Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Master's Thesis Nr. 13

From Speech Recognition to Instruction Learning

Jonah Vincke

Course of Study:	Informatik
Examiner:	Prof. Dr. rer. nat. Marc Toussaint
Supervisor:	Prof. Dr. rer. nat. Marc Toussaint
Commenced:	October 27, 2014
Completed:	April 28, 2015
CR-Classification:	I.2.6, I.2.7, I.2.8

Kurzfassung

In der Robotik ist *learning from demonstration* ein Forschungsfeld, welches viel Beachtung bekommen hat, da es ermöglicht, einen Roboter auf einfache und intuitive Weise komplexe Bewegungen beizubringen. Sofern man einem Menschen etwas beibringt, ist es üblich, natürliche Sprache zu benutzen um zusätzliche Informationen zu vermitteln. Ungeachtet dessen ist es kaum erforscht worden, wie Sprache zusätzlich genutzt werden kann, um Robotern etwas mittels *learning from demonstration* beizubringen. Daher untersucht diese Thesis, wie natürliche Sprache in der Robotik bereits genutzt wird und wie der Lernprozess mittels Sprache besser unterstützt werden kann. Im Weiteren wird ein *learning from demonstration* System auf Basis von *task spaces* implementiert, welche eine intuitive Abstraktion des Zustands des Roboters und dessen Relation zur Umgebung sind und daher stärker mit gesprochenen Anweisungen verknüpft sind. Zusätzlich wird eine *task space-Auswahl-Instruktion* und eine *Verbesserungs-Instruktion* vorgestellt, um natürliche Sprache zur Unterstützung des Lernprozesses zu nutzen. Diese werden auf einem echten Roboter mittels eines Experiments evaluiert, bei dem der Roboter einen Punkt zeichnen muss.

Abstract

In robotics, *learning from demonstration* is a research field that has gained a lot of attention since it provides an easy and intuitive way to teach a robot complex movements. When teaching a human it is common to use natural language to provide additional information. Regardless of that, it is rarely investigated how speech can be used in addition for teaching a robot with *learning from demonstration*. Therefore this thesis examines how natural language is already used in robotics and how the learning process can be supported further by using speech. Furthermore a *learning from demonstration* system is implemented based on *task spaces* that are an intuitive abstraction of the state of the robot and its relation to the environment and therefore are more correlated with spoken instructions. Additional a *task space selection instruction* and a *correction instruction* are introduced to use natural language to support the learning process. They are evaluated on a real robot in an experiment where the robot has to draw a point.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

In robotics, learning is one of the most important but also challenging field of research. Approaches vary strongly depending on the field of application from plan generation for navigation tasks to trajectory representations for low-level movements. They are usually separated in unsupervised learning if the robot learns autonomously and supervised learning if an expert provides additional information. For the latter, *Learning from Demonstration* (LfD) is a research field that has received a lot of attention because it provides an intuitive and easy way to implement complex motions. In contrast to this motivation, one of the most intuitive way to teach a human has not been in the focus: natural language.

Therefore, this thesis investigates how natural language instructions have to be designed to be intuitive and how they can support the learning process of LfD methods.

In recent works the concept of describing skills in task spaces, instead of only using the joint configuration of the robot, has been discovered. They abstract relationships between the information of the environment and the robot's joint configuration in an intuitive way. Accordingly natural language is more correlated to such a description than to the robot's observation or joint configuration. Therefore this work focus on the representation of skills in task spaces. Most approaches only define task spaces that do not contradict and that are needed for the specific task a priori. This is a strong restriction since it must not even be clear for the teacher which task spaces are relevant or contradict the goal. It is also impractical to select them a priori and not during learning. Therefore the work of Muhlig et al. [MGSG09] investigated an implicit variance-based selection together with an explicit on-line selection using vision that is limited to finding a single object of attention.

This thesis investigates a more general explicit selection of task spaces using natural language. Furthermore a correction instruction is introduced that addresses the opportunity of natural language to support the learning process together with the task space selection instruction. In addition, it is shown how natural language can be used to control the learning process intuitively.

Outline

Chapter 2 – Related work: First of all a short overview of the different classes of LfD is given according to their abstraction level. Here the mentioned approach of Muhlig et al. [MGSG09] related to the selection of task spaces is investigated in more detail. Then an overview is given how natural language is used in the robotics community that is not limited to LfD methods.

Chapter 3 – Background: This chapter briefly describes the methods used in this thesis. The *Dynamic Time Warping* (DTW) algorithm that can be used to align signals that vary on time and a linear regression method that finds the optimal parameters of a linear model are presented. After that a recent LfD method is investigated that is later used as the skill representation. The last part gives a brief discussion about motion and kinematics that leads to the formulation of the operational space controller which is used in this approach to reproduce the skill. It also introduces a new parameter for the skill representation.

Chapter 4 – Instruction learning: The three main concepts of the use of natural language as an additional input source are presented. It is investigated how speech can be used to design the learning process more interactive. After that an automatic task space selection similar to the work of Muhlig et al. [MGSG09] is introduced that can be manipulated by a novel spoken task space selection instruction. In addition a correction instruction is investigated that can be used to improve the selection if the skill is not reproduced correctly by the robot.

Chapter 5 – Implementation: First the robot platform and the already existing controller are presented. Then it is described how the chosen speech recognition software is adapted and how the interactive learning process is realized. After that it is described how the LfD method is implemented.

Chapter 6 – Evaluation: A simple point drawing experiment is described and the results for the solely automatic task space selection and the combination with whether the correction instruction or the spoken task space selection is presented.

Chapter 7 – Discussion: First the results of the experiment are analyzed. After that the introduced methods of this thesis are compared with the related works. Furthermore the insights that has been gained in this work about the use of speech recognition software and learning from demonstration are reflected. After that a short conclusion from this work is drawn.

Chapter 8 – Summary: At last a brief summary over the whole thesis is given that highlights the important parts.

2 Related work

The research field of machine learning can be divided into supervised and unsupervised learning. In the unsupervised case, no expert exist who provides information or controls the learning process. Therefore the agent has to decide by itself what to learn and can only use its own observations. For robotics an example would be a robot that has a prediction about its environment. While moving and interacting within that environment the robot would learn by updating its prediction according to the chosen actions and observations. In this case, natural language would be a form of interaction.

In supervised learning, the agent gets additional information by an expert. Learning from demonstration, which is also called imitation learning or programming by demonstration, is a field of research that has gained a lot of attention in the past in the robotics community. Here the additional information is usually only the teacher's movement.

In the following, a brief overview of the different types of LfD methods is presented. Here the use of task spaces is investigated in more detail since their abstraction is higher correlated to spoken description than the sensor information is. In addition an approach that is used to select task spaces from an arbitrary large pool is of most interest since it is related to the introduced instructions in this thesis. The last part gives an overview of the use of natural language in the robotics community that is not limited to learning from demonstration.

2.1 Learning from demonstration

Learning from demonstration is a form of supervised learning where the expert provides trajectories, which represent the recorded movement over time, as data. In most cases this is the set of joint angles $\{q\}_{0:T}$ and velocity $\{\dot{q}\}_{0:T}$. The trajectories are also implicitly labeled as "good" examples for a desired skill. The main goal is to use the provided trajectories to learn a representation of a skill, which is often called the policy, that can be used later with possibly changed conditions. A common motivation is to provide an intuitive or simple way for programming a robot[PHAS09, KKGB11], sometimes such that also non experts can create new skills [CIT12].

Several approaches for learning from demonstration exist. They can be classified on how the policy is represented and how it is learned. Additional, Argall et al. [ACVB09] classify how the data is acquired.

They separate between *demonstration*, where the robot's own movement is recorded, and *imitation*, where the teacher's movement is recorded. The most significant difference between both is, that for imitation the correspondence problem between the teacher's degrees of freedom and the robot's ones has to be considered. In addition, the joint limits are not taken into account directly. Despite that the motion is human-like and the whole body movement can be recorded. For the data acquisition several systems exist that use magnetic, inertial or mechanical sensors. Alternatively the motion can be derived using a vision system.
For demonstration, no correspondence problem has to be considered and no additional acquisition system is needed. Here the only difference can be seen in the way the robot is moved during the data acquisition. One possibility is kinesthetic teaching where the robot is moved physically. It is also possible to teleoperate the robot by using the same data acquisition systems as for imitation but using the information to move the robot during the demonstration. Due to the fact, that in both cases the data has to be stored in or transformed into a space that is relevant for the robot, the skill representation is independent of the acquisition method.
According to Argall et al. [ACVB09] the policy representation can be separated into three classes:

- Plans: A set of pre- and postconditions for each action and possibly a transition model are learned.
- System models: A transition model of the world and possibly a reward function for the states are learned.
- Mapping functions: States are directly mapped to actions.

Since a survey about all LfD methods would go beyond the scope of this work only several approaches are presented in the following.

2.1.1 Plans

In the framework of plans the policy is described as a hierarchy of actions. Each action consists of pre- and postconditions. An example is the work of Lauria et al. [LBKK02] where the teacher provides a spoken route description. It is then translated into a sequence of predefined actions. The pre- and postconditions are then used to check if the spoken description is suitable.
Rybski et al. [RYSV07] represent a plan as a directed acyclic graph that connects several *behaviors*. Each behavior consists of a list of preconditions, a name, a parameter list and a list of behaviors that will be triggered after the execution, dependent on its result. The precondition has to be true before the behavior is executed and the parameters are needed for the execution.

2.1.2 System model

For system models, the most common field of research is *Reinforcement Learning* (RL) [SB98]. Here the goal is to find a policy π, that maps the actual state s to an action a. This is not done directly as before, despite a transition model $P(s'|s,a)$ and a reward function $R(s)$ is used. The agent chooses the sequence of actions that maximizes the expected reward, optionally only over a specific period of time. There exist many different RL methods that are distinct from each other by their policy learning method, the space of actions and states and additional assumptions. As a common learning method the Bellman equation is used:

$$(2.1)\quad V^{\pi}(s) = \sum_{a} \pi(s,a) \sum_{s}^{\prime} P(s'|s,a)R(s') + \gamma V^{\pi}(s')$$

In equation 2.1 γ denotes the discount factor that represents how the temporal distance to future rewards is weighted. An example for the representation of the states and actions is a *Markov Decision Process* (MDP) [KKGB11, AN04]. It has a finite set of states S and actions A and it assumes that future observations only depend on the actual state and the actions taken in the future (the Markov property). The policy can be found by using equation 2.1 as a dynamic programming approach.

The goal of reinforcement learning is to find a policy that maximizes the expected reward. The basic approach is an unsupervised learning method that assumes a given transition model and reward function. In the field of learning from demonstration only demonstrations of an expert are given. In some cases also the transition model is given but the reward function has to be determined to model the demonstrated behavior. This field of research is known as *Inverse Reinforcement Learning* (IRL). An example is *Apprenticeship Learning* introduced by Abdeel and Ng [AN04]. Here the problem is described as a MDP but with an unknown reward function. Given an expert's set of trajectories it is constructed iteratively. This is done by simulating which observations would be made by following the current policy. According to this the predicted reward function is updated such that the difference of the total reward between the trajectories of the expert and the simulated one is minimized. The new reward is then used by a standard RL algorithm to update the policy.

2.1.3 Mapping function

Mapping functions are most often used for low-level motions. There are a lot of different approaches for learning a mapping from a given state to an action. The most important properties of this type of LfD methods are their ability to generalize and their robustness against perturbations [PHAS09].

Several approaches represent this mapping in a form of *Partial Derivative Equations* (PDEs). Here the robustness is ensured by its design in form of PDEs. As an example, *Dynamic Movement Primitives* (DMPs) [PHAS09] define a set of partial derivative equations that are similar to a linear spring system. Therefore only goal directed skills can be learned. In addition to a linear spring system non-linear movements can be modeled by a non-linear function that perturbs the PDEs. Here the generalization to new start conditions is given by the design and it can be adapted to new goals by changing the goal position.
Another approach uses *Gaussian Mixture Models* (GMMs) [KZB11] to model an underlying density function of an arbitrary dynamical system. Global stability is ensured by the introduced parameter estimation method of the GMMs. It is realized as an optimization problem with strict stability constraints.

Despite the general definition of Argall et al. for mapping functions there exist approaches that map the time to a desired state and not the state to an action. The desired state is then mapped to an action using a controller. An example are *Probabilistic Movement Primitives* (ProMPs) [PDPN13]. This statistical approach describes a skill as the time dependent mean trajectory and covariance encoded by a basis function model. The main advantage of ProMPs is that they can be easily adapted to via-points and to new starting- or end conditions.

Other approaches exist that use GMMs to define the density function of the trajectories and not of a dynamical system [CGB07, MGH$^+$09]. In cases of binary spaces a *Bernoulli Mixture Model* (BMM) can be used similar to the GMM [CGB07]. The parameters of the GMMs or BMMs are derived using an *Expectation-Maximization* (EM) algorithm. Optionally a k-means clustering method can be used to find a better initialization of the EM algorithm. To reproduce the skill often *Gaussian Mixture Regression* (GMR) is used.

In both works the use of task spaces, which may be referred to as features [CGB07], ensures the generalization. The robustness to perturbations is given by the design of the controller that is used to reproduce the skill. A task space is any abstraction of the state of the robot, the environment or a combination of both. In most cases it abstracts the relation between the robot's joint configuration and the world in an intuitive way such as the position of the hand or the distance between the robot's body and an object. Therefore a spoken description is usually more correlated with task spaces than with the joint configuration or sensor data. In most works the task spaces are predefined to cover all needed aspects of a task which contradicts the idea of generalization.

2.2 Task space selection

The Problem of choosing which task spaces are relevant during learning and not a priori is considered in the work of Muhlig et al. [MGSG09]. They define three different criteria for choosing relevant task spaces. As the basis pool, the absolute and relative positions and absolute orientation of all objects are used.

Before starting with the demonstration, the object of attention has to be identified. According to this the relevant object has to be shaken by the teacher which is then noticed by the system. This information is used to apply the *Attraction-Based Criterion*. It discards all object-based task spaces that are not related to the object of attention. Additionally a task space is added for each other object that covers its relative position to the object of attention.

After the data acquisition and before the skill learning process, the *Variance-Based Criterion* is used to avoid conflicting task spaces. Therefore the task spaces are sorted into groups of conflicting task spaces, for example all that are related to the position of an object. For each group the task space that has the lowest variance over all time steps is chosen. For this purpose, a *Similarity Criterion* is presented, that is also used in a similar work of Muhlig et al. [MGH$^+$09]. It maps the variance to a linear function with a negative gradient and is limited to a maximal value for a variance of zero and to zero for a value that exceeds a specific maximal variance. The mapped variance is then used as the metric of a least squared cost for the optimization of an operational task space controller.

In the first mentioned approach of Muhlig et al. [MGSG09] a third criterion is introduced. They use an imitation approach to record the demonstrated movement using a VICON motion capture system. These measurements are mapped on a simplified model of a human that is used to define two cost functions, one that represents the effort and a second that stands for the discomfort of a movement. These costs are used for the *Kinetic Criterion* that chooses only task spaces where the sum of both costs exceeds a fixed threshold.

2.3 Natural language and learning

In the robotics community the most common fields of application for natural language are navigation and vision. For the first, in most cases a sequence of actions are provided together with spatial informations by natural language [LBKK02, TKD$^+$11, DKS13]. In the vision community objects are described [HYNK04, MFZ$^+$12] or classified [CCT10, CT10] by speech.

When teaching a human similar to learning from demonstration it is common to use natural language, for teaching a robot it is not as most works do not use speech at all [AN04, CGB07, PHAS09, KZB11]. Apart from that, natural language is only used to guide the user through the learning process [WIC$^+$09] or in a strictly separated combination of learning from demonstration and learning from natural language [RYSV07]. Only Akgun et al. [ACYT12] introduce a method where natural language provides additional information for LfD. Here the teacher can generate a set of *keyframes* instead of a whole trajectory using natural language.

2.3.1 Navigation tasks

When navigating robots, the use of natural language has been discovered in several works. As an example Lauria et al. [LBKK02] introduce *Instruction-Based Learning*, a method that creates a new skill as a plan only by a spoken set of actions. The scenario is not limited to, but only tested for navigation. The users input is limited to predefined actions and already learned skills.

Duvallet et al. [DKS13] introduce a system to follow a natural language route description in a possibly unknown environment. Here the route description is first decomposed into a set of *Spatial Description Clauses* (SDCs) that define the series of instructions that have to be followed. The robot iteratively updates its prediction about the environment and chooses an action according to its policy, which depends on the actual state and SDC. An action is any edge in the actual created map of the environment. The map itself consists of vertices that represent viewpoints from where unknown parts of the map can be discovered and edges that connect the vertices with paths the robot can follow.
The policy is learned by an iterative LfD technique and realized as a multi-class classification problem. According to the actual belief, a trajectory is sampled and then followed. For all visited states, the classifier is updated according to the training set which consists of multiple expert's trajectories and corresponding natural language route descriptions that are also separated into SDCs.

Tellex et al. [TKD$^+$11] also use SDCs to parse the natural language route description. The hierarchy of the SDCs is then used to generate a *Generalized Grounding Graph* (G^3) that maps the single parts of the natural language command to actions and grounded objects, paths and places in the known environment.
To create the graph the system has to be trained first. To do so the corresponding SDC for a part of a given natural language command is manually annotated. The same is true for objects and places. The actions and paths are trained, given a set of recordings of the whole state and the chosen actions together with a set of natural language commands. These commands are collected from different users and should describe the whole shown scene.

All these approaches have in common that the user provides a sequence of actions that the robot has to execute in form of a natural language command. Even if learning from demonstration methods are used, natural language is never addressed as a supporting information source for the learning process.

2.3.2 Natural language and vision

In the vision community the usability of natural language has also gained more attention. Here most works are object related. As an example Yoshizaki et al. [YNK03] use speech to provide the position of an object. A similar work addresses the problem of using vision to handle utterances while referring to an object with speech [HYNK04]. A recent work of Matuszek et al. [MFZ+12] addresses the problem of learning which objects are referred to with a spoken description. They create a perception over which spoken adjectives belong to which features by providing a training set of objects and corresponding descriptions. Cakmak et al. [CCT10] use speech to label if a combination of two objects belongs to a concept, for example a house, or not. In this work, the impact or usability of natural language as input was not in the focus. Instead this scenario was used to analyze the impact of different types of *active learning*.

2.3.3 Active learning

One of the most obvious field in robotics where speech should be used is active learning. Here the learner decides, instead of the teacher, what kind of input would achieve the highest information gain. It can be integrated into LfD systems for example by asking for additional trajectories that can be provided without speech or using natural language for any kind of information. As mentioned before, the work of Cakmak et al. [CCT10] use natural language as input to label combinations of two colored blocks as members or nonmembers of several concepts. Given a new instance the teacher can also demand the robot to classify if it belongs to a certain concept. In this study the impact of several active learning strategies are analyzed. It is shown, that non-expert human teachers prefer that the robot makes queries only if prompted. In a similar work, Cakmak and Thomaz [CT10] also discover, that, in cases of concept learning, non-expert human teachers have trouble giving optimal negative examples.

2.3.4 Learning from demonstration and natural language

Rybski et al.[RYSV07] introduce a combination of pure learning from speech and learning from demonstration. They represent the skill by a plan. In the learning from natural language part, it is created by an *if-otherwise-before* construct. The user defines a precondition followed by a series of action. Here the set of possible actions is predefined. After that the teacher can give an alternative for the case that the precondition is not fulfilled. It is also possible to nest multiple if-otherwise-before constructs to allow complex plans.
The learning from demonstration part is limited to the navigation of the robot. The user can tell the robot to follow him. Then moving tasks are added to the plan automatically as long as the teacher moves. If he or she stops additional *if-otherwise-before* constructs can be added to the plan as described before. After creating the plan the robot searches for undefined *otherwise cases* and if one is found it asks the user to provide a series of actions or *if-otherwise-before* constructs if wanted. This is done to reduce the problem that creating strongly nested plans can be very hard to be provided by the user.

Akgun et al. [ACYT12] introduce new concepts how the data can be recorded in a LfD setting. In addition to the usual case, where a trajectory is recorded, the teacher is able to only add single frames in the so called *Keyframe Demonstration* mode. The whole learning process is controlled using natural language. As an example the learning process is started by saying "new demonstration" and new frames are added when the teacher says "record frame". In a third mode the teacher can switch between the keyframes created in the keyframe demonstration mode. Then the user can delete the frame, modify it or add a new one. In all three cases the skill is represented as a GMM. When training the GMM the only difference between the keyframe demonstration mode and the classical approach is that the number of Gaussians is not fixed but set to the maximal number of keyframes in all demonstrations.

All in all the opportunities provided by natural language has only be discovered in more detail for route descriptions and in the vision community. For learning from demonstrations natural language is used to create another form of demonstrated input data or it is only used separately to create a plan from speech or from demonstration. Therefore this thesis investigates the use of natural language to support existing learning from demonstration methods to address this uncovered field of research. With respect to the limitation of time, the scope is reduced to low level mapping methods. Additional the representation of skills in task spaces are in the focus since they are correlated with spoken descriptions.

3 Background

In the following chapter, the methods used to implement the learning from demonstration system discussed in chapter 5 are presented. First the dynamic time warping algorithm is investigated based on the description of Muhlig et al. [MGSG09]. Then linear regression with non-linear features is explained according to the lecture script of Marc Toussaint [Tou13]. The notation is slightly changed with respect to the subsequent discussed probabilistic movement primitives. ProMPs are a skill representation introduced by Paraschos et al. [PDPN13]. At last it is discussed how the motion of robots can be described in general. It is shown how this leads to the motion generation that is already implemented on the real robot presented in chapter 5 which is used for the experiments. This discussion is based on the lecture script of Marc Toussaint [Tou14].

3.1 Dynamic time warping

To deal with the problem that demonstrations could vary in time the data has to be temporally aligned. This can be done by the often used [CGB07, MGSG09] dynamic time warping algorithm. It is realized as a dynamic programming approach as shown in algorithm 3.1. First the difference between the value for every two combinations of time steps are derived. In this work the euclidean distance is used for this purpose. Then the minimal cost to go is calculated. This is done by adding the minimal cost to go value of a pair that is in one or both time dimensions one step earlier to every distance.

To warp the signal the path from the goal to the start has to be considered that chooses the minimal cost to go. Every time it is cheaper to change the time step of the first signal the current considered value of the second signal is added to the warped signal. The result then represents the optimal transformation of one trajectory onto the other since the over all distance is minimized.

Algorithm 3.1 Dynamic time warping

```
procedure DTW($signal_1$,$signal_2$)
    for $i = 0...signal_1.length$ do
        for $j = 0...signal_2.length$ do
            $\mathsf{distance(i,j)} \leftarrow$ COMPUTEDISTANCE($signal_1(i)$,$signal_2(j)$)
        end for
    end for
    for $i = 0...signal_1.length$ do
        for $j = 0...signal_2.length$ do
            $\mathsf{dtw(i,j)} \leftarrow \begin{cases} \mathsf{distance(i,j)} & i = 0 \wedge j = 0 \\ \mathsf{distance(i,j)} + \mathsf{dtw(i-1,j)} & i > 0 \wedge j = 0 \\ \mathsf{distance(i,j)} + \mathsf{dtw(i,j-1)} & i = 0 \wedge j > 0 \\ \mathsf{distance(i,j)} + min \begin{pmatrix} \mathsf{dtw(i,j-1)} \\ \mathsf{dtw(i-1,j)} \\ \mathsf{dtw(i-1,j-1)} \end{pmatrix} & \text{otherwise} \end{cases}$
        end for
    end for
end procedure
```

3.2 Linear regression with non-linear features

One field of machine learning is regression. Here the goal is to find the parameters of a given model that represent a finite data set the best with respect to a given cost function. In the subfield of linear regression the data is represented by a linear model:

$$f(x) = w_0 + \sum_{i=1}^{n} w_i * x_i = \mathbf{x}^T w \tag{3.1}$$

In equation 3.1 w is a vector containing all parameters and x is the input vector. **x** is the augmented version of the input vector with an additional 1 in the front ($\mathbf{x}^T = (1, x_1, ..., x_n)$). To find the optimal parameters w this can be described as an optimization problem with a least square cost function:

$$L^{ls}(w) = \sum_{i=1}^{n} (y_i - \mathbf{x}_i^T w)^2 = \|\mathbf{y} - \mathbf{X}w\|^2 \tag{3.2}$$

Here $\mathbf{y}$ denotes the combined output vector of all data points and $\mathbf{X}$ is the combined matrix of all augmented input vectors. The optimum can be derived analytically by setting the derivative of equation 3.2 to zero. The result is:

$$(3.3)\quad w^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

To realize non-linear models that can be optimized with linear regression the input vector can be replaced by non-linear features. This feature $\phi(x)$ is a function that maps the input vector to another arbitrary dimensional vector. The optimal parameters can be found the same way as in equation 3.3 with the only difference that the augmented input vectors in $\mathbf{X}$ have to be replaced with the corresponding features.

3.3 Probabilistic movement primitives

As a recent approach for learning from demonstration Paraschos et al. [PDPN13] introduced probabilistic movement primitives. It is a probabilistic formulation that combines several properties in one approach, such like co-activation by the product, modulation by conditioning and learning by maximum likelihood methods.

A recorded trajectory $\{q, \dot{q}\}_i$ is encoded by a linear function model with non-linear features. As the feature Gaussian basis functions are used for striking movements and for rhythmic motions Von-Mises basis functions are chosen. As an example the Gaussian basis function is defined as:

$$(3.4)\quad b_i^G(t) = \exp^{(-\frac{(t-c_i)^2}{2h})}$$

Here c_i denotes the center of the basis function and h the width. For the basis matrix each column can be normalized by dividing each entry with the sum of all other column entries. For a temporal modulation the time can be substituted by a phase variable that may change non-linear. Independent on the type of basis function, the model can be described for each joint as:

$$(3.5)\quad y_t = \begin{pmatrix} q_t \\ \dot{q}_t \end{pmatrix} = \Phi_t^T w + \epsilon_y$$

Here ϵ_y denotes the variation of the movement and Φ_t the basis matrix that consists of n rows depending on the number of basis functions and m columns for each dimension of the output, in this case two for the configuration and velocity. Assuming that the variation ϵ_y is Gaussian distributed, the probability of a trajectory τ is for the given basis function model:

$$(3.6)\;\; p(\tau|w) = \prod_t \mathcal{N}(y_t|\Phi_t^T w, \Sigma_y)$$

To encode the coupling between all joints a joint basis function model can be used instead of treating each one independently. For this purpose a joint weight vector $\mathbf{w} = (w_1^T, ..., w_n^T)^T$ and an extended basis matrix Ψ_t has to be defined as a block diagonal matrix with Φ_t as the diagonal entries:

$$(3.7)\;\; \mathbf{y}_t = \begin{pmatrix} y_{1,t} \\ \vdots \\ y_{d,t} \end{pmatrix} = \Psi_t \mathbf{w} = \begin{bmatrix} \Phi_t & \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \Phi_t \end{bmatrix} \mathbf{w}$$

The coupling can then be encoded by the distribution $p(\mathbf{w};\theta)$ of $\mathbf{w}$. Assuming that the weights are also Gaussian distributed the resulting probability for an observation y_t is given by:

$$(3.8)\;\; p(y_t|\theta) = \int \mathcal{N}(y_t|\Phi_t^T w, \Sigma_y)\mathcal{N}(w|\mu_w, \Sigma_w) = \mathcal{N}(y_t|\Phi_t^T \mu_w, \Phi_t^T \Sigma_y \Phi_t + \Sigma_y)$$

The parameters $\theta = (\mu_w, \Sigma_w)$ can be derived by any maximum likelihood estimation. To model via-points and final positions the model can be conditioned by adding the desired observation and applying Bayes theorem:

$$(3.9)\;\; \mu_w^{new} = \mu_w + \Sigma_w \Psi_t (\Sigma_y^* + \Psi_t^T \Sigma_w \Psi_t)^{-1}(y_t^* - \Psi_t^T \mu_w)$$

$$(3.10)\;\; \Sigma_w^{new} = \Sigma_w - \Sigma_w \Psi_t (\Sigma_y^* + \Psi_t^T \Sigma_w \Psi_t)^{-1} \Psi_t^T \Sigma_w)$$

Here y_t^* denotes the desired observation at time step t and Σ_y^* denotes its accuracy.

3.4 Motion and kinematics

By introducing task spaces the motion, which is performed according to a learned skill depends not only on the general representation of the skill. To investigate other impacts on the generated motion a brief discussion on the motion generation used in this thesis is given which is based on the field of kinematics.

3.4.1 Kinematics

For Robotics, a subfield of kinematics is the problem of finding positions and orientations of the robot's parts in world coordinates, while knowing its joint angles and geometry. For simplification it is assumed that the robot only consists of ideal rigid parts so that its shape does not change under external forces. Another assumption is, that it can be described as an kinematic tree with its body as the root. This means, looking from its root each part has only one "parent" it is attached to. As an example one can consider a humans arm. Each finger has the hand as its parent which itself is attached to the forearm. Together with the upper arm this branch is connected to the shoulder which could be seen as part of the body and therefore as the root.

Looking from the root one can define a fixed coordinate system at each part's end. To describe the transformation between the part j and its parent i the joint configuration and the shape of the part has to be considered. For this a rotation matrix $R_{i \to j}$ for the joint and a translation shift t for the shape is defined. The combination of both describes the complete transformation $T_{i \to j}$ of the joint's coordinate system to the free end's coordinate system.

With the previous assumptions one can derive the transformation from the root to the end-effector's coordinate system, that is the part of the robot that is used to interact with the environment (e.g. for a humanoid robot its hands or for an industrial robot its welding equipment), directly. All transformation matrices from the first joint to the last has to be multiplied in the given order:

$$(3.11)\quad T_{root \to endeffector} = \prod_{i=0}^{max-1} T_{i \to i+1}$$

3.4.2 Kinematic maps and task spaces

It is not necessary to limit kinematics to find the position or orientation of the joints in world coordinates. One can also define other features $\phi(q)$ that are dependent on the joints configuration. An example is the alignment of the robots hand to a tables surface. It can be described as the scalar product between the surfaces normal and the transformed desired direction of the hand. Another example is the distance between the hand and an object, that is described as the euclidean norm of the distance between both positions in world coordinates. The transformation from the joint angles to these measurements are called kinematic maps and their space is further referred to as a task space.

3.4.3 Inverse kinematics

So the problem of finding positions and orientations in the world frame while knowing the robot's joint states is called kinematics. Accordingly the reverse problem, finding a joint state to a given position in world coordinates is called inverse kinematics. Despite before, it is not every time possible to find a unique solution.

One problem is the nullspace motion. It is often possible to find an arbitrary number of solutions. A simple example is a body with two joints that are connected with a straight shape and whose rotation axes are centered in the shape and are pointing along it. Here the endeffector's coordinate system does not change as long one joint rotates exactly the same but in the other direction as the other joint. Another aspect is the singularity. Despite before not every element of the domain can be transformed. This is for example the case if the desired position of the endeffector is not reachable due to the arm's limitations.

This means, that an inverse transformation of the kinematics is not possible. To address this it is often useful to find the desired joint state by solving an optimization problem:

$$q^* = argmin_q \|\phi(q) - y^*\|_C^2 + \|q - q^*\|_H^2 \tag{3.12}$$

Here the first part of equation 3.12 penalizes deviations from the desired goal and the second part penalizes larger changes in the joint configuration. While the first part solves the singularity problem, since it finds the closest solution to the goal, there still may exist several results. This nullspace problem is addressed by the second term which would choose the nearest configuration to the current one if several optimal configurations exist.

3.4.4 Motion profiles

Since any robotic system has a mass and therefore can only change its configuration continuously the motion to the desired state has to be modeled. The form of the movement is often called the motion profile. It can be defined in the joint configuration just as in the task space dependent on the desired behavior. The only difference is that for the latter the corresponding joint configuration has to be calculated using inverse kinematics. For example if a straight movement of the endeffector in the world frame is desired this has to be modeled in the corresponding task space and then be transformed.

The choice of the type of the motion profile can be taken according to additional goals. For example one can desire a smooth motion. To do so the desired acceleration can be modeled by a sinus. If one want the fastest motion possible but without violating a maximum acceleration and speed the motion profile would be created by using the maximum acceleration until the maximum speed is reached and at the appropriate time it would use the maximum negative acceleration to slow down until the desired goal is reached.

3.4.5 Euler Lagrange equation

So for any given kinematic map it is possible to find the optimal joint configuration and a motion profile to reach it. But a general robot is controlled by torques and not by accelerations. To calculate the needed controls the Euler Lagrange equation can be used:

(3.13) $\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = u$

Here $L(q, \dot{q})$ denotes the Lagrangian that is defined as the difference between the kinetic and the potential energy:

(3.14) $L = T - U$

The general solution of equation 3.13 is often referred to as the *general robots dynamic*:

(3.15) $M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) = u$

Here $M(q)$ denotes the inertia, $C(q, \dot{q})$ the Coriolis forces and $G(q)$ the gravity force. Knowing the specifications of the robot and under the assumption that the robot only consists of rigid parts this forces and the inertia can be computed efficiently.

3.4.6 Operational space control

Similar to inverse kinematics (see equation 3.12), the problem of finding an optimal control that increases the satisfaction of all tasks the best can be described as an optimization problem in a least square sense. For each task, the difference between the desired acceleration in the task space and the resulting change by a given acceleration of the joints is penalized by its squared norm:

(3.16) $u^* = argmin_u \; \|u\|_H^2 + \sum_i \|\ddot{\phi}_i(q) - \ddot{y}^*\|_{C_i}^2$

The metric C_i can be used to generate a selection, which task spaces and furthermore which dimensions of a task space are important. Another aspect is that it also influence the resulting acceleration and therefore the time needed to reach a goal. This metric is further referred to as the precision.

The optimal control can be derived analytically by setting the derivative of equation 3.16 to 0 and using equation 3.15 of the general robot dynamics.

4 Instruction learning

When teaching a human in a classical learning from demonstration setting, the teacher provides different information with natural language. To discuss the types of information a common example like a pick and place task that consists of four parts is considered: reaching a cup, grasping it at the handle, moving it to a table and placing it there.

In such a hierarchical task the teacher would use natural language to name which series of actions has to be done. This description is always on a high level. For example he or she could say "First you have to take the cup and then put it on the table.". How the meaning of such a sequence of instructions can be learned to later follow new commands has been already studied in the work of Duvallet et al. [DKS13] and Tellex et al. [TKD+11].

Depending on the belief over the knowledge of the learner and the complexity and uncommonness of the task a teacher would also give additional information using natural language. In the previous example if teaching a baby the teacher may predict that it does not know how to grasp a cup. In this case the teacher could also state what is important, such that it has to be grasped at the handle. The same is true if it is a very complex or uncommon task since the teacher may predict that otherwise something will be forgotten or that the learner just does not know what is important. This type of input is addressed at section 4.2 where the teacher can select important or discard unimportant task spaces.

It is also common to ask the learner to demonstrate the learned skill and then verify the progress of the learning process according to the reproduction. If it is not successful the teacher normally shows the whole skill or only parts of it again. This paradigm was also addressed in the work of Akgun et al. [ACYT12] and Cakmak et al. [CCT10] and is further discussed in section 4.3.
Another possibility is to use natural language to tell what was wrong or has to be done differently. One part of this variety of information is also covered by the task space selection discussed in section 4.2.

4.1 Interactive learning

In contrast to the intuitive interactive learning process when teaching a human some additional interactions has to be considered when teaching a robot. Here the use of natural language also provides a lot of opportunities for making the learning process more interactive and comfortable. Especially for LfD where the movement of the teacher is recorded commands like "start demonstration" and "stop demonstration" to define start end endpoint of the recording are a benefit as in the work of Akgun et al. [ACYT12]. Otherwise a second person is needed to control the process or assumptions as in the work of Muhlig et al. [MGH$^+$09] have to be made. In their approach the objects have to be moving during the demonstrations. Between them there has to be a break where they stand still. This is needed so that the data can be acquired continuously and be separated later.

4.2 Selecting task spaces by natural language

As discussed in the section 3.4 if a skill is represented in task spaces and the movement is generated by an operational space controller, the precision of a task space can be used to represent the skill in addition. But it has to be examined how this precision has to be modeled. For this purpose the previous pick and place task is considered again.

As the pick and place task can be divided into four parts, reaching a cup, grasping it at the handle, moving it to a table and placing it there, it is a good example for a complex task where a hierarchical approach can be used.

For the first part it depends on the object which task spaces are relevant. Here, grasping a cup at its handle, the relative position and orientation of the hand to the object is important. In the case of a bottle only the distance to the bottleneck and the alignment to its surface would be important since it can be grasped from any side. Only regarding this task spaces would fulfill the task but it is not ensured that the movement would look human like since the joints itself are not considered. Therefore a blending between the joint configuration being more important at the beginning to the relative position of the hand to the object at the end would be preferable.

For grasping, the important task spaces depends on the characteristics of the gripper. The relative position of the fingertips to the object may be important or for simpler end-effectors only the relative position of the hand and the degree how opened it is. Additional the pressure at the fingertips would be important to ensure that the object is not dropped too early.

To reach the goal position the relative position to the hand is important. For the dropping task it depends again on the characteristics of the gripper.

Similar to the work of Muhlig et al. [MGSG09] where task spaces are selected using vision, this can be done more elegant with the use of natural language. Despite determining an object as important by shaking it, an arbitrary number of objects can be labeled as important and also as unimportant. Since several task spaces can be defined that relates to the object it is preferable that the user can be more precise than selecting all. As an example, the teacher can say "the right hands position is important" or "the relative position of the object is unimportant". The expert can also give information about the level of importance like "very important" or "a bit important".

Since natural language is not the only input, task selection mechanisms that are based on the demonstration data should also be applied. This can be done with the *similarity criterion* introduced by Muhlig et al. [MGSG09]. Here the time dependent variance can be mapped to the precision p at time sept t for the ith task space by the following equation:

$$p_{t,i} = \begin{cases} p_i^{max} - \frac{p_i^{max}}{\sigma_i^{max}}\sigma_{t,i} & 0 \le \sigma_{t,i} < \sigma_i^{max} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

Here p_i^{max} is a constant that should be chosen according to the operational space controller since it determines the maximum precision. σ_i^{max} is a constant that should be used to model the mapping with respect to the task space dimensions.

After initializing the precision according to the above equation it can be modified by the natural language instructions. Labeling a task space as unimportant should discard it. This can be done by setting the precision to zero. In the other cases a factor or bias can be used. As the *similarity criterion* can create values close to zero the use of a factor cannot ensure a high precision for important labeled task spaces. Therefore a bias is preferable. Since the maximum precision is chosen with respect to the operational space controller to avoid too high torques it should still limit the possible value. According to this equation 4.1 is expanded to the following rule:

$$p_{t,i} = \begin{cases} min(p_i^{max}, p_i^{max} - \frac{p_i^{max}}{\sigma_i^{max}}\sigma_{t,i} + p_i^{min}) & 0 \le \sigma_{t,i} < \sigma_i^{max} \\ p_i^{min} & \text{otherwise} \end{cases} \tag{4.2}$$

Here p_i^{min} can be chosen according to the level of importance.

4.3 Correction instruction

As discussed before for teaching a human a skill, it is normal to correct the learner if he makes a mistake while demonstrating the actual state of learning. In the following a first attempt to introduce such an interactive correction method in robotics is presented.

To avoid correspondence problems between the learner's demonstration and the teacher's correction, positive and negative examples are given only for a single time step. Therefore the teacher can first demand the robot to demonstrate the learned behavior. Then if the robot differs from the desired behavior the teacher can pause the demonstration and correct the robot's state. Only the corrected state is recorded, not the trajectory to reach this state. It is possible to demonstrate several corrected states. Additional the correction can be labeled as a "good" or "bad" behavior.

The question is now how this information can be used to improve the skill. As we focus on works, where instead of only the joint states an arbitrary number of task spaces are used, the most general approach would influence the precision of a task space. To decide which task spaces are relevant two measurements can be used: how heavy the task space was violated at the demonstration, given as the distance between the demonstrated state and the desired state and how heavy the corrected state is violated, given as the distance between the corrected state and the desired state.

4.3.1 Violation criterion

In addition to this measurements also the label if its a good or a counterexample has to be considered. In the case of a good demonstration and according to the two distance measures one can separate four cases:

First of all there could be no violation at all. This could be the case if the correction has nothing to do with the corresponding task space. An example would be how opened the hand is in the reaching step in the pick and place task. Or it could be relevant but the belief over its importance is correct. In the same example as before this could be the case for the relative position of the hand, if only its orientation was wrong. All in all if nothing is violated nothing should be changed.

Another case is that the desired state is more violated if corrected. In this case the belief about the importance of the corresponding task space has to be decreased, since it would prefer the incorrect robot's demonstration over the corrected one. But the level of violation has to be considered, since another task space could have caused the incorrect skill reproduction and the current violation only arises from a perturbation.

The third possibility is that the task space is violated during the demonstration more than during the correction. In this case the belief about the importance of the corresponding task space has to be increased, since it would prefer the corrected demonstration over the incorrect robot's one. Again the distance between both states has to be considered for the same reason.

The last option is that the distance of the desired state is every times big. Since the corrected state violates the desired state an increased precision cannot result in a better reproduction of the skill. For the same reason a reduction of the precision cannot worsen the reproduction. Therefore it should be reduced.

When giving additional corrections that are labeled as "negative" examples it has to be distinguished if it is relevant for the current task space. For example if the position and orientation of the hand is important the teacher could provide a "negative" example where it is only rotated.

Here the position is not addressed but it is close to the desired one. Therefore giving additional "negative" examples is not useful.

This is different for additional, as "good" labeled corrections because it is a stronger statement. It is not necessary to distinguish which task space is of relevance. Here just the worst "good" example would count.

All in all a first attempt would be to check if all "good" examples are closer to the desired state than the "negative" examples to which also the state belongs when the robot has been stopped. In this case the precision has to be increased. Otherwise they have to be decreased.

4.3.2 Temporal smoothing

As discussed most works do not use natural language for additional information in the case of learning from demonstration. All LfD methods implicitly label all expert trajectories as positive examples, despite the teacher could also label demonstrations as counter examples. The reason that the use of negative demonstrations has not been discovered is that it cannot be derived which part of the trajectory leads to the failure. In the case of a correction, even though a concrete time step is given, it is still the problem to derive for which part of the whole skill this information is important. Additionally this problem has to be concerned for positive demonstrations.

As an naive approach to address this, the bias is applied to the whole neighborhood of the correction time step but with decreasing impact according to the temporal distance. This is done by smoothing the bias with a Gaussian but without its normalization:

$$(4.3)\quad p_t = p_t + p_{bias}\ \exp^{-\frac{(t_{cor}-t)^2}{2\sigma}}$$

Here σ can be chosen to represent the region in which the result has to be applied. t_{cor} denotes the time of the correction instruction.

5 Implementation

5.1 Robot platform - PR2

As the robot platform the Personal Robot 2 (PR2) is used which is operated by the *Robot Operating System* (ROS). Both are developed by Willow Garage. The PR2 is an open platform designed for human like behavior as in personal robot applications. Each of its arms has 8 degrees of freedom, its torso can be moved up and down and it has four wheels for navigation. In this work from all of the available sensors only the Microsoft Kinect is used for the object recognition.

5.1.1 Speech synthesis

Speech synthesis is needed to inform the user that the robot has understood a command. As the speech synthesis tool, the ROS package *sound_play* is used. One advantage of *sound_play* is that it is a ROS package which can use the speakers of the PR2 directly. Another advantage is that it provides easy to use C++ and python APIs for playing wave-files, build-in sounds or to synthesis strings to speech.

5.1.2 Object recognition

As the object recognition is not in the focus of this work *Augmented Reality Tags* (ARTags) are recognized and recorded instead of objects. For this purpose the existing ROS package *ALVAR* is used. It uses ARTags to identify and publish the position and orientation every time it recognizes one.

The mapping of the spoken name of an object to its corresponding ARTag is hard coded. This is done because several ways to implement a grounding of natural language to an object has already been introduced in other works [CCT10, TKD$^+$11, MFZ$^+$12].

5.1.3 Activity machine

The main controlling structure of the robot has been realized before by a so called *activity machine*. It manages a list of activities which themselves defines tasks the robot has to solve. For this work the activities to follow a trajectory in a specific task space and to relax the joints are of relevance. The first updates a desired state in the task space which is used by an operational space controller to find the actual best control signal. Instead of a precision metric only a scalar can be passed. The second activity sets the gains of all joints to zero.

5.2 Speech recognition

As the speech recognition software Nuance Dragon NaturallySpeaking is used which is one of the leading dictation software. It has a speech recognition accuracy up to 99% from the beginning [Nua14] which can be increased by training or by just using the software. The premium version provides the possibility to create commands that copy a predefined text into any or only into a specified program each time the command is spoken. It is also possible to dictate freely into most programs that contain a text box.
Since the speech recognition itself is not in the focus of this work, the benefits of Nuance Dragon NaturallySpeaking, such as full automated accuracy improvements and a full predefined vocabulary make it to the tool of choice. Additional for simplification and due to the limitation of time the possibility of creating own commands fits perfectly.
As an disadvantage it is only available for Windows. Therefore an additional interface between the speech recognition software and the LfD system is needed. Since only speech commands are used that pasts a predefined text into any program, this is done by a terminal program. Since the basic functionality of ROS has been ported to windows, it works as a ROS-publisher node. Each word that is written is published directly and can be used by any subscriber. In the case of speech commands the predefined text is pasted into the terminal window and therefore published directly.

5.3 Interactive learning process

As discussed in chapter 4 one goal of using natural language is to make the learning process more interactive. Therefore the whole work flow is controlled by speech commands. During the general initialization the last learned skill is loaded and additional parameters are set. After that, a main state machine can switch between the learning process, the demonstration of the learned skill and a teleoperation mode.

5.3.1 Teleoperation mode

The teleoperation mode provides the ability to navigate the robot with natural language. It is invented as a comfort feature for example to drive the robot closer to a table if needed for a demonstration. The user can rotate the robots base and drive it in all 4 main directions for a fixed time or until he says "stop". The speed of rotation and translation can be set to three levels by saying "full-", "half-" or "low speed".
The PR2 is then controlled internally using its low-level base controllers. They can be used by just publishing <geometry_msgs::Twist> messages to the "/base_controler/command" topic. As a disadvantage, no distance can be passed. The robot moves as long as the message is published. Therefore an internal state machine is used that enters a publishing state and flips back into the idle state when finished. To drive the robot for a fixed period of time the user has to say for example "move a step forward". Otherwise the movement is started by saying "move" plus the direction and finished by saying "stop". This can be done for rotational movements accordingly. To smooth the motion of all commands with a fixed duration a sinus-profile is used for the motion generation as discussed in section 3.4.4.

5.3.2 Demonstration mode

In the demonstration mode the robot performs the learned skill. First it is checked which objects has not been found by the ALVAR node and all related task spaces are deactivated. The user is informed that the robot starts the demonstration and therefore is told to take attention. Then for each active task space an activity to follow the desired trajectory is created. During the robots demonstration the user can say "stop" to start a correction instruction. In this case the robot stops and waits for the user to define which of both hands will be corrected. The teacher has to choose one of both by saying for example "the right hand". This has to be done because both arms will become relaxed and due to the gravity forces they will change their configuration even if the user does not want to correct both. After choosing one hand the user can say "like this" or "like that" to label the actual state as a positive example. It is also possible to say "not like this" or "not like that" to label it as a negative correction. After an arbitrary number of corrections have been provided the user has to say "continue" to let the robot finish his demonstrations. After that the correction instructions are used to update the precision function.

5.3.3 Learning mode

When the state machine switches to the learning mode first of all the robot relaxes its arms so that the user can move them during a demonstration. This is done by a relaxing activity that sets the gains of the joints of both arms to zero. It informs the user about this state change by saying "You have my attention.". Now the user is free to delete the actual skill, to add new demonstrations, to command the robot to demonstrate the actual state of the learning progress or to define which task spaces are relevant.

To delete the skill, the user has to say "forget everything". As expected the user has to confirm this command by saying "yes". By saying "no" or if the command is not confirmed after ten seconds nothing happens. Otherwise the whole learning progress is discarded.

If it is the first time since the learning mode has been started the user can begin with the demonstration by saying "start demonstration" or "start recording" directly. Otherwise "new demonstration" has to be said first. Then the states of all task spaces are recorded with a rate of 100 Hz. When the user says "stop demonstration" or "stop recording" the data is sub sampled to 1000 data points. For simplification it is assumed that only smoothed movements are demonstrated that does not last longer than one minute. According to this a sample rate of about 16.67 Hz is reached at minimum and there is no need for a complex regression method. The model of the skill is only updated using the new demonstration when the learning mode is left or the user commands the robot to demonstrate it. To do so the user can say "show me what you've learned" or "imitate". Then the same procedure as discussed in section 5.3.2 is called.

The last option is to enable or disable task spaces by speech. Because of the limitation of time and since object related language grounding [MFZ$^+$12] and a combination with vision [HYNK04] has already been discovered the relation between an object description and the representation as an ARTag is hard coded. For the same reason the spoken language is not interpreted by a parser. Instead an object's name can be said that defines which task spaces are of interest. Additional the user has the option to give a more detailed description. The descriptions are optional but if used, only task spaces that are of the corresponding task map type are influenced. Otherwise all task spaces that are related to the object are affected. After that the user can say "is very important", "is important" or "is unimportant" to finish the selection.

As shown in table 5.1 and 5.2 the user is able to say for example "the right hand's position is important" or "the position of the right hand is important". In both cases the same result is achieved. All task spaces that are related to the internal name of the right hand and are of a task map type related to the position are labeled as important and its minimal precision is set to half of the most possible. In the case of "very important" its precision is set to three quarter of the maximum and if it is unimportant it is just labeled as unimportant.

5.4 Speech supported learning from demonstration

By using speech, the learning from demonstration work flow is much more interactive as shown in the previous section 5.3. The user can add new demonstrations, can select or discard task spaces or he can advice the robot to show what it has learned. While reproducing the

spoken description	**corresponding task map type**
"the relative position of" / "relative position"	relative position
"the relative orientation of" / "relative orientation"	alignment
"the position of" / "position"	position
"the orientation of" / orientation"	quaternion

Table 5.1: Descriptions that reduce the scope of related task spaces

spoken name	**corresponding object**
"the right hand" / "the right hand’s"	right endeffector
"the left hand" / "the left hand’s"	left endeffector
"the green block" / "the bottle’s"	artag_1
"the blue block" / "the green block’s"	artag_5

Table 5.2: Examples of objectnames and the corresponding internal representation

acquired skill, the robot can be stopped and manipulated by a correction instruction. The motion to reproduce a skill is generated by an operational space control as described in section 3.4. Therefore it calculates the controls that would decrease the optimization problem most which is defined by the different activities.

5.4.1 Skill representation

For this work only the movement of the robots arms and the recognized objects are considered. Therefore the internal state of the robot is described by its joint angles $\{q\}_{0:T}$ and velocity $\{\dot{q}\}_{0:T}$. Together with the position and orientation of the marked objects the following set of partly conflicting task spaces are created for each endeffector:

- the position
- the orientation
- the joint configuration of the gripper
- the relative positions to all objects
- the alignments between the grippers main axes and the objects main axes

Since the development of activities for obstacle and joint limit avoidance would go beyond the scope of this work, corresponding task spaces are not included.

In this work, a skill is represented as a probabilistic movement primitive [PDPN13] as described in section 3.3 but only considering its state and not its derivative. According to this the skill is represented by a basis function model for each task space separately. In this work Gaussian basis functions are used as for simplification only not cyclic movements are concerned:

(5.1) $b_i^G(t) = \exp^{(-\frac{(t-c_i)^2}{2h})}$

The weights of the model are encoded by its mean and covariance $\theta_i = (\mu_{w,i}, \Sigma_{w,i})$. Additional for each task space a time dependent precision trajectory is given.

5.4.2 Kinesthetic teaching

During the data acquisition, the gains of the joints of the PR2 are set to zero. This enables the teacher to move the robots arm during a demonstration. The start and beginning of the recording is indicated by the spoken command "start demonstration" and "stop demonstration". As discussed by Calinon et al. [CGB07] kinesthetic teaching has the advantage that no correspondence problem has to be considered but the disadvantage is that whole body motions cannot be acquired. Therefore this technique is a good choice since in this work only the movements of the arms are of interest.

5.4.3 Policy learning

The skill is represented as a ProMP introduced by Paraschos et al. [PDPN13] but it is learned differently. They used an expectation maximization algorithm to find the parameters of the distribution of the weights of the basis function model. In contrast to that in this work the trajectories are first temporal aligned by a dynamic time warping algorithm as discussed in section 3.1. Then for each trajectory the optimal weights are calculated separately by using linear regression with the Gaussian basis functions as non-linear features as discussed in section 3.2:

(5.2) $w^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},\ with\ X = (\Phi_0^T, \ldots, \Phi_N^T)^T$

Here $\Phi_t = (b_0^G(t), \ldots, b_n^G(t))$ denotes the basic matrix for each of the n time points. After that the parameters are derived as the mean and covariance of the calculated weights.

5.4.4 Task space selection

As discussed in section 4.2 an automated task space selection is done by default that depends linearly on the variance. From the maximum precision the weighted variance is subtracted and negative results are set to zero. After that a bias is added according to the additional information about the importance provided by natural language. Then the values are limited to the range between the maximum and minimum precision.
At last if a reproduction has been corrected anytime, the information is used as discussed in section 4.3. If all positive examples are better than the negative ones and the reproduced skill, a time-scaled bias equal to the maximum precision is added. Otherwise it is subtracted. Since it would not be satisfactory to correct the robot often the bias is chosen as high.

5.4.5 Skill reproduction

To demonstrate a skill first its probabilistic model is updated with the actual position by conditioning as described in section 3.3:

$$\text{(5.3)}\quad \mu_w^{new} = \mu_w + \Sigma_w \Psi_t (\Psi_t^T \Sigma_w \Psi_t)^{-1} (y_t^* - \Psi_t^T \mu_w)$$

Then a trajectory is generated from the updated basis function model by calculating each desired time step as:

$$\text{(5.4)}\quad \mathbf{y}_t = \begin{pmatrix} y_{1,t} \\ \vdots \\ y_{d,t} \end{pmatrix} = \Psi_t \mu_w^{new} = \begin{bmatrix} \Phi_t & \dots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \dots & \Phi_t \end{bmatrix} \mu_w^{new}$$

For each task space a squared cost term is added to the optimization function of the operational space controller as discussed in section 3.4. Whenever a new time step is reached its goal and precision is updated with the corresponding values of the desired state and precision trajectory.

6 Evaluation

To evaluate the implemented learning from demonstration system a drawing experiment is introduced as shown in figure 6.1. Here the robot has to learn to draw a point under a target building block while ignoring another. The learning progress is tested for the solely automatic task space selection and both introduced instruction separately.

6.1 Experiment

To compare the impact of the correction instruction and the automatic and spoken task space selection a drawing skill is investigated. The test and training scene is a graph paper separated into six times five blocks laying on a table. On the paper are two building blocks each marked with an ARTag. The green one that is not relevant for the task is located at the lowest left block in all training and test examples. The relevant, blue building block is located at different positions for each training demonstration. The test settings consists of six different locations.

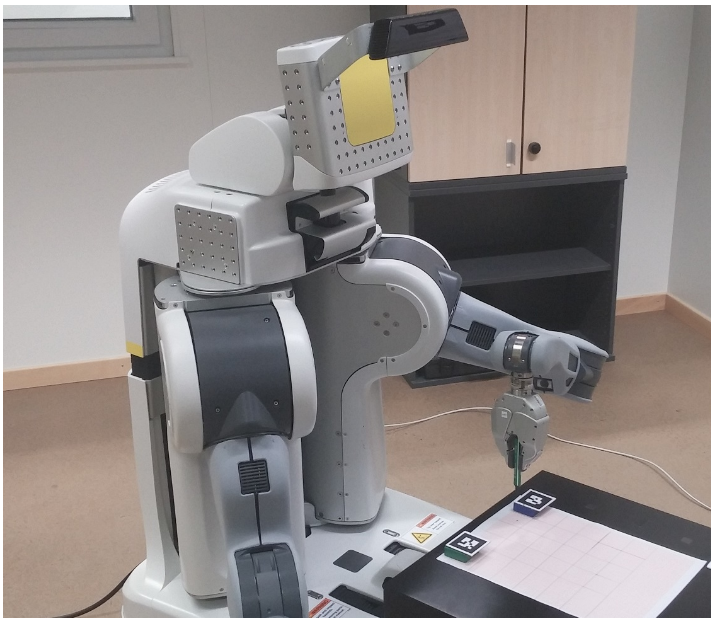

Figure 6.1: The PR2 is shown during the drawing experiment as well as the used ARTags.

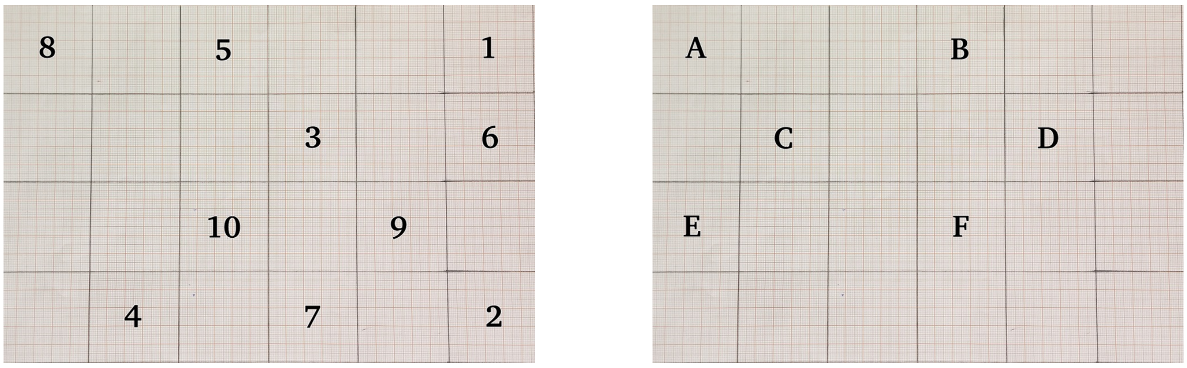

Figure 6.2: The chronological order of settings for the training is presented at the left and the position of the target building block for the tests is shown at the right.

The goal is to draw a point exactly in the middle of the block one lower than the one where the blue building block is located. To train this behavior, the robots arm is first moved from its right side above the target point and then moved down. After that it is moved up and then back to the initial position. This is done for 10 different locations as shown in figure 6.2. Here each position is denoted by a number that represents its chronological order. The parameters for the automatic task space selection are set according to the highest standard deviation when moving the arm and the hand six times randomly. Since in this task the trajectories in all task spaces are much closer the maximum standard deviation is reduced by a factor of five.

For all training sets from one up to 10 demonstrations the automatic task space selection is tested solely and in combination with the spoken task space selection. This is done for six test settings as shown in figure 6.2. The robot is tested three times for each position of the blue building block. When testing the spoken task space selection instruction, the robot is told that the hands position and the green block are unimportant and that the blue block is very important. The correction instruction is only tested with a training set of 10 demonstrations. Here the movement is corrected for the first time at the moment when the pencil is at its lowest point. Only a single positive correction is provided. A second correction instruction is used approximately two seconds before the pencil is at its lowest point. For the standard deviation of the Gaussian smoothing two and a half seconds are chosen.

6.2 Results

As a criterion for the success of the reproduction three measurements are considered. Most important is the shortest distance d_{min} between the drawn lines to the desired point since it was the goal to reach the target point. In addition the furthest distance d_{max} as well as the maximum distance l between two points on the drawn lines themselves are considered. The

N	1	2	3	4	5	6	7	8	9	10
$\varnothing d_{min}$	10.44	8.27	6.73	4.89	4.78	5.14	5.53	4.59	4.96	4.94
$\varnothing d_{max}$	11.59	9.06	7.50	5.67	5.67	5.91	6.19	5.25	5.59	5.54
$\varnothing l$	1.36	0.95	1.02	0.87	1.08	0.87	0.84	0.78	0.68	0.98

Table 6.1: Results of the drawing experiment for the solely automatic task space selection. All values represents the average and are in cm.

N	1	2	3	4	5	6	7	8	9	10
$\varnothing d_{min}$	0.71	0.90	0.98	0.73	0.81	1.00	0.80	0.84	0.87	0.56
$\varnothing d_{max}$	1.38	1.61	1.70	1.57	1.73	1.69	1.63	1.64	1.58	1.23
$\varnothing l$	0.91	1.09	0.88	1.09	1.34	0.90	0.96	0.84	0.90	0.80

Table 6.2: Results of the drawing experiment for the spoken task space selection. All values are in cm and represents the average.

latter is further referred to as the length measurement. Both represent the accuracy of the skill reproduction since only a single point should be drawn. The average results for the automatic task space selection are presented in table 6.1 and for the spoken task space selection in table 6.2. The whole list of measurements can be found in the appendix A.

For the automatic task space selection the average shortest distance is more then halved after four demonstrations from 10.44 cm to 4.89 cm. The same is true for the average furthest distance which is reduced from 11.59 cm to 5.67 cm. These values for both measurement vary only a bit if more than four demonstrations are used as the training set. The standard deviation of the minimal distance is also reduced from 5.06 cm for a single demonstration to 3.28 cm after four training samples. This is also illustrated in figure 6.3. Here the minimal distance of the automatic and the spoken task space selections are compared. While the results of the automatic task space selection varies strongly depending on the position of the target object the standard deviation of the spoken task space selection is equal to or smaller than 0.9 cm.

In figure 6.4 at the left the results for the automatic task space selection are illustrated exemplary for a training set of a single demonstration (red lines), for two (green lines) and for three (blue lines). The automatic task space selection's accuracy increases here from an average minimal distance of 10.44 cm to 6.73 cm. The length measurement is always lower than or equal to 1.36 cm.

The results on the right side illustrate the influence of the spoken task selection for the same number of training demonstrations. Here the colors are vice versa and the used instructions are that the irrelevant construction block and the position of the hand are unimportant and

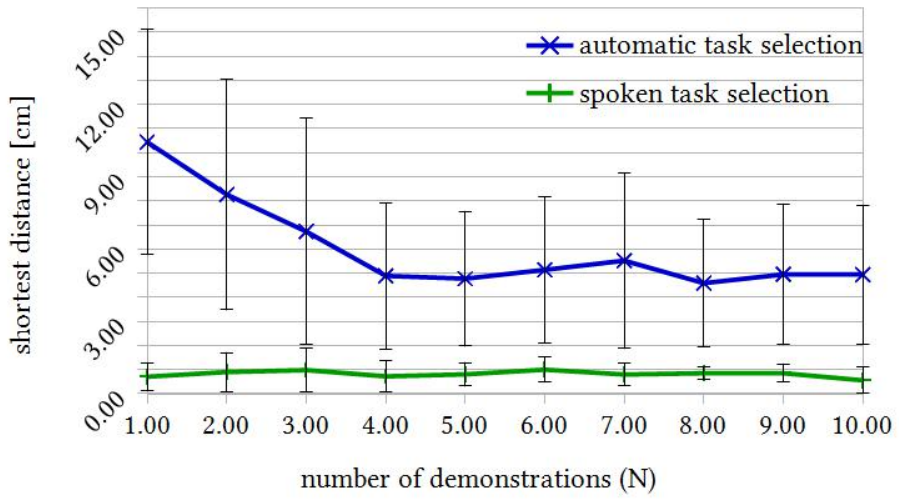

Figure 6.3: Average minimal distance of the automatic (blue line) and spoken task space selection (green line) together with their standard deviations.

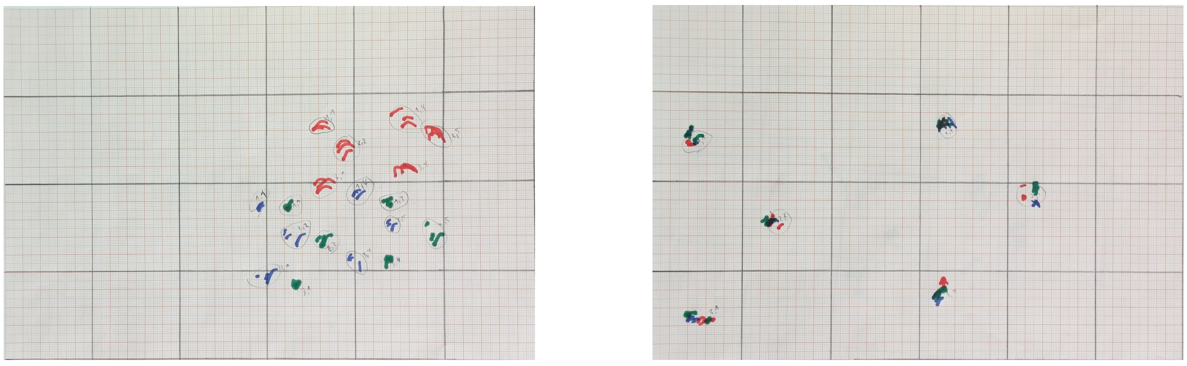

Figure 6.4: Demonstration of the drawing skill only using the automatic task space selection with 6 different positions of the relevant marker (from the left top at:{(1,1), (1,4), (2,2), (2,5), (3,1), (3,4)}). The goal was to draw a point in the middle of the next lower block. The red lines are reproductions with a training set of a single, the green with two and the blue with three demonstrations. At the right image the colors are vice versa.

that the target block is very important. For any number of training demonstrations the average minimal distance is lower than 1 cm and the average furthest distance is lower than 1.73 cm. The length measurement varies between 0.6 cm and 1.1 cm.

	1 correction						2 corrections					
pos.	**1,1**	**1,4**	**2,2**	**2,5**	**3,1**	**3,4**	**1,1**	**1,4**	**2,2**	**2,5**	**3,1**	**3,4**
d_{min}	4.30	0.15	3.80	0.90	4.30	0.90	3.70	0.40	3.20	1.20	3.55	1.40
d_{max}	9.60	1.20	6.30	1.25	7.80	1.10	6.70	1.00	3.90	1.90	5.55	2.10
l	6.75	1.95	3.45	0.35	3.50	0.35	3.70	0.90	0.70	1.40	2.00	0.95

Table 6.3: Results of the drawing experiment for one and two correction instructions and a training set of 10 demonstrations. The position are described from the left and the top. All values are in cm.

The results for the correction instructions are illustrated in figure 6.5. Here the blue lines are drawn after one correction and the green lines after a second correction. Depending on the position of the target building block in some cases a line is drawn instead of a point. Because of that the length measurement of the lines varies strongly from 0.35 cm to 6.75 cm for a single correction instruction and between 0.7 cm and 3.70 cm for two as presented in table 6.3. The average length decreases from 2.73 cm to 1.61 cm and the maximum length is reduced from 6.75 cm to 3.70 cm by using a second correction instruction. This is also well illustrated by the top left lines in figure 6.5. For a single correction the average minimal distance is 2.39 cm while its values lies between 0.15 cm and 4.3 cm. With a second correction instruction the average minimal distance is reduced to 2.24 cm and its values varies between 0.40 cm and 3.70 cm.

The impact of the correction instructions on the precision is illustrated in figure 6.6. The automatic task space selection has determined after 10 demonstrations that both object positions are always more important than the hands position as shown in the top left graph (a). Here the precision of the relevant block's position is up to 96,8715 and for the irrelevant one up to 88,2248. As shown by the bottom left graph (b) the first correction instruction at time step 61 reduced the precision of all task spaces related to the position close to zero for eight time steps except the one of the relevant block. As illustrated by the bottom right graph (c) the second correction instruction at the 46th time step again reduces the precision of all position related task spaces except the one of the relevant building block.

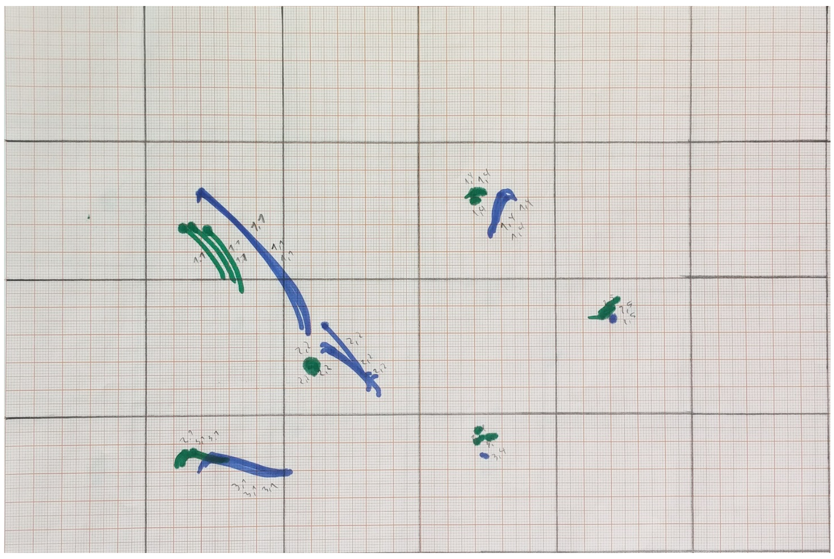

Figure 6.5: Reproduction of the learned skill with a training set of 10 demonstrations using the correction mode together with the automatic task selection. Demonstrated for 6 different positions of the relevant marker (from the left top at:{(1,1), (1,4), (2,2), (2,5), (3,1), (3,4)}) and a fixed position of a second one. The blue lines show the impact of a single and the green lines of two corrections.

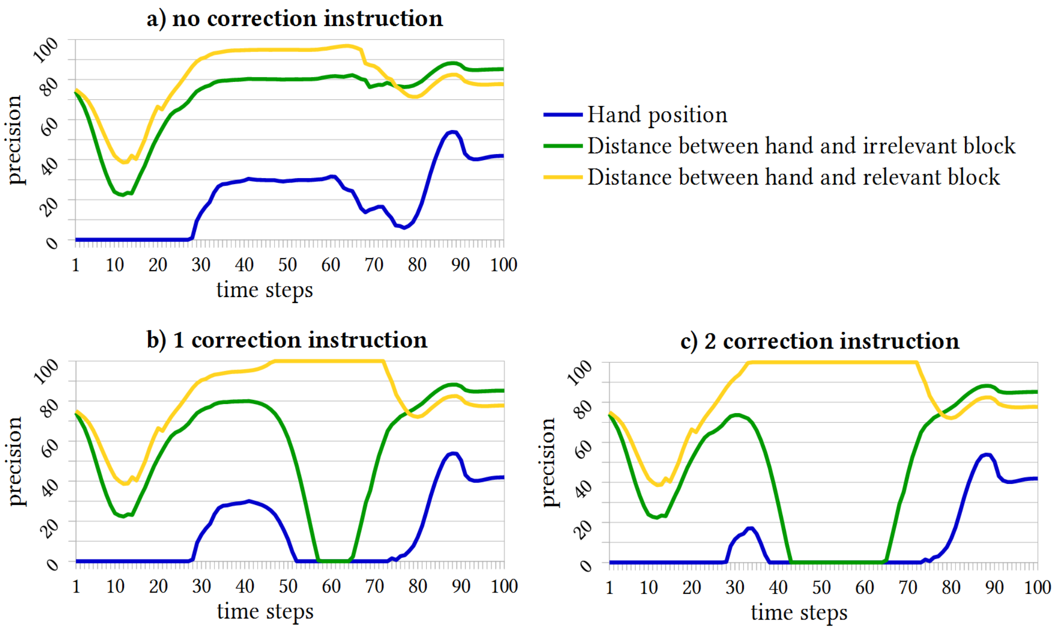

Figure 6.6: Precision of the task spaces that are related to the position with (a) only the automatic task space selection, (b) after applying one or (c) two correction instructions.

7 Discussion

The limitations of the automatic task space selection without the assumption that no concurrent task spaces are allowed are well observable in figure 6.3 and table 6.1. For only one demonstration the average minimal distance is 10.44 cm. In this case all task spaces are fully weighted since there is no derivations. With an increasing number of demonstrations and under the assumption that they are Gaussian distributed the standard derivation converges to its real value. Therefore the precisions also converge to fixed numbers. Under the given assumption the same is true for the mean weight and therefore also for the results.

This process is also illustrated in figure 6.4. Here the center of the lines are shifted down after the skill is trained with a second demonstration. As shown in figure 6.2 at the left this comes from the chronological order of the demonstration which leads to a shifted mean trajectory. Since the irrelevant building block is not moved it leads to a fixed attractor point. The same is true for the position of the hand. According to this the variations between the results for the different positions of the target building block have to be caused by the relative position of the relevant block. As this variation does not increase, the automatic task space selection has to weight the different task spaces wrongly as already shown for a training set of 10 demonstrations in figure 6.6 (a). All in all the better measurements of success and accuracy for an increasing number of demonstrations can only be traced back on the mean trajectory for the relative position of the irrelevant object that is shifted to the middle of the test settings.

In contrast to the automatic task space selection, the spoken instructions leads to a small minimal distances independent of the number of training trajectories. This is best illustrated in figure 6.4 at the right. Here the spoken task space selection was used to discard all task spaces that encode any position information except for the relative distance between the target object and the hand. Therefore only concurrent task spaces that are related to the orientation of the hand or an object were not discarded. Since no object has been rotated and the hand was always in a downwards pointing pose during all demonstrations the remaining ones have no negative effect. A probable reason that in most cases the minimal distance to the goals at (1,4) and (2,5) is the furthest is the fact that they were near to the limits of the robots joints.

The results of the correction instruction highlight its limitation. In figure 6.5 it is well illustrated that the fixed temporal window of the correction is to small for this task. The robots' hand is first moved to the drawing point that results from the automatic task space selection. Then it is moved down and while already drawing the precision of the relative position of the irrelevant building block is reduced. According to this the loss function of the operational space controller is influenced more by the desired goal of the relative position task space of the relevant block. Therefore the hand is moved towards that goal position which leads to drawing a line. By correcting the movement again the insufficient temporal window is enlarged as shown in figure 6.6 which results in shorter lines and lower minimal distances. Nevertheless especially the target goals at the left are never reached. The reason for that is not clear. The results show that for the correcting instruction additional information has to be used to determine in what temporal window the correction has to be applied.

All in all the experiments also showed some advantages and disadvantages of task spaces. On the one hand as long as all task spaces are given that are needed to fully encode a task it is possible that the skill is learned with a single demonstration. On the other hand this is only true if solely the relevant information is represented by them. So a skill can only be represented if the real minimal set of task spaces is found by computing the precision correctly.

7.1 Comparison with other approaches

In this work a skill is represented as a set of ProMPs, one for each task space. To learn the parameters of the ProMPs all trajectories are first time warped according to the first demonstration. After that the optimal weight vector for each trajectory is computed using linear regression. The parameters are then calculated as the mean and standard deviation which fulfills the maximum likelihood condition. The precision is used as a blending between the different task spaces. Since all task spaces are time warped according to the first demonstration this one has the most influence on the temporal behavior of the skill reproduction. For example if at the experimental setting discussed in section 6.1 the first point to draw would be next to the initial position, the robot would only need a short time to reach it. The drawing movement would be as usual. When the robot now has to reach a point far away this would lead to a very fast motion since the point has to be reached in the same time as before. Or if the maximum speed is limited the arm would maybe move downwards too early and therefore draw a line instead of a point. The phase variable introduced by Paraschos et al. [PDPN13] could be used to avoid this behavior. To do so the paths computed by the dynamic time warping algorithm has to be considered.

If using *dynamic movement primitives* [PHAS09] that describe a skill in form of partial derivative equations that is disturbed by a non linear function to model the movement would require to separate the movement in several parts. Then it would generalize better to new starting conditions. When the teacher describe the whole task with natural language as discussed in chapter 4 a series of description may be given as in the pick and place example it is divided into taking the object and placing it. But this separation is not sufficient since it has to be divided into four parts. In the presented experiment the teacher would describe the skill for example by "draw a point under the blue block". Since it is a single action no separation is provided by the teacher. In the training demonstrations the goal and start points are also approximately the same. If only the joint configuration is used the disturbing function would lead to draw the same point independent on the position of the relevant blocks. But also if only the relative position of the relevant block is used the result would be the same. This is caused by the fact, that the disturbing function is only time dependent. Since the start and goal state is set to the initial state which depends on the position of the relevant block and the disturbing function always leads to the same trajectory the movement of the hand is then every time the same. Even if several task spaces are described as DMPs and are blended like in the presented approach the desired skill cannot be trained since using only the relative position of the relevant building block is the optimal blending and this does not fulfill the task. Therefore even if this approach is used together with several task spaces as long as the motion is not separated into sufficient parts it does not represent the motion well. This separation can be made using natural language but it would be on the one hand not intuitive and on the other hand not obvious when using arbitrary many task spaces.

In the case that GMMs are used to define the underlying density function of the trajectories [CGB07, MGSG09, MGH^{+}09, ACYT12] the presented results can be applied directly in several cases. It is a similar approach that only encodes the information given by the demonstrated trajectories as GMMs instead of a Gaussian basis function model. Except for the work of Akgun et al. [ACYT12] the movement is recorded in several task spaces or for different features. To reproduce the skill also a precision matrix is used which is weighted differently as in this thesis.

Calinon et al. [CGB07] use the inverse of the covariance matrix to weight a task space or a feature as they call it. This weighting is only used to calculate the trajectory and is not used by the controller. Therefore the limitations for the weight as presented in this work is not needed. This has the advantage that the additional parameters as presented in this work are not needed. In the drawing experiment this weighting would have produced a much better result. As shown in figure 6.6 the precision of the relevant object is about 95 and 80 for the irrelevant object at the time steps where the point is drawn. According to the linear model the variance for the irrelevant object has to be four times as much as of the relevant one. In contrast to the presented model the mapping of Calinon et al. [CGB07] would lead to a precision for the irrelevant object that is only one forth of the precision of the relevant object. Nevertheless the goal would not be reached perfectly. This could also be improved by a spoken task space selection. The same is true for the correction instruction but it has to be adapted. Since the inverse can lead to arbitrary weightings a bias as used for the correction instruction is not suitable. Instead a factor or the replacement of the inverse by another function has to be used.

The *attraction-based criterion* introduced by Muhlig et al. [MGSG09] discards all task spaces that are not related to the object of attention. It is a very strong assumption since on the one hand only object related task spaces are not discarded and on the other hand it assumes that the teacher can imagine in all cases what task spaces are relevant. The introduced spoken task space selection avoid such strong assumptions but leads to the problem, that all task spaces are taken into account unless they are manually discarded. In the presented experiments only two objects are detected but if more are present this would cause problems. In a second step the *similarity criterion* is used only to select a set of task spaces which members are not concurrent. The remaining ones are fully weighted then. All in all the *attraction-based criterion* can be replaced by the spoken task space selection and the correction instruction has to be adapted to correct the chosen task spaces. In the presented experiments the strict assumption that only a set of independent task spaces has to be chosen would have lead to the best possible result as long as the relative position to the target object is chosen. Since the position of the hand is at the beginning and the end every time the same in contrast to the relative position of the object this cannot be guaranteed. For tasks like the pick and place example in chapter 4 where the target object changes the skill has to be divided in several parts in addition. Here, as discussed before, the spoken description would separate the skill into two parts which would be sufficient.

In the work of Akgun et al. [ACYT12] in addition to classical LfD methods, where trajectories are recorded, the skill can be learned from several sets of keyframes which are only single joint configurations. For both data acquisition methods, the parameters of a GMM is learned using an EM algorithm. The classical approach could be extended to task spaces like in the work of Calinon et al. [CGB07] or Muhlig et al. [MGSG09] and the presented instructions could be used as discussed before. In contrast to that, the keyframes methods are not easily adaptable to be used in different task spaces. Here the single data points are temporally mapped by assuming that the time between two keyframes are depending solely on its distance and a constant average velocity. The time information of the keyframes has to be mapped according to the longest needed time of all task spaces. As mentioned by Akgun et al. [ACYT12] it can be difficult for the user to identify all needed keyframes. As an example sometimes not all keyframes were provided in their experiments to avoid collisions. Since this happens even if only one task space like the joint configurations is recorded this problem will supposable get worse for an arbitrary number of task spaces. To handle this, their second introduced method, the *keyframe iterations*, would be preferable. Here the teacher can switch between the keyframes to add new ones or to modify or delete the current one.
Independent of the used keyframe technique, only the presented task space selection instruction can be used to extend the framework. For the automatic task space selection and the correction instruction the sparse set of data points would not provide enough data to compute a significant variance.

The approach presented in this thesis focus on low-level imitation learning. In contrast to that, the discussed approaches for navigation tasks are represented by system models or plans. All possible actions are predefined or created by a path planner. Speech supported learning from demonstration, as presented in this work would be adaptable as a method to learn new actions. Duvallet et al. [DKS13] and Tellex et al. [TKD+11] use SDCs to parse the natural language commands which are not limited to navigation instructions. For these systems the presented approach has to be extended such that single SDCs are taught.

7.2 Insights into learning from demonstration

The use of task spaces to extend the variety of information that is used to represent a skill affords a great opportunity on the one hand but on the other hand faces new challenges. For the same objects but new positions or orientations they generalize well. But it is not clear how they have to be defined if for example the related object is not recognized. In the presented approach when commanded to reproduce the skill the task space would just be ignored. It is also not easy to define all possible task space that are needed. For example an object could be grasped at several parts depending where it is flat enough or where the material is strong enough. Even the point of gravity could be important.

The choice of kinesthetic teaching was sufficient for this work. But even for drawing a point it is difficult to move the arm correctly. It is a very limited method since for example a simultaneously motion of both arm together with movement of the gripper cannot be realized by a single person. All in all other approaches are preferable.

7.3 Insights into speech recognition

The combination of a simple ROS publisher node and Nuance Dragon NaturallySpeaking Premium provides a great opportunity to control the work flow of any ROS node. Especially for applications like learning from demonstration if the users motion are recorded additional input sources are needed to avoid unfavorable assumptions. The possibility to create new commands in a few seconds which are also directly recognized with a high rate is great. One disadvantage is that the predefined text is not pasted as a whole. Instead each character is printed one after another probably to create the impression of classical typing since the software is mostly used for dictation. Another disadvantage is that there is no information available about the time point the user begins to say the command. The time the software needs to recognize the command also takes up to three seconds and varies strongly. According to this the premium version is not sufficient for time critical applications. Especially the teleoperation mode suffers from the needed recognition time since the PR2 can cover a long distance until the "stop" command is recognized. The same is true for the correction instruction where the needed time would not be sufficient for correcting faster movements. Here the information about the time when the user started to say the command would be sufficient since the robot could be driven back to that state.

7.4 Conclusion

In this thesis a learning from demonstration system based on probabilistic movement primitives has been introduced that is fully controlled by natural language. Therefore intuitive controlling instructions are introduced such as adding additional demonstrations, letting the robot reproduce the skill based on the current learning progress or correcting the reproduction. The latter can improve the accuracy but depends strongly on the provided correction and the setting. Another instruction can be used to manipulate the importance weight of a task space. By discarding and selecting task spaces the accuracy is improved rapidly. In the optimal case skills can then be learned with a single demonstration. The results demonstrates the benefit of such instructions but are limited to predefined connections between the spoken description and task spaces. Learning this relation for new instances will be an important field of research for future works.

8 Summary

In this thesis first an overview over learning from demonstration was given together with examples for the different classes of methods. The use of task spaces and how an subset of a given pool of task spaces can be found was investigated in more detail since they are more correlated to natural language as the robots joint configuration or sensor data. After that an overview over the use of natural language in robotics was given that is not limited to LfD but that showed that in this field of research natural language is rarely used.

After presenting the methods used in this thesis it was reasoned that in a LfD setting speech is normally used to describe the series of actions as already utilized for navigation tasks in the work of Duvallet et al. [DKS13] and Tellex et al. [TKD$^+$11]. In addition information is often provided about what is important to solve the task. To address this, the spoken task space selection was introduced that can select or discard task spaces that are related to the natural language description. It was realized as an extension of the automatic task space selection similar to the work of Muhlig et al. [MGSG09]. After that it was also reasoned that during the learning process, the learner is normally advised to demonstrate the skill to verify the learning progress. If it is not satisfying additional training demonstrations are provided or the learner is corrected. The first option was covered by the interactive design of the learning process and the latter by introducing a correction instruction. Here the teacher can say "stop" and then move the robots arm to the correct position to show the corrected state for that time point.

After that the details about the implementation and the robot platform were presented. The PR2 was used together with Nuance Dragon NaturallySpeaking for the speech recognition and ALVAR for the object recognition. The skill was represented by a ProMP and a precision trajectory for each task space. To train the ProMP, the recorded trajectories were first temporally aligned using dynamic time warping. After that the optimal weights of the Gaussian basis function model were calculated for each trajectory by using linear regression for non-linear features. At last the mean and covariance of the weights were calculated. To reproduce the skill the basis function model of each task space was first conditioned on the current state and according to the updated model the desired trajectory was calculated. Together with the precision an already existing operational space controller was used to follow the desired trajectory.

The introduced instructions and the automatic task space selection were tested on a drawing experiment where the robot was trained with up to ten demonstrations. It was shown that the automatic task space selection was not sufficient but that the spoken task space selection could be used to learn the task with a single demonstration. The correction instruction improved the results but not satisfactory.

At last a comparison to the related work was drawn. It showed that the presented instructions could be used directly or adapted in most other LfD methods. For high level representation the presented system could be used as a method to learn new actions.

Outlook

The presented instructions has to be developed further in future works. The spoken task space selection improved the results most but are limited to predefined commands and relations. Accordingly it would be promising to investigate how the selection of task spaces by unknown description can be learned. Since the correction instruction does not lead to a sufficient result it has to be examined how it can be improved. The variance could be used to decide if the difference between the wrong reproduction and the corrected state is far enough to make a decision. It has also to be investigated how the temporal smoothing of the correction could be improved.

A promising approach in further works would be the generalization of task spaces to new instances of objects. In the presented approach the skill is generalized by the use of task spaces to new positions of the relevant object. But it cannot be used for new instances of an object. As an example it is not possible to tell the robot to draw a point under the blue building block in the discussed experiments. The use of SDCs or similar approaches could address this problem. Instead of learning in task spaces that are related to specific objects they should be related to the different parts of the SDC. When saying the same action's name but another object's name the skill would then be applied to a new instance. In the presented work the connection between the object's name and the grounded object is hard coded. Therefore a grounding of the spoken object description parsed into the SDC and the object in the environment is needed. This could be done similar to the work of Matuszek et al. [MFZ+12]. A problem is that there are many types of task spaces that would not generalize to different instances directly. Therefore an important question will be what types of task spaces have to be added to the pool of possible spaces and how they can be adapted to new instances. In addition unsupervised learning methods could be investigated that learns to adapt an initial knowledge about a skill to new instances of objects.

A Experimental results

Minimal distance [cm]

N	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
(1,1)	15.05	13.90	12.40	8.85	8.70	9.00	10.50	8.30	9.00	9.10
(1,4)	4.60	5.30	3.85	2.75	2.50	2.30	3.60	2.15	3.00	3.20
(2,2)	12.20	10.20	8.40	5.70	5.05	5.70	6.60	4.90	5.65	5.00
(2,5)	5.10	1.4	0.15	2.15	2.10	2.25	1.10	2.65	1.90	2.65
(3,1)	16.70	13.90	11.95	8.50	8.05	8.90	9.50	7.70	8.35	8.15
(3,4)	9.00	4.90	3.65	1.40	2.25	2.70	1.85	1.85	1.85	1.55
$\oslash$	10.44	8.27	6.73	4.89	4.78	5.14	5.53	4.59	4.96	4.94
σ	5.06	5.19	4.97	3.28	3.00	3.22	3.96	2.86	3.20	3.08

Maximal distance [cm]

N	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
(1,1)	16.10	14.50	12.90	9.55	9.35	10.10	10.95	8.85	9.45	10.05
(1,4)	6.00	6.00	4.10	4.50	3.50	3.20	4.50	3.10	3.40	3.70
(2,2)	13.30	11.25	9.70	6.25	5.80	6.30	7.00	5.40	6.20	5.40
(2,5)	5.95	2.5	0.80	2.65	3.05	2.85	2.15	3.10	2.65	3.25
(3,1)	18.00	14.40	13.30	9.15	9.05	9.60	10.15	8.35	8.80	8.65
(3,4)	10.20	5.70	4.20	1.90	3.25	3.40	2.40	2.70	3.05	2.20
$\oslash$	11.59	9.06	7.50	5.67	5.67	5.91	6.19	5.25	5.59	5.54
σ	5.09	5.03	5.20	3.23	2.91	3.30	3.81	2.77	3.02	3.16

Length [cm]

N	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
(1,1)	1.10	0.80	0.65	0.90	0.70	1.30	0.70	0.60	0.40	1.15
(1,4)	1.70	0.70	0.85	1.90	1.10	1.05	1.25	0.95	0.50	1.10
(2,2)	1.20	1.10	1.30	0.55	0.80	0.70	0.55	0.75	0.55	0.45
(2,5)	1.45	1.5	0.75	0.65	1.50	0.75	1.05	0.50	0.80	1.00
(3,1)	1.30	0.70	1.35	0.70	1.15	0.70	0.75	1.05	0.50	1.15
(3,4)	1.40	0.90	1.20	0.50	1.20	0.70	0.75	0.85	1.35	1.00
$\oslash$	1.36	0.95	1.02	0.87	1.08	0.87	0.84	0.78	0.68	0.98
σ	0.21	0.31	0.30	0.53	0.29	0.25	0.26	0.21	0.35	0.27

Table A.1: Results of the automatic task space selection. (a,b) defines the position of the target block from the left and the top. $\oslash$ is the average and σ the standard derivation.

Minimal distance [cm]

N	1	2	3	4	5	6	7	8	9	10
(1,1)	0.20	0.00	0.30	0.15	0.40	0.80	1.20	0.95	0.85	0.30
(1,4)	0.75	0.95	1.10	1.15	0.50	0.70	0.40	0.70	0.60	0.50
(2,2)	0.55	0.70	0.25	0.00	0.40	0.50	0.60	0.80	0.70	0.10
(2,5)	1.25	2	2.15	1.60	1.60	1.90	0.80	1.30	1.55	1.60
(3,1)	0.05	0.10	0.15	0.35	0.80	0.80	1.50	0.70	0.95	0.30
(3,4)	1.45	1.65	1.95	1.10	1.15	1.30	0.30	0.60	0.55	0.55
⊘	0.71	0.90	0.98	0.73	0.81	1.00	0.80	0.84	0.87	0.56
σ	0.56	0.81	0.90	0.65	0.48	0.51	0.47	0.25	0.37	0.54

Maximal distance [cm]

N	1	2	3	4	5	6	7	8	9	10
(1,1)	0.75	0.80	0.80	0.70	1.10	1.30	1.75	1.70	1.60	1.15
(1,4)	1.75	1.80	1.70	2.10	1.85	1.15	1.10	1.40	1.20	1.10
(2,2)	1.25	1.25	1.15	0.45	1.30	0.90	1.30	1.50	1.50	0.45
(2,5)	1.95	2.85	2.95	3.10	2.55	2.80	1.90	2.45	2.20	2.50
(3,1)	0.65	0.85	1.10	1.50	2.00	1.30	2.40	1.65	1.60	1.00
(3,4)	1.90	2.10	2.50	1.55	1.60	2.70	1.30	1.15	1.35	1.20
⊘	1.38	1.61	1.70	1.57	1.73	1.69	1.63	1.64	1.58	1.23
σ	0.58	0.80	0.86	0.96	0.52	0.83	0.49	0.44	0.34	0.68

Length [cm]

N	1	2	3	4	5	6	7	8	9	10
(1,1)	1.10	1.15	0.60	1.10	1.60	0.50	0.65	0.80	0.80	0.85
(1,4)	1.10	0.95	0.80	1.00	2.60	1.00	0.70	0.70	0.85	0.70
(2,2)	0.95	0.90	1.05	0.75	0.90	0.90	0.80	0.75	1.15	0.80
(2,5)	0.70	0.85	1.00	1.75	1.00	0.90	1.30	1.20	0.65	1.10
(3,1)	0.80	1.60	1.10	1.15	1.25	0.55	1.00	1.00	1.10	0.70
(3,4)	0.80	1.10	0.70	0.80	0.70	1.55	1.30	0.60	0.85	0.65
⊘	0.91	1.09	0.88	1.09	1.34	0.90	0.96	0.84	0.90	0.80
σ	0.17	0.27	0.20	0.36	0.69	0.38	0.29	0.22	0.19	0.16

Table A.2: Results of the spoken task space selection. (a,b) defines the number of the column from the left and the row from above where the target block was positioned. ⊘ is the average value and σ the standard derivation.

Bibliography

[ACVB09] B. D. Argall, S. Chernova, M. Veloso, B. Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009. (Cited on page 12)

[ACYT12] B. Akgun, M. Cakmak, J. W. Yoo, A. L. Thomaz. Trajectories and Keyframes for Kinesthetic Teaching: A Human-robot Interaction Perspective. In *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction*, pp. 391–398. 2012. (Cited on pages 15, 18, 27, 28, 49 and 50)

[AN04] P. Abbeel, A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the twenty-first international conference on Machine learning*, p. 1. ACM, 2004. (Cited on pages 13 and 15)

[CCT10] M. Cakmak, C. Chao, A. L. Thomaz. Designing Interactions for Robot Active Learners. *IEEE Transactions on Autonomous Mental Development*, 2(2):108–118, 2010. (Cited on pages 15, 17, 27 and 33)

[CGB07] S. Calinon, F. Guenter, A. Billard. On Learning, Representing, and Generalizing a Task in a Humanoid Robot. *IEEE Transactions on Systems, Man and Cybernetics, Part B*, 37(2):286–298, 2007. (Cited on pages 14, 15, 19, 38, 49 and 50)

[CIT12] L. C. Cobo, C. L. Isbell, Jr., A. L. Thomaz. Automatic Task Decomposition and State Abstraction from Demonstration. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, volume 1, pp. 483–490. 2012. (Cited on page 11)

[CT10] M. Cakmak, A. Thomaz. Optimality of human teachers for robot learners. In *IEEE 9th International Conference on Development and Learning*, pp. 64–69. 2010. (Cited on pages 15 and 17)

[DKS13] F. Duvallet, T. Kollar, A. T. Stentz. Imitation Learning for Natural Language Direction Following through Unknown Environments. In *IEEE International Conference on Robotics and Automation*. 2013. (Cited on pages 15, 16, 27, 51 and 53)

[HYNK04] Z. M. Hanafiah, C. Yamazaki, A. Nakamura, Y. Kuno. Human-robot Speech Interface Understanding Inexplicit Utterances Using Vision. In *Extended Abstracts on Human Factors in Computing Systems*, pp. 1321–1324. 2004. (Cited on pages 15, 17 and 36)

[JT11] N. Jetchev, M. Toussaint. Task space retrieval using inverse feedback control. In *Proceedings of the 28th International Conference on Machine Learning*, pp. 449–456. 2011.

[KKGB11] G. Konidaris, S. Kuindersma, R. Grupen, A. Barto. Robot learning from demonstration by constructing skill trees. *The International Journal of Robotics Research*, 2011. (Cited on pages 11 and 13)

[KZB11] S. Khansari-Zadeh, A. Billard. Learning Stable Nonlinear Dynamical Systems With Gaussian Mixture Models. *IEEE Transactions on Robotics*, 27(5):943–957, 2011. (Cited on pages 14 and 15)

[LBKK02] S. Lauria, G. Bugmann, T. Kyriacou, E. Klein. Mobile robot programming using natural language. *Robotics and Autonomous Systems*, 38(3–4):171–181, 2002. (Cited on pages 12, 15 and 16)

[MFZ$^+$12] C. Matuszek, N. FitzGerald, L. Zettlemoyer, L. Bo, D. Fox. A Joint Model of Language and Perception for Grounded Attribute Learning. In *Proceedings of the 29th International Conference on Machine Learning*, pp. 1671–1678. 2012. (Cited on pages 15, 17, 33, 36 and 54)

[MGH$^+$09] M. Muhlig, M. Gienger, S. Hellbach, J. Steil, C. Goerick. Task-level imitation learning using variance-based movement optimization. In *IEEE International Conference on Robotics and Automation*, pp. 1177–1184. 2009. (Cited on pages 14, 15, 28 and 49)

[MGSG09] M. Muhlig, M. Gienger, J. Steil, C. Goerick. Automatic selection of task spaces for imitation learning. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4996–5002. 2009. (Cited on pages 9, 10, 14, 15, 19, 28, 29, 49, 50 and 53)

[Nua14] Nuance Communications, Inc. *Nuance® Dragon® NaturallySpeaking 13 Premium - datasheet*, 2014. URL http://www.nuance.com/ucmprod/groups/corporate/@web-enus/documents/collateral/dns13premiumdatasheet.pdf. Data sheet. (Cited on page 34)

[PDPN13] A. Paraschos, C. Daniel, J. Peters, G. Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems*, volume 26, pp. 2616–2624. 2013. (Cited on pages 14, 19, 21, 38 and 48)

[PHAS09] P. Pastor, H. Hoffmann, T. Asfour, S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *IEEE International Conference on Robotics and Automation*, pp. 763–768. 2009. (Cited on pages 11, 13, 14, 15 and 48)

[RYSV07] P. Rybski, K. Yoon, J. Stolarz, M. Veloso. Interactive Robot Task Training through Dialog and Demonstration. In *2nd ACM/IEEE International Conference on Human-Robot Interaction*, pp. 49–56. 2007. (Cited on pages 12, 15 and 17)

[SB98] R. S. Sutton, A. G. Barto. *Reinforcement learning: An introduction*. MIT press, 1998. (Cited on page 13)

[TKD$^+$11] S. A. Tellex, T. F. Kollar, S. R. Dickerson, M. R. Walter, A. Banerjee, S. Teller, N. Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proceedings of the National Conference on Artificial Intelligence*. 2011. (Cited on pages 15, 16, 27, 33, 51 and 53)

[Tou13] M. Toussaint. Machine Learning, 2013. Lecture script. (Cited on page 19)

[Tou14] M. Toussaint. Introduction to Robotics, 2014. Lecture script. (Cited on page 19)

[WIC$^+$09] A. Weiss, J. Igelsböck, S. Calinon, A. Billard, M. Tscheligi. Teaching a humanoid: A user study on learning by demonstration with HOAP-3. In *The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pp. 147–152. 2009. (Cited on page 15)

[YNK03] M. Yoshizaki, A. Nakamura, Y. Kuno. Vision-speech system adapting to the user and environment for service robots. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pp. 1290–1295. 2003. (Cited on page 16)

All links were last followed on April 27, 2015.

Declaration

I hereby declare that the work presented in this thesis is entirely my own.
I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations.
Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before.
The electronic copy is consistent with all submitted copies.

place, date, signature