Learning Intent From
Tone of Voice

by

Jennie E. Cochran

Submitted to the
Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the
degree of
Master of Engineering in Electrical Engineering and Computer
Science
at the Massachusetts Institute of Technology
May 19th, 2004

Copyright 2004 Jennie E Cochran. All rights reserved.

The author hereby grants to M.I.T. permission to reproduce and
distribute publicly paper and electronic copies of this thesis
and to grant others the right to do so.

Author __________________________________________

Department of Electrical Engineering and Computer Science

May 17th, 2004

Certified by _______________________________________

Bruce Blumberg

Thesis Supervisor

Accepted by __________________

Arthur C. Smith

Chairman, Department Committee on Graduate Theses
Learning Intent From
Tone of Voice

by

Jennie E. Cochran

Submitted to the
Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the
degree of
Master of Engineering in Electrical Engineering and Computer
Science
at the Massachusetts Institute of Technology
May 19th, 2004

Abstract

The ability to discern information from the tone of voice that a person uses is an
important part of social interactions. Synthetic characters that can interact
naturally with humans could take advantage of this information if they could
discern it. I propose that a synthetic character with a vocalization affect classifier
and the ability to learn associations can use the tone of voice of the person
interacting with her to predict what the person is going to do. In this approach
the classifier learns to distinguish tones in real time allowing the character to
adapt to new tones. I describe the implementation of the system, called Minimus
T.O. Mouse, and its extensions from previous affect classifying systems and
previous synthetic characters.

Thesis Supervisor: Bruce Blumberg
Title: Associate Professor
Acknowledgements

I would like to thank my family, my roommates and my advisor for their support throughout this year.

My parents have always been supportive, especially during my undergraduate years and they never stopped this year. They called just to check up on me and say hi, which, though a small thing, means everything. My siblings have always been there to listen to me babble about things I think are exciting and anything I wanted to talk about. Most importantly, my parents and siblings have always believed in me, knowing I will get everything done even if I don’t know it yet.

My roommates kept me sane this year. Having Stephanie and Miquela to talk to and understand what I’m saying because they’re going through the same things was more than I could ever ask for. We got pissed off together and celebrated together. It was a hard year that in the end was worth it because of their friendship.

Lastly, I would like to thank my advisor, Bruce, for taking me on again this year and letting me find my niche in the group. He allowed me freedom to find what I was interested in, but also made sure the work I did was relevant and worthy of note.

Thank you everyone!!
Table of Contents

1 Introduction ............................................................................................................. 7
  1.1 Synthetic creature learn tone of voice ............................................................. 7
  1.2 Map .................................................................................................................. 11

2 Background and Underlying Architecture ......................................................... 12
  2.1 Synthetic characters ....................................................................................... 12
  2.2 Framework ....................................................................................................... 14
    2.2.1 Overview .................................................................................................... 14
    2.2.2 Flow of operation ..................................................................................... 14
    2.2.3 Sensory system ........................................................................................ 16
    2.2.4 Perception system .................................................................................... 18
    2.2.5 Belief system ............................................................................................ 21
    2.2.6 Action system .......................................................................................... 23
    2.2.7 Motor system ............................................................................................ 26
  2.3 Affect in voice .................................................................................................. 27
    2.3.1 Physical affect of emotions .................................................................... 27
    2.3.2 Automatic classification of affect in vocalizations ................................. 28
  2.4 Association learning ......................................................................................... 29
  2.5 Dobie T. Coyote ............................................................................................... 31
  2.6 Maximillion T. Mouse ...................................................................................... 33

3 Minimus T.O. Mouse ............................................................................................. 36
  3.1 Meet Minni ....................................................................................................... 36
  3.2 Minni is a synthetic character ......................................................................... 36
  3.3 Sensory system ............................................................................................... 37
3.4 Perception system .......................................................... 39
3.5 Action system ............................................................... 43
3.6 Emotion ........................................................................ 46
3.7 How to interact with Minni ............................................ 47

4 Conclusions ....................................................................... 48
  4.1 Evaluation ...................................................................... 48
  4.2 Lessons ......................................................................... 51
  4.3 Contributions .............................................................. 52

5 Future work ....................................................................... 54

References ........................................................................ 56
List of figures

2.1 Flow of World Update..................................................15
2.2 Sensory System Example...............................................18
2.3 Percept Tree Example..................................................21
2.4 Belief System Example..................................................23
2.5 Action System Example................................................26
2.6 Motor System Example................................................27
2.7 Innovational Action Tuples.......................................32
2.8 Dobie T. Coyote.......................................................33
2.9 Maximillian T. Mouse..................................................35
3.1 Minni’s Percept Tree..................................................40
3.2 Pitch Contours and Polynomial Fits................................41
Chapter 1

Introduction

1.1 Synthetic creatures learn tone of voice

Machines are used almost everywhere in daily life now, from waking up to an alarm clock, to driving to work, to emailing colleagues, to using a cellphone to call family, to microwaving a quick dinner, and to watching TV before going to bed. Day after day further uses for computers and machines are found. As machines are used more and more, we need new intuitive interfaces to interact with them. The more naturally a human can interact with machines, the more useful machines will be: as social interactions are integral parts of human society. Structured civilized society sets humans apart from other animals. For machines to be fully integrated into human society, they need the ability to interact socially with each other and with humans. This would make using any machine easy, intuitive, and natural.

Designing synthetic characters is one approach to developing social machines. Synthetic characters provide an interface that allows humans to interact naturally with computers. Instead of embodying an abstract idea of a ‘social machine’ that manages to understand and be understood by humans, they embody a concrete idea of a virtual creature that has an animated body which moves and displays familiar actions and emotions. Synthetic characters have personalities which allows people to anthropomorphize them. These powerful traits allow people to interact with synthetic creatures as they would with beings with whom they are familiar, such as dogs or other people. One guiding goal of designing synthetic characters is for these creatures to understand all aspects of social interactions, including body movement and stance, facial expressions and vocalizations (including speaking and other emotional vocalizations like laughing). The dual goal is for these creatures to in turn have the capacity to use
their bodies and voices to fully express themselves in ways humans understand thus completing both sides of social interaction.

Maximillian T Mouse was developed by the Synthetic Characters group in the fall of 2003. Max is a synthetic character with the body of a mouse and personality similar to a small child. He is continually hungry, and as he is very picky, he always wants cheese and only cheese. A person interacts with Max as she would perhaps with a small child. She can wave a fake cheese around his field of view, bring the cheese close and then give it to him or yank it away from him, thereby teasing him. Max reacts as most children do; obtaining the cheese makes him happy, while he becomes confused, frustrated and eventually angry when he does not receive the cheese. In addition, receiving the cheese after not receiving it surprises him. Max also has other abilities and traits similar to children's. He can see people moving in the area around him and knowing they are there, he then often begs them to give him the cheese. Max does however lack one important social capability, he has no auditory awareness. He cannot hear a person talking to him, therefore he has no idea whether the person is teasing him or soothing him with their voice.

The lack of auditory awareness in Max led to the development of Minimus T.O. Mouse. Minni was designed to have the same primary drive as Max; she wants the cheese, teasing angers her and eating delights her. The main difference between the two creatures however is that Minni can hear people and use their vocalizations to learn about them. She tries to learn, based on what a person does and what sounds he makes, what his different vocalizations indicate about what he will do, i.e. if he uses a certain tone of voice will he give her the cheese or not. Unlike other systems that recognize tone of voice, Minni learns to differentiate and recognize the tone in real time. Learning in real time allows her to adapt to different people using tones in different ways. Being able to adapt in a social setting is an important capability for a social creature as social interactions often change. Humans adapt very well, and I aimed to approximate this capability with Minni.

In order to participate in any social situation, Minni had to have an interface for use between her and humans. The interface used to interact with
her combines both natural and unnatural elements. A person uses a video game pad to control and move the cheese around Minni’s world, which is not quite natural. However, he can speak to her as he would another human. The social aspect of speaking to another person is not lost in her interface as it is when using many other vocal interfaces. Most vocal interfaces focus on the quantitative meaning of the words said. Many systems look for and classify phonemes to identify words to combine into sentences to discern meaning. Other systems try to discern the meaning from command style interfaces. For example a previous creature designed by the Synthetic Characters, Dobie T. Coyote a trainable dog, learns to do actions when a person speaks a command. However, none of these systems use the affect in the vocalization to discern meaning. The tone of voice plays a large role in the overall meaning a person wishes to convey. The systems described lose that extra information which expresses intent and emotional state of the speaker much better than words alone do. Minni does not lose the information contained in the tone of voice; in fact, she focuses on it.

While there are several other interfaces that could convey the same information conveyed in speaking, a natural interface is optimal. Information contained in the content of a vocalization can be easily communicated in other ways, such as typing a message on a keyboard. However, the information contained in the tone of voice during the delivery of a vocalization cannot be easily expressed by other methods. This difficulty indicates that a natural interface would be the most efficient and the most easily understood by humans. Thus I designed an interface in which a person can speak to Minni as he would another person.

Current research on vocalization affect focuses on the emotional state of the speaker and how this affects the tone of the vocalization. Research on automatic classification of vocalization affect has consisted of taking vocal data, labeling each sample with the assumed emotional state of the speaker and training a classifier off-line. For instance, some systems were trained by asking a trained actor to say certain phrases with certain emotional states. These recordings were then labeled with the emotional state, such as angry, sad or happy and then passed into the trainer. A system developed this way has no
means to adapt to changes in how the speaker’s emotional state is expressed. All interactors must use the same tone of voice to convey the same emotion. Humans interactions are constantly changing and the people involved are constantly adapting their beliefs about what the affect in other people’s vocalizations mean. For machines to successfully interact with humans, they must also be able to adapt to changes in speaker affect. Thus we wanted Minni to learn in real time what the speakers intent is, based on associations between tone of voice and previous actions that occurred.

For Minni to learn intent, she not only has to be able to classify affect in vocalizations, she also has to be able to reason about her actions, her classifications and their consequences. Dobie T. Coyote, a trainable dog mentioned previously, was designed to reason about his actions and the commands that people spoke to him. The ultimate design of Minni’s learning system was inspired from Dobie’s behavior system, though it has several key differences that make it more appropriate for the ultimate goal of the system.

I proposed that correlating actions and features of vocalizations makes it possible for a virtual creature to adaptively learn about affect in speech. I developed Minni using this idea. In order to build the new system that learns affect classification online, I used the features of speech that previous systems that automatically classify affect in speech found useful. This new system has two important features previous systems have not had. First, as mentioned, the learning occurs on-line instead of off. Previous systems have recorded data, trained a system off-line and then integrated the system into a bigger system. This system learns classification on-line, using only the data it receives during use. Second, this system labels data based on an intent perceived by the system, not based on an affect specified to or by the speaker. Previous systems have recorded data and labeled it by explicitly asking the speaker to add an affect to his speech or by having people specify the affect they believe is contained within the speech. This system labels the data based on actions the speaker does, eliminating the need for direct access to the speakers thoughts, and mimicking the way humans learn about tone of voice. I also built on systems that learn to make and use associations. Dobie T. Coyote is a synthetic character designed as a
trainable dog. He also learns to associate vocalizations with actions. He learns to associate words with actions, thus enabling the user to train him to perform a specific action on command. Minni’s learning system goes further. While Dobie only learned when he should do an action, Minni learns both when she should and should not do an action. In addition, while Dobie did not use feedback to improve his classification of spoken commands, Minni makes use of this additional information.

1.2 Map

In this paper, I will elaborate the ideas presented above. First, I will further explain what synthetic characters are and how they are designed. I will describe the framework that was used when designing and building Minimus T.O. Mouse. I will then discuss the specifics of the system that make up Minni, explaining differences to existing systems and pointing out where further work can occur. After the Minimus system has been explained, I will discuss how successfully the system met the goals. Finally I will draw conclusions and reiterate further research that can occur.
Chapter 2

Background and Underlying Architecture

2.1 Synthetic characters

Minni was created specifically as a synthetic character because these characters provide an excellent framework for creating systems that adaptively learn. Because of this choice, I shall first explain the ideas underlying synthetic characters. Synthetic characters are virtual creatures that 1) have underlying behavior mechanisms similar to real animal behavioral mechanisms and 2) interact with humans in some way which expresses a personality. Designing entities which embody recognizable creatures is a useful way to implement socially competent machines, as people are more inclined to be able to interact with and understand them without instruction. No explanation is needed as synthetic characters express themselves in easily understandable ways. They have tails, ears and bodies similar to actual animals which they also use and move in similar ways. For instance, Dobie T. Coyote has a dog’s body and wags his tail when he is happy. Max has a mouse body with ears that indicate his happiness level. The wolves in AlphaWolf walk slowly when scared and quickly when excited. (Dobie, Max and the AlphaWolf wolves are all synthetic characters designed using these ideas.)

We design these creatures using ideas from the current theories of animal behavior. This approach has both pros and cons. Using such theories takes advantage of a framework for learning that is based on learning systems (i.e. animals) that have been honed for millions of years. Animals are superb learners as they must learn about their environments in order to survive. Behavior research aims to understand the mechanisms that aid learning. We use this research with the belief that it can be used as an effective basis for developing other systems that learn. Dobie T Coyote is a recent example of this idea. (Blumberg et al (2002) describes the underlying systems in Dobie in detail.)
Dobie T Coyote, as stated above, is a synthetic character designed as a trainable dog. The user watches Dobie onscreen while he speaks commands into a microphone and rewards or punishes Dobie for doing or not doing what the user desires. Dobie’s learning system is based on clicker dog training. In this style of training, dogs are taught to associate a click with food. The dog can then be rewarded with a click during training and rewarded with food later. Teaching the dog using a clicker allows the trainer to pinpoint the exact action of the dogs that was good, instead of introducing ambiguities that arise when the dog must wait for food. When the trainer has chosen an action he wishes to train the dog to do on command, he first clicks whenever the dog does the action. The dog learns to associate the action with ‘goodness’ and does it more often. The trainer then starts speaking the command and clicking only when the dog does the action and the trainer has spoken the command. The dog learns to distinguish from doing the action without the command and doing the action with the command. After learning this, the trainer can then say the command to trigger the dog into performing the action. Dobie’s learning system was designed on these principles. A click is hardwired into his system as a good thing. Actions have values based on what kind of reward he believes he will receive for doing them. When the user clicks while Dobie is performing an action, the actions value increases. Thus if the user clicks every time Dobie sits, Dobie will learn sitting is good and do it more often, similar to a real dog. The user can then start saying a command when Dobie sits. Dobie learns to associate the command and sitting with goodness, which allows him to associate the command with sitting. After learning this, Dobie will sit when he hears the command, as he perceives this as a good thing that will lead to the user rewarding him.

Implementing creatures based on current theories also exposes gaps in said theories. Filling these gaps points out new directions for researching the underlying mechanisms behind behavior. However, these gaps also present challenges. The design of a creature cannot stop merely because the theory does not specify how every mechanism works. Thus the designer must fill the gap while trying to retain biological plausibility. Another challenge associated with implementing creatures in such a way include implementing interfaces which are
natural for humans to understand. Much more research is needed before a machine can ‘see’, ‘hear’ and do other information gathering as well as a human or even a simple animal can. However, even with these challenges, the ideas behind the design of synthetic characters are useful for developing social machines.

2.2 Framework

2.2.1 Overview

The Synthetic Characters group at the MIT Media Lab developed the architecture I used during the development of Minimus T.O. Mouse. This code base provides a foundation to build systems that learn. It specifically aids the development of virtual creatures whose learning mechanisms are based on computational models of current biological theories. The creatures built within this framework appear as animated characters on screen. They have embodiments and personalities which are controlled by the subsystems of the creatures. These subsystems include sensory, perception, belief, action and motor, among other, systems. Similar to the subsystems of real creatures, the sensory system processes all the low level inputs to the creature and outputs the results to the perception system, which uses the results to evaluate perceptions of the world. The belief system then uses the evaluated perceptions to build beliefs about the world. The action system uses the beliefs and perceptions to decide what to do. The motor system executes low level motor actions to implement the physical actions the action system chooses. Other systems that have been developed include navigational and emotional systems, both of which affect the overall behavior of the creature.

2.2.2 Flow of operation
The virtual worlds created using the Synthetic Character architecture run by continually updating all the objects contained in a world. Each world contains a list of 'updatables' that it goes through and repetitively updates. As a creature is an instance of an Updatable, the world updates each creature as it traverses the list. During each creature’s update call, the creature updates each of its systems in turn and then updates any updatables that pertain specifically to it, (i.e. these updatables were added to the creature and not to the world and are only updated if the creature is updated). The creature updates the systems again when its world turn to update next comes around. In most installations created by this architecture, different entities in the world are not run by different threads; the entire world runs on one thread, (with the exception of some input devices that might create their own threads). This choice avoids all deadlock and concurrent modification problems that might arise, however it does impose the restraint of sequentially updating each entity, and does not allow for different computers running different creatures.
**Figure 2.1. Flow of the Update Loop in the Virtual World.** The world updates each entity in the world during each iteration though the update loop. Each entity updates its systems during the entities turn in the world update loop.

### 2.2.3 Sensory System

The sensory system of a creature consists of various sensors and filters that receive and process raw data from the world. This system handles all the inputs to the creature and does low-level processing of these inputs in order to package them into structures used in other parts of the system. We call these structures data records; objects that consist of keys linked to data. For instance, a data record created from seeing a red ball might contain the key for color and the data associated with that key would indicate red. It might also have a key for shape which would point to data indicating a ball shape. Raw data to the sensors can also come in the structure of data records. Most virtual output devices in the world, such as other creatures, output their raw data in the form of data records.

Each sensor pulls raw data from some output device in either the virtual world or the physical world. For instance, Minni has a sensor that pulls data from a microphone in order to obtain raw noise data. This sensor then processes the sound data into pitch contours which are used later in the system to develop beliefs and perceptions about the tone of voice used in speech utterances. Almost all creatures have a sensor called a ‘virtual sensory input sensor’ which pulls data the other virtual entities in the world send out. When each virtual physical object in the world is created, any raw data that should be passed on to other creatures – such as color and shape, is placed in a data record that the physical object sends out to the world on each update along with position information. In some cases, this information is not sent out, and the other creatures must learn this information through other methods – such as machine vision. However, this structure allows the designer of a system to choose what she would like to focus on when designing the system, whether that is machine vision, social learning, or machine training, and not need to worry about other aspects that are less relevant to the current research.
The sensory system also contains filters that filter lists of data records in order to cull superfluous information. Other systems that use the filtered data records then do not have to do unnecessary processing of extraneous information. For instance, if a creature is receiving information about several different other objects in the world, but it only wants to focus on one of these objects, the sensory system can filter out all the data records that do not pertain to the object of interest. Humans and other animals have a similar mechanism that allows them to pay different amounts of attention to different things in their worlds. For instance, many people will tune out any sound when they are reading, making it difficult to get their attention.

When a creature is called to update itself, it updates the sensory system first. The system first calls each sensor to return the list of data records it processed. These lists are compiled into one larger list that is then filtered by each filter. The final list of processed, filtered data records is then passed to the perception system.
Figure 2.2. Sensory System Example. The sensory system processes the raw data in sensors which package the data into data records. It then sends the data records through filters to cull extraneous information. In this example, the creature is paying attention to the red ball and not the brown dog, so its attention filter filters out the data record about the dog.

2.2.4 Perception system

The perception system is structured as a tree of percepts. Percepts are a representation of concepts the creature can perceive, such as shapes in general, a specific shape, an object type, or a specific object. The more general percepts are at the root of the tree and as one traverses the tree, the percepts become more specific. For instance a shape percept would be high in the tree, while the triangle and circle percepts would be the shape percepts children, located lower
in the tree. The triangle percept might have children itself, such as an equilateral triangle percept and an isosceles triangle percept. The perception system and sensory system are closely linked because the sensory system must package data in a way that the perception system can use, i.e. the perception system utilizes a representation of the world and the sensory system processes data to fit into that representation.

The trees of most creatures have certain special percepts including a root precept, often called a true percept, an external data percept root, a derived percept root, and a proprioceptive percept root. The true percept encompasses the idea that the creature can perceive anything at all and is the parent of all other roots and therefore the ancestor of all percepts. The external percept root is the ancestor of all percepts pertaining to perceptions about the external world, i.e. other creatures, objects, and sounds. The derived percept root is the ancestor of all perceptions that are based on combinations of other percepts. For example, a creature could have a ball percept and a red percept. The derived percept for a red ball would be a descendent of the derived percept root. The proprioceptive percept root is the root for all percepts pertaining to things the creature perceives about itself, such as what action it is performing.

As a creature learns about the world, more percepts are added to its tree. Some of the percepts are added immediately after a new concept is encountered. For example, when a creature first starts its ‘life’ it may only have a concept of shapes in general, i.e. it has a shape percept that recognizes any shape. When the creature starts discovering different shapes however, specific shape percepts, such as mouse or tree percepts are added as children of the general shape percept. Other percepts are added by a more controlled process. One example of this is the classifier and model percept structure. A classifier percept is designed to learn about some concept and to develop models for different derivations of this concept. While the classifier percept recognizes any kind of the concept, its children, model percepts, only recognize a specific derivation of this concept. For example, a sound classifier percept may have children percepts to recognize specific sounds, such as certain words. When a new sound is heard that the classifier cannot classify, the sound percept might decide to ignore the deviant
instance, or it might add a new model percept that learns to recognize the new word.

Each percept can return a match and confidence value associated with how closely an instance of a data record matches the percept’s concept and how confident the percept is in that match. For instance, a red color percept would return a match of 1.0 and a confidence of 1.0 (1.0 being the best match and highest confidence 0.0 being no match and no confidence) when presented with an instance of a red ball, and would return a match of 0.0 and confidence of 1.0 when presented with an instance of a blue ball. The match value a percept returns for a specific instance does not change over time, while the confidence value can change. For instance, if an instance of a red ball moves out of the creatures field of view, the red percept will always return a match of 1.0 for that ball, yet the returned confidence value would decay more and more the longer the ball remains out of view.

The perception system sends the list of processed and filtered data records from the sensory system through its tree of percepts during its update call, which occurs right after the sensory system’s call. Each data record in the list traverses the tree of percepts, instead of each percept of the tree traversing the list of data records. While a percept processes a data record, it decides whether to send the data record on to its children for processing. ‘True percepts’ send every data record they receive on to their children. The external data percept root only passes data records that pertain to external world, while the proprioceptive percept root only passes data records that pertain to the creatures perception of itself. Classifier percepts only pass along data records that pertain to what it is classifying. The ability to stop a branch from processing an irrelevant data record restricts the amount of unnecessary processing that occurs. During the processing of a data record, the percept calculates a match and confidence associated with the data and packages these value in the form of a percept evaluation data record. After the entire list of data records has been processed, the system can send the list of percept evaluation data records on to other systems for use in building beliefs about the world.
The perception system evaluates the data record by first asking the root to process it. In this example, the root decides to pass the data record to its children. Only the external data root decides to pass this particular data record on to its children as it is the only one that calculates a high match; the data comes from an external source. Of its children, only the shape percept passes the data record on; its match is 1.0 as the record has shape information. After none of its children compute a high match, the shape percept might add a new child: a ball shape percept.

**2.2.5 Belief system**

The belief system organizes the percept evaluation data records obtained from the perception system in a way that allows the creature to maintain a set of beliefs that the action system can use to decide what behaviors to enact. The belief system consists of structures called beliefs that hold histories of percept
evaluation data records which pertain to percepts which are related to the belief. Often the behavior system wants to know how a percept has been evaluated in order to decide what to do. When this happens, it can query the belief system with its request. Common requests include a request for the evaluation with the highest match value, a request for an evaluation with a match value higher than .5 within a certain time window, or a request for an evaluation with a specific data value retained by the record. Other requests are defined when specific functionally is required.

During the belief systems update call, the system first updates its current beliefs, then it uses the percept evaluation data records obtained from the perception system to create new beliefs. All of these beliefs are then sent through a merger method which merges similar beliefs and separates distinct beliefs. The designer of a system defines what indicates a similar belief and how to update beliefs through metrics installed when the system is created. If no merge metrics are added, then all beliefs are merged into one big belief. During the system update the system also throws away old stale beliefs and percept evaluation data records.
Belief System Example

Figure 3.4. Belief System Example. This creature has two beliefs: one for the red ball and one for the brown dog. The belief for the ball contains histories of the percept evaluation data records pertaining to the shape, color, sound classifier and sound model percepts. The confidence values decrease the older the evaluation record is, while the match remains the same.

2.2.6 Action system

The action system uses the beliefs created and maintained by the belief system to dictate the behavior of the creature. These actions that make up the action system can be broken down into the following types: action groups and general actions.

General actions are the most basic types of actions in the action system; they are applied all the time, i.e. every time the action system is updated. A common general action is the look-at action. This action turns the creature's
head and eyes to look somewhere. It is separate from his other behaviors so that every behavior does not have to implement functionality for head movement.

Actions groups consist of a group of related actions and a mediator to choose between those actions. A mediator can choose to apply all actions, no actions or a subset of actions within the group. The mostly commonly used action group is the probabilistic action group, which is comprised of startle actions and regular actions. The mediator of this group chooses probabilistically between the regular actions unless it is startled into applying a startle action. (‘Startled into applying’ will be explained below.) It picks between regular actions by examining their action tuples. Action tuples contain a trigger, an action to apply, a do until and an object. For instance, an eat action tuple would have a trigger for ‘I’m holding food’, an action to apply of “eat the food”, (in our system an eat animation), a do until of ‘I finished eating the food’ (in our system, until the eat animation finishes) and an object of ‘the food that I am holding’. Each tuple has a value associated with it that identifies the current importance of the action. For instance, if none of the triggers associated with that action are active, the action tuple’s returned value is zero. When choosing which regular action to do, the system multiples the value of each action tuple by a random number, and then chooses the tuple associated with the highest product. Startle actions are also defined by action tuples. The difference however, is that a startle action is applied whenever its triggers evaluate to a number greater than 0.0. When this occurs, the mediator stops the current action in progress (whether or not it has finished) and starts applying the startle action. This is called startling the mediator into applying a startle action.

There are many types of triggers, though the most common are triggers based on percept evaluations. These percept triggers activate when percept evaluation data records in the belief system fit a certain description. Frequently this description consists a match and confidence higher than .5 within a certain time window. The trigger sends a request, (see 2.2.5 Belief System) to the belief system for a percept evaluation data record that fits the description. Other triggers activate when the creature has finished a specific action or randomly.
Action tuples can have multiple triggers, as long as a decision policy is specified; the designer has to define the ands, ors and nots of the decision logic.

In the Synthetic Characters architecture most actions contained in action tuples are actions that apply animations. These actions tell the motor system to apply specific animations while the motor system actually runs the animation. Depending on how the creature is defined, the action system or a separate emotional system informs the motor system how to apply the animation, i.e. how to blend between animations for the same action created with different emotional affect.

Do untils work on the same mechanism as triggers. However, they specify when the action is finished, whereas the trigger indicates readiness to start. A common do until calls for the termination of the action tuple when its animation has finished playing out.

The value of an action tuple depends heavily on what the designer wants to do with the creature. If the designer wants to design a creature that learns certain behaviors, the action tuple values will vary as it learns about each behavior and its importance. If the designer wishes to focus on other aspects of the creature, such as how he perceives the world and forms beliefs, the values might be constant and similar.

Similar to the perception system, the action system can add actions as it learns about the world. Often new actions are added within an action group, though general actions can also be added. Dobie T Coyote adds actions when he has clearer ideas about what should trigger an action. He adds actions that are triggered off verbal commands once he learns both the command and that it is associated with the action.

During the action system update call, the system itself updates each general action and each action group, which in turn update their own actions according to how they were designed.
2.2.7 Motor system

The motor system does the low level implementation of animations. As the systems operate in real time, the motor system was designed to blend and run animations in real time. The system can blend between animations for the same action that have different emotional affects i.e. a stand happy and a stand sad animation, between animations with different orientations, i.e., a walk left and a walk right, and between the end and beginning of different animations. This last trait enables the animator to not need to create animations for every transition that might occur between main animations.
2.3 Affect in voice

2.3.1 Physical affect of emotions on creatures

Humans have mechanisms for learning the affective state of other individuals; one can predict what kind of mood someone else has. On a high level, these mechanisms focus on facial expressions, body position and movement and vocalizations. On lower levels, this mechanism must decide which features of each high level attribute to pay attention to. There has been extensive work done on classifying the influence of affective state in vocalizations of humans and other animals.

Most vocalizations are affected by the psychological state of the animal as this arouses the nervous system. The nervous system affects heart rate, blood pressure, respiratory depth and pattern, and mouth salivation. These factors in turn affect loudness, frequency range, median frequency, rhythm, fluctuations of vocalizations. For instance, when a person is excited, heart rate goes up, breathing becomes faster, and the mouth dries up. This causes the speech to be louder, faster, more enunciated, and shifted toward higher frequencies. These physical effects appear in speech signals and can be used to automatically classify affect in speech.
2.3.2 Automatic classification of affect in vocalizations

The most recent automatic affect recognition systems include Roy and Pentland (1996), Delleart et al (1996), Slaney and McRoberts (1998), Nakatsu et al (1999) and Breazeal and Aryanda (2002). Roy’s system discriminates between approving and disapproving vocalizations using pitch and energy statistics and an open quotient and spectral tilt measures. (Open quotient is the ratio between the time the vocal folds are open to the total pitch period. Spectral tilt is the overall slope of the speech signal’s spectrum.) Delleart distinguishes four affects: happy, sad, angry and fear using only pitch statistics. Slaney’s system distinguishes three affects: approval, attentional bids and prohibition using pitch, mel frequency cepstral coefficients and energy. Nakatsu looks at eight emotional states: angry, sadness, happiness, fear, surprise, disgust, playfulness and neutral using energy, pitch and LPC (linear predictive coding) statistics. Breazeal’s system distinguishes the five features using pitch and energy. Slaney and Roy focused on utterances adults would direct toward small children. Breazeal focused on a similar class of utterances, those a person might direct toward a robot.

All of this work has been done by labeling the affect from the person issuing the vocalization’s standpoint. Either someone specified to the speaker what affect to add to his speech, or people were recorded in natural settings and then asked to label the utterances with an affect. Trainer a classifier this way relies on the assumption that the speaker and listener associate the same features of the utterance with the affect and that the listener can recognize these features in the vocalization.

Each of the systems trained the classifier off line and then used the classifier to recognize affect in real time. This restricts the classifier to the affect spaces found in the training data, not allowing it to adapt to new speaker tones.

The most useful features this research found for classifying affect include pitch contour, mean, variance, maximum and range, pitch segment average length, minimum length, and number, and energy variance, maximum, and range.
2.4 Association learning

Animals obtain information for decision making from their surroundings; they observe and interact with the environment for this purpose. Creatures have both the ability to convey information by their own actions and the ability to deduce information from the actions of others. Common actions that contain information are vocalizing, holding a certain body stance, and moving part or all of the body. These actions all contain information because the receiver associates the actions with something, another action, a mood, a warning. Without an association, the action means nothing at all. For instance, a dog growls to warn it will bite. A child can deduce that the dog will bite him because he has seen a dog growl and then bite before. Because he understands this association, he will stop pulling the dog’s tail and avoid a bite. However, if the child did not associate the growling with a warning, he might not stop pulling the dog’s tail. He would therefore receive a bite because the growl did not contain any relevant information to him. He’ll perhaps learn better next time.

Associations are a central part in how any creature processes information. Though some associations are built into the creature, most are learned through experience giving creatures the ability to adapt to a changing environment. For instance, a child may believe that a certain tone of voice, which we shall call a friendly tone of voice, indicates good things. However, her older brother may use a friendly tone of voice when he tricks her into giving him her candy. She eventually learns a friendly tone of voice is not always associated with good things. This is an important example of learning an association independent of speaker projected intent. The brother uses a friendly tone of voice because he believes his sister associates it with good things, making it easier for him to trick her. However, his intent is not good, which she must learn.

One question that presents itself is how does a creature learn associations when it is being bombarded with so many stimuli? How does a creature know
what features of the stimuli contain information? For instance, how does a creature know that the time when events occur is important information to use to learn associations? Why do we pay more attention to the main pitch of an utterance than other features of an utterance?

One theory is that while creatures do not have many associations built in, they have certain mechanisms built in. These mechanisms help the creature decide what to pay attention to in order to form meaningful associations, Gallistel (2002) presents this idea as adaptive specialization. Creatures extract information from experience, for instance they learn distances, directions, times of day, and duration of events. These are not low level sensory signals, these are features extracted from those low level signals. Gallistel explains that a single feature extractor does not suffice for all the inputs a creature processes into information. Creatures evolved to have adaptively specialized learning mechanisms, each of which specializes in extracting certain information from certain inputs. Once this information has been extracted, the creature can then form associations based on the outcome of an action in the context of the extracted relevant information. The formation of associations based on consequences is often called reinforcement learning. Research in this area includes both on how animals learn these associations and how to implement reinforcement learning in machines.

Blumberg (2002) and Blumberg et al (2002) shows how clicker training, a form of reinforcement learning, in real dogs can be applied to machines to produce similar behavior. Clicker training starts by teaching the dog to associate ‘good’ i.e. food, with a click. Then, whenever a trainer wishes to reward the dog, he can click at that exact moment. Clicking at the moment the dog did something wanted is better than feeding the dog later, as it makes what the trainer actually wanted vague to the dog. After this stage of training, the dog associates the action the trainer wants with good also. After the dog learns the good action the trainer then goes on to add a command during the training. The dog then learns to associate the command with the action which he previously learned was good. This is an example of reinforcement learning as the good click reinforces good behavior.
This idea of clicker can be extended to machines by hard programming a good thing for the machine which serves as an analog to food for the dog. Dobie T. Coyote (see 2.6 Dobie T. Coyote) was implemented in this fashion. Using the idea of forming associations with good things and bad things can be used for general learning, as opposed to just training. Minni has virtual cheese as her food and good thing. Under this paradigm she is trained to recognize that some tones result in a reward of cheese and some do not.

2.6 Dobie T. Coyote

Dobie T. Coyote is a synthetic character designed as a trainable dog. The user, a trainer of the virtual dog, trains Dobie the way a dog trainer would train a real dog. She teaches him commands for actions by rewarding him with clicks associated with food. (See 2.5 Association learning) Dobie’s learning system was designed based on the principles of clicker training. In its design, Dobie learns to classify vocal commands in his perception system. He learns which actions are good and what associations to make in his action system.

Dobie’s perception system contains a voice classifier percept that uses cepstral coefficients and dynamic time warping to measure distance between voiced commands. The classifier labels utterances with an interesting action Dobie was performing during a time window around the utterance. Interesting actions are those that he can learn about – the actions he can be trained to do – sit, play bow, howl etc. Classification is done by finding the nearest neighbor. A new example is classified with the label of the closest utterance retained by the classifier.

Dobie’s action system is structured as a main probabilistic action group with two startles and several regular actions. The startles are triggered from being rewarded and punished. They affect his emotion level. He becomes happier when rewarded and sadder when punished. The regular actions include sit, howl, play and shake. These regular actions are structured as innovational
action tuples. These action tuples are groups of tuples that apply the same action but have different values and triggers. When these groups are first created they contain one tuple with a trigger that is always active and a neutral value. The tuple group keeps statistics on the percepts that are active while its tuples are being applied. The values are variable to allow Dobie to learn which actions are good to perform and which are not. The value of a child tuple decreases when Dobie is punished during the application of that tuple and increases when he is rewarded. Since values affect how often an action is chosen by the action group mediator, an action that is rewarded often will be chosen more often. Thus Dobie will perform an action more often for which he has been rewarded often. Conversely Dobie will perform an action less often for which he has been punished. Both consequences mimic real clicker training.

**Figure 2.7. Innovation Action Tuples.** Dobie has been trained to distinguish between sitting with and without the sit command. He is in the process of learning to distinguish between roll over with and without the verbal command. He has not been trained to howl.

The trainer says a command every time Dobie does the action for which she wants to train. The classifier percept learns to classify this command and spawns a child model percept for the command. This child activates when the
command is spoken. When the innovational action tuple for the action calculates
statistics that indicate a percept (the classifier's child percept) is usually active
during the action's application, the tuple group will spawn a new child tuple with
a percept trigger based on the associated percept. This new child has its own
value distinct from the original tuple's value. After this child is added, Dobie can
begin to distinguish between doing the action anytime and doing the action on
command.

The trainer now only rewards Dobie for doing the desired action when she
says the command. The original tuple's value will start to decrease as it is not
being rewarded, while the value of the tuple associated with the command will
increase as it will be rewarded. After several training iterations, the value of the
tuple associated with the command will be high enough to trigger the action
whenever Dobie heard the command, completing his training of that action.

Figure 2.8. Dobie T. Coyote. Dobie watches the hands that reward and
punish him. The left hand is holding a clicker whose click Dobie associated with
good things - food. (add diagram of innovational action tuples)

2.7 Maximillian T. Mouse
Maximillain T. Mouse is a teasable synthetic character. A person interacts with him by teasing him with a piece of cheese. She waves a physical cheese around and using infrared tracking Max follows the cheese. When the user brings the cheese close enough, Max grasps for it because he wants it - he is perpetually hungry. If she keeps the cheese close for too long, the cheese appears in Max's virtual world and he grabs it, then eats it, becoming very happy. If the user yanks the cheese away however, Max grabs at nothing and becomes sad, confused, frustrated and then angry. Max decides whether he likes the user or not depending on how often he gets the cheese. He only uses the sequence of obtaining and missing the cheese to learn about a person's giving pattern. However, his emotions are more complex than merely being happy when he receives cheese and sad when he does not. After learning about a person's pattern of giving him cheese or not, he becomes surprised or confused if the pattern changes. If a person gives him cheese often and then stops he becomes confused and then he angers more quickly. If a person has not been giving him cheese and then does, he is surprised and becomes extra happy.

Besides learning about a person's giving pattern, Max has the capability to learn about different people and try to influence them. He has the ability remember different people and how he feels about them after they have left and come back. He also sees using computer vision techniques when people move around his area. He tries to get them to interact with him by knocking on his window and begging them for food which he points out is only a few feet away on the counter by his window.

Max also influences people with his very expressive personality. His rendering style and wide range of actions allow him to express despair, confusion, frustration, anger, surprise, joy, excitement and the ranges in between. His personality influences people the way a pet's personality influences them. A person often feels she were too harsh in her teasing when she sees Max so upset.

Max was the inspiration for Minni. As Max could not use normal social clues to decipher people's actions, we were further reminded how important they
are to human interactions. Minni was designed to make up for the lack of social skills in Max.

Figure 2.9. Maximillian T. Mouse. Max looks for the cheese he desperately wants.
Chapter 3

Minimus T.O. Mouse

3.1 Meet Minni

Using the ideas and tools presented above, I designed a system that learns the intent of a person from their tone of voice. The system is in the form of a synthetic character named Minimus T.O. Mouse, (Minni for short). Minni learns to associate different tones of voice with whether she gets food (the cheese) or not. She hears the person interacting with her and starts building associations between getting and missing the cheese with features of the vocalizations she hears. The features are pulled out by algorithms I implemented and the associations are modified every time she tries to grasp the cheese. Her sensory system pulls the sound and computes the features. The perception system learns to classify the sounds based on feedback from the action system. The action system uses the perceptions evaluated by the perception system and beliefs made by the belief system to decide the best action to apply. The action system chooses her actions guided by the probability of something good happening. Minni was hard programmed to think getting the cheese is a good thing and not receiving it is bad.

3.2 Minni is a synthetic character

I chose to build the tone learning system as a synthetic character for several reasons. A system that learns tone of voice is only useful if the tone of voice used in an utterance is relevant to the meaning of an interaction. People are aware that machines do not recognize different tones of voice, therefore they refrain from adding information via that channel. Thus, adding a tone classifier
to a common machine would be useless. However, people do add information via their tone of voice to utterances when they speak to animals. Synthetic characters have personalities which naturally brings humans to talk to them as they would animals. Thus providing a character with the ability to learn about tone allows the character to be more socially competent when interacting with humans. Synthetic characters provide a sound framework for developing systems which interact with humans and learn about the environment. This framework fits well with the goal of building a system that learns about a humans intent.

### 3.2 Minni’s sensory system

Minni’s sensory system is made up of a sensors that pull raw data from the world and process it to pass along to the perception system. The sensors include a tone sensor and virtual sensory input sensor. The virtual input sensor pulls the data coming from the other virtual entities in the world, such as the cheese. Specifically it provides the cheese’s shape and location to the perception system. The tone sensor pulls raw sound data from the world, processes it into a pitch contour which is passed along. The pitch contours are created by the sensory system, while the perception system uses them to classify affect.

The tone sensor pulls raw data from the world by sampling sound at 44.1 kHz from a microphone and quantizing each sample to 4 bytes. By processing buffers of 1024 samples the sensor provides pitch estimates every 1.4 milliseconds.

As pitch is associated with a sinusoid, one can use the properties of sinusoids to extract the pitch. A sample of a sinusoid, \( s_n \), is characterized by its relationship to its neighbors:

\[
s_n = \alpha \frac{(s_{n-1} + s_{n+1})}{2} \quad (1)
\]

Solving a least squares problem on the error function
\[ \varepsilon(\alpha) = \sum_n \left[ x_n - \alpha \left( \frac{x_{n-1} + x_{n+1}}{2} \right) \right]^2 \]  (2)

gives an optimal estimate of \( \alpha^* \):

\[ \alpha^* = \frac{2 \sum_n x_n (x_{n-1} + x_{n+1})}{\sum_n (x_{n-1} + x_{n+1})^2} \]  (3)

This leads to the optimal estimate of the frequency,

\[ \omega^* = \cos^{-1}(1/\alpha^*) \]  (4)

This estimation approach leads to the algorithm developed by Saul et al (2002) that I implemented to extract pitch. First, one low pass filters the signal in question, and then downsamples it. According to Saul’s algorithm, the signal would then be passed through a pointwise nonlinearity, such as squaring or clipping negative values, that concentrates the energy at the fundamental frequency. The signal then passes through a filterbank creating several bandlimited signals. A frequency estimate is obtained from each of these signals using equations (3) and (4). The output of the algorithm is the frequency associated with the lowest uncertainty. Uncertainty is defined as

\[ \mu^* = \frac{1}{\omega^* \left( \frac{\cos^2 \omega^*}{\sin^2 \omega^*} \right) \left( \frac{1}{\varepsilon} \frac{\partial^2 \varepsilon}{\partial \alpha^2} \right)_{\omega^*}^{1/2} } \]  (5)

If certain conditions are not met, i.e., the minimum uncertainty is higher than a specified threshold, the algorithm can output a code for an unvoiced input. Other conditions for a voiced segment include a minimum energy, a valid frequency, and \( \alpha(\alpha^*) \ll \alpha(0) \), (mean squared error is small relative to the mean squared amplitude).

In my implementation, the tone sensor passes windows of 1024 samples into the algorithm. It low pass filters each window using a 9th order chebychev type II low pass filter to get rid of high frequencies created from the window and to prepare for downsampling. It then downsamples from 44.1kHz to 22kHz as
Saul’s algorithm suggests. It does not use a pointwise nonlinearity as I did not find that it helped the algorithm estimate the pitch. The sensor then sends the signal through a filterbank of 6 filters each of which covers 1.2 octaves. The filters are all 9th order chebychev type II bandpass filters with .5 db of ripple. The order was chosen as an effective but low order for the filters. I looked at many different filterbanks, making changes from order of the filters, to size of the passbands, to number of filters, to lowest and highest frequencies of the overall filterbank. The final configuration returned the smoothest pitch contours.

While finding the pitch estimate from the filterbank with the lowest uncertainty the only condition the sensor applies to determine an unvoiced segment is $|a^*| > \frac{1}{2}$ resulting in an invalid frequency. After returning the pitch estimate from the algorithm, the sensor does further voiced versus unvoiced processing to find pitch contours of utterances. It does this by looking at the uncertainty estimates and energy. Specifically it looks for windows of estimates which have low uncertainties and high energies. When such a window is found the pitch contour and energy estimates are packaged into a data record and sent to the perception system.

### 3.4 Perception system

Minni’s perception system receives all the processed data records from the sensory system, including those that provide cheese location and those that contain pitch contours from utterances. The cheese location data records activate proximity percepts that indicate whether the cheese is within grasping distance. (Activate is used here to mean the percept returns a high match and confidence.) The pitch contour data records activate the sound classification percept and its model children and are used to classify the voice affect. The action system later uses the cheese proximity and tone classification to decide what Minni should do.
Figure 3.1. Minni’s Percept Tree. Minni has percepts that allow her to reason about what she is doing, where the cheese is, what sounds she hears and combinations of these. In this snapshot the voice classifier percept classifies two different tones, one that occurs when she misses the cheese and one that occurs when she eats the cheese.

The classification percept receives pitch contour data records during its update call. During this time, it classifies the contours it receives, based on what it previously learned. Based on the classification, the relevant child percept (if one is found) activates. Later in the creature update call, the action system sends feedback to the percept, which the percept then uses along with recent pitch contours to train its classifier.

The classifier percept contains a classifier which uses models to train and recognize tone. The children of the classifier percept are model percepts which house these models. When the classifier percept receives a new data record, it classifies it by finding the closest model. The classifier percept activates every time a pitch contour is received, while model percepts only activate if the classifier chooses their model as the closest model. (Classification is discussed below.)
When the action system finishes running an interesting action (see 3.5 Action System), it sends the time window of the when the action was applied and an action label to all percepts that are interested. In Minni’s design, the voice classification percept is the only interested percept. The classification percept then asks the belief system for the most recent pitch contour data record and checks that the utterance occurred within the appropriate time window. The classifier classifies this contour and if the classification matches the action that actually occurred, the classifier reinforces the appropriate model. If not, a new model is created with the test data record.

Classification is done by computing the distance from the example data record to the models. The distance to the closest model is then compared to a threshold. If the distance is under the threshold the model is returned as the classification, if not the classifier returns a code for no match. The distance from an example to a model is defined here as the average distance between the example and each record the model houses. The distance between two contours is measured by the difference of fitted polynomial coefficients. Each contour is fit to a 2nd and 3rd degree polynomial. Before fitting the contour, the y axis (pitch estimate) is translated down by the minimum pitch and the x axis (time) is stretched by maximum pitch divided by maximum time. First and 2nd degree polynomials are fit to the \( F(\text{time}) = \text{pitch contour} \). A 3rd degree polynomial is fit to the \( G(\text{pitch}) = \text{time contour} \).
Figure 3.2. Pitch Contours and Polynomial Fits. The left pitch contour has less error with a 2nd degree fit while the right contour has a smaller estimation error with a 3rd degree fit.

I translated and stretched the axes so that the errors found from these rotated fits would be comparable. The pitch contour is given a shape label based on the fit with the least error. The distance between two contours is defined as infinite if they have different shape labels. If two contours have the same shape, the distance between them is defined by the norm of the difference of the vectors of coefficients of the polynomial that corresponds to that shape.

\[ e = \| c_1 - c_2 \| \quad (1) \]

\( c_1 \) and \( c_2 \) are the coefficients of the polynomials that fit the contours best

Other ways to define distance between contours include a weighted norms of all differences of polynomial coefficient vectors.

\[ e = \sum_{i=1}^{3} \| c_{1i} - c_{2i} \| \quad (2) \]

\( c_{ki} \) are the coefficients of the \( i^{th} \) degree polynomial fit

and the norm of the error between samples of the fitted contours stretched to have the same time axis.

\[ ts = [0 : 1 : T] \]
\[ Ts = [ts^1, ts^{1-1}, ... , 1] \quad (3) \]
\[ e = \| Ts1 \ast c_{1i} - Ts2 \ast c_{2i} \| \]

Error (1) was chosen as its performed as well as the other errors and requires less computation.

When the classifier classifies an example correctly (according the action that occurred) it reclassifies the example, only using the models that correspond to that action. Looking at the smallest distance to a model, it either adds the example to the model as reinforcement or creates a new model, depending on how far that distance is.
New models are labeled by the action specified by the feedback from the action system. After creation the example that caused the creation of the new model is added to that model. These models are continually updated as new examples are added and old examples are culled.

I used an important aspect of how Dobie learned in Minni’s system. His classifier uses the context of a rewarded action as a label for utterances. It also uses this as implicit feedback that the utterance heard during around the reward is relevant to the action and should therefore be added to a model. Minni’s classifier also uses the context as a label and as feedback that the vocalization is relevant. However, she does not always add the example to a previously existing model. Sometimes these examples cause the creation of new models. New models are added to the classifier for three reasons. A new interesting action has occurred in the time range of a vocalization, a vocalization occurred during an action that already made models but the distance from new example to all the models is too large or a vocalization was classified incorrectly according to the action that it thought would occur and the action that did occur.

The method of checking previous classifications and using this to improve the system was not used in Dobie’s classification system. His system did not need feedback as it used a nearest neighbors classification scheme and clusters cannot be formed in this way. Clusters are formed in Minni’s approach to classification. The feedback helps shape the clusters and create new clusters where appropriate, thus improving classification.

### 3.5 Action system

Several top level actions start the composition of Minni’s action system. These actions include an action that controls where Minni looks, an action that controls where she reaches and grabs, and a main probabilistic action group. The probabilistic action group has several functions. Like all probabilistic action groups, it chooses between startle actions and regular actions for Minni to
perform. However, it also sends feedback information to the perception system so that the voice classifier percept can learn. Finally it decides if an association between an action and a percept should be used to decide information in the future.

When the system is first created the main action group is made up of two startle actions and 3 regular actions. The initial two startle actions are triggered off of a regular action, grasp. When Minni grasps for the cheese one of two things happen, she grabs and eat it or she misses it entirely. If she eats the cheese, the thumbs up startle action is triggered and she gives you a thumbs up to show her immediate happiness. If she misses the cheese entirely, the miss startle action is triggered and she looks at her hands with a confused expression.

The initial regular actions include grasp, beg, and reach. Beg and reach have no special triggers or do untils. Therefore the probabilistic mediator inside the action group chooses among them at random if no startle action has been triggered. The mediator also lets them run for a random amount of time. The mediator adds the grasp action to the contestants when the cheese is within a certain distance – Minni thinks she can reach it. When the grasp action is applied Minni grasps once for the cheese, eats it if she grabs it or misses entirely. The eat or miss then triggers one of the startle actions.

When each action has completed this main group must decide whether and what feedback to send to the perception system. If the action is interesting, it sends the action label and the time window it applied the action to the perception system. Interesting actions are ones that cause an immediate change in Minni’s emotions. The thumbs up action is interesting because she becomes very happy as she just ate cheese. The looking at her hands confused action is interesting because she becomes sad as she just thought she would eat and then did not. The group goes through each percept to see if it is interested in this information. The only percepts that are interested are the classifier percepts and its child model percepts. They use this information to learn and classify tones of voice.

In addition to giving feedback to the perception system to aid learning, the group reasons about actions and percepts to learn. If a good thing occurs after she performs an action while a certain percept is activate, then Minni learns that
she should do that action when the percept is active in order to get the good thing. If a bad thing occurs under the same circumstances, then she learns not to do the action when the relevant percept is active. If Minni gets the cheese after grasping for it when she hears a soothing tone of voice but does not get the cheese after hearing a mocking tone of voice, then she will learn to grasp whenever she hears a soothing tone of voice and never grasp for it when she hears a mocking tone of voice.

This learning occurs by keeping statistics relating active actions to active percepts. When the statistics reach a certain level, an association forms between the action and the percept. If the action is a good action, i.e. eating, Minni learns to grasp when the percept is active. If the action is bad, i.e. missing the cheese, Minni learns to not grasp when that percept is active. Structurally this is manifested by adding new actions to the action group. A good association leads to a new startle action for grasp that triggers off the relevant percept. A bad association leads to a new startle that restricts Minni from grasping at the cheese, no matter how close it is. The learning is also manifested in her behavior. Minni grasps at the cheese, no matter how far it is, when startled into grasping by a good association and never grasps when startled by a bad association. Instead, when startled by a bad association, Minni crosses her arms and taps her foot annoyed.

Though this system of learning associations is similar to the methods Dobie T Coyote used, it is different for several key reasons. Dobie could be trained to do any of the actions in his repertoire; however, he could not be trained not to do something. I wanted Minni to learn to not grasp for the cheese if she knew she was not going to get it, as this is a waste of energy. In general, one does not want a machine doing something if it knows the results will not be good.

The structure of the two systems is different for this reason. The innovation action tuples cannot be used effectively to train the creature to not perform an action. They add action tuples each of whose value can be positively or negatively reinforced, however, unless all tuples are negatively reinforced, the mediator will choose the highest tuple to apply. This unfortunately negates the effectiveness of the negative reinforcement. In order to implement a system that
could be taught not to perform an action, I needed to design a different system. I created the probabilistic action group described above for this purpose. Instead of grouping similar actions together in a subgroup with its own mediator, the final group adds actions to the main group. Thus other different actions with high values will be chosen over the negatively reinforced actions.

3.6 Emotion

Synthetic characters are designed to express emotion and Minni is no exception. Her ears and posture indicate her happiness level. Her happiness is directly tied to the cheese. If she receives the cheese often she is very happy. If she does not receive the cheese very often, she is sad. After learning the intent from tone of voice, hearing certain things can affect her mood also. If she is happy and hears a bad tone of voice, her ears will go down as she thinks she will not receive the cheese again. If she is sad and she hears a good tone of voice, her ears go up as she thinks she will get to eat the cheese. I chose to use her ears as an indicator of her happiness level as people are accustomed to seeing dogs ears indicate their mood.

Max also uses his ears to indicate his happiness level. However, he has more ways to express himself. Max’s renderer changes his color to match his mood. He has a wider range of actions to express a wider range of emotions. Besides the sad to happy range that Minni has, Max can also show surprise, despair, excitement and anger.

Though Max can express more emotions than Minni, he cannot recognize those emotions in his human user. Minni, while she does not understand the emotions expressed in vocalization, can learn to recognize the different affects. This gives her an advantage in social situations as she can better predict what the human will do with the cheese.

Max can be teased on one way that Minni cannot be teased. If someone holds the cheese just out of the range that Max thinks is reachable, he becomes
annoyed. However, though Minni does not think this is a form of teasing, she does still becomes sad. Minni has a hunger level that increases the longer she without receiving the cheese. If a person holds the cheese just out of range and says things she cannot classify from earlier events, she ignores what the person says, but gets sad because she gets hungrier and hungrier. However, if the person says things she does classify, she either gets annoyed because she knows the person is teasing her and will not give her the cheese, or she becomes excited and jumps at the cheese no matter how far it is because she thinks the person will give it to her.

3.7 How to interact with Minni

A person who wishes to interact with Minni does so in the following way. There is a microphone near the installation which picks up what the person says. There is also a gamepad which the person uses to control the cheese in Minni’s virtual world. The person moves the cheese around Minni’s world saying things to her as she grasps for the cheese. If the person is consistent with choosing certain tones of voice when he gives Minni the cheese and when he does not, Minni will learn that the former tones lead to eating while the latter tones lead to more hunger. She will start grasping for the cheese whenever she hears the former tones of voice and never grasp when she hears the latter.
Chapter 4

Conclusions

4.1 Evaluation

Minimus T.O. Mouse is an example of machine that can socially interact with humans and learn from those interactions. She is a unique system when compared to other systems that classify tone of voice and form associations because 1) Minni learns the classification online while she is interacting with a person and 2) she learns when to and when not to perform an action.

The key points of Minni’s system include online training of a classifier to recognize different tones of voice, learning associations between actions and tones of voice, learning whether the association leads to good or bad, and allowing Minni to modify her behavior based on these associations. All of these goals were met to various degrees. Minni learns to recognize different tones of voice and she learns associations between actions and percepts very well. She also understands that eating the cheese is ‘good’ and not eating it is ‘bad’ and she can add actions that change her future behavior based on her experience with a user and the cheese. The next section discusses what worked well and where improvements could be made.

Minni learns to distinguish vocalizations with different pitch contours well. She only needs to hear 2 or 3 samples with similar contours to develop a reliable model for the type of contour. While the learning works well when the estimated pitch contours are accurate representations of the vocalization, it does not work well when they are not, which does happen under some conditions. The algorithm to estimate pitch works well when a person uses a sing-song voice. However, it does not work as well when a person has a voice with several prominent formants. In this case, the estimator cannot distinguish between the different formants and so jumps between them returning very noisy pitch
estimates. In other cases two strong sinusoids are mixed together and the estimator returns a pitch in between the two frequencies resulting in more noise.

Minni learns associations between actions and percepts very well. This learning was made obvious when I added new proprioceptive percepts. I added an “I’m excited” proprioceptive percept that activates when any percept that she has learned is associated with good things activates and startles her into grasping for the cheese. The percept is supposed to alert the emotion manager that something has happened that Minni believes will result in goodness, thus increasing her happiness level because of the expectation. However, if Minni has learned the associations well, she grasps for the cheese and eats it every time she is startled by a good percept. Because she is excited about grasping for the cheese and she eats it every time, she learns to associate the excited percept with goodness. Unfortunately, as the associations spawn startle actions, after associating the excited percept with goodness, she spawns a grasp action that triggers off of excitement. However, this causes an unending cycle of “I know that’s a good percept so I should grasp” causing “I’m excited because I think something good will happen” causing “Excited is a good percept so I should grasp again” causing “I’m excited again because I think something good will happen” causing her to grasp forever. This problem is caused for two reasons, 1) the feedback for grasping – thumbs up or confused looking at hands is not given time to occur when good percepts trigger a new grasp and 2) Minni does not learn to undo associations that are no longer valid. To fix the first reason, I added triggers to the new startles that keep Minni from startling if she is already grasping or if she is in the feedback actions. This ensures she always receives feedback for her actions. To fix the second reason, Minni now does not form associations with proprioceptive percepts and actions. However, this is not a permanent fix. In the future Minni must learn to distinguish between percepts that are triggered directly from what she is doing and those that are triggered indirectly. In order to do this Minni must be able to continue to learn about associations and to unform associations that, as stated earlier, are no longer valid.
Though some associations can cause problems when feedback is not handled correctly, associations with vocalization model percepts still receive feedback and can adapt. If a model percept triggers a startle grasp and then Minni does not get the cheese, the classifier will create a new miss model whose first example is the utterance that caused the other model percept to trigger. As more examples are heard, the two models will separate as the eat utterances will reinforce the eat model and the miss utterances will reinforce the miss model. Because of this, Minni learns associations between eating or missing and tone model percepts very well. While learning the models, she develops associations. After the model has become reliable, she forms the association after 4 or 5 eats or misses.

Minni’s concept of good and bad is hardwired in to her systems, so she understands it very well. To her, eating is good and missing the cheese is bad. However, the actions that actually send feedback to the perception and action system are not the eat and miss actions as they are part of the grasp action. The actions that do send feedback are the thumbs up and confused looking at hands actions. This allowed for flexibility in the future for grasp to have a variable value that depends on the past. For instance, if Minni grasps and continues to miss, grasp’s value should decrease. However, if she later grasps and receives the cheese, the feedback she receives should a high value and not the low value that grasp now has. As eating and missing are part of the grasp action, she needs another mechanism to distinguish between grasp-eat and grasp-miss. The thumbs up and confused looking at hands provides this mechanism.

Minni is capable of changing her behavior based on what she thinks will happen in the future. She adds new actions to her repertoire that trigger off percepts from associations that she has learned will cause bad or good things to happen. If she has learned the associations and tone models well, these new actions help her to achieve her goal of getting cheese or to not waste her energy trying to get it when she knows she cannot. After she has formed what she believes is a reliable association, she adds a new action. These actions work well to show what she believes about the user and his vocalizations. They work well to aid her behavior only if she has learned the associations well. If she has not
learned the associations well, her behavior causes her to not grasp for the cheese when the user would give it to her, or causes her to grasp for it when the user will never give it to her. Minni can be tricked into adding new behavior which does not aid her if 1) the user deliberately starts using a tone when he won’t give her the cheese when previously he would give it to her or 2) the user uses a good tone to trigger Minni into grasping for the cheese and then uses a bad tone while she misses. The feedback to the action system and the perception system is based on the last utterance to occur. In the first trick case mentioned, Minni will develop a new model for bad utterances that will then classify the previously good utterances as bad. However, in the second case the last utterance was classified correctly and the trick utterance will not be reevaluated. In order not to trick Minni, the user should choose a tone and stick with it throughout the action.

4.2 Lessons

Through the creation and evaluation of the system I learned the following things: 1) combining simple systems can often produce the desired complex behavior for a system better than using complicated systems, 2) there are perhaps better ways of finding the pitch contour than the way I chose, and 3) the use of proprioceptive percepts has to be carefully considered.

Choosing to combine several relatively simple systems to produce the desired behavior turned out to be a good way to design the overall system for a couple reasons including: it was easier to understand the underlying behavior and its effects, and this approach uses algorithms that can be combined and run in real time on one machine limiting the need for processing power and increasing the flexibility of the system. During the debugging and testing of the system, when unpredicted effect happened, it was not too hard to track down the cause as the systems are easy to understand. Isolating and fixing any problem was not a long convoluted process, rather a thought exercise in elegance. The entire system can run, albeit slowly, on a laptop computer with 670 MHz.
processor and 512 SDRAM. More intense algorithms would not be able to run on such a computer.

Though the approach of using simple systems has several pros in its favor, the choice of simple systems is important. The pitch estimator I decided to use, though simple and effective for some cases, was not effective for others. As mentioned, it has trouble estimating pitches for voices that do not have a clearly prominent main formant. In the future I would look for a different algorithm to estimate main pitch that does not rely so much on the assumption that the energy of the main sinusoid is much larger than the energy contained in the other sinusoids.

Using proprioceptive percepts can be a powerful tool in the design of synthetic characters, but they have to be carefully considered. Though by definition they are percepts which describe what the creature itself is doing, there must be some mechanism in place that allows the system to know these are percepts it generates itself so that it can distinguish why they are being triggered and not start using circular logic. They are very useful for defining percept triggers that keep a creature from doing one action while it is doing another. However, a creature can form associations with these percepts that are circular.

4.3 Contributions

This work has contributions in the context of social machines, synthetic characters and systems. In the context of social machines, it is an example of a system that classifies affect in voice online instead of off. Previous social machines that make use of affect classifiers trained their classifiers offline. This system classifies affect in real time, allowing it adapt to new tones and people. Though Minni’s classifier’s success rate is not as high as classifiers trained offline, its ability to classify online is a huge advantage as social machines must deal with social situations which are continually evolving.
The ability to adapt to new people and to make use of tone of voice adds a new dimension to the available interfaces for synthetic characters. Developing characters that incorporate this ability allows humans to interact more naturally with synthetic characters. As a goal of designing these characters is to provide an interface that people can intuitively use to interact with machines, this contributes that their field. Minni’s ability to learn both to and not to do an action also furthers the research in the area of synthetic characters. Previous characters learned when to do actions, however, they did not learn when not to do actions. Learning when to perform actions enables the creature to take advantage of circumstances that allow it to succeed when other circumstances would not. In the context of the synthetic characters Dobie, Max and Minni, succeeding means being rewarded with a click or cheese. Learning when not to perform actions, however, keeps the creature from doing something it knows will have a negative result. Extending this idea to machines in general means if a machine can learn to not perform an action, it saves energy it would otherwise be wasting or it refrains from performing an action that could cause the machine to break.

The largest significance of the approach taken is that this system is an example of a system that combines simple systems to obtain one with complicated behavior. This means the individual subsystems would not necessarily be presented at a conference by themselves, but combined they create something unique and powerful which could be presented. As synthetic characters are built to have more and more abilities, the need to keep the subsystems simple grows. Understanding the effects of combining many systems is not possible if the underlying systems are too complicated. This system combined subsystems from different characters and newly designed subsystems to obtain an entirely new system with new behavior. Being able to reuse and combine systems of previous characters makes the development of synthetic characters very useful and insightful to the development of large systems.
Chapter 5

Future work

Minimus T.O. Mouse was designed with the goal of creating an entity that could use affect in vocalizations to effectively interact in social situations with a human. Currently Minni only uses tone of voice to derive social information. To further develop her social skills she needs to be able to perceive other types of social information such as body stance and facial expression. Work in the field of machine vision could be combined with the current systems to allow this to happen. Combining body stance, facial expression, eye direction and tone of voice information would provide more robust classifications and better social behavior from synthetic characters.

In addition to learning more about humans, Minni needs to be able to learn about more actions. Instead of focusing on obtaining the cheese, to be a useful creature she should focus on something more pertinent to human daily life, such as remembering all your appointments or remembering where your important files are located. She could learn which tones correspond to losing a file, forgetting an appointment, having to pay bills, and then provide reminders or pointers in such situations.

Improvements that could be made to the current design of Minni include making the action system more adaptable. Two ways to do this are 1) change the structure of actions that are added when associations are formed and 2) allow the system to continue to evaluate associations and remove previously added actions that are no longer accurate. Currently, when an association is formed, the system adds startle actions to the main action group which trigger every time the percept from the association activates. These actions should instead be regular actions with high values that are based on how reliable the association is. Even after new actions have been added, the system should continues to update the statistics and evaluate the associations. If the associations change they should be reevaluated.
and actions that were added previously but are no longer very strong should be removed from the system. This would allow Minni to adapt to changes in user behavior.

The interface for interacting with Minni could be improved by borrowing parts of Max's interface. Having a physical cheese that a user can move around is much more intuitive than using a gamepad. Also, having the ability to know when people are in the area would be useful for giving Minni more social capabilities.

As Minni was designed because Max could use improvements and Minni could be improved from Max's abilities, combining the two characters is a definite direction for future research. The combined character would be able to learn more about the people interacting with it than Max does and have more complex emotions than Minni does, taking advantage of the strengths of each character.
References


