Service Robot For Elderly and Disabled

Huynh Ngoc Anh
School of Electrical and Electronic Engineering

Dr Shen Zhiqi
School of Electrical and Electronic Engineering

Assoc Prof Miao Chunyan
School of Computer Engineering

Abstract **-** This robot combines advantages of Wi-Fi and robotics technologies to provide daily-life services to people staying at home, the disabled and the elderly. Some of its numerous applications include direct interactions and remote visits to distant friends, relatives and doctors, remote telemedicine, daily reminders, and more. Developed using C# programming language, the robot implements all the basic functions needed by most automatic guided vehicles, such as: real time observation, management and remote control.

Keywords - robotics technology, disabled, elderly

1 INTRODUCTION

Helping elderly and disabled is one of the most important applications of surveillance robots. The robots of this kind have to have some basic functions such as: camera to observe what is going on in surrounding area, algorithm to follow a person and capability to help the people to talk with distant relatives. The complex environment is fairly challenging to this kind of robot. There are many reasons for this, eg. rough terrain, unstructured object and motion of different objects in the surrounding area. Therefore, object tracking capability and estimation for potential collision avoidance are the most essential skills that a robot needs to fulfill the requirements.

There are many different algorithms that can be used to develop tracking capability. This project introduced the two most common algorithms: color-based and human-face-based. In the color-based algorithm, the object is identified by its unique color characteristics and the robot can automatically navigate by itself to keep the distance and angle to the object in certain ranges. Similarly, in the human-face-based algorithm, the face of a person is tracked and the robot can adjust itself so that the distance as well as the angle is in range.

Due to time constraint, this project focuses on the capability of following a human using a single camera. Further applications can be developed based on this work such as remote visits, telemedicine…

2 LITERATURE REVIEW

Developing service robots for elderly and disabled is the subject of active research since the elderly population is increasing very fast in the world due to better health care system. Different companies have put lots of effort to come out with more and more effective robots. Robosoft has recently introduced a robot that make use of the advantages of internet and robotics technologies to provide daily services to people staying at home, the disabled and the elderly [1]. It possesses numerous functions and some of them include social interactions and remote visits with relatives and doctors, cognitive prosthesis, daily reminders, and more. It is based on robuLab10, a multi-purpose mobile robot designed to embed various "application modules", and a robuBox, the generic robot middleware based on Microsoft Robotics Studio, that comes with every robot produced by Robosoft. The robuBOX-AGV is developed using Microsoft Robotics Studio and implements all the basic functions needed by most automatic guided vehicles, such as: anti-collision, path teaching-and-repeating, service orchestration and remote control (by phone).

Another remarkable example is the one that was developed by Carnegie Mellon University, Pittsburgh [2]. Its aim is to develop a service robot based on a systematic software engineering method, particularly for real-time, embedded and distributed systems with UML. They have applied the COMET method, which is an UML-based method for the development of concurrent applications, specifically distributed and real-time applications. They describe their purpose is to apply the COMET/UML method in developing the service robot for the elderly, T-Rot, which is under development at CIR. Here, their main focus was on an autonomous navigation system for the service robot; this is one of the most challenging issues and is essential in developing service robots, especially mobile service robots to help disabled people. It includes hardware design for the integration of various sensors and actuators as well as software development and integration of modules like a path planner and a navigator.

One common feature that all service robots have to possess is object tracking capability. There are many different algorithms that can be used to realize the tracking capability of robot. One of the famous algorithms is to track motion of the object [3]. This algorithm uses different filter and complex techniques to extract motion of the object being tracked from motion of background; hence, for real-time application, its implementation requires powerful hardware that may not be suitable for small robot with limited processing power. Other two simpler algorithms that are more suitable for small scale project are color-based algorithm and human-face-based algorithm. These two algorithms [4,5] detect an object based on its unique color characteristics or shape. With the time constraint and the purpose of prototyping, the two most common algorithms, color-based and human-face-based, will be implemented and experimented. They are simple yet effective and suitable for low processing power robots.

3 APPROACH

The overall design of this implementation comprises of three main components: robot platform with a single camera integrated, communication link with a Wi-Fi module used and computer upon which the main program executed.

The prototype robot platform was built using Programmable Integrated Circuit Microcontroller, Maxim Max 3222 chip and a commercial robot platform [6]. It is shown in figure 1. It has two controllable wheels that are controlled to make the robot move in four directions. A power program was burned into the Microcontroller to handle all the instructions and commands in order to allow the robot to be fully automotive. The robot can be controlled wirelessly by a remote computer upon issuing commands that set the robot’s movement direction, speed and duration accurately.

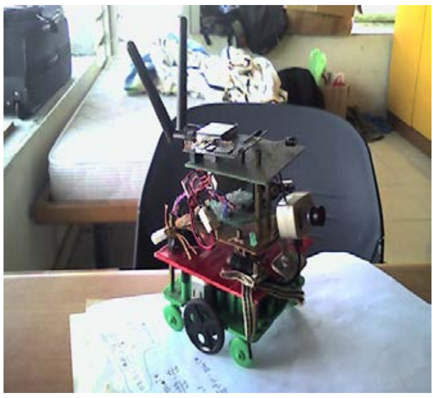


Figure 1: Robot platform

A Wi-Fi module (Digi Connect Wi-EM) is used as a server socket to link robot and computer via a Wi-Fi router [7]. It is shown in figure 2. Once the module is configured, it can be operated with a power source of 3.3 volts. With a wireless router serving as an “instruction transmitter”, the Wi-Fi module can then receive instructions from the router and proceed to initiate robot movement. The module was configured to have IP Address of 192.168.0.100 and Port Number of 2101 initially. These parameters can be changed easily through a web browser by accessing the web server hosted on the Wi-Fi module.

Figure 2: Wi-Fi module

C# programming language is used to write the main software that coordinates the robot. Aforge.NET library is used to detect color and Emgu.NET library is used to detect human face. Once the object, whether it is identified by its color or it is a human face, is detected, the software then estimates its relative position on the image frame. From this position information, the robot is controlled to move and consequently adjust position of the object on the image frame so that it is relatively at the center. This is called auto-correction mechanism. For example, if the object is too far from the camera, it appears to be too small in image frame. Once the software is awared of this fact, it controls the robot to move forward to reduce the distance and make the object appear to be bigger. If the object is too near from the cameara, the object appears to be too big on image frame and the software controls the robot to move backward to increase the distance and make the object appear to be smaller. Similarlly for the case when the object appears to be inclined whichever left or right side on image frame. This software component is where the author put most of the effort.

The control flow diagram goes as follow. The camera mounted on the robot starts by capturing the scene in front of the robot and sends the image frame back to the receiver. The signal received at the receiver is analog hence it must be converted to digital signal before being processed by laptop. This is done by an Analog-to-Digital converter. The main program on the laptop then scans through the image for the object being detected. Once the position of the object has been determined, the laptop sends commands to control the robot so that the object always appears relatively at the center of the image frame. The graphical controlling mechanism is shown in figure 3.

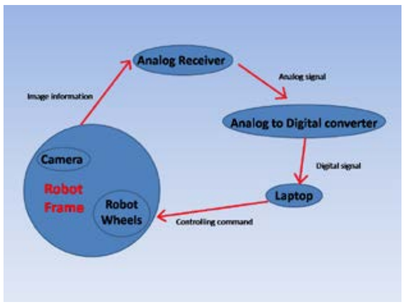


Figure 3: Control flow diagram

4 SOFTWARE DESIGN

The software component comprises of two parts: image processing and control unit. In the image processing part, the tracked object is identified either by its unique color characteristics or by its shape (human face). Different filters, namely RGB filter, HSL filter and grayscale filter [8], are deployed in the former algorithm consecutively to filter out the desired object based on its color composition. Output of these filters is then processed by blob counter algorithm to come out with location of the object being tracked. On the other hand, human face can also be the object of tracking. In this case, the image frame is processed directly by Haar-like algorithm [9] to locate any human face appears on image frame.

In the control unit, location of the object that is determined in the image processing part is used as input to correct position of the object. It controls the robot so that the location of the object always appears to be at the center of image frame.

4.1 IMAGE PROCESSING

4.1.1 Color-based algorithm

Firstly, the RGB filter is used to filter out the desired object in Red-Green-Blue color space. It uses RGB values to focus the attention towards the primary RGB colors. Depending on the color selected this filter will diminish all pixels that are not of the selected colors. This function is different than RGB channel in that white pixels are also diminished even though they may contain the color selected.

For example, if Red is chosen:

R = ((R-B) + (R-G))

G = 0

B = 0

R is then normalized with respect to the maximum red value.

Based on the above formula it can be seen that white pixels result in a zero value whereas pure primary colors (R=255, G=0, B=0) R doubles its value. Thus function does a better job than RGB channel in filtering for a particular color as white light is removed.

Due to normalization, really dark pixels can be elevated in intensity and generate too much noise in the resulting image. The Min Pixel Value allows specifying a minimum value below which pixels are considered to be black and will be ignored when calculating the image results.

Secondly, HSL filter is used to filter out the desired object in Hue-Saturation-Light color space. This model is originally a hexagonal cone and is shown graphically in figure 4. [10]

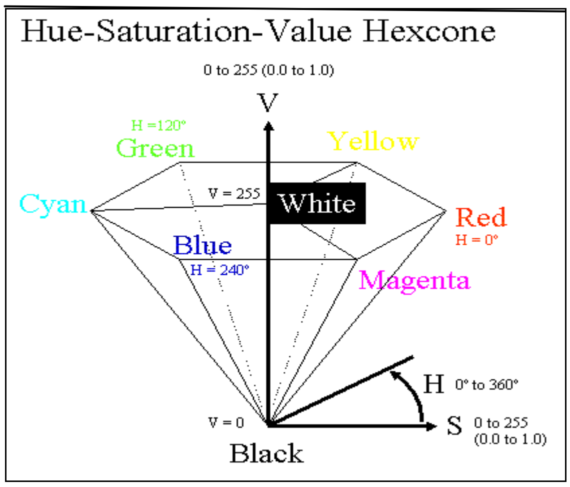


Figure 4: HSL color space

The vertical axis is the Value (Lightness) in the range 0.1

The angle is the Hue in the range 0.360

The relative radius is the Saturation in the range 0.1

In HSL color space, *Hue* is measured by an angle with Red starting at 0 degrees, Green at 120 degrees and Blue at 240 degrees. Complementary colors are in-between: Yellow is at 60 degrees, Cyan is at 180 degrees, and Magenta is at 300 degrees. The black line in the diagram at the lower left on the screen shows the hue angle.

Saturation, S, is a ratio that ranges from 0.0 along the center line of the cone (the V axis) to 1 on the edge of the cone. The gray circle above in the diagram at the lower left of the screen shows the position of the saturation value. The colors along this saturation circle are shown in the color box above the hue trackbar. The colors along the saturation radius are shown in the color box above the saturation trackbar.

Value (Lightness), V (L), ranges from 0.0 (dark) to 1.0 (bright).

Thirdly, after going through the two filters described above, the image was filtered to produce grayscale image before going through blob counter. This step essentially effects speed of the blob counter as it converts processed image frame to grayscale space and speed up the blob counter.

Fourthly, blob counter counts objects in image, which are separated by black background. The algorithm treats all pixels with values less or equal to BackgroundThreshold as background, but pixels with higher values are treated as objects' pixels.

For blobs' searching, the class used supports 8 bpp indexed grayscale images and 24/32 bpp color images.

The processing sequence is graphically described below:

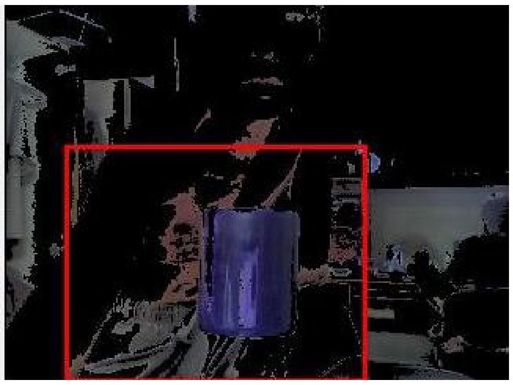


Figure 5: Image frame after being processed by RGB filter

Output of RGB filter is shown in figure 5. As seen from the figure, the area that contains the object is surrounded by the red rectangle. In figure 6, this area is further reduced. The final image is then converted to grayscale space before being processed by blob counter algorithm.

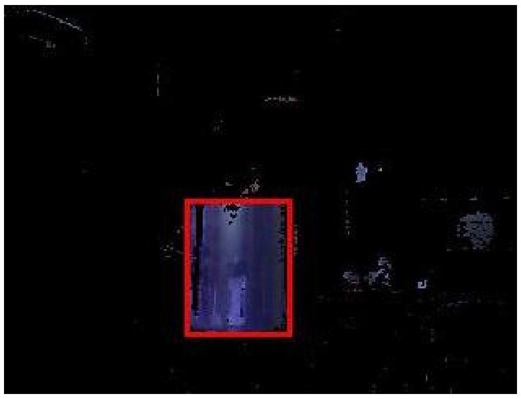


Figure 6: Image frame after being processed by HSL filter

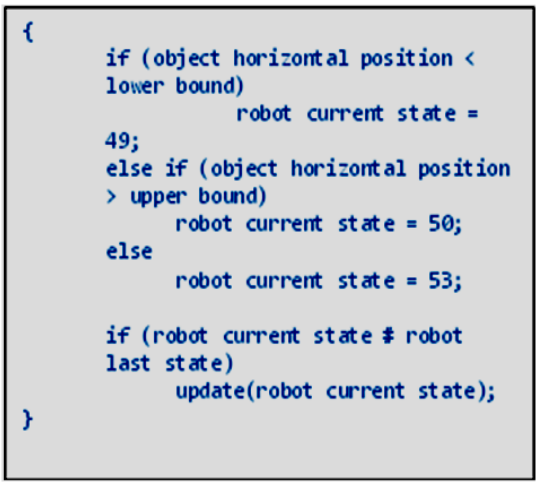
4.1.2 Human-face-based algorithm

There are many different approaches that can be used for face detection purpose such as skin-based approach, shape-based approach and facial-appearance-based approach. HaarCascade algorithm, which is used in this project, is a special technique that utilizes Haar-like features in recognizing human face in the image frame. The algorithm works by finding rectangular regions in the given image that are likely to contain objects that the cascade has been trained for and returns those regions as a sequence of rectangles. It scans the image several times at different scales. Each time it considers overlapping regions. It may also apply some heuristics to reduce number of analyzed regions, such as Canny prunning. After it has proceeded and collected the candidate rectangles (regions that passed the classifier cascade), it groups them and returns a sequence of average rectangles for each large enough group. More detail can be found at materials specializing on HaarCascade algorithm.

4.2 CONTROL UNIT

This piece of software plays the role of manipulating the robot to adjust the position of the object as appearing in the current image. The object that is being detected is supposed to be at center of the current image.

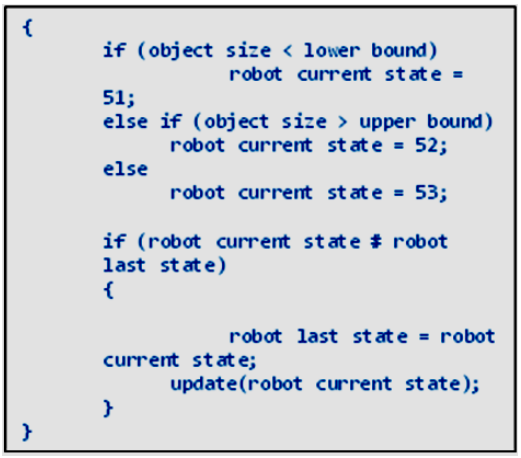
The controlling mechanism begins by checking the horizontal position of the object being detected. If it is lower than a lower bound, the computer will issue a command to turn the robot left to correct the horizontal position. If it is higher than an upper bound, the computer will issue a command to turn the robot right to correct the horizontal position. If it is in the desired range, the computer will issue a command to stop the robot.



```
{
        if (object horizontal position <
        lower bound)
                robot current state =
        49;
        else if (object horizontal position
        > upper bound)
             robot current state = 50;
        else
             robot current state = 53;

        if (robot current state ≠ robot
        last state)
             update(robot current state);
}
```

After correcting the horizontal position of the robot, the size of the object is checked and the robot is controlled to move backward and forward to correct the size with similar mechanism.



```
{
        if (object size < lower bound)
                robot current state =
        51;
        else if (object size > upper bound)
             robot current state = 52;
        else
             robot current state = 53;

        if (robot current state ≠ robot
        last state)
        {
                robot last state = robot
        current state;
             update(robot current state);
        }
}
```

The “robot current state” variable is to keep track of the current state of the robot that is determined in the processing part and based on the current image. The “robot last state” variable is used accompanying with the “robot current state” variable to avoid sending duplicate messages to the robot.

Another programming issue that needs to be considered is the network programming. C# simplifies the network programming through its namespaces like System.Net and System.Net.Sockets. A Socket is an End-Point of To and From (Bidirectional) communication link between two programs (Server Program and Client Program) running on the same network. We need two programs for communicating a socket application in C#. A Server Socket Program (Server) and a Client Socket Program (Client). The communication model is shown in figure 7.

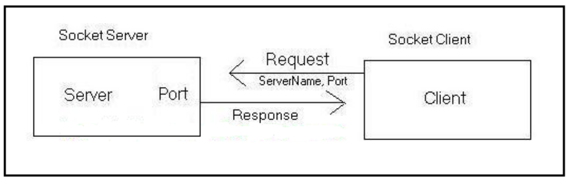


Figure 7: Socket communication model

- A C# Server Socket Program: A C# Server Socket Program running on a computer has a socket that bound to a Port Number on the same computer and listening to the client's incoming requests.
- A C# Client Socket Program: A C# Client Socket Program have to know the IP Address (Hostname) of the computer that the C# Server Socket Program resides and the Port Number assign for listening for client's request (The Wi-Fi module was configured to have IP address 192.168.0.100 and Port Number 2101)

5 CASE STUDY

Most of the computation was performed on laptop having Intel Dual core chip 2.0 GHz. Low resolution (320x240 pixels) input images were chosen for real-time response, and the tracking algorithm was able to process 15 frames per second. Snapshots of the experiment setup using colour-based algorithm are shown in Figure 8. A circular blue lid was used as the object being tracked. Figure 9 shows the object detected by the main program on laptop. This is actually camera view or what is seen by the robot.

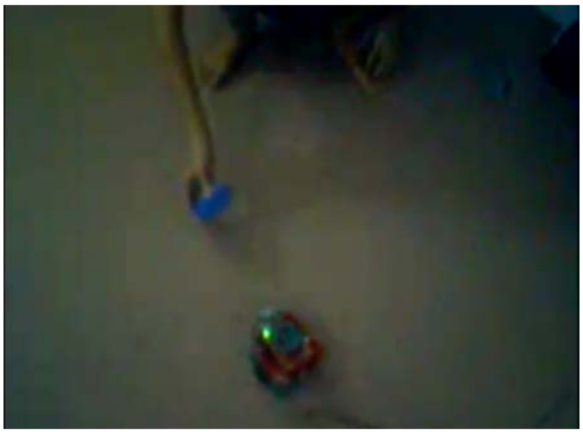


Figure 8: Demonstration using color-based algorithm.

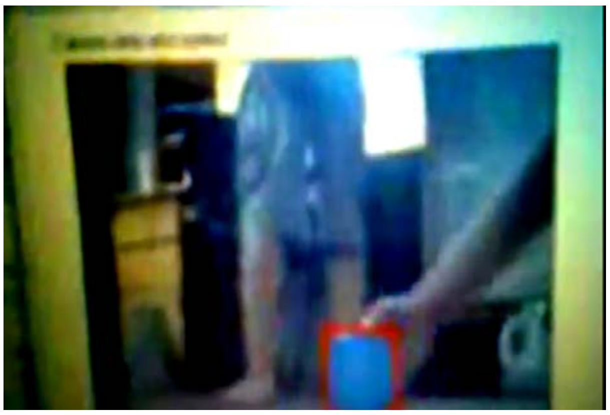


Figure 9: Laptop screen snapshot when the robot is under operation.

6 MOTION-BASED DETECTION

Detecting motion of external objects from a moving robot is the subject of active research. There are two independent motions involved: the motion of the robot (and hence the camera it carries) and the motions of moving objects in the environment. Unfortunately, those two motions are blended together when measured through a sensor such as a camera. In order for a robot to detect moving objects robustly, it should be able to decompose these two independent motions from sensor readings.

Frame differencing, which compares two consecutive images frames and deduce motion from the difference is the most intuitive and fast algorithm. This approach is especially effective when the camera is static as there is no noise from surrounding environment. However, this straightforward approach is not applicable in the case of moving camera because a big difference is generated by slightly moving the camera even if nothing moves in the environment. There are two independent movements involved in the moving camera scenario: ego-motion of the camera and motion of moving objects. Since these two kinds of motion are mixed in a single image frame, the ego-motion of the camera should be eliminated to filter out the motion of the moving objects. The following algorithm is introduced to do the filtering job. Frame differencing is utilized but the ego-motion of the camera in previous image frame is compensated before comparing it with the next image frame.

Firstly, a feature selection algorithm was used to produce an initial feature set. The Lucas-Kanade method was then applied to track those features in the subsequent image. Once the correspondence feature sets $<f_{\_t-1}, f_t>$ is known, the ego-motion of the camera can be estimated using a transformation model and optimization method, namely the bilinear model. Figure 10 [3] shows the final result image with the human body detected based on its relative motion with the bacground, and figure 11 [3] shows the output of the adaptive filter.

Once motion has been identified, objects in the scene can be tracked easily.

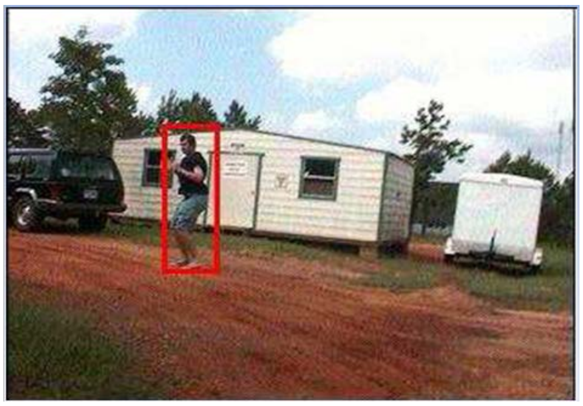


Figure 10: Real scene, the motion of the person being tracked.

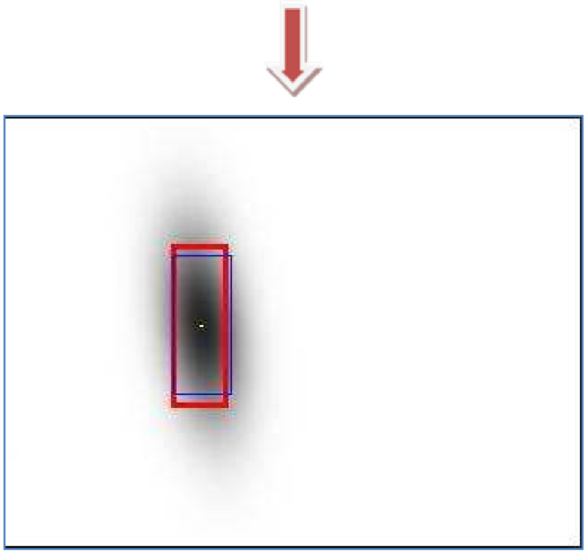


Figure 11: Image through adaptive filter.

7 CONCLUSION

A robust real-time algorithm for object detection was introduced for an outdoor carrying a single camera. The relative position of the object, with respect to the robot, was estimated using its average horizontal position and its size. The performance statistics shows the current system can detect moving object with unique characteristics and shows a basic tracking capabilities.

Further work is required to develop an algorithm to help the robot to track general moving object that does not base on its color or its shape. This is still under developing due to time limitation.

As of now, the robot can follow any object basing on its color characteristics. Due to time constraint, the completed implementation of the human-face-based algorithm has not been tested. Furthermore, more efforts should be done to develop applications in helping elderly and disabled such as reminder, remote talking which this project, due to time constraint, is not successful in fulfilling.

ACKNOWLEDGEMENT

I would like to show my gratitude to my supervisor, Dr Shen Zhiqi, who has advised and guided me to accomplish this project.

I also would like to thank Liu Yong for his advices on the process of completing the project

On top of that, I also wish to acknowledge the funding support for this project from Nanyang Technological University under the Undergraduate Research Experience on Campus (URECA) program.

REFERENCES

[1] Robosoft's robot for elderly and disabled: http://spectrum.ieee.org/automaton/robotics/medical-robots/robosoft-kompai-robot-assist-elderly-disabled

[2] Robot developed by CMU, Pittsburgh http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4799445

[3] Motion tracking

http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.99.3742

[4] Color detection algorithm

http://yekmer.posterous.com/object-detection-by-color-using-aforgenet

[5] Face detection algorithm

http://www.emgu.com/wiki/index.php/Face_detection

[6] Final year project: A Robot Platform Design for Wireless Controlling Via WIFI, HONG KAH HUNG WELSON, Nanyang Technological University, 2008/2009.

[7] Final year project: Action-based communication tool, Ho YongFu, Nanyang Technological University, 2008/2009.

[8] Different color space analysis

http://www.imagecomponent.net/products/pc_hsl_adjustment.aspx

[9] Haar-like features

http://en.wikipedia.org/wiki/Haar-like_features