8-31-2020

Lip Synchronization for ECA Rendering with Self-Adjusted POMDP Policies

Tristan Szucs
University of Windsor

Lip Synchronization for ECA Rendering with Self-Adjusted POMDP Policies

By

Tristan Szucs

A Thesis
Submitted to the Faculty of Graduate Studies
through the School of Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science
at the University of Windsor

Windsor, Ontario, Canada

2020

Lip Synchronization for ECA Rendering with Self-Adjusted POMDP Policies

by

Tristan Szucs

APPROVED BY:

A. Hussein
Department of Mathematics and Statistics

A. Mukhopadhyay
School of Computer Science

X. Yuan, Advisor
School of Computer Science

June 25, 2020

DECLARATION OF ORIGINALITY

ABSTRACT

The recent advancements in virtual reality have allowed for the creation of autonomous agents to aid humans in the retrieval and processing of useful digital information or to aid humans in requesting tasks to be completed by these autonomous agents. Known as embodied conversational agents (ECA), these intelligent agents bridge the physical and virtual worlds by providing natural verbal and non-verbal forms of communication with the user. To provide a positive user experience, it is essential for an ECA not only to appear human-like but also correctly identify the user's intention so the ECA can correctly assist the user. This thesis continues the research done by our research group investigating the further improvement of POMDP-based dialogue management using machine learning on POMDP's belief state history. This thesis integrates a technique to match lip movements with the rendered ECA audio alongside the automatically selected emotion. Finally, this research conducts experiments using machine learning techniques to adjust POMDP policies and compare its effectiveness in terms of dialogue lengths and successful intention discovery rates.

DEDICATION

To my parents and brother

for your unconditional support

ACKNOWLEDGEMENTS

I would like to take this opportunity to thank my advisor Dr. Yuan for his support, wisdom, and guidance. He has supported me since I approached him regarding graduate school and helped make this thesis possible.

I would also like to thank Dr. Hussein and Dr. Mukhopadhyay for agreeing to be on my committee, for their feedback, and for taking this journey with me.

This work builds upon what Ruturaj accomplished, and he was kind enough to walk me through his thesis and answer my questions even after his graduation, helping to ensure that my thesis builds on his correctly. This aid is extremely appreciated.

To the wonderful people working in the Computer Science department, thank you for your aid in everything from solving registration issues to setting up my presentations, you are instrumental. The transition online was not easy, but you helped me through it.

To my friends and family who sat through presentation rehearsals or read my thesis, your aid helped make this work what it is today, and I am glad you were a part of this journey.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

LIST OF APPENDICES

LIST OF ABBREVIATIONS/SYMBOLS

AU – Action Unit

BSH - Belief State History

COCOM - Contextual Control Model

DWT – Discrete Wavelet Transformation

DM – Dialog Manager

ECA – Embodied Conversational Agent

FACS – Facial Action Coding System

HC – Hand-Crafted

HCI – Human Computer Interaction

MDP – Markov Decision Process

ML – Machine Learning

MOS – Mean Opinion Score

NN – Neural Network

NCP – Number of Change Points

OB – Organization Behaviour

POMDP - Partially Observable Markov Decision Process

RL – Reinforcement Learning

SDS - Spoken Dialogue System

SPL – Software Production Line

1. Introduction

Human computer interaction is typically quite different from human to human interaction. This gap is partly due to the increased number and variance of channels humans use to communicate with each other. In a typical conversation, a human will use voice inflections and body language in addition to the spoken words to convey their meaning. Whereas when communicating with a computer, these components have traditionally been lacking. Computers typically use graphics and written words to indicate their current state to the user. However, these indications can be foreign to many users who typically communicate primarily with humans and not with digital systems. Missing these additional communication features increases the chance of miscommunication due to the lack of additional signals.

Computers are also taking on roles that are traditionally done by humans as this automation increases the need for reliable communication between humans and computers.

For these reasons, *embodied conversational agents* (ECA) have been a subject of research. ECAs are intelligent, exhibit some human behaviours and use a "visual representation to reinforce the belief that it is a social entity" [1]. One such behaviour is using a *spoken dialogue system* (SDS) to maintain a realistic spoken conversation. An SDS uses speech for input and output while understanding the user and acting accordingly [2]. At the heart of an SDS system is a *Dialog Manager* (DM), which essentially decides what the next action to take based on the current state of the conversation [2].

One of the critical parts of ECAs is that they exhibit human and social characteristics. While this may seem to be unnecessary, it is essential because it changes how humans interpret and understand the information coming from the ECA. The field of Organizational Behavior (OB) grades communication channels based on their capacity to convey information using the richness scale which considers three factors: the number of cues provided (such as emotions, expressions, and actions), the immediacy of the conversation, and the amount of presence the channel provides [3]. Figure 1, which uses example channels from [3], shows that email, a text-only channel, is not as rich as a phone call, and likewise, a phone call is not as rich as face to face. Therefore, if the ECA moves towards the high end of the scale, it will appear to be a richer channel to the user being able to provide more information in a more timely manner than it would otherwise.

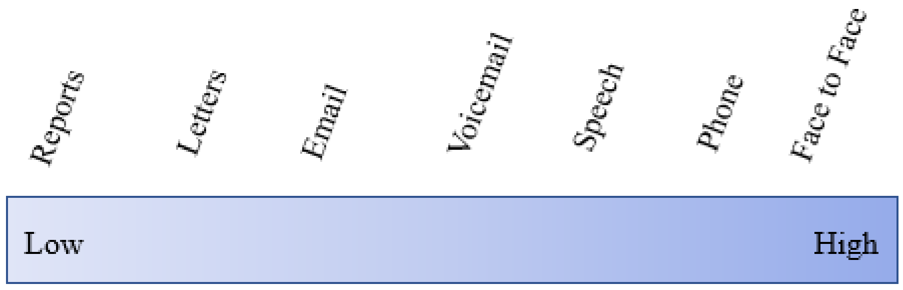


Figure 1 OB communication channel richness

It can be beneficial for an ECA to display emotional qualities as one of its features, as shown in [4], who attempted to use an emotional display to shorten conversation time while still obtaining the needed information from the user. By altering facial expressions, an ECA may demonstrate emotions [4]; however, the ECA's facial features do not attempt to lip-sync with the prescribed audio. Current literature has indicated that it is vital for ECAs to appear natural and use non-verbal cues as aids to create improved outcomes [5].

Text to speech applications are becoming more common in our society with multiple offerings commercially available including “Amazon Polly” and “Google Cloud Text-to-Speech”. Text to speech is the technology used to produce the audio of the ECA in [4], but it does not use a lip-syncing technique to match the expression to the speech. As claimed in [6], the “audience lose their concentration because the line of character and mouth shape does not match, and sometimes it interrupted empathy or absorption to the character”. By not breaking this rule, the proposed ECA will be more relatable.

The thesis will continue as follows: Section 2 will go over background knowledge of ECAs and related topics this thesis uses. Section 3 will go over the prior work that this thesis is built upon or were significant contributors to the topics needed for this thesis. Section 4 will explain the problem this thesis solves and the contributions this thesis has in solving this problem. Section 5 will explain the architecture and related algorithm this thesis proposes before Section 6 explains the experiment and related implementation used to demonstrate the proposed architecture and algorithm. This thesis outlines the created results and discusses them in Section 7, before making the final concluding remarks and discussing multiple future avenues for research in Section 8.

2. Literature Review

This chapter focuses on the previous work done in creating an emotional virtual avatar that can communicate conversationally and understand the user's intention while responding effectively to the user. Additionally, it will also analyze existing research regarding creating avatars that communicate with lip-syncing and realistic emotional voice generation.

2.1 Embodied Conversational Agents (ECA)

Embodied Conversational Agents (ECA) have been around for quite some time and is not strictly a computer science related ambition but can involve fields such as psychology, emotional theory, and human computer interaction (HCI) [7]. This section will review several of these components and areas that make up the methodology.

2.1.1 Sentiment Analysis of Inputs

When a user sends input to the system, they type in or say some arbitrary text. When the user does this, they may, intentionally or inadvertently, write their text in a way that indicates their opinion. While understanding the opinion behind the text is simple for humans, it is more complicated for a computer to do so. Sentiment analysis, also known as opinion mining, is the field of research that analyzes the opinion indicated through written text [8].

In [4], sentiment analysis detects the negative, neutral, or positive opinion the user is expressing in their text. The result is three floating-point numbers between zero and one, representing negative, neutral, and positive opinions present in the text. This sentiment analysis is a function returning a tuple of information, as shown in Eq. (1).

$$(positive, neutral, negative) = SA(input) \qquad (1)$$

Table 1 contains a series of sentiment analyses examples. Looking at Table 1, it appears that the sentiment cannot be a fact like ‘smelly’ but rather something like ‘bad’. One can also see that none of the examples are both negative and positive.

Sentence	**Negative**	**Neutral**	**Positive**
I am good today	0.0	0.408	0.592
I am sick today	0.623	0.377	0.0
I am going to the dentist today	0.0	1.0	0.0
I hate you	0.787	0.213	0.0
I love you	0.0	0.192	0.808
I work with you	0.0	1.0	0.0
The garden is pretty	0.0	0.484	0.516
The garden is smelly	0.0	1.0	0.0
The garden is outside	0.0	1.0	0.0
The garden is ugly	0.524	0.476	0.0
The garden is bad for the environment	0.368	0.632	0.0
The garden is dangerous	0.508	0.492	0.0
Which is better?	0.0	0.408	0.592

Which is worse?	0.608	0.392	0.0
It is sunny outside	0.0	0.517	0.483
It is gloomy outside	0.348	0.652	0.0

Table 1 Sentiment analysis examples

2.1.2 Decision Making with Dialogue Management

Researchers have been working on creating ECAs which communicate in conversational styles with humans for quite some time. These ECAs have a DM controlling their dialogue.

As noted in [2], a DM uses the current state to predict the best action to achieve the best end reward. There are multiple ways to perform this task, each with benefits and weaknesses. In [4], the DM used is the *Partially Observable Markov Decision Processes* (POMDP), which forms the DM for this thesis as well. This section will first provide a brief overview of these other methodologies before going a more in-depth overview of POMDP.

<u>Other DM Systems</u>

The most straightforward DM system is the finite state approach; this system uses a finite state machine to model the entire conversational state, including transitions [9]. Before the system can communicate, it requires modelling of every step of the conversation. Due to this limitation, it only works for short specific conversations that can be modelled. An example of this system is a bank call-in system; it expects precise answers and has the entire conversation laid out.

A more advanced approach is the frame based approach where the system uses a frame, which contains each data point similar to a form, and asks questions from the user to elicit all the information it needs from the user [2]. There are still disadvantages to this approach, notably that it can require a complex grammar to parse the needed information out of the user's response, and the frame itself can typically only handle fairly simple data points [2].

Building on top of the frame based approach is the information state approach. The information state approach is similar to the frame based approach. However, the system also keeps track of additional information that is not directly related to the frame, such as the user's mental state or intention [2]. However, each piece of data is still just a data point that can not be changed and is not probabilistic.

A different approach is a plan based approach in which the user and system have a shared goal, and both use their actions (speech) to progress towards the end state that they both desire [2]. However, it can be difficult to predict what a system using this approach will do [9].

Finally, there is the agent-based approach. In this approach, the system views the conversation as a collaborative process and has the added ability to revise previously captured information. However, it still relies on having the correct user intention [9].

POMDP

This section is an overview of how POMDP functions, the concepts described herein primarily come from [10].

Markov's decision process (MDP), forms the basis of POMDP, which consists of states and actions. States describe the current state that the system is in, and actions are the various things that the system can do – such as to say, "Would you like X". Each action then has effects that it may cause, and the immediate reward POMDP receives for performing the action. The reward determines if it was worth doing the action or not and is based upon what happened when the system performed the action.

However, unlike typical MDPs, the current state of POMDP is not fully known, and the system relies on observations to determine what the current state is. These observations do not specify an exact state but instead give us a probabilistic estimate on what the state is, from which the system can build a probability distribution over all the states using the available transitions and these observations. This probability is known as the belief state, and the history of the belief state is known as *Belief State History* (BSH).

The belief state is made up of probability state pairs where the system needs to identify which state the conversation is in currently. For example, based on the observations obtained from the user, the system may believe there is a 75% chance the user is in state A, and 30% in state B. These probability values and states together make up the belief state. When the system receives a new observation from the user, typically the input, the system will analyze the input to determine how it should adjust the belief states probabilities. For example, it might adjust the belief state to A:80% and B:40% if the input would increase both state probabilities.

To be precise, POMDP is a 7-tuple consisting of states, actions, transitions, rewards, observations, conditional observational probabilities, and a reward discount [4]. The states are the various states the POMDP could believe it is in at any moment. Actions are the available actions the POMDP has available to take while the transitions represent ways in which the states can transition from one state to the next. Rewards are the immediate reward given for the taken transition. Likewise, the observations are the observations the POMDP can observe to decide its belief state. Conditional observational probabilities represent how each observation affects the belief state. Finally, the reward discount determines if the POMDP system prioritizes immediate or future rewards.

Traditional POMDP systems do not maintain a BSH because by keeping the full belief state, one can use the transitions and conditional probability to move from one state to another without the BSH [10]. However, ECA research has shown that using BSH can be useful for other reasons [4].

When used in the context of dialogue management, Figure 2 shows how the POMDP works, emphasizing the dialogue loop. After the system performs an action by speaking, the user provides an input, which is the observation from the perspective of the system. Inside the system, it uses the input to update the belief state. Likewise, pass the last action back to the belief state calculator as it allows for the observations to be taken in context as they are a reply to the last action the system used. This belief state determines the next action taken.

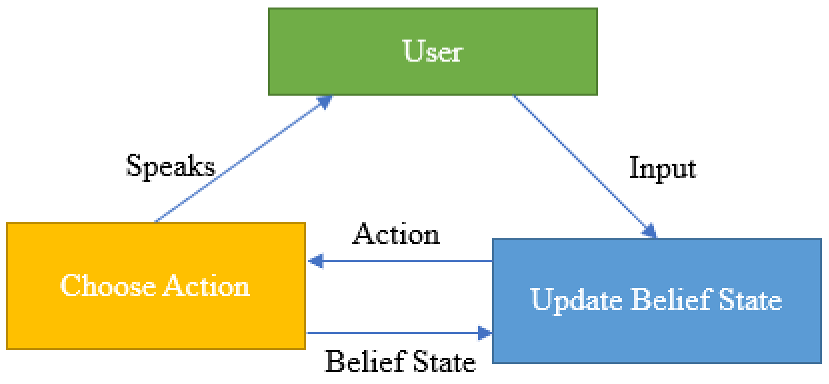


Figure 2 Simplified POMDP loop

Outside of ECAs, POMDP systems have a variety of uses, including optimizing the maintenance of offshore wind substructures [11] or improving pedestrian safety in self-driving cars [12].

One can illustrate how the reward discount can affect the decisions taken using the example of maintenance costs. For example, if the reward discount is low, future rewards, as in future cost savings, are barely taken into consideration. Thus since the immediate reward is high, it may ignore maintenance and take the higher hit later. Conversely, if the reward discount is high, then it will perform the maintenance now if it means a greater reward, or more significant cost savings, later. Finding a balance between these two opposing lines of thought is required to find an optimal way forward.

The system must use a policy to take the belief state and determine which action to perform next.

POMDP is related to other Markov models; Table 2 is based on [13] [14] and describes the relationship to other models.

	No Actions	**Yes Actions**
Known State	Markov Chain	MDP
Estimate State	Hidden Markov Model	POMDP

Table 2 Markov models compared

2.1.3 Wavelet Transform for Trend Analysis

While POMDB can be used to facilitate communication, it does not consider the history of the conversation using only the current state to predict and influence the next best action. Separate research has enabled these POMDB systems to look back and use trends in the conversation to influence the next action.

As "Sampling, Histogram, and Sketches are statistical-based approaches and are inadequate to predict the hidden patterns when applied to time-series data" [15] those techniques cannot be used on the BSH. Instead prior work has turned to wavelets which are a subset of wave transformation which uses "short windows at high frequencies and long windows at low frequencies" [15]. For this thesis, Discrete Wavelet Transformation (DWT) is of importance. DWT also removes the possibility noise in the trends meaning it will become more accurate then trend analysis techniques which do not remove this noise [4]. When used on a wave, DWT will result in the number of sharp variation points, which is the number of change points (NCP) [4]. DWT transforms the BSH wave to find the number of variation points [4].

Figure 3 shows examples of how DWT transforms a wave and acquires its variation points. The asterisk on the top graphs indicates where the trend changes based on the windows used. The lower graph then has an asterisk where the new wave crosses zero, showing how these zero-crossing points represent the change in trend. The left window has a larger window and so finds fewer variation points. While for this thesis, this figure is just an example of finding the sharp variation points on a larger graph, the data represents the vehicle count at an intersection over ten days at a 10-minute interval [16].

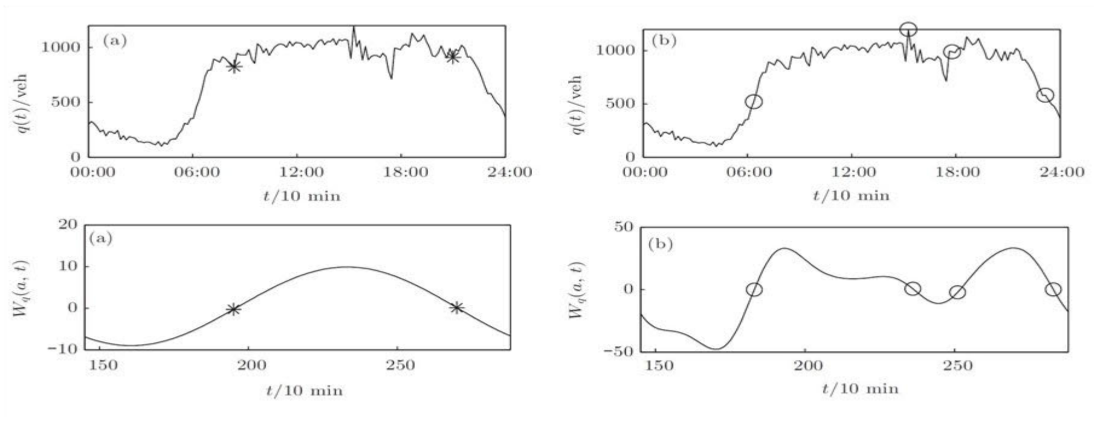


Figure 3 DWT wavelet example [16]

In order to apply DWT to the BSH, the system analyzes each intent individually. Each intent has a history of its probability, which forms a wave. Applying DWT to this wave will result in the number of variation points for this intent [17]. The summation of these numbers is the final number of NCPs.

For example, suppose the ECA is inquiring about the size of pizza requested. It has three intents i.e., small, medium, and large. Table 3 shows a fictitious conversation between this ECA and a human user. The user is not deeply knowledgeable on the sizes and gives contradictory indicators; this is where the NCPs will show up.

ECA Action	User Response	Belief after observation
“Hello, what size of pizza would you want?”	“I would like a king size pizza”	Small:0 Medium:0 Large:0.6
“That is not an option, please choose from small, medium, and large”	“I would like a medium pizza please”	Small:0 Medium:0.8 Large:0.2
“You would like an 8 inch medium pizza.”	“No, that is not big enough, I would like a large”	Small:0 Medium:0.4 Large:0.8
Are you sure you would like a large pizza?	“Yes, I would like a large pizza”	Small:0 Medium:0 Large:1.0

Table 3 DWT conversation example

The easiest intent to analyze is the *small* intent, never being indicated to the ECA as being wanted; thus, there will not be any NCP points. Therefore, this example will focus on medium and large. Figure 4 shows the values of their belief. The top charts are the belief values for medium and large, and the bottom two are their respective Harr wavelet transformations using the code provided in [18]. Note that the high and low points are not the same as the variation points; in fact, the zero-crossing points are not one of the discrete values entered. Instead, the variation points occur in places where the trend would be changing had these been complete waves. From the figure, there are two crossing points in each, making the total NCP value for this conversation four.

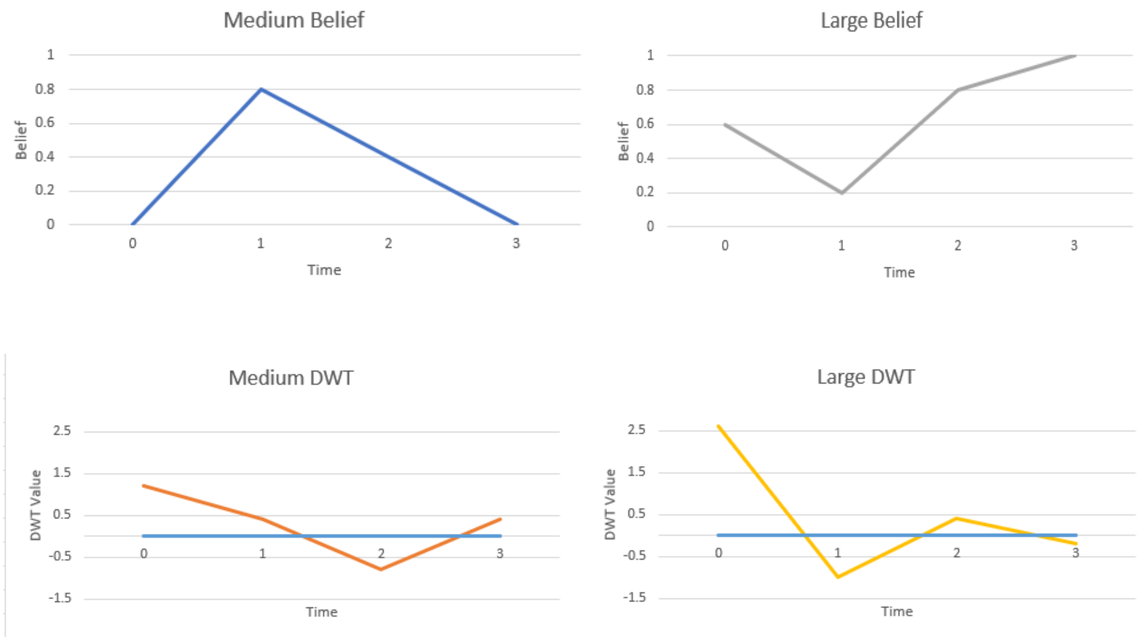


Figure 4 DWT conversation example waves

This analysis repeats at each step of the conversation, with the initial value being 0 as no waves exist.

2.1.4 Contextual Control Model (COCOM)

As described in [19], the COCOM consists of a series of modes that determine which action to take based on the belief state. COCOM prescribes four modes, strategic, tactical, opportunistic, and scrambled. These modes are chosen using the current state of the system and represent the level of control and predictability the system has. From most control to least the modes can be described as follows:

1. **Strategic mode** – strategic mode has the most level of control as the user and system are communicating well, which allows the system to plan the conversation’s long-term direction.
2. **Tactical mode** – in tactical mode, the system has some additional information to use, but it is not enough to plan beyond the current topic.
3. **Opportunistic mode** – in opportunistic mode, the system only has enough information to predict the next action as the system does not have enough information to plan further.
4. **Scrambled mode** – in scrambled mode, there is not even enough information to predict the next action well, and so the system's action is entirely random.

These modes are not permanent, and the system chooses every iteration which mode it should use. It can jump from one mode to another, and if the state suggests it is necessary can jump over the opportunistic mode when it is moving from one end of the control spectrum to the other. Figure 5 shows the available transitions for each mode.

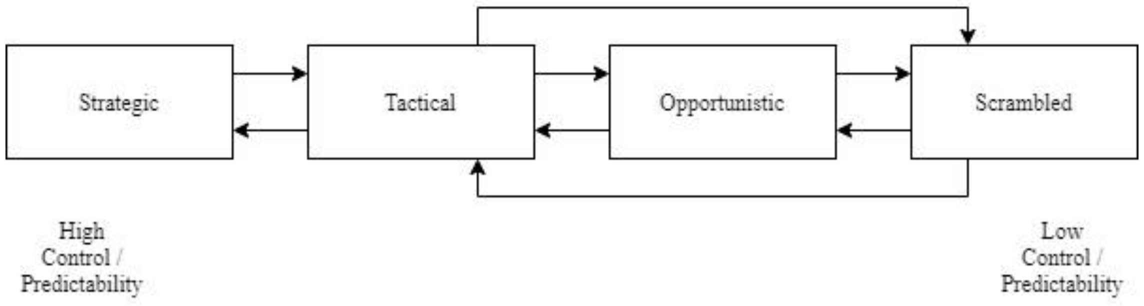


Figure 5 The available transitions in COCOM

2.1.5 Text-to-Speech

Text to speech systems or applications are applications which take in text and produce an audio version that sounds like someone is speaking. Commercial applications providing this technology include Amazon Polly and Google Cloud Text-to-speech. Some applications, such as Voicery, can do accents, sound like a variety of people, and even emulate multiple speech styles [20].

Being able to emulate multiple speech styles is vital for this application as having the speech style match the emotion displayed allows the user to be more immersed in the conversation instead of having a jarring experience. This capability increases the richness of the communication channel, a concept discussed in the introduction.

2.1.6 Facial Action Coding System

In the implementation of [4], facial expressions changed by swapping out 3-D models; but combining every facial expression with every lip shape will result in an untenable number of faces. Instead, this thesis uses the Facial Action Coding System, which describes the movement of various locations on the face [21]. There are well-known face movements which are known as Action Units (AU). These AUs are enumerated and represent various changes the face may go through, such as “Inner Brow Raiser” [22].

Additionally, AUs have an intensity value, as demonstrated in the implementation provided by [23]. To demonstrate changing intensity values, a series of images shown in Figure 6 contains increasing intensities. The starting face contains no action units, whereas the next has an intensity of 0.2 and the last 1.0 for the ‘ee’ phenome. Looking at the images, the middle image with an intensity of 0.2 is starting to show some teeth, but it is harder to see. The final image has the most teeth showing, as the mouth is the most in the ‘ee’ phenome.

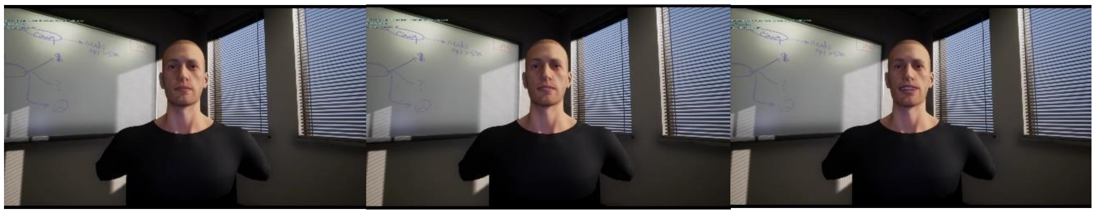


Figure 6 From left to right: normal, 0.2 ee, 1.0 ee

A list of AUs are in [24], but it also shows how combining some action units results in a range of emotions, demonstrating how the AUs can be combined to show emotions. These combinations are how this thesis combines AUs to make the various emotions; see Figure 7 for a demonstration of how AUs can be used to create a happy face. In the first image, the avatar only has its cheeks raised, in the second, only the smile is visible, and finally, the right image is the combination that looks like a realistic happy face. See Appendix A for photos of all six emotions this thesis’ avatar can simulate.

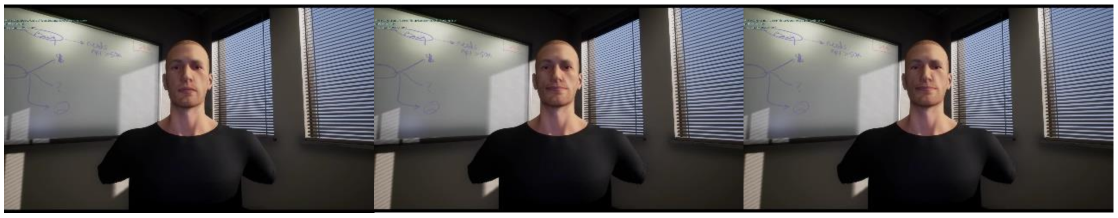


Figure 7 Left to right, cheek raiser (6), lip corner puller (12), happy (6+12)

A happy face is additionally an excellent example of a face that is a combination of two areas; the smile with the mouth, and the raised cheeks. The raised cheeks are typically a necessary part of a genuine smile [3], and so adding the raised cheeks is quite essential.

2.2 Related Topics

The architecture supporting the ECA relies on several topics that are not causally related to the ECA itself. This section will outline their functionality.

2.2.1 Q-Learning for Self-Learning of Policy

Reinforcement Learning (RL) is a form of Machine Learning (ML) where the system learns based on performing actions in an unknown space, observing a reward, and then use the observed reward to inform future action choices [25]. As claimed by [4], previous research using RL to improve dialogue managers has shown its potential.

Q-Learning is a specific RL algorithm that does not require a map of the domain in which it is learning [26]. In [4], Ruturaj suggests using Q-Learning as the RL method citing multiple benefits including its ability to start from a random policy to find the optimal policy, and that it can optimize over all steps conducted not just each single step. The following is a summary of how Q-Learning works, as defined in [26].

Q-Learning consists of states and actions. A state is the current state of the system. In contrast, an action is the various actions the system can take – note that each state does not correspond to an action – meaning that each action is not a simple transition. An action results in an unknown state and not a specified state. The matrix Q consists of each possible current state and action combination. The values contained in Q represent the potential reward by taking this action and are called q-values.

Immediately after taking an action, the system receives a reward value r – the immediate reward. This reward, along with the next best q-value, is used to adjust the previously used q-value. Only adjust the q-value used to choose an action immediately after the action has taken place. Equation (2) shows how to adjust said q-value. This equation is a simplified equation from [26].

$$Q(x,a) = (1-\alpha)Q(x,a) + \alpha(\, r + \Upsilon \max_{b}(\, Q\,(y,b)\,)\,) \tag{2}$$

In Eq. (2), α is the learning rate, a concept found in machine learning. X is the current state, and a is the last taken action. Y is the new state, and b is all actions. Finally, $\Upsilon$is the reward discount and is a constant on how much to consider a future reward.

The first half of the equation, which is the left-hand side of the main addition, refers to the amount of the new value given to the historical value. For example, if the learning rate is 0.8, then 20 percent of the old value is retained – this way, it does not learn solely from the new action but considers previous actions. The second half calculates the value learned due to the last action and represents the remaining 80% of the new value. The new part takes the reward immediately returned by the last action and adds a value to represent the future reward based on the new state observed after the action. This future reward is the best q-value in the newly observed state b. This future reward is tempered by the reward discount Υ, which determines the priority of immediate reward versus future reward.

Note that Eq. (2) is equivalent to the equation provided in [4] for this purpose.

In summary, the algorithm performs as follows:

1. Based on the current state's q-values, the system performs the action with the highest q-value.

2. The system now is in another, or possibly the same, state and receives a reward value *r*.
3. Adjust the previously used q-value based on the new state, best next action, and the reward received.

Figure 8 shows this cycle.

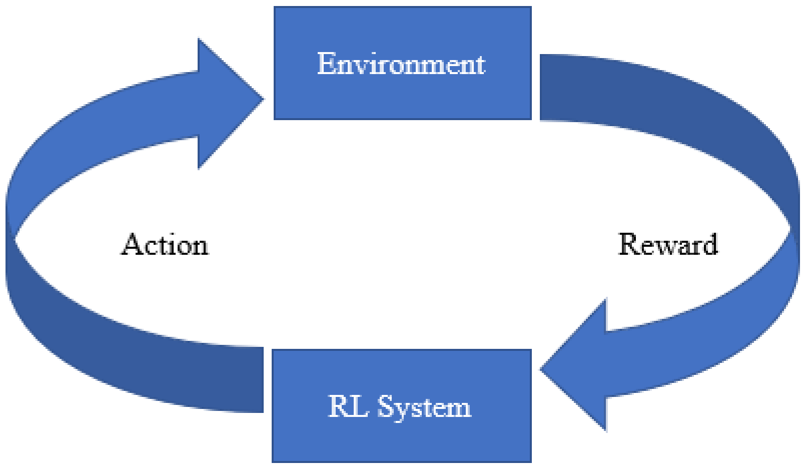


Figure 8 Q-learning cycle

For example, suppose a system has five states and three actions, and Table 4 contains related q-values. This example will use a reward discount of 0.5 and a learning rate of 0.8.

	Action 1	Action 2	Action 3
State 1	1	5	3
State 2	2	7	2
State 3	0	1	5
State 4	4	3	3
State 5	6	2	1

Table 4 Q-learning starting example

Suppose the system is in state 4. Looking at the table, the best action is action 1 as it has a q-value of 4, which is greater than actions 2 and 3. Therefore the system proceeds with action 1.

After doing action 1, the system receives a reward of 3 and upon introspection discovers it is now in state 3. Based on this information, the system uses Eq. (2) to update the q-value for state 4 action 1.

$$Q(4,1) = (1 - 0.8)Q(4,1) + 0.8(\ 3 +\ 0.5\ \max_{b}(\ Q\ (3, b)\ )\ )$$

The best action for state 3 is action 3, which has a q-value of 5.

$$Q(4,1) = (0.2)(4) + 0.8(\ 3 +\ 0.5\ (5)\ )$$

$$Q(4,1) = 5.2$$

Table 5 is the new Q-Value table note how state 4 action 1 is updated:

	Action 1	Action 2	Action 3
State 1	1	5	3
State 2	2	7	2
State 3	0	1	5
State 4	5.2	3	3
State 5	6	2	1

Table 5 Updated q-values example

This process repeats until the system reaches an end state and closes. The end state is not specified in q-learning but is determined externally.

2.2.2 Fuzzy Logic for Emotion Generation

According to [27], the term fuzzy logic was introduced in the late 1960s, referring to the processing of three or more truth values. Its definition expanded, and now it generally means logic where there is some uncertainty in regards to the truth of the values [27].

Ruturaj uses fuzzy logic to determine the emotion the ECA should express using a set of rules that use the current sentiment analysis and COCOM to determine the emotion to express [4].

As described in [4], fuzzy logic has three primary steps; first, the inputs pass through a fuzzifier, which creates fuzzy inputs. These fuzzy inputs pass through an inference engine; the inference engine uses a set of rules to transform the fuzzy inputs into fuzzy outputs again. Finally, the fuzzy outputs are passed through a defuzzifier to get crisp outputs; these are outputs that are output as the result of the fuzzy logic. Figure 9 shows the flow of information and terms.

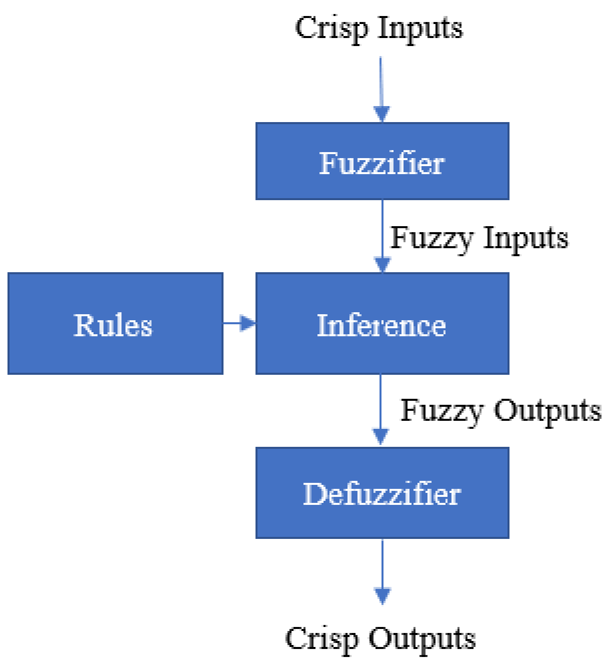


Figure 9 Fuzzy logic overview

2.2.3 Forced Phonetic Alignment for Lip Synchronization

The problem of mapping phonetic sounds to an audio file is not a new problem and is an issue studied in speech science known as the forced alignment problem [28]. Forced alignment aligns the given audio to its transcription typically with a Hidden Markov Model, finding the most likely set of states that are passed through at each point to align the two inputs [28].

As shown in [23], FAC AU can be used to change the avatar face to match each mouth shape needed for each phonetic sound correctly. This thesis uses the same system as [23], which uses a forced phonetic alignment system to acquire the lip-sync schedule.

3. Prior Work

Several primary components build largely upon research previously conducted at the University of Windsor and other research centers. This section analyzes previous research to identify their contributions and describes the inspirations taken.

3.1 POMDP-Based Dialogue Management

While many people have studied using POMDP to manage dialogue, this research is conducted to improve the intention discovery so that POMDP can decipher what the user wants quicker in order to serve the user better and provide an improved customer experience.

COCOM is not present in traditional POMDP but was added to the model by Sathulla in 2010 as a way to overcome some limitations in using POMDP for dialogue management [9]. In particular, as claimed in [9], COCOM overcomes limitations regarding dynamically solving the POMDP. Having multiple policies to solve based on the situation makes sense intuitively.

Additionally, in 2010, Bian proposed adding BSH to dialogue management POMDP [29]. Their research involved using the BSH to aid in determining the next action. It used the previous and current state to determine the next action, including using specific domain knowledge to determine if the change made sense and using a corrector action if it did not. As mentioned in Section 1, traditional POMDP does not use the BSH at all to improve its ability to conduct the conversation, and so this marked a change for POMDP dialogue systems. Using the BSH in this way would require domain-specific knowledge to be set up ahead of time, which may limit its applicability. It also does not look back very far to analyze the full history.

The COCOM proposed in [9] changed the mode based on the conversation, but Mulpur improved this using BSH [30] sort of combining Sathulla [9] and Bian's [29] works. In 2016, Mulpuri proposed the idea of using the BSH to calculate the NCP, which further identifies the type of user [30] interacting with the ECA. This user type then maps to a COCOM mode. Unlike Bian's work [29], the system does not look back to make the action but instead analyzes trends in the BSH to see long-term changes. This improvement enables their proposed POMDP method to take advantage of the full BSH. To demonstrate their work, Mulpuri used the example of requirements engineering to show improvement.

Ruturaj's 2019 thesis [4] supersedes Mulpuiri's work, where Ruturaj proposed using the same knowledge level, along with the sentiment analysis, to choose an emotion to display. Of importance to POMDP, however, he floated the idea of using q-learning to create self-adjusting policies. Despite this, his thesis does not contain implementation details or show weather or not these self-adjusting policies improve intention discovery or shorten dialogue length. The rest of Ruturaj's architecture serves as the basis of the proposed architecture, minus the contributions of this thesis.

Other research groups have investigated using self-refining policies, albeit in ways that are different from this thesis. In 2014 a group published evidence that using Gaussian processes can automate the creation of policies instead of using hand-crafted policies, which can take significant amounts of time [31]. Their policies could learn from human users and learned faster than previous dialogue system training. Their approach, however, does not use COCOM to aid in policy selection. It does, however, show that self-adjusting policies are worthwhile direction and have previously been investigated, albeit without other functionality, such as trend analysis or COCOM, which may have further helped the policies move the conversation forward. These additional functionalities distinguish this thesis from this work in a significant way, while still allowing this prior work to indicate the feasibility of this thesis contribution.

Additionally a group of researchers proposed in 2019 to add additional information to POMDP concerning the user [32]. This paper, while not heavily influencing the proposed architecture or method, does add weight to the feasibility of using reinforcement learning to enhance POMDP. The approach used in this paper is different from the one they used. Primarily, their approach adds the user information to the POMDP space instead of altering COCOM policies. However, their system did use reinforcement learning and a user emulator script to demonstrate that the conversation context matters, and that the system's learning improves other people's conversations as well, not just the ongoing conversation. Not only did they use a script, but it was tested with people, showing that their system was capable of communicating with both. While the contribution is different, many of the goals of this thesis are similar to the goals presented by [32].

The works that primarily contribute to this thesis are summarized in Table 6.

Author(s)	Title	Contribution / Key Points
Vijaya Krishna Mulpuri (2016) [30]	Trend Analysis of Belief-State History with Discrete Wavelet Transform for Improved Intention Discovery	Modelled how to use collect and use the BSH of POMDP. Showed how to use DWT on BSH to get the NCP of BSH. Created multiple POMDP policies for different knowledge levels based on COCOM.
Ruturaj Rajendrakumar RAVAL (2019) [4]	An Improved Approach of Intention Discovery with Machine Learning for POMDP-based Dialogue Management	Suggested using RL policies to create self-adjusting policies to replace handcrafted policies, did not propose an improved implementation.
Floris den Hengst, Mark Hoogendoorn, Frank van Harmelen, Joost Bosman (2019) [32]	Reinforcement Learning for Personalized Dialogue Management	Demonstrated a method of adding additional contextual information by extending the POMDP state space to include information about the users, which works without prior interactions with the user. This technique involved using RL to lead to more complex behaviours.

		Tested with a variety of people, Showed that RL could aid in positively altering the dialogue. Used a user simulator to simulate initial users

Table 6 Previously completed work on POMDP dialogue management

3.2 Facial Rendering with Emotions

It is not a new idea for ECAs to express emotions; there are multiple works in the past that work towards adding emotions to ECAs. This thesis draws from many of these works to create the proposed architecture and methodology.

Initially published in 1978, the facial action coding system is an important work that describes all the motions the face can make [33]. This work describes the various motions that will be needed to create emotions. As described in Subsection 2.1.6, the motions are specific and relatively simple. However, the system itself is quite powerful, being able to build complex expressions with ease and providing stable and precise definitions.

Multiple approaches to giving avatars emotions or expressive facial expressions have been proposed. One of the approaches suggested mapping “pleasure-displeasure, arousal-nonarousal and dominance-submissiveness” [34] to a facial expression. This approach uses multiple steps to create its faces and additionally controls head motions.

This approach does, however, need a head and facial motion database to build this mapping from, and, as it uses neural networks, will require a level of training. While this additional work creates improvements, the additional work and data required is a barrier to use.

Relating to ECAs, in 2016, Kaur [35] conducted a usability study to determine whether facial expressions representing emotions improve the user experience. This study indeed showed that this is the case; that changing the facial expression displayed on the ECA enhances the user experience.

This concept of emotions influencing behaviour has additional backing. A study done enhancing the smiles on virtual avatars in 2017 found that the enhanced smiles created an even more positive reaction than ordinary smiles [36]. More recently, a study conducted on humans indicates that avatar facial expressions do evoke a response in the end-user. However, they are not the same reactions as human facial expressions [37]. This result may mean that there are limits to virtual avatar facial expressions, but the article does state that the author does not believe that this means that emotions are useless. Combining the results from these studies, it is clear that emotions and their appearance do make an effect on the end-user, even if it is not the same as those caused by human facial expressions.

Ruturaj’s work [4] also contributed to the generation of ECA emotions. He proposed using fuzzy logic on the sentiment analysis from the user's input and the knowledge level to decide which emotion to display. This contribution would automate the process of selecting an emotion for the ECA to display.

Separately in 2019, a research group proposed [23] a paper describing how to create an avatar with fundamental emotions and lip-syncing. Their avatar was not only proposed, but the author's open-sourced some essential components, including code to lip-sync audio files to FAC's AUs using forced phonetic alignment. Their open-source repository does not, however, include code to combine emotion FACs with lip-syncing, which is something added by this thesis. This avatar is the basis of the avatar used in this thesis, and the use of forced phonetic alignment is also consistent. Their lip-syncing uses the last phonetic mouth shape to alter the current shape, which this thesis also uses. While not mentioned in their paper, this thesis included this implementation detail as the author believes that it will reduce the repetitiveness of the lip-syncing. This thesis uses a simplified implementation as [23] used a webcam to create their AUs and facial expressions, while this thesis uses only the standard FAC AUs.

Table 7 summarizes the most critical papers from this section, which contribute to this thesis.

Author(s)	**Title**	**Contribution**
Karamjeet Kaur (2016) [35]	An Approach of Facial Expression Modeling with Changing Trend in the History of Belief States	Showed using emotional facial expressions based on user input to have and show emotions on the ECA Conducted a usability study demonstrating how facial expressions enhance the user experience.
Ruturaj Rajendrakumar RAVAL (2019) [4]	An Improved Approach of Intention Discovery with Machine Learning for POMDP-based Dialogue Management	Use fuzzy logic to select a facial expression to have on the ECA automatically. The architecture, including POMDP and BSH components, forms the basis of this thesis' proposed architecture
Deepali Aneja, Daniel McDuff, Shital Shah (2019) [23]	A High-Fidelity Open Embodied Avatar with Lip Syncing and Expression Capabilities	Showed how their ECA was capable of basic actions and lip-syncing based on phenome control. Created an open-source avatar in

		Airsim The proposed avatar, including some components, form their open-source avatar, is the basis of the avatar used in this thesis.

Table 7 Previous work done on ECAs with emotionally expressive faces

3.3 Audio Rendering for Speech Generation

As noted, there exists multiple text to speech systems; however, there is ongoing research, with more modern systems using neural networks to convert their input and transform it into an audio output. Typically research in this field is moving towards improving how the synthesized audio sounds in particular situations such as towards creating an emotional version of the speech [38].

Expressive text-to-speech is not an area in which this thesis contributes; however, it is essential to take an overview of the current research in this area as it needs to co-operate with the other components. Text-to-speech has seen significant improvements in the last few years.

Recent advancements in this field use ML, such as Tacotron [39], which uses a neural network (NN) in an end to end manner which outperformed parametric models, but not concatenative, according to a mean opinion score (MOS). The MOS is a rating based on human subjects that provide their opinion on how realistic the synthesized voice is and is an effective way to compare different text-to-speech generators within the same experiment. Tacotron was succeeded in 2017 by Tacotron2 [40]. The results from these systems are awe-inspiring being difficult to distinguish from real speech, including ways to emphasize specific words. It does not, however, create expressive speech, creating only neutral speech and so would not be able to be used for this thesis.

Deep Voice 3 [41] in 2018 made a further advancement by introducing convolutional NN, which they claim allows for faster training than previous NN models.

In 2019 a separate group extended Tacotron2 by adding emotions to the voice [38] using what they call a global style token. This work would fit into what this thesis needs; however, it could not merely be used as the system was created for the Korean language.

Perhaps one of the more obvious comparisons to this thesis is ObamaNet [41], an end to end neural network which takes the provided text and creates a video of Obama speaking, including lip-syncing. The results are quite impressive, creating a video that is difficult to distinguish between the synthesized and real video. While this may appear to be a better way to perform an ECAs emotion, porting it over is not as simple as it may first appear. The problem is two-fold. First, ObamaNet does not convey emotions; all synthesized speech is neutral. Secondly, ECAs cannot start and stop; in between each phrase spoken, they must still appear to the user since ObamaNet makes a video the transition between videos would need to flow well and not be disconcerting. These two factors are non-trivial to overcome.

Additionally, there are multiple commercial expressive text-to-speech voice synthesizers, including Voicery [20], Amazon Polly, and Google Cloud text-to-speech. All of these works use similar input and outputs. The input is the text, and if applicable, some sort of expression or emotion identifier. The output is the synthesized audio in the form of an audio file. Therefore swapping out the various works would not be a significant hurdle.

This consistency in prior works allows this thesis to use a commercial system that could later be easily replaced by another improved system in the future without changes to the proposed architecture.

The most critical works looked at in making this decision are summarized in Table 8.

Author(s)	**Title**	**Contribution / Key Points**
Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J. Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Quoc Le, Yannis Agiomyrgiannakis, Rob Clark, Rif A. Saurous (2017) [39]	Tacotron: Towards End-To-End Speech Synthesis	Proposed an end to end neural network text to speech architecture. Outperformed parametric models but not concatenative.
Rithesh Kumar, Jose Sotelo, Kundan Kumar, Alexandre de Brébisson, Yoshua Bengio (2017) [41]	ObamaNet: Photo-Realistic Lip-Sync from Text	Proposed and demonstrated a ML model that can take text and produce a video of Obama saying the text. Does not directly fit

		into an ECA as an ECA has gaps in speaking and does not have all text upfront. The system does not produce emotional speech or video. Uses a module that predicts the mouth shape based on the audio as input.
Ohsung Kwon, Inseon Jang, Chung, Hyun Ahn, Hong-Goo Kang (2019) [38]	Emotional Speech Synthesis Based on Style Embedded Tacotron2 Framework	Showed how to use global style tokens with Tacotron to express emotions.

Table 8 Previous work done to render or create audio based on text

4. Problem Statement and Proposed Method

This section will first introduce the problems this thesis aims to solve. To highlight, these are: how to use self-adjusting policies, and how to add expressive voice with lip-syncing to the ECA.

To solve these problems, changes must occur to existing architectures and the conversational algorithm to accommodate these additional components. Once the problems to be solved are fully identified, this section will go over the proposed solution, including the required components previously proposed by [4], that will form the method proposed by this thesis.

4.1 Problem Statement and Contributions

This thesis aims to solve two problems currently existing in the current literature.

How the policies within COCOM are designed has been an issue under research. The issue is under research due to the length of time it may take to hand-craft optimal policies. As a solution, Ruturaj [4] proposed using RL in order to optimize policies to reduce dialogue length and increase accuracy. This improvement would mean that the time spent creating and optimizing policies by hand can instead be done autonomously as the system is used. However, Ruturaj [4] did not provide specifics on the implementation nor show its improvement. This thesis aims to solve this insufficiency by providing an architecture that uses RL to enhance policy selection and, through a larger experiment, shows its practicality.

Additionally, the most recent architecture proposed as part of [4] did not contain components for lip-syncing while maintaining the desired facial expression emotion or for additional action as desired by POMDP, such as nodding or shaking the ECA's head to further. This thesis will address this problem by adding to the architecture previously proposed in [4] to add-in components to add these missing interactive behaviour features. During the experiment phase, screen capture software records the ECA to ensure the system can provide these synchronizations correctly.

4.2 Overall Architecture

Below in Figure 10 is the overall architecture diagram for this methodology. In it, one can see the flow of information from one component to another. The architecture and figure are modified versions from [4].

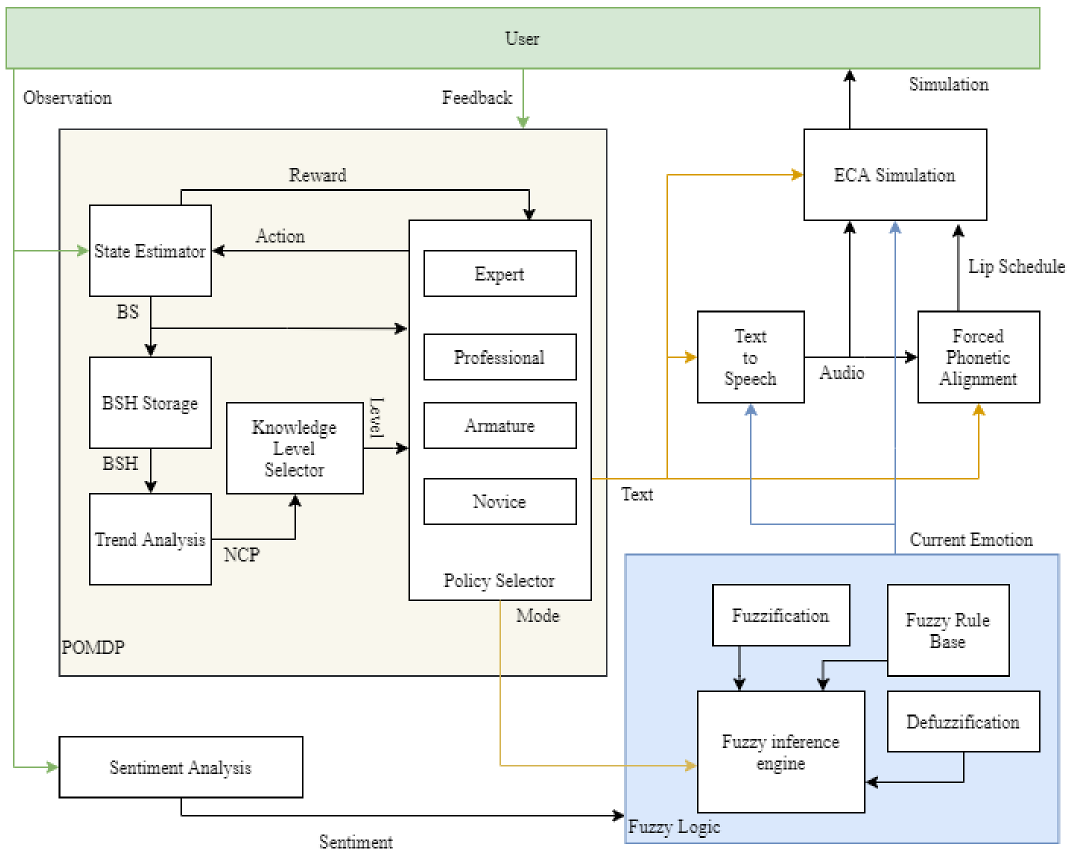


Figure 10 Architecture

In Figure 10, the observation the user provides is the text they input. This is passed into both the POMDP and the sentiment analysis systems.

Within the POMDP one can see how the observation goes into the state estimator, which produces both the reward used for q-learning and the new belief state. The belief state is passed into storage to maintain the history. The history is pulled from the storage system to perform the DWT trend analysis on. The number of sharp points becomes the NCP which is used to select the knowledge level. Within the policy selector the system will select the new policy to use based on the knowledge level. It also receives the reward from the state estimator which trains the last used policy. Finally this module outputs the mode and the best actions text to be used outside of the POMDP system.

The mode is combined in the bottom right portion of the diagram, where fuzzy logic is used to determine the emotion the ECA should display.

In the upper right the steps used to prepare to simulate the emotion and action on the ECA are performed. The emotion and text are combined in the text-to-speech system to create the audio. The audio is combined with the text to get the lip sync schedule.

The lip sync schedule, the audio, and the text are then all used to create the simulation of the ECA. The user than watches the ECA and based on it provides its next input, completing the conversational loop.

4.3 Algorithm

Table 9 outlines the algorithm related to the architecture of Figure 10. Sections of the table highlighted in red represent areas of contribution or improvement.

Algorithm		
Initialization	1	BSH = []
	2	endState = false; last_belief = 1; last_action = null; last_mode = null;
Get Input	3	While true
	4	Text = INPUT()
	5	If (Text == 'exit') break
Estimate Belief and get reward	6	b, reward = AvatarStateEstimator().process(Text, last_belief, last_action)
	7	Words = tokenize(Text)
	8	S = services_match(words) #compares to ontology services
	9	Bel = processBelief(S)
	10	Reward = calculateReward(last_belief, Bel) #based on the amount learned
	11	Return Bel, reward
NCP	12	Bsh.addHistory(b)
	13	Ncp = DWT (bsh)
Get Knowledge Level	14	knowl = AvatarKnowledgeLevelSelector(ncp)
	15	if(ncp < expert_max)
	16	return 'expert'
	17	if(ncp < professional_max)

	18	return ‘professional’
	19	if(ncp < amateur_max)
	20	return ‘amateur’
	21	return ‘novice
Get COCOM Mode	22	mode = modeSelector(knowl) #need to know the policy being used
	23	if(knowl == ‘expert’)
	24	return strategicpolicy()
	25	if(mode == ‘professional’)
	26	return tacticalPolicy()
	27	if(mode == ‘amateur’)
	28	return oppertounisticPolicy()
	29	return scrambledPolicy()
Teach the last mode	31	if(last_mode != null) last_mode.learn(reward, mode)
Get Action	31	action = mode.getAction(b) #This implementation is q-learning / hand crafted
Use sentiment and mode to get emotion	32	sentiment = getSentiment(text)
	33	emotion = avatarEmotionFuzzy(mode, Sentiment) #basically use the rules
	34	f_input = fuzzify(mode, Sentiment)
	35	f_output = apply_rules(f_input)

	36	return crisp(f_output)
Set Face	37	au = getActionUnit(emotion) #could be a combination
	38	eca.setFace(au)
Audio	39	voice_style = getStyleEmotion(emotion) #table lookup
	40	audio = textToSpeech(action.text, voice_style) #contact Voicery API service
ECA Speak / Lip Sync	41	eca.say(audio, action.text)
	42	phones = ps.decode(audio)
	43	foreach(frame in audio)
	44	au = getAU(frame.time, phones)
	45	return getAU (getPrevAU(frame.time, phones) * 0.1 +
	46	getCurAU(frame.time, phones))
	47	eca.mouthAction(au) # add the mouth shape to the emotion
	48	play(frame) #the audio
	49	wait(frame.length) #wait for this segment to play
	50	eca.setFace(emotion)
Carry Over	51	Last_belief = b; Last_mode = mode;

Table 9 Algorithm

4.4 Algorithm Details

The proposed method has a variety of steps and uses a variety of algorithms to accomplish its goal. This section will outline how they function in approximately the order that the systems perform them once implemented. This section will also flesh out some of the implementation details specific to the experiments.

4.4.1 Initialization

When starting, the system will begin at the empty belief state, the simulator will open, and the python code will connect to the simulator.

Once initialization is complete, the first conversation iteration will begin. To start the first iteration the user will type in their first piece of dialogue.

4.4.2 POMDP State Estimation

Upon receiving the dialogue, the POMDP will estimate the new belief state. To do so, it will parse out all the words entered and match them to the available services, then produce the belief state based on the matches and previous belief state. This state will add to the BSH.

4.4.3 Mode Selection with Trend Analysis

The system will perform DWT on the BSH to retrieve the NCP.

The knowledge level is selected based on the NCP based upon the limits provided for that experiment. An example set of limits is in Table 10.

NCP	**Knowledge Level**
NCP < 4	Expert
NCP < 7	Professional
NCP < 10	Amateur
Otherwise	Novice

Table 10 Example knowledge level thresholds

4.4.4 Self Adjustment of POMDP Policy

Based on the new state, the system receives a reward for moving into it. This reward is passed to the q-learning module so the system can perform RL to learn from this action. The new mode is also passed in so that the future reward comes from the new mode. The reward and the mode transition that occurred influences the next action selection using that mode. The addition of mode means that the system uses the following modified Q-Learning equation.

$$Q(x,a) = (1-\alpha)Q(x,a) + \alpha(\, r + \Upsilon \ \max_{b}(\, Q'\,(y,b)\ )\ ) \tag{3}$$

In Eq. (3), Q’ is the new mode – so that it considers the new mode’s future reward.

4.4.5 Emotion Selection with Sentiment Analysis

Based on the NCP, knowledge level, the Q-Learning module will choose the policy by which to choose the action.

4.4.6 ECA Rendering with Speech and Force Phonetic Alignment

Based upon the NCP, reward, and available rules, the system will use Fuzzy logic to determine the emotion the ECA should be expressing for this iteration. Table 11 displays the fuzzy rules which [4] provides.

	Policy			
Sentiment	**Strategic**	**Tactical**	**Opportunistic**	**Scrambled**
Negative	Disgust	Anger		Fear
Neutral	Fear	Sad	Surprise	Sad
Positive	Happy		Surprise	

Table 11 Fuzzy rules for emotion selection

The text to speech module uses the emotion and the action along with a text-to-speech module to create the required speech for the ECA to output. This module will use an expressive voice that matches the requested emotion.

Note that Voicey does not provide the exact emotions prescribed in Table 11, and so the system must translate to a Voicery emotion. Table 12 provides the translation this thesis uses.

Emotion	**Expressive**
Disgust	Flustered
Anger	Angry
Fear	Scared
Sad	Sad
Surprise	Flustered
Happy	Happy

Table 12 Translation to Voicery expressive voices

The forced phonetic alignment module receives the text from the action prescribed and the voice from Voicery. It returns a schedule of which phonetic syllable the ECA should simulate at what time in relation to the audio file provided.

Finally, the ECA uses the audio file, emotion, and lip schedule to simulate an avatar. The ECA will change its face based on the emotion. It will then simultaneously play the audio and alter the mouth to match the shapes specified by the lip schedule – this will mean the mouth syncs with the audio while the rest of the face is still appearing like the the specified emotion.

The ECA uses AUs to make up each facial expression, taking reference from [24] on how to create each emotion from the AUs.

Once complete, the system will return to the first step awaiting a response from the user. Looping in this way means the system continues until the system enters a belief state that terminates the system. If terminated, the system will print to the console information regarding the conversation needed for the experiment.

4.5 Running Examples

This algorithm has many components and can seem quite abstract. This thesis provides two examples to aid in the reader's understanding. One example is provided to aid in understanding how POMDP functions, and another which takes an emotion and text pair to describe how the ECA will simulate the combination. As these steps occur on a larger scale, such as for all intents or multiple audio frames within the same second, these examples will only consider a smaller subset of them.

4.5.1 POMDP Dialogue

These examples will cover the POMDP deciding which action to take, starting from input from the user up to deciding the emotion and action text. Thus this example represents lines 3 – 36, as shown in Table 9. The first of the two given examples assume that this is the first input from the user and will use the hand-crafted policies. The second will start in the middle of a conversation and use a q-learning policy. Note that they both will use fake information for the q-values and use a smaller set of intents.

Example 1 conversation start

This example starts at the start of the conversation and will use hand-crafted policies, which are in Appendix D: Hand Crafted Policies.

Suppose the user says, “I want security”. That text passes into the belief state calculator. The calculator will compare each word to the tags of the intents; in this case, only the low and high security intents are relevant. They both contain the tag ‘security’ within the user input, which significantly increases their belief. High contains the tag ‘secure’ and low ‘insecure’ as these words are like the word security but not exact. Secure being closer than insecure; these comparisons also influence the belief state. The high security value is 0.770, and the low security value being 0.700. These values represent the belief state which the system adds to the BSH. The reward, being based on the change in belief state, which started at all 0.5 to start, is 0.117.

Since this example is using handcrafted policies, no learning takes place.

As there is only one data point, the DWT analysis cannot detect any trends, and as such, the NCP value is zero. The zero is consistent with the idea of NCP as the user needs to change their mind, and they cannot do that in one input.

As the NCP value is zero, this is lower than the max for the expert user type; as such, at this point, the system decides they are an expert. Therefore, the system uses the strategic COCOM policy.

The simplified belief state is (0.770 + 0.700)/2 = 0.735. With the handcrafted strategic policy, this will be the guess action. As ‘high security’ is the highest belief, the action’s text will be ‘Do you mean With High Security?’.

Separately, the sentiment of the user's input was 0.0 negative, 0.0 neutral, 1.0 positive. The knowledge level of expert and the sentiment pass through the fuzzifier, which leaves expert and positive as the fuzzy inputs. Using Table 11 as inference rules, the system gets happy as the fuzzy output. In this application, the defuzzifier does not perform any functions, and happy becomes the final output.

Therefore, the ECA will simulate saying, ‘Do you mean With High Security?’ with happy emotion. The system will then wait until the next user input.

Example 2 mid-conversation q-learning

For this example, the system will start partway in the conversation after there have been three user inputs, and this is the fourth. This assumption means there are already three data points in the BSH. To keep this example simple, the user will be asking for the low security intent, and this example will only cover the low and high security intents.

Input	High Security	Low Security	NCP	Level	Reply
“I want security”	0.770	0.696	0	Expert	“Do you mean with high security”
“I just want it to be secure”	0.792	0	1	Expert	“Do you mean with high security”
“the min security”	0.202	0.732	5	Professional	“Please choose from the following: With High Security, With Low Security”

Table 13 Example conversation history

The last action used the tactical policy, and the action was action four. After the action, the system is in state five. Suppose the q-values at that point looked like:

State	Action 1	Action 2	Action 3	Action 4
1 = 0.0-0.1	1	8	7	8
2 = 0.1-0.2	2	5	25	6
3 = 0.2-0.3	5	5	8	8
4 = 0.3-0.4	3	6	5	1
5 = 0.4-0.5	4	4	6	8
6 = 0.5-0.6	8	2	4	5
7 = 0.6-0.7	5	2	1	3
8 = 0.7-0.8	1	1	5	9
9 = 0.8-0.9	8	4	2	6
10 = 0.9-1.0	3	1	2	5

Table 14 Example 2 initial q-values

Given that history suppose the user says, “I want low security”, a perfectly valid response. This observation will move the belief state to 0.708 for low security and 0 for high security, creating a reward of 0.242. The new simplified belief state is 0.708, as it is the only belief greater than zero.

The system will now calculate the NCPs. To do so, pass the BSH, including from this input, into the DWT to get the transformed wave. Then count each zero-crossing point to get the total NCPs. Figure 11 shows the waves made by the BSH, while Figure 12 is the resultant DWT wave.

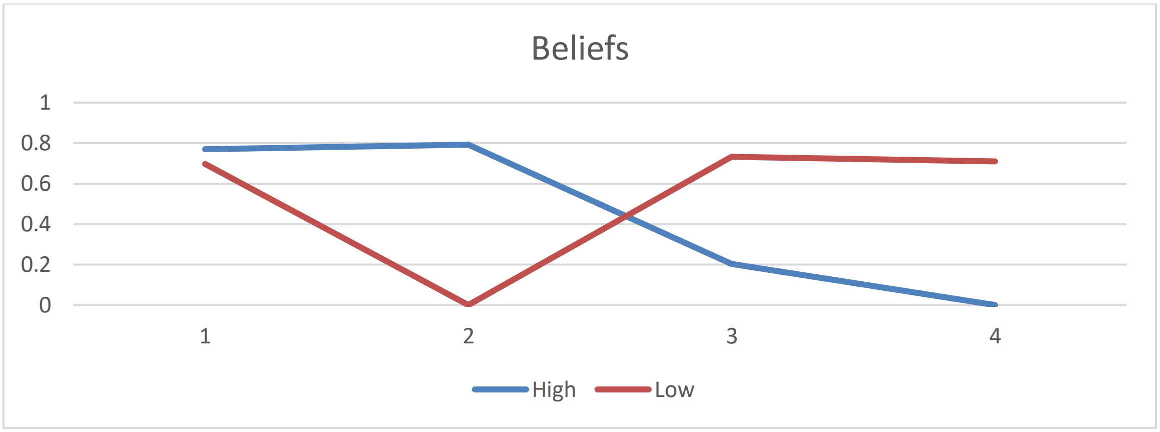


Figure 11 Belief state history example

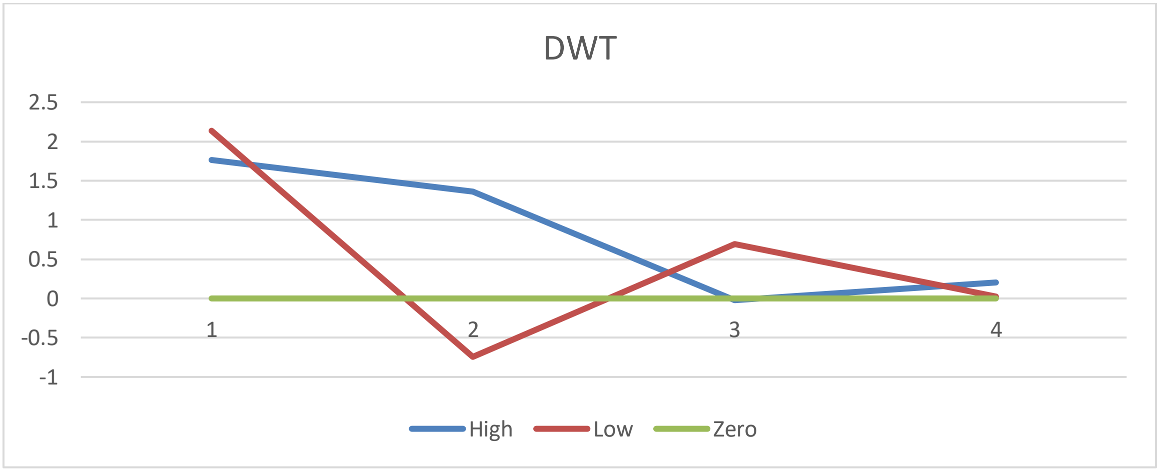


Figure 12 Transformed DWT example

The total number of zero-crossing points is four, which corresponds to the professional user. This user type corresponds to the tactical policy, and so the system will use the same q-values. Looking at Table 14, the best action for state 8 is again action 4 – the selection action. Therefore, the system will say, “Please choose from the following: With High Security”. While this response does not make much sense based on the belief states, as q-learning keeps going, this would ideally improve.

Now that the system has a reward and its new mode, it will update the tactical policies state five action four. This example will use a 0.5 learning rate and a 0.5 discount factor. Note that in this case, the mode did not change, and so Q = Q’ and only the one table is referenced.

$$Q(5,4) = (1-\alpha)Q(5,4) + \alpha(\, r + \Upsilon \max_{b}(\, Q'\,(y,b)\ )\ )$$

$$Q(5,4) = (1-0.5)8 + 0.5(\, 0.242 + \ 0.5 \max_{b}(\, Q'\,(8,b)\ )\ )$$

Using state eight’s q-values, the best q-value, representing future reward, is nine.

$$Q(5,4) = (1-0.5)8 + 0.5(\, 0.242 + \ 0.5\ (9)\ )$$

$$Q(5,4) = 4 + 0.5(\, 4.742\,)$$

$$Q(5,4) = 6.371$$

Therefore, the new state five q-values will look like:

State	Action 1	Action 2	Action 3	Action 4
5 = 0.4-0.5	4	4	6	6.371

Table 15 State five’s new q-values

The sentiment analysis on the user shows that their input of “I want low security” had a sentiment of negative 0.362, neutral: 0, and positive 0.638. This sentiment, in combination with the tactical policy, is passed through the fuzzifier of the fuzzy logic module. The fuzzifier transforms these into positive and tactical for the fuzzy inputs. The inference engine will use the inference rules presented in Table 11, creating a fuzzy output of happy. Happy is, therefore, the selected emotion.

Thus, the system will render “Please choose from the following: With High Security”, in a happy emotion. The system will then wait for input and repeat this process. The system will then train state eight action four based on the next reward.

4.5.2 ECA Simulation

The ECA simulation example covers lines 37 – 50 of the algorithms, as shown in Table 9. The example will start with the two inputs, the emotion and text the ECA will render. Let us assume it is saying, “can you please rephrase your request?” happily.

First, the system will get the AU for a happy face – which is a combination of cheek raiser, which does the eyes, and lip corner puller, which does the smile. Both are set to an intensity of 1.0 and sent to the ECA to render.

Next, the system chooses a Voicery style to use, in this case, happy. The module pulls from Table 12. The Voicery style, and the text, is sent to Voicery using their API, the result is the wave audio file of the voice.

The audio file and transcript pass through the forced phonetic alignment module, which will produce the alignment between the phonetics and the audio file. In this case, the alignment provided by PocketSphinx is in Figure 13.

```
[('SIL', 0, 0, 66), ('T', 0, 67, 72), ('F', 0, 73, 80), ('DH', 0, 81, 84), ('UH', 0,
85, 91), ('R', 0, 92, 101), ('UW', 0, 102, 113), ('ER', 0, 114, 117), ('B', 0,
118, 130), ('IH', 0, 131, 133), ('F', 0, 134, 137), ('W', 0, 138, 146), ('ER',
0, 147, 151), ('UW', 0, 152, 169), ('S', 0, 170, 184), ('G', 0, 185, 191),
('W', 0, 192, 200), ('AA', 0, 201, 207), ('F', 0, 208, 231), ('AA', 0, 232,
243), ('UW', 0, 244, 252), ('UW', 0, 253, 260), ('S', 0, 261, 274), ('HH', 0,
275, 278), ('UW', 0, 279, 287), ('AO', 0, 288, 307), ('AA', 0, 308, 312),
('B', 0, 313, 322), ('OW', 0, 323, 326), ('TH', 0, 327, 334), ('N', 0, 335,
338), ('AW', 0, 339, 345), ('ER', 0, 346, 362), ('OW', 0, 363, 367), ('AH',
0, 368, 370), ('S', 0, 371, 396), ('P', 0, 397, 401), ('S', 0, 402, 410), ('P', 0,
411, 418), ('SIL', 0, 419, 466), ('TH', 0, 467, 477)]
```

Figure 13 Forced phonetic alignment example

The audio splits into a series of small segments, which can also be called frames. Each segment is then outputted one at a time to an audio renderer. At the start of each segment rendering, the system computes a new face rendering using AUs and renders it. To calculate the face: first, zero out all mouth related AU; this will leave the cheeks raised but remove the smile. Now suppose it is at the related point of 195; the current phenome is the ‘w’ phenome, which relates to the phenome f AU. Therefore, set the f AU to an intensity of 0.5. The preceding phenome in the alignment is the g sound, which is the phenome g AU, and therefore is an intensity of 0.1. These two are combined with the raised cheeks and sent to the ECA and rendered. The audio will play, and the system will loop back to the start.

Note, if there were no previous phenome, there is only the one mouth shape with an intensity of 0.5.

Finally, now that the end of the audio file, the system takes the initial happy action units, that is, both the cheeks raised and the smile and sends them to the ECA again to render.

5. Implementation and Experiments

5.1 Software Customization: An Example

To demonstrate the abilities of ECA and intention discovery, this thesis has used software customization as the user's overall intention. The intent the system must discover is determining which feature they want. Software customization is a growing research area as software offerings include many features that users never use in with [42]. This issue is referenced by [42], which claims, "According to a survey, on average, more than 45% functions of a software product are never used by most users" [42]. It is further claimed by [42] that adding these unneeded and additional features can be harmful as it creates a more extensive application and can reduce reliability and security.

This thesis uses the same bookstore system used in [4] and [30] to provide its user intents.

The implementation can be used in three modes. One where the user emulator is used to where multiple conversations occur and summaries are outputted to the terminal. This first mode is useful for experiments and training the self-adjusting policies. Secondly, the user types in the terminal and the system returns the text of the action and emotion. Lastly, the user enters their text to the terminal, and the system renders the avatar to the user.

5.1.1 Interactive Feature Selection

Likely motivated at least in part by the above, software production lines (SPL) is an ideology towards software development that focuses on re-using parts of the software to create customized versions of the software [43]. Additionally, there is an approach to this problem using automated software, such as an ECA to elicit software requirements from users, so that developers can quickly use their SPL to create the customized software [43]. This methodology has the users pick from a variety of pre-defined features that make up the model of the entire software system.

Figure 14 contains the full ontology for this example. Figure 15 shows only the implemented portions.

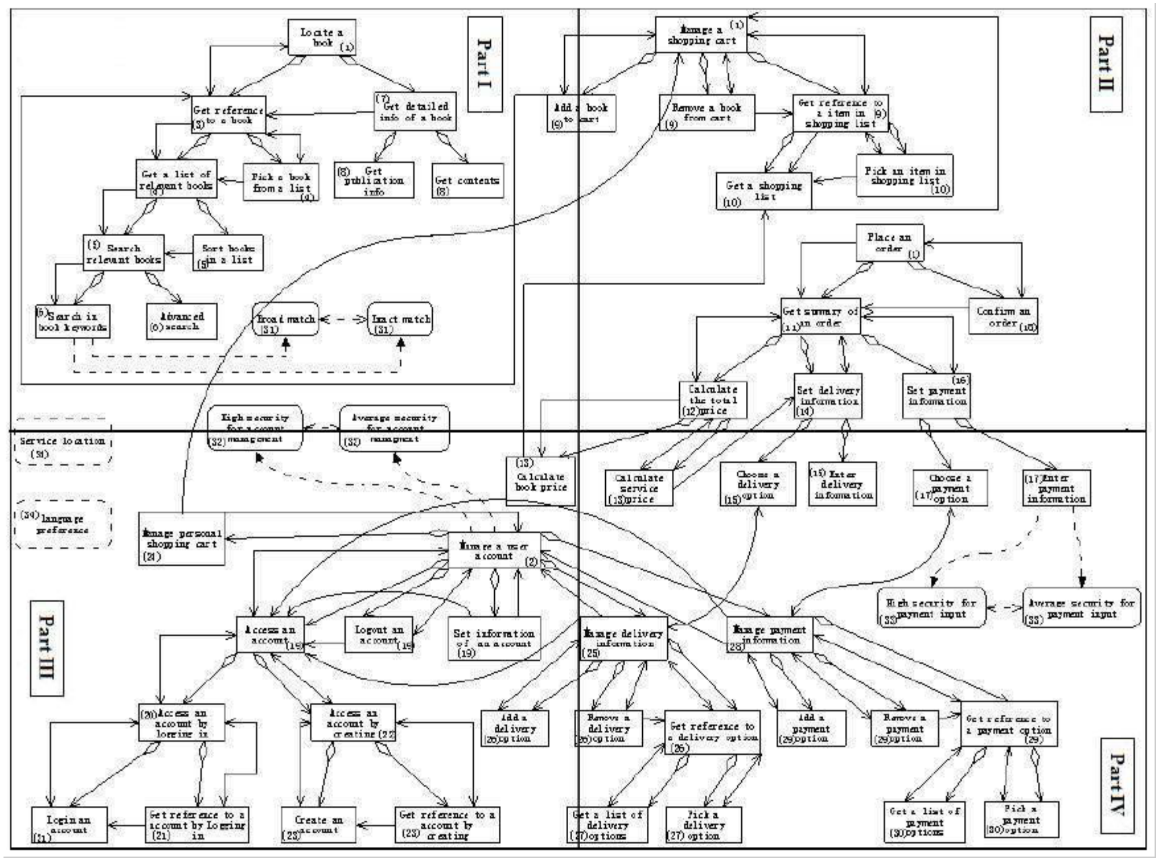

Part I
Part II
Part III
Part IV
Locate a book (1)
Get reference to a book (3)
Get detailed info of a book (7)
Get a list of relevant books (4)
Pick a book from a list (4)
Get publication info (8)
Get contents (8)
Search relevant books (5)
Sort books in a list (5)
Search in book keywords (6)
Advanced search (6)
Broad match (31)
Exact match (31)
Manage a shopping cart (1)
Add a book to cart (9)
Remove a book from cart (9)
Get reference to a item in shopping list (9)
Pick an item in shopping list (10)
Get a shopping list (10)
Place an order (1)
Get summary of an order (11)
Confirm an order (16)
Calculate the total price (12)
Set delivery information (14)
Set payment information (16)
Calculate book price (13)
Calculate service price (13)
Choose a delivery option (15)
Enter delivery information (15)
Choose a payment option (17)
Enter payment information (17)
High security for account management (32)
Average security for account management (32)
Service location (34)
language preference (34)
Manage personal shopping cart (21)
Manage a user account (2)
Access an account (19)
Logout an account (19)
Set information of an account (19)
Manage delivery information (25)
Manage payment information (28)
High security for payment input (33)
Average security for payment input (33)
Access an account by logging in (20)
Access an account by creating (22)
Add a delivery option (26)
Remove a delivery option (26)
Get reference to a delivery option (26)
Add a payment option (29)
Remove a payment option (29)
Get reference to a payment option (29)
Login an account (21)
Get reference to a account by Logging in (21)
Create an account (23)
Get reference to a account by creating (23)
Get a list of delivery options (27)
Pick a delivery option (27)
Get a list of payment options (30)
Pick a payment option (30)

Figure 14 Bookstore ontology [43]

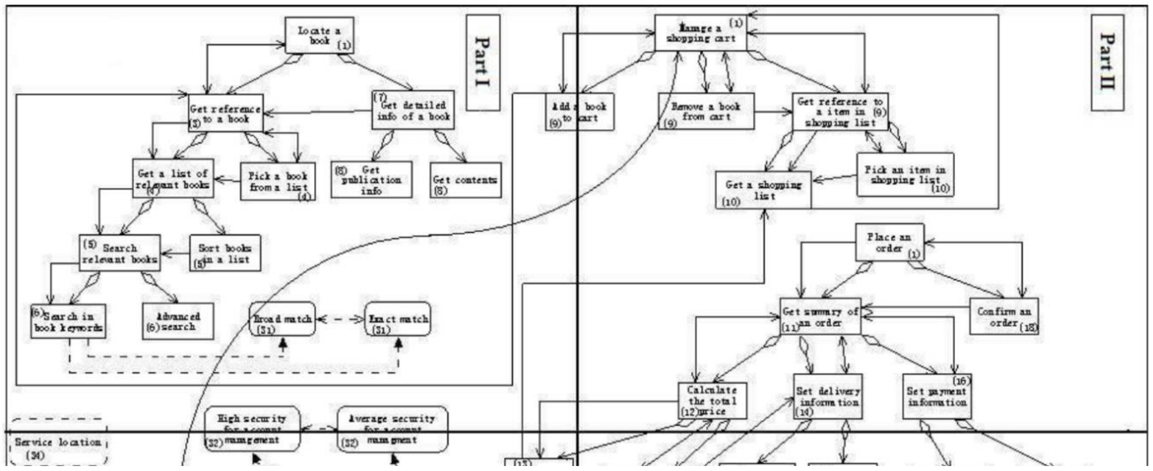

Part I
Part II
Locate a book (1)
Get reference to a book (3)
Get detailed info of a book (7)
Get a list of relevant books (4)
Pick a book from a list (4)
(8) Get publication info
Get contents (8)
(5) Search relevant books
Sort books in a list (5)
(6) Search in book keywords
Advanced (6) search
Broad match (31)
Exact match (31)
High security for account management (32)
Average security for account management (32)
Service location (30)
Manage a shopping cart (1)
Add a book to cart (9)
Remove a book from cart (9)
Get reference to a item in shopping list (9)
Pick an item in shopping list (10)
Get a shopping list (10)
Place an order (1)
Get summary of an order (11)
Confirm an order (15)
Calculate the total price (12)
Set delivery information (14)
Set payment information (16)

Figure 15 Planned ontology implementation [43]

5.1.2 Policy Selection and Dialogue Length

The system can respond to the user to determine what the user would like using both types of policies, the handcrafted policies, and the self-adjusting q-learning policies.

Because testing manually would take an excessive amount of time to perform both training and testing, a test script emulates users, including the intents and user types. It emulates the users so they will decide what to say based on what the system last outputted according to the user type and intent combination. It will then quit after speaking 100 times or when the user has a level of confidence that the system correctly knows what it wants, basing this confidence upon the system's past actions. All automated experiments use the same script to provide a level of consistency. When finished with a conversation, the user script will compare the intent the system believed it was, and the user script will output if it was correct or not, the dialogue length, and other pertinent information.

The experiments use four hand-crafted (HC) policies, one for novice, amateur, professional, and expert, which choose from 4 actions; these policies serve as the COCOM policies when using the HC policies. For machine learning, these policies consist of a q-learning system that has ten states and four actions.   The four actions available are the same for both HC policies and reinforcement learning policies, and the user emulator cannot tell them apart.

The first experiment will be to run the user script against the system, where the system uses only one of the HC policies for each policy. Next, perform the same experiment using a single RL policy – since all four machine learning policies are equivalent, there is no need to use all four. As q-learning tends towards the optimal policy [26], and the reward is based in part on dialogue length, it is expected that the single RL policy will be superior to the single HC policies showing an improvement.

As the NCP limits provided in [4] were from a conversational database, a slightly altered version of the tests to do a grid search on these limits, the best of which the more extensive test uses. The more extensive test increases the sample size, as the sample size for grid searches is less than the sample size used for the other experiments for the grid search to complete in a reasonable timeframe. Similar grid searches find the best results for both the HC and RL version of the policies. Similarly to the single policy experiments, as q-learning should bring us to the optimal policy for each COCOM mode, COCOM experiments are superior to using a single policy [9] it is expected that the RL COCOM experiment will show additional improvement.

5.1.3 Improvement of Interactivity

To demonstrate the lip-syncing and maintaining facial expression capabilities, screen recording technology will record several interactions with a live user. Over time varying emotions will be recorded in addition to the ECA's lip-syncing capabilities. The avatar will also be recorded going through all possible emotions to ensure there is an example for each.

It is expected that this thesis will create an improvement of interactivity over [4] as it did not contain any level of lip-syncing, this lack of lip-syncing can cause audiences to lose interest [6]. Additionally, not only will the expressive voice enhance emotions, which enhances the user experience [35], but it will make the communication channel richer as it has additional social cues. For these reasons, this thesis expects that the proposed system will have an improvement of interactivity when compared to previous systems.

5.2 Evidence of Contributions

This section will outline the outcomes that this thesis expects to occur based on other literature and the correct usage of the proposed method.

5.2.1 Shortened Dialogue Length

Based on the experiment in 5.5.2, this thesis expects to see that the average dialogue length is shorter using the new methodology for choosing the policy to be used. This result will demonstrate that using the self-adjusting policies results in shorter conversations and, as such, has an improved user experience. Furthermore, the rate at which the system discovers the user's intent has increased, thus an increase in the intention discovery rate.

5.2.2 Improved Success Rate (Intention Discovery)

Likewise, the results from the experiment in 5.5.2 should show that the intention discovery has improved, as more of the conversations should perceive the correct intent. This change means that the intention discovery is more correct using the self-adjusting policies than the HC policies.

5.2.3 Improved ECA Interactivity

Unfortunately, without doing a study with human subjects to obtain a mean opinion score (MOS), one cannot quantitatively state the change in interactivity from a voice and facial expressions perspective. Therefore, the author alone will pass judgement on the change of interactivity, which will form the basis of the results of this improvement.

This thesis expects that this implementation will improve the interactivity as the voice is emotional, the face is emotional, and the lip-syncing will match the speech. All of these features were not present in the implementation in [4], which forms the basis of the proposed architecture. Therefore, adding these features should improve the interactivity of the ECA.

5.3 Software Tools and Datasets

A software of this size uses multiple existing software systems so that not everything, including graphics, must be built from scratch. Table 16 contains the list of software and tools used to create the implementation and experiments.

Note that for text-to-speech, this thesis uses Voicery rather than a different published prior work. This thesis uses Voicery primarily for convenience, as it had a speaker that closely matched existing works in terms of emotions, and it means that the training of an

ML model is not required. Since text to speech methods are relatively similar to each other in terms of input and output, it would be simple to swap them out, might just need to change audio formats.

Function	**Software**
Programming Language	Python
Simulation (for avatar)	AirSim (Talking Heads)
Phonetic Alignment	PocketSphinx
For analyzing previous android application	Android Studio
Text to Speech	Voicery
Code Editor (Python)	Notepad++
Code Editor (Android)	Android Studio
Data analysis and Graph Creation	Microsoft Excel

Table 16 List of software tools and datasets used

5.4 Implementation Details

This section will outline how the proposed experiment was implemented, on a simplified level, taking care to describe the steps taken to ensure the created quantitative results are defensible. This section will skip some components of the algorithm when the algorithm is trivial such as an if-else clause.

5.4.1 POMDP

Some of the components in the Android application in [4] are related to this implementation. The author converted these related components into Python. This thesis chooses to use Python to implement its application, as it is a simple language to use, and the Android application had stability issues, randomly crashing. In contrast, the Python system encountered less fatal errors. As the overall system was present, but not all, the author also made many changes and added additional features to create the new implementation.

The POMDP component is central to this system, and so how it works is especially vital to defending the results. POMDP is aware of the series off intents created; each intent has a current belief value and a series of good tags and negative tags. Each good tag is a word related to the intent. For example, for the ‘add to cart’ intent ‘cart’ and ‘add’ are both good tags. The bad tags then are words that, when provided, mean that the intent is less likely to be the user's intent. For example, saying ‘high’ when asking for low security. Using the ‘add to cart’ example, ‘remove’ would be a negative tag. Each inputted word is compared to each tag, considering some variation of spelling and alters the belief for that intent. The belief state changes by considering multiple attributes, including negative and positive tags, the textual difference to the tags, and the parent intent's tags if applicable. This section omits the details from here as they are not needed to understand the overall POMDP. Right after calculating the new belief, it adds to that intent’s belief history.

The reward returned by the change in belief state is based on the increase of the average non-zero belief divided by the number of non-zero intents. This reward formula, presented in Eq. (4), is inspired by the idea that the system wants to head towards having a single intent with a high belief. This reward trains the q-values for q-learning.

$$reward = \frac{average(B) - average(B')}{|B|} \tag{4}$$

$$where$$

$$B = \{b \mid b \neq 0 \ and \ b \in I \} \ and \ I \ is \ all \ intent's \ beliefs$$

$$B' = \{b \mid b \neq 0 \ and \ b \in I' \} \ and \ I' \ is \ all \ intent's \ previous \ beliefs$$

5.4.2 BSH and DWT

To perform a BSH analysis, the Harr wavelet [18] is used on each intent's history to gain the transformed wave. However, as also noted in [4], the Harr wavelet transformation expects an input of length 2^n where n is an integer. As such, to gain more data points when needed, the system randomly chooses two points and adds a point that is the average of the two between the two. This sub-sampling allows for transforming any length of wave. Once capturing the new wave, each point is compared to the next to determine the number of zero-crossing points. This process repeats for each intent and sum the number of zero-crossing points to get the total.

5.4.3 Policies

Based on the mode the main script is in, the system will either select the policy from the selection of q-learning self-adjusting policies, or one of the handcrafted ones based on the knowledge level.

Both policies receive the same information, which is a simplified version of the belief state. This simplified version is the average of all belief states that do not have a non-zero belief.

While the primary system passes in a zero or a one to change the type of policy provided, this value not otherwise used. As such, the action, user emulator, and other components work the same regardless of the policy type.

HC Policies

The HC policies use the simplified belief state to choose the action based upon pre-chosen boundaries. The boundaries were created by thinking about what state the conversation is at if it is that belief state and what the system should say based on the user's knowledge level. For example, one of the actions confirms the maximum believed intent; the boundary where it will ask an expert user to do this is lower than a novice user.

After creating the HC policies, the author did not make further changes. By not changing these further, the author could not sabotage the HC policies had the expected results not came to fruition on their own.

Self-Adjusting Policies

The q-learning policies are similar but use q-learning to choose the action rather than using pre-selected boundaries. As the simplified state is between zero and one, it can evenly divide into ten sub-spaces, each of which correlates with a state. As there are four actions, this results in a ten state four action q-value matrix. All values are initialized to 0 and saved after each conversation.

All four policies are identical at the start but will diverge as training occurs. If the policy uses changes – the system will look ahead into the new policy to get the future reward value.

5.4.4 Actions

As mentioned, the system has four possible actions, each of which is a text template which may include details regarding the belief that needs to be passed to the user for it to determine its response. These actions are not changed between the two policy types or based on the belief state value, and so the other components of the system cannot tell if an HC policy or RL policy created the action. These actions are confirming the max belief, guessing the belief state, asking if the user means all the believed intents, and asking the user to rephrase. The exact templates are in Appendix C: Action Templates.

5.4.5 User Emulator

While this thesis refers to the user emulator as a script, it is one of the portions of the system. To allow for this, the implementation abstracts the input and output into interaction classes based on a value set in the main script. One is for text-only communication using the keyboard for testing; one enables the ECA; and one is the user emulator. In all three cases, the main script calls an input and output function to do the needed steps. The output function only receives the text and the emotion; the object interprets it and responds to the next request for input accordingly. In this way, the object is insulated and does not receive any additional information that the user would not know.

When initiation the user emulator object, the main method passes in whether it is training or not. If it is training, then the first input call returns the user type rather then standard input, and the main script sets the NCP value for the remainder of the conversation, thus training for this user type only. When training, the user output is not produced and thus cannot be a part of the results.

As the purpose of this thesis is to determine the q-learning improvements, and the emulator cannot consider the interactivity of the ECA, the user emulator does not consider the emotion when formulating a response. For performance reasons, the ECA does not run while using the user emulator.

When receiving output, the emulator looks for keywords to determine both its response and its belief in the system. For example, if the system confirms the correct intent, it is increasing its belief more than if it mentions the intent in the selected template.

Based on the user type, intent, and last template used, the emulator uses a hand-crafted dataset of options to reply. The dataset was tweaked based on success rates and NCP values to try and create a realistic dataset.

Note that while the template determines the response to the interaction object, the emulator parses which template it is using the last received output.

Once the user has spoken 100 times, or once the belief that the system is correct has reached a threshold, the user quits, ending the conversation. When exiting, the system returns the estimated intent to the user emulator. This is compared to the true intent which the emulator uses to output a conversation summary for analysis and returns a reward to the system to learn from using (5).

$$reward = \begin{cases} -1, if\ failed \\ \frac{0.7}{length}, if\ success \end{cases} \qquad (5)$$

5.4.6 Avatar

The implemented avatar uses the implementation of [23], where the avatar uses Airsim's Talking Head software. This is a change from [4], which provided the base of the proposed architecture. In [4], the avatar consisted of a 3-d model viewer, which would swap out the 3-d model to change the emotion. This method, however, has some shortcomings, primarily that the 3-d models did not have an easy way to alter them to change the expression besides swapping them out. This shortcoming means the system needs a 3-d model for every complete facial expression. As the proposed method combines not only every eye emotion with every phonetic sound combination but, in many cases, combines phonetic sound mouth shapes creating an infeasible number of possibilities to pre-draw and render. In contrast, basing the system on [23] provided a large amount of flexibility, allowing the system to render combinations of AUs without pre-created 3-d models.

5.5 Experiment Details

After implementing this system, it ran multiple experiments to get the results that could prove the desired contributions.

5.5.1 Single Policies

This thesis ran a set of single policy experiments; these ignore NCP and knowledge levels and always use the same policy. For HC policies, this needs to be done for all four policies as they are different. For the RL policies, only one policy needs to be analyzed, as all four policies would come to the same result. For each HC policy, 8000 conversations occurred. For the single RL policy, 2000 conversations were used for training, followed by 4000 conversations for testing.

5.5.2 NCP Grid Searches

As NCPs in a conversation would likely change based upon the NCP values and actions of the system, it was felt that a better approach would be to use grid searches to find optimal knowledge level boundaries. While grid searches take a significant amount of computing time, this method would result in the best overall result.

For HC, the final grid search ran with 4000 conversations at each combination. As 4000 conversations is a substantial number, a more extensive test is not required. The best result came from using a max expert NCP of 5, max professional of 8, and a max amateur of 10.

For machine learning, a variety of searches were undertaken due to the increased number of hyperparameters. The last search used 4000 training samples where the system knew the user type, followed by 2000 samples where the system did not know the user type. 4000 test samples follow the training samples – enough to compare the result to the other experiments. The idea of the two-stage training is to bootstrap the training of each policy, then run some samples to fine-tune them. The increase in training samples is justified as the system is training four times as many q-values. During training, the learning rate was 0.8 and during test 0.4, with all q-learning reward discounts being set to 0.5. The final boundaries were a max expert NCP of 4, max professional of 8, and a max amateur of 10.

In order to determine the q-learning learning rate (α) and reward discount (Υ) other experiments were conducted using different values. For the learning rate, it was found that lowering the learning rate during testing had better results. For the discount rate, similar adjustments were made, including using different learning rates for each policy. However, the best result came from using the reward discount of 0.5.

6. Results and Discussions

Each conversation results in a status of success and a value of length. Success means whether at the end of the conversation the system had the highest belief matching the user intent of the test script. The length is how many times the user has to say something. It is also important to note that at a length of 101, the user effectively ‘gives up’ and ends the conversation. It is possible for the user to both ‘give up’ and for the conversation to be deemed a success, so long as the intent at the end is still correct.

6.1 Success Rates

The primary metric used to determine the success of the system is the success rate, which is the percentage of times the system correctly acquired the user’s intent. This metric includes times where the user gave up, hitting the maximum length of 101 statements, but at the end of the conversation, the system had the correct intent. To determine success, it does not matter if the intent was similar. If it was not the exact intent, the system failed to acquire it. For example, if the user wants to ‘search by author’ and the system believes they meant ‘search by title’ it is a fail that has the same weight as if the system thought the user wanted ‘low security’. Therefore, an improvement in the success rates indicates an improvement in the system.

Figure 16 shows the success rates of the experiments conducted. The ones beginning with HC are the hand-crafted single policies, the RL uses the self-adjusting polices, and COCOM experiments are the best grid search result. This graph shows the novice being the lowest HC policy, followed by amateur. The next four are very close in success rates, from professional being 57.36% and expert being 59.31%, a difference of less than two percent. Finally, the RL COCOM has the highest accuracy, with a 63.13 % accuracy rate. The full underlying data for this figure can is in Appendix B: Aggregated Results.

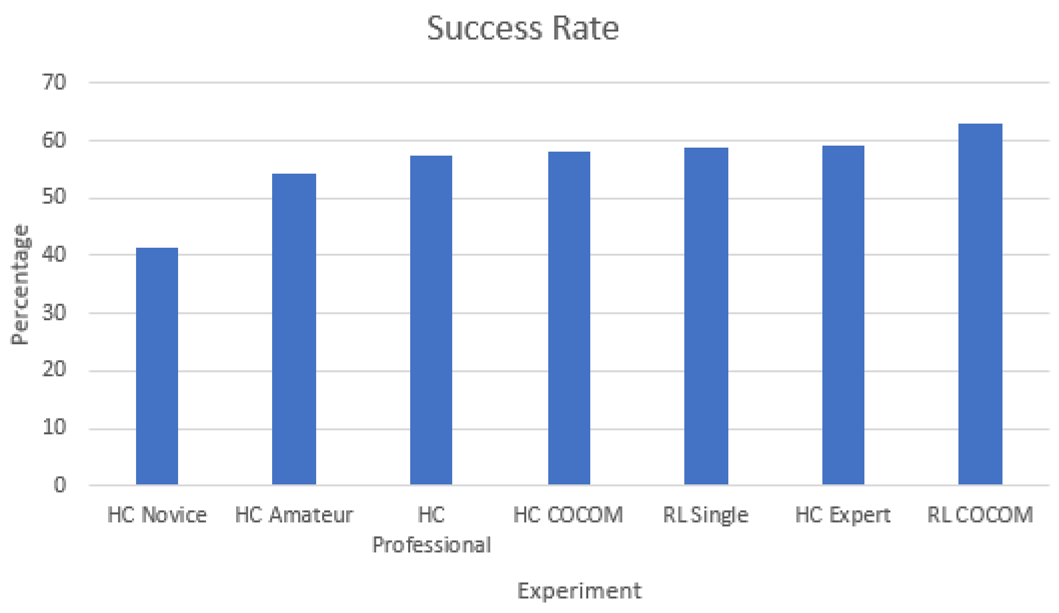


Figure 16 Success rate results

There are some additional points to note with these results. Primarily, that the RL Single experiment is below the HC Expert level, though the two are close. This discrepancy indicates that the HC expert policy is close to the optimal HC policy, and the RL single experiment brings us close to that single optimal policy. Additionally, the HC COCOM is below both – this is likely the novice or amateur policies bringing the overall score down. As proving COCOM is not a subject of this thesis, this thesis conducts no further investigation into this discrepancy. Regardless, these results show that a single self-adjusting policy comes close to optimal and that COCOM aids in creating a better overall experience and success rate for RL policies. This result is in line with what this thesis expected to happen.

6.2 Dialogue Length

A secondary metric used to determine the success of the new system is the length of successful conversations. A shorter conversation where the user does not have to interact as much with the system is an ideal system, as it provides a more positive user experience. However, it is more important for the conversation to be successful; therefore, this section will focus on these successful conversations. There are two categories of successful conversations – those where the user gives up and those where the user also feels confident that the system has the correct intent.

Figure 17 indicates the mean dialogue length for these experiments. A lower average is better than a higher one. This data shows that a single self-adjusting policy did the best, followed by the expert HC policy, then the HC COCOM, and finally, the self-adjusting COCOM. This discrepancy is not what this thesis expected to see; what was expected to be ideal is mediocre.

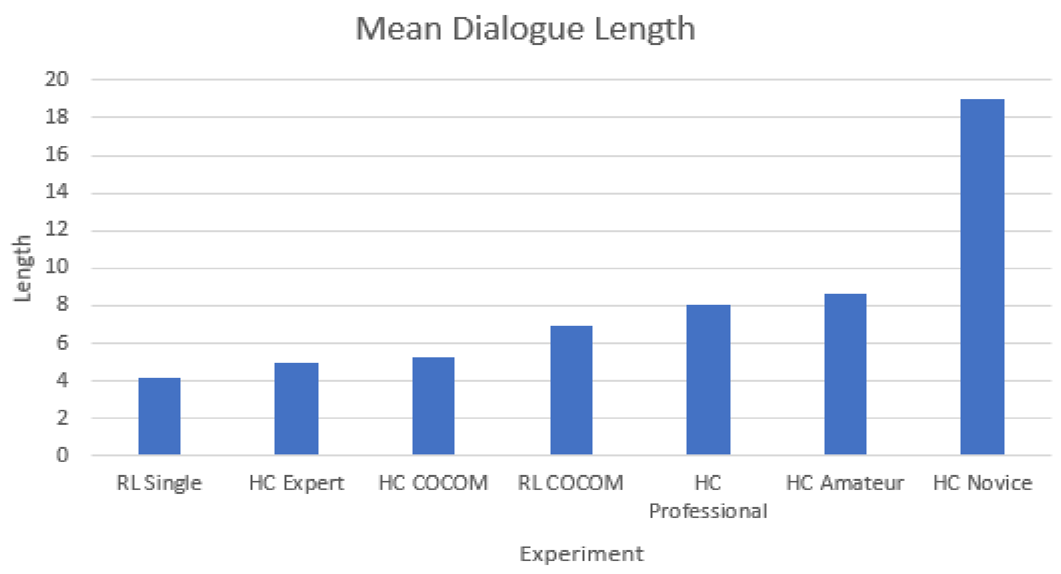


Figure 17 Mean dialogue length

The fact that the single self-adjusting policy performed better than the COCOM version is highly irregular – if the single policy was the better, all four should have gravitated towards it, and the results would have been similar. The fact that the single RL policy beat all others reinforces its place as the single optimal policy.

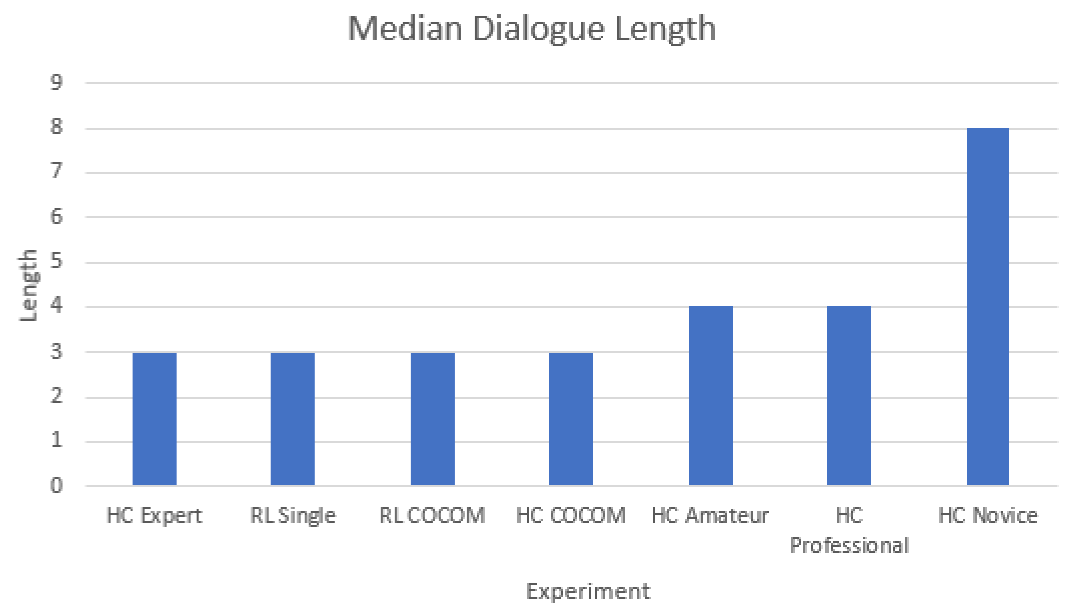


Figure 18 Median dialogue length

Looking at Figure 18, which shows the median dialogue lengths, the results are quite different. The RL Single, RL COCOM, HC COCOM, and HC Expert all have the same median dialogue length. Due to this similarity, it is possible that the RL COCOM's longer dialogues are longer, which may even be caused by additional success conversations. The result is ok because it is better for a conversation to be longer but right than to be wrong. It is also not difficult to see how some long conversations, which may go up to 100, would pull away from a median of 3 and skew the mean.

Looking at Figure 19, the percentage of dialogues ending under three increases for both the self-adjusting experiments. This increase lends further proof to the idea that it is lower with self-adjusting policies.

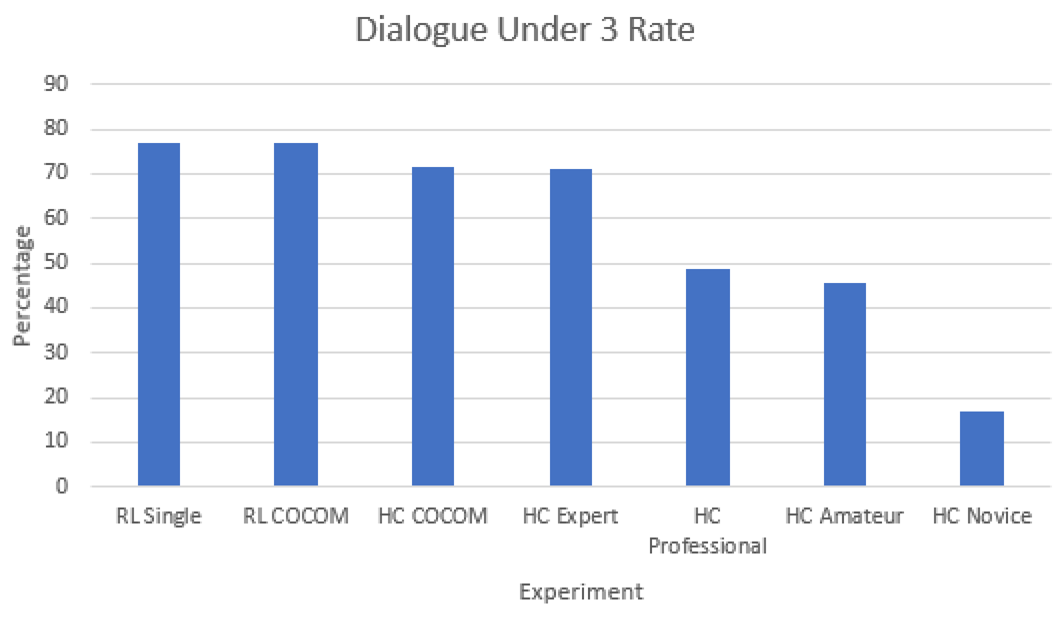


Figure 19 Percent under three

Together, this data shows an improvement in the dialogue length, while it was not an improvement in the mean dialogue length, there is evidence that improvement occurred.

6.3 Recordings

As it is difficult to rank the lip-syncing and overall user experience of the ECA quantitatively, instead several recordings were made to demonstrate the system's capabilities. These recordings have been made available through Google Drive at https://drive.google.com/drive/folders/1vlZXGNK_AY955obj5jN_VKXGA87ZEeZm?usp=sharing . In the author's opinion, the lip-syncing capabilities are excellent overall. While mixing the upper facial expression with the lips for each syllable works well, sometimes the switch back to the emotion on the lips can be jarring to see. This jarring feeling is likely because the change is sudden from having a reasonably neutral mouth throughout the sentence, it looks like the change is more sudden then having the emotion displayed on them throughout as a human would.

Some of the emotions also appear quite intense, which may lead to a more jarring user experience.

6.4 Limitations

While these results do show an improvement, this thesis needs to address some caveats. Primarily, these results came from using a user emulator script and not from using a human usability study. This limitation means that these improvements may not be the same as when applied to humans. The author believes that any system that would do this in a commercial context would likely need both more actions and a higher level grasp of the belief state, not to mention a more detailed POMDP, these are future research directions mentioned in 7.1. Secondly, the grid searches on additional hyper-parameters were limited in scope; they could have missed a result that would have shown an increased improvement.

The avatar itself was not tested on additional test subjects to create a MOS on its interactivity. Therefore, the objective analysis of concepts such as naturalness or realism are the opinions of the author and may not be the same when introduced to a broader audience.

7. Conclusion

In conclusion, this paper furthers the current research into ECAs by providing two contributions. First, this paper demonstrates using a quantitative experiment on how using RL, in the form of Q-Learning, to select the next action in place of each COCOM policy increases intention discovery and decreases dialogue length. The experiments used showed an increase in success rates along with an improvement in dialogue length and an increase in the user's ability to determine the intent of the ECA. All these results show an increase in the usability of the system. Secondly, this paper proposed an improved architecture so the ECA can provide expressive text to speech and lip synchronization. This thesis prototyped the proposed architecture to show its viability producing multiple videos to show others. These contributions improve the current literature surrounding ECAs and their respective capabilities in integrating these various components.

7.1 Future Research

One future work may be allowing for simple actions, such as head nodding or blinking, to be included in this architecture and controlled by the emotions or POMDP. The avatar can likely do this reasonably easily as the basis of the avatar [23], could blink or make other simple motions. As noted in [5], these non-verbal cues, such as gaze or actions, are essential in ECAs. Therefore, improvements in how the ECA acts, in addition to its emotional communication, would be a beneficial research route.

This thesis did not address many of the limitations in the architecture proposed by [4]. These limitations include the system not handling slang words well, and the sentiment analysis not handling some situations such as sarcasm.

This thesis' implementation of POMDP did not cover all features of POMDP; for example, our POMDP system ignores information based on the last action taken when analyzing the observations. Additionally, the system only has four available actions; an increase in the available actions would allow for more finely tuned policies. These steps would be necessary to deploy this system in a commercial environment.

While the Q-Learning policies introduced showed an improvement to the system, the q-learning state uses a simplified belief state, an increase in the available information may result in further improvement. Deep Q-Learning may be the answer as it is a type of Q-Learning that is capable of handling more complex tasks such as playing games [44]. Other reinforcement learning techniques may also provide further improvement.

In the research and discussion section, the author notes that some emotions are quite intense. A future direction may be to reduce this and make it appear more natural. As previously noted, AUs can vary in intensity, which is also a part of [23], which is the basis of this thesis' lip-syncing technique. Changing the intensity of some AUs or having it occur in a randomized range may result in a more natural set off facial expressions.

While the user emulator script shows a significant improvement indicating the utility of the system, it would be interesting to test on real-world users to see how well it fares with real users. Following this topic, the naturalness of the lip-syncing, or facial expressions was untested by others outside of the author. A further study focusing on usability may also determine if users find that it appears natural or very robotic to determine if further research is required.

The emotions fuzzy logic used herein was taken from [4], where the author arbitrarily decided which rules to use. While the oral defence related to [4] indicated these rules might depend on the domain, an exciting research direction could be on using additional RL to decide the emotion. Using RL in this way would be a difficult challenge as creating a user emulator would have more challenges involved, and so it is likely that real users would be required.

REFERENCES

[1] C. Pelachaud and Z. Ruttkay, From Brows to Trust: Evaluating Embodied Conversational Agents, Springer Netherlands., 2005.

[2] K. Jokinen and M. McTear, Spoken Dialogue Systems Synthesis Lectures on Human Language Technologies #5, Morgan & Claypool, 2005.

[3] N. Langton, S. P. Robbins, T. A. Judge and K. Breward, Organizationl Behaviour Concepts, Controversies, Applications, Toronto: Pearson, 2016.

[4] R. Raval, "An Improved Approach of Intention Discovery with Machine Learning for POMDP-Based Dialogue Management.," 2019.

[5] T. Pejsa, S. Andrist, M. Gleicher and B. Mutlu, "Gaze and Attention Management for Embodied," *ACM Transactions on Interactive Intelligent Systems,* vol. 5, no. 1, 2015.

[6] J. Cho, "Research on Animation Lip synchronization technology," *International Journal of Asia Digital Art & Design,* pp. 87-92, 2013.

[7] M. Lankes and R. Bernhaupt, "Using embodied conversational agents in video games to investigate emotional facial expressions," *Entertainment Computing,* vol. 2, no. 1, pp. 29-37, 2011.

[8] B. Liu, Sentiment Analysis and Opinion Mining, Chicago: Morgan & Claypool, 2012.

[9] S. Sathulla, "Four mode based dialogue management with modified POMDP model," 2010.

[10] A. R. Cassandra, "The POMDP Page," POMDP.org, [Online]. Available: http://pomdp.org/. [Accessed 21 2 2020].

[11] P. Gabriel, J. Nielsen and P. RIGO, "POMDP-Based Risk Maintenance Planning for Offshore Wind Substructures," 2016.

[12] F. Pusse and M. Klusch, "Hybrid Online POMDP Planning and Deep Reinforcement Learning for Safer Self-Driving Cars," *2019 IEEE Intelligent Vehicles Symposium (IV),* 2019.

[13] G. Hollinger, "Partially Observable Markov Decision Processes (POMDPs)," 2007. [Online]. Available: https://www.cs.cmu.edu/~ggordon/780-fall07/lectures/POMDP_lecture.pdf. [Accessed 5 March 2020].

[14] "POMDPs (= MDPs + HMMs)," 2010. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.187.7395&rep=rep1&type=pdf. [Accessed 5 March 2020].

[15] K. Shukla and A. K. Tiwari, Efficient Algorithms for Discrete Wavelet Transform: With Applications to Denoising and Fuzzy Inference Systems, London: Springer London, 2013.

[16] Z. Yong and L. Shigao, "Scale-free feature analysis of sudden changes of traffic flow," *Journal of Physics,* 2014.

[17] X. Yuan, M. Kaler and V. Mulpuri, "Personalized visualization based upon wavelet transform for interactive software customization," *Machine Learning and Data Mining in Pattern Recog-nition,* pp. 361-375, 2017.

[18] "Discrete wavelet transform," 7 May 2020. [Online]. Available: https://en.wikipedia.org/wiki/Discrete_wavelet_transform#Code_example.

[19] X. Yuan and R. Vijayarangan, "Emotion Animation of Embodied Conversational Agents with Contextual Control Model," in *2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing,* Beijing, China, 2013.

[20] "voicery," [Online]. Available: https://www.voicery.com/. [Accessed 21 2 2020].

[21] P. Bouissac, "Facial action coding system - Oxford Reference," Oxford Reference, 1998. [Online]. Available: https://www-oxfordreference-com.ledproxy2.uwindsor.ca/view/10.1093/acref/9780195120905.001.0001/acref-9780195120905-e-108. [Accessed 24 2 2020].

[22] P. Ekman and E. L. Rosenberg, What the face reveals: basic and applied studies of spontaneous expression using the facial action coding system (FACS), Oxford University Press, 2005.

[23] D. Aneja, D. McDuff and S. Shital, "A High-Fidelity Open Embodied Avatar with Lip Syncing and Expression Capabilities," in *2019 International Conference on Multimodal Interaction,* 2019.

[24] B. Farnsworth, "Facial Action Coding System (FACS) – A Visual Guidebook - IMotions," IMotions, 18 08 2019. [Online]. Available: https://imotions.com/blog/facial-action-coding-system/. [Accessed 24 2 2020].

[25] C. Szepesvári, Algorithms for Reinforcement Learning, Morgan & Claypool, 2010.

[26] C. J. Watkins and P. Dayan, "Technical Note Q-Learning," *Machine Learning,* vol. 8, no.

3, p. 279–29, 1992.

[27] V. Novak, I. Perfiljeva and J. Mockor, Mathematical Principles of Fuzzy Logic, 1999.

[28] J. Yuan, W. Lai, C. Cieri and M. Liberman, "Using Forced Alignment for Phonetics Research," 2018.

[29] L. Bian, "A modified approach of POMDP-based dialogue management," *Electronic Theses and Dissertations,* 2010.

[30] V. K. Mulpuri, "Trend Analysis of Belief-State History with Discrete Wavelet Transform for Improved Intention Discovery," *Electronic Theses and Dissertations,* 2016.

[31] M. Gasic and S. Young, "Gaussian Processes for POMDP-Based Dialogue Manager Optimization," *IEEE/ACM Transactions on Audio, Speech and Language Processing,* 2014.

[32] F. den Hengst, M. Hoogendoorn, F. van Harmelen and J. Bosman, "Reinforcement Learning for Personalized Dialogue Management," *arXiv,* 2019.

[33] K. Prkachin, "Facial action coding system - Oxford Reference," *Oxford Refernce.*

[34] J. Jia, Z. Wu, S. Zhang, H. M. Meng and L. Cai, "Head and facial gestures synthesis using PAD model for an expressive talking avatar," *Multimedia Tools and Applications,* vol. 73, no. 1, pp. 439-461, 2014.

[35] K. Karamjeet, "An Approach of Facial Expression Modeling with Changing Trend in the History of Belief States," *Electronic Theses and Dissertations,* 2016.

[36] S. Y. Oh, J. Bailenson, N. Krämer and B. Li, "Let the Avatar Brighten Your Smile: Effects of Enhancing Facial Expressions in Virtual Environments.," *PLoS ONE,* vol. 11, no. 9, 2016.

[37] L. C Kegel, P. Brugger, S. Frühholz, T. Grunwald, P. Hilfiker, O. Kohnen, M. L Loertscher, D. Mersch, A. Rey, T. Sollfrank, B. K Steiger, J. Sternagel, M. Weber and H. Jokeit, "Dynamic human and avatar facial expressions elicit differential brain responses," *Social Cognitive and Affective Neuroscience,* 2020.

[38] O. Kwon, . I. Jang, . C. Ahn and . H.-G. Kang, "Emotional Speech Synthesis Based on Style Embedded Tacotron2 Framework," in *2019 34th International Technical Conference on Circuits/Systems, Computers and Communications (ITC-CSCC),* JeJu, Korea (South), Korea (South), 2019.

[39] Y. Wang, R. Skerry-Ryan, D. Stanton, Y. Wu, R. J. Weiss, N. Jaitly, Z. Yang, Y. Xiao, Z. Chen, S. Bengio, Q. Le, Y. Agiomyrgiannakis, R. Clark and R. A. Saurous, "Tacotron: Towards End-to-End Speech Synthesis," *arXiv.org,* 2017.

[40] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerry-Ryan, R. A. Saurous, Y. Agiomyrgiannakis and Y. Wu, "Natural TTS Synthesis by Conditioning WaveNet on Mel Spectrogram Predictions," *arxiv,* p. 2017.

[41] R. Kumar , J. Sotelo , K. Kumar , A. d. Brebisson and Y. Bengio, "ObamaNet: Photo-realistic lip-sync from text," *arXiv,* 2017.

[42] Y. Jiang, C. Zhang, D. Wu and P. Liu, "Feature-Based Software Customization: Preliminary Analysis, Formalization, and Methods," in *2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE),* Orlando, FL, USA, 2016.

[43] X. Yuan and X. Zhang, "An ontology-based requirement modeling for interactive software customization," in *2015 IEEE International Model-Driven Requirements Engineering Workshop (MoDRE),* Ottawa, ON, Canada, 2015.

[44] S. Gu, T. Lillicrap, I. Sutskever and S. Levine, "Continuous Deep Q-Learning with Model-based Acceleration," 2016.

[45] Microsoft, "Home - Airsim," Microsoft, 2018. [Online]. Available: https://microsoft.github.io/AirSim/. [Accessed 22 2 2020].

[46] W. Ping, K. Peng, A. Gibiansky∗, S. O. Arık, A. Kannan, S. Narang, J. Raiman and J. Miller, "DEEP VOICE 3: SCALING TEXT-TO-SPEECH WITH," *arXiv,* 2018.

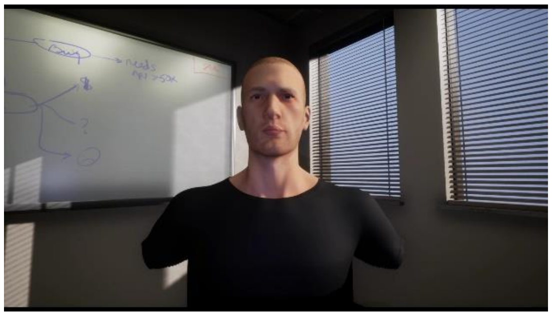
APPENDICES

Appendix A: Avatar Emotions

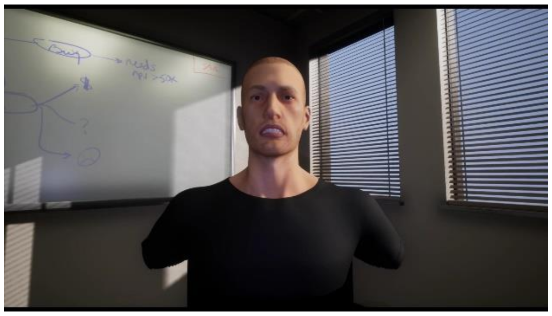
This appendix contains all of the emotions that the avatar can display; the avatar is housed in Airsim's Talking Heads [45]. [23] is the basis upon which many of the techniques used to create the avatar's interactivity. The AU combinations to create emotions are from [24].

The all_emotion video available on https://drive.google.com/drive/folders/1vlZXGNK_AY955obj5jN_VKXGA87ZEeZm?usp=sharing contains all emotions speaking.

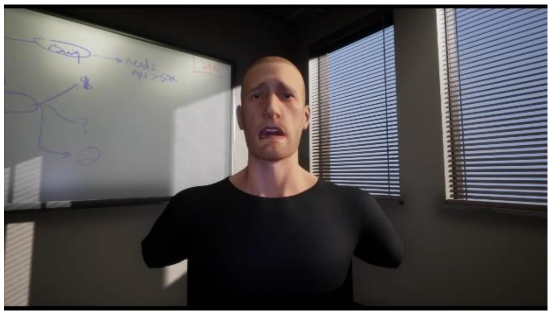
Anger



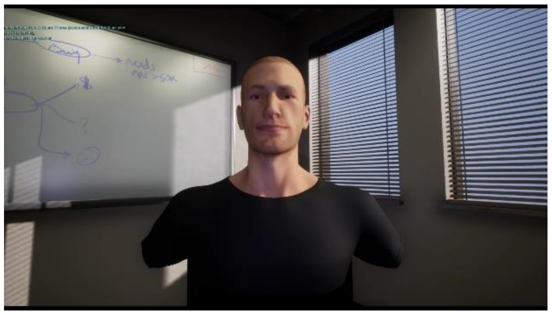
Disgust



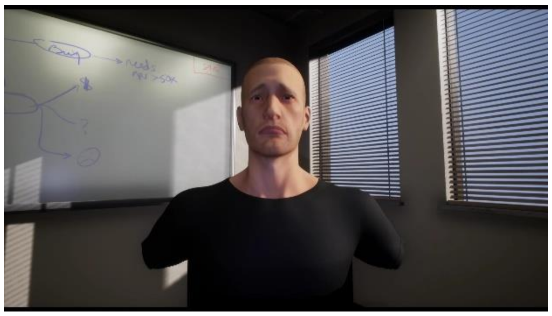
Fear



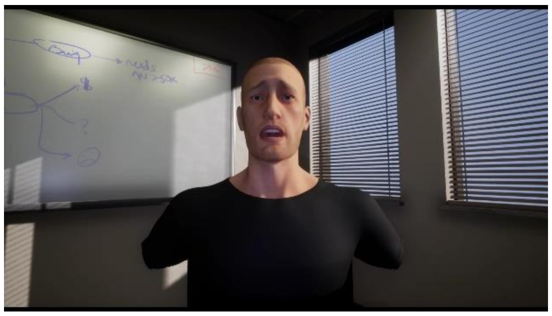
Happy



Sad



Surprise



Appendix B: Aggregated Results

Experiment	Success	Success Mean Length	Success Median Length	Success User Broke Mean Length	Success User Broke Median Length	Total	Total Success	3 and Under	3 and Under (%)
HC Novice	41.21	18.99	8	9.46	8	8000	3297	563	17.08
HC Amateur	54.41	8.63	4	8.46	4	8000	4353	1986	45.62
HC Professional	57.36	8	4	7.35	4	8000	4589	2238	48.77
HC Expert	59.31	4.93	3	4.81	3	8000	4745	3376	71.15
HC COCOM	58.2	5.24	3	4.56	3	4000	2328	1661	71.35
ML Single	58.8	4.12	3	4.64	3	4000	2352	1811	77.00
ML COCOM	63.13	6.97	3	6	3	4000	2525	1943	76.95

Table 17 Aggregated results

Appendix C: Action Templates

The system has four available actions to choose from. Note the following placeholders:

[MAX BELIEF] is the name of the belief with the maximum belief value.

[ALL BELIEFS] a list of the names of all intents that have a non-zero belief value.

Confirmation message template

"Sure, added the service [MAX BELIF]"

Example: "Sure, added the service high security"

Guess message template

"Do you mean [MAX BELIF]"

Example: "Do you mean high security"

Selection

"Please choose from the following: [ALL BELIEFS]"

Example: "Please choose from the following: high security, low security"

Rephrase

"Can you please rephrase your request?"

Appendix D: Hand Crafted Policies

Each of the hand-crafted policies was created by thinking of the user type in mind and when the simplified belief state should be for each action.   Each action has a minimum simplified belief state value. The higher policies are preferred, with strategic being highest. Note that when none of the three higher actions are available, the system uses the selection action. If the belief state is 0, meaning it has no idea, it will always ask to rephrase.

Policy	Confirm	Guess	Rephrase
Scrambled	0.95	0.85	0.70
Opportunistic	0.90	0.75	0.6
Tactical	0.85	0.75	0.50
Strategic	0.80	0.60	0.35

Table 18 Hand-crafted policies

VITA AUCTORIS

NAME:	Tristan Szucs
PLACE OF BIRTH:	Windsor, ON
YEAR OF BIRTH:	1996
EDUCATION:	St. Anne Catholic High School, Windsor, ON, 2014
	University of Windsor, B.Sc., Windsor, ON, 2019
	University of Windsor, M.Sc., Windsor, ON, 2020