PARTIAL CHARACTERIZATION OF COYPU SCENT GLAND COMPOUNDS AND A NEW TECHNIQUE FOR COMPUTER-AIDED PHOTOGRAPHIC IDENITIFICATION OF INDIVIDUAL COYPU

A Thesis

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree Of

Master of Science

by

Steven Michael Finckbeiner

August 2005

ABSTRACT

Coypu are invasive rodents that are ravaging wetlands across their introduced range. Coypu have a sexually dimorphic (male larger) anal gland (AG) used in scent marking behavior. Using GC-MS, this study looked for a similar dimorphism in compounds produced by the AG. Male AG extract contained 15 compounds. Of the 15 compounds, 6 are fatty acids of known structure. Nine of the compounds are partially characterized farnesene isomers. Each male AG chromatograph had the same compounds present, but in differing proportions. This suggests that coypu can use AG secretion to identify individual males. Such knowledge of compounds used in coypu chemical communication could be useful in developing coypu-specific attractants. The second part of this study used a computer program to identify individual coypu by marking and matching whisker-insertion patterns. It was found that whisker insertion pattern can be used as a 'fingerprint' to aid in the photographic identification of individual coypu.

BIOGRAPHICAL SKETCH

Steven Finckbeiner was born in Chalmette, Louisiana in 1979. Steven grew up in Laplace, a town near Lake Ponchartrain outside New Orleans. He graduated fifth in his class from East St. John High School in Reserve, Louisiana in 1997. Steven was awarded a five year Louisiana Honors Scholarship and a Chancellor's Honor Scholarship to attend Louisiana State University in Baton Rouge. While a student at LSU, Steven did undergraduate research on fish olfaction in the labs of Dr. John Caprio and Dr. Richard Bruch. He graduated from LSU *summa cum laude* in 2001. Steven taught eighth grade special education math and science at the Glade School before being accepted in the Field of Neurobiology and Behavior at Cornell. At Cornell, Steven was awarded a Cornell University Fellowship. He was also a teaching assistant in Cornell's Introductory Biology course for two years.

I dedicate this thesis to my family, my first and greatest teachers.

Without them I am nothing.

ACKNOWLEDGEMENTS

I would like to acknowledge the help and guidance of my committee members and co-chairmen, Dr. Tom Eisner, Dr. Robert Johnston, and Dr. Bruce Land. Their patience and understanding is much appreciated. I would also like to acknowledge the essential contributions of Dr. Greg Linscombe, Dr. Dale Nolte, Dr. Vic Nettles, Dr. Jeremy Miller, Steve Kendrot, Susan Jojola, and Lauren Nolfo. Without their help I would never have gotten coypu anal glands for chemical analysis. They were also involved with the development and maturation of many of the ideas presented in the following thesis. Dr Jerrold Meinwald, Dr. Andrew Taggi, Dr. Athula Athygalle and Hyuejen Loo (?) performed the chemical tests central to the first part of this thesis. I would also like to acknowledge my good friend and fellow Wildcat, Bret Acosta. He taught me everything I know about how to find and observe coypu in nature. Bruce Land taught me everything I know about computer programming. The whisker programs are the result of his ability to teach the unteachable. Dr. John 'Doc' Caprio helped me find coypu people in Louisiana and also bounced ideas back and forth in their early stages. Thanks Doc.

My research was funded through a fellowship and teaching assistantships furnished by Cornell University. Dr. Tom Eisner provided direct monetary support of this research. I am also indebted to the Louisiana Department of Wildlife and Fisheries and the United States Department of Agriculture, Animal and Plant Health Inspection Services for access to knowledgeable employees and recently killed coypu.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

INTRODUCTION: ORIGIN, INVASION, AND CONTROL

The stakes involved in coypu research, as measured by the steady annual loss of crucial wetlands to coypu herbivory, are too high *not* to point out at the beginning of this thesis. Feral coypu are destructive pests. To fully understand the importance and scope of the work detailed in this thesis, it is necessary to understand the importance and scope of feral coypu as pests. What follows is meant to highlight where coypu have come from, where they have been, and where humans will allow them to go.

Multiple current hypotheses of rodent systematics support a single colonization event of South America by caviomorph rodents 30 million years ago (Houchon and Douzery 2001). From this initial event, caviomorphs spread across South America and evolved into the most ecologically and behaviorally diverse assemblage of rodents on Earth (Tang-Martinez 2003). Caviomorphs evolved into monogamous antelope-like forms (maras, (Taber and McDonald 1992)), large herding forms ecologically and behaviorally similar to the hippopotamus (capybaras, (Herrera and McDonald 1993), and even tree dwelling porcupines with prehensile tails (Candela and Morrone 2003). In the temperate marshlands of Patagonian South America, coypu (*Myocastor coypus*) evolved to fill the niche associated with beaver and muskrats in the Northern hemisphere (Figure 1). And there they lived relatively undisturbed, hunted first by Amerindians and then the Spanish until their voracious appetite and lustrous fur were noticed by enterprising businessmen.

Figure 1: Natural range of coypu, *Myocastor coypus*. (Adapted from Woods et al 1992)

Coypu, or nutria as they are called outside their native range, were exported to many parts of the world for various reasons in the early 1900s. Chief among these were fur farming, control of canal clogging water hyacinth,

and meat. While still farmed as a cottage industry in parts of Europe, coypu have never successfully caught on as an economically viable meat source. It also became quickly known that even coypu wouldn't eat water hyacinth. It was for fur that coypu were eventually taken to places like Louisiana and Maryland in the United States (Figure 2), England, France, and Poland in Europe, Israel, and even a volcanic crater lake in Africa (Carter and Leonard 2002). Most coypu farmers were people who thought they would get rich quick raising a giant rat that made expensive fur. When faced with the cost of building concrete pens to contain a coypu's massive jaws and correspondingly massive feed bills, many coypu farmers simply opened their cages and released their animals. Destructive storms, particularly hurricanes in Louisiana and Maryland, helped coypu escape farms and also spread them further inland. One hurricane that struck Avery Island in Louisiana is credited with destroying the coypu farm of EA McIllhenny, the inventor of Tabasco sauce. McIllhenny's coypu are popularly believed to be the source of most of the coypu that spread across Louisiana and the coast of the Gulf of Mexico. In reality, coypu were released many times in many places by now unknown private citizens and state and federal agencies intent on making coypu another source of fur revenue. Before it was completely abandoned in the 1950s, coypu farming had succeeded in repeatedly seeding local wetlands with a population of well fed, reproductive age coypu (Carter and Leonard 2002).

Figure 2: Range of feral coypu, Myocastor coypus, in North America. Adapted from LeBlanc (1994).

In the late 1940s, coypu started showing up in leg hold traps set for muskrats and beavers. Muskrat numbers began to decline drastically wherever coypu were found. It is still unknown whether coypu simply out ate muskrats or whether they actively forced them off productive marsh. Eventually muskrats went commercially extinct in Louisiana, a state that had produced more muskrat pelts than the rest of North America for many years (Evans 1970, Bounds and Carowan 2000). Before the invasion of coypu, trappers used to be able to walk the marsh and skin muskrats in the field,

leaving the carcasses in the marsh and bringing the pelts back to market. With no muskrats, trappers had no choice but to go after the heavier and harder-to-process coypu. Compared to muskrats, coypu are incredibly hard to skin. Trappers must bring the entire animal back to camp before it can be properly skinned. To do this in a cost effective manner, a trapper needed larger boats than the home-made pirogues used previously. This was an option open only to trappers with the money to buy them. The first victim of coypu expansion was the muskrat and the culture of the poor trappers who pursued them (Tarver et al 1987, Lowery 1974, Acosta personal communication).

Despite this loss and the difficulty inherent in trapping coypu, coypu fur was still one of the most valuable furs in the world. As long as people wanted coypu fur coats and hats, trappers found a way to get coypu fur to market. Over one million coypu pelts were exported annually between 1960 and 1985 from Louisiana alone (Figure 3). Since these numbers are based on an export tax on fur sellers who only sell the best furs they receive from trappers, this number is actually an underestimate (Kinler et al 1999). Any animal that could yearly lose *at least* one million individuals for more than two decades in only one part of its introduced range was no longer being trapped. It was being fished. Such numbers only hinted at the coming disaster. At the zenith of the fur industry, no one could foresee a time when people would not wear fur.

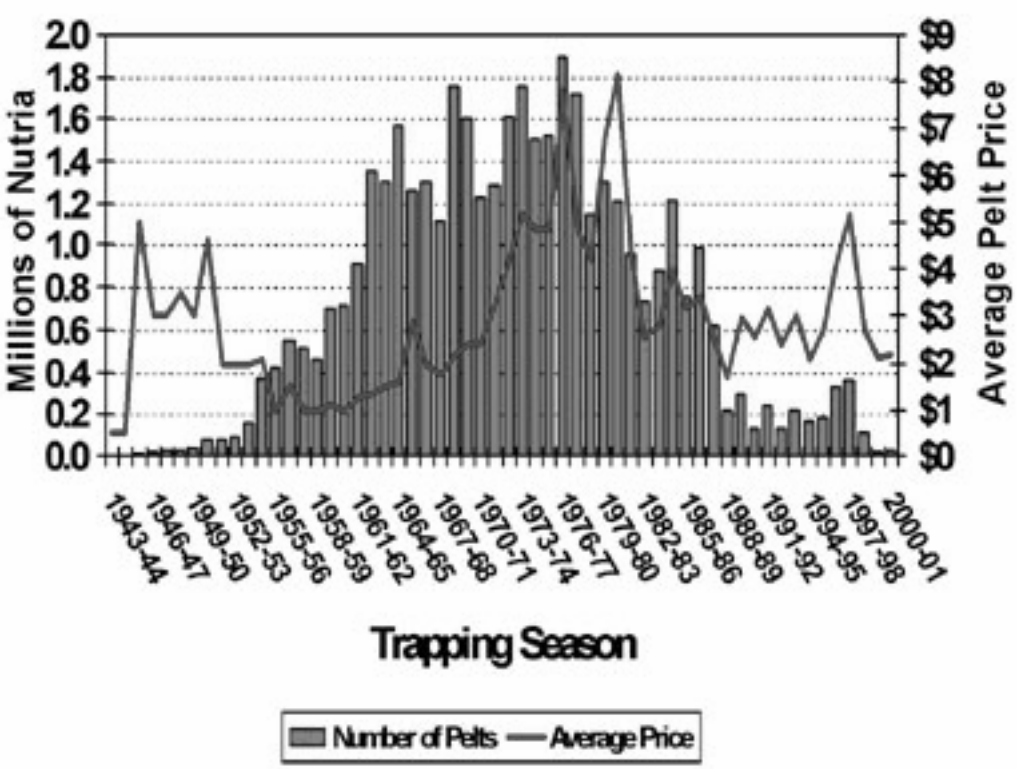


Figure 3: Average pelt price and number of coypu, *Myocastor coypu*, harvested in Louisiana from 1943 to 2001. SOURCE: Comparative Takes of Fur Animals, LDWF Annual Reports, 1944-2001. Used with permission.

Of course, that is exactly what happened. In the late 1980s, the campaigns of various animal rights activist groups had commercially eliminated one of the oldest of all human customs: wearing animal fur. When people stopped wearing fur clothing, the demand for coypu fur fell. It became unprofitable for trappers to carry on with what was left of their livelihood. That this happened at the same time as the expansion of better paying jobs in the booming oil industry of the 1980s only further exacerbated the problem (Gomez 1998).

With their human predators literally out of a job, coypu began to expand inexorably across their introduced range. Reports of “eat-outs,” areas of marsh grass eaten down to the bare mud, began to surface (Linscombe 1992,

Mouton et al 2001) (Figure 4). Eat outs are only the highly visible end stage of a systemic coypu infestation. Once the sustaining network of grass roots are dug up and eaten by hungry coypu, the bare mud left behind is soon eroded by wave action and converted to open water. It is estimated that 100,000 acres of coastal marsh are seriously damaged by coypu every year in Louisiana alone (Mouton et al 2001). These marshes nourish the larval stages of virtually every food organism commonly eaten by humans and larger aquatic predators. Marshes lost to coypu are also essential to protecting coastal cities from tidal surges generated by hurricanes (Chabreck 1972, Lowery 1974, Valentine et al 1972). While marsh loss is the most serious and dramatic result of coypu range expansion and overpopulation, it is not the only result. Efforts to restore cypress swamps have also been thwarted when coypu ate gnawed juvenile trees or ate planted seedlings (Blair and Langlinias 1960, Conner and Toliver 1987, Myers et al 1995). Tunneling coypu have compromised earthen flood control structures (Lowery 1974, Mouton et al 2001). As evidence of such visible damage began to build up, wildlife management professionals realized that coypu had become a classic invasive species. They had no natural predators and were expanding to the detriment of their new environment and its native species (Bounds and Carowan 2000, Carter and Leonard 2002).

Efforts to control or even eradicate coypu from their introduced range began soon after the first reported eat-outs. With one exception, all eradication efforts have failed (Carter and Leonard 2002). The exception, the eradication of coypu in the marshes of southern Great Britain, also became the source of most of our knowledge of coypu physiology and behavior. L.M. Gosling and his team of trappers took seven years to systematically kill every

coypu in the 100,000 acre marshes of East Anglia. Gosling's success is still unique for several reasons. First, coypu were never as widespread in Great Britain as they still are in coypu 'hot spots' like the Gulf Coast of Louisiana, the Chesapeake Bay region of Maryland, or the Marais Poitevin of France. Second, Great Britain's harsh winters led to severe seasonal mortality. In fact, Gosling capitalized on this by increasing the number of trappers following colder winters (Gosling and Baker 1989). Great Britain is now coypu-free.

The rest of the coypu containing-world has spent considerable monies on analyzing and facing their respective infestation problems. Locations with relatively low grade infestations (i.e. Oregon) take the 'open season' approach. Coypu can be killed on sight any time of the year by licensed trappers (Anonymous 2005). The entire southern part of Louisiana is a labyrinthine mixture of marsh, swamp, and river delta. And it is all infested with coypu. An area approximately 20 times the size of Gosling's eradication zone, Louisiana's continuous wetlands are almost unanimously seen as too big and too remote an area to be seriously considered for systematic eradication. Instead, Louisiana coypu biologists decided to tap the deep trapping traditions present in the state by creating and funding the Coastwide Nutria Control Program in 2002. The state began offering a four dollar per coypu (measured by severed tails) bounty to anyone who signed up for the program. In its first year, 300,000 coypu were killed. While this is far short of the number of coypu killed yearly at the peak of the fur industry, it is still a substantial reduction of coypu (Marx et al 2003).

Taking a hard line approach, the state of Maryland and a host of government partners have decided to attempt the second successful eradication effort. Starting in Maryland's Blackwater National Wildlife Refuge,

coypu are being systematically eliminated as part of the Maryland Nutria Project. Their plan, which will eventually cover the entire Chesapeake watershed, is modeled directly on Gosling's tactics and with his cooperation (Steve Kendrot, personal communication). Initial results of this massive undertaking have been positive. Blackwater NWR is now coypu free (Fahrenthold 2004). Monitoring traps set behind the trapping front catch few if any coypu. But, if even one pregnant female coypu is missed, the entire program could be for naught. Continuous monitoring using leg hold traps is expensive, ties up much needed trappers, and also does little to catch coypu that have already eluded the traps at least once. Indeed, this is one of the central problems that the Maryland coypu biologists have identified with their eradication plan (Steve Kendrot, personal communication).

Better methods of monitoring coypu need to be developed if any coypu eradication effort larger than that of England is to be completed successfully. Meaningful data on animal populations is based on mark recapture studies. An animal is caught once, marked, and released at the site of capture. The proportion of marked animals to unmarked animals caught in the next trapping period is used to estimate population size. As the author learned during his first field season in the spring of 2003, coypu are incredibly hard to trap alive. Live trap success rates of 5%/trap night are considered acceptable (Doncaster and Micol 1989). Once a coypu is trapped, they often never go into another trap again. This makes the initial assessment of an area's coypu population, data vital to determining a trapping operation's success, very hard and expensive to obtain (Simpson and Swank 1979, Reggiani et al 1995). These difficulties prompted members of the USGS Nutria Group to place improvements in lure and trap technology at the top of its 'client needs' list.

Continuing in Gosling's tradition of learning about coypu while destroying them, the two free standing chapters below attempt to provide useful tools for improving coypu lure and trap technology while detailing new knowledge gained about coypu as living products of evolution. The first chapter features the partial characterization of compounds found in coypu scent gland. Knowledge of compounds used by coypu in chemical communication could be very useful in creating lures that enhance trapping efficiency and specificity. This also has implications for our basic understanding of coypu chemical communication and physiology. One group of compounds found in coypu scent gland is found in a multitude of other species in an evolutionary pattern described for the first time in this thesis. The second chapter details the programming and implementation of a computer program that helps a user identify individual coypu from photographs using the pattern of white whiskers on the side of a coypu's face. This has direct applications to obtaining mark recapture data in a way that does not involve actually capturing coypu. As an added bonus, the program will work on any animal that has observable whiskers.

Figure 4: A coypu "eat-out" in Big Branch National Wildlife Refuge, Lacombe LA. TOP: Healthy marsh can be seen in the foreground and on the horizon. Paths beaten into the mud by coypu are indicated with arrows. BOTTOM: A coypu leaving the water near another eat-out in Big Branch. Photographs by the author.

CHAPTER 1

PARTIAL CHARACTERIZATION OF COYPU SCENT GLAND COMPOUNDS

INTRODUCTION

Scent marking is an important form of social communication in many mammals. Scent marks can be deposited urine (Beynon and Hurst 2004, Nevison et al 2003, Colins et al 2001, Roberts et al 2001, Young and Henke 1999, Sillero-Zubiri and Macdonald 1998, Miller et al 1998, Johnston et al 1997), feces (Buesching and Macdonald 2004, Ding et al 1998, Ben-david et al 1998, Roper et al 1993), secretion-storing hairs ("osmetrichia" of deer, Ajmat et al 1999), or secretions of specialized exocrine glands (Rosell and Schulte 2004, Irwin et al 2004, Manaf et al 2003, Arkin et al 2003, Lawson et al 2000, Scully et al 2000, Bel et al 1999, Arkin et al 1999, Barling and Shirley 1999, Atoji et al 1998, Welsch et al 1998, Herrera 1992, Zeller et al 1998, Fadem and Schwartz 1986, Gorman et al 1984, Jones and Plakke 1981, Wellington et al 1979, Smith and Hearn 1979). Mammalian scent marks, no matter their source, are invariably complex mixtures of many compounds. It is believed that this complexity is due to the amazing amount of information that an individual mammal can gather from a scent mark. Mammal species use compounds in their scent marks to convey individual identity (Hurst et al 2005, Thom and Hurst 2004, Woodley and Baum 2004, Johnston 2003, Nevison et al 2003, Lawson et al 2000, Wilcox and Johnston 1995, Johnston et al 1995, Johnston 1993), kinship (Hurst et al 2005, Johnston 2003, Mateo 2003, Safi and Kerth 2003, Dobson and Jouventin 2003, Mayeaux and Johnston 2002, Roberts et al 2001, Lawson et al 2000, Sun and Muller-Schwarze 1998, Heth et al 1998, Todrank et al 1998, Sun and Muller-Schwarze 1997), reproductive

state (Ferkin et al 2004, Wirant and McQuire 2004, Zuri et al 2003), territory ownership (Ben-david et al 2005, Begg et al 2005, Rosell and Schulte 2004, Irwin et al 2004, Rostain et al 2004, Barja et al 2004, Buesching and Macdonald 2004, Banack and Grant 2003, Begg et al 2003, Manaf et al 2003, Nevison et al 2003, Zub et al 2003, Miller et al 2003, Schultz and Wilson 2002, Safi and Kerth 2003, Drea et al 2002, Rosell and Bjorkoyli 2002, Revilla and Palomares 2002, Briscoe et al 2002, Rosell and Sundsdal 2001), infection (Zala et al 2004), and even emotional state (Lai and Johnston 2002, Ackerl et al 2002).

Murine rodents, especially laboratory mice, have been the models of choice when studying the chemistry of scent marking (Beauchamp and Yamazaki 2003). Few caviomorph rodents have been studied in this regard (guinea pigs, Beauchamp 1974). Coypu, *Myocastor coypus*, are beaver-like caviomorph rodents native to Patagonian South America. Following accidental or intentional release from fur farms in the 1930s, coypu, or nutria as they are known in their introduced range, now exist as destructive wetlands-destroying pests in large feral populations around the world (Carter and Leonard 2002).

Coypu are polygynous and form long lasting social groups in both their natural and introduced ranges. A dominant male controls a marsh territory with multiple related females and their offspring. Other adult males are excluded. Younger males are tolerated by the dominant male until they reach maturity (Doncaster and Micol 1989, Dagault and Saboureau 1990). Within this simple and typically mammalian social framework, coypu use two documented sources of odorant signals to scent mark.

In the first study of free ranging coypu in their native habitat, Guichon et

al (2003) observed dominant male coypu urine marking tunnel entrances and other coypu following aggressive interactions. Other studies (Gosling and Wright 1994) have observed coypu scent marking with the anal scent gland (ASG) by dragging the area near the base of the tail on the ground or rubbing it over grass or other protruding objects. Coypu tend to ASG-mark before entering water, after exiting water, and when entering tunnel entrances or floating nests. ASG secretion may also contribute to grooming secretions.

The ASG is an unpaired sebaceous gland located ventral to the large intestine near the anus. Internally, yellowish secretory lobules empty into a central duct that feeds into 4-6 nipple-like openings. Ensheathed in skeletal muscle, the gland's opening can be rapidly everted through the anus (Figure 5). Using data from the landmark Great Britain coypu eradication program in the 1970s, Gosling showed that the ASG is sexually dimorphic in size. Male ASGs are 10% larger than female ASGs. Gosling's data also showed distinct seasonal variation in size of male ASG, but not female ASG. This seasonal variation is not correlated with a defined breeding season. Coypu breed year round in their natural and introduced range. Gosling found that male ASGs are larger in the winter when the marsh grass has receded and male-male competition (measured by incidence of fighting scars) is at its peak (Gosling and Wright 1994).

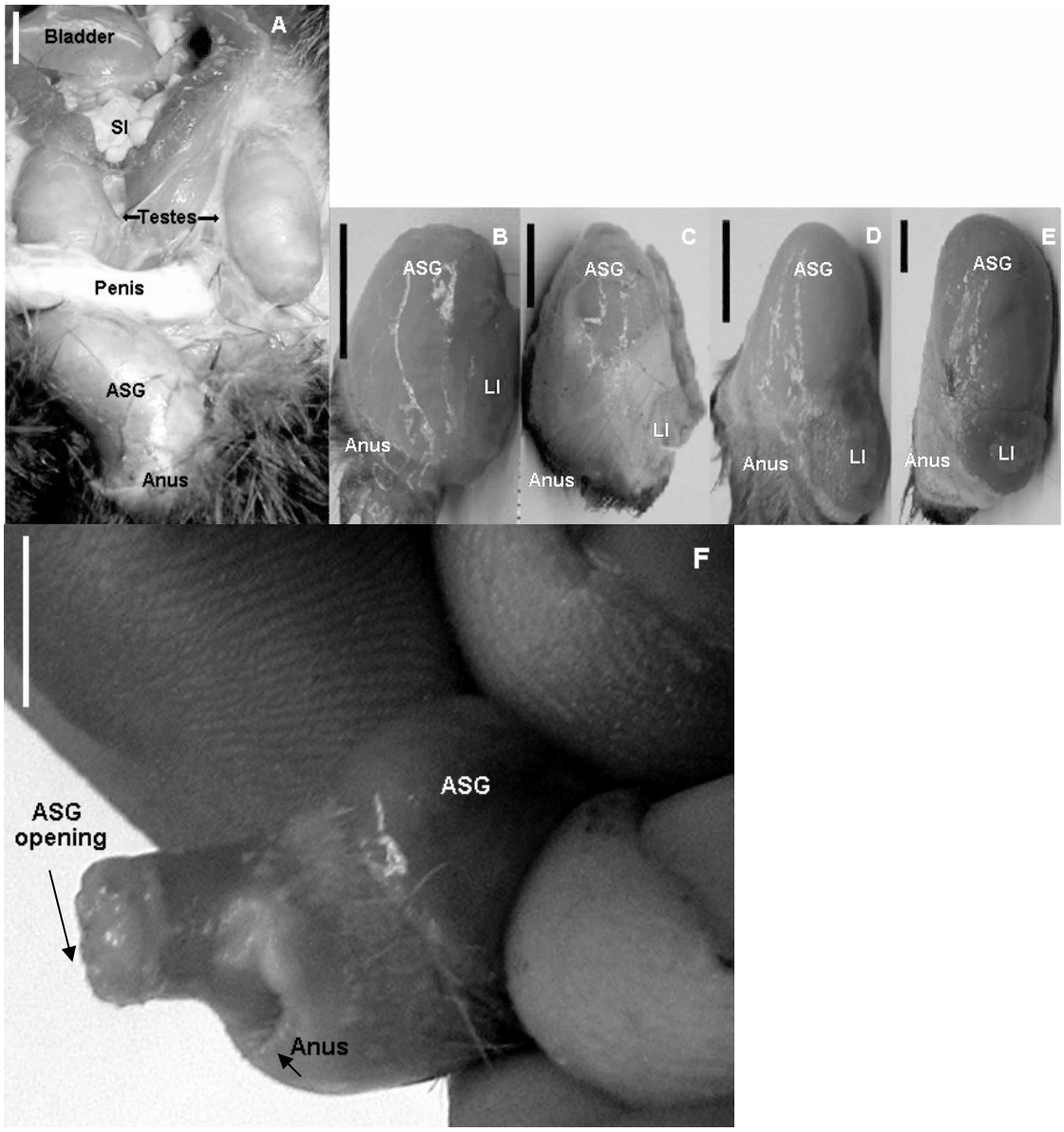


Figure 5: Coypu anal scent gland (ASG). A: Ventral view of ASG in relation to other abdominal organs of adult male coypu (6004). B: ASG of juvenile female coypu (6002). C: ASG of adult female coypu (6013). D: ASG of juvenile male coypu (6017). E: ASG of adult male coypu (6004). F: ASG of juvenile male coypu (6017) with ASG opening everted by manual pressure. B-F, up is ventral. Scale bars = 1 cm. Coypu ID numbers correspond to those in Table 1. SI = Small intestine. LI = Large Intestine. Photographs A and F by the author. B-E by Dave Manns.

Taken together, available evidence suggests that the ASG secretion is a signal involved in male-male aggressive interactions. Gosling hypothesized that increasing the size of the ASG would allow a male coypu to scent mark more. A male who can scent mark more would be signaling that he was healthy enough to expend this extra energy, and was therefore a foe to be avoided. Increased scent marking would thus allow individual coypu to resolve territorial disputes with less fighting (Gosling and Wright 1994). For a chemical signal to perform this role, it needs to have two major design parameters. First, it needs to have compounds that are specific to male secretions and unequivocally identify the sender as male. Second, the secretion should have a component or components that vary in a way that identifies the sender as a specific individual. This design parameter is essential to a species that forms long-lasting social groups that depend on remembering the outcome of social interactions with specific individuals. A scent mark that says “Male” does not help the receiver make a decision about whether to enter a territory. However, if the scent mark says “Male, my name is X,” the receiver can then remember the outcome of his interaction with X and whether it is worth the energy to enter X’s territory again (Gosling and Roberts 2001). How the two design parameters are coded is also of interest. Using the digital/analog approach of Sun and Muller-Schwarze (Sun and Muller-Schwarze 1998), it is possible to see if a signal uses presence or absence of compounds (digital), varying proportions of compounds (analog), or both to code for sex and individuality.

Using tissue from individuals killed in control measures, this study used GC-MS analysis to decipher the coypu ASG. The two goals of the present study were: 1) to look for the two previously mentioned design features and

their coding scheme and 2) to characterize the compounds found in coypu ASG.

METHODS

All tissue used in this study was obtained from coypu killed by employees of the Louisiana Department of Wildlife and Fisheries (LDWF) as part of ongoing coypu control measures in the state of Louisiana. Coypu killed were taken from the Atchafalaya National Wildlife Refuge in January, 2004. Three male coypu (two adult, one juvenile) and two female coypu (one adult, one female) were used in this study (Table 2). Males with scrotal testes were defined as adult. Females with observable embryos were defined as adult.

Table 1: Age, sex, and mass of coypu used in ASG study. ID number is that assigned by LDWF trappers at time of death.

ID #	Age Class	Sex	Mass (kg)	Lab ASG Extract Analyzed by
6004	Adult	Male	6.28	*JM
6009	Adult	Male	3.22	AA
6017	Juvenile	Male	1.48	JM
6013	Adult	Female	4.26	JM
6002	Juvenile	Female	0.86	JM

*JM=Jerrold Meinwald group. AA=Athula Attygalle group.

Carcasses were brought to the field station of the LDWF Fur and Refuge Division (New Iberia, LA) within 6 hours of death. ASGs were immediately dissected free of surrounding tissue and frozen at -20 ^{o}C until they were shipped on ice overnight to Cornell University and stored at -80 ^{o}C. After thawing, yellow glandular tissue was dissected free of surrounding skeletal muscle and homogenized in an 8 mL glass vial with a microspatula. Dissected glandular tissue was refrozen until analysis with GC-MS.

ASG tissue was analyzed in two different labs. First, samples from 2 male and 2 female coypu were analyzed using coupled GC-MS in the lab of

Jerry Meinwald at Cornell University (Ithaca, New York). For GC-MS, 1.5 mL dissected and homogenized ASG tissue was extracted with 1mL pentane. One µL aliquots of extracts were analyzed with a Hewlett–Packard HP5890A gas chromatograph linked to a Hewlett–Packard mass-selective detector (MSD; 70eV EI), using a DB5-MS coated column (30 m x 0.25 mm, J & W Scientific, Folsom, CA). Oven temperature was kept at 35 ^{o}C for 5 min and raised 5 ^{o}C /min to 275. Oven temperature was then held at 275 ^{o}C for 20 minutes.

GC-MS was also carried out in Dr. Athula Attygalle's lab at Stevens Institute of Technology (Hoboken, New Jersey). For gas chromatographic analysis, 0.5 g dissected and homogenized glandular material of one adult male coypu was extracted with 1 mL of dichloromethane (DCM). One µl aliquot of the extract was analyzed on a Shimadzu-QP5050 GC-MS installed with a ZB-FFAP coated column (30 m x 0.25 mm), and electron-ionization (70eV, EI) was obtained. Oven temperature was kept at 40 ^{o}C for 3 min and raised 6 ^{o}C/min to 240 ^{o}C.

For electrospray ionization analysis, 0.5g glandular tissue was extracted with 1 mL of dichloromethane and concentrated into 20 µL. About 10 µL of the dichloromethane was dissolved in acetonitrile (0.7 mL) and 1.0 % aqueous ammonia and analyzed by electrospray-ionization mass spectrometry using a VG Quattro I triple quadrupole instrument under positive and negative modes.

To test for UV reflectance, dissected ASG was exposed to long wavelength ultraviolet light and photographed.

RESULTS

From GC-MS analysis of pentane extracted coypu ASG performed in

the Meinwald lab, it was found that there are size, sex, and individual differences in the compounds made by the ASG. GCs of pentane extracted female ASG (n=2) and male ASG (n=2) show the presence of 1 compound with a boiling point near coypu body temperature that increases in abundance with body mass (peak A, Figures 6, 7, 8, and 9). Peak A was the only peak found in female ASG extract. GCs of pentane extracted male ASG (n=2) show the presence of 15 compounds (labeled in Figures 8 and 9) that occur in both samples. Of these 15 compounds, 2 (peak A and peak B) have boiling points near coypu body temperature. One of these compounds is the previously mentioned peak A. The other compound, peak B, was only found in the ASG of the adult male. The other 13 peaks found in both male ASG extracts have boiling points higher than coypu body temperature. The abundances of these peaks do not seem to correlate with age or size. Mass spectra from all peaks identified in pentane extracted adult male ASG by the Meinwald lab unequivocally identified one compound, (E)-beta-farnesene (Peak C, Figures 8 and 9, MS spectra in Figure 10).

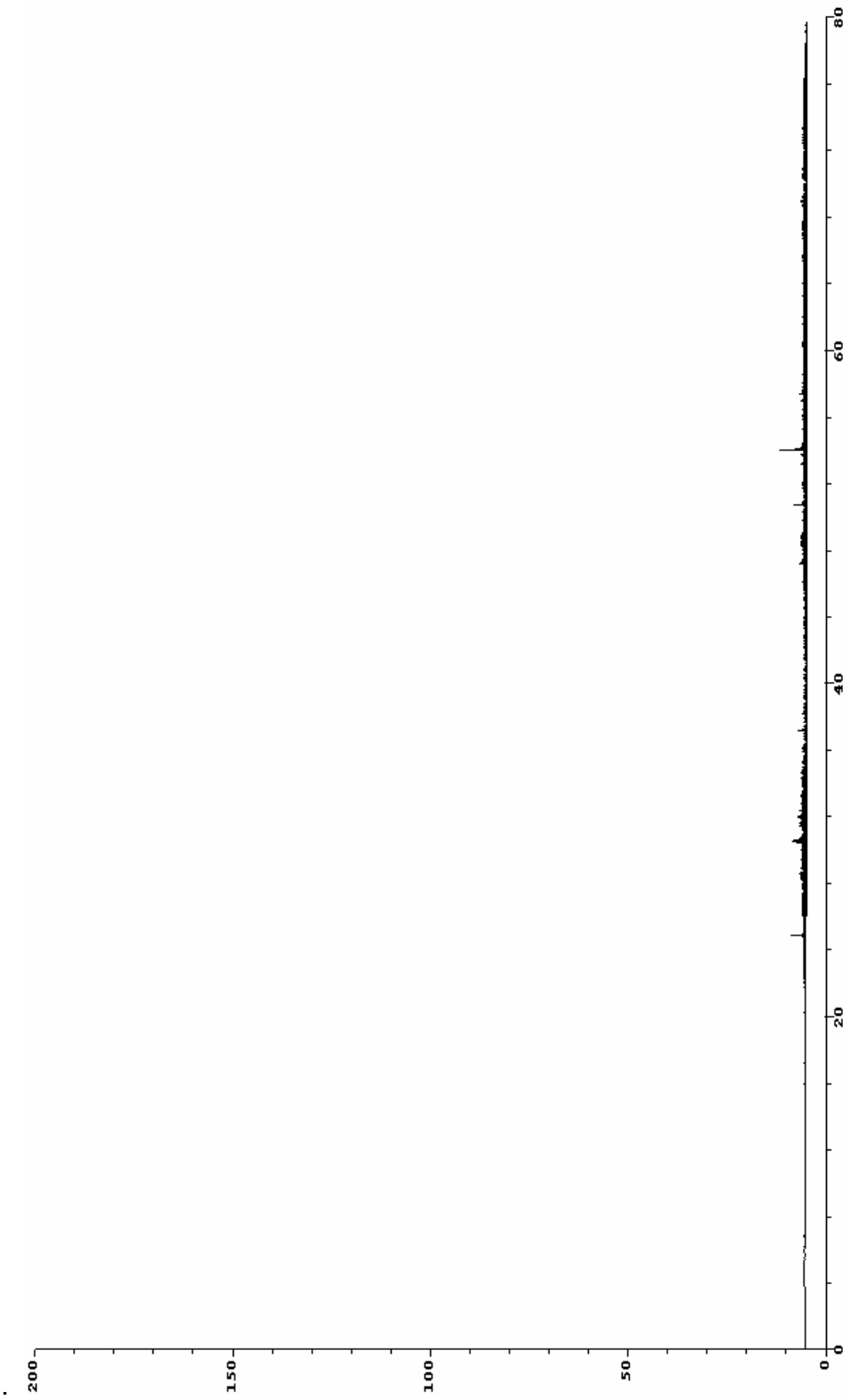


Figure 6: Gas chromatograph of pentane-extracted juvenile female coypu ASG (6002).

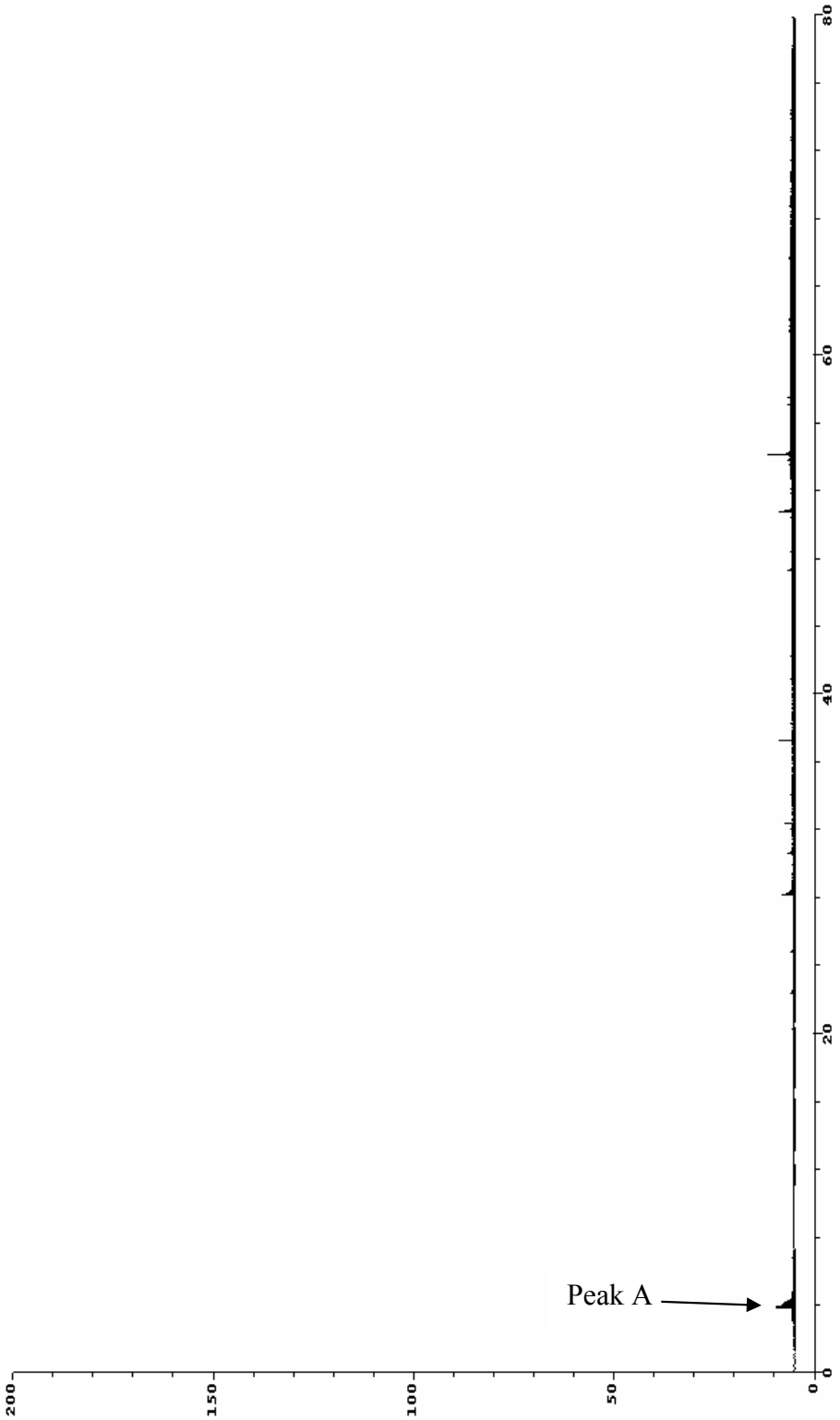


Figure 7: Gas chromatograph of pentane extracted adult female coypu ASG (6013).

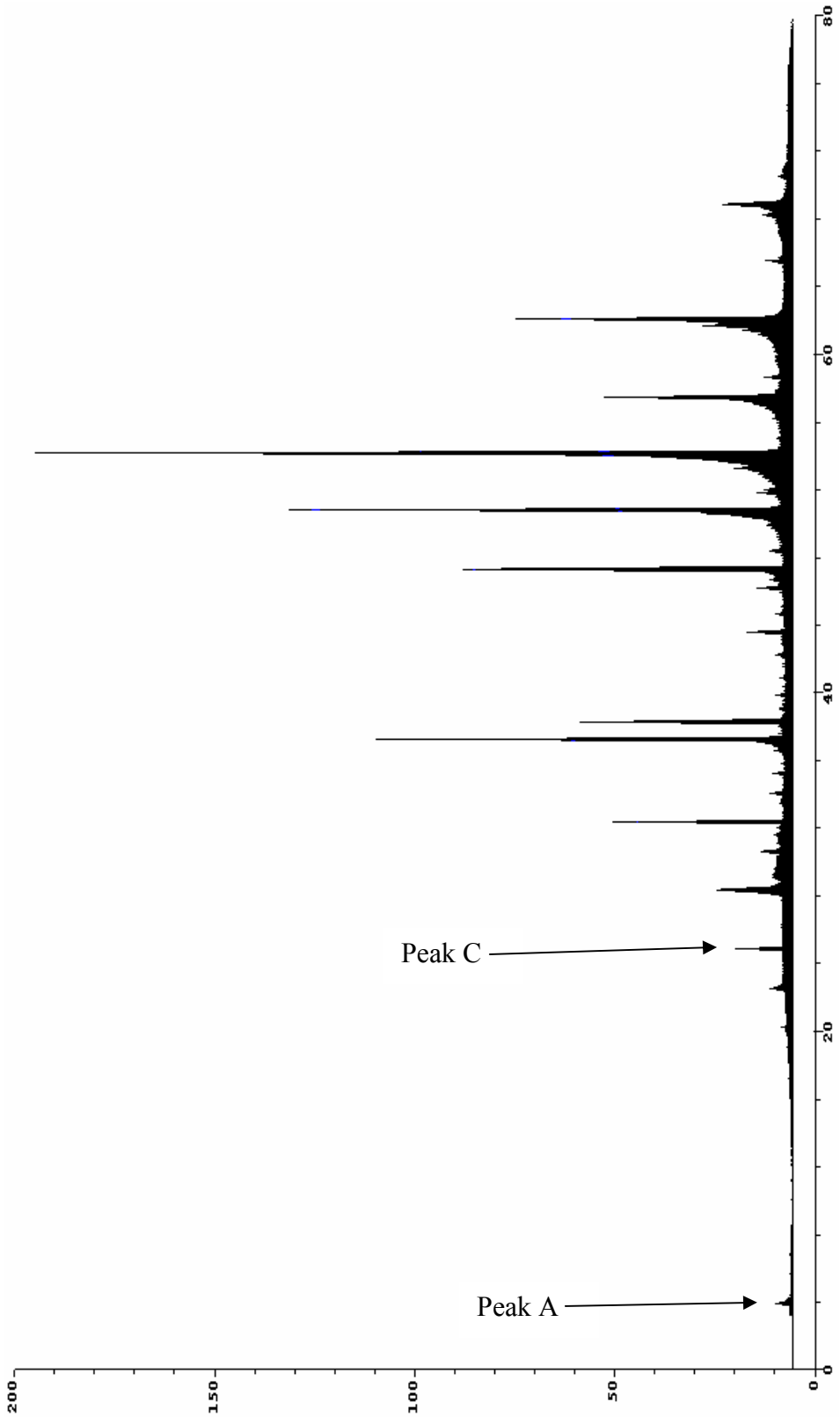


Figure 8: Gas chromatograph of pentane extracted juvenile male coypu ASG (6017).

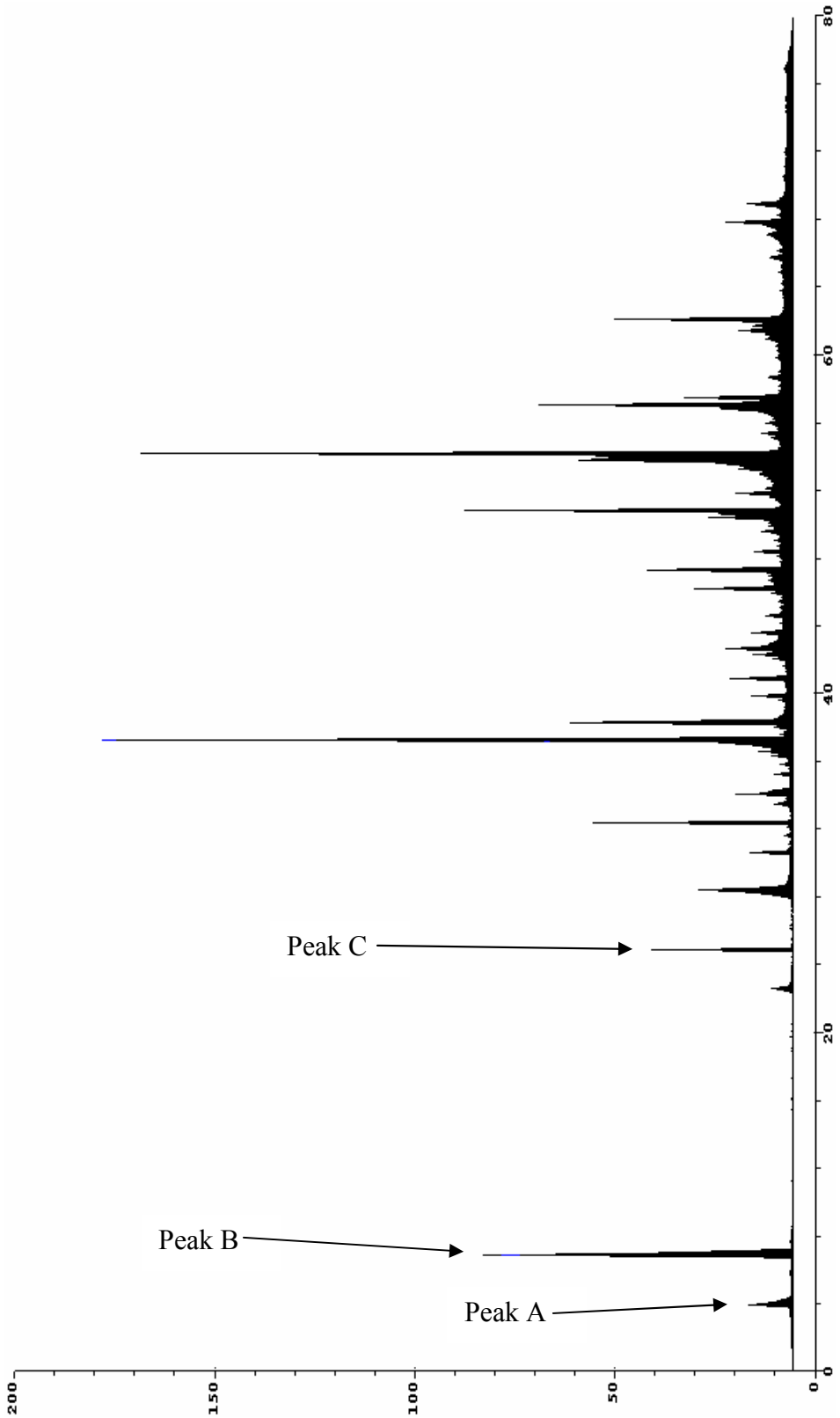


Figure 9: Gas chromatograph of pentane extracted adult male coypu ASG (6004).

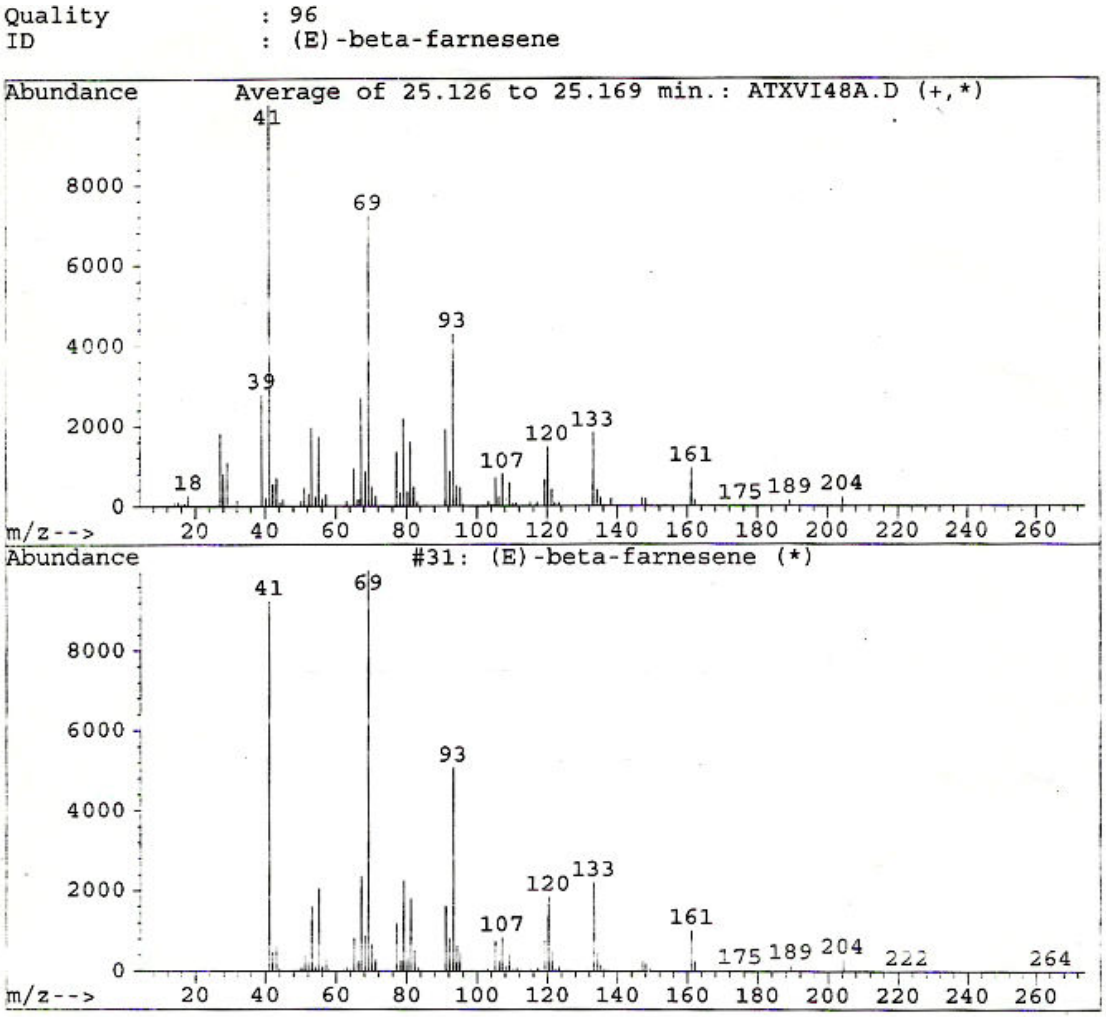


Figure 10: TOP: Mass spectra of peak 4 (Figure 8). BOTTOM: Library reference of pure (E)-beta-Farnesene.

GC-MS of DCM extracted adult male ASG performed in the Attygalle lab also found 15 major peaks (Figure 11, Table 2). MS analysis identified 6 of the peaks as the long chain fatty acids octanoic acid (peak 3), decanoic acid (peak 4), dodecanoic acid (peak 7), myristic acid (peak 9), palmitic acid (peak 11), and stearic acid (peak 14).

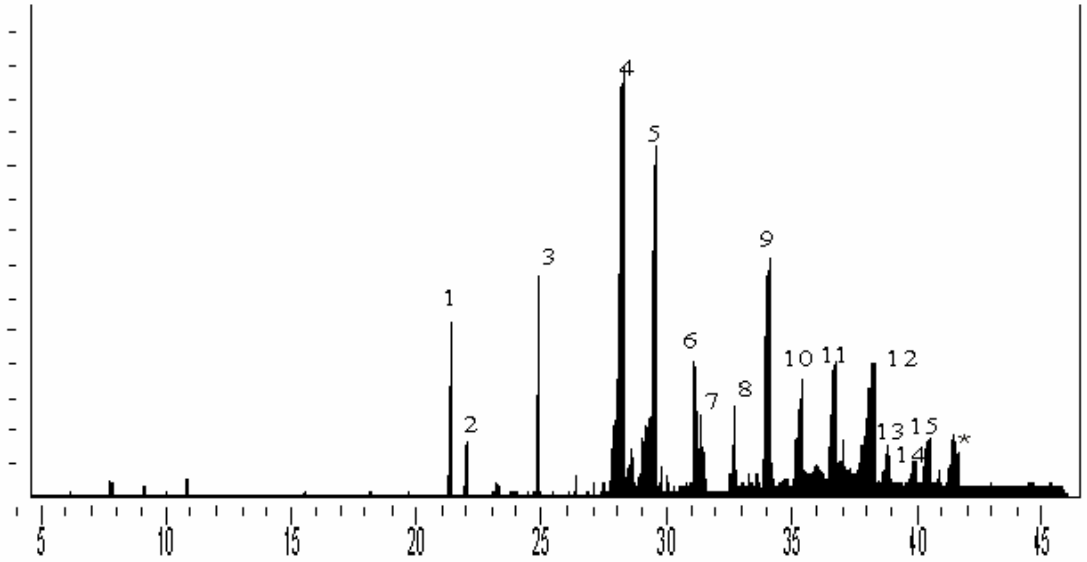


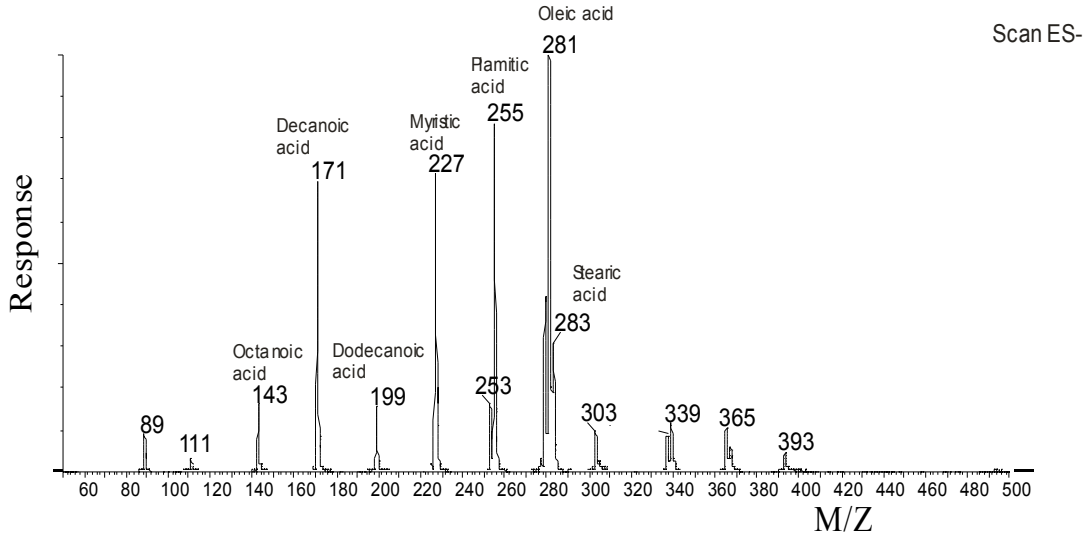
Figure 11: Gas chromatograph of DCM extracted adult male coypu ASG (6009).

Table 2: Volatile compounds characterized from *Myocastor coypus* glands (peak numbers correspond to the GC shown in Figure 10).

Peak #	Compound
1	a farnesene derivative
2	a farnesene derivative
3	Octanoic acid
4	decanoic acid
5	a farnesene derivative
6	a farnesene derivative
7	dodecanoic acid
8	a farnesene derivative
9	Myristic acid
10	a farnesene derivative
11	Palmitic acid
12	a farnesene derivative
13	a farnesene derivative
14	stearic acid
15	a farnesene derivative
*	Impurity (phthalate)

An ESI mass spectrum recorded using the negative-ion mode (Figure 12) confirmed the presence of octanoic acid, decanoic acid, dodecanoic acid, myristic acid, palmitic acid, oleic acid, and stearic acid in the DCM-extracted ASG.

Figure 12: ESI MS (negative-ion mode) of DCM extracted adult male coypu ASG (6009).



A tandem MS/MS examination of the extract using the positive-ion mode showed the presence of a series of large molecules that produces the ion *m/z* 203 upon fragmentation. Figure 13 shows a product ion spectrum recorded from *m/z* 564. The presence of *m/z* 203 and *m/z* 375 ions in this spectrum confirm that they are product ions of the larger *m/z* 564 compound.

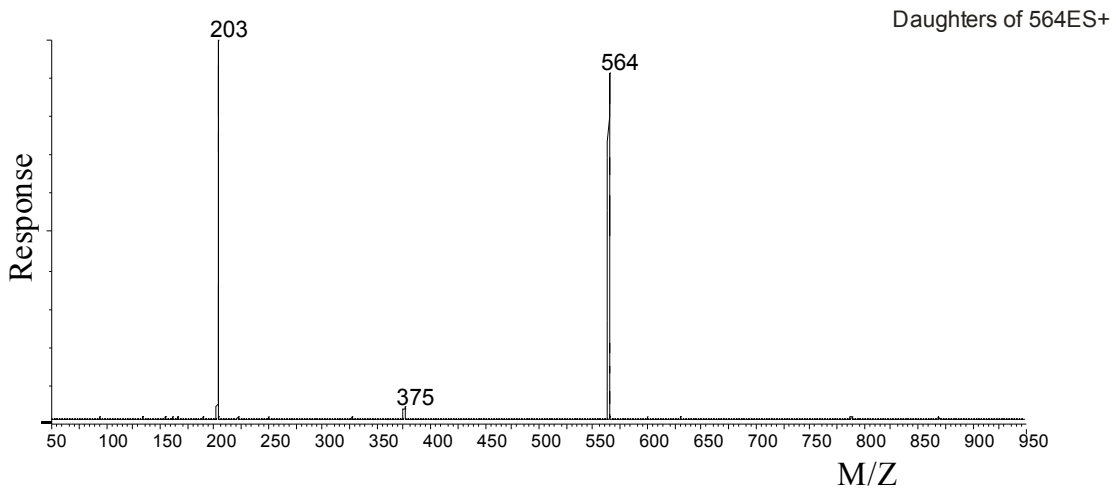


Figure 13: Positive–ion Electrospray ionization spectrum derived from *m/z* 564 ion.

A parent ion scan for the ion *m/z* 203 confirmed the presence of a range of compounds that could produce the ion *m/z* 203 (Figure 14).

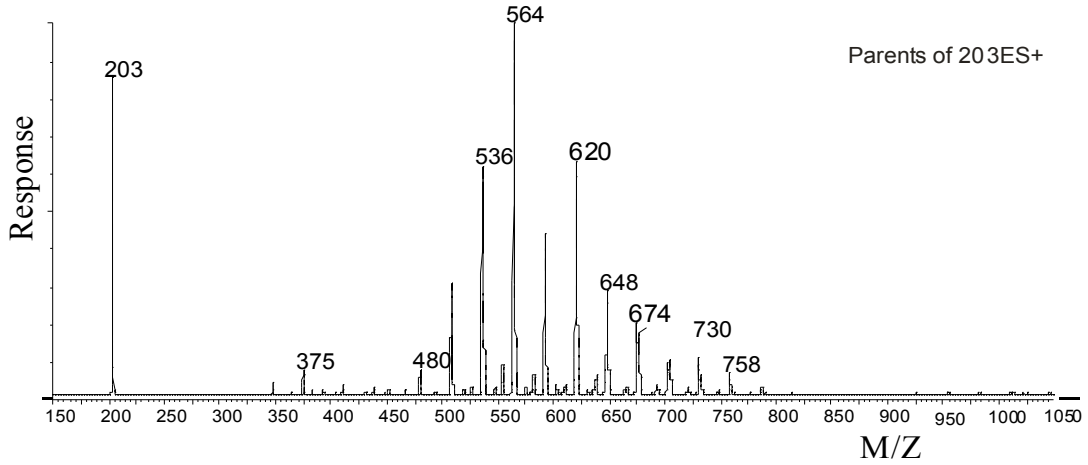


Figure 14: Electrospray ionization mass spectrum of DCM extracted adult male coypu ASG (6009) in positive-ion mode.

These results suggest that the remaining peaks in the DCM extracted ASG GC are farnesyl derivatives (Table 2).

When exposed to long wavelength ultraviolet light (254 nm), the ASG secretory tissue fluoresces brightly (Figure 15).

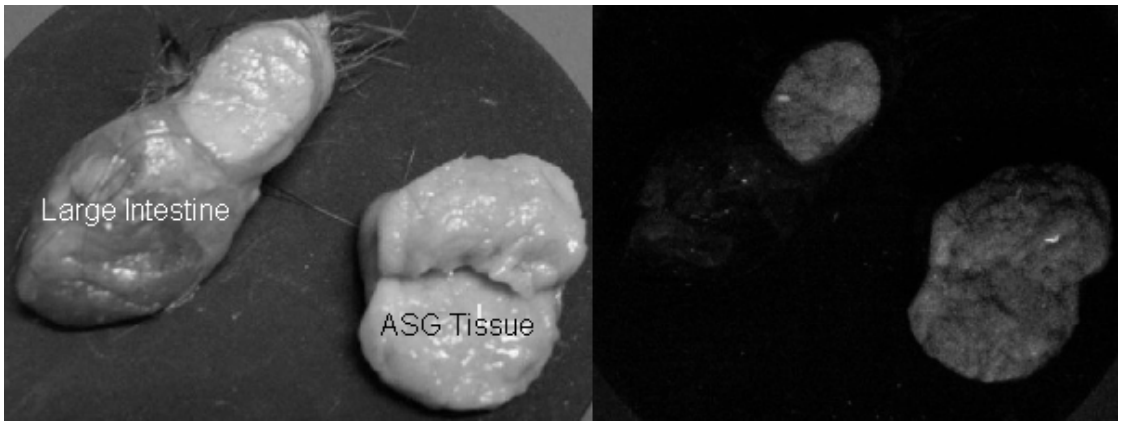


Figure 15: Adult male coypu ASG in visible and UV light. LEFT: Anal gland dissected away from rectum and opened along the midline, visible light. RIGHT: Same as in LEFT, under long wavelength UV light. Photographs by the author.

DISCUSSION

The results of GC-MS analysis carried out in Dr. Meinwald's lab revealed sharp sex differences in compounds found in pentane-extracted ASG. Female ASGs had little or no compounds present. The GC of juvenile female 6013 (Figure 6) is essentially the GC of an inactive gland. The GC of adult female 6013 (Figure 7) reveals only a slightly more productive gland. 6013's GC had one peak, peak A, that was also found in the GCs of both males tested (Figures 8 and 9). Across all four ASGs tested, peak A seems to increase in abundance with increasing overall size in an analog fashion. The fact that both females had none to very few nonpolar compounds is evidence that helps rule out the alternative hypothesis of a grooming function for the ASG.

Despite the paucity of compounds in female ASG, GCs of male pentane-extracted ASG confirmed the presence of 15 compounds found in both juvenile and adult male ASG (Figures 8 and 9). Of these compounds, two (peak A and B), have boiling points low enough for them to evaporate at realistic temperatures encountered by a deposited ASG scent mark. As vapor

phase molecules, they could be perceived by the main olfactory epithelium (MOE) at a distance from the secretion itself. This would be especially interesting for two reasons. First, peak A increases in abundance with size. Second, peak B seems to be adult male-specific. Concentration of peak A could tell the receiver the size of the sender in an analog coding scheme. Presence of peak B could tell the receiver that the sender is an adult male in a digital coding scheme.

The other 13 compounds found in male pentane-extracted ASG have high enough boiling points that it is unlikely they ever dissipate in sufficient quantity to act as volatile odorants. This rules them out as possible odorants capable of being perceived by the MOE. If they are perceived at all, it is more likely the nonvolatile compounds are perceived by the vomeronasal organ (VNO), a “second nose” specialized for the detection of heavier molecules (Takami 2002). It is interesting that these nonvolatile compounds are precisely where most of the variability between the pentane-extracted male ASG GCs is located. The differences in the nonvolatile part of the male ASG GCs are consistent with differences seen in other scent gland secretions that code for individuality using an analog scheme (Zhang et al 2003, Zhang et al 2002).

Specifics of the kinds of compounds that actually form the ASG scent mark were provided by data from the Attygalle lab. Dr. Attygalle’s GC-MS analysis of a DCM-extracted adult male ASG independently confirmed the presence of 15 major peaks in male coypu ASG. It should be noted that this result was obtained using a different nonpolar solvent (DCM vs. pentane) and a faster GC temperature program. Further MS analysis identified 6 of the 15 peaks as common fatty acids of known structure (Figure 16). Far more

surprising is that both labs confirmed the presence of the sesquiterpene farnesene (Figures 10, 13, and 14). Dr. Attygalle's tandem MS/MS data shows the presence of larger compounds that fragment into farnesene (molecular weight of 203) when ionized (Figures 13 and 14). That the ASG secretory tissue fluoresces in ultraviolet light (Figure 15) suggests that these larger structures may contain conjugated double bonds or rings.

Farnesene (Figure 16) is best known as a compound emitted by wounded plants following herbivore attack (Scutareanu et al 2003, Arimura et al 2004, Vuorinen et al 2004). (E)-beta-farnesene is also well known as an alarm pheromone used by many species of aphids (Xiangyu et al 2002). Alpha farnesene and (E)-beta-farnesene are important trail following pheromones of ants and termites (Attygalle and Morgan 1985, Quintana et al 2003). (E)-beta-farnesene has even been found as part of the defensive secretions of marine gorgonians (Gavagnin et al 2003). Perhaps less well known is the role farnesene isomers play in vertebrate olfactory communication. Alligator species make (E)-beta-farnesene in their paracloacal gland (Schulz et al 2003). Springbok make farnesene in their dorsal scent gland (Burger et al 1978). Then there are mice.

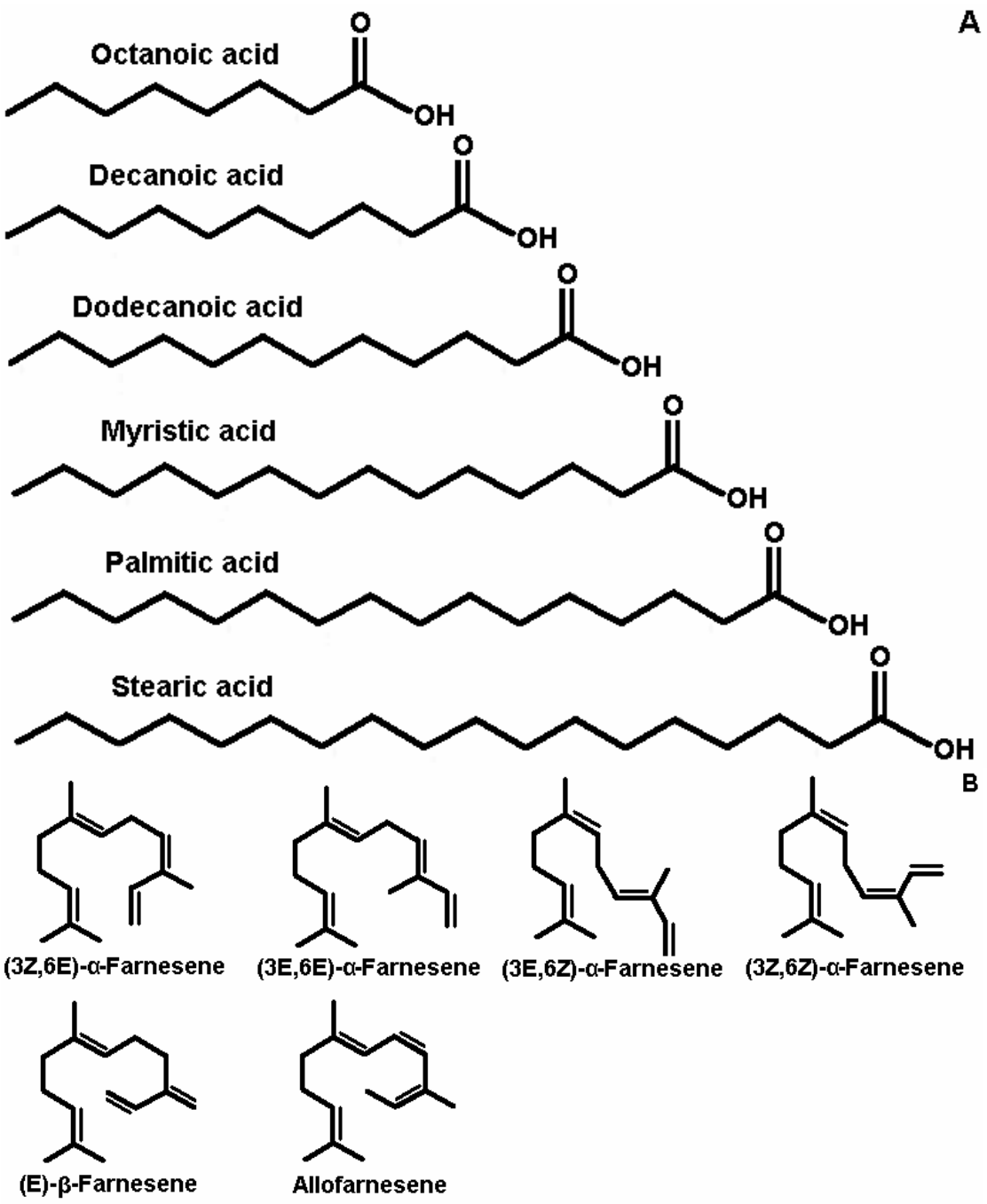


Figure 16: Characterized compounds found in coypu ASG. A: Fatty acids found in coypu ASG. B: Possible isomers of farnesene. (E)-β-farnesene is the isomer identified in coypu ASG.

Mice use urine as a solvent carrier for their chemical signals. The preputial glands of male mice make (E)-beta-farnesene and alpha farnesene

that is added to urine as it leaves the bladder. When compared to subordinate male mice, larger male mice make more farnesenes. Synthesis of the two farnesenes is under gonadal androgen control; castrated mice do not make farnesenes. Both farnesene isomers have measurable effects on the behavior of both sexes. Male mice prefer not to mark on a surface containing natural or synthetic farnesene of either isomer. Female mice are attracted to both isomers of farnesene. Female mice come to estrus sooner after smelling farnesene (Jemiolo et al 1991).

There are intriguing parallels between the mouse farnesene story and production of farnesenes in coypu. First, more farnesene (the (E)-beta-isomer in coypu, peak D, Figures 8 and 9) was found in the larger male. Female coypu do not make farnesenes. Male coypu make farnesene using a sexually dimorphic gland that increases in size with increased male fighting behavior. This suggests that coypu farnesene synthesis is also under gonadal androgen control.

Gonadal androgen control of farnesene synthesis is emerging as a common theme in vertebrate farnesene biosynthesis. While mice and coypu males make farnesene, only immature or female alligators produce (E)-beta-farnesene. This suggests that androgen control can work both ways in an evolutionary sense. As the first volatile precursor on the steroid biosynthetic pathway (Figure 17), farnesene would make a unique volatile indicator of an individual's steroid richness. If an individual has enough steroid (testosterone in males, for example), the steroid itself could induce the synthesis of farnesene. Farnesene would build up at a level indicative of steroid concentration, functioning as an honest signal of an individual's steroid richness.

It should also be noted that apparent gonadal androgen control of farnesene synthesis is evidence that supports a *de novo* synthesis hypothesis for the origin of farnesene. It also helps rule out two other hypotheses for the source of farnesene in coypu ASG (and mouse preputial glands). Farnesene could be accumulated from plants in the diet. It is highly unlikely that lab mice, eating a pelleted lab diet, could accumulate farnesene from their diet. Even if farnesene was accumulated rather than synthesized, the obvious sex differences in its distribution in mice and coypu and castration effects on farnesene synthesis in mice would still point to a gonadal androgen controlled accumulation. Farnesene could also be produced by symbiotic microorganisms living in farnesene-secreting glands. While this is certainly possible, it is highly unlikely due to the fact that sesquiterpenes are known to have potent antimicrobial properties (Gudzic et al 2002). It is also unlikely that laboratory mice would be able to maintain cultures of such microorganisms for the length of their domestication.

Whether or not farnesene has measurable effects on coypu behavior will soon be known. Experiments designed to test hypotheses concerning the behavioral role or roles of farnesene in coypu chemical signaling are currently

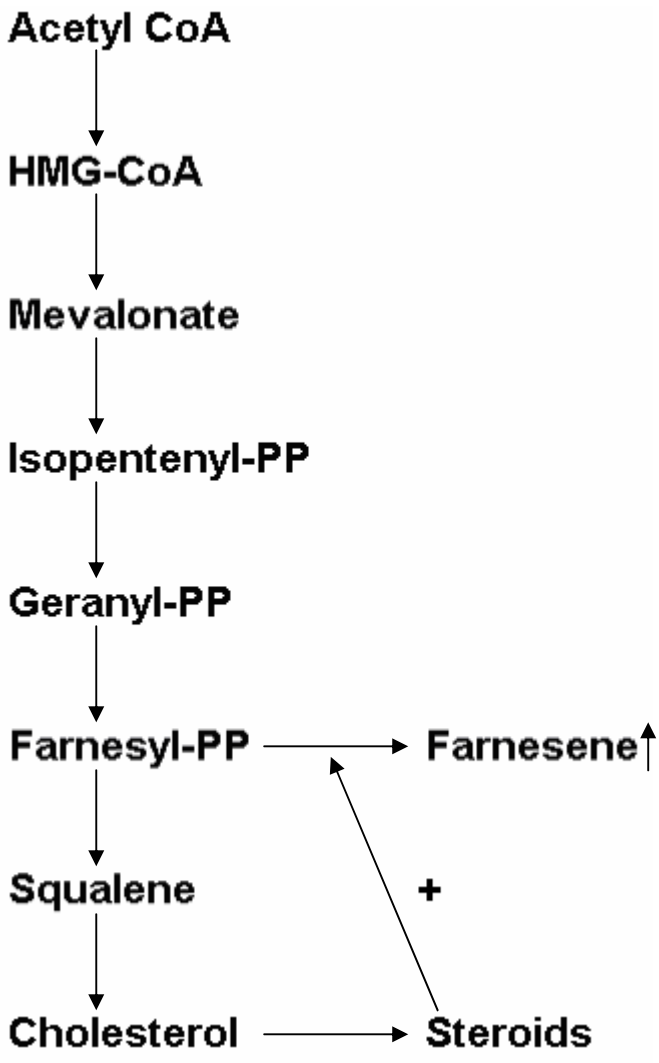


Figure 17: Hypothetical model of steroid control of mammalian farnesene synthesis.

being carried out by USDA scientists in cooperation with the author. Captive coypu of known sex and mass will be exposed to natural and synthetic ASG secretion in large Y-mazes. Time spent sniffing experimental secretion versus control solvent or water will be measured. I predict that male coypu will spend less time sniffing natural ASG secretion from dominant males and synthetic ASG secretion that contains farnesenes. I also predict that females will spend more time sniffing these same experimental samples.

Results of the present study support the hypothesis that coypu ASG secretion is a chemical signal used in aggressive interactions between male coypu. From a signal design perspective, male coypu ASG seems to be an olfactory graffiti of sorts. The signaling coypu leaves a scent mark and resumes the patrolling of his territory. Any receiving coypu wandering nearby can use its MOE to tell the sender's size and sex using a mixed analog (peak A) and digital (peak B) coding scheme. If interested, the receiving coypu can get closer and use its VNO to tell who left the scent mark using nonvolatile compounds that specify individuality using an analog scheme.

Analytical data from the Attygalle group's analysis confirms the presence of fatty acids of known structure. Data from the Attygalle and Meinwald labs independently confirm the presence of the sesquiterpene farnesene. Farnesene was found only in the large male coypu ASG tested. This is identical to the distribution of farnesene in mice, suggesting a similar role for farnesene in both rodent species. The discovery of farnesene in the ASG of male coypu helps support the hypothesis that farnesene is a volatile indicator of steroid richness.

Experiments to determine the role of farnesenes in coypu chemical signaling are underway. These experiments will help answer basic questions about the evolution of olfactory signaling in rodents. It is also possible that knowledge of ASG compounds and their role in coypu chemical signaling could be used to develop coypu-specific attractants that would be useful in controlling coypu numbers while harming fewer nontarget species.

CHAPTER 2

A NEW TECHNIQUE FOR COMPUTER-AIDED PHOTOGRAPHIC IDENITIFICATION OF INDIVIDUAL COYPU (*Myocastor coypus*)

INTRODUCTION

Identifying individual members of a population is a necessary first step of many methods in animal field biology. Long term studies of animal social behavior, like those of chimpanzees, gorillas, and other social animals, require that individuals be reliably identified (Charif et al 2005, Payne et al 2003, Whiten et al 1999, Stanford et al 1994, Sapolsky 1982, Fossey 1982, Fossey 1974). Studies of reproductive cycles, migration, dispersal, predation, and other population parameters require that individual animals be identified and tracked (Porcher 2005, Hake et al 2003, Berthold et al 1997, Henry et al 2005). Data from individuals can then be used in models to better estimate the behavior of populations. This kind of data is crucial when forming hypotheses about the evolution of populations and essential when making decisions that affect such broad themes as conservation and pest management (Bowler and Benton 2005). The importance of data from individual animals in a population has led to the development of many practical methods of individual identification. Although there are many different ways to determine individuality, the limitations of human sensory abilities and the variablitity of field conditions have forced most methods to involve some kind of capture and marking followed by subsequent observation or recapture.

Mark-recapture methods take many forms. Fish are tagged in the tail (Russell and McDougall 2005), ungulates are tagged through the ear (Swenson et al 1999), birds are banded around the leg (Regehr and Rodway, 2003), and carnivores are often radio collared around the neck (Tuyttens et al 2002). Almost all mark-recapture methods involve a period of physical restraint (sometimes with anesthesia) followed by the application of a pigment dye, or a tag inserted through or otherwise attached to skin, or a transmitter device latched to the neck or other unchewable body part. The advantages of physically marking an individual are obvious: a researcher need only look up the tag, band, mark, or radio frequency in a preconstructed index to see what individual is being observed or has been caught. There is little ambiguity about the identity of a bird with a ring band, or a moose with a yellow ear tag. The disadvantages of physically marking an animal are not only obvious, they may be painful as well. Many animals have measurable physiological stress reactions to being restrained. Marking and recapture are assumed to have a similar negative impact on individual animals (Close et al 2003, Perret and Joly 2002, Makinen et al 2000).

This says nothing of animals that are too large, too physically dangerous, or too wary of being physically caught. These animals have forced biologists who study them to come up with creative ways of identifying individuals without physically restraining them. Animals that produce individually unique vocalizations can be monitored with audio recording equipment. In this case, a ‘marked’ individual is one that has had a distinct vocalization recorded and separated from the same kind of vocalizations of other individuals. This individual is recaptured when the same call is recorded

and matched to that individual. Cetaceans of several species (Barlow and Taylor 2005, Boisseau 2005, Watkins et al 2004, Burtenshaw et al 2004) are tracked across oceans by recording the courtship songs of male individuals.

Whales are also good examples of a second kind of 'capture' that does not involve physical contact with individual animals. Humpback whales have white pigmented areas on the ventral surface of their flukes. These areas form patterns that are unique to individual humpbacks. Whale biologists note their own current location and take photographs of these tail patterns when the humpbacks breach or slap the water with their tails. Entire catalogues of tail patterns are built up from many whale sightings and correlated with location. Individual humpbacks can then be matched to a sighting using the pattern of white pigmented areas on the tail and followed across long distances (Smith et al 1999, Friday et al 2000).

Since the advent of inexpensive digital cameras and photograph scanners, the tail pattern catalogues have been converted to digital format. This has allowed some whale biologists to use computer programs to rapidly sort through massive catalogs quickly (Khetarnavaz et al 200315). Using individually unique patterns of spots, cheetahs have also been the subject of computer-aided photographic identification (Kelly 2001). This report details the development of a computer-aided photographic identification method for an invasive hystricomorph rodent, the coypu (*Myocastor coypus*).

Coypu are large, herbivorous, beaver-like rodents originally native to the wetlands of Southern South America. In the 1930s, coypu were introduced to many places around the world for fur farming. Coypu eventually escaped or were released from fur farms. Feral populations were kept under control by commercial trappers until the fur market collapsed in the 1980s

(Carter and Leonard 2002). Since then, coypu have spread beyond their introduced range. This spread has been accompanied by the large scale destruction of fragile coastal wetlands by foraging coypu. Efforts to control or even eradicate coypu have been hampered by a lack of reliable population estimates. Coypu population estimates are hard to estimate because coypu are incredibly hard to mark and recapture. Coypu often will not enter live traps. When they are trapped once, they often will never enter the same kind of trap again (Lowery 1974). In order to develop a way to mark-recapture coypu without physically restraining them, I began to look for naturally occurring, visually observable differences between individual coypu.

With the exception of the brilliantly white fur and facial whiskers around the mouth, coypu are a drab brown (Figure 18). Several of the large, stiff, white, facial whiskers are used for navigation in the dark and when underwater (Lowery 1974). I hypothesized that the insertion pattern of those whiskers into the surrounding brown fur was unique to individual coypu. To test this hypothesis, I wrote a computer program that allows a user to mark the point of insertion of each white facial whisker on a digital photograph of a coypu face. The program then computes an individual-specific 'ID' number using the relative distances between each whisker. The program was tested on photographs and captured video frames of a group of captive coypu.

Figure 18: The face of an adult male coypu. Note the white fur around the mouth and the large easily observed white whiskers projecting from the side of the face. Photograph by the author.

METHODS

Eighteen individual coypu were held in groups of three in concrete pens at the field station of the LDWF Fur and Refuge Division (New Iberia, LA) for 12 hours before being killed as part of ongoing coypu control operations in Louisiana. Still photographs of coypu were taken with a Minolta Dimage Z1 3.2 megapixel digital camera with 10X optical zoom. Video of coypu was taken with a Sony Hi8 Handycam video camera (model # CCD-TRV318) with 20X optical zoom. Analog video was converted to digital video using a Dazzle DVC80 video capture device. Individual frames of video were taken using the frame grabber utility in Microsoft MovieMaker v2.0.

Pens with coypu were numbered 1 through 6. I went to each pen 5 times every 30 minutes and took pictures of a random coypu. Every fifth time, I took 30 seconds of video of a random individual and captured the best frame from that 30 seconds. Of the 30 pictures, 23 were in focus and had clearly

visible individual white whiskers. These 23 pictures were analyzed using Whisker2 and WhiskerView, two programs I wrote for this purpose.

Whisker2 allows a user to mark individual whisker insertion points in a figure window. Once all the whisker insertion points are marked, the program sorts the coordinates on the x axis and finds the coordinates of the whisker insertion points that form the vertices of a polygon that encloses all other whisker insertion points (smallest area polygon). Whisker2 also sequentially finds the distance from the lowest x value whisker insertion point (X_i) to the next highest whisker insertion point and adds all these distances together. Whisker2 divides this sum by the distance from the smallest x value whisker insertion to the largest x value whisker insertion point (X_n) to get an individual specific ID number:

$$ID = \frac{(\overline{X_i X_{i+1}} + \overline{X_{i+1} X_{i+2}} + \cdots + \overline{X_{n-1} X_n})}{(\overline{X_i X_n})}.$$

The program then plots the smallest area polygon as a red shape and plots the whisker coordinates in an output window with a white line connecting the whisker insertion point with the lowest x value to the whisker insertion point with the next highest x value over the red shape (Figure 19).

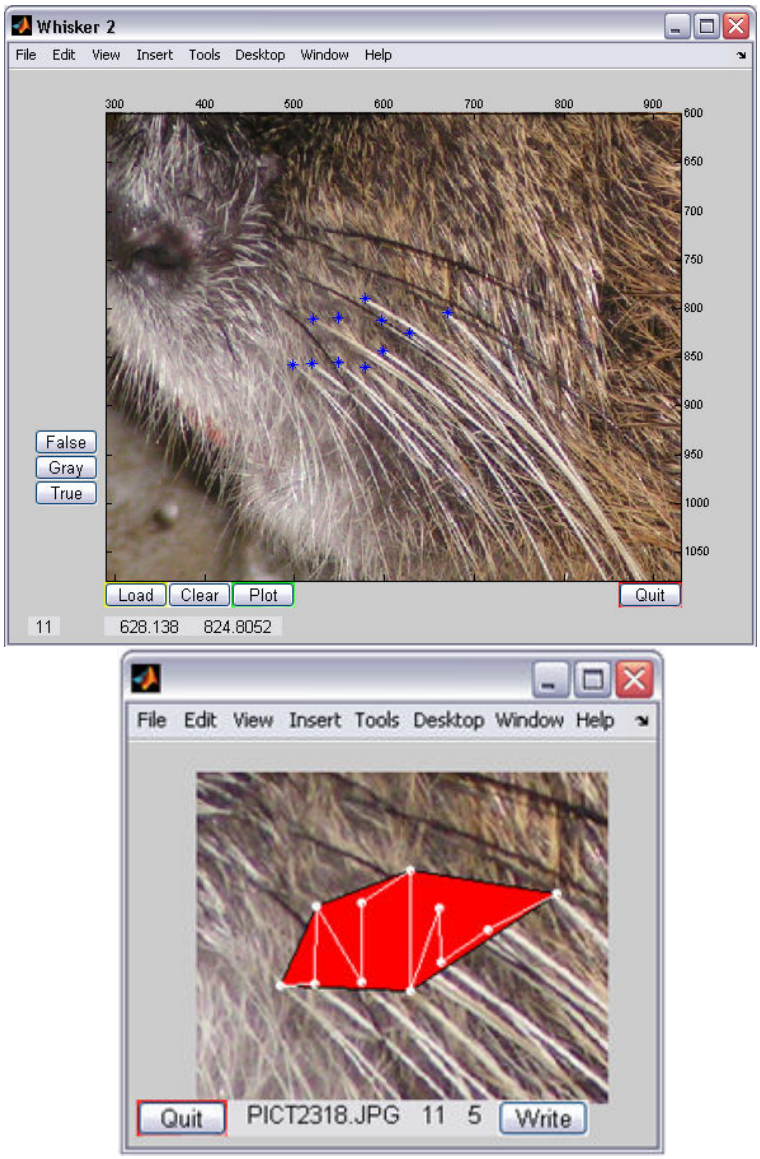


Figure 19: Output of Whisker2. TOP: Whisker insertion points are marked in blue. BOTTOM: Smallest area polygon is in red. Sorted whisker insertion points are marked with white dots and connected with a white line. The text field between the 'Quit' and 'Write' buttons contains the name of the file being analyzed, the number of whiskers, and the number of smallest area polygon vertices.

Whisker2 was used by three different operators to mark whisker insertion points on all 23 pictures. Operators had been trained for approximately an hour before analyzing photographs. Output windows were

exported to JPEG images for later sorting with WhiskerView. This resulted in three whisker insertion plots per photograph.

WhiskerView allows a user to load a target image previously created in Whisker2[1]. The target image is compared to all other plots of whisker insertion points by sequencing through exported output image JPEGs until a match was found.

RESULTS

All three Whisker2 operators obtained identical numbers of whiskers and number of smallest area polygon vertices for each photograph or captured video frame analyzed. Numbers values of ID never varied more than 0.7 between operators. Of 18 coypu photographed and identified, none had identical combinations of the three parameters output by Whisker2. The operators also assigned the same photographs to the same individuals using WhiskerView. Of the 23 pictures taken, 15 individuals were represented by one photograph. Two individuals were represented in 2 pictures. One individual was represented by 4 photographs. Figure 20 (following pages) shows representative photographs and matched output windows for 3 individuals. Table 3 shows all data for all photographs and captured video frames analyzed.

[1] Instructions and documentation for both programs can be found in Appendix II

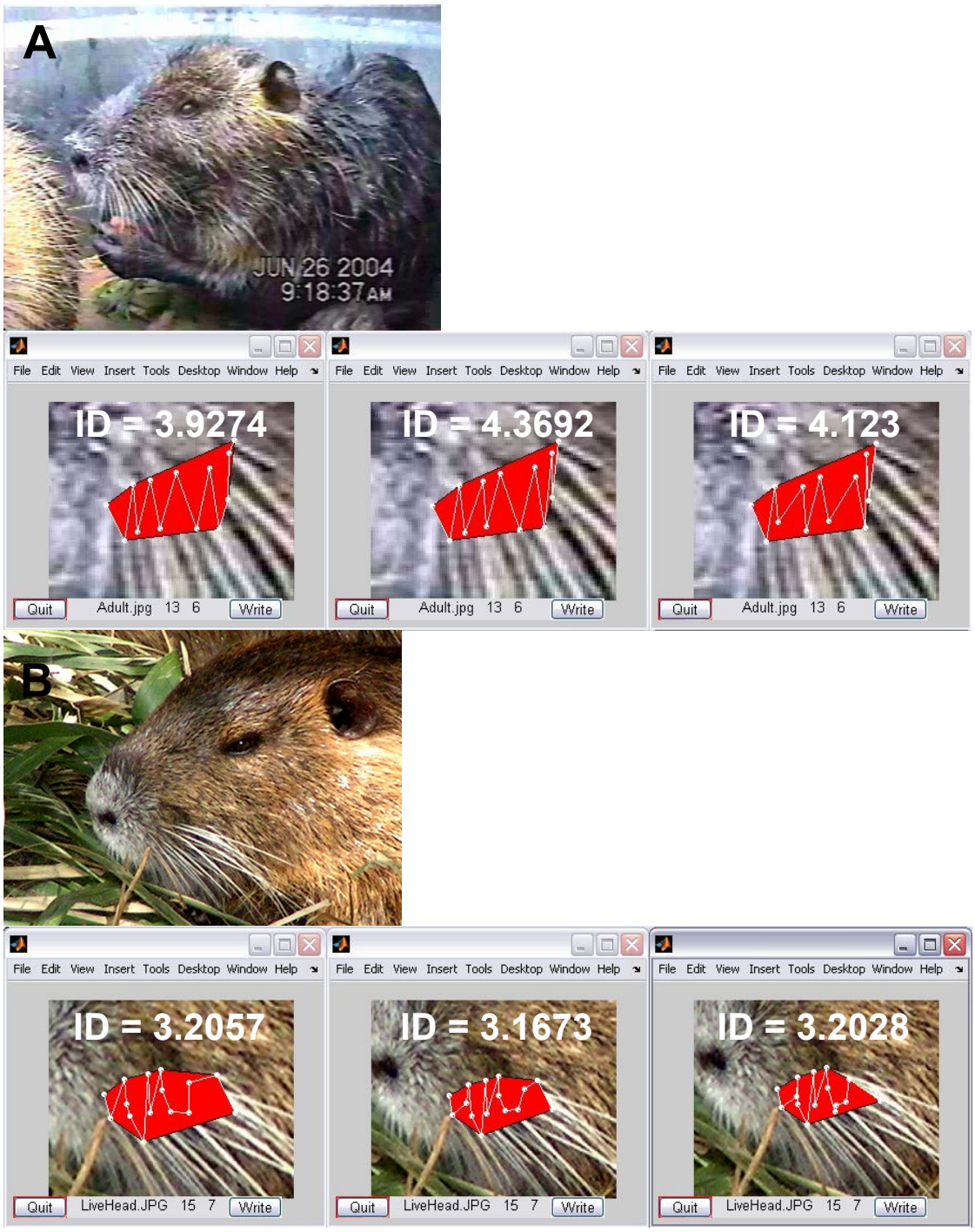


Figure 20: (Pages 44-45) Example photographs and matched Whisker2 output windows. ID values have been added over each output window. A: Captured video frame of adult coypu 1 with output windows from three operators. B: Still photograph of adult coypu 2 with output windows from three operators. C: Still photographs of adult coypu 12 with output windows for each photograph from each operator.

Figure 20 (Continued)

Table 3: All data for each individual coypu and matched Whisker2 output data

Photo #	Coypu #	Filename	Photo or Video Frame ?	Whisker #	Smallest Area Polygon Vertices #	ID
1	1	Adult.jpg	Frame	13	6	3.9274
1	1	Adult.jpg	Frame	13	6	4.3692
1	1	Adult.jpg	Frame	13	6	4.123
2	2	juvy1.jpg	Frame	12	9	5.6854
2	2	Juvy1.jpg	Frame	12	9	5.5339
2	2	Juvy1.jpg	Frame	12	9	5.8772
3	3	Juvy B1.jpg	Frame	14	7	3.033
3	3	Juvy B1.jpg	Frame	14	7	3.3134
3	3	Juvy B1.jpg	Frame	14	7	3.0369
4	4	JuvyA2.jpg	Frame	10	5	2.1081
4	4	JuvyA2.jpg	Frame	10	5	1.9947
4	4	JuvyA2.jpg	Frame	10	5	1.9011
5	5	LiveHead.JPG	Photo	15	7	3.2057
5	5	LiveHead.JPG	Photo	15	7	3.1673
5	5	LiveHead.JPG	Photo	15	7	3.2028
6	6	PICT2250.JPG	Photo	16	9	4.2986
6	6	PICT2250.JPG	Photo	16	9	4.3117
6	6	PICT2250.JPG	Photo	16	9	4.1599
7	7	PICT2256.JPG	Photo	10	5	2.9337
7	7	PICT2256.JPG	Photo	10	5	2.6674
7	7	PICT2256.JPG	Photo	10	5	2.742
8	8	PICT2257.JPG	Photo	18	7	4.7575
8	8	PICT2257.JPG	Photo	18	7	5.0731
8	8	PICT2257.JPG	Photo	18	7	4.9134
9	9	PICT2258.JPG	Photo	16	7	2.5199
9	9	PICT2258.JPG	Photo	16	7	2.6465
9	9	PICT2258.JPG	Photo	16	7	2.5342
10	9	PICT2295.JPG	Photo	16	7	2.6242
10	9	PICT2295.JPG	Photo	16	7	2.6865
10	9	PICT2295.JPG	Photo	16	7	2.4999
11	10	PICT2262.JPG	Photo	17	7	4.521
11	10	PICT2262.JPG	Photo	17	7	4.47
11	10	PICT2262.JPG	Photo	17	7	4.197
12	11	PICT2264.JPG	Photo	12	8	4.1203
12	11	PICT2264.JPG	Photo	12	8	4.1927
12	11	PICT2264.JPG	Photo	12	8	4.0213
13	12	PICT2246.JPG	Photo	16	6	3.7648

Table 3 (Continued)

13	12	PICT2246.JPG	Photo	16	6	3.8515
13	12	PICT2246.JPG	Photo	16	6	4.1086
14	12	PICT2267.JPG	Photo	16	6	3.8854
14	12	PICT2267.JPG	Photo	16	6	3.8658
14	12	PICT2267.JPG	Photo	16	6	3.8658
15	13	PICT2269.JPG	Photo	14	8	2.7315
15	13	PICT2269.JPG	Photo	14	8	3.38
15	13	PICT2269.JPG	Photo	14	8	2.8979
16	14	PICT2272.JPG	Photo	15	7	3.8029
16	14	PICT2272.JPG	Photo	15	7	3.0239
16	14	PICT2272.JPG	Photo	15	7	3.4566
17	15	PICT2276.JPG	Photo	15	7	8.0026
17	15	PICT2276.JPG	Photo	15	7	8.2527
17	15	PICT2276.JPG	Photo	15	7	8.2357
18	16	PICT2279.JPG	Photo	13	7	4.6816
18	16	PICT2279.JPG	Photo	13	7	4.5085
18	16	PICT2279.JPG	Photo	13	7	3.7959
19	16	PICT2289.JPG	Photo	13	7	5.0352
19	16	PICT2289.JPG	Photo	13	7	4.5768
19	16	PICT2289.JPG	Photo	13	7	4.6807
20	16	PICT2299.JPG	Photo	13	7	4.7353
20	16	PICT2299.JPG	Photo	13	7	4.9644
20	16	PICT2299.JPG	Photo	13	7	5.4089
21	16	PICT2303.JPG	Photo	13	7	4.9422
21	16	PICT2303.JPG	Photo	13	7	4.9745
21	16	PICT2303.JPG	Photo	13	7	5.0545
22	17	PICT2302.JPG	Photo	13	6	3.4778
22	17	PICT2302.JPG	Photo	13	6	2.8617
22	17	PICT2302.JPG	Photo	13	6	3.2731
23	18	PICT2308.JPG	Photo	12	7	2.8668
23	18	PICT2308.JPG	Photo	12	7	2.8678
23	18	PICT2308.JPG	Photo	12	7	2.7951

DISCUSSION

Data from this study supports the hypothesis that whisker insertion pattern uniquely identifies individual coypu. Three independent operators were able to reliably determine the same number of whiskers and the same number of smallest area polygon vertices. ID number varied somewhat between photographs matched to the same individual (Table 3). Despite the

decrease in resolution, captured video frames were as easily identified as still photographs. As shown in the example photographs and output windows (Figure 20), the differences in whisker insertion pattern between individuals are obvious when compared side by side.

This is the real strength of the whisker insertion pattern approach: matching marked individuals to previously made output windows is intuitive and extremely simple. There is no complicated discriminant function analysis or principle components analysis to separate individuals. A sample whisker insertion pattern either has the right number of whiskers, the right number of smallest area polygon vertices, a similar ID value, and looks the same or it does not. The price for this simplicity must be paid for in consistency and image quality. The matching process is very sensitive to the presence or absence of one whisker. This is somewhat offset by the computed ID value, which is actually a measure of relative whisker spread and is independent of whisker number. To enhance consistency, Whisker2 has features that help the operator reliably identify whiskers. An operator can zoom in on whiskers and get a false color version of the image that enhances white elements. Selecting what images to analyze is also an important factor in consistency of identification. Images that are out of focus, blurred, or otherwise deficient should not be used. Note that this does not preclude the use of captured video frames. There are often moments in a video clip when an individual coypu is perfectly still. These moments, when captured to digital format, are just as reliable for whisker insertion pattern matching as a mid-resolution still photograph. It is often easier to scan through video and find a better image of an individual's whiskers than it is to take multiple poor quality still pictures of a fleeing coypu.

One way to avoid actually chasing down a coypu is to combine whisker insertion pattern matching with the recently developed technique of camera 'trapping.' Camera trapping uses cameras outfitted with laser or infrared triggers to automatically take photographs when an animal passes through the trigger beam (Wacher and Attum 2005, Hegglin et al 2004, Wegge et al 2004). This would involve stationing a camera near a tunnel entrance or floating platform that coypu frequent. Photographs would be taken automatically when coypu crossed the infrared or laser beam and analyzed offline. Biologists would then be able to identify individual coypu over a long period of time in a completely noninvasive way. Armed with such a tool, coypu could finally be counted without the massive effort currently spent on mark-recapture studies. This would lead directly to better estimates of population size, a parameter crucial to measuring the success of control operations. Plans to develop camera traps that use Whisker2 as an identification method are currently being developed for the Maryland Nutria Project's eradication effort. The project is killing many coypu with traps and poison. However, it is difficult and time consuming to monitor trapped areas for any remaining coypu. By using camera traps and Whisker2, Maryland Nutria Project employees will only need to periodically check the cameras rather than constantly monitoring dangerous traps and poisons. This will allow more manpower to be directed at trapping out areas that are still infested with coypu.

Whisker insertion pattern as a photographic identification tool may also have application to other animal species. Other rodents, seals, and sea otters have observable whiskers that have high contrast with background fur or skin. It is highly likely that these species also have individually unique whisker insertion patterns.

APPENDIX I: Whisker2 and WhiskerView Documentation

A: How to Use Whisker2 and WhiskerView

1. Whisker2 and WhiskerView will only work in MATLAB Release 14. The function 'moveplot' (provided with the Whisker2 and WhiskerView M files) is also needed for Whisker2 to work.

2. Put all pictures to be analyzed in MATLAB's 'work' folder. Only JPG images can be used. There are many free conversion utilities on the web that will convert TIF, GIF, and other picture formats to JPG.

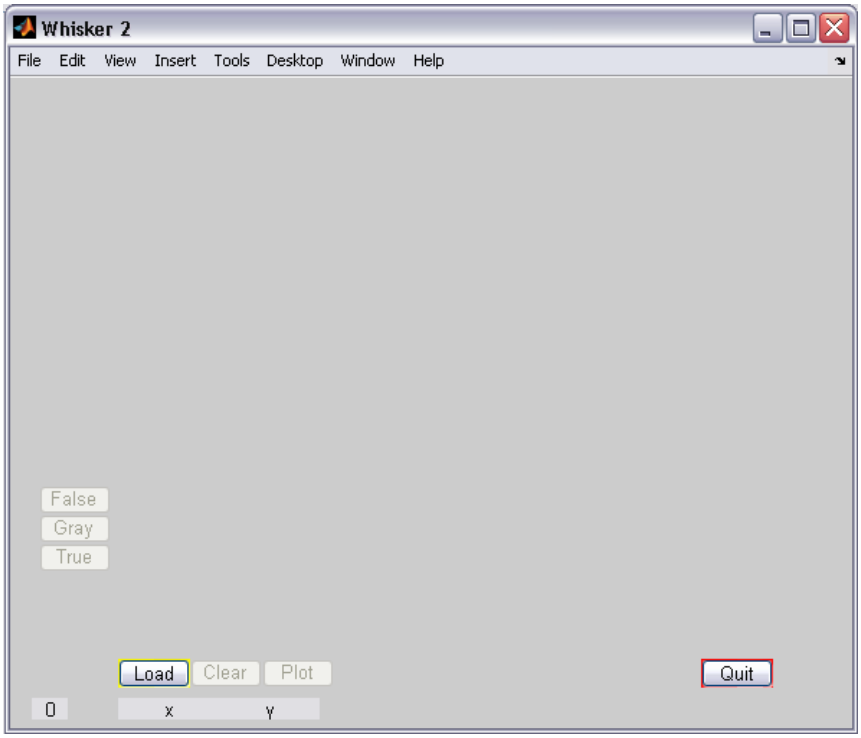
3. Click on 

to start.

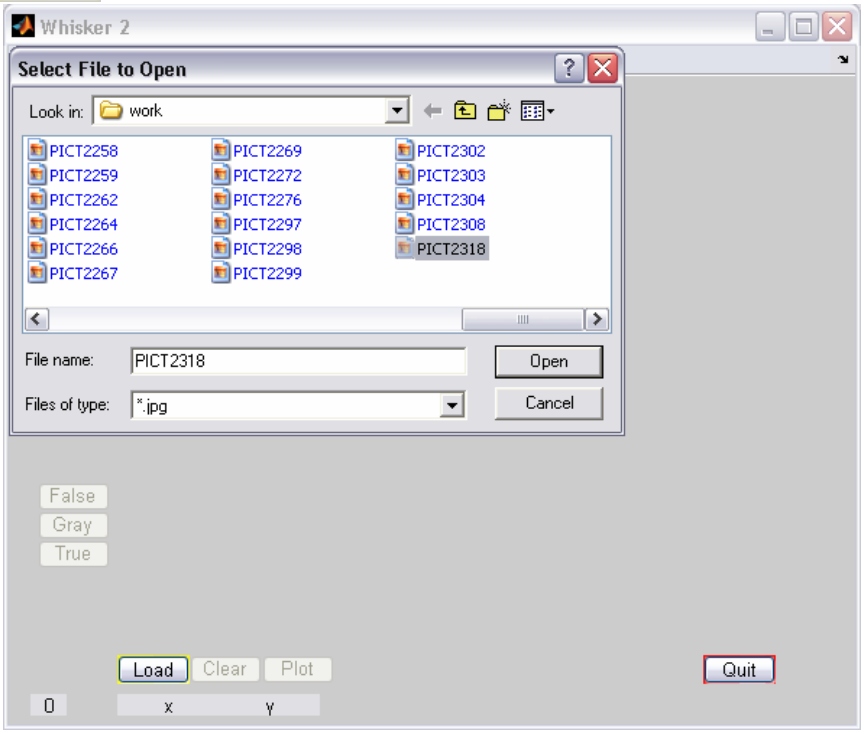
4. Load  whisker2.m into MATLAB's M file editor.

5. To start Whisker2, press  , or F5.

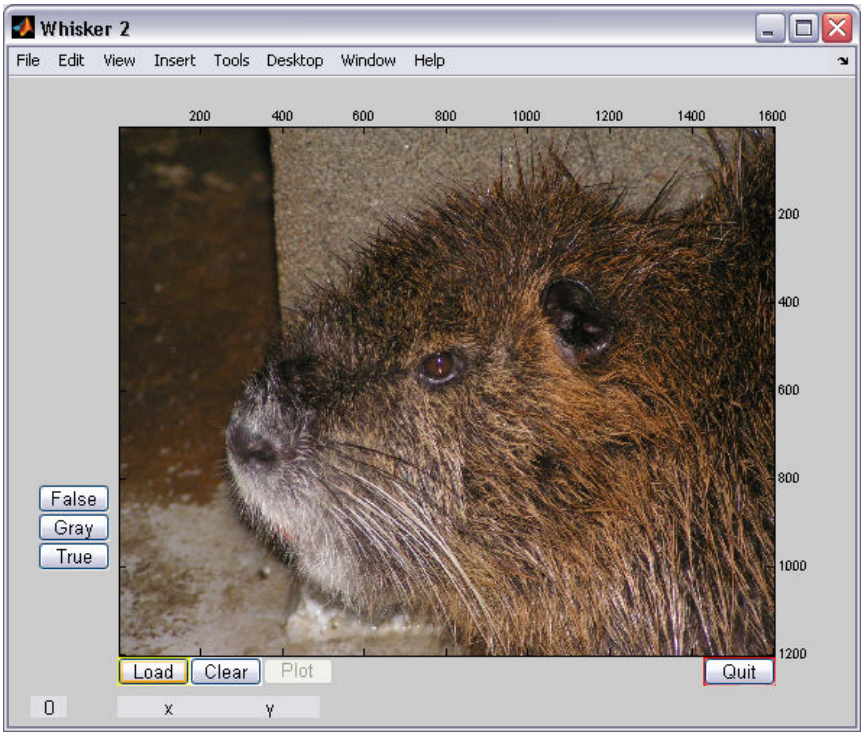
6. The Whisker2 main window looks like this:



7. Press Load . The screen should now look like this:



8. After the file is finished loading, the screen should then look like this:



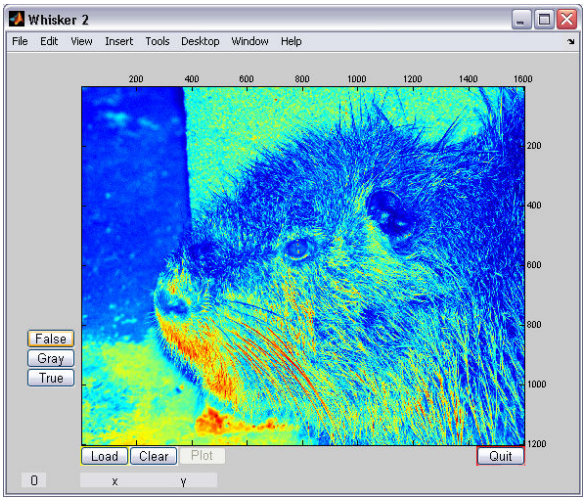
10. There are several options available to help identify whiskers.

- First the user can zoom in or out. Go to 'Tools' on the menubar and click on 'Zoom in.' Once the Zoom in tool is selected, click where you want to zoom in.

- There are three buttons on the left hand side of the main window:

Examples of the results of each button are shown below:

'False': Areas that are white in a true color image become bright red.

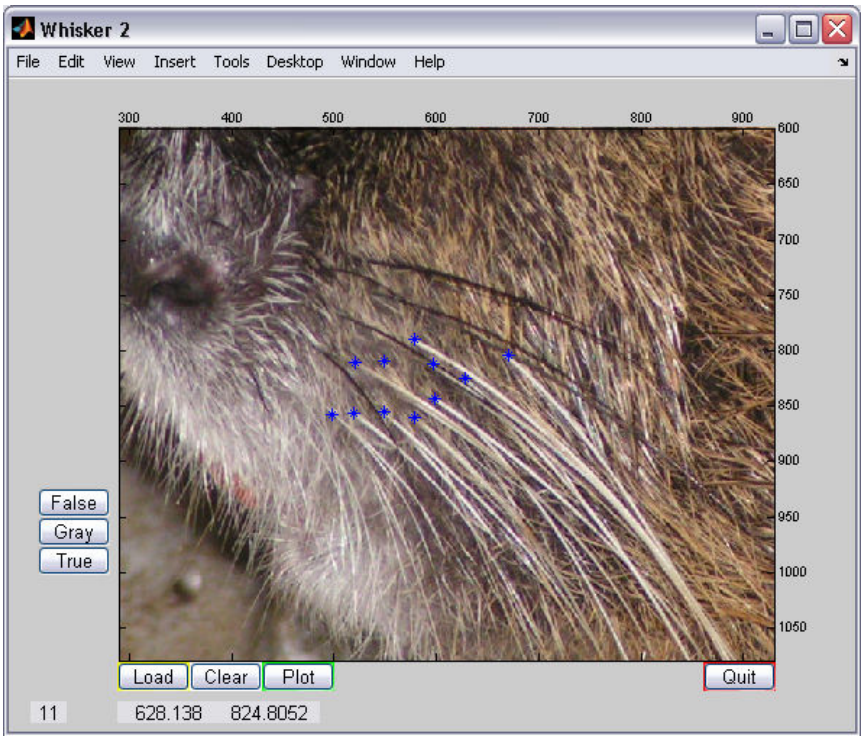


'Gray': Removes all color.

'True' returns the picture to its original color.

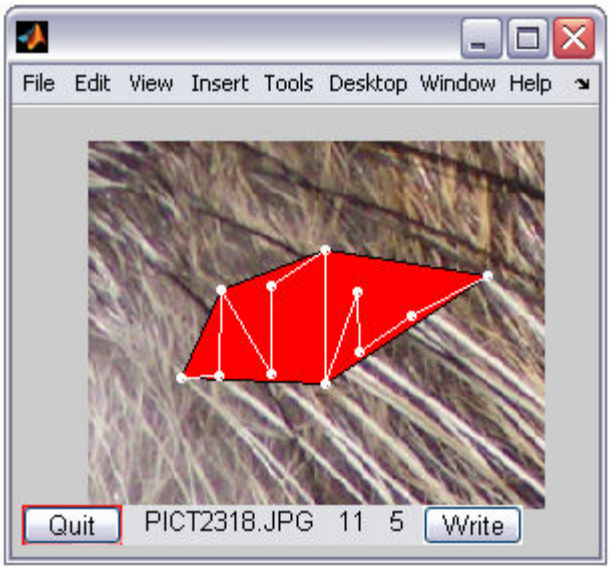
11. Pressing the Clear button clears the current picture and resets all buttons and data.

12. Use the mouse to click on whiskers. A blue asterisk is left on the whisker. The following example is zoomed in. Note that the text fields now display the number of whiskers marked (to the left, bottom) and the coordinates of the marked whiskers (bottom). It is possible to switch between different color styles and still have the same whiskers marked. This is a good way to check a whisker that was just marked. Only mark whiskers that are distinct and show up in multiple pictures of the same animal.



13. When done marking whiskers, press the Plot button. A new window should pop up next to the old one.

14. The new window can also zoom in.



15. This window displays the whisker points as white balls connected by a line. Whiskers are ordered from smallest x coordinate to largest. A line connects them in that order. This makes a pattern that is easier to see than random white balls.

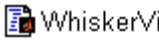
16. It is possible to move the white balls if the user is not satisfied with their location. Click on a ball and drag it with the mouse. The red shape behind the whisker points is the smallest area that contains all the marked whiskers. The text field at the bottom of the window displays the filename of the picture, the number of whiskers, and the number of whiskers that make up the vertices of the smallest-area polygon.

17. Another number is computed from the whisker data but not displayed in the window. Whisker2 takes the coordinates, ordered by their x value, and calculates the distance from one point to the next. It then adds those

distances up. Whisker 2 also calculates the distance from the whisker point with the smallest x coordinate to the whisker point with the largest x coordinate. It takes the sum of all inter-whisker distances and divides it by the distance from the smallest coordinate to the largest to get an individual-specific 'ID' number.

18. If the user clicks on the 'Write' button, the image filename, number of whiskers, number of smallest-area vertices, and ID number are saved to a text file named WhiskerDat.txt. This file is saved in MATLAB's 'work' directory. The entire file can be saved as a MATLAB figure with extension '.fig'. Figures can be exported as JPGs that can be opened with WhiskerView.

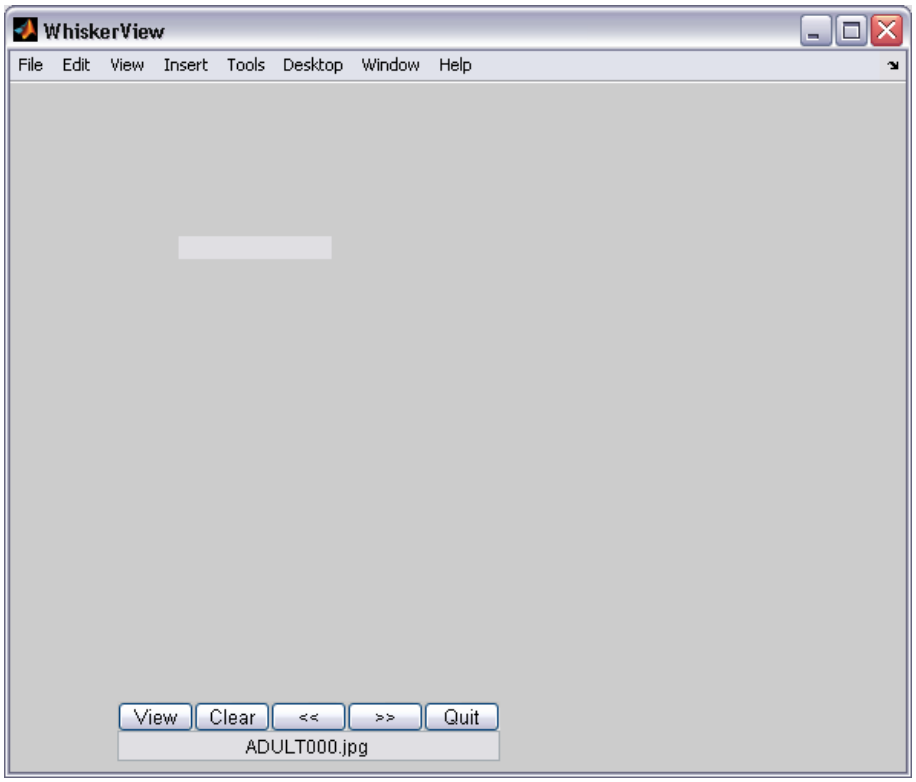
19. Once a large catalogue of whisker prints is built up, it is convenient to have a way to rapidly check one print against others. WhiskerView is designed to do this by allowing the user to load a target picture into one part of a window and scroll through marked images in another part of the same window.

20. Follow steps 1-3 to start MATLAB. Load 

into MATLAB's M file editor.

21. To start WhiskerView, press  , or F5.

22. The WhiskerView window looks like this:



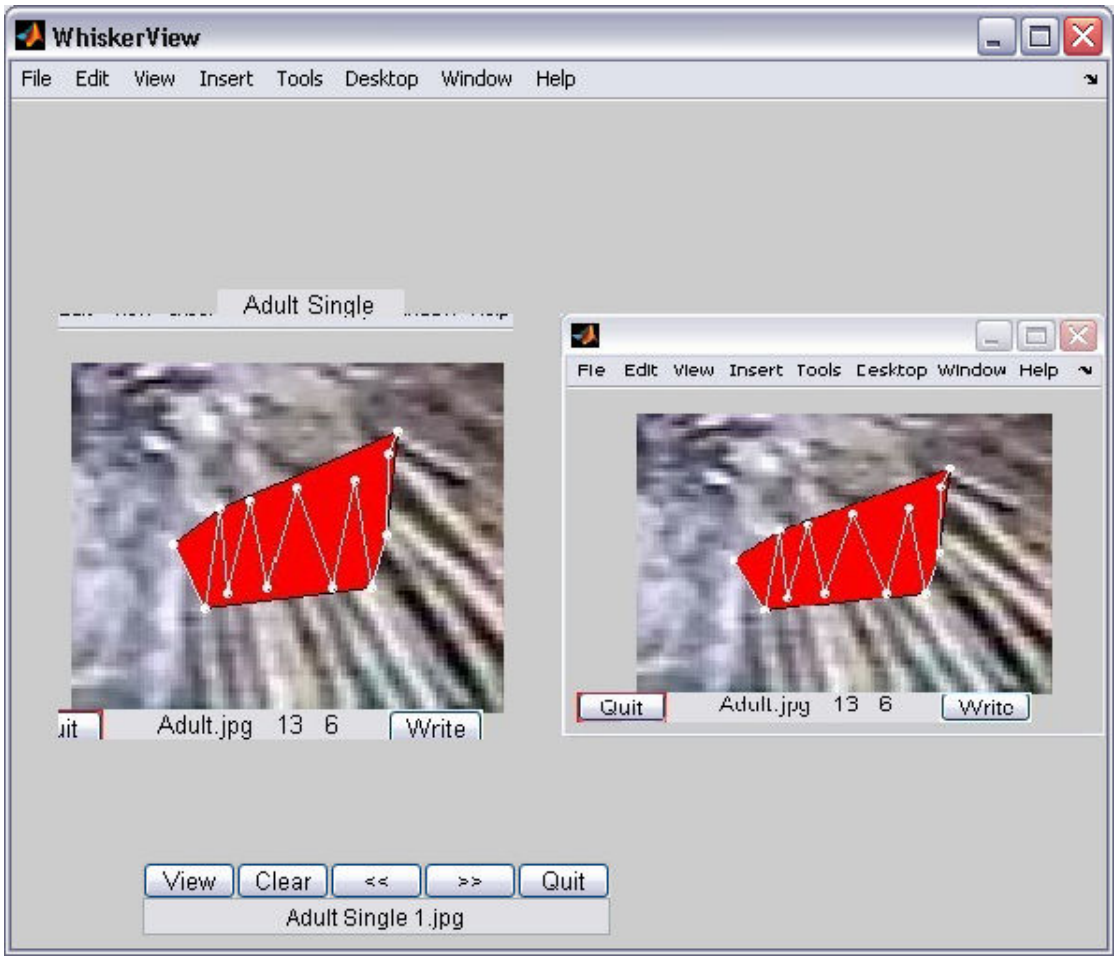
23. To load a target picture, type its filename (with JPG extension) into the target text field (bottom).

22. Click 

. Use the 

buttons to move through pictures on the left side of the window. The window should look like the one below:

23. The window shown above is not a match. Keep cycling through pictures until a picture matches up with the target like the one shown on the next page


WhiskerView
File Edit View Insert Tools Desktop Window Help
Adult Single
Adult.jpg 13 6
Write
File Edit View Insert Tools Desktop Window Help
Quit
Adult.jpg 13 6
Write
View
Clear
<<
>>
Quit
Adult Single 1.jpg

B: HOW WHISKER2 AND WHISKERVIEW WORK

Whisker2 and WhiskerView were written in MATLAB, Release 14. MATLAB is a popular programming language used predominantly by scientists and engineers. MATLAB is a straightforward language that is easily learned, even by programming novices. MATLAB's useful built-in functions allow the beginning programmer to focus on problem solving and not the arcana of getting started commonly found in other programming languages. What follows is as much a brief tutorial in MATLAB programming as it is a dissection of a MATLAB program. Non-programming users of Whisker2 and WhiskerView who are interested in making useful changes to the program can use the following information to make those changes in an intelligent way.

MATLAB saves program code as "M files" with the extension "**.m**". For the following explanation to be useful, the reader should consult the introductory book that comes with MATLAB or some other MATLAB programming reference[2] to learn how to load an M file and how to run an M file within MATLAB. It is also a good idea to learn how to assign variables in MATLAB before reading further.

1. WHISKER2: BUILDING THE GUI

The first 65 lines of code in Whisker2.m build the **G**raphical **U**ser **I**nterface (GUI). All users of modern computers are familiar with *using* GUIs, even if they've never heard the term. In MATLAB, GUIs are figures (essentially backgrounds) with push buttons and other things (sliders, radio buttons, etc) the user can interact with to get an observable outcome. The

[2] The corporation that makes MATLAB, The Mathworks, maintains an infinitely useful website devoted to MATLAB: www.themathworks.com. Do a search for "Getting Started" to get step-by-step instructions from the people who actually made MATLAB.

results of these interactions either change the figure in some way (like turning on context dependent buttons) or display data in axes. The complete Whisker2 GUI is shown above in the section titled **How to Use Whisker2 and WhiskerView**.

1.1 The Figure Window

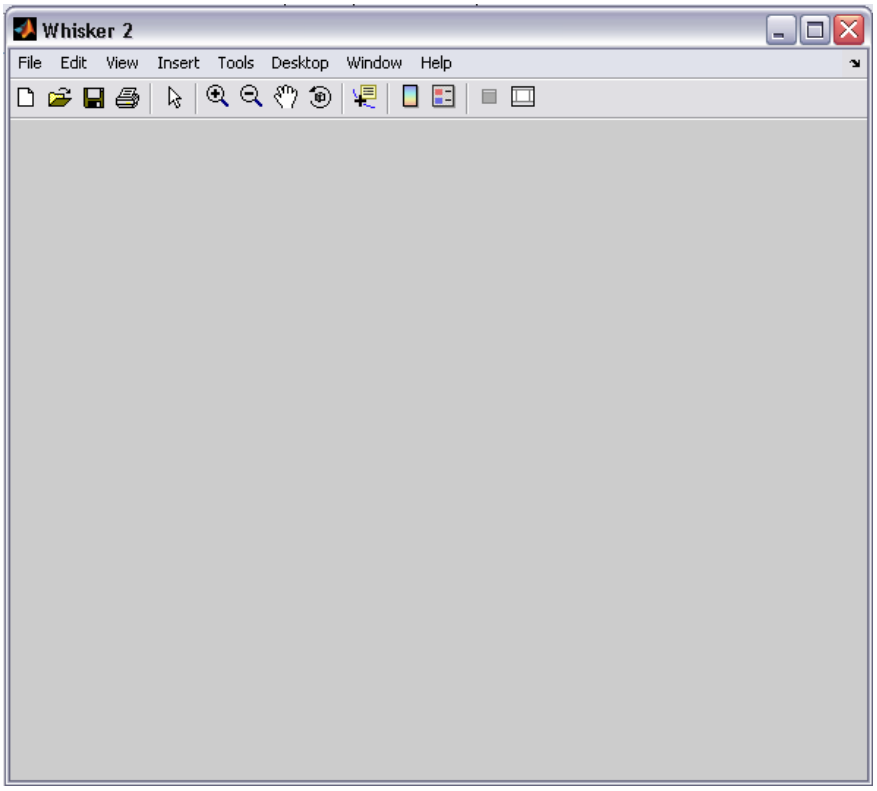
The following lines of code build the figure window:

```
PrimPlot=figure('NumberTitle','off',...
'Name','Whisker 2',...
'position',[10 200 585 450]);
```

Notice the use of ellipses to break up a continuous line of code into shorter segments. All lines of code need to end in a semicolon, unless the user wants a result displayed at the command line. This line makes PrimPlot a figure with certain properties. The properties can be listed in any order, as long as they are formatted properly. The first property is NumberTitle. If MATLAB makes a figure, it numbers it in the upper left hand part of the title bar. If a program is run 8 times, each window will get a particularly annoying number from 1 to 8. By setting this property to 'off,' the figure window remains number-free. The next property is the name of the figure. It is set to Whisker 2. The last line sets the position and dimensions of the figure. MATLAB uses the following format for setting dimensions:

[Distance to left border Distance to bottom border width height].

If a user runs the first line of code, they would get the following empty figure:



Where Whisker2 starts on the screen can easily be changed by manipulating the 'position' term in the first line of code. The next two lines of code set up some important parameters that will be used by the rest of the program:

```
set(gcf,'doublebuffer','on')
scrsize=get(gcf,'position');
```

The 'set' function is one of the most important MATLAB functions used to build GUIs. Set changes figure properties using the following format:

set(figure,'property name', 'property value')

the 'gcf' function means **G**et **C**urrent **F**igure. The current figure is the one previously made in the first line of code, the figure the GUI is built in. This line is **set**ting the current figures 'doublebuffer' to 'on.' Turning the double buffer on increases the amount of memory MATLAB allots to the figure. This helps

make graphics-intensive programs like Whisker2 run without noticeable jumps when the screen redraws. The second line uses the 'get' function to **G**et the **C**urrent **F**igure's position values and set them equal to the variable 'scrsize.' The matrix 'scrsize' is a single row matrix[3] that now looks like this:

[10 200 585 450]

The importance of getting data out of matrices to successful MATLAB programming can not be overstated. It is THE most important thing a beginning programmer can learn. To get the first data value (distance from the left screen border) out of 'scrsize' and store it in the variable 'x,' the user must use the following code:

```
x=scrsize(1,1)
```

Similarly, to get the second data value (distance from the bottom border) out of 'scrsize' and store it in 'x,' the user must use the following code:

```
x=scrsize(1,2)
```

To save a specific member of a matrix into a variable, use the following general format:

```
variable=matrixname(row,column)
```

An entire column can be accessed by using a colon in place of the row value:

```
Variable=matrixname(:,column)
```

This also works in reverse for accessing an entire row:

```
Variable=matrixname(row,:)
```

It should be obvious that using a colon to access an entire column is the same as specifying row #1 for a matrix that only has one row. So-called 'hard coding' of numbers into a program is considered to be weak programming.

[3] It's actually a vector, but that's unimportant for the current discussion.

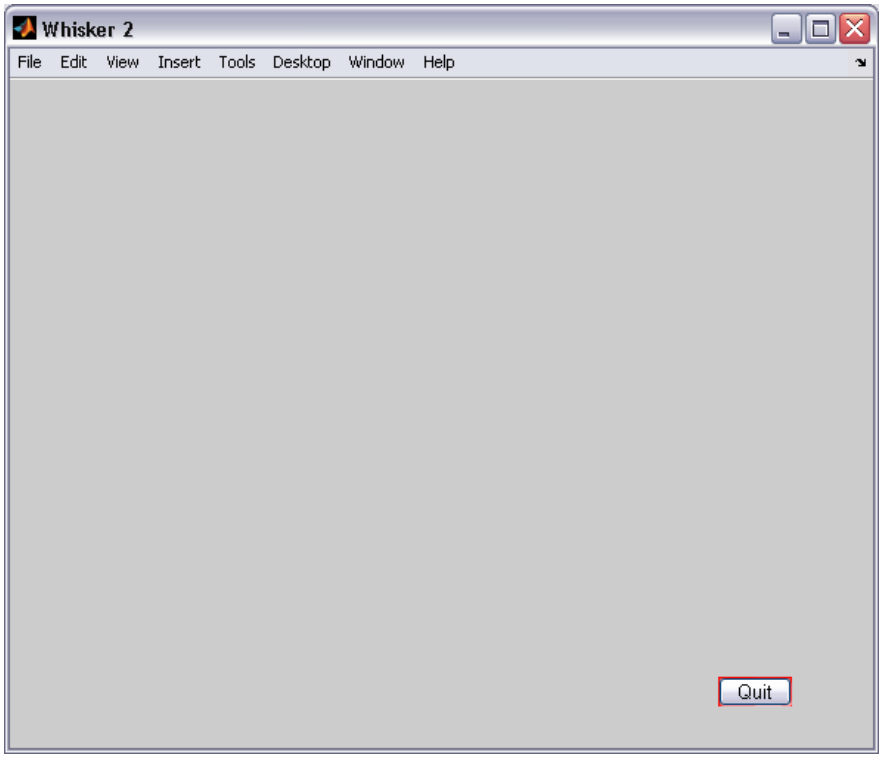
Whenever possible, relationships between different variables should use number-free variables or matrix indicators like the colon. This makes programs that are flexible and easier to change.

1.2 Making Push Buttons

By storing the figure's screen size in the matrix 'scrsize,' it will be available for following lines of code that make pushbuttons proportional to the figure. The 'Quit' button will serve as an example of how to make such a pushbutton. The lines of code that build the 'Quit' button are:

```
quitbutton=uicontrol('style','pushbutton',...
'string','Quit',...
'backgroundcolor',[1,0,0],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+480 ((scrsize(:,2))/35)+23 50 20],...
'callback','s=1;close;');
```

If a user runs the figure definition code (`PrimPlot=`) and the code shown above, they should see the following figure:



These lines of code define the 'Quit' button as the **U**ser **I**nterface **C**ontrol (uicontrol in MATLAB-speak) with 'style' 'pushbutton' and the text 'string' 'Quit." Push button is just one of many uicontrol styles available in MATLAB. The text 'string' in the second line is the text that will appear on the push button. The third line of code sets the 'Quit' button's background color to red. MATLAB specifies colors using a matrix in the following format:

[red green blue]

Values of individual colors can vary from 0 to 1. Any color can be made by varying the proportions of red, green, and blue. The text string that appears on the push button can be changed by varying the 'fontsize' property. The 'position' property uses values found in the matrix 'scrsize' to calculate a proportional distance for the 'Quit' button. This effectively keeps the 'Quit' button in the same place if the figure window is resized. The last property, 'callback,' is the most important part of the 'Quit' button definition. Whenever the 'Quit' button is pushed, the value of the variable 's' is changed to 1 and then the figure is closed. The 'callback' property is how a push button interacts with the underlying program. This will be discussed in Part 2. All the other push buttons in Whisker2 are constructed using the same format as the 'Quit' button.

1.3 Making Text Fields

In Whisker2, text fields are used to display information about the coordinates of marked whiskers and how many whiskers have been marked. How these values actually change following user input will be discussed in Part 2. The code for defining the whisker count text field is shown below:

```
numSize=uicontrol('style','text',...
'string','0',...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+15 ((scrsize(:,2))/35)+1 25 15]);
```

The format is not all that different from the code that defined the 'Quit' button. The 'style' property is set to 'text.' On starting Whisker2, this text field is set to 0 by having the property 'string' set to the character '0.' The coordinate text fields are constructed using the same format as the whisker count text field.

1.4 Mouse Button Action

GUIs are designed to be used with a mouse or some other 'clickable' input device. Whisker2 uses mouse clicks to enter coordinate values for whiskers. How this is actually done will be discussed in Part 2. It is still essential to define what happens when a mouse button is clicked.

```
set(gcf,'windowbuttondownfcn','mousedown=1;');
```

The 'windowbuttondownfcn' is a function present in all MATLAB figure windows. This line of code has that function set the variable 'mousedown' to one.

2. WHISKER2: IMPLEMENTING THE GUI: USING 'WHILE' LOOPS, 'FOR' LOOPS, AND 'IF' STATEMENTS

The previous section detailed the 'building' aspect of GUI design. This section will detail how the GUI of Whisker2 actually works. For a programmer who has never made a GUI, the logic behind their operation can seem counterintuitive. Before the line-by-line workings of Whisker2 are spelled out, it is necessary to give a brief outline of the inner working of the GUI.

2.1 The 'While' Loop

The most important line of code in Whisker2 is the 'while' at line 67:

```
while (s==0)
```

[4] At the MATLAB command line, type in `help relop` for information on relational operators like '=='

A ‘while’ loop consists of a ‘while’ statements followed by lines of code to be executed and then an ‘end’ statement. A ‘while’ loop executes all the lines of code within them as long as their conditional statement is true. This is perhaps easier to visualize if the programmer imagines the computer ‘reading’ through the code line by line until it comes to a ‘while.’ It checks the conditional statement. If s is indeed equal to 0, it then begins to execute the code following the ‘while.’ Then it reaches the ‘end’ statement and goes back to the ‘while.’ It does this over and over again, for as many cycles per second as the programmer’s computer can run per second until the conditional statement is not true. In the case of Whisker2, the conditional statement is the variable associated with the ‘Quit’ button. In a very real sense, all the GUI *is*, is a set of code that runs as long as the ‘Quit’ button *has not* been pushed. When the ‘Quit’ button is pushed, its ‘callback’ property returns a value of s=1. The ‘while’ loop’s conditional statement is no longer true, and the figure window closes. All other push button ‘callback’ properties are evaluated by ‘if’ statements within the ‘while’ loop. It should be obvious that these ‘callback’ properties can not be evaluated if nothing within the ‘while’ loop is being executed.

2.2 The ‘If’ Statement

Recall that the push button definition code has a ‘callback’ property. The ‘callback’ property is used to set what code is **call**ed when that button is pushed. Within the ‘while’ loop, ‘if’ statements check the values of these callbacks. ‘If’ statements begin with an ‘if’ and end with an ‘end.’ Lines of code within the ‘if’ statement are executed when the ‘if’ statement’s conditional statement is true. In the Whisker2 GUI, the variable in the ‘if’ statement’s

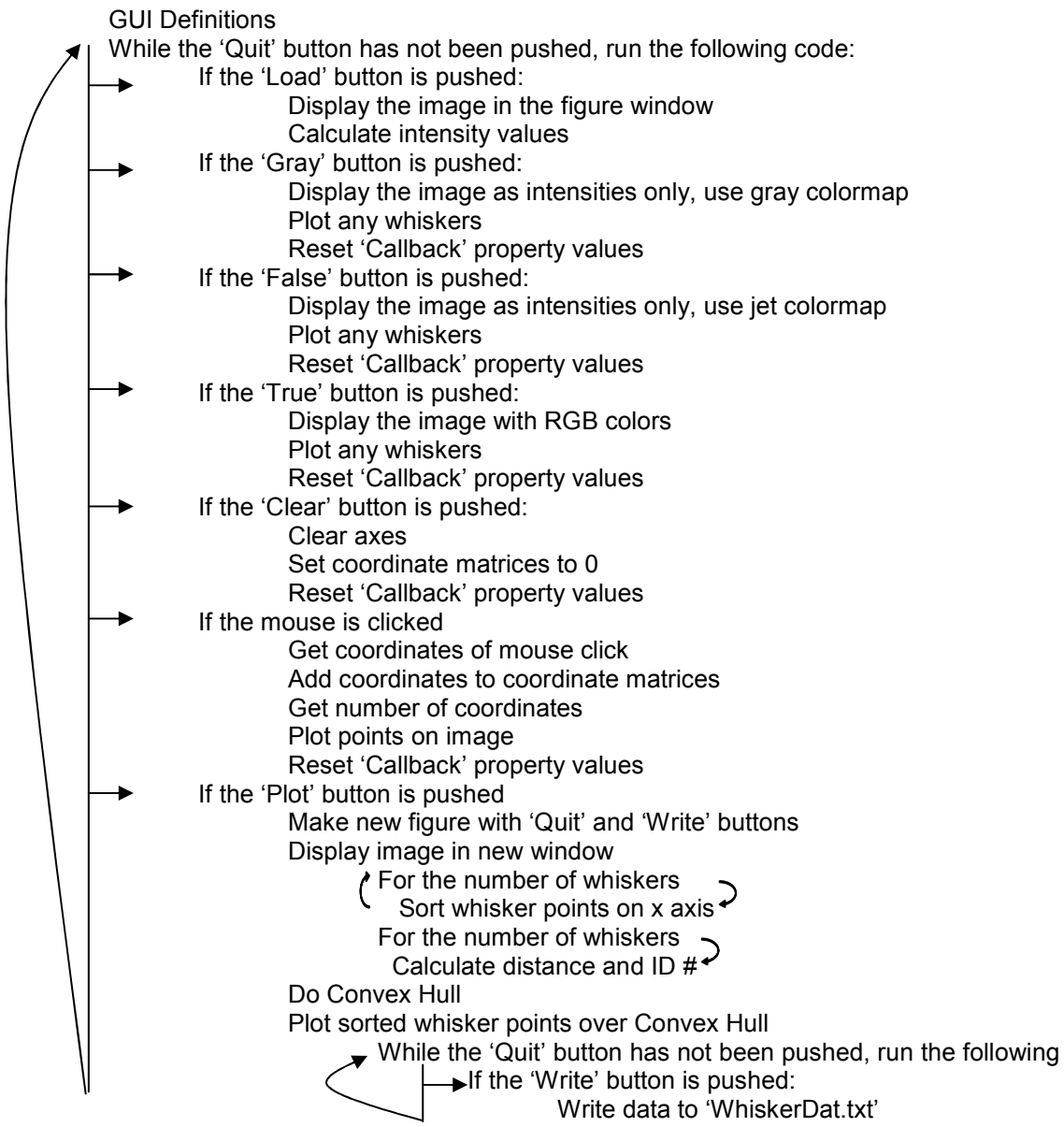
conditional statement is the ‘callback’ of the push button for that code. Examples can be found in the explanation of each push button below.

2.3 The ‘For’ Loop

‘For’ loops begin with a ‘for’ and end with an ‘end.’ Code within the ‘for’ loop is run a certain number of times specified in the initial ‘for’ statement. In Whisker2, ‘for’ loops are used to perform sort and distance calculations for each whisker. How ‘for’ loops are used to do this is shown in the explanations of the push buttons below.

2.4 Pseudocode and the Line-By-Line Workings of Whisker2

What follows is a ‘pseudocode’ version of the GUI control structure. It is common practice to first write out a program in hybrid English/MATLAB ‘pseudocode.’ The ‘pseudocode’ is then translated into MATLAB.



2.4.1 The 'Load' Button

The following lines of code are executed when the 'Load' button is pushed and the callback variable 'v' is set to 1.

```
if(v==1)
 s=0;v=0;clear=0;mousedown=0;mouseup=0;mousemotion=0;pnew=0;
 gry=0;fls=0;tru=0;
 set(truebutton,'enable','on');
 set(truebutton,'backgroundcolor',[.8,.8,.8]);
 set(graybutton,'enable','on');
 set(graybutton,'backgroundcolor',[.8,.8,.8]);
```

```
 set(falsebutton,'enable','on');
 set(falsebutton,'backgroundcolor',[.8,.8,.8]);
 set(clearbutton,'enable','on');
 set(clearbutton,'backgroundcolor',[1,1,1]);
 [filename,pathname]= uigetfile('*.jpg');
   try
     imread([pathname,filename], 'JPEG');
   catch
     errordlg('load did not work, press quit and restart')
   end
 x=imread([pathname,filename], 'JPEG');
 image(x)
 xC=double(x);
 xC=.2989*xC(:,:,1)+.5870*xC(:,:,2)+.1140*xC(:,:,3);
 set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
 hold on
end
```

The first line resets all the other 'callback' variables. This is a common feature of all the push button code in Whisker2. It ensures that a push button doesn't become inactive after it is clicked once. The next lines of code:

```
set(truebutton,'enable','on');
set(truebutton,'backgroundcolor',[.8,.8,.8]);
```

use the 'set' function to turn the 'enable' property to on. This is a way to make push buttons context dependent. Only after the 'Load' button has been pushed, and a picture is displayed, do buttons that can affect that image become clickable. This is a way of limiting the damage a user can do to a program by clicking on buttons that shouldn't be clicked. These lines of code also set each button to a different color. The next lines of code:

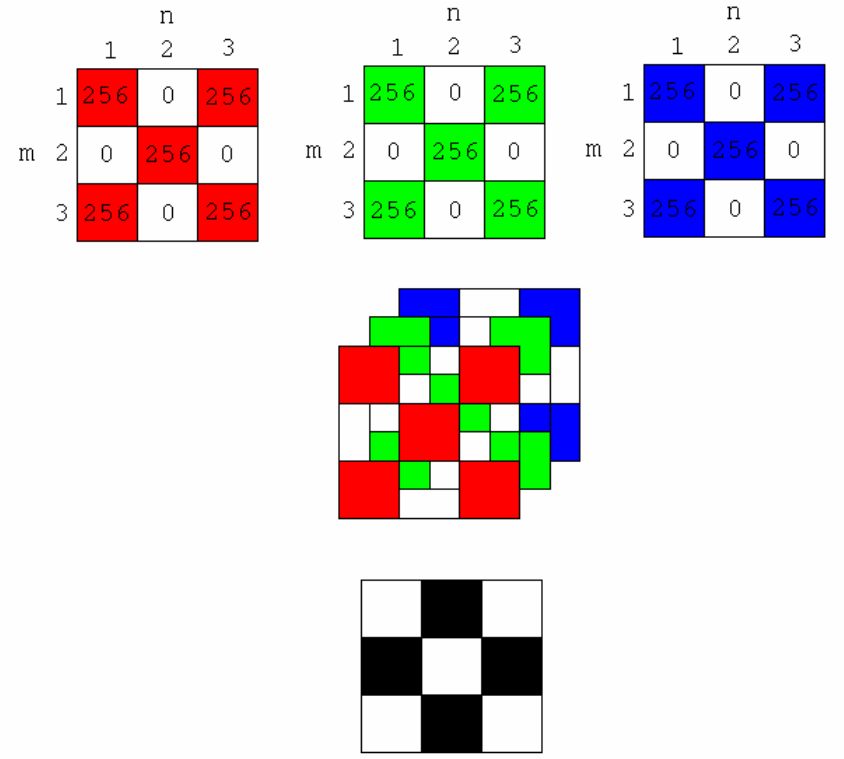
```
[filename,pathname]= uigetfile('*.jpg');
   try
     imread([pathname,filename], 'JPEG');
   catch
     errordlg('load did not work, press quit and restart')
   end
 x=imread([pathname,filename], 'JPEG');
 image(x)
```

use the uigetfile function to launch a pop-up window that allows the user to browse for an JPG compressed image file to load. 'Imread' should be used to download an image file. 'Imread' preserves the unique matrix structure of an

RGB image. The “try”-“catch”-“end” structure allows for the quick addition of an error message if the user botches the load operation. The loaded file is stored in the variable ‘x.’ The ‘image’ function is used to display ‘x’ in the axes of the figure window. The next lines of code are important to the rest of the color management buttons:

```
xC=double(x);
xC=.2989*xC(:,:,1)+.5870*xC(:,:,2)+.1140*xC(:,:,3);
```

The first line takes care of a quirk of working with JPG images in MATLAB. MATLAB’s ‘imread’ function saves a JPG image as an mXnX3 matrix of uint8 integers. Each pixel has m and n coordinates and three color values: red, green, and blue (hence, RGB). This can be visualized as:



Uint8 integers range from 0 to 256. A value of 256 is the maximum value for that color; zero is the minimum. Pixels with the maximum value in all three colors are white. Pixels with the minimum value in all three colors are black.

Using the ‘double’ function on ‘x’ converts uint8 integers to double precision numbers that range from 0 to 1. Using this scheme, a pixel with 1s in all three colors is white, 0 is black. Once the matrix is in double precision numbers, all of MATLAB’s matrix operators can be used. The next line multiplies all the color values in such a way that each pixel now has a single value from 0 to 1. This effectively duplicates NTSC grayscale and converts the image from RGB to a matrix of light intensities (ref). This is done in the ‘Load’ button code because it is computationally expensive. By doing all the heavy image calculations in the same push button, the user does not notice the lag time later when another button is pushed. The last lines of code:

```
 set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
 hold on
end
```

use the ‘set’ and **G**et **C**urrent **A**xes (gca) functions to lock the axes in place and set the font size of the axes at seven. Locking the axes stops them from changing size if the user loads an image that is a different size from the one before it. This wreaks havoc with the push buttons and the aesthetic feel of the program. The ‘hold on’ function is used so subsequent code can draw over the current axes. Without this function, the next executed line of code will create a new set of axes that will white out the old axes. The ‘end’ statement signals the end of the ‘Load’ button ‘if’ statement.

2.4.2 The ‘False’, ‘Gray’, and ‘True’ Buttons

These buttons are grouped together because they are basically the same code but with slight variations. The ‘Gray’ button will be used as an example. The following lines of code are executed when the ‘Gray’ button is pressed and the ‘callback’ variable ‘gry’ is set to 1.

```
if(gry==1)
 hold on
 imagesc(xC);
 colormap gray;
 plot(x1,y1,'b*');
 set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
 s=0;v=0;clear=0;mousedown=0; mouseup=0;
 mousemotion=0;pnew=0;gry=0;fls=0;tru=0;
end
```

The ‘hold on’ function performs the same function as it did in the ‘Load’ button code explained above. The ‘imagesc’ function is used to **image** the **sc**aled intensity matrix created by the ‘Load’ button code and stored in the matrix ‘xC.’ The next line tells ‘imagesc’ to use the colormap ‘gray.’ A colormap is a set of predefined values used to assign color values to intensity images. MATLAB has many colormaps. The ‘False’ button uses the ‘jet’ colormap to highlight areas that are white in the original RGB image. In the next line of code, the ‘plot’ function is used to display the x,y coordinates of marked whiskers (stored in the matrices x1, y1) as blue asterisks (‘b*’). The last three lines perform the same function as in the ‘Load’ button.

2.4.3 The ‘Clear’ Button

The ‘Clear’ button is relatively straight forward in design:

```
if(clear==1)
 cla;
 set(clearbutton,'enable','off');
 set(clearbutton,'backgroundcolor',[.8,.8,.8]);
 set(plotnewbutton,'enable','off');
 set(plotnewbutton,'backgroundcolor',[.8,.8,.8]);
 set(truebutton,'enable','off');
 set(truebutton,'backgroundcolor',[.8,.8,.8]);
 set(graybutton,'enable','off');
 set(graybutton,'backgroundcolor',[.8,.8,.8]);
 set(falsebutton,'enable','off');
 set(falsebutton,'backgroundcolor',[.8,.8,.8]);
 set(numx,'string','x');
 set(numy,'string','y');
 set(numSize,'string','0');
 x1=[];y1=[];
 s=0;v=0;clear=0;mousedown=0;pnew=0;gry=0;fls=0;tru=0;
end
```

The first line uses the ‘cla’ function to **cl**ear the current **a**xes. The next ten lines set all (except ‘Load’ and ‘Quit’) button’s ‘enable’ property to ‘off’ and their background color to gray ([.8,.8,.8]). The ‘Clear’ button is the first place in Whisker2 where a text field is changed. These lines are not different from the other ‘set’ functions that have been used so far. The last three lines reset the coordinate matrices x1 and y1 as well as the ‘callback’ variables and ‘end’ the ‘Clear’ button ‘if’ statement.

2.4.4 Mouse Click

The following lines of code are the heart of Whisker2. This is where mouse clicks are converted to x,y coordinates that the rest of the program uses for calculations. Recall that in the original GUI definition code the ‘windowbuttondownfcn’ was set to return a ‘callback’ variable of ‘mousedown’ equal to one. The following lines of code are executed when the mouse button is clicked:

```
if(mousedown==1 & pnew==0)
 set(plotnewbutton,'enable','on');
 set(plotnewbutton,'backgroundcolor',[0,1,0]);
 pt=get(gca,'currentpoint');
 set(numx,'string',[num2str(pt(1,1))]);
 set(numy,'string',[num2str(pt(1,2))]);
 x1=[x1;pt(1,1)];
 y1=[y1;pt(1,2)];
 xS=size(x1);
 xS=xS(:,1);
 set(numSize,'string',[num2str(xS)]);
 points=[x1 y1];
 hold on
 plot(x1,y1,'b*');
 drawnow
 s=0;v=0;clear=0;mousedown=0;gry=0;fls=0;tru=0;
end
```

The first two lines of code turn on the ‘Plot’ button and turn it gray. The next line of code uses the ‘get’ function to determine the ‘currentpoint’ property of the current axes and assign the value of this property to the matrix ‘pt.’ Pt is a 1X2 matrix. The next two lines of code:

```
set(numx,'string',[num2str(pt(1,1))]);
set(numy,'string',[num2str(pt(1,2))]);
```

use the ‘num2str’ function to convert the number output of the ‘string’ property of the two coordinate text fields to the corresponding coordinate values. The coordinate matrices x1 and y1 are built with each additional coordinate by using the lines:

```
x1=[x1;pt(1,1)];
y1=[y1;pt(1,2)];
```

These lines are a good example of how to sequentially append values of one matrix (pt) to another (x1 and y1). The following lines of code:

```
xS=size(x1);
xS=xS(:,1);
set(numSize,'string',[num2str(xS)]);
```

use the ‘size’ function to get the number of coordinates. Because ‘x1’ is a nX1 matrix , ‘Size’ returns a 1X2 matrix corresponding to the number of rows (n) and columns (1). The actual size value is extracted by getting the value of ‘xS’ at (1,1) which is the same as (:,1). This value is then converted to the ‘string’ property of the whisker count text field (numSize) using the ‘set’ function. The ‘hold on’ function precedes the ‘plot’ function that graphs the whisker coordinates (x1, y1) as blue stars (‘b*’’). The ‘drawnow’ function forces MATLAB to graph the whisker coordinated before exiting the mouse click ‘if’ statement. The last line resets the ‘callback’ variables.

2.4.5. The ‘Plot’ Button

The ‘Plot’ button is really another program nested within Whisker2. It calculates the three crucial data variables used to identify individuals. It results in the creation of another figure window with text fields and another ‘Quit’ button-defined ‘while’ loop GUI control structure. The ‘Write’ button within this window exports data to a text file for later analysis. Because the

code for the ‘Plot’ button is long, it will be explained in sections. The GUI definition code works exactly like the GUI definition code of the primary figure window:

```
if(pnew==1 & v==0)
 set(gcf,'doublebuffer','on')
 st=0;
 write=0;
 scrsize=get(gcf,'position');
 SecPlot=figure('NumberTitle','off',...
 'Units','pixels',...
 'position',...
 [(30*(scrsize(:,1))) ((scrsize(:,2))) ((scrsize(:,3))/2)
  ((scrsize(:,4))/2)]);
 x=imread([pathname,filename], 'JPEG');
 image(x);
 quitbutton2=uicontrol('style','pushbutton',...
 'string','Quit',...
 'backgroundcolor',[1,0,0],...
 'fontsize',10,...
 'position',[((scrsize(:,1))/20)+5 ((scrsize(:,2))/35)+1 50 20],...
 'callback','st=1;close;');
  writebutton=uicontrol('style','pushbutton',...
 'string','Write',...
 'backgroundcolor',[1,1,1],...
 'fontsize',10,...
 'position',[((scrsize(:,1))/20)+205 ((scrsize(:,2))/35)+1 50 20],...
 'callback','write=1;');
  hold on
```

The ‘doublebuffer’ is turned on, ‘callback’ variables for the ‘Quit’ and ‘Write’ buttons are initialized. The ‘scrsize’ matrix is taken from the ‘position’ property of the primary figure window. This is used to make proportional ‘Quit’ and ‘Write’ buttons. The current image is reloaded using variables obtained by the code for the ‘Load’ button. The ‘hold on’ function leaves the figure open for more drawing. The following lines of code initialize matrices that will be used in sorting coordinates and calculating distances between them.

```
z=[];
d=[];
```

This line of code,

```
[points2,I]=sort(points);
```

uses the ‘sort’ function to sort the matrix ‘points.’ Recall that ‘points’ was made in code executed following a mouse button click. It contains x1 and y1 as columns. In order to understand how ‘sort’ works, the following example ‘points’ matrix will be used:

```
points =    174.9812  164.6417
            211.7141  163.3338
            196.1733  182.9523
```

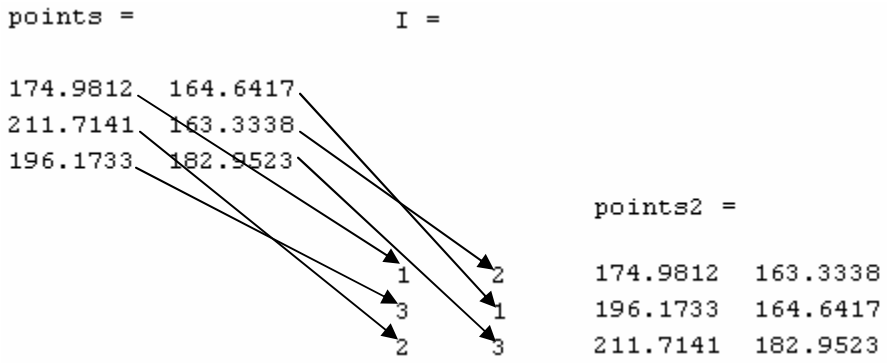
Notice that the coordinates are not sorted at all. Following the ‘sort’ function, ‘points’ becomes ‘points2’:

```
points2 =  174.9812  163.3338
           196.1733  164.6417
           211.7141  182.9523
```

This is a step in the right direction, the x values are sorted in increasing order. However, the x values no longer correspond to the correct y values. Thankfully, one of the outputs of the ‘sort’ function is a matrix that maps how values moved during the sort. This map is stored in ‘I’:

```
I =     1     2
        3     1
        2     3
```

The following illustration shows how ‘sort’ moved the values in ‘points’ to get the values in ‘points2’.



Whisker2 uses a ‘for’ loop to put the sorted x values back with their corresponding y values. To do the ‘for’ loop, the following lines of code calculate the number of times the ‘for’ loop should run:

```
N=[size(x1)];
N2=N(1,1);
```

This is simply the number of whiskers. The ‘for’ loop takes advantage of the fact that the ‘y1’ coordinate matrix is still in the proper order. The following code runs as many times as there are whiskers,

```
for j=1:N2,
    z=[z;(points2(j,1)),y1((I(j,1)));];
    x3=[z(:,1)];
    y3=[z(:,2)];
end
```

Following our example data, the ‘for’ loop rapidly does the following steps:

```
N2=3
For j=1:1,                                    Points2     y1
 z=[z;(points2(1,1)),y1((I(1,1)));];  174.9812  164.6417
 x3=[z(:,1)];
 y3=[z(:,2)];

For j=1:2,                                    196.1733  163.3338
 z=[z;(points2(2,1)),y1((I(2,1)));];
 x3=[z(:,1)];
 y3=[z(:,2)];

For j=1:3,                                    211.7141  182.9523
 z=[z;(points2(3,1)),y1((I(3,1)));];
 x3=[z(:,1)];
 y3=[z(:,2)];
end
```

After running N2 times, the ‘for’ loop has constructed the following matrices:

```
z =   174.9812  164.6417      x3 =  174.9812          y3 =  164.6417
      196.1733  182.9523            196.1733                182.9523
      211.7141  163.3338            211.7141                163.3338
```

Now the coordinates are sorted along the x axis. The next 'for' loop calculates the distance from one whisker point to the next using the MATLAB version of the distance formula:

```
for i=1:(N2-1)
  d=[d;sqrt(((z((i+1),1)-z(i,1))^2)+((z((i+1),2)-z(i,2))^2))];
end
```

Again using our example data, the 'for' loop rapidly does the following steps:

```
                                                          Whisker
N2=3; N2-1=2;                                             distance
for i=1:2
  d=[d;sqrt(((z(3,1)-z(2,1))^2)+((z(3,2)-z(2,2))^2))];  | (3-2)
for i=1:1
  d=[d;sqrt(((z(2,1)-z(1,1))^2)+((z(2,2)-z(1,2))^2))];  | (2-1)
end
```

Note that the 'for' loop only goes to N2-1. This stops the distance formula from trying to find the distance from the last whisker to a nonexistent whisker. The next line of code computes the distance from the whisker with the smallest x coordinate to the whisker with the largest x coordinate:

```
dWide=sqrt(((z(1,1)-z(N2,1))^2)+((z(1,2)-z(N2,2))^2));
```

Using the example data,

```
                                                          Whisker
                                                          distance
N2=3;
dWide=sqrt(((z(1,1)-z(3,1))^2)+((z(1,2)-z(3,2))^2));   | (3-1)
```

All of the distances are added up (using the 'sum' function) and then divided by the distance from the smallest x coordinate to the largest x coordinate to get 'ID':

```
dSum=sum(d);
ID=dSum/dWide;
```

The next lines of code set up and perform the Convex Hull. Using the MATLAB function 'convhull,' a polygon equal to the smallest area that encloses all the whiskers is found:

```
a1=[];
```

```
a=[x1,y1];
cvh2=(convhull(x1,y1));
a1=a(cvh2,:);
x2=a1(:,1);
y2=a1(:,2);
N3=size(x2);
NCH=(N3(1,1))-1;
fill(x2,y2,'r');
```

The matrix ‘a1’ is initialized. Matrix ‘a’ is then filled with the unsorted coordinates from the coordinate matrices x1 and y1. The matrix ‘cvh2’ is set equal to the results of ‘convhull’ performed on x1 and y1. ‘Cvh2’ becomes a matrix that specifies the whiskers that form the vertices of the polygon. The matrix ‘a1’ is then set equal to the coordinates found in ‘a’ and specified by ‘cvh2.’ New coordinate matrices, x2 and y2, are set equal to their respective parts of ‘a1.’ This is shown using the example data below:

```
y1 =        x1 =        a =                    cvh2 =      a1 =

164.6417    174.9812    174.9812    164.6417       1        174.9812    164.6417
163.3338    211.7141    211.7141    163.3338       2        211.7141    163.3338
182.9523    196.1733    196.1733    182.9523       3        196.1733    182.9523
                                                   1        174.9812    164.6417

                                                            x2 =        y2 =

                                                            174.9812    164.6417
                                                            211.7141    163.3338
                                                            196.1733    182.9523
                                                            174.9812    164.6417
```

The number of coordinates that specify the convex hull polygon is saved in the variable ‘NCH.’ Note that NCH is the number of convex hull polygon vertices minus one. This is because the ‘convhull’ function adds a coordinate to make a closed loop. This has to be accounted for when getting a true value of the number of convex hull polygon vertices. Using the ‘fill’ function, a red (‘r’) polygon is drawn using the coordinates in x2 and y2.

The following lines of code add the x value sorted whisker coordinates (x3 and y3) as a white line (‘w-‘) over the red convex hull polygon.

```
hold on
y6=plot(x3,y3,'w-');
```

A function called ‘moveplot’ allows individual points to be moved in real time before the figure is saved.

```
moveplot(y6,'axy');
```

Text fields that display the image’s filename and computed data are defined and displayed.

```
num2=uicontrol('style','text',...
'string','',...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+55 ((scrsize(:,2))/35)+1 150 20]);
set(num2,'string',[filename   '         '   (num2str(N2))   '         '
(num2str(NCH))]);
```

Then the axes are locked and turned off:

```
set(gca,'xaxisloc','top',...
'yaxisloc','right',...
'XDir','normal',...
'YDir','reverse',...
'fontsize',8);
axis off;
```

All of the previous code is executed once the ‘Plot’ button in the primary window is pushed, but before the new figure window’s ‘Quit’ or ‘Write’ buttons are pushed. The following lines of code control the push button actions of the second figure window:

```
while (st==0)
 if (write==1)
  wrt=[filename ' ' (num2str(N2)) ' ' (num2str(NCH)) '...
    '(num2str(ID))];
  fid=fopen('WhiskerDat.txt','a+');
  fprintf(fid,'%s\n',wrt);
  fclose(fid);
  write=0;
```

```
 end
```

The ‘while’ loop executes its contained code as long as the ‘Quit’ button is not pushed (st=0). When the ‘Write’ button is pushed, the ‘Write’ button’s ‘if’ statement saves the current image’s filename, the number of whiskers, the number of convex hull polygon vertices, and the ID number as a ‘string’ matrix ‘wrt’:

```
wrt=[filename ' ' (num2str(N2)) ' ' (num2str(NCH)) '...
    '(num2str(ID))];
```

The following line of code uses ‘fopen’ to open the text file WhiskerDat. This function will also create the file if it doesn’t already exist. The property ‘a+’ allows more data to be appended to the file.

```
fid=fopen('WhiskerDat.txt','a+');
```

This line uses ‘fprintf’ to save data in ‘wrt’ to the file represented by ‘fid’:

```
fprintf(fid,'%s\n',wrt);
```

‘%s’ is the format for saving a ‘string.’ There are many %-letter combinations used in saving data with ‘fprintf.’ The ‘\n’ specifies a carriage return after the data string. This creates an ordered file of columns and rows that can be easily exported to Excel for further analysis. ‘Fclose’ closes the file represented by ‘fid.’ The ‘callback’ variable ‘write’ is reset to zero. The ‘Write’ button’s ‘if’ statement is ended with an ‘end’ statement.

```
  fclose(fid);
  write=0;
 end
```

2.4.6 Ending the Program

The following code ends the program. First, there is a ‘drawnow’ function that follows the ‘end’ of the second figure window’s ‘Write’ button’s ‘if’ statement. This and subsequent ‘drawnow’ functions force MATLAB to output

the results of preceding code to the screen immediately. Without these functions, the results of pushed buttons will not be seen immediately. 'Callback' variables are reset before the 'end' of the 'Plot' button's 'if' statement. Then the 'Plot' button's 'if' statement is followed by the first 'while' loop's 'end' statement. The last line of the program is a 'close' statement.

```
   drawnow
  end <------------------------- 'End' of the 2nd 'while' loop
  s=0;v=0;clear=0;mousedown=0;pnew=0;gry=0;fls=0;tru=0;
  drawnow;
 end <-------------------------- 'End' of the 'Plot' button 'if'
 drawnow
end <---------------------------- 'End' of the 1st 'while' loop
close
```

3. WhiskerView: Building the GUI

WhiskerView is actually modified from a very early version of Whisker2, a version I wrote when I was figuring out how to display pictures in MATLAB. WhiskerView is no where near as complicated as Whisker2. All it does is load and display pictures. However, it does a few interesting things that Whisker2 *does not* do. Some of these show up in the GUI definition code.

3.1 The Figure Window

The figure window is defined exactly the same as in Whisker2, 'doublebuffer' is turned on, and position values are loaded into the matrix 'scrsize'.

3.2 Making Push Buttons.

The ‘Quit’, ‘Clear’ and ‘View’ buttons are defined like a standard push button. Although there is somewhat of a precedent for this in the code that defines the ‘Quit’ button, things do get tricky in the code for defining the buttons that move through pictures. The ‘>>’ button will be used as an example:

```
nextbutton=uicontrol('style','pushbutton',...
    'string','>>',...
    'backgroundcolor',[.8,.8,.8],...
    'fontsize',9, ...
    'position',[((scrsize(:,1))/20)+220 ((scrsize(:,2))/35)+21 50
20],...
    'callback', 'nxt=1;cnt=cnt+1');
```

There’s nothing unusual here until the last line, the ‘callback’ definition. The callback variable ‘nxt’ is set equal to one. And then there’s some math:

```
cnt=cnt+1
```

The variable ‘cnt’ is the heart of WhiskerView. It is what the code in the ‘>>’ button ‘if’ statement uses to figure out what picture to display. It is a counter variable that keeps track of how many times the ‘>>’ or ‘<<’ button has been pushed. It is in the ‘callback’ definition for a very good reason. That will be explained in the section below that explains that piece of code.

3.3 Making Text Fields

In WhiskerView, text fields are used to display the filename of the current image and to enter the file name of the target image to be loaded. The difference between these two is reflected in the definition code:

```
numFile=uicontrol('style','text',...
    'string','',...
    'fontsize',10,...
    'position',[((scrsize(:,1))/20)+110 ((scrsize(:,2))/35)+330 100
15]);
edittext=uicontrol('style','edit',...
    'string','ADULT000.jpg',...
```

```
    'fontsize',9,...
    'position',[((scrsize(:,1))/20)+70 ((scrsize(:,2))/35)+2 250
20]);
```

'numfile' defines the text field that will display the filename of the current image. 'edittext' defines the text field that will be used to enter the file name of the target image to be loaded. As far as defining the two, the only difference is in the 'style' property. They will, of course, be handled somewhat differently in the 'if' statements below.

3.4 Getting Image Filenames Into WhiskerView.

Whisker2 did not have any important functions or calculations before its main 'while' loop. WhiskerView has one key function that gets data that will be used by the rest of the program, it comes before WhiskerView's main 'while' loop.

```
files=dir('*.jpg');
```

The 'dir' function looks in MATLAB's 'work' folder for files with a specified extension. In this case it is looking for JPG files. The 'dir' function returns an array of structures with the following fields: name, date, bytes, and isdir. Defining what a structure is or even better, what an *array* of structures is, is outside the scope of this documentation. Suffice it to say, the 'dir' function gets the names of all the JPG files in the MATLAB 'work' folder. How this is used will be described in the 'if' statement descriptions below.

4. WHISKERVIEW: IMPLEMENTING THE GUI USING 'WHILE' LOOPS AND 'IF' STATEMENTS

The previous section detailed the 'building' aspect of GUI design. This section will detail how the GUI of Whisker2 actually works.

4.1 The Main 'While' Loop.

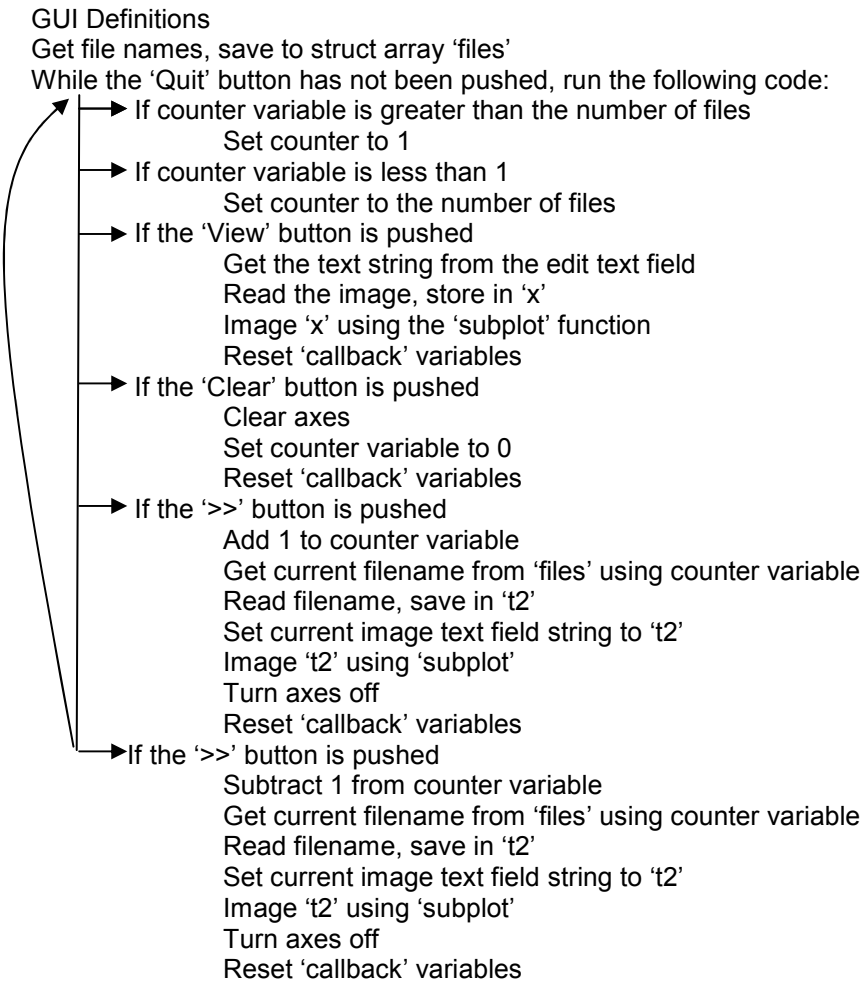
WhiskerView's GUI control structure is a single 'While' loop that contains 'if' statements. The 'while' loop's conditional statement is the 'callback' variable of the 'Quit' button. As long as the 'Quit' button has not been pushed, the 'if' statements within the 'while' loop are evaluated and the program runs. If the 'Quit' button is pushed, the variable 's' is set to 1 and the figure window closes. See this section for Whisker2 to get a feel for how important the 'while' loop is and the big idea behind the GUI control structure.

4.2 'If' Statements

WhiskerView's push buttons were defined at the beginning of the program. Lines of code within the 'if' statement are executed when the 'if' statement's conditional statement is true. In the Whisker2 GUI, the variable in the 'if' statement's conditional statement is the 'callback' of the push button for that code. Examples can be found in the explanation of each push button below.

4.3 Pseudocode and the Line-By-Line Workings of WhiskerView

As established in this section for Whisker2 (above), it is common to write out 'pseudocode' for a program and then translate that into MATLAB. What follows is the pseudocode and then a full explanation of each working part of WhiskerView.



4.3.1 Counter Variable Control

The counter variable 'cnt' is crucial to the operation of WhiskerView. The '>>' and '<<' buttons directly control the state of 'cnt.' They do so through their 'callback' variables:

For the '>>' button:

```
'callback', 'nxt=1;cnt=cnt+1');
```

For the '>>' button:

```
'callback', 'prv=1;cnt=cnt-1');
```

Incrementing and decrementing the counter in the 'callback' ensures that 'cnt' is changed only when the button is actually pushed.

Incrementing and decrementing have their limits. For example, if there are 16 JPG images in MATLAB's 'work' directory, WhiskerView will go right along and add 1 to 'cnt' past 16. This generates an error when WhiskerView tries to open the 17th file in the structure array 'files' and finds there is no 17th filename. The same kind of problem holds for the opposite direction. WhiskerView will continue to decrement below 1, even though there is no 0th or -1th file! The first 'if' statements within the 'while' loop check the status of 'cnt' and make sure it is not in an impossible state:

```
if(cnt>((length(files))))
    cnt=1;
end
```

The 'if' statement above first gets the length of the filename field of the struct array 'files' using the 'length' function. If the 'cnt' variable is greater than the number of possible filenames, it resets 'cnt' to 1. One can imagine the user clicking past the last picture, this 'if' statement resets 'cnt' to 1, and the 1st picture comes up again. Problem solved. The 'if' statement below checks for impossibly low 'cnt' values.

```
if(cnt<1)
    cnt=(length(files));
end
```

If the user has gone backwards (using the '<<' button) to the last filename and clicks again, 'cnt' becomes 0. The previous 'if' statement sets 'cnt' to the number of filenames. This is equivalent to setting 'cnt' equal to the last filename.

4.3.2 The ‘View’ Button

The ‘View’ button isn’t all that different from the ‘Plot’ button of Whisker2. When the ‘View’ button is pushed, the following code is executed:

```
if(v==1)
    x=imread(get(edittext, 'string'));
    subplot('position',[0.5 0.25 0.5 0.5])
    image(x);
    axis off
    drawnow
    s=0;v=0;clr=0;nxt=0;prv=0;
 end
```

The ‘imread’ function opens the filename entered into the edit text string and stores it in ‘x’. The ‘subplot’ function is used before the ‘image’ function so the image doesn’t take up the entire figure window. There are many different ways to use ‘subplot.’ The method used here is to directly specify the position and size by using the ‘position’ property. This is similar to the ‘position’ property used in GUI definition code that makes figure windows. The format of the ‘position’ property matrix is the same:

[Distance to left border Distance to bottom border width height].

However, the units are in proportion of the figure window. Using the ‘View’ button’s code as an example, the width of the displayed image would be 0.5, or 50% of the current figure window.

Axes are turned off and a ‘drawnow’ forces the image to be displayed. “Callback’ variables are reset.

4.3.3 The ‘Clear’ Button

The ‘Clear’ button is even simpler than the ‘Clear’ button of Whisker2.

```
if(clr==1)
    cla;
    cnt=0;
    s=0;v=0;clr=0;nxt=0;prv=0;
 end
```

Axes are cleared, the ‘cnt’ counter variable is reset to 0, then ‘callback’ variables are reset.

4.3.4 The ‘>>’ and ‘>>’ Buttons

These buttons are identical after the ‘if’ statement. The ‘>>’ button will be used as an example. When the ‘>>’ button is pushed, the ‘cnt’ variable is incremented, and the following code is executed:

```
if(nxt==1)
    t=(files(cnt).name);
    t2=imread(t);
    set(numFile,'string',[num2str(t)]);
    subplot('position',[0 0.25 0.5 0.5])
    subimage(t2);
    axis off
    s=0;v=0;clr=0;nxt=0;
```

The first line uses the ‘cnt’ variable to specify the number of the filename as stored in the ‘files’ struct array defined at the beginning of the program. If ‘cnt’ happens to be set at 10 when the ‘>>’ button is pushed, the filename stored at files(10).name is saved into the variable ‘t’ and loaded using the ‘imread’ function. The current file display text field is updated using the ‘set’ command. The ‘subplot’ function specifies exactly where the following ‘subimage’ function will display the image store in ‘t2.’ ‘Subimage’ is used instead of image to ensure smoother function and the correct assignment of pictures to their respective windows.

4.3.5 Ending the Program

The ‘while’ loop has an ‘end’ statement that closes the loop.

C: Complete MATLAB Code

Whisker2.m

```
PrimPlot=figure('NumberTitle','off',...
'Name','Whisker 2',...
'position',[10 200 585 450]);
```

```
set(gcf,'doublebuffer','on')
scrsize=get(gcf,'position');
quitbutton=uicontrol('style','pushbutton',...
'string','Quit',...
'backgroundcolor',[1,0,0],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+480 ((scrsize(:,2))/35)+23 50 20],...
'callback','s=1;close;');
clearbutton=uicontrol('style','pushbutton',...
'string','Clear',...
'enable','off',...
'backgroundcolor',[.8,.8,.8],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+125 ((scrsize(:,2))/35)+23 50 20],...
'callback', 'clear=1');
plotbutton=uicontrol('style','pushbutton',...
'string','Load',...
'backgroundcolor',[1,1,0],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+75 ((scrsize(:,2))/35)+23 50 20],...
'callback', 'v=1;');
plotnewbutton=uicontrol('style','pushbutton',...
'string','Plot',...
'enable','off',...
'backgroundcolor',[.8,.8,.8],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+175 ((scrsize(:,2))/35)+23 50 20],...
'callback', 'pnew=1;');
graybutton=uicontrol('style','pushbutton',...
'string','Gray',...
'enable','off',...
'backgroundcolor',[.8,.8,.8],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+20 ((scrsize(:,2))/35)+123 50 20],...
'callback', 'gry=1;');
falsebutton=uicontrol('style','pushbutton',...
'string','False',...
'enable','off',...
'backgroundcolor',[.8,.8,.8],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+20 ((scrsize(:,2))/35)+143 50 20],...
'callback', 'fls=1;');
truebutton=uicontrol('style','pushbutton',...
'string','True',...
'enable','off',...
'backgroundcolor',[.8,.8,.8],...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+20 ((scrsize(:,2))/35)+103 50 20],...
'callback', 'tru=1;');
numx=uicontrol('style','text',...
'string','x',...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+75 ((scrsize(:,2))/35)+1 70 15]);
numy=uicontrol('style','text',...
'string','y',...
```

```
'fontsize',10,...
'position',[((scrsize(:,1))/20)+145 ((scrsize(:,2))/35)+1 70 15]);
numSize=uicontrol('style','text',...
'string','0',...
'fontsize',10,...
'position',[((scrsize(:,1))/20)+15 ((scrsize(:,2))/35)+1 25 15]);
set(gcf,'windowbuttondownfcn','mousedown=1;');
s=0;v=0;clear=0;mousedown=0;save=0;
pnew=0;gry=0;fls=0;tru=0;x1=[];y1=[];
while (s==0)
    if(v==1)
       s=0;v=0;clear=0;mousedown=0;mouseup=0;mousemotion=0;pnew=0;
       gry=0;fls=0;tru=0;
       set(truebutton,'enable','on');
       set(truebutton,'backgroundcolor',[.8,.8,.8]);
       set(graybutton,'enable','on');
       set(graybutton,'backgroundcolor',[.8,.8,.8]);
       set(falsebutton,'enable','on');
       set(falsebutton,'backgroundcolor',[.8,.8,.8]);
       set(clearbutton,'enable','on');
       set(clearbutton,'backgroundcolor',[1,1,1]);
       [filename,pathname]= uigetfile('*.jpg');
        try
            imread([pathname,filename], 'JPEG');
        catch
            errordlg('load did not work, press quit and restart')
        end
       x=imread([pathname,filename], 'JPEG');
       image(x)
       xC=double(x);
       xC=.2989*xC(:,:,1)+.5870*xC(:,:,2)+.1140*xC(:,:,3);
       set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
       hold on
    end
    if(gry==1)
       hold on
       imagesc(xC);
       colormap gray;
       plot(x1,y1,'b*');
       set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
       s=0;v=0;clear=0;mousedown=0; mouseup=0;
       mousemotion=0;pnew=0;gry=0;fls=0;tru=0;
    end
    if(fls==1)
       hold on
       imagesc(xC);
       colormap jet;
       plot(x1,y1,'r*');
       set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
       s=0;v=0;clear=0;mousedown=0; mouseup=0;
       mousemotion=0;;pnew=0;gry=0;fls=0;tru=0;
    end
    if(tru==1)
       hold on
       image(x);
```

```
    plot(x1,y1,'b*');
    set(gca,'xaxisloc','top','yaxisloc','right','fontsize',7);
    s=0;v=0;clear=0;mousedown=0; mouseup=0;
    mousemotion=0;;pnew=0;gry=0;fls=0;tru=0;
end
if(clear==1)
    cla;
    set(clearbutton,'enable','off');
    set(clearbutton,'backgroundcolor',[.8,.8,.8]);
    set(plotnewbutton,'enable','off');
    set(plotnewbutton,'backgroundcolor',[.8,.8,.8]);
    set(truebutton,'enable','off');
    set(truebutton,'backgroundcolor',[.8,.8,.8]);
    set(graybutton,'enable','off');
    set(graybutton,'backgroundcolor',[.8,.8,.8]);
    set(falsebutton,'enable','off');
    set(falsebutton,'backgroundcolor',[.8,.8,.8]);
    set(numx,'string','x');
    set(numy,'string','y');
    set(numSize,'string','0');
    x1=[];y1=[];
    s=0;v=0;clear=0;mousedown=0;pnew=0;gry=0;fls=0;tru=0;
end
if(mousedown==1 & pnew==0)
    set(plotnewbutton,'enable','on');
    set(plotnewbutton,'backgroundcolor',[0,1,0]);
    pt=get(gca,'currentpoint');
    set(numx,'string',[num2str(pt(1,1))]);
    set(numy,'string',[num2str(pt(1,2))]);
    x1=[x1;pt(1,1)];
    y1=[y1;pt(1,2)];
    xS=size(x1);
    xS=xS(:,1);
    set(numSize,'string',[num2str(xS)]);
    points=[x1 y1];
    hold on
    plot(x1,y1,'b*');
    drawnow
    s=0;v=0;clear=0;mousedown=0;gry=0;fls=0;tru=0;
end
if(pnew==1 & v==0)
    set(gcf,'doublebuffer','on')
    st=0;
    write=0;
    scrsize=get(gcf,'position');
    SecPlot=figure('NumberTitle','off',...
    'Units','pixels',...
    'position',...
    [(30*(scrsize(:,1))) ((scrsize(:,2))) ((scrsize(:,3))/2)
        ((scrsize(:,4))/2)]);
    x=imread([pathname,filename], 'JPEG');
    image(x);
    quitbutton2=uicontrol('style','pushbutton',...
    'string','Quit',...
    'backgroundcolor',[1,0,0],...
```

```
        'fontsize',10,...
        'position',[((scrsize(:,1))/20)+5 ((scrsize(:,2))/35)+1 50
20],...
        'callback','st=1;close;');
        writebutton=uicontrol('style','pushbutton',...
        'string','Write',...
        'backgroundcolor',[1,1,1],...
        'fontsize',10,...
        'position',[((scrsize(:,1))/20)+205 ((scrsize(:,2))/35)+1 50
20],...
        'callback','write=1;');
        hold on
        z=[];
        d=[];
        [points2,I]=sort(points);
        N=[size(x1)];
        N2=N(1,1);
         for j=1:N2,
             z=[z;(points2(j,1)),y1((I(j,1)));];
             x3=[z(:,1)];
             y3=[z(:,2)];
         end
         for i=1:(N2-1)
          d=[d;sqrt(((z((i+1),1)-z(i,1))^2)+((z((i+1),2)-z(i,2))^2))];
         end
        dWide=sqrt(((z(1,1)-z(N2,1))^2)+((z(1,2)-z(N2,2))^2));
        dSum=sum(d);
        ID=dSum/dWide;
        a1=[];
        a=[x1,y1];
        cvh2=(convhull(x1,y1));
        a1=a(cvh2,:);
        x2=a1(:,1);
        y2=a1(:,2);
        N3=size(x2);
        NCH=(N3(1,1))-1;
        fill(x2,y2,'r');
        hold on
        y6=plot(x3,y3,'w-');
        moveplot(y6,'axy');
        num2=uicontrol('style','text',...
        'string','',...
        'fontsize',10,...
        'position',[((scrsize(:,1))/20)+55 ((scrsize(:,2))/35)+1 150
20]);
        set(num2,'string',[filename '   ' (num2str(N2)) '   '
(num2str(NCH))]);
        set(gca,'xaxisloc','top',...
        'yaxisloc','right',...
        'XDir','normal',...
        'YDir','reverse',...
        'fontsize',8);
         axis off;
        while (st==0)
              if (write==1)
```

```
                wrt=[filename ' ' (num2str(N2)) ' ' (num2str(NCH)) ' '
(num2str(ID))];
                fid=fopen('WhiskerDat.txt','a+');
                fprintf(fid,'%s\n',wrt);
                fclose(fid);
                write=0;
              end
              drawnow
          end
          s=0;v=0;clear=0;mousedown=0;pnew=0;gry=0;fls=0;tru=0;
          drawnow;
          end
      drawnow
end
close
```

WhiskerView.m

```
PrimPlot=figure('NumberTitle','off','Name','WhiskerView','position',[
10 200 585 450]);
set(gcf,'doublebuffer','on')
scrsize=get(gcf,'position');
quitbutton=uicontrol('style','pushbutton',...
    'string','Quit',...
    'backgroundcolor',[.8,.8,.8],...
    'fontsize',10,...
    'position',[((scrsize(:,1))/20)+270 ((scrsize(:,2))/35)+21 50
20],...
    'callback','s=1;close;');
clearbutton=uicontrol('style','pushbutton',...
    'string','Clear',...
    'backgroundcolor',[.8,.8,.8],...
    'fontsize',10,...
    'position',[((scrsize(:,1))/20)+120 ((scrsize(:,2))/35)+21 50
20],...
    'callback', 'clr=1');
plotbutton=uicontrol('style','pushbutton',...
    'string','View',...
    'backgroundcolor',[.8,.8,.8],...
    'fontsize',10,...
    'position',[((scrsize(:,1))/20)+70 ((scrsize(:,2))/35)+21 50
20],...
    'callback', 'v=1;');
nextbutton=uicontrol('style','pushbutton',...
    'string','>>',...
    'backgroundcolor',[.8,.8,.8],...
    'fontsize',9, ...
    'position',[((scrsize(:,1))/20)+220 ((scrsize(:,2))/35)+21 50
20],...
    'callback', 'nxt=1;cnt=cnt+1');
prevbutton=uicontrol('style','pushbutton',...
    'string','<<',...
    'backgroundcolor',[.8,.8,.8],...
    'fontsize',9,...
    'position',[((scrsize(:,1))/20)+170 ((scrsize(:,2))/35)+21 50
20],...
```

```
    'callback', 'prv=1;cnt=cnt-1');
numFile=uicontrol('style','text',...
    'string','',...
    'fontsize',10,...
    'position',[((scrsize(:,1))/20)+110 ((scrsize(:,2))/35)+330 100
15]);
edittext=uicontrol('style','edit',...
    'string','ADULT000.jpg',...
    'fontsize',9,...
    'position',[((scrsize(:,1))/20)+70 ((scrsize(:,2))/35)+2 250
20]);
s=0;v=0;clr=0;nxt=0;prv=0;
files=dir('*.jpg');
cnt=1;
while (s==0)
 if(cnt>((length(files))))
    cnt=1;
 end
 if(cnt<1)
    cnt=(length(files));
 end
 if(v==1)
    x=imread(get(edittext, 'string'));
    subplot('position',[0.5 0.25 0.5 0.5])
    image(x);
    axis off
    drawnow
    s=0;v=0;clr=0;nxt=0;prv=0;
 end
 if(clr==1)
    cla;
    cnt=0;
    s=0;v=0;clr=0;nxt=0;prv=0;
 end
 if(nxt==1)
    t=(files(cnt).name);
    t2=imread(t);
    set(numFile,'string',[num2str(t)]);
    subplot('position',[0 0.25 0.5 0.5])
    subimage(t2);
    axis off
    s=0;v=0;clr=0;nxt=0;
 end
 if(prv==1)
    t=(files(cnt).name);
    t2=imread(t);
    set(numFile,'string',[num2str(t)]);
    subplot('position',[0 0.25 0.5 0.5])
    subimage(t2);
    axis off
    s=0;v=0;clr=0;prv=0;
 end
 drawnow
end
```

REFERENCES

Ackerl, K., Atzmueller, M. and Grammer, K. (2002) The scent of fear. *Neuroendocrinology Letters* **23,** 79-84.

Acosta, B. (2004) Nutria and Trapping: Interview and Personal Recollections, (ed. S. Finckbeiner), Garyville, LA.

Ajmat, M. T., Chamut, S. and Black-Decima, P. (1999) "Osmetrichia" in the grey brocket deer (Mazama gouazoubira). *Biocell* **23,** 171-176.

Anonymous. (2005) NUTRIA Myocastor coypus*, Vol. 2005*, Oregon Fish and Wildlife Department.

Arimura, G., Huber, D. P. W. and Bohlmann, J. (2004) Forest tent caterpillars (Malacosoma disstria) induce local and systemic diurnal emissions of terpenoid volatiles in hybrid poplar (Populus trichocarpa x deltoides): cDNA cloning, functional characterization, and patterns of gene expression of (-)-germacrene D synthase, PtdTPS1. *Plant Journal* **37,** 603-616.

Arkin, A., Saito, T. R., Takahashi, K., Amao, H., Aoki-Komori, S. and Takahashi, K. W. (2003) Age-related changes on marking, marking-like behavior and the scent gland in adult Mongolian gerbils (Meriones unguiculatus). *Experimental Animals* **52,** 17-24.

Arkin, A., Saito, T. R., Takahashi, K., Sugiyama, M., Aoki-Komori, S., Amao, H. and Takahashi, K. W. (1999) Observation of marking-like behavior, marking behavior and growth of the scent gland in young Mongolian gerbils (Meriones unguiculatus) of an inbred strain. *Experimental Animals* **48,** 269-276.

Atoji, Y., Yamamoto, Y. and Suzuki, Y. (1998) Apocrine sweat glands in the circumanal glands of the dog. *Anatomical Record* **252,** 403-412.

Attygalle, A. B. and Morgan, E. D. (1985) Ant Trail Pheromones. *Advances in Insect Physiology* **18,** 1-30.

Banack, S. A. and Grant, G. S. (2003) Reproduction and behaviour of the Samoan flying fox, Pteropus samoensis (Chiroptera, pteropodidae). *Mammalia* **67,** 419-437.

Barja, I., de Miguel, F. J. and Barcena, F. (2004) The importance of crossroads in faecal marking behaviour of the wolves (Canis lupus). *Naturwissenschaften* **91,** 489-492.

Barling, P. M. and Shirley, J. (1999) Diaphorase activity in sebaceous glands and related structures of the male red deer. *Comparative Biochemistry and Physiology B-Biochemistry & Molecular Biology* **123,** 17-21.

Barlow, J. and Taylor, B. L. (2005) Estimates of sperm whale abundance in the northeastern temperate Pacific from a combined acoustic and visual survey. *Marine Mammal Science* **21,** 429-445.

Beaucham.Gk. (1974) Perineal Scent Gland and Social Dominance in Male Guinea-Pig. *Physiology & Behavior* **13,** 669-673.

Beauchamp, G. K. and Yamazaki, K. (2003) Chemical signalling in mice. *Biochemical Society Transactions* **31,** 147-151.

Begg, C. M., Begg, K. S., Du Toit, J. T. and Mills, M. B. L. (2003) Scent-marking behaviour of the honey badger, Mellivora capensis (Mustelidae), in the southern Kalahari. *Animal Behaviour* **66,** 917-929.

Begg, C. M., Begg, K. S., Du Toit, J. T. and Mills, M. G. L. (2005) Spatial organization of the honey badger Mellivora capensis in the southern Kalahari: home-range size and movement patterns. *Journal of Zoology* **265,** 23-35.

Bel, M. C., Coulon, J., Sreng, L., Allaine, D., Bagneres, A. G. and Clement, J. L. (1999) Social signals involved in scent-marking behavior by cheek-rubbing in Alpine marmots (Marmota marmota). *Journal of Chemical Ecology* **25,** 2267-2283.

Ben-David, M., Blundell, G. M., Kern, J. W., Maier, J. A. K., Brown, E. D. and Jewett, S. C. (2005) Communication in river otters: Creation of variable resource sheds for terrestrial communities. *Ecology* **86,** 1331-1345.

Ben-David, M., Bowyer, R. T., Duffy, L. K., Roby, D. D. and Schell, D. M. (1998) Social behavior and ecosystem processes: River otter latrines and nutrient dynamics of terrestrial vegetation. *Ecology* **79,** 2567-2571.

Berthold, P., vandenBossche, W., Leshem, Y., Kaatz, C., Kaatz, M., Nowak, E. and Querner, U. (1997) Satellite-tracking of White Storks Ciconia ciconia: Migration of an eastern individual to South Yemen. *Journal Fur Ornithologie* **138,** 546-549.

Beynon, R. J. and Hurst, J. L. (2004) Urinary proteins and the modulation of chemical scents in mice and rats. *Peptides* **25,** 1553-1563.

Blair, R. M. and Langlinias, M. J. (1960) Nutria and swamp rabbits damage baldcypress plantings. *Journal of Forestry* **58,** 388-389.

Boisseau, O. (2005) Quantifying the acoustic repertoire of a population: The vocalizations of free-ranging bottlenose dolphins in Fiordland, New Zealand. *Journal of the Acoustical Society of America* **117,** 2318-2329.

Bounds, D. and Carowan, G. (2000) A nonnative nemesis. In *Transactions of the North American Wildlife and Natural Resources Conference, Vol. 65*, pp. 405-413.

Bowler, D. E. and Benton, T. G. (2005) Causes and consequences of animal dispersal strategies: relating individual behaviour to spatial dynamics. *Biological Reviews* **80,** 205-225.

Briscoe, B. K., Lewis, M. A. and Parrish, S. E. (2002) Home range formation in wolves due to scent marking. *Bulletin of Mathematical Biology* **64,** 261-284.

Buesching, C. D. and Macdonald, D. W. (2004) Variations in scent-marking behaviour of European badgers Meles meles in the vicinity of their setts. *Acta Theriologica* **49,** 235-246.

Burger, B. V., le Roux, M., Spies, H. S. C., Truter, V. and Bigalke, R. C. (1978) Mammalian pheromone studies - III. (e,e)-7,11,15-Trimethyl-3-methylenehexa-deca-1,6,10,14-tetraene, a new diterpene analogue of [beta]-farnesene from the dorsal gland of the springbok, Antidorcas marsupialis. *Tetrahedron Letters* **19,** 5221-5224.

Burtenshaw, J. C., Oleson, E. M., Hildebrand, J. A., McDonald, M. A., Andrew, R. K., Howe, B. M. and Mercer, J. A. (2004) Acoustic and satellite remote sensing of blue whale seasonality and habitat in the Northeast Pacific. *Deep-Sea Research Part Ii-Topical Studies in Oceanography* **51,** 967-986.

Candela, A. M. and Morrone, J. J. (2003) Biogeography of neotropical porcupines (Rodentia, Hystricognathi): Integrating recent and fossil data through a panbiogeographic approach. *Ameghiniana* **40,** 361-378.

Carter, J. and Leonard, B. P. (2002) A review of the literature on the worldwide distribution., spread of, and efforts to eradicate the coypu (Myocastor coypus). *Wildlife Society Bulletin* **30,** 162-175.

Chabreck, R. H. (1972) Vegetation, water and soil characteristics of the Louisiana coastal region. In *Bulletin No. 664*, pp. 72, Louisiana State University Experimental Station, Baton Rouge, LA, USA.

Charif, R. A., Ramey, R. R., Langbauer, W. R., Payne, K. B., Martin, R. B. and Brown, L. M. (2005) Spatial relationships and matrilineal kinship in African

savanna elephant (Loxodonta africana) clans. *Behavioral Ecology and Sociobiology* **57,** 327-338.

Close, D. A., Fitzpatrick, M. S., Lorion, C. M., Li, H. W. and Schreck, C. B. (2003) Effects of intraperitoneally implanted radio transmitters on the swimming performance and physiology of Pacific lamprey. *North American Journal of Fisheries Management* **23,** 1184-1192.

Collins, S. A., Gosling, L. M., Watkins, R. W. and Cowan, D. P. (2001) Artificially increasing scent mark rate increases urogenital gland size in mice Mus musculus. *Physiology & Behavior* **74,** 517-522.

Conner, L. C. and Toliver., J. A. (1987) The problem of planting Louisiana swamplands when nutria are present. In *Proceedings of the Eastern Wildlife Damage Conference, Vol. 3*, pp. 42-49.

Dagault, N. and Saboureau, M. (1990) Reproductive-Behavior of the Male Coypu (Myocastor-Coypus M) near the Poitou Marshlands. *Canadian Journal of Zoology-Revue Canadienne De Zoologie* **68,** 1584-1589.

Ding, B., Zhang, Y. P. and Ryder, O. A. (1998) Extraction, PCR amplification, and sequencing of mitochondrial DNA from scent mark and feces in the giant panda. *Zoo Biology* **17,** 499-504.

Dobson, F. S. and Jouventin, P. (2003) How mothers find their pups in a colony of Antarctic fur seals. *Behavioural Processes* **61,** 77-85.

Doncaster, C. P. and Micol, T. (1989) Annual Cycle of a Coypu (Myocastor-Coypus) Population - Male and Female Strategies. *Journal of Zoology* **217,** 227-240.

Drea, C. M., Vignieri, S. N., Kim, H. S., Weldele, M. L. and Glickman, S. E. (2002) Responses to olfactory stimuli in spotted hyenas (Crocuta crocuta): II. Discrimination of conspecific scent. *Journal of Comparative Psychology* **116,** 342-349.

Evans, J. (1970) About nutria and their control. *United States Bureau of Sport Fisheries and Wildlife, Resource Publication* **86,** 1-65.

Fadem, B. H. and Schwartz, R. A. (1986) A Sexually Dimorphic Suprasternal Scent Gland in Gray Short-Tailed Opossums (Monodelphis-Domestica). *Journal of Mammalogy* **67,** 205-208.

Fahrenthold, D. (2004) Blackwater Refuge Now Nutria-Free

Marsh-Demolishing Rodent Fended Off Earlier Efforts. In *Washington Post*, pp. B01, Washington, DC, USA.

Ferkin, M. H., Lee, D. N. and Leonard, S. T. (2004) The reproductive state of female voles affects their scent marking behavior and the responses of male conspecifics to such marks. *Ethology* **110,** 257-272.

Fossey, D. (1974) Observations on Home Range of One Group of Mountain Gorillas (Gorilla-Gorilla-Beringei). *Animal Behaviour* **22,** 568-581.

Fossey, D. (1982) Reproduction among Free-Living Mountain Gorillas. *American Journal of Primatology* **2,** 97-104.

Friday, N., Smith, T. D., Stevick, P. T. and Allen, J. (2000) Measurement of photographic quality and individual distinctiveness for the photographic identification of humpback whales, Megaptera novaeangliae. *Marine Mammal Science* **16,** 355-374.

Gavagnin, M., Mollo, E., Castelluccio, F. and Crispino, A. (2003) Sesquiterpene metabolites of the antarctic gorgonian Dasystenella acanthina. *Journal of Natural Products* **66,** 1517-1519.

Gomez, G. M. (1998) Furbearers, University of Texas Press, Austin, Texas.
Gorman, M. L., Kruuk, H. and Leitch, A. (1984) Social Functions of the Sub-Caudal Scent Gland Secretion of the European Badger Meles-Meles Meles-Meles (Carnivora, Mustelidae). *Journal of Zoology* **203,** 549-559.

Gosling, L. M. and Baker, S. J. (1989) The Eradication of Muskrats and Coypus from Britain. *Biological Journal of the Linnean Society* **38,** 39-51.

Gosling, L. M. and Roberts, S. C. (2001) Scent-marking by male mammals: Cheat-proof signals to competitors and mates. In *Advances in the Study of Behavior, Vol 30, Vol. 30*, pp. 169-217.

Gosling, L. M. and Wright, K. H. M. (1994) Scent Marking and Resource Defense by Male Coypus (Myocastor Coypus). *Journal of Zoology* **234,** 423-436.

Gudzic, B., Djokovic, D., Vajs, V., Palic, R. and Stojanovic, G. (2002) Composition and antimicrobial activity of the essential oil of Hypericum maculatum Crantz. *Flavour and Fragrance Journal* **17,** 392-394.

Guichon, M. L., Borgnia, M., Righi, C. F., Cassini, G. H. and Cassini, M. H. (2003) Social behavior and group formation in the coypu (Myocastor coypus) in the Argentinean pampas. *Journal of Mammalogy* **84,** 254-262.

Hake, M., Kjellen, N. and Alerstam, T. (2003) Age-dependent migration strategy in honey buzzards Pernis apivorus tracked by satellite. *Oikos* **103,** 385-396.

Henry, C., Poulle, M. L. and Roeder, J. J. (2005) Effect of sex and female reproductive status on seasonal home range size and stability in rural red foxes (Vulpes vulpes). *Ecoscience* **12,** 202-209.

Herrera, E. A. (1992) Size of Testes and Scent Glands in Capybaras, Hydrochaeris-Hydrochaeris (Rodentia, Caviomorpha). *Journal of Mammalogy* **73,** 871-875.

Herrera, E. A. and Macdonald, D. W. (1993) Aggression, Dominance, and Mating Success among Capybara Males (Hydrochaeris-Hydrochaeris). *Behavioral Ecology* **4,** 114-119.

Heth, G., Todrank, J. and Johnston, R. E. (1998) Kin recognition in golden hamsters: evidence for phenotype matching. *Animal Behaviour* **56,** 409-417.

Huchon, D. and Douzery, E. J. P. (2001) From the old world to the new world: A molecular chronicle of the phylogeny and biogeography of Hystricognath rodents. *Molecular Phylogenetics and Evolution* **20,** 238-251.

Hurst, J. L., Thom, M. D., Nevison, C. M., Humphries, R. E. and Beynon, R. J. (2005) MHC odours are not required or sufficient for recognition of individual scent owners. *Proceedings of the Royal Society B-Biological Sciences* **272,** 715-724.

Irwin, M. T., Samonds, K. E., Raharison, J. L. and Wright, P. C. (2004) Lemur latrines: Observations of latrine behavior in wild primates and possible ecological significance. *Journal of Mammalogy* **85,** 420-427.

Jemiolo, B., Xie, T.-M. and Novotny, M. (1991) Socio-sexual olfactory preference in female mice: Attractiveness of synthetic chemosignals. *Physiology & Behavior* **50,** 1119-1122.

Johnston, R. E. (1993) Memory for Individual Scent in Hamsters (Mesocricetus-Auratus) as Assessed by Habituation Methods. *Journal of Comparative Psychology* **107,** 201-207.

Johnston, R. E. (2003) Chemical communication in rodents: From pheromones to individual recognition. *Journal of Mammalogy* **84,** 1141-1162.

Johnston, R. E., Munver, R. and Tung, C. (1995) Scent Counter Marks - Selective Memory for the Top Scent by Golden-Hamsters. *Animal Behaviour* **49,** 1435-1442.

Johnston, R. E., Sorokin, E. S. and Ferkin, M. H. (1997) Scent counter-marking by male meadow voles: Females prefer the top-scent male. *Ethology* **103,** 443-453.

Jones, T. R. and Plakke, R. K. (1981) The Histology and Histochemistry of the Perianal Scent Gland of the Reproductively Quiescent Black-Tailed Prairie Dog (Cynomys-Ludovicianus). *Journal of Mammalogy* **62,** 362-368.

Kehtarnavaz, N., Peddigari, V., Chandan, C., Syed, W., Hillman, G. and Wursig, B. (2003) Photo-identification of humpback and gray whales using affine moment invariants. In *Image Analysis, Proceedings, Vol. 2749*, pp. 109-116.

Kelly, M. J. (2001) Computer-aided photograph matching in studies using individual identification: An example from Serengeti cheetahs. *Journal of Mammalogy* **82,** 440-449.

Kendrot, S. (2003) Challenges faced by maryland nutria project.

Kinler, N., Linscombe, G. and Hartley, S. (1999) A survey of nutria herbivory damage in coastal Louisiana in 1999, Fur and Refuge Division, Louisiana Department of Wildlife and Fisheries., New Iberia, LA, USA.

Lai, W. S. and Johnston, R. E. (2002) Individual recognition after fighting by golden hamsters: A new method. *Physiology & Behavior* **76,** 225-239.

Lawson, R. E., Putman, R. J. and Fielding, A. H. (2000) Individual signatures in scent gland secretions of Eurasian deer. *Journal of Zoology* **251,** 399-410.

LeBlanc, D. J. (1994) Nutria. In *Prevention and Control of Wildlife Damage*, (eds. S. E. Hygnstrom, R. M. Timm and G. E. Larsen), pp. B71-B80, Nebraska Cooperative Extension Service, University of Nebraska-Lincoln., Lincoln, Nebraska.

Linscombe, G. (1992) Nutria population, habitat use and damages. In *Proc. Nutria and Muskrat Management Symposium*, pp. 27-33, Louisiana Cooperative Extension Service, Louisiana State University Agricultural Center.

Linscombe, G. (2000) 1999-00 annual report: fur and alligator advisory council, pp. 35, Louisiana Department of Wildlife and Fisheries, New Iberia, Louisiana.

Lowery, G. H. (1974) The mammals of Louisiana and its adjacent waters, Louisiana State University Press, Baton Rouge.

Makinen, T. S., Niemela, E., Moen, K. and Lindstrom, R. (2000) Behaviour of gill-net and rod-captured Atlantic salmon (Salmo salar L.) during upstream migration and following radio tagging. *Fisheries Research* **45,** 117-127.

Manaf, P., de Brito-Gitirana, L. and Oliveira, E. S. (2003) Evidence of chemical communication in the spiny rat Trinomys yonenagae (Echimyidae): anal scent gland and social interactions. *Canadian Journal of Zoology-Revue Canadienne De Zoologie* **81,** 1138-1143.

Marx, J., Mouton, E. and Linscombe, G. (2003) NUTRIA HARVEST DISTRIBUTION 2002-2003 And A SURVEY OF NUTRIA HERBIVORY DAMAGE IN COASTAL LOUISIANA IN 2003, pp. 44, Fur and Refuge Division, Louisiana Department of Wildlife and Fisheries, New Iberia, LA, USA.

Mateo, J. M. (2003) Kin recognition in ground squirrels and other rodents. *Journal of Mammalogy* **84,** 1163-1181.

Mayeaux, D. J. and Johnston, R. E. (2002) Discrimination of individual odours by hamsters (Mesocricetus auratus) varies with the location of those odours. *Animal Behaviour* **64,** 269-281.

Miller, K. E., Laszlo, K. and Dietz, J. M. (2003) The role of scent marking in the social communication of wild golden lion tamarins, Leontopithecus rosalia. *Animal Behaviour* **65,** 795-803.

Miller, K. V., Jemiolo, B., Gassett, J. W., Jelinek, I., Wiesler, D. and Novotny, M. (1998) Putative chemical signals from white-tailed deer (Odocoileus virginianus): Social and seasonal effects on urinary volatile excretion in males. *Journal of Chemical Ecology* **24,** 673-683.

Mouton, E., Linscombe, G. and Hartley, S. (2001) A survey of nutria herbivory damage in coastal Louisiana in 2001., pp. 19, Fur and Refuge Division, Louisiana Departmetn of Wildlife and Fisheries, New Iberia, LA, USA.

Myers, R. S., Shaffer, G. P. and Llewellyn, D. W. (1995) Baldcypress (Taxodium distichum L rich) restoration in Southeast Louisiana-the relative effects of herbivory, flooding, competition, and macronutrients. *Wetlands* **15,** 141-148.

Nevison, C. M., Armstrong, S., Beynon, R. J., Humphries, R. E. and Hurst, J. L. (2003) The ownership signature in mouse scent marks is involatile.

Proceedings of the Royal Society of London Series B-Biological Sciences **270,** 1957-1963.

Payne, K. B., Thompson, M. and Kramer, L. (2003) Elephant calling patterns as indicators of group size and composition: the basis for an acoustic monitoring system. *African Journal of Ecology* **41,** 99-107.

Perret, N. and Joly, P. (2002) Impacts of tattooing and pit-tagging on survival and fecundity in the alpine newt (Triturus alpestris). *Herpetologica* **58,** 131-138.

Porcher, I. F. (2005) On the gestation period of the blackfin reef shark, Carcharhinus melanopterus, in waters off Moorea, French Polynesia. *Marine Biology* **146,** 1207-1211.

Quintana, A., Reinhard, J., Faure, R., Uva, P., Bagneres, A. G., Massiot, G. and Clement, J. L. (2003) Interspecific variation in terpenoid composition of defensive secretions of European Reticulitermes termites. *Journal of Chemical Ecology* **29,** 639-652.

Regehr, H. M. and Rodway, M. S. (2003) Evaluation of nasal discs and colored leg bands as markers for Harlequin Ducks. *Journal of Field Ornithology* **74,** 129-135.

Reggiani, G., Boitani, L. and Destefano, R. (1995) Population-Dynamics and Regulation in the Coypu Myocastor Coypus in Central Italy. *Ecography* **18,** 138-146.

Revilla, E. and Palomares, F. (2002) Spatial organization, group living and ecological correlates in low-density populations of Eurasian badgers, Meles meles. *Journal of Animal Ecology* **71,** 497-512.

Roberts, S. C., Gosling, L. M., Thornton, E. A. and McClung, J. (2001) Scent-marking by male mice under the risk of predation. *Behavioral Ecology* **12,** 698-705.

Roper, T. J., Conradt, L., Butler, J., Christian, S. E., Ostler, J. and Schmid, T. K. (1993) Territorial Marking with Feces in Badgers (Meles-Meles) - a Comparison of Boundary and Hinterland Latrine Use. *Behaviour* **127,** 289-307.

Rosell, F. and Bjorkoyli, T. (2002) A test of the dear enemy phenomenon in the Eurasian beaver. *Animal Behaviour* **63,** 1073-1078.

Rosell, F. and Schulte, B. A. (2004) Sexual dimorphism in the development of scent structures for the obligate monogamous Eurasian beaver (Castor fiber). *Journal of Mammalogy* **85,** 1138-1144.

Rosell, F. and Sundsdal, L. J. (2001) Odorant source used in Eurasian beaver territory marking. *Journal of Chemical Ecology* **27,** 2471-2491.

Rostain, R. R., Ben-David, M., Groves, P. and Randall, J. A. (2004) Why do river otters scent-mark? An experimental test of several hypotheses. *Animal Behaviour* **68,** 703-711.

Russell, D. J. and McDougall, A. J. (2005) Movement and juvenile recruitment of mangrove jack, Lutjanus argentimaculatus (Forsskal), in northern Australia. *Marine and Freshwater Research* **56,** 465-475.

Safi, K. and Kerth, G. (2003) Secretions of the interaural gland contain information about individuality and colony membership in the Bechstein's bat. *Animal Behaviour* **65,** 363-369.

Sapolsky, R. M. (1982) The Endocrine Stress-Response and Social-Status in the Wild Baboon. *Hormones and Behavior* **16,** 279-292.

Schultz, R. N. and Wilson, P. C. (2002) Territorial marking by lone male Gray Wolves, Canis lupus. *Canadian Field-Naturalist* **116,** 311-313.

Schulz, S., Kruckert, K. and Weldon, P. J. (2003) New terpene hydrocarbons from the alligatoridae (Crocodylia, reptilia). *Journal of Natural Products* **66,** 34-38.

Scully, W. M. R., Fenton, M. B. and Saleuddin, A. S. M. (2000) A histological examination of the holding sacs and glandular scent organs of some bat species (Emballonuridae, Hipposideridae, Phyllostomidae, Vespertilionidae, and Molossidae). *Canadian Journal of Zoology-Revue Canadienne De Zoologie* **78,** 613-623.

Scutareanu, P., Bruin, J., Posthumus, M. A. and Drukker, B. (2003) Constitutive and herbivore-induced volatiles in pear, alder and hawthorn trees. *Chemoecology* **13,** 63-74.

Sillero-Zubiri, C. and Macdonald, D. W. (1998) Scent-marking and territorial behaviour of Ethiopian wolves Canis simensis. *Journal of Zoology* **245,** 351-361.

Simpson, T. R. and Swank., W. G. (1979) Trap avoidance by marked nutria; a problem in population estimation. In *Proceedings of the Annual Conference*

Southeast Association Fish and Wildlife Agencies, Vol. 33, pp. 11-14.

Smith, J. D. and Hearn, G. W. (1979) Ultrastructure of the Apocrine-Sebaceous Anal Scent Gland of the Woodchuck, Marmota-Monax - Evidence for Apocrine and Merocrine Secretion by a Single Cell Type. *Anatomical Record* **193,** 269-&.

Smith, T. D., Allen, J., Clapham, P. J., Hammond, P. S., Katona, S., Larsen, F., Lien, J., Mattila, D., Palsboll, P. J., Sigurjonsson, J., Stevick, P. T. and Oien, N. (1999) An ocean-basin-wide mark-recapture study of the North Atlantic humpback whale (Megaptera novaeangliae). *Marine Mammal Science* **15,** 1-32.

Stanford, C. B., Wallis, J., Mpongo, E. and Goodall, J. (1994) Hunting Decisions in Wild Chimpanzees. *Behaviour* **131,** 1-18.

Sun, L. X. and MullerSchwarze, D. (1997) Sibling recognition in the beaver: a field test for phenotype matching. *Animal Behaviour* **54,** 493-502.

Sun, L. X. and Muller-Schwarze, D. (1998) Anal gland secretion codes for family membership in the beaver. *Behavioral Ecology and Sociobiology* **44,** 199-208.

Swenson, J. E., Wallin, K., Ericsson, G., Cederlund, G. and Sandegren, F. (1999) Effects of ear-tagging with radiotransmitters on survival of moose calves. *Journal of Wildlife Management* **63,** 354-358.

Taber, A. B. and Macdonald, D. W. (1992) Spatial-Organization and Monogamy in the Mara Dolichotis-Patagonum. *Journal of Zoology* **227,** 417-438.

Takami, S. (2002) Recent progress in the neurobiology of the vomeronasal organ. *Microscopy Research and Technique* **58,** 228-250.

Tang-Martinez, Z. (2003) Emerging themes and future challenges: Forgotten rodents, neglected questions. *Journal of Mammalogy* **84,** 1212-1227.

Tarver, J. G., Linscombe, G. and Kinler, N. (1987) Fur animals, alligators and the fur industry in Louisiana, Louisiana Wildlife and Fisheries Commission, Baton Rouge, LA, USA.

Thom, M. D. and Hurst, J. L. (2004) Individual recognition by scent. *Annales Zoologici Fennici* **41,** 765-787.

Todrank, J., Heth, G. and Johnston, R. E. (1998) Kin recognition in golden hamsters: evidence for kinship odours. *Animal Behaviour* **55,** 377-386.

Tuyttens, F. A. M., Macdonald, D. W. and Roddam, A. W. (2002) Effects of radio-collars on European badgers (Meles meles). *Journal of Zoology* **257,** 37-42.

Valentine, J. M., Walther, J. R., McCartney, K. M. and Ivy, L. M. (1972) Alligator diets on the Sabine National Wildlife Refuge, Louisiana. *Journal of Wildlife Management* **36,** 809-815.

Vuorinen, T., Reddy, G. V. P., Nerg, A.-M. and Holopainen, J. K. (2004) Monoterpene and herbivore-induced emissions from cabbage plants grown at elevated atmospheric CO2 concentration. *Atmospheric Environment* **38,** 675-682.

Wacher, T. and Attum, O. (2005) Preliminary investigation into the presence and distribution of small carnivores in the Empty Quarter of Saudi Arabia through the use of a camera trap. *Mammalia* **69,** 81-84.

Watkins, W. A., Daher, M. A., George, J. E. and Rodriguez, D. (2004) Twelve years of tracking 52-Hz whale calls from a unique source in the North Pacific. *Deep-Sea Research Part I-Oceanographic Research Papers* **51,** 1889-1901.

Wegge, P., Pokheral, C. P. and Jnawali, S. R. (2004) Effects of trapping effort and trap shyness on estimates of tiger abundance from camera trap studies. *Animal Conservation* **7,** 251-256.

Wellington, J. L., Byrne, K. J., Preti, G., Beauchamp, G. K. and Smith, A. B. (1979) Perineal Scent Gland of Wild and Domestic Guinea-Pigs - Comparative Chemical and Behavioral-Study. *Journal of Chemical Ecology* **5,** 737-751.

Welsch, U., van Dyk, G., Moss, D. and Feuerhake, F. (1998) Cutaneous glands of male and female impalas (Aepyceros melampus): seasonal activity changes and secretory mechanisms. *Cell and Tissue Research* **292,** 377-394.

Whiten, A., Goodall, J., McGrew, W. C., Nishida, T., Reynolds, V., Sugiyama, Y., Tutin, C. E. G., Wrangham, R. W. and Boesch, C. (1999) Cultures in chimpanzees. *Nature* **399,** 682-685.

Wilcox, R. M. and Johnston, R. E. (1995) Scent Counter-Marks - Specialized Mechanisms of Perception and Response to Individual Odors in Golden-Hamsters (Mesocricetus-Auratus). *Journal of Comparative Psychology* **109,** 349-356.

Wirant, S. C. and McGuire, B. (2004) Urinary behavior of female domestic dogs (Canis familiaris): influence of reproductive status, location, and age. *Applied Animal Behaviour Science* **85,** 335-348.

Woodley, S. K. and Baum, M. J. (2004) Differential activation of glomeruli in the ferret's main olfactory bulb by anal scent gland odours from males and females: an early step in mate identification. *European Journal of Neuroscience* **20,** 1025-1032.

Woods, C. A., Contreras, L., Willner-Chapman, G. and Whidden, H. P. (1992) Myocastor coypus. *Mammalian Species* **398,** 1-8.

Xiangyu, J. G., Zhang, F., Fang, Y. L., Kan, W., Zhang, G. X. and Zhang, Z. N. (2002) Behavioural response of aphids to the alarm pheromone component (E)-beta-farnesene in the field. *Physiological Entomology* **27,** 307-311.

Young, J. G. and Henke, S. E. (1999) Effect of domestic rabbit urine on trap response in cottontail rabbits. *Wildlife Society Bulletin* **27,** 306-309.

Zala, S. M., Potts, W. K. and Penn, D. J. (2004) Scent-marking displays provide honest signals of health and infection. *Behavioral Ecology* **15,** 338-344.

Zeller, U., Epple, G., Kuderling, I. and Kuhn, H. J. (1988) The Anatomy of the Circumgenital Scent Gland of Saguinus-Fuscicollis (Callitrichidae, Primates). *Journal of Zoology* **214,** 141-156.

Zhang, J. X., Ni, J., Ren, X. J., Sun, L. X., Zhang, Z. B. and Wang, Z. W. (2003) Possible coding for recognition of sexes, individuals and species in anal gland volatiles of Mustela eversmanni and M-Sibirica. *Chemical Senses* **28,** 381-388.

Zhang, J. X., Sun, L. X., Zhang, Z. B., Wang, Z. W., Chen, Y. and Wang, R. (2002) Volatile compounds in anal gland of Siberian weasels (Mustela sibirica) and steppe polecats (M-eversmanni). *Journal of Chemical Ecology* **28,** 1287-1297.

Zub, K., Theuerkauf, J., Jedrzejewski, W., Jedrzejewska, B., Schmidt, K. and Kowalczyk, R. (2003) Wolf pack territory marking in the Bialowieza primeval forest (Poland). *Behaviour* **140,** 635-648.

Zuri, I., Su, W. and Halpern, M. (2003) Conspecific odor investigation by gray short-tailed opossums (Monodelphis domestica). *Physiology & Behavior* **80,** 225-232.