<u>ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY</u>

HUMAN ASSISTED HUMANOID ROBOT PAINTER

M.Sc. THESIS

Cemal GÜRPINAR

Department of Computer Engineering

Computer Engineering Programme

JUNE 2012

<u>**ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY**</u>

HUMAN ASSISTED HUMANOID ROBOT PAINTER

M.Sc. THESIS

Cemal GÜRPINAR
(504091509)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Asst. Prof. Dr. Hatice KÖSE

JUNE 2012

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

İNSAN DESTEKLİ İNSANSI ROBOT RESSAM

YÜKSEK LİSANS TEZİ

Cemal GÜRPINAR
(504091509)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Yrd. Doç. Dr. Hatice KÖSE

HAZİRAN 2012

Cemal GÜRPINAR, a **M.Sc.** student of ITU **Graduate School of Science Engineering and Technology 504091509**, successfully defended the **thesis** entitled "**HUMAN ASSISTED HUMANOID ROBOT PAINTER**", which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Asst. Prof. Dr. Hatice KÖSE** Istanbul Technical University	....
Jury Members :	**Prof. Dr. Nadia ERDOĞAN** Istanbul Technical University	....
	Assoc. Prof. Dr. Nafiz ARICA Turkish Naval Academy	....

Date of Submission : 4 May 2012
Date of Defense : 6 June 2012

To my love of my life,

FOREWORD

First of all, I would like to express my sincere gratitude to my advisor, Asst. Prof. Dr. Hatice KOSE, for her very valuable support and direction in my thesis. She kindly accepted me as her master degree student and encouraged me in this study. When I ran into a problem, I found myself next to her and she did not withhold her assistance from me for a moment.

I would also like to express my sincere gratitude to both of my families that I feel their support always with me.

I want to admit that I could not finish this study without her. Every time I scooped up my head from my work with a fatigue and distress, I saw her sweety smile and I turned to my study with a relaxed mood. She did not leave me alone for a moment, thank you for being with me, my lovely wife.

June 2012 Cemal GURPINAR

TABLE OF CONTENTS

ABBREVIATIONS

AI	**:** Artificial Intelligence
API	**:** Application Programming Interface
CCD	**:** Charged-Coupled Device
CMOS	**:** Complementary Metal Oxide Semiconductor
CoP	**:** Central of Pressure
CoM	**:** Center of Mass
CPU	**:** Central Processing Unit
DOF	**:** Degrees of Freedom
HD	**:** High Definition
IDE	**:** Integrated Development Environment
IR	**:** Infrared
ITD	**:** Interaural Time Difference
LED	**:** Light Emitting Diode
PDF	**:** Probability Density Function
pdf	**:** Probability Distribution Function
RGB	**:** Red Green Blue
MST	**:** Minumum Spanning Tree

LIST OF TABLES

LIST OF FIGURES

Page

HUMAN ASISTED HUMANOID ROBOT PAINTER

SUMMARY

Currently, research on humanoid robots is one of the most exciting and hot topics. Many researches have been studied on humanoid robots like creating a mobile robot that is able to coexist and collaborate with humans and to perform tasks that humans cannot. Another attractive research area is creating artist robots that are able to perform arts.

In this research, NAO H25 human assisted humanoid robot painter is described. The aim of this study to reproduce the whole painting process by a humanoid robot with a vision system with the assistance of a human.

To create a robot painter, solutions for some key problems are generated. The key problems regarding vision are obtaining the image of the environment in which the objects that the robot paints, 2D object segmentation, extracting the objects from environment, color perception, reducing the number of color to the optimum set of colors that can be used by the robot and determining orientation for brush strokes. The key problems regarding NAO humanoid robot are standing in front of the painting table, grasping the paintbrush, moving down the paintbrush to the start point, drawing a line and moving up the paintbrush from end point.

Painting process are divided into three phases: obtaining the image of the environment that the objects are in, extracting the objects to paint and painting by a robot to be in interaction with its assistant (i.e. Robot says to human: “Could you give me the blue color?”).

NAO H25 humanoid robot has two CMOS-Complementary Metal Oxide Semiconductor digital cameras on its head. It acquires the image of the environment via its camera.

After taking the image of the environment, the foreground and background pixels should be segmented to extract objects from the environment. Graph-based object segmentation is used for this purpose. Once this algorithm applied to the image, all colors are determined and can be percept by the robot.

The robot must analyze color distribution of the object to determine the appropriate color segments for the optimum set of colors that can be used by the robot. This issue is related with the color reduction. K-means clustering method is used for color reduction.

When the color reduction is finished, the orientation used to guide brush strokes should be computed in order to imitate human painting style. The gradient information is used to calculate orientation.

After computing orientation, NAO tries to find an appropriate area in each color segment derived from the second phase (extracting objects from the environment). If the robot finds such an area, it calls its assistant and asks for the color that it will paint. Its assistant gives the color case to it. NAO controls the color and if it is true, it thanks to child. If the color is incorrect, the robot asks for the right color. When the robot gets the right color, it starts drawing. NAO moves its arm to the start position and pushes the paintbrush down. Then it moves its arm to the end position and pulls the brush up.

In this system, it is important to paint in a stable direction. The grasping becomes unstable when a brush moves from near side to far side. However, it is stable when a brush moves from left to right and from far side to near side.

For the next time, the robot looks for a new area to paint. All of the color in one level is painted; the next level starts with a thinner brush. The number of level is related with the brush number used for painting.

Webots is a development environment used to model, program and simulate mobile robots. This painting art is simulated in a generic environment in Webots with NAO. A chair, a can on the chair and two balls of different sizes in front of the chair are located. The humanoid robot paints these objects on a canvas. In the simulator, the robot moves its arm and its hand, and the simulated picture is filled with colors, as the robot's hand is moved.

In real environment, a table, a cardboard used as a canvas placed on a table, a pot the robot paints and gouache paints are used. Currently, the robot paints on a table top, using its arm like a plotter on an x-y plane and moves its upper torso with the arm to reach different locations on the table.

Once the system is moved on the real robot, there are some technical challenges, primarily due to the requirement that the NAO uses its articulated arms to paint on a fixed surface (canvas). It has some limitations on its articulated arms.

NAO H25 humanoid robot does not have any force and tactile sensors on its fingers. Because of this constraint, the paintbrush is fixed in the robot's hand to manipulate it while painting.

İNSAN DESTEKLİ İNSANSI ROBOT RESSAM

ÖZET

Günümüzde, insansı robotlar üzerindeki araştırmalar en heyecan verici ve sıcak konulardan biridir. İnsansı robotlar üzerinde insanlarla bir arada ve işbirliği içinde çalışabilen ve insanların yapamadığı görevleri yapabilen hareketli robot yaratma gibi birçok araştırma yapılmıştır. Bir diğer ilgi çekici araştırma alanı ise sanat icra edebilen robotlar yaratmaktır.

Bu araştırmada, NAO H25 insan destekli insansı robot ressam anlatılmaktadır. Bu çalışmanın amacı, bir insan tarafından resim yapılırken yerine getirilen işlemlerin tamamının, görü sistemi olan insansı bir robot tarafından bir insan yardımıyla gerçekleştirebilmesidir.

Bir ressam robot yaratmak konusunda, bazı önemli sorunlar için çözümler üretilmiştir. Görü ile ilgili çözülmesi gereken önemli sorunlar, robot tarafından resmi yapılacak nesnelerin bulunduğu ortamın görüntüsünün elde edilmesi, iki boyutlu nesne bölütlemesinin yapılması, nesnelerin yeraldığı ortam görüntüsünde nesneler ile arka planın birbirinden ayrılması, nesnelerin sahip olduğu renk sayısının robot tarafından kullanılabilecek en uygun renk sayısına indirgenmesi, renklerin robot tarafından algılanabilmesi, fırça darbelerinin yönünün belirlenmesi ve kullanılacak doğru rengin robot tarafından tespit edilebilmesidir. NAO insansı robotu ile ilgili çözülmesi gereken önemli sorunlar, insansı robotun resmin çizileceği yatay resim masası önünde ayakta durması, boya fırçasını kavraması, boya fırçasını başlangıç noktasında aşağı doğru indirmesi, fırça ile bir çizgi çizilmesi ve bitiş noktasında yukarı doğru kaldırmasıdır.

Resim yapma işlemi üç bölüme ayrılmıştır: 1) Resmi yapılacak nesnelerin bulunduğu ortamın resminin robot tarafından elde edilmesi, 2) Resmi yapılacak nesneler ile arka planın birbirinden ayrılması, 3) Asistanı ile etkileşime geçerek robot tarafından nesnelerin resminin yapılması (örneğin robot yardımcısına “Bana mavi boyayı verebilir misin?” diyerek etkileşimde bulunabilir.)

NAO H25 insansı robotun başının üzerinde bir tanesi alın, diğeri ise ağız hizasında olmak üzere iki tane dijital kamerası vardır. Robot, nesnelerin bulunduğu ortamın resmini alın hizasındaki kamerası aracılığıyla elde eder. Bu işlem neticesinde birinci bölüm tamamlanmış olur.

Ortam görüntüsü elde edildikten sonra nesne ile arka planın birbirinden ayrılması maksadıyla nesneye ait görüntü elemanları ile arka plana ait görüntü elemanlarının bölütlenmesi gerekmektedir. Bu amaçla çizge tabanlı nesne bölütleme algoritması kullanılmaktadır. Bu algoritma neticesinde nesnelerin sahip olduğu renkler belirlenmiş olur ve bu renkler robot tarafından algılanır.

Robotun, resmini yapacağı nesneler çok farklı renklere sahip olabilirler. Buna karşın robotun elinde var olan renk sayısı ise sınırlıdır. Bu nedenle robotun uygun renk bölütlerini belirleyebilmesi için nesnenin renk dağılımını analiz etmesi gerekmektedir. Bu konu renk azaltma ile ilgilidir. Bu çalışmada renk azaltma işlemi için K-means bölütleme algoritması kullanılmıştır.

Nesne ile arka planın birbirinden ayrılması, renklerin algılanması ve renk azaltma işlemlerinin tamamlanmasından sonra, robotun fırça darbelerini tuval üzerine nasıl uygulayacağına karar vermesi gerekmektedir. Bu maksatla fırça darbelerinin yönü belirlenmelidir. Fırça darbelerinin yönünün belirlenmesi için ortam resmindeki görüntü elemanlarının yön bilgileri kullanılmıştır.

Fırçanın yönünün belirlenmesinden sonra NAO her bir renk bölütünde boyayabileceği uygun bir alan bulmaya çalışır. Böyle bir alan bulduğunda, yardımcısını çağırır ve kendisinden boyama yapacağı rengi vermesini ister. Yardımcı renk kâsesini NAO'ya verir. NAO, boya kâsesini kontrol eder ve doğru boya olup olmadığını tespit eder. Eğer doğru boya ise yardımcısına teşekkür eder, doğru boya değilse o zaman yardımcısından doğru boyayı kendisine vermesini ister. NAO doğru boyayı elde ettiğinde resim yapmaya başlar. Öncelikle kolunu başlangıç noktasına götürür ve boya fırçasını aşağı indirir. Daha sonra kolunu bitiş noktasına götürür ve fırçayı kaldırır.

NAO bulduğu ilk uygun alanı boyadıktan sonra bir başka uygun alan arar. İlk seviyedeki bütün renkler boyandıktan sonra, bir sonraki seviyeyi daha ince fırça ile boyamaya başlar. Resim yapma işleminde kullanılan seviye sayısı, boyama için kullanılan fırça sayısı ile ilgilidir. Kullanılacak fırça sayısı yapılacak olan resmin tamamlanacağı seviye sayısını belirler.

Bu sistemde boyama yaparken fırçayı dengeli bir yönde hareket ettirmek önemlidir. Aksi taktirde fırça, robotun elinden kayabilir veya çizgiler yanlış bir şekilde çizilebilir. Boya fırçası, robota yakın olan taraftan uzak olan tarafa doğru hareket ederse kavrama dengeli bir hale gelir. Bununla birlikte, boya fırçası soldan sağa doğru ve uzak taraftan yakın tarafa doğru hareket ederse o zaman kavrama dengesiz bir hale gelir.

Webots, hareketli robotların modellenebilmesi, programlanabilmesi ve benzetiminin yapılabilmesi için kullanılan bir geliştirme ortamıdır. Resim yapma işlemi, Webots benzetim programında NAO H25 insansı robotu kullanılarak yaratılan jenerik bir ortamda gerçekleştirildi. Bu ortama bir adet sandalye, sandalyenin üzerine bir adet küçük teneke kutu, sandalyenin önüne ise bir tanesi küçük bir tanesi büyük olmak üzere iki adet farklı renkte top yerleştirildi. Insansı robot bu nesnelerin resimlerini jenerik ortamdaki tuval üzerine yaptı. Benzetim ortamında robot kolunu ve ellerini oynatırken, aynı zamanda resim de yavaş yavaş renklerle doldu ve resim ortaya çıktı.

Gerçek ortamda gerçekleştirilen çalışmalarda yatay bir masa, masa üzerine yerleştirilmiş tuval olarak kullanılan bir kağıt, resmini yapacağımız bir saksı ve guaj boya kullanılmıştır. NAO, kollarını x-y düzlemi üzerinde bir çizici gibi kullanarak ve üst gövdesini koluyla birlikte hareket ettirerek masa üzerindeki farklı noktalara erişebilmektedir.

Sistem gerçek robot üzerine taşındıktan sonra, özellikle de NAO'nun eklemli kollarını sabit bir yüzey üzerinde resim yapması için hareket ettirmesi gereksiniminden kaynaklanan bazı teknik zorluklar ortaya çıkmıştır. Çünkü NAO'nun eklemli kollarının hareket kısıtlaması vardır. Sabit yüzey üzerinde istediği bir noktaya istediği bir şekilde hareket edememektedir. Uzak noktalara erişebilmesi için kollarıyla birlikte üst gövdesini de hareket ettirmesi gerekmektedir. Üst gövdesini hareket ettirirken aynı zamanda ayakta olması nedeniyle kendi dengesini de sağlaması gerekmektedir.

NAO'nun kolları ile ilgili kısıtlamalarının yanında elleri ile ilgili de bazı kısıtlamaları vardır. NAO'nun her bir elinde üç parmağı vardır. Parmaklarından hiçbirisini tek başına hareket ettirememekte, sadece parmaklarını açıp kapayabilmektedir. Bununla birlikte parmaklarının üzerinde herhangi bir güç veya dokunma sensörü bulunmamaktadır. Bu kısıtlamadan dolayı insansı robot, fırçayı kendi imkânları ile insana benzer bir şekilde hissederek kavrayamamaktadır. Elini açtığı anda fırça elinin içine sabitlenmekte ve daha sonra elini kapatmak suretiyle fırçayı kavramaktadır.

1. INTRODUCTION

Currently, research on humanoid robots is one of the most exciting and hot topics. Development of the humanoid robot has recently been progressing, and the technologies surrounding it have progressed simultaneously. Applications for the humanoid robot are expanding as well as its hardware and software development.

Many researches have been studied on humanoid robots like creating a mobile robot that is able to coexist and collaborate with humans and to perform tasks that humans cannot. Another attractive research area is creating artist robots that are able to perform arts.

In this research, a NAO H25 human assisted humanoid robot painter is described. The aim of this study is to reproduce the whole painting process by a humanoid robot with a vision system with the assistance of a human.

The painting process is divided into three phases (Figure 4.1): obtaining the image of the environment that the objects are in, extracting the objects to paint and painting by a robot to be in interaction with its assistant (i.e. Robot says to human: “Could you give me the blue color”).

In this painting system, first, NAO H25 humanoid robot acquires the image of the environment in which objects through its camera.

In the second phase, the foreground and background pixels are segmented to extract objects from the environment.

The robot analyzes the color distribution of the object to determine the appropriate color segments for the optimum set of colors that can be used by the robot. K-Means clustering method is used for color reduction.

Until this section, NAO extracts the objects and determines the colors of the objects for painting. Before starting to paint there is only one process which maps the question that is in which direction NAO moves its brush. This issue is related with the orientation.

In this stage, it is required to guide NAO’s brush strokes to paint properly. For this purpose, gradient information to lead brush strokes that is robust to texture is used. NAO computes the normal orientation at every pixel in image using gradient operator.

Generally, a human painter does not paint at once. He/she starts painting with a large brush to paint rough scene and pass over the painting again to add fine details.

In the last phase, NAO tries to find an appropriate area in each color segment derived from the second phase. If the robot finds such an area, it calls its assistant and asks for the color that it will paint. Its assistant gives the color case to it. NAO controls the color and if it is true, it thanks to child. If the color is incorrect, the robot asks for the right color. When the robot gets the right color, it starts drawing. NAO moves its arm to the start position and pushes the paintbrush down. Then it moves its arm to the end position and pulls the brush up.

For the next time, the robot looks for a new area. All of the color in one hierarchy is painted, the next hierarchy start with a thinner brush to add fine details. The number of hierarchy is related with the brush number used for painting.

Webots is a development environment used to model, program and simulate mobile robots. This painting art is simulated in a generic environment in Webots with NAO. A chair, a can on the chair and two balls of different sizes in front of the chair are located in the environment. The humanoid robot paints these objects on a canvas. In the simulator, the robot moves its arm and its hand, and the simulated picture is filled with colors, as the robot’s hand moves.

In real environment, a table, a cardboard used as a canvas placed on a table, a pot that the robot will paint and gouache paint are used.

Currently, the robot paints on a tabletop, using its arm like a plotter on an x-y plane and moves its upper torso with the arm to reach different locations on the table.

The ultimate goal of this study is to use this sytem as a part of an interaction game between the robot and disabled children to contribute to the rehabilitation of children with disabilities whenever the system is transfered to the real robot. The steps of the interaction game are planned as follows;

- The humanoid robot asks disabled child to find the correct color case,

- The disabled child put the color case in front of the robot,
- If the color isn't true, the robot asks for a right color,
- If the color is true, the robot says "Thank You" to the disabled child.

NAO has some limitations because of its articulated arms. It can not do every action like getting the painting color itself and can not reach every point on the canvas. It is intended to use the interaction between the child and robot to deal with the limitations of the robot, i.e. to get the right color or to sink the paintbrush into the color case. It is also expected that some emergent actions, i.e. child's helping the robot by coloring some parts which are hold for robot to color and which robot does not reach itself.

It is desired that the children to understand the robot's limitations and emergently play an active role to assist the robot to achieve the painting task.

In Chapter 2, other robots that perform arts are introduced. Chapter 3 explains the overview of the NAO humanoid robot. Chapter 4 demonstrates how NAO acquires the image of the environment. Object segmentation methods are explained in Chapter 5. Chapter 6 shows how NAO percept colors and make color reduction. Chapter 7 is about contour matching. Chapter 8 contains orientation for brush stroke. Chapter 9 explains how NAO performs painting process. Chapter 10 gives information about our experiments. Chapter 11 demonstrates our conclusions and recommendations.

2. RELATED WORK

Research into humanoid robots generates great interest in both the scientific community and popular culture. Ideally, a humanoid robot could perform a cooperative task with a human or work in a dangerous area such as a construction site instead of a human. Another interesting research area is cerating robots performing arts. There are some examples of artist robots. AARON [1], for example, is a unique robot that creates a piece of art by itself based on adapting a geometrical model of a subject. ISAC [2], is a robot that track artist's hand trajectory and mimic it. Dot-Chan [3], focuses on force feedback along with grasping technology.

2.1 AARON

Harold Cohen, who is an English painter, at Stanford University's Artificial Intelligence Lab in 1973, developes AARON, which is the first robot in human history to paint original art.

Cohen implements AARON's program to discover what an independent (machine) intelligence might do, given some knowledge of the world and some rudimentary physical capabilities. And, in the process, to have it teach him about possibilities he hasn't imagined. He wants AARON's work is less like human work, not more like human work.

AARON that has no physical or visual feedback mechanism could only paint objects or humans given information about so it has to know what it's painting.

AARON has two parts. In the first part (Figure 2.1), artificial intelligence program creates image as a software. This image can be printed or can be shown on the screen. In the second part, cartesian robot (Figure 2.2) basically serves as a plotter and draws the image.

AARON, firstly, is built as a screen saver. However, it is different from other screen savers, because it can draw infinite number of original picture and therefore on every computer that AARON is executed, AARON draws different pictures.

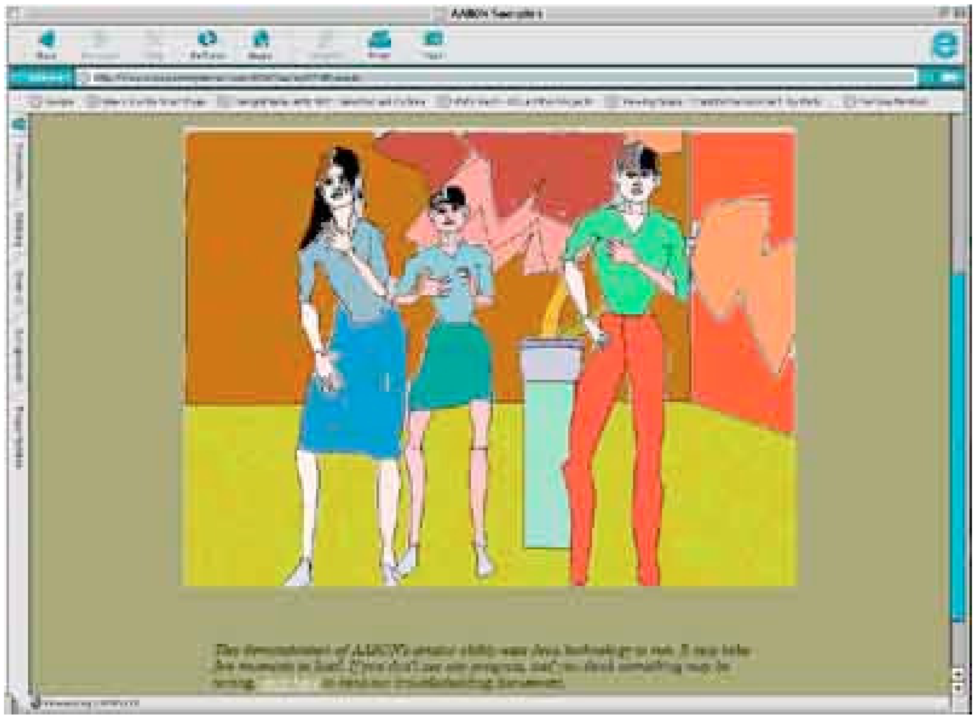

This demonstration of AARON's artistic ability uses Java technology to run. It may take few moments to load. If you don't see any progress, [illegible] something may be wrong. [illegible] troubleshooting document.

Figure 2.1 : First Part Of The AARON (Software Of The AARON-Screensaver).

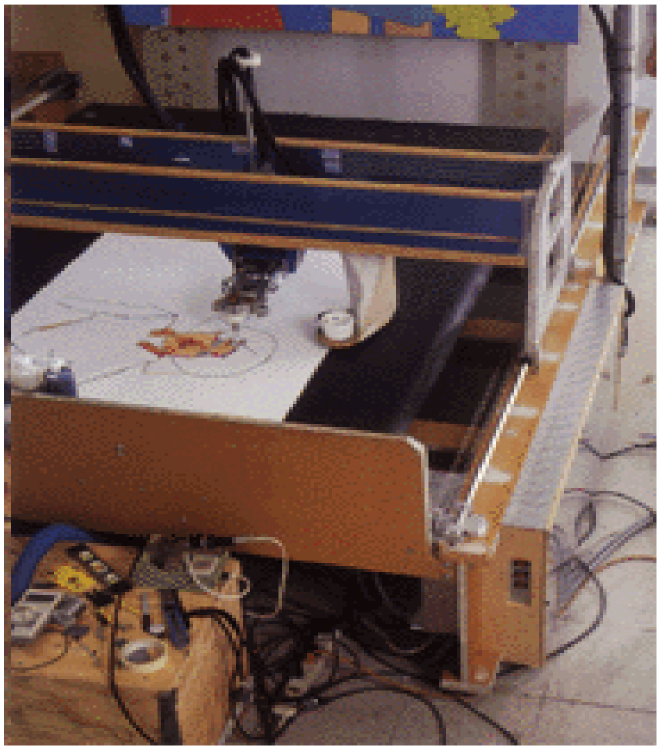


Figure 2.2 : Second Part Of The AARON (Cartesian Robot).

2.2 ISAC

ISAC (Figure 2.3), a Dual Armed Humanoid Robot, explores the relationship between man's creativity and automation in the field of visual art. ISAC has the ability to observe an artist in the process of drawing, and to mimic the motions of the artist.

The robot can record the motions and then reproduce its own "robotic" interpretation of the drawing. Because of the nature of its soft arms, ISAC is unlikely reproduce the drawing exactly. Each time it reproduces the artist's actions the robot will add its own subtle variations.

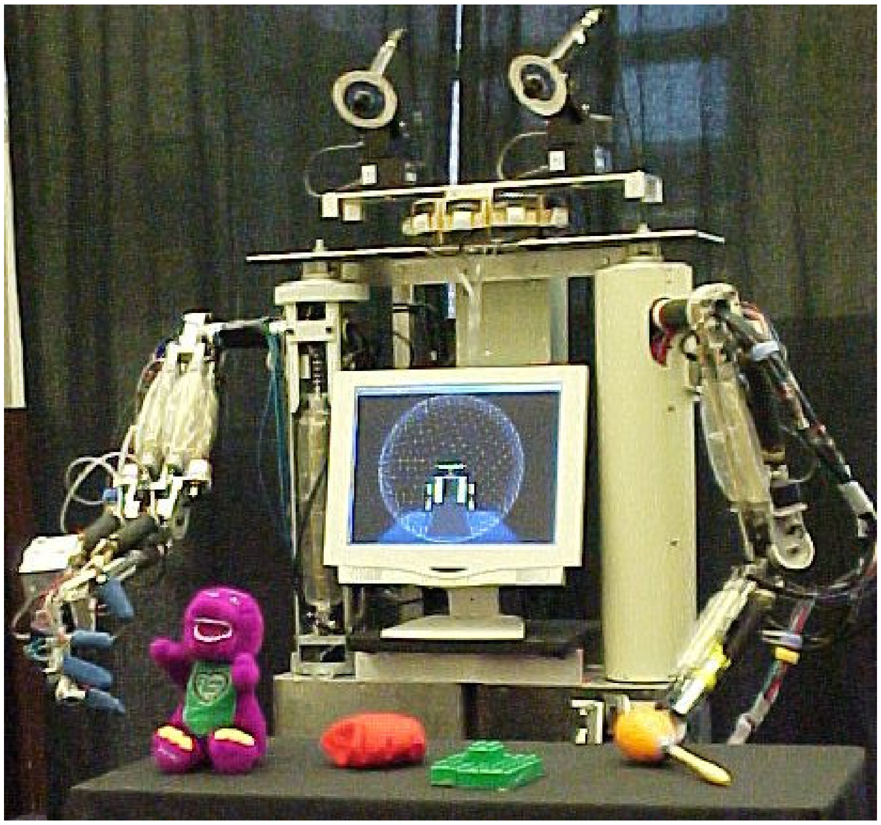


Figure 2.3 : ISAC.

ISAC, can perform human-like drawing if the human hand/arm movement can be tracked. In order to mimic human-hand movement, some knowledge about the hand is needed. ISAC's visual tracking module performs this task.

The objective of the visual tracking module is to provide information (such as position) about the hand to the robot system. The tracking system of the ISAC is composed of a camera head with two color cameras, a color image acquision board and two Pentium PC's for color image processing, hand tracking, and camera control. Figure 2.4 shows two color CCD-Charged-Coupled Device cameras mounted on a pan/ tilt/ verge system. The hand tracker utilizes the 2-D position of the hands in the image planes and subsequently controls the camera head to center the hand in the camera views. After the target is fixated, the 3-D position can be computed and used by the robot system to find the position of the hand relative to the robot coordinate system.

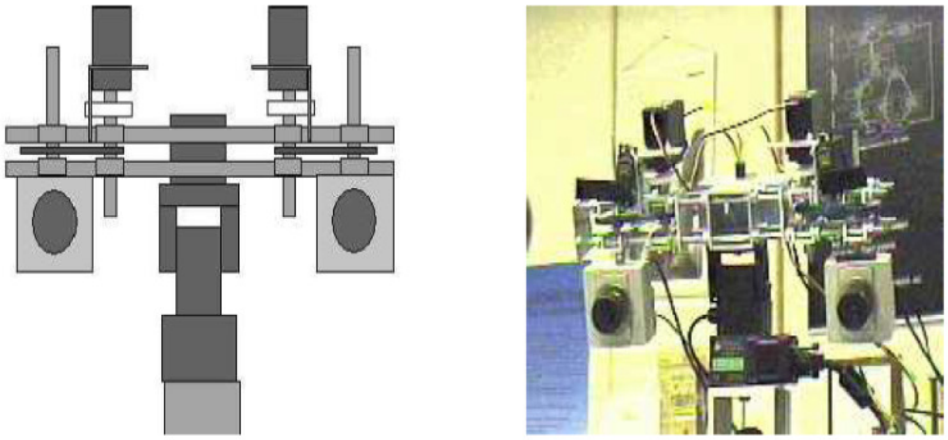


Figure 2.4 : ISAC's 4 DOF camera head.

2.3 DOT-CHAN

DOT-CHAN (Figure 2.5) is a painting robot with multi-fingered hands and stereo vision, which uses a paintbrush to paint a picture. The robot's ultimate goal is to reproduce whole painting process related with human painting.

DOT-CHAN has two arms with seven DOF-Degrees-of-Freedom and two hands with four fingers. At the top of the each finger, there is a force sensor and on the palm of each hand, there are 20 tactile sensors. These force and tactile sensors are used to grasp paintbrush similar to the human painter.

The robot has a camera head with nine cameras and stereo vision system. The robot can move its head vertically and horizontally. This stereo vision system is used to determine the position of the object in 3D space.

Painting action is carried out in three stages by the robot: acquiring 3D model, composing a picture model, and painting by the robot.

In the first stage, 3D shape is reconstructed from multi-view camera images to simulate the human painter observing an object from several viewpoints. Multi-view camera images are obtained from nine cameras of DOT-CHAN.

In the second stage, an interpretation of a 3D model is carried out in order to determine how to represent features of the object. In fact, the robot cannot start painting with the 3D model of the object. A transition is needed between the 3D model and painting. For this purpose, a picture model is established to represent the target object on a canvas. A picture model is composed by extracting the features of the object from its 3D model and texture.

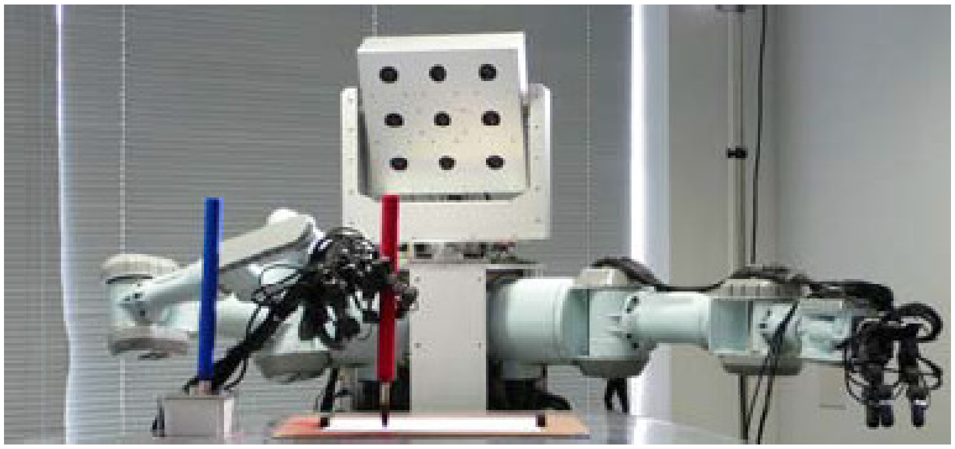


Figure 2.5 : DOT-CHAN Humanoid Robot Painter.

In the third stage, robot paints the object with multi-fingered hands. Figure 2.6 shows the flowchart of a robot painting.

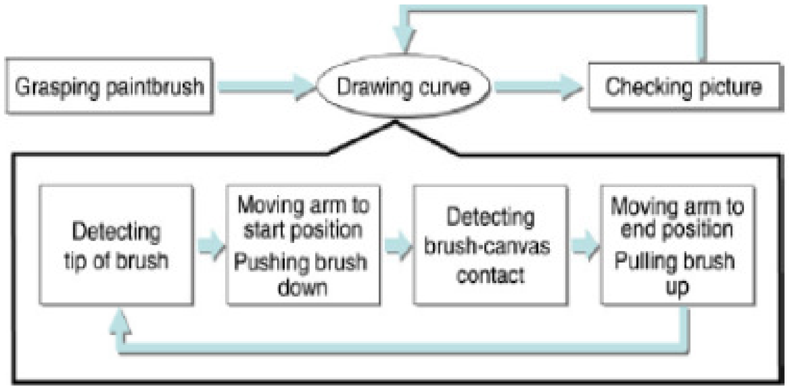


Figure 2.6 : The Flowchart of a Robot Painting.

3. OVERVIEW OF THE NAO H25 HUMANOID ROBOT

3.1 Hardware Platform

NAO H25 (Figure 3.1) [4], is a programmable, 57-cm tall humanoid robot with the following key components:

- NAO's body has 25 DOF (Table 3.1) and its key elements are electric motors and actuators.
- NAO's sensor network is composed of 2 cameras, 4 microphones, sonar rangefinder, 2 IR-Infrared emitters and receivers, 1 inertial board, 9 tactile sensors, and 8 pressure sensors.
- NAO's communication devices are voice synthesizer, LED-Light Emitting Diode lights, and 2 high-fidelity speakers.
- NAO has Intel ATOM 1,6ghz CPU- Central Processing Unit (located in the head) that runs a Linux kernel and supports Aldebaran's proprietary middleware (NAOqi).
- NAO has second CPU (located in the torso).
- NAO has 27,6-watt-hour battery that provides NAO with 1.5 or more hours of autonomy, depending on usage.

Table 3.1: The Construction of NAO H25.

Part	DOF
Head	x2 DOF
Arm (in each)	x5 DOF
Pelvis	x1 DOF
Leg (in each)	x5 DOF
Hand (in each)	x1 DOF

3.2 Vision

The overview of the NAO's Vision System is summarized in table 3.2.

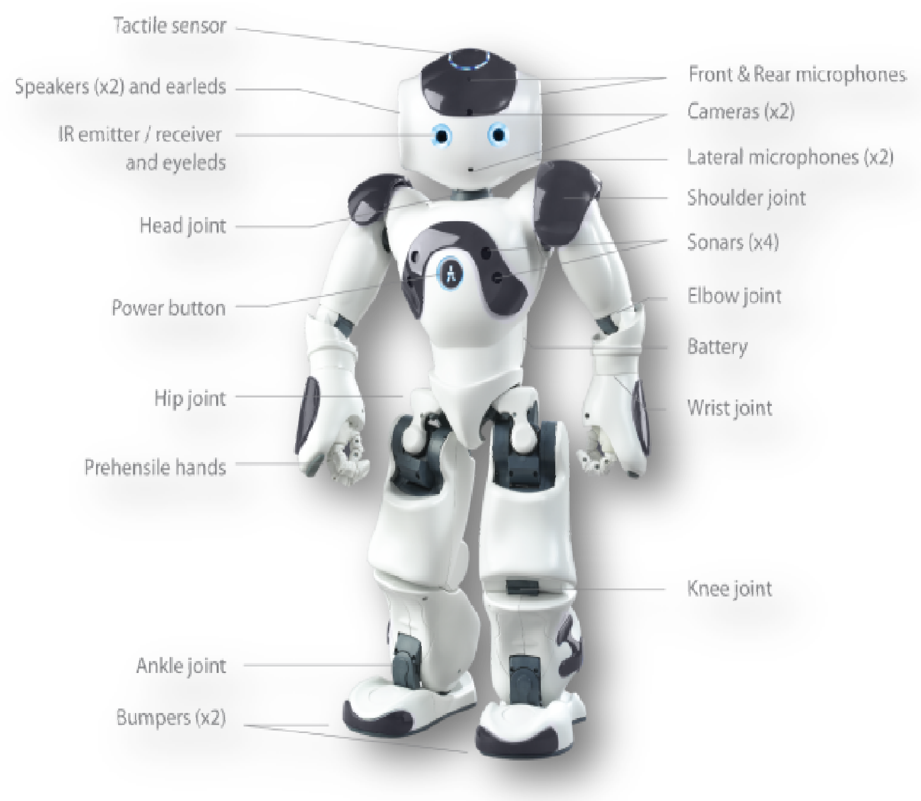


Figure 3.1 : NAO Humanoid Robot.

3.2.1 Camera

NAO H25 humanoid robot has two CMOS digital cameras on his head. The first camera is located on NAO's forehead and scans the horizon. The second one which is located at mouth level scans the immediate surroundings.

Cameras can capture up to 30 mages per second in HD-High Definition resolution. NAO can move the head by 239°horizontally and by 68° vertically, and his camera can see at 61° horizontally and 47°vertically (Figure 3.2-3.3).

The Choregraphe software (explained in section 10.4) and Webots simulation environment (explained in section 10.1) lets the user recover photos and video streams of what NAO sees. But eyes are only useful for interpreting what NAO sees. For instance, NAO can track a red ball, a face or another object which is learned before by NAO. To determine whether NAO sees the specified object or face currently, NAO blinks eyes if the object or face is in the field of its vision (Figure 3.4).

Table 3.2: The Overview of The NAO's Vision System.

Part	Value
Cameras	**x2 on front**
Sensor Model	MT9M114
Sensor Type	SOC Image Sensor
Imaging Array	
Resolution	1.22MP
Optical Format	1/6 inch
Active Pixels (HxV)	1288x968
Sensitivity	
Pixel Size	1.9µm
Dynamic Range	70dB
Signal/Ratio (max)	37dB
Responsivity	2.24 V/lux-sec (960p) 8.96 V/lux-sec (VGA)
Output	
Camera Output	960p@30fps
Data Format	YUV422
Shutter Type	ERS (Electronic Rolling Shutter)
View	
Field of View	72.6°DFOV (60.9°HFOV,47.6VFOV)
Focus Range	30cm ~ infinity
Focus Type	Fixed Focus

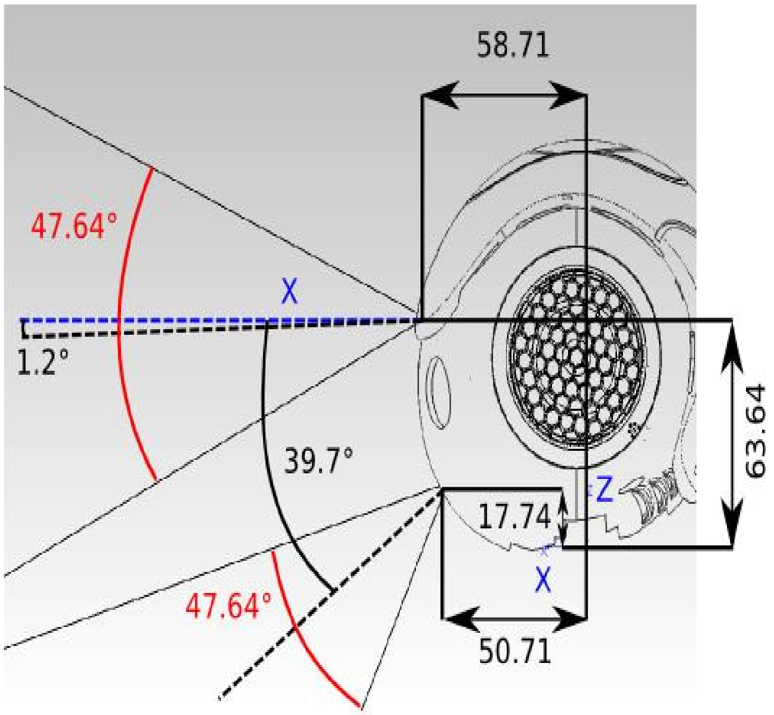


Figure 3.2: Camera Position-Vertical.

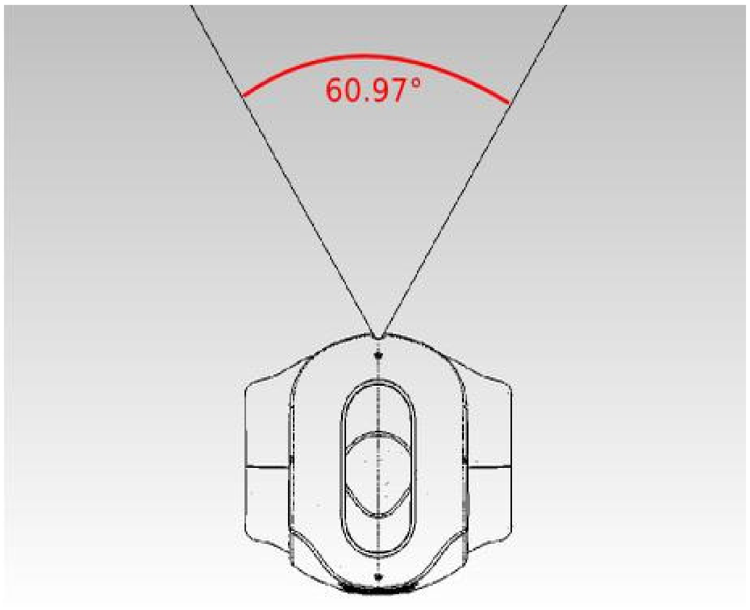


Figure 3.3: Camera Position-Horizontal.

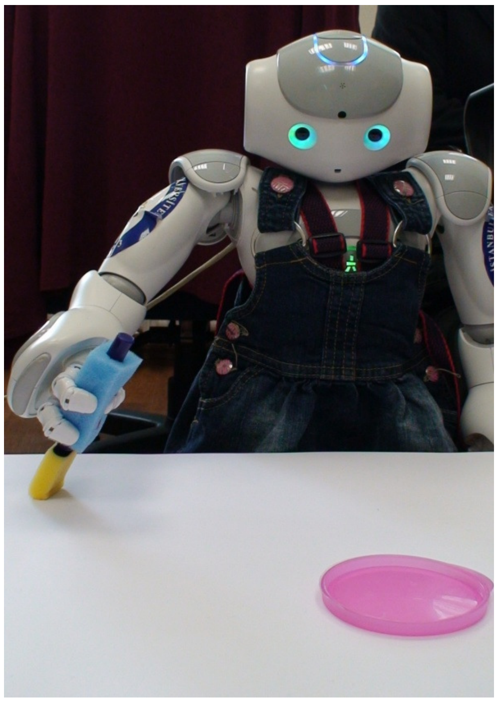


Figure 3.4: NAO blinks its eyes when the color case is in the field of vision.

In this study, NAO takes the image of the environment in which the objects through its forehead digital camera at 640x480 resolution.

3.2.2 Object Recognition

NAO has the capability of recognizing a large quantity of objects. Once the object is saved to the internal database of NAO thanks to Choregraphe software, if it sees this object again, NAO is able to recognize and say what it is.

NAO contains a set of algorithms for detecting and recognizing objects. NAO can detect and recognize a ball or, eventually, more complex objects.

In this study, object recognition is used to identify color cases used by the robot to paint objects. Every color is stored in a different color case. Initially, every color case is shown to the NAO, and NAO saves these color cases with their names to its internal database.

When NAO's assistant gives him a color case, NAO controls the color case to identify it. If it is matched with the one in its database, NAO thanks to his assistant. If it is not matched with the one in its database, NAO asks for a right color.

3.2.3 Face Detection and Recognition

Face detection and recognition is one of the best known features for interaction between human and the robot. NAO can detect and learn a face in order to recognize it next time. Besides, NAO can track a face.

NAO contains a set of algorithms for detecting and recognizing faces. NAO can recognize who is talking to it.

These algorithms have been specially developed, with constant attention to using a minimum of processor resources.

This key feature is used to recognize NAO's assistant. Initially, NAO's assistant is shown to it and it saves face to its internal face database. When NAO sees its assistant for the next time, it recognize him/her and track his/her face while talking to it.

3.3 Audio

The overview of the NAO's Audio System is summarized in Table 3.3.

Table 3.3: The Overview of The NAO's Audio System.

Part	**Value**
Loud Speakers	**x2 later**al
Diameter	36mm
Impedance	8ohms
Sp Level	87dB/w +/- 3dB
Freq Range	Up to ~20kHz
Input	2W
Microphone	**x4 on the head**
Sensitivity	~40 +/-3dB
Frequency Range	20Hz-20kHz
Signal/Noise Ratio	58dBA

3.3.1 Text To Speech

NAO has two loud speakers and two leds in its ears (Figure 3.5).

With these speakers, NAO can speak up to 9 languages including English, German, French, Spanish, Italian, Chinese, Japanese, Korean, Portuguese. With a "say box" in Choregraphe user can insert text and modify voice parameters as he/she wish. NAO is able to say the text correctly, with the right punctuation and intonation.

With this feature, NAO can ask for a color to its assistant after determining the color used in painting and can interact with its assistant.

3.3.2 Automatic Speech Recognition

Speech recognition is critical in humanrobot interaction. If there is no interaction between a robot and a human, they cannot coexist and cooperate in order to achieve a certain task. The important point is to develop stable and powerful speech recognition.

NAO has four microphones: one of them is on front, the other is on back and others are on lateral (Figure 3.6). With these microphones, NAO is able recognize speech who is talking to it.

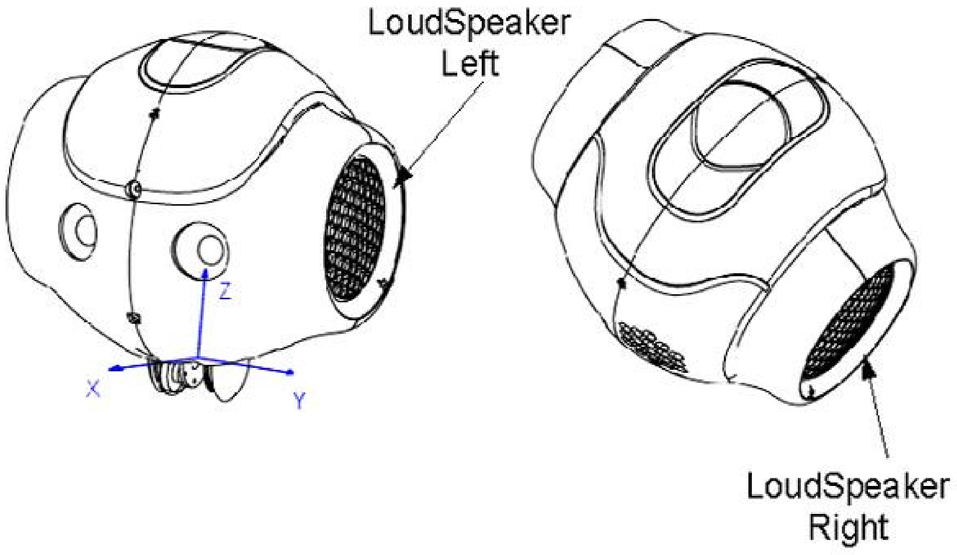


Figure 3.5: Loudspeakers of NAO.

NAO can hear someone from 2 meters away and recognize a complete sentence or just few words in the sentence in 9 languages including English**,** German**,** French**,** Spanish**,** Italian**,** Chinese, Japanese, Korean.

In this work, NAO can talk with its assistant more fluid and make conversation close to natural using this feature.

3.3.3 Speech Detection and Localization

NAO's environment is made of sounds that he, like people, is able to detect and localize in the space thanks to microphones all around his head.

One of the main purposes of humanoid robots is to interact with people. Thanks to sound localization, a robot can identify the direction of sounds. NAO sound source localization is based on an approach known as "Time Difference of Arrival."

When a sound is made by a source, each of NAO's four microphones receives the sound wave at slightly different times.

For example, if someone talks in front of NAO , the corresponding sound wave first hits the front microphones, then the left and right microphones a few milliseconds later, and finally the rear microphone.

These differences is known as ITD-Interaural Time Difference (Figure 3.7) [5]. They are used to determine the current location of the emitting source.

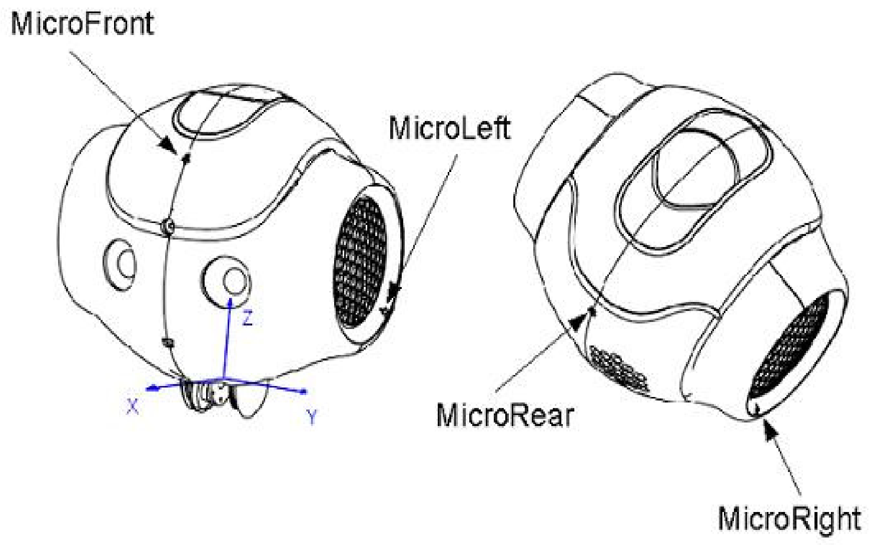


Figure 3.6: Microphones of NAO.

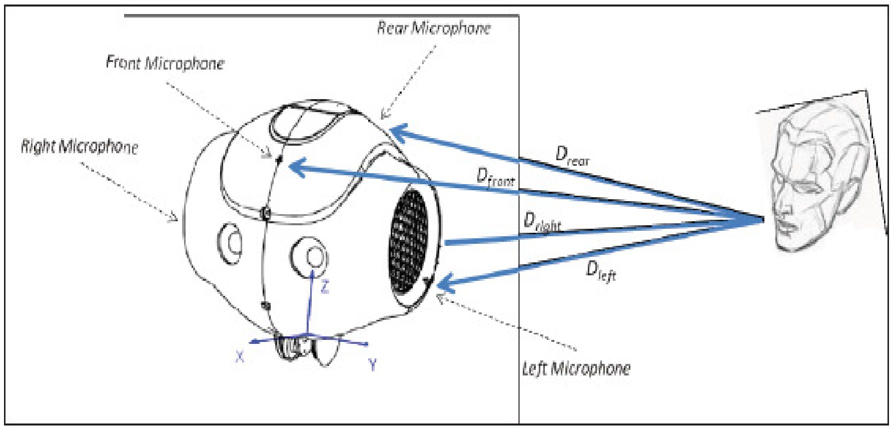


Figure 3.7: Interaural Time Difference (ITD).

By solving the equation every time it hears a sound, NAO can determine the direction of the emitting source (azimuthal and elevation angles) from ITDs between the four microphones.

This feature is used to make localization of NAO's assistant while he/she is talking to him. NAO is able to turn to the direction that sound comes. Therefore, natural conversation is simulated.

4. ACQUIRING THE IMAGE OF THE ENVIRONMENT

Painting action is divided into three stages (Figure 4.1). First stage is acquiring the image of the environment.

NAO H25 humanoid robot has two CMOS digital cameras on his head. The robot takes the image of the environment which has the objects to be painted through the its digital camera located on its forehead.

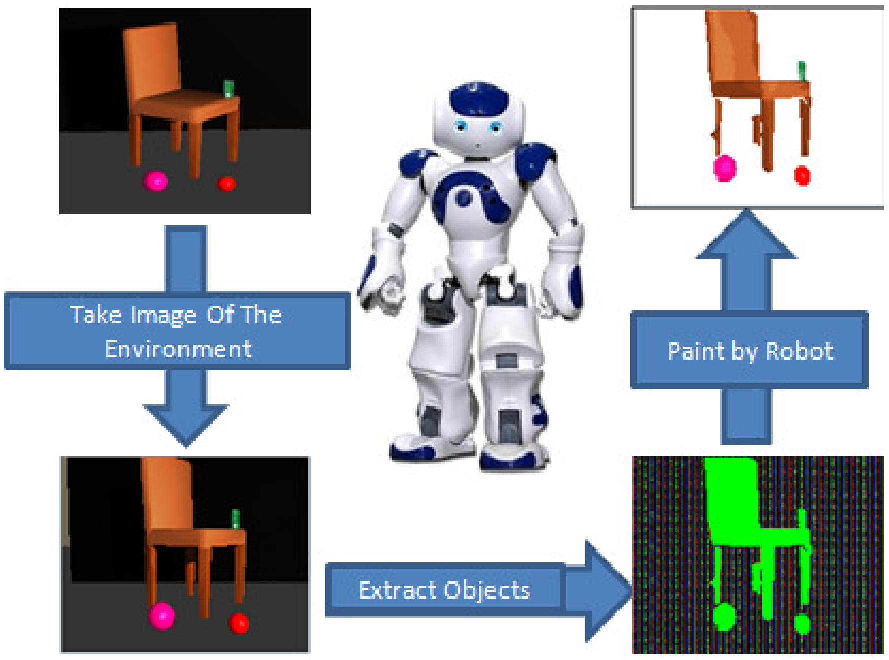


Figure 4.1: Overview of The Painting System.

5. OBJECT SEGMENTATION

Image segmentation is related with subdividing an image into parts that are closely identified with objects, regions or areas with similar properties. According to the time of the Gestalt movement in psychology [6], it is known that perceptual grouping plays a powerful role in human visual perception.

Image segmentation is very useful phase for object recognition. Because proper object recognition is very tough task and often needs high-level knowledge. This knowledge can be obtained from segmentation phase. Objects can be detected and can be seperated from background using this high-level knowledge. For instance, in medical images, red blood cells can be thought as objects. To count red blood cells, segmentation is applied to the image.

Generally, image segmentation algorithms are depends on two basic properties of intensity values [7]. One of them is discontinuity and the other one is similarity. In the first classification, the approach is about to subdivide an image into regions based on sudden changes in intensity, such as edges or corners in an image. Thresholding [8-10], is one of the examples of this classification. In this method, an appropriate threshold T is selected to seperate object pixels and background pixels from each other. All images are different from each other. The success of this method is completely based on how well the histogram can be separated. An ideal solution is to look at the histogram of the image and determine the threshold value only for this image. Nevertheless, determining the threshold value manually or automatically is a tough work.

In the second classification, the approach is about to subdivide an image into regions based on similarity according to a set of pre-defined criteria [11-17]. Special clustering, region growing and region splitting and merging are some examples of methods in this classification. There are some drawbacks in these methods. In region growing method, seed points and growing criteria should be determined according to the user's desire. Nevertheless, in order to segment an image correctly, the image content has to be known and almost for every image, different seed points and different growing criterion has to be determined by the user. As it turned out from the situation that is a time consuming task. In region splitting and merging method segmented image is square shaped. This is the drawback of this method. However, this method does not require connectivity between pixels. This situation makes this method be sensitive to noise.

In this study, efficient graph-based image segmentation [18] is used. Because,

1. This method is not only interested in local properties of the image but also it satisfies global properties.
2. The result of the segmentation can be adjusted according to the purpose of the user. In the result of segmentation, the user can keep only large areas or desired shapes.
3. Segmentation result is not square shape but it is square shape in the Region Split and Merge method.
4. The histogram of the image is not concerned but it is important in the Thresholding method.
5. Seed points and similarity criterion is not required, but it is necessary in region growing method.
6. This method is resistant to noise.
7. The result is efficient and fast.

5.1 Graph-Based Segmentation

Graph-based segmentation, represents the problem as a graph $\mathbf{G} = (\boldsymbol{N}, \boldsymbol{B})$. Each node $\boldsymbol{n_i} \in \boldsymbol{N}$ corresponds to a pixel in the image and edge $(\boldsymbol{n_i}, \boldsymbol{n_j}) \in \boldsymbol{B}$ binds nodes $\boldsymbol{n_i}$ and $\boldsymbol{n_j}$. The edge set **B** is constructed by connecting pairs of pixels that are neighbors in an **8-connected** sense. A weight function $\mathrm{l}(\boldsymbol{n_i}, \boldsymbol{n_j})$ is assigned to each edge based on a certain property such as measuring dissimilarity between pixels in the image. In this method, an edge weight function that depends on absolute gray level difference between the pixels connected by an edge is used,

$$\boldsymbol{l}(\boldsymbol{n_i}, \boldsymbol{n_j}) = |\boldsymbol{I}(\boldsymbol{n_i}) - \boldsymbol{I}(\boldsymbol{n_j})| \qquad \textbf{(5.1)}$$

where $\boldsymbol{I}(\boldsymbol{n_i})$ is the gray level value of the pixel $\boldsymbol{n_i}$.

This method defines a predicate to detect whether there is a boundary between regions or not using graph-based representation of the image. It compares two quantities: one of them is the intensity differences of nodes along the boundary, the other one is gray level differences between neighboring pixels within each component (internal difference).

The internal difference of a region is defined to be the largest weight in minumum spanning tree of the region, MST (R,B). That is,

$$\boldsymbol{IntDif(R) = \max_{b \in MST(R,B)} l(b)}. \qquad \textbf{(5.2)}$$

The diffrence between regions is defined minumum weight edge binding the two regions. That is,

$$\boldsymbol{Dif(R_1, R_2) = \min_{n_i \in R_1, n_j \in R_2, (n_i, n_j) \in B} l(n_i, n_j)}. \qquad \textbf{(5.3)}$$

If $\boldsymbol{R_1}$ ***and*** $\boldsymbol{R_2}$ is not binded with an edge, the difference between these regions are equal to infinitive, $\boldsymbol{Dif(R_1, R_2)} = \infty$.

The predicate that is predefined in this method examines whether there is a boundary between a pair of regions by controlling if the difference between the regions, $\boldsymbol{Dif(R_1, R_2)}$ is greater than the internal difference within one of the regions, $\boldsymbol{IntDif(R_1)}$, and $\boldsymbol{IntDif(R_2)}$.

In this segmentation method, it is possible to adjust the result of segmentation according to user's needs. For example, the user wants larger regions remain at the end of segmentation or want segmentation method prefer regions of certain shapes. This would cause the algorithm to merge regions that do not fit desired shape. For this purpose a threshold function can be defined.

This function controls the level to which the difference between components must be grater than minumum internal difference. A region comparison predicate $\boldsymbol{P(R_1, R_2)}$, is defined as,

$$\boldsymbol{P(R_1, R_2)} = \begin{cases} \boldsymbol{true}, & if\ Dif(\boldsymbol{R_1}, \boldsymbol{R_2}) > MinInt(\boldsymbol{R_1}, \boldsymbol{R_2}) \\ \boldsymbol{false}, & otherwise \end{cases} \qquad \textbf{(5.4)}$$

Minumum internal difference, MinInt, is defined as,

$$MinInt = \min\big(IntDif(R_1) + \varphi(R_1), IntDif(R_2) + \varphi(R_2)\big). \qquad \textbf{(5.5)}$$

The threshold functon that controls the level to which the difference between components must be grater than minumum internal difference is based on the size of the region and defined as,

$$\varphi(R) = t/|R| \qquad \textbf{(5.6)}$$

t is a constant parameter while |R| indicates of the size of region.

The steps of the graph-based segmentation algorithm are as follows:

1. To make the images resistent to the noise, a gaussian filter (sigma = 0.8) is used to smooth the image before calculating the edge weights.
2. The input is a graph G = (N,B), where N are the **n** nodes and B are **m** edges. The edge set B is constructed by connecting pairs of pixels that are neighbors in an 8-connected sense. Every edge has a corresponding weight that is a measure of dissimilarity between neighboring pixels. An edge weight function, l_i that based on the absolute gray level difference between the pixels connected by an edge is used.
3. Sort B into ascending order $(l_1, l_2, \dots, l_m)$.
4. Begin with a segmentation S^0, where each node n_i is in its own region. At the beginning, every pixel constitutes a single region. According to the result of comparing an edge between pixels, these regions are merged or left separate.
5. Create S^k from S^{k-1} as follows. Let R_i^{k-1} be the region of S^{k-1}that contains n_i and R_j^{k-1} the region that contains n_j. Let n_i and n_j indicates the nodes binded by the q-th edge in the ordering, $z_q = (n_i, n_j)$. If $R_i^{k-1} \neq R_j^{k-1}$ and $l(z_q) \leq MinInt\,(R_i^{k-1}, R_j^{k-1})$ then S^k is acquired form S^{k-1} by merging regions $R_i^{k-1} and\ R_j^{k-1}$. Otherwise, do nothing.
6. Return S^m after the final iteration.

This algorithm treats monochrome (intensity) and color images in different ways. Color images are processed as three separate monochrome images. For color images, the algorithm is run three times, one for each of the red, green and blue color channels. Then the results of these three channels are intersected. If two neighboring pixels are in the same region in all three of the color channel segmentation, these pixels are put in the same region.

In Figure 5.1, there is an apple with water drops and light reflection on it. Figure 5.2-5.4 shows the result of graph-based segmentation with different parameters.

This algorithm takes three parameters, which are:

- **Sigma**: Value for smoothing the input image before segmentation.
- **K** : Value for the threshold function.
- **MinReg**: Minimum region size enforced by post-processing.

In Figure 5.1, there are water drops, color variability caused by the nature of apple and illumination variation on apple. All of these factors cause the apple splitted into many regions as shown Figure 5.2. If we think that our study is not affected by the small regions created by these factors, we can make them merge with larger regions by changing the parameters.

In Figure 5.3 and 5.4, it can be seen that many small regions are eliminated. There are three distinct regions in these images: the leaf, apple stem and the apple itself.

There is a drawback of this algorithm. If background and foreground is similar, then some part of objects can be grouped together. In Figure 5.5, there is a girl whose name is Lena. Len's skin color and hat's color is very similar to the background and the object on the right. Because of this similarity factor, in Figure 5.6, Lena's hat is segmented with the wall. In addition, her face and her shoulder are clustered with the object on the right of Lena. If the parameters are changed, almost coarse silhouette of Lena appears in Figure 5.7.

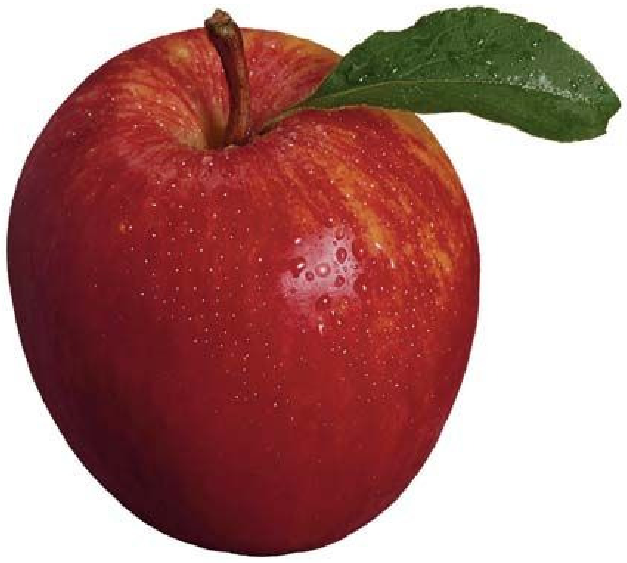


Figure 5.1: Original Image.

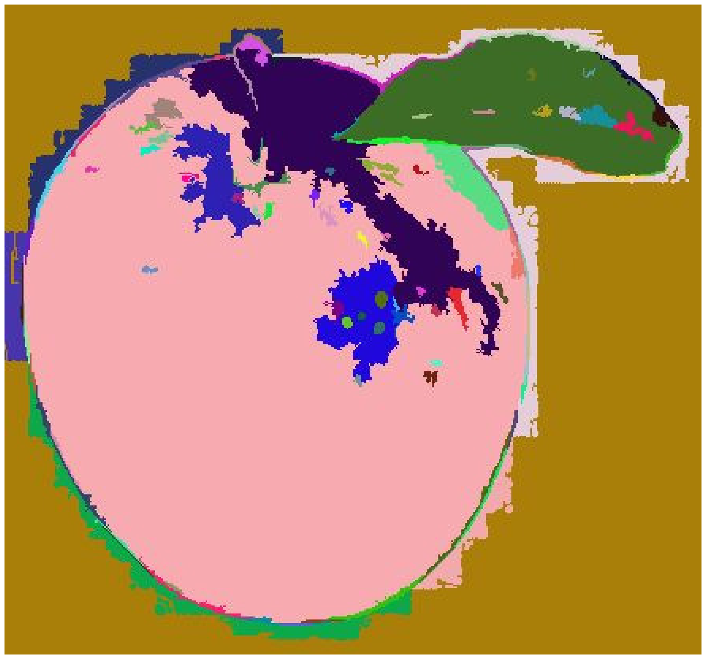


Figure 5.2: Graph-Based Segmentation with σ = 0.8, t = 500, MinReg= 20.

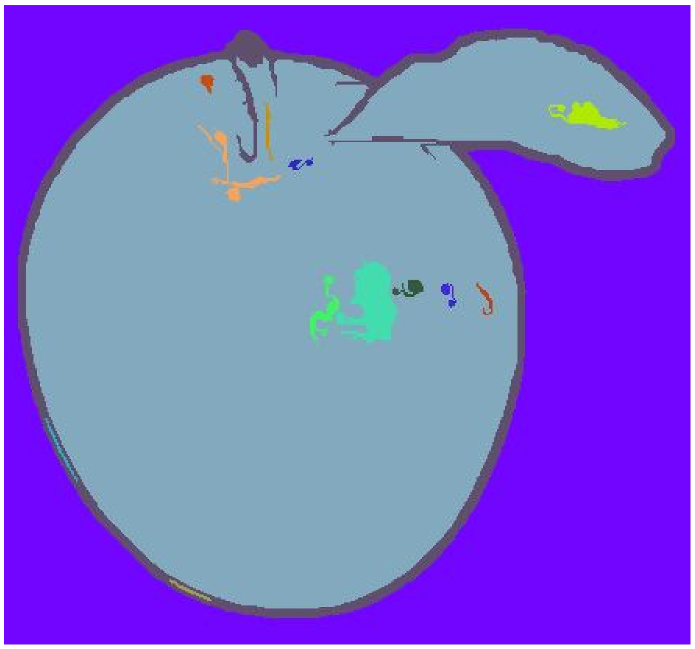


Figure 5.3: Graph-Based Segmentation with σ = 0.8, t = 750, MinReg = 40.

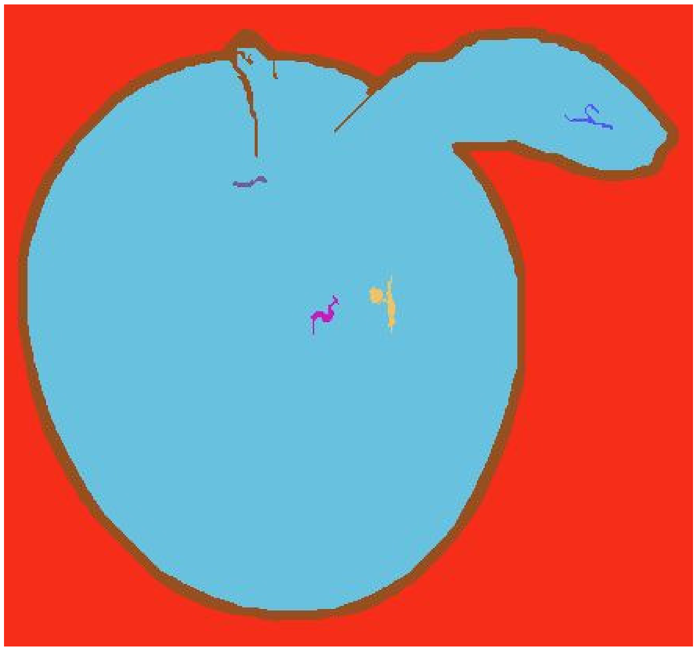


Figure 5.4: Graph-Based Segmentation with σ = 0.8, t = 100, MinReg = 50.

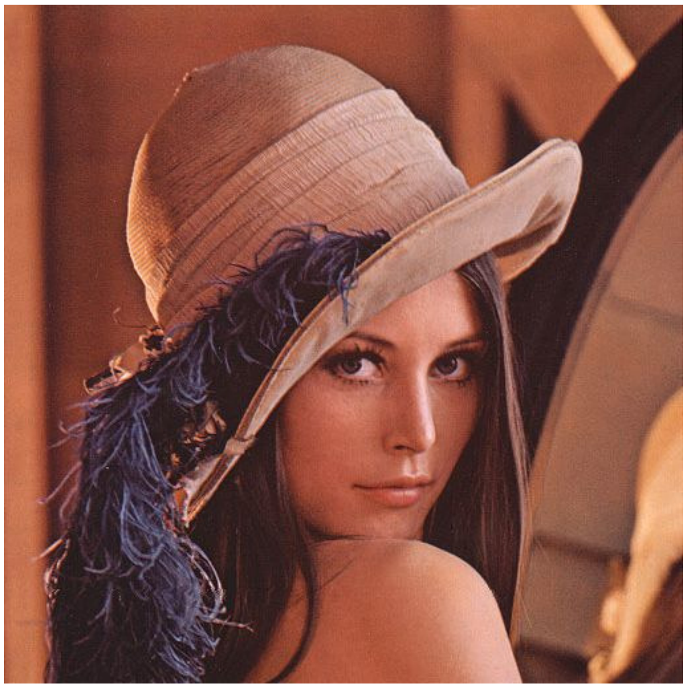


Figure 5.5: Original Image.

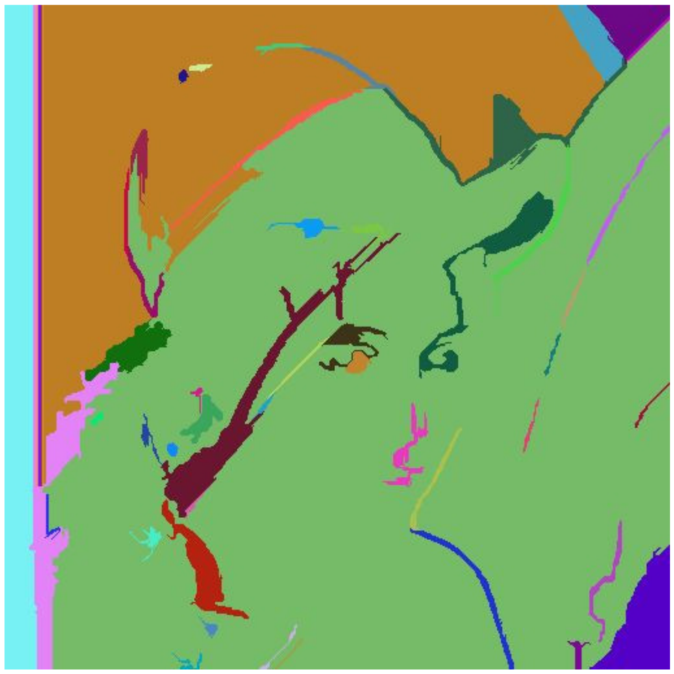


Figure 5.6: Graph-Based Segmentation with σ = 0.8, t = 700, MinReg = 50.

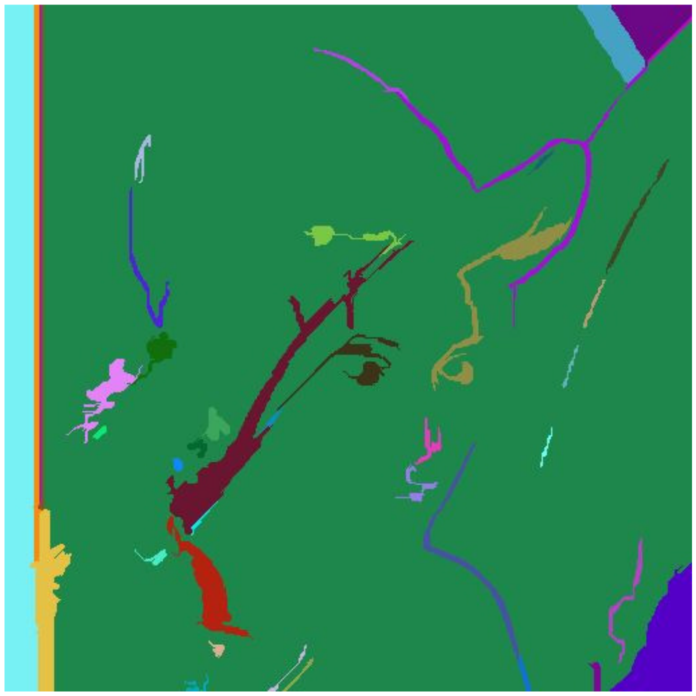


Figure 5.7: Graph-Based Segmentation with σ = 0.8, t = 1000, MinReg = 50.

6. COLOR PERCEPTION AND REDUCTION

After extracting the objects from the environment, appropriate color segments should be determined for the optimum set of colors that can be used by a robot.

This issue is related with the color reduction. Color reduction is an action of representing a full color image to an image with a smaller number of colors, by substituting each original image color with the closest color from the reduced set of color. The aim of this process is to minimize the perceived distortion by human eye while reducing the representative image colors.

In color reduction process, maybe the easiest way of representing full image colors with reduced number of colors is to make zero the least significant bits in RGB-Red Green Blue values. Although this algorithm is fast and simple, it's results are insufficient for many applications.

Recently, several algorithms for color reduction have been developed. Some common algorithms are median-cut [19], octree [20], and variance-based algorithm [21]. These algorithms splits the color space into smaller regions. However, their drawbacks differs in implementation difficulty, producing colors with low contrast, losing memory and time . Neural network color quantization algorithm [22] provides good results, but it's computation is very expensive.

We use K-means clustering [23] for color reduction in this research. Because it is simple to implement, robust and commonly used in various fields of study.

The K-means is a clustering method that tries to find clusters in the data. K-means is used first to find the clusters and then look for the pattern in each cluster. Because the K-means is not exactly a pattern recognition method.

If the number of clusters in the data is not known (K), the user has to make a guess and adjust this number later, if necessary. The method interactively searches for K centers of mass inside the data. This is one of the most used techniques for grouping.

In K-means clustering, initial partition is done according to the K centers of mass and this partition can quitely affect the final clusters.

The K-means clustering algorithm itself works as follows :

1. Input: data and the number of groups (K).

2. K centers of mass in the data is defined, in other words, the dataset is divided into K clusters and the data points are randomly assigned to the clusters so that all clusters have roughly the same number of data points (Figure 6.1).

3. The distance from each data point to each cluster is computed. If the data point is closer to its own cluster, it is left in its initial cluster. Otherwise, it is moved into closest cluster (Figure 6.2).

4. Compute the new center of clusters (Figure 6.3).

5. If the error between the previous centers of masses and the new ones is below a certain limit accept and terminate (Figure 6.4), otherwise return to step 3.

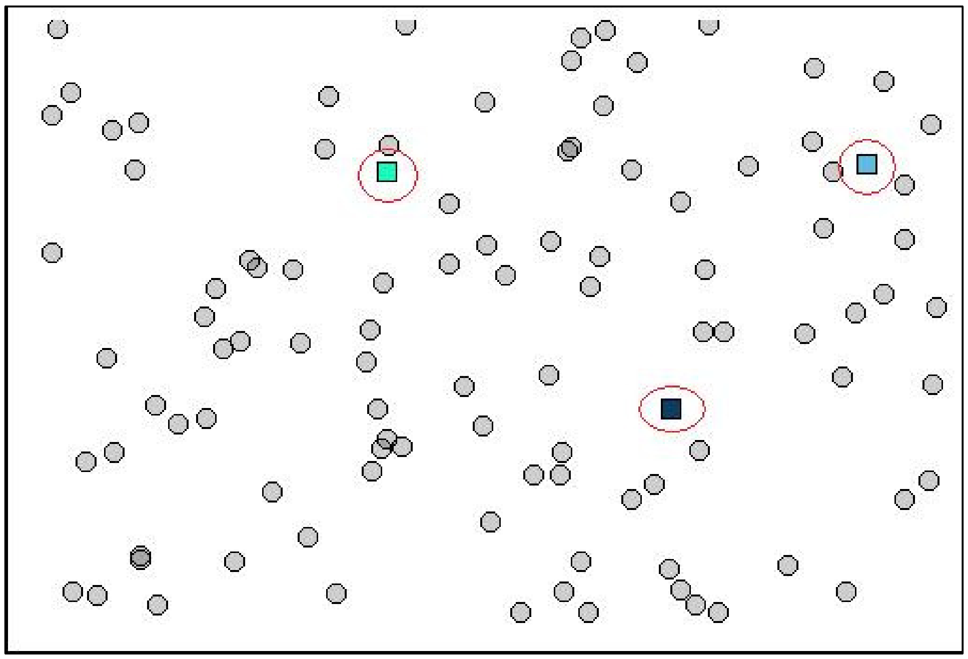


Figure 6.1: 3 Center of Mass is determined initially for K-means Algorithm (This dataset has 100 data points and 3 cluster).

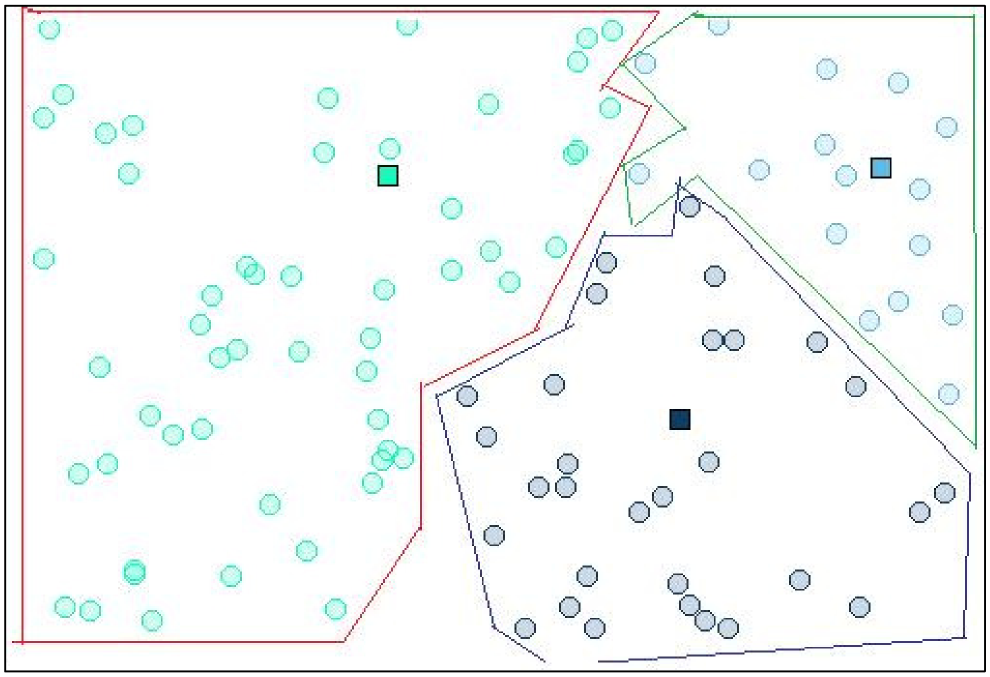


Figure 6.2: Associate each data point with a center of mass.

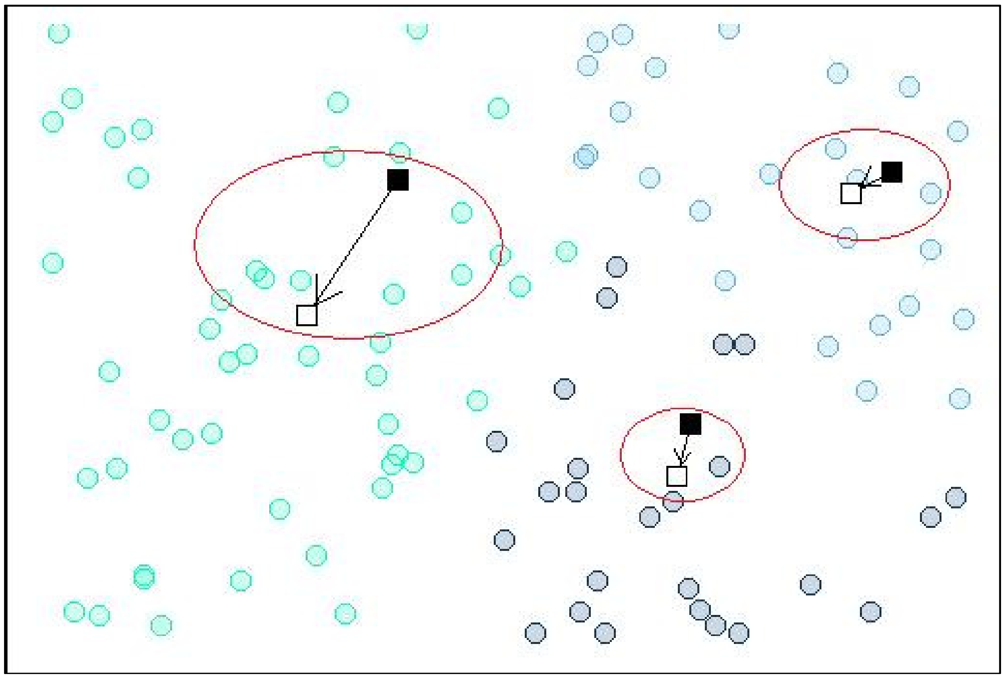


Figure 6.3: The new center of mass is computed.

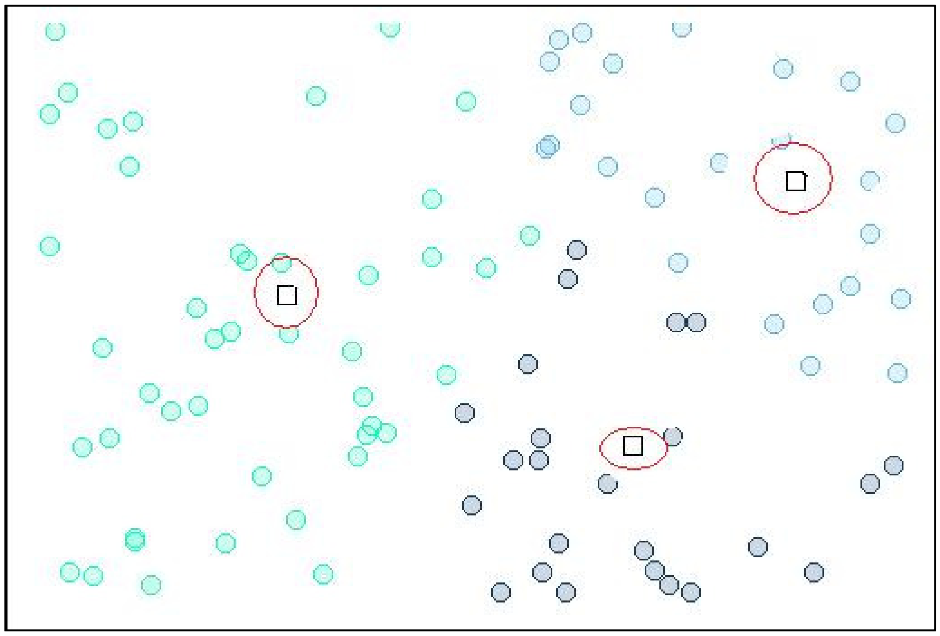


Figure 6.4: The Result of K-Means Clustering.

Figure 6.5 shows the image we used in our study in Webots environment and Figure 6.6 shows the result of K-Means clustering used for color reduction.

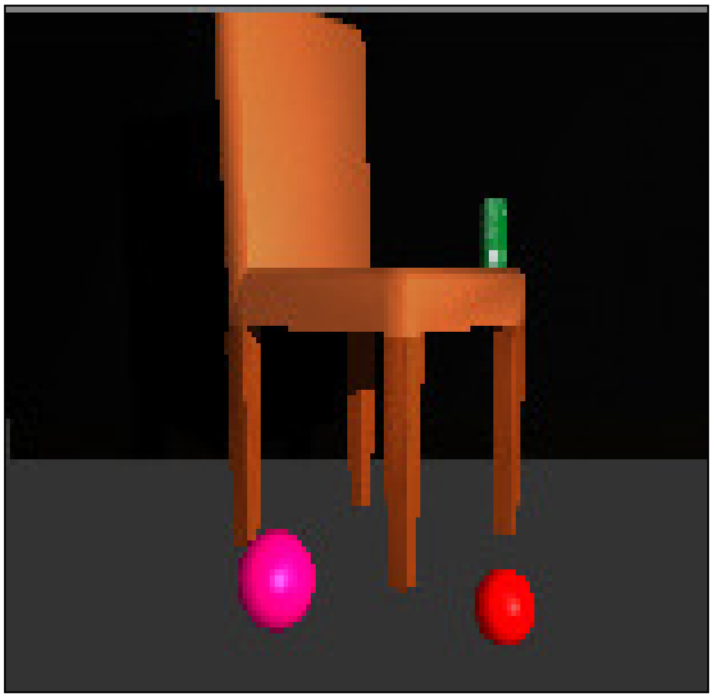


Figure 6.5: Original Image.

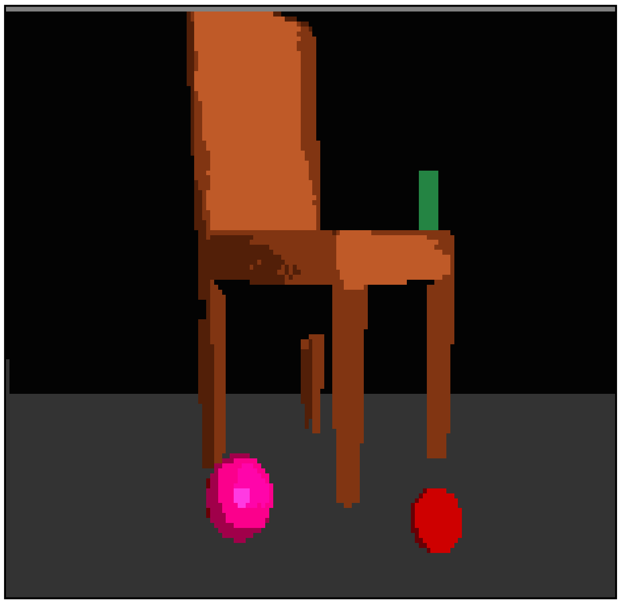


Figure 6.6: The Result of K-means Clustering
(Full Image Colors are reduced to 16 colors).

7. CONTOUR MATCHING

The result of segmentation of an image is a set of groups of pixels that each group represents a region. These segmented regions can be transformed into a compact way that facilitates the region's description and help to compare and match with a given pattern. In order to convert a region to a compact object, contours in the image should be found.

Contour is an outline or a boundary of an object. It appears the places where most change happens. Especially, these changes are seen on the edges.

Contours are usually computed from a grayscale image where it is easier to define contrast. So it is better to convert the color image to the grayscale image [24] (Figure 7.1). The conversion can be realized as follows:

```
for(i=0; i<sourceimage->height; i++)
        for(j=0; j<sourceimage->width; j++)
                destinationimage[i][j]= (uchar)(sourceimage[i][j].b*0.114 +
                sourceimage [i][j].g*0.587 +
                sourceimage [i][j].r*0.299);
```

In the grayscale image, edges are detected using Canny edge detector algorithm [25]:

1. Calculate first derivatives of the image in both x and y direction.

$$P_x = G_\sigma^x * P \qquad P_y = G_\sigma^y * P \tag{7.1}$$

2. Calculate the magnitude of gradient at every pixel.

$$Mag(x, y) = |\nabla P| = \sqrt{P_x^2 + P_y^2} \tag{7.2}$$

3. Select only those pixels that are local maxima of the magnitude in the direction of the gradient.
4. Apply hysteresis thresholding.
   a. Select the pixels such that $Mag(x, y) > T_h$ (high threshold).
   b. Gather the pixels such that $Mag(x, y) > T_l$ (low threshold) that are neighbors of already gathered edge points.

A contour can be represented in different ways; the most common way to represent a contour is Freeman chain codes [24].

Freeman chain code represents a boundary by a connected sequence of straight line segments, each with a specified length and direction. It uses a 4- or 8-connectivity and the segment's direction is coded following a numbering code (Figure 7.2).

In this study Freeman chain code is used to represent contours. After edge detection in the grayscale image, contours are detected using Freeman chain code.

When contours in the image are found, the objects to paint are also found (Figure 7.3).

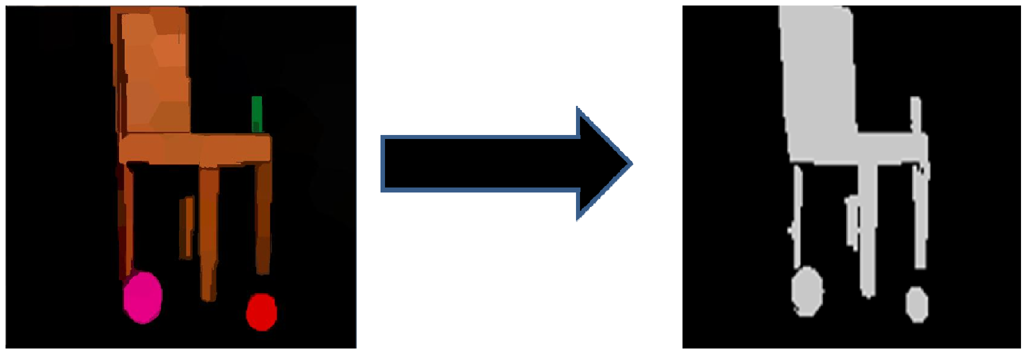


Figure 7.1: Converting Original Image to Binary Image.

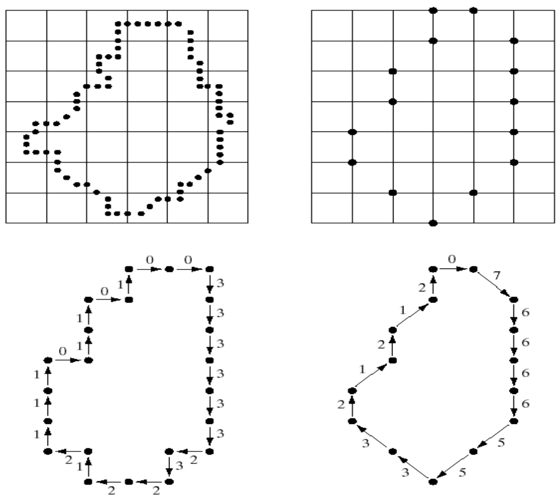


Figure 7.2: Freeman Chain Code.

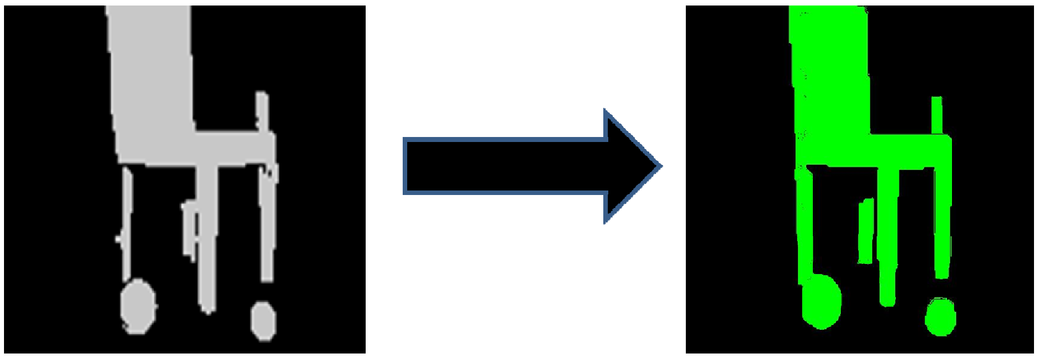


Figure 7.3: (a) Binary Image (b) Contours in the image.

8. ORIENTATION FOR BRUSH STROKE

Until this point, NAO obtains the objects to paint and percepts the color of the objects. Before starting to paint there is only one process which maps the question that is in which direction NAO moves its brush. This issue is related with the orientation.

In this stage, it is required to guide NAO's brush strokes to paint properly. For this purpose, gradient information to lead brush strokes that is robust to texture can be used [26].

NAO computes the normal orientation at every pixel in image using gradient operator.

$$\theta = \arctan\,(dy/dx) \tag{8.1}$$

θ is betwwen $(-\pi/2, \pi/2)$.

Generally, a human artist does not paint at once. He/she starts painting with a large brush to paint rough scene and pass over the painting again to add fine details.

In our research, painting process is carried out layer by layer. These layers are determined by the number of paint brushes. In the first layer, largest brush is used to paint roughly and in the other layers smaller brushes are used to add details.

A brush stroke in the image consists of breaking points that forms curve, color and a brush radius and is painted as follows:

- Start point (x_0, y_0), of the brush stroke is placed on the canvas and the color of that point is used for the curve's color. (Figure 8.1.(a))
- Next breaking point is computed in the direction, S_0, which is normal to the calculated gradient, G_0, on previous point along the curve. From the second point, there are two directions normal to the gradient, but $(\theta + \pi/2)$ is selected. (Figure 8.1.(b)).
- The distance between two points are determined by the brush radius.
- The other points are calculated by repeating the actions above.

- Painting stroke is stopped when the length of stroke is reached the predefined stroke length.

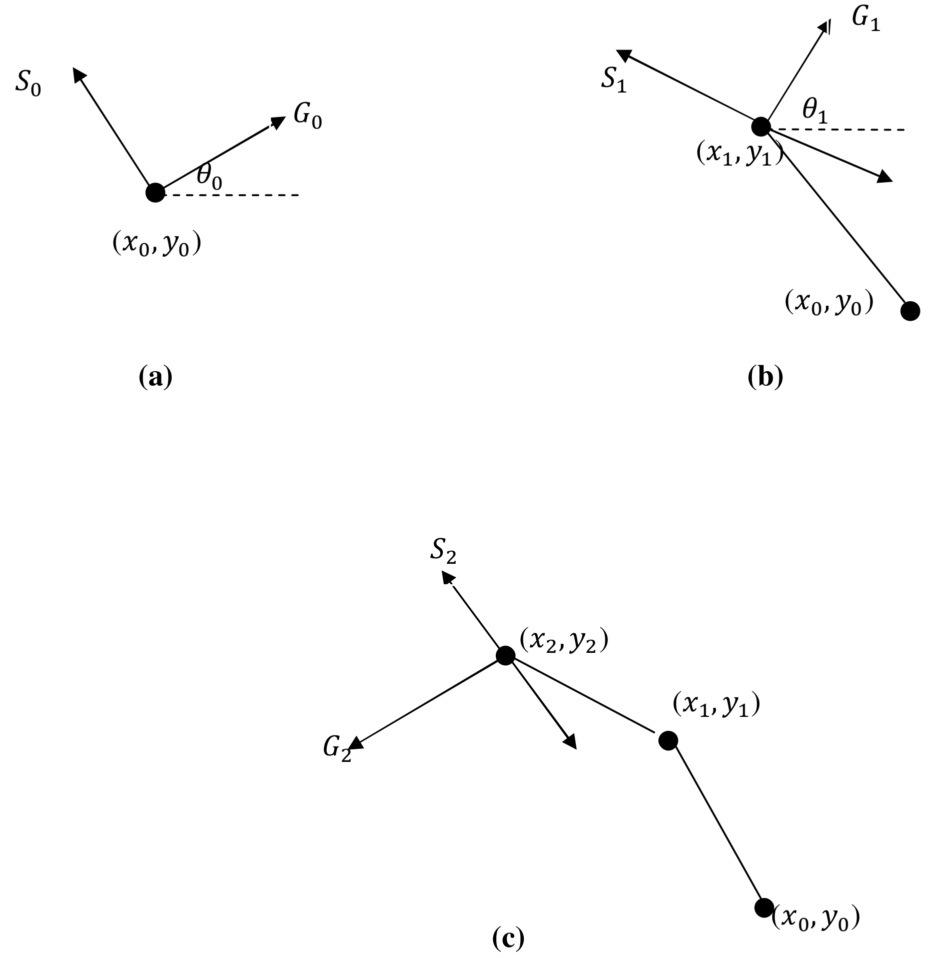


Figure 8.1: (a) Start Point of Brush Stroke (b) Brush Stroke continues in the direction of the normal to the gradient (Second Point) (c) Third Point of Brush Stroke.

9. PAINTING BY ROBOT

Human painters generally paint pictures step by step. He/she starts painting with a large brush to paint rough scene and pass over the painting again to add fine details.

In our research, painting process is carried out layer by layer. These layers are determined by the number of paint brushes. In the first layer, largest brush is used to paint roughly and in the other layers smaller brushes are used to add details.

After objects are extracted from image, NAO records this image as a target image. NAO's final goal is to paint this target image on the canvas. As indicated before, NAO paints image layer by layer. At every layer, NAO applies Gaussian blurring to the image with

$$sigma_i = blurFactor\ x\ r \qquad \textbf{(9.1)}$$

where sigma is the sigma of Gaussian, blurFactor regulates sigma and its value of 0.5 is used in this study and r is the pre-computed radius of the brush. Then the robot enregisters this image as a referenced image in its mind.

Before NAO starts drawing, it tries to find an area in each color segment derived from color perception section that contains a number of pixels larger than [27],

$$Region_i = t\ x\ r^2 \qquad \textbf{(9.2)}$$

where **r** is the pre-computed radius of the brush, and **t** is the constant that must be depending on the drawing style.

If the robot finds such an area, it calls its assistant for selected color and begins filling this area with paint using paintbrush. It moves its arm to the start position and pushes the brush down. Then it moves its arm in the direction of the brush stroke which is calculated by the robot as indicated in section 8 to the end position that is the maximum length of brush stroke and pulls the brush up.

In the begining, the canvas is white which means it is empty. Every time NAO draws a brush stroke on the canvas, it compares the referenced image at any layer with the image on the canvas and decides whether it is time to stop and change the brush size or not.

For the next time, the robot looks for a new area with the same color. All of the color in one layer is painted, the next layer start with a thinner brush to add fine details. The number of layer is related with the brush number used for painting.

In our system, it is important to paint in a stable direction. The grasping becomes unstable when a brush moves from the near side to far side. But it is stable when a brush moves from left to right and from far side to near side.

10. EXPERIMENTS

10.1 Webots

Webots [28], is a development environment used to model, program and simulate mobile robots. It offers a rapid simulating environment that allows the user to create 3D virtual worlds close to the real worlds with physics properties such as mass, joints, friction coefficients, etc.

With Webots the user can build complicated robotic setups, with similar or different robots in the same environment. The properties of each object, such as shape, color, texture, mass, friction, etc. are determined by the user. A large choice of simulated sensors and actuators is feasible to equip each robot. The robot controllers can be programmed with the built-in IDE-Integrated Development Environment or with third party development environments. The robot behavior can be tested in physically realistic worlds.

The controller programs can be transported to real robots.

The user can do with Webots :

- Mobile robot building for academic research, the automotive industry, the toy industry etc.
- Multi-agent research like collaborative mobile robots groups,
- Adaptive behavior research like AI-Artificial Intelligence, genetic algorithm,
- Teaching robotics like robotics lectures,
- Robot contests.

10.2 Painting Experiment with Webots

We simulated this painting art in a generic environment in Webots with NAO as in Figure 10.1. In NAO's world, a chair, a can on the chair and two balls of different sizes in front of the chair are located.

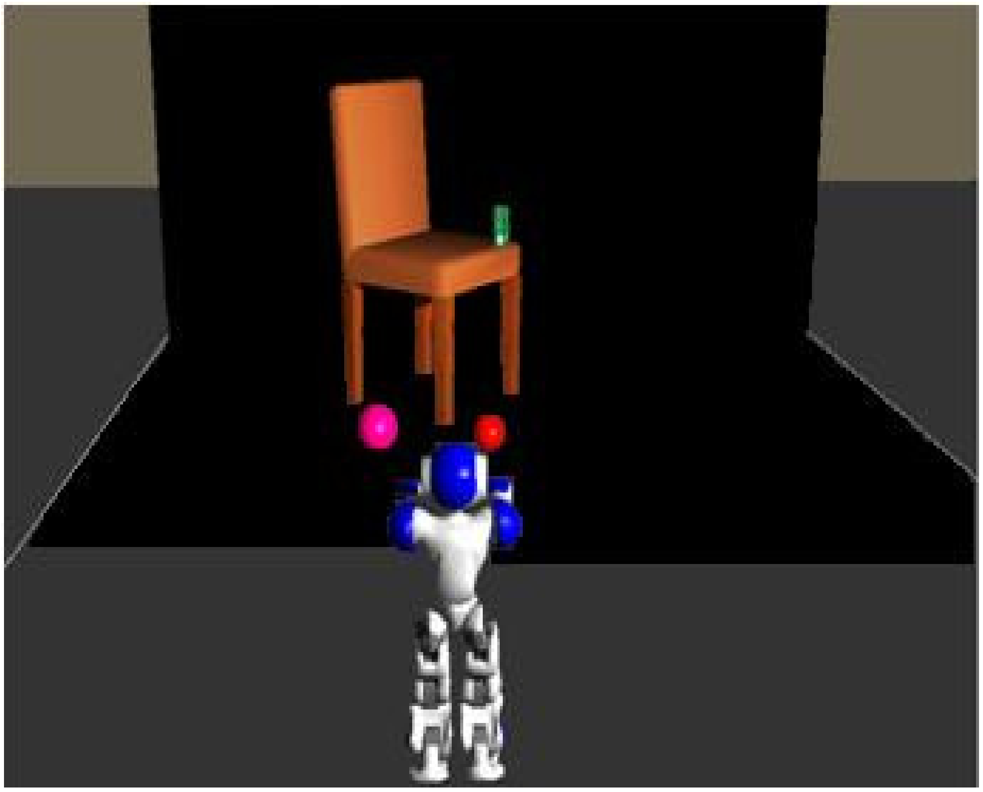


Figure 10.1: NAO in Webots.

The humanoid robot paints objects on a canvas in three phases. In the first phase, we obtain the image of the environment in which the objects that we are interested in through the humanoid robot’s camera on the forehead (Figure 10.2).

Human artists generally focus on specific areas of a painting in which details in each area are different. For example, the objects that are closer are painted detailly while far objects are painted roughly. We apply this basic observation about the actions of a human painter little differently to a robot to make generated paintings seem more meaningful. We want robot paints only the objects in the image.

In the second phase, graph-based segmentation is applied to the image to find and group the foreground and background pixels (Figure 10.3). Then we find active counters in the image to extract the objects that the humanoid robot paints (Figure 10.4).

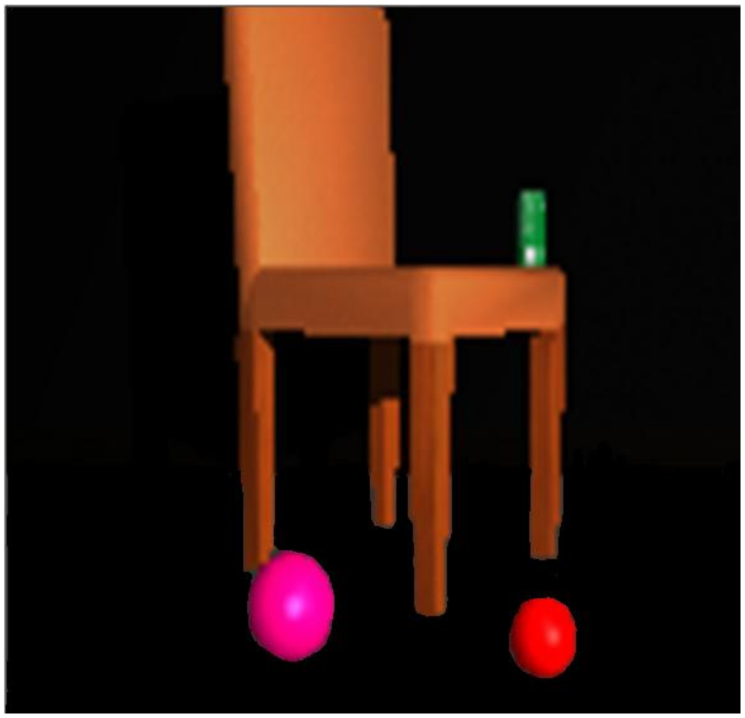


Figure 10.2: Obtaining Image Of The Environment by NAO.

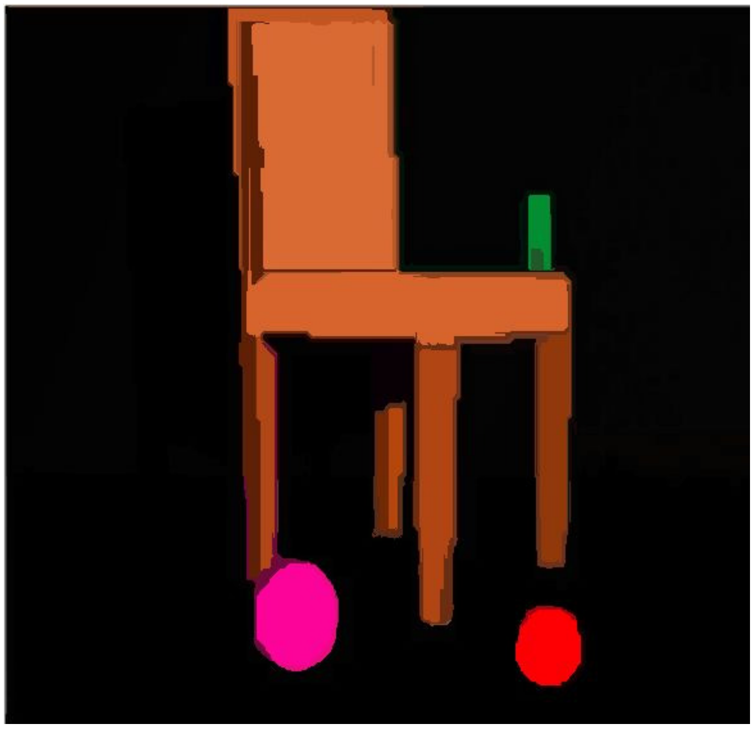


Figure 10.3: The Result Of Graph-Based Segmentation.

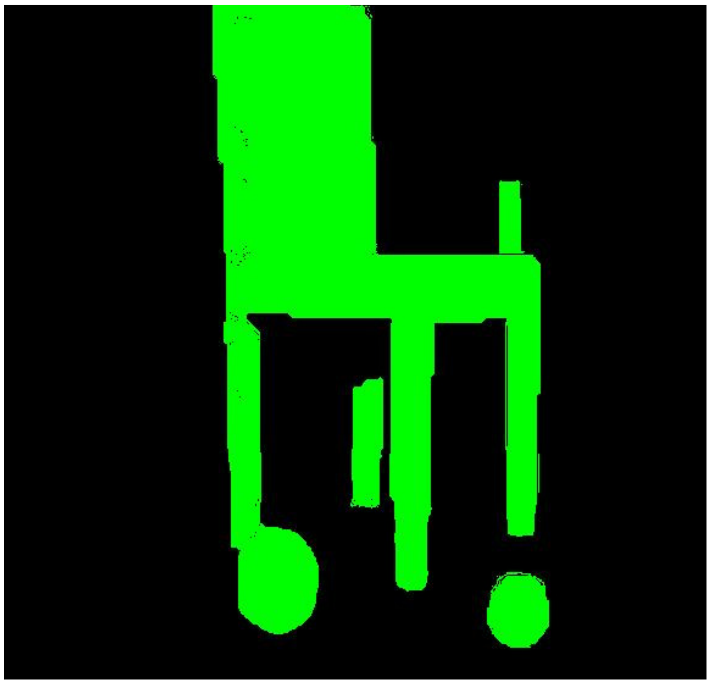


Figure 10.4: The Result Of Active Contour Method.

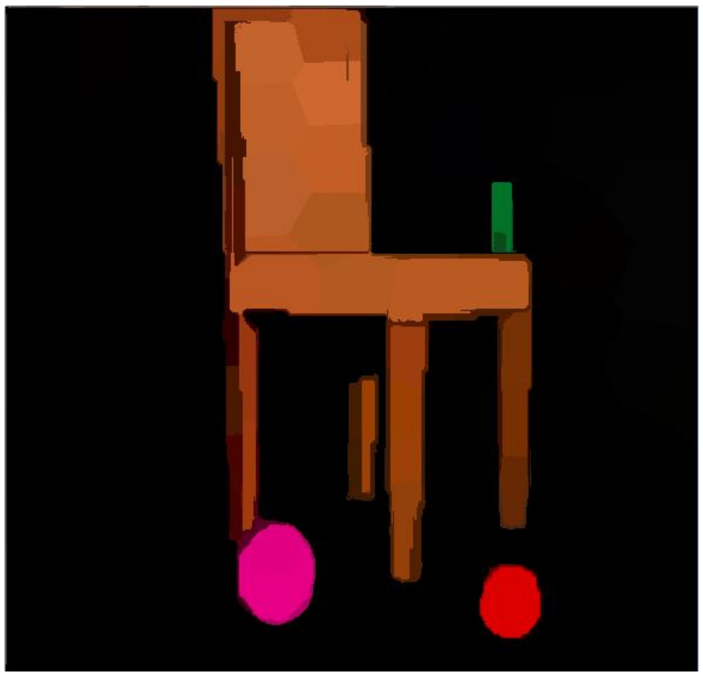


Figure 10.5: The Result Of Color Reduction with K-Means Algorithm.

After the objects are extracted, the robot must represent the object's color using a proper set of colors. In fact, this problem is related with the color reduction. We achieve color reduction with K-Means algorithm (Figure 10.5).

In the third phase, painting by a robot with fingers is realized. In the simulator, three brushes with different sizes are used. Their radii are 16 pixel, 8 pixel and 4 pixel, respectively. The results of each brush are below (Figure 10.6-10.8).

In order to give a hand-made effect to the picture, a circle with the brush radius is located at every breaking point of brush stroke.

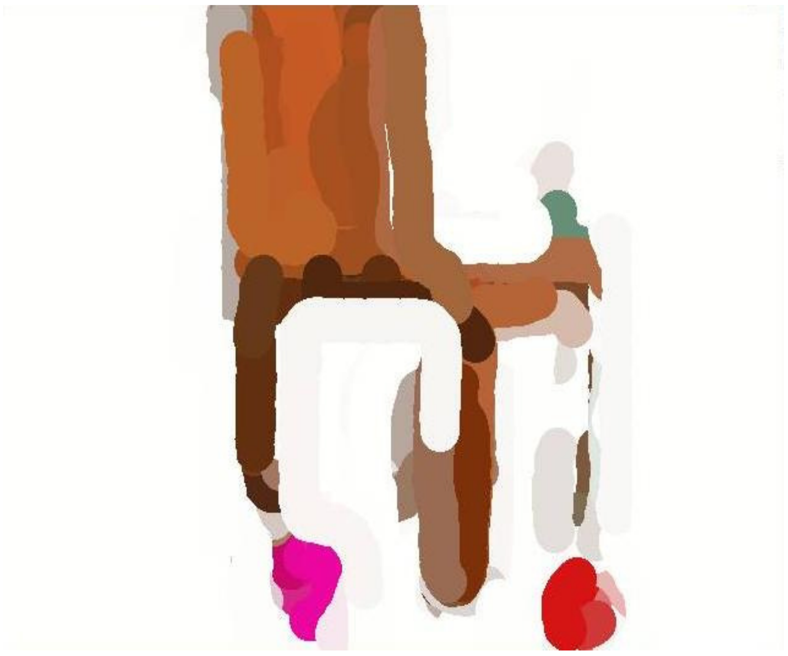


Figure 10.6: The Result of First Brush (Radius with 16 pixel).

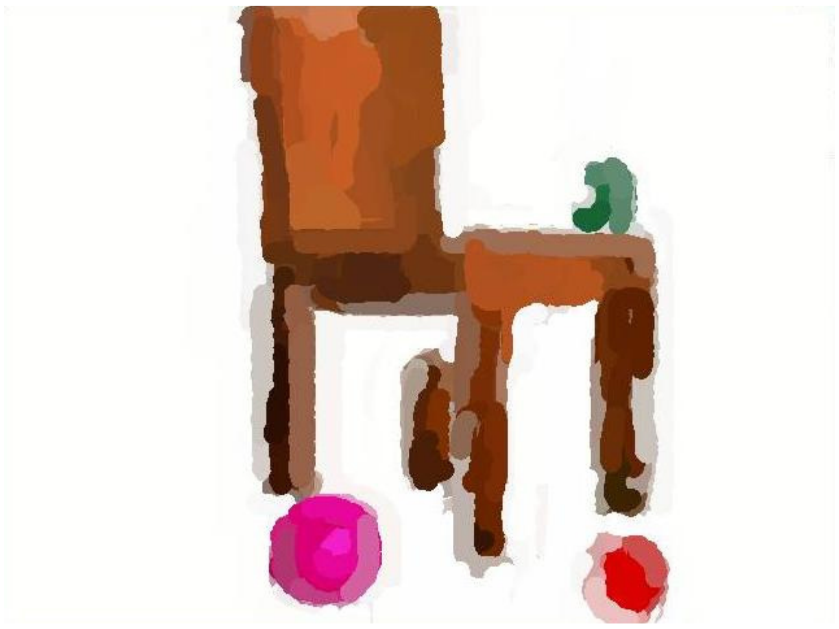


Figure 10.7: The Result of Second Brush (Radius with 8 pixel).

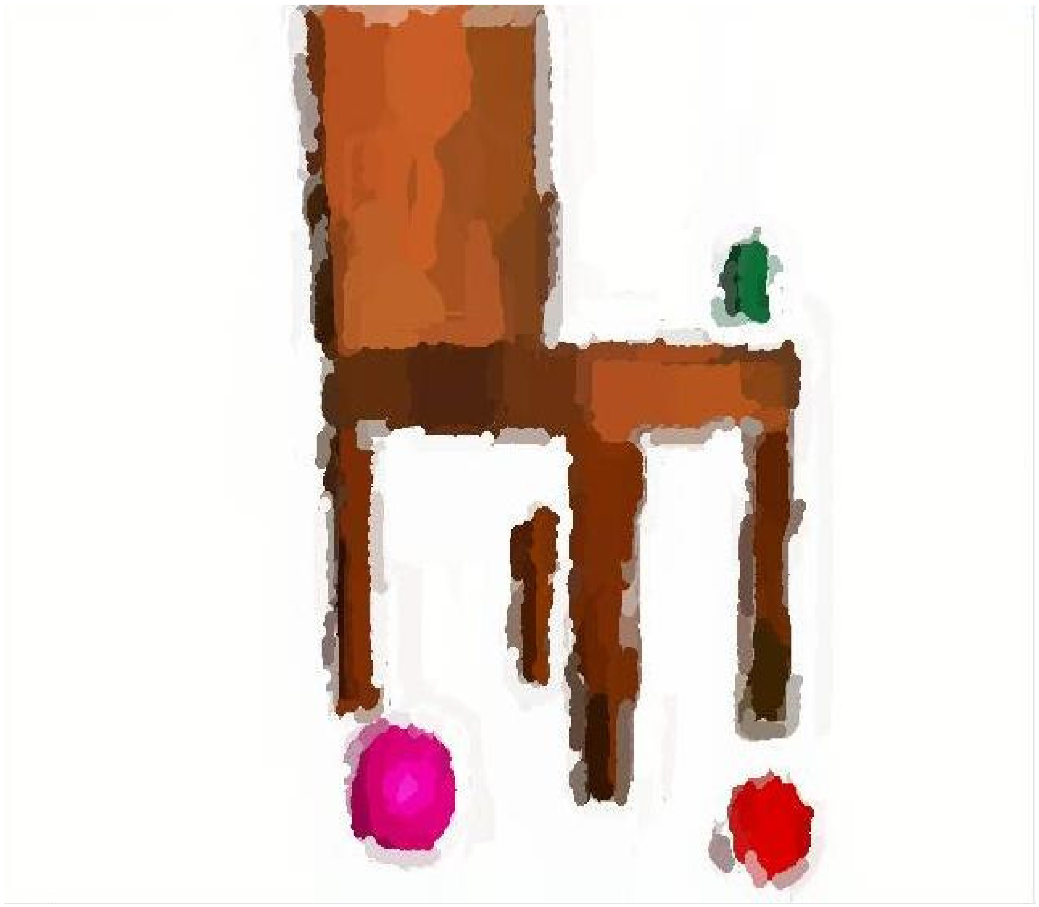


Figure 10.8: The Result of Third Brush (Radius with 4 pixel).

10.3 Painting Experiment with Real NAO H25 Humanoid Robot

One of the important issue in painting with real NAO is the manipulation of a paintbrush by robot's fingers. NAO H25 humanoid robot doesn't have any force and tactile sensors on its fingers and on the palm of its hands. Because of this constraint, NAO is not be able to grasp the paintbrush itself. It, first, open its hand, then its assistant fix the paintbrush in robot's hand. Paintbrush is thin to be hold tightly in robot's hand, so we cover the outside of brush with sponge (Figure 10.9).

The details of physical and kinematic problems are still worked ng on the real robot. Currently, the robot paints on a table top, using its arm like a plotter on an x-y plane and moves its upper torso with the arm to reach different locations on the table (Figure 10.10).

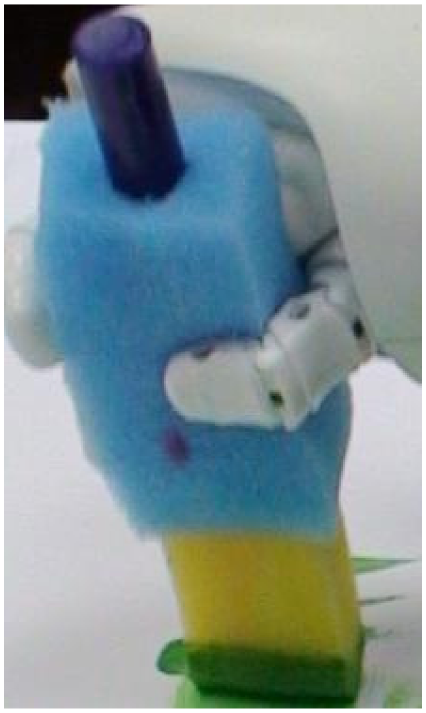


Figure 10.9: Paintbrush.

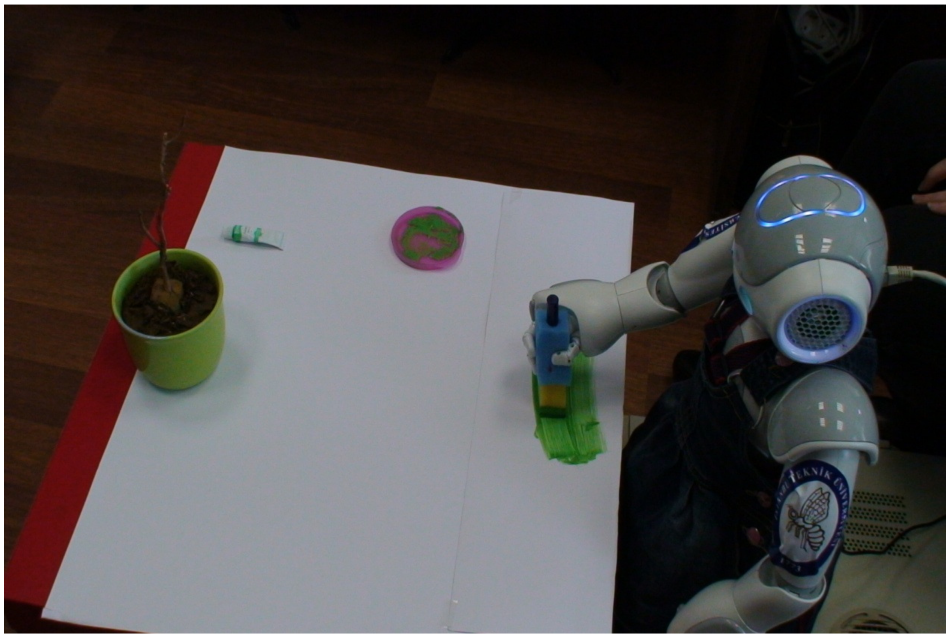


Figure 10.10: Painting with real NAO.

10.4 Controlling NAO with Choregraphe

Choregraphe [29], is a platform that allows users :

- Creating animations and behaviors for NAO,
- Examining the created animations and behaviors on a simulated robot before moving them to the real NAO,
- Monitoring and controlling NAO.

Choregraphe lets the users make very complicated behaviors or animation without writing a code. However, NAO is accessible via C++ and python.

The behaviors created with Choregraphe are interpreted with NAOqi that is loaded on NAO and programs and controls real NAO.

When Choregraphe starts, the following interface meets the user (Figure 10.11).

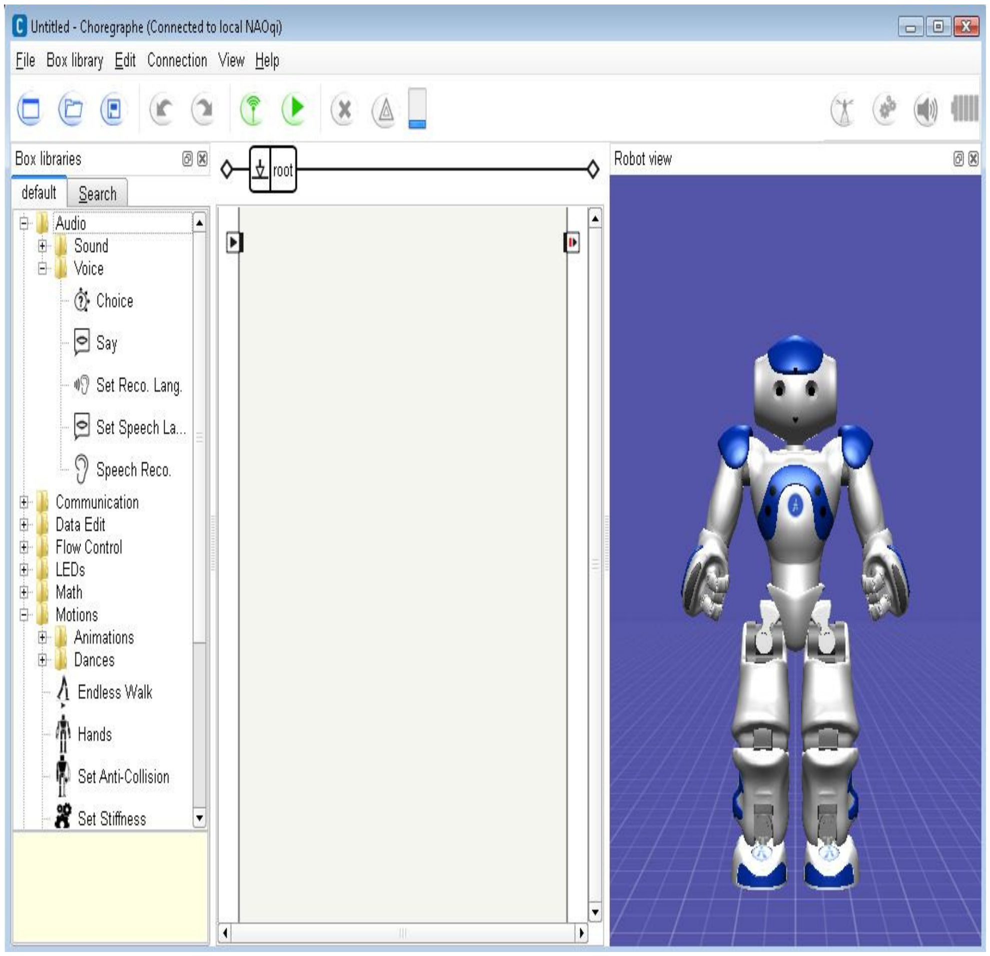


Figure 10.11: Choregraphe's Starting Interface.

Choregraphe's interface consists of following panels (Figure 10.12) :

1. The Box Libraries Panel,

2. The Flow Diagram Panel,

3. Robot View Panel,

4. Video Monitor Panel

5. Pose Library Panel

6. Project Content Panel

7. Behavior Manager Panel

8. Debug Window Panel

9. Script Editor Panel

10. Undo Stack Panel

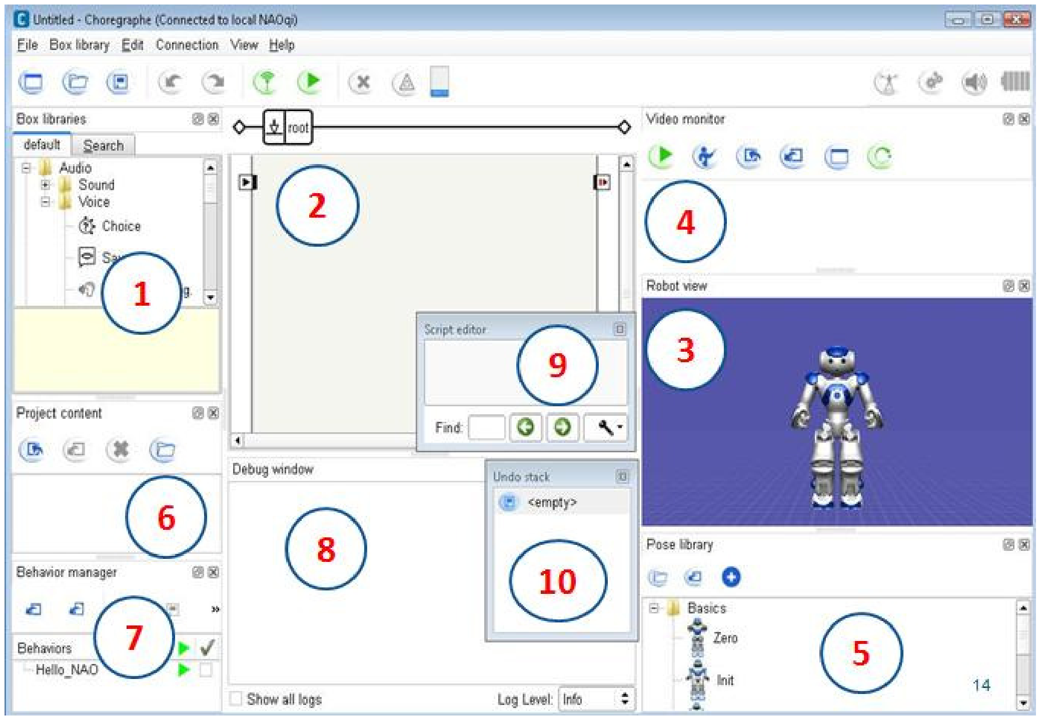


Figure 10.12: Whole Panels of Choregraphe.

10.5 Creating Behavior for NAO

As shown in Figure 10.9 with 1, there is a built-in box library in box library panel. To create a behavior, user must drag and drop boxes from a box library to the flow diagram, and connect them to do the behavior in that box. For example, if the user wants NAO to say "Could you give me the pink color", he/she has to :

- Chose Audio->Voice->Say box from box library,

- Drag and drop the say box from box library to the flow diagram (Figure 10.13),

- Connect the play button ▶ to the beginning of the behavior (Figure 10.13),

- Double click the say box to get inside the box and change the text in the Localized Text box to make NAO say what you want (Figure 10.14).

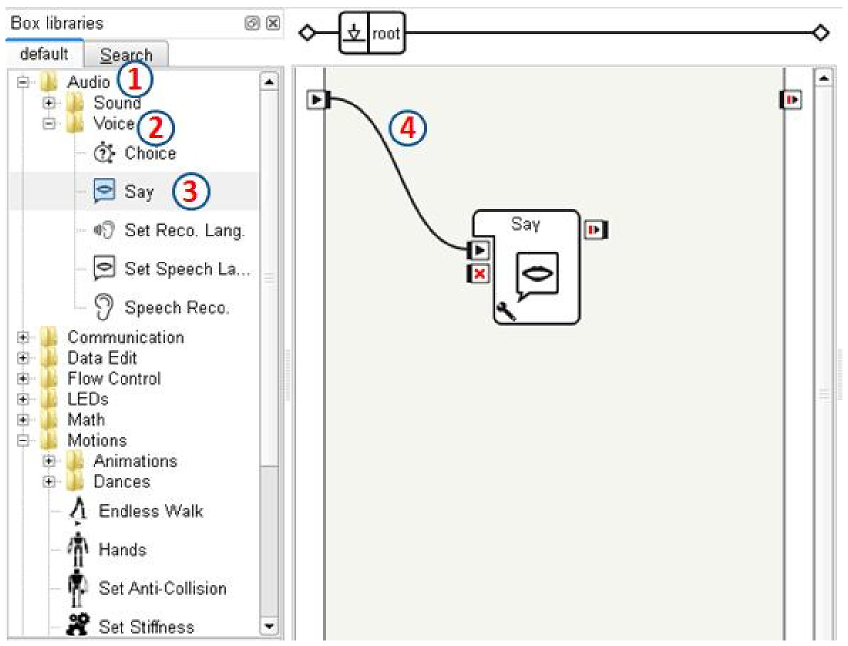


Figure 10.13: How To Add a Say Box.

10.6 Creating Own Behavior for NAO

Choregraphe platform allows users to create their own behaviors that are convenient to their purpose instead of using built-in box library.

For instance, there is no avaliable box for moving NAO's hand from left to right by holding the brush in its hand. To realize this behavior, the user has to create his own box. The steps for creating a box are as follows (Figure 10.15):

1. Right click on mouse and select "Add a new box" option,

2. Write the name of the box and give a brief description of the box,

3. Add an image for the box (optional),

4. Write the parameters of the box for input and output,

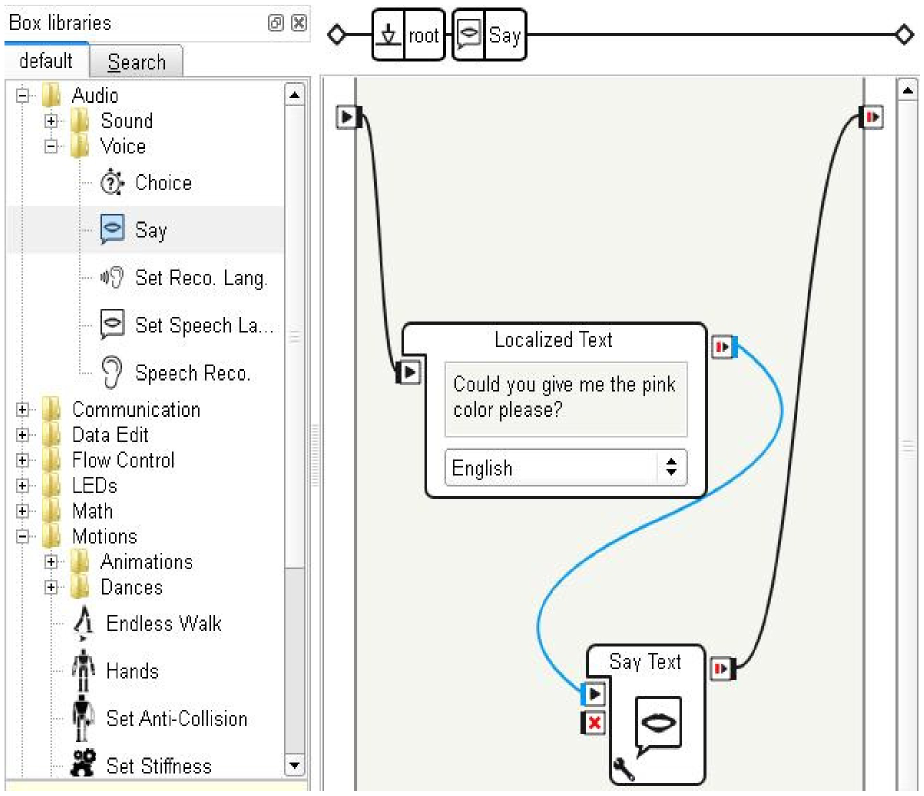
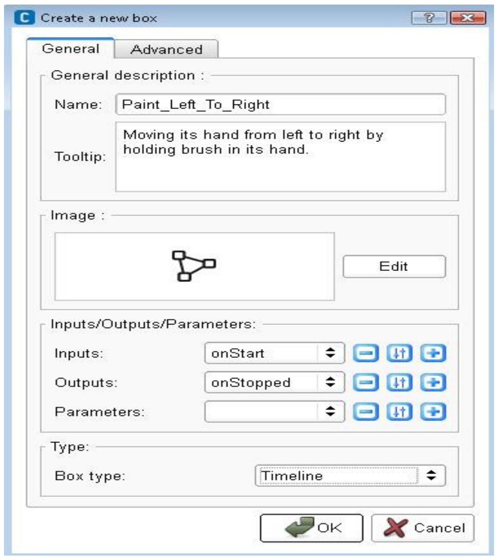


Figure 10.14: How To Add Text into the Say Box.

5. Determine the type of the box:

   a. Script (Figure 10.16) : The Script is in Pyton, so any Pyton module can be imported and any Python function can be used to make NAO do the behavior that the user wants.



General description :

Name: Paint_Left_To_Right

Tooltip: Moving its hand from left to right by holding brush in its hand.

Image :

Inputs/Outputs/Parameters:

Inputs: onStart

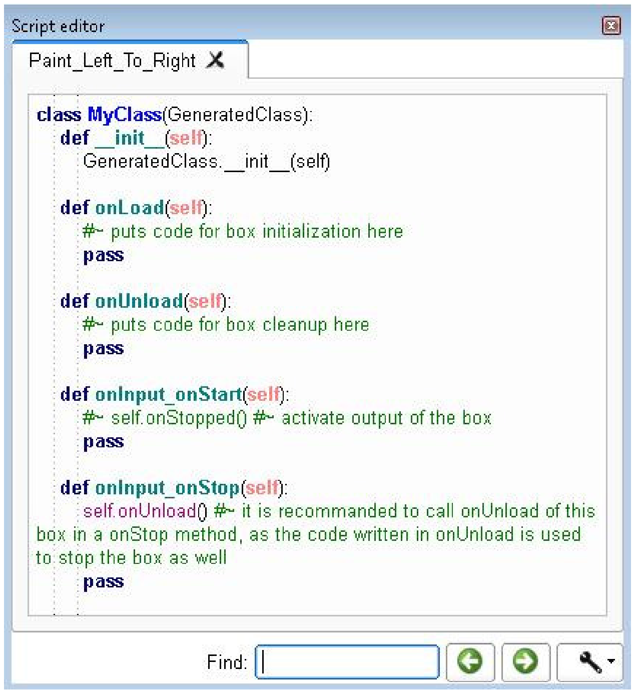
Outputs: onStopped

Parameters:

Type:

Box type: Timeline

Figure 10.15: Creating A New Box.



```
class MyClass(GeneratedClass):
    def __init__(self):
        GeneratedClass.__init__(self)

    def onLoad(self):
        #~ puts code for box initialization here
        pass

    def onUnload(self):
        #~ puts code for box cleanup here
        pass

    def onInput_onStart(self):
        #~ self.onStopped() #~ activate output of the box
        pass

    def onInput_onStop(self):
        self.onUnload() #~ it is recommanded to call onUnload of this
box in a onStop method, as the code written in onUnload is used
to stop the box as well
        pass
```

Figure 10.16: Script Box Type.

b. Flow Diagram (Figure 10.17) : In this type, users do not deal with scripts or codes. They only order the behaviors and connect them together respectively.

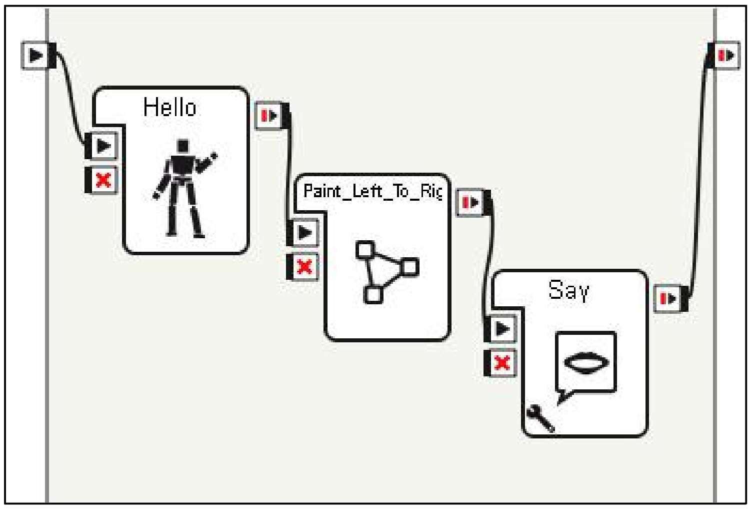


Figure 10.17: Flow Diagram Box Type.

c. Timeline (Figure 10.18) : Timeline allows the user to program every part of NAO like hands, legs etc. in different time frames. In this example, we use timeline, because moving from left to right is a continues action and it requires different motions in different time frames.

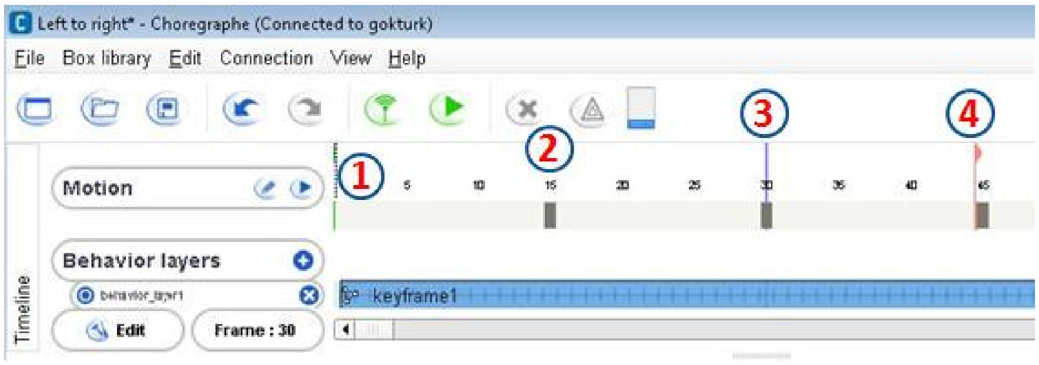

Left to right* - Choregraphe (Connected to gokturk)
File Box library Edit Connection View Help
1
2
3
4
Motion
Behavior layers
behavior_layer1
Edit
Frame : 30
keyframe1
Timeline

Figure 10.18: Timeline- (1) represents start frame, (2) represents motion keyframe, (3) represents currently selected fame and (4) represents end frame.

As shown in Figure 10.18, right click at any time label, the user can create a behavior using NAO's different parts (Figure 10.19).

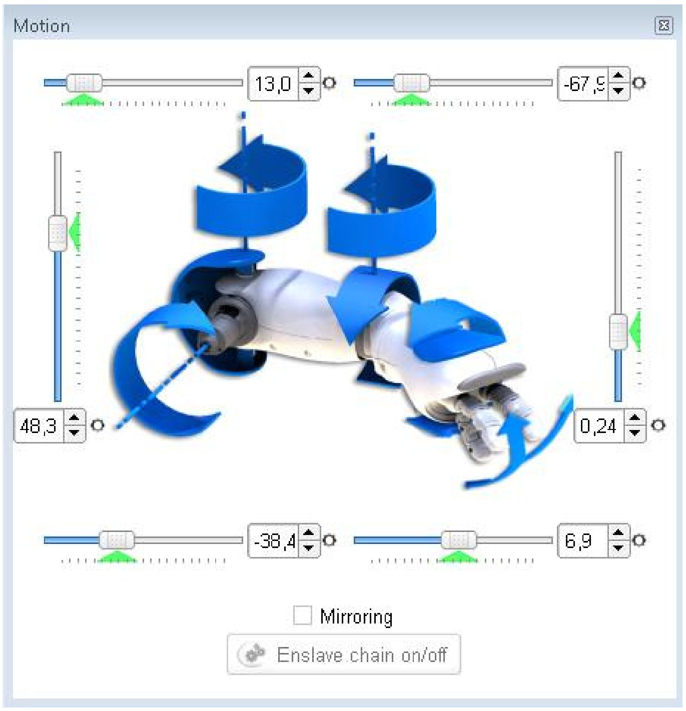


Figure 10.19: Motion Parts of NAO used while forming motion at timeline.

10.7 Recognizing Objects

NAO can learn faces and objects. For the next time, NAO sees that face or object, he can recognize it.

This process takes place as follows:

- Open the video monitor panel and click the play button (Figure 10.20) to see what NAO sees,
- Click learn button and wait for four seconds for the object to be placed correctly by the NAO (Figure 10.20),
- Take the contour of the object and close the form (Figure 10.21),
- Write the information about the face or object (Figure 10.22),
- If there are more objects to be learned, turn to the second step,

After all objects are learned, send the database to NAO by clicking on Send current vision recognition database to NAO button.

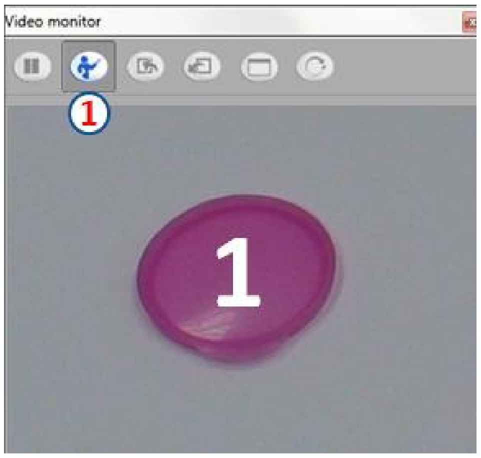


Figure 10.20: Video Monitor Panel- (1) represents learn button.

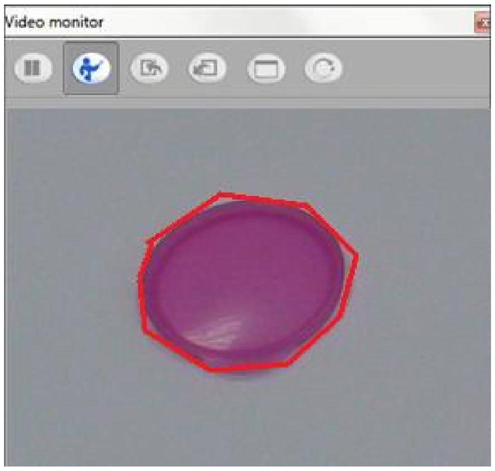
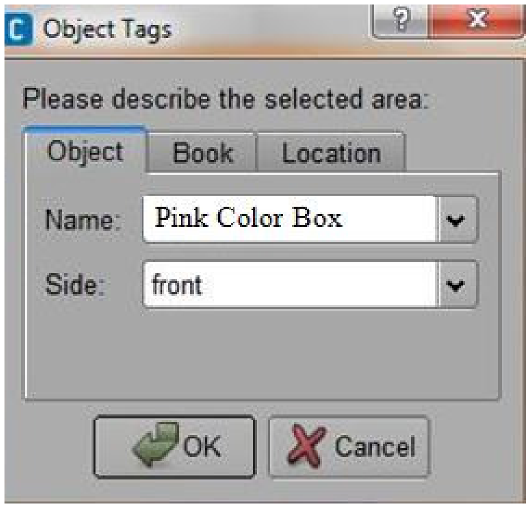


Figure 10.21: Video Monitor Panel- (1) represents learn button.



Object Tags

Please describe the selected area:

Name: Pink Color Box

Side: front

Figure 10.22: Object Information Form.

11. CONCLUSIONS AND RECOMMENDATIONS

In this study, the NAO humanoid robot paints the pictures of the objects it observes, with the assistance of a human. NAO takes the image of the environment, extracts the objects in the environment and finally paints them, asking the human for help when it needs assistance.

In this current work, the first simulation results are represented, mostly focusing on the painting part, but the ultimate goal is focused on the assistance part. The details of physical and kinematic problems are stil worked on the real robot. Currently, the robot paints on a table top, using its arm like a plotter on an x-y plane and moves its upper torso with the arm to reach different locations on the table.

Once the system is moved on the real robot, it is planned to use with disabled children as a part of rehabilitation like an interaction game. Interaction games with robots and children, based on drumming, interaction [30-31] and sign language [32-37] is used previously. Robot and the disabled child will play an interaction game with together. Within the scope of interaction game we plan actions as follows;

- The humanoid robot asks disabled child to find the correct color case,
- The disabled child put the color case in front of the robot,
- If the color isn't true, the robot asks for a right color,
- If the color is true, the robot says "Thank You" to the disabled child.

It is considered to use the interaction between the child and robot to overcome the limitations of the robot, i.e. to get the right color. It is also expected that some emergent actions, i.e. child's helping the robot by coloring some parts which are hold for robot to color.

It is desired that the children to understand the robot's limitations and emergently play an active role to help the robot to fulfill the painting task.

In [38-39], the first outputs of this study with the humanoid robot painter are presented.

12. REFERENCES

[1] **Url-1**<http://www.kurzweilcyberart.com/>, date retrieved 03.06.2012.

[2] **Srikaew A., Cambron M.E., Northrup S., Peters II R. A., Wilkes D. M. and Kawamura K.** (2000). Robotics and Autonomous Systems: Painting robot with multi-fingered hands and stereo vision vol. 15, pages 38-45.

[3] **Kudoh S., Ogawara K., Ruthanurucks M., Ikeuchi K.** (2009). Intelligent Systems and their Applications, IEEE : ISAC: Foundations in Human-Humanoid Interaction, vol. 57, pages 279-288.

[4] **Url-2** <http://www.aldebaran-robotics.com/documentation/nao/hardware/ index.html>, date retrieved 03.06.2012.

[5] **Url-3** <http://www.aldebaran-robotics.com/Downloads/94-Documents/ 102-Feature-papers/View-category.html>, date retrieved 03.06.2012.

[6] **Url-4** <http://webspace.ship.edu/cgboer/gestalt.html>, date retrieved 03.06.2012.

[7] **Gonzalez R. C. and Woods R. E.** (2002). Digital Image Processing.

[8] **Somapa F. and Asano A.** (2009). International Journal of Computer Science and Network Security : Hybrid image thresholding method using edge detection, Vol. 9, No. 4.

[9] **Otsu N.** (1979). IEEE Transactions on Systems, Man and Cybernetics : A threshold selection method from gray-level histograms, Vol. 9, No. 1.

[10] **Henden C.** (2004)**,** A comparison of Thresholding Methods.

[11] **Url-5** <http://en.wikipedia.org/wiki/File:Regiongrowing_histogram1.jpg>, date retrieved 03.06.2012.

[12] **Url-6** <http://en.wikipedia.org/wiki/File:Regiongrowing_figure_Original.jpg>, date retrieved 03.06.2012.

[13] **Url-7** <http://en.wikipedia.org/wiki/File:Lightningg_1.jpg>, date retrieved 03.06.2012.

[14] **Url-8** <http://en.wikipedia.org/wiki/File:Lightningg_2.jpg>, date retrieved 03.06.2012.

[15] **Url-9** <http://en.wikipedia.org/wiki/File:Lightningg_3.jpg>, date retrieved 03.06.2012.

[16] **Url-10** <http://en.wikipedia.org/wiki/File:Lightningg_4.jpg>, date retrieved 03.06.2012.

[17] **Url-11** <http://en.wikipedia.org/wiki/Region_growing>,
date retrieved 03.06.2012.

[18] **Felzenszwalb P. F. and Huttenlocker D.P.** (2004). International Journal of Computer Vision : Efficient graph-based segmentation, Vol. 59, No. 2, pages 167-181.

[19] **Heckbert P.** (1982). Comput. Graph.: Color Image Quantization for Frame Buffer Display, Vol. 16, pages 297–307.

[20] **Gervautz M. and Purgathofer W.,** (1990). Graphics Gems, A. S. Glassner, Ed. New York: A Simple Method for Color Quantization: Octree Quantization, pages 287–293.

[21] **Wan S.J., Prusinkiewicz and Wong S.K.M.** (2007). Color Research and Application: Variance-based color image quantization for frame buffer display, Vol. 15, pages 52-58.

[22] **Dekker A. H.** (1994). Network: Computation in Neural Systems: Kohonen neural networks for optimal color quantization, Vol. 5 No. 3, pages 351-367.

[23] **Yoon K. J. and Kweon I. S.** (2004). IEEE International Conference on Pattern Recognition: Human Perception Based Color Image Quantization,17.

[24] **Marengoni M., Stringhini D**. (2011). SIBGRAPI Conference on Graphics, Patterns, and Images Tutorials: High Level Computer Vision Using OpenCV.

[25] **Canny J**. (1986). Pattern Analysis and Machine Intelligence, IEEE Transactions on PAMI-8(6): A computational approach to edge detection, 679–698.

[26] **A. Hertzmann** (1998). Proceedings of ACM SIGGRAPH : Painterly Rendering with Curved Brush Strokes of Multiple Sizes.

[27] **Ruchanurucks M., Kudoh S, Ogawara K., Shiratori T. and Ikeuchi K.**(2007). International Conference on Robotics and Automation : Human Robot Painter: Visual Perception and High-Level Planning.

[28] **Url-12** <http://www.cyberbotics.com/>, date retrieved 03.06.2012.

[29] **Url-13** <http://users.aldebaran-robotics.com>, date retrieved 03.06.2012.

[30] **Kose-Bagci H., Ferrari E., Dautenhahn K., Syrdal D. S., and Nehaniv C. L.** (2010**).** Connection Science: Effects of emodiment and gestures on social interaction in drumming games with a humanoid robot, Vol. 22, No. 2, pp. 103– 134.

[31] **Dautenhahn K., Nehaniv C. L., Walters M. L., Robins B., Kose-Bagci H., Mirza N. A., Blow M.** (2009). Special Issue on "Humanoid Robots", Applied Bionics and Biomechanics 6(3) : KASPAR - A Minimally Expressive Humanoid Robot for Human-Robot Interaction Research, pp. 369-397.

[32] **Kose H., Yorganci R., and Algan H. E.** (2011). 5th European Conference on Mobile Robots (ECMR11) : Evaluation of the Robot Sign Language Tutor using video-based studies, pp. 109-114.

[33] **Kose, H. and Yorganci R.** (2011). 11th IEEE-RAS International Conference on Humanoid Robots, Bled, Slovenia (HUMANOIDS 2011) : Tale of a robot: Humanoid Robot Assisted Sign Language Tutoring, pp 105 – 111, **ISSN:** 2164-0572.

[34] **Kose, H., Yorganci R., Algan H. E., and Syrdal D.S.** (2012). SORO special issue on "Measuring Human-Robot Interaction: Evaluation of the Robot Assisted Sign Language Tutoring using video-based studies, DOI: 10.1007/s12369-012-0142-2.

[35] **Kose, H., Yorganci R., and Itauma I.I.** (2011). IEEE ROBIO conference, Thailand: Robot Assisted Interactive Sign Language Tutoring Game , pp. 2247-2249.

[36] **Url-14** <http://humanoid.ce.itu.edu.tr>, date retrieved 03.06.2012.

[37] **Kindiroglu, A.A., Yorgancı, R., Kirac, F., Kose, H. and Akarun, L.** (2012). IEEE T-RAM Special issue on Assistive Robotics: Nao Robot Assisted Multimodal Sign Language Tutoring Framework, submitted.

[38] **Gurpinar, C., Alasag T., and Kose H.** (2012). IEEE Advanced Robotics and its Social Impacts (ARSO), Munich: Humanoid Robot Painter Assisted By A Human, pp. 79-82.

[39] **Gurpinar, C., and Kose H.** (2012). 21st IEEE International Symposium on Robot and Human Interactive Communication, Paris, France : Human Assisted Humanoid Robot Painter, submitted.

CURRICULUM VITAE

Name Surname	**: Cemal GÜRPINAR**
Place and Date of Birth	**: İstanbul-1983**
E-Mail	**: cemalpg@hotmail.com**
B.Sc.	**: Turkish Naval Academy**
M.Sc.	**: Istanbul Technical University**

PUBLICATIONS/PRESENTATIONS ON THE THESIS

Gurpinar, C., Alasag T., and Kose H. (2012). IEEE Advanced Robotics and its Social Impacts (ARSO), Munich,Germany: Humanoid Robot Painter Assisted By A Human, pp. 79-82.