M.S. RAMAIAH INSTITUTE OF TECHNOLOGY
(Autonomous Institute Affiliated to VTU)
MSR Nagar, MSRIT Post, Bangalore - 560054

Project entitled

Cognitive Intelligence for Perception in a Humanoid

submitted in partial fulfillment for the award of degree in

Bachelor of Engineering in
Electronics and Communication Engineering

by

Nihal Winston D'Cunha	1MS10EC066
Nischal K N	1MS10EC068
Prashanth G L	1MS10EC083
Praveen Nayak M	1MS10EC084

under the guidance of

M. S. Srinivas
Professor

Department of Electronics and Communication
M. S. Ramaiah Institute of Technology

June 2014

M.S. RAMAIAH INSTITUTE OF TECHNOLOGY

MSR Nagar, MSRIT post, Bangalore - 560054

Department of Electronics and Communication Engineering

CERTIFICATE

This is to certify that the project entitled '**Cognitive Intelligence for Perception in a Humanoid**' is a record of original work by Nihal Winston D'Cunha [USN: 1MS10EC066], Nischal K.N [USN: 1MS10EC068], Prashanth G L [USN: 1MS10EC083] and Praveen Nayak M [USN: 1MS10EC084] in partial fulfilment for the award of degree in Bachelor of Engineering in Electronics and Communication, Visvesvaraya Technological University, Belgaum, during the year 2013-2014. The project report has been approved as it satisfies the academic requirements with respect to project work prescribed for Bachelor of Engineering degree.

Signature of the guide	Signature of the HOD
Prof. M S Srinivas, Professor, Dept of E&C, MSRIT	Dr. S. Sethu Selvi, Head of the Department, Dept of E&C, MSRIT

External examiners

Name	Signature and date
1.	
2.	

M.S. RAMAIAH INSTITUTE OF TECHNOLOGY

MSR Nagar, MSRIT post, Bangalore - 560054

Department of Electronics and Communication Engineering

DECLARATION

We hereby declare that the entire work embodied in this project has been carried out by us at M. S. Ramaiah Institute of Technology under the guidance of **Prof. M. S. Srinivas**. This project has not been submitted in part or full for the award of any diploma or degree of this or any other University.

Team Members:

Nihal Winston D'Cunha	1MS10EC066
Nischal K.N	1MS10EC068
Prashanth G L	1MS10EC083
Prveen Nayak M	1MS10EC084

Date:

"The true delight is in the finding out rather than in the knowing."

Isaac Asimov

Abstract

This project proposes a novel architecture to link high-level cognitive skills to the perceptual and physical abilities of a humanoid robot. The proposed system determines relative visual importance based on inherent object attributes, high-level task constraints, and the attentional states of other people. Objects of interest are tracked in real-time to produce motion paths which are analyzed by a set of native physical laws designed to discriminate between objects, both animate and inanimate. Animate objects can be the source of attentional states, detected by finding faces and recognizing them as well. In addition, we introduce the sense of auditory perception and comprehension. The robot proposed is capable of recognizing basic spoken instructions and intensity of sound and linking the meaning to attentional states it perceives from its sense of sight. Individual components are evaluated by comparisons to human performance on similar tasks, and the complete system is evaluated in the context of a basic social learning AI mechanism that allows the robot to learn from and respond to observed stimuli.

Acknowledgements

We express our sincere thanks to Dr. S.Y Kulkarni, Principal, MSRIT, for providing us with the necessary technical and administrative support. We express our gratitude to Dr. S. Sethu Selvi, HOD, Department of Electronics and Communication, who mentored us and provided us with necessary advice and support for carrying out this project.

We also express our sincere gratitude to Prof. M.S. Srinivas for having whole-heartedly agreed to guide our project. His timely advice and guidance inspired us to explore the avenues in detail and come out with extra-ordinary solutions. His insightful suggestions and constant support have been instrumental in the positive outcome of this project. We once again offer our sincere thanks for guiding our project successfully.

We also express out sincere thanks to Brahma3 for their inputs and support in 3D printing and Li2 Technologies Ltd., Bangalore.

We profoundly thank the MSRIT Alumni Association for having given us monetary assistance. We thank Mr. C G Raghavendra, Project Co-ordinator, for the support and guidance. We also thank the laboratory staff for providing us with the necessary equipments and support.

Contents

List of Figures

List of Tables

Dedicated to the world of RESEARCHERS, TEACHERS, OPEN SOURCE COMMUNITY and OUR PARENTS.

Chapter 1

Introduction

Project C.H.L-O.E (CHLOE- Cognitive Humanoid Learning- Online Embodiment) takes inspiration from the human brain. A baby during the initial days of its life knows very little. Some things are inbuilt, like the ability to distinguish a human face from its surroundings. Another hardwired attribute of a baby is 'how to learn'. The baby sees things around it and asks people for information about the object it has seen. Later the baby learns the things gathered and grows up. CHLOE's working coincides with the working of the human brain.

Cognitive learning is defined as the acquisition of knowledge and skill by mental or cognitive processes the procedures we have for manipulating information 'in our heads'. In cognitive learning [1], the system learns by listening, watching, touching, reading, or experiencing and then processing and remembering the information. Cognitive learning differs a lot from conventional learning. Many researches done in the past have proved that cognitive learning is faster and it is a better form of learning compared to conventional learning. In cognitive learning, the learning takes place in the mind and not according to the conventional behavior exhibited. Cognition involves the formation of mental representations of the elements of a task and the discovery of how these elements are related. Cognitive learning stresses on the idea that learning comes about as a result of processes related to experience, perception, memory, as well as overtly verbal thinking and interaction with the user. Cognitive learning is a powerful mechanism that provides the means of knowledge, and goes well beyond simple imitation of others. Information processing is based on a learning that describes the processing of storage and retrieval of knowledge from the system's memory.

Cognitive learning might seem to be passive learning, but that is what is running in the background. The proposed system is a mechanism for motor response, i.e., the system gives back to the environment, and is a social being. The learner is quite active, in

a cognitive way, in processing and remembering newly incoming information. Because cognitive activity is involved in many aspects of human behavior, it might seem that cognitive learning only takes place in human beings. However, we have developed a system to show that cognitive learning is not only limited to human behavior but it can be extended to machines and other mechanical devices, leading to reduced human intervention in tough environments.

As a holistic approach, our attempt is to impart the cognitive abilities of a 2 year old child to a machine. Hence, it tends to a social being, and gives back to the environment as much as it takes in. It is capable of understanding simple visual and auditory stimuli and knows how to react to these.

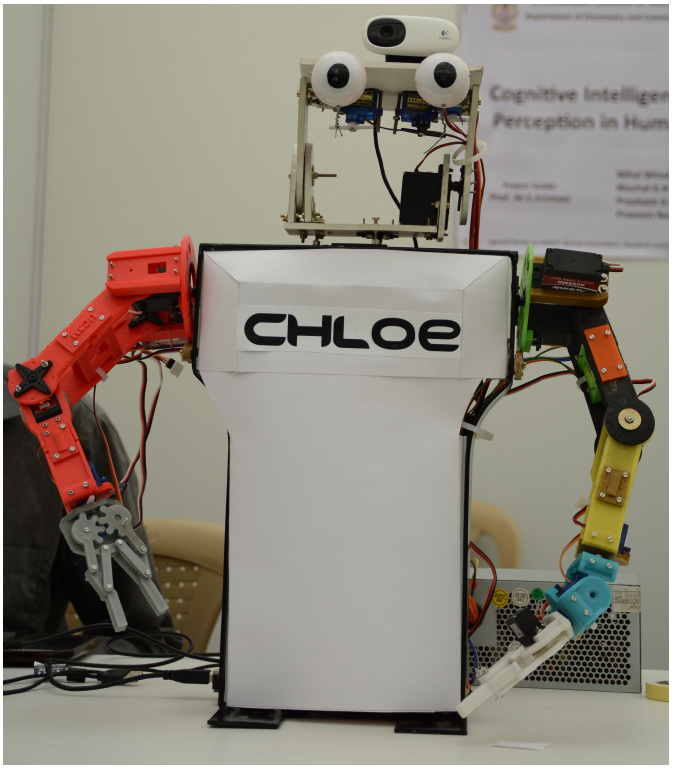


FIGURE 1.1: Introducing Cognitive Humanoid Learning-Online Embodiment (C.H.L-O.E) v1.0.

Chapter 2

Problem statement and Contributions

Cognitive Robotics is a field of extensive research and is the future towards which Cognitive science, robotics and control, Signal processing and Psycho-Neuro Analysis are going to merge. Though extensive research is on-going in this field, a relatively low number of products employs such algorithms due to the high cost involved. Also, individuals and start-ups cannot work on cognitive robotics due to the high funding required which can be afforded by few entities such as the government and defence. We have tried to bridge this gap by building a low cost cognitive platform deployed on 3D printed humanoid body. The key contributions of CHLOE to the research community are:

1. A perceptive humanoid robot, with social skills, that knows or learns how to respond in different circumstances.
2. A Versatile open-Source software architecture, easily upgradable modules and compatibility with new modules.
3. A Parallel Distribution Processing (PDP) based Cognitive framework, to embody the cognitive function of a 2-year old child in a machine
4. A low cost Cognitive platform to aid low budget research and development in developing countries like India, making use of trending technologies like 3D printing.

Chapter 3

Literature Survey

The present day world is moving towards cognitive learning systems. iCub and Nao, are the leading ones in this regard. Project C.H.L-O.E derives motivation from these two cognitive learning based robots Variety of architectures are developed for cognitive systems. Wide variety of functions are handled by these systems which includes assisting the elderly or a person confined to a bed or a wheelchair, perform certain tasks that are dangerous to humans, such as fighting fires, etc. There are three important systems in this aspect which are described in the following subsections.

3.1 ASIMO

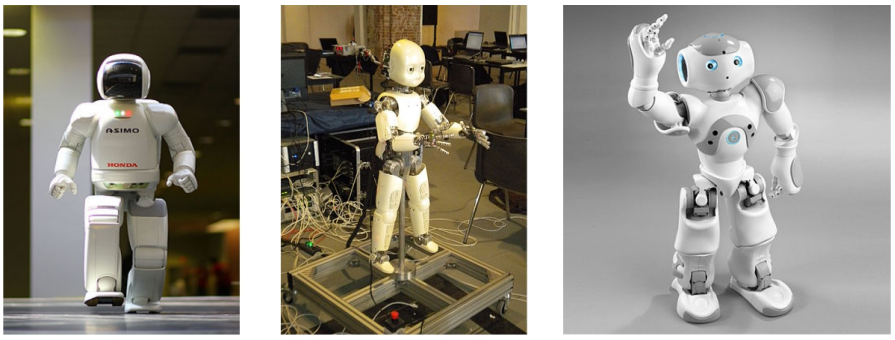
ASIMO, an acronym for Advanced Step in Innovative MObility, is a humanoid robot (Fig. 3.1) designed and developed by Honda. With aspirations of helping those who lack full mobility, ASIMO is frequently used in demonstrations across the world to encourage the study of science and mathematics.

In addition to ASIMO's ability to walk like we do, it can also do the below tasks :

- Understand pre-programmed gestures and spoken commands
- Recognize voices
- Recognize faces
- Interface with IC Communication cards.
- ASIMO has arms and hands so it can do things like turn on light switches, open doors, carry objects, and push carts.

3.2 iCub

iCub is a 1 metre high humanoid robot testbed for research into human cognition and artificial intelligence. It was designed by the RobotCub Consortium of several European universities and built by Italian Institute of Technology. The robot is open-source, with the hardware design, software and documentation all released under the GPL license.



(a) (b) (c)

FIGURE 3.1: Various Cognitive Humanoid Systems (a) ASIMO, (b) iCub Robot, (c) Nao Robot.

It has 53 motors that move the head, arms & hands, waist, and legs. It can see and hear, it has the sense of proprioception (body configuration) and movement (using accelerometers and gyroscopes).

3.3 Nao

Nao (pronounced now) is an autonomous, programmable humanoid robot developed by Aldebaran Robotics, a French robotics company headquartered in Paris. NAO has:

- 25 degrees of freedom, for movement
- Two cameras, to see its surroundings
- An inertial measurement unit, which lets him know whether he is upright or sitting down
- Touch sensors to detect your touch
- Four directional microphones, so he can hear you

This combination of technologies (and many others) gives NAO the ability to detect its surroundings.

Chapter 4

Proposed System

CHLOE (Cognitive Humanoid Learning-Online Embodiment) is an automated humanoid which has the ability to interact with its environment and learn accordingly. Possessing a clean slate for a memory, it remembers objects and faces that it encounters, and can then be trained to perform specific actions when it comes across those objects or faces again.

Also proposed is the implementation of the well concept of attentional states for perception. Human attention is never dormant. Always eager to learn, humans absorb information from the surroundings and give back to the environment. The perceptive response is manifested using motor skills like moving arms and giving "attention" to the cause of the attentional states.

A Parallel Distributed Processing (PDP) based Cognitive Framework versatile to modular upgrades and modifications is also proposed.

Also, experimentation is performed with the concept of networked Machine Learning systems. With improvement in internet technology, it is possible for a system to perform complex computations on a remote server. Google Speech to Text API access is presented as a proof of this concept.

Chapter 5

Stimulus Control and Cognition

To behave adaptively is to behave differently in one situation than in another. As we move up from paramecia to pigeons and people, the number of different modes of possible behavior increases enormously, and with it the number of different situations for which a unique behavior is appropriate. How do animals organize this knowledge? And how are situations recognized?

The two questions are not really separate, since some kinds of organization make recognition quick and accurate, while others make it slow and unreliable. The ability to recognize when particular adaptive behaviors are appropriate has been taken for granted. In this section we look at one aspect of recognition and the organization of individual knowledge.

What does it mean to *recognize* something? In a formal sense the answer is simple: it means to be in a unique state, so that in the presence of object w, the being is always in state W, and state W never occurs at any other time. This is also a necessary condition for the being to discriminate w from things that are not w. But the formal answer conceals a great deal. For example, it isn't much help in constructing machines that can recognize, for which much more specific information is required: We need to know how to process particular visual (or auditory, touch, or whatever) inputs; how to direct the visual apparatus on the basis of past information where should the machine look next? How to distinguish objects from their backgrounds; how to identify the same object from different points of view; and how to encode all this information so that it provides a useful basis for action.

Most of these questions are about *perception*, and that is what we have implemented. In order to get on with the study of learning and motivation we must take for granted the

processes that translate a particular physical environment into some internal representation that allows the being to recognize the environment on subsequent occasions (or, more cautiously phrased, the processes that allow the animal to behave in the same way or ways that are the same in essential aspects every time he is in the same environment). Perceptual processes are not trivial. This section is about the last step in the process: the encoding of information in ways useful for action. We are concerned not with how the being "sees" a phone or a ball or a colored light, but with how these things resemble or differ from other things in his world.

It is not at all clear that the best way to answer this question is to begin with the concept of *stimulus*, where a stimulus is some very specific, physically defined event. Nevertheless, because the study of learning in animals grew up under the influence of reflex-oriented behaviorists, the relevant terms, experimental methods, and concepts have all evolved from "stimulus-response" psychology. Hence, "stimulus" is the natural place to start. The terms behaviorist ancestry at least means that we will know what we are talking about, even if we are uncertain about how useful it will prove to be.

5.1 Object Detection

Keeping in mind, a holistic view of the Cognitive learning system in itself, we have designed composite Color-Contour based object detection algorithm. Since we require the humanoid to behave with the social and cognitive intelligence of a 2 year old child, we did some research into what a child can actually perceive with higher acuity. And the following can be inferred:

1. Compared to polygons, a child's perception of a spherical and oval objects is much higher. Fewer the edges seen, the more interesting it is to a human. This stems from the fact that human faces tend to an ellipsoid. Hence, detection and interest toward spherical objects must be greater.
2. A child shows heightened perception towards prominently brighter colors. Strong, bright colours have the effect of shocking the childs inner vibrations, which can make it unsettled and restless. Bright, intense colours such as primary red, yellow and orange are perceived as attractive and are first detected, wherever in the field of vision.

Also, amorphous abject detection is not a cognitive quality, but is developed in later stages, with development in memory. Hence, we propose a simplistic scheme than the one in fig 5.1 to mimic the same.

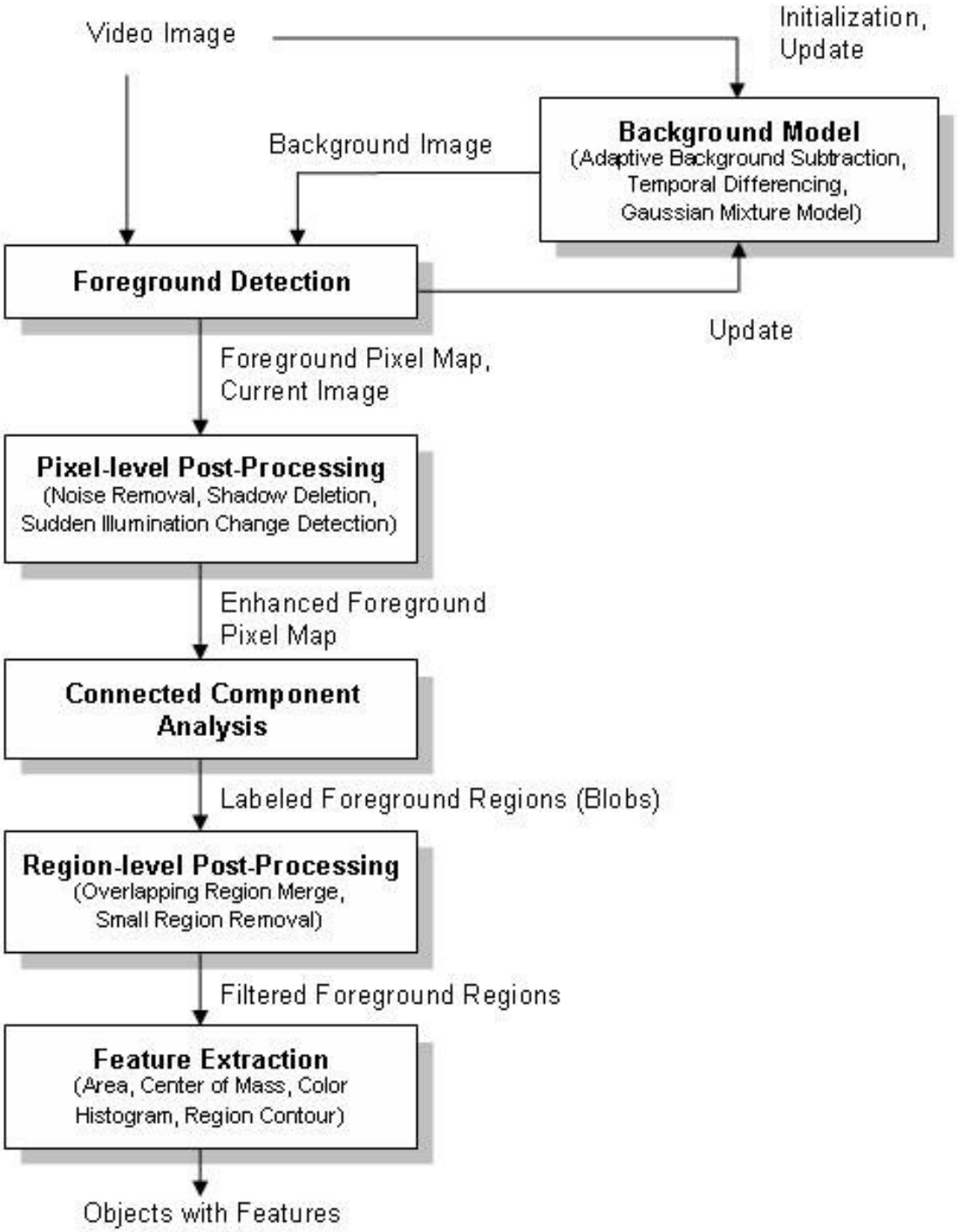


FIGURE 5.1: General Object detection system.

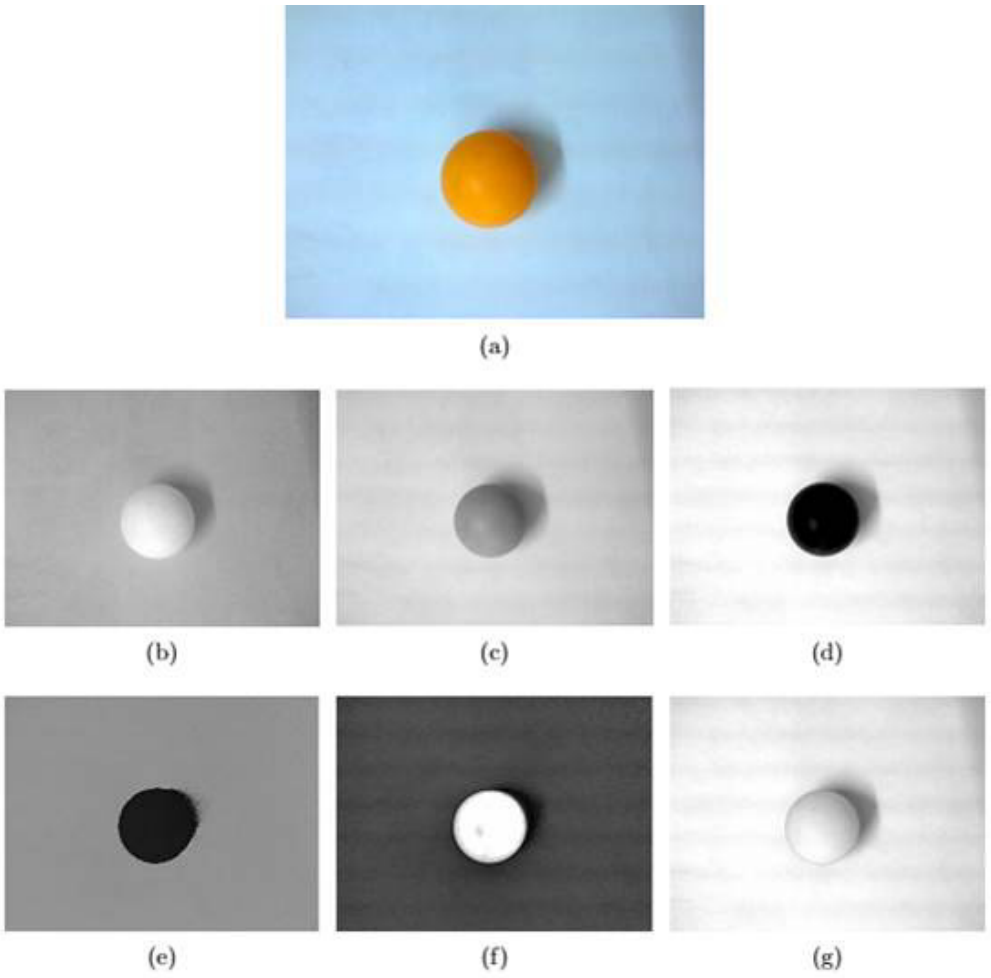


FIGURE 5.2: Colorspaces (a) Sample image of a ball and its (b) Red, (c) Green, (d) Blue (RGB), (e) Hue, (f) Saturation, (g) Value (HSV) components.

5.1.1 Color based detection Using HSV Filtering

The choice of the colorspace used is extremely critical since we need a characteristic property to distinguish between the object and the scene around it. Fig 5.2 shows a sample frame containing a yellow ball, and the various components of its RGB (Red, Green, Blue) and HSV(Hue, Saturation, Value) colorspaces. The primary feature of a good visual system is color-independence, i.e. the component chosen should behave in almost the same way irrespective of the color of the object. Thus, the red, green and blue components shown in Figs. 5.2b, 5.2c and 5.2d are not useful owing to their rich color content. The hue component shown in Fig. 5.2e is also ruled out since it represents the amount of color in HSV colorspace. The saturation component shown in Fig. 5.2f is promising since the white background is unsaturated (very low color content) and hence, a lot of distinction can be seen between the highly saturated object and its background. We are making use of the property that saturation is a measure of how much color

is contained in the image and hence, perfect grayscale images (no color content) are unsaturated.

Another reason to choose the saturation component is its inherent property of illumination-invariance. Under slight illumination variations, the white background appears to be a little gray; however, it remains unsaturated and thus, the saturation component is not affected much. This provides the system with robustness to illumination, to a great extent.

Steps for detection the largest bright object are as follows:

1. Capture a frame from the device
2. Flip the frame horizontally to reflect a virtual image
3. Blur the frame using GaussianBlur
4. Convert the frame from BGR to HSV colour space (Fig 5.3(a))
5. Get a scaled down version to reduce the processing time
6. Define the lower and upper bounds of the red colour (use other ranges for other colours)
7. Get the components within the colour range
8. Get the contours of these components (Fig 5.3 (b))
9. Determine the largest contour
10. Determine if the mass of the contour is large enough to track
11. Draw a minimum-area bounding rectangle on the original frame and pass centre of mass to main AI program

5.1.2 Shape based detection

Now, just color detection is not sufficient. the intersection of the color and shape maps gives the desired interest object. Hence, we use Hough transform to detect the shape of a circle. Smoothing of the image by linear convolution with a `size1`×`size2` Gaussian kernel and Median filter is found to improve results.

The purpose of the Hough transform [2] technique is to find imperfect instances of objects within a certain class of shapes by a voting procedure. This voting procedure is carried out in a parameter space, from which object candidates are obtained as local maxima in a so-called accumulator space that is explicitly constructed by the algorithm

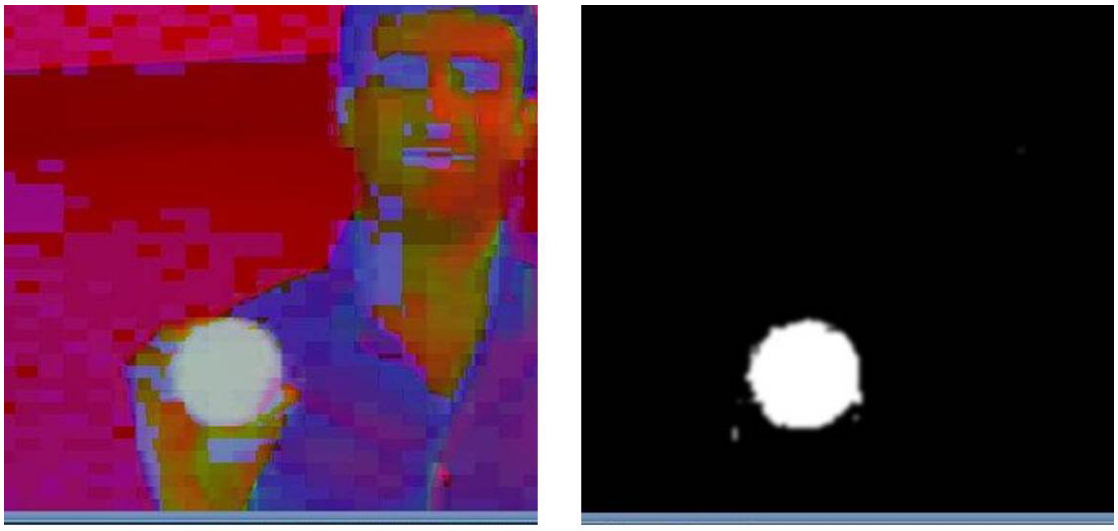


FIGURE 5.3: Considering a Red ball, (a) HSV colospace, (b) After filtering the red-yellow color range (bright range).

for computing the Hough transform.

The process of identifying possible circular objects in Hough space is relatively simple:

- First we create our accumulator space which is made up of a cell for each pixel, initially each of these will be set to 0.
- For each(edge point in image(i, j)): Increment all cells which according to the equation of a circle($(i-a)^2 + (j-b)^2 = r^2$) could be the center of a circle, these cells are represented by the letter 'a' in the equation.
- For all possible value of a found in the previous step, find all possible values of b which satisfy the equation.
- Search for the local maxima cells, these are any cells whose value is greater than every other cell in its neighbourhood. These cells are the one with the highest probability of being the location of the circle(s) we are trying to locate.

The result is pretty impressive. The code functions very well and detects the circles under most circumstances (i.e. the ball is far from the camera, close to the camera, slow movement, fast movement, etc). Also, entire processing is completed between two consecutive frames in an Intel Core i5-3210 system operating on 40fps. So the computation is also suitably fast.

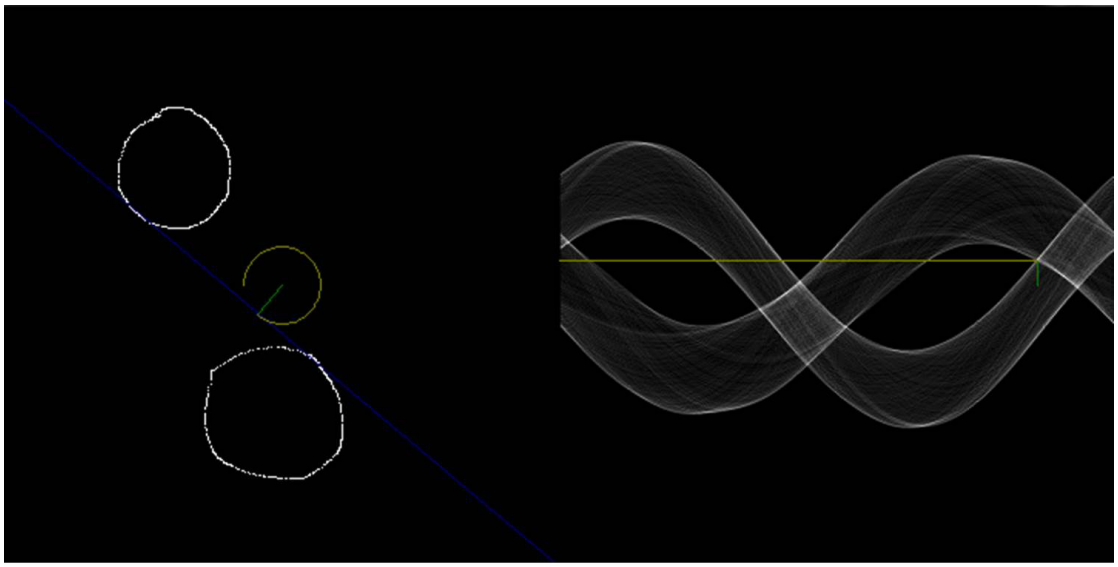


FIGURE 5.4: Hough transform on a circular object (a)test image generated using MS-paint having circles and lines, (b) generated projections to be constrained by circle equation to detect a circle.

FIGURE 5.5: Shape detection (a) The detected color contour to be subjected the Hough transform, (b) Rendering of transform results. The local minima is searched to obtain a, b and r.

5.1.3 Intelligence in Detection

As stated earlier, the brighter circular tending objects have to be detected. A simple if-else intelligence module does the job. The worst case performance may still be bad, as all exceptions may not be handled. But using 4 possible outcomes of shapes and colors (3 levels of brightness nested), the required module is completed. The results are shown in Fig. 5.7.

FIGURE 5.6: Shape based selection (a) showing detected polygon, (b) Circular object given priority over polygon.

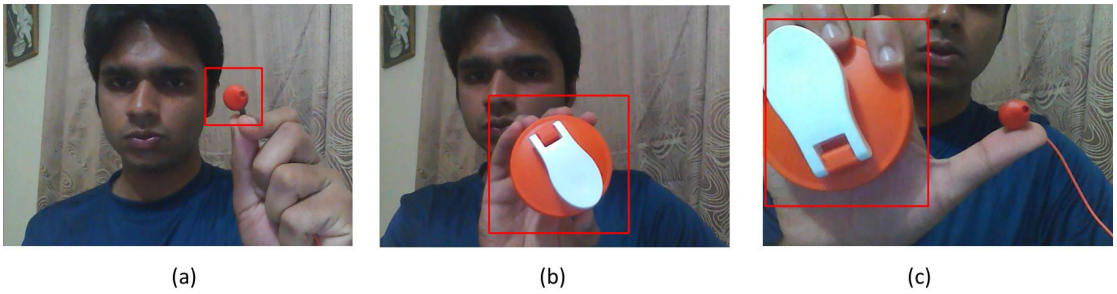


FIGURE 5.7: Shape and color based selection. We take two circular plane objects (a) and (b), the brighter orange is given more priority and is selected.

5.2 Object Recognition

We propose a modified version of the extremely popular SIFT algorithm by combining it with FLANN for keypoint matching.

5.2.1 SIFT

Scale-invariant feature transform (or SIFT) is an algorithm in computer vision to detect and describe local features in images. The algorithm was published by David Lowe

in 1999. SIFT is a method to detect distinct, invariant image feature points, which easily can be matched between images to perform tasks such as object detection and recognition, or to compute geometrical transformation between images [3].

For any object in an image, interesting points on the object can be extracted to provide a "*feature description*" of the object. This description, extracted from a training image, can then be used to identify the object when attempting to locate the object in a test image containing many other objects. To perform reliable recognition, it is important that the features extracted from the training image be detectable even under changes in image scale, noise and illumination. Such points usually lie on high-contrast regions of the image, such as object edges.

Another important characteristic of these features is that the relative positions between them in the original scene shouldn't change from one image to another. For example, if only the four corners of a door were used as features, they would work regardless of the door's position; but if points in the frame were also used, the recognition would fail if the door is opened or closed. Similarly, features located in articulated or flexible objects would typically not work if any change in their internal geometry happens between two images in the set being processed. However, in practice SIFT detects and uses a much larger number of features from the images, which reduces the contribution of the errors caused by these local variations in the average error of all feature matching errors.

SIFT can robustly identify objects even among clutter and under partial occlusion, because the SIFT feature descriptor is *invariant to uniform scaling, orientation, and partially invariant to affine distortion and illumination changes.*

SIFT is quite an involved algorithm. It has a lot going on and can become confusing. So we split up the entire algorithm into multiple parts. Heres an outline of what happens in SIFT.

1. **Constructing a scale space**: This is the initial preparation. You create internal representations of the original image to ensure scale invariance. This is done by generating a *scale space*.
2. **LoG Approximation**: The Laplacian of Gaussian is great for finding interesting points (or key points) in an image. But its computationally expensive. So we cheat and approximate it using the Difference of Gaussian. (Fig. 5.8)
3. **Finding keypoints**: With the super fast approximation, we now try to find key points. These are maxima and minima in the Difference of Gaussian image we calculate in step 2. (Fig 5.10 (a), Fig 5.9 shows keypoint detection results for a database of 5 images)

4. **Get rid of bad key points**: Edges and low contrast regions are bad keypoints. Eliminating these makes the algorithm efficient and robust. A technique similar to the Harris Corner Detector is used here. (In Fig 5.10(a), the red marked features are bad keypoints)
5. **Assigning an orientation to the keypoints**: An orientation is calculated for each key point. Any further calculations are done relative to this orientation. This effectively cancels out the effect of orientation, making it rotation invariant. (Fig 5.10(b), (c))
6. **Generate SIFT features**: Finally, with scale and rotation invariance in place, one more representation is generated. This helps uniquely identify features. Lets say you have 50,000 features. With this representation, you can easily identify the feature youre looking for (say, a particular eye, or a sign board).

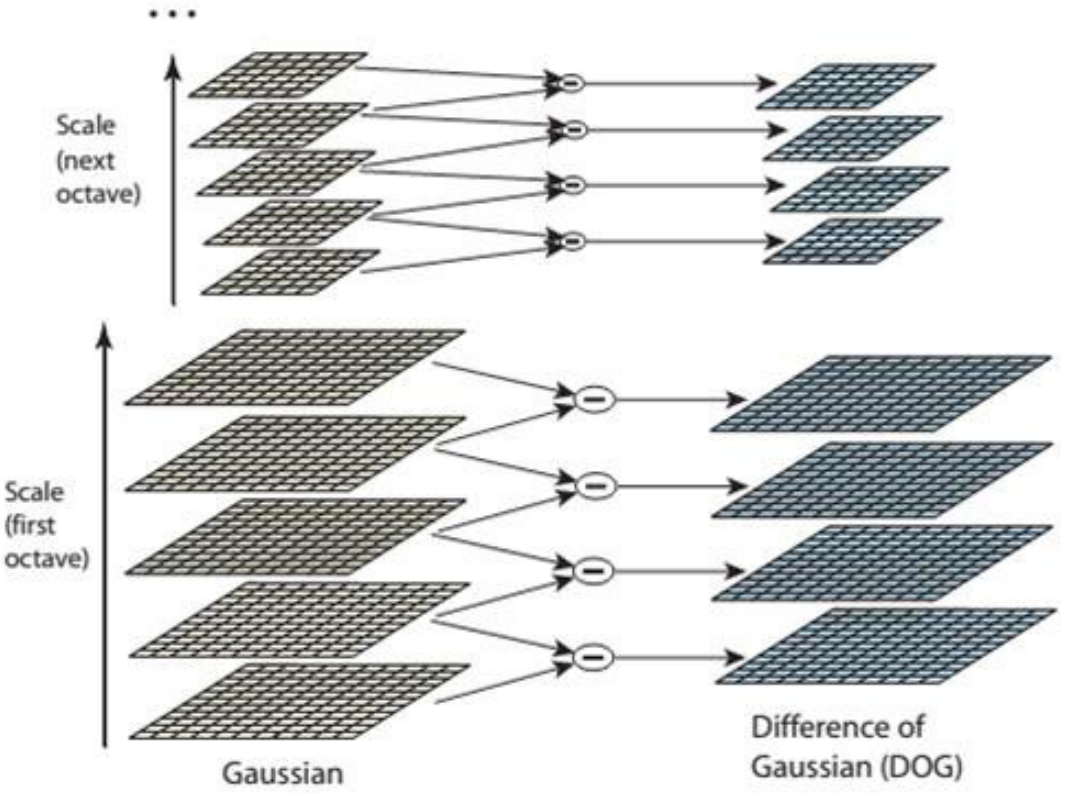


FIGURE 5.8: For each octave of scale space, the initial image is repeatedly convolved with Gaussians to produce the set of scale space images shown on the left. Adjacent Gaussian images are subtracted to produce the difference-of-Gaussian images on the right. After each octave, the Gaussian image is down-sampled by a factor of 2, and the process repeated.

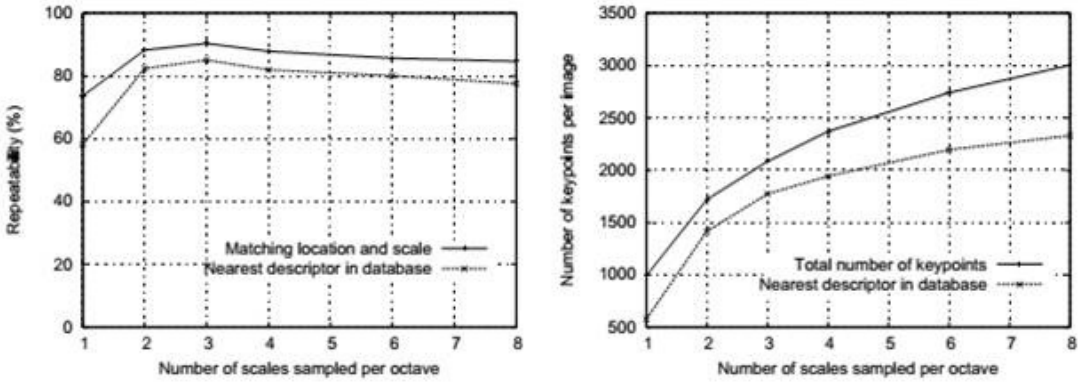


FIGURE 5.9: The top line of the first graph shows the percent of keypoints that are repeatably detected at the same location and scale in a transformed image as a function of the number of scales sampled per octave. The lower line shows the percent of keypoints that have their descriptors correctly matched to a large database. The second graph shows the total number of keypoints detected in a typical image as a function of the number of scale samples.

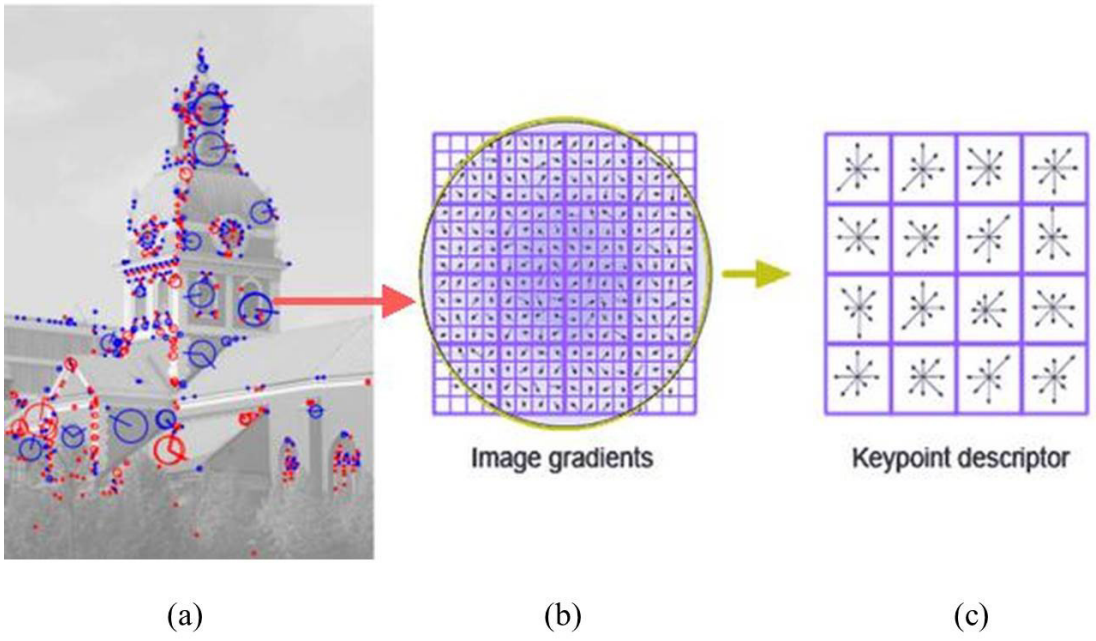


FIGURE 5.10: SIFT feature generation (a) Image showing bad keypoints in red and good ones in blue, (b) Image gradients for assigning orientation, (c) Keypoint descriptor is actually how the feature is stored in memory, as a vector with magnitude and orientation.

5.2.2 FLANN

After successfully finding SIFT keypoints in an image, we can use these keypoints to match the keypoints from other image. FLANN [4] is a library for performing fast approximate nearest neighbor searches in high dimensional spaces. Created by Marius Muja in 2009, It contains a collection of algorithms we found to work best for nearest neighbor search and a system for automatically choosing the best algorithm and optimum parameters depending on the dataset. The type of index trees are KD, randomized KD and Hierarchical K-Means.

We find that FLANN uses either the Randomized KD Tree or Hierarchical K-means Tree for generic feature matching.

5.2.2.1 Randomized KD Tree

Improve the approximation of nearest neighbors from the query point by searching simultaneously across a number of randomized trees. The tree are built from the same set of samples. The author argues that in practice the variances among dimensions do not vary much. Randomly picking from the highest few will be enough to make the different trees. (RKD) It could be further improved by performing PCA. Re-align the data to the highest principle components. (PKD)

5.2.2.2 Hierarchical K-Means Tree

It is a tree of which every inner node is split in K-ways. K-means clustering is used to classify the data subset at each node. A L level tree would have approximately K^L leaf nodes. They are all at level L.

FLANN is written in C++ and contains bindings for MATLAB and Python. In our experiments we have found FLANN to be about one order of magnitude faster on many datasets (in query time), than previously available approximate nearest neighbor search software. It is still a work in progress and is open-source and state-of-the-art.

5.2.3 Prediction of the new Bounding Box, point of interest

Once the relevant keypoints have been found, using the minimum bounding box function in opencv, a Region of interest is found which contains all objects in the image.

So now, CHLOE knows the region where there are maximum number of matched contours. But, what we need is a point of interest so that a response can be shown. This

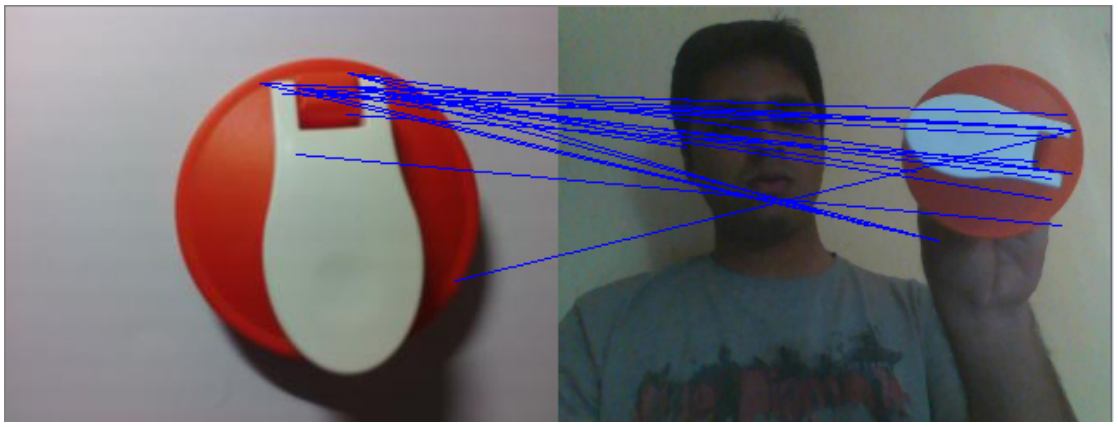


FIGURE 5.11: Results of SIFT+FLANN keypoint matching showing rotation and scale invariance.

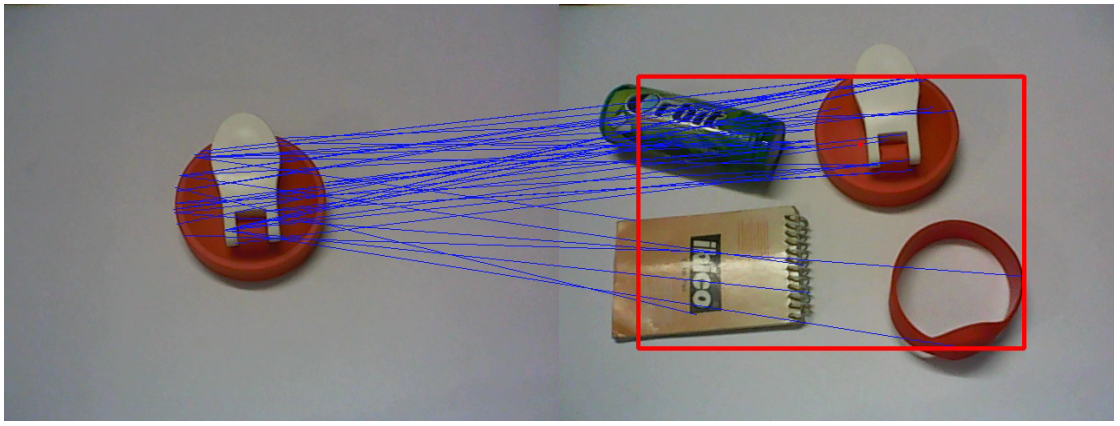


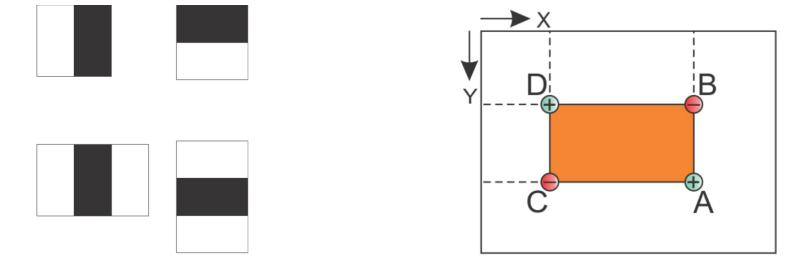
FIGURE 5.12: On-system test results for an object labeled "Orbit" among other objects. The Box shows the region of Interest, where all objects lie. The Red point, shows the centroid of all points, the point of interest.

point is approximated to be the centroid of the matched keypoints and passed to the master control.

Hence the **overall object detection procedure**, simply put, is as follows

1. Select ROI in the image. Slice the image and extract the ROI.
2. Find SIFT Keypoints in ROI image.
3. Find SIFT Keypoints in the image.
4. Match Keypoints using FLANN.
5. Predict new bounding box/ ROI and repeat.

The drawback to using SIFT? Well, its a **patented Algorithm**. If applied to commercial means and ends, one has to pay royalty to Prof. David Lowe and University of British Columbia. A small price to pay for a brilliant and accurate technique, we would say.



a) Illustration of Haar features used in the algorithm

b) Illustration of Integral Image concept

FIGURE 5.13: Haar and Integral Image concepts.

5.3 Face Detection and Recognition

Face detection is concerned with finding whether or not there are any faces in a given image (usually in gray scale) and, if present, return the image location and content of each face. This is the first step of any fully automatic system that analyzes the information contained in faces (e.g., identity, gender, expression, age, race and pose).

After the face is detected, next important challenge is to get the identity. Facial recognition (or face recognition) is a type of biometric software application that can identify a specific individual in a video frame or digital images by analyzing and comparing patterns. Face recognition has emerged as an active research area due to its large number of useful applications. These applications mainly include detective agencies, military purpose, surveillance, information security, smart cards and entertainment.

5.3.1 Face Detection using Viola-Jones Algorithm

The face detection problem is challenging as it needs to account for all possible appearance variation caused by change in illumination, facial features, occlusions, etc. In addition, it has to detect faces that appear at different scale, pose, with in plane rotations. In spite of all these difficulties, tremendous progress has been made in the last decade and many systems have shown impressive real-time performance. The recent advances of these algorithms have also made significant contributions in detecting other objects such as humans/pedestrians, and cars. One such advancement is the Viola Jones algorithm which is being implemented in our project.

Viola Jones method is applied to detect the face efficiently. The basic idea is to slide a window across image and evaluate a face model at every location. Here, training is slow, but detection is very fast. It consists of three phases which are dealt in the following sections.

5.3.1.1 Integral Image

The *haar* features as shown in Fig. 5.13a are used to evaluate the facial features. When these features are applied to the sample image shown in Fig 5.14, the pixels which comes under black region and white region are summed independently, then white region pixels are subtracted from the black region pixels. This is time consuming, since 160,000+ features exists and should evaluate all over the image. Hence we go for the concept of Integral Image.

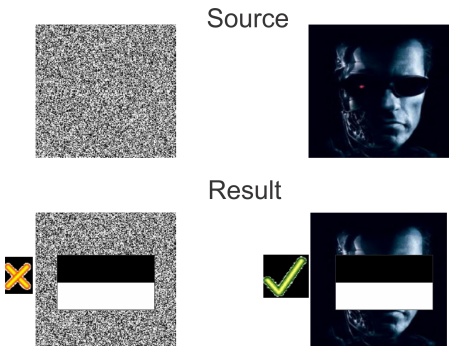


FIGURE 5.14: Illustration of Haar Feature Evaluation.

Integral image, also known as a summed area table, is an algorithm for quickly and efficiently computing the sum of values in a rectangle subset of a grid. Let A,B,C,D be the values of the integral image at the corners of a rectangle as shown in the Fig. 5.13b. Then the sum of original image values within the rectangle can be computed as:

$$Sum = A - B - C + D \tag{5.1}$$

Only 3 additions are required for any size of rectangle. This is the power of Integral Image concept.

5.3.1.2 Adaboost

The AdaBoost learning algorithm is used to boost the classification performance of a simple learning algorithm. It does this by combining a collection of weak classification functions to form a stronger classifier. The pseudocode of this algorithm is shown below.

Input

- Training examples $S = \{(x_i, z_i), i = 1, ..., N\}$
- T is the total number of weak classifiers to be trained

Initialize

- Initialize example score $F^0(x_i) = \frac{1}{2}\ln(\frac{N_+}{N_-})$ where N_+ and $N\_$ are the number of positive and negative examples in the training data set.

Adaboost Learning

For t = 1,...., T :

1. For each Haar-like feature h(x) in the pool, find the optimal threshold H and confidence score c_1 and c_2 to minimize the Z score L^t.
2. Select the best feature with the minimum L^t.
3. Update $\mathrm{F}^t(x_i) = F^{t-1}(x_i) + f_t(x_i), i = 1, ..., N$
4. Update $W_{+1j}, W_{-1j}, j = 1, 2$.

Output Final classifier $F^T(x)$.

5.3.1.3 The Attentional Cascade Structure

Attentional cascade is a critical component in this detector. The key insight is that smaller, and thus more efficient, boosted classifiers can be built which reject most of the negative sub-windows while keeping almost all the positive examples (Fig. 5.15). Consequently, majority of the subwindows will be rejected in early stages of the detector, making the detection process extremely efficient.

5.3.1.4 Jitter reduction in face detection

We find there is a very visible jitter in the face detection, when actually implemented. Since this may cause trouble in pointing and looking at space, we use a simple ***Moving Average filter*** to reduce noise and average out the detected points. A moving average (rolling average or running average) is a calculation to analyze data points by creating a series of averages of different subsets of the full data set. Given a series of coordinates and a fixed subset size, the first element of the moving average is obtained by taking the average of the initial fixed subset of the number series. Then the subset is modified by "shifting forward", that is, excluding the first number of the series and including the next number following the original subset in the series. This creates a new subset of numbers, which is averaged. This process is repeated over the entire data series. The

plot line connecting all the (fixed) averages is the moving average. A moving average is a set of numbers, each of which is the average of the corresponding subset of a larger set of datum points. Viewed simplistically it can be regarded as smoothing the data. This is implemented using a simple dynamic circular queue of fixed size. (Currently, size=16). For higher orders, greater stability is obtained at the cost of higher settling time. Some of the sample images which are detected from this method is shown in Fig. 5.16.

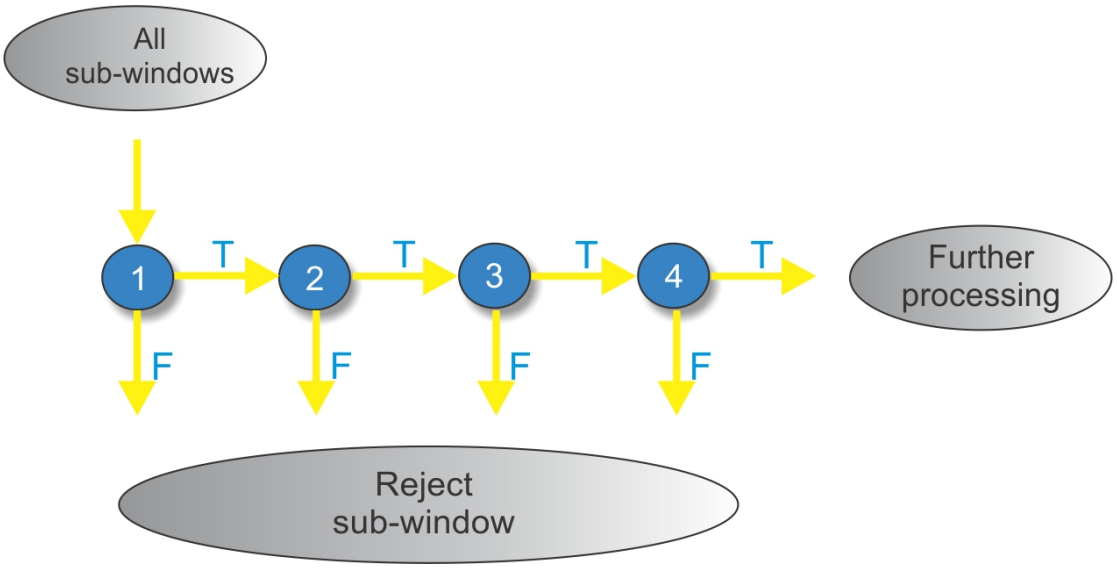


FIGURE 5.15: Figure showing the cascade process.

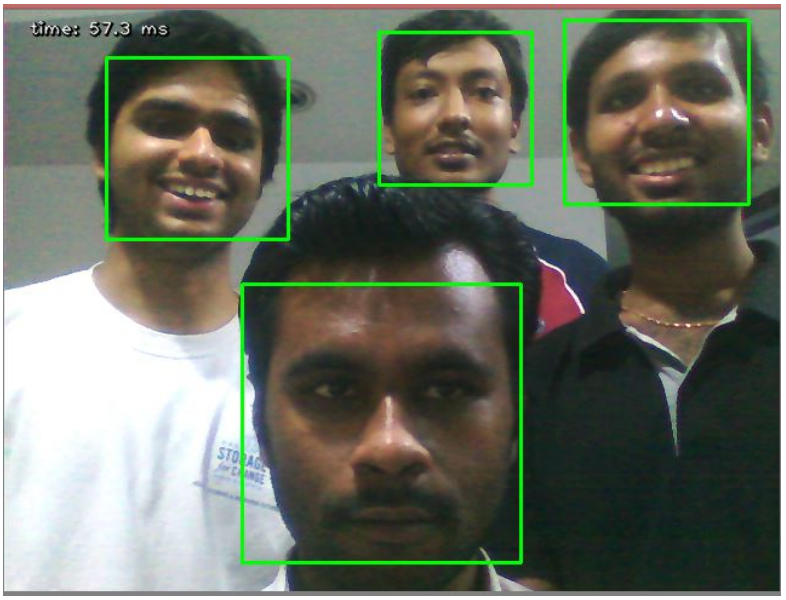


FIGURE 5.16: Face detector output on Real Time Test image.

5.3.2 Face Recognition

Face Images are similar in overall configuration, which add to the complexity involved in recognizing the face. These variations are mainly classified in to following categories.

- Illumination variations
- Pose variations
- Expression variations
- Time change
- Occlusion

5.3.2.1 Eigenfaces for Recognition

We have focused our method of recognition toward developing a sort of unsupervised pattern recognition scheme that does not depend on excessive geometry and computations like deformable templates. Eigenfaces approach seemed to be an adequate method to be used in face recognition due to its simplicity, speed and learning capability.

A previous work based on the eigenfaces approach was done by M. Turk and A. Pentland [6], in which, faces were first detected and then identified. In this project, a face recognition system based on the eigenfaces approach, kindred to the one presented by M. Turk and A. Pentland, is proposed.

The scheme is based on an information theory approach that decomposes face images into a small set of characteristic feature images called eigenfaces, which may be thought of as the principal components of the initial training set of face images.

Recognition is performed by projecting a new image onto the subspace spanned by the eigenfaces and then classifying the face by comparing its position in the face space with the positions of known individuals. Actual system is capable of both recognizing known individuals and learning to recognize new face images. The eigenface approach used in this scheme has advantages over other face recognition methods in its speed, simplicity, learning capability and robustness to small changes in the face image.

5.3.2.2 Algorithm

Algorithm can be explained with the help of sample images (Fig. 5.17) and effects on them as the algorithm proceeds.

FIGURE 5.17: Set of images used to create our eigen space for face recognition.

1. The first step is to obtain a set S with M face images. In our example M = 25 as can be seen in Fig. 5.17. Each image is transformed into a vector of size N and placed into the set.

$$S = \{\tau_1, \tau_2, \tau_3, ..., \tau_M\} \tag{5.2}$$

2. After obtaining the set, we need to calculate the mean image Ψ.

$$\psi = \frac{1}{M}\sum_{n=1}^{M}\tau_n \tag{5.3}$$

FIGURE 5.18: Mean Image.

3. Then the difference (Φ) between the input image and the mean image is found out.

$$\Phi_i = \Gamma_i - \Psi \tag{5.4}$$

4. Next we seek a set of M orthonormal vectors, u_n , which best describes the distribution of the data. The k^{th} vector, u_k, is chosen such that

$$\lambda_k = \frac{1}{M}\sum_{n=1}^{M}\left({u^T}_k\Phi_n\right)^2 \tag{5.5}$$

is a maximum, subject to

$$u^T{}_l u_k = \delta_{lk} = \begin{cases} 1 & if l = k \\ 0 & otherwise \end{cases} \tag{5.6}$$

where u_k and λ_k are the eigenvectors and eigenvalues of the covariance matrix C

5. We obtain the covariance matrix C in the following manner

$$\begin{aligned} C &= \frac{1}{M}\sum_{n=1}^{M} \Phi_n {\Phi_n}^T && (5.7) \\ &= AA^T && (5.8) \\ A &= \{\Phi_1, \Phi_2, \Phi_3, ..., \Phi_n\} && (5.9) \\ L_{mn} &= {\Phi_m}^T \Phi_n && (5.10) \end{aligned}$$

6. Once we have found the eigenvectors, v_l, u_l, we can then use them for recognition purpose

$$u_l = \sum_{k=1}^{M} v_{lk}\Phi_k \qquad l = 1, 2, .., M \tag{5.11}$$

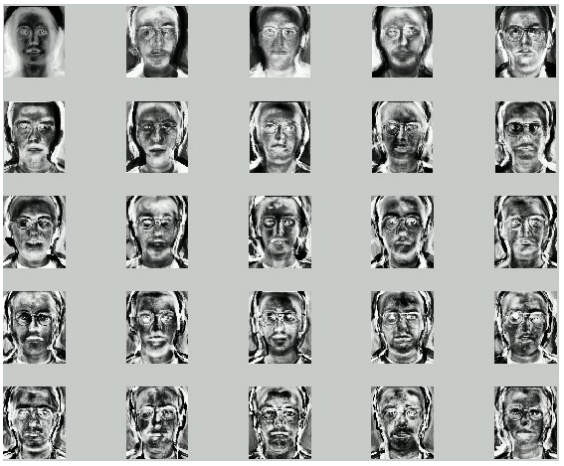


FIGURE 5.19: Eigenfaces of our set of original images.

5.3.2.3 Recognition Procedure

The proposed Face Recognition (FR) system is shown in Fig. 5.20. The procedure is explained as follows.

1. The image containing a face is first preprocessed for illumination normalization using histogram equalization.

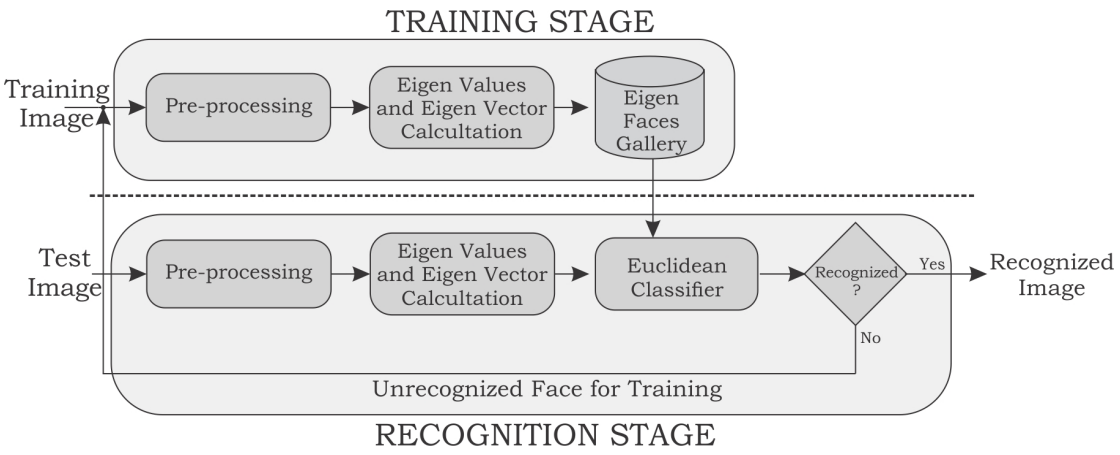


FIGURE 5.20: Block Diagram of the Proposed Face Recognition System.

2. A new face is transformed into its eigenface components. First we compare our input image with our mean image and multiply their difference with each eigenvector of the L matrix. Each value would represent a weight and would be saved on a vector Φ.

$$\omega_k = {u_k}^T(\Gamma - \Psi) \quad \Omega^T = [\omega_1, \omega_2, ..., \omega_M] \tag{5.12}$$

3. We now determine which face class provides the best description for the input image. This is done by minimizing the Euclidean distance

$$\varepsilon_k = ||\Omega - \Omega_k||^2 \tag{5.13}$$

4. The input face is consider to belong to a class if ε_k is below an established threshold θ_ε. Then the face image is considered to be a known face. If the difference is above the given threshold, but bellow a second threshold, the image can be determined as a unknown face. If the input image is above these two thresholds, the image is determined NOT to be a face.
5. If the image is found to be an unknown face, we could decide whether or not we want to add the image to your training set for future recognitions. We would have to repeat steps 1 trough 6 to incorporate this new face image.

5.3.2.4 Advantages and Disadvantages

The primary advantage of the eigenface method is the system's speed and efficiency. The eigenface approach reduces the amount of data needed to identify an individual to 1/1000th of a full sized image (Lau Technologies, 1999). Presence of objects like beard,glasses etc does not decrease performance.

On the other side, the eigenface method has problems identifying faces in different light levels and pose positions. Better results are obtained when the input faces are preprocessed to remove background details, and all faces are equally oriented geometrically. The face must be presented to the system as a frontal view in order for the system to work.

5.3.3 Interest Point Determination

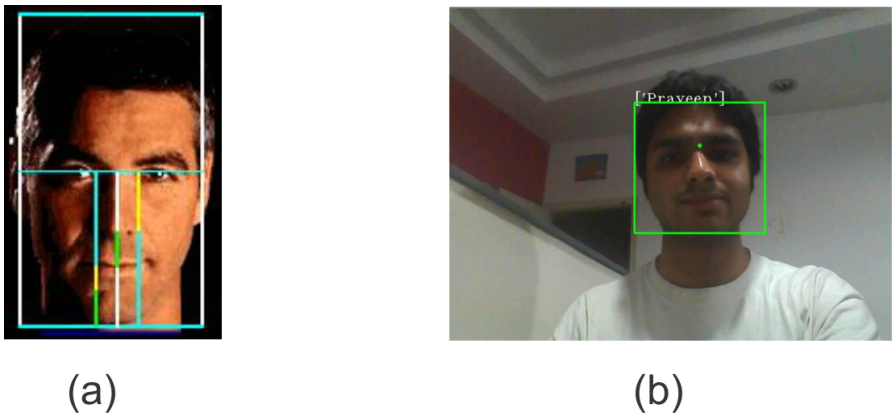


FIGURE 5.21: Interest Point Determination in Human face (a) Standard Golden Ratio governed dimensions in Human face, (b) Result of face detection, recognition, interest point determination.

During interactions with other humans, generally, the point of focus is the bridge between the eyes. We have implemented this so that the humanoid gives the feel of "looking at" you. This simulates the natural human tendency and makes CHLOE's visual behaviour similar to that of humans.

The interest point or the point of focus is obtained using the golden ratio attribute of the human face. The head forms a golden rectangle with the eyes at its midpoint. The mouth and nose are each placed at golden sections of the distance between the eyes and the bottom of the chin (Fig. 5.21(a)). This property is used to find the bridge between the human eyes once the face is recognized.

Fig. 5.21(b) shows the final output of this submodule.

5.4 Speech Recognition

5.4.1 Introduction

The theme of Social interaction and intelligence is important and interesting to an Artificial intelligence and Robotics community. It is one of the challenging areas in

Human-Robot Interaction (HRI). Speech recognition technology is a great aid to admit the challenge and it is a prominent technology for Human-Computer Interaction (HCI) and Human-Robot Interaction (HRI) for the future.

Spoken language is a natural and convenient means to instruct a robot if it is processed reliably. Modern speech recognition systems can achieve high recognition rates, but their accuracy often decreases dramatically in noisy and crowded environments. This is usually dealt with by either requiring an almost noise-free environment or by placing the microphone very close to the speakers mouth.

Speech recognition can be achieved in many ways. But we used Google speech API. This API allows fine control and flexibility over the speech recognition capabilities. The API is designed to enable both brief (one-shot) speech input and continuous speech input. Speech recognition results are provided as a list of hypotheses, along with other relevant information for each hypothesis. This section aims to cover everything a person need to know in order to access the Speech API using any language they choose. Currently the API is used primarily by Google Chrome's Speech Input JavaScript API, which is defined in the W3C Web Speech API Specification [8].

The 4 main sections are

- Acquiring an API Key
- Speech API v2
- Common Parameters
- Working

5.4.2 Acquiring an API key

5.4.2.1 Develop API key

In order to gain access to an API key for Google Speech, you will need to be a member of the Chromium Development Group2 [9]. Once you are a member of the group, a new service will appear on your list of APIs. The API Key that you get from being a member of the Chromium Dev group only allows you access to 50 calls a day. Fairly limiting, especially since you will probably make that many in just trying to figure out the API.

5.4.2.2 Chrome API key

Another option is to use the API Key built in to Chrome. You can do this by capturing the headers that Chrome sends when using the speech input in Chrome.

5.4.3 Speech API Version 2

The first information that became publicly available concerning Googles speech recognition referenced the one shot API endpoint.

https://www.google.com/speech-api/v2/recognize.

This endpoint only allows at most a 40 second clip and limited file size. The exact specifications are not known, but initial testing proves that this endpoint is only good for a few words at best.

Basically this is a POST request to the URL above, with the audio binary in the body of the post. A Content-Type header must be specified in the following format:
Content-Type: audio/x-flac; rate=16000
Where rate is the sampling rate of the file. If this is incorrect then Google will have a hard-time recognizing the audio.

The API takes 2 known formats: FLAC and Speex. However the Speex implementation does not seem to be the standard format and so the recommended format is FLAC. When you call this API you will pass the parameters regarding your request in the URL.

For example:
...api/v2/recognize?xjerr=1&client=chromium&lang=en-US
Tells the server that the file is in English-US, and that the error tolerance should be set to 1, and that the requesting client is chromium. A successful call to this API endpoint will return the results in JSON, with an array of possible responses and their Neural Networks confidence that is correct.

5.4.4 Parameters

The important parameter used in the code are listed below:

1. **Chunk:** Chunks of bytes to read each time from mic.
2. **Rate:** Rate at which the speech signal is sampled.
3. **Threshold:** The threshold intensity that defines silence and noise signal .An intensity lower than THRESHOLD is silence.
4. **Silence limit:** Silence limit in seconds. The max amount of seconds where only silence is recorded. When this time passes the recording finishes and the file is delivered.

5. **Previous audio:** Previous audio in seconds to prepend. When noise is detected, the quantity of previously recorded audio is pre-pended. This helps to prevent chopping the beginning of the phrase.
6. **Flac:** FLAC stands for Free Lossless Audio Codec, an audio format similar to MP3, but loss-less, meaning that audio is compressed in FLAC without any loss in quality. This is similar to how Zip works, except with FLAC you will get much better compression because it is designed specifically for audio, and you can play back compressed FLAC files in media players just like you would an MP3 file.

   FLAC stands out as the fastest and most widely supported loss-less audio codec, and the only one that at once is non-proprietary, is unencumbered by patents, has an open-source reference implementation, has a well documented format and API, and has several other independent implementations.

Some common important parameters that we use in the POST is listed below:

1. **Key:** This is the API key used for the service.
2. **pFilter:** This is the profanity of the filter, 0 for Off, 1 for medium and 2 for Strict
3. **lang:** This is the language of the recording/transcription. Use the standard web-codes for your language. i.e. en-US for English-US, ru for Russian, etc.
4. **output:** The output of the stream. Some values include “pb” for binary, “json” for json string.

5.4.5 Working

Initially Microphone stream configuration is done using correct sampling rate, chunk memory value, intensity threshold and silence limit. Then we take average audio intensity from the mic. If the Intensity is greater than threshold we start listening to Microphone, extract phrases from it and sends it to Google’s TTS service and returns response. a “phrase” is sound surrounded by silence according to threshold. Number of phrases to be processed before finishing the sound record can be controlled and sound is recorded in the .WAV format. This is a temporary .WAV file which is then converted to FLAC format using the code shown below:

```
os.system(FLAC_CONV+ filename+'.wav')
f = open(filename+'.flac','rb')
flac_cont = f.read()
f.close()
```

Now a user agent like Mozilla Firefox or Chrome is required post the query to Google Web speech API service. The following code snippet shows the query request to Google.

```
lang_code='en-US'
googl_speech_url = """https://www.google.com/speech-api/v2/recognize?output=
json&lang=%s&key=AIzaSyCnl6MRydhw_5fLXIdASxkLJzcJh5iX0M4""" %(lang_code)
hrs = {"User-Agent": "Mozilla/5.0 (X11; Linux i686) AppleWebKit/535.7
        (KHTML, like Gecko) Chrome/16.0.912.63 Safari/535.7",'Content-type':
       'audio/x-flac; rate=16000'}
req = urllib2.Request(googl_speech_url, data=flac_cont, headers=hrs)
p = urllib2.urlopen(req)
```

We set the spoken language for the speech recognizer "lang" to the "en-US", i.e., English-United States value that the user has selected. If this is not set, it defaults to the lang of the HTML document root element and hierarchy. Google speech recognition supports numerous languages, as well as some right-to-left languages that are not included in this demo, such as he-IL and ar-EG.

HTTPS POST to Google web service, gives JSON object as a Response. JSON object contains result with 5 utterances and confidence of each utterance. We need to parse the JSON object to get the required text.

5.4.6 Why Google STT?

This makes the entire speech recognition system language independent!! Hence, ensuring its portability to different countries and locations across the globe. Google has full support for 36 languages across the world with numerous accents. Wherever you are, CHLOE can interact with you!

Also, the software is accurate and immune to noise. It has a robust noise reduction and removal mechanism and provides extremely accurate results even in accented speech.

This is also a *proof of concept* that any web server that provides an API can be accessed by an AI system. Basically, what we have, is a robot system that can learn over the largest treasure trove of information available- The internet. It can connect with other cognitive systems, read facebook updates and notifications, the news, etc. etc. The list is infinite. the possibilities are infinite.

There are only two drawbacks of using the Google API (That we can think of). Firstly, its response speed is dependent on your internet connection and bandwidth. In the absence of internet, auditory stimulus fails. Secondly, the Google API is available for only a limited number of hits per day for general users. To get a higher number of requests, one has to register as a business owner requiring use of the API and pay to get access to the servers.

Chapter 6

Mechanical Design

6.1 Insight

It has been argued that to build a machine with human like intelligence, it must be embodied in a human like body. Others argue that for humans to interact naturally with a robot, it will be easier for the humans if that robot has humanoid form. A third, and perhaps more concrete, reason for building a humanoid robot is to develop a machine that interacts naturally with human spaces. The architectural constraints on our working and living environments are based on the form and dimensions of the human body. Consider the design of stairs, cupboards and chairs; the dimensions of doorways, corridors and benches. A robot that lives and works with humans in an unmodified environment must have a form that can function with everyday objects. The only form that is guaranteed to work in all cases is the form of humanoid.

The mechanical design of the humanoid requires careful and complex trade-offs between form, function, power, weight, cost and manufacturability. For example, in terms of form, the robot should conform to the proportions of a human with upper body length of 520mm. However, retaining the exact proportions compromises the design in terms of the selection of actuation and mechanical power transmission systems. This section describes the final mechanical design and how the balance between conflicting design requirements has been achieved.

6.2 Proportions

The target proportions for the robot are based on biomechanical data of the human form. Fig. 6.1 shows the proportions of the frontal plane dimensions of upper body of

TABLE 6.1: Comparison of CHLOE's mass distribution with human mass distribution.

Body Component	CHLOE's mass	CHLOE	Human
Head	0.58kg	12.36%	15.63%
Torso	2.86kg	61.21%	66.40%
Arm	1.24kg	26.43%	17.97%
Total	4.68kg		

a 50th percentile male based on data from a United States survey [10]. The dimensions shown in millimetres indicate the appropriate sizes of anatomical features against the comparable dimensions on CHLOE.

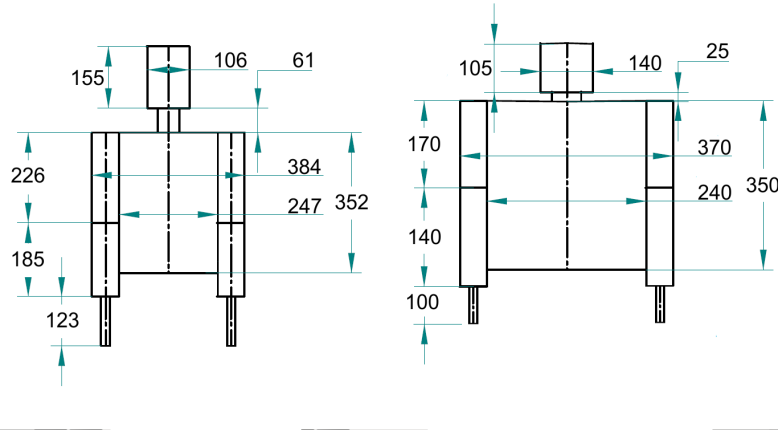


FIGURE 6.1: The proportions of typical human anatomy compared to the matching proportions of CHLOE's anatomy.

By comparison, the proportions of length and width of the head is unlike the human counterparts. This was dictated by the presence of actuators in the head to control the motion of eyes. But however even in humans the size and shape of faces changes from person to person, hence it does not appear out of proportions or inhuman. It is also seen that the proportions of the torso match exactly with that of a human. The overall effect is still convincingly human-like.

The changes in volume required to house the actuators, as well as the mass of the actuators themselves have an effect on the mass distribution. Table 6.1 shows the mass distribution of CHLOE's compared to that of a human [11]. Even though the arms are heavier than the human counterpart, they are significantly inferior in power. This excess weight is due to the large number of actuators which provide numerous degree's of freedom to make up for the lack of power. CHLOE's torso is as heavy as a human even though it lacks any functional parts. This is done to provides a stable platform to anchor arms that have a high torque and to maintain the center of gravity.

TABLE 6.2: DOFs at different parts of the system.

Part	No of DOF
Eye	4
Neck	2
Shoulder	4
Elbow	2
Wrist	4
Hand	4
Total	20

6.3 Architecture

The extent to which human joint function can be replicated is another key factor in robot design. Fig. 6.2 shows the degrees of freedom (DOF) contained in each joint area of the robot. In the cases where there are multiple degrees of freedom (for example, the shoulder) the joints are implemented sequentially through short links rather than as spherical joints. Other key differences to the human form are the lack of a continuous flexible spine. The design incorporates a pair of fully functioning arms, neck, as well as eye socket. Each Eye ball has 2 DOF that enables the person to interact with the robot more naturally and also helps in exhibiting emotions. The neck has 2 DOF that helps to track the object of interest and keep it within the camera frame. Also these two DOFs provided for the head can be used in the situations where normal human gestures indicated by head such as acceptance, rejection and different emotions are to be expressed. The arms have joints at shoulder, elbow, wrist and hand like a human. This provides an additional 7 DOF for each arm. The total DOF is given in Table 6.2.

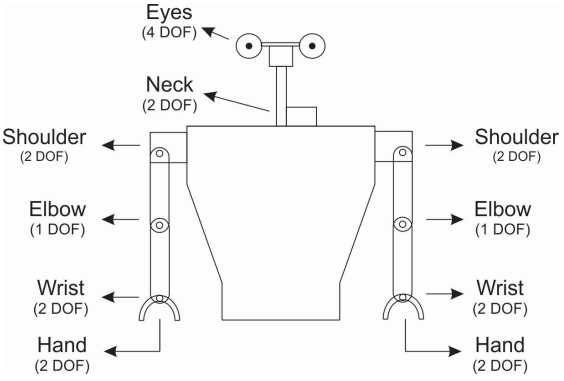


FIGURE 6.2: The location of the joints, indicating the degrees of freedom in each joint.

6.4 Head

The head and the neck have been developed to match the proportions and functionality of a human. It has been designed with a modular approach to create a backbone structure to which sensors and response units can be incorporated at later stages. It forms a framework with provisions to attach sensors and actuators as and when required. It consists of 2DOF at the neck to providing lateral and vertical movements similar to a human neck. The entire structure is made up of 5mm foam board material. This helps us to achieve low weight and high strength characteristics that reduces the strain on the actuators. These advantages were evident when evaluated against PVC pipes, wood and aluminium which are not only difficult to fabricate but also expensive. The 3D rendered model of the structure is shown in Fig. 6.3.The head structure was designed in AutoCAD 2014 (Appendix E).

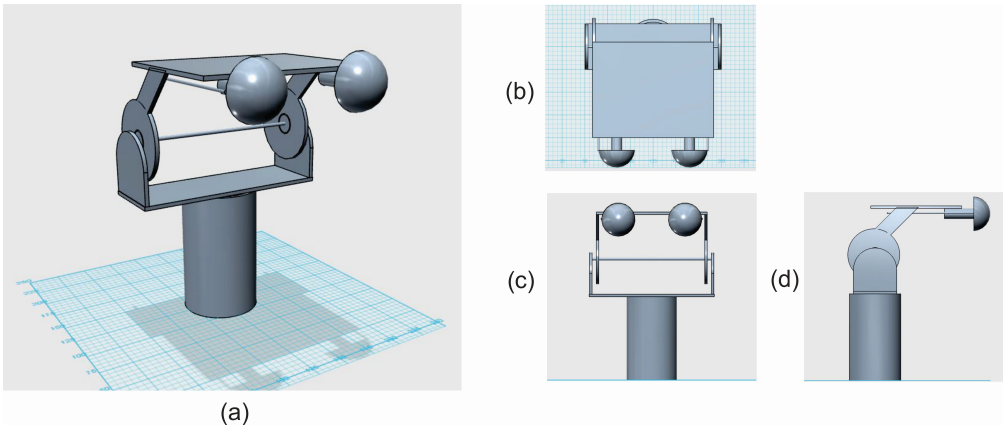


FIGURE 6.3: 3D rendered model of head (a) Isometric View, (b) Top view, (c) Front View, (d) Side View.

It has been said that eyes are the windows to soul. Humans extensively use the eye not only for the purpose of vision but also for expressing emotions and reflecting their inner feelings. We use eyes for maintaining eye contact while conversation which is a very important posture. In order for a humanoid to be as human as possible it is necessary for it to have eyeballs for the above said reasons.

Hence one of the peripherals we include in the preliminary design is eyeballs. The design consists of 2 eyeballs that are harbored into the head structured through universal joints. This enables the rotation of eyeballs over 180° on both the axis. This motion of this eyeball is controlled by a system of wire frames. It consist of 2 servo's each controlling 1DOF. Both the eye balls are connected to both the servo's over a wire frame design system as shown in Fig. 6.4. The yellow arrow's indicate the movements of various parts when the servo rotates in anti-clockwise direction resulting in both the eyeballs rotating right. Similarly the Blue arrow's indicate the series of motion involved in rotating the

eyeballs down. A combination of these 2 motions can be used to rotate the eyeball to any location in a solid angle of 2π steradians.

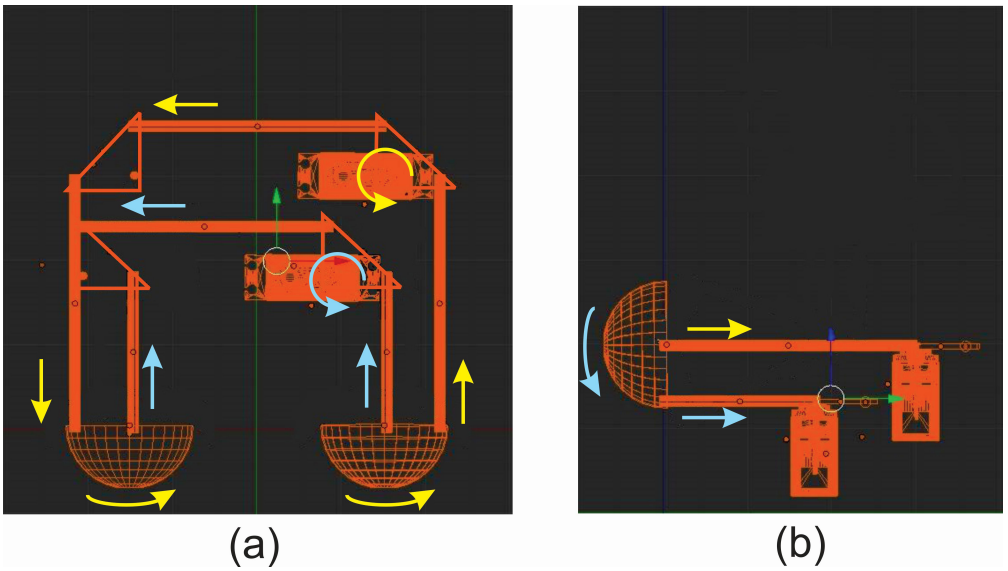


FIGURE 6.4: Cad model of Eye Ball Wire Frame system. (a) Top View, (b) Side View

6.5 Robotic Arm

An important feature of upper human body is the arms. The human arm has evolved over the due course of millions of years to become a very dexterous tool for handling a multitude of tasks of varying complexity. A similar feature on an humanoid can enable it to perform operations beyond the scope of a normal human such as lifting heavy loads, handling objects at extreme temperatures, etc. Apart from these arms are also used by humans as a means of communication. Humans perceive this information subconsciously and do not realize the importance of hand gestures and postures in day to day communication. To enable a robot to be perceived and interacted with as a human it is also required for it to perform these tasks. Hence we incorporate both the left and the right arm into the design.

The arms are made up of PLA and ABS plastic through a 3D printing process. The 3D printing process used here was fused material deposition method that uses multiple layers of printed plastic in a CNC system. This method was chosen because it enabled ease of design and fabrication. The process is relatively simple and produces accurate models with a resolution of ± 0.1mm at a affordable cost compared to a similar process of milling using Aluminium or any other metal. It has acceptable strength and temperature tolerance. Since it is molded out of plastic it provides the flexibility of changing the design post printing to accommodate prototyping errors. The parts were printed on

TABLE 6.3: 3D printing specifications.

Specification	Value
Layer height	0.2mm
Shell thickness	0.8mm
Bottom/Top layer thickness	0.9mm
Fill Density	25%-60%
Print Speed	60
Print Temperature	ABS: 230°, PLA: 180°
Filament Diameter	1.75mm

Brahma3 anvil 3D printer with the print specifications shown in Table 6.3. Some of the 3D printed parts are shown in Fig. 6.5.

(a) (b)

FIGURE 6.5: (a) All 3D printed parts for 1 arm before assembly. (b)3D printing of a part in progress.

The design of the arm is broken down into 5 sections, viz., shoulder, arm, forearm, wrist and hand. Each of these were designed independently considering the dimensions of various actuators that were to be incorporated into the design. All the 5 parts were further subdivided into smaller parts that are easy to be printed and assembled. This modular construction enabled us to modify the design without the need to print the entire arm. The Table 6.4 shows the number of sub parts for different sections of the arm. Fig. 6.6 shows the various sections of the arm assembled. The objectives of the design were

- Linking the parts in a way to achieve the highest rigidity possible
- Saving maximum plastic possible and avoiding heavy links to decrease print time.
- Achieving maximum aesthetic satisfaction by maintaining proportions and wights

The various DOF's are shown in Fig. 6.7 using different color codes. The design provides 2 DOF (orange) for the shoulder to simulate the ball and socket joint of an human

TABLE 6.4: Table depicting the number of subparts and actuators in each section of the arm.

Section	No. of Subparts	No. of Actuators
Shoulder	4	2
Arm	8	0
Forearm	9	2
Wrist	6	2
Hand	11	1
Total	38	7

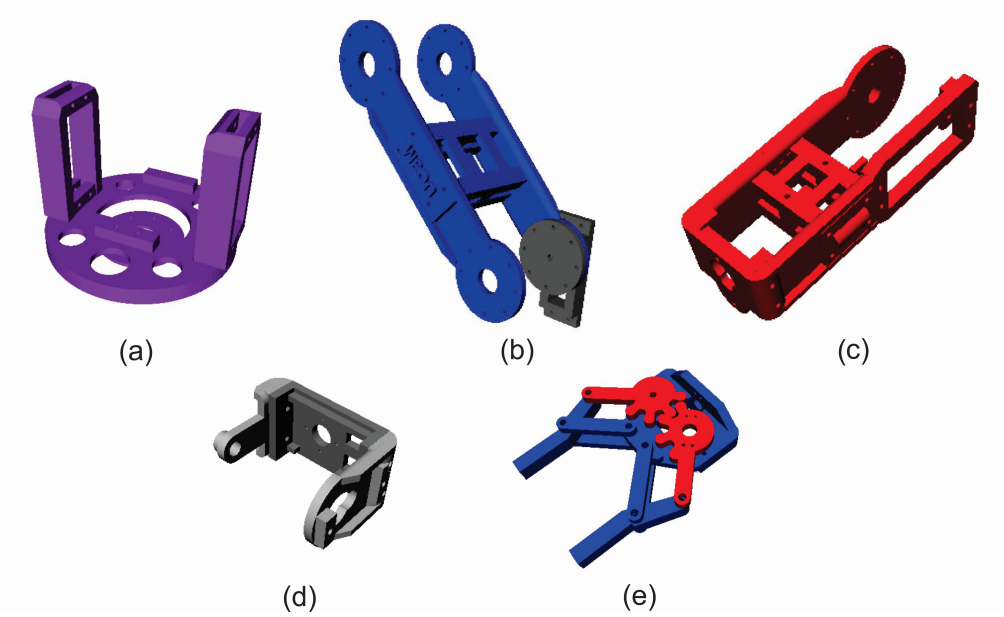


FIGURE 6.6: Different subsections of the arm (a) Shoulder, (b) Arm, (c) Forearm, (d) wrist, (e) hand.

shoulder enabling each arm to cover 2π steradians on its respective side. 1 DOF (Yellow) for the elbow joint provides a 180° movement of the joint which is more than what human's can achieve. The 2 DOF (Blue) for the Wrist enables the robot to manipulate the hand to required orientation for picking up objects which is similar to the ball and socket joint present is humans and finally the 2 DOF (Green) for the Hand enables the humanoid to pick up objects with its gripper and also rotate it. The assembling are depicted in Appendix A.

6.6 Torso

Torso in a human plays an important role of harboring various vital organs and also protecting them. But in a humanoid it acts as a support structure to host other functional parts such as the head and the arm. To make the humanoid as close as possible to the

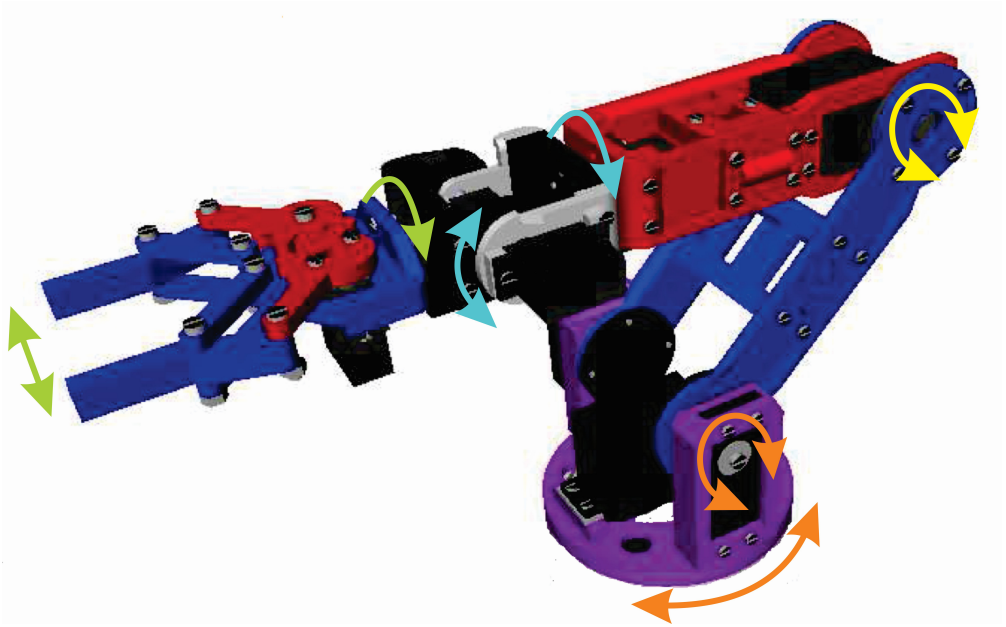


FIGURE 6.7: Completely assembled arm depicting various DOF's.

man it is giving a shape similar to a human. The basic requirements for the torso is to strong and heavy enough to maintain the center of gravity even when the arms are at full stretch. It should also be able to support the stresses of high torque actuators used. The empty place inside the torso enables us to act as a common collection points of various signals and space for equipment to process them. It also harbors the power supply and actuator control system for the humanoid. The frame of the body is fabricated using mild iron plates of 50mm wide. They are held together by arc welds. To improve the appearance it is given an aesthetic cover.

Chapter 7

Electronics Design

The electrical subsystem consists of actuators, a device to produce signals for the actuators, a device to decide the control signals, a device to provide the power to all the other electrical systems, sensors, etc. A brief explanation on the functionality and specification of these devices is explained in this section

7.1 Actuators

The various degrees of freedom at various joints of the robot are controlled by servo actuators. Servo motors are a special kind of motors that provide a feedback on the current position enabling the system to move to a precise position and also hold it. Each servo motor has 1 degree of freedom and can rotate from 0° to 180°. The current system uses a total of 18 Servo motors of varying torque capacity. Table 7.1 shows the total number of servos used and their characteristics. The servos have a operating voltage range of 4.8V to 6V. We have chosen an operating voltage of 5V for all servos. The servos draw a minimum current of about 150mA at no load and can go up to 900mA at full load. At stall, the biggest servo's can draw a current drawn can go upto 1.3A. The servos have a maximum speed of 0.16sec/60°. The process of calculating the required servo torque at a joint is given in Appendix B.

The position of these servos are controlled by a PWM signal from a controller. The frequency of this signal is 50Hz. The angle to which the servo is suppose to rotate is decided by the high signal period of each pulse. A high signal period of 0.5ms corresponds to a 0° and a high signal period of 2.5ms corresponds to an angle of 180°. The rate at which this high signal period varies can be used to control the speed of the servos. A controller is used to produce these signals.

TABLE 7.1: Servo Actuators.

Servo	Weight (g)	Torque (kg/cm)	Speed (s/60°)	Size (mm)	Places Used
Avionic AV55MG	55	8.5	0.2	40.7 x 19.7 x 42.9	Shoulder
Avionic AV52MG	58	6.0	0.18	40.7 x 19.7 x 42.9	Shoulder, Elbow
Vigor VTS-08B	43	4	0.25	43 x 23 x 38.2	Neck
Vigor VTS-05A	8	1	0.19	22.5 x 12.8 x 23.3	Hand
TowerPro SG90	9	1.8	0.10	22.2 x 11.8 x 31	Wrist, Eyeballs

7.2 Actuator control

An atmega328 on an Arduino Uno (Fig. 7.1) is used as bridge between the high level processor and the actuators. The function of this controller is to convert the servo position data sent by the master processor into PWM signals that are required to control the servo motors. To control the 18 servo motors individually we require 18 PWM signals. But this is beyond the scope of atmega328. Hence we use an auxiliary PWM generator chip TLC5940. This chip interfaces to the controller via SPI bus and can produce 16 16-bit PWM signals. In addition to this it also supports daisy chaining that enables us to obtain more than 16 PWM signals. Here we use 2 TLC5940 daisy chained and interfaced with atmega328 via SPI. The circuit diagram for the same is shown in Appendix C. This has been implemented on a general purpose PCB (Fig. 7.2). The communication protocol between arduino and TLC5940 is explained in Appendix F

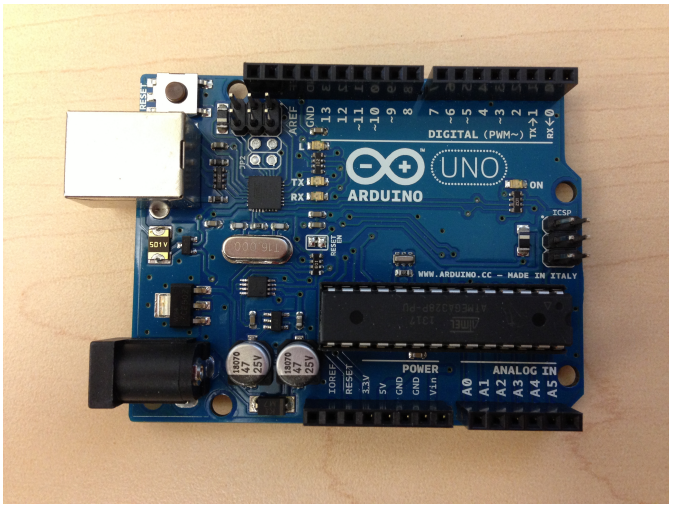


FIGURE 7.1: Arduino Uno.

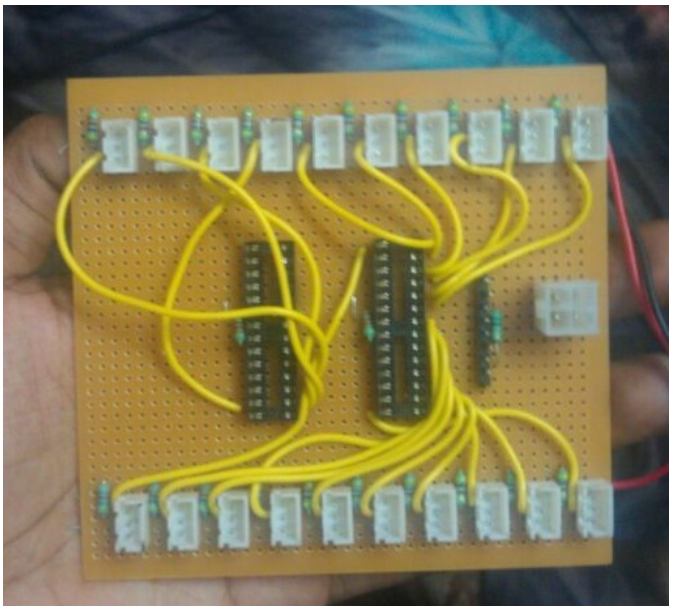


FIGURE 7.2: PCB board for TLC5940.

Another function performed by the micro-controller is to control the speed of the servo motors. As seen earlier servo motors can rotate 60° in less than 0.2s. Such speeds are not acceptable in most of the humanoid motions and can cause undesired situation. To control the speed of rotation an algorithm has been developed that remembers the current servo position and rotates relative to it, based on the speed specified by the master processor. A pseudo code of the algorithm used to control the speed is given below

```
1  Servos = No of Servos
2  angle_current[Servos] = Initial angles
3  angle_new[Servos] = 0
4  speed[servos] = 0
5  while 1:
6          if new data from master
7                  update new angle and speed data to arrays
8          for all servos
9                  if angle_current != angle_new
10                         angle_current = angle_current + speed
11                         send angle_current to TLC5940
12         sleep for 100ms
```

We first initialize 3 arrays angle_current to store the current angle of the servo, angle_new to store the new angle the servo is suppose to reach and speed that holds the rate at which that particular servo must turn. It is defined in terms of number of steps for every 100ms. When a new position data arrives from the master processor it updates the angle_new and speed arrays. Then the code iteratively checks for servos that needs to be rotated. Those servo's that have to be rotated are updated with the new incremented angle into angle_current array. Then this value is sent to the TLC5940 chip to update its

position data. This enables us to control the speed of all the servo motors with precision to mimic the human behavior.

7.3 Communication Protocol

The communication between the master processor and the arduino happens through serial over USB protocol and is always one way. A new protocol has been built over this protocol to effectively communicate the angles and speeds of the servo motors from the master to the arduino. A 3 byte packet is used to carry the position information of each servo. The packet can be broken up as follows

- Byte 1: Servo number - range 1-20
- Byte 2: New servo angle - range 0-180
- Byte 3: Speed - range 1-180

Ex: To rotate servo 2 to an angle of 80° at a speed of 5°/100ms, the 3 byte packet to be sent is 0x025005. The first byte 0x02 represents the servo number, the second byte 0x50 represents the servo angle and the 3rd byte 0x05 represents the speed.

7.4 Camera

The sense of vision for an humanoid is accomplished with a camera that is mounted on the head. This camera helps to track people and objects that travel in its vicinity. A camera with a resolution of 640 x 320 pixels is sufficient to perform the necessary algorithms. Some important factor that were considered were

- Physical dimensions of the camera as it is mounted on a mobile mount
- Auto-focus capability to automatically focus on near and far objects like our human eye
- Good light sensitivity in room lightning as well as bright sunlight
- A frame rate sufficient enough to track the movement of objects
- Provide a USB2.0 interface

Based on these requirement Logitech C270 was chosen. Some of the specifications are

- Dimensions: 70x18x30mm, Weight 73g
- Maximum Frame Size: 1280x720 pixels
- USB 2.0 port
- Build in microphone

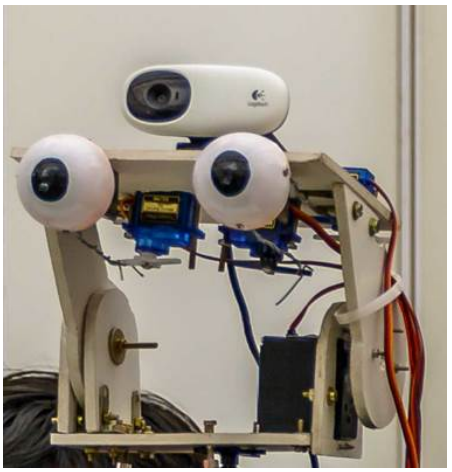


FIGURE 7.3: Webcam mounted on the head.

TABLE 7.2: Power specifications of the system components.

Component	Operating Voltage(V)	Maximum stall Current(A)	Continuous Current (A)	Quantity	Power Consumption(W)
Arduino	5V	0.2	0.2	1	1
TLC5940	5V	0.3	0.3	2	3
1kg Servo motors	5V	0.7	0.25	2	7
1.8kg Servo motors	5V	0.8	0.3	8	32
4kg Servo motors	5V	1.1	0.4	2	11
6kg Servo motors	5V	1.4	0.6	4	28
9kg Servo motors	5V	1.9	0.74	2	19
Total					101

7.5 Power Requirements

The system incorporates a large number of Servo motors for motion, an efficient power system is required to manage them. Table 7.2 showing the power consumption of the different modules of the system. Hence to provide this power the best and efficient option is to use a switched mode power supply. A conventional 450W SMPS is used in this system. It not only provides the required current but also provides protection against shorts, surge currents and over voltages.

Chapter 8

Cognitive Engine and the Master Control framework (The COG)

Cognition can be defined as "the act or process of knowing in the broadest sense; specifically, an intellectual process by which knowledge is gained from perception or ideas" (Webster's Dictionary). While cognitive psychology is the dominant school of thought today, the *information processing approach* is the dominant view within this area. The information processing approach focuses on the study of the structure and function of mental processing within specific contexts, environments, or ecologies.

The term *Cognition* has a two fold definition. One it implies an ability to understand how things might possibly be, not just now but at some point of time in future, and to take this into consideration when determining how to act. Second the *Memory* which is an act of remembrance of what happened at some point in the past helps in anticipating future events. Memory is very important aspect of cognition because it helps to predict the future using past and then imbibing what does actually happen to adapt and improve the systems anticipatory ability.

Cognitive systems exhibits effective behaviour through perception, action, deliberation, communication and through either individual or social interaction with the environment. The characteristic of a cognitive system is that it can function effectively in circumstances that were not planned for explicitly when the system was designed, that is , it has some degree of plasticity and is resilient in the face of the unexpected.

The beautiful concept of physical embodiment is as follows. During the early stage of human development, interactions with various physical environments have a major role in determining the information structuring inside the individual such as body representation, motor image and object permanency. On the other hand, at the later stage,

due to the interactions with other agents, social behaviours such as early communication, joint attention, imitation of various actions including vocalization, empathy and verbal communication gradually emerge. Whether the individual is in mature or premature state, the common aspect of these developmental processes is a sort of scaffolding by the environment including other agents that triggers the sensorimotor mapping and promotes the infants autonomy, adaptability and sociality, directly or indirectly, and explicitly or implicitly. This is exactly how we have tried to implement the master control framework

There are four major theories of how we humans process information:

- Stage Approach
- Levels-of-Processing Theory
- Parallel Distributed processing approach
- Connectionistic models

We use The Stage approach for memory and Parallel Distributed processing approach for Information processing and determination of attentional states.

8.1 Memory organization and Stage approach

The focus of this model is on how information is stored in memory. The model is based on the work of Atkinson and Shriffin (1968) and proposes that information is processed and stored in three stages:

- Sensory memory
- Short term memory
- Long term memory

8.1.1 Sensory memory

During every moment of an organism's life, sensory information is being taken in by sensory receptors and processed by the nervous system.The information people received which is stored in sensory memory is just long enough to be transferred to short-term memory. Two aspects of our system happen to be of sensory memory type.

1. The servos for the arms *reset to equilibrium* every time the system is restarted. This is analogous to the gait and comfort positions in a human body, or the

positions we are most accustomed to and are in, when not in any kind of test or pressure.

2. The Camera has an in-built buffer that stores around 64kB of images (Around 3-5 frames, based on Camera refresh rate and sampling rate used). This is similar to the human eye, which has a visual acuity of 0.0625s. Everything seen is not posted to short term or long term memory. Only the appealing ones, set by the main process are actually comprehended as changes and shifted to permanent or short term memory. (Fig. 8.1)

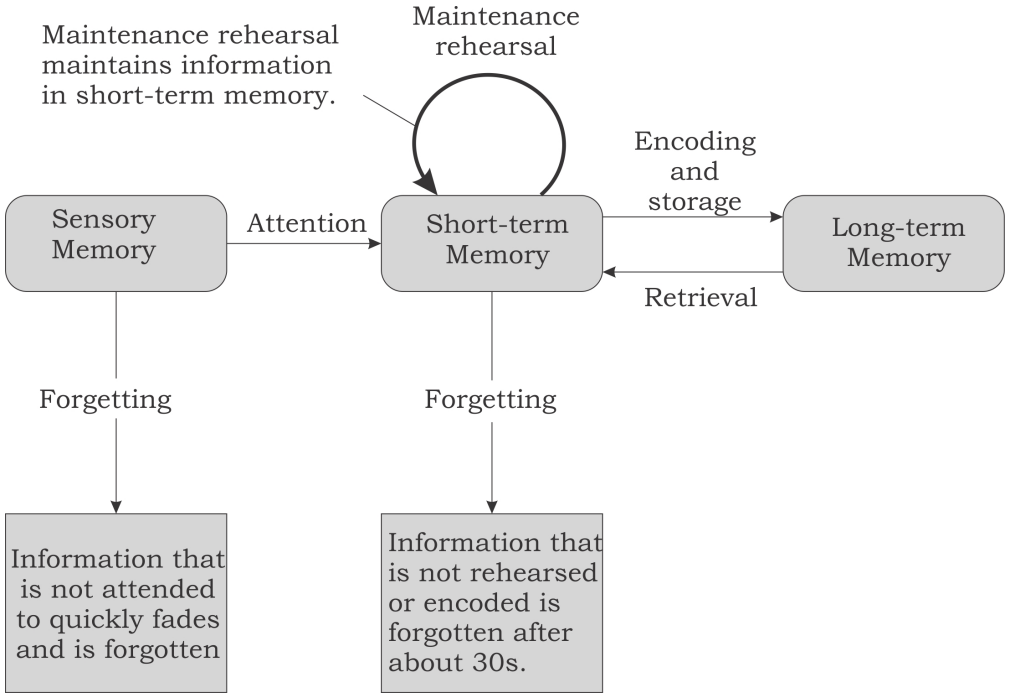


FIGURE 8.1: Memory organization in CHLOE and their relationship.

8.1.2 Short term memory

Implemented as Chunk memory for Voice stimuli. Humans remember instructions and statements of importance that they attach knowledge to. But commands and such stimuli requiring immediate response do not need to be stored in permanent memory. hence, after processing, these Chunks are destroyed. Chunking, in psychology, is a phenomenon whereby individuals group responses when performing a memory task. Tests where individuals can demonstrate "chunking" commonly include serial and free recall tasks. Both tasks require the individual to reproduce items that he or she had previously been instructed to study. Test items generally include words, syllables, digits/numbers, or lists of letters. Presumably, individuals that exhibit the "chunking" process in their responses are forming clusters of responses based on the items' semantic relatedness or perceptual features. The chunks are often meaningful to the participant.

Humans can retain five to nine pieces of information in their short term memory. We have programmed CHLOE to store upto 11 pieces of information as chunk memory, only from the advent of sound to the first long pause, as mentioned in Chapter 5, Section 5.4. Hence a group of words or syllables associated with one another due to their *order* or *manner of occurrence* are stored as a chunk .flac file and deleted when the limit of 11 chunks is reached.

8.1.3 Long Term Memory

Long-term memory (LTM) is the final stage of the dual memory model proposed by Atkinson and Shiffrin, in which data can be stored for long periods of time. While short-term and working memory persists for only about 20 to 30 seconds, information can remain in long term memory indefinitely. According to Mazur (2006), long-term memory has also been called reference memory, because an individual must refer to the information in long-term memory when performing almost any task.

TABLE 8.1: Some sample commands and objects trained and stored in a .csv file- Permanent memory.

Command	Audiomode label
good morning	1
hello	2
how are you	3
what is your name	4
look down	5
learn	6
find	7
yes	8
no	9
ok	10
orbit	11
book	12
mobile	13
cap	14

In CHLOE's memory, the long term components are for three functions and stored in 3 ways:

1. Firstly, the auditory commands that it can understand. These are stored as a .csv file with a label. All newly learned commands and object identities are stored as the label or *Audiomode* (Table 8.1).

2. Secondly, a gallery of images containing objects linked to the label of *Audiomode* are stored in memory (Fig. 8.3). The linking is done automatically by the writing segments in the main control framework.
3. The faces learned are stored as an eigen face gallery after capturing images on command and training the system. The eigen vectors are stored for the purpose of comparison. These are linked to a separate .csv file where names of the people learned are stored with a corresponding label. Hence, when face is recognised, name is shown (Fig. 8.2).

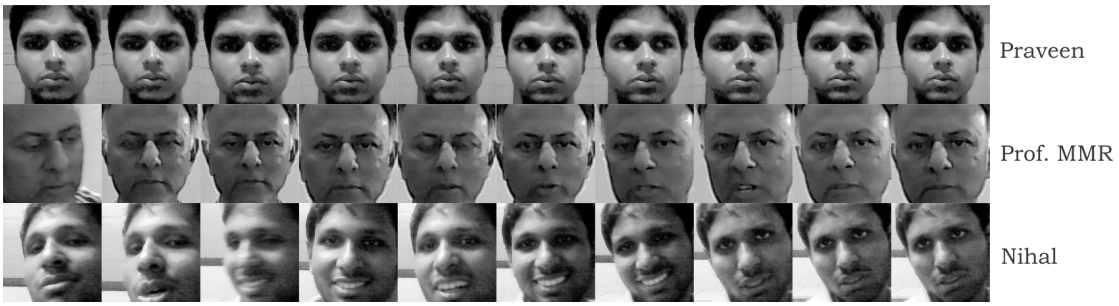


FIGURE 8.2: Learned Face Image database, stored with name and label- Permanent memory.

FIGURE 8.3: Learned object database, stored with name and label- Permanent memory.

8.2 Information handling using Parallel distributed processing

The parallel-distributed processing model states that information is processed simultaneously by several different parts of the memory system, rather than sequentially as hypothesized by Atkinson-Shiffrin. We propose are eight major aspects to a parallel distributed processing model:

- A *set of processing units*
- A *state of Activation*

- An *output function for each unit*
- A *pattern for connectivity* among units
- A *propagation rule* for propagating patterns of activities through the network of connectivities
- An *activation rule* for combining the inputs impinging on a unit with the current state of that unit to produce a new level of activation for the unit.
- A *learning rule* by which patterns of connectivity are modified by experience.
- An *environment* within which the system must operate

Python has good support for both *thread-based parallelism* and *process-based parallelism.* Since our framework does not require all processes to be parallel, we opt for thread-based parallelism. This is less computationally intensive, but sometimes slow to respond. But it is within tolerable limits, which is what helped us decide.

CHLOE's brain (Which we call the COG), is implemented using two threads which run simultaneously. These are the *vision thread* and the *speech thread.* This, again, simulates the Human brain and conforms to Parallel Distributed theory of Information handling as a cognitive method.

8.2.1 Vision Thread

As stated earlier, PDP systems consist of a state of activation and an activation rule. The three submodules in the vision module are Bright-object detection, Object recognition and face detection with recognition. Of these, bright-object detection and face recognition are purely controlled by a visual stimulus. Object recognition is controlled by the Audiomode parameter obtained from the speech thread.

Hence, first, the activation state for each submodule is defined (Table 8.2). Next, the modules are divided to two parts- the *Detect* part and the *Find* part. The former is used to find if the activation state is encountered. Only if it is, the latter is executed. The vision thread constantly runs the *Detect* part of each submodule in a round-robin fashion, and vectors to one process if the activation state is found to be true.

8.2.2 Speech Thread

This thread involves just one process, i.e., speech recognition. Once speech is detected, it is recorded in chunks and stored as short term memory. A request is then made from the Google STT Server using the API, following which the detected string is obtained from the server. The string is compared with the commands and objects known to

TABLE 8.2: Activation states for all possible stimuli causing information with respect to CHLOE.

Thread	Submodule	Activation state	Return parameter
Speech	Speech recognition	Always 1	Audiomode label
Vision	Bright-Object Detection	if(contour found)	Contours
	Face Detection and Recognition	if(!contour)	Point of interest or point of focus
	Object Recognition	if(Audiomode)	Point of interest

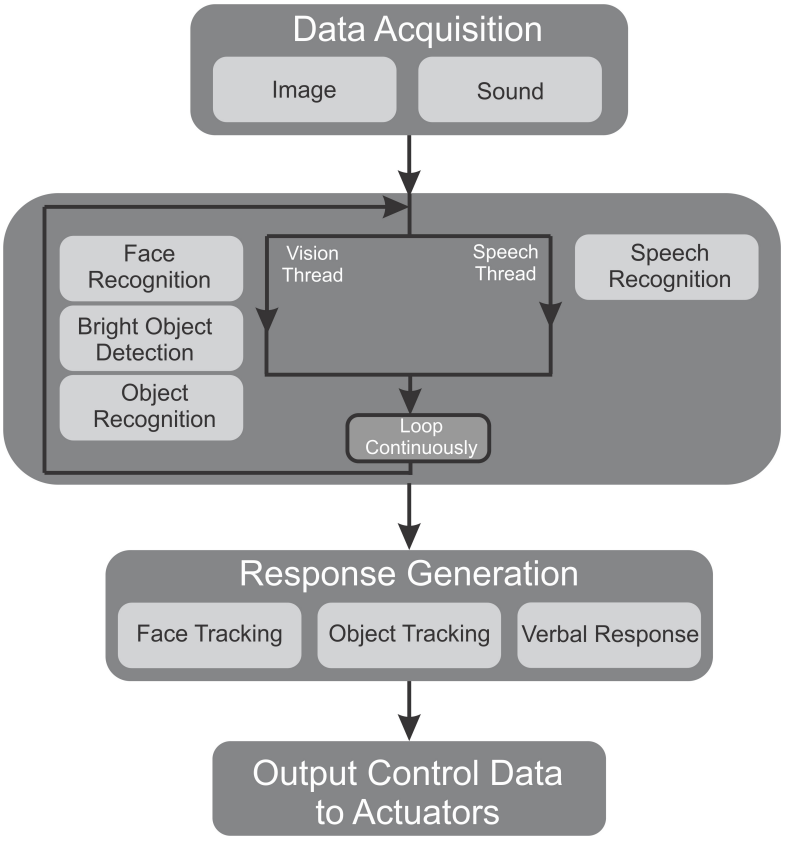


FIGURE 8.4: Flowchart showing the main processing levels in CHLOE, a system with a PDP Cognitive framework.

CHLOE and appropriate action is taken. This response could be a mechanical action, a change in attentional states or a change in activation states of the processes in the vision thread (Table 8.2).

8.2.3 Learning Mechanism or Learning Rule

This is the most important section of the COG. It answers the question 'When to Learn?'. A cognitive system is defined by its ability to learn, and ability to assimilate memory.

The face recognition has an in-built facility to comprehend that a visible face is NOT recognised. This is an internal cue to learn the visible new face. On the other hand, the object recognition takes cues to learn from a valid Audiomode. Simple instructions like 'learn' and 'find' are understood by CHLOE, as mentioned earlier.

8.3 Response Generation

As mentioned before, CHLOE is a social being. Hence, it is not suitable if the results are something to be seen on a computer screen. The response has to be a visible one, and something a human may do. We include this section in this module as these response generation codes are called by the Master control functions and do not have any utility on their own. We have three parts in this submodule.

8.3.1 Speech Synthesis and speech response

Speech synthesis is the artificial production of human speech. A computer system used for this purpose is called a speech synthesizer, and can be implemented in software or hardware products. A text-to-speech (TTS) system converts normal language text into speech. An application or other process sends text to a speech synthesizer, which creates a spoken version that can be output through the audio hardware or saved to a file. The ultimate goal of text-to-speech (TTS) synthesis is to create natural sounding speech from arbitrary text. Moreover, the current trend in TTS research calls for systems that enable producing speech in different speaking styles with different speaker characteristics and even emotions.

Speech.py is a Python module that provides a clean interface to Windows's voice recognition and text-to-speech capabilities. It's very easy to use within a program that needs to listen for specific phrases or general speech, or that needs to speak. It is a powerful and unified module to developers who want to use speech synthesis in their software.

The idea behind using speech as a response is to make the CHLOE as human as possible, and able to respond to speech with requisite speech.

8.3.2 Face/Bright object tracking

Attention is the behavioral and cognitive process of selectively concentrating on one aspect of the environment while ignoring other things. Attention has also been referred to as the allocation of processing resources. Hence, in perception, a system assigns, what we call, attentional states, to objects and faces in the field of vision.

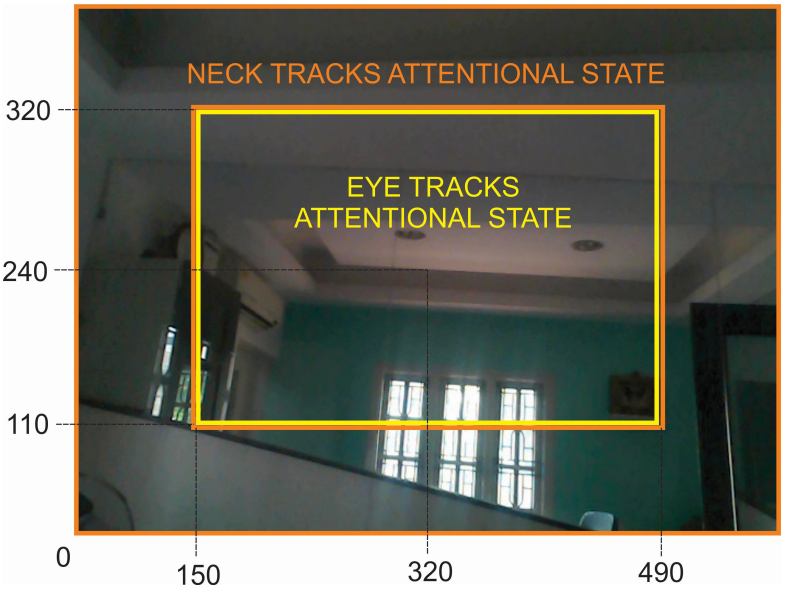


FIGURE 8.5: Division of field of vision for eye and neck based tracking of attentional state.

We show this change in attentional states by manifesting response as eyeball tracking. This means, that the attentional state is tracked by the head, with the point of focus as the point of interest. CHLOE's eyes seem to look at the cause of the attentional state, which could be a brightly colored object or a human face. This is exactly how a human would behave, hence our impetus in its implementation.

The ultimate objective of any resultant motion of the neck and eyeballs is to place the cause of heightened attention in the center of field of vision. In Fig. 8.5, the field of vision is divided into two regions: The inner box, for which the eye moves; and the outer box, for which the neck moves.

A simple linear relationship is established between the differential of the ideal point of focus and the nearest ideal point. The ideal point is the boundary for the neck, and the center of the FOV for the eyes.

- For the neck motion, the distance between the inner boundary and the point of interest is found. This differential is mapped to the servo rotation degree by a linear one-to-one relationship using the map function in python.
- For the eye motion, the differential between the point of interest and the center of field of vision is found and resolved to x and y coordinates. These x and y differentials are linearly mapped to max and min of the eye vertical and eye horizontal servos.

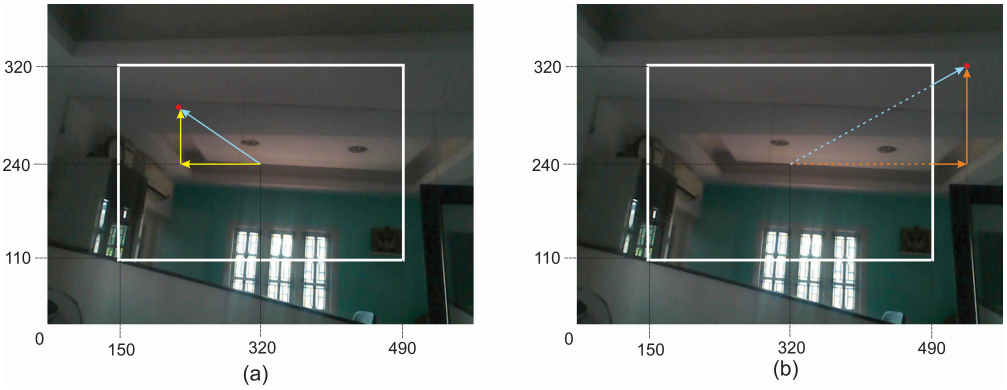


FIGURE 8.6: Vectors showing motion (a) Eye horizontal, vertical and resultant, when the interest point in skewed from the center, (b) Neck horizontal, vertical and resultant, when the point of interest is outside the eye motion region.

8.3.3 Arm motion

For object recognition, the incorporated response is arm motion to *point at* the object. Once the Audiomode is received ("Look down"), the neck motors enable CHLOE to look at the scene in front of it. The scene viewed by the camera, despite appearing to be rectangular in nature, actually gets mapped onto a trapezoidal field-of-view (FOV) (Fig. 8.7) in reality. It is extremely difficult and also impractical to expect the arm to reach every point in the FOV. Hence, we divide the trapezoidal region into discrete regions as shown in Fig. 8.8. We have chosen 16 regions depending on the object size.

Ideally, the arm has to be maneuvered onto the exact centers of the regions, as indicated by the blue dot in each region in Fig. 8.8. However, in the regions on the edges of the FOV, the object is more likely to be placed inwards in order to avoid cutting across the edges of the FOV. Thus, the arm is programmed to move to re-aligned centers of the regions, indicated by the red dots. This increases the probability of the arm reaching the exact position of the object wherever it is placed in the FOV. By dividing the FOV into a higher number of regions, we can reach more areas of the scene albeit at an increased cost of complexity (Appendix D).

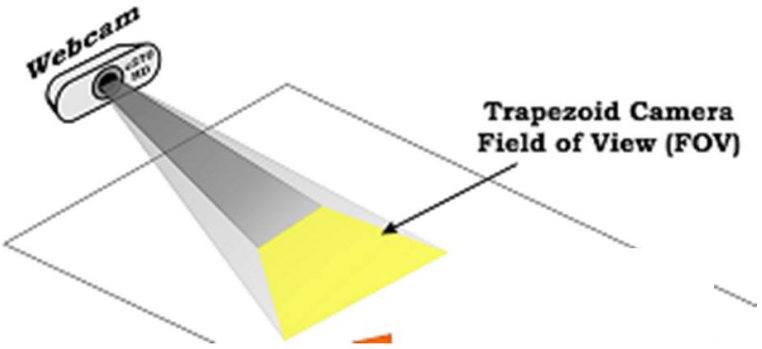


FIGURE 8.7: Camera field of view.

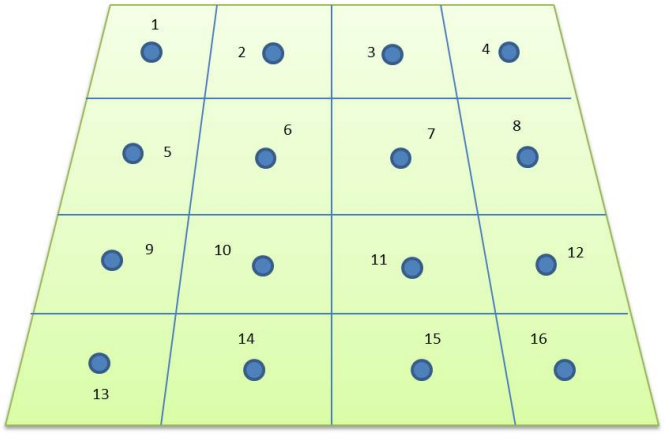


FIGURE 8.8: Camera field of view mapped to a trapezoid- Blue lines indicate sectors, blue dots indicate geometric centers.

Chapter 9

Potential Applications

9.1 Rehabilitation for the Autistic

This was the main intended use for a cognitive humanoid, when first created in 2003. Humanoid robots may be used for the rehabilitation of the people who are affected with autism, who often find communication easier with regimented structure that computer-based programs provide. There are numerous autism centers across the nation and worldwide. So, this will be a handy tool in using at those centers as the affected people can be attracted more towards the robotic movement than the bare teaching. One such robot form aldebaran for kids suffering from Autism is shown in Fig. 9.1.

FIGURE 9.1: Aldebaran's robot for Autism affected kids

9.2 Industrial Applications

Humanoids are used extensively in industries where human-like dexterous jobs are performed, such as automobile factories. This robot can work in a space equivalent to that taken up by a person (Fig. 9.2. Nextage, developed by Kawada Industries, is a human

shaped robot designed to work alongside people. Here, the requirement is that one or more robots can work in mutual coordination as they would in actual factory work. They pick up parts, assemble them, and tighten screws, until they have completed the task at hand. The workbench has markers, which the robots use to position themselves, so even if they're moved, they can start working right away. Hence, the designed robot can be extensively used in industries.

FIGURE 9.2: Nextage robot as industrial player.

9.3 Personal Assistant

The main application is in the medical field where it can used as a personal assistant to the required people. The robot will be able to assist the elderly and visually-challenged people at home using gesture and voice recognition. In addition, it will also be able to locally navigate, cooperate with people using social cues, take instruction from people, recognize location context, recognize classes of objects, manipulate objects and learn from experience. It can manipulate daily objects including doors, dishes and keys, and it will able to help a person get up in case of a fall. Humanoids, thus have a major role to play as a personal assistant.

9.4 Cognitive Platform

CHLOE uses open source hardware called Arduino for its implementation. Hence, it could be used as an open-source hardware platform to educate people about cognitive learning. Owing to its low cost of implementation, anyone can implement CHLOE at home and deploy more complex algorithms on it. Since CHLOE uses the processor of a desktop computer or a laptop for all its computations, really complex algorithms can be developed. It can also be used by universities which perform cognition-related research without incurring high costs.

9.5 Space Exploration

The form factor and overall design of CHLOE is inspired by humans. Hence, with additional elements for dexterity, it may be used in Space exploration as an assistant to Astronauts in their experiments. Such a system would be similar to NASA's Robonaut. Robonaut is a humanoid robotic development project conducted by the Dextrous Robotics Laboratory at NASA's Johnson Space Center (JSC) in Houston, Texas. Robonaut differs from other current space-faring robots in that, while most current space robotic systems (such as robotic arms, cranes and exploration rovers) are designed to move large objects, Robonaut's tasks require more dexterity.

The core idea behind the Robonaut series is to have a humanoid machine work alongside astronauts. Its form factor and dexterity are designed such that Robonaut can use space tools and work in similar environments suited to astronauts. With a lot of work, CHLOE could also be deployed in such fields.

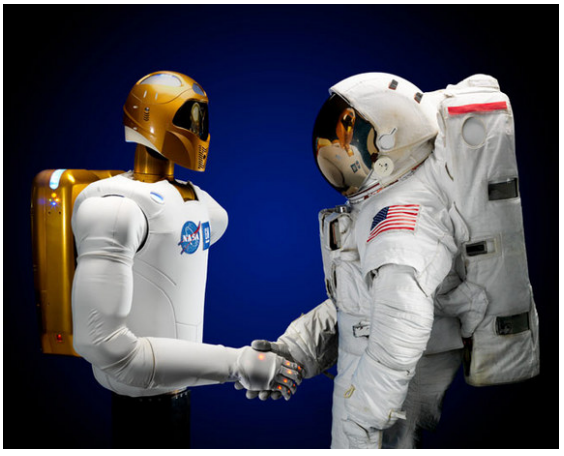


FIGURE 9.3: NASA's Robonaut, use of cognitive humanoids in space exploration

Chapter 10

Scope and Future work

The robot proposed is capable of recognizing basic spoken instructions, intensity, tone of sound and linking the meaning to attentional states it perceives from its sense of sight. Individual components are evaluated by comparisons to human performance on similar tasks, and the complete system is evaluated in the context of a basic social learning AI mechanism that allows the robot to respond to observed stimuli. Objects of interest are tracked in real-time to give the robot some perception. Further advantage of the designed system is that it can communicate with other parts of the world as it is connected to online continuously, such as, interacting with other robots in real time, etc. Also, the cognitive framework devised is extremely versatile, and supports addition of extra modules without disturbing the integrity of the existing ones.

10.1 Stereo Vision

Stereoscopic Vision system is based on the human visual system. Here, two cameras, perfectly matched and horizontally aligned, are used to create a depth map of the field of view. The depth map is calculated using the horizontal disparity between the left and right images. This depth map can be used to perform functions like picking up objects. This is one application that we aimed to implement, while first planning the specifics of the project. But, we were unable to, owing to the cost of stereo cameras (Such as Bumblebee) and complexity of usage.

10.2 Emotional Intelligence

The ability to emote is part of the human character, one element of what it means to be human. Some of us have an acute ability to feel and express emotion; others, such as those on the autism spectrum, have less. We watch each other closely to read emotional clues and react in response. This is an extremely challenging topic, to understand and generate emotions. It requires a certain element of behavioral learning and psychology, but is doable. A necessary pre-requisite for this is presence of a lips and eyelids. A combination of expressions of these can be used to generate emotions.

10.3 Haptics

Haptics is any form of nonverbal communication involving touch. It is a tactile feedback technology which recreates the sense of touch by applying forces, vibrations, or motions to the user. This mechanical stimulation can be used to assist in the creation of virtual objects in a computer simulation, to control such virtual objects, and to enhance the remote control of machines and devices (tele-robotics). It has been described as "doing for the sense of touch what computer graphics does for vision". Haptic devices may incorporate tactile sensors that measure forces exerted by the user on the interface. Haptic devices can be used for the robot to have touch as a stimulus.

10.4 Auto Calculation of servo values

In the present version of CHLOE, we have 16 regions and, for each region, the values to be given as input to each servo is converted to a Look Up Table (LUT) using the trial and error method. The input to the servo is the angle at which the servo is required to rest. This method proves to be cumbersome if the FOV is divided into a large number of regions. A viable alternative is to model the joints of the arm in the form of *kinematic relations*, and try to deduce the servo values by solving the simultaneous equations for a particular point in the FOV. This technique would introduce unmatched resolution to the arm, since it can decide the required servo values by itself to move to a particular point in the FOV.

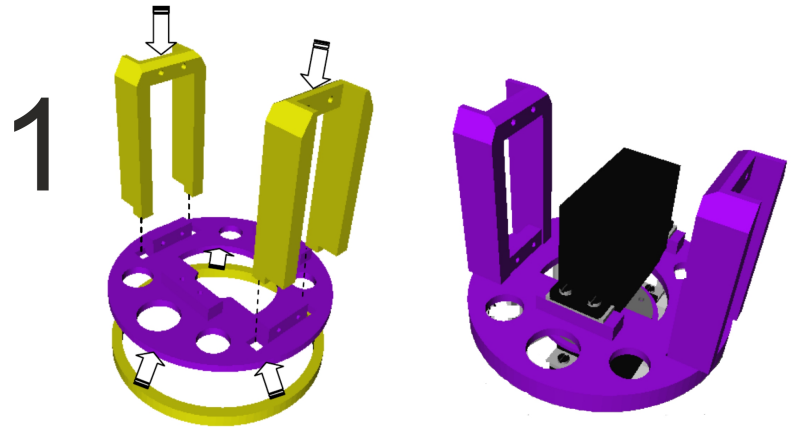
10.5 Smoothing of movements

The smoothness of movements is determined by the rate at which the servo turns. The current work uses a linear speed control mechanism to control the speed which causes abrupt acceleration and deceleration at the ends. To behave more human like it is required for the humanoid to move gracefully. One way of achieving this is by using a PID control for determining the step level. By tuning the PID constants precisely we can produce such graceful motions.

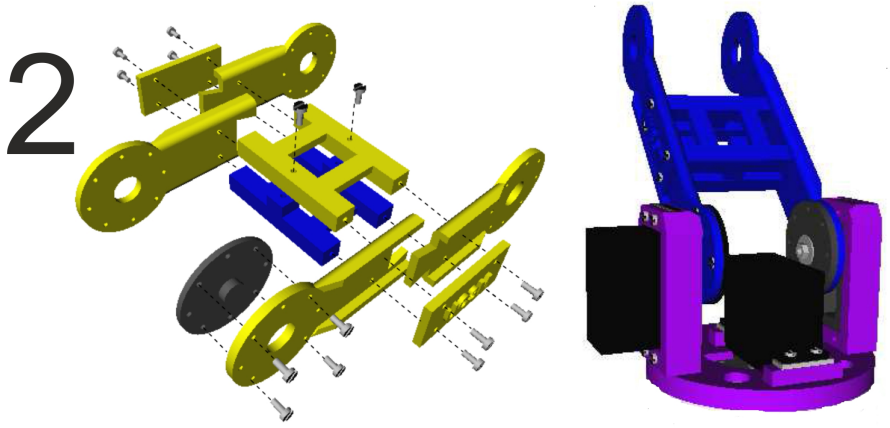
Appendix A

Assembling the Arm

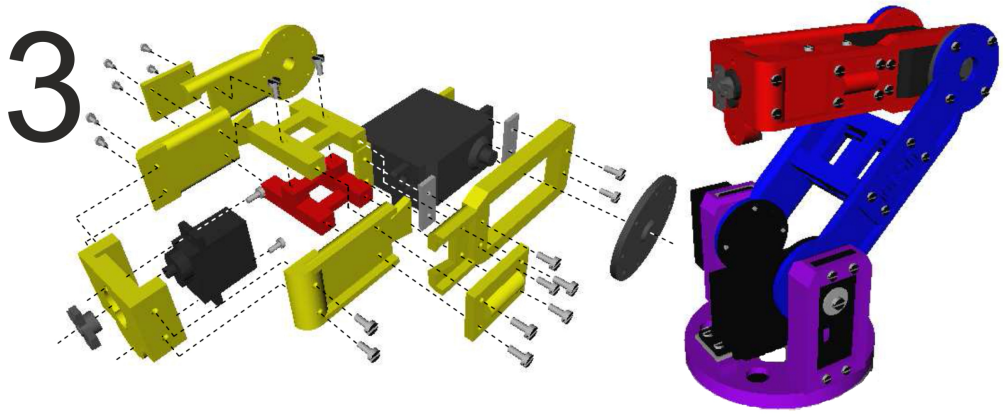
Assembly of Shoulder. No.of subparts=4, No. of Servos=2.



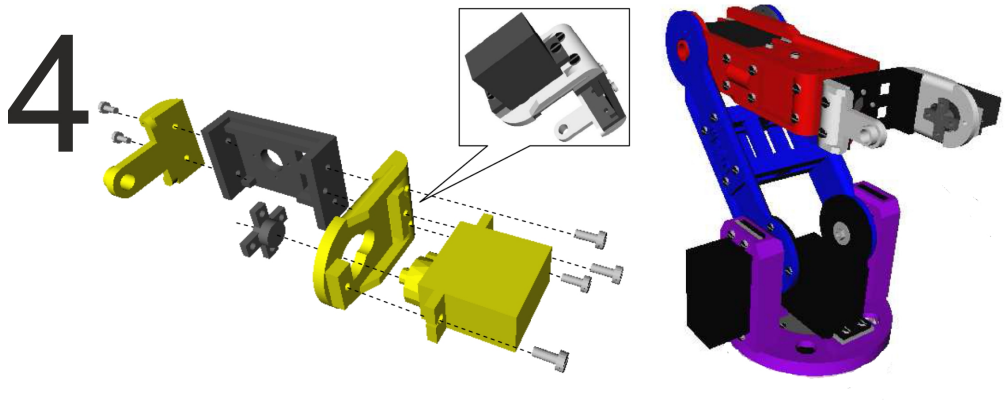
Assembly of arm. No.of subparts=8, No. of Servos=0.

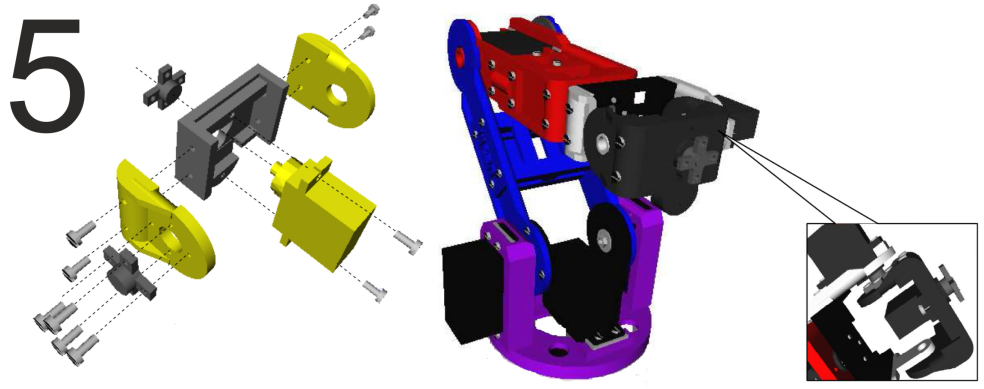


Assembly of forearm. No.of subparts=9, No. of Servos=2.

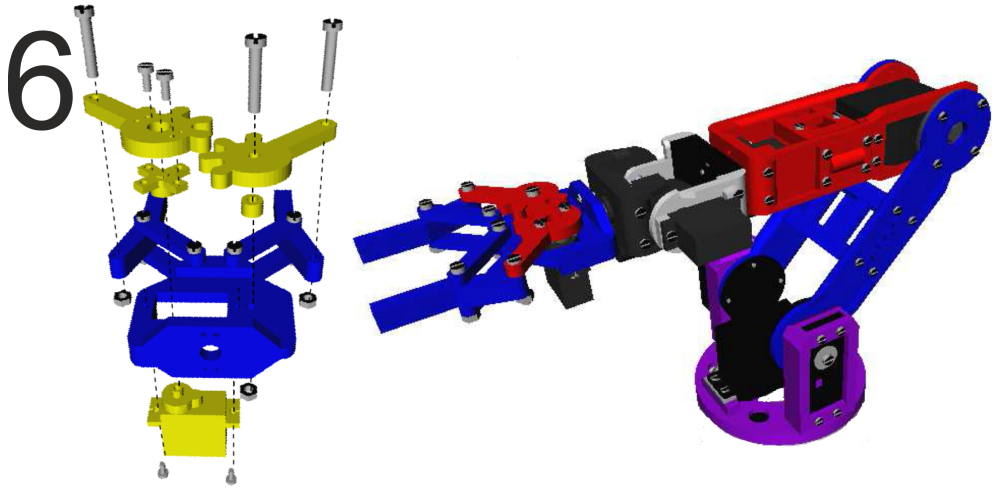


Assembly of wrist. No.of subparts=6, No. of Servos=2.





Assembly of hand. No.of subparts=11, No. of Servos=1



Appendix B

Servo Torque Ratings

The torque rating required for the servos shown in Fig. B.1 can be calculated as follows:

Torque about joint 1:

$$M1 = \frac{L1}{2} \times W1 + L1 \times W4 + \left(L1 + \frac{L2}{2}\right) \times W2 + (L1 + L3) \times W3$$

Torque about joint 2:

$$M2 = \frac{L2}{2} \times W2 + L3 \times W3$$

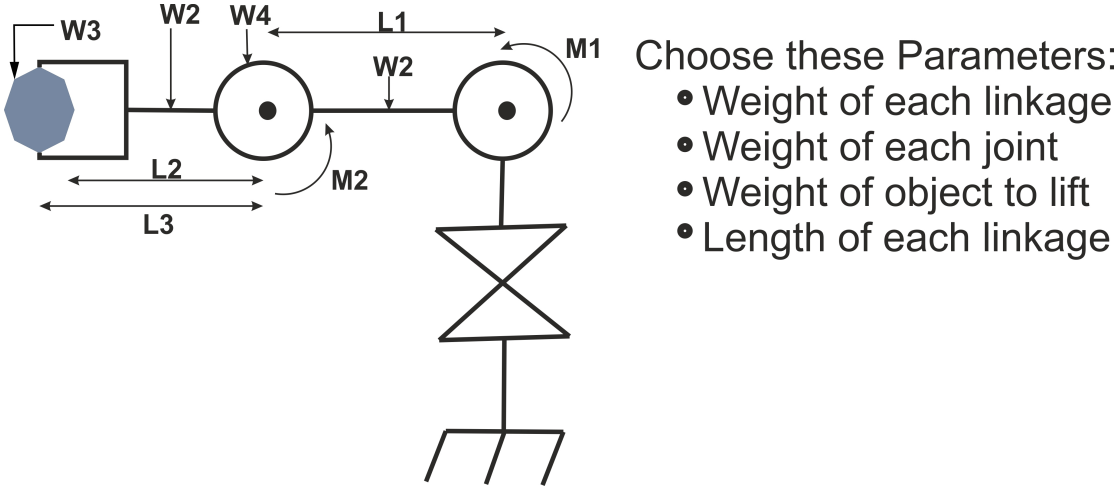


FIGURE B.1: Determination of servo torque ratings.

Appendix C

Schematics for Interfacing Arduino, TLC5940 and servos

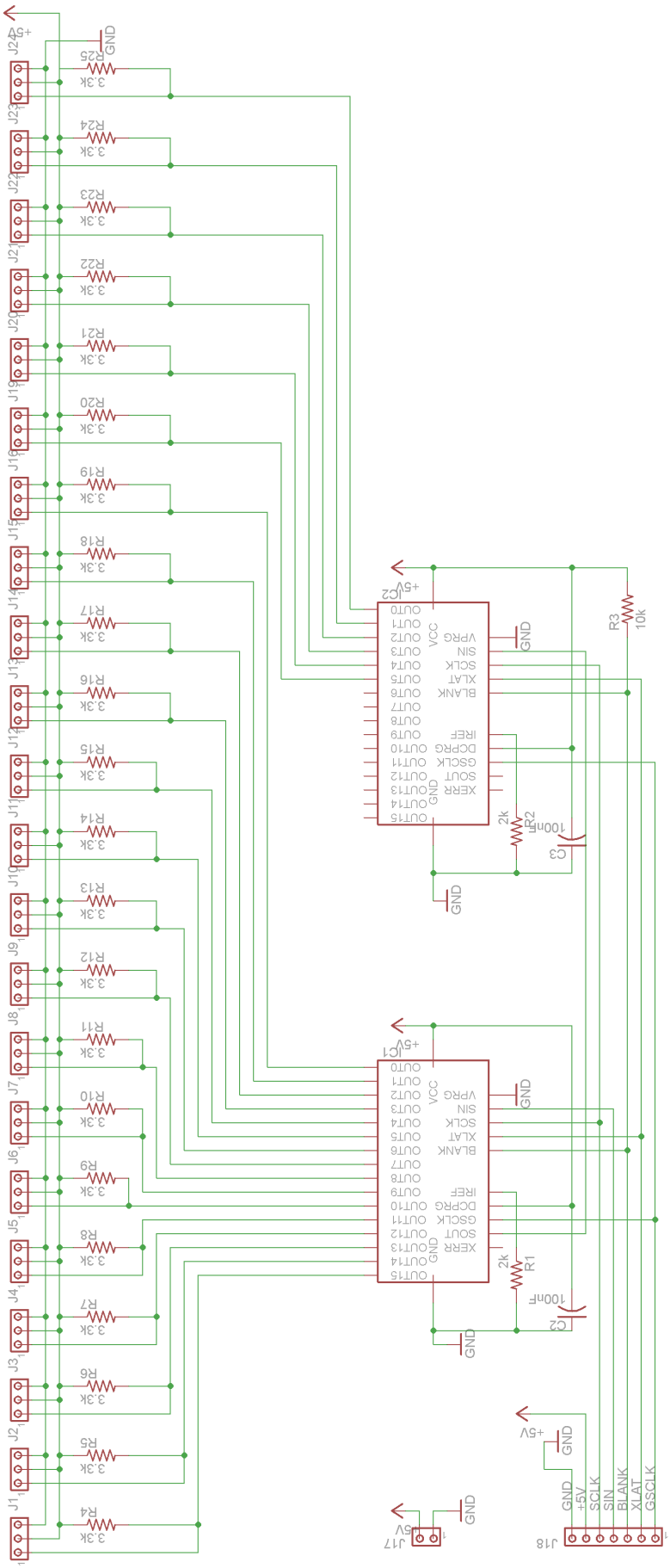


FIGURE C.1: Schematics to interface TLC5940 to Servos.

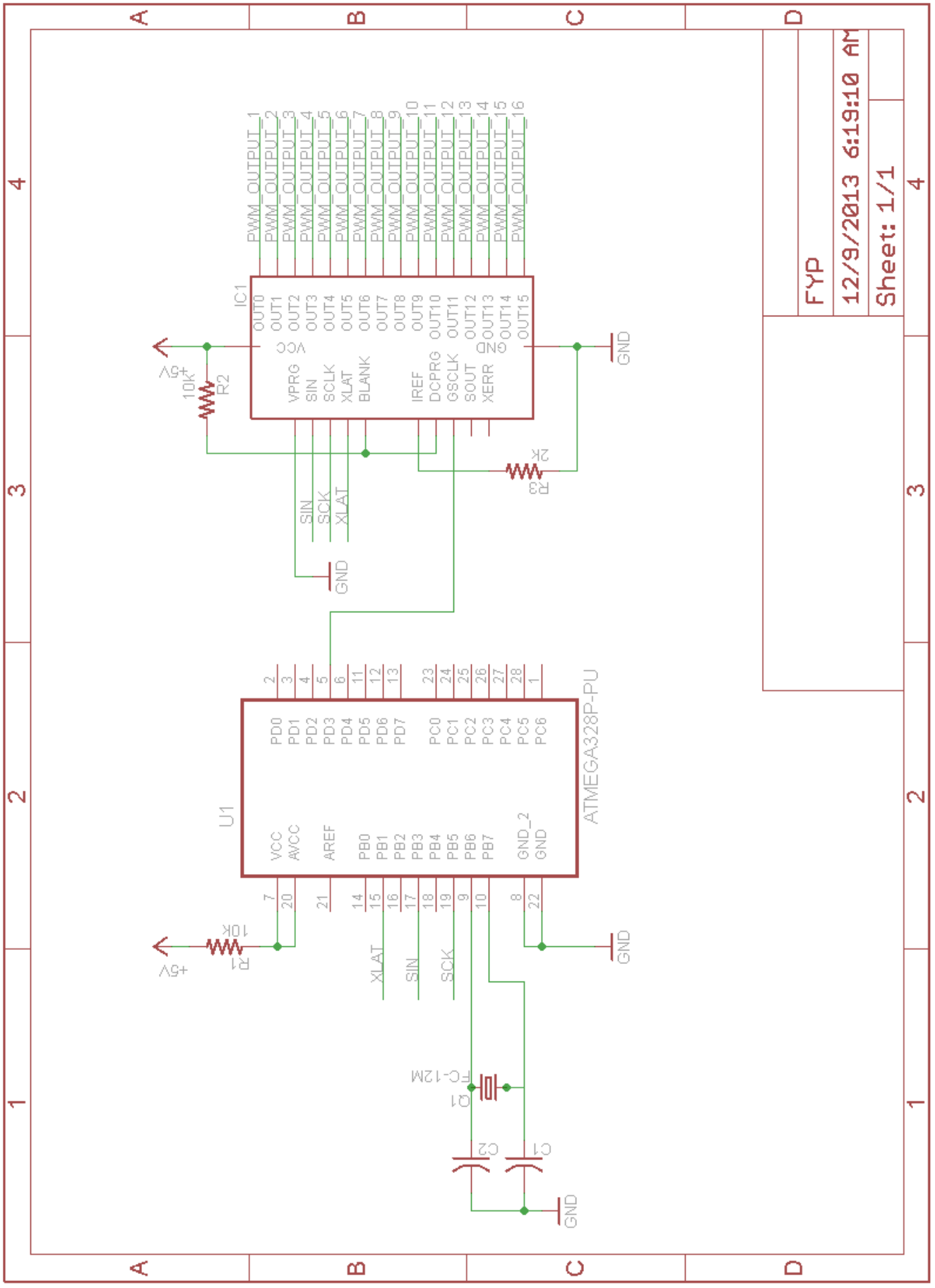


Figure C.2: Schematics to interface TLC5940 to Arduino UNO.

Appendix D

Servo look up tables

As explained previously, servos take values between 0 and 180 as input, ideally. But practical ranges any be different. So we constrain the max and min positions of each servo, to serve a dual purpose: Ensure values remain within range, and to limit motion so as to make movements as human as possible (Table 8.1).

For each of the 9 regions of the trapezoidal field-of-view (FOV) shown in Chapter 6, the values to be given to each servo have to be determined by trial and error, and then stored in the form of a look-up-table (LUT). Table D.2 contains the LUT used in CHLOE's working. The terms base, shoulder, elbow and wrist refer to the corresponding servos used in the arm. The term home refers to the servo values required to maintain the arm at its reference position of minimum stress.

Table D.1: Max, min and default values for all servos.

Servo Channel	Maximum	Minimum	Default	Joint
0	140	50	90	Eye Vertical
2	120	60	90	Eye Horizontal
4	160	45	75	Neck
5	130	70	103	Neck
6	150	0	0	Right Shoulder
7	180	70	140	Right Shoulder
8	170	70	110	Left Elbow
9	140	0	70	Left Wrist
10	180	0	150	Left Wrist
11	30	0	0	Left Gripper
12	180	0	180	Left shoulder
13	170	0	170	Left shoulder
14	160	65	80	Right Elbow
15	170	0	90	Right Wrist
16	120	0	155	Right Wrist
17	140	0	0	Right Gripper

Table D.2: Arm motion Look-up table for object recognition, for 16 divisions of FOV.

Position	Ch 6	Ch 7	Ch 8	Ch 9	Ch 10	Ch 11	Ch 12	Ch 13
1	0	140	110	70	150	0	180	170
2	0	175	110	70	150	20	180	170
3	0	140	140	70	160	0	180	170
4	0	130	140	70	150	0	180	170
5	0	140	110	70	150	50	180	180
6	10	180	110	80	165	0	180	170
7	10	180	90	80	150	0	180	170
8	10	160	90	70	150	0	180	170
9	0	140	110	70	150	0	160	180
10	20	180	100	70	150	0	180	170
11	20	180	85	70	150	80	180	170
12	20	160	85	60	170	0	180	170
13	0	140	110	70	150	0	150	180
14	30	180	100	70	150	0	180	170
15	30	180	90	80	150	0	180	170
16	30	160	70	80	150	0	180	170

Appendix E

2D projections of Head Structure

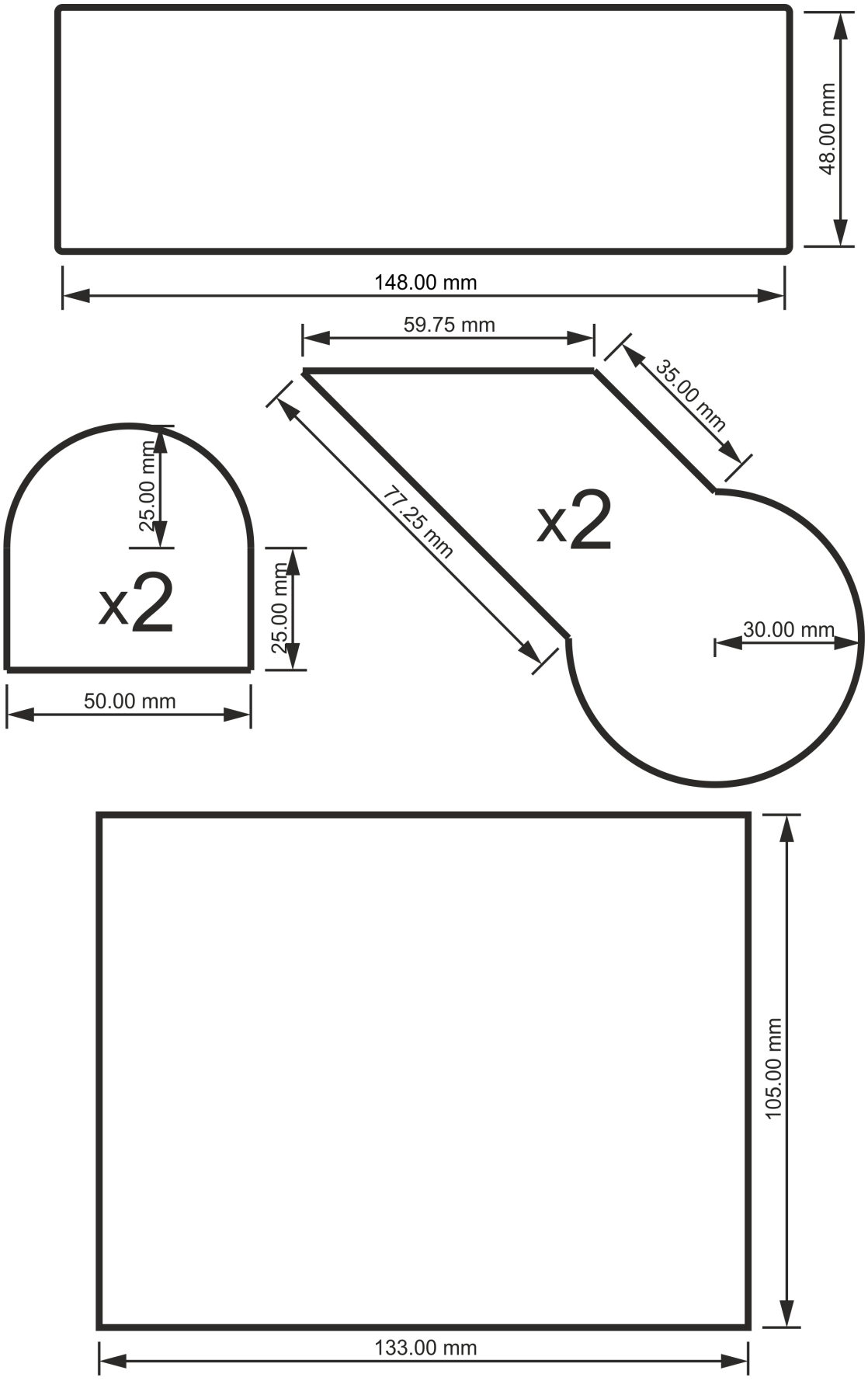


FIGURE E.1: 2D projections of the head structure- parts of the Head.

Appendix F

TLC5940 datasheets

16 CHANNEL LED DRIVER WITH DOT CORRECTION AND GRAYSCALE PWM CONTROL

FEATURES

- **16 Channels**
- **12 bit (4096 Steps) Grayscale PWM Control**
- **Dot Correction**
  - **6 bit (64 Steps)**
  - **Storable in Integrated EEPROM**
- **Drive Capability (Constant-Current Sink)**
  - **0 mA to 60 mA (V_{CC} < 3.6 V)**
  - **0 mA to 120 mA (V_{CC} > 3.6 V)**
- **LED Power Supply Voltage up to 17 V**
- **V_{CC} = 3 V to 5.5 V**
- **Serial Data Interface**
- **Controlled In-Rush Current**
- **30MHz Data Transfer Rate**
- **CMOS Level I/O**
- **Error Information**
  - **LOD: LED Open Detection**
  - **TEF: Thermal Error Flag**

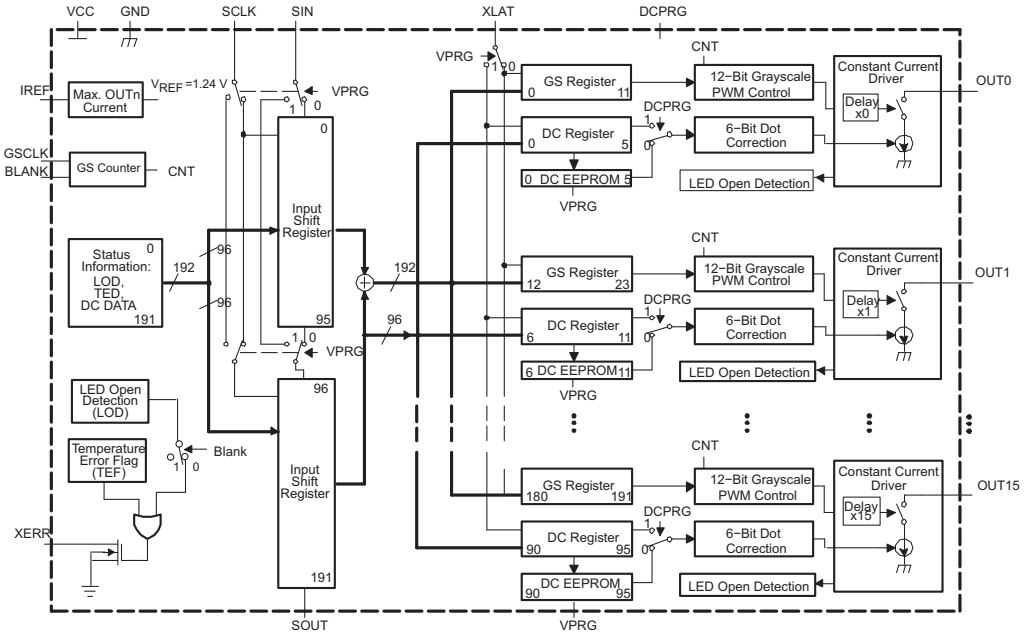
APPLICATIONS

- **Monocolor, Multicolor, Full-Color LED Displays**
- **LED Signboards**
- **Display Backlighting**
- **General, High-Current LED Drive**

DESCRIPTION

The TLC5940 is a 16-channel, constant-current sink LED driver. Each channel has an individually adjustable 4096-step grayscale PWM brightness control and a 64-step, constant-current sink (dot correction). The dot correction adjusts the brightness variations between LED channels and other LED drivers. The dot correction data is stored in an integrated EEPROM. Both grayscale control and dot correction are accessible via a serial interface. A single external resistor sets the maximum current value of all 16 channels.

The TLC5940 features two error information circuits. The LED open detection (LOD) indicates a broken or disconnected LED at an output terminal. The thermal error flag (TEF) indicates an overtemperature condition.



Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

PowerPAD is a trademark of Texas Instruments.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of the Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

Copyright © 2004–2007, Texas Instruments Incorporated

PRINCIPLES OF OPERATION

SERIAL INTERFACE

The TLC5940 has a flexible serial interface, which can be connected to microcontrollers or digital signal processors in various ways. Only 3 pins are needed to input data into the device. The rising edge of SCLK signal shifts the data from the SIN pin to the internal register. After all data is clocked in, a high-level pulse of XLAT signal latches the serial data to the internal registers. The internal registers are level-triggered latches of XLAT signal. All data are clocked in with the MSB first. The length of serial data is 96 bit or 192 bit, depending on the programming mode. Grayscale data and dot correction data can be entered during a grayscale cycle. Although new grayscale data can be clocked in during a grayscale cycle, the XLAT signal should only latch the grayscale data at the end of the grayscale cycle. Latching in new grayscale data immediately overwrites the existing grayscale data. Figure 11 shows the timing chart. More than two TLC5940s can be connected in series by connecting an SOUT pin from one device to the SIN pin of the next device. An example of cascading two TLC5940s is shown in Figure 12 and the timing chart is shown in Figure 13. The SOUT pin can also be connected to the controller to receive status information from TLC5940 as shown in Figure 22.

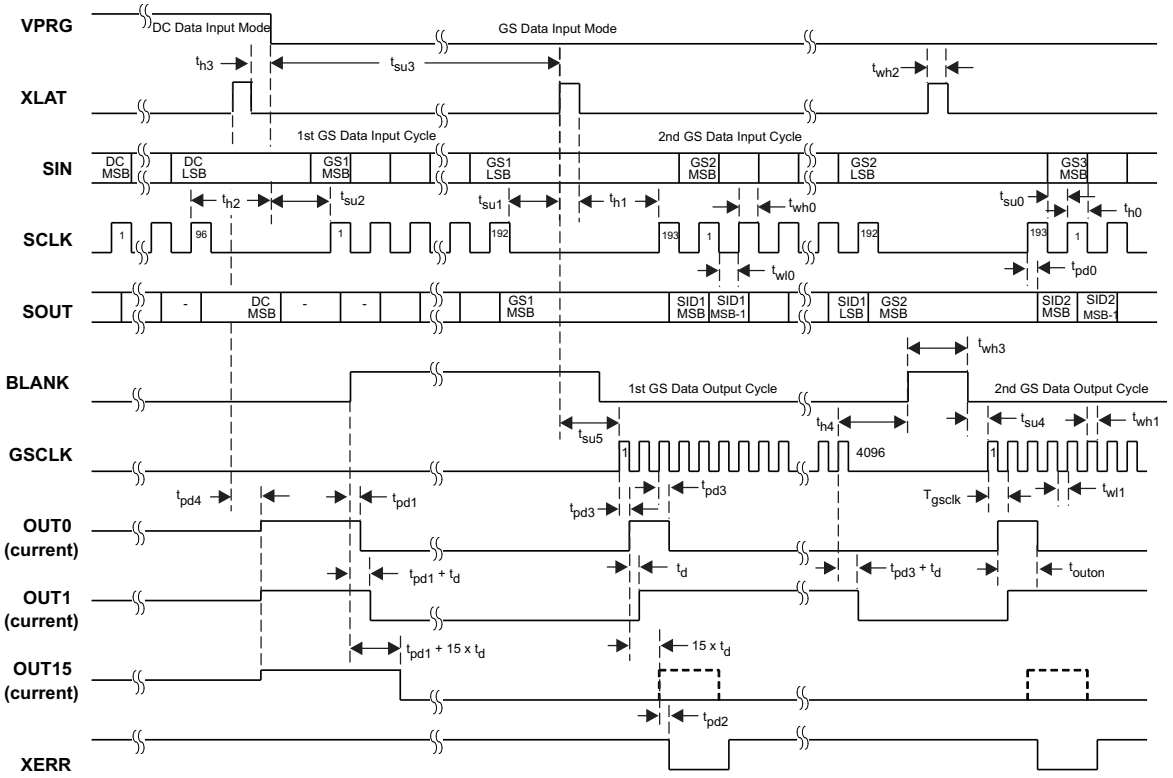


Figure 11. Serial Data Input Timing Chart

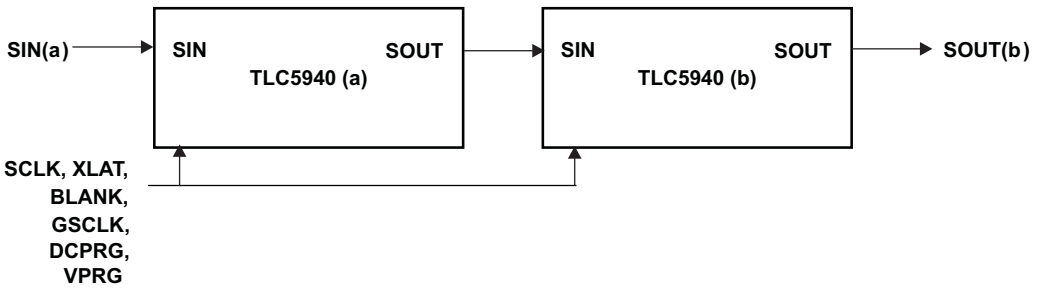


Figure 12. Cascading Two TLC5940 Devices

Copyright © 2004–2007, Texas Instruments Incorporated

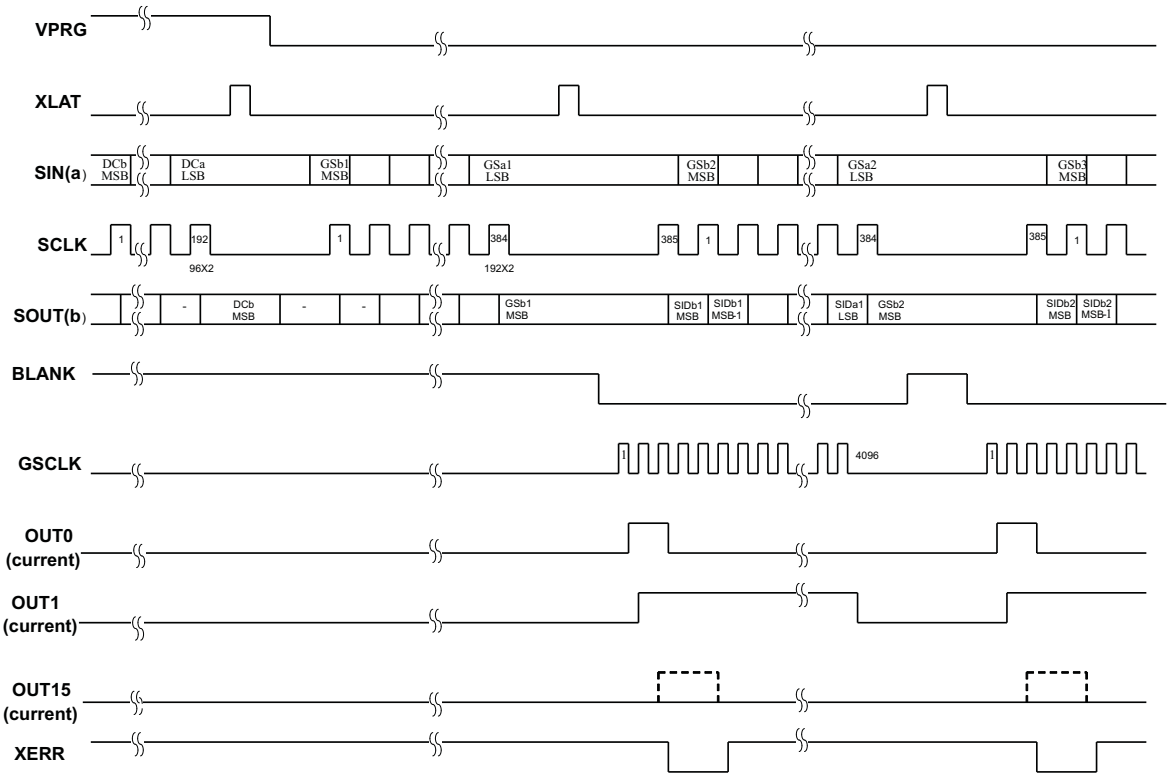


Figure 13. Timing Chart for Two Cascaded TLC5940 Devices

ERROR INFORMATION OUTPUT

The open-drain output XERR is used to report both of the TLC5940 error flags, TEF and LOD. During normal operating conditions, the internal transistor connected to the XERR pin is turned off. The voltage on XERR is pulled up to V_{CC} through an external pullup resistor. If TEF or LOD is detected, the internal transistor is turned on, and XERR is pulled to GND. Since XERR is an open-drain output, multiple ICs can be OR'ed together and pulled up to V_{CC} with a single pullup resistor. This reduces the number of signals needed to report a system error (see Figure 22).

To differentiate LOD and TEF signal from XERR pin, LOD can be masked out with BLANK = HIGH.

Table 2. XERR Truth Table

ERROR CONDITION		ERROR INFORMATION		SIGNALS	
TEMPERATURE	OUTn VOLTAGE	TEF	LOD	BLANK	XERR
$T_J < T_{(TEF)}$	Don't Care	L	X	H	H
$T_J > T_{(TEF)}$	Don't Care	H	X		L
$T_J < T_{(TEF)}$	OUTn > $V_{(LED)}$	L	L	L	H
	OUTn < $V_{(LED)}$	L	H		L
$T_J > T_{(TEF)}$	OUTn > $V_{(LED)}$	H	L		L
	OUTn < $V_{(LED)}$	H	H		L

TEF: THERMAL ERROR FLAG

The TLC5940 provides a temperature error flag (TEF) circuit to indicate an overtemperature condition of the IC. If the junction temperature exceeds the threshold temperature (160C typical), TEF becomes H and XERR pin goes to low level. When the junction temperature becomes lower than the threshold temperature, TEF becomes L and XERR pin becomes high impedance. TEF status can also be read out from the TLC5940 status register.

Copyright © 2004–2007, Texas Instruments Incorporated

Appendix G

Bill of Materials (BOM)

Sl. No.	Item	Amount (INR)	Quantity	Total Amount (INR)
1	Logitech Webcam C270	1300.00	1	1300.00
2	Servo motor with 1Kg Torque	400.00	2	800.00
3	Servo motor with 1.8Kg Torque	300.00	8	2400.00
4	Servo motor with 4Kg Torque	500.00	2	1000.00
5	Servo motor with 6Kg Torque	775.00	4	3100.00
6	Servo motor with 9Kg Torque	990.00	2	1980.00
5	Zebronics 1001SM Microphone	200.00	1	200.00
6	Arduino Uno	1800.00	1	1800.00
7	TLC5940	100.00	2	200.00
8	3D printing material: 1Kg PLA spool	1500.00	1	1500.00
9	Metal Fabrication (Torso)	600	1	600
10	450W SMPS	500	1	500
11	Foam Board	250	1	250
10	Miscellaneous hardware	600	-	600
11	Miscellaneous Electronics	300	-	300
12	Aesthetics	500	-	500
			Total	17030

Bibliography

[1] Vernon, David, Giorgio Metta, and Giulio Sandini. "A survey of artificial cognitive systems: Implications for the autonomous development of mental capabilities in computational agents." IEEE Transactions on Evolutionary Computation, vol. 11, no. 2, (2007): 151–180.

[2] Ballard, Dana H. "Generalizing the Hough transform to detect arbitrary shapes." Pattern recognition vol. 13, no. 2 (1981): 111-122.

[3] Lowe, David G. "Object recognition from local scale-invariant features." 1999. The proceedings of the seventh IEEE international conference on Computer vision, Vol. 2. IEEE, 1999.

[4] Marius Muja and David G. Lowe. "Scalable Nearest Neighbor Algorithms for High Dimensional Data". Pattern Analysis and Machine Intelligence, Vol. 36, 2014.

[5] Viola, Paul, and Michael J. Jones. "Robust real-time face detection." International journal of Computer Vision vol. 57, no. 2 (2004): 137–154.

[6] Turk, Matthew, and Alex Pentland. "Eigenfaces for recognition." Journal of cognitive neuroscience vol. 3, no. 1, (1991): 71–86.

[7] Baron, Robert J. "Mechanisms of human facial recognition." International Journal of Man-Machine Studies vol. 15, no. 2 (1981): 137–178.

[8] https://dvcs.w3.org/hg/speech-api/raw-file/tip/speechapi.html,Glen Shir es & Hans Wennborg, Google Inc., 19 October 2012, W3C

[9] Chromium Development Group: https://groups.google.com/a/chromium.org/forum/#!forum/chromium-dev

[10] W. T. Dempster and G. R. Gaughran, "Properties of body segments based on size and weight," American Journal of Anatomy, vol. 120, no. 1, pp. 33–54, 1967. [section-1.2]

[11] H. G. Armstrong, Anthropometry and mass distribution for human analogues," Military male aviators, vol. 1, 1988. [section-1.2]