PLANNING TO LISTEN : USING AUDIO FEEDBACK FOR COMPLEX MANIPULATION TASKS UNDER UNCERTAINTY

JIHIRSHU NARAYAN

A report submitted for the course
COMP 8755 : INDIVIDUAL COMPUTING PROJECT
Supervised by: DR. MARCUS HOERGER
DR. HANNA KURNIAWATI
The Australian National University

November 2020

Except where otherwise indicated, this report is my own original work.

JIHIRSHU NARAYAN
8 November 2020

Acknowledgments

First and foremost, I would like to thank Dr. Marcus Hoerger and Dr. Hanna Kurniawati for giving me the opportunity to work on this project. They have been extremely supportive throughout the entirety of this project. It was my honour and privilege to be working with them on such a fascinating project. I would also like to thank Jimy Cai, who provided me with rich insights whenever I was stuck on a problem. I also thank the entire RDL team for providing such an enjoyable working environment. In addition, I would like to thank Dr. Felipe Trevizan, who took time out of his busy schedule, and agreed to be the examiner for this project. Finally, I would like to thank my family who have been a pillar of support in my life as always.

Abstract

Robust manipulation is an essential feature for efficient operation of autonomous robots in industries where they are expected to interact with the environment. In certain situations, the robot is expected to plan its strategy based upon the object or properties of the object it is trying to manipulate. Some of the popular means for gaining such information have been observations from sensor modality such as vision or torque. In this project, we have implemented a manipulation strategy that is based solely on the audio feedback generated by the robot's interaction with the object. We aimed to build a model that could accommodate uncertainty in observation and object properties. We also aimed that the model would exhibit a certain level of situational awareness when deciding upon the sound profile which would maximize the information it was receiving. Our model performed consistently, exhibiting robust decision making under uncertainty in observation and object property.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

It is a popular belief amongst researchers that we are in the middle of the 4^{th} industrial revolution, which is being driven by large scale automation of industrial practices using smart technology [1]. The use of automated machinery in industries is not a novel idea. However, providing these automated entities with intelligent behaviour is considered as one of the key steps that will push our current technology well forward into the envisioned "Industry 4.0" [1]. In industries such as manufacturing and healthcare, one of the most sought after behaviour in autonomous robots is robust manipulation. An important aspect of such behaviour is the ability to quickly gain information about the target object and its properties. Researchers have traditionally relied on a robot's vision, torque or haptic sensor to attain information about the object. For instance, a robot's vision module provides effective information regarding an object's shape and size. Similarly, the torque feedback from a robot's gripper may provide information about the object's weight. This project aims to create a model that moves forward with this idea of designing intelligent behaviour for automated entities with the help of a sensor modality that has seen less adoption in robot manipulation so far: Audio.

The prime motivation for this project is to come up with a model which enables a robot to use audio observations as a meaningful source of information. This particular functionality could prove useful in situations where the previously discussed sensor modules fail to pick up a critical object property. Imagine a scenario where the robot is presented with two objects of similar color, shape, size, weight and rigidity but the surface of the two objects are distinct, and the way the robot manipulates each object is dependent on the object's surface property. In such cases, conventional sources of information would prove to be insufficient, but audio observations generated by the object and its interaction with the environment will enable the robot to distinguish between the two objects.

In an ideal situation, the problem statement could be handled solely through effective audio signal processing and analysis. However, observations generated in realistic conditions are noisy and dependent on various factors. For instance, if we are generating observations by striking the target object softly on a surface, the speed and angle of impact of the object on the surface will affect the generated observation. Therefore,

the model needs to be constructed in a way that it is capable of handling these uncertainties in the observation. A Partially Observable Markov Decision Process (POMDP) model provides us with a principled mathematical framework for constructing such a model. Another key aspect of this project is to identify viable audio features that are computationally inexpensive but at the same time provide rich information about the audio signal.

1.1 Problem Statement

The goal of this project is to develop a strategy for robust manipulation of objects based on audio observations. The aim is to use audio feedback generated from the object's interaction with the environment as the primary source of decision-making observation. This includes enabling the robot to make decision regarding which sound profile (e.g. sliding an object across a table) in a given situation would yield the most relevant information in the context of the manipulation task at hand, i.e., exhibit situational awareness regarding observation generating actions.

The project will be implemented using Kinova's Movo as our agent for manipulation. In order to estimate the model's performance, it will be tested with 4 different objects and it is expected that the model identifies each of them with a very high accuracy. The objects are a pringles can (Object 0), a coffee mug (Object 1), a coffee plastic cup (Object 2) and a paper base object (Object 3). In order to further test the robustness of the model, it will be tested with modified versions of Object 0 and Object 1. All of these objects can be observed in Fig 1.1. It is also important to understand that the model does not account for noise in terms of signal's background noise to a great extent and therefore all of its empirical evaluation is restricted to lab conditions.

One of the 4 objects is placed on the table in front of the robot and the initial configuration (robot pose and object location) is assumed to be constant and perfectly known. The physical properties of the object, such as the surface properties, are unknown. There are two terminal actions - move to location A and move to location B. The robot has to identify the object based solely on the audio observations generated from its interaction with the object. Then based on the reward model, the robot decides where the target object should be moved - location A or location B. A summary of the nomenclature used in this report can be found in Table 1.1.

1.2 Report Outline

The coming chapters in the report expand upon important aspects of the project such as Background and Related Work, Design and Implementation, Experimental Methodology, Results and Conclusion. Each chapter has a short introduction and summary.

Chapter 2 provides the reader with an overview of the essential concepts which are at the heart of this project. For instance, it provides a short summary of audio

Table 1.1: Nomenclature used in this report

Term	Meaning
Object 0	Pringles Can
Object 1	Coffee Mug
Object 2	Coffee Plastic Cup
Object 3	Paper base Object
Action 1	Sliding object across the table
Action 2	Lifting and Striking the object softly on the table

classification research and also introduces the fundamentals of Partially Observable Markov Decision Process, which eases the reader's transition when the model is being constructed in chapter 3.

Chapter 3 mainly focuses on the methodology adopted to construct the POMDP model. It explains how the POMDP parameters, introduced in chapter 2, are defined for the model. The chapter also discusses in depth how the observation model was built, including the necessary audio signal analysis.

Chapter 4 discusses the hardware setup and software architecture for the implementation of the model. It also discusses how information flows between the various entities, and how observations are captured and analyzed.

Chapter 5 mainly presents the results and analysis for the experiments, and explains how these results demonstrate some of the salient features of the model. Chapter 6 presents a short discussion about what kind of research can be expected in this field of study in the future and some concluding thoughts for the project.

Figure 1.1: Target Objects for Identification

Chapter 2

Background and Related Work

2.1 Introduction

This chapter presents the relevant background knowledge which informs the reader about pre-existing work in the concerned areas of the project. This chapter will allow the readers to familiarize themselves with the fundamental principles and frameworks on which the rest of the report depends heavily. There are two broad topics that will be discussed in this chapter : the existing research in audio classification which will form the basis of our audio feature analysis in Section 3.2.5 and the formal definition of the POMDP model will ease the reader's transition when we are building the model in Section 3.2.

2.2 Background

2.2.1 Audio Classification

Audio classification has been a subject of focused research in the past two decades. Extracting relevant information from complex, high dimensional data makes the research area extremely challenging and interesting. Another aspect of audio analysis which makes the field so challenging is the presence of varied noise in signals. Similar to image classification, advancement in audio classification coincided with the development of powerful deep learning tools. Convolutional neural network based models have been applied to varied objectives such as speaker identification, speaker gender identification, music genre identification, etc. [2]. However, an efficient neural network model needs to be trained on a large dataset. In most of the cases, a neural network is only as good as the dataset on which it is trained.

As mentioned before, the idea of audio feedback being used as a source of information in robotics has been largely unexplored. However, the topic has generated some interesting work in the recent past. Gandhi et al. [3] have recently concluded a study where they explored the same idea with a different approach. They used a combination of audio and visual observations to identify the object and the corresponding action

which caused the audio observation. Their classification model was based on image recognition, treating the visualization of the audio signal as an input to a convolutional neural network along with the visual data of the object, which required a large sample dataset.

2.2.2 POMDP Planning

Partially Observable Markov Decision Process (POMDP) is an extension of the Markov Decision Process where the true state of the agent cannot be determined with absolute certainty [4]. It provides a generalized framework which allows us to frame a decision problem where the model can account for uncertainty. The formal definition of a POMDP can be represented as the following tuple : { S, A, O, T, Z, R, b0, γ } , where S is a finite set of discrete states, A is a set of discrete actions and O is a set of observations. At every time step, the agent is in a state s $\in$ S, executes an action a $\in$ A, receives an observation o $\in$ O and transitions to the next state s' $\in$ s. The transition function $T(s, a, s') = P(s'|s, a)$ is a conditional probability distribution which determines the probability for the model to land in state s' given current state s and action a. The observation function $Z(s, a, o) = P(o|s, a)$ is a conditional probability distribution which determines the probability of receiving an observation o given current state s and action a. The transition function and observation function represent the uncertainty in state transition and sensing respectively. The reward function $R(s, a)$ is a state-action dependent function which serves as our objective function [5]. b_0 represents our initial belief and $\gamma \in (0,1)$ is our discount factor.

The solution of a POMDP problem is a mapping from beliefs to the best action, and it is called an optimal policy. A policy π is a value function $V_\pi(b)$ which determines the expected total reward for executing policy π from belief b, and is calculated as $V_\pi(\mathrm{b}) = E[\sum_{t=0}^{\infty} \gamma^t \mathrm{R}(s_\mathrm{t}, a_\mathrm{t})|b,\pi]$. An optimal policy is the policy π^* whose whose value function $V_{\pi*}(b)$ is the highest among all possible policies for any belief b [6].

Instead of operating over actual states, which is not possible due to the partial observability of the states, a POMDP model operates on a belief state. A belief state is a probability distribution over the actual states and encapsulates the agent's 'belief' of its current state [4]. Since a belief state is a probability distribution over the actual states, a belief space of 'n' states has 'n-1' dimensions. This is referred to as the 'curse of dimensionality' and makes POMDPs notoriously difficult to solve [7]. However, recent development of tractable algorithms [8; 9; 10; 11] has made the use of this framework practical for robotics use, where real time planning is essential. For the purpose of this project, we will be using Online POMDP Planning Toolkit (OPPT) [12] which is a software toolkit for approximating POMDP solutions in real time. We will be discussing OPPT in greater depth in the next section.

2.3 Summary

Although there is a wealth of information in the field of audio signal processing, feature extraction and classification, the application of audio classification in the field of robotics has been fairly unexplored. The POMDP model is a powerful variation of the Markov Decision Process which allows us to account for uncertainty in a model. Using the fundamental principles discussed in this chapter, we will see how we can build an effective model in Chapter 3.

Design and Implementation

3.1 Introduction

The previous chapter discussed the fundamental principles and framework within which the problem would be modelled. This chapter moves forward with the model definition and specifications. The entire problem is framed as a POMDP model. However, the most important aspect of the POMDP model would be an effective observation model. Therefore, the POMDP model specification will be discussed in the first few sections, and then the majority of the chapter is dedicated to building the observation model using effective audio feature analysis.

3.2 POMDP Model

3.2.1 State-Action space

In our POMDP model definition, the state space is composed of 11 variables. The state space of the model is defined as $S = \theta \times \alpha \times K$, where $\theta \in \mathbb{R}^7$ is the set of robot's arm joint angles, $\alpha \in \mathbb{R}^3$ is the relative distance of the object from the robot's end-effector and K is the discrete set of objects.

The action space is designed to accommodate two classes of actions – Observation generating actions and manipulation actions. As the name suggests, the observation generating actions enforce the target object's interaction with the environment, thus producing observations. The manipulation actions consist of actions to be implemented once the model has made its decision about the target object's identity. There are 2 observation generating actions, referred to as Action 1 and Action 2. Action 1 positions the end-effector at the object's location and then the end-effector pushes the object 5 cm in the x direction along the surface. Action 2 positions the end-effector at the location of the object, closes the robot gripper, lifts the object 5 cm in the positive z direction and then brings down the object 5 cm in the negative z direction with a velocity of 10 cm/s, an then opens the gripper. There are 2 manipulation actions, referred to as 'Move to Location A' and 'Move to Location B'. The manipulation actions are also terminal

actions, i.e, the resulting state of those actions, irrespective of the current state, is the terminal state.

3.2.2 Initial Belief

The initial belief state is a distribution over a range of state values, pre-defined by the user. Given the definition of the model's state space, the initial configuration determines the initial pose of the robot arm, the relative distance of the object from the robot's end-effector and the object itself. Since the primary aim of this project is to identify the object and a constant initial configuration is assumed for the robot pose and object location, the initial belief is uniformly distributed across the last state variable, representing the object, keeping all other state variables constant. Since the initial distribution over the object state is uniform, the model executes information gathering actions (Action 1, Action 2) in order to make its final decision.

3.2.3 Transition Model

As mentioned earlier, the transition function is a conditional probability function $T(s, a, s') = P(s'|s, a)$ which determines the probability that the agent will land in state s' given current state s and action a. In other words, it models the uncertainty in state transition. Given that the object or its property will not change because of any of the defined actions, the transition probability for the object state variable is defined as $P(s'_{11} = s_{11} \mid s_{11}, a) = 1, \forall\ a \in A$, where s_{11} is the current object state, s'_{11} is the next object state. For the rest of the state variables, change in the resulting state is calculated based on the given actions without addition of any noise, i.e, assuming a perfect transition model.

3.2.4 Reward Model

The reward model is quite straightforward. Every observation generating action incurs a penalty of 1 , i.e, $R(s,a)$ = -1 $\forall\ s \in S$ where $a \in$ {Action 1, Action 2}. For Object 0 and Object 2, 'Move to Location A' results in a reward of 10 and 'Move to Location B' incurs a heavy penalty of 100, i.e, $R(s_{11} = Object0$ or $Object2$, $a = 'Move\ to\ Location\ A') = 10$ and $R(s_{11} = Object0$ or $Object2$, $a = 'Move\ to\ Location\ B') =$ -100. For Object 1 and Object 3, 'Move to Location B' returns a reward and 'Move to Location B' incurs a penalty, i.e, $R(s_{11} = Object1$ or $Object3$, $a = 'Move\ to\ Location\ B') = 10$ and $R(s_{11} = Object1$ or $Object3$, $a = 'Move\ to\ Location\ A') =$ -100. It should also be noted that any action that results in the collision with the environment (table for this model) will result in a penalty of 100 and termination of the model.

3.2.5 Observation Model

An effective observation model is one which provides the most relevant information regarding the true state of the object. The most naïve solution for this would be to build a neural network based classifier that provides indication regarding the object's property. Since the time and resource for this project did not allow building a sufficiently large dataset for such an observation model, and it would be counter intuitive to expect the user to build such a dataset every time a new object is introduced in the model, it was more prudent to explore the observation signal's audio features that would serve the same purpose based on a much smaller dataset.

Audio Features

One of the primary goals is to transform the captured observation into meaningful information for the model. However, an audio signal is a high dimensional data that cannot be directly plugged into a model as observation. Therefore, there is a need to extract features from the audio sample that provide insight into its properties.

A signal can be represented in two domains: time and frequency. While the time domain represents how a signal's amplitude changes with respect to time, the frequency domain tells us the composition of the signal with respect to frequency. Features which are extracted from the time domain representation are known as temporal features while the frequency domain features are known as spectral features. The interpretation of an audio signal has numerous aspects to it, such as tone, pitch, loudness etc. Each feature is associated with one or more such aspect of auditory perception. Some of the features that were explored through the course of this project are as follows:

- Temporal Features:
  - RMS : The root mean square value is calculated by taking the square root of the mean square values of the amplitude. It represents the loudness of an audio signal.
  - ZCR : The zero-crossing rate is defined as the rate of amplitude's sign-changes. It is a prominent feature used to classify the type of audio event, such as classification into speech and non-speech signal [13].
- Spectral Features:
  - Spectral Centroid – It represents the tone or timbre of an audio signal and it is calculated by taking the weighted mean of the frequency spectrum obtained by the Fourier transform, with the magnitude of each frequency serving as the weights [14]. It can also be interpreted as the center of mass of the frequency spectrum. There are two methods of calculating the spectral centroid, each of which differs in the way the signal is transformed into the frequency domain. In the first method, the entire signal is analyzed as a

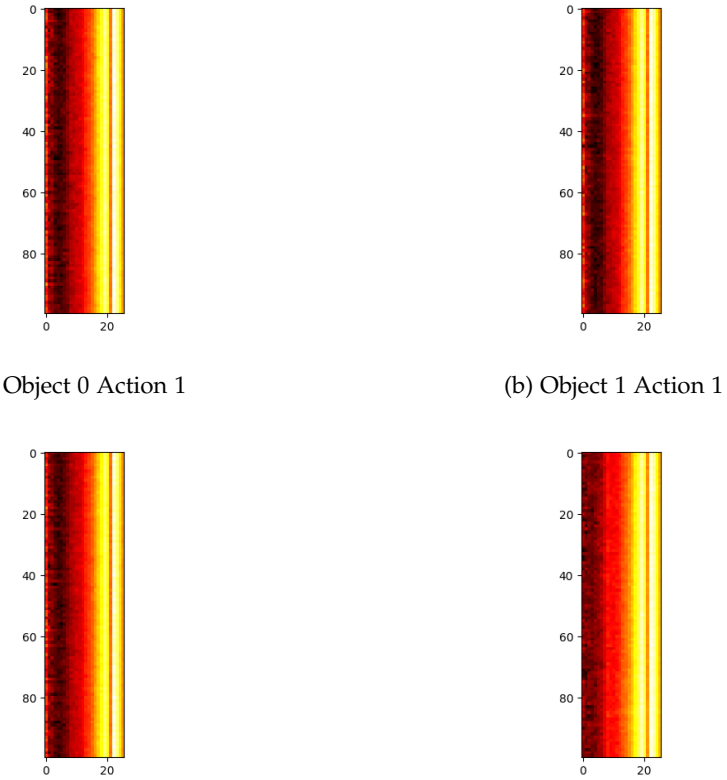
single entity, without intermediate sampling, and therefore providing us the frequency distribution of the entire signal, independent of time. The second method uses the short-time Fourier transform which uses a sliding window of short length to provide us with spectral information at each time interval, thus preserving some of the temporal information as well.

– Mel Frequency Cepstral Coefficients (MFCC) [15] – MFCC is a short-term spectral feature that is believed to be one of the best representation of the perceptually relevant aspects of an audio signal. The idea is to transform the signal to Mel scale, which is a mapping between actual signal frequency and the scale on which human auditory system operates. The formula for converting frequency to Mel scale is $M(f) = 1125ln(1 + f/700)$. Depending on the range of frequency that needs to be covered and the resolution of each bank, the frequency range for each filterbank is calculated on the Mel scale, following which, the short time Fourier transform of the original signal is calculated. Based on these two sets of data, each windowed time sample of the original signal is assigned a coefficient for each filterbank.

Observation Model Construction

Out of the various audio features explored, the most promising were MFCC and a combination of spectral centroid and rms value. The first iteration of the observation model was based on a combination of spectral centroid and rms value. A sample set of 10 observations was collected for each object and observation generating action pair. A multi-variate gaussian distribution was created for each of the object and observation generating action pair based on the mean and standard deviation of the collected observations. These distributions were used for sampling observations during POMDP planning and calculating the likelihood of the observations received after robot execution. The results from this iteration of the observation model were not robust enough and so this observation model was discarded.

A different approach was used for building the observation model using MFCC. The MFCC value for a single audio signal is represented as a 2D array where each row represents a temporal windowed sample of the audio signal and each column represents a filterbank with a specific frequency range. Keeping the recording time for the observation sufficiently long and assuming that the robot's actions are deterministic to an extent with respect to the time duration for execution, the signals are trimmed accordingly to obtain a fixed size of 2D observation. Similar to the previous method, a sample set of 10 observations was collected for each object and observation generating action pair and a matrix of mean values was computed for each of those pairs. These mean values serve as cluster centers and any new incoming observation can be classified to a cluster center based on its distance from the centers. The heat map for the mean MFCC values for each object and observation generating action pair can be observed in Figure 3.1 and 3.2. The x-axis represents the fiterbanks while



(a) Object 0 Action 1

(b) Object 1 Action 1

(c) Object 2 Action 1

(d) Object 3 Action 1

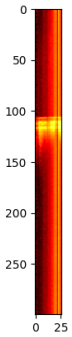
Figure 3.1: MFCC Heat Map for Mean values

the y-axis represents the segmented audio samples. The lighter color areas represent higher coefficients. For instance, for object 0 action 2 image, a peak in frequency can be observed halfway into the sample. It means that at that instant in the audio sample, a higher co-efficient for the corresponding frequency filterbanks was observed. The previous iteration of the observation model returned the values for spectral centroid and rms value as observation, while in this model the observation is the object class itself. The results from this observation model were encouraging and the final results of identification based on this model can be observed in Chapter 5.

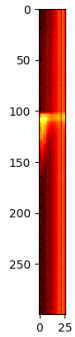
3.3 Summary

Using the framework discussed in Chapter 2, a model was prepared for the problem statement defined in section 1.1. A formal summary of the model is as follows :

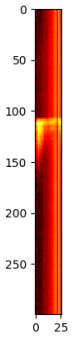
State space $S = \theta \times \alpha \times K$, $\theta \in \mathbb{R}^7$ is the set of robot's arm joint angles, $\alpha \in \mathbb{R}^3$ is the relative distance of the object from the robot's end effector and K is the discrete set of



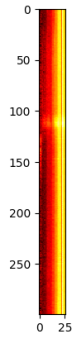
(a) Object 0 Action 2



(b) Object 1 Action 2



(c) Object 2 Action 2



(d) Object 3 Action 2

Figure 3.2: MFCC Heat Map for Mean values

objects. s_{11} represents the state of the object.

Actions = [Action 1, Action 2, Move to Location A, Move to Location B]

O = [O_0,O_1,O_2,O_3] where O_x represents the observation that the target object belongs to class $Object_x$

$P(s'_{11} = s_{11} \mid s_{11}, a) = 1, \forall\ a \in A$ where s_{11} is the current object state, s'_{11} is the next object state

Observation Model :

$P(O_0 \mid s_{11} = Object\ 0, a = Action\ 1) = 0.5$
$P(O_1 \mid s_{11} = Object\ 0, a = Action\ 1) = 0.3$
$P(O_2 \mid s_{11} = Object\ 0, a = Action\ 1) = 0.15$
$P(O_3 \mid s_{11} = Object\ 0, a = Action\ 1) = 0.05$
$P(O_0 \mid s_{11} = Object\ 0, a = Action\ 2) = 0.6$
$P(O_1 \mid s_{11} = Object\ 0, a = Action\ 2) = 0.05$
$P(O_2 \mid s_{11} = Object\ 0, a = Action\ 2) = 0.3$
$P(O_3 \mid s_{11} = Object\ 0, a = Action\ 2) = 0.05$
$P(O_0 \mid s_{11} = Object\ 1, a = Action\ 1) = 0.11$
$P(O_1 \mid s_{11} = Object\ 1, a = Action\ 1) = 0.67$
$P(O_2 \mid s_{11} = Object\ 1, a = Action\ 1) = 0.11$
$P(O_3 \mid s_{11} = Object\ 1, a = Action\ 1) = 0.11$
$P(O_0 \mid s_{11} = Object\ 1, a = Action\ 2) = 0.05$
$P(O_1 \mid s_{11} = Object\ 1, a = Action\ 2) = 0.6$
$P(O_2 \mid s_{11} = Object\ 1, a = Action\ 2) = 0.3$
$P(O_3 \mid s_{11} = Object\ 1, a = Action\ 2) = 0.05$
$P(O_0 \mid s_{11} = Object\ 2, a = Action\ 1) = 0.11$
$P(O_1 \mid s_{11} = Object\ 2, a = Action\ 1) = 0.11$
$P(O_2 \mid s_{11} = Object\ 2, a = Action\ 1) = 0.67$
$P(O_3 \mid s_{11} = Object\ 2, a = Action\ 1) = 0.11$
$P(O_0 \mid s_{11} = Object\ 2, a = Action\ 2) = 0.11$
$P(O_1 \mid s_{11} = Object\ 2, a = Action\ 2) = 0.11$
$P(O_2 \mid s_{11} = Object\ 2, a = Action\ 2) = 0.67$
$P(O_3 \mid s_{11} = Object\ 2, a = Action\ 2) = 0.11$
$P(O_0 \mid s_{11} = Object\ 3, a = Action\ 1) = 0.01$
$P(O_1 \mid s_{11} = Object\ 3, a = Action\ 1) = 0.01$
$P(O_2 \mid s_{11} = Object\ 3, a = Action\ 1) = 0.48$
$P(O_3 \mid s_{11} = Object\ 3, a = Action\ 1) = 0.5$
$P(O_0 \mid s_{11} = Object\ 3, a = Action\ 2) = 0.11$
$P(O_1 \mid s_{11} = Object\ 3, a = Action\ 2) = 0.11$
$P(O_2 \mid s_{11} = Object\ 3, a = Action\ 2) = 0.11$
$P(O_3 \mid s_{11} = Object\ 3, a = Action\ 2) = 0.67$

Reward Model :

$R(s,a)$ = -1 $\forall$ $s \in S$ where $a \in$ {Action 1, Action 2}
$R(s_{11} = Object0, a = 'Move\ to\ Location\ A') = 10$
$R(s_{11} = Object0, a = 'Move\ to\ Location\ B') = -100$
$R(s_{11} = Object1, a = 'Move\ to\ Location\ A') = -100$
$R(s_{11} = Object1, a = 'Move\ to\ Location\ B') = 10$
$R(s_{11} = Object2, a = 'Move\ to\ Location\ A') = 10$
$R(s_{11} = Object2, a = 'Move\ to\ Location\ B') = -100$
$R(s_{11} = Object3, a = 'Move\ to\ Location\ A') = -100$
$R(s_{11} = Object3, a = 'Move\ to\ Location\ B') = 10$

Chapter 4

Experimental Methodology

4.1 Introduction

Now that we have finished constructing the model and estimated its parameters, we proceed with the implementation. This chapter presents the details of how we implemented the tests for our designed model keeping the problem statement in sight. We first understand our planned experimental setup. Then we present details regarding the hardware resources utilized for implementation and their specifications. Lastly, we describe the components of our software architecture.

4.2 Experimental Setup

For testing the model, 4 target objects were used : an empty pringles can (object 0), a coffee mug (object 1), a plastic coffee cup (object 2) and a paper-base object (object 3). The experiments were conducted in a controlled environment, which means the height of the table and the initial object position is assumed to be kept constant. It is also assumed that the environment has minimal background noise. This statement might seem contradictory because of the claim that the model is robust enough to handle noise. However, the noise in observation being referred to is produced by the object's interaction with the environment such as speed and angle of impact of the object on the surface of the table.

A discount factor of 0.99 was used for the reward model. It should also be noted that since the Adaptive Belief Tree [9] (ABT) is an on-line solver, it's planning time was limited to 2 seconds per step. After every step, the belief is updated using a Sequential Importance Resampling particle filter [16]. The total number of belief particles at any step was chosen to be 1000. The initial configuration of the robot and the placement of the object for the chosen initial belief can be observed in Figure 4.1. For the purpose of these experiments, it is assumed that the location of the table and the object is perfectly known.

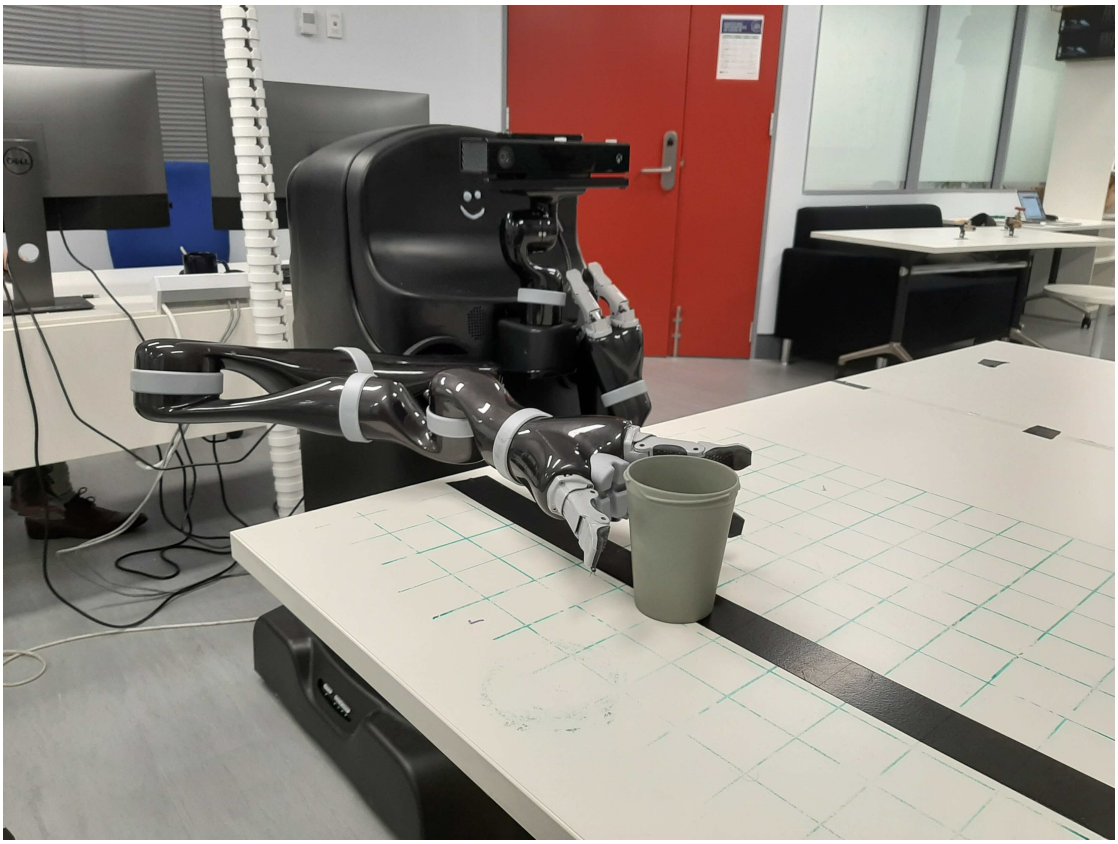


Figure 4.1: Initial configuration of the setup representing the initial belief state for Object 2 (Plastic coffee cup)

4.3 Hardware Platform

The hardware setup for the experiments consisted of two entities: the kinova movo mobile robot for object manipulation and a planning PC. The planning PC did not have any extraordinary requirement. Any system with a reasonable processing power and a working microphone can be used. The details of the particular planning PC used for this experiment can be viewed in Appendix 3.

The kinova movo is a mobile manipulator with two seven degree of freedom Jaco arms. In addition to the two robotic arms, the robot has a linear joint for adjusting its height, a mobile base for movement and a Microsoft Kinect vision module for perception. The end-effector for both the arms has a 3 fingered gripper, capable of effective object manipulation. The arms can reach up to 20 cm/s of linear velocity.

The robot has two Intel Next Unit of Computing (NUC) computers – Movo1 and Movo2. Both of the NUC is an Intel Core-i7@3.1Ghz with 16GB of RAM. Movo 1 is mainly used for sensor processing while Movo 2 is used for process control and scheduling. The entire system runs on Robot Operating System (ROS). The designed model does not use the mobile base of the robot under the assumption of a static experimental setup. Since the location of the object is also assumed to be perfectly known before execution, the perception module was also not utilized in any way.

4.4 Software Platform

4.4.1 Online POMDP Planning Toolkit (OPPT)

As mentioned earlier, POMDP models are very difficult to solve due to the 'curse of dimensionality' [7]. OPPT, which is a software-toolkit capable of approximating POMDP solutions online [12], was used in order to plan with the designed POMDP model. The idea behind OPPT is that it allows users to frame their POMDP problem using a plugin-type library. Each tenet of a POMDP problem such as the intial belief, observation model, heuristic behaviour, state transition behaviour, reward model and terminal conditions is expressed as a plugin-type C++ library. The rest of the essential parameter values are obtained through a configuration file, also to be prepared by the user under a pre-defined format. Once the problem is adequately framed, the POMDP engine churns out an approximating solution using a solver of our choice. An overview of the contents of each of these plugins is as follows :

- Initiali Belief Plugin : As the name suggests, this is where the initial condition of the model is defined. Generally, a range of state values is defined and the initial belief particles are sampled uniformly from that range. However, for the designed model, a distribution over all the states was not necessary . Since the focus of the project is not robustness in terms of object position error, the first 10 variables of the state space, representing the right arm joint angles and the

relative distance of the object from the right arm end-effector, were kept at a constant value for all the belief particles. The 11th state variable representing the object (or object property) was an integer value drawn uniformly from the range [0,3]. The plugin also sets up the kinematic and collision model for the robot, target object and environment (the table), which aids in further planning and collision checking.

- Observation Plugin : The observation plugin was used to implement the observation model that was derived in the previous chapter. As mentioned before, under the selected model, the observation received is the class to which the target object belongs. Two sets of probability functions are defined – the first one determines an observation given a state and action pair while the second one gives the likelihood for the received observation for each state-action pair. It should be noted that there are two implementations of the observation plugin – one for planning and the other for robot execution, the only difference being that the execution plugin calls a ROS service for an actual observation instead of a simulated one.

- Transition Plugin – This plugin defines the behaviour of the transition function of a POMDP model. It defines the transition behaviour of state variables based on the actions taken. For instance, action 1 represents the sliding action where the end-effector moves the target object 5cm horizontally (in the x direction). Based on that the first 10 variables of the state space are updated accordingly. The kinematic and collision model of the robot and the target object is updated as well. There is a single plugin for both planning and execution in this case. When the belief particle for robot execution is detected, we connect with the robot's control system through an API and implement the action on the robot. As mentioned before, in the model, the last variable of the state space, representing the object or object property is independent of any action.

- Reward Plugin – The reward plugin defines the reward function of the model. A penalty of 1 is allotted for each observation generating action (action 1 and action 2). If the true state of the object is object 1 or object 3, action 3 generates a positive reward of 10 while action 4 generates a heavy penalty of 100. Similarly, if the true state of the object is object 2 or object 4, action 4 generates a positive reward of 10 while action 4 generates a penalty of 100. A penalty of 100 is assigned to any action that results in the robot's collision with the table.

- Terminal Plugin – The terminal conditions for the model is rather simple. The model is terminated if either one of action 3 or action 4 was implemented, or any other action which results in a collision with the table.

OPPT also gives us the option to choose the solver which the engine uses. Users may choose their own implementation for a solver. The model uses OPPT's default solver

which is based on Adaptive Belief Tree (ABT) [9]. ABT uses a modified Monte-Carlo tree search for large online POMDP problems [10] where a belief tree is maintained which can be reused and improved as the algorithm progresses.

4.4.2 ROS Listener Node

A very important aspect of this project was to effectively capture the audio observations generated by the target object's interaction with the environment. In order to keep the implementation as simple as possible, a ROS node was setup separate from the entire OPPT operation. This node operates by constantly scanning for the boolean value of a ROS parameter and whenever its true, the microphone of the planning PC is activated. Once the observation generating action's execution is completed by the robot, the parameter is set to false from the transition plugin. The listener node, which is constantly observing the parameter's value, takes the false value as an indication that the recording is completed and the recorded signal is saved as a 2-channel audio recording in the 'wav' format with a sampling rate of 44.1 KHz. The ROS parameter is controlled by the 'Transition Plugin', where in whenever action 1 or action 2 is detected for the robot's execution, the parameter is set to true and when the action is completed, it is set to false. This is a simple implementation with minimal requirements that produces effective results.

4.4.3 ROS Observation Service Node

Similar to the 'Listener' ROS node, there is an 'Observation Service' ROS node which analyzes the most recently captured observation and assigns it to one of the object class using the observation model described in the previous chapter. Since in this case a value needed to be returned from the ROS node as well, using ROS parameters was not sufficient. Therefore, the node uses a 'ROS Service' which enables two-way communication. As a result of this two-way communication, ROS service provides an added flexibility in the way observations are analyzed. For instance, different analysis pipelines can be configured based on different observation generating actions. The call for the ROS service is controlled by the observation plugin's execution version.

4.5 Summary

The robot used for the implementation of the model was Kinova Movo. The model uses OPPT for obtaining an optimal policy of the POMDP problem in real time. The block diagram in Figure 4.2 provides an idea regarding the architecture of the software model, and how information is exchanged between modules. There are 3 entities running on the planning PC : OPPT, observation service node and listener node. When OPPT decides that an observation generating action needs to be executed, it sets the value of a ROS param to be true. The listener node, which is repeatedly scanning the

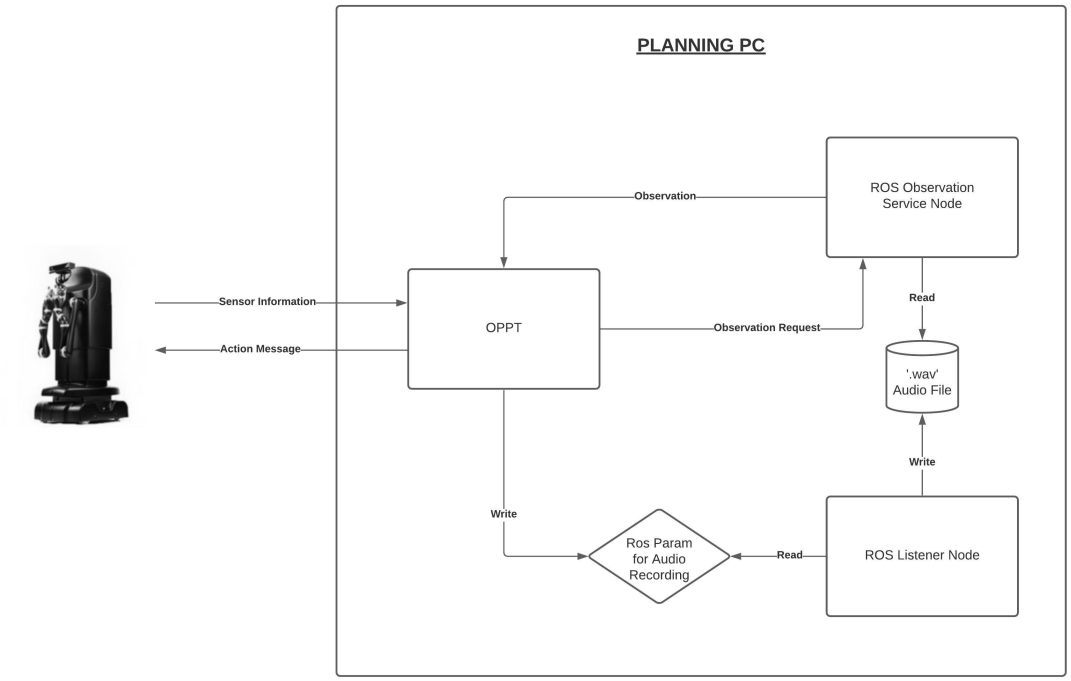


Figure 4.2: Block Diagram representing software overview

value of that param, starts recording the observation and when the param is set to false by OPPT, the recording stops and the audio file is saved to disk. Then, when OPPT requires the observation, a service request through ROS is sent to Observation service, which analyzes the saved audio signal and classifies the signal to its nearest cluster mean. OPPT is also linked to the robot via ethernet and ROS and transmits action messages to the robot and reads joint angle data periodically.

Chapter 5

Results

5.1 Introduction

In this chapter, the results of the project are presented and discussed with reference to the aim of the project, which was to build a complex manipulation strategy based on audio feedback under uncertainty. The primary aim was to demonstrate that the model can correctly identify the objects through audio feedback. One of the key secondary aims was to build a model that is aware of the consequences of its actions, i.e., be aware of which observation profile (e.g sliding across the table) would yield the maximum information. Some key evidence will also be presented that suggests that our model is robust enough to handle uncertainty in observations.

5.2 Object Identification

In the first iteration of trials, using the observation model results discussed in chapter 3, a total of 40 trials (10 for each object) were conducted. Although, the model performed with an accuracy of 95%, the fact that the model was jumping to the final state on the basis of 1, at most 2 observations, was concerning . It was an indication that the model did not account for enough noise in observation as expected. It was operating under the premise that the observation accuracy was high. Based on a few manual observations of a few trial runs, a tuned observation model was developed that is presented in section 3.3.

40 trials of experiments were conducted again and this time the model achieved 100% accuracy. The tuned model was able to cope with wrong observations well and in some cases recovered from 2 wrong observation in the same trial. Table 5.1 gives a summary of the key statistics of the trials. It can be noticed that in certain object-action scenarios, where our model was still sure of its observations, the model quickly converges to the final state with a high reward. In other instances, where the model is expecting noisy observation or received contradicting observations, the model takes its time to execute additional observation generating actions before deciding the terminal action.

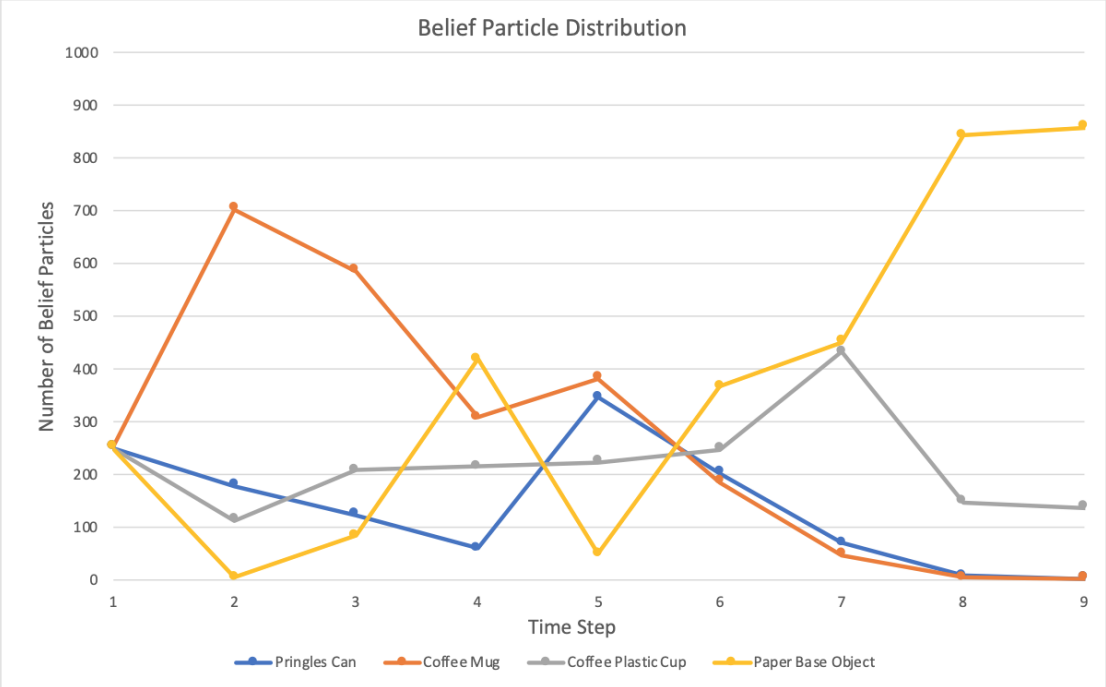


Figure 5.1: Belief Distribution for a single trial - Target Object : Paper Base Object

Additionally, it can be observed that the model retains the capacity to recover from bad observations. In Figure 5.1,it can be observed how the distribution of the belief particles varied across a particular experiment trial. The target object for this trial was the paper base object. The total number of belief particles is derived from a user defined preset value, 1000 in this case. So, initially it is observed that there is a uniform distribution of all the object states. However, the very first observation received is erroneous (observation class - coffee mug). Since the tuned observation model is more conservative in nature, the next stage does not have enough belief particles for the model to converge to the wrong state and the model continues generating observations. The model receives correct observations for the next two actions and the belief particle count for paper base object recovers. The model receives yet another wrong observation, this time object 2 (coffee plastic cup). There is no scope for convergence at this stage and the model remains absolutely undecided. The model receives 3 consecutive correct observations in the final stage of the run and eventually the belief particle count of the paper base object is enough for the model to converge to the correct state.

5.3 Situational Awareness

One of the key features that we wanted our model to exhibit was situation awareness. In the context of our model, this simply means deciding which observation generating

Table 5.1: Results of Experiment Trials for the original 4 objects

True State	Final Belief State	No. of Steps	Action 1	Action 2	Discounted Reward
0	0	3	0	2	7.811
0	0	4	2	1	6.73289
0	0	2	0	1	8.9
0	0	4	1	2	6.73289
0	0	2	1	0	8.9
0	0	2	1	0	8.9
0	0	2	1	0	8.9
0	0	3	0	2	7.811
0	0	2	0	1	8.9
0	0	2	0	1	8.9
1	1	2	0	1	8.9
1	1	3	1	1	7.811
1	1	3	1	1	7.811
1	1	2	0	1	8.9
1	1	2	0	1	7.811
1	1	3	1	1	7.811
1	1	3	2	0	7.811
1	1	2	0	1	8.9
1	1	2	0	1	8.9
1	1	3	1	1	7.811
2	2	3	0	2	7.811
2	2	3	1	1	7.811
2	2	5	3	1	5.665561
2	2	3	1	1	7.811
2	2	3	1	1	7.811
2	2	3	1	1	7.811
2	2	3	1	1	7.811
2	2	3	0	2	7.811
2	2	3	0	2	7.811
2	2	3	0	2	7.811
3	3	2	1	0	8.9
3	3	5	3	1	5.665561
3	3	8	3	4	2.527188
3	3	7	3	3	3.562816
3	3	2	0	1	8.9
3	3	2	0	1	8.9
3	3	3	1	1	7.811
3	3	3	2	0	7.811
3	3	3	1	1	7.811
3	3	3	1	1	7.811

Table 5.2: Data demonstrating situational awareness regarding observation generating actions

Object 0 Belief Particles	Object 1 Belief Particles	Object 2 Belief Particles	Object 3 Belief Particles	Action
251	640	96	13	Action 2
203	658	117	22	Action 2
250	520	81	149	Action 2
242	546	162	50	Action 2
273	451	109	167	Action 2
238	468	102	192	Action 2
457	458	85	0	Action 2
230	524	76	170	Action 2
44	353	383	220	Action 1
168	637	191	4	Action 1
121	305	570	4	Action 1
70	47	432	451	Action 2
10	16	389	385	Action 2
90	76	456	378	Action 2
90	90	608	212	Action 2

action would yield the maximum information regarding the object and its property. This is actually a fundamental trait of POMDPs and it is the reason why they are used for a variety of problems. From the model's point of view, it is simply taking the action which would maximize its discounted reward. However, this translates into a wonderful feature for this application where it does not waste time executing actions that leave it with little to no new information about the object's true state.

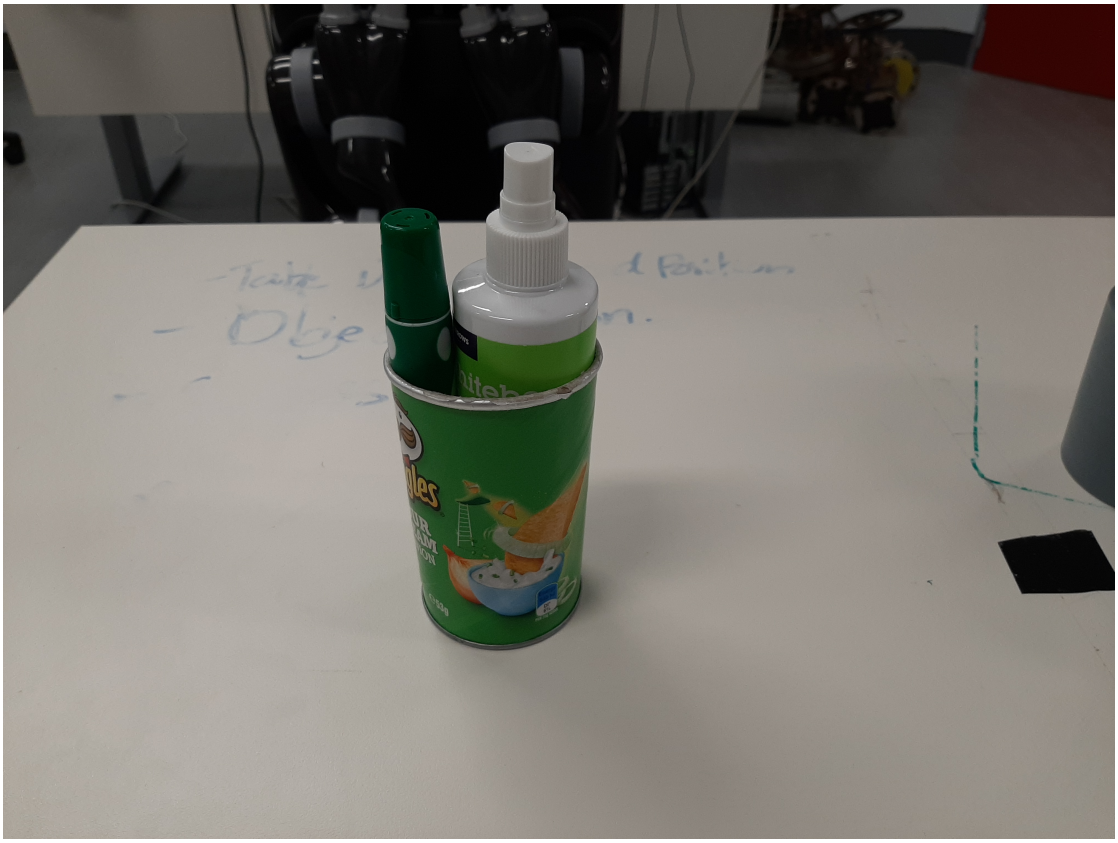
We studied the log files of our trials and analyzed the relationship between distribution of belief states and the observation generating action selected in that situation. From the given observation probabilities, we can see that when the belief states are predominantly distributed between object 0 and object 1, action 2 seems to be a more trustworthy source of observation. A similar relationship exists for object 1 - object 2 and action 1, and object 2 - object 3 and action 2. The data provided in Table 5.2 demonstrates that whenever, the model was confused between these pair of objects, it executed the corresponding action which would provide most promising observations. The log files of the trials are provided in our repository, link to which can be found in Appendix 3.

Table 5.3: Results of Experiment Trials for modified object 0 and modified object 1

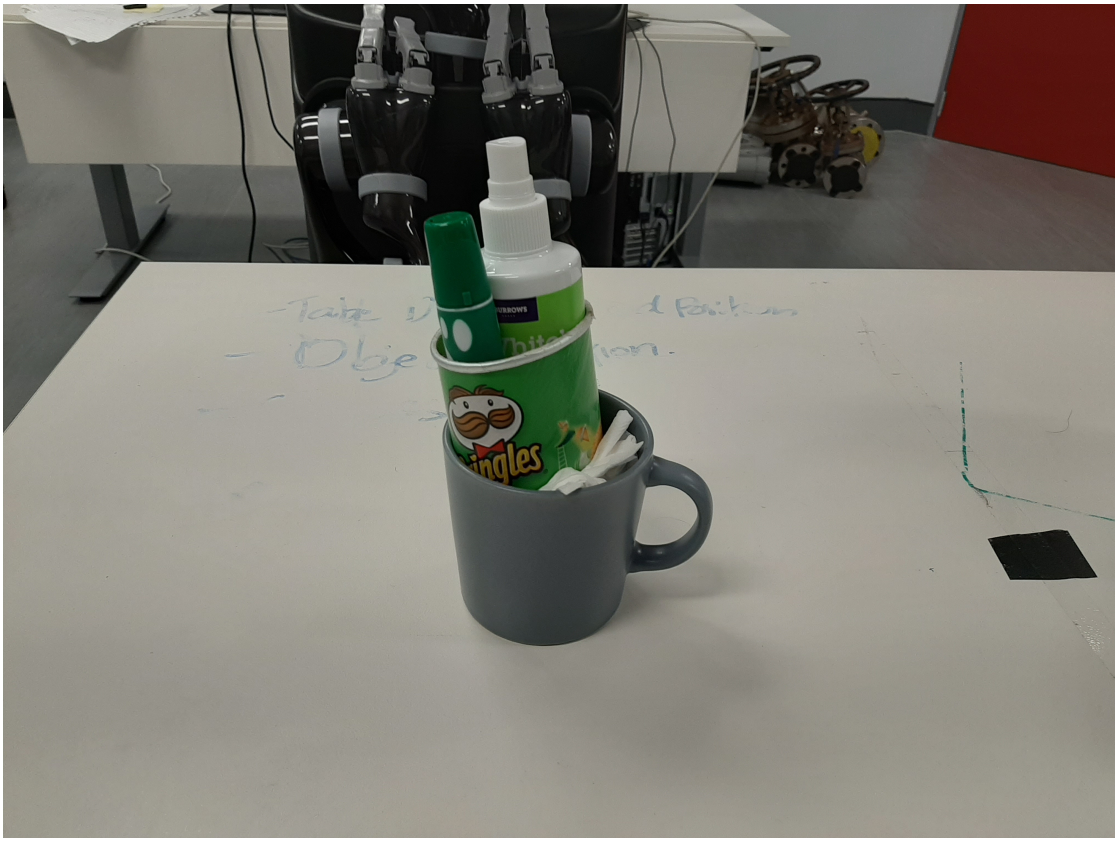
True State	Final Belief State	No. of Steps	Action 1	Action 2	Discounted Reward
0	1	2	1	0	-100
0	0	2	0	1	8.9
0	1	3	2	0	-100
0	0	2	1	0	8.9
0	0	2	0	1	8.9
0	0	3	1	1	7.811
0	0	2	0	1	8.9
0	0	3	0	2	7.811
0	2	6	3	2	4.608905
0	0	2	0	1	8.9
1	1	4	1	2	6.73289
1	0	5	3	1	-100
1	1	2	0	1	8.9
1	1	3	1	1	7.811
1	1	3	1	1	7.811
1	1	3	0	2	7.811
1	1	2	0	1	8.9
1	1	2	0	1	8.9
1	1	2	0	1	8.9
1	1	3	2	0	7.811

5.4 Trials with modified objects

In order to understand how the model responds to unaccounted change in object properties, object 0 and object 1 was modified using additional weights, as shown in Figure 5.2(a) and Figure 5.2(b). Without any change in parameter in the model, it achieved an accuracy of 70% and 90%, for modified object 0 and modified object 1 respectively. The results of these trials can be observed in Table 5.3. Even though the objects have been modified visually and in terms of weight, which means that any visual or haptic sensor based classifier would confuse it with another object, our model was able to identify the original object.



(a) Modified Pringles Can



(b) Modified Coffee Mug

5.5 Summary

Empirical evidence was provided to support the claim that the model performs consistently and is robust enough to handle uncertainty in observation and object property. Evidence indicating intelligent behaviour on part of the agent was also presented. It should be noted that while situational awareness regarding observation generating actions may look trivial in such a small action space, it will be a massively important factor in cutting down run-time and maximizing reward as the state and action space scales up.

Chapter 6

Conclusion

6.1 Future Work

Our model presents promising results for the defined scope. However, it can be further improved by using advanced signal processing techniques which can handle background noise outside of controlled lab conditions. For example, Wark [17] shows that audio classification based on singular short-term frame features are rendered ineffective when presented with considerable variance between training and testing dataset, and instead suggests extracting multiple features versions from a plurality of the original sample, which handles background noise much effectively. We also anticipate that the environment surface may affect the model behavior, and its dependencies can be studied as part of any future work. It would also be interesting to see the results of our model coupled with other sensory observations such as vision and torque. Our present model had a well defined object set from which it had to identify, thus limiting its ability to identify properties. We believe that a more general version of the model can be developed which focuses on object properties rather than the object itself. Such a model would be extremely useful in object manipulation tasks. For instance, making the agent conscious of its choice of manipulation strategy depending on the object's structural strength, surface properties, etc.

6.2 Conclusion

The current trend for industrial automation indicates that the demand for robots with robust manipulation capability will only increase [1]. As such, it is only natural that the expectation of robotic intelligence increases as well. As human beings, we study our immediate environment primarily using our vision and so it was logical that we expect the robots working on our behalf to do the same, which prompted heavy research in the field of robotic vision in the last 15 years [18]. However, we cannot deny the fact the observations from any single sensor does have limitations. A visual observation can reveal information about color, shape and size, but it cannot be used to judge weight. A torque sensor can judge weight, but it cannot reveal information about certain object

properties. A fully automated robot that many have envisioned in the future must be equipped to handle all possible observations.

Through this project, we have demonstrated that audio observation modelled in POMDP problem can be successfully used to identify objects with great success. We have showed that given a small prior information, our model can be very effective in distinguishing between a small set of objects. We have also showed, with reasonable success, that our model is capable of identifying objects with slight alteration in its properties. We believe that this additional capability, of interpreting audio observations, will result in a more complete and versatile agent.

Bibliography

[1] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, and M. Hoffmann, "Industry 4.0," *Business information systems engineering*, vol. 6, no. 4, pp. 239–242, 2014. (cited on pages 1 and 29)

[2] H. Lee, P. Pham, Y. Largman, and A. Y. Ng, "Unsupervised feature learning for audio classification using convolutional deep belief networks," in *Advances in neural information processing systems*, pp. 1096–1104, 2009. (cited on page 5)

[3] D. Gandhi, A. Gupta, and L. Pinto, "Swoosh! rattle! thump!–actions that sound," *arXiv preprint arXiv:2007.01851*, 2020. (cited on page 5)

[4] M. J. Kochenderfer, *Decision making under uncertainty: theory and application*. MIT press, 2015. (cited on page 6)

[5] M. Hoerger, J. Song, H. Kurniawati, and A. Elfes, "Pomdp-based candy server: Lessons learned from a seven day demo," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, pp. 698–706, 2019. (cited on page 6)

[6] Z. Littlefield, D. Klimenko, H. Kurniawati, and K. E. Bekris, "The importance of a suitable distance function in belief-space planning," in *Robotics Research*, pp. 683–700, Springer, 2018. (cited on page 6)

[7] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998. (cited on pages 6 and 18)

[8] H. Kurniawati, D. Hsu, and W. S. Lee, "Sarsop: Efficient point-based pomdp planning by approximating optimally reachable belief spaces," in *Robotics: Science and systems*, vol. 2008, Zurich, Switzerland., 2008. (cited on page 6)

[9] H. Kurniawati and V. Yadav, *An online POMDP solver for uncertainty planning in dynamic environment*, pp. 611–629. Springer, 2016. (cited on pages 6, 16, and 20)

[10] D. Silver and J. Veness, "Monte-carlo planning in large pomdps," in *Advances in neural information processing systems*, pp. 2164–2172, 2010. (cited on pages 6 and 20)

[11] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "Despot: Online pomdp planning with regularization," in *Advances in neural information processing systems*, pp. 1772–1780, 2013. (cited on page 6)

[12] M. Hoerger, H. Kurniawati, and A. Elfes, "A software framework for planning under partial observability," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1–9, IEEE, 2018. (cited on pages 6 and 18)

[13] R. Bachu, S. Kopparthi, B. Adapa, and B. Barkana, "Separation of voiced and unvoiced using zero crossing rate and energy of the speech signal," in *American Society for Engineering Education (ASEE) Zone Conference Proceedings*, pp. 1–7, 2008. (cited on page 10)

[14] J. Marozeau, A. de Cheveigné, S. McAdams, and S. Winsberg, "The dependency of timbre on fundamental frequency," *The Journal of the Acoustical Society of America*, vol. 114, no. 5, pp. 2946–2957, 2003. (cited on page 10)

[15] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences," *IEEE transactions on acoustics, speech, and signal processing*, vol. 28, no. 4, pp. 357–366, 1980. (cited on page 11)

[16] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking," *IEEE Transactions on signal processing*, vol. 50, no. 2, pp. 174–188, 2002. (cited on page 16)

[17] T. J. Wark, "Robust detection and classification of objects in audio using limited training data," Aug. 28 2007. US Patent 7,263,485. (cited on page 29)

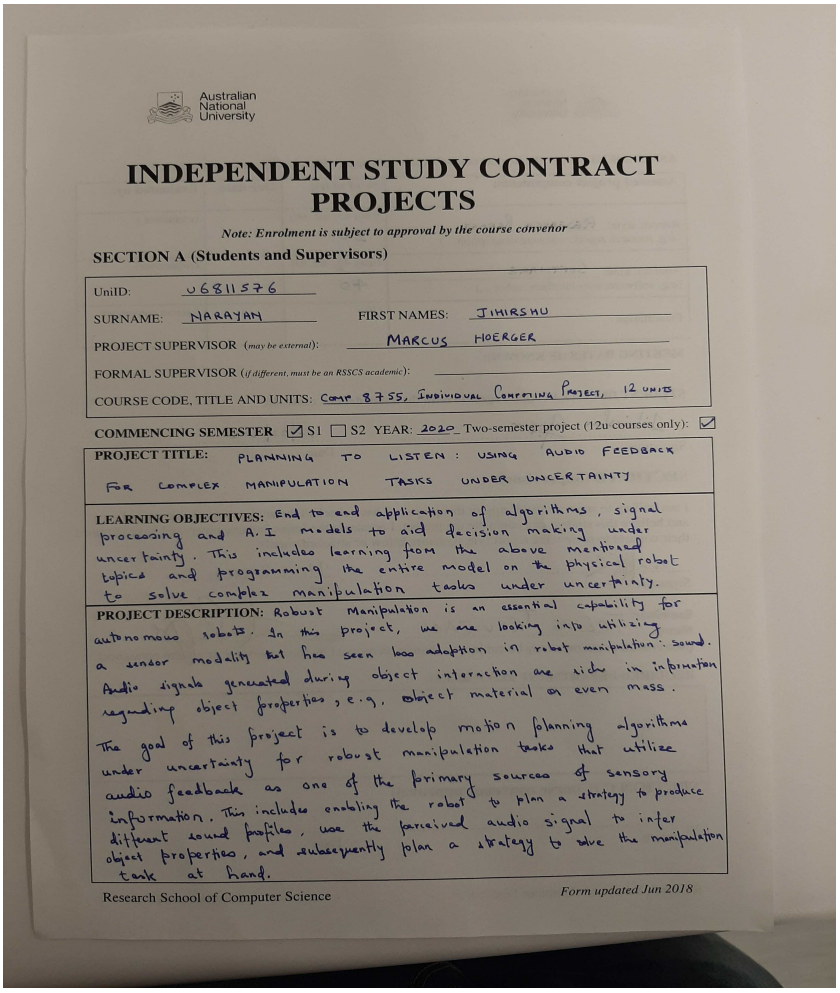
[18] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1343–1377, 2011. (cited on page 29)

Appendix 1 - Project Description

The scope of the project is the same as the problem statement described in Section 1.1. The expectation was to come up with a POMDP model that was capable of planning a robust manipulation strategy based solely on audio observations. The initial target was to distinguish two objects successfully and later on modified to 4 objects. In terms of output, it was agreed that there should be substantial work in the following areas:

- Identify potential audio features for observation model
- Select the best features which provide maximum separation
- Building a POMDP model on paper
- Implementing the said POMDP model using OPPT
- Implementing the OPPT plugins for the POMDP model
- Constructing a feasible observation model using the selected audio features
- Conducting the experiments to assess model performance

Appendix 2 - Study Contract



Australian National University

INDEPENDENT STUDY CONTRACT PROJECTS

Note: Enrolment is subject to approval by the course convenor

SECTION A (Students and Supervisors)

UniID: u6811576

SURNAME: NARAYAN FIRST NAMES: JIHIRSHU

PROJECT SUPERVISOR (*may be external*): MARCUS HOERGER

FORMAL SUPERVISOR (*if different, must be an RSSCS academic*):

COURSE CODE, TITLE AND UNITS: COMP 8755, INDIVIDUAL COMPUTING PROJECT, 12 UNITS

COMMENCING SEMESTER ☑ S1 ☐ S2 YEAR: 2020 Two-semester project (12u courses only): ☑

PROJECT TITLE: PLANNING TO LISTEN : USING AUDIO FEEDBACK FOR COMPLEX MANIPULATION TASKS UNDER UNCERTAINTY

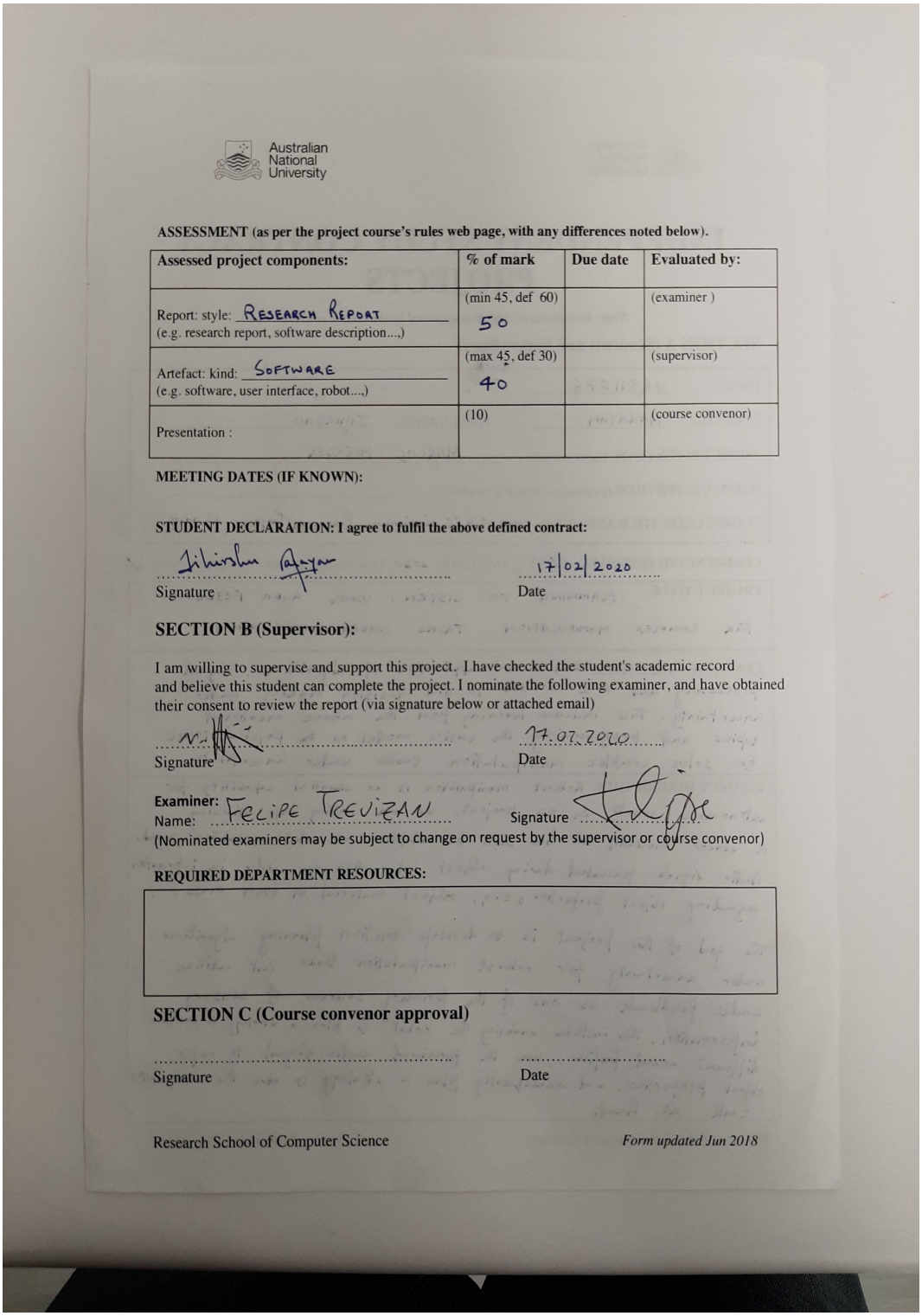
LEARNING OBJECTIVES: End to end application of algorithms, signal processing and A.I models to aid decision making under uncertainty. This includes learning from the above mentioned topics and programming the entire model on the physical robot to solve complex manipulation tasks under uncertainty.

PROJECT DESCRIPTION: Robust Manipulation is an essential capability for autonomous robots. In this project, we are looking into utilizing a sensor modality that has seen less adoption in robot manipulation : sound. Audio signals generated during object interaction are rich in information regarding object properties, e.g., object material or even mass.

The goal of this project is to develop motion planning algorithms under uncertainty for robust manipulation tasks that utilize audio feedback as one of the primary sources of sensory information. This includes enabling the robot to plan a strategy to produce different sound profiles, use the perceived audio signal to infer object properties, and subsequently plan a strategy to solve the manipulation task at hand.

Research School of Computer Science *Form updated Jun 2018*

Figure 1: Study Contract Page 1



Australian National University

ASSESSMENT (as per the project course's rules web page, with any differences noted below).

Assessed project components:	% of mark	Due date	Evaluated by:
Report: style: RESEARCH REPORT (e.g. research report, software description...,)	(min 45, def 60) 50		(examiner)
Artefact: kind: SOFTWARE (e.g. software, user interface, robot...,)	(max 45, def 30) 40		(supervisor)
Presentation :	(10)		(course convenor)

MEETING DATES (IF KNOWN):

STUDENT DECLARATION: I agree to fulfil the above defined contract:

Jihwahn Ahyan — Signature

17/02/2020 — Date

SECTION B (Supervisor):

I am willing to supervise and support this project. I have checked the student's academic record and believe this student can complete the project. I nominate the following examiner, and have obtained their consent to review the report (via signature below or attached email)

Signature

17.07.2020 — Date

Examiner:
Name: FELIPE TREVIZAN — Signature

(Nominated examiners may be subject to change on request by the supervisor or course convenor)

REQUIRED DEPARTMENT RESOURCES:

SECTION C (Course convenor approval)

Signature — Date

Research School of Computer Science — *Form updated Jun 2018*

Figure 2: Study Contract Page 2

Appendix 3 - Artefact Description

.1 Code Ownership

Git Repo

List of Code files submitted and person responsible :

Table 1: List of code modules and Individuals responsible

File Name	Author
CmakeLists	Dr. Marcus Hoerger
Robot Control API	Dr. Marcus Hoerger
Transition Plugin	Dr. Marcus Hoerger / Jihirshu Narayan
Initial Belief Plugin	Dr. Marcus Hoerger
Observation PLugin	Jihirshu Narayan
Terminal Plugin	Jihirshu Narayan
Reward Plugin	Jihirshu Narayan
ROS Observation Service Node	Jihirshu Narayan
ROS Listener Node	Jihirshu Narayan
Python Speech Features (MFCC extraction)	James Lyons *

*** https://github.com/jameslyons/python_speech_features**

.2 Experiment Details

Description of how the experiments were conducted is discussed at length in Section 4.2

.2.1 Hardware Requirements

- Kinova Movo with 7 Degree of Freedom Jaco Arm
- Planning PC
  - Hardware Specification
    - Intel Core-i7 8 core @2.8 GHz
    - 64 bit architecture
    - 16 GB RAM
    - Intel HD Integrated Graphics
    - 512 GB HDD
  - Software Requirements
    - Ubuntu 18 LTS
    - ROS Melodic(Robot Operating System)
    - Gazebo v9
    - Cmake 3.10.2
    - OPPT v0.5
    - python 2.7.17/3.6.9
    - C++ 11
    - Python libraries - librosa, numpy, matplotlib, pyAudio

Appendix 4 - Readme

listings

Code from an external file.

```
# AudioClassification


Clone the repo into a desired location. In this example we will
use the local Desktop and install the module.
    cd ~/Desktop
    git clone https://github.com/hoergems/AudioClassification.git
    mkdir Observation && cd Observation
    mkdir src
    ln -s ~/Desktop/AudioClassification/ObservationService ~/Desktop/Observation/
    catkin_make
    cd ~/Desktop
    cd AudioClassification && mkdir build && cd build
    source ~/Desktop/Observation/devel/setup.bash
    cmake -DCMAKE_INSTALL_PREFIX=<install folder> -DUSE_ARM_HACK=ON ..
    make && make install

## Usage
On the MOVO2 computer, open a terminal and run

    movostop

Then run

    roslaunch movo_bringup_simple main.launch

On your Planning PC, open 3 terminal windows. On the first window run the follow
```

```
    source <install folder>/share/oppt/setup.sh
    source ~/Desktop/ObservationService/devel/setup.bash
    cd <oppt folder>/bin
    ./abt --cfg <folder where this repo is cloned into>/cfg/AudioClassification.c

On the second window run the following commands :

    source <install folder>/share/oppt/setup.sh
    source ~/Desktop/ObservationService/devel/setup.bash
    rosrun ObservationService serviceNode.py


On the third window run the following commands :

    source <install folder>/share/oppt/setup.sh
    source ~/Desktop/ObservationService/devel/setup.bash
    rosrun ObservationService listener.py
```