Report on Research and Development

Use of Recommender System in Robotics

Deebul Nair*
Matrikel Nr.: 9023573

B-IT Master Studies Autonomous Systems

University of Applied Sciences Bonn-Rhein-Sieg
Fraunhofer Institute for Autonomous Intelligent Systems

Advisor: Prof. Dr. Paul Plöger and M.Sc. Iman Awaad

August 14, 2015

*deebul.nair@smail.inf.fh-bonn-rhein-sieg.de

Abstract

Learning from Demonstration covers methods by which a robot learns new skills using human guidance. Learning from Demonstration in robotic systems involves a number of important aspects, such as deducing the relevant features of the observations, modelling based on the relevant features and execution based on the model. We consider the problem of inferring the pertinent features using a set of demonstrations, that should be reproduced by the robot. This is challenging because the number of demonstrations on which the learning has to be made are few. In this work, we improvise an approach which is inspired by recommender systems to have an expert knowledge base. This knowledge base is then used to recommend the relevant features to imitate . The expert knowledge base is a reduced subset of the whole search space of the features to be imitated. The key novelty of our work is in the expert knowledge base used for recommendation, which is structured based on effect metrics. We argue that the structured knowledge base helps in inferring the relevant features using very less demonstrations. In this work we use the skill based framework for robot programming. Under this framework we try to learn the "move" motion primitive. The relevant features for successful reproduction of the "move" motion primitive are learnt using the method proposed. The approach was evaluated in simulation and using data acquired from the youBot robot. The results shows promising progress as it could learn with less number of demonstrations.

Contents

List of Figures

List of Tables

*Acronyms

GMM Gaussian Mixture Model. 5

HMI Human Machine Interaction. 34

HMM Hidden Markov Model. 5

KPIECE Kinodynamic Planning by Interior-Exterior Cell Exploration. 39

LfD Learning from Demonstration. 1, 3, 5, 6, 22, 33, 34, 37

nDCG normalized Discounted Cumulative Gain. iii, 22, 30

OMPL Open Motion Planning Library. 39

PDDL Planning Domain Definition Language. 36

RRT Rapidly-Exploring Random Tree. 39

STRIPS Stanford Research Institute Problem Solver. 36

TEC Theory of event coding. 7, 8

1 Introduction

Learning from Demonstration (LfD) is a technique of learning new skills based on demonstrations by a teacher[Argall et al., 2009]. In 2013, robot sales increased by 12 % to 178,132 units, by far the highest level ever recorded for one year. Sales of industrial robots to the automotive, the chemical, and the rubber and plastics industries, as well as to the food industry continued to increase in 2013. Between 2008 and 2013 the average robot sales increase was at 9.5% compounded annual growth rate per year [1]. With the increasing number of robots being deployed in the industries the amount of efforts to program the robots is also increasing. It is a major requirement to program the robots with new tasks even by persons who are non experts in robotics. LfD comes as an appropriate solution to this problem, in which a non expert can demonstrate the expected skill to the robot and the robot can learn from it.

In our work, we work on the problem of inferring the relevant aspects of the demonstration for a successful reproduction of a motion primitive. A motion primitive is defined as a single elementary movement [Schaal et al., 2000]. Motion primitive forms the building blocks for the robotic skills, which in turn forms building blocks of robotic tasks. For example we want the robot to do a task *"Get milk bottle from the refrigerator."*. This task can be broken down into various robotic skills *moveto(refrigerator) + open(refrigerator) + pickup(milk bottle) + close(refrigerator) + moveto(table)*. Each of the robotic skills can be composed of several atomic movements i.e. motion primitives . So the skill of *pickup* can be composed of *move_arm(milk bottle) + grasp(milk bottle) + move_arm(tray) + release(milk bottle)* motion primitives. Thus each task of the robot can be broken down to the level of atomic actions of the motion primitives. The motion primitives are executed in a sequence to achieve the higher goals.

Motion primitives in robotics is a highly researched topic. There are various representation of motion primitives available in the literature. In our approach we propose a feature based representation for the motion primitive. A feature is a quantitative element of the world the robot can measure. In our approach we identify the relevant features that completely describe a motion primitive. The general approach taken in machine learning for the problem is to create a large training dataset and use feature selection algorithms to determine the relevant features. But this approach cannot be used in LfD since the demonstrations which are the training dataset, are sparse.

[Abdo et al., 2014a] came up with a novel idea for determining relevant features using few demonstrations by taking inspirations from recommender systems . They build an offline expert knowledge base using experts, which is similar to the user preferences created by recommender system. Now the expert knowledge base is used to make a recommendations

[1] `http://www.ifr.org/uploads/media/Executive_Summary_WR_2014_02.pdf`

on the relevant features of the motion primitive.

Our work advances the approach of [Abdo et al., 2014a], such that the knowledge base created is more structured using effect metrics. Effect metrics are a means of measuring changes which have occurred in the robot, environment and interactions of robot-environment due to an action. The knowledge base is collected off-line and before any demonstrations. When a new motion primitive is learned, the demonstrations are shown by an user. The robot, based on the data available from demonstrations and the off-line knowledge base makes recommendations on what are the relevant features for the motion primitive.

1.1 Why not the term "recommender system" ?

The work should be ideally named as "Learning by Demonstration, Recommendation using Expert knowledge base". The name "recommender system" is an established term in literature. Recommender systems are a subclass of information filtering systems that seek to predict the 'rating' or 'preference' that a user would give to an item [Bobadilla et al., 2013]. The prediction of the preference is done based on a knowledge base generated using the user data. This knowledge base generation is always an online process. But in our work we try to create the knowledge base off-line that is based on expert knowledge. Also in our case the system recommends set of features that could be used to explain a demonstrated motion primitive by doing a greedy search on all the templates available in the knowledge base. This is not the case in recommender systems. So even though in principle the idea of both the systems are same, we would like to refrain from using the term recommender system and rather use a more generic term "recommendations using knowledge base".

2 Problem Statement

Learning from Demonstration (LfD) is defined as a technique that develops policy from examining state to action mappings [Argall et al., 2009]. The state are expressed in terms of the features and the actions are the motion primitives. The action causes changes in the state of the world. The world consist of the robot and the environment.

The LfD problem can be defined as follows [Argall et al., 2009].

The world consist of states S and actions A. The mapping between states by way of actions is defined by a probabilistic transition function $T(s`|s,a)$ ; $S \times A \times S \rightarrow [0,1]$. $s`$ is the resultant state due to the execution of the action a on the initial state s. As we all know the world is not fully observable all the time. Only a part of the world Z is observable. The learning has to be made on the observable states Z.

A policy $\pi : Z \rightarrow A$ selects action based on observations of world state. A policy determines which action a to execute from A based on the observations Z.

The demonstrations used in our approach consist of pairs of initial state and final state. Initial state is the state before the execution of an action while final state is the state after the execution of the state. The demonstrations d consist of pairs of demonstrated initial state z and final state $z`$ Demonstrations $d \in D$ formally as $k$ pairs of initial-final states: $d = (z^i, z^{`i})\text{where}(z^i, z^{`i}) \in Z\text{and}i = 0, ..., k$

2.1 Deriving a policy: Goal based mapping

In these methods of policy learning of LfD, the policy tries to predict the state of the world once the action is executed based on the current state of the world.

The perspective adopted is that multiple demonstrations of the same action can help to understand the key aspects of the action. In other words, when an action shares similar goal across multiple demonstrations, it might suggest that corresponding goal must be achieved in-order to reproduce the action. So these methods of policy learning takes the demonstrations as input and provides a list of goals to be satisfied as output for reproduction of the action.

This prediction is the passed to a planner who tries to find a plan to execute the action to reach the predicted goal state as illustrated in figure 1([Argall et al., 2009])

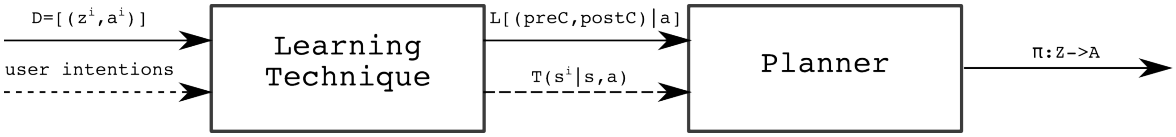


Figure 1: Policy Derivation by determining the goal of the actions [Argall et al., 2009]

2.2 Goal based mapping on relevant features

The policy learning with the complete set of features of the observed states faces a drawback that it requires a lot of demonstrations. So to learn on fewer demonstrations we need to do determine the relevant features of the states and do the policy learning only based on these relevant features.

When an action shares similar components across multiple demonstrations, it might suggest that these components are the essential features of the state that should be reproduced by the robot. In these methods the relevant features R of the states are determined based on the likelihoods of the demonstrations, before the goal state prediction.

The policy learning technique first determines which are the relevant aspects of the states for reproducing the action and these are then provided to the predictor function which predicts the goal state as illustrated in figure 2 (modified from [Argall et al., 2009])
.

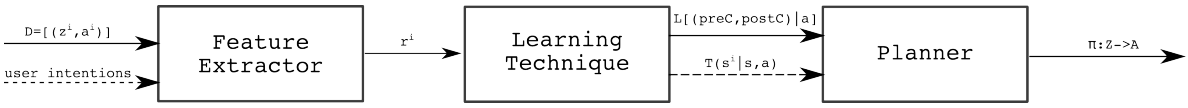


Figure 2: Policy derivation using first extracting relevant features and then determining the goal state of the robot (modified from [Argall et al., 2009])

3 Related Work

3.1 Motion Primitives

Motion primitives is a well defined concept for representing modular and re-usable skills in robotics. Motion primitive can be considered as elementary building blocks for skill representation [Kruger et al., 2010]. Motion primitives are commonly used for representing and learning skills in robotics.

There is generally two distinct approaches to represent motion primitives, one at the trajectory level and the other at the symbolic level [Argall et al., 2009]. One of the early techniques in the trajectory level approaches was as a spline by [Ude, 1993]. The other most common method is to represent motion primitives as a probabilistic representation [Calinon, 2009]. Gaussian Mixture Model (GMM) is used to encode motion primitives through probabilistic representation. GMM allowed autonomous and incremental construction of the motion primitives. This was beneficial for LfD. Hidden Markov Model (HMM) is used to encode motion primitives. In HMM the motion primitives are represented as a single multivariate Gaussian Distribution. [Paraschos et al., 2013] introduced the concept of probabilistic movement primitives (Promotion primitives) as generalistic framework for representing and learning motion primitives.

A popular alternative to this approach has considered modelling the intrinsic dynamics of motion [Schaal et al., 2000] . This approach was advantageous for LfD as it was time-independent and can be modulated to produce trajectories with similar dynamics in area of workspace not covered during learning. One of the drawbacks of the trajectory based techniques was that it lacks cognitive sense [Aein et al., 2013]. These drawbacks are also been addressed in current research activities like [Manschitz et al., 2015] where the transition between the motion primitives are also being learned to execute complex tasks.

In symbolic representations a compact descriptions of motion primitives is defined, but the execution details are not included. A large body of work uses symbolic representation for encoding motion primitives ([Muench et al., 1994], [Friedrich et al., 1996], [Pardowitz et al., 2007]). [Alissandrakis et al., 2005] encoded human motions as pre-defined postures, positions or configurations. [Aein et al., 2013] creates a library of motion primitives. In a more recent work by [Andersen et al., 2014] the skill based framework is used to learn predefined skills. The skills are made of combinations of motion primitives. The configuration parameters of the motion primitives are learnt by demonstrations. The paper proposes an approach to represent skill for learning inside a skill based framework. But in the above work the representation of motion primitives in a skill based framework is not well defined.

3.1.1 Proposed Motion Primitive

We represent motion primitives as foundation blocks for a robot to complete a skill. For example consider a "pick up skill" can be executed using combinations of "move" Motion primitive. So a *pick up screw* skill gets fragmented as *move arm camera pose + move arm grasp pose + move arm place pose*. Which motion primitives are available to a robot depends on the hardware specification of the robot.

Each motion primitives has a pre-condition and post-conditions to ensure and verify a proper functioning. Using LfD we try to learn the post-condition(goal) of the motion primitives. The motion primitives in our framework consist of a teaching part and an execution phase, as illustrated in Figure 3 (extended from [Bogh et al., 2012] and [Andersen et al., 2014]) The motion primitive using this approach is well suited for the skill based framework as explained in [Pedersen et al., 2015].

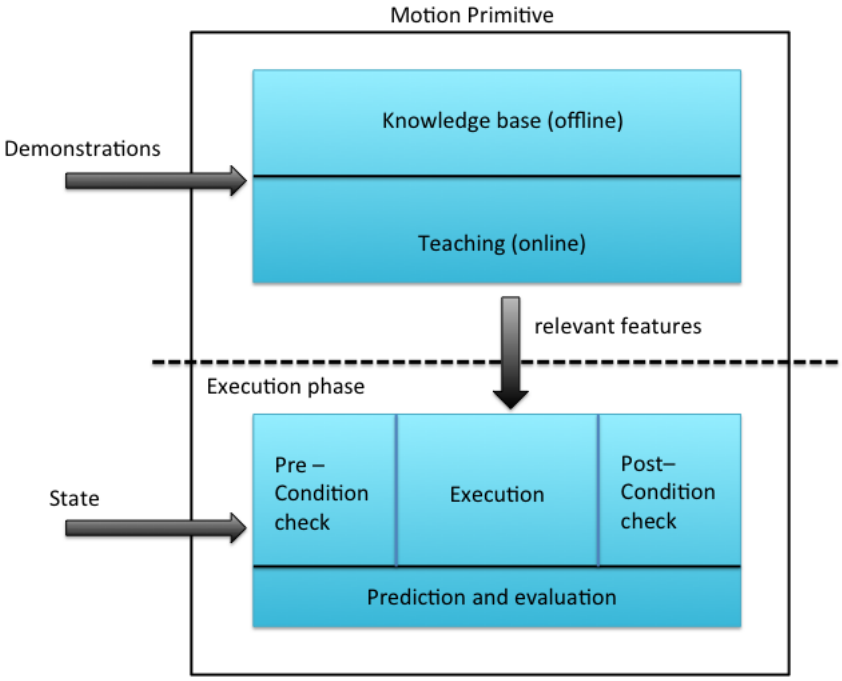


Figure 3: Proposed general structure of a motion primitive. A motion primitive consist of Learning Phase and Execution Phase.

The motion primitive as shown in figure 3 consist of two parts the learning part and the execution part. In the learning part the demonstrations are taken as an input and the relevant features are extracted. This extracted features are then fed to the execution part. The execution part is represented as a combination of pre and post conditions of features. Based on the relevant features the execution phase predicts the post conditions

and executes the motion primitive. In section 4 the mathematical formulation of the motion primitive is discussed in details.

3.2 Goal Based Learning from Demonstration

A large body of work uses goal based or plan based representation of learning ([Friedrich and Dillmann, 1995], [Nicolescu and Mataric, 2003], [Csibra and Gergely, 2007], [Demiris, 2007], [Veeraraghavan and Veloso, 2008], [Lee et al., 2009], [Hommel, 2009], [Abdo et al., 2013]). The majority of the work in these try to infer the underlying goals and intent of the demonstrator. The terms goals and intent are used interchangeable; but goal refers to the immediate desirable state while intent refers to longer term desirable state. [Tomasello et al., 2005] defines intent as *" a plan of action the organisam chooses and commits itself to the pursuit of a goal, an intention thus includes both a means(action plan) as well as goal"*(p676).

The idea of learning goals is inspired by learning techniques in humans. It has been studies that humans show a strong and early indication to interpret observed behaviours of others as goal directed actions. This is termed as 'teleological obsession' [Csibra and Gergely, 2007] . They argue this 'teleological obsession' serves for on-line prediction and social learning.

The problem of recognizing goals and intentions of the action can be formulated as a problem of model matching. The agent which is observing deploys sensors, each reporting its observations of the state. Based on the state observations, two approaches are available for analysis *descriptive* and *generative* [Demiris, 2007]

In *generative* approach the observed states are mapped into a latent(hidden) space. The latent space is usually of lower dimension than the observation space. The main problem of inferring in the observation space is the dimensionality of the space. So by reducing the observations to latent space learning is made on smaller dimensions. Generative model is highly popular in machine learning community and robotics community ([Buxton, 2003], [Bishop, 2006] , [Calinon, 2009])

In contrast, in the *descriptive* approach, the patterns are characterised by extraction of a number of low level features, and to use the set of restrictions of the features level([Isham, 1981], [Jain et al., 1998], [Abdo et al., 2013]). The observer agent subsequently matches the observed data against pre-existing representations and depending on what the task is show generates the action corresponding to their representations. Pre -existing representations can have associated data that label these representations with the goals, beliefs and intentions that underlie in the demonstrations. This approach corresponds to the Theory of event coding (TEC) method for intention interpretation by [Csibra and Gergely, 2007]. TEC was formulated to provide an alternative prespective

that allows to take intentions and the goal directed nature of action into consideration. TEC explains how human learning works; basically how human action is anticipatory in nature, how anticipation emerge from experience and, how anticipation comes to regulate human behaviour.

Our work is an extension of the *descriptive* approach of learning. We also learn using the restircted feature space.

3.2.1 Proposed Expert Knowledge Base

A recent work by [Abdo et al., 2014a] leverages feedback from experts to create a recommendations and use this recommendations to recommend features for actions. Our approach advances over [Abdo et al., 2014a], by creating a more structured knowledge base . The expert knowledge base used in the previous is not structured. In this work we try to create a more structured generic framework based on effect metrics concept. The expert knowledge base is explained in details in the section 4.

3.3 Recommender system in Robotics

Use of the recommender system in robotics is a very recent affair. Recommender system is a special branch in machine learning which specializes in determining the user preferences, based on previous activites of the user. It then uses this user preferences to predict products(movies, videos, news etc) which the user will like to see or purchase. Recommender systems are broadly classified into two types *Content Based Recommender System* and *Collabrative Recommender System* [Bobadilla et al., 2013]. The use of recommender system in robotics is relative a new approach. The first to use was by [Matikainen, 2012], who used it to the model recommendation problem. They tried to recommend based on previous history which machine learning algorithm to use for a specific image processing situation. In [Matikainen et al., 2013] they tried to solve the n-bandit problem of selecting food floor coverage statistics using recommender system.

Further Recommender system was used to learn user preferences of the users for doing daily chorus for robots. [Abdo et al., 2014b] and [Abdo et al., 2015] tried to learn user preferences concerning a given tasks using small number of known preferences. Using recommender system they enable robots to predict user preferences with respect to tidying up objects in containers, such as shelves or boxes.

4 Learning Motion Primitive using Kinesthetic Demonstrations

The main idea of the work is to infer the relevant set of features, which describe the demonstrated motion primitive. Based on the learnt relevant features, during the execution phase the robot tries to predict the goal of the motion primitive, starting from a new initial condition.

The demonstrations consist of recording two states of the world called as the start state and the end state. So each demonstrations have 2 snapshots of the world in the initial state and the final state. Based on these snapshots we try to infer the intent of the demonstrations.

The motion primitive in our representation consist of pre-conditions and the post-conditions. The execution of each motion primitive depends on the features used to explain it. Each motion primitive is defined by a set of features which are a subset of the complete feature space, which are also relevant for the robot to execute the respective motion primitive.

4.1 Features

Features are the basic building block, which are used to describe a motion primitive. A Feature f can be defined as any quantitative parameter of the world W. For example the pose of the robot, color of the box, distance between box and robot, displacement of the tooltip in time.

Features are broadly classified into :

1. features describing object properties $f(O)$
   (a) features describing robot properties $f(O_r)$ (eg : pose of the robot)
   (b) features describing environment properties $f(O_e)$(eg: color of the box)
2. features describing relation between object properties $g(O_1, O_2)$
   (a) relation between different objects $g(O_1, O_2)$ (eg : distance between box and robot)
   (b) relation between same object in time $g(O_{1i}, O_{1f})$(eg : displacement of tooltip)

Thus the collection of featues can be defined as

$$\begin{aligned} F &= f(O) + g(O_1, O_2) \\ &= (f(O_r) + f(O_e)) + (g(O_1, O_2) + g(O_{1i}, O_{1f})) \end{aligned} \tag{1}$$

As explained in section 3.1.1 the proposed motion primitive is defined using pre and post conditions of the features.

$$\text{Motion primitive} := (f_s, f_e)$$

$$where$$

$$f_s = \text{features in start of demonstrations}$$

$$f_e = \text{features in end of demonstrations}$$

4.2 Modelling Motion Primitive

The learning problem can be considered as a supervised learning problem, where the model has to learn from the set of demonstrations to predict the output f_e when a new input f_s is provided as explained in figure 4.

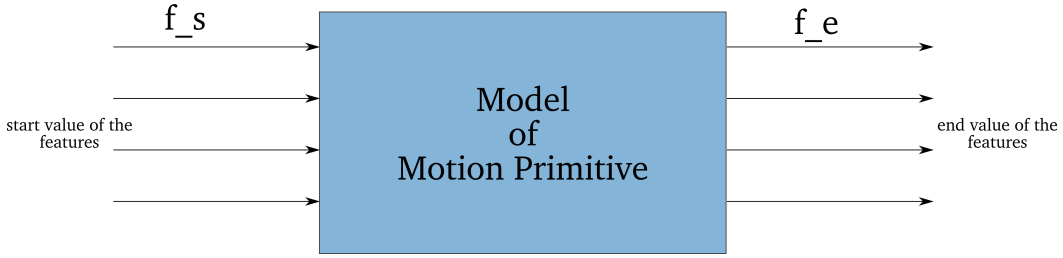


Figure 4: Model of motion primitive

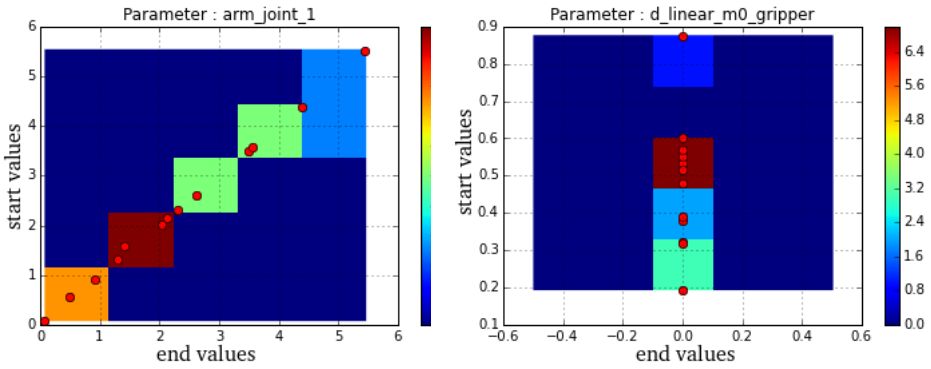
When the number of training set is limited,learning on all the features of the world is not a feasible solutions. The approach taken here is to find the relevant features on which the learning can be made most effectively. The prespective adopted in finding the relevant features is that in multiple demonstrations of the same action, some features will share similar values in all the demonstrations. Since we are demonstrating a single action there has to be consistency in some features of the action. We need to identify these consistent features and these become the relevant features. The consistency increases the relevance of the feature to sucessfully predict from new unseen start states.

We will consider a bivariate probability density function ϕ_i by considering 2 variables, the value of a feature in start of demonstration and its value at the end of the demonstration.

Let s_i be the i^{th} feature of f_s, and e_i be the i^{th} feature of f_e,
The bivariate distribution is given by :

$$\phi_i(s_i, r_i) = \eta_i I_i(\lfloor s_i \rfloor, \lfloor e_i \rfloor) \tag{2}$$

where operator $\lfloor . \rfloor$ is a quantization operator, that returns the bin unit in the histogram



(a) Distribution of the feature describing arm joint 1.

(b) Distribution of feature describing distance of object 0 and tooltip.

Figure 5: An example of bivariate distributions of two features for the *move* arm relative to object motion primitive.The x-axis is the end values of demonstration while the y-axis is the start value of the demonstration. The dots are the feature values across 12 demonstrations. Based on the distribution in 5a, we can say that there is no certaininty in the end values. This results in lower relvance to the motion primitive . Based on the distribution in 5b,we can say that the end states are more concentrated.This ensures that for any value in the start state we are sure that the end values always remain same. This results in a higher relevance with respect to the motion primitive.

I_i, that corresponds to the feature.

The bi-variate distribution is an appropriate criteria to comment on the relevance of the feature. Using the distribution we can conclude on the convergence of the feature. A feature which has converged, the final values will lie on a straight line in the distribution. If all the final values lie on a straight line it ensures that whatever maybe the initial value the end values always remain constant. The constant value in all the demonstration concludes that the corresponding feature is relevant with respect to the intent of the motion primitive. This is explained in the figure 5. Higher the degree of convergence, higher the relevance of the feature for the motion primitive.

To calculate this relevance in the distribution, we use two different measuring methods. 1) Entropy 2) Conditional entropy. We compute the entropy H_i and conditional entropy CH_i of the bi-variate distribution $\phi_i(s_i, r_i)$.

Entropy of a discrete random variable is given by,

$$H(X) = -\sum P(X) \log P(X) \tag{3}$$

For a pair of discrete random variables X and Y, which are co related conditional

entropy of X given Y $h(X|Y)$ is given by :

$$H(X|Y) = -\sum_{k=-K}^{K} p_X(x|y)p_Y(y|x)\log\frac{p_Y(y|x)}{p_X(x|y)p_Y(y|x)} \tag{4}$$

Based on the entropy or conditional entropy we define the model relevance of a motion primitive.

$$p(f|\theta) = \prod_i e^{-E_i} \tag{5}$$

where θ is the set of all ϕ, and E_i can be H_i or CH_i

4.3 Expert knowledge base

In our work we create the knowledge base based on expert knowledge of the tasks being performed. An individual subset of feature space F is called as a template t. $t_i \subset F$. For example the template t_1 contains all the features describing the joint angles of the robot, template t_2 contains features which describes distance of tooltip to manipulated object.

A collection of the templates is called as a knowledge base K.

$K = t_1 t_2 \dots t_n$

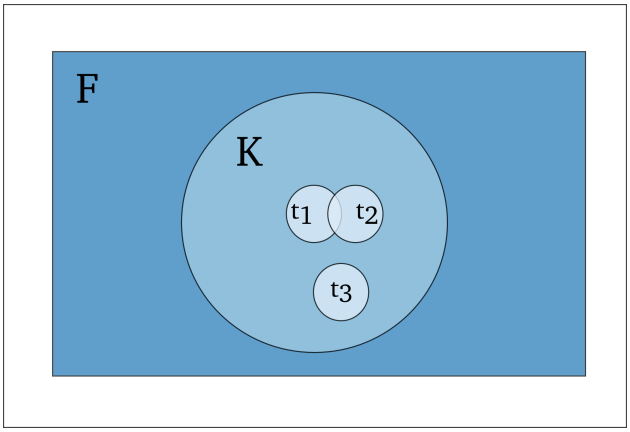


Figure 6: Relation between feature space F, knowledge base K and the templates t

Thus for each template t_i we have created a low dimension subset of the feature space F. Our aim is not to find minimum set of features but to find relevant set of features which can describe an action.

The novelty in our approach is the representation of the expert knowledge base. The knowledge base has been structured using the effect metrics. The templates in the knowledge base are organised on the basis of the effect metrics. The structured nature of the knowledge base helps in determining the relevant features with less number of demonstrations.

4.3.1 Effect Metrics

Effects are defined as changes to the robot-world relationship and/or to positions, orientations and states of external objects and robot [Alissandrakis et al., 2006]

In this work we only consider changes to the robot-world relationship and changes to positions, orientations and states of the robot. Based on the orientation and position of the robot in environment 2 types of effect metrics can be used, *position* and *orientation*

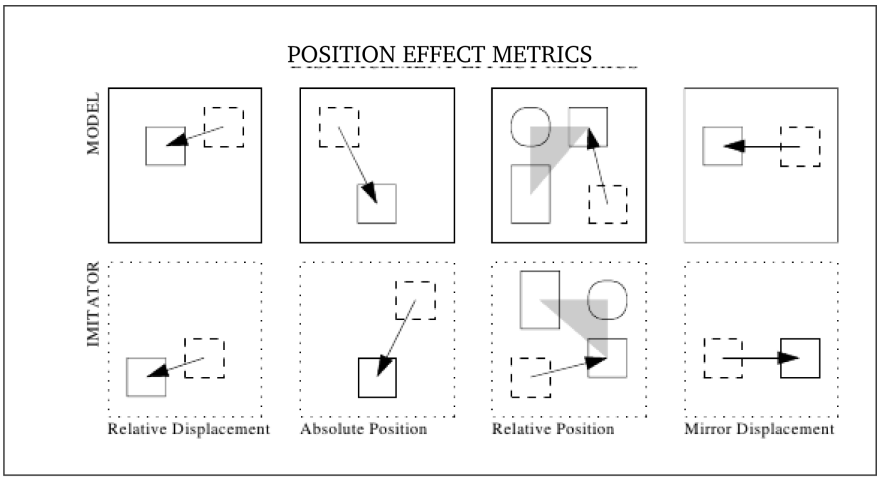


Figure 7: Position effect metrics. To measure the change in the relation of change in positions between robot and environment objects *relative displacement, absolute position, relative position and mirror position* effect metrics can be used. First row shows the demonstration and their resulting effects. The second rows represents the corresponding object (in different workspace) how it needs to be moved (from dashed to solid outline) by an imitator to match the corresponding effects according to the metric. The grey triangle are superimposed to show the relative position of the objects are the same in final state. [Alissandrakis et al., 2006]

The different position metrics are

- relative displacement
- absolute position
- relative position
- mirror position

the different orientation metrics are

- relative rotation
- absolute orientation

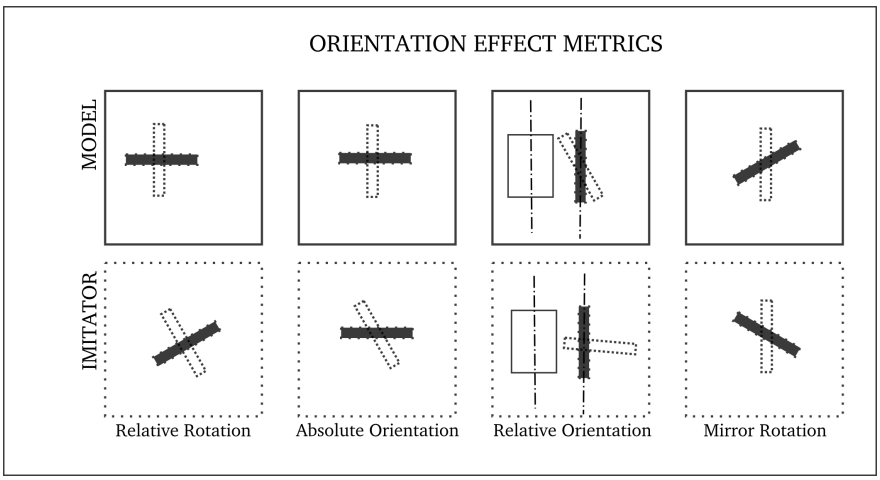


Figure 8: Orientation effect metrics. To measure the change in the relation of change in orientation between robot and environment objects *relative rotation, absolute orientation, relative orientation and mirror rotation* effect metrics can be used.First row shows the demonstration and their resulting effects. The second rows represents the corresponding object (in different workspace) how it needs to be moved. (from dashed to solid outline) by an imitator to match the corresponding effects according to the metric. The guide lines are superimposed to show the relative orientation of the objects are the same in final state. [Alissandrakis et al., 2006]

- relative orientation
- mirror rotation

Depending on the effect metric the same demonstration can be interpreted as different. The example in figure 9 explains this.

Based on the effect metrics the knowledge base is organized in different templates which describe an action. The templates developed in this work are categorized based on the effect metrics.

4.4 Selecting template from the knowledge base

For computing the relevance, we use a 3 stage procedure:

1. Comparing mean of start value and end value of the feature. → To determine that the feature has changed. (This was used because features which were not changed in demonstration also have low entropy. So to remove these features this condition check was introduced.)

2. Comparing the standard deviation of start value and end value of the featue → To determing the feature shows convergence in information. (This was used because

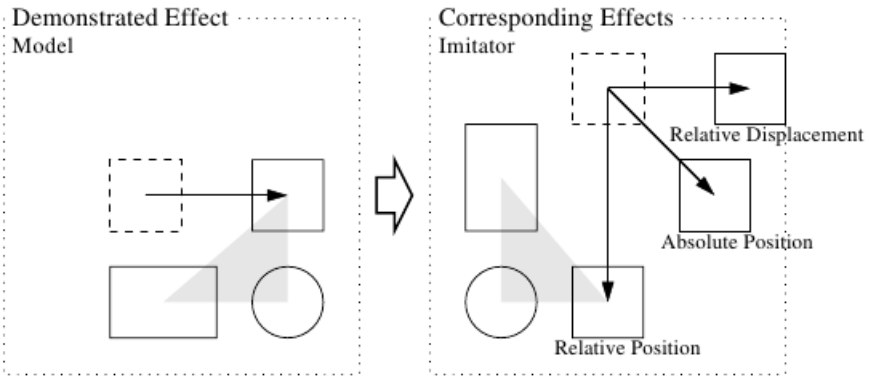
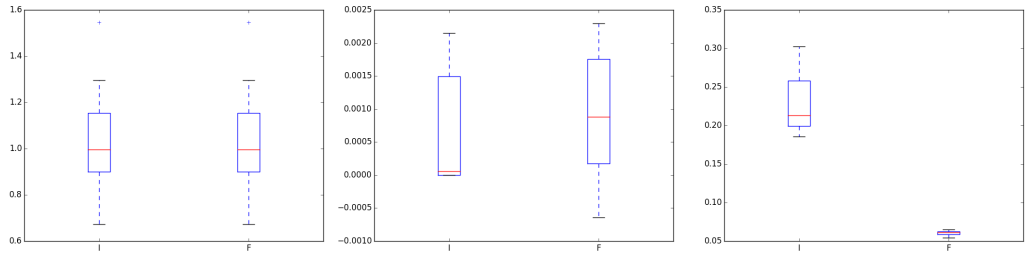


Figure 9: The figure illustrates three examples of the imitation based on a single demonstration. The left box is the demonstrated action. The right box is the possible final state based on the position metrics used. The grey triangles are superimposed to show the relative distance is maintained [Alissandrakis et al., 2006]

there were features which changed and had low entropy but they were divergent in nature. So to remove the effect of these features this condition check was introduced as explained in figure 10)

3. Calculating the entropy/conditional entropy of the final value of features whose values have changed.



(a) (b) (c)

Figure 10: Box plots of features. Each figure has 2 box plots. Plot I is of the start value of the feature while plot F is of the end value of the feature. Figure 10a is box plot of a featue whose mean values have not changed for both start and end values of the feature . Figure 10b is the box plot of a feature whose mean value has changed but the final values has not converged. Figure 10c is the box plot of a feature whose mean value has changed and the final values have converged.

Data:
f_s : start value of the features
f_e : end value of the features
K : knowledge base
Result:
relevance L of each Template
R=0;
for *each template t in K* **do**
  for *each feature f in t* **do**
    read start values s of feature f from f_s;
    read end values e of featue f from f_e;
    if *mean(s) equals mean(e)* **then**
      $R+ = 1$;
    end
    if *standard deviation(s) is less or equal to standard deviation(e)* **then**
      $R+ = 1$;
    end
    $R+ =$ entropy(e)*or* conditional entropy($e|s$);
  end
end

Algorithm 1: Algorithm for computing relevance of the templates in knowledge base

For each $t_i \in K$, we compute a score β_i that combines model fitting and the number of features in the template.

$$\beta_i = -2\log(p(f|\theta_i)) - \alpha_i L_i \tag{6}$$

where θ_i are the distributions related to the features of t_i and L_i is the number of features in the template. The first term of equation computes relevance and the second term, weighted by α, encourages the usage of templates consisting of a large number of features L_i .

The most relevant template T^* is selected as :

$$T^* = \underset{i}{\operatorname{argmin}}(\beta_i) \tag{7}$$

5 Experiments and Evalutaion

The proposed motion primitive representation and recommending valid features using modified knowledge base was validated by learning the *move* motion primitive on both simulation and on the real robot youBot.

5.1 Experiment Setup

The experiments were conducted on data obtained from both simulation as well on the real robot. The learning of the movement of the base was done in the simulation software. Gazebo [2] simulator was used to simulate the workspace of the robot. The model of the workspace used was taken from the robocup@work competition [3]. An illustration of the workspace is shown in figure 11. The demonstrations of moving the base was done by teleoperation using a joy-pad.

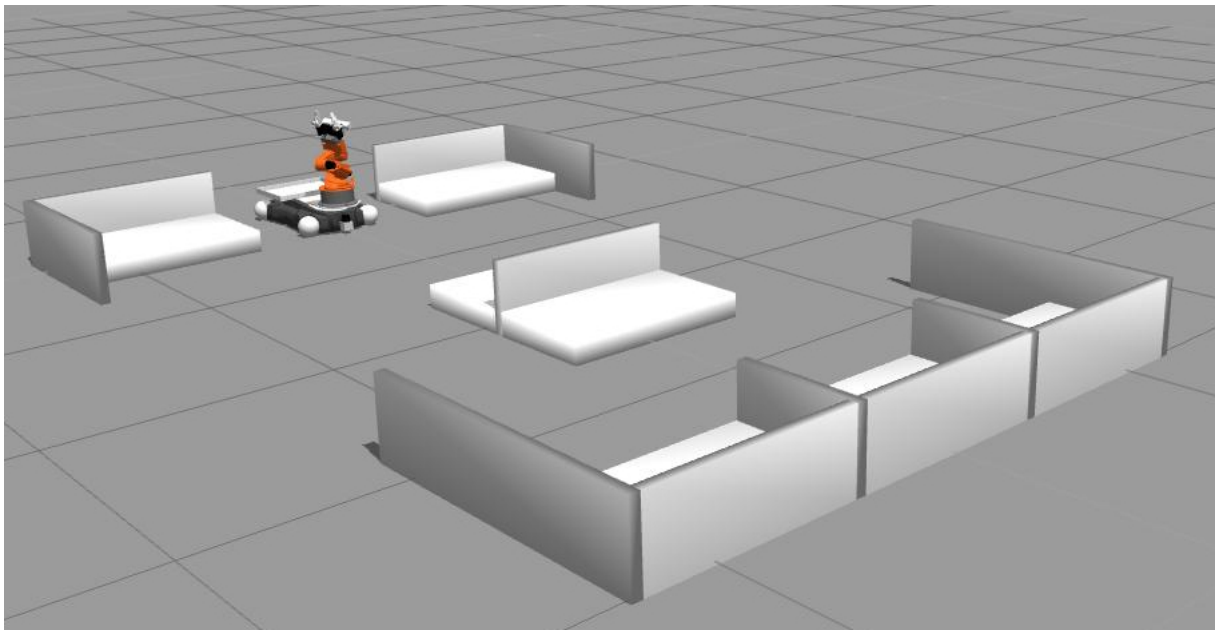


Figure 11: Gazebo simulator and robocup@work workspace with KUKA youBot

The learning of the arm movements was done on the real robot. The robot used for experimentation is the KUKA youBot [4]. The youBot has an omnidirectional mobile platform on which a five-axis robot arm is mounted. The arm was used to teach the move motion primitive . Kinesthetic demonstrations was used for teaching various arm poses as shown in figure 12. As we do not address perception in our scope of our work, we used

[2]http://gazebosim.org/
[3]http://www.robocupatwork.org/resources.html

QR markers attached to the objects and measured their pose using the camera mounted on the robot's arm.

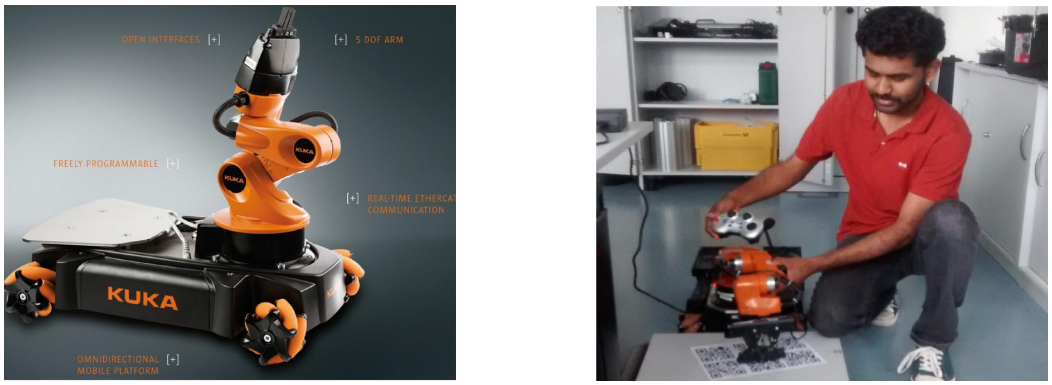


Figure 12: a) KUKA youBot [5]. b) The teacher demonstration manipulative motion primitive to the youBot

5.1.1 Steps to recording demonstrations

The demonstrations used for learning consist of two parts. The first part is the start values and the second part is the end values. For each demonstration the values of the features are recorded first in the start state and then in the end state. The steps involved in recording the demonstrations is explained in the figure 13

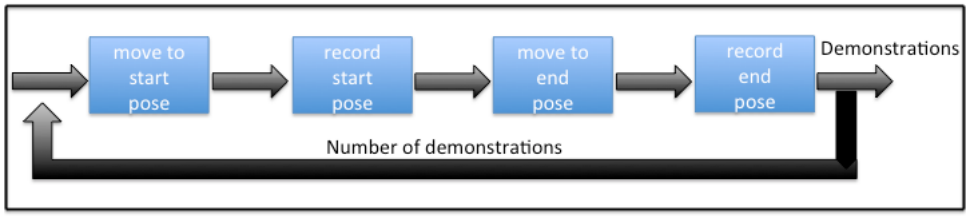


Figure 13: Steps involved in recording the demonstrations

5.2 Experiments

The experiments conducted for the work were to teach some of the motion primitives related to the tasks in RoboCup@Work competition.

[5]http://www.kuka-robotics.com/germany/en/products/education/youBot/

5.2.1 RoboCup@Work

Robocup@work competition [6] is conducted on various tasks, which are involved in work-related scenarios. The tasks involved in the competition are taken directly from a list of scientifically and industrial challenges. The table 1 list the various test conducted during the competition. In our experiments we tried to learn the motion primitives related to the task executed in the competition.

Task	**Description**
Basic Navigation Test	can the robots navigate well in their environment, i.e. in a goal-oriented, autonomous, robust, and safe way
Basic Manipulation Test	demonstrate basic manipulation capabilities by the robots, like grasping, turning, or placing an object
Basic Transportation Test	assess the ability of the robots for combined navigation and manipulation tasks
Precision Placement Test	drop objects precisely into cavities

Table 1: Robocup Task list

5.2.2 Motion primitives learnt

The motion primitives which were learnt for validating the approach are as follows

1. *move* base to absolute pose
2. *move* base to relative pose to object
3. *move* arm to absolute pose
4. *move* arm to relative pose to object

5.3 *move* Motion Primitive

The *move* motion primitive is an important building block for all the skills used in an industrial setup. For example lets consider the 'pick up' skill in an industrial setup. This skill can be fragmented into the following motion primitives:

- **pick up skill :**
    1. *move* base to platform

[6] http://www.robocupatwork.org/

2. *move* arm to "look platform" pose
3. locate objects
4. *move* arm to pre-grasp pose
5. grasp
6. *move* arm to post-grasp pose

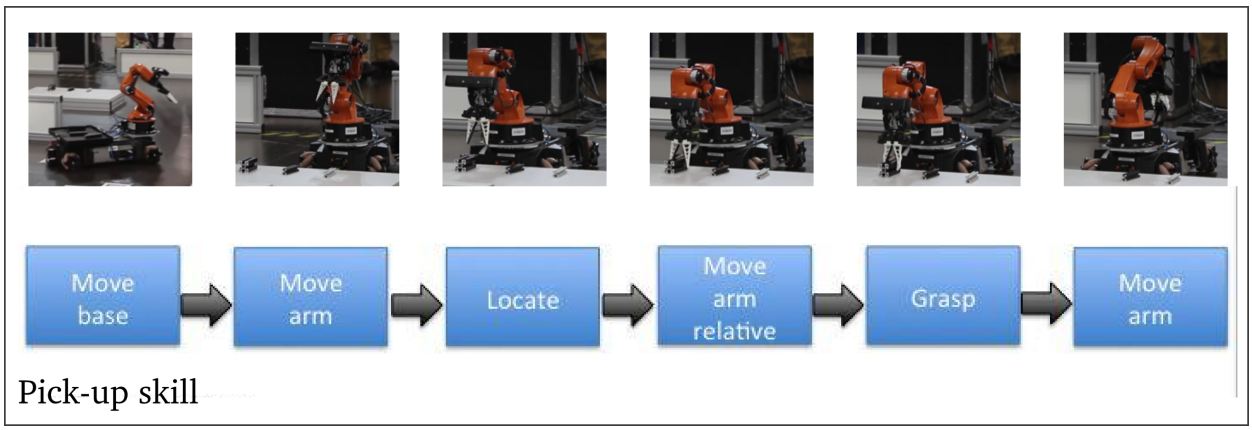


Figure 14: 'Pick up' skill : The pick up skill consist of a series of motion primitives. The motion primitives and their sequence for executing 'pick up' skill. Images courtesy b-it bots robocup@work (German open 2015)

The *move* motion primitive can be broadly classified for a mobile manipulator for 2 purposes.

1. Base movements
2. Arm movements

The *move* motion primitive can be further classified based on the effect metrics

1. Absolute pose
    - position
    - orientation
2. Relative pose
    - position
    - orientation
3. Relative displacement

5.3.1 *move* Knowledge base

The template as defined earlier is a subset of the feature space, which defines a particular motion primitive. A collection of templates, form the knowledge base for a particular robot. The templates formulated for the *move* motion primitive in youBot are mentioned in the table 2

5.4 Evaluation metrics

We designed a dataset to quantitatively evaluated the performance of the recommendations made by our algorithm. An expert labeled the templates in the knowledge base on a 3 point Likert scale of 1-3 (where 3 is the best) on the basis of the his expert knowledge. We use this metrics to quantify the quality of a ranked list of recommendations by its normalized Discounted Cumulative Gain (nDCG) [Manning, 2008] at positions 1 and 3. nDCG@1 is suitable metric for LfD since we use the top recommendation for finding our goal. We consider nDCG@3 suitable for evaluating the quality of the recommendations.

5.5 Results and Discussions

This section summarizes the evaluation of our approach conducted using kinesthetic demonstrations of the *move* motion primitive.

5.5.1 *move* base absolute pose

In this experiment the youBot is taught to *move* base to an absolute location in the workspace. The experiments consist of recording different poses of the robot in the workspace. First the robot is moved to any arbitary position and the start pose is recorded then the robot is moved to the intended final position which needs to be taught and the end pose is recorded. The above steps are repeated till all the observations are recorded as shown in figure 15. The motion primitive can be used as a part of the "basic navigation test" task. In these task the robot has to go to pre-defined locations in the workspace. We ran the experiments using different number of demonstrations and using both entropy and conditional entropy for measuring relevance. The success of the experiments are shown in figure 16. The template describing the pose of the base of the youBot was recommended as the most relevant template from the demonstrations.

move Knowledge Base				
part	metric	type	features	description
arm	absolute	position	joint 1, joint 2, joint 3, joint 4, joint 5	Joint angles of the youBot
		position	tooltip link	tool tip frame of youBot
		orientation	tooltip link	tool tip frame of youBot
		pose	tooltip link	tool tip frame of youBot
	relative	position	tooltip link, object 1	Relative distance between position of tooltip and object 1
		orientation	tooltip link, object 1	Relative distance between orientation of tooltip and object 1
		pose	tooltip link, object 1	Relative distance between pose of tooltip and object 1
		position	tooltip link, object 2	Relative distance between position of tooltip and object 2
		orientation	tooltip link, object 2	Relative distance between orientation of tooltip and object 2
		pose	tooltip link, object 2	Relative distance between pose of tooltip and object 2
		position	tooltip link, object 1, object 2	Relative distance between position of tooltip and object 1, object 2
		orientation	tooltip link, object 1, object 2	Relative distance between orientation of tooltip and object 1, object 2
		pose	tooltip link, object 1, object 2	Relative distance between pose of tooltip and object 1, object 2
		displacement	tooltip link	tool tip frame of youBot
base	absolute	position	base link	base link frame of youBot
		orientation	base link	base link frame of youBot
		pose	base link	base link frame of youBot
	relative	position	base link, object 1	Relative distance between position of base and object 1
		orientation	base link, object 1	Relative distance between orientation of base and object 1
		pose	base link, object 1	Relative distance between pose of base and object 1
		position	base link, object 2	Relative distance between position of base and object 2
		orientation	base link, object 2	Relative distance between orientation of base and object 2
		pose	base link, object 2	Relative distance between pose of base and object 2
		position	base link, object 1, object 2	Relative distance between position of base and object 1, object 2
		orientation	base link, object 1, object 2	Relative distance between orientation of base and object 1, object 2
		pose	base link, object 1, object 2	Relative distance between pose of base and object 1, object 2
		displacement	base link	base link frame of youBot

Table 2: Knowledge base for *move* motion primitive

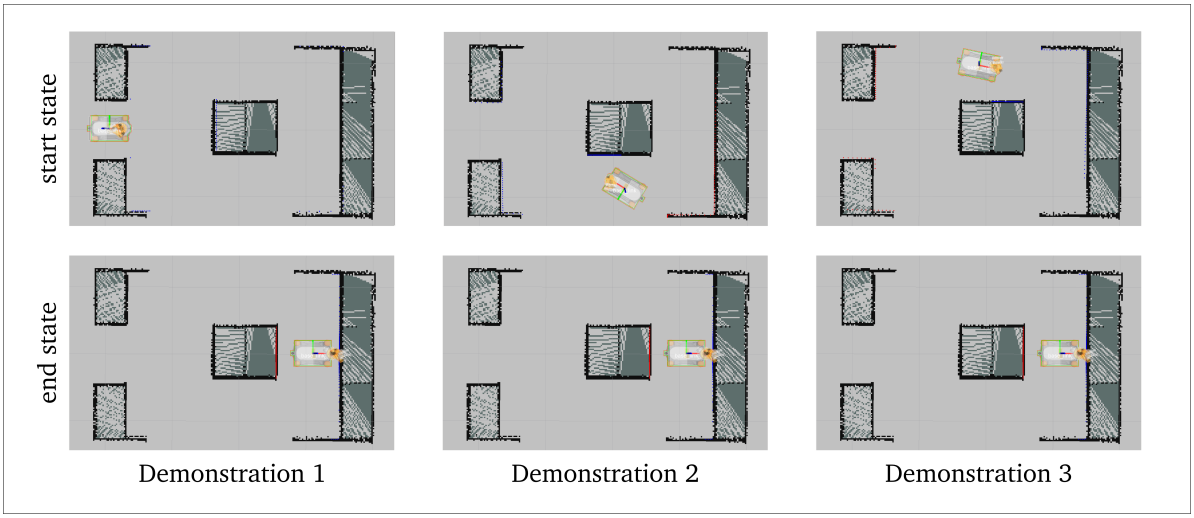


Figure 15: Demonstrations for teaching the robot to *move* base to an absolute pose in the workspace. The rows show the recorded start pose and the end pose. The columns show the different demonstrations

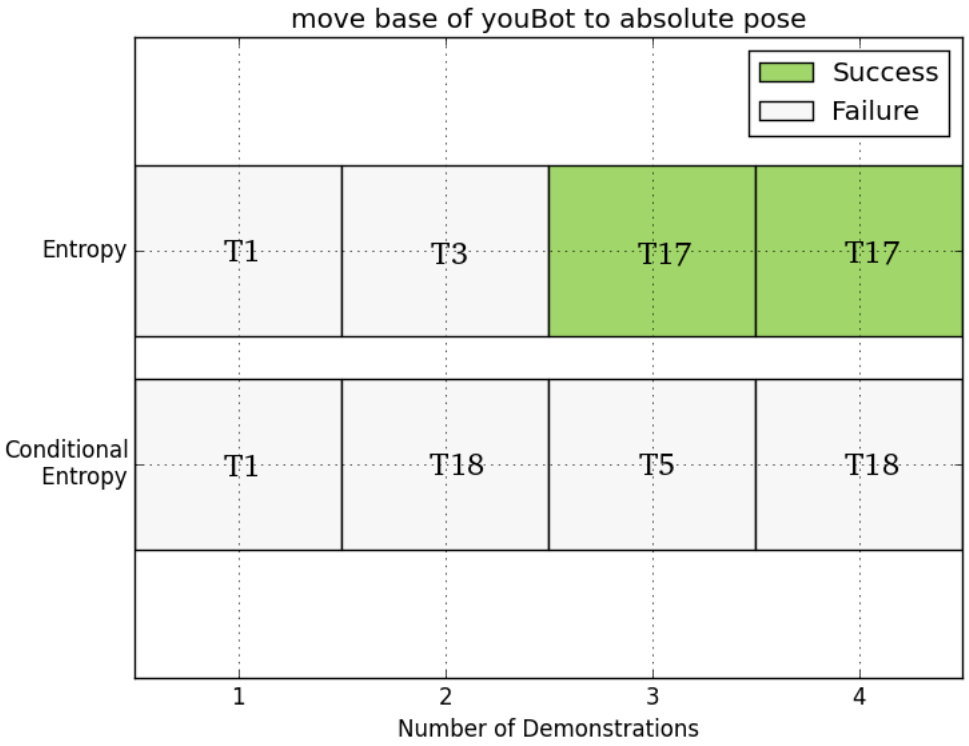


Figure 16: Results for *move* base to an absolute pose. The x-axis denotes the number of demonstrations used for learning. The y-axis denotes the method used for computing relevance.Each box is marked with the selected template name. Green color denotes the relavant template was successfully recommended

5.5.2 *move* base relative pose to object

In this experiment the youBot is taught to *move* base to a relative location in the workspace. The experiments consist of recording different start and end poses of the robot in the workspace. First the robot is moved to any arbitary position and the start pose is recorded then the robot is moved to the intended final position which needs to be taught and the end pose is recorded. The above steps are repeated till all the observations are recorded as displayed in figure . As you can see the object was placed on different platform on each demonstrations.

This particular motion primitive can be used as a part of the "basic manipulation test" task. In these task the robot has to go near a platform containing an object. We ran the experiments using different number of demonstrations and using both entropy and conditional entropy for measuring relevance. The success of the experiments are shown in figure 18. The template describing the relative distance between the base of youBot and the object was recommended as the most relevant template from the demonstrations.

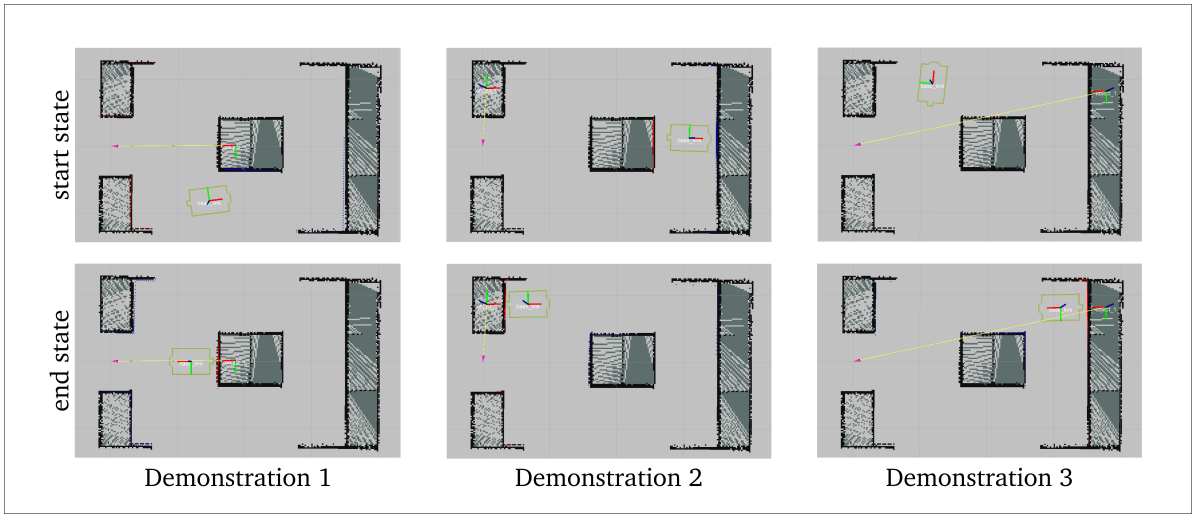


Figure 17: Demonstrations for teaching the robot to *move* base to a relative pose in the workspace. The rows show the recorded start pose and the end pose. The columns show the different demonstrations

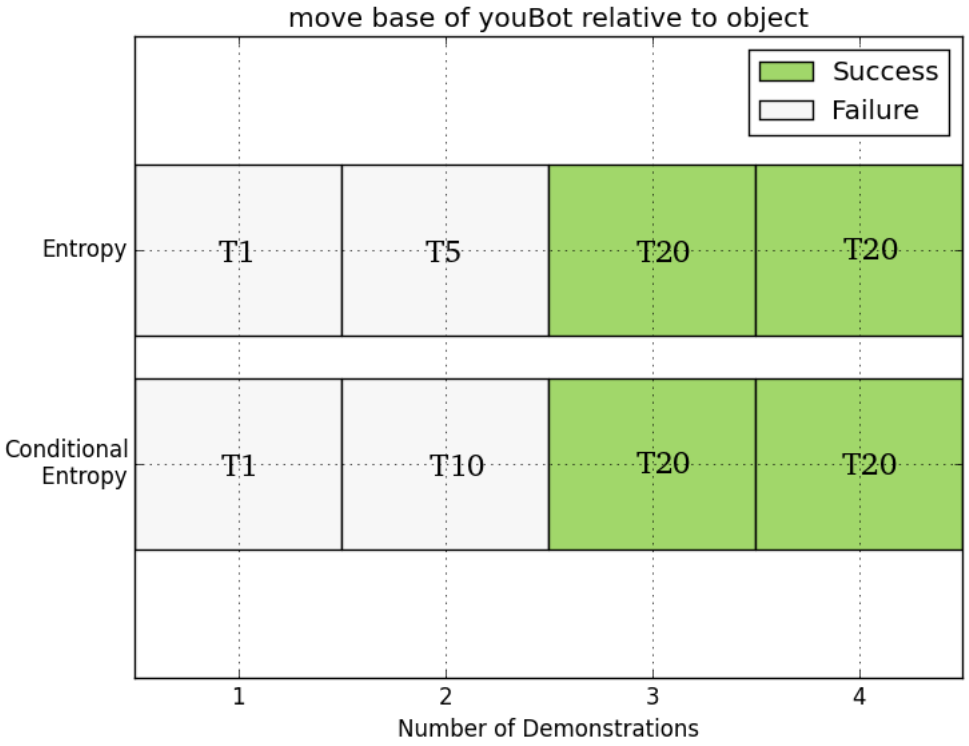


Figure 18: Results for *move* base to a relative pose. The x-axis denotes the number of demonstrations used for learning. The y-axis denotes the method used for computing relevance.Each box is marked with the selected template name. Green color denotes the relavant template was successfully recommended

5.5.3 *move* arm absolute pose

In this experiment the youBot is taught to *move* arm to an absolute pose. The experiments consist of recording different start and end poses of the robot arm. First the robot arm is moved to any arbitary position and the start pose is recorded then the robot arm is moved to the intended final position which needs to be taught and the end pose is recorded. The above steps are repeated till all the observations are recorded as shown in figure 19

This particular motion primitive can be used as a part of the "basic manipulation test" task. In these task the robot has to be taught the "look platform" pose for looking at the platform. We ran the experiments using different number of demonstrations and using both entropy and conditional entropy for measuring relevance. The success of the experiments are shown in figure 20. The template describing the joint angles of the arm was recommended as the most relevant template from the demonstrations.

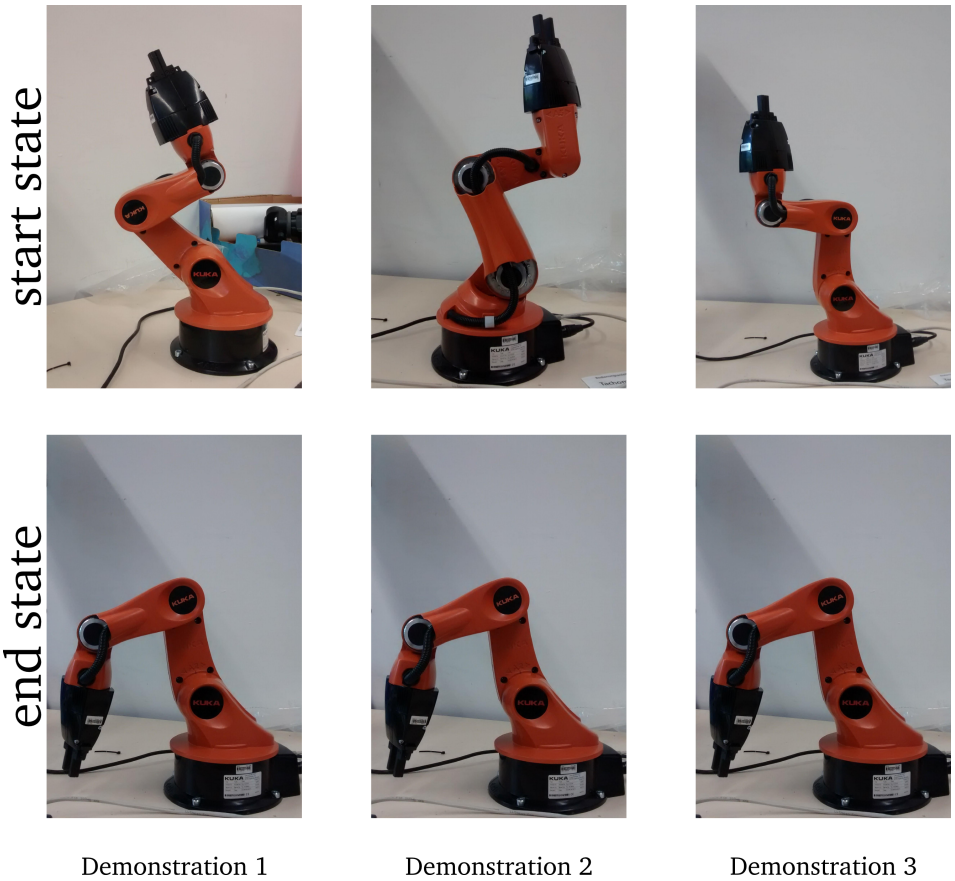


Figure 19: Demonstrations for teaching the robot to *move* arm to an absolute pose in the workspace. The rows show the recorded start pose and the end pose. The columns show the different demonstrations

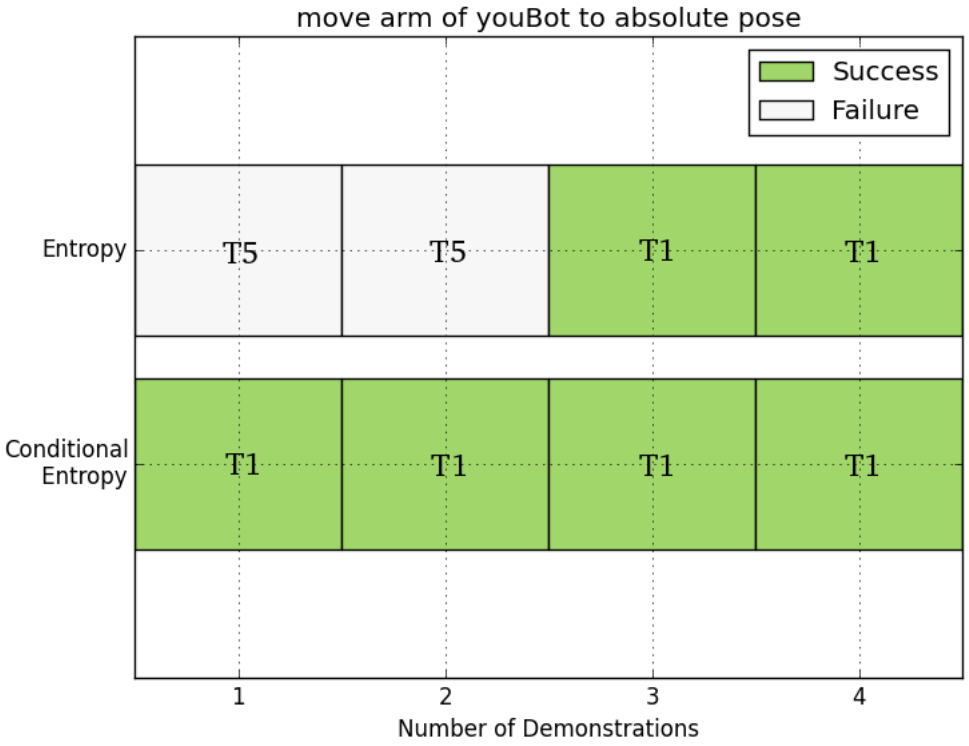


Figure 20: Results of *move* arm to an absolute pose. The x-axis denotes the number of demonstrations used for learning. The y-axis denotes the method used for computing relevance.Each box is marked with the selected template name. Green color denotes the relavant template was successfully recommended

5.5.4 *move* arm relative pose to object

In this experiment the youBot is taught to *move* arm to a relative pose to an external object. The experiments consist of recording different start and end poses of the robot arm. First the robot arm is moved to any arbitary position and the start pose is recorded then the robot arm is moved to the intended final position which needs to be taught and the end pose is recorded. The above steps are repeated till all the observations are recorded.

This particular motion primitive can be used as a part of the "precision placement test" task. In these task the robot has to be place the object precisely inside a cavity. So the robot has to be taught to place the object relative to placement cavity. We ran the experiments using different number of demonstrations and using both entropy and conditional entropy for measuring relevance. The success of the experiments are shown in figure 21. The template describing the relative distance between the tooltip and the object was recommended as the most relevant template from the demonstrations.

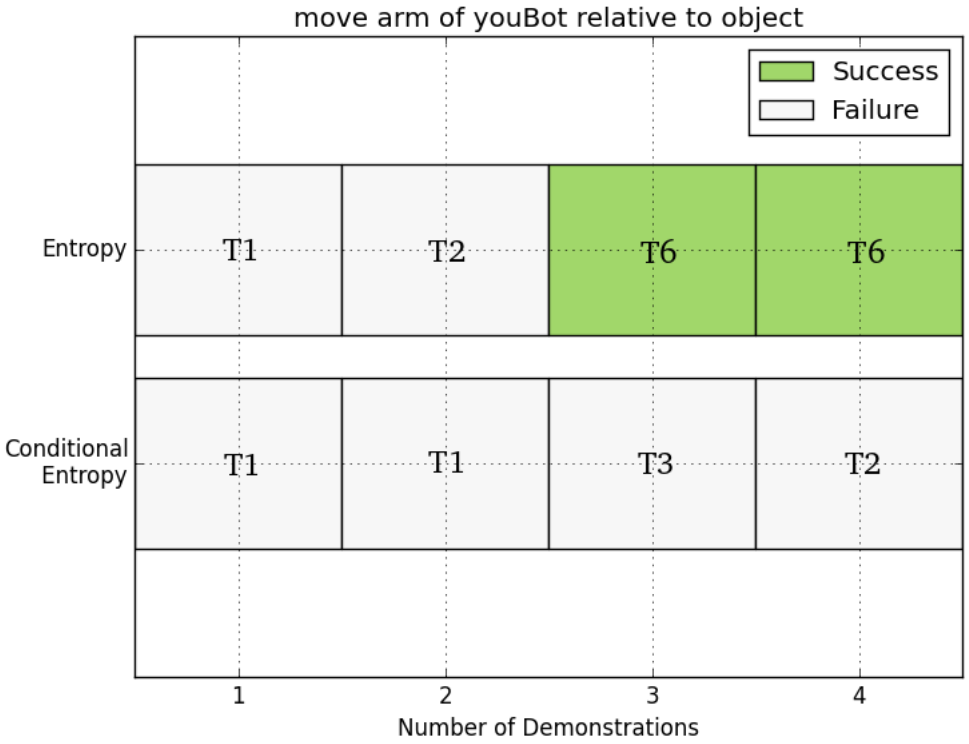


Figure 21: Results of *move* arm to an relative pose. The x-axis denotes the number of demonstrations used for learning. The y-axis denotes the method used for computing relevance.Each box is marked with the selected template name. Green color denotes the relavant template was successfully recommended

5.5.5 normalized Discounted Cumulative Gain based evaluation

The following section has the evalutaion based on the nDCG@1 and nDCG@2 metrics. For evalutation the templates in the knowledge base was labeled on a 3 point likert scale. We ran the algorithm using entropy as out relevance function.

Figure 22 shows the nDCG@1 plots for different number of demonstrations. Its clear that with 3 demonstrations the algorithm is able to predict the expected template in all the experiments.

Figure 23 shows the nDCG@3 plots for different number of demonstrations. We can observe that with increasing number of demonstrations the quality of the recommendations are improving. With 3 demonstrations the quality of the recommendations is well above 60% for all the experiments.

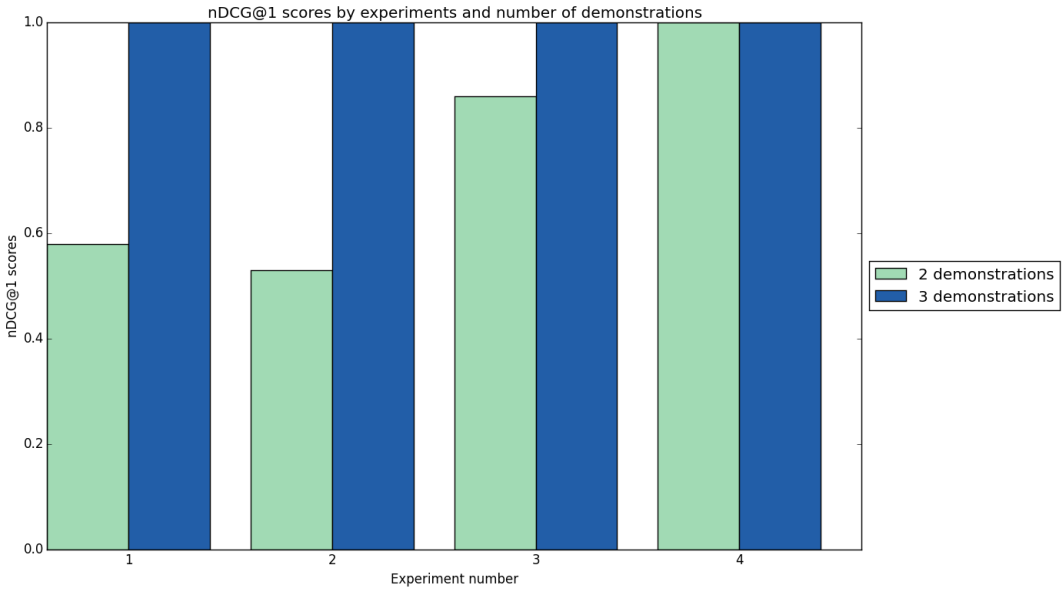


Figure 22: nDCG@1 values for the experiments conducted. The y-axis is the nDCG@1 score. The x-axis consist of groups of 2 bars arranged as experiments conducted. Each bar in the experiment corresponds to the number of demonstration used to make the recommendations.

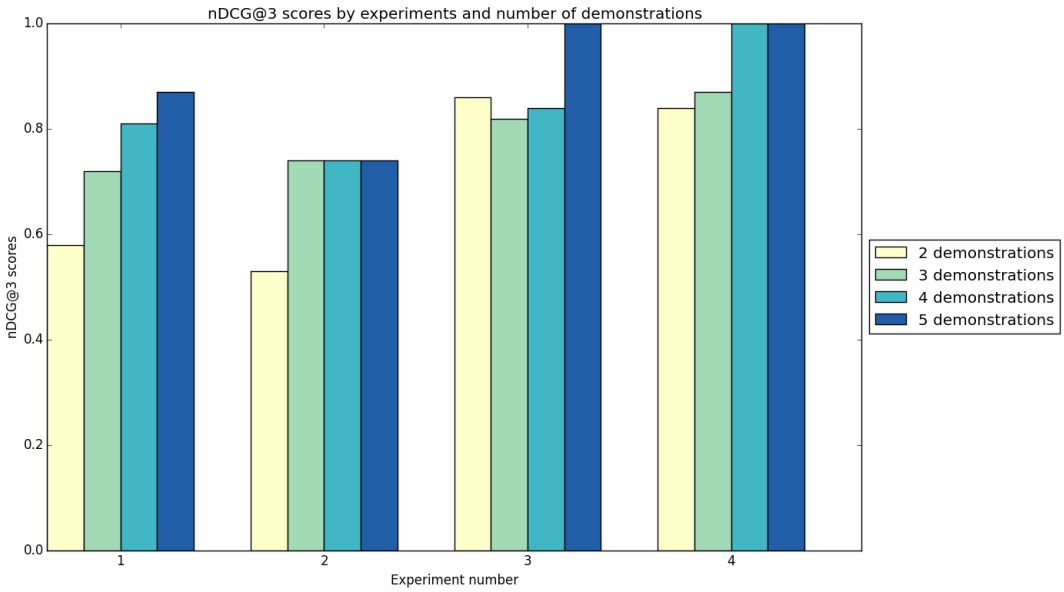


Figure 23: nDCG@3 values for the experiments conducted. The y-axis is the nDCG@3 score. The x-axis consist of groups of 4 bars arranged in the sequntial order of the experiments conducted. Each bar in the experiment corresponds to the number of demonstration used to make the recommendations.

5.6 Discussions

The proposed approach recommends the relevant features with higher probability. It has been observed that using entropy based likehood measuring method we could achieve a success rate of 100% using just 3 demonstrations. The higher success rate ensure that the entropy based method is a reliable way to determine the relevant features. While using the conditional entropy based relevance measuring method we could only achieve a success rate of 62.8%. The lower success rate means that conditional entropy cannot be used for recommending the relevant features.

6 Summary

In this work we presented relevant feature selection based on a knowledge base recommendation. The proposed approach was an extension to the framework proposed by [Abdo et al., 2014a]. We also have proposed a possible framework for representing a motion primitive in a skill based framework by [Pedersen et al., 2015] . The method was evaluated on experiments conducted on simulation and the real robot

- **Successful learning on fewer demonstrations**
  The result showed that we could successfully learn on fewer demonstrations. All the experiments conducted the relevant features could be recommended using just **three** demonstrations.

- **Structured nature of the knowledge base helps in recommending**
  The knowledge base was re structured to segregate templates based on the effect metrics. This ensured that the correct templates are made available for recommendations. This structure of the knowledge base acted as a catalyst for recommending in less number of demonstrations.

- **Generalization to un-demonstrated start states**
  Our results show that robot trained using our recommendations not only learns the goal on demonstrated motion primitives, but also generalizes well to do motion primitives from initial configurations not demonstrated before.

- **Use of proposed method to non-experts**
  Using the knowledge base generator B we could easily add new features and templates. This helps in ease for non-experts to teach the robot new motion primitives.

- **Scalability of the approach**
  The approach was specifically made to work under the skill based framework A , the approach is highly scalable. New motion primitives can be easily added and the existing motion primitives can be resued to form new skills. Specially in the case of industrial robotics, the author claims that it can be utilized for solving majority of the tasks.

Our application scenarios of learning goal from demonstrations goes beyond the application scenarios that previous work has considered.

7 Limitations of work

This section lists some of the major limitations of the approach.

- *Poor quality of demonstration*
  As LfD systems are totally dependent on the demonstration of the data, its quality greatly reflects on the results. Poor quality of demonstration will lead to very poor results. For example in our approach its very much necessary that there has to be significant ddifference in the features which are not to be learned. So the demonstrations has to vary for all the features not being learned. So great care has to be taken by the user to teach a correct set of demonstration.

- *Lack of element for feedback*
  The proposed approach there is not provisions for providing feedback to the robot after the learning. This is major limitation in terms of learning. Feedback are a good way for the robot to generalize on new situations.

- *Lack of continuous learning*
  The proposed approach the learning is done in a single shot. There is no provision for feedback and continuos learning. A scalable approach would be if there are provisions for the robot to continuos learn while executing. Such provisions will ensure that the robot is continuously learning and improving its decision making.

8 Future Work

This section proposes areas that can be improved or extended for future works in the field

- *Learning "When to imitate"*
  The current work only focussed on the "what to imitate" part for identifying the relevant aspects of the demonstrations. "When to imitate" is a section of LfD to determine when the demonstrated action should be executed. The proposed approach can be used to identifying the start conditions (pre-condition) by identifying the relevant aspects in the start values of the demonstrations.

- *Better evaluation strategies*
  The current evaluation method is manual and are subject ot faults. Automated methods for evaluting the results need to be used for better confidence on the results.

- *Combining both entropy and conditional entropy*
  The current results showed that entorpy based relevance performed better while conditional entorpy didnt perform better for all the cases. But conditional entropy performed better in some cases. This needs to be further investigated why the conditional entropy didnt perform better in some experiments and how we can modify the approach to use it for getting better results.

- *Creating a complete skill based on the proposed motion primitive framework*
  A motion primitive framework is proposed in this work. The relevant features were also learnt. Now the work has to be extended to use the relevant feature and execute a particular motion primitive successfully. Also a set of motion primitives has to be learnt and a complete skill has to be executed.

- *Better Human Machine Interaction (HMI)*
  The current methods of recording the demonstrations are not sutiable for outside lab. Since one of the big benefits of LfD is that non-experts would be able to teach the robot with new skills, its of high importance to work on better more intuitive HMI needs to be developed.

- *Learning of the templates on-line*
  The templates used in the approach is generated manually using experts of manipulation. But in future these futures has to be learnt on-line. The base templates can be used from the off-line process but these needs to be updated online based on the demonstrations. For example for a motion primitive representing the distance between tool-tip and object template will be recommended, also if joint 4 is showing relevance as in all the demonstrations joint 4 was show in a particular angle, then it should be added on-line to the template list.

- *Completenes of the approach*
  The results of the approach shows positive signs of the usefullness of the approach. But we need to have a critical look at the completeness of the solution i.e. can the approach be generalized for learning all the motion primitives required for a general robot. Works like [Bogh et al., 2012] have claimed that for all the task in an industrial setup, a limited list of skills is sufficient. The completeness of our approach needs to be verified.

A Use of Motion Primitive in Skill Based Framework

Similar to the concept that speech consist of syllables , it has been observed that tasks in can be broken down into smaller elements which are called as skills. The skills in turn can be decomposed of smaller atomic movements called as Motion Primitives. A three layer Framework was introduced by [Pedersen et al., 2015] and is illustrated in figure 24.

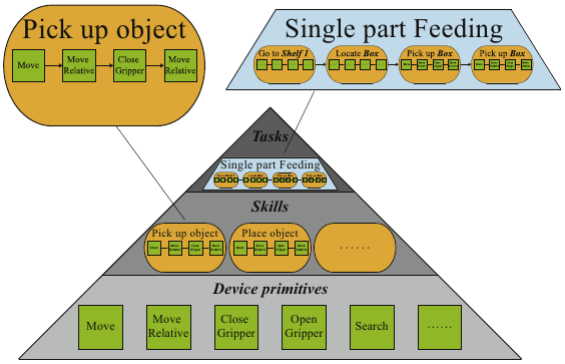


Figure 24: Three layers of primitives, skills and tasks. The components in each layer is essentially a combination of lower layer. [Pedersen et al., 2015]

The three layers of the framework are :

- Motion Primitives
- Robotics Skills
- High Level Tasks

A.1 Motion Primitives

The lowest layer is the called the motion primitives. It consist of atomic movement.

A.2 Robotics Skills

Robotic skills form the base of the skill based Framework. Robotic skills are object-centred robot abilities, which can be easily parametrized.

A.3 High level Tasks

The Higher level description of a task to be done. Ususally in this layer planners like Stanford Research Institute Problem Solver (STRIPS) or Planning Domain Definition Language (PDDL) are used to choose which robotic skills has to be executed.

The proposed motion primitive approach fits exactly inside the skill based framework mentioned here. The motion primitives can be combined together and higher level skills can be executed.

B Knowledge base generator

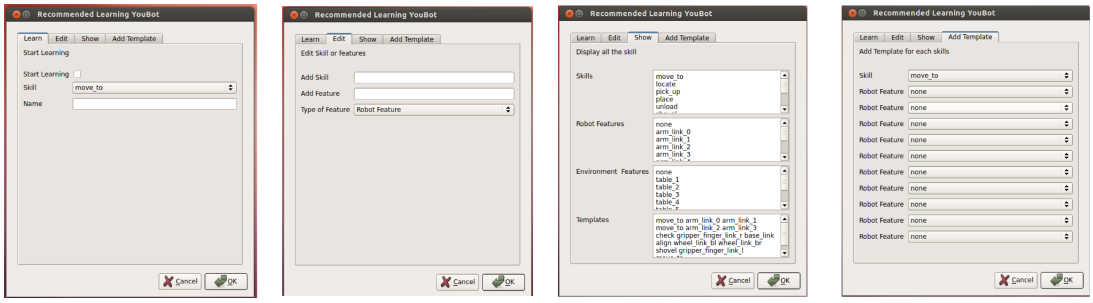


Figure 25: Tool for generating knowledge base. 1) window for learning a new skill 2) window for editing skill or features 3) Display all features, skills and templates 4) adding new templates

On of the limitations of the current efforts in LfD is adding new features for new task [Argall et al., 2009]. Due to the intuitive nature of demonstration, LfD algorithms have the potential to make robot programming accessible for everyday users who have no programming experience and want to customize robot behaviours. The use of current approaches by the researchers , however, is likely to lead to situations in which users attempt to teach the robot tasks that cannot be learned simply due to the lack of features describing some aspect of the task.

The solution to such an attempt would be have good tool where the user on identification of new features can add new features to the system on which the robot can learn.

Also the template creation is an actively changing scenario with various permutations and combinations possible. The above mentioned tool can be used for rapid generation and modification of the templates.

C Use of Recommender system in robotics

We started our work based on the notion of using Recommender system in the field of robotics. We could partially use the notion of recommender system in our work. In the time period of the research various other fields where the recommender system could be used were identified. This section list some of the fields of robotics in which technology of recommender system fits appropriately.

C.1 Preferences of users in Task Planing

With the rise of service robotics in home environments, personalized task planning is emerging as a new field of interest. The robots has to learn the user preferences based on its experience with the user and able to modify its decisions regularly. A common example would be suppose if a user asks the robot for a cup of tea, the robot has to make a decision on which cup would be an appropriate choice for the user. This choice of the user can be learnt using recommender systems. Works by [Abdo et al., 2013] and [Abdo et al., 2015] have already made progress in this field by using crowd sourcing data.

C.2 Kernel Recommendation in learning algorithms

"Firstly, linearity is rather special, and outside quantum mechanics no model of a real system is truly linear. Secondly, detecting linear relations has been the focus of much research in statistics and machine learning for decades and the resulting algorithms are well understood, well developed and efficient. Naturally, one wants the best of both worlds. So, if a problem is non-linear, instead of trying to fit a non-linear model, one can map the problem from the input space to a new (higher-dimensional) space (called the feature space) by doing a non-linear transformation using suitably chosen basis functions and then use a linear model in the feature space. This is known as the 'kernel trick'. The linear model in the feature space corresponds to a non-linear model in the input space. This approach can be used in both classification and regression problems. The choice of kernel function is crucial for the success of all kernel algorithms because the kernel constitutes prior knowledge that is available about a task. Accordingly, there is no free lunch (see No Free Lunch Theorems) in kernel choice." [7]

Collaborative based recommender system can be used to determine the kernel for learning algorithms. This problem is similar to the n-bandit problem where the user has to decide on which kernel to keep the bet.

[Matikainen et al., 2013] tried to solve the n-bandit problem to determine which state machine gives better coverage of the floor. Similar attempts can be made in learning the

[7] `http://www.svms.org/kernels/`

kernel .

C.3 Motion Path Planning algorithms

Open Motion Planning Library (OMPL) has implemented the following planners `http://ompl.kavrakilab.org/planners.html` The selection of the planner for a particular tasks is active scientific problem. If you use the OMPL control , then OMPL will automatically select an appropriate planner (unless you have explicitly specified one). If the state space has a default projection (which is going to be the case if you use any of the built-in state spaces), then it will use Kinodynamic Planning by Interior-Exterior Cell Exploration (KPIECE). This planner has been shown to work well consistently across many real-world motion planning problems, which is why it is the default choice. In case the state space has no default projection, Rapidly-Exploring Random Tree (RRT) will be used. Can we recommend which planner to use in which situation based on previous knowledge of the planner in similar situations. Content based recommender system can be used in such situations to determine which planner will be suitable in current situation.

C.4 Learning trajectory preference of users

[Jain et al., 2013] have worked on learning the trajectory preferences of users by on-line learning their feedbacks. Similar approach can be used as a recommender system to learn the user preference of a trajectory for doing a particular tasks.

D Softwares and tools

A software implementation and number of tools have been included in the attached *CD-ROM*. The tools and the software developed as part of the project. Below is a brief description of the packages

- **Recommender Learning**
  Python based implementation of the learning approach. The class has to be invoked with the folder containing the JSON file of the demonstrations. The JSON file is read and learning is made on base of the demonstrations. It also reads the knowledge base and creates the set of templates. It implements the approach mentioned in section 4. Available online : `https://github.com/deebuls/RecommenderSystemInRobotics/tree/master/code/recommended_learning`

- **mir_teleop_record**
  ROS node for teleoperation of the youbot arm and the base. The node was an extension of the work of b-it bots team `https://github.com/mas-group/robocup-at-work/tree/brazil-2014/mas_common_robotics/mcr_tools/mcr_teleop`. The node was modified to record the features and to create a JSON file with the readings. Available online : `https://github.com/deebuls/RecommenderSystemInRobotics/tree/master/code/mir_teleop_record`

- **Data accumulator**
  The python script for accumulating all the recorded json file and creating a consolidated JSON file of the demonstration. Available online : `https://github.com/deebuls/RecommenderSystemInRobotics/tree/master/code/tools`

- **Knowledge base creator**
  An interactive online tool for creating of the knowledge base. Menu driven GUI for active interaction with the user and easy to use template creation for the knowledge base. Available online : `https://github.com/deebuls/RecommenderSystemInRobotics/tree/master/code/knowledge_base_creator`

References

[Abdo et al., 2013] Abdo, N., Kretzschmar, H., Spinello, L., and Stachniss, C. (2013). Learning manipulation actions from a few demonstrations. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 1268–1275. IEEE.

[Abdo et al., 2014a] Abdo, N., Spinello, L., Burgard, W., and Stachniss, C. (2014a). Inferring What to Imitate in Manipulation Actions by Using a Recommender System. In *ICRA*, volume 31, pages 189–206. Springer$}$.

[Abdo et al., 2014b] Abdo, N., Spinello, L., Stachniss, C., and Burgard, W. (2014b). Collaborative Filtering for Learning User Preferences for Robotic Tasks.

[Abdo et al., 2015] Abdo, N., Stachniss, C., Spinello, L., and Burgard, W. (2015). Robot, Organize my Shelves! Tidying up Objects by Predicting User Preferences.

[Aein et al., 2013] Aein, M. J., Aksoy, E. E., Tamosiunaite, M., Papon, J., Ude, A., and Worgotter, F. (2013). Toward a library of manipulation actions based on semantic object-action relations. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 4555–4562. IEEE.

[Alissandrakis et al., 2006] Alissandrakis, A., Nehaniv, C. L., and Dautenhahn, K. (2006). Action, state and effect metrics for robot imitation. In *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*, pages 232–237. IEEE.

[Alissandrakis et al., 2005] Alissandrakis, A., Nehaniv, C. L., Dautenhahn, K., and Saunders, J. (2005). An approach for programming robots by demonstration: Generalization across different initial configurations of manipulated objects. In *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on*, pages 61–66. IEEE.

[Andersen et al., 2014] Andersen, R. S., Nalpantidis, L., KrÃ$\frac{1}{4}$ger, V., Madsen, O., and Moeslund, T. B. (2014). Using robot skills for flexible reprogramming of pick operations in industrial scenarios. In *International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pages 678–685.

[Argall et al., 2009] Argall, B. D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern recognition and machine learning*. Information science and statistics. Springer, New York.

[Bobadilla et al., 2013] Bobadilla, J., Ortega, F., Hernando, A., and GutiÃ©rrez, A. (2013). Recommender systems survey. *Knowledge-Based Systems*, 46:109–132. 92.

[Bogh et al., 2012] Bogh, S., Nielsen, O. S., Pedersen, M. R., KrÃ¼ger, V., and Madsen, O. (2012). Does your robot have skills? In *The 43rd Intl. Symp. on Robotics (ISR2012)*.

[Buxton, 2003] Buxton, H. (2003). Learning and understanding dynamic scene activity: a review. *Image and vision computing*, 21(1):125–136.

[Calinon, 2009] Calinon, S. (2009). *Robot programming by demonstration : a probabilistic approach*. EPFL Press Distributed by CRC Press, Lausanne, Switzerland Boca Raton, FL.

[Csibra and Gergely, 2007] Csibra, G. and Gergely, G. (2007). âobsessed with goalsâ: Functions and mechanisms of teleological interpretation of actions in humans. *Acta psychologica*, 124(1):60–78.

[Demiris, 2007] Demiris, Y. (2007). Prediction of intent in robotics and multi-agent systems. *Cognitive Processing*, 8(3):151–158.

[Friedrich and Dillmann, 1995] Friedrich, H. and Dillmann, R. (1995). Robot programming based on a single demonstration and user intentions. In *3rd European workshop on learning robots at ECML*, volume 95.

[Friedrich et al., 1996] Friedrich, H., MÃ¼nch, S., Dillmann, R., Bocionek, S., and Sassin, M. (1996). Robot programming by demonstration (RPD): Supporting the induction by human interaction. *Machine Learning*, 23(2-3):163–189.

[Hommel, 2009] Hommel, B. (2009). Action control according to TEC (theory of event coding). *Psychological Research Psychologische Forschung*, 73(4):512–526.

[Isham, 1981] Isham, V. (1981). An introduction to spatial point processes and markov random fields. *International Statistical Review/Revue Internationale de Statistique*, pages 21–43.

[Jain et al., 2013] Jain, A., Wojcik, B., Joachims, T., and Saxena, A. (2013). Learning trajectory preferences for manipulators via iterative improvement. In *Advances in Neural Information Processing Systems*, pages 575–583.

[Jain et al., 1998] Jain, A. K., Zhong, Y., and Dubuisson-Jolly, M.-P. (1998). Deformable template models: A review. *Signal processing*, 71(2):109–129.

[Kruger et al., 2010] Kruger, V., Herzog, D., Baby, S., Ude, A., and Kragic, D. (2010). Learning Actions from Observations. *IEEE Robotics & Automation Magazine*, 17(2):30–43.

[Lee et al., 2009] Lee, K. H., Lee, J., Thomaz, A. L., and Bobick, A. F. (2009). Effective robot task learning by focusing on task-relevant objects. In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 2551–2556. IEEE.

[Manning, 2008] Manning, C. (2008). *Introduction to information retrieval*, chapter Evaluation in information retrieval. Cambridge University Press, New York.

[Manschitz et al., 2015] Manschitz, S., Kober, J., Gienger, M., and Peters, J. (2015). Learning movement primitive attractor goals and sequential skills from kinesthetic demonstrations. *Robotics and Autonomous Systems*.

[Matikainen, 2012] Matikainen, P. (2012). *Model Recommendation for Action Recognition and Other Applications*. PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213.

[Matikainen et al., 2013] Matikainen, P., Furlong, P. M., Sukthankar, R., and Hebert, M. (2013). Multi-armed recommendation bandits for selecting state machine policies for robotic systems. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 4545–4551. IEEE.

[Muench et al., 1994] Muench, S., Kreuziger, J., Kaiser, M., and Dillman, R. (1994). Robot programming by demonstration (rpd)-using machine learning and user interaction methods for the development of easy and comfortable robot programming systems. In *Proceedings of the International Symposium on Industrial Robots*, volume 25, pages 685–685. INTERNATIONAL FEDERATION OF ROBOTICS, & ROBOTIC INDUSTRIES.

[Nicolescu and Mataric, 2003] Nicolescu, M. N. and Mataric, M. J. (2003). Natural methods for robot task learning: Instructive demonstrations, generalization and practice. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems*, pages 241–248. ACM.

[Paraschos et al., 2013] Paraschos, A., Daniel, C., Peters, J., and Neumann, G. (2013). Probabilistic movement primitives. In *Advances in Neural Information Processing Systems*, pages 2616–2624.

[Pardowitz et al., 2007] Pardowitz, M., Knoop, S., Dillmann, R., and Zollner, R. D. (2007). Incremental Learning of Tasks From User Demonstrations, Past Experiences,

and Vocal Comments. *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 37(2):322–332.

[Pedersen et al., 2015] Pedersen, M. R., Nalpantidis, L., Andersen, R. S., Schou, C., BÃ¸gh, S., KrÃ¼ger, V., and Madsen, O. (2015). Robot skills for manufacturing: From concept to industrial deployment. *Robotics and Computer-Integrated Manufacturing*.

[Schaal et al., 2000] Schaal, S., Kotosaka, S., and Sternad, D. (2000). Nonlinear dynamical systems as movement primitives. In *IEEE International Conference on Humanoid Robotics*, pages 1–11.

[Tomasello et al., 2005] Tomasello, M., Carpenter, M., Call, J., Behne, T., and Moll, H. (2005). Understanding and sharing intentions: The origins of cultural cognition. *Behavioral and Brain Sciences*, 28:675–691.

[Ude, 1993] Ude, A. (1993). Trajectory generation from noisy positions of object features for teaching robot paths. *Robotics and Autonomous Systems*, 11(2):113–127.

[Veeraraghavan and Veloso, 2008] Veeraraghavan, H. and Veloso, M. (2008). Teaching sequential tasks with repetition through demonstration. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3*, pages 1357–1360. International Foundation for Autonomous Agents and Multiagent Systems.