TÉCNICO LISBOA

Active Robot Learning for Efficient Body Schema Online Adaptation

Gonçalo Carvalho e Cunha

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Advisors/Supervisors:
Prof. Alexandre José Malheiro Bernardino
Prof. Plinio Moreno López

Examination Committee

Chairperson: Prof. João Fernando Cardoso Silva Sequeira
Advisor: Prof. Plinio Moreno López
Examiner: Prof. Ruben Martinez-Cantin

January 2021

Declaration

I declare that this document is an original work of my own authorship and that it fulfils all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa

Acknowledgments

I would like to give my thanks to my thesis supervisors Alexandre Bernardino and Plínio Moreno along with Pedro Vicente and Ricardo Ribeiro for guiding me and for being available to listen to any doubts and problems I faced throughout the making of this thesis. Thank you to my family and, especially, to my parents, Norma Carvalho and Fernando Cunha, for raising me, for always making sure I had everything I needed and for the healthy household we always lived in. A very special thank you to Luna França for all the love, support, good times we spent together from High-School to Masters of Science and for showing me how to not be lazy. An honourable mention to Pentes for always questioning my intelligence, pushing me to prove him wrong. Thank you to André Meneses, Edgar Pasadas, Francisco Mendes and Ivan Carapinha for all the help throughout the degree and for the good times spent outside classes. Finally, thank you to Ricardo Coutinho and Pedro Meneses for our tactical adventures with André Meneses to take our minds off our degrees.

Abstract

Humanoid robots have complex bodies and kinematic chains with several Degrees-of-Freedom (DoF) which are difficult to model. Learning the parameters of a kinematic model can be achieved by observing the pose of the robot links during prospective motions and minimising the prediction errors. This thesis proposes a movement efficient approach for estimating online the body-schema of a humanoid robot arm in the form of Denavit-Hartenberg (DH) parameters. A cost-sensitive active learning approach based on the A-Optimality criterion is used to select optimal joint configurations. The chosen joint configurations simultaneously minimise the error in the estimation of the body schema and minimise the movement between samples. This reduces energy consumption, along with mechanical fatigue and wear, while not compromising the learning accuracy. The work was implemented in a simulation environment, using the 7 DoF arm of the iCub robot simulator. The hand pose is measured with a single camera via markers placed in the palm and back of the robot's hand. It is proposed a pose dependent noise model to reduce the impact of non-uniform measurement noise in the system and a non-parametric occlusion model is proposed to avoid choosing joint configurations where the markers are not visible, thus preventing worthless attempts. The results show cost-sensitive active learning has similar accuracy to the standard active learning approach, while reducing in about half the executed movement.

Keywords: cost-sensitive active learning, body-schema, calibration, humanoid, robotics

Resumo

Os robôs humanóides possuem corpos complexos e cadeias cinemáticas com vários graus de liberdade que são difíceis de modelar. É possível aprender os parâmetros de um modelo cinemático ao observar a pose dos corpos rígidos da cadeia durante movimentos prospetivos e minimizando os erros de predição. Esta tese propõe um método para estimar a estrutura de um braço robótico humanóide, na forma de parâmetros de Denavit-Hartenberg, de forma eficiente (que permita reduzir o consumo de energia e o desgaste mecânico). É usada uma abordagem de aprendizagem ativa, baseada no critério *A-Optimality*, que considera a eficiência de movimentos ao selecionar ângulos ótimos para as articulações. Estes ângulos são informativos para a estimação e minimizam o movimento efetuado durante o processo de calibração, simultaneamente. A implementação foi feita em ambiente de simulação, utilizando o braço robótico com 7 graus de liberdade do iCub Simulator. A pose da mão é medida com uma única câmera usando marcadores ArUco, colocados na palma e nas costas da mão. É proposto um modelo de ruído *pose-dependent*, para reduzir o impacto do ruído de medição não uniforme, e um modelo de oclusão não paramétrico, para evitar a escolha de configurações do braço em que os marcadores não estão visíveis e reduzir o número de tentativas de amostragem falhadas. Os resultados mostram que a abordagem de aprendizagem ativa com consideração pela eficiência dos movimentos tem um desempenho semelhante a uma abordagem de aprendizagem ativa convencional e reduz significativamente o movimento efetuado durante a calibração.

Keywords: aprendizagem ativa *cost-sensitive*, *body-schema*, calibração, humanóide, robótica

Contents

List of Tables

List of Figures

Acronyms

AL Active Learning. 28–30, 34, 37

CCSAL Constrained Cost-Sensitive Active Learning. 28, 30, 34, 39, 41

CSAL Cost-Sensitive Active Learning. 30

DH Denavit-Hartenberg. xiii, 4, 6, 8, 10, 11, 23, 25–28, 41

EKF Extended Kalman Filter. 6, 8–13, 15, 18, 22, 23, 25–28, 30, 37, 38, 41, 42

R Random. 28–30, 32

UCSAL Unconstrained Cost-Sensitive Active Learning. 28, 30, 34, 39, 41

YARP Yet Another Robot Platform. 23, 26

Chapter 1

Introduction

Robots are generally deployed to have a fixed behaviour in low uncertainty environments (e.g. factories) and they rely on their body-schema to accomplish many of their tasks. [1] defines body-schema as an ”implicit knowledge structure that encodes the body’s form, the constraints on how the body’s parts can be configured, and the consequences of this configuration on touch, vision and movement” and it is how the term is used in this thesis. In [2] several approaches are shown on how to represent the body-schema, as well as an interesting discussion about what is still missing in modern robots to be as robust as animals or human beings. Generally, robots require expensive and time consuming calibrations performed by experts since body parts: i) may not have the exact dimensions they should and ii) they may be affected by material wear and fatigue. Even with these offline calibration procedures, the presence of abnormal conditions or disturbances, such as changes in room temperature causing materials to expand or contract, may affect their performance if they lack the ability to adapt their models online (self-calibrate).

Humans are able to learn their own body-schema, which is a process started at a very young age, as stated in [3], by using information from our senses. This continuous learning and adaptation is what allows humans to be able to adapt to different conditions, and producing robots possessing similar behaviours is essential for many areas of robotics where an extended period of autonomous behaviour is required, such as exploration robots in inaccessible areas, rescue robots, social robots and for human-robot cooperation.

Online body-schema adaptation requires the use of machine learning algorithms, a subject which has been evolving constantly, allowing machines to perform previously unimaginable tasks. These algorithms sometimes require significant amounts of data, often provided by humans or, in the context of body-schema learning, by performing random movements and obtaining measurements. This may result in acquiring irrelevant data or data which does not improve the models, wasting human, computing and time resources. Active learning is a sub-field of machine learning which aims to reduce the amount of training data required to build a model, with a certain precision. This is done by having the learning algorithm decide which data it wants to label or sample next. A general introduction for this area of research can be found in [4]. Cost-sensitive active learning is a concept also explored in [4]. The main idea is that the learning task may be associated with other costs which do not decrease necessarily if the amount of training data reduces. As an example, robots designed to perform chemical experiments, such as [5], use optimisation algorithms to decide the next experiment to perform based on the learning potential, as well as the monetary costs. Active learning has been used in several works and, empirically, seems to succeed. In the context of robotics and body-schema learning, requiring less data means the robot is able to adapt faster to whichever unpredictable conditions it has to face.

The calibration problem consists of the estimation of a set of parameters associated with the arm’s

physical characteristics. The parameter estimation (i.e. calibration routine) requires samples that are obtained from arm movement.

The proposed calibration routine should make use of samples from the position and orientation of the hand, i.e. the hand pose, while knowing the readings of proprioceptive sensors (joint encoders). An active learning approach should be used to optimise the calibration routine. It should choose the best joint configurations in order to reduce the number of samples, as well as reduce the amount of movement needed which would improve time and energy efficiency as a consequence.

The proposed method will be compared with random selection of values for the joints and with a conventional active learning approach, which only aims to reduce the number of samples. The comparison will be made by assessing the ability of the methods to reduce error in the body-schema, the number of samples needed and the amount of movement needed.

1.1 Document Outline

The structure of the document is the following:

- Chapter 2 presents the state of the art work in robot calibration, robot active learning and cost-sensitive active learning and the contributions of this thesis.
- In Chapter 3, theory and methods required for achieving the goals of this thesis are presented.
- Chapter 4 mentions some aspects regarding the implementation of the methods, such as used software, and information regarding the robot simulator.
- In Chapter 5 the obtained results are shown and analysed.
- Chapter 6 contains the conclusions and suggestions for future work.

Chapter 2

Related Work and Expected Contributions

Recent works have succeeded in employing different strategies for body schema adaptation. [6] achieved this by simulating how the robot hand would appear in the robot camera images, based on the body schema and the proprioceptive information, and comparing the simulation with the actual camera images. Sequential Monte Carlo techniques were used to feed a particle-based Bayesian estimation method to estimate the parameters of the body schema. [7] uses known planar surfaces to estimate the robot's model's inaccuracies, using an Extended Kalman Filter for the parameter estimation. In [8], an automated calibration method is proposed relying on redundant information from multiple sensors, more concretely using self touch and self observation. Their results showed how using redundant information is superior to using only one kinematic chain, since fewer samples were needed to achieve similar performance, and it is superior in terms of observability. All these works successfully adapt their body models to account for the robot's body errors, but they fail at choosing the most informative samples to do so. Using active learning could reduce the number of samples needed as well as improve the achieved results.

Active learning methods have been employed and have shown empirical success in multiple areas of robotics. In [9], active Bayesian perception is used to guide the exploration of the unknown contour of an object, based on previous sensory data, showing clear improvements when compared to passive perception. The follow up work [10] again uses active Bayesian perception, this time guiding the exploration of an object by a robot hand and achieving less error in object classification. In the context of grasping, [11] used an active deep learning approach to plan multi-fingered grasps and showed that fewer samples were needed, comparing to a passive supervised learning method. [12] uses active learning for learning a model, based on Gaussian Mixture Models, of a virtual musical instrument in order to be able to imitate a music performance. They showed the robot was able to acquire the required knowledge faster than if using a random exploration strategy. In the context of body-schema learning, [13] used active learning to estimate a kinematic model of a serial robot, by selecting optimal joint configurations and end-effector observations. The learning process is done online using Recursive Least Squares estimation. In [14], an intrinsically motivated goal exploration mechanism is proposed, allowing active learning of inverse models in high-dimensional redundant robots and showing that exploration in the task space can be a lot faster than exploration in the actuator space for learning inverse models in redundant robots. All these works have shown the advantages of using active learning, since less iterations of the learning algorithms were needed to achieve a certain error threshold. However, the main focus was to minimise the number of samples, instead of focusing on minimising the actual effort needed.

In [15], a novel criterion is proposed for active touch point selection for fast object shape estimation which considers both uncertainty and movement cost to touch. Doing this also decreased execution time, by reducing the time spent moving instead of computing. A similar technique is used in [16] for shape estimation of 3-dimensional objects, by proposing an exploration algorithm, choosing the next best touch target maximising the estimated information gain and minimising the expected costs of exploration actions. These works support the use of cost-sensitive active learning since they were able to minimise the accumulated path length needed for accurate estimation with low impact on the number of touches necessary. This is similar to what is desired for this thesis, since the cost of an exploration action in the calibration routine is also related to the required movement to perform such action.

2.1 Expected Contributions

This thesis aims to create a calibration routine which can be performed autonomously by the robot, with no human intervention, using active learning for sampling and movement efficiency. The purpose of the calibration routine is for the robot to improve the knowledge about its own body-schema. Recall that this consists in estimating a set of parameters associated with the physical properties of its body, such as the length of the forearm or the length of the upper arm, called the DH parameters. The estimation of these parameters will be done using observations of the pose (position and orientation) of the hand, using visual input. This will be tested using the iCub arm, shown in Fig. 2.1, which has seven rotational joints.

Similarly to [15] and [16], we argue that using active learning to reduce the number of samples taken may not be the best approach, since some of the best samples may require unnecessary long movements, increasing execution time and energy spent. The main contribution of this work is a cost-sensitive active learning approach for body-schema learning, which chooses the best joint angles to sample the hand pose attempting to reduce the number of samples required and the required movement. The proposed calibration routine is composed of the key steps shown in Figure 2.2.

Using visual input to obtain samples of the hand pose comes with additional challenges to which solutions are also proposed in this thesis. The first challenge is occlusion. Many arm joint configurations may lead the hand to a position or orientation which makes it undetectable by the cameras, i.e. it may be hidden behind the robot's head. To address this issue, a method is used to learn the likelihood of observing the hand at a given joint configuration, using information from the previous attempts on other joint configurations. The second challenge is heteroscedastic measurement error, in other words, measurement error is not uniform for every sample. This is due to the way the hand pose is detected in the camera images and image resolution limitations. This is relevant to the problem, since not having a rough estimate of the expected noise may lead to failed estimations. It is proposed a noise predictor for the used measurement method, based on previous studies on this matter, and even though it is a rough predictor, the results show how it improves performance significantly.

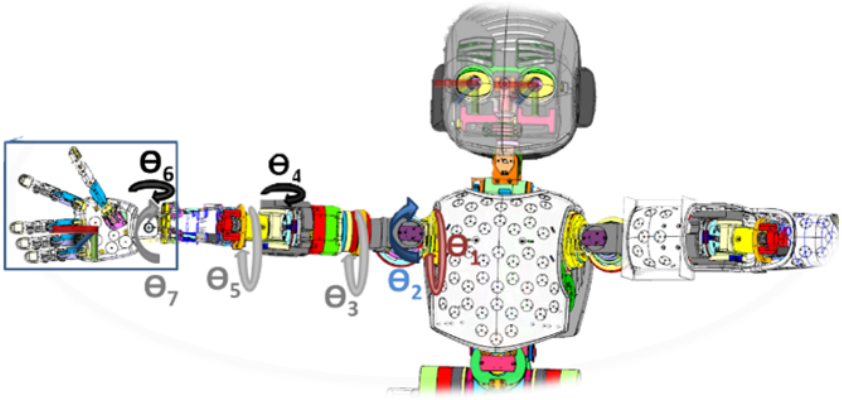


Figure 2.1: iCub Computer Aided Design (CAD) model.

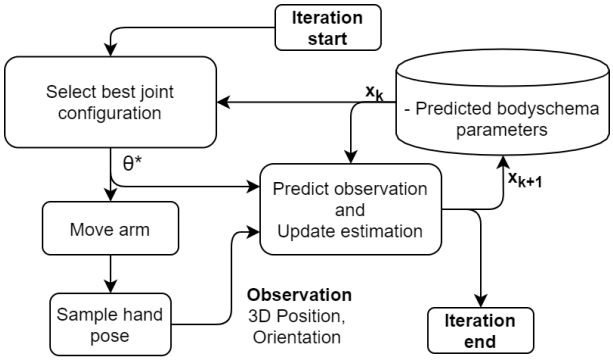


Figure 2.2: Key steps in the structure of the required program to use active learning for joint value selection and observations of the hand pose to estimate the robot's physical parameters, x. The subscript k indicates the algorithm's iteration and θ^* is the selected joint configuration.

Chapter 3

Methods

This Chapter is dedicated to theory and methods required for achieving the goals of this thesis.

Sections 3.1 and 3.2 explain two essencial subjects in a body-schema learning work, which are forward kinematics and Denavit-Hertenberg parameters, respectively. The proposed system requires a recursive estimator to update the estimation of the DH parameters after each sample, for which it is used an Extended Kalman Filter (EKF), explained in Section 3.3. Section 3.4 is about the angle-axis convention, which is used in the EKF to represent the hand orientation. To guarantee sampling and movement efficiency, a cost-sensitive active learning criterion must be defined, which is presented in Section 3.5. As already mentioned in the previous Chapters, this thesis proposes acquiring samples of the hand pose for the calibration procedure using visual input. Since the hand may not be detected by the cameras, in Section 3.6 an occlusion model is explained, used to predict whether the hand should be visible or not, based on previous attempts, using non-parametric smoothed beta distributions.

3.1 Forward Kinematics

A robotic arm consists of rigid bodies sequentially connected by joints, which can be revolute or prismatic. The robot used in this thesis possesses seven joints, all revolute. One end of the kinematic chain is fixed to a base and the other end is loose, the hand, often referred to as end-effector in the literature, allowing it to interact with the environment. Computing the forward kinematics is the process of obtaining the pose of the hand, in relation to a base reference frame. This Section is based on [17], where more detailed information can be found, regarding this topic.

The values for the coordinates of a particular point, P, in space may vary, depending on the reference frame it is being measured from. For example, if a robotic arm is gripping a ball, using its end-effector, the 3-dimensional position of the ball with respect to the end-effector may be $(0, 0, 0)$, to simplify, but it will be a very different value with respect to the base, because it is farther away. The same can be said for the orientation of a vector. The relation between the values of a particular vector in different reference frames is given by a matrix, usually called transformation matrix. Given two reference frames, frame 0 and frame 1, the relation between the coordinates of point P in frame 1, $\boldsymbol{p}^1$, and the coordinates of point P in frame 0, $\boldsymbol{p}^0$, is given by

$$\boldsymbol{p}^0 = \boldsymbol{o}_1^0 + \boldsymbol{R}_1^0 \boldsymbol{p}^1, \tag{3.1}$$

where $\boldsymbol{o}_1^0$ is the position of the origin of frame 1 with respect to frame 0 and $\boldsymbol{R}_1^0$ is a rotation matrix given by

$$\boldsymbol{R} = \begin{bmatrix} \boldsymbol{x}_1^0 & \boldsymbol{y}_1^0 & \boldsymbol{z}_{1,}^0 \end{bmatrix} \tag{3.2}$$

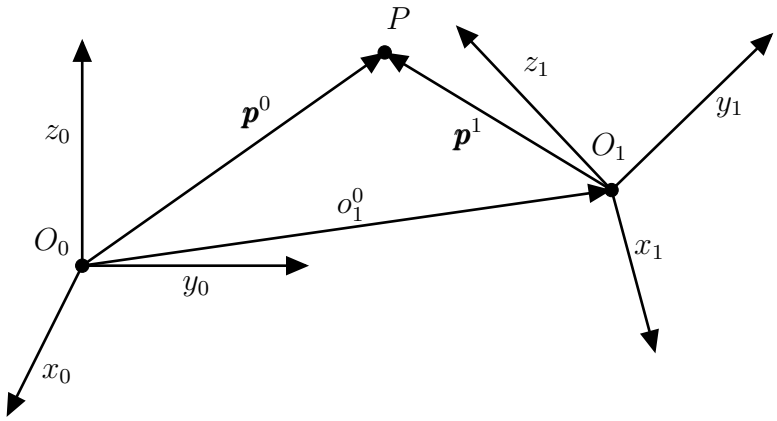


Figure 3.1: Coordinate transformation between reference frames.

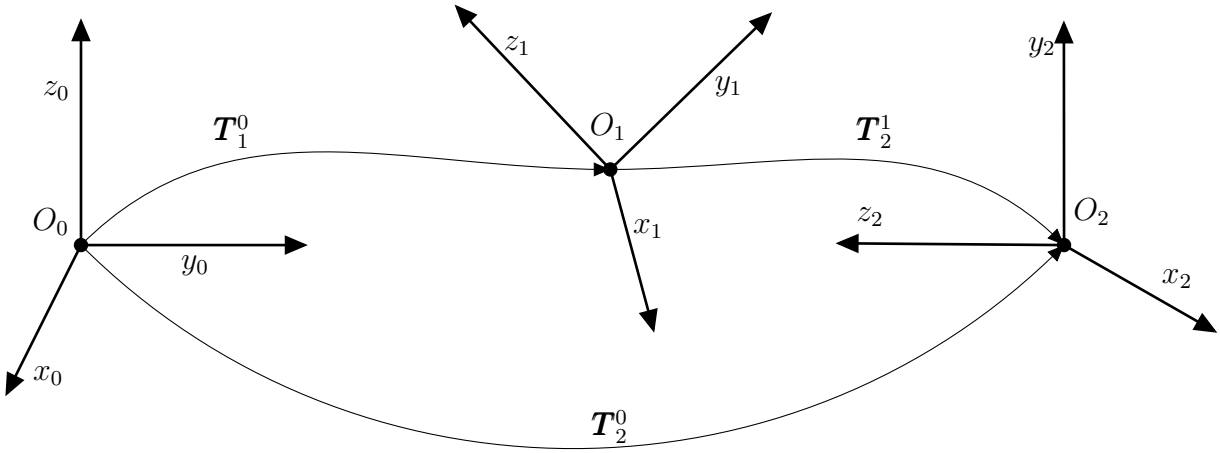


Figure 3.2: Sequential coordinate transformation between multiple reference frames.

where $\boldsymbol{x}_1^0$, $\boldsymbol{y}_1^0$ and $\boldsymbol{z}_1^0$ are the axes of frame 1 with respect to frame 0. These notations are depicted in Figure 3.1.

To achieve a compact notation for (3.1), another representation for a generic vector p is introduced

$$\tilde{\boldsymbol{p}} = \begin{bmatrix} \boldsymbol{p} \\ 1 \end{bmatrix}. \tag{3.3}$$

Now, the coordinate transformation between $\boldsymbol{p}^0$ and $\boldsymbol{p}^1$ is given by

$$\tilde{\boldsymbol{p}}^0 = \boldsymbol{T}_1^0 \tilde{\boldsymbol{p}}^1, \tag{3.4}$$

where $\boldsymbol{T}_1^0$ is a 4×4 matrix

$$\boldsymbol{T}_1^0 = \begin{bmatrix} \boldsymbol{R}_1^0 & \boldsymbol{o}_1^0 \\ \boldsymbol{0}^T & 1 \end{bmatrix}. \tag{3.5}$$

Considering a kinematic chain with n links, obtaining the description of end-effector kinematics can be done recursively, by sequentially multiplying transformation matrices describing the kinematic relations between consecutive joints, as shown in Figure 3.2. A coordinate frame is attached to each joint, from joint 0 to joint $n-1$, and frame n to the end-effector. Then, the coordinate transformation between

frame n with respect to frame 0 is given by

$$\boldsymbol{T}_n^0(\boldsymbol{\theta}) = \boldsymbol{T}_1^0(\theta^{(1)})\boldsymbol{T}_2^1(\theta^{(2)})\dots\boldsymbol{T}_n^{n-1}(\theta^{(n)}), \tag{3.6}$$

where $\boldsymbol{\theta} = \left[\theta^{(1)}\theta^{(2)}\cdots\theta^{(n)}\right]^T$ is a vector of size n containing the values for the joint variables.

3.2 Denavit-Hartenberg Convention

The DH convention offers a systematic way of obtaining the transformation matrix between consecutive frames attached to the joints of a kinematic chain. It is able to do this using only two linear and two angular distances, whereas, normally, transformation matrices between reference frames require the 3-D position offset and 3 angles to compute the rotation matrix, which may be Euler angles, for example.

The DH convention requires reference frames to be attached to each joint following certain rules. This is not necessary for this thesis, since the DH parameters are estimated using the EKF, but, for more information regarding this aspect, see [17]. As already mentioned, there are 4 DH parameters. Considering Figure 3.3, these are defined according to Table 3.1. Since in the iCub arm all joints are revolute, the parameter o_i acts as an offset for the corresponding joint.

a_{i-1}	Distance from F_{i-1} to F_i along the X_{i-1} axis.
α_{i-1}	Angle around the X_{i-1} axis between the Z_{i-1} and Z_i axes.
d_i	Distance from F_{i-1} to F_i along the Z_i axis.
o_i	Angle around the Z_i axis between the X_i and X_{i-1} axes.

Table 3.1: Definition of the Denavit-Hartenberg parameters.

Knowing the DH parameters for link i and the joint angle, $\theta^{(i)}$, [17] shows the transformation matrix between frames i and $i+1$, is given by

$$\boldsymbol{T}_{i+1}^i(\theta^{(i)}) = \begin{bmatrix} c_{o_i+\theta^{(i)}} & -s_{o_i+\theta^{(i)}}c_{\alpha_i} & s_{o_i+\theta^{(i)}}s_{\alpha_i} & a_i c_{o_i+\theta^{(i)}} \\ s_{o_i+\theta^{(i)}} & c_{o_i+\theta^{(i)}}c_{\alpha_i} & -c_{o_i+\theta^{(i)}}s_{\alpha_i} & a_i s_{o_i+\theta^{(i)}} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{3.7}$$

3.3 Extended Kalman Filter

The EKF, explained in detail in [18], allows recursive parameter estimation of systems represented by a nonlinear model, which is the case for the relation between the DH parameters, x, and the hand pose, z, given by the forward kinematics, $h(x, \boldsymbol{\theta})$, computed using (3.6) and (3.7). A measurement of the hand pose, $\boldsymbol{z}_k$, is modelled by

$$\boldsymbol{z}_k = h(\boldsymbol{x}_k, \boldsymbol{\theta}_k) + \boldsymbol{v}_k, \tag{3.8}$$

where $\boldsymbol{\theta}_k$ represents the joint encoder values, $\boldsymbol{v}_k$ represents the measurement noise and k is the time-step. In the EKF, it is also considered a non-linear state transition function, f,

$$\boldsymbol{x}_{k+1} = f(\boldsymbol{x}_k, u_k) + \boldsymbol{w}_k, \tag{3.9}$$

where u_k is a control input of $\boldsymbol{x}_k$ and $\boldsymbol{w}_k$ is the associated process noise. Since in this thesis x are the DH parameters of the iCub's arm, they are approximately constant in time, only affected by the process

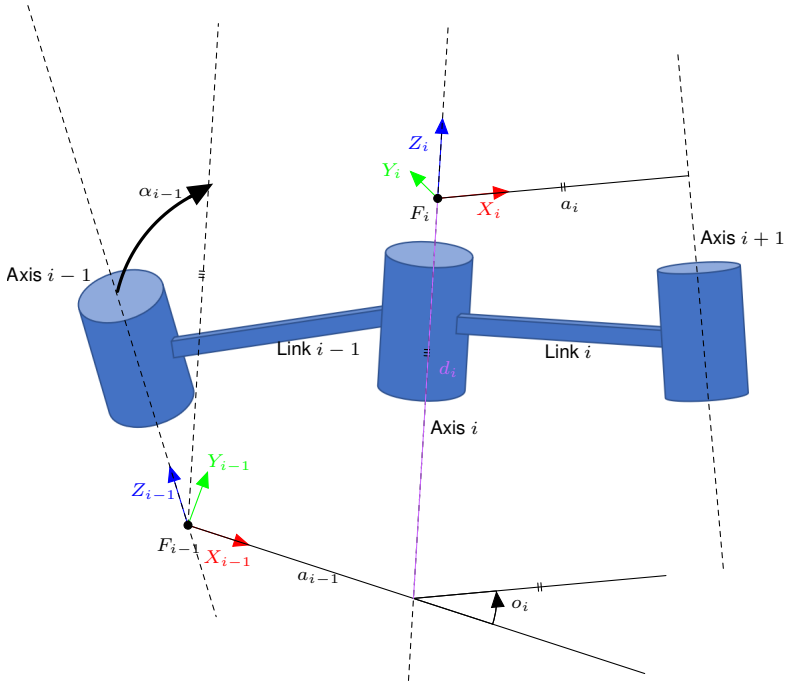


Figure 3.3: Denavit-Hartenberg parameters of sequential links in a kinematic chain and corresponding reference frames.

noise

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{w}_k. \tag{3.10}$$

It is assumed that the measurement and process noises are Gaussian, uncorrelated and zero-mean, have no cross-correlation, and have known co-variance matrices, $\boldsymbol{R}_k$ and $\boldsymbol{Q}_k$, respectively. Therefore,

$$E[\boldsymbol{v}_k \boldsymbol{v}_l^T] = \begin{cases} \boldsymbol{R}_k & k = l \\ 0 & \text{otherwise,} \end{cases} \tag{3.11}$$

$$E[\boldsymbol{w}_k \boldsymbol{w}_l^T] = \begin{cases} \boldsymbol{Q}_k & k = l \\ 0 & \text{otherwise,} \end{cases} \tag{3.12}$$

$$E[\boldsymbol{w}_k \boldsymbol{v}_l^T] = 0, \quad \forall \quad k, l. \tag{3.13}$$

The EKF is divided in two steps. The predict step and the update step.

Predict

This step aims to predict the state x and the prediction variance $\boldsymbol{P}$ at time $k+1$, using only information available at time k. These predictions are given by

$$\hat{\boldsymbol{x}}_{k+1|k} = \hat{\boldsymbol{x}}_{k|k} \tag{3.14}$$

and

$$\hat{\boldsymbol{P}}_{k+1|k} = \hat{\boldsymbol{P}}_{k|k} + \boldsymbol{Q}_k. \tag{3.15}$$

Update

This step updates the state, $\boldsymbol{x}$, and co-variance, $\boldsymbol{P}$, using a combination of the prediction and the new observation obtained of the hand pose, $\boldsymbol{z}_k$. These updates are given by

$$\hat{\boldsymbol{x}}_{k+1|k+1} = \hat{\boldsymbol{x}}_{k+1|k} + \boldsymbol{K}_{k+1}[\boldsymbol{z}_k - h(\hat{\boldsymbol{x}}_{k+1|k}, \boldsymbol{\theta}_k)] \tag{3.16}$$

and

$$\boldsymbol{P}_{k+1|k+1} = \boldsymbol{P}_{k+1|k} - \boldsymbol{K}_{k+1}\boldsymbol{S}_{k+1}\boldsymbol{K}_{k+1}^T, \tag{3.17}$$

where $\boldsymbol{K}_{k+1}$ is the Kalman gain, given by

$$\boldsymbol{K}_{k+1} = \boldsymbol{P}_{k+1|k}\boldsymbol{H}_{k+1}^T\boldsymbol{S}_{k+1}^{-1}, \tag{3.18}$$

where

$$\boldsymbol{S}_{k+1} = \boldsymbol{H}_k\boldsymbol{P}_{k+1|k}\boldsymbol{H}_k^T + \boldsymbol{R}_{k+1}, \tag{3.19}$$

and H is the jacobian matrix of the observation function in (3.8), with respect to $\boldsymbol{x}$,

$$\boldsymbol{H} = \frac{\partial h}{\partial \boldsymbol{x}}. \tag{3.20}$$

3.3.1 Predicting the Pose of the End-effector

At any given iteration, k, of the calibration routine, there is a current prediction of the DH parameters of the arm given by the EKF, $\hat{\boldsymbol{x}}_k$. Knowing the values of the joint encoders, $\boldsymbol{\theta}_k$, the forward kinematics are computed using (3.6), where each transformation matrix is obtained by replacing the predicted DH parameters, $\hat{\boldsymbol{x}}_k$, in (3.7).

Knowing the structure of a transformation matrix from (3.5), (3.6) gives the necessary information about the hand position and the associated rotation matrix. Knowing its rotation matrix, it is expressed in using the angle-axis convention. The prediction, $\hat{\boldsymbol{z}} = h(\hat{\boldsymbol{x}}_k, \boldsymbol{\theta}_k)$, used in (3.16), is the resulting 7-dimensional vector containing the hand position and orientation.

3.3.2 Initialisation

When using the EKF, one must initialise the measurement noise covariance matrix, $\boldsymbol{R}$, defined in (3.11), the initial estimate, $\boldsymbol{x}_0$, its covariance, $\boldsymbol{P}_0$, and the process noise covariance, $\boldsymbol{Q}$, defined in (3.12). These values can be crucial for the success or failure of the EKF, as argued in [19], where also systematic methods of defining these initial values are proposed to reduce the likelihood of failure. These initial guesses, for this thesis, are chosen based on the information from this Section, followed by fine tuning, with trial and error.

Measurement Noise Covariance

The measurement noise covariance matrix, $\boldsymbol{R}$, has the role of defining how much the EKF should trust a particular measurement. This role is evident by analysing (3.18).

If an estimate of the standard deviation of the i-th measurement, σ_i, is known, a simple way to specify the measurement noise covariance matrix, $\boldsymbol{R}_i$, is

$$\boldsymbol{R}_i = \sigma_i^2 I, \tag{3.21}$$

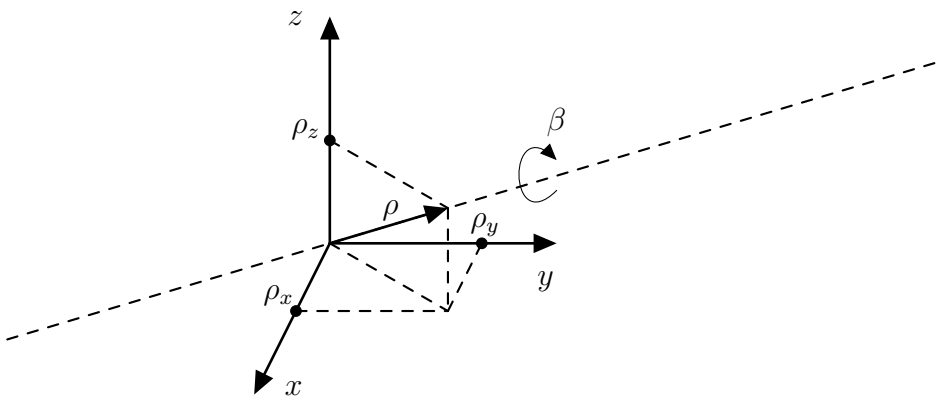


Figure 3.4: Rotation of an angle β around an arbitrary axis, defined by the unit vector ρ.

where I is the identity matrix. If one expects a larger measurement variance, σ_i should be increased, and if a lower measurement error is expected, it should be decreased. Increasing σ_i makes the filter more robust to measurement errors, but over-increasing it may cause the filter to be unaffected by measurements. Oppositely, over-decreasing σ_i can make the filter too sensitive to noisy measurement. Obtaining an estimate of σ_i is, objectively, the most difficult step, since it can never be known exactly. It can be obtained experimentally or from the measurement device's manufacturer. More information regarding this estimation is in Section 4.2.1.

Initial Estimate

For this particular application, the initial estimate, $\hat{\boldsymbol{x}}_0$, is very straight forward, since most robots possess an estimate of their forward kinematics at every moment. This should be the initial estimate.

For the initial estimate covariance matrix, $\boldsymbol{P}_0$, [19] suggests using

$$\boldsymbol{P}_0 = (\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_0)^T(\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_0)I, \tag{3.22}$$

by approximating $\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_0$ as

$$\hat{\boldsymbol{x}}_0 - \boldsymbol{x}_0 = 0.5 \cdot (\boldsymbol{x}_u - \boldsymbol{x}_l), \tag{3.23}$$

since upper, $\boldsymbol{x}_u$, and lower, $\boldsymbol{x}_l$, bounds for the state estimate can frequently be known with reasonable accuracy.

Process Noise Covariance

In [19], it is stated that selecting values for the process noise covariance is usually the biggest challenge, regarding the EKF, and it is shown a systematic method to compute it. For this thesis, since the state, $\boldsymbol{x}$, is representing the DH parameters of the robotic arm, it varies very slowly with the passage of time, therefore the process noise covariance was selected as

$$\boldsymbol{Q} = \sigma_Q^2 \boldsymbol{I}, \tag{3.24}$$

where $\sigma_Q^2 < 10^{-4}$.

3.4 Angle-axis Convention

In Section 3.3, regarding the EKF, it was defined the observation, z, containing the hand pose. It is 7-dimensional, containing the 3-dimensional position and rotation of the hand, using the angle-axis convention.

The angle-axis convention derives the rotation matrix, $\boldsymbol{R}$, by expressing the rotation of an angle β around an arbitrary axis, defined by a unit vector $\boldsymbol{\rho} = [\rho_x \quad \rho_y \quad \rho_z]^T$, as in Figure 3.4. Obtaining the rotation matrix is done by computing

$$\boldsymbol{R}(\beta, \boldsymbol{\rho}) = \begin{bmatrix} \rho_x^2(1-c_\beta)+c_\beta & \rho_x\rho_y(1-c_\beta)-\rho_z s_\beta & \rho_x\rho_z(1-c_\beta)+\rho_y s_\beta \\ \rho_x\rho_y(1-c_\beta)+\rho_z s_\beta & \rho_y^2(1-c_\beta)+c_\beta & \rho_y\rho_z(1-c_\beta)-\rho_x s_\beta \\ \rho_x\rho_z(1-c_\beta)-\rho_y s_\beta & \rho_y\rho_z(1-c_\beta)+\rho_x s_\beta & \rho_z^2(1-c_\beta)+c_\beta \end{bmatrix}, \tag{3.25}$$

where $c_\cdot$ and $s_\cdot$ are the cosine and sine functions, respectively. This matrix holds the property

$$\boldsymbol{R}(-\beta, -\boldsymbol{\rho}) = \boldsymbol{R}(\beta, \boldsymbol{\rho}). \tag{3.26}$$

The inverse problem of computing the axis and angle corresponding to a given rotation matrix is done by using the known information about the structure of the matrix from (3.25). Given an arbitrary rotation matrix

$$\boldsymbol{R} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \tag{3.27}$$

the angle, β, and unit vector, $\boldsymbol{\rho}$, with $\sin(\beta) \neq 0$, are given by

$$\beta = \cos^{-1}\left(\frac{r_{11}+r_{22}+r_{33}-1}{2}\right) \tag{3.28}$$

$$\boldsymbol{\rho} = \frac{1}{2\sin(\beta)} \begin{bmatrix} r_{32}-r_{23} \\ r_{13}-r_{31} \\ r_{21}-r_{12} \end{bmatrix}. \tag{3.29}$$

If $\beta = 0$, it means there is no rotation, therefore $\boldsymbol{\rho}$ is arbitrary. If $\beta = \pi$, $\boldsymbol{R}(\beta, \boldsymbol{\rho})$ simplifies to

$$\boldsymbol{R}(\beta, \boldsymbol{\rho}) = \begin{bmatrix} 2\rho_x^2-1 & 2\rho_x\rho_y & 2\rho_x\rho_z \\ 2\rho_x\rho_y & 2\rho_y^2-1 & 2\rho_y\rho_z \\ 2\rho_x\rho_z & 2\rho_y\rho_z & 2\rho_z^2-1 \end{bmatrix} \tag{3.30}$$

and three of the equations from this matrix must be used to compute the three components of the unit vector $\boldsymbol{\rho}$.

3.5 Active Learning

An active learning problem is one where the learner has the ability to select its own training data, either from i) a limited pool of unlabelled data (pool-based sampling), ii) by looking at a stream of input data and deciding whether to query or discard it (selective sampling) or iii) by generating its own queries (query synthesis) [4]. This Section will focus on query synthesis, since this thesis is a regression problem, and this approach has been shown to work when the measurements do not depend on human interpretation [20].

The goal of an active learner is to identify what is the optimal instance to query or action to perform. There are several criteria to achieve this. For instance, expected entropy reduction, expected error reduction, expected variance reduction or expected model change. As shown in [20], the right criterion may depend on the problem itself and if the wrong one is chosen, it may even perform worse than querying randomly. After choosing the appropriate criterion, one must choose an appropriate cost function, $C(\cdot)$, and then the next instance to query, $\boldsymbol{\theta}$, is given by

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\text{argmin}} \quad C(\boldsymbol{\theta}). \tag{3.31}$$

For this given problem, it is desired to choose the joint angles, $\boldsymbol{\theta}$, for the next action, which will most reduce the expected error in the estimation model.

A-optimality Criterion

The A-optimality criterion is proposed by [13] for active selection of the joint angles, $\boldsymbol{\theta}$. This criterion can be used, since the EKF linearises the observation function around $\hat{\boldsymbol{x}}_k$. It consists of defining the cost function as the expected mean squared error of the robot parameters, x. Since the observation noise from (3.8) is considered to be Gaussian, the cost function is given by the expected trace of the covariance matrix of x, given the previous observations, $\boldsymbol{z}_{1:k}$, for joint configurations $\boldsymbol{\theta}_{1:k}$,

$$\begin{aligned} C_0(\boldsymbol{\theta}) &= \mathbb{E}\left[(\hat{\boldsymbol{x}}_{k+1} - x)^T(\hat{\boldsymbol{x}}_{k+1} - x)|\boldsymbol{z}_{1:k}, \boldsymbol{\theta}_{1:k}\right] \\ &\approx \mathbb{E}\left[tr(\boldsymbol{P}_{k+1})|\boldsymbol{z}_{1:k}, \boldsymbol{\theta}_{1:k}\right]. \end{aligned} \tag{3.32}$$

Since the robot should estimate the hand pose using visual input, it needs to be able to find the hand. Some adaptations were made to (3.32) to avoid the selection of joint configurations where, for example, the hand would be behind the robot's back or hidden by some other parts of the body. As observable in Figure 3.5, positive values of x for the hand position are not desirable due to the robot frame having the x axis pointing backwards. In order to discourage this area, a term was added to the cost function from (3.31) to penalise positive values for x in the predicted position, $\hat{p}$, for a given joint configuration $\boldsymbol{\theta}$. Therefore it was adapted to

$$C\left(\boldsymbol{\theta}\right) = \frac{C_0\left(\boldsymbol{\theta}\right)}{\bar{\eta}} + a \cdot \arctan\left(b \cdot \hat{p}_x\right), \tag{3.33}$$

where $a > 0$ and $b > 0$ are tunable, and $\bar{\eta}$ is the likelihood of the hand being detected by the cameras for the given joint configuration $\boldsymbol{\theta}$. How to estimate this likelihood is explained in Section 3.6.

Computing $C(\boldsymbol{\theta})$

The information used to compute the co-variance matrix $\boldsymbol{P}$ is shown schematically in Figure 3.6. In order to compute it, the predict and update step of the EKF are computed with (3.15) and (3.17). This is possible since the prediction and update of the co-variance matrix do not depend on the sample taken, only on the jacobian $\boldsymbol{H} = \frac{\partial h}{\partial \boldsymbol{x}}$ computed at the selected joint configuration, $\boldsymbol{\theta}$, and on the measurement noise co-variance, $\boldsymbol{R}$.

3.5.1 Cost-sensitive Active Learning

Throughout most active learning researches and works as [13], [9] and [10], the main focus of active learning is to reduce the number of measurements acquired to train a good model. This assumes that the cost of acquiring those measurements is approximately uniform, so reducing the number of iterations indeed reduces the acquisition cost.

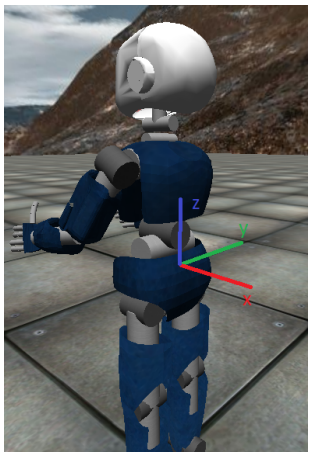


Figure 3.5: iCub simulator base reference frame. The red, green and blue lines represent the x, y and z axes, respectively.

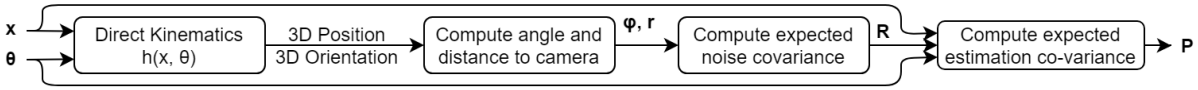


Figure 3.6: Steps to compute the expected estimation co-variance, given the current estimation of the DH parameters, $\boldsymbol{x}$, and a joint configuration, $\boldsymbol{\theta}$. ϕ and r represent the angle and distance of the hand to the camera, respectively, according to Figure 4.4a.

In this thesis, the sample acquisition cost is directly related to the amount of movement performed by the arm, since (3.32) selects the next joint configuration to where the arm moves to. This thesis mitigates the amount of movement performed in two separate ways.

Unconstrained Optimisation

In order to penalise a great amount of movement in the calibration routine, one can simply change (3.31) to

$$\boldsymbol{\theta}_k^* = \underset{\boldsymbol{\theta}}{\text{argmin}} \quad C(\boldsymbol{\theta}) + \gamma d(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1}^*), \tag{3.34}$$

where $d(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1}^*)$ is a distance measurement, representing the cost associated to moving to the next joint configuration, $\boldsymbol{\theta}$, accounting for the previous one, $\boldsymbol{\theta}_{k-1}^*$, and γ is a parameter which can be changed according to the relative weight intended. Increasing the value of γ will increase the probability of $\boldsymbol{\theta}_k^*$ being closer to $\boldsymbol{\theta}_{k-1}^*$, thus reducing the total movement of the arm.

Regarding the actual implementation of the method, $d(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1}^*)$ is the squared euclidean norm,

$$d(\boldsymbol{\theta}, \boldsymbol{\theta}_{k-1}^*) = \|\boldsymbol{\theta} - \boldsymbol{\theta}_{k-1}^*\|^2. \tag{3.35}$$

Several different functions can be used for this effect. The squared l_2-norm was chosen due to the fact that it is closer to zero around its minimum than the regular l_2-norm or the l_1-norm, which allows more exploration in those regions.

Constrained Optimisation

Reducing the amount of arm movement during the calibration routine can also be done by constraining the optimisation problem, instead of changing the function to optimise, as in (3.34). The choice of the

next joint configuration is given by

$$\boldsymbol{\theta}_k^* = \underset{\boldsymbol{\theta} \in \left[\boldsymbol{\theta}_{k-1}^* - \boldsymbol{\Delta}, \boldsymbol{\theta}_{k-1}^* + \boldsymbol{\Delta}\right]}{\operatorname{argmin}} C(\boldsymbol{\theta}), \tag{3.36}$$

where $\boldsymbol{\theta}_{k-1}^*$ is the previous joint configuration selected and $\boldsymbol{\Delta}$ is a vector of size equal to the number of joints, n. Considering normalised joint values in the interval $[0, 1]$, $\boldsymbol{\Delta}$ is defined as

$$\boldsymbol{\Delta} = \delta \cdot \mathbf{1}_{\boldsymbol{n}}, \tag{3.37}$$

where $\mathbf{1}_{\boldsymbol{n}}$ is a unit vector of size n and δ is a tuning parameter that defines the relative movement every joint can perform around the current configuration. This is a stricter option than (3.34), since it is guaranteed that the next joint configuration will be inside the n-dimensional box defined by the interval $\left[\boldsymbol{\theta}_{k-1}^* - \boldsymbol{\Delta}, \boldsymbol{\theta}_{k-1}^* + \boldsymbol{\Delta}\right]$.

3.6 Non-parametric Occlusion Model

When a new joint configuration is selected, the hand may or may not be visible to the cameras. If the hand is not visible, a sample cannot be obtained to update the EKF. Consequently, the same joint configuration is selected again when solving (3.31), since the cost function from (3.32) remains unaltered. Using non-parametric smoothed beta distributions removes this issue, by having the robot retain the knowledge about the successful and unsuccessful attempts at sampling the hand pose and using it to predict the likelihood of a new joint configuration yielding a successful sample.

3.6.1 Beta-Binomial Model

Consider, firstly, the robot making m attempts at sampling the hand pose at a particular joint configuration, $\boldsymbol{\theta}_i$. It is successful S times and unsuccessful U times. Each of those attempts is a Bernoulli trial with an unknown probability, η, of success. Then,

$$p(S|\eta, m) \propto \eta^S \left(1 - \eta\right)^{m-S} = \eta^S \left(1 - \eta\right)^U \tag{3.38}$$

The unknown probability, η is a random variable modeled as a Beta distribution, $\eta \sim \text{Beta}(a, b)$. This means

$$p(\eta) = \frac{\eta^{(a-1)}(1-\eta)^{(b-1)}}{B(a,b)} \propto \eta^{(a-1)}(1-\eta)^{(b-1)}, \tag{3.39}$$

where $B(a, b)$ is the Beta function. Using Bayes rule, it is possible to compute the posterior of η, given the number of successful and unsuccessful trials

$$p(\eta|S, m) \propto p(S|\eta, m) \cdot p(\eta), \tag{3.40}$$

given by

$$p(\eta|S, m) \propto \eta^{a+S-1}(1-\eta)^{b+U-1}, \tag{3.41}$$

meaning the posterior of η is also a Beta distribution with parameters $a + m_S$ and $b + m_U$.

3.6.2 Kernel Extrapolation

In [21], a kernel based non parametric approach is used to predict the probability of a successful grasp by extrapolating past data to new unseen features. The same method is applied in this thesis to predict the likelihood, η_*, of having a visible marker to the cameras in a particular joint configuration, $\boldsymbol{\theta}_*$, given the past k observations, $Y = \{y_1, y_2, \ldots, y_k\}$, in other configurations, $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_0, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_k\}$, where each y_i contains the number of successful an unsuccessful sampling attempts, S_i and U_i, respectively.

The Bayes rule is used to obtain the posterior

$$p(\eta_*|\boldsymbol{\theta}_*, \boldsymbol{\Theta}, Y) \propto p(Y|\eta_*, \boldsymbol{\theta}_*, \boldsymbol{\Theta}) p(\eta_*|\boldsymbol{\theta}_*, \boldsymbol{\Theta}), \tag{3.42}$$

where the prior $p(\eta_*|\boldsymbol{\theta}_*, \Theta)$ is a Beta distribution, with parameters a_0 and b_0, $Be(a_0, b_0)$, and the likelihood term,

$$p(Y|\eta_*, \boldsymbol{\theta}_*, \boldsymbol{\Theta}) = \prod_{i=0}^{k} p(y_i|\eta_*, \boldsymbol{\theta}_*, \boldsymbol{\theta}_i), \tag{3.43}$$

represents the likelihood of the observation, y_i, at each joint configuration, $\boldsymbol{\theta}_i$, given $\boldsymbol{\theta}_*$ and the success rate, η_*. For each observation, [21] models the likelihood function as a binomial distribution

$$p(y_i|\eta_*, \boldsymbol{\theta}_*, \boldsymbol{\theta}_i) = Bin(S_{*i}; \eta_*, S_{*i} + U_{*i}), \tag{3.44}$$

where $S_{*i} = K(\boldsymbol{\theta}_*, \boldsymbol{\theta}_i) S_i$ and $U_{*i} = K(\boldsymbol{\theta}_*, \boldsymbol{\theta}_i) U_i$ are the number of successful and failed sampling attempts at the joint configuration $\boldsymbol{\theta}_*$, given the attempts at $\boldsymbol{\theta}_i$, propagated by a kernel function, K. The books [22] and [23] provide more information on kernel functions. It was used a squared exponential kernel so that the diffusion process keeps $S_* = S_i$ and $U_* = U_i$ for $\boldsymbol{\theta}_* = \boldsymbol{\theta}_i$ and decreases as $\boldsymbol{\theta}_*$ is farther in the joint space. Using the deduced posterior in [21],

$$p(\eta_*|\boldsymbol{\theta}_*, \boldsymbol{\Theta}, Y) = \text{Be}\left(\eta_*; \sum_{i=0}^{k} S_{*i} + a_0, \sum_{i=0}^{k} U_{*i} + b_0\right). \tag{3.45}$$

Finally, the predicted sampling success mean probability is given by

$$\bar{\eta}_* = \frac{\sum_{i=0}^{k} S_{*i} + a_0}{\sum_{i=0}^{k} S_{*i} + a_0 + \sum_{i=0}^{k} U_{*i} + b_0}. \tag{3.46}$$

This result is used to select joint configurations more likely to have a visible marker in the cost function from (3.33).

As an example, Figure 3.7 shows a simple 1-dimensional example, using 10 training data points sampled from a square wave function. As in the ArUco marker observation, 1 indicate a successful and 0 an unsuccessful attempt. Using non-parametric smoothed beta distributions allows the prediction of the probability of occurring a success on any point of the input space.

Squared Exponential Kernel

A kernel is a measure of similarity between different points in the input space [23]. One of the most commonly used kernels is the squared exponential, given by

$$k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = \sigma \cdot \exp\left(-\frac{1}{2}(\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)^T \boldsymbol{L}^{-2} (\boldsymbol{\theta}_i - \boldsymbol{\theta}_j)\right), \tag{3.47}$$

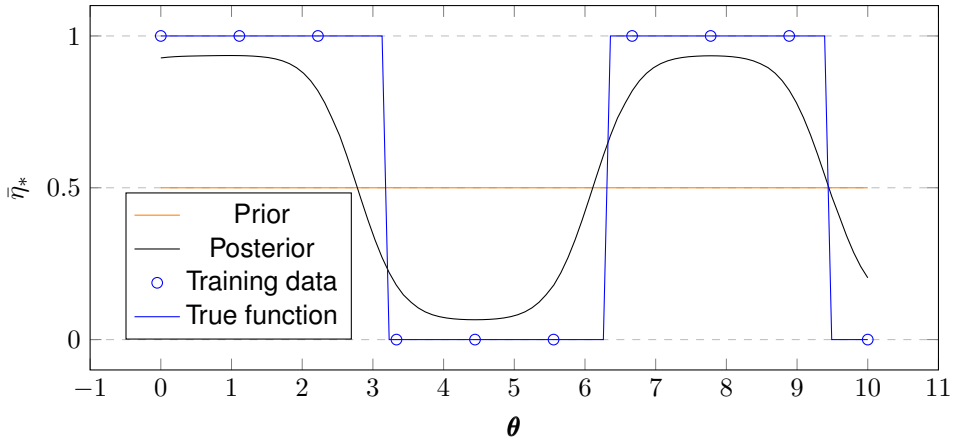
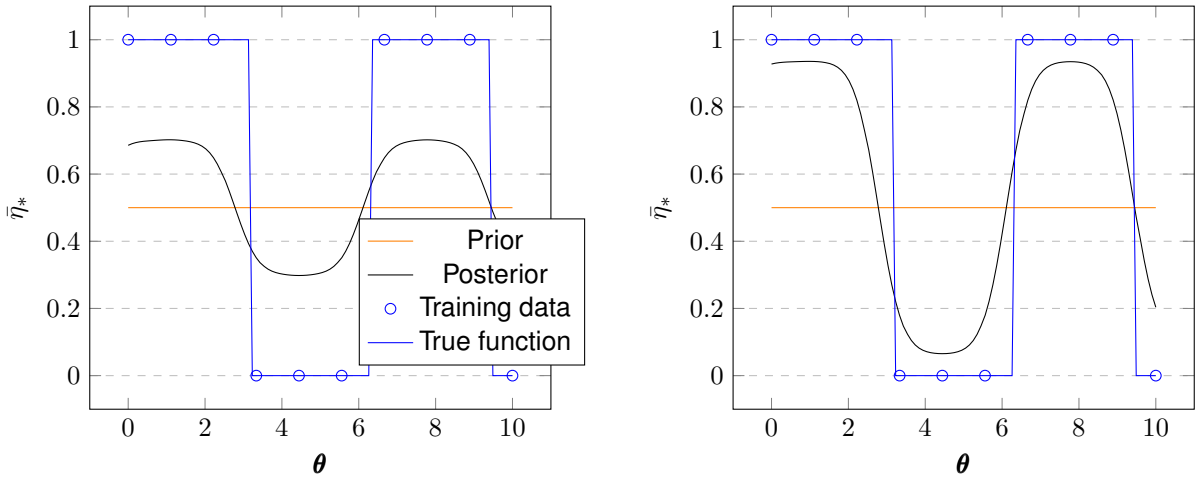


Figure 3.7: Predicting sampling success mean probability for 1-dimensional example.



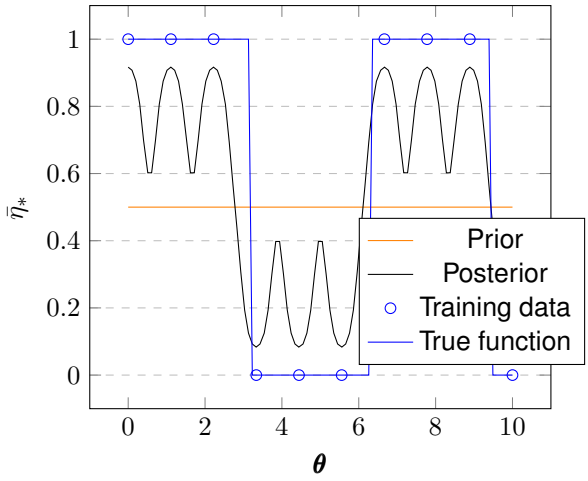
(a) $\sigma = 1$ (b) $\sigma = 10$

Figure 3.8: Different posteriors obtained using different σ values for the squared exponential kernel.

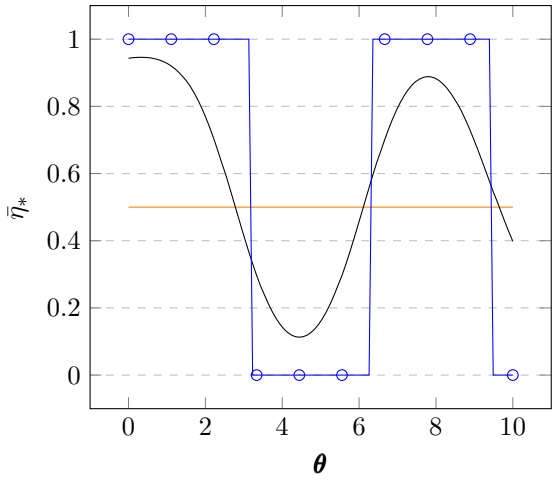
where $\sigma > 0$ and $\boldsymbol{L}$ is a diagonal matrix. The parameter σ controls how much the posterior should deviate from the prior after the measurements. A higher value for σ obtains a posterior trusting more on the measurements, whereas a lower value for sigma obtains a posterior closer to the prior, as the examples in Figure 3.8 show. Each of the n values on the diagonal of $\boldsymbol{L}$ controls the influence of each of the n dimensions of $\boldsymbol{\theta}_i$ and $\boldsymbol{\theta}_j$ on each other, respectively. Lower values for the diagonal elements of $\boldsymbol{L}$ mean the training data is only relevant to closer points, while high values mean the training data influences other points farther away. Lower values should be used when the function is expected to be rapidly changing, while higher values should be used for functions expected to be smoother. A 1-dimensional example can be seen in Figure 3.9.

Selected Prior

Choosing a prior for the sampling success probability consists of selecting the values for a_0 and b_0 from (3.45) and (3.46). With no knowledge about whether a marker is visible or not, a sensible prior is $a_0 = b_0 = 1$, which means the prior Beta distribution, $p(\eta_*|\boldsymbol{\theta}_*, \Theta) = Be(a_0, b_0)$, is equivalent to a uniform distribution with an expected value of 0.5, since $\eta_* \in [0, 1]$. This is the prior used in Figures 3.7, 3.8 and 3.9. However, the values $a_0 = 1$ and $b_0 = 0$ were chosen for the prior of the occlusion model, which means the robot expects to see the marker for every joint configuration prior to any training

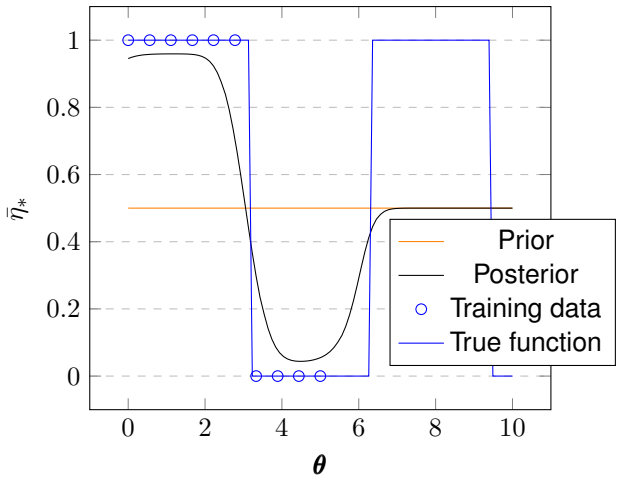


(a) $L = 0.2$

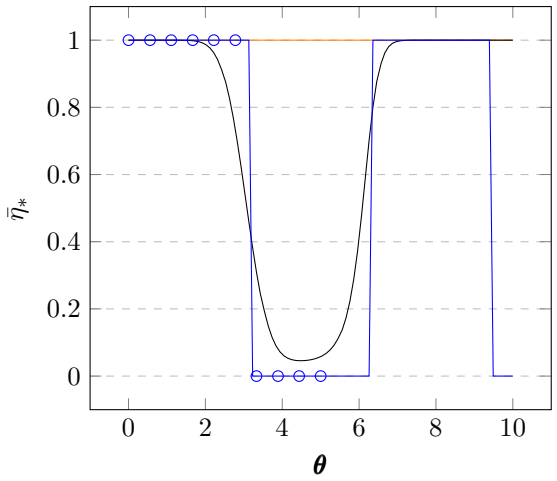


(b) $L = 1$

Figure 3.9: Different posteriors obtained using different L values for the squared exponential kernel.



(a) Prior with $\bar{\eta}_* = 0.5$.



(b) Prior with $\bar{\eta}_* = 1$.

Figure 3.10: Posterior data for the sampling success mean probability, $\bar{\eta}_*$, obtained from different priors, given a set of training data, for a 1-dimensional $\boldsymbol{\theta}$.

data, $\bar{\eta_*} = 1$.

Recall from the active learning cost function (3.33) that the robot is more likely to choose joint configurations where the marker is more likely to be visible (higher values of $\bar{\eta_*}$). Now, consider Figure 3.10, where the training data is not evenly spread. Notice on Figure 3.10b how using a prior for which $\bar{\eta_*} = 0.5$ may make (3.33) become biased towards an already explored region where the marker was visible. On Figure 3.10a the same training data is plotted, but the posterior is obtained from a prior for which $\bar{\eta_*} = 1$. With this prior the robot will only be less likely to explore a certain region if there are failed sample attempts nearby. This allows more exploration, which is desirable, since repeatedly selecting safer configurations can cause the EKF to overfit.

Chapter 4

Experimental Setting

The active learning approach proposed requires optimisation of non-differentiable functions, therefore a global optimisation algorithm called DIRECT is used, explained in detail in Section 4.1. Sampling the hand pose using visual input is done using ArUco markers, shown in Section 4.2. These are placed on the back and palm of the hand, providing an easy and reliable way to detect its position and orientation. Section 4.3 gives some details regarding the iCub simulator and the gaze interface used. Section 4.4 describes the performance metrics used to evaluate the different methods. Finally, Section 4.5 shows the program flowcharts and identifies the responsibilities of each method in them.

4.1 Optimisation Algorithm DIRECT

This thesis is aimed for robots that need to be responsive to external stimuli, therefore the optimisation of the cost function must obey real-time constraints. In addition, optimising the cost function (3.33) requires a global optimisation algorithm, since it is not easily differentiable. The DIRECT algorithm, proposed in [24], lowers the computational complexity by dividing the input space into hyper-rectangles and only sample their midpoints. This allows the algorithm to work in high-dimensional spaces, being efficient for 10-20 dimensions, according to the authors. The NIopt library [25] provides an implementation for the DIRECT algorithm, as well as for multiple other global and non-linear optimisation algorithms.

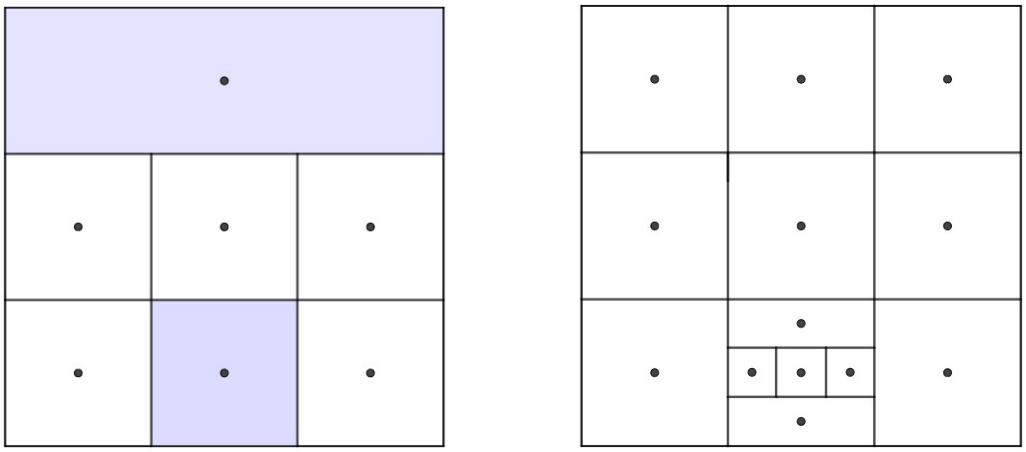
Firstly, DIRECT requires the search space to be normalised to be the n-dimensional unit hyper-cube. For this thesis, a cost function $C(\boldsymbol{\theta})$ requires optimisation, where each joint $\theta^{(i)}$ is constrained to an interval $[a^{(i)}, b^{(i)}]$, therefore the normalisation will be given by applying the transformation

$$\theta^{(i)}_{norm} = \frac{\theta^{(i)}}{b^{(i)} - a^{(i)}}. \tag{4.1}$$

As the algorithm proceeds, the search space defined by the unit hyper-cube will be divided into hyper-rectangles, each with a sampled point at their centre. Each iteration of the algorithm possesses two major steps, depicted in Figure 4.1:

- Identify potentially optimal hyperrectangles.
- Divide the identified hyperrectangles and sample the new hyperrectangles' centres.

These steps are performed repeatedly for a limited number of iterations or until there is no significant improvement between iterations. The following Sections 4.1.1 and 4.1.2 provide a description of these steps and Section 4.1.3 provides the complete algorithm.



(a) DIRECT algorithm identifying potentially optimal rectangles, shaded.

(b) DIRECT algorithm division of the identified hyper-rectangles in Figure 4.1a.

Figure 4.1: DIRECT algorithm major steps.

4.1.1 Identify Potentially Optimal Hyper-rectangles

For each hyper-rectangle, the function function value is known at its centre c, as well the distance d from the centre point to the vertices. This information allows to form the diagram from Figure 4.2. A hyper-rectangle j is potentially optimal, in a set of m hyper-rectangles, if there exists some $\tilde{K} > 0$, such that

$$f(c_j) - \tilde{K}d_j \leq f(c_i) - \tilde{K}d_i, \quad \forall\, i = 1, \cdots, m \tag{4.2}$$

and

$$f(c_j) - \tilde{K}d_j \leq f_{min} - \epsilon|f_{min}|, \tag{4.3}$$

where ϵ is a positive constant and f_{min} is the minimum value of the function found at that given time. The second condition is needed to prevent the algorithm from wasting function evaluations in pursuit of extremely small improvements. The authors of [24] claim the algorithm is fairly insensitive to the value of ϵ and provides good results for values between 10^{-3} to 10^{-7}.

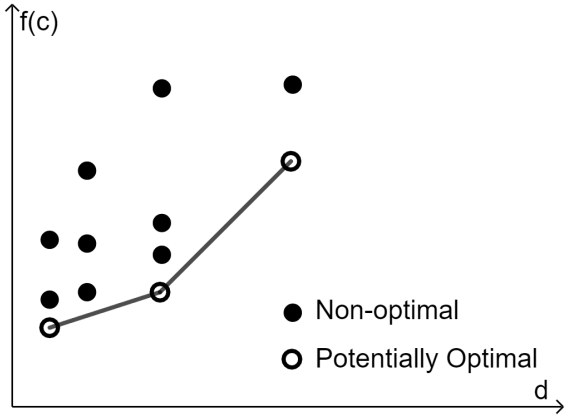


Figure 4.2: Illustration of the identification of potetially optimal rectangles.

4.1.2 Dividing Hyper-rectangles

Figure 4.1 provides a 2-dimensional visual representation of how the algorithm divides hyper-cubes and hyper-rectangles. The division is always done along the dimensions with the highest length, in order to ensure that the hyper-rectangles shrink on every dimension, which is determinant for convergence, as proven in [24]. Algorithm 1 describes the procedure for the division, covering both hyper-cubes and hyper-rectangles.

Algorithm 1 Dividing Hyper-rectangles - adapted from [24]

1: Identify the set I of dimensions with the maximum side length. Let δ equal one-third of this maximum side length.
2: Sample the function at the points $c \pm \delta e_i$ for all $i \in I$, where c is the centre of the rectangle and e_i is the ith unit vector.
3: Divide the rectangle containing c into thirds along the dimensions in I, starting with the dimension with the lowest value of $w_i = \min\{f(c + \delta e_i), f(c - \delta e_i)\}$ and continuing to the dimension with the highest w_i.

4.1.3 Implementation of the DIRECT Algorithm

The search begins by sampling at the centre of the unit hyper-cube. Then, at each iteration, a set of the potentially optimal hyper-rectangles is identified, as described in Section 4.1.1. Those hyper-rectangles are then sampled and subdivided, with Algorithm 1. This process is repeated until a certain number of iterations or function evaluations is reached. The formal statement of the algorithm, adapted from [24], is in Algorithm 2.

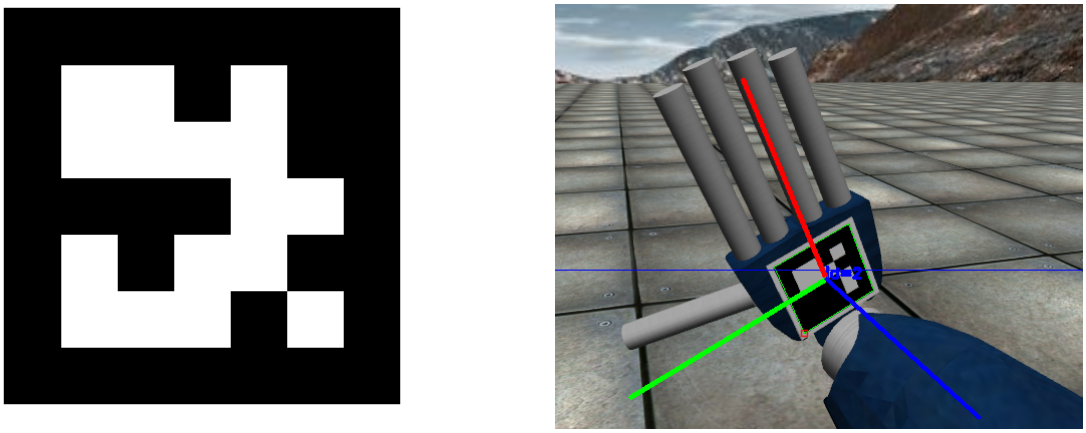
Algorithm 2 Multivariate DIRECT Algorithm - adapted from [24]

1: Let c_1 be the centre point of the unit hyper-cube and evaluate $f(c_1)$. Set $f_{min} = f(c_1)$, $m = 1$, and maxIter be the number of iterations.
2: **for** $k = 0, \cdots$,maxIter **do**
3: Identify the set S of potentially optimal hyper-rectangles, as described in Section 4.1.1.
4: **while** $S \neq \emptyset$ **do**
5: Select any hyper-rectangle $j \in S$.
6: Using the procedure described in Section 4.1.2, determine where to sample within the hyper-rectangle j and how to divide the hyper-rectangle into subrectangles.
7: Update f_{min} and set $m = m + \Delta m$, where Δm is the number of new points sampled.
8: Set $S = S - \{j\}$.
9: **end while**
10: **end for**

4.2 ArUco Module

The ArUco module is based on the ArUco library [26], which serves for detection of square fiducial markers (ArUco markers), like the one in Figure 4.3a. The ArUco module provides functions for marker creation, marker detection and marker pose estimation, in addition to functions to draw the detected markers and their orientation in the input image for visualisation, as shown in Figure 4.3b.

Each ArUco marker encodes a unique ID by a binary code, given by different patterns, and belongs to a set of markers, referred to as dictionary. The ArUco module provides multiple pre-defined dictionaries and also offers configurable dictionaries in size and number of bits. The set of markers in the dictionaries follow a criterion to maximise inter-marker distance, in other words, they are as different as possible from



(a) Example of an ArUco marker.

(b) ArUco marker pose estimation.

Figure 4.3: ArUco marker and pose detection in the iCub simulator. The red, green and blue lines represent the $\boldsymbol{x}$, y and z axes, respectively.

each other, to avoid possible errors. Only two markers are used to measure the hand pose. One for the palm of the hand and one for the back of the hand, therefore the smallest 4×4 bits predefined dictionary is used, to reduce the inter-marker confusion rate.

As stated in [26], the marker pose with respect to the camera is estimated by minimising the re-projection error of the corners, using an iterative algorithm. This re-projection error measures how close the real image is to the 2-D projection of the 3-D estimate.

The ArUco module, available in OpenCV[1], will be responsible for acquiring observations of the hand pose using the iCub cameras.

4.2.1 Measurement Noise Model

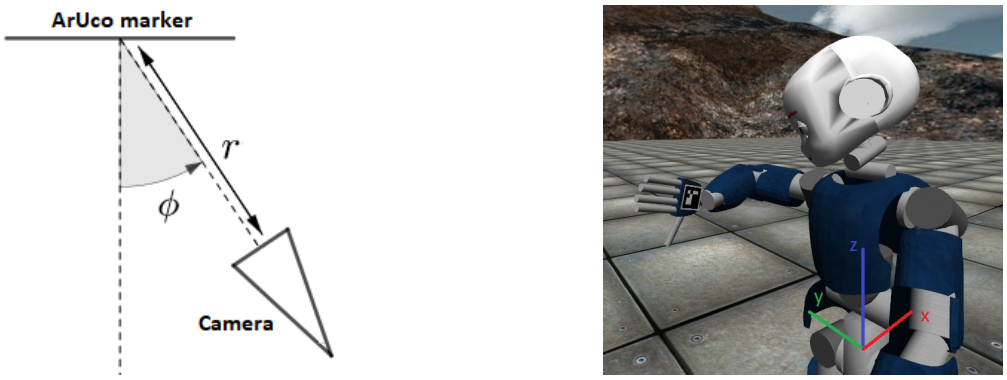
In the research field of estimation and tracking, it is crucial to have knowledge about what is the expected error for a given sample and if the expected error is constant for every sample or if it heteroscedastic, meaning it is not constant in time or in space. This allows to give more importance to a sample which has a lesser amount of error expected and vice-versa.

From Section 4.2, it is clear that the accuracy of the pose obtained depends highly on camera calibration parameters and on the amount of pixels the marker occupies in the image. It is of paramount importance to estimate the expected measurement error for each sample since it will be different. For instance, a marker positioned closer to the camera will yield a more reliable estimate than a marker which is farther away. Having prior knowledge about this allows estimators, in this case the EKF, to know how much to rely on a particular sample and change the measurement co-variance from (3.11) accordingly.

Measurement error and variance in fiducial markers have been studied in various works, such as [27] and [28]. They show how measurement error and variance change with the distance and angle to the camera. The information in these works allowed building a rough predictor of the error, given a particular obtained sample. Both the works achieve similar conclusions:

- The closer to the camera the fiducial marker is, the better is the pose estimation.
- The rotation estimation seems to be worse when the marker is directly facing the camera and the systematic error and variance are lower when the marker is angled 30º to 50º from the camera.

[1] A tutorial of how to use the module is at `https://docs.opencv.org/trunk/d5/dae/tutorial_aruco_detection.html`.



(a) ArUco marker pose in relation to the camera.

(b) ArUco marker placed on the right hand of the iCub simulator.

Figure 4.4: iCub camera facing ArUco marker. The distance from the camera to the marker is represented by r and ϕ represents the angle between the perpendicular direction of the marker and the direction defined by the centre point of the marker and the camera position.

In this thesis, the measurement noise co-variance matrix structure was given by (3.21). Given the studies in the mentioned works, a very rough approximation of the expected error can be made as a function of camera distance and camera angle to the marker, in order to change σ_i^2 according to the measurement obtained.[2] The mentioned works showed the error seemed to have a quadratic relation with distance and it was lower when the camera is angled 45º from the marker, increasing towards 0º and 90º. Given the distance, r, and angle, ϕ, to the camera, as in Figure 4.4a, an expression was created to compute a value for σ_i^2

$$\sigma_i^2 = ar^2 + b\left(\phi - 45\right)^2, \tag{4.4}$$

where a and b are adjustable coefficients, which were selected by trial and error, with an initial guess based on the previously mentioned studies. This is by no means the most accurate error prediction and improving this prediction could improve the final results. Nevertheless, using this rough expression is essential for improving the performance of the EKF, as will be shown in Section 5.3.3.

4.3 iCub Simulator

In order to test the implementation, the open-source simulator for the humanoid robot iCub will be used, presented in [29]. This simulator was designed to replicate the physics and dynamics of the real robot, as close as possible. The DH parameters of this robot are known exactly from Table 4.1.

The iCub library[3] provides interfaces to interact with the simulator, either by sending commands or reading information. It allows to send commands to move all joints connecting the rigid bodies, contains proprioceptive information about the joint encoders and it possesses 2 cameras to act as eyes. The Yet Another Robot Platform (YARP) middleware [30] is used to communicate with the robot, which allows building a robot control system as a collection of programs communicating in a peer-to-peer way. It provides multiple connection types (tcp, udp, multicast, local, MPI, mjpg-over-http, XML/RPC, tcpros, ...). The properties of the cameras are in Table 4.2.

The YARP middleware provides interfaces to easily interact with the environment of the simulator, including adding and moving custom objects with custom textures. By adding planes to the environment with the ArUco markers as textures, it is possible to write a small script which moves and rotates the

[2] Recall the measurement co-variance is considered to be diagonal, where σ_i is the standard deviation of the i-th measurement.
[3] The library is available in the git-hub repository `https://github.com/robotology/icub-main`.

Link	0	1	2	3	4	5	6
$\boldsymbol{a}$ [mm]	0	0	-15	15	0	0	62.5
$\boldsymbol{d}$ [mm]	-107.74	0	-152.28	0	-137.4	0	16
$\boldsymbol{\alpha}$ [°]	90	-90	-90	90	90	90	0
$\boldsymbol{o}$ [°]	-90	-90	-105	0	-90	90	180

Table 4.1: Actual Denavit-Hartenberg parameters of the iCub right arm in the iCub simulator. These parameters are very similar to the actual DH parameters of the actual iCub robot.

Resolution	f_x	f_y	c_x	c_y
640×480	514.681	514.681	320.0	240.0

Table 4.2: Image resolution and intrinsic parameters of the cameras of the iCub simulator. These parameters are very similar to the parameters from the actual iCub cameras.

markers to be placed on the palm and on the back of the hand, since the kinematics of the arm are known exactly, as can be seen in Figure 4.4b.

4.3.1 Gaze

In order to obtain samples of the hand pose using the ArUco markers, it is necessary to control the gaze of the robot for them to appear in the camera images. It is assumed the robot is able to find its hand, as long as it is physically possible to do so.

To control the gaze of the robot, it is used the gaze interface proposed in [31], which is a complete architecture for gaze control of a humanoid robot. It is able to control the neck and the eyes to track a Cartesian fixation point in space. It implements gaze stabilisation, saccadic movements, and vestibulo-ocular reflex. It is implemented on the iCub's head, which has three degrees of freedom for the neck and three degrees of freedom for the eyes. Knowing, roughly, the 3-D position of the hand, this position is given as a fixation point to the robot, using the gaze interface, and produces the movement as in Figure 4.5.

4.4 Comparison Metrics

In this Section, a few metrics will be defined for the evaluation of the performance of the different methods, regarding prediction errors and the amount of movement done by the arm.

In order to measure prediction errors, both for position and orientation, a set of N points are generated in the joint space, using a uniform distribution, at each iteration. The forward kinematics is computed using (3.6) and, from the resulting matrix, both the predicted 3D position in the Cartesian space, $\hat{\boldsymbol{p}}$, and rotation matrix, $\hat{R}$, are obtained. The respective errors are evaluated at each of these joint configurations. To obtain an overall error estimate of the calibration in the robot work-space, the arithmetic mean of both position and orientation errors is computed.

Average Position Error

For each of the N configurations, the position error is computed using the euclidean norm and the expression for the average position error results in

$$\frac{1}{N}\sum_{i=1}^{N}\|\boldsymbol{p}_i - \hat{\boldsymbol{p}}_i\| \quad \text{[mm]}. \tag{4.5}$$

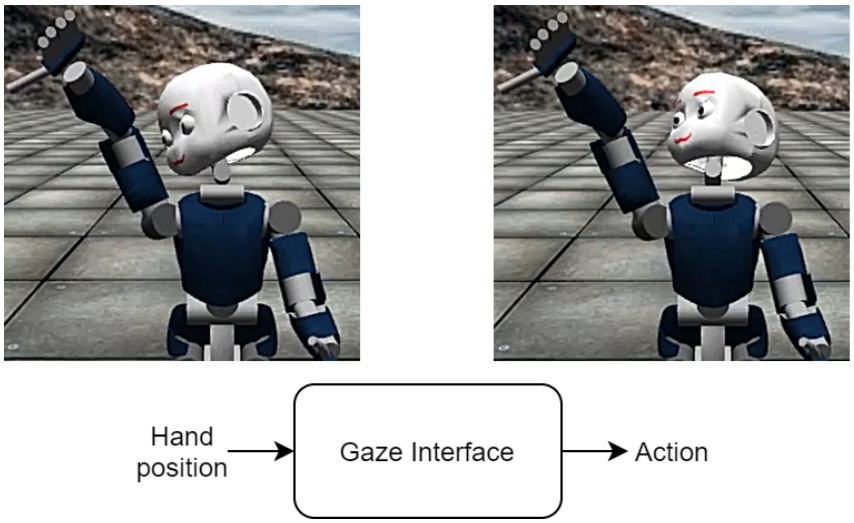


Figure 4.5: iCub using the gaze interface to look at its hand.

Average Orientation Error

For each of the N configurations, the orientation error is computed using,

$$d(\boldsymbol{R}_A, \boldsymbol{R}_B) = \sqrt{\frac{\|logm(\boldsymbol{R}_A^T \boldsymbol{R}_B)\|_F^2}{2}} \cdot \frac{180}{\pi} \quad [^\circ], \tag{4.6}$$

where $logm$ is the principal matrix logarithm and $\|\cdot\|_F$ is the Frobenius norm. The average orientation error results in

$$\frac{1}{N} \sum_{i=1}^{N} d(\boldsymbol{R}_i, \hat{\boldsymbol{R}}_i) \quad [^\circ]. \tag{4.7}$$

Accumulated Joint Movement

For each iteration of the algorithm, the accumulated joint movement is given by the sum of the movements each individual joint had to perform up to that point in past iterations. So, at iteration K, the accumulated joint movement is given by

$$\sum_{i=1}^{K} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{i-1}\|_1 \quad [^\circ], \tag{4.8}$$

where $\|\cdot\|_1$ represents the $l1$ norm and $\boldsymbol{\theta}_0$ is the known initial joint configuration.

4.5 Calibration Routine

The calibration routine algorithm runs in a loop and starts by selecting a joint configuration for the robot to move its arm to. When active learning is being used, the robot will select a joint configuration by solving the problems defined in (3.31), (3.34) or (3.36), depending on the previously selected criterion, using (3.33) as the cost function, $C(\boldsymbol{\theta})$. To solve the optimisation problems, the DIRECT algorithm will be used, described in Section 4.1. When active learning is not used, the robot will select a random joint configuration, uniformly distributed. After this decision, the arm moves to where decided and an observation of the hand pose is obtained. The EKF uses the observation for estimating the DH parameters, $\boldsymbol{x}$, by comparing it to the prediction, obtained as explained in Section 3.3.1. The experiments are divided

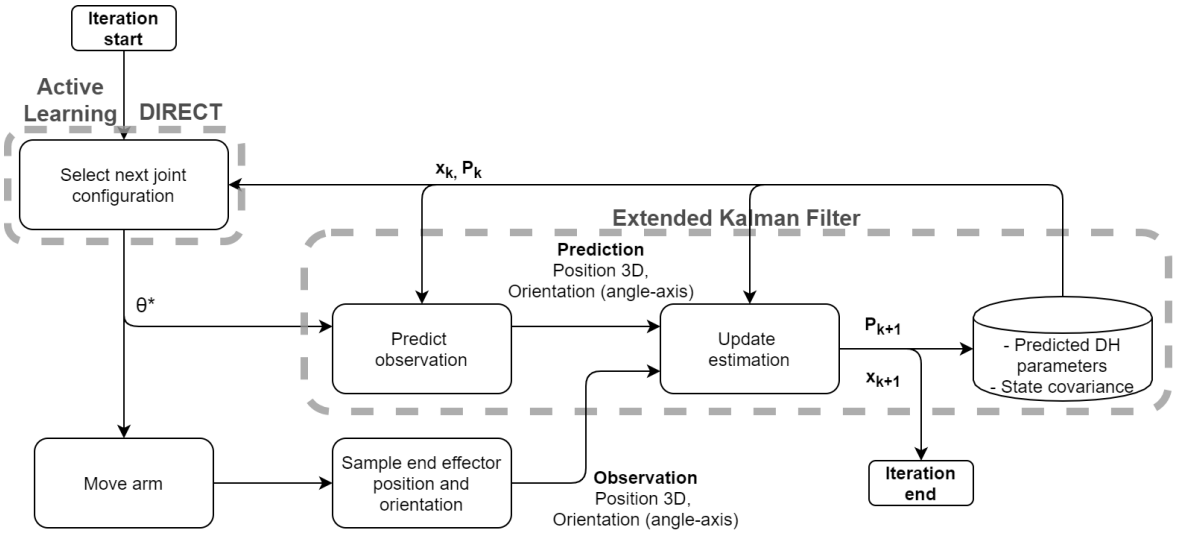


Figure 4.6: Geometric simulation flowchart to estimate the robot's Denavit-Hartenberg parameters, $\boldsymbol{x}$. The subscript k indicates the algorithm's iteration, $\boldsymbol{P}$ represents the EKF estimation co-variance and $\boldsymbol{\theta}^*$ is the selected joint configuration.

in two incremental settings. The first one, explained in Section 4.5.1, where the observation is based on geometric simulation which is a simpler implementation of the proposed system. It is assumed the end-effector pose can be retrieved, for instance, from an external vision-based algorithm and the real end-effector pose is directly retrieved on the simulation. The second one, graphical simulation, which is a more realistic scenario, where the end-effector pose is retrieved by using the iCub eyes (cameras) and fiducial markers placed on the hands. It is explained in detail in Section 4.5.2.

4.5.1 Geometric Simulation Setting

In the iCub Simulator, the DH parameters of the robot are known exactly. This provides a reliable way to sample the hand pose. It removes the error associated with the use of the ArUco markers, so it allows a quicker way to test the initial system implementation composed of the EKF, the DIRECT algorithm, connection to the YARP server and use of the iCub interface. It can also serve as validation for the use of the cost-sensitive active learning methods and it can be thought of as a setting where a more reliable way of sampling is available. Using this sampling method, the program flowchart is the one shown in Figure 4.6.

Sampling the hand pose

As already mentioned, the DH parameters of the iCub robot in the simulation environment are known exactly and are in Table 4.1. In order to obtain measurements of the pose of the hand without relying on its visual sensors, these values can be used to compute the exact pose, as explained in Section 3.3.1, using the direct kinematics. In order to maintain some realism for these samples, random Gaussian noise is added to each sample, after the exact pose is computed, with standard deviations according to Table 4.3.

4.5.2 Graphical Simulation Setting

This setting consists on the full system implementation from Figure 4.7, using ArUco marker to sample the hand pose, described in Section 4.2. The noise model, explained in Section 4.2.1, is used in the

Position	Orientation	
	Unitary axis	Angle
2 mm	8%	5°

Table 4.3: Standard deviations of the added Gaussian noise to the simulated samples for the geometric simulation setting. After the error is added to the 3 components of the unitary axis, the axis is re-normalised.

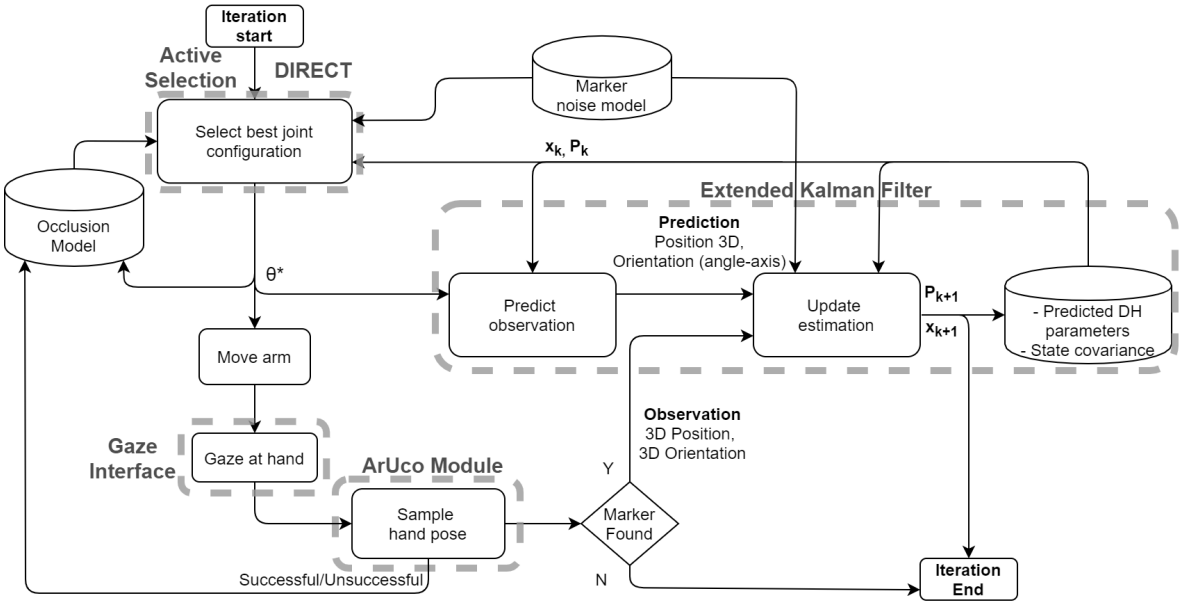


Figure 4.7: Graphical simulation flowchart to estimate the robot's DH parameters, $\boldsymbol{x}$. The subscript k indicates the algorithm's iteration, $\boldsymbol{P}$ represents the EKF estimation co-variance and $\boldsymbol{\theta}^*$ is the selected joint configuration.

joint selection step, to avoid noisy samples, and in the EKF after the sample is obtained. If the ArUco marker is not found, no update is performed on the DH parameters. Whether this was a successful or unsuccessful attempt, that information is used as training data for the non-parametric occlusion model used for the next iteration of the loop.

Chapter 5

Results

This chapter is dedicated to displaying and analysing the results obtained from the performed experiments. Section 5.1 explains the experimental conditions regarding initialisation and summarises the different methods used for joint selection to be compared. Section 5.2 is dedicated to the results obtained without using fiducial markers for obtaining the hand pose and Section 5.3 is dedicated to the results obtained using fiducial markers.

5.1 Experimental Conditions

The calibration routines from Figure 4.6 and Figure 4.7 are performed for the geometric and graphical simulations, respectively, using four different methods for the selection of the joint values of the iCub arm, which are summarised in Table 5.1. The first method is Random (R), which selects random joint configuration uniformly distributed, which usually serves as a base line for comparison with methods using active learning. The second method is the Active Learning (AL) method, which does not consider movement costs. The third and fourth methods are the cost-sensitive approaches proposed in this work, Unconstrained Cost-Sensitive Active Learning (UCSAL) and Constrained Cost-Sensitive Active Learning (CCSAL).

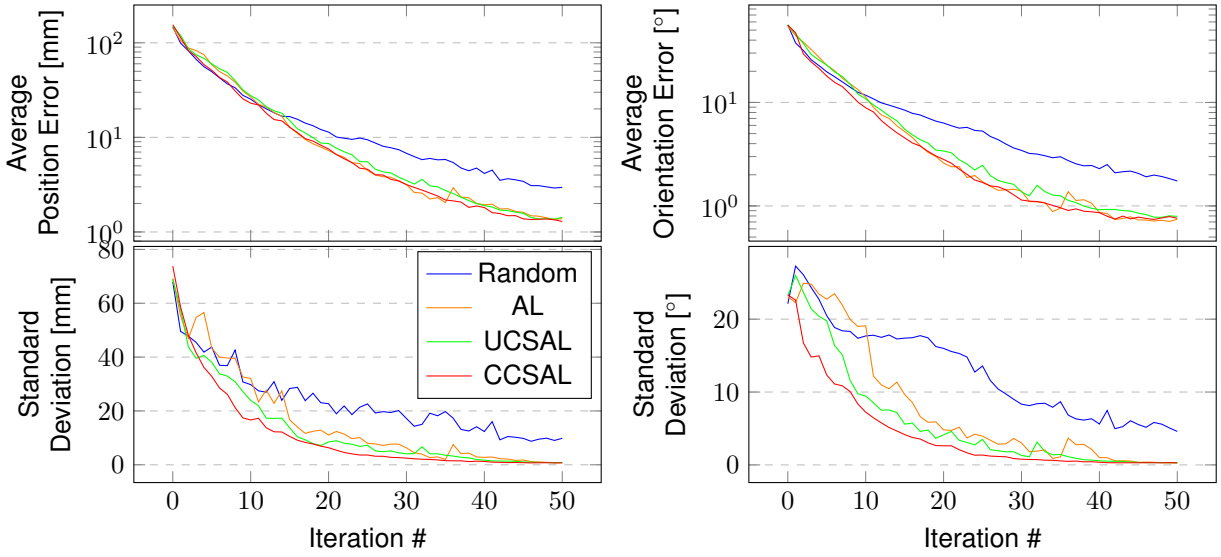
A random initial joint configuration for the right arm of the robot is set and the algorithm from Figure 2.2 executes for all different methods. The main loop of Figure 2.2 runs 50 times, which means 50 samples are taken of the hand pose. At each new run, the DH parameters of the robot are initialised with values from a uniform distribution, where the means are the actual values of the DH parameters from Table 4.1 and the width of the distribution is 30% of the highest value from all the linear and angular DH parameters, 46 mm and 54°, respectively, as summarised in Table 5.2. This is relatively high, comparing to the error it is desired to find in a robot for it to run reliably. Ideally, the calibration routine would be done before reaching this point. Lower error values help achieving convergence and increasing the speed at which the EKF converges. In order to obtain solid conclusions, the results displayed are the average of 50 repetitions of each experiment.

Method	**Joint Selection**
Random (**R**)	Uniform random selection
Active Learning (**AL**)	Solves (3.31)
Unconstrained Cost-Sensitive Active Learning (**UCSAL**)	Solves (3.34)
Constrained Cost-Sensitive Active Learning (**CCSAL**)	Solves (3.36)

Table 5.1: Summary of the different used joint selection methods.

# of samples per experiment	# of experiments	Uniform distribution width		Initial arm pose
		Linear	Angular	
50	50	46 mm	54°	Random

Table 5.2: Conditions of the performed experiments with and without fiducial markers. The uniform distribution width is regarding the random initialisation of the DH parameters.



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

Figure 5.1: Geometric simulation results of the different joint selection methods regarding the mean position and orientation errors, with respect to the loop iterations of the calibration routine and corresponding standard deviations.

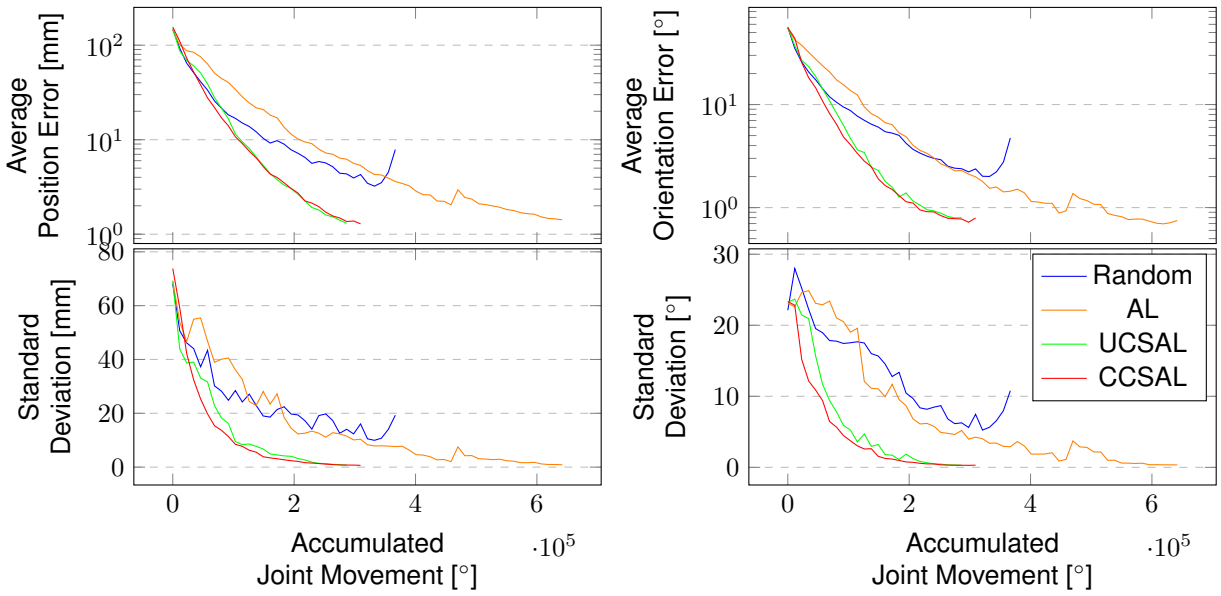
All the conditions mentioned in this Section are valid for both the graphical simulation and the geometric simulation.

5.2 Geometric Simulation Results

In this Section, the results of the implementation for the geometric simulation setting, described in Section 4.5.1, are presented, as well as their analysis/discussion.

For each method, the plots from Figure 5.1 show how the average error evolves at each iteration and the plots from Figure 5.2 show how the average error evolves as movement is performed by the arm. Both representations provide useful information, regarding the trade-offs of error reduction, number of samples and movement performed. For the proposed cost-sensitive active learning methods, the values $\gamma = 10^{-5}$ and $\delta = 0.4$ were used for the unconstrained and constrained methods, respectively, regarding (3.34) and (3.36).

By observing Figure 5.1, the first relevant observation is the improved performance of AL, in comparison to the R method. The R method often leads to uninformative samples, resulting in small error decrements. By analysing the standard deviation plots, it is possible to see how using AL produces consistent results, which is not true for R, which maintains higher standard deviation values. Looking at Fig. 5.2, it shows the AL method does, roughly, double the movement of R and it is less efficient in the amount of movement performed. For instance, after moving the arm for 2×10^5 deg, the R method has a



(a) Mean position error in millimetres with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

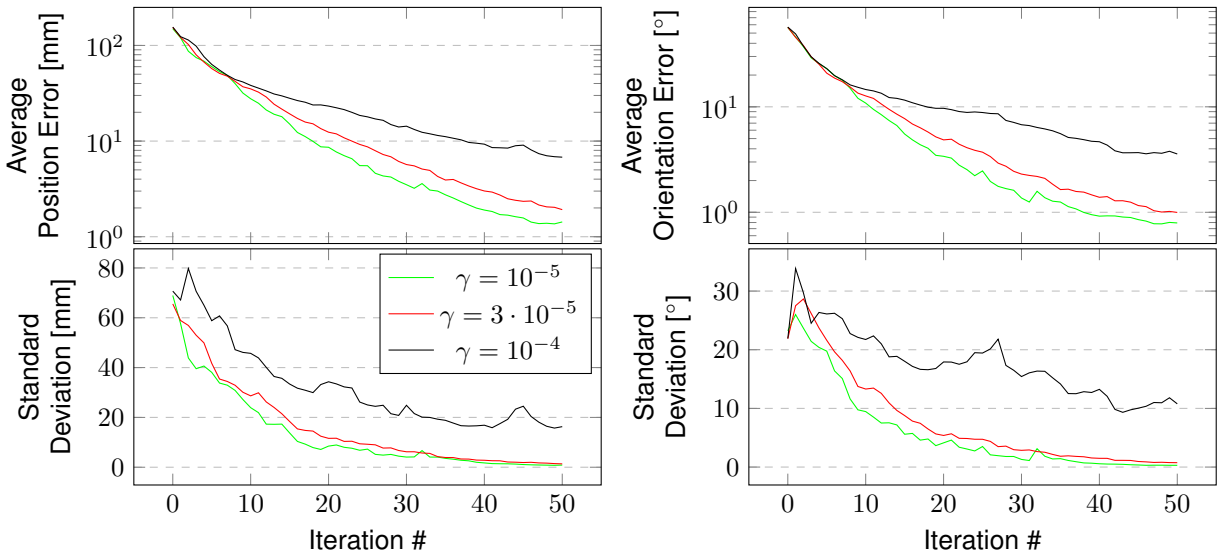
(b) Mean orientation error in degrees with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

Figure 5.2: Geometric simulation results of the different joint selection methods regarding the mean position and orientation errors, with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

lower error than the AL method. However, for the same number of iterations (assuming a constant cost for each sample) the AL method is more efficient. This is not desirable when trying to sample efficiently since, even if the R method would require more samples.

Both the cost-sensitive methods, UCSAL and CCSAL, can perform very similarly to the AL method with the selected values for γ and δ, when considering the number of samples taken in Figure 5.1. In the results from Figure 5.2, they reveal to be much more advantageous since it took, roughly, less than half the movement and the same amount of iterations to achieve similar results. The standard deviation plots from Figure 5.1, also show similar reliability, regarding the convergence of the EKF, and the cost-sensitive approaches even seem to be slightly more consistent. This may be due to the fact that, with the cost-sensitive approaches, the DIRECT algorithm searches the closer areas more thoroughly and may find better solutions to the respective optimisation problems.

Regarding the two Cost-Sensitive Active Learning (CSAL) methods, UCSAL and CCSAL, the values for the parameters in the respective optimisation problems, γ in (3.34) and δ in (3.36), were selected by finding the ones that could match the iteration wise performance of the AL method and would reduce the amount of movement significantly. When comparing the two, their lines almost overlap in the error plots from Figures 5.1 and 5.2, indicating that both methods allow similar amounts of exploration and exploitation. The only noticeable difference is in the standard deviation plots from Figures 5.1 and 5.2, where the CCSAL method seems to be slightly more consistent than UCSAL, having a lower standard deviation of the error values sampled at every moment. This may be due to the fact that the optimisation problem solved using the CCSAL method is constrained, therefore the DIRECT algorithm does not lose time searching for solutions in undesirable areas of the joint space, i.e. areas which would require the arm to move more significantly. Even though the UCSAL method also encourages searching for solutions in a closer region, the DIRECT algorithm will still explore outside of it. Another justification for this may be the fact that the optimisation problem solved for the UCSAL method, given in (3.34), deforms



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

Figure 5.3: Geometric simulation results regarding the mean position and orientation errors, with respect to the loop iterations of the calibration routine and corresponding standard deviations, using unconstrained optimisation, for different values of γ from (3.34).

the cost function defined in (3.33), by adding the distance penalty directly, defined in (3.35). This will lead to some joint configurations chosen only because of them being really close, rather than being good potential samples for the model.

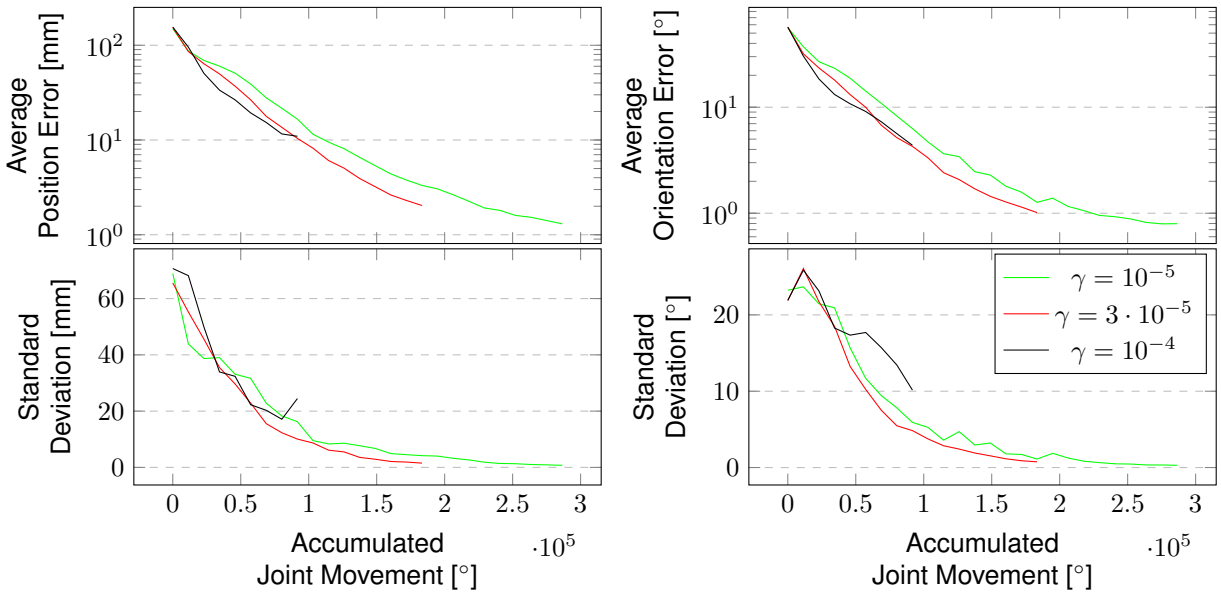
5.2.1 Unconstrained Optimisation - Influence of γ

The unconstrained optimisation for cost-sensitive Active Learning is implemented by adding a term to the cost function to penalise movement, multiplied by a constant γ, as proposed in (3.34). Different values were tested for this coefficient and the most relevant ones are plotted in Figure 5.3 and Figure 5.4.

As (3.34) suggests, when γ increases, less movement tends to happen between iterations. Looking at Figure 5.4, this effect is clearly present in the results obtained. Having a lower value, as $\gamma = 10^{-5}$, allows to reduce the error with less samples obtained, as observed in Figure 5.3. The results using the method with $\gamma = 10^{-4}$ show the consequences of a higher movement penalty. Figure 5.4 shows the movement efficiency of using $\gamma = 10^{-4}$ is similar to using $\gamma = 3 \cdot 10^{-5}$, at the cost of requiring many more iterations to eventually reach the same error values. The standard deviation graphs, both from Figure 5.4 and Figure 5.3, show that the results with $\gamma = 10^{-4}$ may be less consistent. If movement efficiency is a priority, using $\gamma = 3 \cdot 10^{-5}$ may be more advantageous than $\gamma = 10^{-5}$. It would require more samples to reach the same error values, but reducing the time the arm spends moving may make up for the extra computation time. Indeed, this will depend on computing power and on the velocity the joints are rotating.

5.2.2 Constrained Optimisation - Influence of δ

The constrained optimisation for cost-sensitive Active Learning is implemented by restricting the optimisation search space to an area around the current joint configuration, as proposed in (3.36). Different values for δ are displayed in the results from Figures 5.5 and 5.6.



(a) Mean position error in millimetres with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

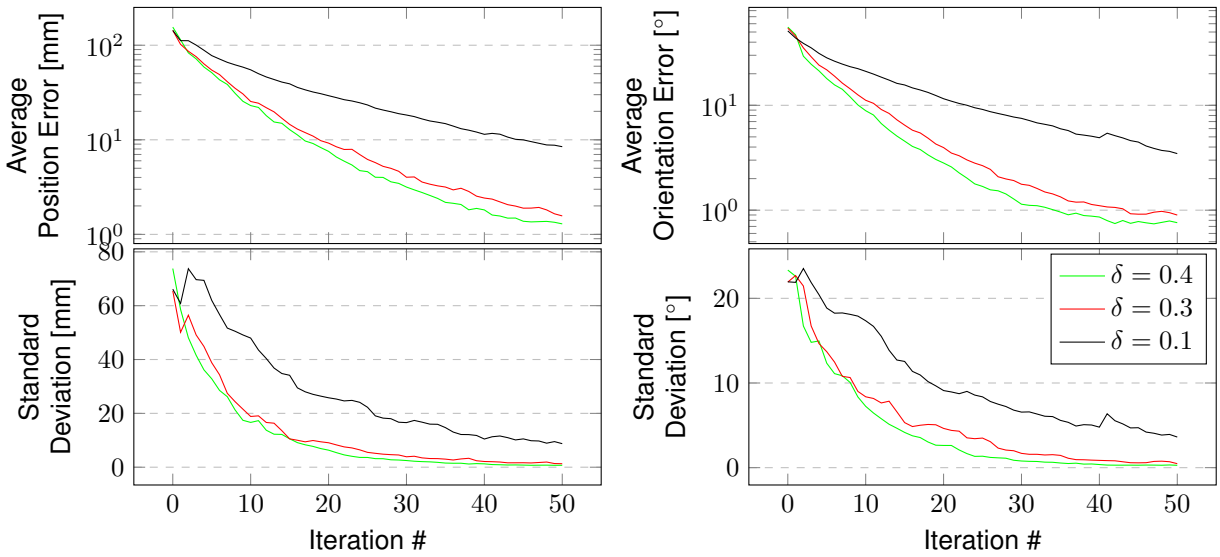
Figure 5.4: Geometric simulation results regarding the mean position and orientation errors, with respect to the accumulated joint movement during the calibration routine, using unconstrained optimisation, for different values of γ from (3.34).

The influence of δ in the performance of the calibration routine is similar to the effect γ has for the unconstrained method. Lower values of δ reduce the optimisation region, therefore less movement happens between iterations. Having a higher value of δ, such as $\delta = 0.4$, meaning that each joint is allowed to move 40% of its maximum movement possible at each iteration, provides a solution capable of reducing the number of samples needed to reduce the error and decreases the movement needed significantly. Decreasing δ to 0.3 does not show particular improvements, when comparing to $\delta = 0.4$, and reinforces that a balance must be found to be able to reduce movement, but maintain the ability to find decent joint configurations to sample. The movement efficiency is similar, as is seen in Figure 5.6, but it would require a larger number of iterations to reach similar final errors in Figure 5.5. The same is true for using $\delta = 0.1$.

5.3 Graphical Simulation Results

The results in Figure 5.7 and 5.8 were taken using the same methods from Section 5.2, where the selected values for γ and δ, regarding (3.34) and (3.36), were $\gamma = 10^{-5}$ and $\delta = 0.5$, respectively.

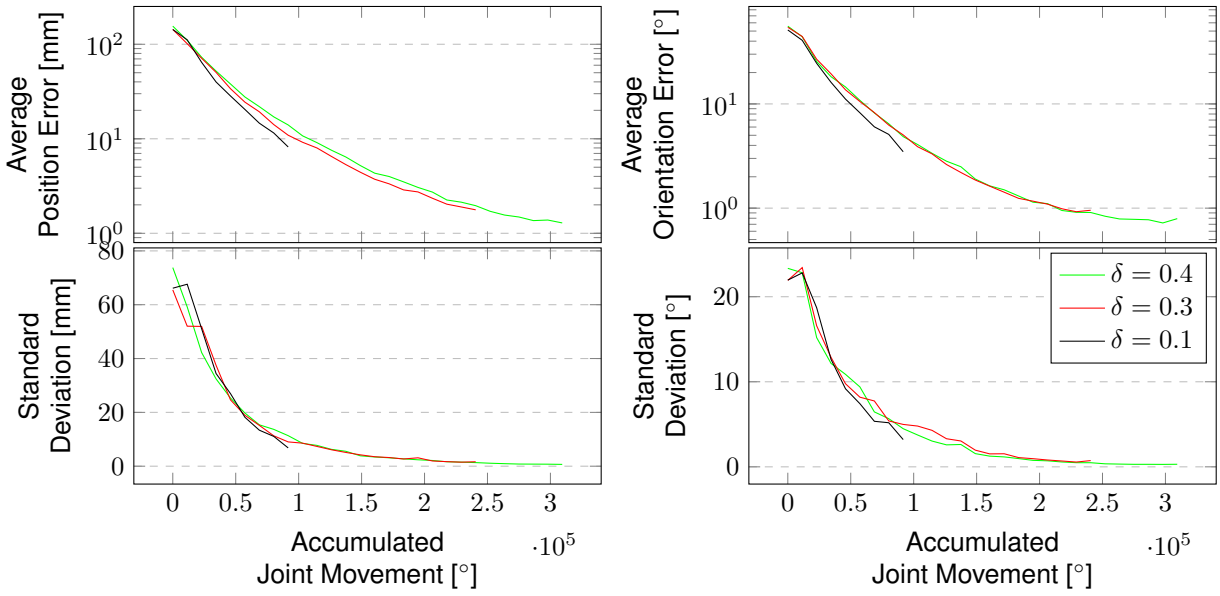
By observing the plots from Figure 5.7, there are two major differences, when comparing them to the results of the geometric simulation, in Figure 5.1. Firstly, the final errors obtained for the position and orientation errors are much larger for all methods. The pose measurement method using fiducial markers introduces a significantly larger amount of noise into the system, resulting in worse estimations. This could be improved by using cameras with a better resolution, which are not available in the iCub simulator. The second difference is regarding the position errors. In Figure 5.7a, the R method seems to perform much closer to the active learning methods than in Figure 5.1a. This may be, again, due to the limitations imposed by the pose measurement method. When using the fiducial markers, it seems there is not much of a margin to improve after a few samples and the random method catches up rather



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

Figure 5.5: Geometric simulation results regarding the mean position and orientation errors, with respect to the loop iterations of the calibration routine and corresponding standard deviations, using constrained optimisation, for different values of δ from (3.34).

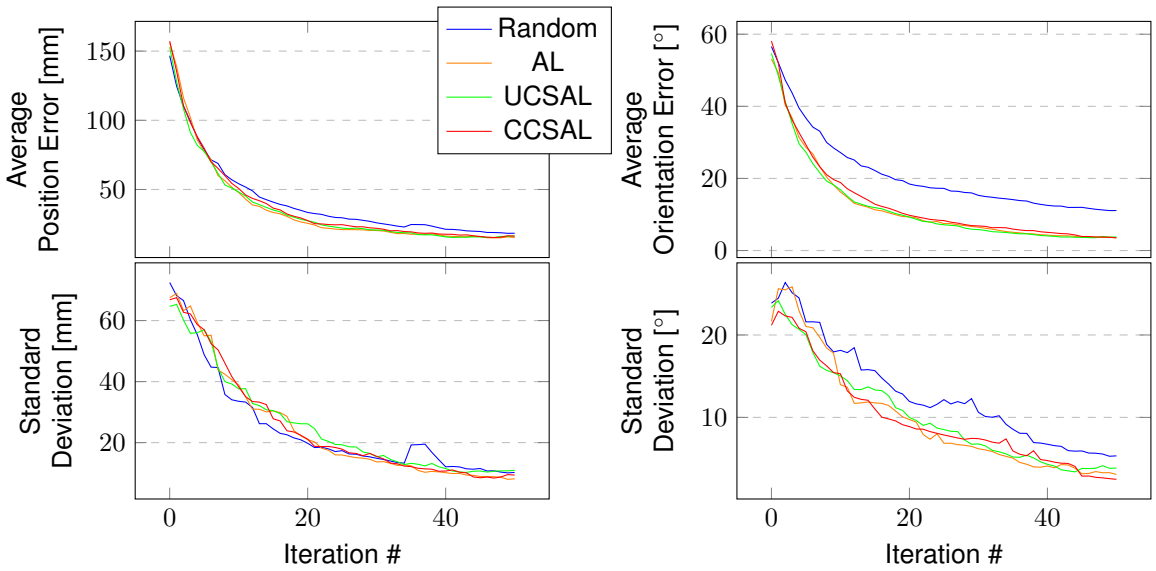


(a) Mean position error in millimetres with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

Figure 5.6: Geometric simulation results regarding the mean position and orientation errors, with respect to the accumulated joint movement during the calibration routine, using constrained optimisation, for different values of δ from (3.34), with respect to joint movement and corresponding standard deviations.

quickly. Oppositely, in Figure 5.7b, the random method can hardly improve the orientation estimate of the hand, performing much worse than the active learning methods. This may mean that choosing a joint configuration more likely to have a visible marker in a favourable pose, as explained in Section 4.2.1, is



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

Figure 5.7: Graphical simulation results of the different joint selection methods regarding the mean position and orientation errors, with respect to the loop iterations of the calibration routine.

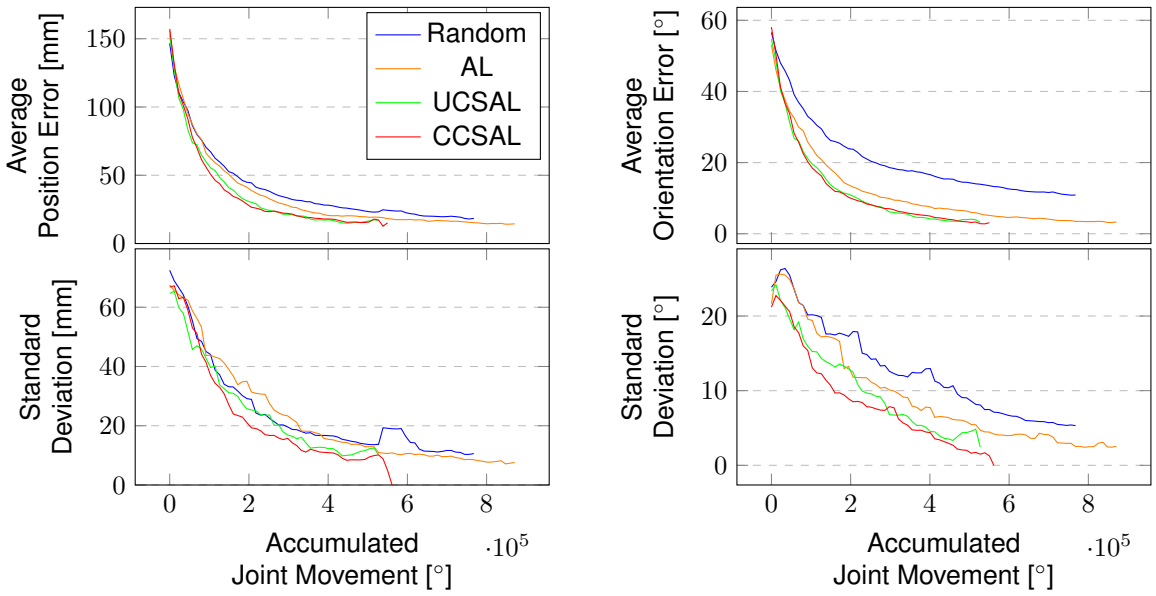
more important for estimating orientation than position. When observing the standard deviation plots, all methods seem to be equally consistent for position error and the random method is slightly less consistent for orientation error.

The efficiency of the cost-sensitive methods, UCSAL and CCSAL, seems to be less noticed in Figure 5.8 than in Figure 5.2, but it is still fairly noticeable regarding the evolution of the orientation errors. Both cost-sensitive methods require less movement than the AL method to achieve the same results in the same number of iterations. The standard deviation plots also show how both the cost-sensitive methods are more consistent at any given joint movement performed. The CCSAL method also seems to be slightly more consistent than the UCSAL method. This may not be significant, but it was also the case for the geometric simulation results. This is most likely due to the constraints added for CCSAL, allowing the DIRECT algorithm to search the region more thoroughly, or the fact that the UCSAL method explicitly penalises movement in the cost function, whereas the CCSAL method gives equal weight to farther and closer joint configurations, as long as they are within the bounds.

5.3.1 Unconstrained Optimisation - Influence of γ

When selecting optimal joint configuration using unconstrained optimisation for movement efficiency, i.e. solving (3.34), different values of γ were tested to study its influence on the calibration performance. The results of those tests are plotted in Figures. 5.9 and 5.10.

In Figure 5.9 it is visible how the position and orientation errors evolve for the iterations of the calibration routine for multiple γ values. As γ increases, more relative importance is given to selecting closer consecutive joint configurations. This may reduce the quality of the samples obtained from the joint configurations selected, reducing performance. This is slightly noticeable for $\gamma = 10^{-4}$ and fairly noticeable for $\gamma = 10^{-3}$. The standard deviation plots show how inconsistent the results are for $\gamma = 10^{-3}$, since it becomes much harder to select optimal joint configurations, and much slower to move away from joint configurations where the marker is not visible.



(a) Mean position error in millimetres with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

Figure 5.8: Graphical simulation results of the different joint selection methods regarding the mean position and orientation errors, with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

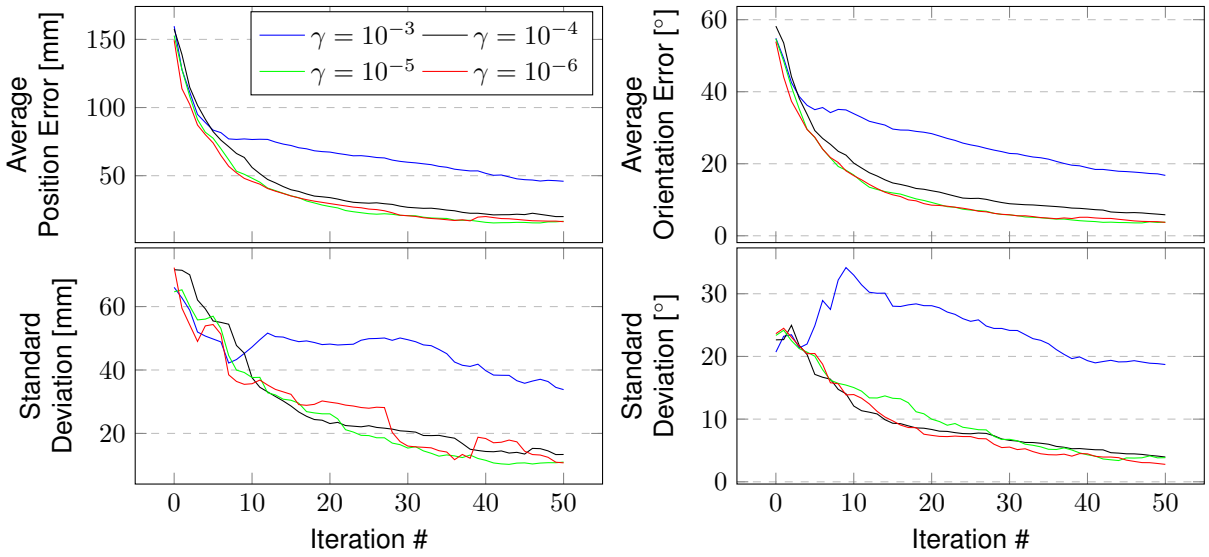
Figure 5.10 shows how the position and orientation errors evolve as movement is performed by the arm. For both errors, efficiency increases as γ increases, i.e. the curve is steeper in the beginning, although for $\gamma = 10^{-3}$ the standard deviation plots show the results are more inconsistent. Increasing γ past this point is not advantageous. $\gamma = 10^{-4}$ is a value for which the calibration routine seemed to do very well regarding both sampling and movement efficiency, performing almost a quarter and half of the movement of when $\gamma = 10^{-6}$ and $\gamma = 10^{-5}$, respectively, and performing almost identically, regarding the results from Figure 5.9, meaning informative samples can still be selected without requiring large movements.

5.3.2 Constrained Optimisation - Influence of δ

When selecting optimal joint configuration using constrained optimisation for movement efficiency, i.e. solving (3.36), different values of δ were tested to study its influence on the calibration performance. The results of those tests are plotted in Figs. 5.11 and 5.12.

Figure 5.11 shows how the average position and orientation errors evolve at each iteration. As expected, as δ decreases, performance decreases, since the search space is being reduced, it is more likely that sub-optimal joint configurations are selected. This decrease is very slight and only fairly noticeable when $\delta = 0.2$, where both the average position and orientation errors decrease slower. The standard deviation plots also show it becomes far less consistent at this value. This should be mainly due to the random initial joint configuration given to the arm. If it is fortunate to start in an area of the joint space where it is easy to sample the hand pose, it performs fairly well, but if it starts in an area of the joint space where it is hard to visualise the markers, that run does worse, since it is slower at leaving those areas due to the stricter movement restrictions.

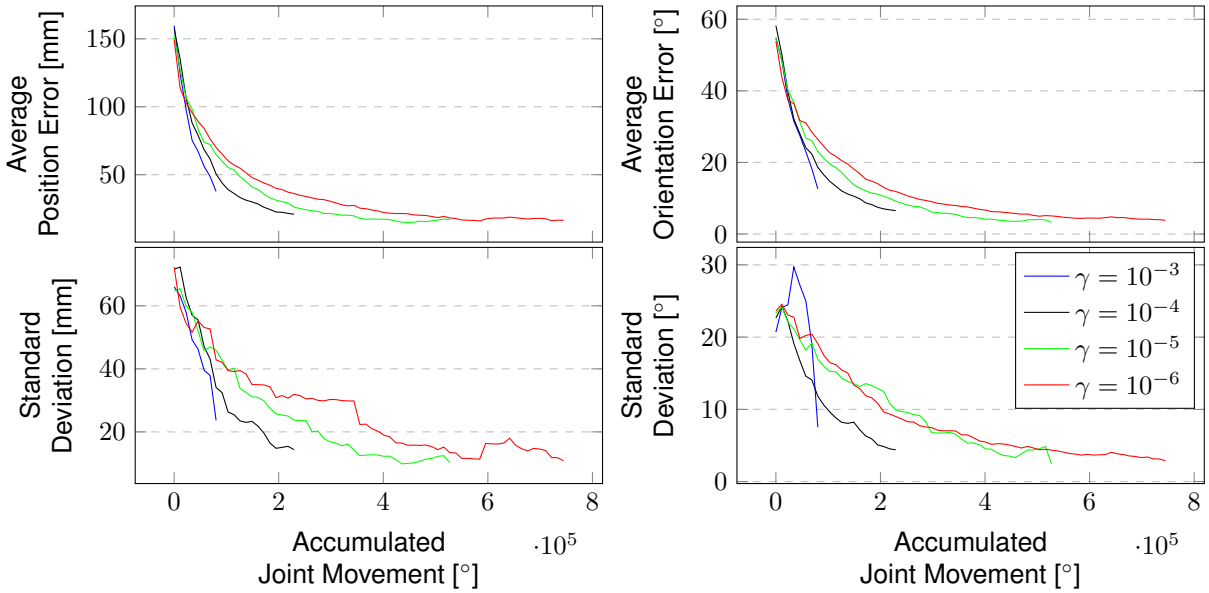
Figure 5.12 shows how the average position and orientation errors evolve as movement is performed



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

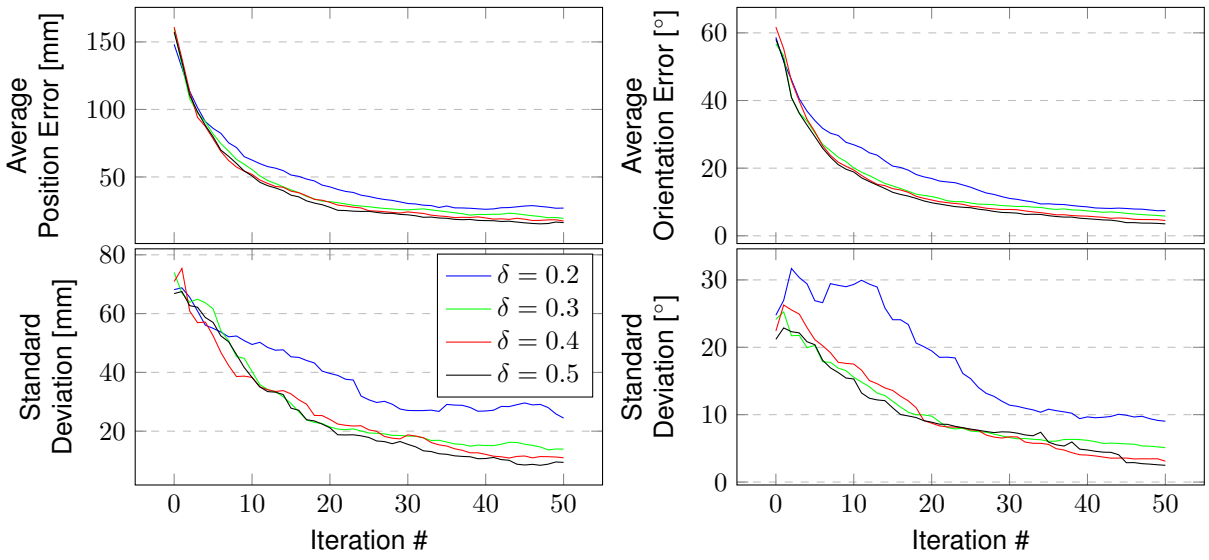
Figure 5.9: Graphical simulation results regarding the mean position and orientation errors, with respect to the loop iterations of the calibration routine, using unconstrained optimisation, for different values of γ from (3.34).



(a) Mean position error in millimetres with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

Figure 5.10: Graphical simulation results of the different joint selection methods regarding the mean position and orientation errors, with respect to the accumulated joint movement during the calibration routine. using unconstrained optimisation, for different values of γ from (3.34).



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

Figure 5.11: Graphical simulation results regarding the mean position and orientation errors, with respect to the loop iterations of the calibration routine, using constrained optimisation, for different values of δ from (3.34).

Method	**Measurement Noise Model**	
	EKF	**Joint Value Selection**
Naive EKF	No	No
Naive AL	Yes	No
AL	Yes	Yes

Table 5.3: Summary of the different experiments performed to evaluate the performance impact of the measurement noise model.

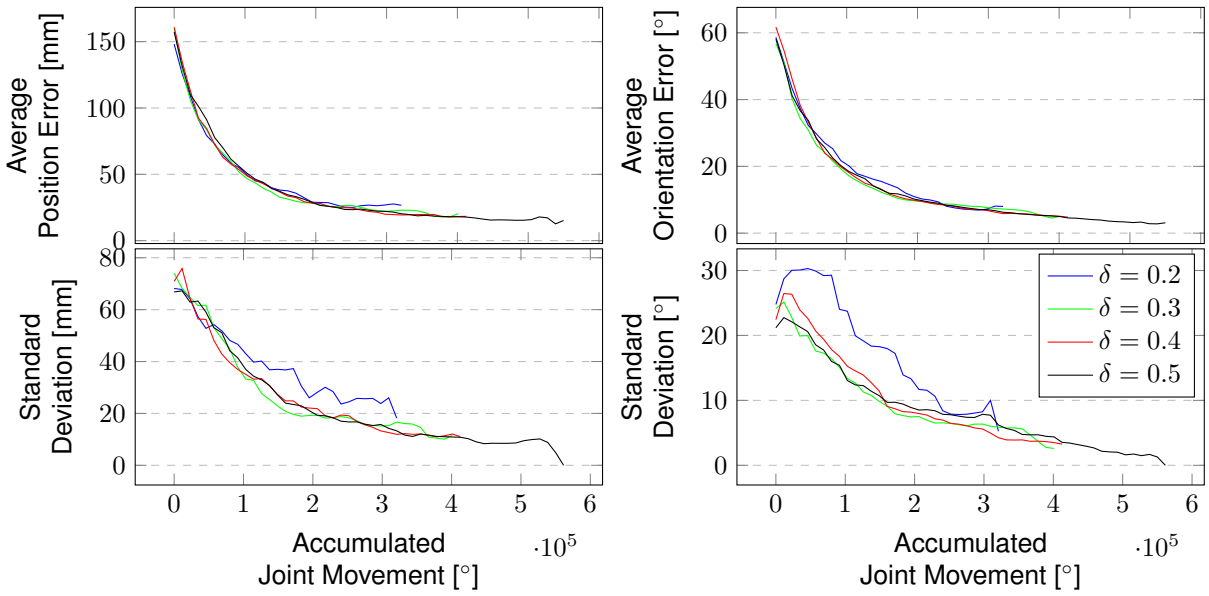
by the arm joints. All δ values seem to perform similarly, since all lines remain very close or overlapped. $\delta = 0.5$ ends with less error both for position and orientation, but more movement was performed and all methods seem to be able to reach the same point if the same movement was performed. Of course, these would have to obtain more samples, adding to execution time.

5.3.3 Predicting Measurement Noise

This Section aims to show how the use of the rough prediction of the measurement noise from (4.4) impacts the performance of the EKF.

Using the standard AL method, for simplicity, two groups of 50 executions were performed, where one group had the measurement noise co-variance matrix, $\boldsymbol{R}$, from (3.21) constant, $\boldsymbol{R} = \sigma^2 I$, which is called the Naive EKF approach, and the other group changed $\boldsymbol{R}$ according to Section 4.2.1, but did not predict it in the active learning step, which is called the Naive AL approach. The average results regarding the position and orientation errors are plotted in Fig 5.13 and compared with the used method throughout this thesis, AL. This information is summarised in Table 5.3.

In Figure 5.13, it is possible to see the differences in performance for both conditions. The Naive EKF approach achieves the least performance, since it is not capable of reducing the impact of measurements more likely to be wrong on the EKF. The Naive AL approach performs slightly better, but,



(a) Mean position error in millimetres with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the accumulated joint movement during the calibration routine and corresponding standard deviations.

Figure 5.12: Graphical simulation results regarding the mean position and orientation errors, with respect to the accumulated joint movement during the calibration routine, using constrained optimisation, for different values of δ from (3.36).

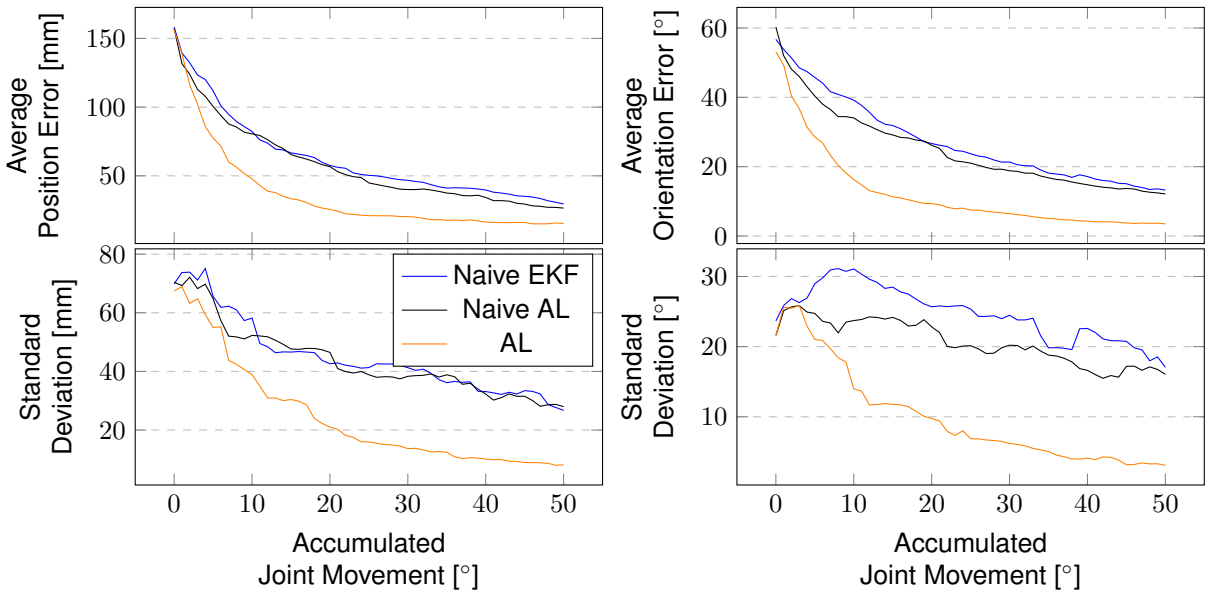
by not predicting the expected error in a sample before selecting the next joint configuration, the EKF converges slower, since the samples obtained are more likely to be less reliable. Both the methods converge much slower than the full method used in this thesis. The standard deviation plots also show the inconsistency of the Naive approaches, which is due to the noisy measurements entering the system. The Naive AL approach does change the measurement noise co-variance matrix according to the measured hand pose, but, even though this reduces the impact of a noisy measurement, it still may disturb the estimation, since (4.4) is not the most accurate prediction. Incorporating the estimation (4.4) in the selection of the next joint configuration seems to be more important, promoting the acquisition of samples more likely to be less noisy and achieving faster convergence.

This Section shows how using the rough expression from (4.4) to estimate measurement noise when selecting new joint configurations is advantageous for the results obtained using ArUco markers. It also shows how using it only for the EKF does not improve the performance significantly, but this may mean the prediction from (4.4) could be improved. This is possible by performing similar studies to [27] and [28] with the ArUco markers in the iCub simulator.

5.3.4 Marker Occlusion

Some statistics were obtained regarding the number of discarded samples in the 50 iterations of the calibration routine and they are shown in Figure 5.14. These are discarded due to occlusion of the marker or due to the ArUco module not being able to locate the marker in the camera images. Every time this happens, it means movement is performed, yet no sample is obtained. The statistics obtained are regarding the main results from Figures 5.7 and 5.8.

Figure 5.14 shows the active learning methods tend to suggest fewer arm configurations where the marker is hidden by its own body. This shows the impact of the smooth beta distributions, explained in



(a) Mean position error in millimetres with respect to the loop iterations of the calibration routine and corresponding standard deviations.

(b) Mean orientation error in degrees with respect to the loop iterations of the calibration routine and corresponding standard deviations.

Figure 5.13: Graphical simulation results regarding the use of the proposed noise model from Section 4.2.1 in the EKF and active joint selection. Naive EKF does not use the model at any point, while Naive AL uses the model only in the EKF, but not for active joint selection, and AL uses the noise model in both instances.

Section 3.6. It is crucial to avoid failed sampling attempts, since, after a few failed attempts, it is able to discourage those and the surrounding joint configurations in the joint selection step. The CCSAL method seems to discard slightly more samples than the other active learning methods. Since the CCSAL method reduces the search bounds to reduce movement, occasionally, the search space may be reduced to one where it is harder to find the markers, while the other methods are allowed to leave those regions quickly. Even though the CCSAL method discards more samples on average, the results from Figure 5.8 showed it is as efficient as the UCSAL method and even has lower standard deviation values. Reducing search space creates a trade-off between being able to search it more thoroughly, but increasing the chance of being in a region where it is hard to spot the markers.

Improving these results would be difficult without more prior knowledge. The smooth beta distribution could be trained before the calibration routine instead of starting with no prior knowledge, as done in this thesis, but having too much training data could increase the computation time significantly, since (3.42) has complexity $\mathcal{O}(n)$ and the DIRECT algorithm must compute it for every point it evaluates.

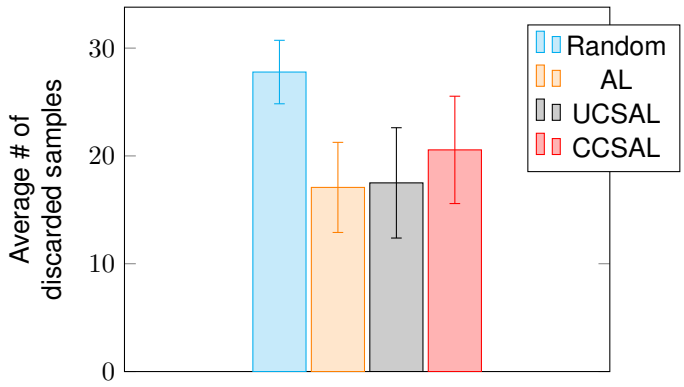


Figure 5.14: Average number of discarded samples due to marker occlusion for each joint selection method in the graphical simulation results.

Chapter 6

Conclusions

This thesis proposed a cost-sensitive active learning approach to estimate the DH parameters of 7 joints of the iCub arm, using measurements of the hand pose, in order to prioritise movement efficiency. This was achieved in two separate manners, revealing similar results. Both use the A-optimality criterion for active learning. The first, UCSAL, combines the criterion with a movement penalty in the cost function to discourage movement, while the second, CCSAL, reduces the search space.

The results show there is an advantage in discouraging or restricting movement during the optimisation stage. It is possible to reduce the movement performed by roughly half and still maintain the iteration wise performance. If movement efficiency is a priority, one can restrict the movement even more, at the cost of more iterations. It is worth mentioning, more iterations does not mean lower time-efficiency, since reducing the amount of time spent moving may make up for the extra computing time. Indeed, it will depend on the computing power and the speed at which the arm moves.

The cost-sensitive active learning methods, UCSAL and CCSAL, showed similar performances with only slight differences, such as the constrained method discarded slightly more samples due to marker occlusion and the standard deviation plots showed it was able to obtain slightly more consistent results than the unconstrained method. These differences were not significant and the disadvantages of both methods could only be showed clearly in more extreme cases, where the movement penalty was much higher or when the search space was made to small, for UCSAL and CCSAL, respectively. Optimal choices for each of the exploration parameters, γ and δ, requires an evaluation of the priorities of the calibration procedure. If movement efficiency is a priority, higher values of γ or lower values of δ provide that, at the cost of needing a few more samples to reach lower error values, adding to execution time. Computing power and movement speed would also be relevant in this matter, since if both of these are high, the overhead of acquiring a sample is lower.

Using ArUco markers in the iCub simulator provided an insight to a more realistic setting, where the samples obtained are impacted by measurement error. Even in these conditions, the active learning showed robustness, still performing better than random exploration. The cost-sensitive methods were still more efficient than the standard active learning method and maintained a similar performance regarding the number of samples needed. The results also showed the importance of having prior knowledge about measurement noise, especially to avoid noisy samples, since when the noise model was applied only to the EKF there was only a slight improvement. This may be due the fact that it is not a rigorous model, but it is as an attempt to approximate previously performed studies on similar markers. Using the non-parametric occlusion model proved to be successful in predicting whether the markers were visible to the cameras, even with little training data, obtained only during the actual calibration routine. Not only it prevented the robot from repeatedly selecting a joint configuration with hidden markers, as it increased the number of samples acquired, significantly, when comparing to random joint selection.

6.1 Future Work

For future work, it would be interesting to proceed to an implementation on the actual iCub robot to evaluate how the methods perform in the real world. It could also be built a more adequate noise model to check if it could improve the results significantly, since Section 5.3.3 showed the model only increased the estimation performance when combined with the active learning criterion to select the next joint configuration, but not when used only on the EKF. This can be achieved by performing similar studies to [27] and [28], but using the ArUco markers in the iCub simulator. Finally, the results could also be improved if both cameras would be used to sample the pose of the ArUco markers, similarly to the work done in [8], which showed significant estimation improvements. This could add some robustness to measurement noise and more information regarding the reliability of each individual sample. Furthermore, using both cameras for pose detection with ArUco markers would not add any significant overhead, since it is fairly quick in comparison with movement and optimisation times.

Bibliography

[1] M. S. A. Graziano and M. M. Botvinick, "How the brain represents the body: insights from neurophysiology and psychology," *Common mechanisms in perception and action: Attention and performance XIX*, vol. 19, pp. 136–157, 2002.

[2] M. Hoffmann, "Body models in humans, animals, and robots," *arXiv preprint arXiv:2010.09325*.

[3] A. v. d. Meer, "Keeping the arm in the limelight: advanced visual control of arm movements in neonates.," *European Journal of Paediatric Neurology*, vol. 1, no. 4, pp. 103–8, 1997.

[4] B. Settles, "Active Learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 6, pp. 1–114, 6 2012.

[5] B. Burger, P. M. Maffettone, V. V. Gusev, C. M. Aitchison, Y. Bai, X. Wang, X. Li, B. M. Alston, B. Li, R. Clowes, N. Rankin, B. Harris, R. S. Sprick, and A. I. Cooper, "A mobile robotic chemist," *Nature*, vol. 583, no. 7815, pp. 237–241, 2020.

[6] P. Vicente, L. Jamone, and A. Bernardino, "Online body schema adaptation based on internal mental simulation and multisensory feedback," *Frontiers Robotics AI*, vol. 3, no. MAR, 2016.

[7] R. Zenha, P. Vicente, L. Jamone, and A. Bernardino, "Incremental adaptation of a robot body schema based on touch events," *2018 Joint IEEE 8th International Conference on Development and Learning and Epigenetic Robotics, ICDL-EpiRob 2018*, pp. 119–124, 2018.

[8] K. Stepanova, T. Pajdla, and M. Hoffmann, "Robot Self-Calibration Using Multiple Kinematic Chains-A Simulation Study on the iCub Humanoid Robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1900–1907, 2019.

[9] U. Martinez-Hernandez, T. J. Dodd, M. H. Evans, T. J. Prescott, and N. F. Lepora, "Active sensorimotor control for tactile exploration," *Robotics and Autonomous Systems*, vol. 87, pp. 15–27, 2017.

[10] U. Martinez-Hernandez, T. J. Dodd, and T. J. Prescott, "Feeling the Shape: Active Exploration Behaviors for Object Recognition with a Robotic Hand," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 12, pp. 2339–2348, 2018.

[11] Q. Lu, M. Van der Merwe, and T. Hermans, "Multi-Fingered Active Grasp Learning," 6 2020.

[12] A. Ribes, J. Cerquides, Y. Demiris, and R. Lopez de Mantaras, "Active Learning of Object and Body Models with Time Constraints on a Humanoid Robot," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 8, no. 1, pp. 26–41, 2015.

[13] R. Martinez-Cantin, M. Lopes, and L. Montesano, "Body schema acquisition through active learning," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 1860–1866, 2010.

[14] 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.

[15] T. Matsubara and K. Shibata, "Active tactile exploration with uncertainty and travel cost for fast shape estimation of unknown objects," *Robotics and Autonomous Systems*, vol. 91, pp. 314–326, 2017.

[16] S. Ottenhaus, L. Kaul, N. Vahrenkamp, and T. Asfour, "Active Tactile Exploration Based on Cost-Aware Information Gain Maximization," *International Journal of Humanoid Robotics*, vol. 15, no. 1, pp. 1–21, 2018.

[17] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. Advanced Textbooks in Control and Signal Processing, Springer London, 2010.

[18] N. Reid, "Estimation," *International Encyclopedia of Statistical Science*, pp. 455–459, 2011.

[19] R. Schneider and C. Georgakis, "How to NOT make the extended kalman filter fail," *Industrial and Engineering Chemistry Research*, vol. 52, no. 9, pp. 3354–3362, 2013.

[20] J. O'Neil, "An Evaluation of Selection Strategies for Active Learning with Regression," *Dublin Institute of Technology*, 2015.

[21] L. Montesano and M. Lopes, "Learning grasping affordances from local visual descriptors," *2009 IEEE 8th International Conference on Development and Learning, ICDL 2009*, pp. 1–6, 2009.

[22] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2. 2006.

[23] B. Schölkopf and A. J. Smola, *Learning with Kernels*. MIT, 2002.

[24] D. R. Jones, C. D. Perttunen, and B. E. Stuckman, "Lipschitzian optimization without the Lipschitz constant," *Journal of Optimization Theory and Applications*, vol. 79, no. 1, pp. 157–181, 1993.

[25] S. G. Johnson, "The NLopt nonlinear-optimization package," *http://ab-initio.mit.edu/nlopt*.

[26] S. Garrido-Jurado, R. Muñoz-Salinas, F. J. Madrid-Cuevas, and M. J. Marín-Jiménez, "Automatic generation and detection of highly reliable fiducial markers under occlusion," *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, 2014.

[27] D. F. Abawi, J. Bienwald, and R. Dörner, "Accuracy in optical tracking with fiducial markers: An accuracy function for ARToolKit," *ISMAR 2004: Proceedings of the Third IEEE and ACM International Symposium on Mixed and Augmented Reality*, no. January 2004, pp. 260–261, 2004.

[28] K. Pentenrieder, P. Meier, and G. Klinker, "Analysis of Tracking Accuracy for Single-Camera Square-Marker-Based Tracking," *Proc. Dritter Workshop Virtuelle und Erweiterte Realität der GI-Fachgruppe VR/AR*, no. August 2016, p. 15, 2006.

[29] V. Tikhanoff, A. Cangelosi, P. Fitzpatrick, G. Metta, L. Natale, and F. Nori, *An open-source simulator for cognitive robotics research: The prototype of the iCub humanoid robot simulator*. 2008.

[30] G. Metta, P. Fitzpatrick, and L. Natale, "YARP: Yet Another Robot Platform," *International Journal on Advanced Robotics Systems*, 2006.

[31] A. Roncone, U. Pattacini, G. Metta, and L. Natale, "A cartesian 6-DoF gaze controller for humanoid robots," *Robotics: Science and Systems*, vol. 12, 2016.