Local Online Motor Babbling: Learning Motor Abundance of A Musculoskeletal Robot Arm

Lokales Online-Motor Babbling: Lernen der Motorredundanzen eines muskuloskeletalen Roboterarmes

Master-Thesis von Zinan Liu aus Henan

Tag der Einreichung:

1. Gutachten: Prof. Dr. Jan Peters, Prof. Dr. Koh Hosoda
2. Gutachten: Prof. Dr. Heinz Koeppl
3. Gutachten: Asst. Prof. Dr. Shuhei Ikemoto, Svenja Stark

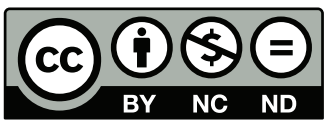
TECHNISCHE UNIVERSITÄT DARMSTADT

Local Online Motor Babbling: Learning Motor Abundance of A Musculoskeletal Robot Arm
Lokales Online-Motor Babbling: Lernen der Motorredundanzen eines muskuloskeletalen Roboterarmes

Vorgelegte Master-Thesis von Zinan Liu aus Henan

1. Gutachten: Prof. Dr. Jan Peters, Prof. Dr. Koh Hosoda
2. Gutachten: Prof. Dr. Heinz Koeppl
3. Gutachten: Asst. Prof. Dr. Shuhei Ikemoto, Svenja Stark

Tag der Einreichung:



For Anqi

Erklärung zur Master-Thesis

Hiermit versichere ich, die vorliegende Master-Thesis ohne Hilfe Dritter nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 25. April 2019

_____

(Zinan Liu)

Abstract

Sensorimotor learning of continuum or soft robots has been a challenging problem due to the highly redundant, non-linear system with hysteresis. Recent works in goal babbling have demonstrated successful learning of inverse kinematics (IK) on such systems and suggest that babbling in the goal space better resolves motor redundancy by learning as few sensorimotor mapping as possible. However, for the musculoskeletal robot, which is a hard-bodied system with soft actuation, motor redundancy can be of useful information to explain muscle activation patterns, thus the term motor abundance. This thesis aims to learn the IK and motor abundance of a 10 degree-of-freedom (DoF) bio-inspired upper limb robot actuated by 24 pneumatic artificial muscles (PAMs), which is a highly redundant and over-actuated musculoskeletal system with an unknown task space. Firstly some simple heuristics are introduced to empirically estimate the unknown task space, so as to facilitate IK learning using directed online goal babbling. The results show that the learned IK is able to achieve 1.8 cm average accuracy given the best possible average is 1.2 cm. We then further propose local online motor babbling using the Covariance Matrix Adaptation Evolution Strategy (CMA-ES), which bootstraps on the collected samples in goal babbling for initialization, such that motor abundance can be queried for any static goal within the defined task space. The result shows that our motor babbling approach can efficiently explore motor abundance, and gives useful insights in terms of muscle stiffness and synergy.

Zusammenfassung

Motor Babbling und Goal Babbling sind bereits für sensorimotorisches Lernen auf hochgradig redundanten Systemen weicher Roboter benutzt worden. Neue Arbeiten auf dem Gebiet des Goal Babblings haben das erfolgreiche Lernen der inversen Kinematic solcher Systeme demonstriert und deuten darauf hin, dass Exploration im Zielraum Motorredundanzen durch das Lernen von so wenig wie möglich nötigen Sensorimotorzuordnungen besser auflösen kann. Für muskuskeletale Robotersysteme können die Motorredundanzen nützliche Informationen zur Erklärung von Muskelaktivierungsmustern liefern. In der vorliegenden Thesis werden die inverse Kinematik und die Motorredundanzen eines hochgradig redundanten und übersteuerten Muskuskeletalsystems mit unbekanntem Zielraum gelernt. Zunächst wird eine Heuristik vorgestellt mithilfe derer der unbekannte Zielraum geschätzt werden kann um das Lernen der inversen Kinematik unter Benutzung von gerichtetem Online-Goal Babbling zu ermöglichen. Die Ergebnisse zeigen, dass die gelernte inverse Kinematik in der Lage ist, durchschnittlich 1,8cm Genauigkeit zu erzielen. Weiterhin stellen wir lokales Online-Motor Babbling unter der Benutzung der Covariance Matrix Adaptation Evolution Strategy (CMA-ES) vor, welches sich mittels Bootstrapping von den beim Goal Babbling gelernten Beispielen initialisiert, sodass die Motorredundanz für jedes statische Ziel im definierten Zielraum abgefragt werden kann. Die Ergebnisse zeigen dass unser Motor Babbling Ansatz effectiv die Motorredundanz explorieren kann und nützliche Einsichten in Hinblick auf Muskelsteifheit und Synergien liefert.

Acknowledgments

I would like to express my appreciation and gratitude to Prof. Jan Peters and Prof. Koh Hosoda, who made this joint thesis project possible for me, to conduct research and experiments at *Adaptive Robotics Laboratory* in Osaka University, Japan. I would also like to thank Prof. Heinz Koeppl, who let me explored my interests in robotics as a HiWi before I start my thesis, and who has also made this multidisciplinary thesis possible in *Fachbereich Elektrotechnik und Informationstechnik*. In addition, my courtesy to *Department International Affairs* at TU Darmstadt and DAAD for their financial support. During my thesis work, Prof. Ikemoto has been a great mentor in leading me onto the right track, Arne Hitzmann has been of indispensable help to me in programming, software, and setting up of the robot, Svenja Stark has always kept my work checked and organized thanks to her midnight counseling and proofreading, and last but not least, Hiroaki Masuda, who has helped me to mechanically take care of the high-maintenance robot. My sincere thanks go to all of you, and this thesis would not have been possible without you.

Contents

Figures and Tables

List of Figures

List of Tables

1 Introduction

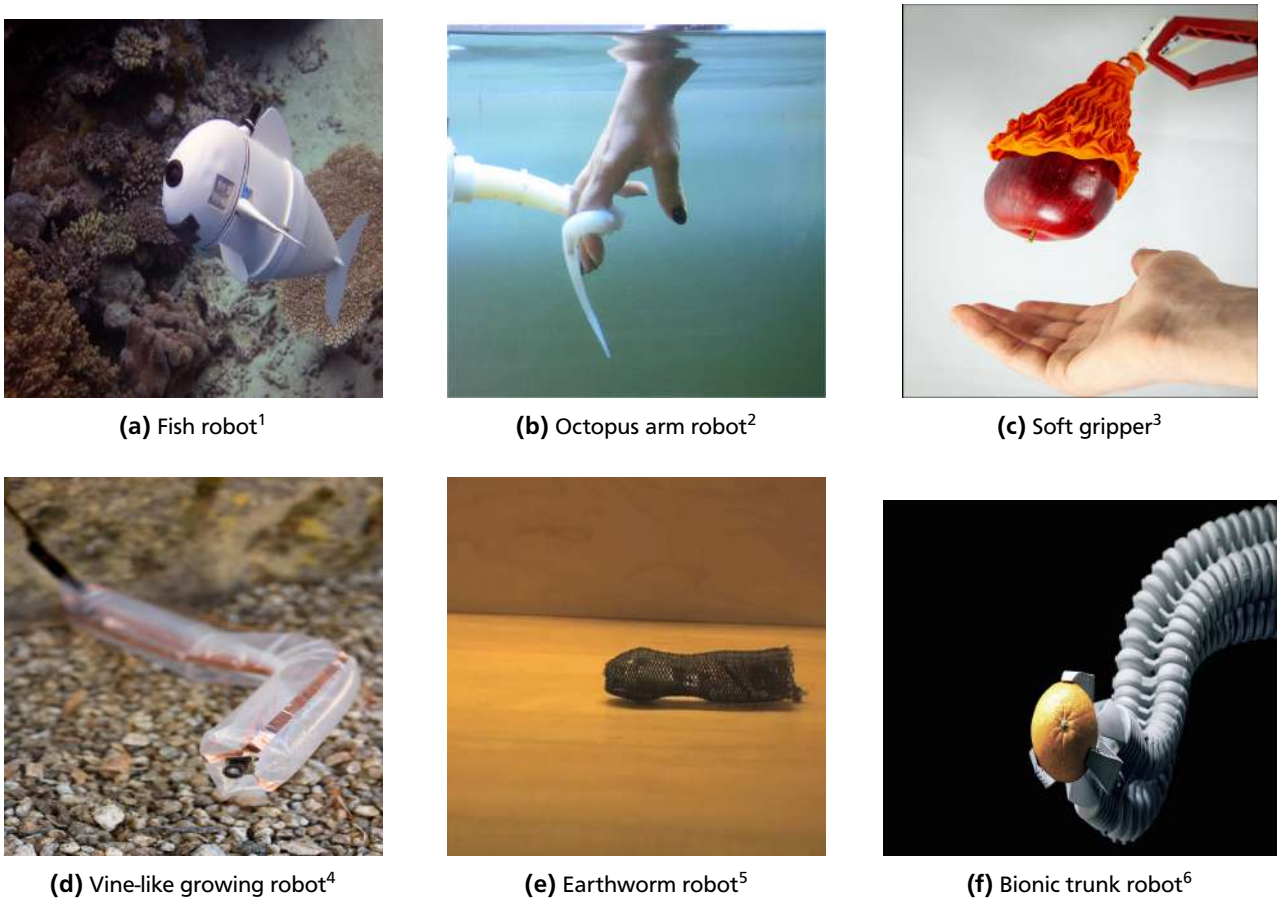
1.1 Motivation

Soft robotics has attracted great attention in the past decade, due to the enhanced flexibility and adaptability compared to hard-bodied robots with rigid body links. Soft robots made with highly compliant materials allow the possibility of working in unstructured and congested environments, as well as improved safety when working around humans [1, 2]. On the other hand, the most commonly used industrial robots are hard-bodied robots, which are usually fixed in well-defined environments such that they can perform a prescribed motion repeatedly with great precision. These hard-bodied robots are designed to be stiff to counter the vibration and deformation of the structure and to maintain the accuracy of the movement. The joints are usually flexible only in one rotary or translational direction to provide one degree of freedom (DoF), and all the possible motion combinations of all DoFs and their attached rigid links define the complete task space [2].

From another perspective, soft robots use distributed deformation with theoretically an infinite number of DoFs, leading to a hyper-redundant configuration of the motor space, but also enabling great dexterity and adaptability, where they have little resistance to compressive forces and therefore can conform to objects and obstacles, making it possible to to carry soft and fragile payloads without causing damage, such as the soft gripper in Figure 1.1c [3]. The large strain deformation of the compliant materials also enables squeezing through openings smaller than the nominal dimensions of the robot, such as the vine-like growing robot in 1.1d, which is able to squeeze through and navigate in an unstructured environment and perform weight lifting thousand times of its own weight [4]. Recent advances in soft robotics have led to many bio-inspired designs that display the aforementioned characteristics, as illustrated in Figure 1.1.

Although Young's modulus model only defines the level of "softness" as homogeneous prismatic bars that are subject to axial loading and small deformations [5], it nevertheless offers a useful measure of the material rigidity used in soft robots. Materials typically used for hard-bodied robots have moduli in the order of $10^9 - 10^{12}$ pascals, whereas natural skin and muscle tissue usually have moduli in the order of $10^4 - 10^9$ Pa. Thus soft robots are usually defined as systems capable of autonomous behavior, and primarily composed of materials with moduli in the range of that of soft biological materials [1]. However, in this thesis work, we address soft robots either as a major soft body with compliant materials, or a hard robot body equipped with soft and compliant actuation. Soft robots are usually actuated in one of the two ways: variable length tendons or pneumatic actuators [1]. Variable length tendons are usually in the form of tension cables [6] or shape-memory alloy actuators [7] embedded in soft segments, such as the octopus arm in Figure 1.1b. Pneumatic actuation can be used to inflate channels in a soft material and lead to the desired deformation. Pneumatic artificial muscles (PAMs), or McKibben actuators are one way of pneumatic actuation where rubber tube is inflated inside a braided sleeve that guides the inflation [8]. Another way of pneumatic actuation is to use fluidic elastomer actuators (FEAs), a highly extensible, adaptable, and low-power soft actuator, which comprises synthetic elastomer films operated by the expansion of embedded channels under pressure [9–11]. The actuation of soft robots usually align the actuators in a muscle-like, biologically inspired agonist-antagonist arrangement that allows bidirectional actuation and co-contraction of muscle pairs for adaptable compliance [1]. Note that hard-bodied robots with hyper-redundant continuum structures or soft compliant actuation also has the potential to work in unstructured environments and provide high dexterity, much as the soft robots do. The bionic handling assistant inspired by the elephant trunk actually uses hard exterior material built with Additive Manufacturing (AM) technology and actuated with pneumatics to achieve the continuum deformation [12].

Aside from the compliant materials and actuation, compliant robots are also facilitated for sensing, actuation, computation, power storage, and communication, embedded in the compliant material. Algorithms that match the impedance of the soft body structure are also crucial to delivering the desired motion. This tight coupling blurs the line between the body and the brain, i.e., the body of the robot and the control unit since the control algorithms can be simplified by outsourcing to the morphology of the body. The mechanical adaptability and dexterity of the compliant materials and actuation can be viewed as embodied intelligence that augments the brain with morphological computation [1, 13, 14]. The behavior of the system does not solely come from the internal control structure but are also shaped by the interaction with the environment, and its own morphology, i.e., body shape, as well as the placing of sensors and effectors. These physical constraints shape the dynamics of interaction, which the body morphology, as well as the coupled sensory-motor activity, induce statistical regularities and therefore enhance the internal information processing [15].



(a) Fish robot[1] **(b)** Octopus arm robot[2] **(c)** Soft gripper[3]

(d) Vine-like growing robot[4] **(e)** Earthworm robot[5] **(f)** Bionic trunk robot[6]

Figure 1.1.: Various bio-inspired soft robots: **(a)** Soft-bodied fish robot self-contained with an array of fluidic elastomer actuators, capable of rapid, continuum body motion that emulates escape responses in addition to forward swimming [16]. **(b)** An octopus arm robot mimicking the muscular hydrostat structure by using Electro-Active Polymer (EAP) technology and emulating the antagonistic muscle contractions to provide varied stiffness and elongation other than bending in all directions [17]. **(c)** A light-weight soft gripper robot made of an origami "magic-ball" and a flexible membrane, driven by vacuum, is capable of lifting a large variety of objects from delicate foods to heavy bottles [3]. **(d)** A vine-line soft pneumatic robot that navigates through growth, where the pressurization of the inverted thin-walled vessel allows rapid asymmetric lengthening of the robot tip for locomotion and direction control [4]. **(e)** An earthworm robot equipped with nickel titanium coil actuators in a flexible braided mesh-tube, capable of generating sequential antagonistic motion that leads to peristaltic locomotion [18]. **(f)** The bionic handling assistant inspired by the elephant trunk, driven by pneumatic actuators in the continuum chambers made with Additive Manufacturing (AM) technology [12]. This is, however, a hard-bodied continuum robot yet with soft actuation, i.e., pneumatic actuators.

Recent researches in compliant robots has made evident of the embodied intelligence: [19] demonstrates multiple quadruped gaits by using linear readout weights to combine the sensor values from an actuated multi-joint spine embedded with force sensors, [20] exploits the dynamics of the soft octopus arm robot to approximate non-linear dynamical systems and embedded non-linear limit cycles, and [21] deploys the spring-loaded inverted pendulum model on a compliantly actuated that achieves various quadruped locomotion. However, these approaches rely heavily on the design and manufacturing of the robot, and the interplay with the environment. [22] has proposed a theoretical framework for morphological computation on compliantly embodied robots, which models the physical bodies as mass-spring systems implemented with complex nonlinear operators. By adding a simple static readout to the morphology, complex mappings

1 Taken from: http://news.mit.edu/2018/soft-robotic-fish-swims-alongside-real-ones-coral-reefs-0321
2 Taken from: http://www.octopus-project.eu/gallery.html
3 Taken from: https://techxplore.com/news/2019-03-robot-soft-strong.html
4 Taken from: https://news.stanford.edu/2017/07/19/stanford-researchers-develop-new-type-soft-growing-robot/
5 Taken from: https://newatlas.com/meshworm-robotic-earthworm/23674/
6 Taken from: https://www.festo.com/group/de/cms/10241.htm

of input to output streams can be emulated in continuous time. Nevertheless bridging the gap between artificial and natural systems still poses great theoretical and technological challenges.

A reliable long-term application of such continuum/soft robots is strongly dependent on the real-time kinematic and/or dynamic controllers for fast, accurate, and energy-efficient control, not only because of the uneasy manipulation compared to simple translation and rotations of hard-bodied robots, but also due to the highly redundant and nonlinear characteristics, e.g., compliance and hysteresis, that restrict high-frequency accurate control [23]. Even the research is still in its infancy, the survey [23] summarized the state-of-art control strategies for continuum/soft robotic manipulators, which are categorized as follows:

- **Model-based static controllers**: assuming steady-state under force equilibrium, the full configuration of the soft manipulator can be defined by a low-dimensional state space representation. The simplest and most common kinematics model assumes a 3D configuration space, and that the continuum/soft module can be parametrized by three variables, known as the constant curvature (CC) approximation [24]. However the CC method reduces an infinite dimensional structure to 3D, ignoring a large portion of the manipulator dynamics, and only suitable if the manipulator is uniform in shape and symmetric in actuation design, in addition to negligible external loading effect and minimal torsional effects [23].

- **Model-free static controllers**: machine learning and data-driven approaches for model-free control of continuum/soft manipulators is relatively new yet a field with great potential: [25] was the first to propose a feedforward neural network to learn the inverse kinematics (IK), which proves to be significantly better than the analytical method [26, 27]. Online goal babbling scales better to hyper-redundant systems and bootstraps fast on efficiently generate samples in the task space for IK mapping using self-organizing maps [22, 28]. A "model-less" technique is later developed to empirically estimate the kinematic Jacobian matrix online by incrementally moving each actuator, leading to a highly robust, accurate, and generic approach for closed-loop task space control of continuum robots [29]. Recent advances in reinforcement learning have been approaching the problem by learning deterministic stationary policies [30] and fuzzy logic-based controllers [31]. Other attempts in transfer learning [32] and differential IK [33, 34] has also been carried out, yet only in simulations so far. Model-free approaches circumvent the need for parameter definition of the configuration space, and treat the joint space and the manipulator space independently, making it a better choice for systems that are highly nonlinear and nonuniform [23, 26].

- **Model-based dynamic controllers**: model-based approaches in developing dynamic controllers would require the formulation of the kinematic model with an associated dynamic formulation, which is extremely challenging given that the kinematics are difficult to model for continuum/soft robots, to begin with, a dynamic formulation is subjected to the aggravation of model uncertainties [23]. This field is still in the nascent stage where most works are only in simulations. Based on the formulated kinematics using the CC model for a simulated 2D multisection robot, the dynamic model is represented in the Euler-Lagrangian form using lumped dynamic parameters, making [35] the first to demonstrate a closed-loop task space dynamic controller for continuum robots. A different approach then proposed the same kinematic and dynamic model, but with sliding mode controller [36], along with other variant attempts of [35, 36] in [23, 37–41].

- **Model-free dynamic controllers**: this field is relatively unexplored for continuum/soft robots. Early attempts first use machine learning techniques to compensate for dynamic uncertainties [42], however only for closed-loop dynamic control of the joint variables. In the field of reinforcement learning, reaching dynamics of a simulated multisegmented dynamical planar model of the octopus arm was demonstrated, and solved the hidden Markov model by using a nonparametric Gaussian temporal difference learning algorithm [43]. Recently, a forward dynamic model using a class of recurrent neural network and trajectory optimization first experimentally demonstrated direct actuator space to task space dynamic control on a 3D soft pneumatic manipulator [44]. Model-free approaches offer a simpler way to develop dynamic controllers, however, challenges still remain in bringing reinforcement learning onto real robots in practice, or in closing the open loop of high computational complexity [23].

Aside from hard-bodied continuum robots and soft robots, musculoskeletal robot systems also exhibit similar compliant and deformable characteristics. The movement coordination of these robots is achieved primarily through peripheral mechanical feedback loops and the biomechanical constraints provided by the musculoskeletal system. Effective exploration strategies can be exploited by the emerging behavior from the synergistic coupling of the system's morphology [22]. As suggested by the name, the musculoskeletal structure consists of hard-bodied skeleton actuated with soft or compliant muscles or/and tendons. Many types of musculoskeletal robots have been developed in recent years, to name a few a humanoid with flexible spined torso and whole-body muscle-tendon driven systems [45], a human-like robot arm with

Festo's fluidic muscle actuator [7] [46], the 114 DoF tendon-driven humanoid *Kengoro* actuated by the sensor-driver integrated muscle module [47]. Different from PAMs, these muscle modules consist of motors, motor drivers, tension sensors and thermal sensors, covered by sheet metals, enabling flexible tension control [48].

The human-inspired musculoskeletal system mimics the skeleton and muscle structures of a human. However, the human body is an over-actuated system, not only does it have a higher dimension in motor space than the degree of freedoms in the action space, i.e., more number of muscles than joints, it also has more degree of freedoms (DoFs) than necessary to achieve a certain motor task. How the effector redundant system of the human body adaptively coordinates movements remains a challenging problem. The most common approaches deal with compound action such as walking using a simple rule-based control system and outsource the complex dynamics to the morphology of the robot while interacting with the environment [24, 49, 50], such as achieving door opening task and human-like throwing motion generated by a musculoskeletal upper limb robot [51, 52]. However, the simple rule scheme is not enough for accurate control of general reaching tasks, nor for complex dynamic motions with torque control. And given that the musculoskeletal design is inherently non-uniform and asymmetric, typical model-based controllers do not apply, thus we investigate in applying model-free approaches on musculoskeletal systems, in particular on learning the inverse kinematics, and interpreting motor redundancy as variability and abundance.

1.2 Related Work

Various model-free approaches have been tried on the control and sensorimotor learning of musculoskeletal systems. [53] has demonstrated direct teaching on a musculoskeletal robot arm by controlling the internal pressures or the axial tensions of the PAMs under specific constraints, which forces the PAMs' lengths in the reproducing phase similar to that of the teaching demonstration. [54] extracted analytical linear models from neural networks and applied feedback control on a two-link musculoskeletal manipulator. [55] developed real-time inverse dynamics learning for musculoskeletal robots using goal babbling to effectively reduce the search space, a recurrent neural network, specifically the echo state network, to represent the state of the robot arm, and novel online Gaussian processes for regression. However these methods in direct teaching [53] and various learning schemes with artificial neuro networks [54, 55] are highly training dependent, which can be expensive to conduct experiments and inefficient to collect sample data, and do not generalize well to the complete task space for accurate control.

By extending to other domains such as sensorimotor learning in hard-bodied robots, or continuum robots, this problem is typically addressed by learning the forward kinematics via motor babbling, and explore the motor-sensory mapping from scratch [56–58] until eventually the robot can predict the effects of its own actions. However autonomous exploration without prior knowledge in motor babbling doesn't scale well to high dimensional sensorimotor space, due to the rather inefficient sampling of random motor commands in over-actuated systems. Inspired by infant development, i.e., reaching a goal by trying to reach. [59] and [60] introduced online goal babbling. This alternative suggests that learning inverse kinematics by goal babbling avoids the curse of dimensionality simply because the goal space is of much smaller dimension than the redundant motor space [59, 60]. [60] aims to solve the inverse kinematics by actively and autonomously generating goals, based on the progress measure in the region of interest. The progress measure is computed as the derivative of the summed reaching error in a certain time window of the region, which in turn also serves the criteria to recursively split the goal space using the k-d tree into further interest regions. In this way, unreachable regions or explored regions with high reaching accuracy will have low progress, such that goal generation can emphasize on the relatively unexplored yet reachable regions. Nonetheless, [60] assumes that the sensorimotor space can be entirely explored, which is not feasible in practice for high dimensional motor systems [28]. [59] then proposed to specify the goal space a priori as a grid, and sampling the goal grid points to guide exploration [61], such that sensorimotor mapping can be sufficiently generalized and bootstrapped for efficient online learning. It has also been quantitatively evaluated for an average of sub-centimeter reaching accuracy on an elephant trunk robot [28] with reasonable experiment time. We therefore pursuit in this direction of research by bringing goal babbling of hard-bodied continuum robot [28] to musculoskeletal robots [62].

Given the above works aiming to reduce motor redundancy for learning [28, 56–58, 60, 61], it can be argued that motor redundancy in human musculoskeletal systems is actually the keystone to natural movements with flexibility and adaptability, hence should be termed motor abundance rather than redundancy [63] [64]. It is also suggested that joint redundancy facilitates motor learning, whereas task space variability does not [65]. Thus based on the studies and implementation of goal babbling and IK learning on the 10 DoFs musculoskeletal robot arm, we further extend on interpreting motor redundancy as motor abundance, share insights on muscle stiffness and muscle synergies, which enables the generalized yet accurate IK control with variable muscle strengths and configurations, and lays the foundation for future research on model-free dynamic learning approaches on musculoskeletal robot systems.

[7] https://www.festo.com/cat/en_gb/data/doc_engb/PDF/EN/DMSP_EN.PDF

1.3 Overview

In this thesis, we work with a 10 DoF musculoskeletal upper limb robot actuated by 24 PAMs. We investigate on sensorimotor learning of this system using model-free approaches with machine learning algorithms, starting with learning the inverse kinematics using directed goal babbling [59], which reduces the motor redundancy and bootstraps online to efficiently explore and generate IK mapping samples [28,61]. However since the goal space of the robot arm is unknown and nonconvex [62], we empirically estimate the goal space with randomly generated postures, forcing the convex hull such that directed goal babbling can be applied, and subsequently remove the outlier goals in the goal space after IK learning for further online motor babbling to learn the queried motor abundance. On the other hand, the inherent redundancy of the musculoskeletal system gives rise to embodied intelligence [22] and natural movements with great adaptability and flexibility, it is suggested as motor abundance rather than motor redundancy [63, 64]. Particularly for musculoskeletal systems, motor abundance can be of useful information to explain muscle stiffness and muscle synergy. Therefore this thesis also further explores the motor abundance by local motor babbling, where online learning takes place with the initialization of the local sampled data from the learned IK, and use Covariance Matrix Adaptation-Evolution Strategies (CMAES) [66] to explore motor variability by fixing the end effector and the queried goal point. The intuition of using CMA-ES, is to intentionally set the starting point of the optimization away from the optimum, i.e., the motor commands that lead to the desired queried goal, which is the neighbouring goal from the learned IK, and conducts the CMA-ES trials multiple times to collect varied data series of different initialization and evolution paths, such that muscle stiffness and synergies can be reproduced for the queried goal.
This thesis is organized as follows: in Chapter 2 the musculoskeletal robot upper limb is introduced as the search platform. Chapter 3 reviews directed online goal babbling [28, 59, 61], the black box optimization method CMA-ES [66], and how to bootstrap the initialization on the local goal babbling data, and learn motor abundance via goal babbling using CMA-ES. Chapter 4 shows the experiments and evaluations of the learned IK and motor abundance. Chapter 5 then concludes the thesis with an outlook on future research potential with the musculoskeletal robot platform.

2 The Upper Limb Robot

2.1 Musculoskeletal Structure

The upper limb robot that we use for experiments in this thesis is of musculoskeletal systems design that aims to mimic the biological structure and motoric characteristics of a human. The upper limb consists of a shoulder linkage mechanism and an arm, similar to the human upper limb skeleton structure and muscle arrangement. The skeleton provides attachment points for the tendons and muscles, while the linkage structure imposes mechanical constraints onto the body's range of motion and defines the task space [62]. As illustrated in Figure. 2.1, the structural proportions and the composition of the robot demonstrate great resemblance to that of a human. Bones as the humerus, ulna carpals, metacarpals, and phalanges are adopted directly as well as their spatial relation, proportions, and mechanical properties. An earlier design of the robot shoulder mechanism imitates the human scapula and clavicle, which however tend to separate unless held in place by a coupling muscle. In order to ease the design and enhance mechanical durability, a series of sliding joints in combination with four ball joints are applied in the robot shoulder complex, while also allowing to replicate the range of motion of the human shoulder [67].

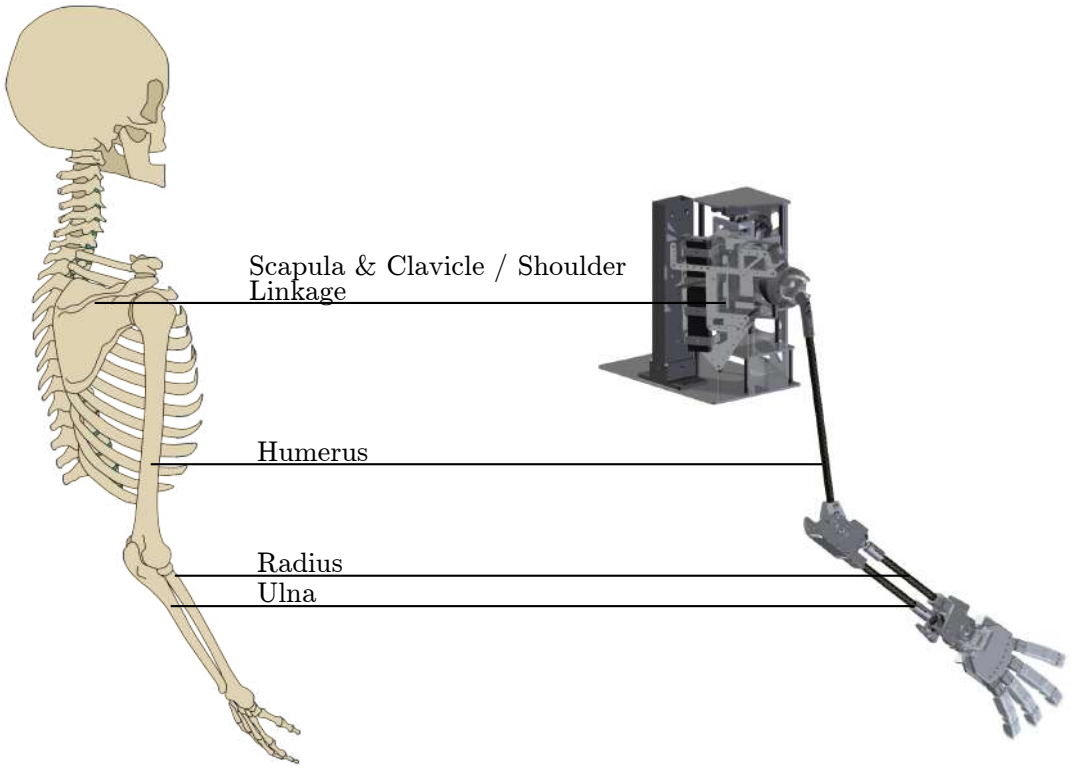


Figure 2.1.: Taken from [62], this figure shows the comparison of a human upper limb skeleton and the CAD rendering of the robot, where the structural resemblance between the robot skeleton design and the skeleton can be observed. The shoulder joint that consists of clavicle and scapula is replaced with a shoulder linkage mechanism [67] for mechanical integrity and durability. Movements generated in the imposed task space are also human-like as they adhere to the same underlying physical limitations.

To mimic the motoric characteristics and anthropomorphism of a human, the robot needs to be actuated with comparable flexibility and compliance. Pneumatic Artificial Muscles (PAMs) make a viable choice due to the lightweight yet strong, and inherently compliant characteristics. The most common PAMs are the McKibben muscles, or McKibben actuators, which are compliant linear soft actuators composed of elastomer tubes in braided fiber sleeves [8]. The actuation is

realized by controlling compressed air with proportional valves, allowing continuous contraction at varying speeds. As shown in Figure 2.2, the muscle consists of a rubber tube plugged at one end and connected to the air supply line on the other, covered by a braided sleeve to limit and guide the inflation of the rubber tube. The braided sleeve expands as the rubber tube expands, while the muscle bulges and the length decreases, where the sleeve's diameter is approximately inversely proportional to its length. The airline is monitored by a pressure sensor such that the muscle pressure can be controlled with a simple PID controller [62].

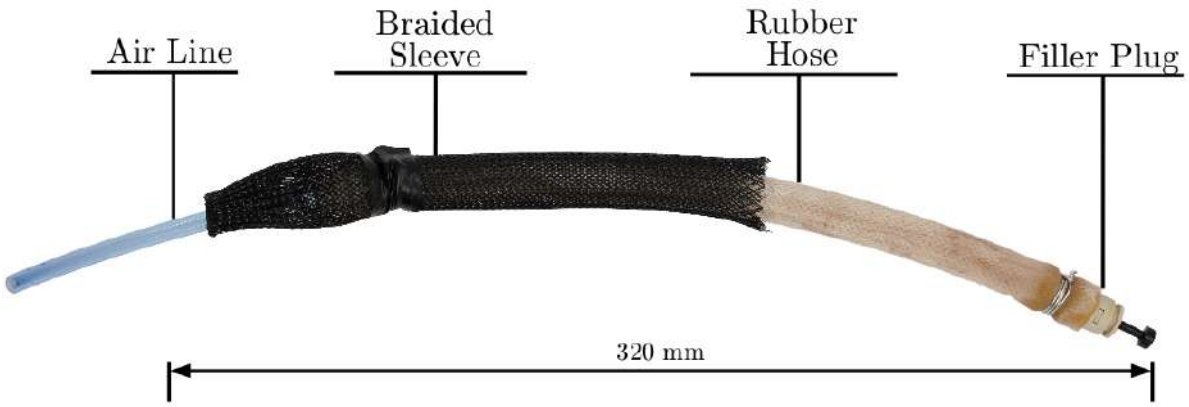


Figure 2.2.: Taken from [62], this figure illustrates the cutaway model of a pneumatic aritificial muscle (PAM) used in the presented upper limb robot

Assuming no external forces, the volume and tension of a single muscle can be analytically modeled as in [68]. However, the abrasion and friction between the rubber tubes and the braided sleeves, the bent routing of the muscles, and the antagonistic yet synergetic nature of muscle groups not only let the real values deviate away from the modeled predictions, but also wear and tear the muscles and reduce the life span of a PAM around 10,000 cycles [68]. The rubber tube ruptures when pressurized up to 0.87MPa, and due to the routing of the muscles and strappings with the skeleton, it is advised to keep the actuation range below 0.4MPa for a relatively smooth forward kinematics, which also prolongs the life cycles of the PAMs. Thanks to the simple McKibben design, the muscles can be easily produced and replaced when they're worn out [62].
In order to attach PAMs onto the skeleton, the correspondence between the muscles of the human upper limb and the robot's muscles is taken into consideration, as shown in Figure 2.3. When selecting which muscles to adopt on the robot, often times the sets of adjacent muscles are combined and approximated into a single muscle, whereas small single muscles are omitted. Thus a bundle of PAMs does not quite resemble a bundle of real muscles, lacking the fine yet intricate fibers, and small single muscles cannot be accurately emulated by the PAMs [62].

2.2 Firmware, Control, and Software

The robot operates on a custom made printed circuit board (PCB) as a single control system providing low-level control of the air valves and reading the pressure sensors, where digital-analog (DAC) and analog-digital (ADC) converters are implemented correspondingly for the air valve and pressure sensor control for each muscle. A custom driver is implemented to read the pressure sensor values and write the actuator commands of the valve at the rate of 500Hz in a realtime-enabled context. Therefore, the individual muscles are updated at the same rate. Non-realtime contexted communication is initiated by the driver, such that an interruption-free main control loop is guaranteed by orchestrating the two differently prioritized contexts when exchanging and accessing data. The PCB communicates to a single-board-computer (SBC), i.e., Raspberry Pi 3(B), where the Robot Operating System (ROS) is used to provide a well-documented communication framework for high-level control [62].
ROS compatibility is also integrated within the main driver for topic advertisement and subscription. The custom controller for each muscle is spawned and the configurations from the ROS parameter server are loaded, so as to update the muscle's PID controllers. This design pattern is based on the `ros_control` interface [69], which provides multi-threading capabilities in real-time along with the real-time context of the main driver. Instead of torque, velocity, and position control for typical rigid link robots offered by `ros_control`, a new controller class resembling the muscle actuator is implemented, which provides the control of the internal pressure of the individual muscles [62]. Note that the tendon organ inspired tension sensors in [62] are not used for experiments in this thesis, as the measurements are noisy due to the strapped muscles and mechanical constraints of the skeleton, such that goal babbling exploration in the task space would introduce even more inconsistencies and redundancies to learn the inverse kinematics.

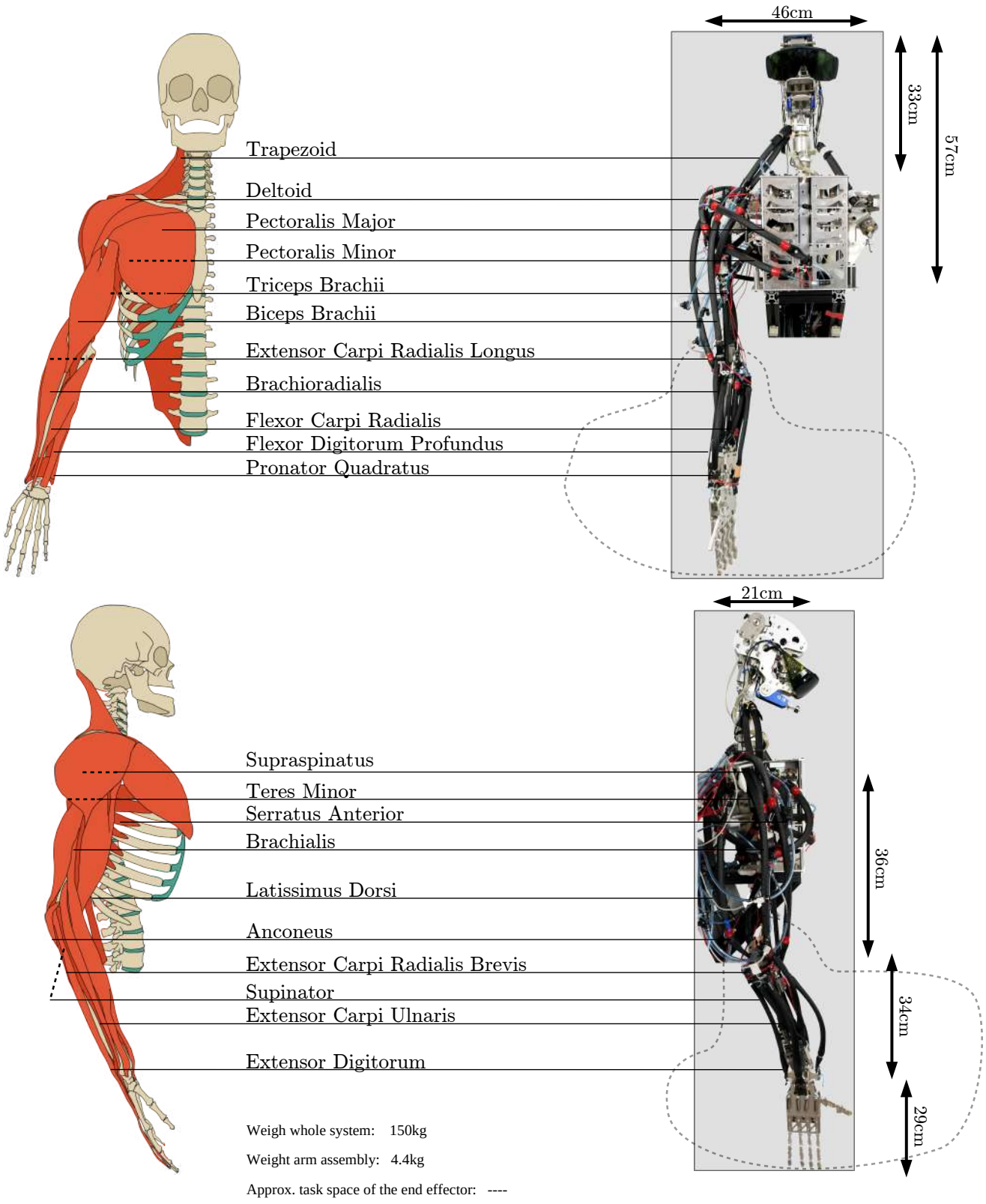


Figure 2.3.: Taken from [62], the above comparisons of human upper limb anatomy and the robot illustrates the musculoskeletal system design with identical dimensional relations to the human body parts, i.e., shoulder, chest, upper arm, fore arm, and the hand. The system displays similar morphology characteristics and sufficiently large task space compared to the human upper limb, where the major artificial muscles are adopted correspondingly from the human anatomy. This musculoskeletal robot arm is a highly redundant system consisting of 10 DoFs and 32 PAMs [62].

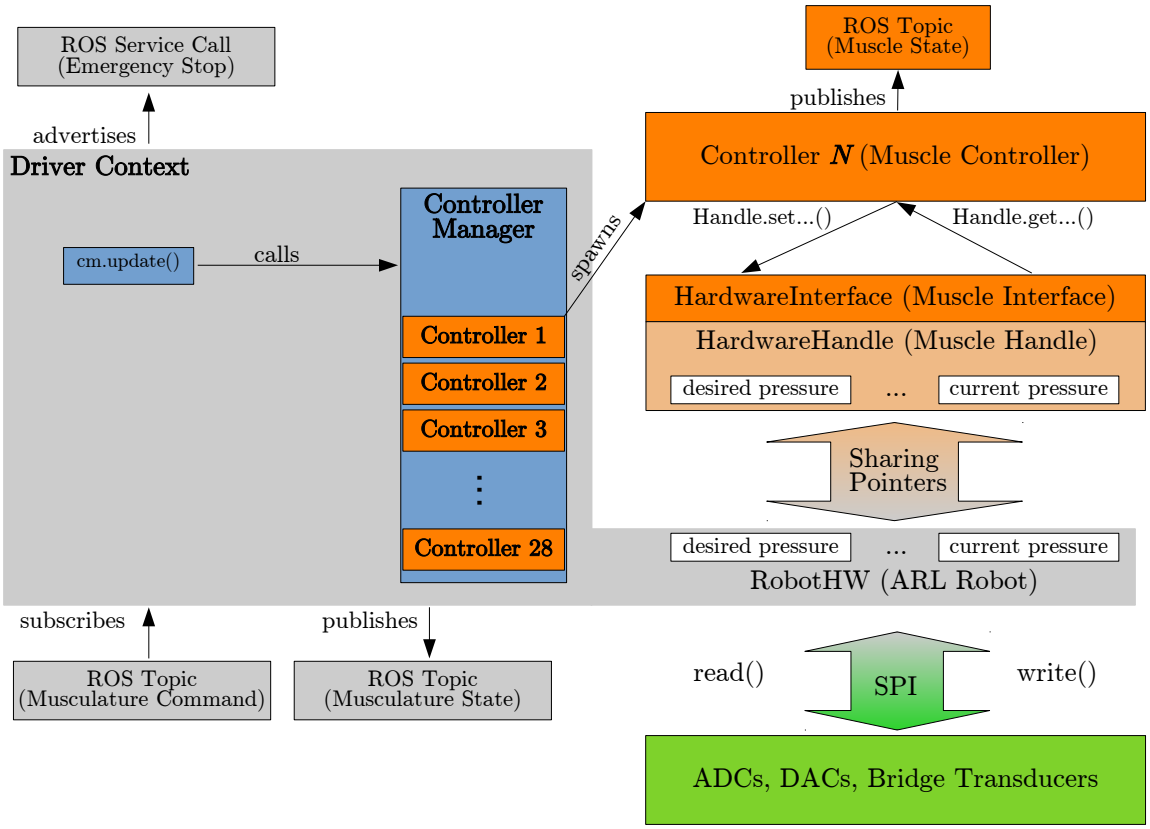


Figure 2.4.: Taken from [62], the figure illustrates the software stack of the robot, which consists of the main drive, which updates the data of the hardware controllers via SPI and stores them inside of the RobotHW interface, and the muscle controller threads spawned by the controller manager to calculate the individual control parameters for each muscle [62].

2.3 Detecting the End of A Movement

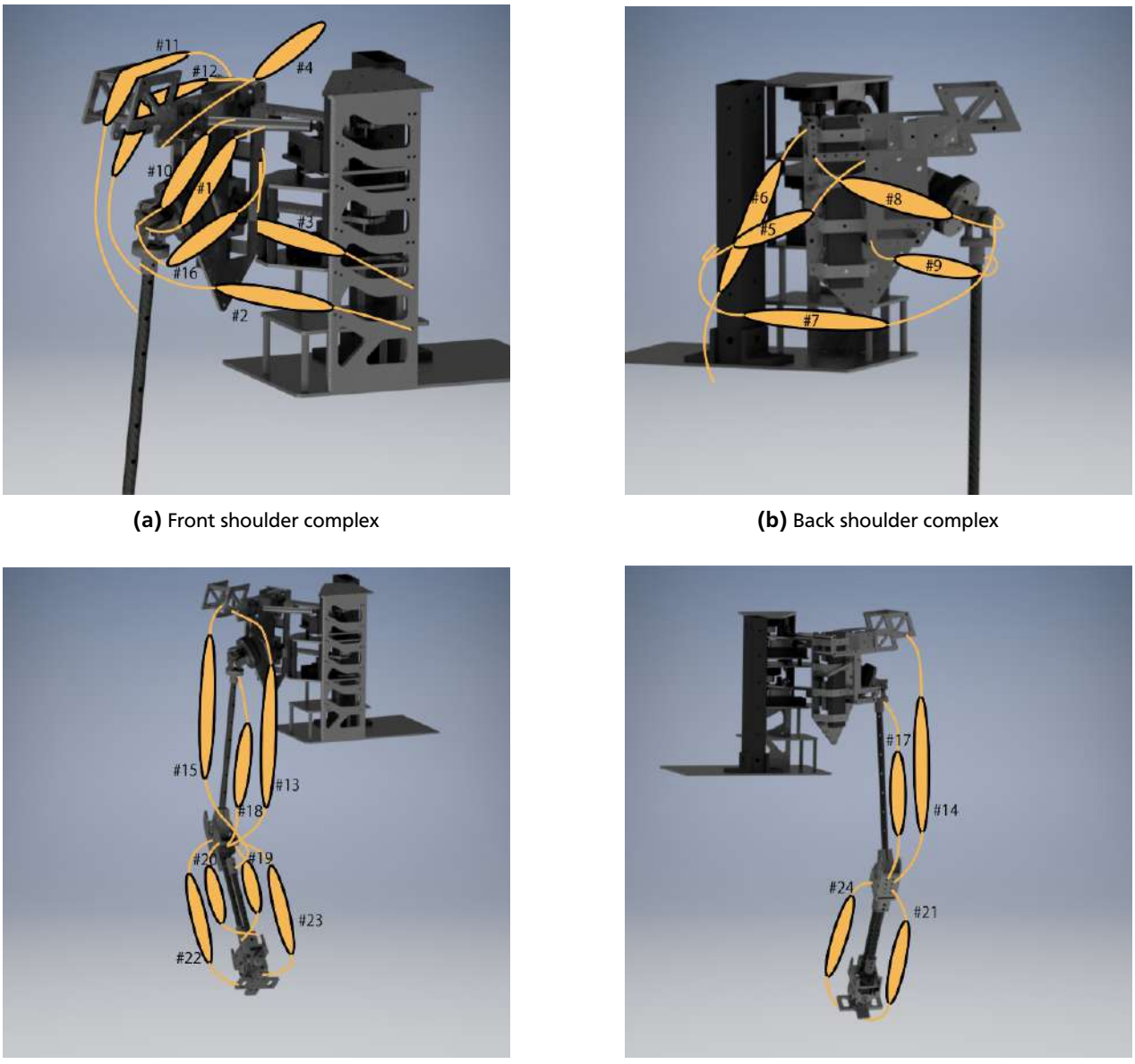
In online goal babbling and online motor babblng, tens of thousands movements are carried out to generate samples for sensorimotor learning. Therefore the dectection of the end of a movement is needed to efficiently generate movements and to speedup the experiments. Given the controller for each muscle and the published muscle state ROS topic that monitors the current pressure of the muscle, the end of an execution can be detected as all muscle pressures have reached the desired pressures within a small threshold ϵ governed by the PID controller. However, since the muscles have different sizes, different routings, and different mechanical constraints while interacting in the environment, the ϵ would be different for each muscle and is non-stationary over time. Therefore as long as the average muscle pressure changes within a windowed time is below ϵ, we conclude the execution. Given the internal pressures of all muscles $\mathbf{q}$ in a first-in last-out (FILO) buffer stack of size p, i.e., $\mathbf{q}^1, \cdots, \mathbf{q}^p$, if

$$\frac{\Sigma_{i=1}^{p-1} ||\mathbf{q}^p - \mathbf{q}^i||}{p} < \epsilon, \tag{2.1}$$

where $||\cdot||$ is the Euclidean norm, the execution could be considered finished. Nevertheless, there are cases where a sudden change in pressure actuation could lead to an oscillating or jittering end effector due to the elasticity and compliance of the PAMs. In order to address this issue, we also consider the stability of the end effector as the other indicator for the end of an execution. Similar to (2.1), given the tracking of the end effector position $\mathbf{x}$,

$$\frac{\Sigma_{j=1}^{l-1} ||\mathbf{x}^l - \mathbf{x}^j||}{l} < \xi, \tag{2.2}$$

where the FILO buffer stack is of length l and ξ is the threshold. If both conditions (2.1) and (2.2) are met, the execution then has ended and a new execution command can be sent. In the experiments both p and l is set to 100, $\epsilon = 4.5 \times 10^{-3}$, and $\xi = 2 \times 10^{-3}$. The rate of updating $\mathbf{q}^i$ and $\mathbf{x}^j$ is set to 100Hz, thus the detection is initialized 1 second before any motor command is sent out.



(a) Front shoulder complex

(b) Back shoulder complex

(c) Front arm linkage

(d) Back arm linkage

Figure 2.5.: The current muscle arrangements of the 24 muscles with extended task space [70], reduced from the 32 PAMs as in [62]. The muscles are labelled and displayed in either shoulder or arm groups from the front or back view. The names and functionalities of the muscles can be looked up in Table 2.1. The major muscles groups of the upper limb, such as the trapezius, deltoid, triceps, and biceps etc., are all preserved to replicate human-like upper limb movements.

In order to extend the task space, and arrange the muscles in an easily maintainable and replaceable fashion, we reduced the number of PAMs from 32 in [62] to 24 in [70] and adjusted the muscle arrangements. The new arrangement is shown in Figure 2.5, where the major muscle groups of a human are preserved in the upper limb robot, and the corresponding muscle names and functions can be looked up in Table 2.1. It can be observed that the major muscle groups of the shoulder complex, namely the trapezius and deltoid, are preserved to reproduce the abduction/adduction and flexion/extension movements of the shoulder. The major muscles of the upper arm, the triceps, and biceps are responsible for the flexion/extension and the supination of the arm. Antagonistic pairs of muscles are also implemented, such as muscle 21 and 24, i.e., the extensor carpi ulnaris extends and the flexor carpi ulnaris flexes the wrist antagonistically.

#	Muscle Name	Function
1	Pectoralis major	flexes, abducts, and rotates the humerus
2	Pectoralis minor	depresses the points of the shoulder and drawing the scapula superior
3	Serratus anterior	pulls scapula forward
4	Trapezius (superior)	rotates the scapula
5	Trapezius (middle)	retracts the scapula
6	Trapezius (inferior)	rotates the scapula
7	Latissimus dorsi	rotates scapula downward
8	Deltoid (rear)	abducts, flexes,
10	Deltoid (front)	and extends
9	Infraspinatus	externally rotates the humerus and stabilize the shoulder joints
11	Deltoid (medial)	the shoulder
12	Supraspinatus	abducts the arm
13	Biceps (short head)	supinate the forearm and flex the elbow
14	Triceps (long head)	extend the shoulder and elbow
15	Biceps (long head)	flex and supinate the forearm
16	Subscapularis	rotates and adducts the humerus
17	Triceps (lateral head)	extend the elbow
18	Brachialis	flexes the elbow
19	Pronator	pronates the hand
20	Supinator	supinates the hand
21	Extensor carpi ulnaris	extends and adducts the wrist
22	Extensor carpi radialis brevis	extends and abducts the wrist
23	Extensor carpi radialis longus	extends and radially abducts the wrist
24	Flexor carpi ulnaris	flexes the wrist

Table 2.1.: Muscle numbers with the corresponding names and functions of the 24 PAMs used for experiments. The major muscle groups of a human upper limb are preserved in the robot to mimic the motions of human arm and shoulder.

3 Theoretical Foundations

For learning the inverse kinematics, we use directed online goal babbling, and bootstrap on the generated samples to initialize for further motor babbling using CMA-ES. We will first review directed online goal babbling and CMA-ES, then introduce the local online motor babbling algorithm, which queries any static point within the space of learned IK for motor abundance.

3.1 Directed Online Goal Babbling

Given the specified convex goal space $\mathbf{X}^* \in \mathbb{R}^n$ encapsulating K goal points, and denoting all the reachable set of commands in the motor space as $\mathbf{Q} \in \mathbb{R}^m$, the aim is to learn the inverse kinematics model $\mathbf{X}^* \rightarrow \mathbf{Q}$, that generalizes all points in the goal space to a subset of solutions in the motor space. Starting from the known home position $\mathbf{x}_0^{home}$, and home posture $\mathbf{q}_0^{home}$, i.e., the inverse mapping $g(\mathbf{x}_0^{home}) = \mathbf{q}_0^{home}$, the goal-directed exploration is generated by

$$\mathbf{q}_t^* = g(\mathbf{x}_t^*, \boldsymbol{\theta}_t) + E_t(\mathbf{x}_t^*), \tag{3.1}$$

where $g(\mathbf{x}_t^*, \boldsymbol{\theta}_t)$ is the inverse mapping given learning parameter $\boldsymbol{\theta}_t$, and $E_t(\mathbf{x}_t^*)$ adds perturbation noise to discover new positions or more efficient motor commands in reaching goals. At every time step, the motor system forwards the perturbed inverse estimate, $\mathbf{x}_t, \mathbf{q}_t = \text{fwd}(\mathbf{q}_t^*)$, and the actual $(\mathbf{x}_t, \mathbf{q}_t)$ samples are used for regression, where prototype vectors and local linear mapping [72] is used as the regression model, and to monitor the progress of exploration in the defined goal space.

The major part of directed goal babbling is to direct the babbling of the end effector at specified goals and target positions. Each trial of goal babbling is directed towards one goal randomly chosen from $\mathbf{X}^*$, and continuous piecewise linear targets are interpolated along the path

$$\mathbf{x}_{t+1}^* = \mathbf{x}_t^* + \frac{\delta_x}{||\mathbf{X}_g^* - \mathbf{x}_t^*||} \cdot (\mathbf{x}_g^* - \mathbf{x}_t^*), \tag{3.2}$$

where $\mathbf{x}_t^*, \mathbf{X}_g^*$ are the target position and final goal of the trial, and δ_x being the step size. Target positions are generated until $\mathbf{x}_t^*$ is closer than δ_x to $\mathbf{X}_g^*$, then a new goal $\mathbf{X}_{g+1}^*$ is chosen. The purpose of directed goal babbling is to generate smooth movement around the end effector position, such that the locally learned prototype vectors can bootstrap and extend the exploration of the goal space, and allow the integration of the following weighting scheme

$$\begin{aligned} w_t^{\text{dir}} &= \frac{1}{2}(1 + \arccos(\mathbf{x}_t^* - \mathbf{x}_{t-1}^*, \mathbf{x}_t - \mathbf{x}_{t-1}) \\ w_t^{\text{eff}} &= ||\mathbf{x}_t - \mathbf{x}_{t-1}|| \cdot ||\mathbf{q}_t - \mathbf{q}_{t-1}||^{-1} \\ w_t &= w_t^{\text{dir}} \cdot w_t^{\text{eff}}, \end{aligned} \tag{3.3}$$

w_t^{dir} and w_t^{eff} measure direction and kinematic efficiency of the movement, such that inconsistency of a folded manifold, and redundant joint positions can be optimized [61]. The multiplicative weighting factor w_t is then integrated to the gradient descent that fits the currently generated samples by reducing the weighted square error.

To prevent drifting to irrelevant regions and facilitate bootstrapping on the local prototype centers, the system returns to $(\mathbf{x}^{\text{home}}, \mathbf{q}^{\text{home}})$ with probability $\mathbf{p}^{\text{home}}$ instead of following another goal directed movement. Returning to home posture stablizes the exploration in the known area of the sensorimotor space [28, 59], similar to infants returning their arms to a comfortable resting posture between practices:

$$\mathbf{q}_{t+1}^* = \mathbf{q}_t^* + \frac{\delta_q}{||\mathbf{q}_{\text{home}} - \mathbf{q}_t^*|| \cdot (\mathbf{q}^{\text{home}} - \mathbf{q}_t^*)}, \tag{3.4}$$

the system moves from the last posture $\mathbf{q}_t^*$ to the home posture $\mathbf{q}^{\text{home}}$ in the same way as in equation (3.2) by linearly interpolating the via-points along the path, until $||\mathbf{q}_{\text{home}} - \mathbf{q}_t^*|| < \delta_q$.

3.1.1 Local Linear Mapping Regression

The inverse mapping uses the Local Linear Model (LLM) for regression [72]. Let's define $\mathbf{x}$ to be the input, which in the goal babbling inverse mapping problem corresponds to the end effector position in 3D space, and $\mathbf{q}$ is the inferred motor command in the joint space. The LLM then comprises of $l = 1, \cdots, L$ linear models

$$g_l(\boldsymbol{x}) = \boldsymbol{W}_l \boldsymbol{x} + \boldsymbol{b}_l$$

which are combined as

$$\boldsymbol{q}(\boldsymbol{x}) = \frac{1}{n(\boldsymbol{x})} \sum_{l=1}^{L} b\left(\frac{\boldsymbol{x} - \boldsymbol{p}_l}{d}\right) g_l\left(\frac{\boldsymbol{x} - \boldsymbol{p}_l}{d}\right)$$

with Gaussian responsibilities

$$b(\boldsymbol{x}) = \exp(-||\boldsymbol{x}||^2)$$

and the normalization

$$\frac{1}{n(\boldsymbol{x})} = \sum_{l=1}^{L} b\left(\frac{\boldsymbol{x} - \boldsymbol{p}_l}{d}\right).$$

The linear functions g_l, weighted by the Gaussian responsibilities, are activated in the vicinity of the prototype vectors, i.e., the local centers, $\boldsymbol{p}_l$. New local functions g_l and prototype centers $\boldsymbol{p}_l$ are added dynamically during online training. Starting with the first training sample $(\boldsymbol{x}_1, \boldsymbol{y}_1)$, the first linear model with center $\boldsymbol{p}_1 = \boldsymbol{x}_1$, linear weight $\boldsymbol{W}_1 = \mathbf{0}$, and bias $\boldsymbol{b}_1 = \boldsymbol{y}_1$ is initialized. When new input $\boldsymbol{x}$ is received, that has a distance of at least d, i.e., the radius of the prototype centers, to all existing prototypes, a new prototype is then created.
In order to avoid abrupt changes in the inverse estimate, the new weight matrix to the newly added local model is initialized to be the Jacobian of the inverse estimate, i.e., $\boldsymbol{W}_l^{K+1} = J(\boldsymbol{x}) = \partial g_l(\boldsymbol{x}) / \partial \boldsymbol{x}$, and the offset is set to be the last inverse estimate before inserting the new local function, i.e., $b_l^{K+1} = g_l(\boldsymbol{x})$
In each time step, the inverse estimate is then fitted to the current sample $(\boldsymbol{x}_t, \boldsymbol{q}_t)$ by reducing the weight square error

$$\mathbf{E}_w^X = w_n \cdot ||\mathbf{q}_t - g(\mathbf{x}_t)||$$

using online gradient descent on $\mathbf{E}_w^{\mathbf{x}}$ with a learning rate η

$$\mathbf{W}_{t+1}^{(k)} = \mathbf{W}_t^{(k)} - \eta \frac{\partial \mathbf{E}_w^X}{\partial \mathbf{W}^{(k)}}, \quad \mathbf{b}_{t+1}^{(k)} = \mathbf{b}_t^{(k)} - \eta \frac{\partial \mathbf{E}_w^X}{\partial \mathbf{b}^{(k)}},$$

where w_t is the weighting scheme in equation (3.3).

3.1.2 Exploratory Noise

The exploratory noise, or motor perturbation in 3.1, is crucial for discovering new postures that would otherwise not be found by the inverse estimate [28, 59]. By exploring the local surrounding of the inverse estimate with i.i.d normal distribution in each motor dimension, and varying these distribution parameters with a normalized Gaussian random walk, the noise is modeled as:

$$E_t(\mathbf{x}_t^*) = \mathbf{A}_t \cdot \mathbf{x}_t^* + \mathbf{b}_t, \quad \mathbf{A}_t \in \mathbb{R}^{m \times n}, \quad \mathbf{b}_t \in \mathbb{R}^m, \tag{3.5}$$

where all entries e_t^i in the matrix $\mathbf{A}_t$ is initialized and varied as follows:

$$e_0^i \sim \mathcal{N}(0, \sigma^2), \quad \delta_{t+1}^i \sim \mathcal{N}(0, \sigma_\Delta^2)$$
$$e_{t+1}^i = \sqrt{\frac{\sigma^2}{\sigma^2 + \sigma_\Delta^2}} \cdot (e_t^i + \delta_{t+1}^i) \sim \mathcal{N}(0, \sigma^2).$$

3.1.3 Feedback Controller

After learning, the average reaching accuracy is evaluated by querying the inverse model for every goal within the defined goal space $\mathbf{X}^*$, and a simple feedback controller to adapt execution failures. Execution failure occurs when the inverse estimate is not possible to execute, i.e., $\mathbf{q}^* \notin \mathbf{Q}$, due to interference, non-stationary bionic robot design and $\mathbf{Q}$ constantly changing overtime. Given the queried goal $\mathbf{x}^*$ and the predicted posture $\mathbf{q}^* = g(\mathbf{x}^*)$, where $\mathbf{q}^* \notin \mathbf{Q}$, the feedback controller would slightly shift the queried goal from $\mathbf{x}^*$ to $\hat{\mathbf{x}}_t^*$, then forwarding the inverse estimate $\mathbf{x}_t = \text{fwd}(g(\hat{\mathbf{x}}_t^*))$. Target shifting follows the current observed error $\text{err}_t = \mathbf{x}^* - \mathbf{x}_t$, and integrated over time:

$$\hat{\mathbf{x}}_0^* = \mathbf{x}^*, \quad \hat{\mathbf{x}}_{t+1}^* = \hat{\mathbf{x}}_t^* + \alpha \cdot \text{err}_t. \tag{3.6}$$

3.2 Covariance Matrix Adaptation - Evolution Strategies (CMA-ES)

CMA-ES is a method of black box optimization that minimizes the objective function $f : \mathbf{Q} \in \mathbb{R}^m \to \mathbb{R}$, $q \to f(q)$, where f is assumed to be a high dimensional, non-convex, non-separable, and ill-conditioned mapping of the state space modelled by a multi-variate normal distribution.
The idea of CMA-ES is introducing a multi-variate normal distribution to sample a population, evaluating the population $f(\mathbf{q})$ to select the good candidates, and updating the search distribution parameters by adapting the covariance and shifting the mean of the distribution according to the candidates.
Given a start point q^0 and initializing the covariance to identity matrix $\mathbf{C^0} = \mathbf{I}$, the search points in one population iteration is sampled as follows:

$$\mathbf{q}_k^{(g+1)} \sim \mathbf{m}^{(g)} + \sigma \mathbf{y}_k^{(g)} \quad k = 1, \cdots, \lambda \quad \mathbf{q}_k, \mathbf{m} \in \mathbb{R}^n, \sigma \in \mathbb{R}_+, \mathbf{C} \in \mathbb{R}^{n \times n} \tag{3.7}$$

where $\mathbf{y}_k^{(g)} = \mathcal{N}_i(\mathbf{0}, \mathbf{C}^{(g)})$, $\mathbf{m}$ being the mean vector, σ being the step-size, and λ is the population size [66]. For notation simplicity, the generation index (g) is henceforth omitted.

3.2.1 Updating the Mean

The mean vector m is updated by using the non-elitistic selection [66], where a weighted intermediate recombination of the best μ solutions is applied, rather than choosing the only best as in the elitistic selection. Let $q_{i:\lambda}$ denote the ith best solution in the population of λ, the best μ points from the sampled population are then selected, such that $f(\mathbf{q}_{1:\lambda}) \leq \cdots \leq f(\mathbf{q}_{\mu:\lambda})$, and weighted intermediate recombination is applied:

$$\mathbf{m} \leftarrow \mathbf{m} + \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} =: \mathbf{m} + \mathbf{y}_w, \tag{3.8}$$

$$\text{where } w_1 \geq \cdots \geq w_\mu > 0, \sum_{i=1}^{\mu} w_i = 1, \tag{3.9}$$

$$\text{and } \mu_{\text{eff}} = \left(\frac{||\mathbf{w}||_1}{||\mathbf{w}||_2}\right)^2 = \frac{||\mathbf{w}||_1^2}{||\mathbf{w}||_2^2} = \frac{(\sum_{i=1}^{u} |w_i|^2)}{\sum_{i=1}^{u} w_i^2} = \frac{1}{\sum_{i=1}^{u} w_i^2} =: \mu_w \approx \frac{\lambda}{4}. \tag{3.10}$$

μ_{eff} is refered as the *variance effective selection mass*. From the definition of w_i in (3.9) we can derive that $1 \leq \mu_{\text{eff}} \leq \mu$, and $\mu_{\text{eff}} = \mu$ for equal recombination weights. Usually $\mu_{\text{eff}} \approx \lambda/4$ indicates a reasonable setting of w_i. A a simple and feasible setting could be $w_i \propto \mu - i + 1$, and $\mu \approx \lambda/2$ [66].

3.2.2 Adapting the Covariance Matrix

The covariance matrix evolves the scale of distribution, shaping the population solutions in the favored direction according to the evaluated object function values. Starting from the empirical estimation of the covariance matrix, both rank-μ-update and rank-one-update are derived and used to adapt the covariance matrix.

Estimating the Covariance Matrix

Assume that the popuplation ccontains enough information to estimate a covariance metrix reliably, we can empirically estimate the covariance matrix from scratch using the sampled population from (3.7), $\mathbf{q_1}, \cdots, \mathbf{q_\lambda}$, via the empirical covariance matrix

$$\mathbf{C}_{emp} = \frac{1}{\lambda - 1} \sum_{i=1}^{\lambda} \left(\mathbf{q}_i - \frac{1}{\lambda} \sum_{j=1}^{\lambda} \mathbf{q}_j \right) \left(\mathbf{q}_i - \frac{1}{\lambda} \sum_{j=1}^{\lambda} \mathbf{q}_j \right)^T,$$

where $\mathbf{C}_{emp}$ uses the *actually realized* sample as the reference mean value, and the covariance estimator can be interpreted as the estimated distribution variance within the sampled points. However in order to estimate the variances of sampled steps and the evolution between generations, the reference mean value $\mathbf{m}$ can be used for the covariance estimation,

$$\begin{aligned} \mathbf{C}_\lambda &= \frac{1}{\lambda} \sum_{i=1}^{\lambda} (\mathbf{q_i} - \mathbf{m})(\mathbf{q_i} - \mathbf{m})^T \\ &= \frac{1}{\lambda} \sigma^2 \sum_{i=1}^{\lambda} \mathbf{y_i} \mathbf{y_i}^T. \end{aligned}$$

For a more robust estimation, the same *weighted selection* as in (3.8) can be applied

$$\mathbf{C}_\mu = \sigma^2 \sum_{i=1}^{\mu} w_i \mathbf{y_i} \mathbf{y_i}^T, \tag{3.11}$$

the matrix $\mathbf{C}_\mu$ is an estimator for the distribution of the selected steps, whereas $\mathbf{C}_\lambda$ estimates the original distribution of steps before selection. Choosing the best μ samples from the population tend to reproduce successful steps when sampling from the adapted new covariance $\mathbf{C}_\mu$ [66].

Rank-μ-Update

In order to get a reliable estimation, the variance effective selection mass μ_{eff} must be larget enough, however evaluation of the objective function $f(\mathbf{q})$ can be expensive as each trial would require the upper limb robot to execute the movement $\mathbf{q}$. Thus to achieve a fast search, both the population size λ and μ_{eff} must be small, e.g., we may assume $\mu_{\text{eff}} \leq 1 + \ln m$, where m is the dimension of the motor space, i.e., $m = 24$ PAMs. In addition, we use information from previous generations, i.e., using the mean of all the estimated covariance matrices after a sufficient number of generations,

$$\mathbf{C}^{(g+1)} = \frac{1}{g+1} \sum_{i=1}^{g} \frac{1}{\sigma^{(i)^2}} \mathbf{C}_\mu^{(i+1)}$$

then becomes a more reliable estimator for the selected steps. However all generation steps have the same weight. Exponential smoothing is then introduced to assign recent generations a higher weight. Choosing $\mathbf{C^{(0)}} = \mathbf{I}$ to be the unity matrix and a learning rate $0 < c_\mu \leq 1$, then $\mathbf{C}^{g+1}$ is updated as

$$\begin{aligned} \mathbf{C}^{(g+1)} &= (1 - c_\mu)\mathbf{C}^{(g)} + c_\mu \frac{1}{\sigma^{(g)^2}} \mathbf{C_\mu^{(g+1)}} \\ &= (1 - c_\mu)\mathbf{C}^{(g)} + c_\mu \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda}^{(g+1)} \mathbf{y}_{\mathbf{i:\lambda}}^{\mathbf{(g+1)}^T} \\ \text{omiting (g), i.e.,} \quad \mathbf{C} &\leftarrow (1 - c_\mu)\mathbf{C} + c_\mu \sum_{i=1}^{\mu} w_i \mathbf{y}_{i:\lambda} \mathbf{y}_{\mathbf{i:\lambda}}^T. \end{aligned} \tag{3.12}$$

This update is then called the rank-μ-update [74], since the sum of the outer products in (3.12) is of rank $\min(\mu, m)$ with probability one (given μ non-zero weights) [66].

Rank-One-Update Utilizing the Evolution Path

When estimating the covariance matrix from scratch, all selected steps are used from a single generation, an opposite viewpoint would suggest to repeatedly update the covariance matrix in the generation sequence using a single selected step only, where $\mu = 1$ and $\mathbf{y}_{1:\lambda} = (\mathbf{x}_{1:\lambda} - \mathbf{m})/\sigma$, then the rank-one update of the covariance matrix reads

$$\mathbf{C} \leftarrow (1 - c_1)\mathbf{C} + c_1 \mathbf{y}_{1:\lambda} \mathbf{y}_{1:\lambda}^T, \tag{3.13}$$

where c_1 is the learning rate and $0 < c_1 \leq 1$. Rank-one update has been independently found in several domains [75–78], in light of CMA-ES and adapting parameters, specifically the covariance, the evolution strategies mainly rely on the covariance adaptation of arbitrary normal mutation distributions [76]. Consider the vectors $\mathbf{y}_1, \cdots, \mathbf{y}_{g_0} \in \mathbb{R}^m, g_0 \geq m$, which span $\mathbb{R}^m$, and let $\mathcal{N}(0,1)$ denote independent (0, 1)-normally distributed random numbers, an n-dimensional normal distributions with zero mean can then be produced as

$$\mathcal{N}(0,1)\mathbf{y}_1 + \cdots + \mathcal{N}(0,1)\mathbf{y}_{g_0} \sim \mathcal{N}\left(\mathbf{0}, \sum_{i=1}^{g_0} \mathbf{y_i} \mathbf{y_i}^T\right), \tag{3.14}$$

which can be interpreted as a normally distributed random vector with zero mean covariance matrix $\sum_{i=1}^{g_0} \mathbf{y_i}\mathbf{y_i}^T$ generated by adding "line distributions" $\mathcal{N}(0,1)\mathbf{y_i}$. Considerring all normal distributions with zero mean, the singular distribution $\mathcal{N}(0,1)\mathbf{y}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{y}_i \mathbf{y}_i^T)$ generates the vector $\mathbf{y}_i$ with maximum likelihood, which must live on a line that includes $\mathbf{y}_i$, and thus the distribution must obey $\mathcal{N}(0,1)\sigma\mathbf{y}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{y}_i \mathbf{y}_i^T)$, otherwise any other line distribution with zero mean cannot generate $\mathbf{y}$ at all. Choosing a proper σ then reduces to choosing the maximum likelihood of $||\mathbf{y}||$ for the one-dimensional gaussian $\mathcal{N}(0, \sigma^2 ||\mathbf{y}_i^2||)$, which is $\sigma = 1$. Having the rank of one, the only eigenvectors of the covariance matrix $\mathbf{y}_i \mathbf{y}_i^T$ are $\{\alpha \mathbf{y}_i | \alpha \in \mathbb{R}_{\setminus 0}\}$ with the eigenvalue $||\mathbf{y}_i||^2$. Thus any normal distributions can be realized using (3.14) when $\mathbf{y}_i$ are chosen appropriately, as long as the vectors $\mathbf{y}_i$ are not the eigenvectors of the covariance matrix, which they usually are not. Considering (3.14) and a slight simplification of (3.12), the sum in (3.12) can be consist of a single summand only, thus leading to the rank-one update of the covariance matrix in (3.13).
In (3.12) and (3.13), the covariance matrix is updated using the matrix multiplication of the selected steps, i.e., $\mathbf{y}_i \mathbf{y}_i^T$, however since $\mathbf{y}_i \mathbf{y}_i^T = -\mathbf{y}_i(-\mathbf{y}_i)^T$, the sign information is lost when updating the covariance. Thus a so-called *evolution path* is introduced to restore the sign information. Evolution path refers to the sum of a sequence of successive steps over a number of generations, weighted by exponential smoothing. Starting with $\mathbf{p}_c^{(0)} = \mathbf{0}$, and $\mathbf{p}_c \in \mathbb{R}^n$, the evolution path is updated as

$$\mathbf{p}_c^{(g+1)} = (1 - c_c)\mathbf{p}_c^{(g)} + \sqrt{c_c(2 - c_c)\mu_{\text{eff}}} \frac{\mathbf{m}^{(g+1)} - \mathbf{m}^{(g)}}{\sigma^{(g)}}, \tag{3.15}$$

where $c_c \leq 1$ being the learning rate, and $1/c_c$ can be considered as the backward time horizon of the evolution path $\mathbf{p}_c$ that contains roughly 63% of the overall weight [66]. The factor $\sqrt{c_c(2 - c_c)\mu_{\text{eff}}}$ is then a normalization constant for $\mathbf{p}_c$. When $c_c = 1$ and $\mu_{\text{eff}} = 1$, the factor reduces to one, and $\mathbf{p}_c^{(g+1)} = (\mathbf{q}_{1:\lambda}^{(g+1)} - \mathbf{m}^{(g)})/\sigma^{(g)}$.
By integrating rank-one-update in (3.13) with the evolution path (3.15), the update of covariance matrix then reads [88]

$$\mathbf{C} \leftarrow (1 - c_1)\mathbf{C} + c_1 \mathbf{p}_c \mathbf{p}_c^T. \tag{3.16}$$

The rank-one-update with revolution path (3.16) significantly improves the the covariance update compared to (3.13) for small μ_{eff}, due to the heavily explored correlations between consecutive steps and the restored sign information [66].

Update the Covariance Using Both Rank-One-Update and Rank-μ-Update

Combining rank-μ-update in (3.12) and rank-one-update in (3.16), the final CMA update of the covariance matrix reads

$$\mathbf{C} \leftarrow (1 - c_1 - c_\mu)\mathbf{C} + c_1 \mathbf{p}_c \mathbf{p}_c^T + c_\mu \sum_{i=1}^{\lambda} w_i \mathbf{y_{i:\lambda}} \mathbf{y_{i:\lambda}}^T, \tag{3.17}$$

where

$$c_1 \approx \frac{2}{m^2}, \quad c_\mu \approx \min\left(\frac{\mu_{\text{eff}}}{m^2}, 1 - c_1\right),$$
$$\mathbf{y}_{i:\lambda}^{(g+1)} = \frac{(\mathbf{x}_{i:\lambda}^{(g+1)} - \mathbf{m}^{(g)})}{\sigma^{(g)}}, \quad \sum w_j = \sum_{i=1}^{\lambda} w_i \approx -\frac{c_1}{c_\mu}.$$

The combined update (3.17) combines both the advantages of rank-μ-update and the rank-one update. Rank-μ-update efficiently uses information from the entire population, suitable for large populations, while rank-one-update exploits the correlations between generations using the evolution path, which is crucial in small populations [66].

3.2.3 Step-Size Control

The covariance matrix adaptation in (3.17), increases or decreses the scale of the distribution only in a single direction for each selected step, or decreases the scale by fading out old information by a given, non-adaptive factor. However CMA does not explicity control the "overall scale" of the distribution, i.e., step length σ, mainly due to the fact that the largest reliable learning for the CMA in (3.17) is too slow to achieve competitive change rates for the overall step length. Thus a step-size control is needed in addition to the adaptation rule (3.17) [66].
The step size σ is updated using cumulative step-size adaptation (CSA). The intuition is when the evolution path, i.e., the sum of successive steps, is short, single steps tend to be uncorrelated and cancel each other out, thus the step-size should be decreased. On the contrary, when evolution path is long, single steps points to similar directions and tend to be correlated, therefore increasing the step size. When in desired situations, the steps are then approximately perpendicular in expectation and therefore uncorrelated. The same idea of evolution path as in (3.15) is constructed, however the difference here is that a conjugate evolution path is considered since $\mathbf{p}_c$ from (3.15) depends on its direction. Initializing the evolution path vector $\mathbf{p}_\sigma = \mathbf{0}$, the conjugate evolution path reads:

$$\mathbf{p}_\sigma \leftarrow (1-c_\sigma)\mathbf{p}_\sigma + \sqrt{c_\sigma(2-c_\sigma)\mu_{\text{eff}}}\mathbf{C}^{-\frac{1}{2}}\frac{\mathbf{m}^{(g+1)}-\mathbf{m}^{(g)}}{\sigma^{(g)}}, \tag{3.18}$$

where $c_\sigma < 1$ being the learning rate, similar to c_1 and c_c in (3.17), and $1/c_\sigma$ is the backward time horizon of the evolution path. $\sqrt{c_\sigma(2-c_\sigma)\mu_{\text{eff}}}$ is the normalization constant. And $\mathbf{C}^{-0.5} \triangleq \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^T$, where $\mathbf{C} = \mathbf{B}\mathbf{D}^2\mathbf{B}^T$ is an eigendecomposition of $\mathbf{C}$. $\mathbf{B}^T$ is the transform of the orthonormal basis of eigenvectors, which rotates the space such that the principal axes of the distribution $\mathbf{0}, \mathbf{C}$ is aligned in the coordinate axes. The diagonal elements of $\mathbf{D}$ are the square roots of the corresponding positive eigenvalues, where $\mathbf{D}^{-1}$ applies a (re)-scaling so that all axes are equally sized. $\mathbf{B}$ then rotates the result back into the original coordinate system, ensuring that the principal axes of the distributions are not rotated by the overall transformation and directions of the consecutive steps are comparable. Therefore the transformation $\mathbf{C}^{-0.5}$ makes the expected length of $\mathbf{p}_\sigma$ independent of its direction, and for any sequence of realized covariance matrices $\mathbf{C}^{(g)}_{g=0,1,2,\cdots}$, given $\mathbf{p}_\sigma^{(0)} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$, the evolution path is under random selection $\mathbf{p}_\sigma^{(g+1)} \sim \mathcal{N}(\mathbf{0},\mathbf{I})$ [78].
In order to update the step length σ, the $||\mathbf{p}_\sigma||$ is "compared" with its expected length $E||\mathcal{N}(\mathbf{0},\mathbf{I})||$, and updated in log scale which is unbiased. With some manipulations, the final update of σ reads [66]

$$\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{||p_\sigma||}{E||\mathcal{N}((\mathbf{0},\mathbf{I}))||}-1\right)\right), \tag{3.19}$$

where $d_\sigma \approx 1$ being the damping factor $\ln\sigma$ [78], and $E||\mathcal{N}((\mathbf{0},\mathbf{I}))||$ is the expectation of the Euclidean norm of a random vector that's normally distributed with zero mean [66].

3.3 Local Online Motor Babbling

When learning inverse kinematics using online goal babbling, since there are multiple postures $\mathbf{q}$ reaching x^*, it is assumed that we don't need to know all redundancy, and only learn the ones with most direction and kinematic efficiency by integrating the weighting scheme (3.3) in the optimization. In fact, $\mathbf{Q}$ is not only unknown, and may never be exhaustively explored on a physical system, but also non-stationary due to the nature of musculoskeletal robot design with PAMs. This can be addressed by using the simple feedback controller in (3.6), where execution failures due to the changing of $\mathbf{Q}$ are adapted when the queried goal x^* is slightly shifted based on the proportion of the euclidean error $x^* - x_t$.
As illustrated in Algorithm 1, the queried goal x^*, the learned inverse model $g(x)$, and the neighboring postures $\mathbf{Q_x}$: $q_t \forall x_t \iff ||x_t - x^*|| < r$, which is collected from the goal babbling process, are the input to online motor babbling. The aim of the algorithm is to output a new posture configuration set $\mathbf{Q_{cma}}$, from which different muscle stiffness can be generated while keeping the end effector position fixed. The initialization sets the gain and number of iteration of the feedback controller to $\alpha = 0.05$, $T = 30$, t number of trials for CMA-ES $N = 5$, and the prototype sphere radius is $r = 0.02$. We use `pycma` library [79] to implement CMA-ES, where we encode variables q in the objective function implicitly $f(\text{fwd}(q))$ [66]. The objective function is simply set as the euclidean norm to the goal scaled with a constant, i.e., $c \cdot ||x^* - x_t||$, where $c = 10$, and the optimum objective function value is set to $f^* = 0.03$, meaning that an empirical optimum of $f^*/c = 3mm$ to the goal, which is also the stopping criteria for each CMA-ES trial.
Each trial of CMA-ES starts by iterating the feedback controller and finding the posture q_t that leads closest to the neighboring goal, and q_t is subsequently used to initialize the mean vector m. The covariance is initialized to be an identity matrix, which allows isotropic search and avoids bias. In order to initialize the step-size, an empirical variance is estimated from $\mathbf{Q_x} \cup \mathbf{Q_{fb}}$, and the mean of the variance is taken as initialization. The union of the two sets is to ensure

Algorithm 1: Motor Babbling Using CMA-ES

input : x^*, $g(x)$, $\mathbf{Q_x}$
output : $\mathbf{Q_{cma}}$
initialize: $\alpha = 0.05,\ T = 30,\ N = 5,\ \lambda = 13,\ r = 0.02,\ c = 10,\ f^* = 0.03,\ \mathbf{Q_{cma}} = \{\}$

select N closest goals $x_1^*, \cdots, x_N^*$ *to* x^*;
for $n \leftarrow 0$ **to** N **do**
  $\hat{x}_0^* = x_n^*$;
  $\mathbf{Q_{fb}} = \{\}$;
  for $t \leftarrow 0$ **to** T **do**
    $x_t, q_t = \text{forward}(g(\hat{x}_t^*))$;
    $\hat{x}_t^* = \hat{x}_{t-1}^* + \alpha \cdot (x^* - x_t)$;
    if $||x^* - x_t|| < r$ **then**
      collect $(x_t,\ q_t)$ In $\mathbf{Q_{fb}}$;
  select q_t *for the minimum* $||x_t - x^*||$ *in* $\mathbf{Q_{fb}}$;
  initialize $m = q_t,\ \sigma = mean(var(\mathbf{Q_x} \cup \mathbf{Q_{fb}})),\ \mathbf{C} = \mathbf{I}$;
  while $\hat{f} < f^*$ **do**
    sample posture population $\mathbf{q_s} : q_1 \cdots q_\lambda$ as in (3.7);
    for $k \leftarrow 1$ **to** λ **do**
      $x_t, q_t = \text{forward}(q_k)$;
      $\hat{f} = f(x_k) = c \cdot ||x^* - x_t||$;
      if $||x^* - x_t|| < r$ **then**
        collect q_k in $\mathbf{Q_{cma}}$;
    update $\mathbf{m}$ as in (3.8);
    update $\mathbf{p}_\sigma$ and σ as in (3.18), (3.19);
    update $\mathbf{p_c}$ and $\mathbf{C}$ as in (3.15), (3.17);

sufficient data for a feasible estimation. Near the home position, which is the centroid of the goal space, many data samples are available as online goal babbling often comes back to (x_{home}, q_{home}). However around the edges of the goal space, there are often very few local samples, sometimes less than the action space dimension, i.e., the 24 muscles. By taking in the samples generated by feedback controller, a better initialization of σ can be robustly estimated.

3.4 Reproducing Muscle Abundance

In order to visualize muscle abundance, namely in terms of reproducing muscle stiffness and muscle synergy encoded in the evolved covariance matrix, we assume the distribution of parameters to be multi-variate Gaussian and multi-modal, as the motor space is of high dimension, and there can be different muscle group posture configurations while keeping the end effector fixed. Therefore a multi-variate Gaussian Mixture Model [80] is fit to the collected data in $\mathbf{Q}$. By assuming a distribution of Gaussian parameters over the data samples $p(\mathbf{Q}|\theta)$, a prior multi-variate Gaussian distribution is introduced

$$p(\theta) = \sum_{i=1}^{K} w_i \mathcal{N}(\mu_\mathbf{i}, \mathbf{\Sigma_i}),$$

w_i are the weights for each Gaussian mixture component, and the posterior distribution is estimated by using Bayes rule [80], such that the posterior distribution would preserve the form Gaussian mixture model, i.e.,

$$p(\theta|\mathbf{Q}) = \sum_{i=1}^{K} \tilde{w}_i \mathcal{N}(\mu_\mathbf{i}, \tilde{\mathbf{\Sigma}}_\mathbf{i}),$$

where the parameters $(\mu_\mathbf{i}, \tilde{\mathbf{\Sigma}}_\mathbf{i})$ and weights $\tilde{w}_i$ are updated using Expectation Maximization (EM) to maximize the likelihood [80]. The number of mixture models P is estimated using Bayesian Information Criterion (BIC) [80] for $P \in [1, 10]$, where the lowest BIC of P is taken. Finally, we sample from the mixture model with updated parameters and weights $q^* \sim \sum_{i=1}^{K} \tilde{w}_i \mathcal{N}(\mu_\mathbf{i}, \tilde{\mathbf{\Sigma}}_\mathbf{i})$ and forward q^* on the robot.

4 Experiments and Evaluations

We implement directed online goal babbling to learn the IK, and local online motor babbling to explore for motor abundance for the queried goal point, respectively. Since the task space is unknown, we first empirically estimate the goal space using some simple heuristics. Here the term task space and goal space are slightly abused. The original literature refers to the defined space for goal babbling as the goal space, which is technically a subset of the task space that encapsulates all the reachable positions by the end effector. In our experiment setup, we try to maximize the defined goal space to equate to the size of the unknown task space, thus the term goal space and task space is henceforth interchangeably used.

4.1 Experiment Setup

The general setup for goal babbling and motor babbling experiments consists of the mentioned robot hardware above, firmware and software. The sensory space so far only consists of the pressure sensor in each muscle for PID control, thus in order to learn the IK with sensorimotor mapping, tracking is needed to obtain the end effector position in 3D. In the experiments, the tracking is performed using Intel RealSense ZR300 and its open source API [71]. As shown in Figure 4.1, the hand of the robot is replaced with a tennis ball as the color marker, and tracking of the end effector is performed in reference to the center of the red marker as the origin. However, the tracking introduces an error up to 1 cm in depth, i.e., x-axis, and sub-millimeter error in y and z-axis. The colored point cloud overlayed in ROS rviz is the specified convex goal space, which will be introduced in Section 4.2.1.

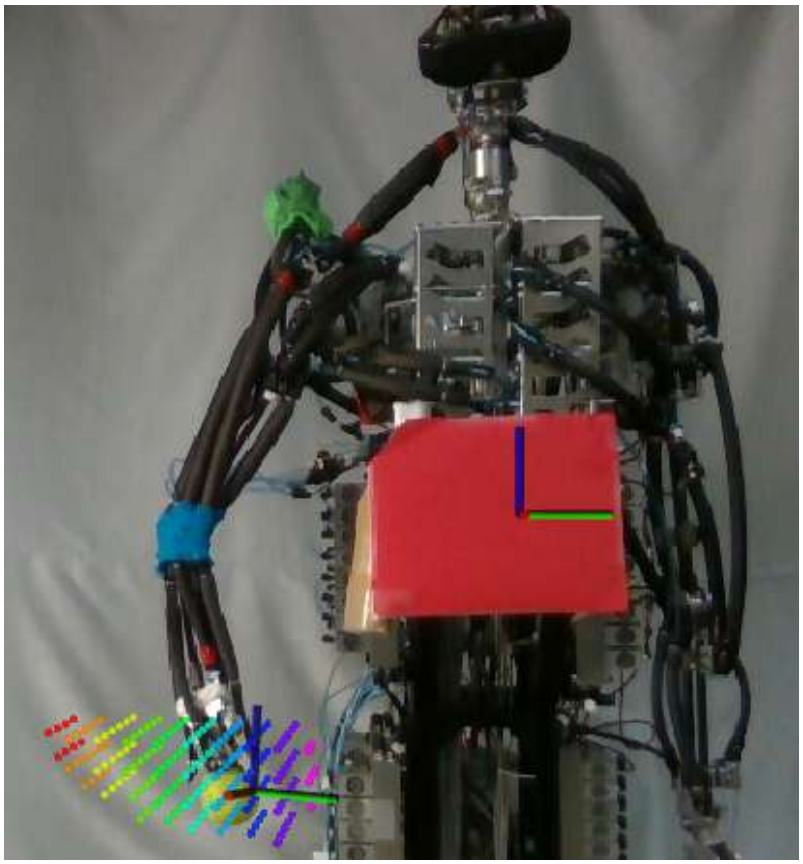


Figure 4.1.: The final experiment setup of the robot, ready to conduct goal babbling. The 10 DoF musculoskeletal robot arm is actuated by the 24 PAMs shown in Figure 2.5, with an empirically defined goal space in reference to the red marker on the chest, visualized in `rviz`.

The ROS architecture of the setup is shown as a node graph in Figure 4.2. The `/muscle_controller_spawner` spawns the PID parameters to the ROS server, as shown in Figure 2.4. `/arl_driver_node` provides information for `/diagnostics`,

and current muscle pressure to `/musculature/state`, which further publishes the internal pressures for `exec_dection` as in equation (2.1). The defined goal space that is reachable by the end effector of the robot is published by `/pointcloud_pub`, displayed as the colorful point cloud in Figure 4.1. The red marker on the chest of the robot is used as a static `/tf` frame for the reference as the origin, the tracked end effector position is then transformed with respect to the origin in `/tf` and publish to `/hand_pose` for `/eff_detection` as in equation (2.2).

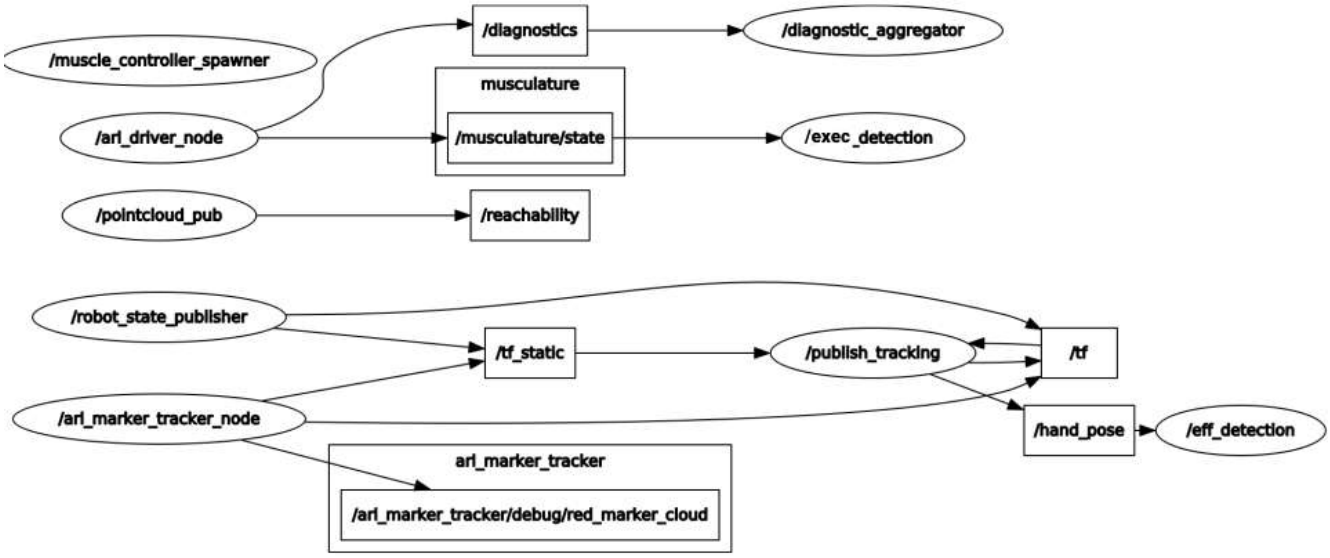


Figure 4.2.: Node graph of the ROS architecture for the experiment setup: The driver communicates the muscle state pressure for diagnostics, and publishes for execution detection as in equation (2.1). The defined goal space is published for visualization. The tracking uses the red frame on the chest of the robot as a static tf frame for origin, transforms the current end effector position in `/tf` and publishes to `/hand_pose`, which is used to detect the end of a movement shown in equation (2.2) as the end effector stabilizes.

Before learning the IK, the best achievable accuracy in reaching tasks is estimated as in [28]

$$\bar{x}_p = \frac{1}{R}\sum_r x_p^r,$$
$$D = \frac{1}{P}\sum_p \frac{1}{R}\sum_r ||x_p^r - \bar{x}_p||.$$

By repeating $P = 20$ random postures for $R = 20$ times each, the average Euclidean norm error is computed to be $D = 1.2$ cm, meaning that the musculoskeletal upper limb robot can reach with the best average error of 1.2 cm. In comparison to accurate and agile control in typical robots with rigid links, this is of quite low accuracy. For soft robots with a musculoskeletal structure, reaching accuracy is traded for better flexibility and compliance. Nevertheless, this highly non-linear non-stationary redundant system still poses great challenges to learn an accurate IK, due to friction, hysterisis, and the strong mechanical interplay between the bended muscles and the skeleton. The nonstationarity and the quick wear-and-tear of the PAMs can lead to deviations in end effector positions even when supplying the same air pressures, and thus changing the reachable goal space, which is not even analytically known in the first place. These challenges will be addressed later by using directed online goal babbling that bootstraps on the inverse mapping, and a simple feedback controller to adapt to the nonstationary changes during the learning process [28, 61]. The unknown goal space will be empirically estimated and approximated using simple heuristics, which are introduced in Section 4.2.

4.2 Learning Inverse Kinematics

4.2.1 Defining the Goal Space

The complete task space of the upper limb robot is unknown and non-convex, however directed goal babbling would require the specified goal space to be convex to efficiently bootstrap and allow the integration of the weighting scheme in equation (3.3). Thus we first empirically estimate the goal space by randomly generating 2000 random postures for each muscle within $[0, 0.4]$ MPa, and denote as the empirical goal space $\mathbf{X}_E$.

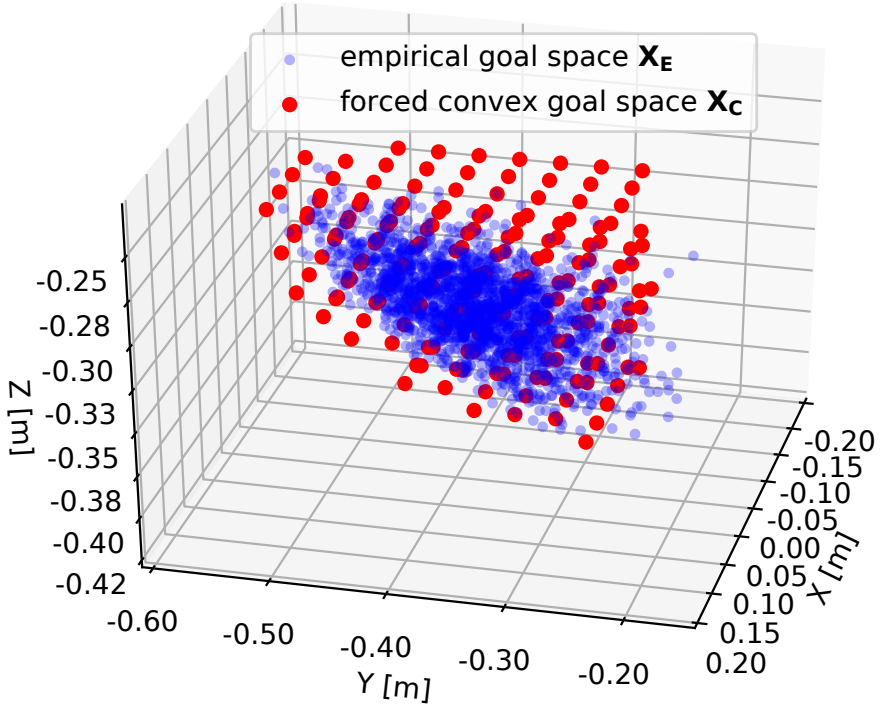


Figure 4.3.: Empirical goal space $\mathbf{X}_E$ (in blue) is obtained by sampling from 2000 random postures. A uniformly sampled convex goal space $\mathbf{X}_C$ (in red) is first defined by a cube grid $\mathbf{C}$ with 3 cm spacing encapsulating $\mathbf{X}_E$, and intersected with the forced convex hull of $\mathbf{X}_E$ using the quickhull algorithm [73], i.e., $\mathbf{X}_C = \text{Conv}(\mathbf{X}_E) \cap \mathbf{C}$. $\mathbf{X}_C$ is then used for learning IK, as shown in Figure 4.1

In order to approximate the uniform samples in $\mathbf{X}_E$ for efficient online learning and evaluations, a cube grid $\mathbf{C}$ with $3cm$ spacing encapsulating $\mathbf{X}_E$ is defined, where $\mathbf{X}_E \subset \mathbf{C}$. The sampled convex hull goal grid $\mathbf{X_C}$ in Figure 4.1 is then made from the intersection of all points in the empirical goal space and the cube grid, i.e., $\mathbf{X}_C = \text{Conv}(\mathbf{X}_E) \cap \mathbf{C}$, where Conv forces the convex hull using the quickhull algorithm [73]. However, as shown in Figure 4.3, $\mathbf{X}_E$ is a slanted non-convex irregular ellipsoid, forcing a convex hull in the 2000 random posture samples would introduce non-reachable regions in the goal space. This is addressed later with the similar set operation to remove the outlier goals using the learned prototype vector space.

4.2.2 Evaluations and Results

The experiment is conducted with $T = 20000$ samples, with target step length $\delta_x = 0.02$, which corresponds to the target velocity of 2 cm/s, allowing the robot to generate smooth local movements. The sampling rate is set to $5Hz$, generating 5 targets and directed micro movements for learning. After every 4000 samples, performance evaluation is carried out online. The learning experiment including online evaluations amount to less than 2 hours real time. As illustrated in Figure 4.4, the learning bootstraps quite fast in the first 4000 samples, followed by a slow convergence until 16000 samples. At $T = 20000$, the feedback controller is applied, the performance error drops to 3.4cm. However in $\mathbf{X_C}$ there are still many outlier goals, which are the non-reachable regions introduced by forcing the convex hull. A similar set intersection operation is applied with the learned prototype spheres $\mathbf{S}$ and the goal space $\mathbf{X_C}$, where $\mathbf{S}$ is taken as the encapsulated space of the prototype spheres, and the final goal space is $\mathbf{X}_S = \mathbf{S} \cap \mathbf{X}_C$, as shown in Figure 4.6, where the number of goals has been reduced from 179 in $\mathbf{X}_C$ to 94 in $\mathbf{X}_S$. We then evaluate again these 94 goals with the feedback controller, the performance error reduces further to an average of 1.8 cm in 4.4. However due to the forced convex hull $\mathbf{X}_C$, local inverse models cannot efficiently regress at the edge of the task space, the error distribution still shows a few errors larger than 3 cm, which can be further reduced later by motor babbling using CMA-ES.

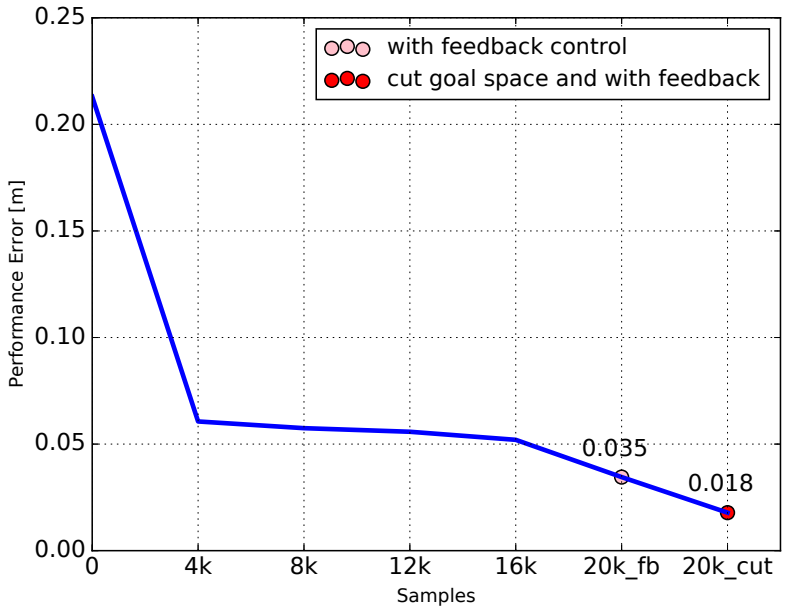


Figure 4.4.: Decreasing performance error up to 20000 samples, i.e., the average Euclidean norm to the goals evaluated throughout the convex goal space $\mathbf{X_C}$, the feedback controller is applied at 20000 samples, resulting an average error of 3.5 cm. However there are still outlier goals remaining from forcing the convex hull, thus we take the explored prototype sphere space $\mathbf{S}$ and intersect with $\mathbf{X_C}$, i.e., $\mathbf{X_S} = \mathbf{S} \cap \mathbf{X_C}$ to remove the outliers. Evaluating on the cut goal space $\mathbf{X_S}$ reduces the error to 1.8 cm.

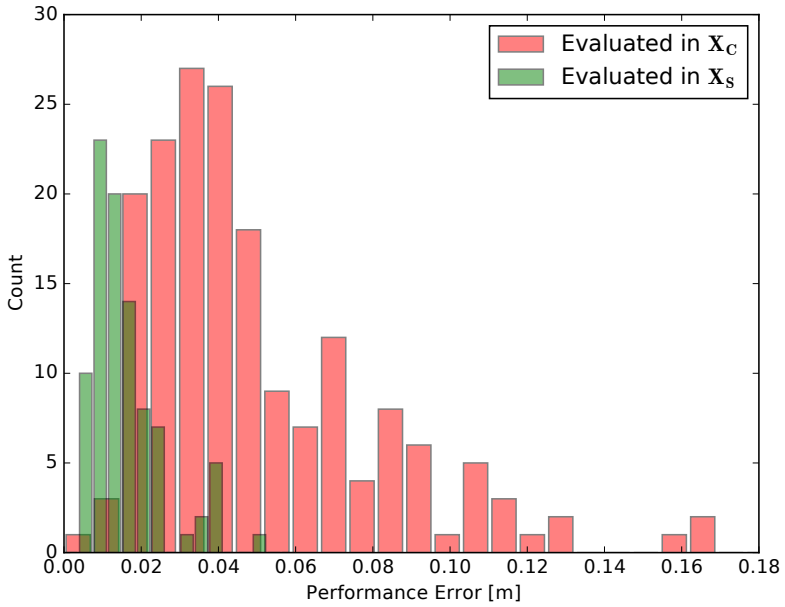


Figure 4.5.: Performance error distribution evaluated on the 179 goals of the convex goal space $\mathbf{X_C}$ used for IK learning in Figure 4.3, compared with the error distribution evaluated on the 79 goals of $\mathbf{X_S}$, as in Figure 4.6, where the cut goal space with outlier goals removed. The comparison shows that performance error has been significantly reduced after removing outlier goals, reaching an average error of 1.8 cm as shown in Figure 4.4, where most errors are distributed below 2 cm, and the worst error not more than 5 cm.

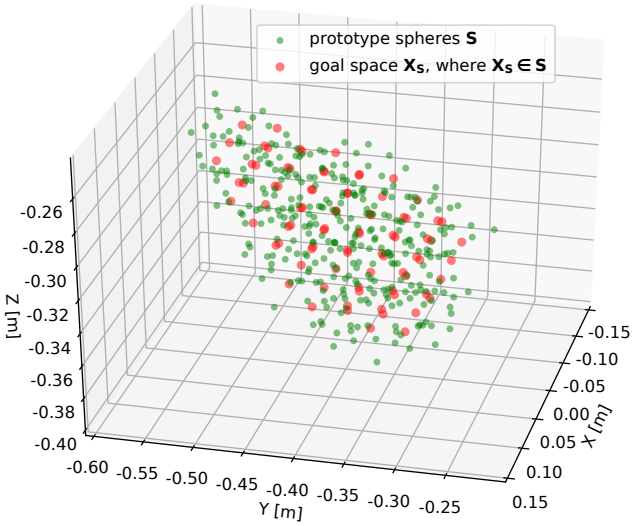


Figure 4.6.: Prototype spheres S (green) encapsulating the final goal space $\mathbf{X}_S$ (red) with outlier goals removed, which is later used for motor babbling. $\mathbf{X}_S$ is made with a similar set intersection operation, where the encapsulated space populated with learned prototype spheres $\mathbf{S}$ and the goal space $\mathbf{X_C}$ are intersected, i.e., $\mathbf{X}_S = \mathbf{S} \cap \mathbf{X}_C$. As shown in the figure, the number of goals has been reduced from 179 in $\mathbf{X}_C$ to 94 in $\mathbf{X}_S$

4.3 Quering Goal Space for Motor Babbling

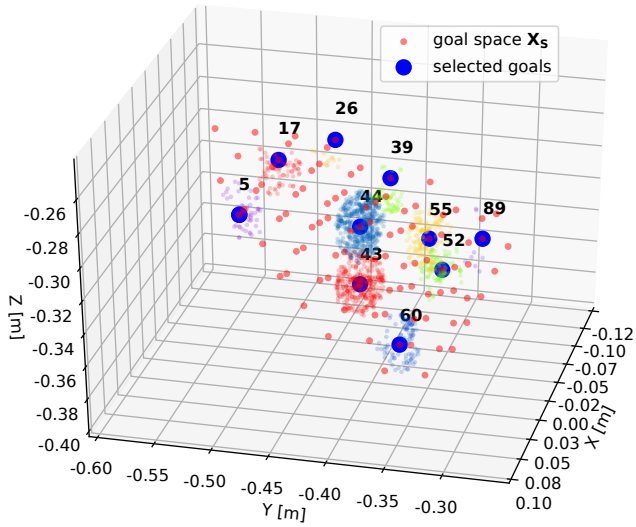


Figure 4.7.: 10 goal points for motor babbling in the final goal space $\mathbf{X_S}$ are evenly selected to show case the generality of local online motor babbling using CMA-ES. The color point clouds are the local samples within 2cm radius of the queried goals, generated during the goal babbling online IK learning process. These local data around the queried goal is then used to initialize the step-size σ for the CMA-ES trials.

We evenly selected 10 goals in the final goal space $\mathbf{X_S}$ to perform online motor babbling. The selected goals and their local samples within 2cm radius are shown in Figure 4.7. The goals are selected to show case the generality of querying any goal within the goal space for motor babbling. Around the edges, goal 26, 5, 89, 52 are chosen, and near the centroid home position, goal 44 and 39 are selected. The rest goals 17, 43, 55, and 60 are to populate the rest of the goal space. It can be expected and observed that more samples were generated near the home posture, since in online goal babbling the arm returns to (x_{home}, q_{home}) with probability p_{home}, whereas goals around the edges have only a few samples, such as goal 26 and 89.

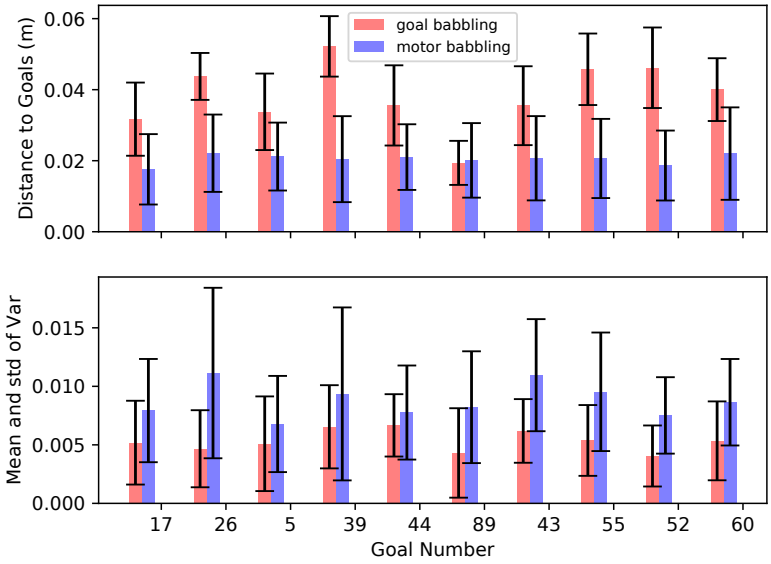


Figure 4.8.: Comparing the reaching error and muscle variability of directed goal babbling and local motor babbling using CMA-ES. The data is obtained by sampling 200 motor commands from the fitted GMM, which reproduces the learned local motor abundance for each queried goal as mentioned in Section 3.4. The motor commands are then fed to the robot for execution, and the end effector positions are recorded. The upper bar plot shows the average distance and standard deviation of the 200 trials to the queried goal, and the lower one shows the mean and standard deviation of pressure variances for all the PAMs. It can be observed that CMA-ES not only increased the means and standard deviations of all 24 muscle variances for the 10 queried goals, but the reaching error has also been reduced, meaning that motor abundance has been efficiently explored with varied muscle stiffness and synergies, while keeping the end effector position fixed for the queried goal.

For each selected goal, $N = 5$ trials of CMA-ES is performed as in Algorithm 1, where each trial takes on average 5 minutes experiment time on the robot. Muscle stiffness is then reproduced by first fitting the collected neighboring samples $\mathbf{Q_x}$ to the Gaussian mixture model, which serves as a baseline learned during goal babbling, followed by another experiment fitting $\mathbf{Q_{cma}}$ to the mixture model and the subsequent sampling. 200 samples from the mixture model is evaluated on the robot, the mean and standard deviation of the reaching error, and of pressure variance are plotted. As illustrated in Figure 4.8, CMA-ES outperforms the baseline in terms of both larger muscle pressure variance and smaller goal reaching error, where the average lies close to the 2cm prototype sphere radius. Since online goal babbling favors kinematic and direction efficiency by reducing motor redundancy, the sampled muscle pressure generally varies trivially compared to the ones generated from the CMA-ES GMM model, which expands the variance in search of global optimum while keeping the goal reaching accuracy. Due to non-stationary changes of the possible posture configurations $\mathbf{Q}$, the local neighboring samples $\mathbf{Q_x}$ no longer lead to a close position to the goal, however $\mathbf{Q_x}$ of the neighboring goals can be used for initializing the step-size σ, and initializing the mean vector m from $\mathbf{Q_{fb}}$, to adapt to non-stationary changes and maintain the goal reaching accuracy while performing motor babbling.

It can be observed that for goal 44, which is closest to the home position, the motor variance doesn't increase much as other queried goals. This is because every time the interpolated directed goal path comes across the centroid home region, goal 44 has a higher chance of collecting more samples q_t of varied motor configurations within the neighborhood. Nevertheless, CMA-ES still explores motor redundancy rather efficiently. As shown in Figure 4.9, the evolution trial expands the maximum and minimum standard deviation of the search, i.e., such that the optimum f^* is reached. After 5 such evolution trials, the sampled GMM data is used to estimate the covariance, compared with the covariance estimate from the baseline GMM data. As shown in 4.10, CMA-ES preserves the structure while enhancing the variance on the diagonal, while also discovers more correlation within different groups of muscles, which can be prominently observed on the robot in Figure 4.11.

4.4 Interpreting Muscle Abundance

The muscle pressure variability in the covariance encodes muscle abundance, which can be interpreted as muscle stiffness and static muscle synergies. Loosely speaking, muscle synergy is defined as a co-activation pattern of muscles in a certain

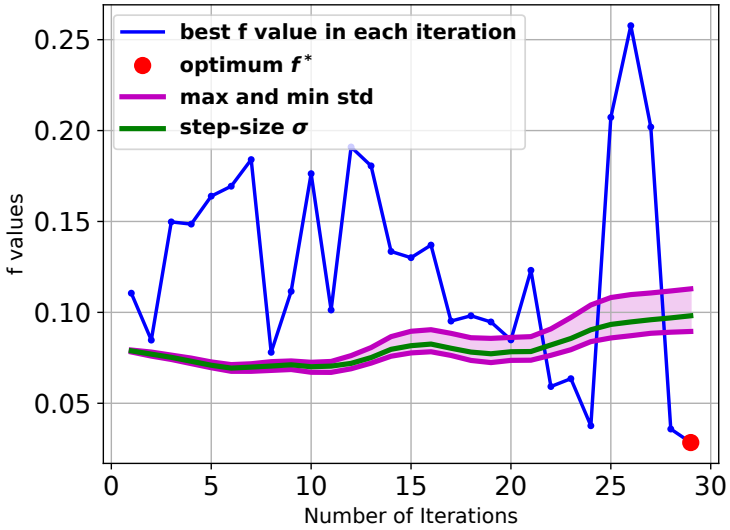


Figure 4.9.: One evolution trial for goal 44 is selected to demonstrate the evolution of the optimization, or in the case of motor babbling, to illustrate the effective way of exploring motor variability. It can be observed that the value of the objective function fluctuates while the maximum and minimum standard deviations of the step size keep on expanding, indicating the growing search distribution of the covariance matrix, until the defined optimum objective function value is found.

#	Muscle Name	Function
3	serratus anterior	pulls scapula forward
7	latissimus dorsi	rotates scapula downward
8	rear deltoid	abducts, flexes,
10	front deltoid	and extends
11	medial deltoid	the shoulder
15	biceps brachii	flexes and supinates the forearm
18	brachialis	flexes the elbow
19	pronator	pronates the hand
20	supinator	supinates the hand

Table 4.1.: Names and functions for the muscles of interest

movement from a single neural command signal [81]. It can be argued that muscle synergy is a way of kinetically constraining the redundant motor control of limited DoFs, or as neural strategies to handle the sensorimotor systems [82]. We claim no sides in the sensorimotor learning of humans, however, by constraining the end effector position of the musculoskeletal robot arm, the static muscle synergies and stiffness can be encoded in the covariance matrix and provide some useful insights. In Figure 4.10, muscles of high variances, namely muscle 3, 7, 8, 10, 11, 15, 18, 19, 20 are of particular interest, where muscle 7 and 8, 20 and 8 are highly negatively correlated. Inspired by the human's upper limb, the PAMs of the robot arm mimics the function of human arm muscles, as illustrated in Table 4.1. By fitting the data $\mathbf{Q}_{\mathbf{cma}}$ in the mixture models and subsequently applying sampling, we can observe the co-activation patterns of the muscles. As shown in Figure 4.11a and 4.11c, the upper limb first reaches goal 44 with a relaxed arm posture and a lowered adducted shoulder, whereas in Figure 4.11b and 4.11d the end effector position is maintained by stiffening the arm, lifting the extended shoulder, and pronating the hand. The negative correlation of muscle 7 and 8 can be interpreted as the coordination of extension and abduction, as well as the flexion and adduction of the shoulder. Muscle 8 and 20 coordinate shoulder abduction with a supinating hand, and by adducting shoulder while pronating the hand.

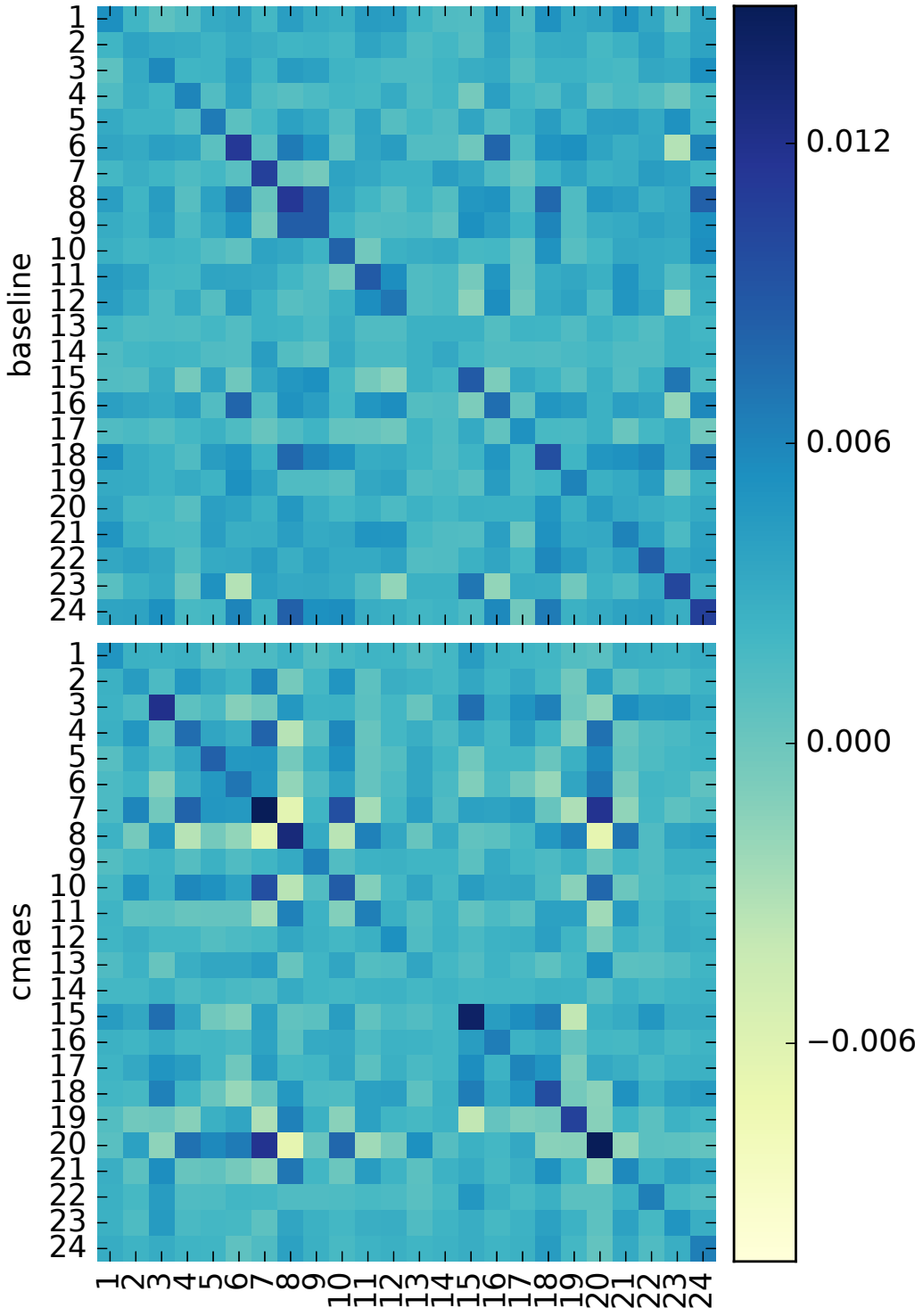
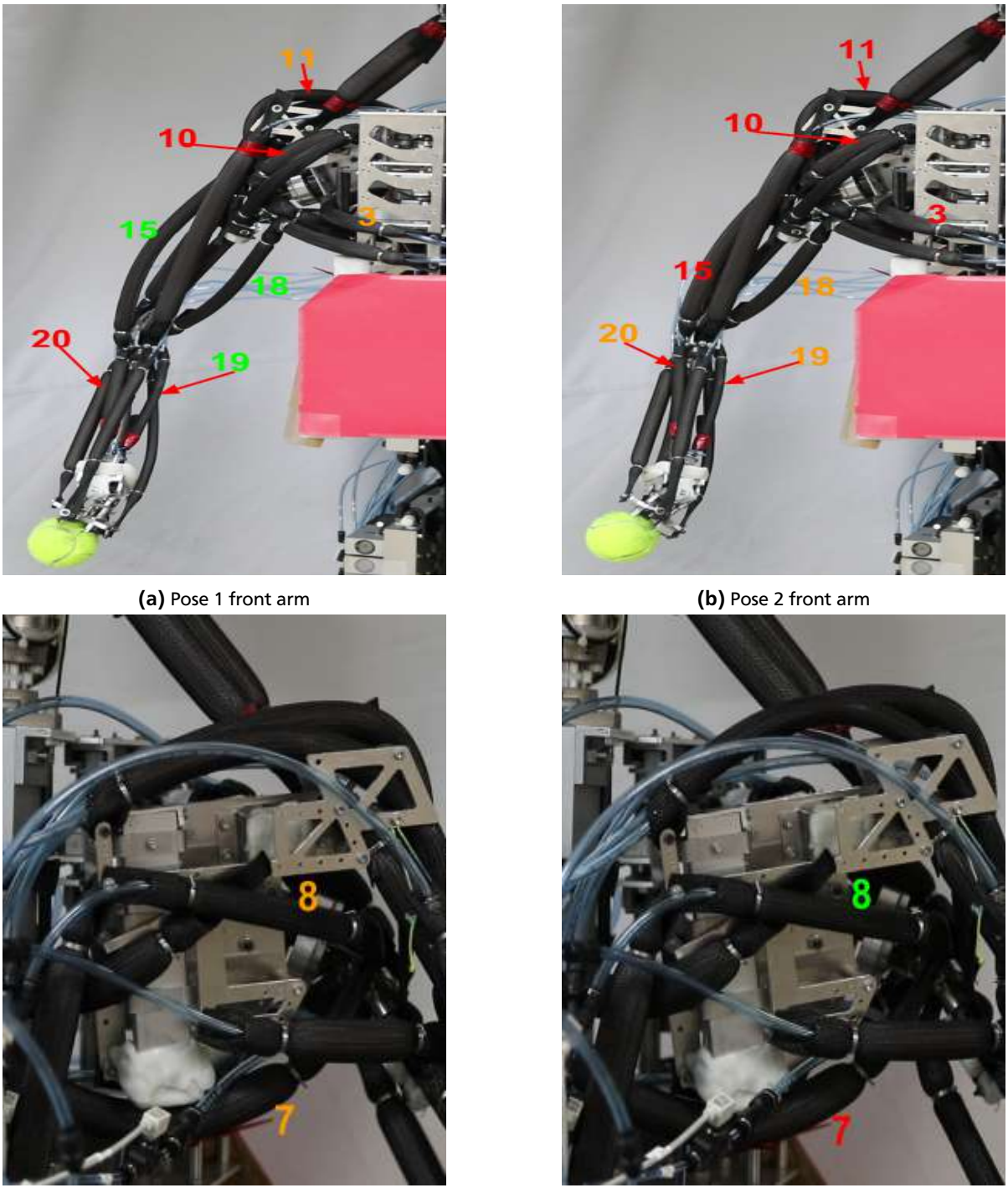


Figure 4.10.: Comparing baseline and CMA-ES covariances of goal 44 to illustrate the effective exploration for motor abundance. The reason of choosing goal 44 is that the baseline motor variability, which is encoded in the local samples generated throughout the whole goal babbling process, is already quite abundant, and that further exploration for more motor abundance would be much more difficult than the other queried goals. Since goal 44 is near the home position $\mathbf{x}_{\text{home}}$, various motor configurations has been collected when being swept across in the directed goal babbling process. Desipte the difficulty, we can still observe the enhancement of motor abundance encoded in the covariance matrix after motor babbling. As shown in the figure, the largest change of variance occurs at muscle pair (8,20), changing from 0.003 to -0.01, where the -0.01 covariance corresponds to the standard deviation of 0.1 MPa pressure change, consituting 25% of the PAM actuation range.



(a) Pose 1 front arm

(b) Pose 2 front arm

(c) Pose 1 back shoulder

(d) Pose 2 back shoulder

Figure 4.11.: Static muscle synergies reproduced from the learned motor abundance of goal 44, the labelled muscles are color coded in green (low), orange (medium), and red (high) to indicate the state of pressure actuation. A relaxed arm posture with a lowered shoulder can be observed in (a) and (c), whereas a stiffened arm with pronating hand and a lifted shoulder and can be observed in (b) and (d), while keep the end effector position fixed.

5 Conclusion

We have implemented directed goal babbling [61] to learn the inverse kinematics of a 10 DoF musculoskeletal robot arm actuated by 24 PAMs. We defined the unknown goal space by empirically sampling 2000 random postures and forcing a convex hull ready for learning, and post-processed the goal space to remove outlier goals. The result shows an average reaching error of 1.8 cm, where the reaching accuracy achievable by the robot is 1.2 cm. The simple heuristics and approximation of the goal space allows us to use directed goal babbling to learn a larger sensory space compared to a well-defined yet small partial task space, and promote more efficient mapping to the motor space compared to active exploration [60]. Nevertheless, learning with a forced convex goal space where the intrinsic task space is non-convex introduces outlier goals, which the corresponding directed babbling can be misleading. A future research direction of integrating directed goal babbling with active exploration could be of interest, where the goal space grid can be defined large enough to encapsulate the whole task space, and active exploration guided by the k-d tree splitting and progress logging can indicate the learned task space while still keeping the bootstrapping flavor of the inverse model.

We further extended the directed goal babbling method to local online motor babbling using CMA-ES to effectively search for more motor abundance. By initializing the evolution strategy with local samples generated from goal babbling, any point within the goal space can be queried for motor abundance. The idea is to intentionally initialize the mean vector of CMA-ES slightly away from the queried goal. By expanding covariance and setting the stop condition to meeting the set optimum of the objective function value, efficient motor babbling data can be generated locally around the queried goal with a few CMA-ES trials of different initializations from the neighboring goals. We evenly selected 10 goals within the goal space to showcase the generality of local online motor babbling. The results show that our proposed method has significantly increased the average muscle pressure variances, while keeping the end effector more stable and closer to the queried goals, compared with the goal babbling baseline. Even in the home position where motor abundance has already been well-explored, local motor babbling shows a maximum increased standard deviation of 0.1 MPa, constituting 25% of the muscle pressure actuation range. Our method also adapts to queried goals near the edges of the goal space where samples for initialization are sparse due to the uneasy posture of the robot arm around such goals.

By fitting Gaussian mixture models to the data collected using local motor babbling, the sampling of the GMMs can reproduce motor abundance in terms of muscle stiffness and muscle synergies encoded in the evolved single-mode covariance matrix. Muscle stiffness can be seen on the inflating and deflating muscles, and muscle synergies can be clearly observed in the covariance where variances and correlations are strong, as well as when GMM sampled postures are applied on the robot correspondingly. The bonus that comes with the encoded covariance and mixture models is that the queried motor abundance can be captured and reproduced by distributions, which enables the formulation of trials for reinforcement learning in future research, such as learning weight lifting with varied muscle stiffness, planning trajectories and learning dynamics using via-points and the locally queried motor abundance library.

Bibliography

[1] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, p. 467, 2015.

[2] D. Trivedi, C. D. Rahn, W. M. Kier, and I. D. Walker, "Soft robotics: Biological inspiration, state of the art, and future research," *Applied bionics and biomechanics*, vol. 5, no. 3, pp. 99–117, 2008.

[3] S. Li, J. J. Stampfli, H. J. Xu, E. Malkin, E. V. Diaz, D. Rus, and R. J. Wood, "A vacuum-driven origami "magic-ball" soft gripper," 2019.

[4] E. W. Hawkes, L. H. Blumenschein, J. D. Greer, and A. M. Okamura, "A soft robot that navigates its environment through growth," *Science Robotics*, vol. 2, no. 8, p. eaan3028, 2017.

[5] C. Majidi, "Soft robotics: a perspective—current trends and prospects for the future," *Soft Robotics*, vol. 1, no. 1, pp. 5–11, 2014.

[6] M. Calisti, M. Giorelli, G. Levy, B. Mazzolai, B. Hochner, C. Laschi, and P. Dario, "An octopus-bioinspired solution to movement and manipulation for soft robots," *Bioinspiration & biomimetics*, vol. 6, no. 3, p. 036002, 2011.

[7] C. Laschi, M. Cianchetti, B. Mazzolai, L. Margheri, M. Follador, and P. Dario, "Soft robot arm inspired by the octopus," *Advanced Robotics*, vol. 26, no. 7, pp. 709–727, 2012.

[8] C.-P. Chou and B. Hannaford, "Measurement and modeling of mckibben pneumatic artificial muscles," *IEEE Transactions on robotics and automation*, vol. 12, no. 1, pp. 90–102, 1996.

[9] K. Suzumori, S. Iikura, and H. Tanaka, "Applying a flexible microactuator to robotic mechanisms," *IEEE Control systems magazine*, vol. 12, no. 1, pp. 21–27, 1992.

[10] C. D. Onal, X. Chen, G. M. Whitesides, and D. Rus, "Soft mobile robots with on-board chemical pressure generation," in *Robotics Research*, pp. 525–540, Springer, 2017.

[11] R. F. Shepherd, F. Ilievski, W. Choi, S. A. Morin, A. A. Stokes, A. D. Mazzeo, X. Chen, M. Wang, and G. M. Whitesides, "Multigait soft robot," *Proceedings of the national academy of sciences*, vol. 108, no. 51, pp. 20400–20403, 2011.

[12] A. Grzesiak, R. Becker, and A. Verl, "The bionic handling assistant: a success story of additive manufacturing," *Assembly Automation*, vol. 31, no. 4, pp. 329–333, 2011.

[13] C. Paul, "Morphological computation: A basis for the analysis of morphology and control requirements," *Robotics and Autonomous Systems*, vol. 54, no. 8, pp. 619–630, 2006.

[14] H. Hauser, A. J. Ijspeert, R. M. Füchslin, R. Pfeifer, and W. Maass, "Towards a theoretical foundation for morphological computation with compliant bodies," *Biological cybernetics*, vol. 105, no. 5-6, pp. 355–370, 2011.

[15] R. Pfeifer, M. Lungarella, and F. Iida, "Self-organization, embodiment, and biologically inspired robotics," *science*, vol. 318, no. 5853, pp. 1088–1093, 2007.

[16] A. D. Marchese, C. D. Onal, and D. Rus, "Autonomous soft robotic fish capable of escape maneuvers using fluidic elastomer actuators," *Soft Robotics*, vol. 1, no. 1, pp. 75–87, 2014.

[17] C. Laschi, B. Mazzolai, V. Mattoli, M. Cianchetti, and P. Dario, "Design and development of a soft actuator for a robot inspired by the octopus arm," in *Experimental Robotics*, pp. 25–33, Springer, 2009.

[18] S. Seok, C. D. Onal, K.-J. Cho, R. J. Wood, D. Rus, and S. Kim, "Meshworm: a peristaltic soft robot with antagonistic nickel titanium coil actuators," *IEEE/ASME Transactions on mechatronics*, vol. 18, no. 5, pp. 1485–1497, 2013.

[19] Q. Zhao, K. Nakajima, H. Sumioka, H. Hauser, and R. Pfeifer, "Spine dynamics as a computational resource in spine-driven quadruped locomotion," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1445–1451, IEEE, 2013.

[20] K. Nakajima, H. Hauser, R. Kang, E. Guglielmino, D. G. Caldwell, and R. Pfeifer, "A soft body as a reservoir: case studies in a dynamic model of octopus-inspired soft robotic arm," *Frontiers in computational neuroscience*, vol. 7, p. 91, 2013.

[21] D. Lakatos, K. Ploeger, F. Loeffl, D. Seidel, F. Schmidt, T. Gumpert, F. John, T. Bertram, and A. Albu-Schäffer, "Dynamic locomotion gaits of a compliantly actuated quadruped with slip-like articulated legs embodied in the mechanical design," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3908–3915, 2018.

[22] R. Pfeifer, M. Lungarella, and F. Iida, "Self-organization, embodiment, and biologically inspired robotics," *science*, vol. 318, no. 5853, pp. 1088–1093, 2007.

[23] T. George Thuruthel, Y. Ansari, E. Falotico, and C. Laschi, "Control strategies for soft robotic manipulators: A survey," *Soft robotics*, vol. 5, no. 2, pp. 149–163, 2018.

[24] M. W. Hannan and I. D. Walker, "Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots," *Journal of robotic systems*, vol. 20, no. 2, pp. 45–63, 2003.

[25] M. Giorelli, F. Renda, G. Ferri, and C. Laschi, "A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5033–5039, IEEE, 2013.

[26] M. Giorelli, F. Renda, M. Calisti, A. Arienti, G. Ferri, and C. Laschi, "Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature," *IEEE Transactions on Robotics*, vol. 31, no. 4, pp. 823–834, 2015.

[27] M. Giorelli, F. Renda, M. Calisti, A. Arienti, G. Ferri, and C. Laschi, "Learning the inverse kinetics of an octopus-like manipulator in three-dimensional space," *Bioinspiration & biomimetics*, vol. 10, no. 3, p. 035006, 2015.

[28] M. Rolf and J. J. Steil, "Efficient exploratory learning of inverse kinematics on a bionic elephant trunk," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 6, pp. 1147–1160, 2014.

[29] M. C. Yip and D. B. Camarillo, "Model-less feedback control of continuum manipulators in constrained environments," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 880–889, 2014.

[30] Y. Ansari, M. Manti, E. Falotico, M. Cianchetti, and C. Laschi, "Multiobjective optimization for stiffness and position control in a soft robot arm module," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 108–115, 2018.

[31] P. Qi, C. Liu, A. Ataka, H.-K. Lam, and K. Althoefer, "Kinematic control of continuum manipulators using a fuzzy-model-based approach," *IEEE Transactions on Industrial Electronics*, vol. 63, no. 8, pp. 5022–5035, 2016.

[32] M. S. Malekzadeh, S. Calinon, D. Bruno, and D. G. Caldwell, "Learning by imitation with the stiff-flop surgical robot: a biomimetic approach inspired by octopus movements," *Robotics and Biomimetics*, vol. 1, no. 1, p. 13, 2014.

[33] T. G. Thuruthel, E. Falotico, M. Cianchetti, and C. Laschi, "Learning global inverse kinematics solutions for a continuum robot," in *Symposium on Robot Design, Dynamics and Control*, pp. 47–54, Springer, 2016.

[34] T. Thuruthel, E. Falotico, M. Cianchetti, F. Renda, and C. Laschi, "Learning global inverse statics solution for a redundant soft robot," in *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics*, vol. 2, pp. 303–310, 2016.

[35] A. Kapadia and I. D. Walker, "Task-space control of extensible continuum manipulators," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1087–1092, IEEE, 2011.

[36] A. D. Kapadia, I. D. Walker, D. M. Dawson, and E. Tatlicioglu, "A model-based sliding mode controller for extensible continuum robots," in *Proceedings of the 9th WSEAS international conference on Signal processing, robotics and automation*, pp. 113–120, World Scientific and Engineering Academy and Society (WSEAS), 2010.

[37] A. D. Kapadia, K. E. Fry, and I. D. Walker, "Empirical investigation of closed-loop control of extensible continuum manipulators," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 329–335, IEEE, 2014.

[38] V. Falkenhahn, A. Hildebrandt, and O. Sawodny, "Trajectory optimization of pneumatically actuated, redundant continuum manipulators," in *2014 American Control Conference*, pp. 4008–4013, IEEE, 2014.

[39] A. D. Marchese, R. Tedrake, and D. Rus, "Dynamics and trajectory optimization for a soft spatial fluidic elastomer manipulator," *The International Journal of Robotics Research*, vol. 35, no. 8, pp. 1000–1019, 2016.

[40] V. Falkenhahn, A. Hildebrandt, R. Neumann, and O. Sawodny, "Model-based feedforward position control of constant curvature continuum robots using feedback linearization," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 762–767, IEEE, 2015.

[41] V. Falkenhahn, A. Hildebrandt, R. Neumann, and O. Sawodny, "Dynamic control of the bionic handling assistant," *IEEE/ASME Transactions on Mechatronics*, vol. 22, no. 1, pp. 6–17, 2017.

[42] D. Braganza, D. M. Dawson, I. D. Walker, and N. Nath, "A neural network controller for continuum robots," *IEEE transactions on robotics*, vol. 23, no. 6, pp. 1270–1277, 2007.

[43] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *ICML*, 2014.

[44] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, "Learning dynamic models for open loop predictive control of soft robotic manipulators," *Bioinspiration & biomimetics*, vol. 12, no. 6, p. 066003, 2017.

[45] M. Inaba, I. Mizuuchi, R. Tajima, T. Yoshikai, D. Sato, K. Nagashima, and H. Inoue, "Building spined muscle-tendon humanoid," in *Robotics Research*, pp. 113–127, Springer, 2003.

[46] I. Boblan, R. Bannasch, H. Schwenk, F. Prietzel, L. Miertsch, and A. Schulz, "A human-like robot hand and arm with fluidic muscles: Biologically inspired construction and functionality," in *Embodied Artificial Intelligence*, pp. 160–179, Springer, 2004.

[47] Y. Asano, T. Kozuki, S. Ookubo, M. Kawamura, S. Nakashima, T. Katayama, I. Yanokura, T. Hirose, K. Kawaharazuka, S. Makino, *et al.*, "Human mimetic musculoskeletal humanoid kengoro toward real world physically interactive actions," in *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pp. 876–883, IEEE, 2016.

[48] Y. Asano, T. Kozuki, S. Ookubo, K. Kawasaki, T. Shirai, K. Kimura, K. Okada, and M. Inaba, "A sensor-driver integrated muscle module with high-tension measurability and flexibility for tendon-driven robots," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5960–5965, IEEE, 2015.

[49] R. Niiyama, K. Kakitani, and Y. Kuniyoshi, "Learning to jump with a musculoskeletal robot using a sparse coding of activation," in *Proc. ICRA 2009 Workshop on Approaches to Sensorimotor Learning on Humanoid Robots*, pp. 30–31, 2009.

[50] R. Niiyama, S. Nishikawa, and Y. Kuniyoshi, "Athlete robot with applied human muscle activation patterns for bipedal running," in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pp. 498–503, IEEE, 2010.

[51] K. Hosoda, H. Saito, and S. Ikemoto, "Muscular-skeletal humanoid robot for body image construction," in *2016 International Symposium on Micro-NanoMechatronics and Human Science (MHS)*, pp. 1–3, IEEE, 2016.

[52] S. Ikemoto, F. Kannou, and K. Hosoda, "Humanlike shoulder complex for musculoskeletal robot arms," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4892–4897, IEEE, 2012.

[53] S. Ikemoto, Y. Nishigori, and K. Hosoda, "Direct teaching method for musculoskeletal robots driven by pneumatic artificial muscles," in *2012 IEEE International Conference on Robotics and Automation*, pp. 3185–3191, IEEE, 2012.

[54] Y. Duan, S. Ikemoto, and K. Hosoda, "Optimal feedback control based on analytical linear models extracted from neural networks trained for nonlinear systems," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8689–8694, IEEE, 2018.

[55] C. Hartmann, J. Boedecker, O. Obst, S. Ikemoto, and M. Asada, *Real-time inverse dynamics learning for musculoskeletal robots based on echo state gaussian process regression*. 2012.

[56] P. Gaudiano and S. Grossberg, "Vector associative maps: Unsupervised real-time error-based learning and control of movement trajectories," *Neural networks*, vol. 4, no. 2, pp. 147–183, 1991.

[57] Y. Demiris and A. Dearden, "From motor babbling to hierarchical learning by imitation: a robot developmental pathway," 2005.

[58] A. D'Souza, S. Vijayakumar, and S. Schaal, "Learning inverse kinematics," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, vol. 1, pp. 298–303, IEEE, 2001.

[59] M. Rolf, J. J. Steil, and M. Gienger, "Goal babbling permits direct learning of inverse kinematics," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 3, pp. 216–229, 2010.

[60] A. Baranes and P.-Y. Oudeyer, "Active learning of inverse models with intrinsically motivated goal exploration in robots," *Robotics and Autonomous Systems*, vol. 61, no. 1, pp. 49–73, 2013.

[61] M. Rolf, J. J. Steil, and M. Gienger, "Online goal babbling for rapid bootstrapping of inverse models in high dimensions," in *Development and Learning (ICDL), 2011 IEEE International Conference on*, vol. 2, pp. 1–8, IEEE, 2011.

[62] A. Hitzmann, H. Masuda, S. Ikemoto, and K. Hosoda, "Anthropomorphic musculoskeletal 10 degrees-of-freedom robot arm driven by pneumatic artificial muscles," *Advanced Robotics*, vol. 32, no. 15, pp. 865–878, 2018.

[63] M. Latash, "There is no motor redundancy in human movements. there is motor abundance," 2000.

[64] M. L. Latash, "The bliss (not the problem) of motor abundance (not redundancy)," *Experimental brain research*, vol. 217, no. 1, pp. 1–5, 2012.

[65] P. Singh, S. Jana, A. Ghosal, and A. Murthy, "Exploration of joint redundancy but not task space variability facilitates supervised motor learning," *Proceedings of the National Academy of Sciences*, vol. 113, no. 50, pp. 14414–14419, 2016.

[66] N. Hansen, "The cma evolution strategy: A tutorial," *arXiv preprint arXiv:1604.00772*, 2016.

[67] S. Ikemoto, F. Kannou, and K. Hosoda, "Humanlike shoulder complex for musculoskeletal robot arms," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4892–4897, IEEE, 2012.

[68] F. Daerden and D. Lefeber, "Pneumatic artificial muscles: actuators for robotics and automation," *European journal of mechanical and environmental engineering*, vol. 47, no. 1, pp. 11–21, 2002.

[69] A. R. Tsouroukdissian, "ros-control: An overview," *ROSCon*, 2014.

[70] K. H. Hiroaki Masuda, Arne Hitzmann and S. Ikemoto, *Common dimensional autoencoder for learning redundant muscle-posture mappings of complex musculoskeletal robots*. Adaptive Robotics Laboratory, School of Engineering Science, Osaka University, 2019.

[71] S. Dorodnicov, "Intel realsense cross platform api." `https://github.com/IntelRealSense/librealsense/tree/v1.12.1`, 2016.

[72] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[73] C. B. Barber, D. P. Dobkin, D. P. Dobkin, and H. Huhdanpaa, "The quickhull algorithm for convex hulls," *ACM Transactions on Mathematical Software (TOMS)*, vol. 22, no. 4, pp. 469–483, 1996.

[74] N. Hansen, S. D. Müller, and P. Koumoutsakos, "Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es)," *Evolutionary computation*, vol. 11, no. 1, pp. 1–18, 2003.

[75] H.-G. Beyer and K. Deb, "On self-adaptive features in real-parameter evolutionary algorithms," *IEEE Transactions on evolutionary computation*, vol. 5, no. 3, pp. 250–270, 2001.

[76] G. Collange, N. Delattre, N. Hansen, I. Quinquis, and M. Schoenauer, "Multidisciplinary optimization in the design of future space launchers," *Multidisciplinary Design Optimization in Computational Mechanics*, pp. 459–468, 2013.

[77] T. Glasmachers, T. Schaul, S. Yi, D. Wierstra, and J. Schmidhuber, "Exponential natural evolution strategies," in *Proceedings of the 12th annual conference on Genetic and evolutionary computation*, pp. 393–400, ACM, 2010.

[78] N. Hansen, "Verallgemeinerte individuelle schrittweitenregelung in der evolutionsstrategie," *Mensch & Buch Verlag, Berlin*, 1998.

[79] N. Hansen, Y. Akimoto, and P. Baudis, "CMA-ES/pycma on Github." Zenodo, DOI:10.5281/zenodo.2559634, Feb. 2019.

[80] C. M. Bishop, *Pattern recognition and machine learning*. springer, 2006.

[81] G. Torres-Oviedo, J. M. Macpherson, and L. H. Ting, "Muscle synergy organization is robust across a variety of postural perturbations," *Journal of neurophysiology*, 2006.

[82] M. C. Tresch and A. Jarc, "The case for and against muscle synergies," *Current opinion in neurobiology*, vol. 19, no. 6, pp. 601–607, 2009.

[83] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.

[84] R. P. Paul, *Robot manipulators: mathematics, programming, and control: the computer control of robot manipulators*. Richard Paul, 1981.

[85] K. Hosoda, S. Sekimoto, Y. Nishigori, S. Takamuku, and S. Ikemoto, "Anthropomorphic muscular–skeletal robotic upper limb for understanding embodied intelligence," *Advanced Robotics*, vol. 26, no. 7, pp. 729–744, 2012.

[86] R. Niiyama, S. Nishikawa, and Y. Kuniyoshi, "Biomechanical approach to open-loop bipedal running with a musculoskeletal athlete robot," *Advanced Robotics*, vol. 26, no. 3-4, pp. 383–398, 2012.

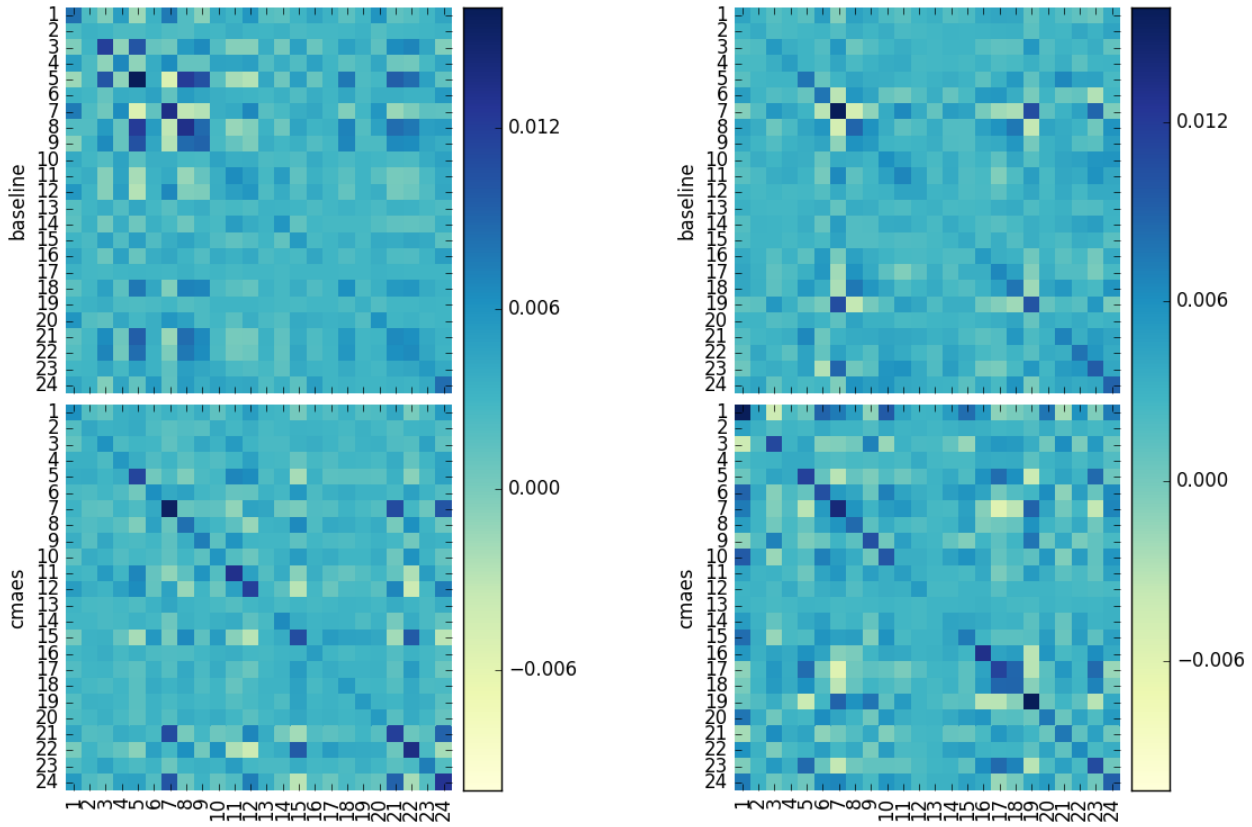
[87] D. M. Wolpert, R. C. Miall, and M. Kawato, "Internal models in the cerebellum," *Trends in cognitive sciences*, vol. 2, no. 9, pp. 338–347, 1998.

[88] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proceedings of IEEE international conference on evolutionary computation*, pp. 312–317, IEEE, 1996.

A Appendix

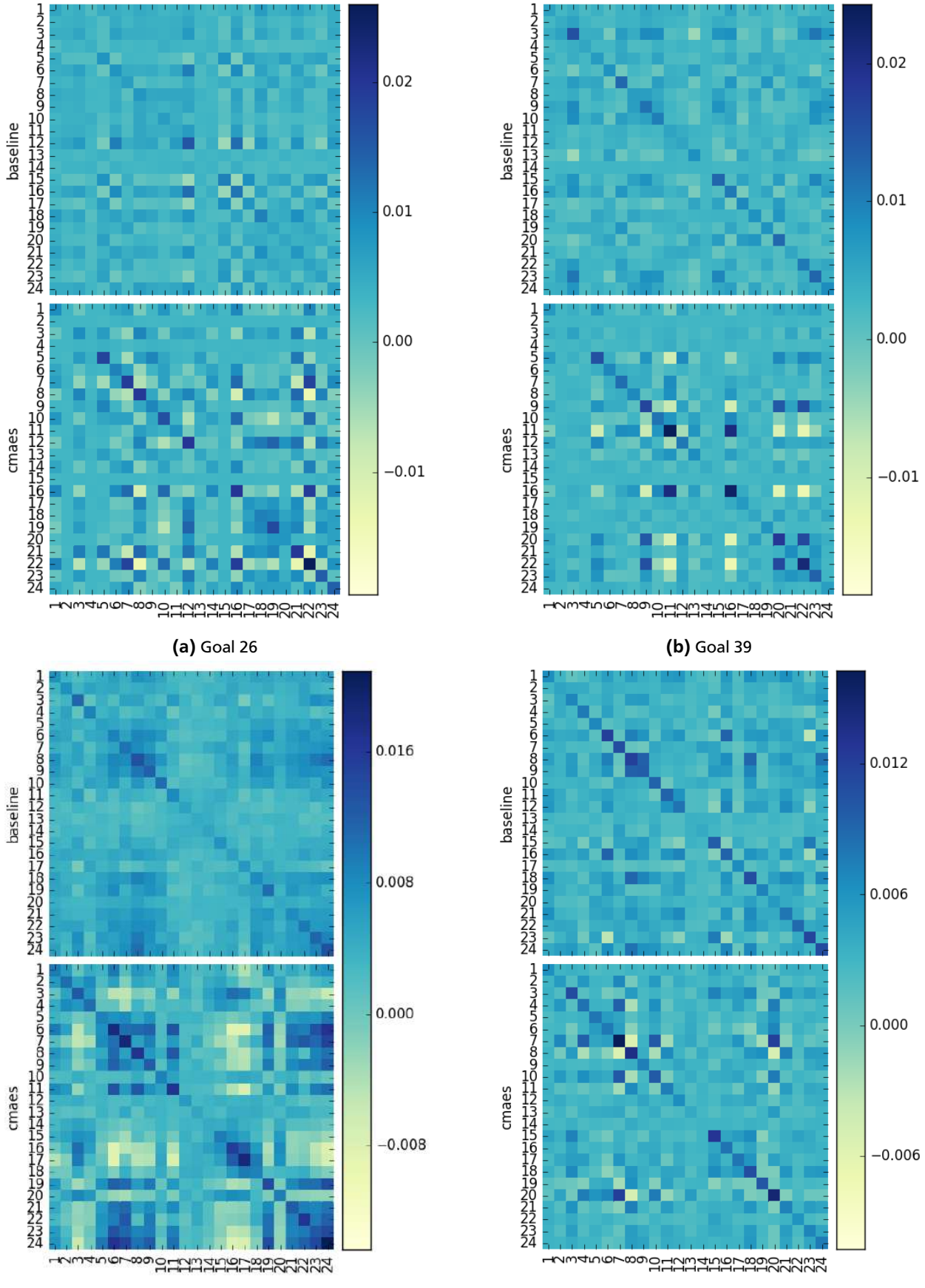
A.1 Evaluation of the ten queried goals for motor abundance

In Figure 4.7, 10 goals are selected to show case the generality of local online motor babbling using CMA-ES. The learned motor babbling results for all of the 10 goals are illustrated below, where the comparisons of the final adapted covariance matrix in CMA-ES and the one estimated from the local neighbouring samples generated throughout the goal babbling process. We can see that for all goals the covariance matrix learned from motor babbling has been enhanced, namely increased variances and increased correlations, including negative ones.



(a) Goal 5 **(b)** Goal 17

Figure A.1.: Evaluations of goal 5 and 17: Covariance comparison of local online motor babbling using CMA-ES with the baseline using directed online goal babbling



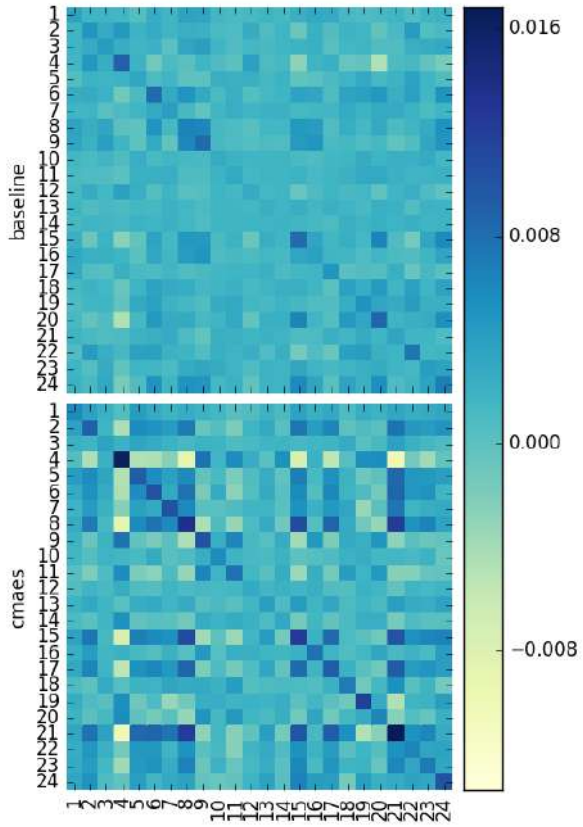
(a) Goal 26

(b) Goal 39

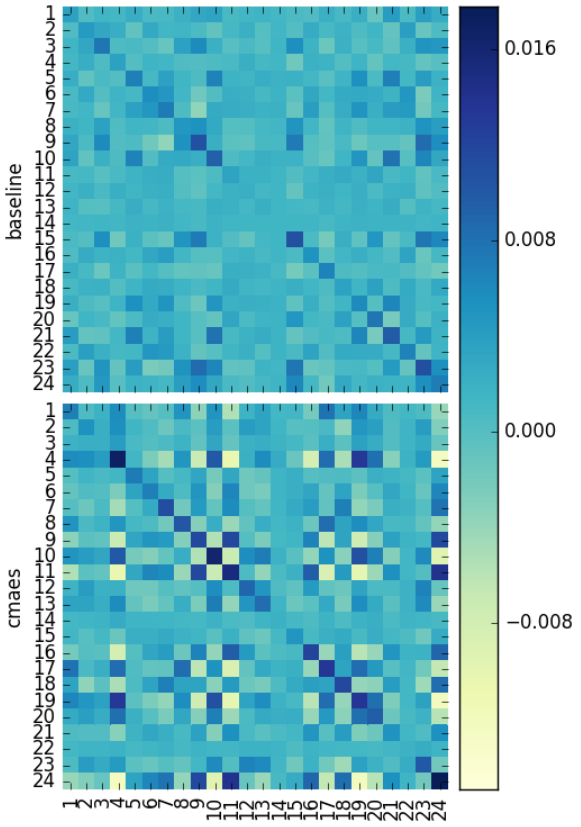
(c) Goal 43

(d) Goal 44

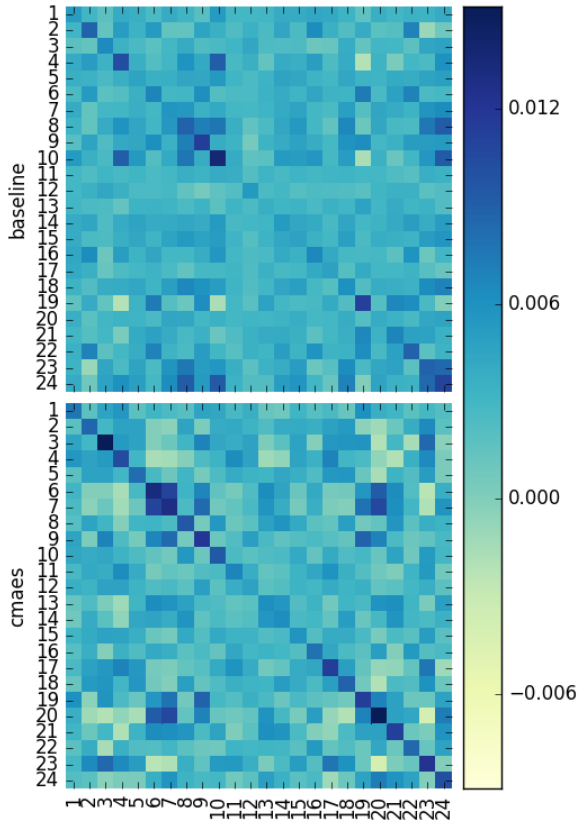
Figure A.2.: Evaluations of goal 26, 39, 43, and 44: Covariance comparison of local online motor babbling using CMA-ES with the baseline using directed online goal babbling



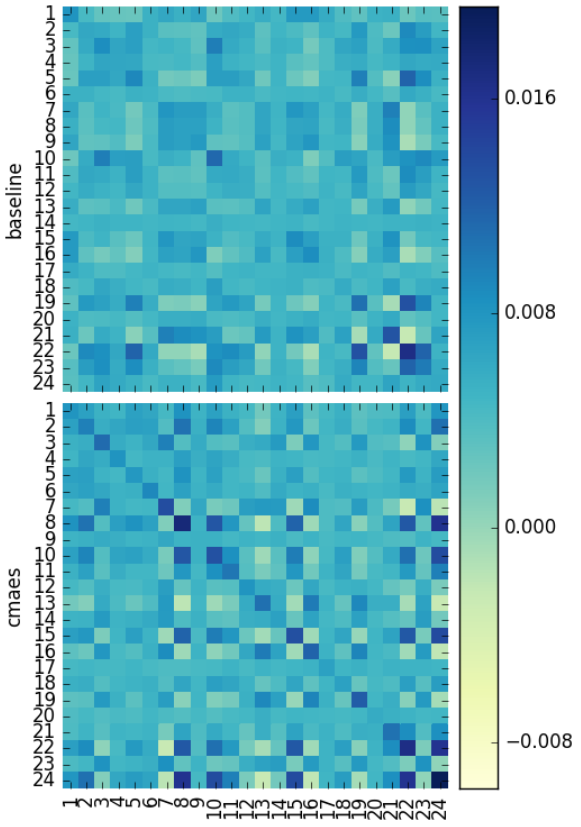
(a) Goal 52



(b) Goal 55



(c) Goal 60



(d) Goal 89

Figure A.3.: Evaluations of goal 52, 55, 60, and 89: Covariance comparison of local online motor babbling using CMA-ES with the baseline using directed online goal babbling