TECHNICAL UNIVERSITY OF CRETE
SCHOOL OF ELECTRONIC & COMPUTER ENGINEERING

Olfaction as a Digital Sense: Implementation of Real Time Video Capture & Processing System for Insect Gathering Detection Stimulated by Olfaction in Restricted Environment

by

Alkiviades Klotsonis

Thesis

Submitted in partial fulfillment of the

Requirements for the acquisition of diploma degree of

Electronic & Computer Engineering

May 2015

THESIS COMMITTEE

Professor Kostas Kalaitzakis, *Thesis Supervisor*

Professor Michalis Zervakis

Assistant Professor Vasilis Samoladas

Dedication

I would like to dedicate this to my family. My beautiful mother who taught me how to live a healthy life full of potential, my fastidious father who made me smart, strong and rational, my little brother who has always been my other half and my extraordinary grandmother who kept me motivated, always reminding me how long life can be and how human brain is evolving over time. I would also like to dedicate it to my best friend who always believed in me and was always there at the hard times, to the first love of my life who had been supportive for a long period in the effort of the acquisition of this degree and last but not least to my beautiful aunt, that if it wasn't for her I might had never made it!

Acknowledgements

I would like to thank my parents for the financial support through all of my studies. My supervisor professor for believing in my idea; giving me the opportunity to work on it and keeping me motivated and organized.

The Technical University of Crete as I got the most I could of this institution and specially the department of Electronic & Computer Engineering which finally made me a worth-standing scientist and most of all human. It might took me long to appreciate our department's teaching methodology while our professors let us sometimes be self-educated and "helpless" but at the end of the day this is what makes a graduate student of such a departments being able to claim that he/she follows a technological field that is evolving every second.

I am also grateful to Mr. Christodoulakis; a professor that I followed closely during my studies despite that I could never even imagine my self being able to handle a few of what he does daily. After completing his course of "Human Computer Interaction" I have to say that I got closer to the meaning of being an Engineer in general.

I should also not forget about Mr. Sidiropoulos who was the first of my professors that actually taught me that there are mathematical minds that you will never overtake. Funny but I will always carry the knowledge I got from the last course he gave us before he left the institution. I still step up into problems in my everyday life that I wouldn't be able to find the optimum solution if I didn't knew about the "Theory of Optimization".

Last but not least my greetings to Mr. Pantinakis the professor that most students do not like but I do. He might not remember our disagreement but I still have to say to him that by a cosmologist's point of view you cannot ever be damaged by kinetic energy!

Abstract

Olfaction as a Digital Sense: Implementation of Real Time Video Capture & Processing System for Insect Gathering Detection Stimulated by Olfaction in Restricted Environment

Alkiviades Klotsonis, E.C.E.

Technical University of Crete, 2015

Supervisor: Konstantinos Kalaitzakis

The following thesis is organized in 4 chapters:

1st Chapter discusses about the sense of olfaction. The way this sense is conceived by the human brain is analyzed and the measurement conventions about it are recorded. Then a brief study of olfaction mechanisms that are met in vertebrates, fish, insect and the nematode olfactory system are presented. At the end there is a reference on how the animal's olfaction have been useful to our life and a comparison of their olfactory systems.

2nd Chapter introduces the technological progress of artificial olfaction. First, the electronic nose sensors are studied and their applications are presented. Next we analyze the concept and development of the bio-electronic nose. Finally we discuss about bioengineering methods that implement the artificial olfaction metaphor and conclude on the future trends.

3rd Chapter presents the designing process of the system that is described at the tittle. The setup requirements are discussed and the olfactometer is introduced. Moreover the algorithm is designed, implemented in Matlab and tested. A brief study on image binarization techniques is presented and finally there is a research about the optimal hardware for the system deployment.

4th Chapter implements the system. Additionally it contains a getting started guide for BeagleBone Black. All the necessary for our system implementation software, drivers and libraries are explained into detail. The chapter also contains an educational cross compiling section with tips on how to cross compile and remote debug for BeagleBone Black. Finally the results of the experiments are presented followed by a discussion on the conclusions.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

CEN EN	European Committee of Standardization
OU_E	European Odor Unit
ODT	Odor Detection Threshold
VNO	VomeroNasal Organ
ORs	Odorant Receptors
OSNs	Olfactory Sensory Neurons
OE	Olfactory Epithelium
OB	Olfactory Bulb
TAARs	Trace Amin-Associated Receptors
ORNs	Olfactory Receptor Neurons
ORCO	Odorant Receptor CO-receptor
IRs	Ionotropic glutamate Receptors
MOx	Metal Oxide
EAG	Electroantennogram
EMG	Electromyographic
BBB	BeagleBone Black
V4L	Video for Linux
OpenCV	Open Source Computer Vision Library
SSH	Secure Shell
FTP	File Transfer Protocol
IDE	Integrated Developing Environment

Chapter 1
A Brief Olfaction Study

Abstract

Olfaction is the most complex and less understood human sense. Although humans do not have a lot olfactory receptors compared to many species of the animal kingdom, which makes our noses insensitive, we are still the main subject of qualitative odor measurement. Early in this chapter we will discuss about the human sense of smell and odor measurement. On the other hand, molecular mechanisms of olfaction have been intensively studied in the last quarter century and it was found that receptors by which olfactory stimuli are detected are vastly different between different animal species and even between different olfactory organs of the same species. This chapter includes a general description of the anatomy of the mammalian olfactory system. Also contains a brief comparison of the fish, insect and nematode olfactory receptors capability.

1.1 Olfaction as a Human Sense

We have five senses: vision, hearing, touch, smell, and taste. Although some others including balance, pain, itch, and temperature senses have been reported, the first five remain the major classical senses. Among those, three of them: vision, hearing, and touch, recognize physical stimuli, and the other two senses recognize chemical stimuli. The science and technology concerned with vision and hearing have been advanced enormously, while the understanding of the chemical senses, especially the sense of smell, has been very limited.

1.1.1 Human Senses at the Digital World

The stimulus energy for the sense of vision and hearing is light and sound, respectively. The camera creates recorded signals that can be delivered to remote places. This technology allows us to transfer audio and video so we can watch a football game at home. The sense of touch has been also integrated with information technology in the form of the tablet PC as well as augmented reality sometimes. On the contrary, there is no such a device, which can capture smell or taste. The sense of smell is even more complicate and mysterious than the sense of taste. Electronic noses have been intended to mimic the signal processing of the sense of smell; however, elemental receptor materials of the conventional electronic noses are totally different from human olfactory receptors. If we consider that the sense of smell is a chemical sense, the same receptor materials as those in the human nose should be employed to accurately realize the human sense of smell.

In the last two decades, much has been learned about the smell sensing mechanism in biological systems. With knowledge about the biological olfactory system and the techniques for the expression of biological receptor proteins, we are able to utilize biological materials and systems to mimic the biological olfactory system. In addition to the advances in biological and biotechnological area, nanotechnology has progressed to a great degree. Creating a device that has a similar function to the human smell sensing system can be realized by combining the olfactory cells or receptors with nanotechnology.

1.1.2 Olfaction Modeling

A parallel investigation to replicating the human nose would be the digitization of olfaction. In order to reach the Digital Odor, we need to create a model to describe it and reproduce any known to human odor today respectively to the digital image or sound. This is the road down the place where physics meet chemistry or science leads to a principle theory as Einstein had imagined. A relative chemical approach is done at [1] where odors are categorized to simple and complex.

In short, a *simple* odor is usually a small, polar molecule that can become airborne, enter the nasal cavity and then be sensed by the mammalian olfactory system. There are many different functional groups found in odorous compounds; some of which are listed below in Table 1.1.

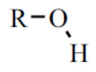
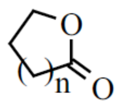
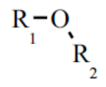
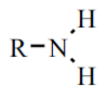
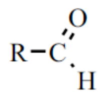
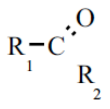
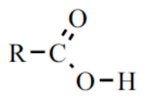
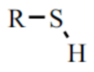
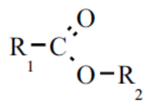
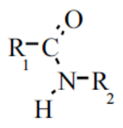
Functional group	Structure	Functional group	Structure
Alcohol		Lactone	
Ether		Amine	
Aldehyde		Nitrile	$R-C\equiv N$
Ketone		Isonitrile	$R-N\equiv C$
Acid		Thiol	
Ester		Amide	

Table 1.1: Odorous Compounds (taken from [1] page 13)

A *complex* odor (as opposed to a simple odor) is a collection of two or more different volatile chemical compounds that produces a smell. The number of different compounds in an odor can vary from just a few to several thousand – each with a different concentration and each with a different detection threshold for the human nose. The resultant smell may have various olfactory notes within it, such as green, floral, citrus etc. The problem is further complicated by the fact that the detection threshold of a chemical compound in a complex odor can differ from that when it is on its own, i.e. as a simple odor. For example, compound A and B may not smell as individuals but when mixed together they do elicit an olfactory response. Thus, the structure-activity relationship may be straightforward and exist for simple odors but can be highly non-linear in mixtures. This makes it very difficult to understand and model the mechanisms by which the human olfactory system works and thus to develop an electronic metaphor.

1.2 Odor Conception & Measurement Conventions

Odor measurement even without a complete model of representation can be achieved based on factors that are deduced from the conception of odor by the human brain. Defining a process and making the necessary conventions can lead to quantity and quality units that describe an odor [2]. Measurement is essential for odor regulation and control in nowadays applications. An odor emission often consists of a complex mixture of many odorous compounds. Analytical monitoring of individual chemical compounds present in such an odor is usually not practical. As a result, odor sensory methods, instead of instrumental methods, are normally used to measure complex odors. Odor sensory methods are available to monitor odor both from source emissions and in the ambient air. Those two diverse circumstances require different approaches.

Field measurement with portable Olfactometers seems more effective, but the use of “Field Olfactometers” is not regulated in Europe so far, while it is popular in the U.S. and Canada, where several States set limits at the receptor sites or along the perimeter of odor emitting plants, expressed in units of dilution to threshold (D/T) [3]. Different aspects of odor can be measured through a number of quantitative methods, such as assessing concentration or apparent intensity. Initial entry into a room provides the most accurate sensing of smell, before habituation begins to change the perception of an odor. Sensation of odor has 4 properties related to threshold and tolerance. Those are Concentration, Intensity, Quality and Hedonic Tone.

1.2.1 Odor Detection Threshold

The odor detection threshold (ODT) is the lowest concentration of a certain odor compound that is perceivable by the human sense of smell. The threshold of a chemical compound is determined in part by its shape, polarity, partial charges and molecular mass. The olfactory mechanisms responsible for a compound's different detection threshold is not well understood, as such, these thresholds cannot yet be accurately predicted. Rather, they must be measured through extensive tests using human subjects in laboratory settings. The ODT is the concentration of an odor in air when 50% of a population can distinguish between the odorous sample and an odor free blank. The recognition threshold is the concentration of an odor in air in which 50% of a population can discern from an odorous sample and odor free blank. The recognition odor threshold is usually a factor of 2 to 5 times higher than the ODT [9]. Odor concentration is an odor's pervasiveness. To measure odor sensation, an odor is diluted to certain amounts to reach a detection or recognition threshold.

1.2.2 Measuring Concentration

The measurement of odor concentration is the most widespread method to quantify odors. It is standardized in CEN EN 13725:2003 [4]. The method is based on dilution of an odor sample to the ODT. The numerical value of the odor concentration is equal to the dilution factor that is necessary to reach the ODT. Its unit is the *European Odor Unit*, OU_E. Therefore, the odor concentration at the ODT is 1 OU_E by definition.

To establish the odor concentration, an olfactometer is used which employs a group of panelists. A diluted odorous mixture and an odor-free gas (as a reference) are presented from sniffing ports to a group of panelists. In comparing the odor emitted from each port, the panelists are asked to report if they can detect a difference between the ports. The gas-diluting ratio is then decreased by a factor of 1.4 or two (i.e., the concentration is increased accordingly). The panelists are asked to repeat their judgment. This continues until the panelists respond certain and correct twice in a row. These responses are used to calculate the concentration of the odor in terms of OU_E/m^3.

To complete such a task a few additional factors should be taken into account. First, the test persons must fulfill certain requirements, for example regarding their sensitivity of odor perception. The main panel calibration gas to verify this requirement used is n-Butanol (as 1 $OU_E/m^3 \equiv 40$ ppb/v n-butanol) [5]. Second, to collect an odor sample, the samples must be collected using specialized sample bags, which are made from an odor free material e.g. Teflon. The most accepted technique for collecting odor samples is the lung technique, where the sample bag is placed in a sealed drum, and a vacuum is placed on the drum, which fills the sample bag as the bag expands, and draws the sample from the source into the bag. Critically, all components that touch the odor sample must be odor free, which includes sample lines and fittings. Also a human's odor detection threshold is variable. Repeated exposure to an odorant leads to enhanced olfactory sensitivity and decreased ODT for a number of different odorants [6]. It was found in a study that humans that were completely unable to detect the odor of androstenone developed the ability to detect it after repeated exposure [7]. Another study showed that humans could discriminate between two odorants that differ in concentration by as little as 7% [8].

1.2.3 ODOR INTENSITY

Odor intensity is the perceived strength of odor sensation. This intensity property is used to locate the source of odors and perhaps most directly related to odor nuisance [9]. Perceived strength of the odor sensation is measured in conjunction with odor concentration. This can be modeled by the Weber-Fechner law:

$$I = a * \log(c) + b$$

Where:

- I is the perceived psychological intensity at the dilution step on the butanol scale.
- a is the Weber-Fechner coefficient, c is the chemical concentrations.
- and b is the intercept constant (0.5 by definition).

Odor intensity can be expressed using an odor intensity scale, by assigning a numerical value that corresponds to a verbal description in order to describe an odor sensation [10].

Odor intensity can be divided into the following categories according to intensity:

0 – no odor

1 – very weak (odor threshold)

2 – weak

3 – distinct

4 – strong

5 – very strong

6 – intolerable

This method is applied by in the laboratory and is done so by a series of suitably trained panelists/observers who have been trained to appropriately define intensity.

1.2.4 Hedonic Tone Assessment

Hedonic assessment is the process of scaling odors on a scale ranging from extremely unpleasant via neutral up to extremely pleasant. It is important to note that intensity and hedonic tone, whilst similar, refer to different things. That is, the strength of the odor (intensity) and the pleasantness of an odor (hedonic tone). Moreover, it is important to note that perception of an odor may change from pleasant to unpleasant with increasing concentration, intensity, time, frequency, and previous experience with a specific odor (correlated memories); all factors determining a response [9].

The overall set of qualities is sometimes identified as the "*FIDOL factors*", (short for **F**requency, **I**ntensity, **D**uration, **O**ffensiveness and **L**ocation) [11].

1.2.5 Odor Character

The character of an odor is a critical element in assessing an odor. This property is the ability to distinguish different odors and is only descriptive. First a basic description is used such as sweet, pungent, acrid, fragrant, warm, dry, or sour. The odor is then referenced to a source such as sewage or apple, which can then be followed by a reference to a specific chemical such as acids or gasoline [9].

Most commonly, a set of standard descriptors is used, which may range from fragrant to sewer odor [12]. Although the method is fairly simplistic, it is important for the FIDOL factors to be understood by the person recording the character. This method is most commonly used to define the character of an odor that can then be compared to other odors. It is common for olfactometry laboratories to report character as an additional factor post sample analysis.

Odor character after all is a very objective factor. Although some descriptions remain the same at most applications where odor characterization is necessary, at most cases odor descriptors vary. Table 1.2 below, taken from [13], show another odor characterization proposal together with their corresponding chemical structure.

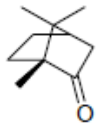
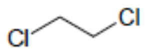
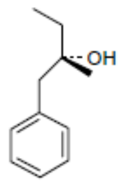
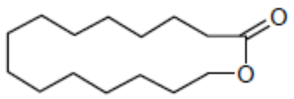
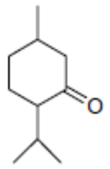
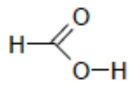
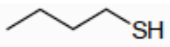
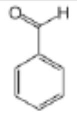
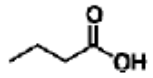
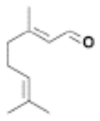
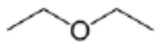
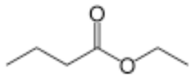
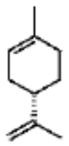
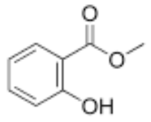
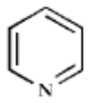
Primary aromas	Example compound	Chemical structure	Common source
Camphoraceous	camphor		mothballs
Ethereal	ethylene dichloride		dry cleaning fluid
Floral	phenylethyl methyl ethyl carbinol		rose fragrance
Musky	ω-pentadecalactone		angelica root oil
Pepperminty	menthone		peppermint oil
Pungent	formic acid		ant secretion
Putrid	butyl mercaptan		skunk odor

Table 1.2: Some primary aroma categories proposed by Amoore in 1964

The ODT as it was described above can vary between different categories/group of odors. A correlation between the odor type and the ODT is shown at the Table 1.3 below, taken from [13].

Odorant	Aroma type or source	Chemical structure	Human detection Threshold (mg dm^{-3})†
Benzaldehyde	bitter almond		3.0×10^{-3}
Butyric acid	rancid butter		9.0×10^{-3}
Citral	lemon		3.0×10^{-6}
Ether	ether		5.8
Ethyl butyrate	fruity		1.0
Limonene	lemon		0.1
Methyl salicylate	wintergreen		0.1
Pyridine	pungent		3.0×10^{-2}

† A human detection threshold concentration of 0.1 mg dm^{-3} for a gas or particulate odorant in dry air is equivalent to 77.1 parts per million (ppm) at standard temperature and pressure (STP).

Table 1.3: Range of human detection thresholds for some common odorants

1.3 Animals Olfactory System

Sense of smell is the most mysterious sense among five senses, and its biological mechanism was revealed relatively later than with other senses. This sense has been instinctually used for the perception of dangerous situations or subtle changes in the environment. For instance, fire can easily be recognized through the smell of smoke, and the spoilage of food can be determined by their putrid odors. However, humans have insensitive noses. Non-human vertebrates such as dogs and mice have more types and numbers of olfactory sensory neurons (OSNs) in their noses [14–16]. This makes them more sensitively smell something out than human beings. The analysis is taking place separately for vertebrates, fish, insects and nematodes as their olfaction mechanisms differ significantly. Studying their olfaction system though is a necessary step before attempting to create any bio-electronic nose and knowing their difference is important when designing an application specific bio-electronic nose as we might have to mimic the most suitable olfactory system. Sufficient neuroscience knowledge is considered requisite to go through that subsection. Anyone that does feel comfortable with that should step forward to 1.3.5 where the upshots of the previous start unfolding.

1.3.1 The Vertebrate Olfactory System

The peripheral olfactory system is comprised of the main olfactory epithelium and the accessory olfactory system, which contains the vomeronasal organ (VNO), the Grueneberg ganglion and the septal organ of Masera in rodents (Figure 1.1). The main olfactory epithelium's primary function is the detection of volatile odorants in the air [17] whereas the VNO detects semiochemicals such as pheromones [18–20]. The Grueneberg ganglion is implicated in the detection of stress signals from conspecifics [21–23] and the septal organ contains neurons expressing a subset of odorant receptors (ORs) that are also expressed in the ventral domain of the olfactory epithelium [24, 25].

At Figure 1.1 below we see a sagittal view of a mouse head. The anterio-dorsal tip contains the Grueneberg ganglion. The VNO is located on the ventral side and is divided into 2 halves, each expressing a unique receptor repertoire, which project to the accessory olfactory bulb located at the distal end of the olfactory bulb. The dorsal receptors in the olfactory epithelium are depicted in red and the ventral receptors in blue. The areas they project to on the olfactory bulb are labeled with the same colors. The septal organ is shown at the ventral base of the nasal septum.

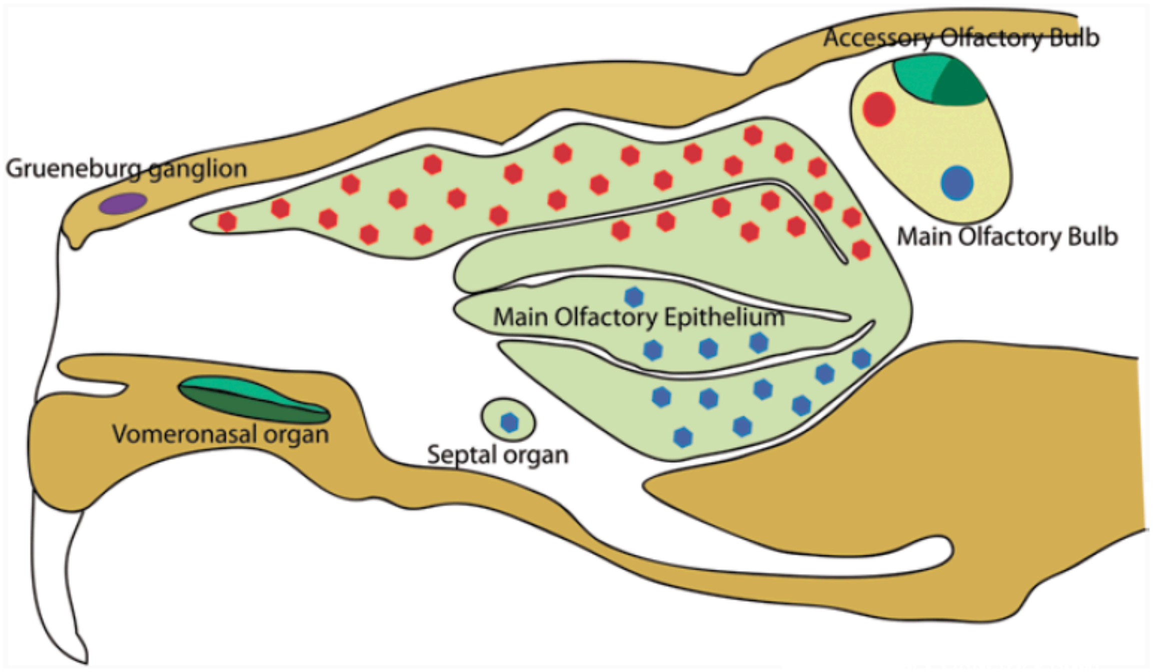


Figure 1.1: Sagittal view of a mouse head

1.3.2 Olfactory System in Aquatic Vertebrates

Fish carry out chemosensation via olfaction, gustation and general chemosensation mediated by solitary chemosensory cells. Unlike terrestrial animals, the fish olfactory system detects water-soluble chemicals. They detect four main classes of odorants: amino acids, gonadal steroids, bile acids and prostaglandins [26]. Fish have a single olfactory organ called an olfactory rosette, which contains 3 types of OSNs: ciliated, microvillous and crypt cells, and projects a tightly fasciculated olfactory nerve to the olfactory bulb (OB), (Figure 1.2) [27]. These 3 types of neurons differ from each other based on morphology (number of cilia, length of dendrite), position (depth in the OE) and their receptor expression profile. Each class of these receptors is believed to use the same or similar signaling pathways as their mammalian orthologs. The fish receptor repertoire is smaller than that of mammals, but their receptors are more diverse in sequence [28, 29] and the repertoire size itself varies significantly amongst species. For example, pufferfish have fewer than 50 ORs compared to the 102 intact and 35 pseudogenized OR genes found in zebrafish [30, 31]. Fish OE have also been shown to express trace amine-associated receptors (TAARs) that act as candidate receptors for polyamines [32, 33, 34], and have a much larger repertoire than mammals in some cases like zebrafish with 109 TAAR genes [32]. Again, different species of fish have large variations in the number of TAAR genes their genomes contain.

A single OSN in zebrafish may either express one OR or multiple ORs as in the case of the zOR103 family of receptors, which express either 2 or 3 receptors in some ciliated neurons [35]. Ciliated neurons in the zebrafish project to the dorsal and medial regions in the OB while the microvillous neurons project to the lateral regions [35, 36]. In contrast, catfish ciliated neurons project to the medial and ventral regions and the microvillous OSNs project to the dorsal OB [37, 38]. Ciliated and microvillous OSNs project to mutually exclusive glomeruli in the zebrafish, where no glomeruli were observed to be innervated by both types of OSNs [35, 36].

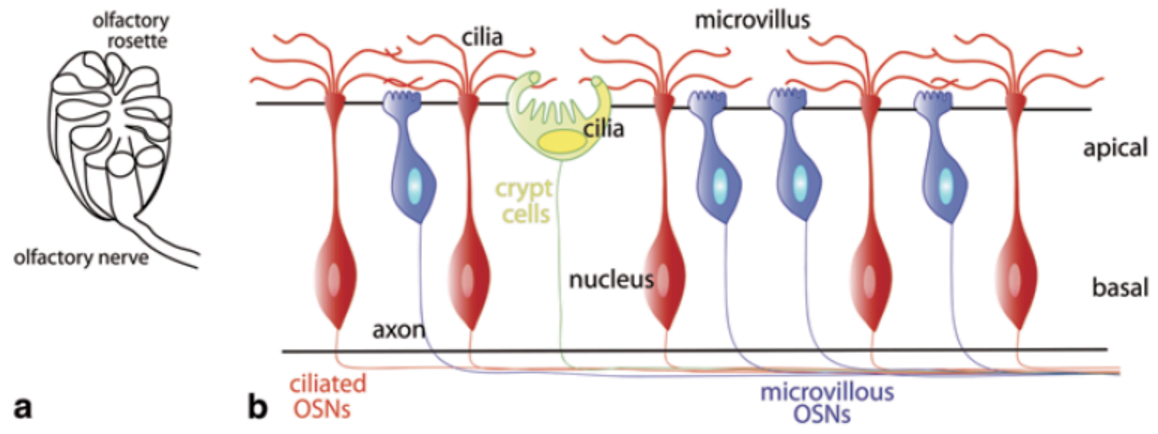


Figure 1.2: (a) Single olfactory rosette, (b) OSNs found in the fish OE.

1.3.3 The Insect Olfactory System

Most insects have two major olfactory organs on their heads, the antenna and the maxillary palp (Figure 1.3a). These organs are covered in sensory hairs called sensilla, which contain up to four sensory neurons, olfactory receptor neurons (ORNs), bathed in the sensory lymph. There are small openings on the surface of these sensilla that allow odorants to dissolve in the lymph and come in contact with the ORN dendrites (Figure 1.3b). These ORNs project a single axon to a single spatially invariant glomerulus, which is also innervated by other ORNs expressing the same OR, and synapses with second order projection neurons in the antennal lobe in a manner analogous to the mammalian olfactory system [39]. There are a total of 43 glomeruli in the antennal lobe in Drosophila, and output neurons project ipsilaterally to Kenyon cells of the mushroom bodies and the lateral horn of the proto cerebrum in the central brain [40]. There are some sexually dimorphic neural circuits which may be involved in the courtship and mating behavior [41].

Insect OR gene families have been identified for Drosophila (62 ORs) [42, 43], Anopheles (79 ORs) [44], Aedes (131 ORs) [45], Apis (157 ORs) [46] and many more. These highly divergent OR genes are not homologous to vertebrate G protein-coupled receptors ORs and they display a novel topology where their N terminus is intracellular and C terminus is extracellular [47, 48]. An inside-out single channel recording of cell membrane excised from an insect OR expressing mammalian cell cultures gave rise to evidence that the insect OR itself is an ion channel [49, 50] that is gated by its cognate ligands (Figure 1.3c). The metabotropic signaling pathway that contributes to activation of olfactory receptor neurons (ORNs) remains a controversial field with many players seemingly playing odor specific roles. RNAi knockdown of $G_{\alpha q}$ in the Drosophila antenna results in the flies becoming insensitive to high concentrations of isoamyl acetate, which normally repels flies [51]. Flies with mutations in norp A phospholipase C have defective maxillary palps [52].

Insects undergo metamorphosis and their larval repertoire of ORs is different from that of adults. Some ORs expressed exclusively in the adult may be responsible for the detection of pheromones responsible for mating. ORNs express up to three different ORs along with odorant receptor co-receptor (ORCO), also known as Or83b. Unlike other insect ORs, ORCO is conserved across diverse species [44–46] and co-expressed with the other ORs [53]. This receptor does not seem to participate directly in odor binding, but instead it forms a heteromer responsible for the transport and transduction of the receptors [53, 54].

Much progress has been made in deorphanizing Drosophila ORs with the use of an “empty neuron” strategy where Or22a/b is deleted, giving rise to a Δ halo mutant fly which has an “empty” ab3A neuron. Using a UAS/Gal4 expression system, the ectopic expression of any OR expression can be driven in ab3A neurons that can then be electrophysiologically tested to study its response to various odorants [55–57]. These studies also showed that insects also detect odors using a combinatorial code of activated glomeruli, just like mammals.

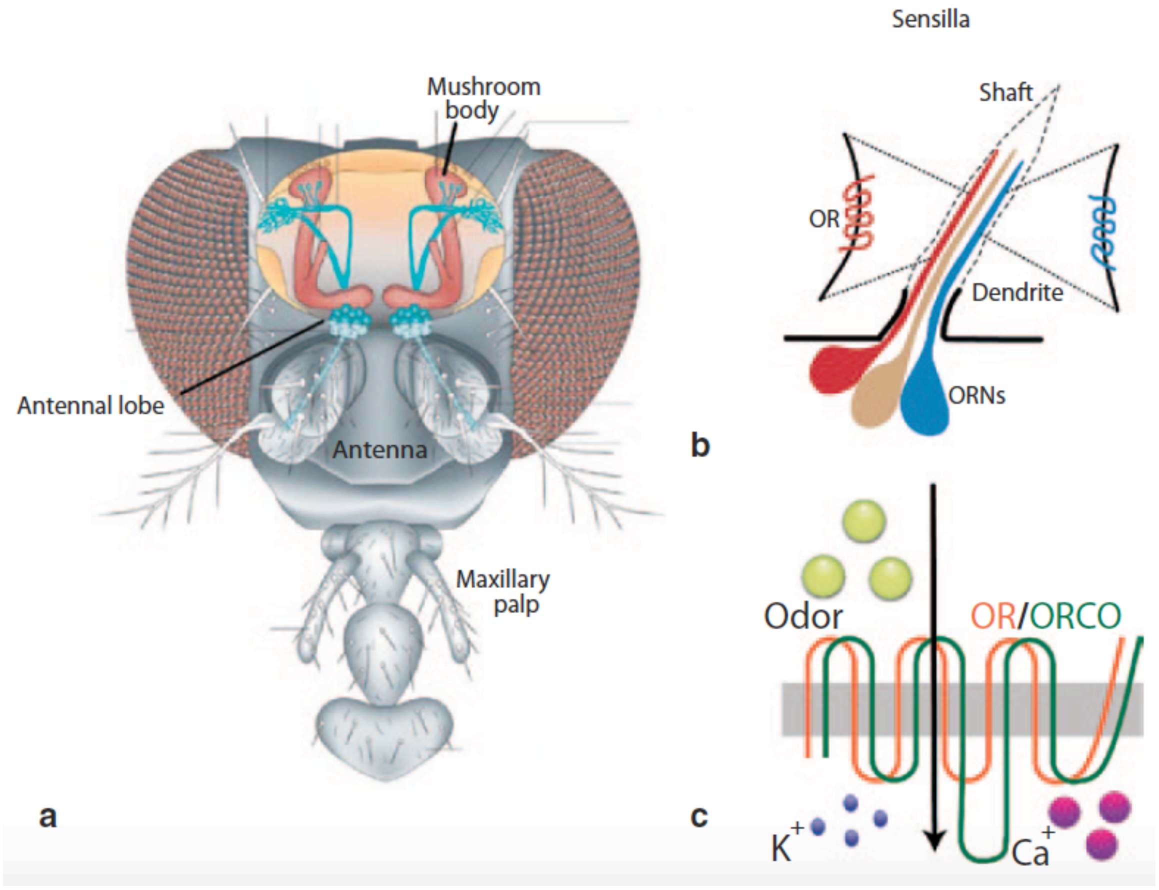


Figure 1.3: (a) Drosophila melanogaster, (b) Sensilla, (c) OR activated

A family related to the ionotropic glutamate receptors (IRs) that mediate neuronal communication at synapses has also been shown in insect ORNs. These ORNs respond to a number of distinct odors without expressing canonical ORs. IRs are extremely divergent with an overall sequence identity of 10–70 % and up to 3 IRs are expressed in an individual ORN [58].

1.3.4 Nematode Olfactory System

Nematodes such as C. elegans carry out chemosensation with chemosensory neurons that extrude their sensilla into the environment via openings made by glial cells called the socket and sheath cells [59]. There are 32 such neurons arranged in bilaterally symmetric pairs along the amphid, phasmid and the inner labia. C. elegans can detect volatile molecules (mostly byproducts of bacterial metabolism) in the nanomolar range and exhibit long-range chemotaxic behavior towards them [60]. Amphid sensory neurons AWC and AWA detect attractants, and their bilateral symmetry aids the detection of gradients [61].

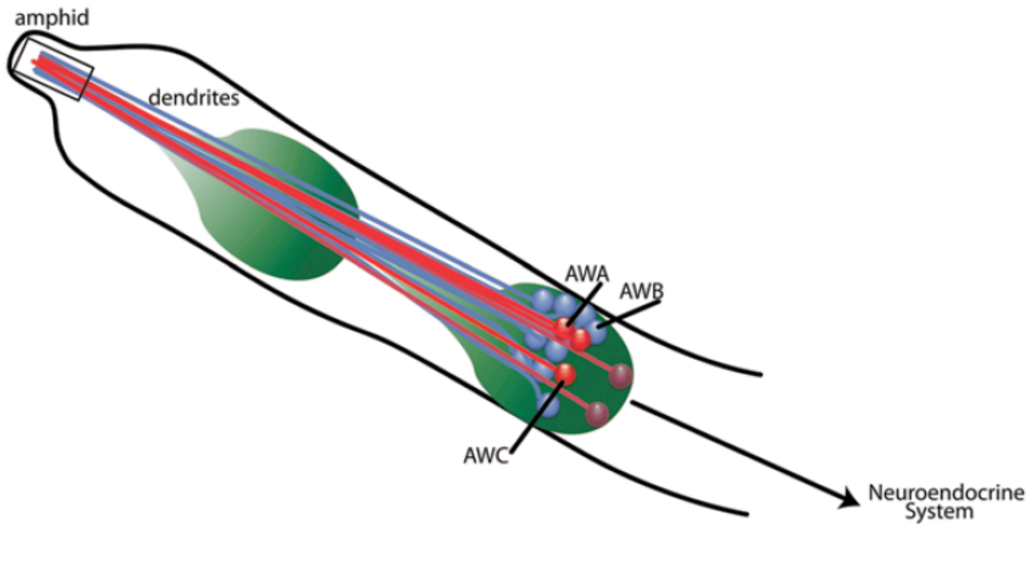


Figure 1.4: The head of C. elegans

As in vertebrates, chemosensation is mediated by 7 transmembrane domain GPCRs that make up 7 % of the genome (more than 500 functional chemosensory receptors) [62–65] and are concentrated on chromosome V, but unlike higher animals, worms do not have the same anatomical structures such as glomeruli to process olfactory information. Their olfactory receptors are evolutionarily divergent from both mammalian and insect ORs, and each neuron expresses a number of olfactory receptors as worms have only 32 neurons [66]. The activation of a single neuron is sufficient for directing behavioral output to environmental cues. Ectopic expression of the olfactory receptor ODR-10 in AWB neurons, which detects repellents, makes the transgenic worms avoid the normally attractive odorant diacetyl, an active ligand for ODR-10 [64, 67], which suggests that this behavior is encoded in the neural circuit (Figure 1.4).

1.3.5 ANIMAL'S OLFACTION APPLICATIONS

In airports, the scene of trained dogs sniffing for explosives and drugs is quite natural. Many animal trainers and scientists have attempted to train animals for the various purposes. Dogs have been trained to detect illegal narcotics and explosives, and their reliability has been examined by many scientists [68, 69]. Trained dogs are able to find narcotics and explosives with very high accuracy [70, 71]. Therefore, they are commonly used in various places such as customs. More recently, dogs and insects have been trained for the diagnosis of diseases. Various types of intractable diseases such as lung, bladder, and breast cancers require early diagnosis. These diseases cause tiny changes in body or urine odors. Although people cannot perceive such small changes, dogs and insects can [72–74]. Mice and rats are also good odor detectors [75–78]. They can recognize compounds that are regarded as odorless, such as carbon dioxide [79]. For these advantages, animals are still trained as alternatives to the human nose.

1.3.6 Animal Training Procedure

The training procedure used for every animal is based to the same logic. Pavlov first issued the basic idea that is today called the Classical Conditioning. Classical conditioning (also Pavlovian or respondent conditioning) is a process of behavior modification in which an innate response to a potent biological stimulus becomes expressed in response to a previously neutral stimulus. This is achieved by repeated pairings of the neutral stimulus and the potent biological stimulus that elicits the desired response. For example, putting some TNT around the bee's antennas and simultaneously hitting the antennas with some sugar water and repeating for 3 times within 1 minute will make bees want to go after TNT at the end.

1.3.7 A comparison

When attempting to investigate the case of creating a bio-electronic nose, except from the capabilities of odor detection that where presented above when describing the olfaction mechanisms we should also take into account the nanotechnology capabilities. Nanotechnology progress should be studied in order to decide what it is possible.

If the case is bioengineering, thus using a trained animal to perform a sniffing task then multiple factors can be taken into account. First it could be the sensitivity. Insects have more ORs than any other animal close or not to their size. The shark might have more ORs but is definitely not convenient to handle by the mankind. Now image leaving a rubbish bag outside the window; probably the first living organism that will sniff it, if you don't have a pet, would be a fly. While dogs, rats and pigs have been used for the detection of explosives, contraband and sometimes diseases [68, 69, 80], it is hard to embed their abilities in an automated system, expensive to train and maintain and they usually need a supervisor to handle them. Insects on the other hand are very handy to restrict and monitor, cheap to maintain, easy to train and extremely capable. Finally, their behavior has been extensively studied in the past [81-95].

Another interesting field contains genetically engineered plants with nematode olfactory system, which have been used for the detection of explosives [80]. As we already mentioned the nematode olfactory mechanism has great capabilities but it is usually met in microorganisms that do not have the capacity to be conditioned.

Chapter 2
Towards Artificial Olfaction

Abstract

An overview of the technological progress in the field of olfaction is presented in this chapter. In the beginning we will refer to the devices that are known as electronic noses, which are not something more than sensor arrays combined with knowledge based systems for the purpose of odor prediction. The technological progress over the years and the numerous applications of electronic noses are mentioned here. Then we will introduce the idea of mimicking the biological system of olfaction and the technological advances to the bio-electronic nose. The recent concept of smell visualization is also mentioned and finally we refer to the interesting bioengineering methods that have been used in the past to overcome the limitations of electronic noses. Bioengineering systems are necessary for the experimental processes towards a bio-electronic nose and have already put into practice to solve real life problems as it shown in this chapter.

2.1 Electronic Nose

An electronic nose from here and on called as an e-Nose used to be a controversial term during the past 20 years when it actually met a huge development. Theoretically with the term e-Nose we refer to an electronic device capable of detecting the odor of one or more volatile compounds. Made up of electronic components, those devices are able to recognize the chemical structure or other physical characteristics of the target volatile compound and give us information about a known odor presence and concentration. An e-Nose can be odor specialized or general purpose according to the technology used. On the way to Digital Olfaction and while studying the progress of this technological field there are a few moments in history that worth mentioning towards to the current state of the art.

2.1.1 HISTORY OF THE TERM

Dravnieks at 1968 [1] envisioned an artificial (or electronic) nose as *''an instrument that would inspect samples of odorous air and report the intensity and quality of an odor without the intervention of a human nose''*. Taking that as a definition, at 1982 [96] a novel smelling device was constructed using semiconductor-based transducers, and it was demonstrated that the sensor was able to reproducibly discriminate various odors. This study became the flagship of future trends and most of the studies that where conducted back then. It was that sensor that many tried to make it perform better using more advanced materials and improved methods even sometimes till today, but it was not called an electronic nose yet. The term e-Nose first began to be used by scientists in the mid to late eighties but the following definition was not published until 1994 [97]:

"*An electronic nose (e-Nose) is an instrument which comprises an array of electronic chemical sensors with partial sensitivity and an appropriate pattern recognition system capable of recognizing simple or complex odors"*

It was from then and on when great progress to the field was conducted. Different sensor technologies, various signal processing algorithms and techniques have been combined together in order to built more efficient systems. Sometimes the purpose of optimization was the cost and some others the sensitivity. Purpose oriented sensor became better and cheaper and the field of usage was widened. E-noses have been used from the food industry to military applications and medicine.

2.1.2 Electronic Nose Architecture

On the meanwhile, important studies supporting the scientific background of e-noses have been reported. In the early 1990s, Buck and Axel revealed the olfactory mechanism [98, 99]. In the nasal cavity, there are numerous OSNs, which are the primary odor sensing cells. Each OSN expresses a single type of OR on its surface membrane. ORs have an excellent selectivity capable of precisely discriminating ligand molecules among a mixture of analogous compounds [100]. Once a certain odorants bind with specific ORs, a signal cascade is activated and olfactory signals are generated [101]. The generated signals are transmitted to the brain, and a person becomes aware of the characteristics of odors using the combination of activated OSNs [102].

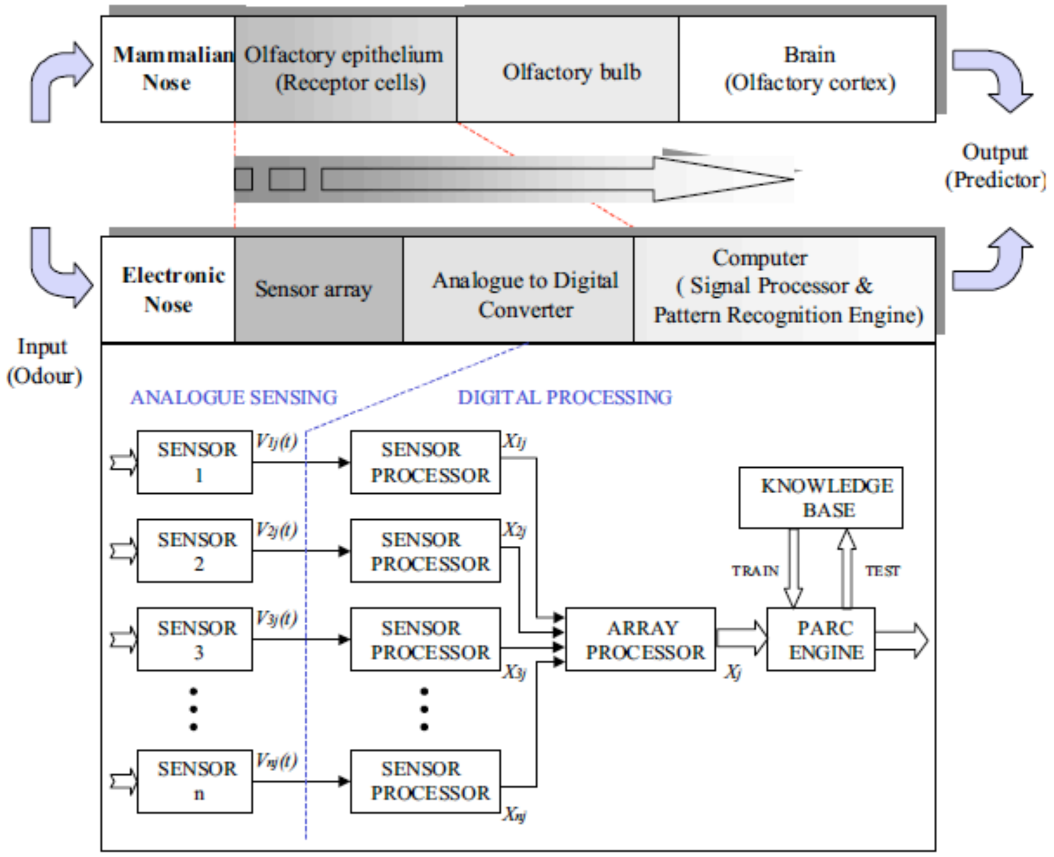


Figure 2.1: e-Nose architecture vs the mammalian model schematically

The odor-discriminating mechanism of the natural olfactory system is very similar to that of e-noses. E-noses are commonly composed of an array of several sensors. Each sensor produces specific responses by reacting to chemical compounds. Thus, an e-nose can generate ‘odor fingerprints’, which are unique patterns of odorants [103]. Odorants can be identified by analyzing the response pattern. This similarity is shown schematically in Figure 2.1 (taken from [104]). The stimulation of odorants generates specific response patterns from olfactory cells in the natural olfactory system or sensor arrays in the e-nose [105]. The generated patterns are analyzed as a specific characteristic of odors in the brain or electric devices. Both mammalian and e-noses recognize odors through this process.

2.1.3 Sensor Types and Materials

Many compounds have been used as a sensor material for the development of e-noses. Metal oxide (MOx)-based sensors have mostly been developed [106–110]. The surface of MOx is modified with diverse chemical compounds. An array of MOx allows the sensor to generate specific response patterns. Conducting polymers or field-effect transistors have also been applied to e-nose systems [111–114]. Such devices have been utilized in many fields requiring the detection of toxic molecules such as amines, alcohols, and sulfur compounds. Also, surface acoustic wave transducers with an array of polymer layers have been extensively used for the development of e-noses [115–118]. Those are the materials that compose chemical sensors, which are mainly used to build general-purpose e-noses. Besides chemical type sensors, e-noses have been constructed application specific with a large variety of sensor types [119], with most common the optical and mass sensors, where gas chromatography and mass spectrometry are used respectively.

However, general-purpose e-noses have critical limitations for application in practical fields. First, the sensitivity is insufficient. It was reported that people could detect odorants at concentrations lower than the ppt level, whereas the sensitivity of e-noses has mainly been in the ppm or ppb range. General-purpose e-noses could not specifically distinguish one odorant within a mixture of odorants. Moreover, e-noses cannot fundamentally mimic the human biological olfaction until it is clearly understood and then is where the conversation about bio-electronic noses steps in.

2.1.4 Data Processing

A set of output signals from individual sensors within an e-nose must be converted into signals appropriate for computer processing. Electronic circuits within a sensing instrument usually perform an analog-to-digital conversion of output signals from sensors, feature extraction of useful information, and interfacing to an external computer for pattern analysis. For unattended automated sampling, it is common to have a mass flow control system that delivers the odor from the source to the sensor array. The final response vectors generated by the sensors are then analyzed using various pattern recognition techniques.

In the 1990s the possible application of artificial neural networks to e-nose systems was suggested as well as the analysis of variance. Gardner and coworkers implemented a three-layer back-propagation network with 12 inputs and 5 outputs architecture for the discrimination of several alcohols. Later at 1992 cluster analysis and principal component analysis were used to test 5 alcohols and 6 beverages from 12 tin oxide sensors. The same group at 1994 developed a stand-alone microprocessor-based instrument that can classify the signals from an array of odor-sensitive sensors. Data from the odor sensor array were initially trained on a personal computer using a neural network program, and then the neuronal weights were sent to an artificial neural emulator, which consisted of a microprocessor, an ADC chips, a ROM, and a RAM. Another approach to odor sensing was studied using a quartz resonator sensor array where the mechanism of odor detection is based on the changes in oscillation frequencies when gas molecules are adsorbed onto sensing membranes. Nakamoto and his team employed neural network pattern recognition, including three-layer back-propagation and principal component analysis, for the discrimination of several different types of alcoholic drinks using a selection of sensing membranes at the 1990s. They also proposed a new processing element model based on fuzzy theory and Kohonen's learning vector quantization for the discrimination of known and unknown odors.

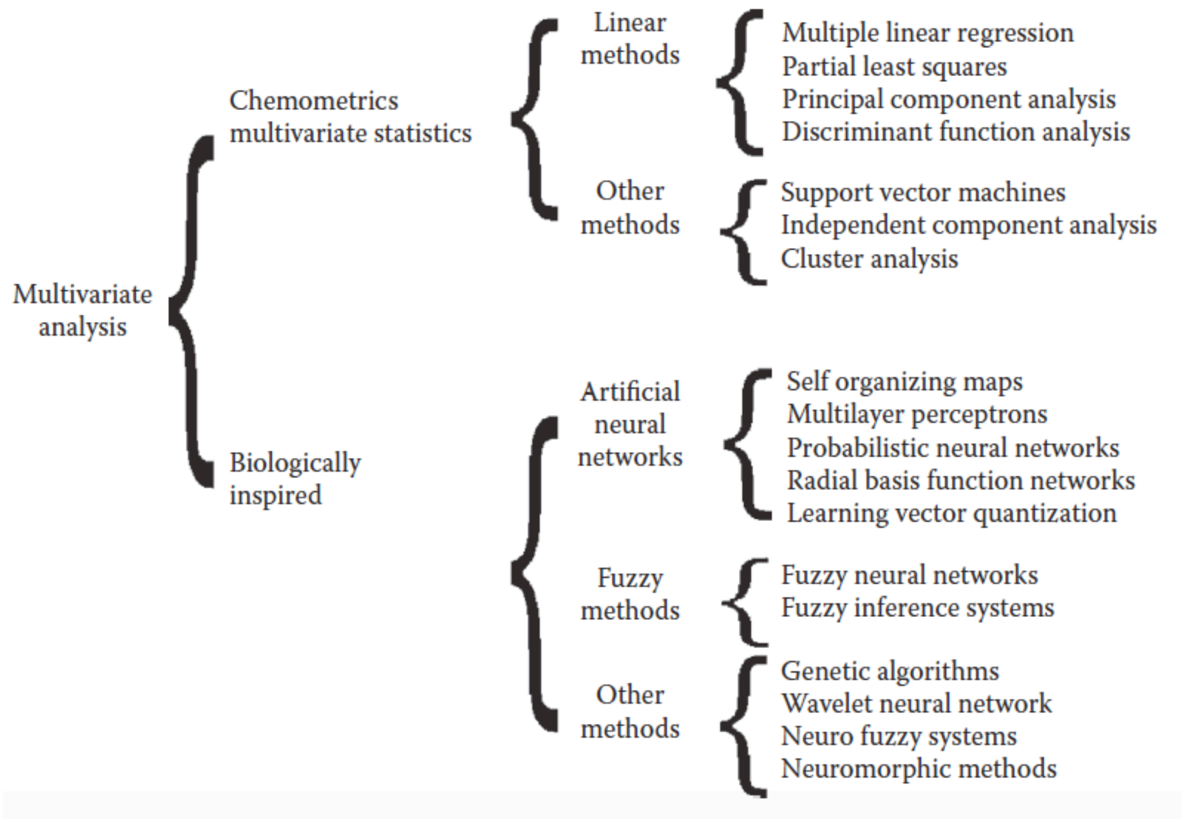


Figure 2.2: Multidimensional data processing methods from sensor arrays

Pattern recognition in the e-nose system may be regarded as a branch of artificial intelligence that involves the mimicking of human intelligence to solve chemical problems. Figure 2.2 summarizes methods commonly used in the field. There are two main approaches to pattern recognition: parametric and nonparametric. Parametric methods rely upon obtaining or estimating the probability density function of the parameters used to characterize the response of a system. Conversely, nonparametric methods require no assumption about the fundamental statistical distributions of data.

Two types of nonparametric learning or classification methods are available: supervised and nonsupervised. Supervised methods involve the learning of data based on advance knowledge of the classification, whereas unsupervised methods make no prior assumption about the sample classes but try to separate groups or clusters. Mapping methods in pattern recognition are an unsupervised way of analyzing chemical inputs. Many authors since these early days of e-nose research have used pattern recognition techniques for a number of practical applications.

2.1.5 E-NOSE APPLICATIONS

E-noses are widely used for the improvement of life quality. The have been used for environmental monitor [120], while specific sensors design for food quality control, have been done for an application range from wine ageing detection and variety discrimination [121-128], to olive oil characterization [129-132] and herb classification [133].

Foraging tasks are also a field that e-noses can contribute significantly. Although the governmental agencies are made out of very strict structures and it will take time to use the e-noses as they still trust and are sentimentally connected with the mans best friend, there are a few serious studies about contraband detection [134] and detection of explosives [135] out there. Even NATO has published a book about the use of e-noses for detection of explosives [136].

Industry sector	Application area	Specific use types and examples
Agriculture	crop protection	homeland security, safe food supply
	harvest timing & storage	crop ripeness, preservation treatments
	meat, seafood, & fish products	freshness, contamination, spoilage
	plant production	cultivar selection, variety characteristics
	pre- & post-harvest diseases	plant disease diagnoses, pest identification
		detect non-indigenous pests of food crops
Airline transportation	public safety & welfare	explosive & flammable materials detection
	passenger & personnel security	
Cosmetics	personal application products	perfume & cologne development
	fragrance additives	product enhancement, consumer appeal
Environmental	air & water quality monitoring	pollution detection, effluents, toxic spills
	indoor air quality control	malodor emissions, toxic/hazardous gases
	pollution abatement regulations	control of point-source pollution releases
Food & beverage	consumer fraud prevention	ingredient confirmation, content standards
	quality control assessments	brand recognition, product consistency
	ripeness, food contamination	marketable condition, spoilage, shelf life
	taste, smell characteristics	off-flavors, product variety assessments
Manufacturing	processing controls	product characteristics & consistency
	product uniformity	aroma and flavor characteristics
	safety, security, work conditions	fire alarms, toxic gas leak detection
Medical & clinical	pathogen identification	patient treatment selection, prognoses
	pathogen or disease detection	disease diagnoses, metabolic disorders
	physiological conditions	nutritional status, organ failures
Military	personnel & population security	biological & chemical weapons
	civilian & military safety	explosive materials detection
Pharmaceutical	contamination, product purity	quality control of drug purity
	variations in product mixtures	formulation consistency & uniformity
Regulatory	consumer protection	product safety, hazardous characteristics
	environmental protection	air, water, and soil contamination tests
Scientific research	botany, ecological studies	chemotaxonomy, ecosystem functions
	engineering, material properties	machine design, chemical processes
	microbiology, pathology	microbe and metabolite identifications

Table 2.1: e-nose examples of industry-based applications

At Table 2.1 above (taken from [13]) more applications of e-noses are described, but most important is that e-noses have contributed in medicine, as they are able to detect from skin diseases to lung, stomach, liver and intestine cancer [137]. Generally any disease that causes the presence of odorous compounds to the breath [138], the skin, the urine or excrement can be diagnosed by the proper e-nose.

2.2 Bio-electronic Nose

In the late 1990s, a novel and more advanced concept of sensor devices was suggested [139]. The challenge was to use ORs as a sensing material in order to mimic a human or animal olfactory system. This new device is called a 'bio-electronic nose'.

2.2.1 COMPARATIVE ADVANTAGES

The bio-electronic nose is based on OR proteins or cells expressing ORs on their surface membrane. ORs are odorant-recognition elements, and are combined with sensor devices that convert biological signals into electrical or optical signals. Since ORs provide odorant-discriminating ability, the bio-electronic nose can closely mimic a human or animal olfactory system. The concept of odorant analysis using a bio-electronic nose fundamentally differs from the odor-discriminating strategy of e-noses based exclusively on pattern recognition using sensor arrays. When the ORs are utilized as a primary sensing material, the sensors can precisely distinguish a target molecule among a mixture of various compounds. In addition, sensors based on ORs are more sensitive than electronic noses. The limit of detection reaches the femto-molar range in liquid conditions and the ppt range in gaseous conditions, which is similar to that of a human nose [140, 141]. By virtue of these excellent characteristics, the bio-electronic nose is now receiving great attention from diverse fields such as disease diagnosis, food safety assessment, and environmental monitoring.

2.2.2 The Concept

A bio-electronic nose consists of two main parts: an odorant-recognition element and a signal transducer, as shown in Figure 2.3 [142]. For the odorant-recognition, cells expressing ORs in their surface membrane, OR proteins, and nanovesicles have generally been used. In the human nose, approximately 390 different types of functional ORs exist [143]. However, humans can discriminate thousands of types of odors. This asymmetry originates from the excellent odorant-recognition ability of ORs, which are capable of distinguishing between their specific ligands and partial ligands, as well as irrelevant molecules [101, 102]. A single odorant activates various types of ORs, and one OR is activated by several odorants. Thus, numerous combinations of activated ORs can be generated. These combinations are recognized as a unique property of an odor in the brain [144].

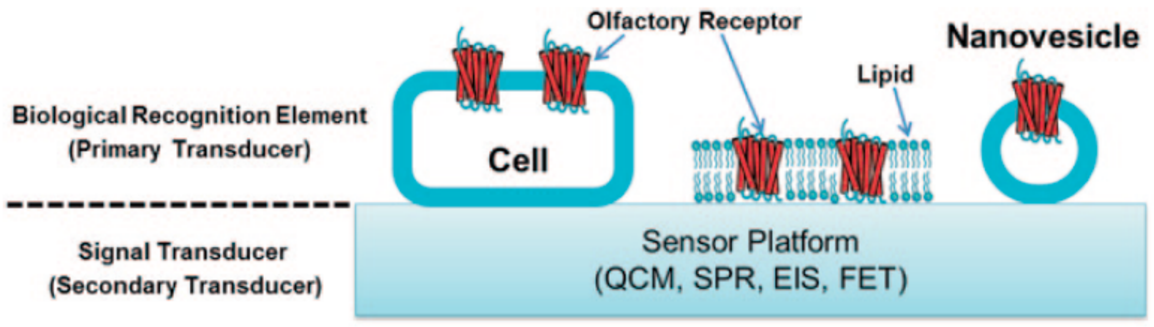


Figure 2.3: Composition of a bio-electronic nose

2.2.3 Development Attempts

A bio-electronic nose utilizes this odor-discriminating ability of ORs. Thus, they can detect specific odors with great selectivity. For instance, the odor from decomposed seafood can be easily distinguished among other odors from various spoiled foods when a bio-electronic nose is functionalized with receptors that can selectively detect the odor of spoiled seafood [145]. However, problems still remain to be overcome. ORs have a seven-transmembrane structure with a high hydrophobicity in the transmembrane region, which makes the expression of ORs in a heterologous system very difficult [146]. Several attempts have been made to achieve functional expression in various heterologous systems. Among various systems, human embryonic kidney cells are broadly used because they allow for a relatively high expression level [147–149]. The identification of membrane-targeting tags and accessary proteins assisting the membrane expression, such as rho-tag and receptor transporting protein 1S, facilitated the high-level expression of ORs in mammalian cells [150–152]. Insect cells, such as SF9 cells, have also been used for the functional expression of ORs [153, 154]. Saccharomyces cerevisiae and Escherichia coli are also good OR expression systems [155, 156]. These systems allow for mass production and efficient purification processes. Thus, they are effectively being used to produce OR proteins for bio-electronic noses.

2.2.4 Current State

Bio-electronic noses cannot yet be extensively used in all areas where a person can smell. This is because the functions of all ORs have not yet been fully revealed. Three hundred and ninety types of human ORs react to different odorants. Moreover, numerous odorants can affect ORs. Thus, incalculable combinations between ORs and odorants exist, and the relationship should be investigated further [157]. Following the further investigation of the specific function of ORs, the application area of bio-electronic noses will continue to grow.

The most interesting thing in bio-electronic noses is that OR-based sensors can mimic the olfactory system. Types of odors have not yet been fundamentally and scientifically classified. Thus, there is no other choice but to recognize specific odors based on personal experience. However, bio-electronic noses will offer a way to classify odors. Each odor has a unique response pattern with activated ORs [102, 157]. A sensor functionalized with a multi-array of all types of ORs can represent whole response patterns in vitro. This means that bio-electronic noses can classify the types of odor without the help of the human nose. Thus, multi-array sensor platforms are being developed to fundamentally understand the characteristics of odors.

2.2.5 VISUALIZATION OF SMELL

Odor as a chemical sense has not yet been decoded. The lack of a complete knowledge about the human olfactory system incommodes the process of creating an absolute replica of the human nose in the form of bio-electronic. Still in order to discriminate among thousands of odors, we do not have any reliable artificial sensing device, but still depend on our human sensory system. However, the information obtained through our natural sensory system does not provide us with the objective information about the odors. Up to now, complex experimental steps, large-scale equipment, well-trained experts, or e-noses have partially fulfilled the need for the detection or analysis of odors. In addition, it is difficult to collect the data on the odorant response pattern to be used as an objective index. However, if the odor is visualized, the visualized pattern can be used as a code for the odor like a QR code. On the other hand creating a visual metaphor might allow the human brain to recognize a possible pattern that a bio-electronic nose might create, but the human brain cannot conceive. Several approaches have been taken to visualize the olfactory signal transduction in the cell-based bio-electronic nose system.

2.3 Bio-engineering

While e-noses exhibit limitations to sensitivity and selectivity, bio-electronic noses are not yet established to the market, as their construction is still too expensive and their application range limited. Definitely a lot of research is expected to be presented on the field of bio-electronic noses the upcoming years as their concept is very promising. That includes advances in nanotechnology and neuroscience as well as to the integration of those two. From the engineering point of view, in the meanwhile we need to overcome the limitations of e-noses in a practical manner and aid at the same time the field of research regarding bio-electronic noses.

Bioengineering in the terms of olfaction refers to the integration of the existed technology in cooperation with any biological form of olfaction. This have been done in past utilizing the olfactory system of animals with a strong sense of smell to nematode olfactory mechanisms of plants [80] and other living organism. As we shown at chapter 1 insects has one of strongest olfactory system that combined with their small size make them ideal for bio-engineering applications. In addition someone can find within the many insect species, physiological responses that can be directly applied in to practice without any conditioning. For instance, the larvae of jewel beetles of the genus Melanophila (Buprestidae) can develop only in the wood of trees freshly killed by fire. To arrange this, the beetles need to approach forest fires from as far as 50 kilometers away. Those larvae may act with a bioengineering method as the most sensitive infrared sensor ever made [158]. In general, the smaller the living organism is the more convenient the bioengineered sensor. When condition is necessary, brain capacity should also be taken into account. Some of the most interesting bioengineering methods developed are presented below.

2.3.1 The Electro-antennogram Method

Insect antennae are highly sensitive to odors of survival interest but also to compounds such as explosives and controlled drugs (unpublished data). The electro-antennogram (EAG), the biopotential developed between two points on an insect antenna, is the result of the massed response of the ORNs to an odor stimulus; several groups have shown the potential use of insect antennae and the EAG in a hybrid-device biosensor [159–161]. A portable system capable of discriminating odors in real-time was designed at 2008 and utilizes the electrophysiological responses recorded from a sensor array composed of male or female antennae from four or eight different species of insects EAG [162]. This EAG method became very popular due to the bee learning ability. Bees and EAG method integration have led in many portable devices currently available on the market and can be used from foraging applications to cancer detection.

2.3.2 Insects On Field

The bees extremely sensitive olfactory system combined with their tendency to move at long distances and their conditioning/learning ability makes them great agents even as free moving insects. After conditioning, the bees can be left on the field to search for the target compound. Systems have been designed that track their journey either using passive integrated receivers or laser array radars that scan the area and track the insects by their winging modulated frequency. Those methods were developed mainly for detection of explosives and land mines and then for agriculture [163, 165].

2.3.3 The Hard Wired Moth Example

Insects, such as moths, can also be trained to respond to specific odors. The electromyographic (EMG) signals of insects can then be compared to deduce the presence of an odor. A prototype system that uses moths to detect explosives was designed, assembled, and tested [166]. It compares the EMG signals of moths trained to respond to those trained not to respond to the explosive signature in order to determine the extent to which it is detected. The device was designed to be portable by making it lightweight, battery-powered, and energy efficient. The prototype performed successfully during a demonstration for DARPA. Though portable, this prototype is still rather bulky. At two pounds each, the ten EMG amplifiers constitute the bulk of the entire system weight. A much smaller and lighter handheld version of this device could be made by integrating the signal conditioning and data processing stages, using ASIC chips, and constructing a one-piece animal exposure chamber and electronics package.

2.3.4 INSECTS IN OLFACTOMETER

The whole idea is almost as it is described in the tittle of this thesis. Conditioned insect(s) is positioned in a monitored restricted environment where odorous air is injected from a specified direction. The conditioned insect movement then is tracked by a video monitoring system to determine if it was affected by the odorous air. Some insects follow a movement pattern at the presence of the target compound, while insect populations conditioned to be attracted to an odor will show concentration to the direction that odorous air is coming from. Usually that restricted environment is an olfactometer and its design, geometry and properties are extensively described at the next chapter, as it is a part of our system design. Our design though differs from most of the systems that have been implemented in the past [167-170]. The interesting thing about those systems is that most of the insects used are parasites. The parasitoid wasp, Microplitis Croceipes, showed dozens times greater sensitivity and smaller response times that e-noses [171].

2.4 Conclusions

In contrast to the senses of vision and hearing, the sense of smell does not have a method to precisely express the information of smell or flavors, even though it plays an important role in our daily life. It is very difficult to create a database and to standardize the information obtained from the olfactory sense, because it usually responds to complex components consisting of a vast variety of chemical elements. The classification and description of smell depends on quite subjective and abstractive expressions and smell cannot be precisely described or quantified using these kinds of expressions.

Various studies are being currently carried out also for the display of olfaction, for example, the development of a device exuding flavor through the internet, and the development of a method and device for smell transmission. However, these kinds of trials have a limitation, in that the device does not deliver the exact information of a smell because there is no criterion for a variety of smells. Modeling the odor will be an important criterion for digitalization of emotional expression of smell, as well as for various applications of smell and flavor, and therefore, needs to be achieved by mimicking human olfaction as closely as possible.

The bio-electronic nose may be the best concept mimicking the human olfactory system, because it utilizes the human olfactory receptors as a sensing element and employs the olfactory signaling processes. It is expected that a method for the standardization of smell can be developed through the qualitative and quantitative measurements from bio-electronic noses, and eventually, that a relationship between data from bio-electronic noses and sensory evaluation data from human olfaction would be established. The established standard of smell can be widely utilized in various industries such as food, beverage, agriculture, flavor and perfume, as well as biomedicine and environmental monitoring.

In the meanwhile, a bio-electronic nose that can recognize every smell that a human nose can, have not yet been created. Generally they are expensive to produce, hard to maintain and not too practical. Thus, we need to monitor the olfactory systems and behaviors of many living organisms with sensitive “noses” that could lead us to practically useful sensing systems for all the above-mentioned applications. For that sake, any application that the minimum requirements are not met due to e-nose limitations and at least for experimental purposes bioengineering techniques development is necessary.

Chapter 3
System Design

Abstract

In this chapter unfolds the results of research that was made towards the system implementation. The physical conditions under which the system will work are described and a simulation environment is proposed. The basic logic behind the algorithm is discussed. Then the logic is deployed and tested with Matlab. Extensive report about Image Binarization and Thresholding techniques is written focusing on the methods that seem to be more useful during the implementation. Finally a research about the available technology that could support the system comes with a discussion about all the candidate devices, their specifications and prospective. Overviewing also the computational power needs for out experimental purposes as well as the cost and the availability, we come up BeagleBone Black, which will be finally used to implement the system.

3.1 Setup Physical Requirements

The first step of the designing process includes the decisions that should be taken about the system setup. The geometry and functionality of the restricted environment are discussed below, as well as the placement, the properties and the environmental conditions for the capturing device. Finally a setup simulation is proposed to serve the experimental purposes in a rational cost.

3.1.1 The Olfactometer

If somebody asks the question: What is an Olfactometer? Then the most precise answer that someone could give and might be close to a definition would be: An Olfactometer is an instrument that measures an odor. As there is no general model describing an odor there isn't any universal instrument either.

At this thesis with the term Olfactometer we refer to the devices that molecular biologists use in order observe insects and other small animals with strong olfactory system behavior. It is still a controversial term as many scientists in the past have given that name to plenty of other devices including one that was invented by German laboratories together with a few conventions about the measurable properties of olfaction. As this device was made in order to observe the human olfaction it worth mentioning a few more details about it. It is a device that it is able to isolate known volatile compounds and then present them to the human nose. The object of decision thus the human standing behind the Nasal Ranger Field Olfactometer [172] as it was later called, have to judge the sense of the properties of the odor. Those properties are the 'concentration' the 'intensity' the 'character' and the 'hedonic tone'. More details about the properties that an odor could have according to the human conception can be found in the 1st chapter of this thesis.

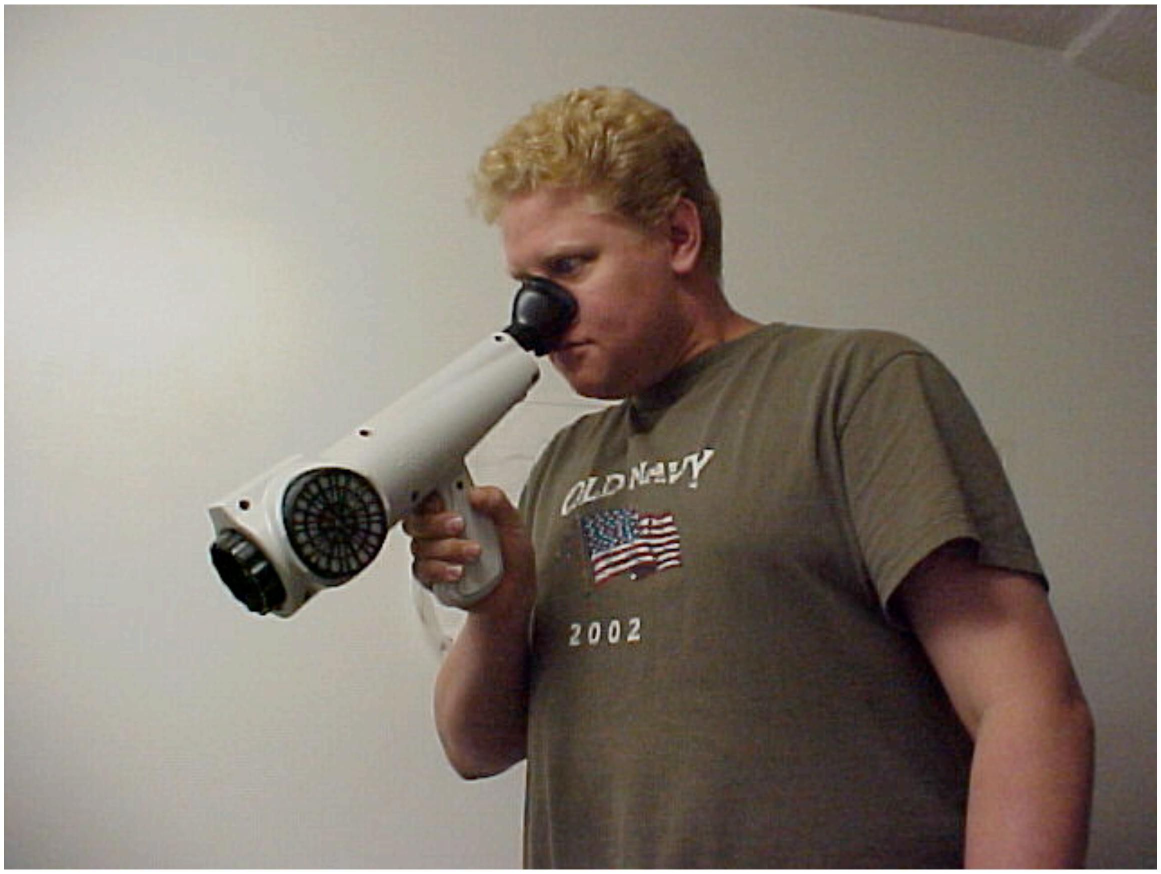


Figure 3.1: The Nasal Ranger Field Olfactometer

The instrument that our developing system is using is the one that is shown in the picture below. This is a 4-way Olfactometer but one with more or less inputs could have been used. This instrument/device is constructed as follows. A solid peace of Teflon with dimensions about 30x30x3cm is getting dig by the lathe in order to create the desired shape witch look like a corner-less star with as many edges as the ways of the Olfactometer. The digging depth might be about 2cm. A hole must be opened at the center of the square surface and at every edge/way on the side. The surface opposite to the central hole is covered by a see through plastic film and sealed. Because the area inside the Teflon will be the restricted environment that we will place the insects all the holes that we have opened must be covered with net after the insect placement. At the central hole on the bottom a vacuum is attached that pumps air out of the environment to create a negative pressure state, which will force air get inside from the other holes. Each of those holes have a valve attached that controls the flow and air champers that are holding known volatile compounds are also attached on those valves.

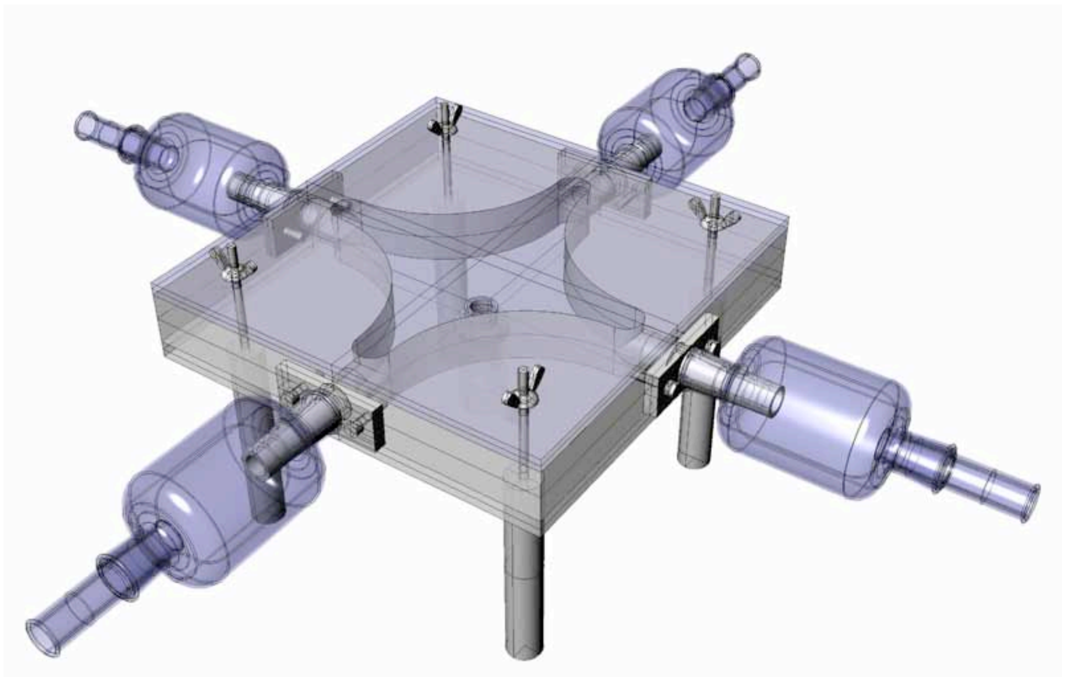


Figure 3.2: The 4-way Olfactometer

3.1.2 Camera Setup & Lighting Conditions

Assuming that we have the 4-way Olfactometer ready for use, now we have to attach a camera at the top of out construction in such a placement that the captured frame will be filled and centered with the top view of the Olfactometer. Some dimensional setting like crops can be done software side but still the centering at this step is important for our setup to work. Black insects color opposed to the white background of the Teflon gives a grate contrast to the frame which will later on simplify the processing procedure while still some other factors must be taken into account. First of all the light in the environment must be uniformly distributed with minimum intensity variation. In case of no light condition a thermal or night vision camera will be necessary. The light restrictions are important for two reasons. Firstly to make sure that it will not affect the image processing procedure and also because the insects change their behavior in different light condition. The option of absolute darkness is considered if we would like to use a similar setup for an application specific portable device.

3.1.3 SETUP SIMULATION

Although the 4-way Olfactometer presented above might seem a quite simple device to either buy or construct by yourself, it is expensive either ways. The reason is the cost of Teflon, which is necessary and cannot be replaced by another material due to its properties. Teflon is very dense and slick, letting even the most sticky volatile compounds slide on the surface, so all the chamber needs between two experiments to be cleaned is some fresh air.

As the above setup covers a complete insect response experiment which is not what we will be dealing with at this thesis, an instance of the Olfactometer with the specific placement of the insects inside of it as it is captured by the camera will be printed on piece of paper. In other words pictures that have taken from a similar experiment will be printed and our camera will be placed above them in uniform light condition in order to complete the image processing and algorithmic part of the experiment. Given that simplification we are having the opportunity to test minimum processing power requirements for real time response as well as the correctness of our algorithm.

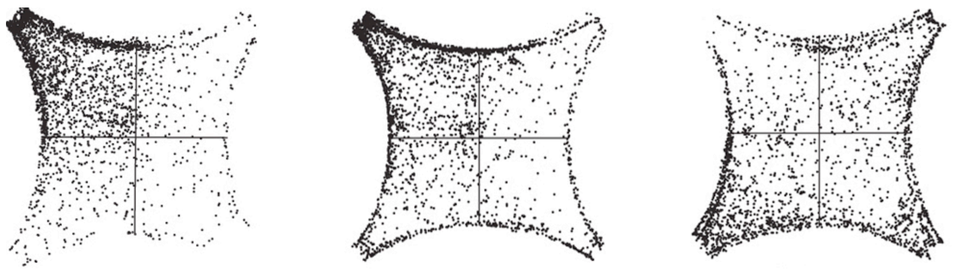


Figure 3.3: Square Frames examples

3.2 Algorithm Design

What we need to do is nothing more than capture and process one frame of the video the camera is sending us within a period of time that would consider our system responsive to the outside world. Given that our system is only answering if there is a gathering of insects at one side of the Olfactometer, an interval of something less than a second can be considered small enough and gives us a responsive system. This is a recurring process that goes on as far as the system is alive. On the next chapter we will describe how we use the camera to capture video, get a frame from that video, handle any frame cropping that might be necessary after the setup and so on. For this implementation we need specific hardware, operating system, libraries and sometimes we also need to interfere on the video driver, but at the moment we will be designing the image processing part of the algorithm, assuming that it is handling a square frame that is filled by the top surface of the 4-way Olfactometer as it is shown at the picture above.

3.2.1 Frame Processing

Given that we have already captured the square frame, it is now necessary to transform that image to a binary form. This is considered a good practice as it reduced the following computations of our algorithm while we are not losing any crucial information. The way we measure the concentration of the insects in a specific region of the binary image, is counting the number of the black pixels in that region. Dividing that with the overall number of black pixel of the frame will give us the concentration as a percentage for the target region. The regions for us couldn't be other than the four edges of the 4-way Olfactometer, so the width and the length of the picture are divided by two and rounded to closest integer in order to split it into quadrants. The final step is to define a threshold percentage that will be considered as a gathering of insects and the system is ready to give the requested outputs.

3.2.2 MATLAB IMPLEMENTATION

Before dealing with raw video format and the capturing process that will later on take place platform specific by the help of libraries and drivers we have to prove that the algorithm described above is functional. For that purpose we will be using Matlab and the functions provided with the Matlab toolbox. At this point no image binarization thresholding techniques are discussed yet, thus for the algorithm verification on Matlab we used an empirical binarization threshold. The Matlab work is following.

Finding Concentrations Algorithm Implementation

This script was written to demonstrate the functionability of the algorithm designed to find the concentrations of insects in the restricted environment of an Olfactometer experiment.

```
close all;
clear all;
clc;
```

Image Reading and Binarization

```
% Reading the RGB picture from file
rgb_img = imread('Figure.jpg');
figure;
imshow(rgb_img);
% Converting the RGB image to Binary
bw_img = im2bw(rgb_img,0.7);
figure;
imshow(bw_img);
```

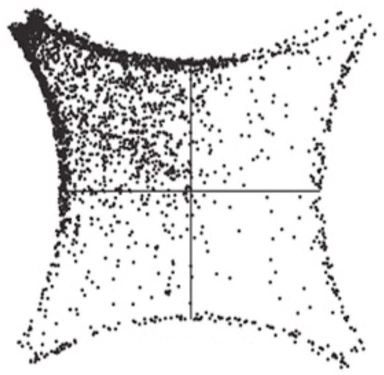


Figure 3.4: RGB Image, Matlab

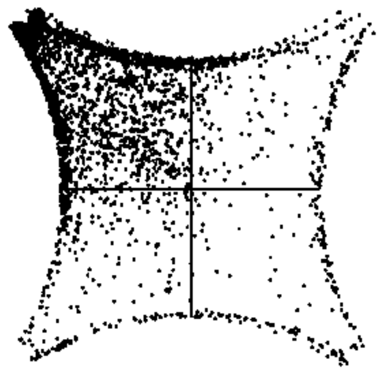


Figure 3.5: Binary Image, Matlab

Splitting Binary Image into Quadrants

```
% Counting the image dimensions
[img_height, img_width] = size(bw_img);
```

```
% Splitting into quadrants
I1=bw_img(1:round(size(bw_img,1)/2),1:round(size(bw_img,2)/2),:);
I2=bw_img(round(size(bw_img,1)/2)+1:size(bw_img,1),1:round(size(bw_img,2)/2),:)
I3=bw_img(1:round(size(bw_img,1)/2),round(size(bw_img,2)/2)+1:size(bw_img,2),:)
I4=bw_img(round(size(bw_img,1)/2)+1:size(bw_img,1),round(size(bw_img,2)/2)+1:si
ze(bw_img,2),:);
% Proof of image cutting into quadrants
figure;
subplot(2,2,1);
imshow(I1);
subplot(2,2,3);
imshow(I2);
subplot(2,2,2);
imshow(I3);
subplot(2,2,4);
imshow(I4);
```

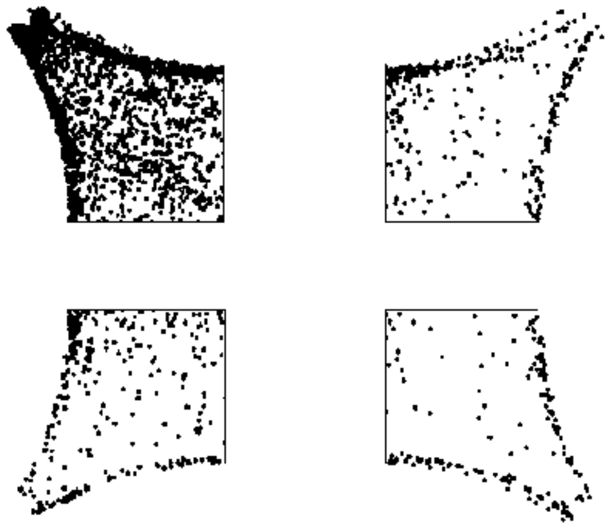


Figure 3.6: Binary Image into Quadrants, Matlab

Calculating the Concentrations

```
% Counting the black pixels at each quadrant
black_num(1) = sum(not(I1(:)));
black_num(2) = sum(not(I2(:)));
black_num(3) = sum(not(I3(:)));
black_num(4) = sum(not(I4(:)));
% Finding the concentrations in percentage
black_per = black_num/sum(black_num)
```

```
black_per =      0.5911    0.1532    0.1603    0.0953
```

3.3 Image Binarization

At this point a basic knowledge about digital image processing is necessary. We will be dealing with terms like pixel format and image histogram so anyone reading must be familiar at least with the representation of a digital image. The Image Binarization process is analyzed and most common Image Thresholding methods are discussed in theoretical approach. The optimum method for our case is decided during the system implementation.

3.3.1 Pixel Formats

The most common pixel format that can hold all the information needed for a colorful picture is based on the RGB color model. The RGB is an additive color model in which red, green, and blue lights are added together in various ways to reproduce a broad array of colors. While the majority of devices use the RGB, color printers are subtractive color devices and use another popular color model the CMYK. As the technology evolves many pixel formats are used for graphic acceleration purposes, sometimes even application specific and hardware based.

A simpler pixel format, which is the one we are dealing with here, is the Grayscale. Grayscale pixels can take up to 256 different values/colors. A Grayscale picture looks like the one that we are used to commonly call as black & white nowadays but is actually composed out of a range of shades of gray without apparent color. The darkest possible shade is black, which is the total absence of transmitted or reflected light. The lightest possible shade is white, the total transmission or reflection of light at all visible wavelengths. So a Grayscale pixel has only one attribute and this is the intensity. As we already mentioned the intensity can take values between 0 and 255.

On the other hand, a Binary image pixel can only take two values, either 0 or 255 and this is usually represented in a binary form where we use 0 for absolute white and 1 for absolute black or vice versa. A couple of other colors/wavelengths can also be used for representation but this is not as common as black & white.

3.3.2 Image Thresholding

Sometimes, like our case it is convenient to transform a Grayscale image into Binary. This is might done in order to isolate objects, like foreground from background, or just for simplicity reasons given that a little to no valuable information is lost during that process. The main logic behind that action is choosing a value between 0 and 255 that is called the threshold. Any pixel with a value smaller than the threshold is changed to 0 and any bigger one to 255 or just 1. This process is called Image Thresholding and it is critical in order to achieve the desired results.

Image Thresholding for some cases it can be real easy while sometimes it can be a greedy process or almost impossible to get good results. So not all Grayscale images are eligible for binarization. The most common practice and also the first thing to consider when attempting to convert a Grayscale image to binary form, is to check out the image histogram.

3.3.3 IMAGE HISTOGRAM

The histogram of a Grayscale image is a figure that shows among the total pixels, how many correspond to a specific intensity (0-255). In other words, how many pixels have intensity with value 10, 21 etc. Imagine a two dimensional figure where in the horizontal axis we have all the possible intensities of a Grayscale image and the vertical axis represents the times that we meet every intensity in the image. If this figure has only two peaks, then the value that should serve as the threshold will be between those two peaks. So if this happens to occur it is very easy and quick for the computer to find the threshold alone and proceed to a good Image Binarization as it happens to our experiment and saves us from a considerable amount of computational power.

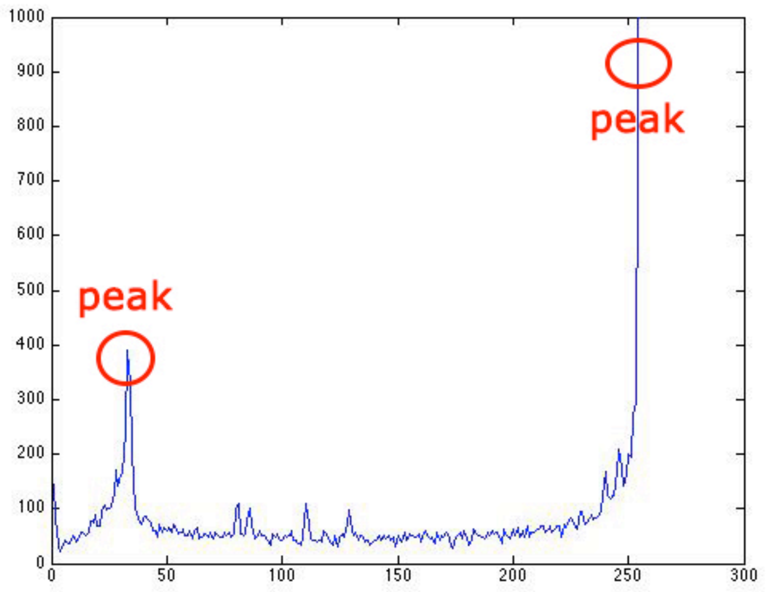


Figure 3.7: Grayscale Image Bimodal Histogram, Matlab

3.3.4 THRESHOLDING METHODS

Generally many methods have been used to achieve binarization to an image without a bimodal histogram and most of them use clustering or entropy models, while other take advantage of local threshold adaptivity. Any of those need additional computational power and usually the histogram of the Grayscale image is an indicator of the proper method. Also someone may develop a new method that fits to needs of the specific application, though today almost any case falls into one of the existing methods. Here we are not going to proceed any deeper analyzing thresholding techniques, as all of the images that we use in our experiment happen to have a bimodal histogram. We should mention though the Otsu's method, as it is deterministically describing how the threshold should be chosen at a bimodal histogram. Later on this chapter and while describing the OpenCV we will see that it is not necessary to implement Otsu's method as it is provided in the form of function within the library. It is good to know though how it works.

3.3.5 THE OTSU METHOD

In Otsu's method we exhaustively search for the threshold that minimizes the intra-class variance (the variance within the class), defined as a weighted sum of variances of the two classes:

$$\sigma_w^2(t) = \omega_1(t)\,\sigma_1^2(t) + \omega_2(t)\,\sigma_2^2(t)$$

Weights w_i are the probabilities of the two classes separated by a threshold t and σ_i^2 are variances of these classes. Otsu shows that minimizing the intra-class variance is the same as maximizing inter-class variance: [173]

$$\sigma_b^2(t) = \sigma^2 - \sigma_w^2(t) = \omega_1(t)\,\omega_2(t)\,[\mu_1(t)\,\mu_2(t)]^2$$

Which is expressed in terms of class probabilities ω_i and class means μ_i.

The class probability $\omega_1(t)$ is computed from the histogram as t:

$$\omega_1(t) = \sum_0^t p(i)$$

While the class mean $\mu_1(t)$ is:

$$\mu_1(t) = \left[\sum_0^t p(i)x(i)\right]/\omega_1$$

Where $x(i)$ is the value at the center of the ith histogram bin. Similarly, you can compute $\omega_2(t)$ and μ_2 on the right-hand side of the histogram for bins greater than t.

The class probabilities and class means can be computed iteratively. This idea yields an effective algorithm:

1. Compute histogram and probabilities of each intensity level.
2. Set up initial $\omega_i(0)$ and $\mu_i(0)$.
3. Step through all possible thresholds $t = 1$ ... maximum intensity
    a. Update ω_i and μ_i
    b. Compute $\sigma_b^2(t)$
4. Desired threshold corresponds to the maximum $\sigma_b^2(t)$
5. You can compute two maxima (and two corresponding thresholds). $\sigma_{b1}^2(t)$ is the greater max and $\sigma_{b2}^2(t)$ is the greater or equal maximum.
6. Desired threshold $= \frac{threshold_1+threshold_2}{2}$

3.3.6 Local Adaptivity

Anyone that would like to go further with the Binarization process may consider a recently developed local adaptive Thresholding method [174]. During the implementation on the next chapter we will notice that the Local Adaptive Thresholding method implemented by the OpenCV library might be a useful as Otsu method in our experiment.

3.4 Hardware Requirements

Given that we have designed the algorithm that solves the problem of insect monitoring in a restricted environment, now we have to choose the necessary equipment before we proceed to our experiments. For sure a camera is needed to capture the video and then a programmable device that will process the data and answer the requested. For the purpose of our application a device equipped with a computational power capable of handling image processing standalone is not necessarily enough. It also has to be able to complete the action in real time.

3.4.1 Real Time Response

Assuming that we need the system output to be renewed every 0.7s in order to be considered responsive, means that it should take not longer than 0.7s to capture a frame from the incoming video, perform an Image Binarization to that frame and then process the binary image as it was described above, while holding the I/O communication with the video camera open.

While meeting the computational power needs is the first device choosing criterion the cost as well as the availability should be also considered. Last but not least we should keep in mind that for the purpose of this thesis we have significantly simplify the experimental process, so choosing a more powerful device if this is not a huge trade of to the cost might be a smart move for future expansions. For instance, repeating the experiment with a real Olfactometer, the same device that handles the image processing would be desired to be able to handle a set of electronic valves at the inputs of the Olfactometer in order our system to answer broader questions.

3.4.2 Microprocessors

The term microprocessor and microcontroller have always been confused with each other. Both of them have been designed for real time applications. They share many common features and at the same time they have significant differences. Both the IC’s i.e., the microprocessor and microcontroller cannot be distinguished by looking at them. They are available in different version starting from 6 pins to as high as 80 to 100 pins or even higher depending on the features.

Microprocessor is an IC, which has only the CPU inside them i.e. only the processing powers such as Intel’s Pentium 1,2,3,4, core 2 duo, i3, i5 etc. These microprocessors don’t have RAM, ROM, and other peripheral on the chip. A system designer has to add them externally to make them functional. Application of microprocessor includes Desktop PC’s, Laptops, notepads etc. Microprocessors find applications where tasks are unspecific like developing software, games, websites, photo editing, creating documents etc. In such cases the relationship between input and output is not defined. They need high amount of resources like RAM, ROM, I/O ports etc. The clock speed of the Microprocessor is quite high as compared to the microcontroller. Whereas the microcontrollers operate from a few MHz to 1GHz, today’s microprocessors operate above 1GHz as they perform complex tasks.

3.4.3 MICROCONTROLLERS

This is not the case with Microcontrollers. Microcontroller has a CPU, in addition with a fixed amount of RAM, ROM and other peripherals all embedded on a single chip. At times it is also termed as a mini computer or a computer on a single chip. Today different manufacturers produce microcontrollers with a wide range of features available in different versions. Some manufacturers are ATMEL, Microchip, TI, Freescale, Philips, Motorola etc. Microcontrollers are designed to perform specific tasks. Specific means applications where the relationship of input and output is defined. Depending on the input, some processing needs to be done and output is delivered. For example, keyboards, mouse, washing machine, digicam, pendrive, remote, microwave, cars, bikes, telephone, mobiles, watches, etc. Since the applications are very specific, they need small resources like RAM, ROM, I/O ports etc. and hence can be embedded on a single chip. This in turn reduces the size and the cost.

3.4.4 The Cost Factor & FPGAs

Comparing microcontroller and microprocessor in terms of cost is not justified. Undoubtedly a microcontroller is far cheaper than a microprocessor. However microcontroller cannot be used in place of microprocessor and using a microprocessor is not advised in place of a microcontroller as it makes the application quite costly. Microprocessor cannot be used stand-alone. They need other peripherals like RAM, ROM, buffer, I/O ports etc. and hence a system designed around a microprocessor is quite costly.

Somewhere between the microprocessors and microcontrollers come the FPGAs. Field Programmable Gate Arrays or FPGAs were once simple blocks of gates that can be configured by the user to implement the logic that he or she wants. In comparison, a microprocessor is a simplified CPU or Central Processing Unit. It executes a program that contains a specific set of instructions. The main difference between FPGAs and microprocessors is the complexity. Although both vary in complexity depending on the scale, microprocessors tend to be more complex than FPGAs. FPGA as an IC can contain RAM or other peripheral is generally much more expensive and powerful than a microcontroller but those two (FPGAs and microcontrollers) nowadays vary significantly. You can find microcontrollers that are more powerful than an FPGA and vice versa.

3.4.5 THE BEAGLEBONE BLACK

The revolutionary microcontrollers, BeagleBone & Raspberry Pi, came on the market and changed the cost-capability curve. Those low cost powerful devices are able to run from Android to Linux distribution and they do not transcend in size a normal cell phone. It is more like a pocket PC or the hacker's paradise as they use to be called. The best thing about them is the price witch is usually lower that 70€.

It is obvious that for the purposes of our experiment one of those devices would be ideal. Provided that we have one BeagleBone Black available there is nothing left than trying it! BeagleBone Black is equipped with:

- AM335x 1GHz ARM® Cortex-A8 processor
- 512 DDR3 RAM
- 4GB 8-bit eMMC on-board flash storage
- 3D graphics accelerator
- Neon floating-point accelerator
- 2x PRU 32-bit microcontrollers

It can run Debian, Android, Ubuntu, Cloud9 IDE on Node.js w/ Bonescript library and more. Also supports connectivity with USB client, USB host, Ethernet, HDMI and 2x 46 pin headers. Prima facie seems enough, remains to be seen.

Chapter 4
System Implementation

Abstract

In this chapter we are introducing the BeagleBone Black (from now and on mentioned as BBB) device and using it to develop our system. Basic operations in order to use BBB are discussed, as well as the tools that cooperate with BBB and will help us with the implementation. The way of programming, compiling and debugging on BBB is explained and a cross compiler & remote debugger is proposed. We follow with the system setup and configuration before we present the experimental process. The code that is implementing the algorithm, the way that it is used to perform the experiments, the deduced results and the mathematical models are discussed in detail. After all there is a discussion about the possible application and future extensions of the system.

4.1 Getting Started with BeagleBone Black

We are going to be using a BBB flashed with a Linux Angstrom distribution. In order to use it we can either attach an HDMI available screen, a mouse and a keyboard or just connect the device to the local network via Ethernet and then use telnet, ssh and ftp in order to handle and communicate with the device. The first choice might seem much more handy especially to anyone that is not familiar with the ssh & ftp protocols but for our experiment is not even an option as we do not want to burden BBB's CPU handling peripherals devices. There is still the option of using the USB cable that comes with the BBB to connect to it or even configuring it to be reachable via internet (outside the local network), but here we will stick with the local network configuration. So we connect the Ethernet and then the power cable. This will make the device to boot. Booting process should last about 10 seconds.

4.1.1 Setting a Static IP Address

Now that the BBB is on and running first thing we need to do is to find out the IP address of the device. This can be done in many ways, like checking the gateway, using a freeware application, that lists all the network devices, on our computer that is also connected to the same network with the BBB or just use the “arp” command through terminal (we assume that the computer that we are using to handle the BBB is running a UNIX based operating system). After finding out the BBB’s IP address, we use that to connect as “root” to the device using ssh. The default root password for a BBB flashed with an Angstrom Linux distribution is either “beaglebone” or no password at all. We might change that if we want but we are not going to describe this case any further here, as there is plenty of documentation on the net for anyone interested to. After logging into the device as root we need to navigate to the directory “/usr/lib/connman/test” where we will find the available binaries that safely do the network configuration. Note that the basic command set for Angstrom is the same as any other UNIX based system. Setting up a static IP for BBB here is done by running the binaries “set-nameserves” and “set-ipv4-method” which you will find under the current directory (usr/lic/connman/test). Running the binaries without properties will return some usage information and for anyone that still does not feel familiar with that process, there is a step-by-step guide [175].

Another optional action to consider, associated with the above task, is adding the BBB's IP to the hosts file of the machine that is used to handle the device. This will save us from remembering and typing this address every time we want to reference to it. We can use the word "beaglebone" for instance. So instead of typing "ssh root@192.168.1.xxx" it will be "ssh root@beaglebone".

4.1.2 Available Libraries

So now that we are connected to the BBB, we need to know what libraries come pre-installed with Angstrom, what is available for installation and mainly which programming language is better supported in order to use it for our system's algorithm implementation. With a brief web search someone would shortly crash on the openCV (CV stands for Computer Vision), which is ideal library for our purpose, well documented on the web and Linux Angstrom supported. The functions provided by OpenCV could be an integral part of our algorithm implementation, covering the entire image processing part of it. It is also a programming language and machine independent library. We still need though a camera driver and maybe video codec software for our BBB before we capture the video and eventually the frame to process.

4.1.3 USB Camera Drivers

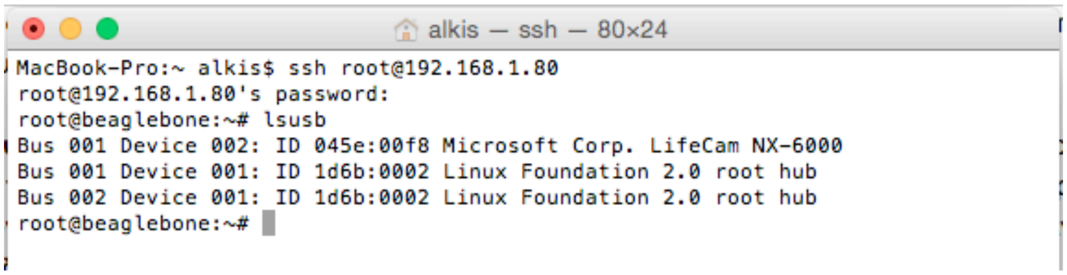
The USB camera to work with BBB needs support for the connection and support for the actual camera hardware. Camera drivers on Linux devices are usually available in one of these three ways: within the kernel, as a compliable stand-alone module, or available as a pre-compiled (packaged) binary driver from the Linux distribution. Angstrom Linux distribution in our case comes with "Video For Linux", in short "V4L" pre-installed. V4L is an API that allows control of a video capture card (camera) on Linux machines. A card-specific driver controls a video capture card and offers a semistandard interface to the system. V4L is an implementation that uses this interface to add additional features, and in turn, provides an API of its own. V4L provides a uniform API for a variety of capture cards, which can be TV cards, Radio cards, or cards used just to capture images from a camera. In our case V4L and the driver for the camera are already installed on a the machine, so as application developers we can use V4L's API (details are described later on this chapter), without knowing much about the camera hardware or it's driver.

4.1.4 Video Codex Software

Another issue that comes up as we are handling a video capturing device is the video encoding framework. The capturing device driver is sending us video in raw format, which means no header and no compression if we where about to save this video to a file. For the purposes of this thesis we are not going to keep/save the capturing video. We only need to process some frames of it and getting those frames from the raw video is what we will do, as this is simpler and will save our device's CPU processing power from any encoding and decoding process. Worth mentioning though that if in future expansions of this project the captured video should be stored, encoding it would be necessary. Raw format video takes big amount of storing space and is inaccessible. For that purpose we would use the "FFMPEG codec". FFMPEG is the leading multimedia framework, able to decode, encode, transcode, mux, demux, stream, filter and play pretty much anything that humans and machines have created. It supports the most obscure ancient formats up to the cutting edge. Building and using the FFMPEG on BBB is trivial and very well documented on the web [176].

4.1.5 CAMERA CONNECTION CHECK

Now that we already know about the average tips that will contribute building the system and before we start to use the necessary API and libraries we have to make sure that the camera we have already connected is recognized by the BBB. It is a Microsoft LifeCam NX-6000 web camera connected via USB. To do that, been logged into BBB as root, we type the command "lsusb" and that should list all the available USB devices. As it is shown in the picture below our camera is listed and we are ready to go on.



```
MacBook-Pro:~ alkis$ ssh root@192.168.1.80
root@192.168.1.80's password:
root@beaglebone:~# lsusb
Bus 001 Device 002: ID 045e:00f8 Microsoft Corp. LifeCam NX-6000
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
root@beaglebone:~#
```

Figure 4.1: USB devices listing on BBB

We should also be able to see our camera as video device under "/dev" directory named as "video0". In that case the Microsoft camera is the default video device and we can reference to it as "/dev/video0".

4.2 Compiling, Building & Debugging for BBB

Compiling, building and debugging on Linux are GNU Toolchain jobs. The Angstrom Linux distribution on our BBB comes with a complete pre-installed GNU Toolchain. Generally, installing and building the GNU Toolchain under Linux is a trivial task and always included in the distribution's package manager at least.

As a microcontroller running Linux, the BBB is able to get through the above tasks. Developing though a large-scale application with the command line is not always convenient. Command line text editors are not so handy and debugging becomes even more painful. Also due to the bounded processing power and memory of devices such BBB, compiling and building can be a very time consuming process. Considering Cross Compiling & Remote Debugging in that cases might be a wise option.

4.2.1 THE GNU TOOLCHAIN

The GNU Toolchain is a broad collection of programming tools produced by the GNU Project. These tools form a toolchain (a suite of tools used in a serial manner) used for developing software applications and operating systems. The GNU Toolchain is free and includes the following projects:

- GNU make: Automation tool for compilation and build
- GNU Compiler Collection (GCC): Suite of compilers for several programming languages
- GNU Binutils: Suite of tools including linker, assembler and other tools
- GNU Bison: Parser generator
- GNU m4: m4 macro processor
- GNU Debugger (GDB): Code debugging tool
- GNU build system (autotools):

GCC, make and Binutils are necessary for developing any application and many will also find the GDB very useful. Those four projects are the tools that we will be using during our system implementation and anyone that would like to follow on the chapter should know about them and have at least used them once before.

4.2.2 Cross compiler

A cross compiler is a compiler capable of creating executable code for a platform other than the one on which the compiler is running. For example, a compiler that runs on a Windows 7 PC but generates code that runs on Android smartphone is a cross compiler. A cross compiler is necessary to compile for multiple platforms from one machine. This is necessary for a platform that is infeasible for a compiler to run on, such as for the microcontroller of an embedded system because those systems contain no operating system. It is also useful for platforms that even they contain an operating system that is theoretically capable of running a compiler, when it comes to action it is actually taking too much time to compile.

4.2.3 UNIX Toolchain Naming Convention

In order to find the suitable toolchain is not only the target processor that we must care about. Also working in a development environment with multiple compilers, means that choosing the write cross compiler every time is a basic precondition. There is a UNIX naming convention for the toolchains referring to the architecture, the vendor, the operating system and the application binary interface. The name should be like [arch]-[vendor]-[os]-[abi]. First comes the architecture of the processor. Are we talking about ARMs, PowerPCs, Intels? This field is mandatory, so you will find it in every toolchain naming. Second, we have the vendor as an optional field. Could be intel, apple, etc. Sometimes, to be explicit, you will also find a -none- within this field, which means that the vendor is not given. Third comes the operating system, again as an optional field. This point also defines what libraries you will have to use in your toolchain – in our case we should use the glibc. If we are compiling for an embedded system without an operating system (bare metal as it is used to be called) then we should use the toolchain named with –none- at this point. Fourth, the type of the application binary interface is given. It defines the low-level interface between your program or functions of the operating system or other programs. Conventions about data types, the calling conventions of functions or to operating system are defined. In our case, we will use the EABI toolchain, the embedded-abi reasonable for PowerPC, ARM and MIPS. So in order to cross compile for BBB we need a toolchain named after “arm-non-linux-eabi”.

4.2.4 INSTALLING & BUILDING THE TOOLCHAIN

Installing the toolchain is not always an easy task. If we want to build the cross compiler on Ubuntu for example it is more likely that we can find the toolchain precompiled as a package on the net. Getting and building the toolchain in that case can be a few minutes process using any of the many step-by-step guides found online. This is not the case if we want to build the same cross compiler on Mac OS though. The configuration here has to be done by hand. To built a cross compiler for ARM on a mac, you must either have an excellent knowledge about compilers or go through a very time consuming process that includes repeated trial and error actions. During this thesis, we did build a cross compiler on a Mac, used it by the Eclipse IDE and linked a remote debugger with the BBB as well. That worked a few times and then for an unknown reason the cross compiler could not find some libraries and resetting the environmental variables did not solved that problem. So we cannot advise anyone to do so, as in the one hand we are not in a position to deterministically describe the process, on the other hand it takes more time than it will save you from waiting. It is preferable for Mac users to create a Virtual Machine with a Linux distribution and install the toolchain for BBB on that.

4.2.5 USING AN INTEGRATED DEVELOPMENT ENVIRONMENT

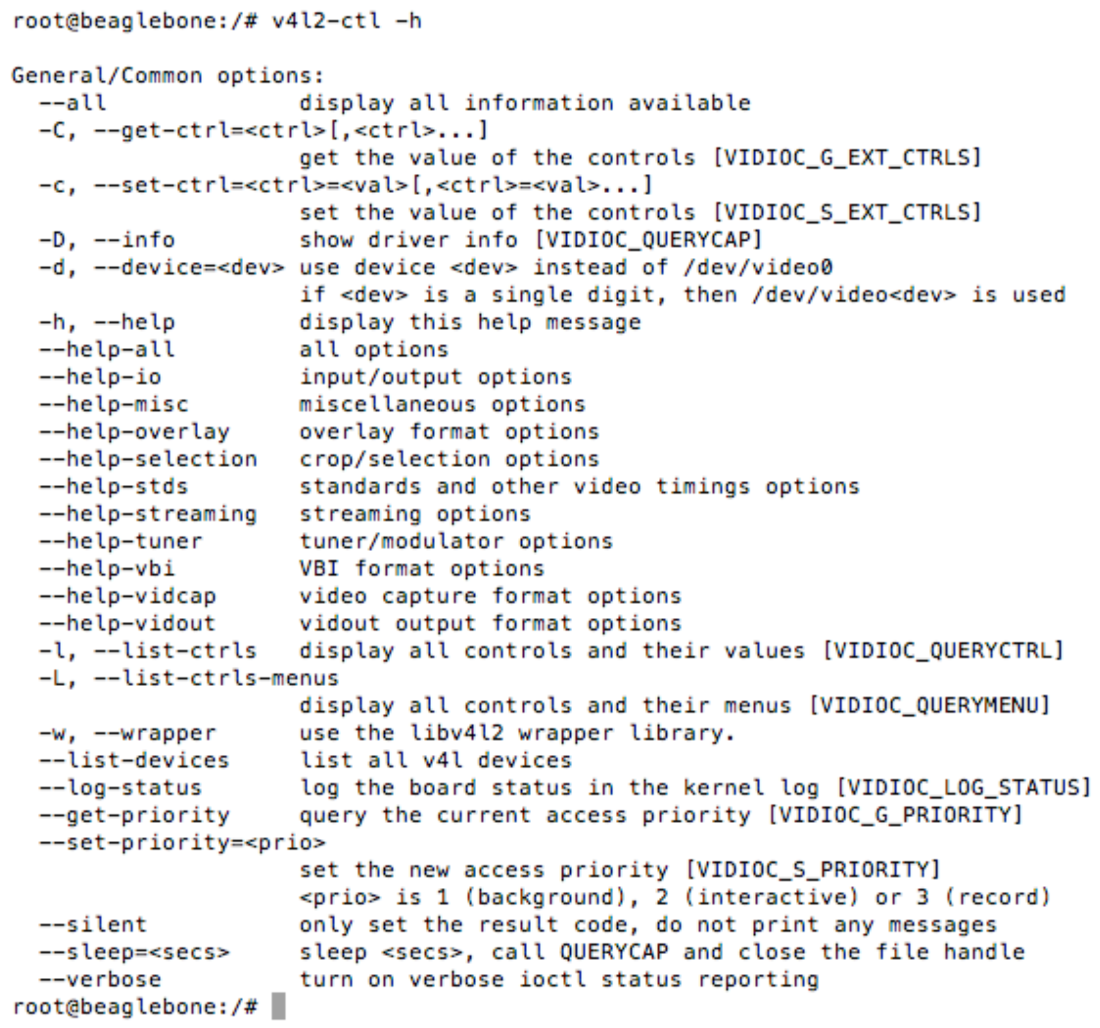
Anyone that will build a Cross Compiler for BBB should also consider linking an Integrated Development Environment (IDE) with it. That process falls up to the IDE settings and its ability to use another toolchain other than the local. In order to link an IDE with a Cross Compiler we must replace the binaries with those that correspond to our toolchain of preference and also set the paths to those and the libraries. A good choice for that purpose is an Eclipse IDE. It is a freeware piece of software with great extensions support. Most programming languages are supported for development with Eclipse and the Cross Compiler can be easily set up for use through its graphical interface.

4.2.6 REMOTE DEBUGGING & MORE

After we have successfully developed and compiled our project for BBB, the Eclipse IDE gives us the opportunity to automatically transfer the binary to the BBB file system as well as running it even in debug mode. Adding the Remote Tools extension to the Eclipse IDE and configuring an FTP & SSH connection with our BBB we are able to follow the debugging process of our project on Eclipse while this is running on BBB. This is Remote Debugging and having that configured we get an open FPT & SHH connection with our device from a graphical interface in the same time. Sounds handy, right? There is a step-by-step guide on how to configure whatever was discussed at this subsection on a machine that is running Debian [177]. The procedure is pretty much the same for other Linux distributions as well.

4.3 Using the V4L2 API

The V4L2 API is acting as the camera driver for us and gives us the opportunity to handle the video capturing options. V4L2 comes preinstalled with Angstrom and the command “v4l2-ctl” gives us control over the video devices. For an overview of the available options and how to use them we can type “v4l2-ctl –h”.



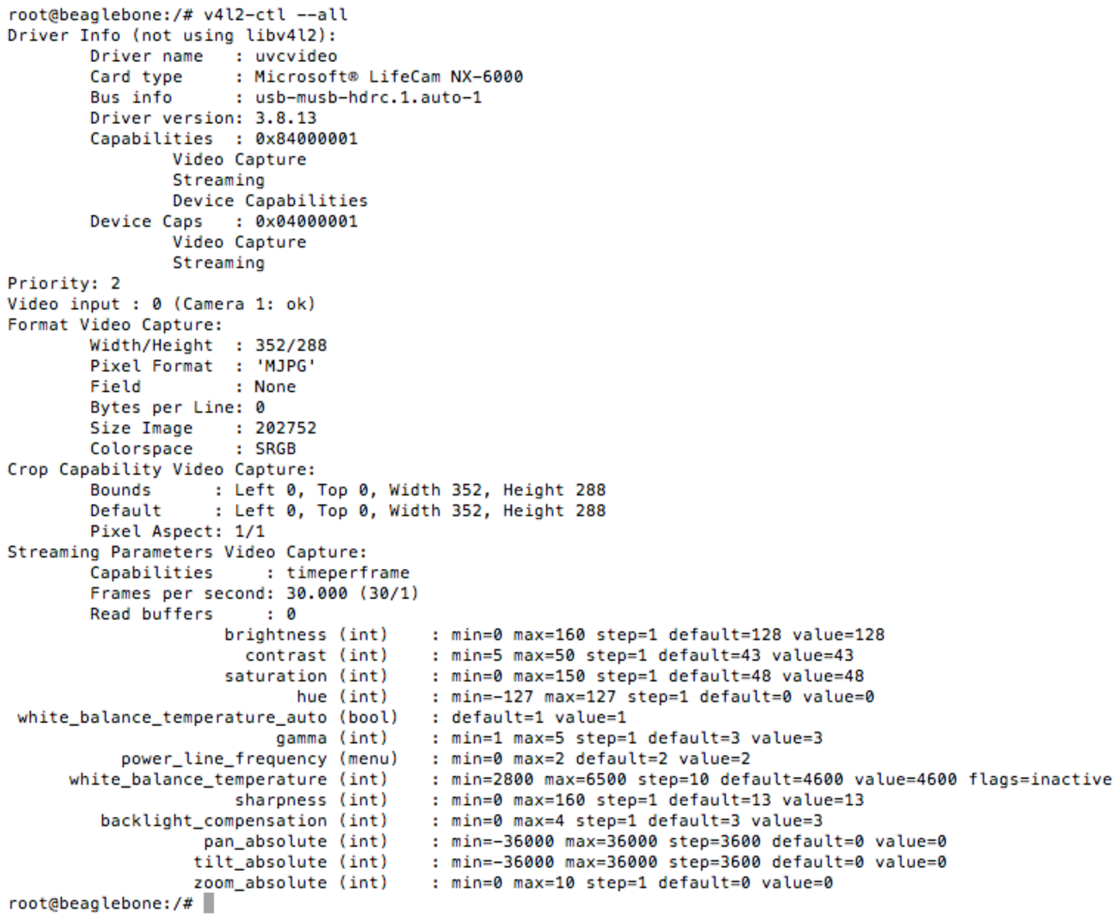
```
root@beaglebone:/# v4l2-ctl -h

General/Common options:
  --all              display all information available
  -C, --get-ctrl=<ctrl>[,<ctrl>...]
                     get the value of the controls [VIDIOC_G_EXT_CTRLS]
  -c, --set-ctrl=<ctrl>=<val>[,<ctrl>=<val>...]
                     set the value of the controls [VIDIOC_S_EXT_CTRLS]
  -D, --info         show driver info [VIDIOC_QUERYCAP]
  -d, --device=<dev> use device <dev> instead of /dev/video0
                     if <dev> is a single digit, then /dev/video<dev> is used
  -h, --help         display this help message
  --help-all         all options
  --help-io          input/output options
  --help-misc        miscellaneous options
  --help-overlay     overlay format options
  --help-selection   crop/selection options
  --help-stds        standards and other video timings options
  --help-streaming   streaming options
  --help-tuner       tuner/modulator options
  --help-vbi         VBI format options
  --help-vidcap      video capture format options
  --help-vidout      vidout output format options
  -l, --list-ctrls   display all controls and their values [VIDIOC_QUERYCTRL]
  -L, --list-ctrls-menus
                     display all controls and their menus [VIDIOC_QUERYMENU]
  -w, --wrapper      use the libv4l2 wrapper library.
  --list-devices     list all v4l devices
  --log-status       log the board status in the kernel log [VIDIOC_LOG_STATUS]
  --get-priority     query the current access priority [VIDIOC_G_PRIORITY]
  --set-priority=<prio>
                     set the new access priority [VIDIOC_S_PRIORITY]
                     <prio> is 1 (background), 2 (interactive) or 3 (record)
  --silent           only set the result code, do not print any messages
  --sleep=<secs>     sleep <secs>, call QUERYCAP and close the file handle
  --verbose          turn on verbose ioctl status reporting
root@beaglebone:/#
```

Figure 4.2: Set of available V4L2 options

4.3.1 Camera Specifications & Settings

Using the option “--list-devices” should show the model of our camera as the “/dev/video0” device, assuming that is the only video device connected to our BBB. With the option “--all” we should see something like the following.



```
root@beaglebone:/# v4l2-ctl --all
Driver Info (not using libv4l2):
        Driver name   : uvcvideo
        Card type     : Microsoft® LifeCam NX-6000
        Bus info      : usb-musb-hdrc.1.auto-1
        Driver version: 3.8.13
        Capabilities  : 0x84000001
                Video Capture
                Streaming
                Device Capabilities
        Device Caps   : 0x04000001
                Video Capture
                Streaming
Priority: 2
Video input : 0 (Camera 1: ok)
Format Video Capture:
        Width/Height  : 352/288
        Pixel Format  : 'MJPG'
        Field         : None
        Bytes per Line: 0
        Size Image    : 202752
        Colorspace    : SRGB
Crop Capability Video Capture:
        Bounds      : Left 0, Top 0, Width 352, Height 288
        Default     : Left 0, Top 0, Width 352, Height 288
        Pixel Aspect: 1/1
Streaming Parameters Video Capture:
        Capabilities     : timeperframe
        Frames per second: 30.000 (30/1)
        Read buffers     : 0
                     brightness (int)    : min=0 max=160 step=1 default=128 value=128
                       contrast (int)    : min=5 max=50 step=1 default=43 value=43
                     saturation (int)    : min=0 max=150 step=1 default=48 value=48
                            hue (int)    : min=-127 max=127 step=1 default=0 value=0
 white_balance_temperature_auto (bool)   : default=1 value=1
                          gamma (int)    : min=1 max=5 step=1 default=3 value=3
           power_line_frequency (menu)   : min=0 max=2 default=2 value=2
      white_balance_temperature (int)    : min=2800 max=6500 step=10 default=4600 value=4600 flags=inactive
                      sharpness (int)    : min=0 max=160 step=1 default=13 value=13
         backlight_compensation (int)    : min=0 max=4 step=1 default=3 value=3
                   pan_absolute (int)    : min=-36000 max=36000 step=3600 default=0 value=0
                  tilt_absolute (int)    : min=-36000 max=36000 step=3600 default=0 value=0
                  zoom_absolute (int)    : min=0 max=10 step=1 default=0 value=0
root@beaglebone:/#
```

Figure 4.3: Camera Specs & Current Settings

Above we see an overview of our camera specifications, the driver’s capabilities and the current video settings. We can edit those settings by using the “--set-ctrl-…”

option as described at Figure 8. Those settings are universal and will be applied if we do not use the “libv4l2” within our code.

4.3.2 The libv4l2 Library & Grab Example

Using functions provided in the “libv4l2” library can overwrite universal settings. Thanks to Mauro Carvalho Chehab at the Appendix E of V4L’s documentation we can find a solid piece of c code [178], which is an example of how to grab a picture using the “libv4l”. It is a good example to study in order to get a first impression of how to use “libv4l” but with a little bit of modification it will also be useful to us, as we will use it to grab pictures during the camera setup. Those pictures will act as the proof of the correct camera placement. Indisputably this code saves us from a great amount of work. In order to use it though, we must first understand it and then make the necessary modifications.

4.3.3 Modification of Grab Example

First of all we must set the path to “libv4l” on BBB at the corresponding #include statement, which is not defined. Then we might also want to change some values of the V4L2 capturing settings like the dimensions and maybe the pixel format too. Here we should note that changing the dimensions would not force the driver to send us an image at the exact numbers that we set, but at the closest possible resolution according to the width dimension. A last modification should be done at the number of the captured frames. This example is grabbing 20 pictures with an interval of 2 seconds, but we only need a single picture. Executing the modified code at the BBB will save at the same directory with the executable an uncompressed image that shows us, what the camera’s view looks like. Below we see the modified example from linuxtv.org and the results of that code. We can always play with the driver’s control values for better results.

4.3.4 Picture Grabbing Modified Code & Results

```
/* V4L2 video picture grabber
   Copyright (C) 2009 Mauro Carvalho Chehab <mchehab@infradead.org>
   This program is free software; you can redistribute it and/or modify
   it under the terms of the GNU General Public License as published by
   the Free Software Foundation version 2 of the License.

   This program is distributed in the hope that it will be useful,
   but WITHOUT ANY WARRANTY; without even the implied warranty of
   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
   GNU General Public License for more details.

   This copy have been edited by Alkiviadis Klotsonis
   in order to work with BeagleBone Black
   and a Microsoft LifeCam NX-6000
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <fcntl.h>
#include <errno.h>
#include <sys/ioctl.h>
#include <sys/types.h>
#include <sys/time.h>
#include <sys/mman.h>
#include <linux/videodev2.h>
#include <libv4l2.h>

#define CLEAR(x) memset(&(x), 0, sizeof(x))

struct buffer {
        void   *start;
        size_t length;
};

static void xioctl(int fh, int request, void *arg)
{
        int r;

        do {
                r = v4l2_ioctl(fh, request, arg);
        } while (r == -1 && ((errno == EINTR) || (errno == EAGAIN)));

        if (r == -1) {
                fprintf(stderr, "error %d, %s\n", errno, strerror(errno));
                exit(EXIT_FAILURE);
        }
}

int main(int argc, char **argv)
{
        struct v4l2_format              fmt;
        struct v4l2_buffer              buf;
        struct v4l2_requestbuffers      req;
        enum v4l2_buf_type              type;
        fd_set                          fds;
```

```
struct timeval                      tv;
int                                 r, fd = -1;
unsigned int                        i, n_buffers;
char                                *dev_name = "/dev/video0";
char                                out_name[256];
FILE                                *fout;
struct buffer                       *buffers;

fd = v4l2_open(dev_name, O_RDWR | O_NONBLOCK, 0);
if (fd < 0) {
        perror("Cannot open device");
        exit(EXIT_FAILURE);
}

CLEAR(fmt);
fmt.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
fmt.fmt.pix.width       = 640;
fmt.fmt.pix.height      = 480;
fmt.fmt.pix.pixelformat = V4L2_PIX_FMT_RGB24;
fmt.fmt.pix.field       = V4L2_FIELD_INTERLACED;
xioctl(fd, VIDIOC_S_FMT, &fmt);
if (fmt.fmt.pix.pixelformat != V4L2_PIX_FMT_RGB24) {
        printf("Libv4l didn't accept RGB24 format. Can't proceed.\n");
        exit(EXIT_FAILURE);
}
if ((fmt.fmt.pix.width != 640) || (fmt.fmt.pix.height != 480))
        printf("Warning: driver is sending image at %dx%d\n",
                fmt.fmt.pix.width, fmt.fmt.pix.height);

CLEAR(req);
req.count = 2;
req.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
req.memory = V4L2_MEMORY_MMAP;
xioctl(fd, VIDIOC_REQBUFS, &req);

buffers = calloc(req.count, sizeof(*buffers));
for (n_buffers = 0; n_buffers < req.count; ++n_buffers) {
        CLEAR(buf);

        buf.type        = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        buf.memory      = V4L2_MEMORY_MMAP;
        buf.index       = n_buffers;

        xioctl(fd, VIDIOC_QUERYBUF, &buf);

        buffers[n_buffers].length = buf.length;
        buffers[n_buffers].start = v4l2_mmap(NULL, buf.length,
                      PROT_READ | PROT_WRITE, MAP_SHARED,
                      fd, buf.m.offset);

        if (MAP_FAILED == buffers[n_buffers].start) {
                perror("mmap");
                exit(EXIT_FAILURE);
        }
}
```

```
        for (i = 0; i < n_buffers; ++i) {
                CLEAR(buf);
                buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
                buf.memory = V4L2_MEMORY_MMAP;
                buf.index = i;
                xioctl(fd, VIDIOC_QBUF, &buf);
        }
        type = V4L2_BUF_TYPE_VIDEO_CAPTURE;

        xioctl(fd, VIDIOC_STREAMON, &type);

      do {
               FD_ZERO(&fds);
               FD_SET(fd, &fds);

               /* Timeout. */
               tv.tv_sec = 2;
               tv.tv_usec = 0;

               r = select(fd + 1, &fds, NULL, NULL, &tv);
        } while ((r == -1 && (errno = EINTR)));
        if (r == -1) {
               perror("select");
               return errno;
        }

        CLEAR(buf);
        buf.type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        buf.memory = V4L2_MEMORY_MMAP;
        xioctl(fd, VIDIOC_DQBUF, &buf);

        sprintf(out_name, "out.ppm");
        fout = fopen(out_name, "w");
        if (!fout) {
               perror("Cannot open image");
               exit(EXIT_FAILURE);
        }
        fprintf(fout, "P6\n%d %d 255\n",
        fmt.fmt.pix.width, fmt.fmt.pix.height);
        fwrite(buffers[buf.index].start, buf.bytesused, 1, fout);
             fclose(fout);

        xioctl(fd, VIDIOC_QBUF, &buf);

        type = V4L2_BUF_TYPE_VIDEO_CAPTURE;
        xioctl(fd, VIDIOC_STREAMOFF, &type);
        for (i = 0; i < n_buffers; ++i)
                v4l2_munmap(buffers[i].start, buffers[i].length);
        v4l2_close(fd);
        return 0;
}
```

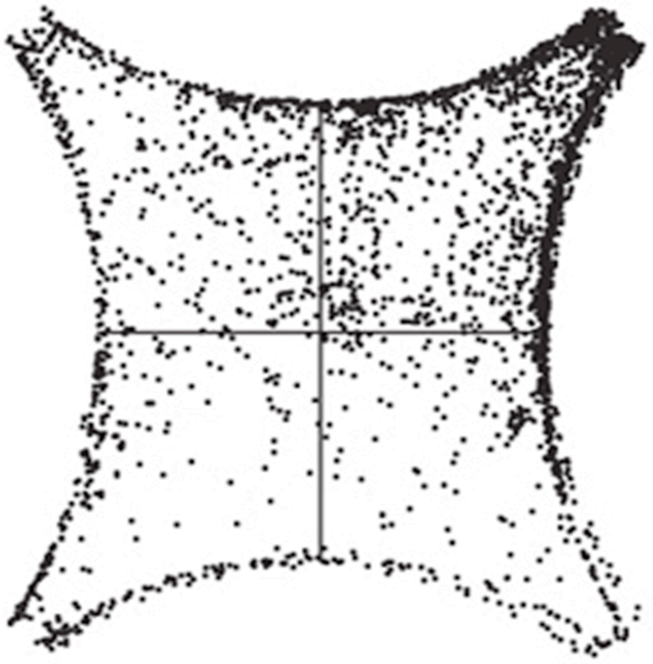


Figure 4.4: The image that was printed

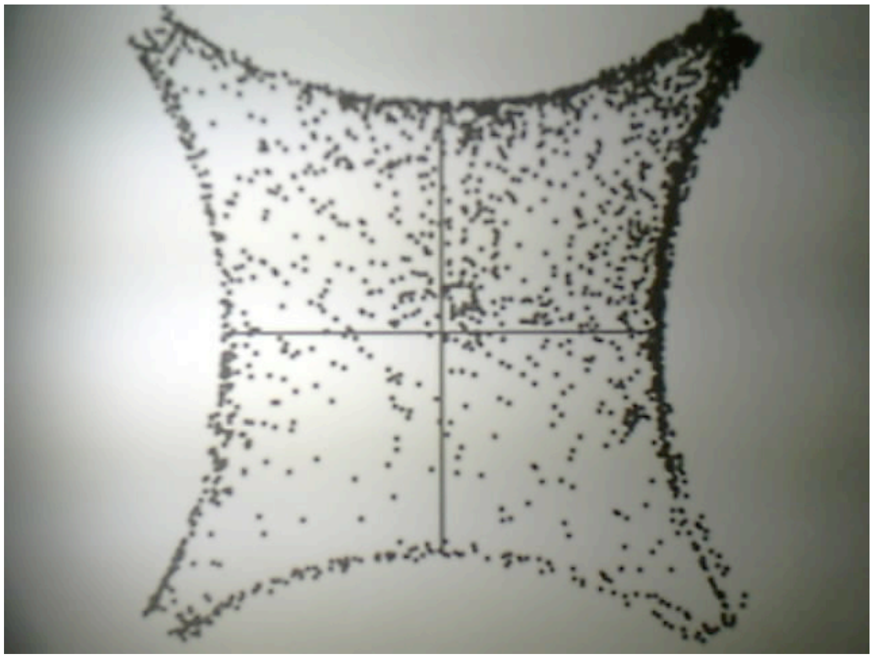


Figure 4.5: Printed image as it was captured

The V4L2 API except from helping us the way it was shown above is also a necessary precondition for the OpenCV library to work correctly. Now lets see what the OpenCV library have to offer and why we need it for our project.

4.4 The OpenCV Library

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community. The library is used extensively in companies, research groups and by governmental bodies.

4.4.1 APPLICATION RANGE

OpenCV's deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New

York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan. It is obviously a flawless solid library that we could use doubtless in any application.

4.4.2 Supported Environments & BeagleBone Black

It has C++, C, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a templated interface that works seamlessly with STL containers. There is a full documentation of the library online [179].

The OpenCV library implemented on C++ comes preinstalled at a Beaglebone Black with an Angstrom Linux distribution. In each case that it doesn't, there is at the package manager so you can easily install it from there. Remember that is always a good practice to know what is installed before applying any actions.

4.4.3 Usage of OpenCV

OpenCV can do most of the work to our experiment. From basic image operations [180] to tasks like video capturing [181], frame processing and Image Miscellaneous Transformations [182] (more attention should be given at the functions "threshold", "addaptiveThreshold" and "ctvColor"), can be done by OpenCV library. It can also help us process the region of interest and not the entire frame [183] that is captured by the camera and sent to us by the driver. On support of this, working with C & C++ on any UNIX based system is always a good idea. Except from being able to do almost everything with those two programming languages, there are plenty of libraries implemented on those especially for Linux. Thus we will use C++ and the corresponding OpenCV implementation to develop our system.

4.5 Experiments with BeagleBone Black

So we have a Beaglebone Black with an Angstrom Linux distribution, we have connected a Microsoft LifeCam NX-6000, both the V4L2 API and the OpenCV library are working flawlessly with it and we are ready to implement the algorithm we have already designed. We found convenient to proceed in the implementation with C++ and we have already studied and presented most of the tool/libraries that we are going to use.

4.5.1 Comparing Thresholding Methods

Before proceeding we need to approve the thresholding method that we will be using for the Image Binarization. The method that we will choose here might not be the most suitable when the camera captures the image. The lighting conditions then, should be taken into account, as we will later on this chapter see.

For the sake of the experiments we have 18 different instances of the Olfactometer with distinct insects placement in that (Olfactometer Instances). We choose randomly one instance/image; we load it from the file system and apply Image Binarization to it with the four candidate thresholding methods. We choose the best binary image based on the optical results (Figure 4.7) compared to the original (Figure 4.6). Then we apply the thresholding method that we chose at the previous step to all the 18 images/instances and record the concentrations as shown in Table 4.1.

```
/* allBinMethods.cpp
 *
 * Klotsonis Alkiviadis
 * School of Electronics & Computer Engineering
 * Technical University of Crete 2015
 *
 * Redistribution is allowed provided that it will
 * retain this notice
 *
 * This program loads an image from file as Grayscale
 * and then performs 4 different thresholding methods
 * to it, generating and saving to the file system 4
 * discrete Binary Images.
 */

#include<iostream>
#include<opencv2/opencv.hpp>
using namespace std;
using namespace cv;

int main()
{
    Mat frame, bin_otsu, bin_man, bin_ad_mean_c, bin_ad_gaus_c;
    frame = imread("figure.jpg",CV_LOAD_IMAGE_GRAYSCALE);
    if(frame.empty()){
      cout << "Error loading the image" << endl;
      return -1;
    }

    threshold(frame,bin_num,180,255,CV_THRESH_BINARY);
    threshold(frame,bin_otsu,0,255,CV_THRESH_BINARY |
CV_THRESH_OTSU);

adaptiveThreshold(frame,bin_ad_mean_c,255,ADAPTIVE_THRESH_MEAN_C,THR
ESH_BINARY,61,5);

adaptiveThreshold(frame,bin_ad_gaus_c,255,ADAPTIVE_THRESH_GAUSSIAN_C
,THRESH_BINARY,61,5);

    cout << "Finished" << endl;

    imwrite("manual.jpg", bin_man);
    imwrite("otsu.jpg", bin_otsu);
    imwrite("meanC.jpg", bin_ad_mean_c);
    imwrite("gaussianC.jpg", bin_ad_gaus_c);
    return 0;
}
```

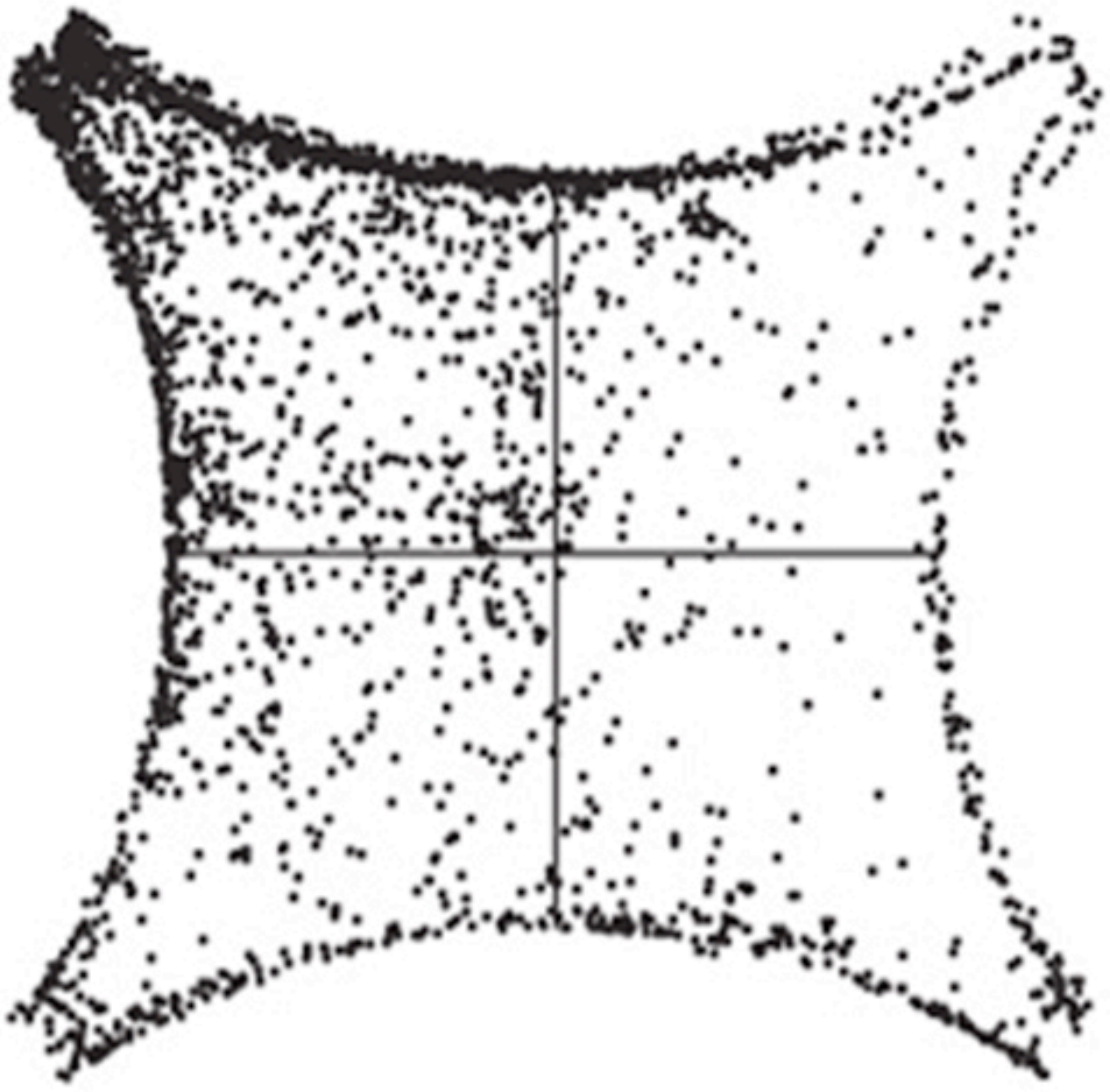


Figure 4.6: RGB Random Image

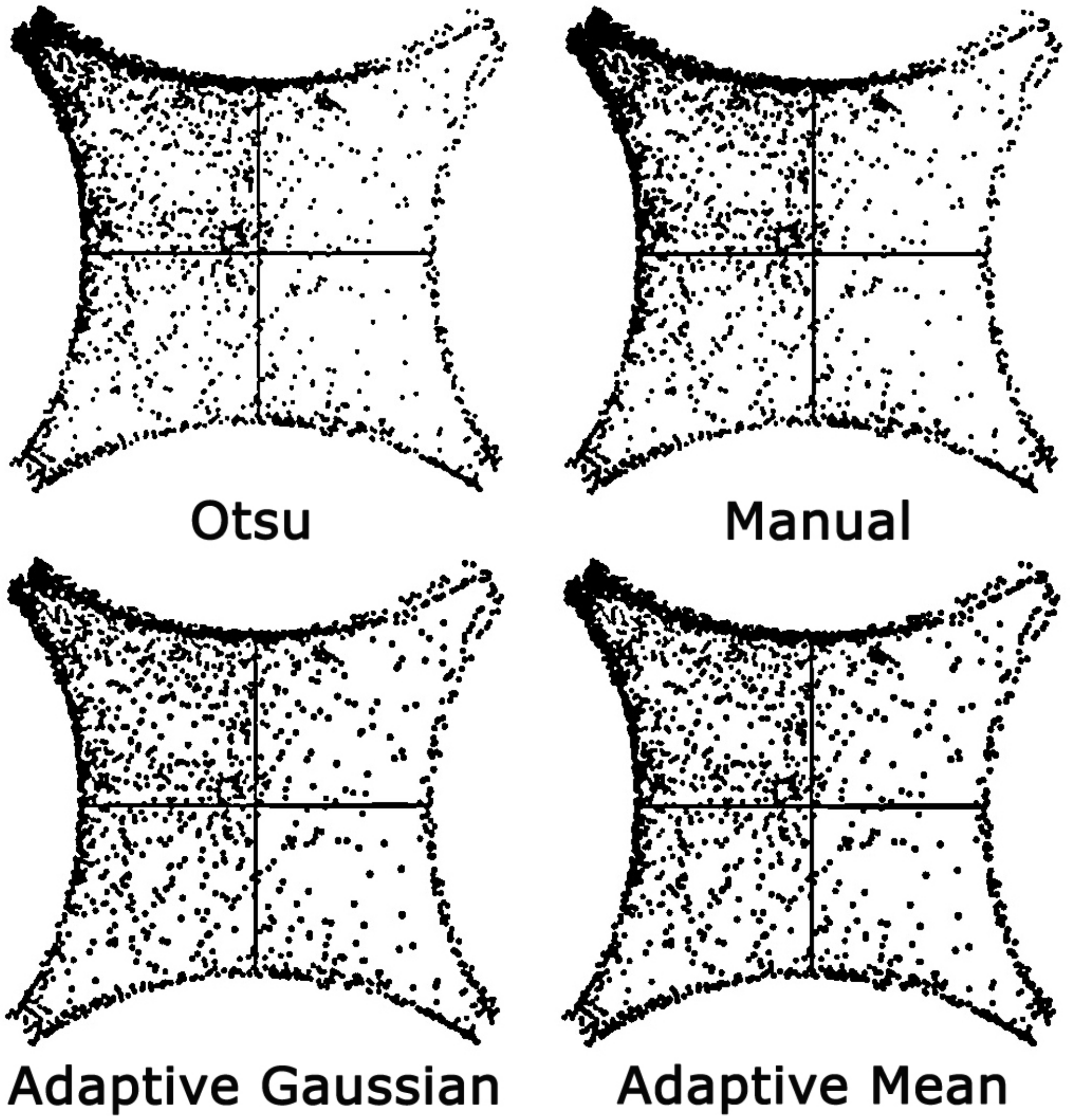


Figure 4.7: Binary Images

Based on the optical results shown at Figure 4.7, the Otsu Thresholding is giving us excellent results. We will proceed with the Otsu method to count the concentrations for all the 18 instances of the Olfactometer (images). This is done by the code that follows:

```
/* allFigureBin.cpp
 *
 * Klotsonis Alkiviadis
 * School of Electronics & Computer Engineering
 * Technical University of Crete 2015
 *
 * Redistribution is allowed provided that it will
 * retain this notice
 *
 * This program loads an image from file as Grayscale
 * and it converts it to Binary, performing the Otsu
 * Thresholding method. Then it counts the concentrations
 * at each quadrant and prints the as percentage.
 * The above is performed recursively to all the 18 images.
*/


#include<iostream>
#include<opencv2/opencv.hpp>
using namespace std;
using namespace cv;

int main()
{
    for( int n = 1; n <= 18; n++ ) {
      Mat figure;
      stringstream sstm;
      sstm << "Figure" << n << ".jpg";
      figure = imread(sstm.str(),CV_LOAD_IMAGE_GRAYSCALE);
      if(figure.empty()){
         cout << "Error loading the image" << endl;
         return -1;
      }

      threshold(figure,figure,0,255,CV_THRESH_BINARY |
CV_THRESH_OTSU);

      int b[4] = { };
      int total = 0;
      for( int y = 0; y < figure.rows; y++ ) {
          for( int x = 0; x < figure.cols; x++ ) {
            if(x<(figure.rows/2) && y<(figure.rows/2)){   //top left
                if(figure.at<uchar>(y,x) != 255) b[0]++;
              }
            if(x>=(figure.rows/2) && y<(figure.rows/2)){  //top right
                if(figure.at<uchar>(y,x) != 255) b[1]++;
            }
            if(x<(figure.rows/2) && y>=(figure.rows/2)){  //bot left
                if(figure.at<uchar>(y,x) != 255) b[2]++;
            }
            if(x>=(figure.rows/2) && y>=(figure.rows/2)){ //bot right
                if(figure.at<uchar>(y,x) != 255) b[3]++;
            }
        }
    }
    total = b[0] + b[1] + b[2] + b[3];

    cout << "Top Left " << (b[0]*100/total) << " Top Right " <<
(b[1]*100/total) << " Bottom Left " << (b[2]*100/total) << " Bottom
Right " << (b[3]*100/total) << " for " << n << " loaded image" <<
endl;

  }
  return 0;
}
```

Figure	Optical Concentration	Top Left Conc. %	Top Right Conc. %	Bottom Left Conc. %	Bottom Right Conc. %
1	Top Left	64	14	16	5
2	None	29	25	21	22
3	Top Left	64	11	17	6
4	None	26	22	24	25
5	Top Left	45	16	23	14
6	Top Left	54	19	16	9
7	Top Left	53	15	18	12
8	Top Left	59	15	15	10
9	None	29	25	21	23
10	Top Left	64	11	13	9
11	Top Left	58	15	13	12
12	None	35	24	22	17
13	None	30	25	21	22
14	None	31	25	25	17
15	Top Left	72	14	8	4
16	None	28	20	29	20
17	Top Left	77	7	11	4
18	None	16	25	34	22

Table 4.1: All Figures Concentrations after Otsu Thresholding

4.5.2 Gathering Decision Formula Design

Now that the algorithm has already counted the concentrations as a percentage at every quadrant of the captured frame we will need a way to decide if a gathering is occurring in one of them. Here we have to transform an objective decision to deterministic, based on objective criteria. As we are not entomologists we need to get based on the previous optical assumption to proceed.

We will express gathering as the ratio between the bigger concentration to the summary of the rest. After calculating this number that will be named the Gathering Ratio, we then must define a threshold empirically that any value above will be considered a gathering to the quadrant that shows the bigger concentration. Thus, the Gathering Ratio (GR) is calculated by the following equation:

$$GR = \max\{a_i\} / (sum\{a_i\} - \max\{a_i\})$$

Where a_i is the concentration at the i quadrant.

Images that show concentration				
Top Left %	Top Right %	Bottom Left %	Bottom Right %	Gathering Ratio
64	14	16	5	1.83
64	11	17	6	1.88
45	16	23	14	0.85
54	19	16	9	1.23
53	15	18	12	1.18
59	15	15	10	1.48
64	11	13	9	1.94
58	15	13	12	1.45
72	14	8	4	2.77
77	7	11	4	3.5

Table 4.2: High Gathering Ratios

Images with no concentration				
Top Left %	Top Right %	Bottom Left %	Bottom Right %	Gathering Ratio
29	25	21	22	0.43
26	22	24	25	0.37
29	25	21	23	0.42
35	24	22	17	0.56
30	25	21	22	0.44
31	25	25	17	0.46
28	20	29	20	0.43
16	25	34	22	0.54

Table 4.3: Low Gathering Ratios

The empirically defined threshold (EDT) will be the mean between the lowest gathering ratio of the images that show concentration and biggest gathering ratio of those that do not. As simple as:

$$EDT = \frac{\min\{GR_{high}\} + \max\{GR_{low}\}}{2}$$

Based on Tables 4.2 & 4.3 our $EDT = 0.7$

4.5.3 Capturing Divergence & Lighting Conditions

Different lighting conditions are giving an altered Grayscale image so another thresholding method might be better. An objective way to evaluate the thresholding methods would be to follow the pixel error rate (PEER) technique [184] or goal-driven evaluation [185]. However when our images were loaded from the file system, it was obvious that the Otsu's method was giving us excellent results. Otsu's method seems to be the best for the pictures captured by the camera at most lighting conditions as well, but that should be examined and remains to be proved.

For that purpose we must choose an image from the 18 that shows either gathering or not but marginally and print it, in order to get captured by the camera. Afterwards we process the captured frame and apply to it the 4 candidate thresholding methods. Then counting the concentrations per quadrant at the 4 binary images and we calculate the weighted mean divergence with the concentrations that were emerged from "allFigureBin.cpp" for the corresponding image. The following code implements the above description.

```
/* allCaptureBin.cpp
 *
 * Klotsonis Alkiviadis
 * School of Electronics & Computer Engineering
 * Technical University of Crete 2015
 *
 * Redistribution is allowed provided that it will retain this notice
 *
 * This piece of code captures a frame from the camera converts it to
 * grayscale and performs Binarization with 4 different thresholding
 * methods. It also counts the concentration of black pixels at every
 * quadrant of the converted to square frame. It finally output the
 * concentrations expressed as percentage and all the binary image as
 * well as the captured one.
 */

#include<iostream>
#include<opencv2/opencv.hpp>
using namespace std;
using namespace cv;

int main()
{
    VideoCapture capture(0);
    capture.set(CV_CAP_PROP_FRAME_WIDTH,640);
    capture.set(CV_CAP_PROP_FRAME_HEIGHT,480);
    if(!capture.isOpened()){
      cout << "Camera Connectivity Error." << endl;
    }
    Mat frame, bin_otsu, bin_man, bin_ad_mean_c, bin_ad_gaus_c;
    capture >> frame;
    if(frame.empty()){
      cout << "Empty Image, steam is dead." << endl;
      return -1;
    }
    frame = frame(Rect(80,0,480,480));
    cvtColor(frame, frame, CV_BGR2GRAY);
    threshold(frame,bin_otsu,0,255,CV_THRESH_BINARY | CV_THRESH_OTSU);
    threshold(frame,bin_man,180,255,CV_THRESH_BINARY);

adaptiveThreshold(frame,bin_ad_mean_c,255,ADAPTIVE_THRESH_MEAN_C,THRESH
_BINARY,61,5);

adaptiveThreshold(frame,bin_ad_gaus_c,255,ADAPTIVE_THRESH_GAUSSIAN_C,TH
RESH_BINARY,61,5);

    int otsu[4] = { };
    int manual[4] = { };
    int mean_c[4] = { };
    int gaussian_c[4] = { };
    for( int y = 0; y < frame.rows; y++ ) {
      for( int x = 0; x < frame.cols; x++ ) {
          if(x<(frame.rows/2) && y<(frame.rows/2)){   //top left
            if(bin_otsu.at<uchar>(y,x) != 255){
               otsu[0]++;
            }
            if(bin_man.at<uchar>(y,x) != 255){
               manual[0]++;
            }
            if(bin_ad_mean_c.at<uchar>(y,x) != 255){
               mean_c[0]++;
            }
            if(bin_ad_gaus_c.at<uchar>(y,x) != 255){
               gaussian_c[0]++;
            }
          }
          if(x>=(frame.rows/2) && y<(frame.rows/2)){  //top right
            if(bin_otsu.at<uchar>(y,x) != 255){
               otsu[1]++;
            }
            if(bin_man.at<uchar>(y,x) != 255){
               manual[1]++;
            }
            if(bin_ad_mean_c.at<uchar>(y,x) != 255){
               mean_c[1]++;
            }
            if(bin_ad_gaus_c.at<uchar>(y,x) != 255){
               gaussian_c[1]++;
            }
          }
          if(x<(frame.rows/2) && y>=(frame.rows/2)){  //bottom left
            if(bin_otsu.at<uchar>(y,x) != 255){
               otsu[2]++;
            }
            if(bin_man.at<uchar>(y,x) != 255){
               manual[2]++;
            }
            if(bin_ad_mean_c.at<uchar>(y,x) != 255){
               mean_c[2]++;
            }
            if(bin_ad_gaus_c.at<uchar>(y,x) != 255){
               gaussian_c[2]++;
            }
          }
          if(x>=(frame.rows/2) && y>=(frame.rows/2)){ //bottom right
            if(bin_otsu.at<uchar>(y,x) != 255){
               otsu[3]++;
            }
            if(bin_man.at<uchar>(y,x) != 255){
               manual[3]++;
            }
            if(bin_ad_mean_c.at<uchar>(y,x) != 255){
               mean_c[3]++;
            }
            if(bin_ad_gaus_c.at<uchar>(y,x) != 255){
               gaussian_c[3]++;
            }
          }
      }
    }
    int otsu_total = otsu[0] + otsu[1] + otsu[2] + otsu[3];
    int manual_total = manual[0] + manual[1] + manual[2] + manual[3];
    int mean_total = mean_c[0] + mean_c[1] + mean_c[2] + mean_c[3];
    int gaus_total = gaussian_c[0] + gaussian_c[1] + gaussian_c[2] +
gaussian_c[3];

    cout << "Top Left " << (otsu[0]*100/otsu_total) << " Top Right " <<
(otsu[1]*100/otsu_total) << " Bottom Left " << (otsu[2]*100/otsu_total)
<< " Bottom Right " << (otsu[3]*100/otsu_total) << " with Otsu
Thresholding" << endl;

    cout << "Top Left " << (manual[0]*100/manual_total) << " Top Right
" << (manual[1]*100/manual_total) << " Bottom Left " <<
(manual[2]*100/manual_total) << " Bottom Right " <<
(manual[3]*100/manual_total) << " with Manual Thresholding" << endl;

    cout << "Top Left " << (mean_c[0]*100/mean_total) << " Top Right "
<< (mean_c[1]*100/mean_total) << " Bottom Left " <<
(mean_c[2]*100/mean_total) << " Bottom Right " <<
(mean_c[3]*100/mean_total) << " with Adapt. Mean Thresholding" << endl;

    cout << "Top Left " << (gaussian_c[0]*100/gaus_total) << " Top
Right " << (gaussian_c[1]*100/gaus_total) << " Bottom Left " <<
(gaussian_c[2]*100/gaus_total) << " Bottom Right " <<
(gaussian_c[3]*100/gaus_total) << " with Adapt. Gaussian Thresholding"
<< endl;

    imwrite("capture.jpg", frame);
    imwrite("binOtsu.jpg", bin_otsu);
    imwrite("binManual.jpg", bin_man);
    imwrite("binMean.jpg", bin_ad_mean_c);
    imwrite("binGaus.jpg", bin_ad_gaus_c);
    return 0;
}
```

Now that we have counted the concentrations for each method we need to calculate what we previous named as weighted mean divergence (WMD). WMD will be expressed here as the weighted variance of the concentration values. As mean we will adapt the concentration values for the corresponding figure from Table 4.1. The equations below are describing the calculation of WMD in a general form for anyone that would like to confirm the minimum WMD with more that one figures.

The variance from Table 4.1:

$$Variance = \frac{\sum_{i=1}^{n}(m_i - \mu_i)^2}{n}$$

Where:

- $n = 4$ for a 4-way Olfactometer
- μ_i the corresponding values from Table 4.1 and
- m_i the values from the corresponding method that we are counting the variance for.

We will expand the above to give more emphasis at the possible gathering side:

$$WMD = \frac{\sum_{i=1}^{n} a_i(m_i - \mu_i)^2}{n+1}$$

Where $a_i = \begin{cases} 2 \text{ , } if\ m_i = \max(m) \\ 1 \text{ , } otherwise \end{cases}$

Results From	Top Left %	Top Right %	Bottom Left %	Bottom Right %	WMD To Table 4.1
Table 4.1 Figure ("mean")	**45**	**16**	**23**	**14**	
Otsu Thresholding	47	23	19	10	17.8
Manual Thresholding	40	31	20	6	69.6
Adaptive Mean Thresh.	36	19	26	17	37.8
Adaptive Gaussian Thr.	37	19	26	16	30

Table 4.4: WMD for all thresholding methods

As we see at the Table 4.4 the Otsu method performs better than the other 3 even when the image is capture by the camera under intense and uniform lighting conditions. It remains to be proved that it gives the right results as well.

4.5.4 Formula Evaluation

The thresholding method that showed the lower weighted variance above, will now be used in order to capture, process, transform to binary and count concentrations for the other 17 images as well. The concentrations as percentage on each quadrant for all the 18 images are shown in Table 4.5 below.

We also need to verify the formula that we designed at 4.5.2. For that purpose we have added at Table 4.5 two extra rows, the “Optical Concentration” and the “Gathering Ratio”. As was shown during the formula design if the empirically defined threshold (EDT) is over 0.7 there is a gathering. At Table 4.5 all Gathering Ratios comply with the optical results, so we are good to go on. Otherwise we should consider revising thresholding method, redesigning the formula, or even changing the light conditions.

For the purpose of the evaluation it is appropriate to use extreme conditions. The values at Table 4.5 are result of an experiment that took place under the worst possible lighting conditions that could produce acceptable results. The light intensity was low and the distribution medium. Even in such an environment the decision method that was described above was able to predict right and the concentrations did not show significant divergence to the values that where emerged when the figures were read from file. The optical results were acceptable as well. During the real-time response test later, we will notice that excellent results can be produced under better lighting conditions and even faster.

Figure	Optical Concentration	Top Left %	Top Right %	Bottom Left %	Bottom Right %	Gathering Ratio
1	Top Left	63	18	14	3	1.8
2	None	31	30	21	17	0.46
3	Top Left	61	18	17	3	1.61
4	None	28	28	23	20	0.39
5	Top Left	43	25	20	10	0.78
6	Top Left	52	25	15	6	1.13
7	Top Left	52	22	17	8	1.11
8	Top Left	58	21	14	6	1.41
9	None	31	30	20	17	0.46
10	Top Left	62	18	12	6	1.72
11	Top Left	57	21	12	8	1.39
12	None	36	30	20	12	0.58
13	None	32	31	19	16	0.48
14	None	32	31	23	11	0.49
15	Top Left	59	31	7	1	1.51
16	None	33	32	24	10	0.5
17	Top Left	71	16	9	2	2.63
18	None	21	31	30	16	0.46

Table 4.5: Formula Evaluation with all Figures

4.5.5 Real Time Response Test

Assuming that we have deterministically found the optimal thresholding method, now remains that our device is capable of performing this series of actions in real time. We haven't mentioned yet how we are going to measure the time in order to examine the real-time capability of our system, but if we are developing it with C++, the tool for that purpose couldn't be other than the <time.h> library. Including that library to our code solves that problem as simple as declaring two variables of type timespec (ex. start_time & end_time) and using the clock_gettime function to get the time of the clock, once before the main algorithm implementation and another after. Then subtracting those values will give us the time needed for our algorithm to process a single frame. If that number is considered a satisfactory interval within our system can renew the output and be considered responsive then we have a real-time capable implementation. With the following code that does this real time capability check, we will be capturing frames and applying the algorithm more that once recursively in order to educe a reliable frame per second (FPS) ration capability. The FPS is educed by counting the overall time needed to process the given number of frames divided by the frames number.

```
/* algorithmCount.cpp
 *
 * Klotsonis Alkiviadis
 * School of Electronics & Computer Engineering
 * Technical University of Crete 2015
 *
 * Redistribution is allowed provided that it will retain this notice
 *
 * This piece of code captures a frame from the camera converts it to
 * grayscale, performs Binarization with the Otsu method and counts
 * the concentration of black pixels at every quadrant.
 * This process is repeated a number of times that is defined as a
 * constant, while the time that it takes is counted. At the end the
 * overall time is divided by repetitions and the fps is deduced.
 */

#include<opencv2/opencv.hpp>
#include<time.h>
#include<iostream>

#define FRAMES 100         //repetitions for the timing test
```

```
#define FRAME_HEIGHT 480
#define FRAME_WIDTH 640

using namespace std;
using namespace cv;

int main()
{
    /*Variable Declaration*/
    Mat frame, bw_frame, bin_frame;
    Mat tl_frame, tr_frame, bl_frame, br_frame;
    struct timespec start, end;
    double interval;
    int total;
    int count[FRAMES][4]={ };
    Int frame_height, frame_width;

    /*Connecting to the Camera*/
    VideoCapture capture(0);
    capture.set(CV_CAP_PROP_FRAME_WIDTH,FRAME_WIDTH);
    capture.set(CV_CAP_PROP_FRAME_HEIGHT,FRAME_HEIGHT);
    if(!capture.isOpened()){
          cout << "Camera Connectivity Error." << endl;
          return -1;
    }

    /*Interval counting starts here*/
    clock_gettime(CLOCK_REALTIME, &start);
    for(int i=0; i<FRAMES; i++){
       capture >> frame;
       if(frame.empty()){
             cout << "Empty Image. Video Stream is Dead." << endl;
             return -1;
       }
       frame = frame(Rect((FRAME_WIDTH-
FRAME_HEIGHT)/2,0,FRAME_WIDTH,FRAME_HEIGHT));
       cvtColor(frame, bw_frame, CV_BGR2GRAY);
       threshold(bw_frame,bin_frame,0,255,CV_THRESH_BINARY | CV_THRESH_OTSU);

       tl_frame = bin_frame(Rect(0,0,bin_frame.cols/2,bin_frame.rows/2));
       tr_frame =
bin_frame(Rect(bin_frame.cols/2,0,bin_frame.cols/2,bin_frame.rows/2));
       bl_frame =
bin_frame(Rect(0,bin_frame.rows/2,bin_frame.cols/2,bin_frame.rows/2));
       br_frame =
bin_frame(Rect(bin_frame.cols/2,bin_frame.rows/2,bin_frame.cols/2,bin_frame.row
s/2));

       for( int y = 0; y < bin_frame.rows/2; y++ ) {
         for( int x = 0; x < bin_frame.cols/2; x++ ) {
           if(tl_frame.at<uchar>(y,x) == 255)
              count[i][0]++;
           if(tr_frame.at<uchar>(y,x) == 255)
              count[i][1]++;
           if(bl_frame.at<uchar>(y,x) == 255)
              count[i][2]++;
           if(br_frame.at<uchar>(y,x) == 255)
              count[i][3]++;
         }
       }
    }
    clock_gettime( CLOCK_REALTIME, &end );
    /*Interval counting just ended*/

    /*Calculating the overall time*/
    interval = (end.tv_sec - start.tv_sec) +
       (double)(end.tv_nsec - start.tv_nsec)/1000000000.0d;

    /*
     * Outputs:
     * 1. The overall time
     * 2. The fps ratio
     * 3. the concentrations
     * 4. All frames & Quadrants
     */
    cout << "It took " << interval << " seconds to process " << FRAMES << "
frames" << endl;
    cout << "Capturing and processing " << FRAMES/interval << " frames per
second " << endl;

    total = count[FRAMES-1][0] + count[FRAMES-1][1] + count[FRAMES-1][2] +
count[FRAMES-1][3];
    cout << "Concentrations: \nTop Left " << (count[FRAMES-1][0]*100/total) <<
" Top Right " << (count[FRAMES-1][1]*100/total) << " Bottom Left " <<
(count[FRAMES-1][2]*100/total) << " Bottom Right " << (count[FRAMES-
1][3]*100/total) << endl;

    imwrite("tl.jpg", tl_frame);
    imwrite("br.jpg", br_frame);
    imwrite("tr.jpg", tr_frame);
    imwrite("bl.jpg", bl_frame);
    imwrite("capture.jpg", frame);
    imwrite("bwCapture.jpg", bw_frame);
    imwrite("binCapture.jpg", bin_frame);
    return 0;
}
```

We have found that light intensity can dramatically affect the frame per second ratio. After testing the algorithm for different numbers of iterations we deduce at the Table 4.6 below the average frame per second ratio.

Light Intensity	Light Distribution	Average FPS
High	Good	21.06
Low	Moderate	14.68

Table 4.6: Frame Rates & Lighting Conditions

4.5.6 Comparison

We have also found that the values of the concentrations under better lighting conditions showed lower variance to Table 4.1 according to the WMD factor. Table 4.7 below shows the values under high intensity light with good distribution together with the WMD factor to Table 4.1.

Figure	Optical Concentration	Top Left %	Top Right %	Bottom Left %	Bottom Right %	Gathering Ratio	WMD to Table 4.1
1	Top Left	68	12	15	3	2.27	8.2
2	None	34	26	21	17	0.53	15.2
3	Top Left	67	11	17	3	2.16	5.4
4	None	30	24	23	21	0.44	10.6
5	Top Left	48	17	22	11	0.96	5.8
6	Top Left	57	19	16	6	1.39	5.4
7	Top Left	57	13	19	9	1.39	9.2
8	Top Left	64	14	14	6	1.88	13.6
9	None	35	23	23	17	0.56	23.2
10	Top Left	68	10	14	6	2.27	8.6
11	Top Left	60	15	14	9	1.58	3.6
12	None	38	24	22	13	0.64	6.8
13	None	32	25	22	19	0.48	3.6
14	None	33	24	27	14	0.51	4.4
15	Top Left	75	14	7	2	3.26	4.6
16	None	30	22	31	15	0.46	8.2
17	Top Left	77	6	12	4	3.5	0.4
18	None	18	26	36	18	0.58	5.8

Table 4.7: All Figure Values at Good Light Conditions

As we can see at Table 4.7 the highest WMD value is even smaller than the any other at Table 4.4 excluding the Otsu method. This is a more extensive indicator that Otsu's method is performing far better at good lighting conditions. A comparison between the qualitative factors under different lighting conditions is presented below.

Figure	Table 4.1 Gathering Ratio	Table 4.5 Gathering Ratio	Table 4.6 Gathering Ratio	Table 4.5 WMD to Table 4.1	Table 4.6 WMD to Table 4.1
1	1.83	1.8	2.27	5.2	8.2
2	0.43	0.46	0.53	11.6	15.2
3	1.88	1.61	2.16	15.2	5.4
4	0.37	0.39	0.44	14	10.6
5	0.85	0.78	0.96	22.8	5.8
6	1.23	1.13	1.39	10.8	5.4
7	1.18	1.11	1.39	13.6	9.2
8	1.48	1.41	1.88	11	13.6
9	0.42	0.46	0.56	14	23.2
10	1.94	1.72	2.27	13.4	8.6
11	1.45	1.39	1.58	11	3.6
12	0.56	0.58	0.64	13.4	6.8
13	0.44	0.48	0.48	16.8	3.6
14	0.46	0.49	0.51	15.6	4.4
15	2.77	1.51	3.26	127.4	4.6
16	0.43	0.5	0.46	63.8	8.2
17	3.5	2.63	3.5	32.2	0.4
18	0.54	0.46	0.58	27.6	5.8

Table 4.8: Qualitative Factors Comparison

An optical representation of the values shown in Table 4.7 is presented at the next figures. When the light is enough and evenly distributed over the Olfactometer, the gathering ratio seems to be closer to the actual and the WMD is usually lower without great outliers.

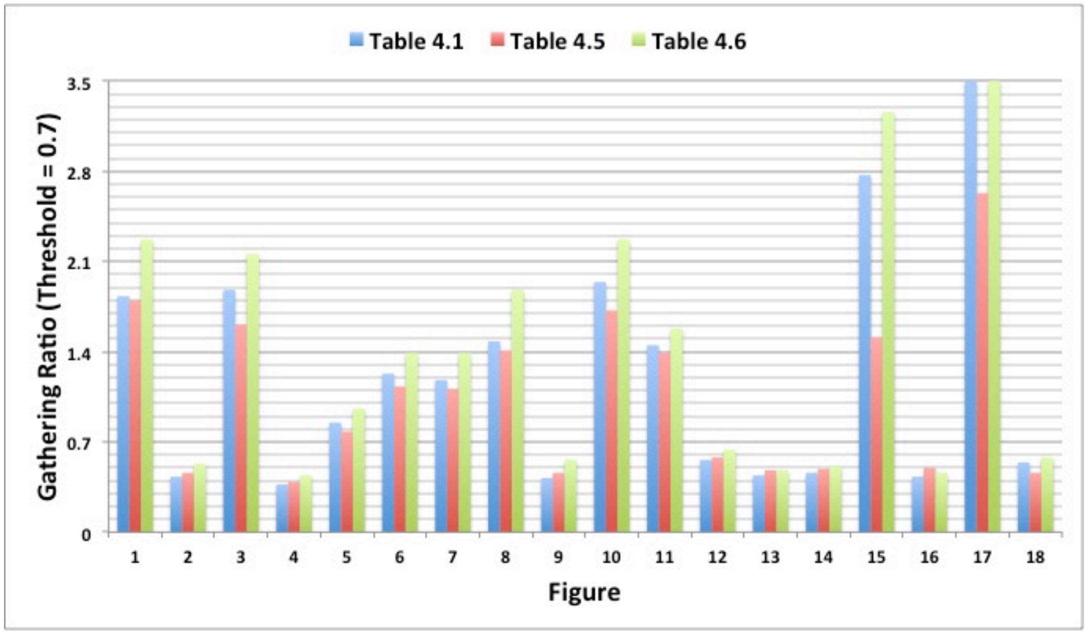


Figure 4.8: Gathering Ration Comparison

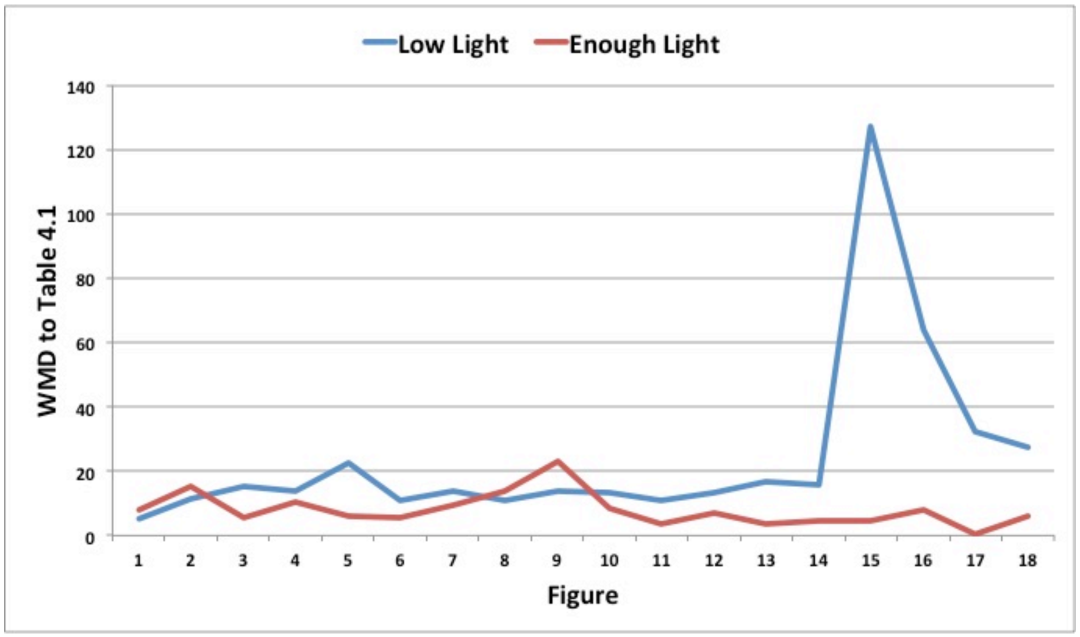


Figure 4.9: WMD Comparison

4.6 Conclusions

The above implementation was done after making a lot of assumptions. It was useful to the scope of a thesis in order to examine the technological tools that where presented as well as basic design for someone to step on when creating the system that will solve the real problem. Many applications that might use such a system would require more expensive equipment and people interested about the use of those applications could financially support the extensive design and implementation. A brief discussion about those factors follows below.

4.6.1 Application Perspectives

The system that was implemented at this chapter can easily be attached and work with a real Olfactometer in an Entomologist's lab. Scientists that study insects behavior either to learn their interests or use insect responses that have previously be trained to evaluate memory correlations would find such a system very handy, as it can save them from staying on top of the Olfactometer holding a pen and a timer. Others study the insect responses to geometrically altered chemical compounds that in their original form attract specific insects and have indications that this might be the key to the secret of the olfaction mechanism. If this system could contribute the minimum to the proof of that assessment and that could open a window to the path of creating a model that could describe any scent, then we would be towards the digitization of olfaction.

Using a system like the one we implemented here in a lab with real insects, except that it could record results with great accuracy it could also open the perspective to count the insects response times to chemical compounds and compare them with the existing electronic equipment that is detecting those chemical compounds. Even it is almost impossible to overcome the existing technology in the terms of response with such a system it is more likely that we will when it comes to sensitivity. Applications that need to localize the presence of a compound detected might find such a system extremely advantageous. Some insects have biological sensors that can detect compounds present many meters away. A system like the one we described is even small enough to be attached on a small drone. Drones today have high development growth and with an extremely sensitive sensor on them, they could be useful from detecting a rotten orange in a big warehouse to locating mines or other kind of explosives on the open field.

4.6.2 FUTURE EXTENSIONS

According to the application perspectives that were discussed the first extension for our system would be to the ability to describe more extensively the insects gathering. Except from answering in which quadrant a gathering is present it might be useful to reconsider a formula that will use directional criteria that could be combined and interact with a drone piloting system.

An implementation to a more powerful device together with the designing decision that should be made is the next step. The BBB was a very handy device to get to know the existent technological tools that helped this system implementation but in order to meet the requirements of a real world application and the stability needed for such a system to get accepted we need to move to more powerful and stable devices.

At the same time an extension of the experimental process using an Olfactometer with real insects inside is consider a must. Factors like captured image differences that will arise when the camera is place above the Olfactometer as well as any different lighting conditions and camera types that might be considered necessary should be taken into account. As we have already mentioned different applications might enforce other lighting conditions which by the might change the insects behavior. So comparing results with various camera types and lighting conditions could be interesting.

We should also mention here that the emerging ratio of frames per second that is achieved by the BBB and the Microsoft camera at our experiment it is expected to be a little bit higher if we had real moving insects in an Olfactometer, due to the dynamic frame renewal. This can be expected to happen basically because of a potential momentary video-capturing lag where the capturing process becomes the bottleneck of the overall efficiency. This though is not absolute and less luckily to happen with even just a better camera.

Last but not least and given that the progress to a real Olfactometer experiment have been made we should consider a design where the Olfactometer input valves are electronic and able to communicate with our the main system. In such a case we would be able to develop a model that correlates the insects gathering factor with the concentrations of the chemical compounds that induced by the valves. Creating this model will give the opportunity to accurately investigate the insect's sensing ability, thus our system's sensitivity and sensibility.

References

[1] J.W. Gardner, P.N. Bartlett (1999). Electronic Noses: Principles and Applications, Oxford University Press: p.245.

[2] Wikipedia, Odor, http://en.wikipedia.org/wiki/Odor

[3] Benzo Maurizio, Mantovani Alice, Pittarello Alberto (2012). Measurement of Odour Concentration of Immissions using a New Field Olfactometer and Markers' Chemical Analysis, Chemical Engineering Transactions 30: 103.

[4] CEN EN 13725:2003, Air quality – Determination of odour concentration by dynamic olfactometry

[5] Van Harreveld A. P., Heeres P., Harssema H. (1999). A review of 20 years of standardization of odor concentration measurement by dynamic olfactometry in Europe, Journal of the Air & Waste Management Association 49 (6): 705–715.

[6] Cain W. S., Gent J. F. (1991). Olfactory sensitivity: Reliability, generality, and association with aging. Journal of experimental psychology. Human perception and performance 17 (2): 382–91.

[7] Wysocki CJ, Dorries KM, Beauchamp GK (1989). Ability to perceive androstenone can be acquired by ostensibly anosmic people, Proc Natl Acad Sci USA 86 (20): 7976–7978.

[8] Cain WS (1977). Differential sensitivity for smell: "noise" at the nose, Science 195 (4280): 796–798.

[9] Spengler John D., McCarthy John F, Samet Jonathan M. (2000). Indoor Air Quality Handbook. New York, NY, USA: McGraw-Hill Professional Publishing. ISBN 978-0-07-445549-4.

[10] Jiang J, Coffey P, Toohey B (2006). Improvement of odor intensity measurement using dynamic olfactometry. Journal of the Air & Waste Management Association 56 (5): 675–830.

[11] http://www.odorimpact.com/en/fidol.shtml#axzz3Z1XKG7JN

[12] http://www.mfe.govt.nz/publications/air/odour-guidelines-jun03/html/page6.html

[13] Alphus D. Wilson, Manuela Baietto (2009). Applications and Advances in Electronic-Nose Technologies. Sensors 9(7): 5099-5148

[14] Zhang X, Firestein S (2002). The olfactory receptor gene superfamily of the mouse. Nat Neurosci 5(2): 124–133

[15] Godfrey PA, Malnic B, Buck LB (2004). The mouse olfactory receptor gene family. Proc Natl Acad Sci USA 101(7): 2156–2161

[16] Quignon P, Giraud M, Rimbault M, Lavigne P, Tacher S, Morin E, Retout E, Valin A-S, Lindblad-Toh K, Nicolas J, Galibert F (2005). The dog and rat olfactory receptor repertoires. Genome Biol 6(10): R83

[17] Buck L, Axel R (1991). A novel multigene family may encode odorant receptors: a molecular basis for odor recognition. Cell 65(1): 175–187

[18] Dulac C, Axel R (1995). A novel family of genes encoding putative pheromone receptors in mammals. Cell 83(2): 195–206

[19] Bean NJ (1982). Modulation of agonistic behavior by the dual olfactory system in male mice. Physiol Behav 29(3): 433–437

[20] Papes F, Logan DW, Stowers L (2010). The vomeronasal organ mediates interspecies defensive behaviors through detection of protein pheromone homologs. Cell 141(4): 692–703

[21] Fleischer J et al (2006). A novel population of neuronal cells expressing the olfactory marker protein (OMP) in the anterior/dorsal region of the nasal cavity. Histochem Cell Biol 125(4): 337–349

[22] Fleischer J et al (2006). Olfactory receptors and signalling elements in the Grueneberg ganglion. J Neurochem 98(2):543–554

[23] Brechbühl J, Klaey M, Broillet M-C (2008). Grueneberg ganglion cells mediate alarm pheromone detection in mice. Science 321(5892): 1092–1095

[24] Tian H, Ma M (2004). Molecular organization of the olfactory septal organ. J Neurosci 24(38): 8383–8390

[25] Storan MJ, Key B (2006). Septal organ of Grüneberg is part of the olfactory system. J Comp Neurol 494(5): 834–844

[26] Hara T (1994). The diversity of chemical stimulation in fish olfaction and gustation. Rev Fish Biol Fisheries 4(1): 1–35

[27] Yoshihara Y (2009). Molecular genetic dissection of the zebrafish olfactory system, in chemosensory systems in mammals, fishes, and insects. Springer. p.1–19

[28] Ngai J et al (1993). Coding of olfactory information: topography of odorant receptor expression in the catfish olfactory epithelium. Cell 72(5): 667–680

[29] Weth F, Nadler W, Korsching S (1996). Nested expression domains for odorant receptors in zebrafish olfactory epithelium. Proc Natl Acad Sci 93(23): 13321–13326

[30] Alioto T, Ngai J (2005). The odorant receptor repertoire of teleost fish. BMC Genomics 6(1): 173

[31] Niimura Y, Nei M (2005). Evolutionary dynamics of olfactory receptor genes in fishes and tetrapods. Proc Natl Acad Sci 102(17): 6039–6044

[32] Hashiguchi Y, Nishida M (2007). Evolution of trace amine-associated receptor (TAAR) gene family in vertebrates: lineage-specific expansions and degradations of a second class of vertebrate chemosensory receptors expressed in the olfactory epithelium. Mol Biol Evol 24(9): 2099–2107

[33] Michel W et al (2003). Evidence of a novel transduction pathway mediating detection of polyamines by the zebrafish olfactory system. J Exp Biol 206(10): 1697–1706

[34] Rolen SH et al (2003). Polyamines as olfactory stimuli in the goldfish Carassius auratus. J Exp Biol 206(10): 1683–1696

[35] Sato Y, Miyasaka N, Yoshihara Y (2007). Hierarchical regulation of odorant receptor gene choice and subsequent axonal projection of olfactory sensory neurons in zebrafish. J Neurosci 27(7): 1606–1615

[36] Sato Y, Miyasaka N, Yoshihara Y (2005). Mutually exclusive glomerular innervation by two distinct types of olfactory sensory neurons revealed in transgenic zebrafish. J Neurosci 25(20): 4889–4897

[37] Hansen A et al (2003). Correlation between olfactory receptor cell type and function in the channel catfish. J Neurosci 23(28): 9328–9339

[38] Morita Y, Finger TE (1998). Differential projections of ciliated and microvillous olfactory receptor cells in the catfish, Ictalurus punctatus. J Comp Neurol 398(4): 539–550

[39] Sato K, Touhara K (2009). Insect olfaction: receptors, signal transduction, and behavior, in chemosensory systems in mammals, fishes, and insects. Springer, p. 203–220

[40] Stocker R et al (1990). Neuronal architecture of the antennal lobe in Drosophila melanogaster. Cell Tissue Res 262(1): 9–34

[41] Datta SR et al (2008). The Drosophila pheromone cVA activates a sexually dimorphic neural circuit. Nature 452(7186): 473–477

[42] Clyne PJ et al (1999). A novel family of divergent seven-transmembrane proteins: candidate odorant receptors in Drosophila. Neuron 22(2): 327–338

[43] Gao Q, Chess A (1999). Identification of candidate Drosophila olfactory receptors from genomic DNA sequence. Genomics 60(1): 31–39

[44] Hill CA et al (2002). G protein-coupled receptors in anopheles gambiae. Science 298(5591): 176–178

[45] Bohbot J et al (2007). Molecular characterization of the Aedes aegypti odorant receptor gene family. Insect Mol Biol 16(5): 525–537

[46] Robertson HM, Wanner KW (2006). The chemoreceptor superfamily in the honey bee, Apis mellifera: expansion of the odorant, but not gustatory, receptor family. Genome Res 16(11): 1395–1403

[47] Benton R et al (2006). Atypical membrane topology and heteromeric function of Drosophila odorant receptors in vivo. PLoS Biol 4(2): e20

[48] Lundin C et al (2007). Membrane topology of the Drosophila OR83b odorant receptor. Febs Letters 581(29): 5601–5604

[49] Sato K et al (2008). Insect olfactory receptors are heteromeric ligand-gated ion channels. Nature 452(7190): 1002–1006

[50] Wicher D et al (2008). Drosophila odorant receptors are both ligand-gated and cyclic-nucleotide-activated cation channels. Nature 452(7190): 1007–1011

[51] Kalidas S, Smith DP (2002). Novel genomic cDNA hybrids produce effective RNA interference in adult Drosophila. Neuron 33(2): 177–184

[52] Riesgo-Escovar J, Raha D, Carlson JR (1995). Requirement for a phospholipase C in odor response: overlap between olfaction and vision in Drosophila. Proc Natl Acad Sci 92(7): 2864–2868

[53] Vosshall LB et al (1999). A spatial map of olfactory receptor expression in the Drosophila antenna. Cell 96(5): 725–736

[54] Larsson MC et al (2004). Or83b encodes a broadly expressed odorant receptor essential for Drosophila olfaction. Neuron 43(5): 703–714

[55] Dobritsa AA et al (2003). Integrating the molecular and cellular basis of odor coding in the Drosophila antenna. Neuron 37(5): 827–841

[56] Hallem EA, Carlson JR (2006). Coding of odors by a receptor repertoire. Cell 125(1): 143–160

[57] Hallem EA, Ho MG, Carlson JR (2004). The molecular basis of odor coding in the Drosophila antenna. Cell 117(7): 965–979

[58] Benton R et al (2009). Variant ionotropic glutamate receptors as chemosensory receptors in Drosophila. Cell 136(1): 149–162

[59] White JG et al (1986). The structure of the nervous system of the nematode Caenorhabditis elegans. Philos Trans R Soc Lond B Biol Sci 314(1165):1–340

[60] Chen N et al (2005). Identification of a nematode chemosensory gene family. Proc Natl Acad Sci U S A 102(1): 146–151

[61] Bargmann CI (2006). Comparative chemosensation from receptors to ecology. Nature 444(7117): 295–301

[62] Bargmann CI, Hartwieg E, Horvitz HR (1993). Odorant-selective genes and neurons mediate olfaction in C. elegans. Cell 74(3): 515–527

[63] Troemel ER et al (1995). Divergent seven transmembrane receptors are candidate chemosensory receptors in C. elegans. Cell 83(2): 207–218

[64] Sengupta P, Chou JH, Bargmann CI (1996). odr-10 encodes a seven transmembrane domain olfactory receptor required for responses to the odorant diacetyl. Cell 84(6): 899–909

[65] Robertson HM (2001). Updating the str and srj (stl) families of chemoreceptors in Caenorhabditis nematodes reveals frequent gene movement within and between chromosomes. Chem Senses 26(2): 151–159

[66] McCarroll SA, Li H, Bargmann CI (2005). Identification of transcriptional regulatory elements in chemosensory receptor genes by probabilistic segmentation. Curr Biol 15(4): 347–352

[67] Troemel ER, Kimmel BE, Bargmann CI (1997). Reprogramming chemotaxis responses: sensory neurons define olfactory preferences in C. elegans. Cell 91(2): 161–169

[68] Ashton EH, Eayrs JT, Moulton DG (1957). Olfactory acuity in the dog. Nature 179(4569): 1069–1070

[69] Moulton DG, Ashton EH, Eayrs JT (1960). Studies in olfactory acuity. 4. Relative detectability of n-aliphatic acids by the dog. Anim Behav 8(3–4): 117–128

[70] Furton KG, Myers LJ (2001). The scientific foundation and efficacy of the use of canines as chemical detectors for explosives. Talanta 54(3): 487–500

[71] Gazit I, Lavner Y, Bloch G, Azulai O, Goldblatt A, Terkel J (2003). A simple system for the remote detection and analysis of sniffing in explosives detection dogs. Behav Res Methods Instrum Comput 35(1): 82–89

[72] Cornu J-N, Cancel-Tassin G, Ondet V, Girardet C, Cussenot O (2011). Olfactory detection of prostate cancer by dogs sniffing urine: a step forward in early diagnosis. Eur Urol 59(2): 197–201

[73] Ehmann R, Boedeker E, Friedrich U, Sagert J, Dippon J, Friedel G, Walles T (2012). Canine scent detection in the diagnosis of lung cancer: revisiting a puzzling phenomenon. Eur Respir J 39(3): 669–676

[74] McCulloch M, Jezierski T, Broffman M, Hubbard A, Turner K, Janecki T (2006). Diagnostic accuracy of canine scent detection in early- and late-stage lung and breast cancers. Integr Cancer Ther 5(1): 30–39

[75] Bodyak N, Slotnick B (1999). Performance of mice in an automated olfactometer: odor detection, discrimination and odor memory. Chem Senses 24(6): 637–645

[76] Clarke S, Trowill JA (1971). Sniffing and motivated behavior in the rat. Physiol Behav 6(1): 49–52

[77] Uchida N, Mainen ZF (2003). Speed and accuracy of olfactory discrimination in the rat. Nat Neurosci 6(11): 1224–1229

[78] Youngentob SL, Mozell MM, Sheehe PR, Hornung DE (1987). A quantitative analysis of sniffing strategies in rats performing odor detection tasks. Physiol Behav 41(1): 59–69

[79] Hu J, Zhong C, Ding C, Chi Q, Walz A, Mombaerts P, Matsunami H, Luo M (2007). Detection of near-atmospheric concentrations of CO2 by an olfactory subsystem in the mouse. Science 317(5840): 953–957

[80] Maki K. Habib (2007). Controlled biological and biomimetic systems for landmine detection. Biosensors and Bioelectronics 23: 1–18

[81] T. Faber, J.

Joerges, R. Menzel (1999). Associative learning modifies neural representations of odors in the insect brain. Nature neuroscience 2(1): 74-78.

[82] Maria Isabel Franco, Luca

Turin, Andreas

Mershin, Efthimios M C Skoulakis (2011). Molecular vibration-sensing component in Drosophila melanogaster olfaction. Proc Natl Acad Sci U S A 108(9): 3797-3802.

[83] Christiane Linster, Brian H. Smith (1997). A computational model of the response of honey bee antennal lobe circuitry to odor mixtures: Overshadowing, blocking and unblocking can arise from lateral inhibition. Behavioural Brain Research 87(1): 1-14.

[84] Torsten Meiners, Felix Wäckers, Wallace Joe Lewis (2003). Associative learning of complex odours in parasitoid host location. Chemical Senses 28(3): 231-236.

[85] Torsten Meiners, Felix Wäckers, Wallace Joe Lewis (2002). The Effect of Molecular Structure on Olfactory Discrimination by the Parasitoid Microplitis croceipes. Chemical Senses 27: 811-816.

[86] Müller Uli (2002). Learning in honeybees: from molecules to behaviour. Zoology (Jena, Germany) 105(4): 315-320.

[87] D.M. Olson, G.C. Rains1, T. Meiners, K. Takasu, M. Tertuliano, J.H. Tumlinson, F.L. Wäckers and W.J. Lewis (2003). Parasitic wasps learn and report diverse chemicals with unique conditionable behaviors. Chemical Senses 28(6): 545-549.

[88] G. C. Rains, J. K. Tomberlin, M. D'Alessandro, W. J. Lewis (2004). Limits of volatile chemical detection of a parasitoid wasp, Microplitis croceipes, and an electronic nose: a comparative study. Trans. of the ASAE Vol. 47(6): 2145–2152

[89] Glen C. Rains, Samuel L. Utley, W. Joe Lewis (2006). Behavioral monitoring of trained insects for chemical detection. Biotechnology Progress 22(1): 2-8.

[90] M. Tertuliano, D.M. Olson, G.C. Rains, W.J. Lewis (2004). Influence of handling and conditioning protocol on learning and memory of Microplitis croceipes. Entomologia Experimentalis et Applicata 110(2): 165-172

[91] Moukaram Tertuliano, Jeffery K. Tomberlin, Zeljko Jurjevic, David Wilson, Glen C. Rains, W.J. Lewis. (2005). The ability of conditioned Microplitis croceipes (Hymenoptera: Braconidae) to distinguish between odors of aflatoxigenic and non-aflatoxigenic fungal strains. Chemoecology 15(2): 89-95

[92] Karl von Frisch (1950). Bees - Their vision, chemical senses, and language. Cornell University Press

[93] Barbara Webb (2004). Neural mechanisms for prediction: Do insects have forward models? Trends in Neurosciences 27(5): 278-282

[94] Randolf Menzel, Jochen Erber (1978). Learning & Memory in Bees. Scientific Americal 239(1): 80-87

[95] Keiji Takasu, W.J. Lewis (1993). Host- and food-foraging of the parasitoid Microplitis croceipes- learning and physiological-state effects. Biological Control 3: 70-74

[96] Persaud K, Dodd G (1982). Analysis of discrimination mechanisms in the mammalian olfactory system using a model nose. Nature 299(5881): 352–355

[97] J.W. Gardner, P.N. Bartlett (1994). A brief history of electronic noses, Sens. Actuators B(18): 211-220

[98] Buck LB, Axel R (1991). A novel multigene family may encode odorant receptors: a molecular basis for odor recognition. Cell 65(1): 175–187

[99] Ressler KJ, Sullivan SL, Buck LB (1993). A zonal organization of odorant receptor gene expression in the olfactory epithelium. Cell 73(3): 597–609

[100] Ressler KJ, Sullivan SL, Buck LB (1994). Information coding in the olfactory system: Evidence for a stereotyped and highly organized epitope map in the olfactory bulb. Cell 79(7): 1245–1255

[101] Firestein S (2001). How the olfactory system makes sense of scents. Nature 413(6852): 211–218

[102] Malnic B, Hirono J, Sato T, Buck LB (1999). Combinatorial receptor codes for odors. Cell 96(5):713–723

[103] Branca A, Simonian P, Ferrante M, Novas E, Negri RMn (2003). Electronic nose based discrimination of a perfumery compound in a fragrance. Sens Actuat B-Chem 92(1–2): 222–227

[104] J. Gardner, Jehuda Yinon (2004). Electronic Noses and Sensors for the Detection of Explosives. Nato Sience Series II Vol. 159

[105] Oh EH, Song HS, Park TH (2011). Recent advances in electronic and bioelectronic noses and their biomedical applications. Enzyme Microb Technol 48(6–7): 427–437

[106] Capone S, Siciliano P, Quaranta F, Rella R, Epifani M, Vasanelli L (2000) Analysis of vapours and foods by means of an electronic nose based on a sol–gel metal oxide sensors array. Sens Actuators B-Chem 69(3): 230–235

[107] Cerrato Oliveros MC, Pérez Pavón JL, García Pinto C, Fernández Laespada ME, Moreno Cordero B, Forina M (2002). Electronic nose based on metal oxide semiconductor sensors as a fast alternative for the detection of adulteration of virgin olive oils. Anal Chim Acta 459(2): 219–228

[108] Dutta R, Hines EL, Gardner JW, Kashwan KR, Bhuyan M (2003). Tea quality prediction using a tin oxide-based electronic nose: an artificial intelligence approach. Sens Actuat B-Chem 94(2): 228–237

[109] El Barbri N, Amari A, Vinaixa M, Bouchikhi B, Correig X, Llobet E (2007). Building of a metal oxide gas sensor-based electronic nose to assess the freshness of sardines under cold storage. Sens Actuat B-Chem 128(1): 235–244

[110] González Martín Y, Cerrato Oliveros MC, Pérez Pavón JL, García Pinto C, Moreno Cordero B (2001). Electronic nose based on metal oxide semiconductor sensors and pattern recognition techniques: characterisation of vegetable oils. Anal Chimica Acta 449(1–2): 69–80

[111] Crone B, Dodabalapur A, Gelperin A, Torsi L, Katz HE, Lovinger AJ, Bao Z (2001). Electronic sensing of vapors with organic transistors. Appl Phys Lett 78(15): 2229–2231

[112] Doleman BJ, Lewis NS (2001). Comparison of odor detection thresholds and odor discriminablities of a conducting polymer composite electronic nose versus mammalian olfaction. Sens Actuat B-Chem 72(1): 41–50

[113] Hatfield JV, Neaves P, Hicks PJ, Persaud K, Travers P (1994). Towards an integrated electronic nose using conducting polymer sensors. Sens Actuat B-Chem 18(1–3): 221–228

[114] Torsi L, Dodabalapur A, Sabbatini L, Zambonin PG (2000). Multi-parameter gas sensors based on organic thin-film-transistors. Sens Actuat B-Chem 67(3): 312–316

[115] Hao HC, Tang KT, Ku PH, Chao JS, Li CH, Yang CM, Yao DJ (2010). Development of a portable electronic nose based on chemical surface acoustic wave array with multiplexed oscillator and readout electronics. Sens Actuat B-Chem 146(2): 545–553

[116] Gan HL, Man YBC, Tan CP, NorAini I, Nazimah SAH (2005). Characterisation of vegetable oils by surface acoustic wave sensing electronic nose. Food Chem 89(4): 507–518

[117] García M, Fernández MJ, Fontecha JL, Lozano J, Santos JP, Aleixandre M, Sayago I, Gutiérrez J, Horrillo MC (2006). Differentiation of red wines using an electronic nose based on surface acoustic wave devices. Talanta 68(4): 1162–1165

[118] Li C, Heinemann P, Sherry R (2007). Neural network and Bayesian network fusion models to fuse electronic nose and surface acoustic wave sensor data for apple defect detection. Sens Actuat B-Chem 125(1): 301–310

[119] Himanshu K. Patel (2014). The Electronic Nose: Artificial Olfaction Technology. Springer ISBN 978-81-322-1547-9, ch. 6.4

[120] Laura Capelli, Selena Sironi, Renato Del Rosso (2014). Electronic Noses for Environmental Monitoring Applications. Sensors 14: 19979-20007

[121] M. Aleixandre, J. Lozano, J. Guti´errez, I. Sayago, M.J. Fern´andez, M.C. Horrillo (2008). Portable e-nose to classify different kinds of wine. Sensors and Actuators, B: Chemical 131(1): 71-76.

[122] M. Garcia, M.J. Fernandez, J.L. Fontecha, J. Lozano, J.P. Santos, M. Aleixandre, I. Sayago, J. Gutierrez, M.C. Horrillo (2006). Differentiation of red wines using an electronic nose based on surface acoustic wave devices. Talanta 68(4): 1162-1167

[123] M. Garcia, M. Aleixandre, J. Gutierrez, M.C. Horrillo (2006). Electronic nose for wine discrimination. Sensors and Actuators, B: Chemical 113(2): 911-916.

[124] A. Guadarrama, J.A. Fernández, M. Íñiguez, J. Souto, J.A. de Saja (2000). Array of conducting polymer sensors for the characterisation of wines. Analytica Chimica Acta 411(1-2): 193-200.

[125] J. Lozano, T. Arroyo, J.P. Santos, J.M. Cabellos, M.C. Horrillo (2008). Electronic nose for wine ageing detection. Sensors and Actuators, B: Chemical 133(1): 180-186

[126] J. Lozano, M.J. Fernandez, J.L. Fontecha, M. Aleixandre, J.P. Santos, I. Sayago, T. Arroyo, J.M. Cabellos, F.J. Gutierrez, M.C. Horrillo (2006). Wine classification with a zinc oxide SAW sensor array. Sensors and Actuators, B: Chemical 120(1): 166-171.

[127] J. Lozano, J.P. Santos, M.C. Horrillo (2005). Classification of white wine aromas with an electronic nose. Talanta 67(3): 610-616.

[128] J.P. Santos, M.J. Fernandez, J.L. Fontecha, J. Lozano, M. Aleixandre, M. Garcia, J. Gutierrez, M.C. Horrillo (2005). SAW sensor array for wine discrimination. Sensors and Actuators, B: Chemical 107(1): 291-295.

[129] A. Guadarrama, M.L. Rodriguez-Méndez, J.A. de Saja (2004). Influence of electrochemical deposition parameters on the performance of poly-3-methyl thiophene and polyaniline sensors for virgin olive oils. Sensors and Actuators, B: Chemical 100(1-2): 60-64.

[130] A. Guadarrama, M.L. Rodriguez-Mendez, J.A. de Saja, J.L. Rios, J.M. Olias (2000). Array of sensors based on conducting polymers for the quality control of the aroma of the virgin olive oil. Sensors and Actuators, B: Chemical 69(3): 276-282.

[131] A. Guadarrama, M.L. Rodriguez-Méndez, C. Sanz, J.L. Rios, J.A. de Saja (2001). Electronic nose based on conducting polymers for the quality control of the olive oil aroma - Discrimination of quality, variety of olive and geographic origin. Analytica Chimica Acta 432(2): 283-292.

[132] Rita Stella, Joseph N. Barisci, Giorgio Serra, Gordon G. Wallace b, Danilo De Rossi (2000). Characterization of olive oil by an electronic nose based on conducting polymer sensors. Sensors and Actuators, B: Chemical 63(1): 1-9.

[133] OS. Chew, M.N. Ahmad, Z. Ismail, A.R. OthmanJ (2003). Virmal Chemical Sensor For Classification Of Herb – Orthosipnon stamineus According To Its Geographical Origin. AsinSense Sensor: p. 173 – 178

[134] Edward J. Staples, Shekar Viswanathan (2008). Detection of contrabands in cargo containers using a high-speed gas chromatograph with surface acoustic wave sensor. Ind. Eng. Chem. Res. 47(21): 8361-8367

[135] A.O Afolabi, T.J Afolabi (2013) Implementation of Electronic Nose Technique In Explosives Detection. IJES 2(7): 10-17

[136] Julian W. Gardner, JehudaYinon (2004). Electronic Noses & Sensors for the Detection of Explosives. NATO Science Series II Vol. 159.

[137] Babak Kateb, M.A. Ryan, M.L. Homer, L.M. Lara, Yufang Yin, Kerin Higa, Mike Y. Chen (2009). Sniffing out cancer using the JPL electronic nose: A pilot study of a novel approach to detection and differentiation of brain cancer. NeuroImage 47: T5–T9

[138] Silvano Dragonieria, Marc P. van der Schee, Tommaso Massaro, Nunzia Schiavulli, Paul Brinkman, Armando Pinca, Pierluigi Carratú, Antonio Spanevello, Onofrio Resta, Marina Musti, Peter J. Sterk (2012). An electronic nose distinguishes exhaled breath of patients with Malignant Pleural Mesothelioma from controls. Lung Cancer 75: 326– 331

[139] Göpel W (1998). Chemical imaging: I. Concepts and visions for electronic and bioelectronics noses. Sens Actuat B-Chem 52(1–2): 125–142

[140] Kim TH, Lee SH, Lee J, Song HS, Oh EH, Park TH, Hong S (2009). Single-carbon-atomicresolution detection of odorant molecules using a human olfactory receptor-based bioelectronics nose. Adv Mater 21(1): 91–94

[141] Lee SH, Kwon OS, Song HS, Park SJ, Sung JH, Jang J, Park TH (2012). Mimicking the human smell sensing mechanism with an artificial nose platform. Biomaterials 33(6): 1722–1729

[142] Lee SH, Park TH (2010). Recent advances in the development of bioelectronic nose. Biotechnol Bioprocess Eng 15(1): 22–29

[143] Malnic B, Godfrey PA, Buck LB (2004). The human olfactory receptor gene family. Proc Natl Acad Sci U S A 101(8): 2584–2589

[144] Buck LB (2004). Olfactory receptors and odor coding in mammals. Nutr Rev 62: S184–S188

[145] Lim JH, Park J, Ahn JH, Jin HJ, Hong S, Park TH (2013). A peptide receptor-based bioelectronics nose for the real-time determination of seafood quality. Biosens Bioelectron 39(1): 244–249

[146] Kiefer H, Krieger J, Olszewski JD, von Heijne G, Prestwich GD, Breer H (1996). Expression of an olfactory receptor in Escherichia coli: purification, reconstitution, and ligand binding. Biochemistry 35(50): 16077–16084

[147] Ko HJ, Park TH (2006). Dual signal transduction mediated by a single type of olfactory receptor expressed in a heterologous system. Biol Chem, 387(1): 59–68

[148] Krautwurst D, Yau K-W, Reed RR (1998). Identification of ligands for olfactory receptors by functional expression of a receptor library. Cell 95(7): 917–926

[149] Zhao H, Ivic L, Otaki JM, Hashimoto M, Mikoshiba K, Firestein S (1998). Functional expression of a mammalian odorant receptor. Science 279(5348): 237–242

[150] Wu L, Pan Y, Chen G-Q, Matsunami H, Zhuang H (2012). Receptor-transporting Protein 1 short (RTP1S) mediates translocation and activation of odorant receptors by acting through multiple steps. J Biol Chem 287(26): 22287–22294

[151] Zhuang H, Matsunami H (2007). Synergism of accessory factors in functional expression of mammalian odorant receptors. J Biol Chem 282(20): 15284–15293

[152] Zhuang H, Matsunami H (2008). Evaluating cell-surface expression and measuring activation of mammalian odorant receptors in heterologous cells. Nat Protoc 3(9): 1402–1413

[153] Kiely A, Authier A, Kralicek AV, Warr CG, Newcomb RD (2007) Functional analysis of a Drosophila melanogaster olfactory receptor expressed in Sf9 cells. J Neurosci Methods 159(2): 189–194

[154] Matarazzo V, Clot-Faybesse O, Marcet B, Guiraudie-Capraz G, Atanasova B, Devauchelle G, Cerutti M, Etiévant P, Ronin C (2005). Functional characterization of two human olfactory receptors expressed in the baculovirus Sf9 insect cell system. Chem Sens 30(3): 195–207

[155] Minic J, Persuy MA, Godel E, Aioun J, Connerton I, Salesse R, Pajot-Augy E (2005). Functional expression of olfactory receptors in yeast and development of a bioassay for odorant screening. FEBS J 272(2): 524–537

[156] Song HS, Lee SH, Oh EH, Park TH (2009). Expression, solubilization and purification of a human olfactory receptor from Escherichia coli. Curr Microbiol 59(3): 309–314

[157] Saito H, Chi Q, Zhuang H, Matsunami H, Mainland JD (2009). Odor coding by a mammalian receptor repertoire. Sci Signal 2(60): ra9

[158] N. Birbaumer, N. Ghanayim, T. Hinterberger, I. Iversen, B. Kotchoubey, A. Kübler, J. Perelmouter, E. Taub, H. Flor (1999). Insect antenna as a smoke detector Rings of single-walled carbon nanotubes Taxon sampling revisited. Nature 398: 3-4

[159] Baker T C and Haynes K F (1989). Field and laboratory electroantennographic measurements of pheromone plume structure correlated with oriental fruit moth behavior. Physiol. Entomol. 14: 1–12

[160] Karg G and Sauer A E (1994). Spatial distribution of pheromone in vineyards treated for mating disruption of the grape vine moth Lobesia botrana measured with electroantennograms. J. Chem. Ecol. 21: 1299–314

[161] Van der Pers J N C, Minks A K (1998). A portable electroantennogram sensor for routine measurements of pheromone concentrations in greenhouses. Ent. Exp. Appl. 87: 209–15

[162] A J Myrick, K-C Park, J R Hetling, T C Baker (2008). Real-time odor discrimination using a bioelectronic sensor array based on the insect electroantennogram. Bioinspiration & Biomimetics 3(4)

[163] Kevin S. Repasky, Joseph A. Shaw, Ryan Scheppele, Christopher Melton, John L. Carsten, and Lee H. Spangler (2007). Range-resolved optical detection of honeybees by use of wing-beat modulation of scattered light for locating land mines. Applied optics 46(15): 3007-3012

[164] Joseph A. Shaw, Nathan L. Seldomridge, Dustin L. Dunkle, Paul W. Nugent, Lee H. Spangler (2005). Polarization lidar measurements of honey bees in flight for locating land mines. Optics express 13(15): 5853-5863

[165] J.R. Riley, A.D. Smith (2002). Design considerations for an harmonic radar to investigate the flight of insects at low altitude. Computers and Electronics in Agriculture 35(2-3): 151-169

[166] Tony L. King, Frank M. Horine, Kevin C. Daly, and Brian H. Smith (2004). Explosives detection with hard-wired moths. IEEE Transactions on Instrumentation and Measurement 53(4): 1113-1118

[167] S. L. Utley, G. C. Rains, W. J. Lewis (2007). Behavoral Monitoring of microplitis croceipes, a parasitoid wasp, for setecting target odorants using a computer vision system. Transactions of the ASABE 50(5): 1843-1849

[168] Jean-René Martin (2004). A portrait of locomotor behaviour in Drosophila determined by a video-tracking paradigm. Behavioural Processes 67(2): 207-219

[169] Glen C. Rains, Samuel L. Utley, W. Joe Lewis (2006). Behavioral monitoring of trained insects for chemical detection. Biotechnology Progress 22(1): 2-8

[170] Glen C. Rains, Jeffery K. Tomberlin, Don Kulasiri (2008). Using insect sniffing devices for detection. Trends in Biotechnology 26(6): 288-294

[171] G. C. Rains, J. K. Tomberlin, M. D'Alessandro, W. J. Lewis (2004). Limits of volatile chemical detection of a parasitoid wasp, Microplitis croceipes, and an electronic nose: a comparative study. Transactions of the ASAE 47(6): 2145-2152

[172] www.nasalranger.com

[173] Nobuyuki Otsu (1979). A threshold selection method from gray-level histograms. IEEE Trans. Sys. 9(1)

[174] T.Romen Singh, Sudipta Roy, O.Imocha Singh, Tejmani Sinam, Kh.Manglem Singh (2011). A New Local Adaptive Thresholding Technique in Binarization, International Journal of Computer Science Issues 8(6): 2

[175] http://derekmolloy.ie/set-ip-address-to-be-static-on-the-beaglebone-black/

[176] www.ffmpeg.org

[177] http://michaelhleonard.com/cross-compile-for-beaglebone-black/

[178] http://linuxtv.org/downloads/v4l-dvb-apis/v4l2grab-example.html

[179] http://opencv.org/documentation.html

[180] http://docs.opencv.org/master/de/d91/tutorial_ug_mat.html

[181] http://docs.opencv.org/java/org/opencv/highgui/VideoCapture.html

[182] http://docs.opencv.org/modules/imgproc/doc/miscellaneous_transformations.html

[183] http://docs.opencv.org/java/org/opencv/core/Rect.html

[184] Pavlos Stathis, Ergina Kavallieratou, Nikos Papamarkos (2008). An Evaluation Technique for Binarization Algorithms. Journal of Universal Computer Science 14(18): 3011-3030

[185] O.D. Trier, A.K. Jain, (1995). Goal-directed evaluation of binarization methods. IEEE Trans. Sys. 17(12): 1191-1201