Electronics and Computer Science
Faculty of Physical Sciences and Engineering
University of Southampton

Yi Farn Lim

10 December, 2021

Improving robotic grasping agility using Iterative Learning Control

Project supervisor: Prof. Christopher Freeman
Second examiner: Dr. Nicolas Green

A project progress report submitted for the award of
MEng Electrical and Electronics Engineering

Statement of Originality

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.

You must change the statements in the boxes if you do not agree with them.

We expect you to acknowledge all sources of information (e.g. ideas, algorithms, data) using citations. You must also put quotation marks around any sections of text that you have copied without paraphrasing. If any figures or tables have been taken or modified from another source, you must explain this in the caption and cite the original source.

I have acknowledged all sources, and identified any content taken from elsewhere.

If you have used any code (e.g. open-source code), reference designs, or similar resources that have been produced by anyone else, you must list them in the box below. In the report, you must explain what was used and how it relates to the work you have done.

I have not used any resources produced by anyone else.

You can consult with module teaching staff/demonstrators, but you should not show anyone else your work (this includes uploading your work to publicly-accessible repositories e.g. Github, unless expressly permitted by the module leader), or help them to do theirs. For individual assignments, we expect you to work on your own. For group assignments, we expect that you work only with your allocated group. You must get permission in writing from the module teaching staff before you seek outside assistance, e.g. a proofreading service, and declare it here.

I did all the work myself, or with my allocated group, and have not helped anyone else.

We expect that you have not fabricated, modified or distorted any data, evidence, references, experimental results, or other material used or presented in the report. You must clearly describe your experiments and how the results were obtained, and include all data, source code and/or designs (either in the report, or submitted as a separate file) so that your results could be reproduced.

The material in the report is genuine, and I have included all my data/code/designs.

We expect that you have not previously submitted any part of this work for another assessment. You must get permission in writing from the module teaching staff before re-using any of your previously submitted work for this assessment.

I have not submitted any part of this work for another assessment.

If your work involved research/studies (including surveys) on human participants, their cells or data, or on animals, you must have been granted ethical approval before the work was carried out, and any experiments must have followed these requirements. You must give details of this in the report, and list the ethical approval reference number(s) in the box below.

My work did not involve human participants, their cells or data, or animals.

Table of Contents

1. Literature review

1.1 Overview of robotics grasping

Robot’s manipulators nowadays are capable of grasping objects in a controlled environment with a fix task moving from the same point A to point B [1][2]. It is difficult for robots to interact with objects far beyond what it is programmed for as unlike humans, robots do not have millions of sensory neurons in their gripper, therefore it is hard for robots to feel the object and grasp it.

In the task of designing better grippers both hardware and control algorithm aspect, one of the largest e-commerce company, Amazon hosts a competition annually to challenge participants to build and demonstrate grippers that is capable of sorting a box full of random items to their designated boxes. [1]

Amazon, one of the largest e-commerce company hosted an annual competition to challenge teams of researchers and engineers to build a robot that is capable of identifying items from various boxes and sort them accordingly [1].

One of the leading grasping research project Dex-Net 4.0 iteration is able to separate objects with an accuracy of 95% with a speed of 300 picks per hour [3]. An average human on the other hand is able to pick 400 to 600 items per hour [1]. This is a reduction of average of about 62% efficiency between robot grasping and human.

1.2 Problem analysis

Robot grasping speed is still very far off of what humans are capable. Besides improving robotic hardware such as better gripper, gripper with sensors tactically placed to gather as much information as possible; another area of focus is to improve the control algorithm of the gripper both in detecting objects and in gripper the object in a fast pace while ensuring it does not damage the object. Therefore, to improve robotic grasping, the following criteria should be fulfilled:

1. Detecting objects of different sizes reliability.
2. Grasp the object with care, using minimal force.
3. Able to correct any mishap from previous iteration.
4. Grasp in a smooth, continuous motion.

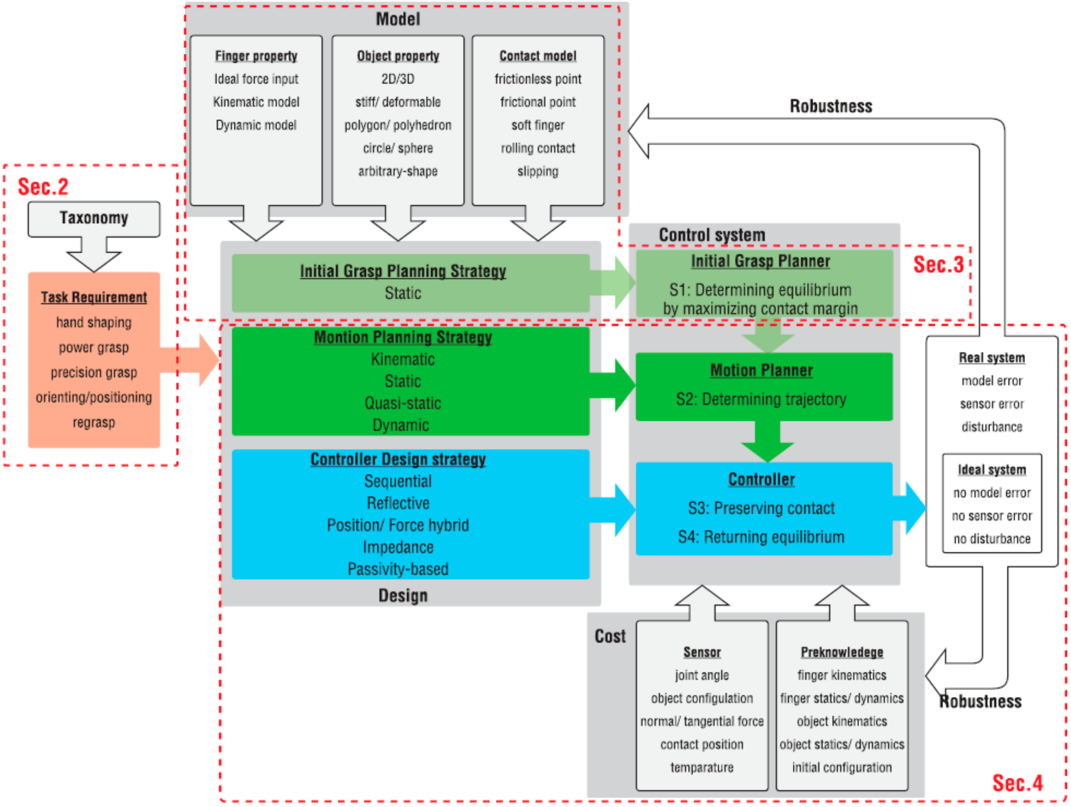
The following section will give an overview of the process to plan and control manipulator motion, the advantages and disadvantages of different Iterative Learning Controller (ILC).

1.3 Planning process for robotic grasping

Motion planning for robotic grasping can be dissected into 4 main portions shown in Figure 1. The first portion find points that the robot can safely grasp the object without causing it to fall. Second section plans out a route that ensures the robot can grasp an object and transport it safely. The third and forth portion builds up the controller of the grasping where it tries to correct any disturbances caused by external environment. [2]

To ensure the object is firmly grasped, it is crucial that the contact force is always above the minimum friction force of lifting and transporting the object. There are a few strategies that is implemented to ensure the condition is always fulfil. One of the methods is to use a feedback loop such as PD controller as demonstrated by [4]. Other methods are to model grasping and object as a spring damper system

[5], using impedance control by setting the object equilibrium point inside the object and dynamically changing the force by detecting object slippage [2].



Model based controller has also been tested to grasp and transport objects securely. However, this requires the dynamics to be known. A workaround is to introduce an adaptive grasp controller on a system where the dynamics of the object is unknown or estimated [6, 7]. With the feedback of velocity and force error, a learning controller can be implemented to hold the object rigidly and has been proven via calculation and simulation [8]. Iterative learning controller has been recently been implemented to a 7DOF robot arm which gradually decreases the contact force and duration of motion [9].

1.4 Overview of Iterative control

Robotic grasping is a repetitive motion. Therefore, it is beneficial to have a learning controller controlling the grasping motion even if the object to be grasped is slightly different, either in dimension or weights where it is able to take in previous iteration error and corrects them. There are different types of learning controller namely, Iterative learning control (ILC), adaptive control, neural network, and repetitive control (RC). Both adaptive controller and neural network controller modifies the controller while ILC and RC modifies the input signal.

The distinctive difference between RC and ILC is that RC does not reset its initial condition every iteration whereas ILC resets every iteration. This allows ILC not to bring over any error into the current iteration but merely changes the input signal based on the error. Since ILC consider the error of previous iteration, it is able to reduce the tracking error due to modelling uncertainly and repeating external disturbances.

The goal of ILC is to ensures the error reduces to as close to zero as possible while using the least amount of trails. It has to be robust in such that it can still converge while having modelling errors and repeated external environment disturbances. ILC are typically separated into 2 categories. Non-model based and model based, and they are discussed briefly in the next section.

1.5 Non-model based ILC

1.5.1 D type

The first ILC was proposed by Arimoto et al. [10] where the error from previous iteration is integrated into the current iteration to improve the performance as long as the previous iteration gives desired output response. The equation governing is defined as follows:

$$e_k(t) = y_d(t) - y_k(t) \tag{1}$$

$$u_{k+1}(t) = u_k(t) + \beta \dot{e}_k(t) \tag{2}$$

where k is the number of iterations, $y_d(t)$ is the desired trajectory, $y_k(t)$ is the trajectory output of the plan for the current iteration, $u_{k+1}(t)$ is the input of the next iteration, $u_k(t)$ is the input of current iteration, β is a constant mxm gain matrix and $\dot{e}_k(t)$ is the derivative of current iteration error.

The model will converge to zero error as long as the convergence criteria is satisfied when the number of iterations tends to infinity.

1.5.2 P type

P type ILC is very similar to D type. Instead of the derivative of the error used to calculate the next iteration input, the error is used instead, yielding the following equation following Arimoto et al. [11]:

$$u_{k+1}(t) = u_k(t) + \beta e_k(t) \tag{3}$$

This controller however may lack the ability to detect and correct small noise signal as there is no derivative term which is known to significantly amplified small signals. [12]

1.5.3 PD type

Arimoto et al. also proposed a mix type ILC where the P and D type works together similar to a PD feedback controller. This iteration is popular among the use for non-linear system as it does not require a plant to operate [13]. Although it can achieve convergence more readily than using P or D alone [12], it is not a guarantee that such success is always achieved unless the number of iterations is low [13].

1.6 Model based ILC

1.6.1 Inverse plant ILC

When the plant of the system is known, one of the simplest methods of ILC is just to invert the plant and use that to calculate the desired optimal input explained in [14] in the form of:

$$u_d(t) = f_p^{-1}(y_d(t), t) \tag{4}$$

where $u_d(t)$ is the desired input signal, f_p^{-1} is the inverse of the plant and $y_d(t)$ is the desired output.

The next input iteration and error is formulated as eqn 5 and 6.

$$u_{k+1}(t) = u_k(t) + f_p^{-1}(t)e_k(t) \tag{5}$$

$$e_{k+1} = r - Gu_{k+1} - d = (I - Gf_p^{-1})e_k \tag{6}$$

Since this take into account the inverse of the plant, it is capable of converge in just one iteration. However, the performance is greatly hinder with any modelling error and random disturbances. Hence, it is generally not suitable to be used in real system where modelling error is unavoidable.

1.6.2 NORM-Optimal ILC

Norm-optimal ILC is consider the 'best' ILC controller. The algorithm developed by [15] aims to solve the optimisation problem in eqn 7 by ensuring reduction of norm of error is achieved in each iteration, the step size for each iteration is automatically chosen and improve robustness by utilising the feedback from current iteration and feedforward data from previous iterations.

$$u_\infty = \arg min_u\{\left||e|\right|^2 \colon\ e = r - y, y = Gu\} \tag{7}$$

The algorithm proposed that on completion of k iteration, the input of the next iteration is simply the solution of eqn 8 optimisation problem:

$$u_{k+1} = \arg min_{u_{k+1}}\{J_{k+1}(u_{k+1})\} \tag{8}$$

where the optimality criterion is given in eqn 9. Eqn 9 differs slightly from [15] as 2 scalar matrix Q and R are added to aid the tuning between reducing tracking error and limit the change of input.

$$J_{k+1}(u_{k+1}) = Q||e_{k+1}||^2 + R||u_{k+1} - u_k||^2 \tag{9}$$

After taking the derivative of eqn 9 with respect to u_{k+1} would yield the following equation:

$$u_{k+1} = u_k + (R + QG^TG)^{-1}QG^Te_k \tag{10}$$

Details of the steps are omitted here, but the steps can be found in [15].

2. Detailed Specification

Different model and non-model ILC control algorithm is discussed in section 2 along with their limitation. It was discussed that model based ILC offers a higher likely hood of convergence as long as the plan model is known. On the other hand, the need of better robotic grasping algorithm was discussed as current generation of robotic grasping is significantly slower than an average human.

Therefore, a novel approach is the developed a control algorithm that utilise the fast convergence and robustness of model based ILC particularly using Norm-Optimal ILC in the hope of increasing the mean pick per hour rate.

2.1 Design Requirements [DR]

DR1. Model will be able to be **simulated in simulation.** This ensures that the ILC algorithm is tested prior to implementing onto the actual physical model.

DR2. Grasper will be able to **use minimal grasping force.** This ensures that the grasper does not use excessive force that could damage the grasped object.

DR3. Grasper will be able to **adapt to different object dimensions.** This allows the grasper the ability to grasp different types of objects whether different in sizes of weights in a safe manner.

DR4. Grasper will be able to **calculate its own trajectory.** This allows the grasper to adjust its reference target in response to misaligned objects and objects of different sizes.

DR5. Grasper will be able to **learn from previous iteration.** This ensures that any previous mishap will be learnt by the algorithm and ensure the grasper will adjust itself to other disturbances such as object misalignment, external repeating force.

DR6. Grasper will be able to **grasp fast and accurately.**

3. Basic Gantry Model & Simulation

4. Future Plans

5. References

[1] R. Hodson. (2018, 10 May) A Gripping Problem. *NATURE* 3. Available: https://media.nature.com/original/magazine-assets/d41586-018-05093-1/d41586-018-05093-1.pdf

[2] R. Ozawa and K. Tahara, "Grasp and dexterous manipulation of multi-fingered robotic hands: a review from a control view point," *Advanced Robotics,* vol. 31, no. 19-20, pp. 1030-1050, 2017/10/18 2017, doi: 10.1080/01691864.2017.1365011.

[3] J. Mahler *et al.*, "Learning ambidextrous robot grasping policies," *Science Robotics,* vol. 4, no. 26, p. eaau4984, 2019, doi: doi:10.1126/scirobotics.aau4984.

[4] A. B. A. Cole, J. E. Hauser and S. S. Sastry, "Kinematics and control of multifingered hands with rolling contact," in IEEE Transactions on Automatic Control, vol. 34, no. 4, pp. 398-404, April 1989, doi: 10.1109/9.28014.

[5] T. Wimboeck, C. Ott and G. Hirzinger, "Passivity-based Object-Level Impedance Control for a Multifingered Hand," 2006 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2006, pp. 4621-4627, doi: 10.1109/IROS.2006.282170.

[6] P. R. Pagilla and M. Tomizuka, "Adaptive control of two robot arms carrying an unknown object," Proceedings of 1995 IEEE International Conference on Robotics and Automation, 1995, pp. 597-602 vol.1, doi: 10.1109/ROBOT.1995.525349.

[7] H. Kawasaki, T. Shimizu, and S. Ito, "Adaptive Coordinated Control of Multiple Robot Arms," IFAC Proceedings Volumes, vol. 33, no. 27, pp. 261-266, 2000/09/01/ 2000, doi: https://doi.org/10.1016/S1474-6670(17)37939-9.

[8] T. Nakayama, S. Arimoto and T. Naniwa, "Coordinated learning control for multiple manipulators holding an object rigidly," Proceedings of 1995 IEEE International Conference on Robotics and Automation, 1995, pp. 1529-1534 vol.2, doi: 10.1109/ROBOT.1995.525492.

[9] J. Bös, A. Wahrburg and K. D. Listmann, "Iteratively Learned and Temporally Scaled Force Control with application to robotic assembly in unstructured environments," 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 3000-3007, doi: 10.1109/ICRA.2017.7989344.

[10] S. Arimoto, S. Kawamura and F. Miyazaki, "Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems," The 23rd IEEE Conference on Decision and Control, 1984, pp. 1064-1069, doi: 10.1109/CDC.1984.272176.

[11] S. Arimoto, S. Kawamura, F. Miyazaki and S. Tamaki, "Learning control theory for dynamical systems," 1985 24th IEEE Conference on Decision and Control, 1985, pp. 1375-1380, doi: 10.1109/CDC.1985.268737.

[12] Ratcliffe, J.D. (2005) An Iterative Learning Control Implemented on a Multi-Axis System. PhD Thesis, School of Electronics and Computer Science, University of Southampton, Southampton.

[13] D. A. Bristow, M. Tharayil and A. G. Alleyne, "A survey of iterative learning control," in IEEE Control Systems Magazine, vol. 26, no. 3, pp. 96-114, June 2006, doi: 10.1109/MCS.2006.1636313.

[14] Jian-Xin Xu and Qian Ji, "New ILC algorithms with improved convergence for a class of non-affine functions," Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No.98CH36171), 1998, pp. 660-665 vol.1, doi: 10.1109/CDC.1998.760759.

[15] N. Amann, D. H. Owens and E. Rogers, "Iterative learning control for discrete time systems using optimal feedback and feedforward actions," Proceedings of 1995 34th IEEE Conference on Decision and Control, 1995, pp. 1696-1701 vol.2, doi: 10.1109/CDC.1995.480384.

6. Appendix