, and Figure 36 is for the rectangle.

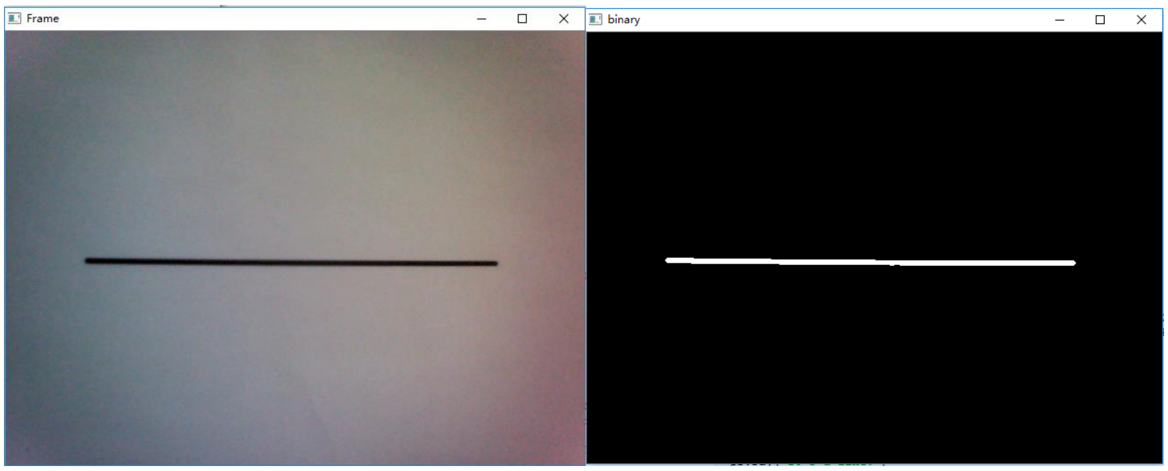


**Figure 35.** Comparison images of Line

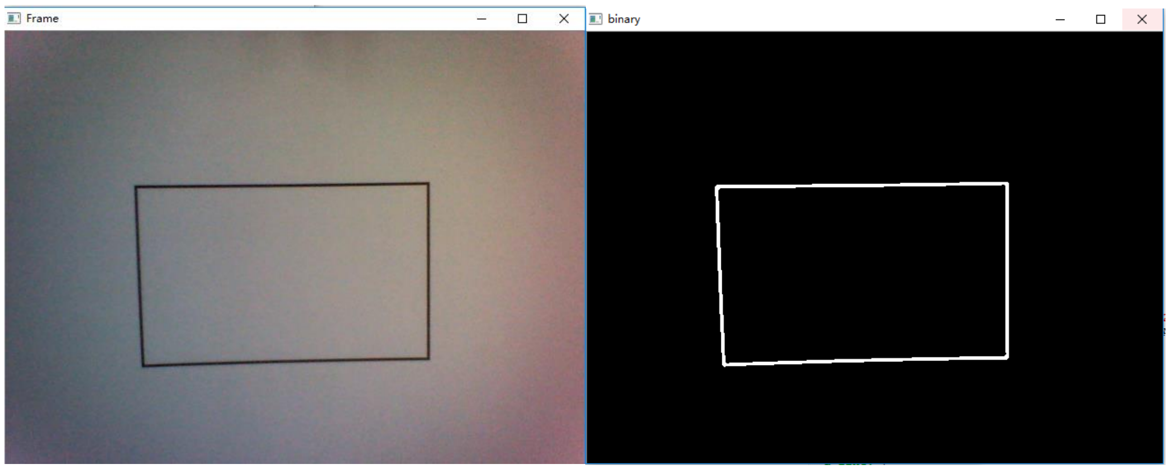
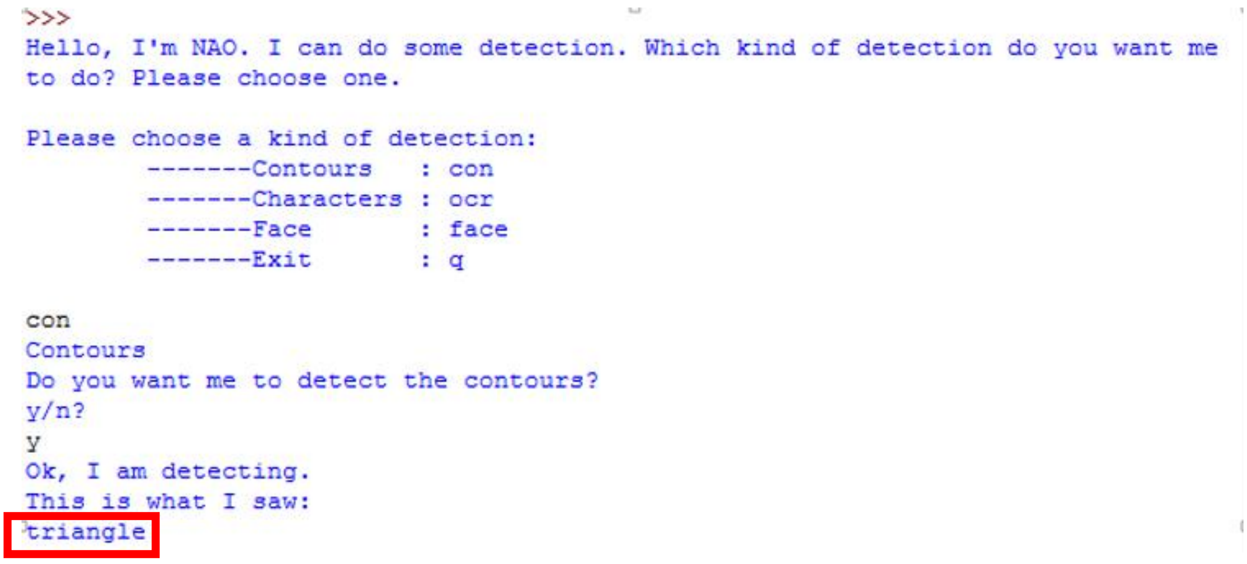


**Figure 36.** Comparison images of Rectangle

#### 3.3.1.3 Results of Contours Detection

The final image is the format that can be detected which means the end of the image process. Then the detection file will be called and the contours detected by the algorithm of detection.



```
>>>
Hello, I'm NAO. I can do some detection. Which kind of detection do you want me
to do? Please choose one.

Please choose a kind of detection:
        -------Contours   : con
        -------Characters : ocr
        -------Face       : face
        -------Exit       : q

con
Contours
Do you want me to detect the contours?
y/n?
y
Ok, I am detecting.
This is what I saw:
triangle
```

**Figure 37.** The output of the detection result

After the detection, the contour will be drawn on the original image.

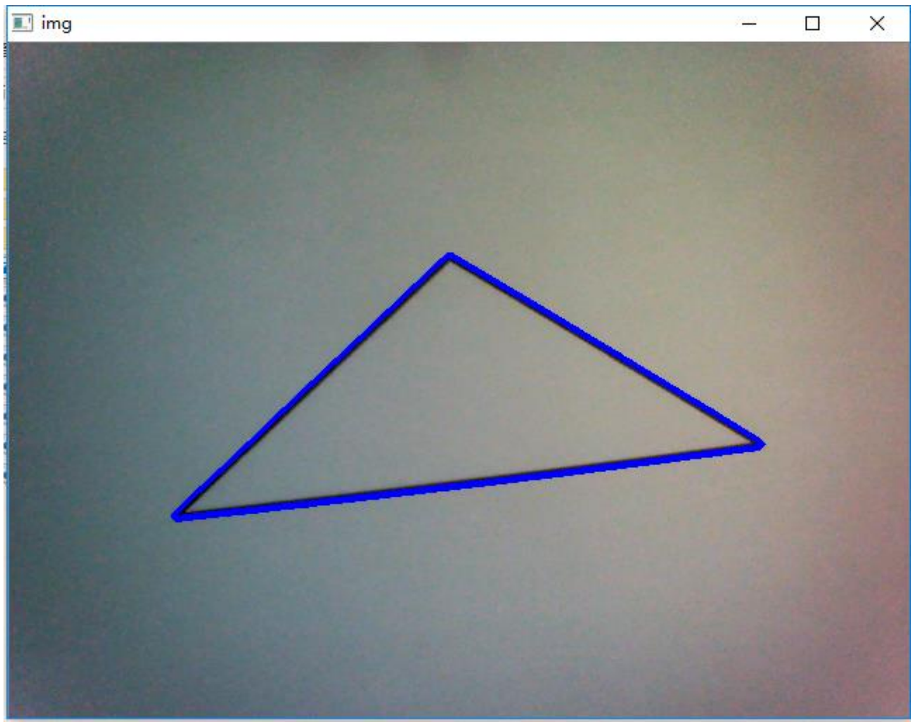


**Figure 38.** Result of the contour

Figure 39 is the detection results of line and rectangle.

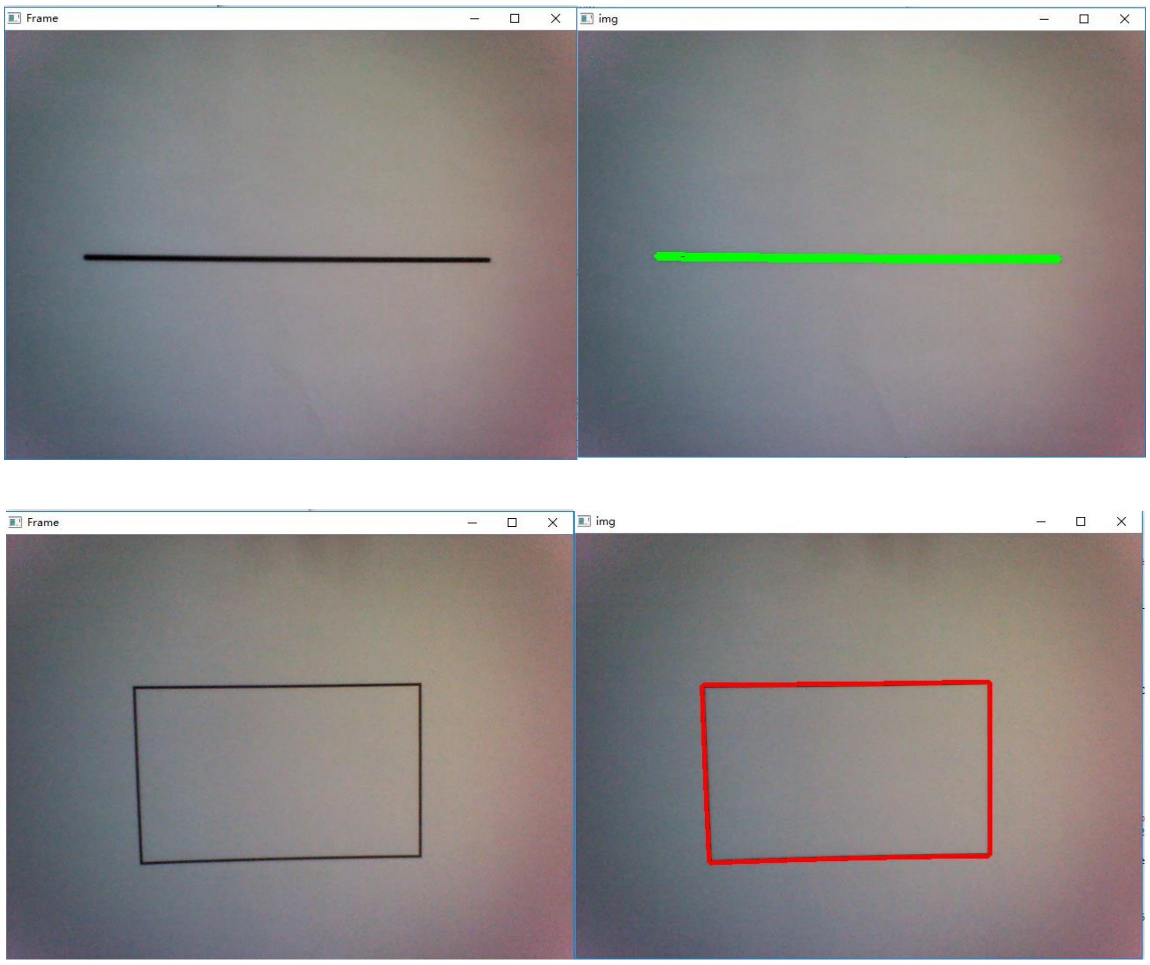


**Figure 39.** Results of line and rectangle

### 3.3.2 Character Recognition

#### 3.3.2.1 Flowchart of Preprocessing

The process for characters is very simple. Only a few steps are involved.

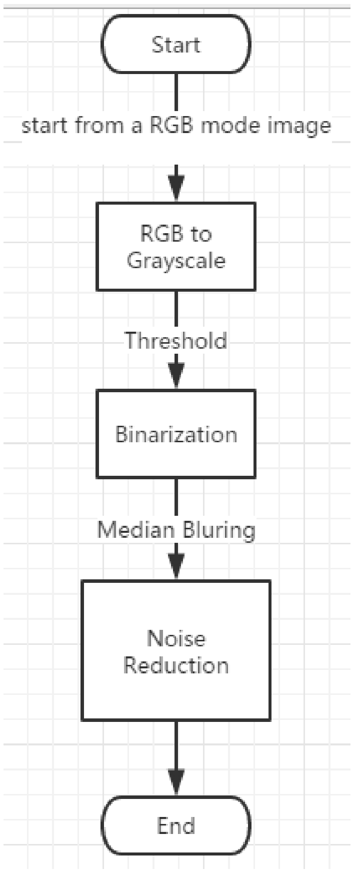


**Figure 40.** Flowchart of preprocessing (Characters)

### 3.3.2.2 Details and Results

Figure 41 is the character A which is saved on a local computer and 41 shows the result of processing.

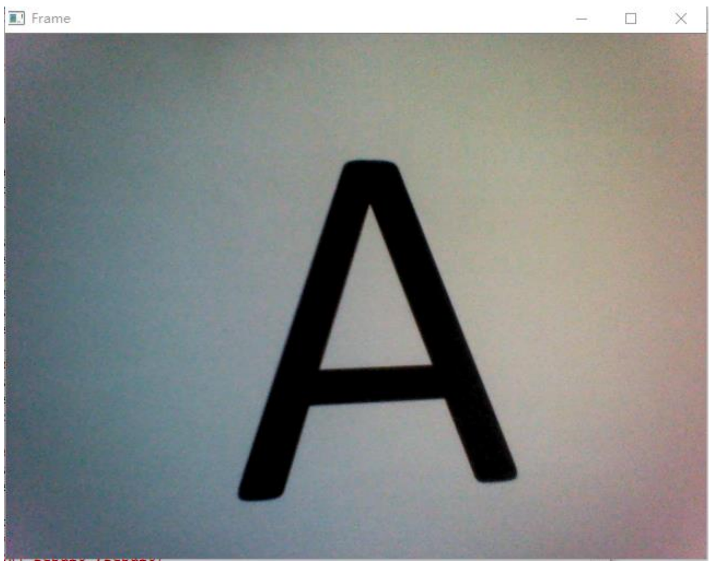


**Figure 41.** Saved on a local computer of Character A

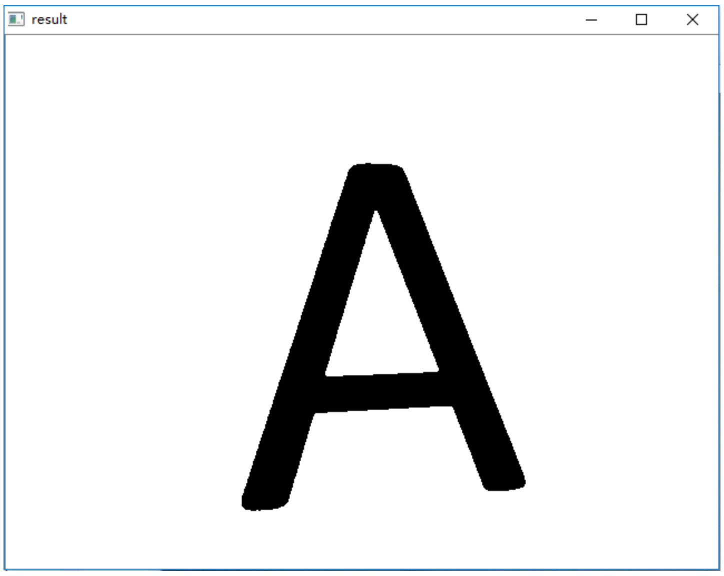
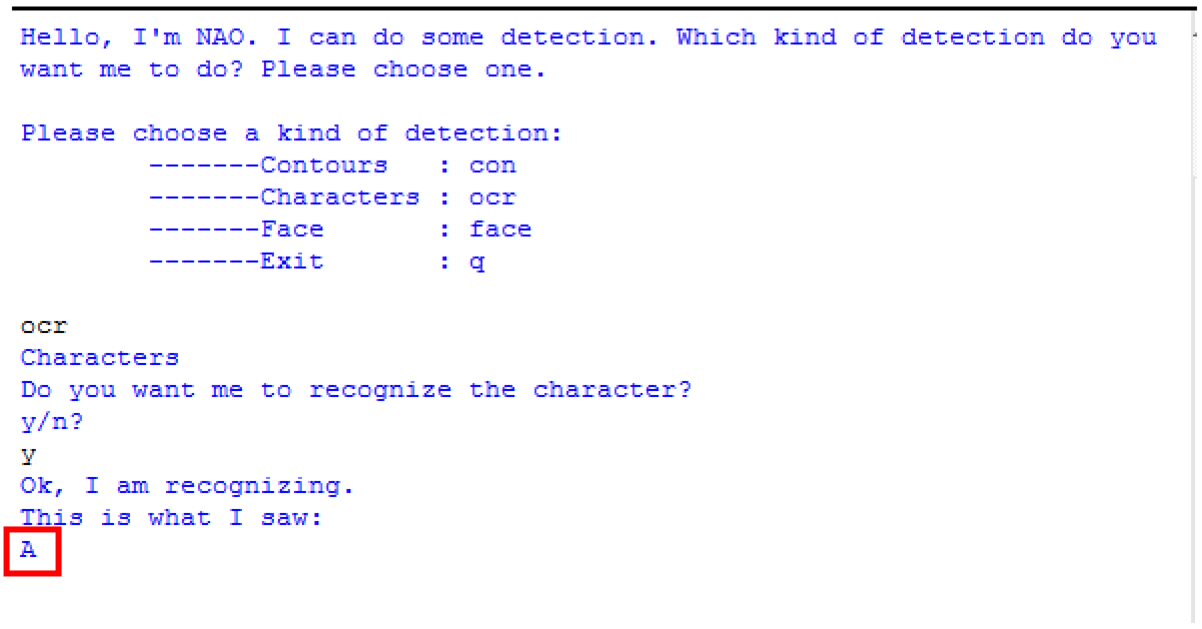


**Figure 42.** The result of processing

#### 3.3.2.3 Result of Character Recognition



```
Hello, I'm NAO. I can do some detection. Which kind of detection do you
want me to do? Please choose one.

Please choose a kind of detection:
        --------Contours   : con
        --------Characters : ocr
        --------Face       : face
        --------Exit       : q

ocr
Characters
Do you want me to recognize the character?
y/n?
y
Ok, I am recognizing.
This is what I saw:
A
```

**Figure 43.** The result of character recognition

### 3.3.3 Face Detection

#### 3.3.3.1 Flowchart of Preprocessing

The flowchart of the face is almost the same with that of contours. But before capturing one frame, the robot will detect faces itself and print the information on IDLE. After that, the robot will capture one frame and detect the face on the image and trim the face down. Then, the input of the image is only the face and the processing is for face only.

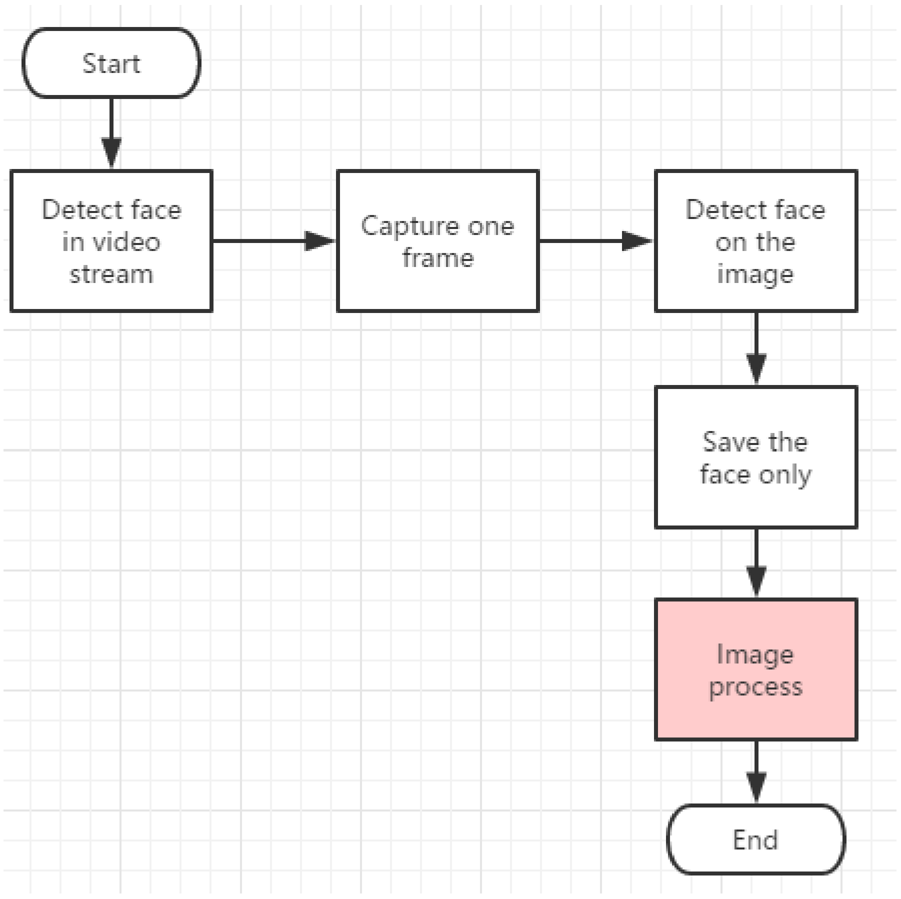
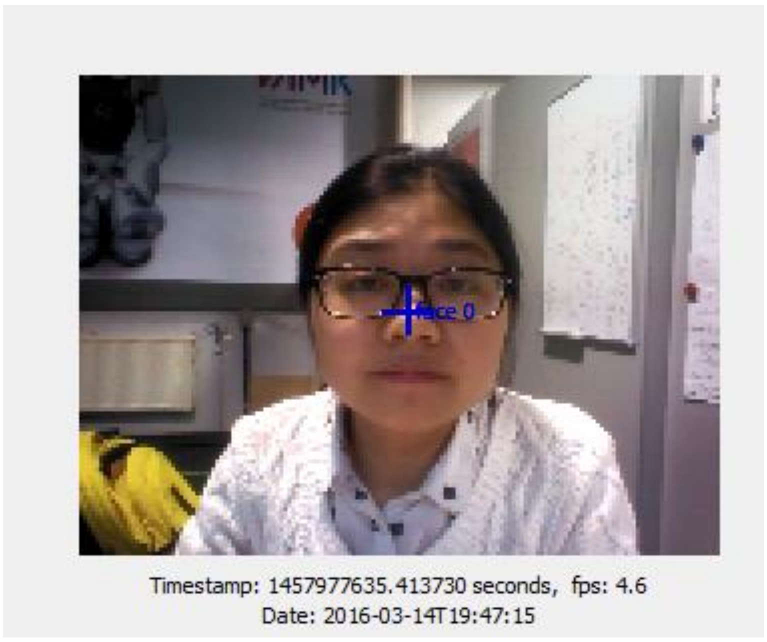


**Figure 44.** Flowchart on whole face detection

The red process is the same with that in contour detection.

**3.3.3.2 Details of Face Detection**

Figure 45 to 48 shows the process of proprocessing.



Timestamp: 1457977635.413730 seconds, fps: 4.6
Date: 2016-03-14T19:47:15

**Figure 45.** Detect face from real-time video stream

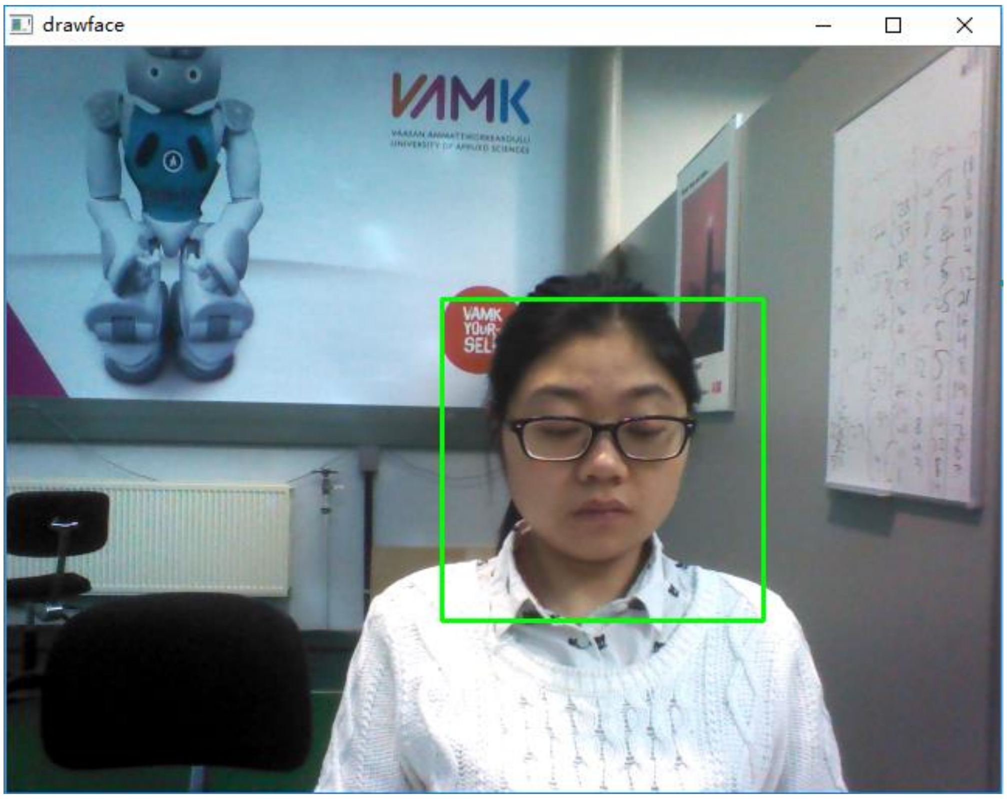


**Figure 46.** Detect face on the image

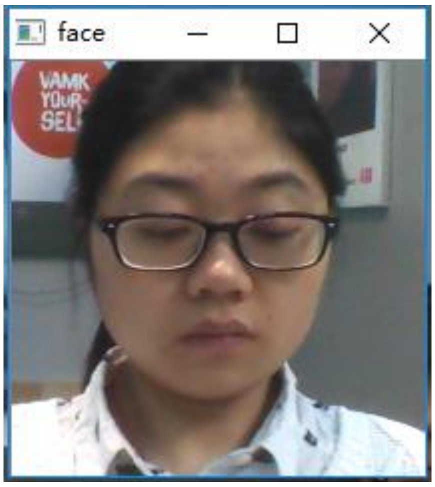


**Figure 47.** Only face

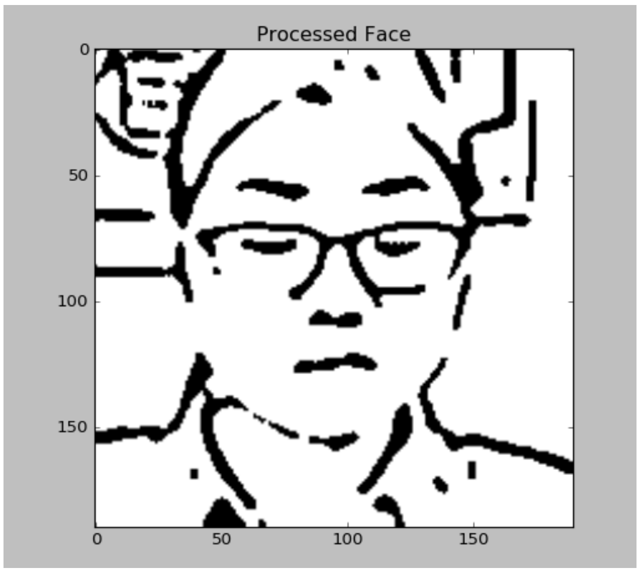
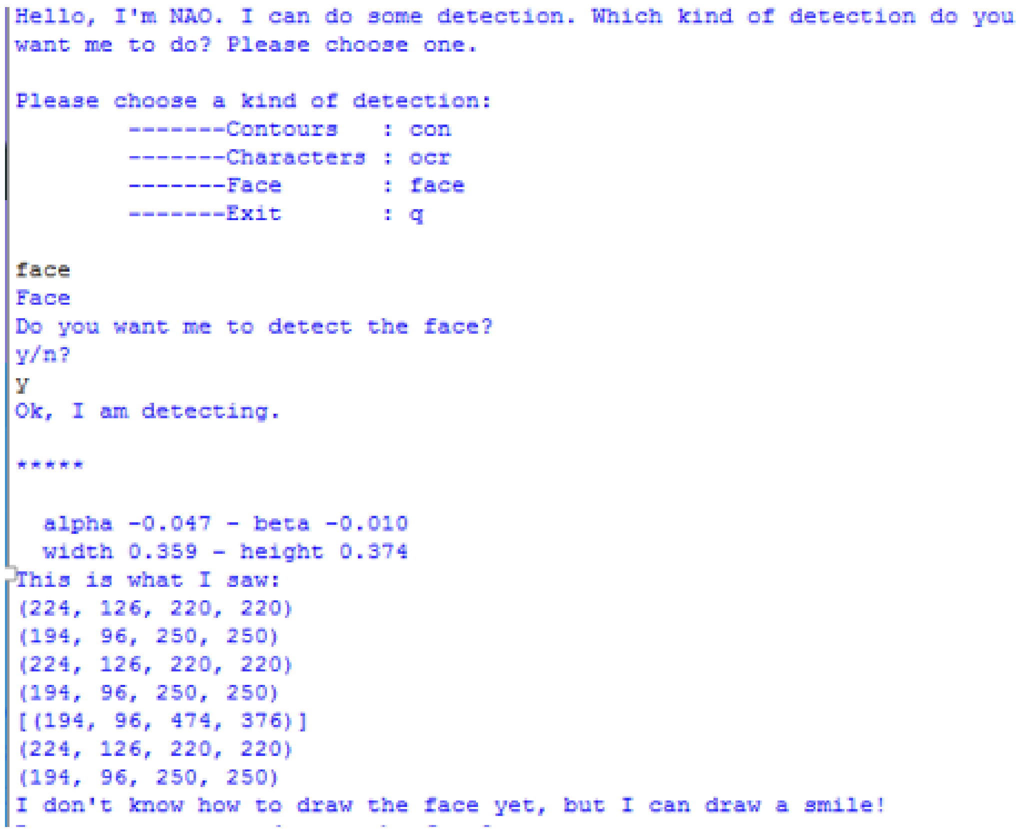


**Figure 48.** The result after preprocessing

### 3.3.3.3 Result of Face Detection



```
Hello, I'm NAO. I can do some detection. Which kind of detection do you
want me to do? Please choose one.

Please choose a kind of detection:
        -------Contours   : con
        -------Characters : ocr
        -------Face       : face
        -------Exit       : q

face
Face
Do you want me to detect the face?
y/n?
y
Ok, I am detecting.

*****

  alpha -0.047 - beta -0.010
  width 0.359 - height 0.374
This is what I saw:
(224, 126, 220, 220)
(194, 96, 250, 250)
(224, 126, 220, 220)
(194, 96, 250, 250)
[(194, 96, 474, 376)]
(224, 126, 220, 220)
(194, 96, 250, 250)
I don't know how to draw the face yet, but I can draw a smile!
```

**Figure 49.** The basic information of the face

# 4 MOTION MODULE

The motion module is the second topic area of the whole thesis. A drawing system will be built and through this drawing system the basic drawing of what the robot sees will be realized.

## 4.1 Hardware

When it comes to motion, joints cannot be bypassed. One joint links two parts of the robot's body. The body part that is close to the trunk is considered to be fixed and the other body parts that are far from the trunk are the ones that rotate around the joint axis. A frame is placed at each joint to perform the rotation of the body parts. When the robot is at the zero pose, all joint frames have the same direction. Roll rotations take place around the x-axis, pitch rotations around the y-axis and yaw rotations around the z-axis. /9/

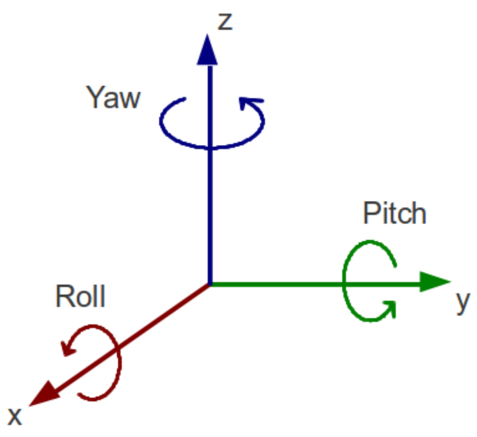


**Figure 50.** Rotations direction of joints /9/

When talking about the joints. Stiffness needs to be discussed**.** The stiffness of the joint is equivalent to torque limitations in the motors. If the joint stiffness is set to 0.0, the joint controller does nothing and the joint is free. With a value at 1.0 the joint is allowed to use full torque power to reach a given position. /10/

NAO robot has 24 joints that coordinate its movements. However, in this case, only one arm the right arm, will be needed. There are five joints on one arm. Each joint has its own effect and range angle. Figure 51 shows the position and range angles of the five joints in the right arm.

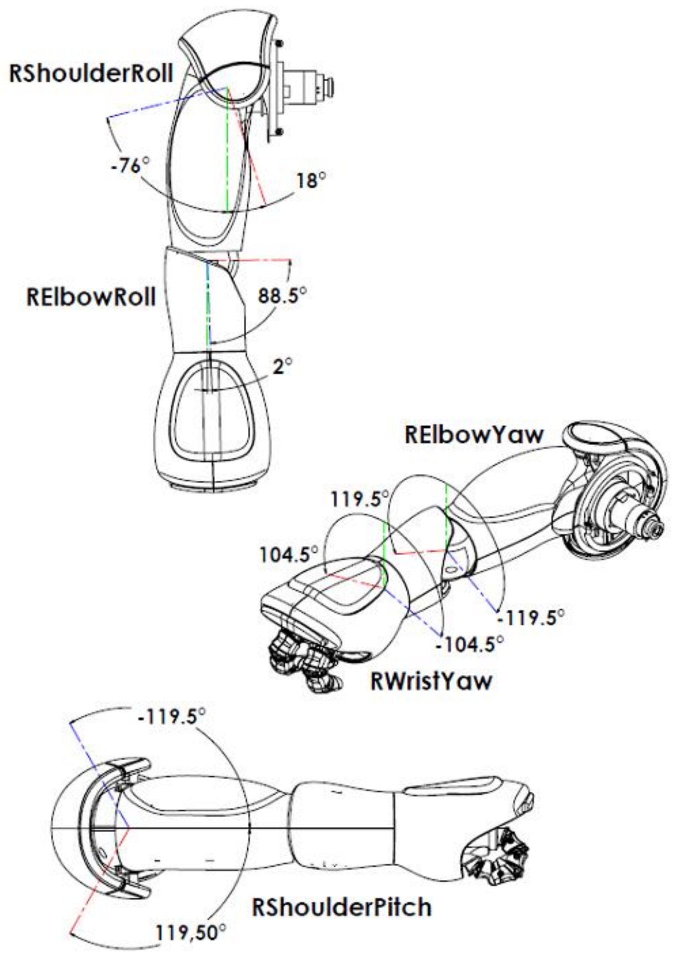


**Figure 51.** The position and range angle of each joint

## 4.2 Mathematical Model

Two 2D coordinate systems are built. One is the expectation locus of the graph, and the other one is the changes of the joints' angles. The two systems have the same x-axis, in which the physical unit lengths are not necessarily the same. In other words, the unit length of x is the same as the unit length of x', which means x and x' have the same meaning and in the unit length $\boxed{\text{ab = ab'}}$ in the Figure 52 below.

The axis x' can be any line at any angle on the plane of the x-y coordinate system.

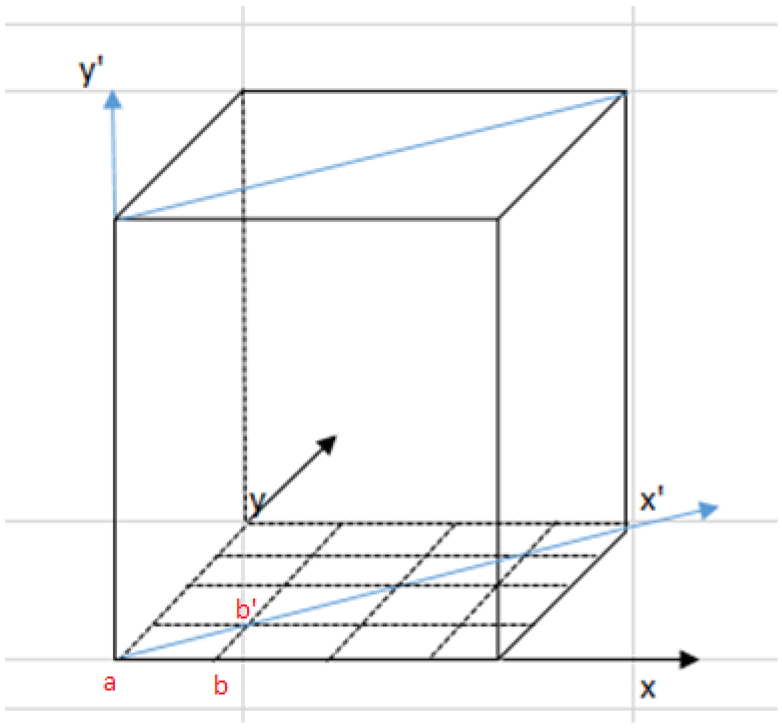


**Figure 52.** The model of the two systems

The coordinate in the horizontal direction (x-y) is the locus of the graph, and the coordinate in the vertical direction (x’-y’) is the locus of angles change of joints, which means the regular pattern of the joints movement.

Some points from the x-y coordinate system will be gathered and the corresponding angles when the robot arm moves to these points respectively will be got. Then through the locus, the regular pattern will be gotten, which means the function about changes in angles can be calculated.

Then some classes will be built, such as the class of line, the class of triangle, the class of rectangle, the class of character A etc. Through the mathematical model and algorithms, the robot will “learn” how to draw the graph in order to represent the class. For example, the robot will learn how to draw a triangle. This triangle represents the class of triangles, which means when it detects the geometric shape of a triangle, no matter what kind of triangle it is, (right triangle, acute triangle, or obtuse triangle; as long as it is a triangle), the robot will call for the drawing system to draw a triangle, which just represents the class of triangles.

As mentioned above, the whole module includes two 2D coordinate systems. One is x-y coordinate system on paper and the other one is x’-y’ on a space which is perpendicular to the paper. The space consists of innumerable planes that are perpendicular to the paper. It likes a 3D system, but it is not, it is just countless 2D coordinate systems which can be seen as the same one.

Figure 53 is the coordinate system on paper. It is used to gather the points on the locus of the drawing graph.

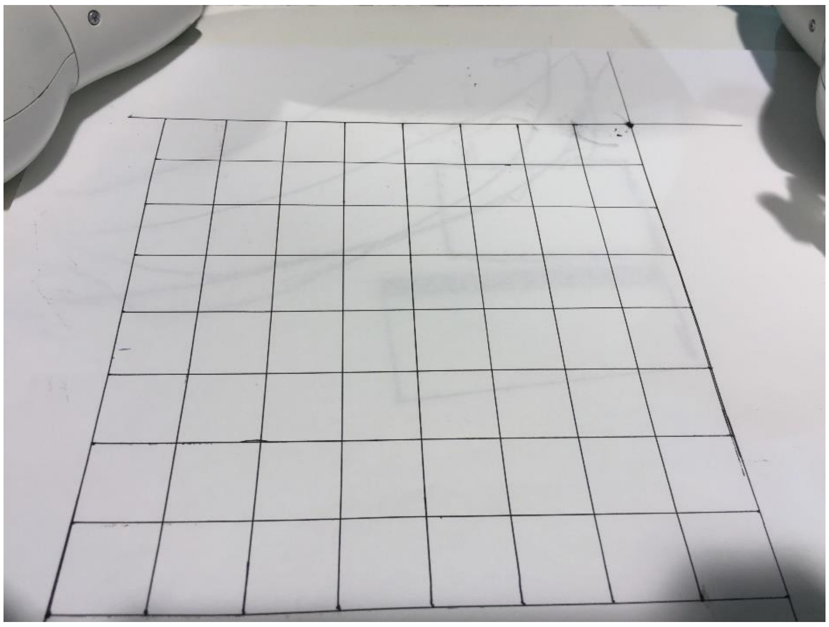
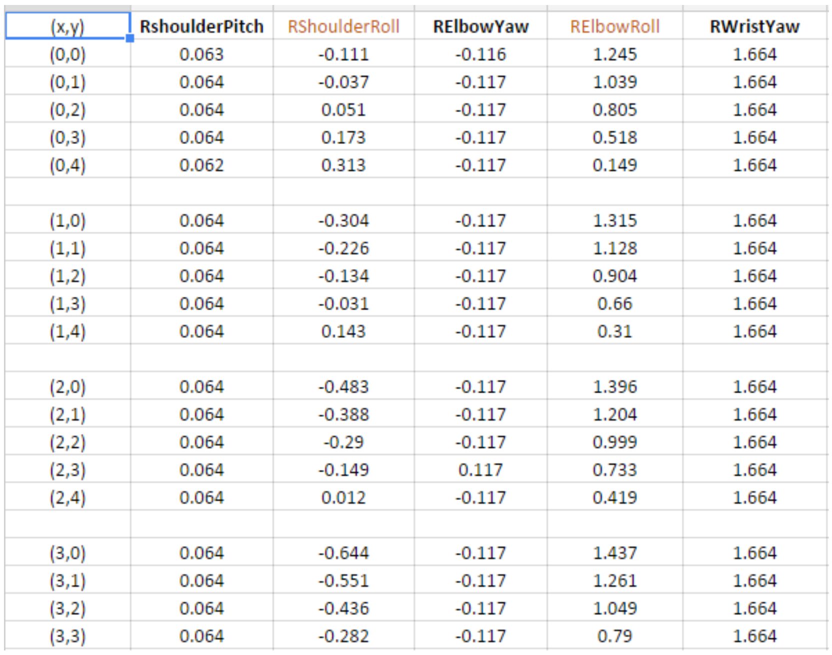


**Figure 53.** The coordinate system used to gather the points on the locus

Figure 54 is some of the points gathered. From figure it can be seen that three joints are almost fixed, which will make the calculation easier.



| (x,y) | RshoulderPitch | RShoulderRoll | RElbowYaw | RElbowRoll | RWristYaw |
|---|---|---|---|---|---|
| (0,0) | 0.063 | -0.111 | -0.116 | 1.245 | 1.664 |
| (0,1) | 0.064 | -0.037 | -0.117 | 1.039 | 1.664 |
| (0,2) | 0.064 | 0.051 | -0.117 | 0.805 | 1.664 |
| (0,3) | 0.064 | 0.173 | -0.117 | 0.518 | 1.664 |
| (0,4) | 0.062 | 0.313 | -0.117 | 0.149 | 1.664 |
| | | | | | |
| (1,0) | 0.064 | -0.304 | -0.117 | 1.315 | 1.664 |
| (1,1) | 0.064 | -0.226 | -0.117 | 1.128 | 1.664 |
| (1,2) | 0.064 | -0.134 | -0.117 | 0.904 | 1.664 |
| (1,3) | 0.064 | -0.031 | -0.117 | 0.66 | 1.664 |
| (1,4) | 0.064 | 0.143 | -0.117 | 0.31 | 1.664 |
| | | | | | |
| (2,0) | 0.064 | -0.483 | -0.117 | 1.396 | 1.664 |
| (2,1) | 0.064 | -0.388 | -0.117 | 1.204 | 1.664 |
| (2,2) | 0.064 | -0.29 | -0.117 | 0.999 | 1.664 |
| (2,3) | 0.064 | -0.149 | 0.117 | 0.733 | 1.664 |
| (2,4) | 0.064 | 0.012 | -0.117 | 0.419 | 1.664 |
| | | | | | |
| (3,0) | 0.064 | -0.644 | -0.117 | 1.437 | 1.664 |
| (3,1) | 0.064 | -0.551 | -0.117 | 1.261 | 1.664 |
| (3,2) | 0.064 | -0.436 | -0.117 | 1.049 | 1.664 |
| (3,3) | 0.064 | -0.282 | -0.117 | 0.79 | 1.664 |

**Figure 54.** Points gathered

Due to the limit of the length of the arm, only 35 points are gathered. These points form an area and in this area the points can make up many shapes. When the arm stays in one position, the corresponding angle of the joint can be recorded.

In the case of the triangle, the triangle is marked by red points on Figure 55.

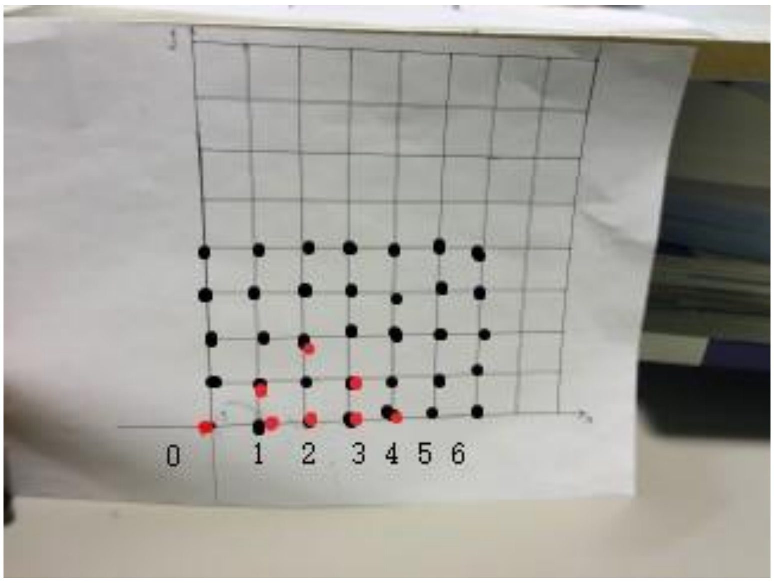


**Figure 55.** The locus of triangle

We can get the information from the figure that the coordinates three vertices are (0,0), (2,2), (4,0). We name each side as a, b, c.

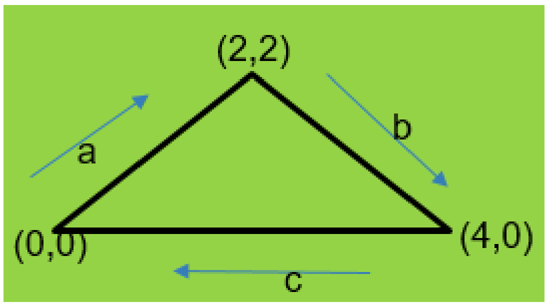


**Figure 56.** The triangle and the direction of the locus

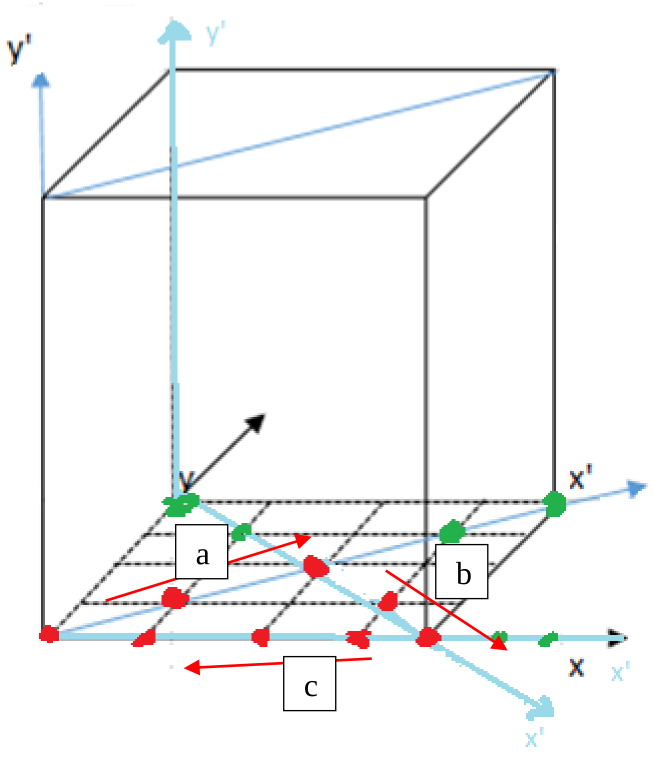


**Figure 57.** The triangle in the system

For side a, it has three exact coordinates (0,0), (1,1), (2,2). This means that, on this line, there are five known points, and the x-coordinates are 0, 1, 2, 3 and 4. As mentioned above, the other coordinate system has the same x-coordinates.

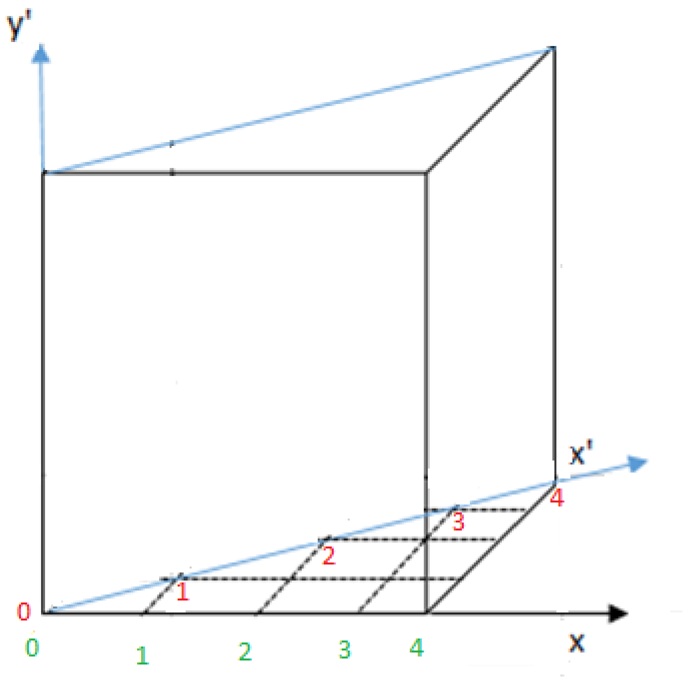


**Figure 58.** The two coordinate system have the same x-axis in unit length

From Figure 55 we get the information needed to calculate two joints RShoulderRoll and RelbowRoll. Figure 55 is the table of the joints' angle. This data, and the x-coordinates can be used to calculate the function of angle change. In case of RshoulerRoll, when the points of the x-y coordinate system are (0,0), (1,1), (2,2), (3,3) and (4,4), the x-coordinates on the x-axis, which is represented by x'-axis, are 0, 1, 2, 3 and 4. The joints' angle of corresponding points are

-0.111, -0.226, -0.29, -0.028, -0.295

The data is put into MATLAB in order to calculate the function.

```
>>> y = [0, 1, 2, 3, 4]

y      =

       0    1     2     3     4

>>> x = [-0.111, -0.226, -0.29, -0.028, -0.295]

x      =

       -0.111     -0.226     -0.29      -0.028      -0.295
```

```
>>> plot (x, y, 'o')
```

**Figure 59** shows figure be plotted of the five points.

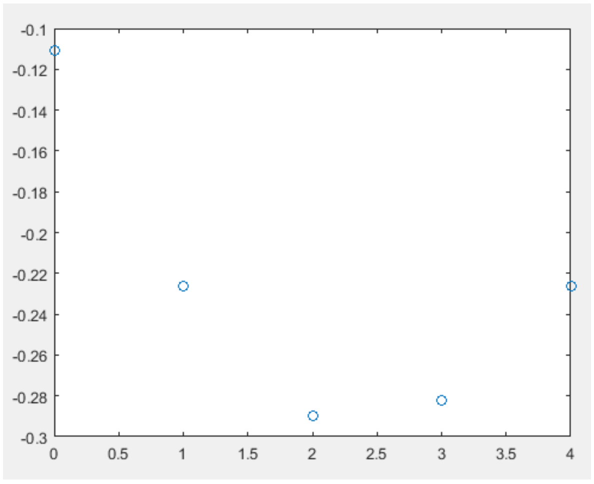


**Figure 59.** The distributed of the angles

From the figure above, the distribution of the angles looks like a quadratic function. So, MATLAB is used to fit the quadratic function.

```
>>> p = ployfit (x, y, 2)

p       =

            0.0296        -0.1469         -0.1107
```

So the equation of the quadratic function is

$$y = 0.0296x^2 - 0.1469x - 0.1107$$

This is for the joint RshoulderRoll. It can be checked by plot the function.

```
>>> x1 = [0 : 0.1 : 4]

>>> y1 = polyval (p, x1)

y1      =

            some data

>>> plot (x, y, 'o', x1, y1)
```

Then the figure of the equation will be known.

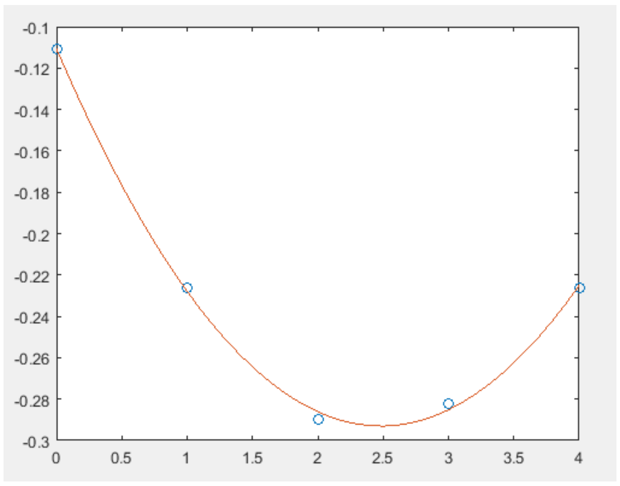


**Figure 60.** The graphic of the equation

It can be put in into the coordinate systems module.

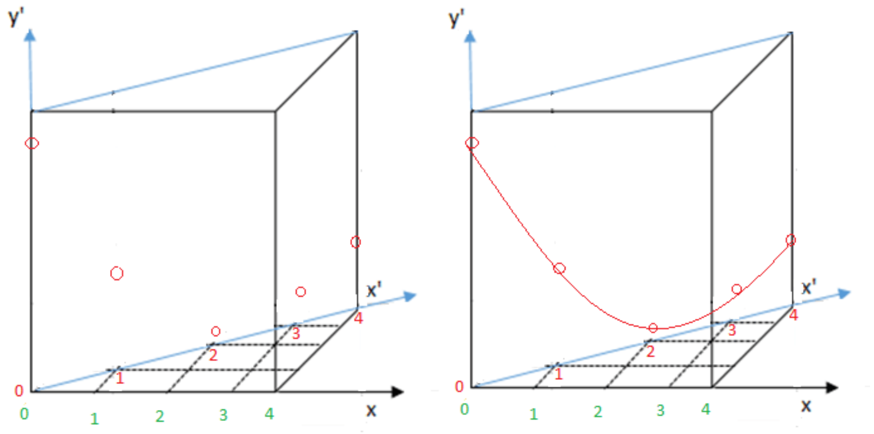


**Figure 61.** Put the graphic in the coordinate systems module
Use the same theory and method we can calculate the equation of the other joint RelbowRoll and the two joints that of side b and side c. However, for side c, the points should be taken oppositely, from (6,0) to (0,0), because related to the axis the motion of the arm is reverse.

```
#line a
for j in range(1,4):
    b = 0.0296*j**2-0.1469*j-0.1107
    d = -0.0283*j**2-0.0667*j+1.2384
```

RShoulderRoll

RElowRoll

The same theory and method are used in drawing characters and faces. However, when drawing faces, the robot has to learn how to draw on a graph in order to represent one class. Therefore, the robot just draws a simple smile instead of faces.

# 5 EXECUTION AND RESULTS

## 5.1 Execution

Because of the limitations of the hardware of NAO, the robot cannot grasp the open tightly, so a pen is tied to its hand.

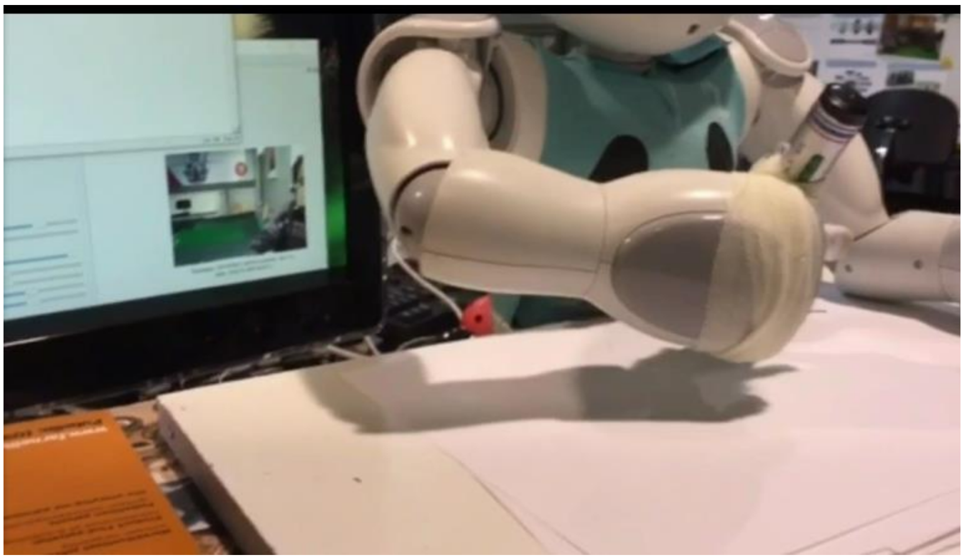


**Figure 62.** The robot is drawing

## 5.2 Results

In the thesis, the robot draws three different kinds of graphs, contours, characters and faces. There are three different contours, lines, triangles, and rectangles.

Figure 63 to 65 show the results:

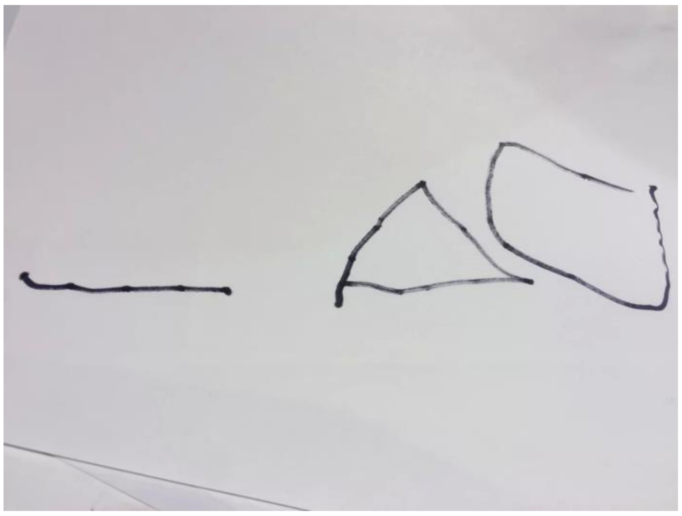


**Figure 63.** Results of contours drawing

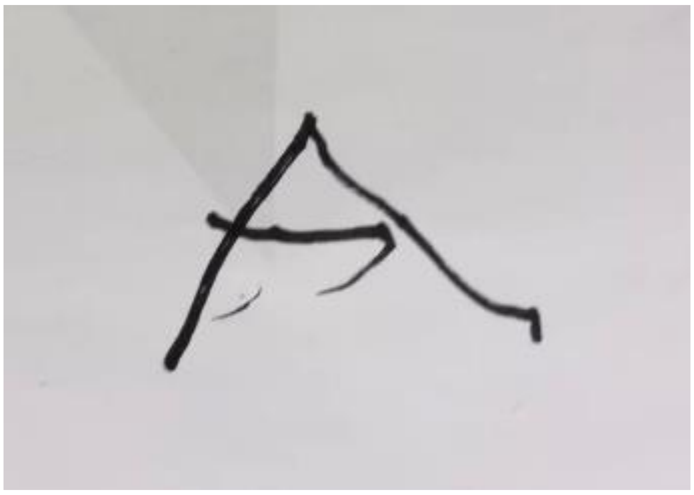


**Figure 64.** Result of character drawing

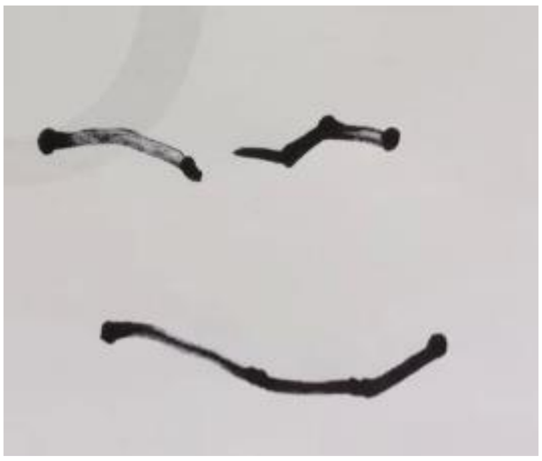


**Figure 65.** Result of smile drawing

# 6 FURTHER RESEARCH

In this thesis, the robot has already known the fixed drawing method which will recognize the object firstly and then calls the accordingly drawing files to draw.

In theory the robot should be able to draw different graphs when it sees different objects even the objects are in the same class. That means the robot needs to have the ability to connect the vectorization vision to the motion itself.

A method of vectorization is needed. This will be another research, by analyzing various questions in vectorization, bitmap vectorization will be proposed and realized.

Bitmap vectorization means convert the pixels of face contour into a series of connected points, and then detect the points on one line which is of each connected points. Save the coordinates of the first point of the line to the array and prepare for the connection with NAO robot.

Then the robot gets the information about the array and uses the array to calculate its motions. The image and the paper have the mapping relations, the coordinates in the image will be showed on the paper. That is in the theory. The robot will draw exactly what it sees.

# 7 SUMMARY

This thesis introduces the development of the vision system concerning contours, characters, and face detection and the basic drawing of contours, characters and faces with a NAO robot. Based on the development of the vision system of the robot, three different kinds of detections are discussed. Different detections use different methods for realization. In contour detection, the central idea is a vertex detector through the number of the vertex to recognize the geometric shape. Three geometric shapes are detected. In terms of character recognition, a technique called OCR is used to convert the image format into text format and the library Tessract is used to recognize it. In this thesis only one character is discussed as the instance. In terms of face detection, the robot can detect faces itself but it cannot do further processing. Thus, when capturing one frame from the real-time video stream, it must do another face detection and further process by OpenCV.

Another critical discussion of this thesis is the second main area, the motion module. Due to the limitations of the robot hardware, the drawing could not be very meticulous, and this is the main problem of this thesis. Even when running the same code, different results were got; the graphs drawn by the robot were not exactly the same each time. As mentioned when discussing further research, the further purpose of this thesis is to discuss how to build a stable and intelligent drawing system on NAO robot. However, this idea just stayed on a theoretical level. Then a method of making one drawing represent one class of drawing was used.

Collecting, recording and processing data was very cumbersome work. Sometimes when collecting the data for one side of the geometric shape, a large number of points had to be collected. It was very easy to make mistakes.

During the thesis work, there was a dead end concerning how to make the robot draw. Earlier in our studies, how to detect a line and draw down it in a project has been studied. Due to the limitations of the hardware, the robot was not stable enough, and the conversion from vision to motion was difficult. It cost me an additional period of time. However, when the direction of thinking was changed the drawing was realized on the robot successfully. There is a long way to go to build a stable and intelligent drawing system. Hopefully this can be achieved in the near future.

## REFERENCES

/1/ Relation between computer vision and various other fields.
https://en.wikipedia.org/wiki/Computer_vision#/media/File:CVoverview2.svg

/2/ NAO humanoid robot.
http://www.generationrobots.com/en/246-buy-the-humanoid-robot-NAO

/3/ NAO, A Humanoid Robot To Debut At Japan's Mitsubishi Bank. By Meera Dolasia on February 6, 2015.
http://www.dogonews.com/2015/2/6/NAO-a-humanoid-robot-to-debut-at-japans-mitsubishi-bank

/4/ NAOqi OS – user accounts
http://doc.aldebaran.com/2-1/dev/tools/opennao.html

/5/ Key concepts
http://doc.aldebaran.com/2-1/dev/naoqi/index.html

/6/ Aldebaran documentation
http://doc.aldebaran.com/2-1/family/robots/video_robot.html#robot-video

/7/ Aldebaran documentation
http://doc.aldebaran.com/2-1/naoqi/vision/alvideodevice.html#alvideodevice

/8/ ALVideoDevice API
http://doc.aldebaran.com/2-1/naoqi/vision/alvideodevice-api.html#ALVideoDeviceProxy::subscribeCamera__ssCR.iCR.iCR.iCR.iCR

/9/ Joints
http://doc.aldebaran.com/2-1/family/robots/joints_robot.html

/10/ Aldebaran documentation. Glossary.
http://doc.aldebaran.com/2-1/glossary.html#term-stiffness