MinistryA of Science and Higher Education of the Russian Federation

ITMO University

GRADUATION THESIS

STUDY ON DESIGNING COMPUTER VISION SYSTEM FOR MOVING OBJECTS TRACKING

Author_____Ali Shakkouf__________________ ________________
(full name) (signature)

Subject area __________________15.04.06 _____________________
(code, name of program track)
__________________ Intelligent technologies in robotics __________

Degree level __________________Master________________________
(Bachelor, Master, Engineer)*

Thesis supervisor___ Gromov Vladislav Sergeevich ______ ______
(surname, initials, academic title, degree) (signature)

St. Petersburg, 20___20___

Student_______Ali Shakkouf____

(full name)

Group___R42332_____ Faculty/Institute/Cluster __Control Systems and Robotics ________

Subject area, program/major ________ Intelligent technologies in robotics __________________

____

Consultant(s):

a) ____ _____________

(surname, initials, academic title, degree) (signature)

b) ____ _____________

(surname, initials, academic title, degree) (signature)

Thesis received "____"________________________20 ____

Originality of thesis: ______________%

Thesis completed with the grade: _______________________________

Date of defense "____"________________________20 ____

Secretary of State Exam Commission_________________________ __________________

(full name) (signature)

Number of pages ___________________________________

Number of supplementary materials/Blueprints _________________________________

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

“САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ”

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

ИССЛЕДОВАНИЕ СИСТЕМЫ ТЕХНИЧЕСКОГО ЗРЕНИЯ ДЛЯ СЛЕЖЕНИЯ ЗА ПОДВИЖНЫМИ ОБЪЕКТАМИ/ STUDY ON DESIGNING COMPUTER VISION SYSTEM FOR MOVING OBJECTS TRACKING

Автор *Шаккуф Али* ____________
(Фамилия, Имя, Отчество) (Подпись)

Направление подготовки (специальность) *15.04.06 «Интеллектуальные технологии в робототехнике»*

Квалификация *магистр*
(бакалавр, инженер, магистр)

Руководитель *Громов Владислав Сергеевич* ____________
(Фамилия, И., О., ученое звание, степень) (Подпись)

К защите допустить

Руководитель ОП *Бахолдин А.В., доцент, к.т.н.* ____________
(Фамилия, И., О., ученое звание, степень) (Подпись)

«____ »___________________ 20 _____ г.

Санкт-Петербург, 20 _20_ г.

Студент *Шаккуф Али* **Группа** *R*42332 **Факультет** *Систем Управления и Робототехники*

(ФИО)

Направленность (профиль), специализация *«Интеллектуальные технологии в робототехнике»*

Консультант (ы):

а) ______________________________ ______________

(Фамилия, И., О., ученое звание, степень) (Подпись)

б) ______________________________ ______________

(Фамилия, И., О., ученое звание, степень) (Подпись)

ВКР принята “ ____ “ ____________ 20 ____ г.

Оригинальность ВКР ____________ %

ВКР выполнена с оценкой ________________

Дата защиты “ ____ “ ____________ 20 __ г.

Секретарь ГЭК ____________ ____________

ФИО Подпись

Листов хранения ________________

Демонстрационных материалов / Чертежей хранения ____________

Ministry of Science and Higher Education of the Russian Federation

ITMO University

APPROVED

Head of educational program

____________________ ____________

(Surname, initials) (signature)

«____» «________________» 20____

OBJECTIVES

FOR A GRADUATION THESIS

Student______Ali Shakkouf ____

(full name)

Group___R42332______ **Faculty/Institute/Cluster** __Control systems and robotics ___________

Degree level _________________________________Master _________________________________

Subject area ________15.04.06 Mechatronics and Robotics ______________________

Major _____________________ Mechatronics and Robotics ________________________________

Specialization _________________ Intelligent robotics ____________________________________

Thesis topic __________ **STUDY ON DESIGNING COMPUTER VISION SYSTEM FOR MOVING OBJECTS TRACKING** ____

Thesis supervisor________ Gromov Vladislav Sergeevich ________________________________

________________ candidate of technical Sciences ____

2 Deadline for submission of complete thesis «______» «___________________» 20_____

3 Requirements and premise for the thesis

This research is dedicated to answering the following question: Can we estimate the future trajectory of an object if we observed that object for some interval of time?

The approach involves on oberving the object for some time 15-20 seconds, then we analysis the movement pattern to find what frequencies it contains. DREM is used as a method to estimate the frequencies. Once we estimated the frequencies, we can generate the future trajectory of the object. To test the accuracy of our algorithm, we use an arm robot that try to grip the object, in the time after the observation finished. ____

4 Content of the thesis (list of key issues)

introduction.

1. Overview of existing technical solution.
2. System Description
3. Implementing system software with MATLAB, Calibrator and Simulink
4. KUKA youBot arm manipulator
5. Dynamic Regressor Extension and Mixing (DREM)
6. Real Experiments in Laboratory

Conclusion

5 List of graphic materials (with a list of required material)

Presentation PowerPoint

6 Source materials and publications

[1] S. Aranovskiy, A. Bobtsov, R. Ortega and A. Pyrkin, "Performance Enhancement of Parameter Estimators via Dynamic Regressor Extension and Mixing*," in IEEE Transactions on Automatic Control, vol. 62, no. 7, pp. 3546-3550, July 2017, doi: 10.1109/TAC.2016.2614889.

[2] Doganis, Fivos (Yerres, FR). Dynamical camera calibration. 20200005491A1, Dassault Systemes (Velizy Villacoublay, FR), January 2020. http://www.freepatentsonline.com/y2020/0005491.html.

[3] Bobtsov, Alexey & Pyrkin, Anton. (2012). Cancelation of unknown multiharmonic disturbance for nonlinear plant with input delay. International Journal of Adaptive Control and Signal Processing. 26. 10.1002/acs.1283.

7 Objectives issued on «_24_» «_____May______» 2020___

Thesis supervisor __Gromov Vladislav Sergeevich _____

(signature)

Objectives assumed by ____________________ «____» «______________» 20____

(signature)

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО"

УТВЕРЖДАЮ

Руководитель ОП

_________________________ ______________
(Фамилия, И.О.) (подпись)

«____» «_______________» 20____ г.

З А Д А Н И Е
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ

Обучающийся_________Шаккуф Али ____
(ФИО полностью)

Группа__R42332 ______**Факультет/институт/кластер**_систем управления и робототехники

Квалификация ___ магистр ____
(магистр, инженер, бакалавр)**

Направление подготовки _15.04.06 Мехатроника и робототехника ______________________
(код, название направления подготовки)

Направленность (профиль) образовательной программы
_________________________Мехатроника и робототехника______________________________

Специализация ____Интеллектуальная робототехника______________________________

Тема ВКР_______ ИССЛЕДОВАНИЕ СИСТЕМЫ ТЕХНИЧЕСКОГО ЗРЕНИЯ ДЛЯ СЛЕЖЕНИЯ ЗА ПОДВИЖНЫМИ ОБЪЕКТАМИ__________________________________

Руководитель___Громов Владислав Сергеевич ____
(ФИО полностью, место работы, должность, ученая степень, ученое звание)
_____ кандидат технических наук ____

2 Срок сдачи студентом законченной работы до «______» «________________» 20_____г.

3 Техническое задание и исходные данные к работе

Это исследование посвящено ответу на следующий вопрос: можем ли мы оценить будущую траекторию объекта, если мы наблюдали этот объект в течение некоторого интервала времени?

Этот подход включает в себя оберегание объекта в течение некоторого времени 15-20 секунд, затем мы анализируем паттерн движения, чтобы найти, какие частоты он содержит. Дрем используется в качестве метода оценки частот. После того, как мы оценили частоты, мы можем сгенерировать будущую траекторию движения объекта. Чтобы проверить точность нашего алгоритма, мы используем руку робота, который пытается захватить объект, в то время как наблюдение закончено.

4 Содержание выпускной квалификационной работы (перечень подлежащих разработке вопросов)

вступление.

1. Обзор существующего технического решения.
2. описание системы
3. Внедрение системного программного обеспечения с использованием MATLAB, калибратора и Simulink
4. Компания KUKA youBot манипулятора
5. Динамическое расширение регрессора и смешивание (DREM)
6. Реальные эксперименты в лаборатории

Вывод

5 Перечень графического материала (с указанием обязательного материала)

Презентация PowerPoint

6 Исходные материалы и пособия

[1] S. Aranovskiy, A. Bobtsov, R. Ortega and A. Pyrkin, "Performance Enhancement of Parameter Estimators via Dynamic Regressor Extension and Mixing*," in IEEE Transactions on Automatic Control, vol. 62, no. 7, pp. 3546-3550, July 2017, doi: 10.1109/TAC.2016.2614889.

[2] Doganis, Fivos (Yerres, FR). Dynamical camera calibration. 20200005491A1, Dassault Systemes (Velizy Villacoublay, FR), January 2020. http://www.freepatentsonline.com/y2020/0005491.html.

[3] Bobtsov, Alexey & Pyrkin, Anton. (2012). Cancelation of unknown multiharmonic disturbance for nonlinear plant with input delay. International Journal of Adaptive Control and Signal Processing. 26. 10.1002/acs.1283.

7 Дата выдачи задания «_24_» «_____мая_______» 2020__г.

Руководитель ВКР__ Громов Владислав Сергеевич _______
(подпись)

Задание принял к исполнению___________________ «____» «______________» 20____г.

Ministry of Science and Higher Education of the Russian Federation

ITMO University

SUMMARY
OF A GRADUATION THESIS

Student_____Ali Shakkouf ____
(full name)

Title of the thesis__ STUDY ON DESIGNING COMPUTER VISION SYSTEM FOR MOVING OBJECTS TRACKING____

Name of organization _________ITMO university__________ ___________________________

DESCRIPTION OF THE GRADUATION THESIS

1 Research objective____________to study and design a computer vision system to track moving objects with periodic moving patterns.____

2 Research tasks_object detection, camera calibration, DREM, forward and inverse kinematic_____

3 Number of sources listed in the review section _________10________________

4 Total number of sources used in the thesis ___________28____________________

5 Sources by years:

Russian			Foreign		
In the last 5 years	5 to 10 years	More than 10 years	In the last 5 years	5 to 10 years	More than 10 years
4	1		8	8	8

6 Use of online (internet) resources ______________Yes, 2______________________________
(Yes/No, number of items in the list of references)

____

7 Use of modern computer software suites and technologies (List which ones were used and for which section of the thesis)

Software suites and technologies	Thesis section
MATLAB	3,5,6

8 Short summary of results/conclusions ____________________________________

The implemented system is useful for tracking objects in the conditions of a room, laboratory or big hall especially for those objects that move with a predefined trajectory and don't change their trajectory depending on the environment. The result shows that the tracking error is relatively small.

____

____

____

9 Grants received while working on the thesis ____
(Grant name)

____

10 Have you produced any publications or conference reports on the topic of the thesis? _Yes___
(Yes, no)

a) Congress of young scientists, "COMPUTER VISION SYSTEM TO TRACK a moving OBJECT"
b) Scientific and educational-methodical conference of ITMO University, "Pinhole camera calibration"

(Bibliographical description of a conference report)

Student______Ali Shakkouf____________ _________________
(Full name) (signature)

Thesis supervisor_ Gromov Vladislav Sergeevich ____ _________________
(Full name) (signature)

“__________”________________20___

Министерство науки и высшего образования Российской Федерации

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

"НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО"

АННОТАЦИЯ

ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

Обучающийся__________Шаккуф Али____
(ФИО)

Наименование темы ВКР: ___ ИССЛЕДОВАНИЕ СИСТЕМЫ ТЕХНИЧЕСКОГО ЗРЕНИЯ ДЛЯ СЛЕЖЕНИЯ ЗА ПОДВИЖНЫМИ ОБЪЕКТАМИ________________________________

Наименование организации, где выполнена ВКР_____ Университет ИТМО ________

ХАРАКТЕРИСТИКА ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЫ

1 Цель исследования_____ изучить и спроектировать систему компьютерного зрения для отслеживания движущихся объектов с периодическими движущимися паттернами.

2 Задачи, решаемые в ВКР __обнаружение объектов, калибровка камеры, дрем, прямая и обратная кинематика ___

3 Число источников, использованных при составлении обзора_____10__________________

4 Полное число источников, использованных в работе ___________28__________________

5 В том числе источников по годам

Отечественных			Иностранных		
Последние 5 лет	От 5 до 10 лет	Более 10 лет	Последние 5 лет	От 5 до 10 лет	Более 10 лет
4	1		8	8	8

6 Использование информационных ресурсов Internet________Да,2_______________________
(Да, нет, число ссылок в списке литературы)

7 Использование современных пакетов компьютерных программ и технологий (Указать, какие именно, и в каком разделе работы)

Пакеты компьютерных программ и технологий	Раздел работы
MATLAB	3,5,6

8 Краткая характеристика полученных результатов ________________________________

Реализованная система полезна для слежения за объектами в условиях помещения, лаборатории или большого зала, особенно за теми объектами, которые движутся по заданной траектории и не меняют своей траектории в зависимости от окружающей среды. Результат показывает, что ошибка отслеживания относительно невелика.

9 Полученные гранты, при выполнении работы ____
(Название гранта)
____

10 Наличие публикаций и выступлений на конференциях по теме выпускной работы__Да___
(Да, нет)

a) Congress of young scientists, "COMPUTER VISION SYSTEM TO TRACK a moving OBJECT"
b) Scientific and educational-methodical conference of ITMO University, "Pinhole camera calibration"

Обучающийся____Шаккуф Али________________ _________________
(ФИО) (подпись)

Руководитель ВКР__ Громов Владислав Сергеевич____ _________________
(ФИО) (подпись)

“__________”________________20___г.

Contents

Introduction

Technology occupies a major role in various areas of our modern life. This role is fundamental and can never be dispensed with. Technology is now found in the various types of devices that we use in everyday life, such as the mobile phone, smart watch, and many other household items. Also, don't forget the military fields. Without technology, radars could not be built, directing missiles and aircraft with precise accuracy, tracking targets ... where here there must be an element of high-speed response that a person cannot provide. And many, many other applications in the medical fields, highly efficient medical diagnosis and various surgical procedures. In automated manufacturing applications where we now see automated mega factories operating using technology and requiring very little human intervention.

What is amazing about technology is that it never stops updating, developing and improving, especially in the past 30 years there is a terrible technological acceleration. That is, we cannot say so-and-so invented an ideal device that could not be better. On the other hand, this means that in every technology seen today there are certain defects or incomplete aspects that sometimes make them vulnerable to penetration and thus a violation of user privacy. Today, we see a terrible competition between technology giants in all fields to find a better product than all the products of other companies. This process of development drains large amounts of money and long working hours for many researchers, developers and scientists. But if the development process succeeds and a new product is found, this returns with huge profits and money for the developing company or the developing team, which is multiple times more than what was spent on the development process.

If we wanted to materialize the word technology with concrete things in real life, we would start with an endless list of products. The mobile devices, surveillance cameras, and nano sensors in our personal devices are technological products. There are also the arms of automated robots, military robots, loading and unloading robots, modern smart missiles, sensors and programs of modern transportation means, manufacturing and canning lines of all kinds, drones and unmanned airplane, 3D printers, CNC machining, shearing and cutting machines, Medical diagnostic devices, interactive auxiliary home robots .... .

No one can imagine our daily life without these technological products. They have an effective, fast-acting and intelligent role in many activities, and they really deserve that importance for many reasons, the main of which are:

- **Economic growth:** Significant improvements in agricultural productivity have altered the way people function in Europe, free farmers to do other activities and allow them to move to the city and establish industrial career. The shift from handcrafted to automated products increased productivity, directly impacting living standards and growth.
- **Increased luxury:** in general, innovation and economic growth increase welfare because living standards rise. According to the Brookings Institution, life expectancy is higher in countries with per capita GDP. Other research also shows that there is a link between innovation and self-luxury.
- **Reducing disease, poverty and hunger:** What comes to reduce hunger, for example, agricultural productivity is crucial in developing countries where the next population boom is likely to occur. Smallholder farms in developing counties play an important role as up to 80% of food is produced in these communities.
- **Increased productivity:** Economic growth is driven by innovation and technological improvements, which reduce production costs and enable production to increase. If we look at this from an enterprise perspective, different automation solutions reduce manual and repetitive work and release time for more important tasks and value creation.
- **Speed and accuracy of implementation:** Robotic products enabled us to manufacture a large number of products with precision, professionalism, aesthetics, and very great speed if compared with the same production by humans. A good example of this is laser cutting and lathe fabrication, where we get super-accurate

products very quickly. Sometimes some products cannot be reached without these technological technologies.

It is very important to know that the manufacture of any modern product requires the intervention of multiple sciences that are consistent and interact with each other to reach the desired goal. And in our time, we do not mean only those sciences like physics, chemistry and programming ... Science has become more specialized and branched. There is traditional control science, adaptive control science, mathematical modeling, mechanical design, image processing science, computer vision science, objects recognition and tracking science, virtual reality, augmented reality, machine education, neuronal networks and many other sciences.

For example, we rarely can talk about objects recognition and tracking without incorporating the science of image processing and computer vision. As it often requires the existence of optical sensors (often cameras) that record sequential snapshots or video stream of the subject. Then comes the role of image processing and the computer vision that takes upon itself the task of identifying the objects in many different ways related to each of the environment in which the target moves, the object geometrical and color information, and the nature of movement. Finally comes the role of objects tracking science that uses the camera information, and the information provided by the science of image processing as input to the tracking algorithm.

This research represents a combination of three applied sciences: image processing, computer vision and robotic arm control. The most important part of this research is to present a new study and application of the frequency estimation algorithm in objects tracking and predict their future trajectory for the next 10-15 seconds.

Also, this research serves as a tester of the accuracy of tracking systems in small environments like room, or big hall. In other words, if someone developed an algorithm for objects tracking, then the software product of this research serves as tester of the accuracy of that algorithm.

What is this research based on?

According to the important theorem formulated by the French mathematician Jean Baptiste Joseph Baron Fourier, *any periodic function, no matter how trivial or complex, can be expressed in terms of converging series of combinations of sines and/or cosines*, known as Fourier series. Therefore, any periodic signal is a sum of discrete sinusoidal components.

This research is dedicated to answering the following question:

Can we estimate the future trajectory of an object if we observed that object for some interval of time?

So, once we observed the object (took enough information about his trajectory for some interval of time) we analysis the movement pattern to find what frequencies it contains. The method to estimate the frequencies is explained later. We can’t use Fast Fourier Transform (FFT) to find the frequencies, this will be clarified later too. Once we estimated the frequencies, we can generate the future trajectory of the object. To test the accuracy of our algorithm, we use an arm robot that try to grip the object, in the time after the observation finished.

The use of such a system in essentially in tracking objects, in the time when we lose the connection with the object or we can’t detect it. For example, in Figure 0.1, if we are tracking cars and label them at a continuous matter, then because of some obstacles (like trees), two or more objects get lost. We can by generating the future trajectory of each car predict their future positions, so that we would not lose the first label we assigned.

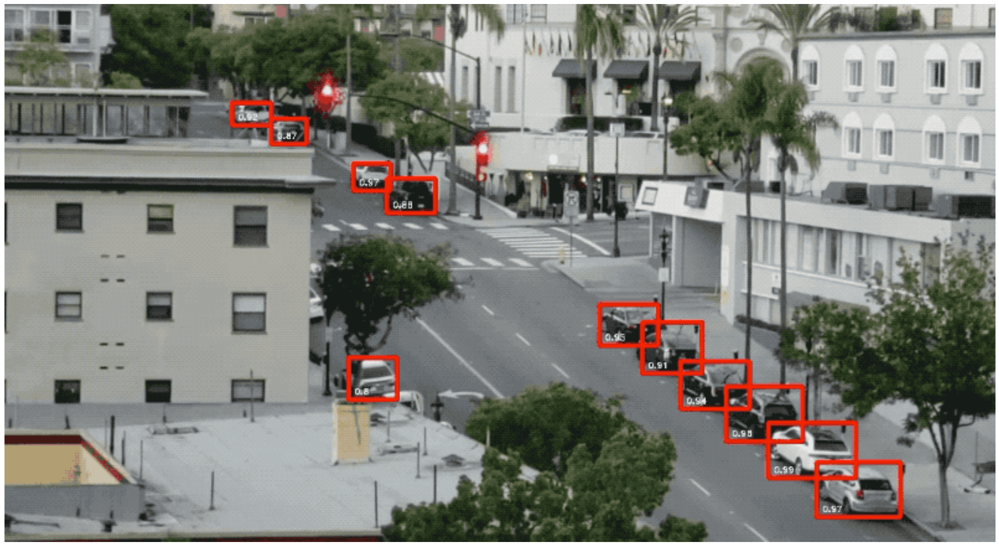

Figure 0.1 – example of cars tracking with existing obstacles

The same principle is applied when tracking missiles or drones. When we get in a situation of radar jamming, we can use the observed trajectory of that missile/aircraft/drone to predict where it is going to be in the next few seconds (until we can track it again with the radar). This is not the case for all types of missiles or drones. There are missiles that moves in totally arbitrary pattern and could not be predicted. On the other hand, there are plenty of missiles that travel huge distances with predefined trajectory.

It is to be noticed that this process is repeated as much as we need to track the object. This process could be represented as cycle (see Figure 0.2).

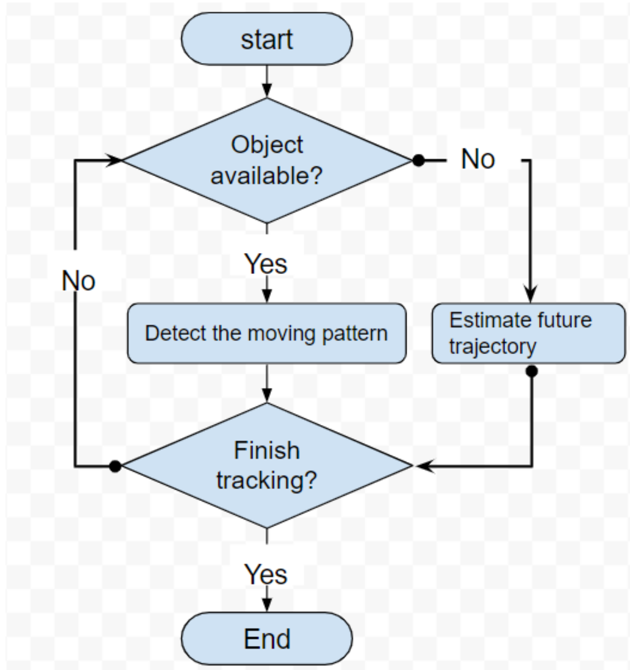

Figure 0.2 – tracking cycle

This research is the starting point of such systems. There are a lot of work in this field. And the required software for such systems is quite heavy.

1 Overview of existing technical solutions

1.1 Object estimation/detection and tracking

A system to detect and track moving objects was represented in Erokhin 2017[1]. Such a system could be implemented onboard and used for unmanned aircrafts. Here a video stream is processed, and we have no information about objects we are tracking. in other words, we don't know the nature of that objects. This work is divided into multiple steps:

- To find the Euclidian transformation between the consecutive frames. An algorithm based on phase correlation used to estimate rotation and shift between consecutive frames. This helps us in background estimation process. The later process is used in moving objects detection process.
- The fact that we don't have any information about the object leads to the need of both background and object estimation. A mathematical model is developed for image estimation. It assumes that for one object, we have a binary representation of it called L, then an observed frame equal to the summation of three elements:
  - Convolution of object image with L.
  - Convolution of background image with complement of L.
  - Normal Gaussian noise.

  Due to some errors of estimation, the author came with the idea of representing the object using two or more segments. It means that one object is detected as two or more objects. This is later processed in tracking algorithm.

- Objects tracking: this task was done using trajectory graph. This graph represents the correspondence between the list of tracked objects 'TO' and the list of segments detected 'S' in the current frame. The correspondence is quantitative and represented as a Euclidian distance between TO and S.

To reduce the calculation cost of Euclidian transformation, the author used a window 128*128 or 256*256 inside the image, and tried to make a good match between them. The larger the window is the more accuracy the calculation is.

Sang 2013[2] work describes a vision system to detect moving objects on a conveyor belt, then use arm robot to pick the objects. The main two tasks solved here are objects recognition and tracking.

Detection task: here a 6-DOF arm robot is used. The vision sensor is attached to the end-effector. Since the camera is stationary, and the background is not changing, then the detection task could be solved by subtraction of two consecutive frames. This solution provides suitable calculation speed for real time detection, especially if the vision sensor provides gray images.

Tracking task: in general, the conveyor belt has a constant speed. But the author supposed that may one conveyor functions on multiple values of speed. For this reason, the speed is measured by dividing the passed distance on the time needed to pass it.

The system works in the following order:

1. Acquire images during fixed time period.
2. If an object was detected go to 3, else go back to 1.
3. Acquire another frame to detect the conveyor speed.
4. Calculate the future position of the object.
5. Stop images acquiring process, move the arm, pick and place the object.
6. Go back to 1.

The arm motion was optimized so that needed time to move arm from point A to B is minimum. This is done by optimizing the robot trajectory.

The discussed approach in Tsarouchi 2013[3] aspires to replace the manual ways for packaging, by fully automated lines equipped with vision sensors and a robot arm. The image-based vision system presented in this paper belongs to the category "Static in the workspace". The system is tested on shaver knobs that look perfectly identical. The system was built in the following sequence:

1. Calibrate the digital camera, so that we can make some real measurement on the shaver handle from 2D image.
2. Extract feature points on shaver handle. The shaver handle could lay on in three different ways; back up, laying on its side and back down. for each of the said laying ways there are different algorithm to pick the shaver handle.
3. Use information from camera calibration and detected features to calculate the real coordinates of target points on the shaver handle and to detect the orientation of shaver handle.
4. Send appropriate commands to the arm robot to pick and place the objects.

The accuracy and errors of object position detection was measured with the help of millimeters paper the is placed under the working area. It has been found that relative error in the axis X and Y respectively are 1.975% and 3.572%.

The error value is large enough to cause a failure in the picking process in many cases. For such a system with unchangeable z axis, it is recommended to repeat the calibration process, because it is the one that provides mapping between 3D world and 2D image.

Shen 2009[4] represents a novel location prediction model. The new model is based on grey theory. It uses this theory to predict the future location of objects move with uncertain pattern.

The main idea of prediction is the following:

1. Suppose we have dataset ‘S’ about the position of an object in 3D space.

2. Divide the data set 'S' into m mini data set such that each mini dataset satisfies the condition of correlativity sequence:

$$\left|x_i - x_j\right| + \left|y_i - y_j\right| < \eta$$

Where η is a value selected to be appropriate for the application. It represents to what degree the data is correlative. Those mini data sets called division sets SD_i. Pseudo-code to find SD_i is:

```
Algorithm Construct_Division
 Input: S - the given location sequence of a moving
           object at discrete sampling points.
 Output: SD = {S1, S2, …, Sm} - the division of S.
1.  Let i=1, SD = ϕ;
2.  Let Si = ϕ;
3.  While (S ≠ ϕ)
4.    Take out the first element e from S and let S =
    S − {e};
5.    if (Si == ϕ)
6.      Si = Si ∪ {e};
7.    else
8.      if (∀ γ ∈ Si (ξ(e, γ) < η ))
9.        Si = Si ∪ {e};
10.     else
11.       { i ++;
12.         Put Si into SD;
13.         Goto step 2; }
14. if (Si ≠ ϕ)
15.   Put Si into SD;
```

3. For each division set calculate two functions F_i, G_i. those functions are calculated using the Algorithm GM(1,1).
4. Find the future position of our object using the following relations:

$$x_t = x_{ip} + f_p(t)$$
$$y_t = y_{ip} + g_p(t)$$

The negative side of such an approach is it could predict the future position for maximum five seconds after the last position of the object was taken, later it has no information how this object will act.

Husain 2014[5] represents a new approach to track moving object in 3D space using depth sensor. problem statement is: given an object moving in 3D space in arm robot workspace, build a vision and control system to control the arm robot such that distance between object and End-effector will stay smaller than a threshold value h.

The system works according to the following sequence:

1. Initialization: user select an object to track by drawing a box around it.
2. Kinect camera provides us with depth map. This map is used to obtain 3D position of the object. A mathematical model is used for that, takes camera position with relative to world coordinate system as an input. Also the orientation of the object is detected here. Taking the first orientation in step1 as the unity orientation.
3. The Inverse kinematic solution is used to obtain the desired joints values.
4. Trajectory planning achieved using Dynamic Movement Primitives Algorithm DMP.
5. Desired trajectory is sent to the controller. The last generates the torque commands and sent the appropriate signals to the arm.
6. Calculate the distance between End-effector and tracked object using vision information and forward kinematic solution.
7. If the error:

$$e = \left\| \begin{bmatrix} e_p \\ e_0 \end{bmatrix} \right\|_2 + \delta$$

Is below a threshold value, then stop tracking else go back to step 2. Where $e_p, \mathrm{e}_0, \delta$ are the position error, orientation error and half the length of the bounding box respectively.

Alahi 2016[6] carried out a study on predicting the future trajectory of people based of their past trajectory.

LSTM-based model was introduced (Long Short Term Memory). This model can meet across multiple individuals to predict human paths in a scene. The researchera use one LSTM per track and share information between LSTMs by introducing a new social gathering layer. They refer to the resulting model as "Social" LSTM. The proposed method outperforms the most recent in two publicly available data sets. In addition, we show qualitatively that Social-LSTM successfully predicts different non-linear behaviors resulting from social interactions, such as a group of individuals moving together.

This approach takes in consideration that people may change their motion based on the behavior of other people around them. Person may stop for a sec, or wait to accommodate a group of people towards him.

The model was tested on multiple datasets to predict future trajectory in the next 5 seconds. Trajectory prediction error was compared with results of other methods used for the same purpose. The predicting error is one of the smallest between the used methods.

Wiest 2012[7] carried out a study on predicting future trajectory of cars for some seconds in advance. The goal of this research is to help driver assistance systems in avoiding accidents as much as possible.

The observed trajectory of the car is represented in 2D space as a sequence of X,Y coordinates with timestamps of the form:

$$T = ((x_0, y_0), t_0), \ldots, ((x_{N-1}, y_{N-1}), t_{N-1})$$

To obtain an uniform representation of all trajectories, the researcher uses Chebyshev decomposition on the three elements of the trajectory (X,Y and time). The coefficients of the obtained polynomial are used as input to the probabilistic model which is used to predict future trajectory.

Instead of learning the regression function, the observed motion patterns in the past are used to deduce the probability distribution as the motion model. Probability modeling allows the conversion of statistical parameters to the x-field of a vehicle's coordinate system.

For the evaluation of the proposed algorithms, a dataset of which consists of 69 real world trajectories is used, recorded at three different intersections with varying maneuvers (passing straight, left and right turns).

1.2 Frequency estimation

Borisov 2017[8] represents a useful use of the so-called dynamic regressor extension and mixing (DREM) to track a multi-sinusoidal signal. And the result was tested using web camera, arm robot and LCD screen.

The author considered LTI system of the form:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t-h), \\ y(t) &= Cx(t), \\ e(t) &= g(t) - y(t), \end{aligned}$$

The problem statement is to find a control law such that the error e goes to zero. Given that the reference signal $g(t)$ is a combination of biased sinusoidal signals with different frequencies.

In the carried-out experience, it had been shown that DREM has big advantage on the well-known gradient method for frequency estimation. DREM is faster and more stable. In MATLAB simulation it has been shown that both DREM and Gradient method are stable but DREM is faster. In real life, due to the fact that camera sensor doesn’t give us information without disturbance, DREM has proved to be less sensitive for disturbance.

1.3 Camera calibration

Doganis 2020[9] published an invention which represents a modification for Zhang method for digital camera calibration. The new approach uses a calibration pattern with the following procedure:

a. Display the calibration pattern on a video screen.
b. Acquire a video stream in which the said pattern exists.
c. Determine the required modification for calibration pattern and make the screen display it.
d. Repeat steps from a to c until acquiring good video stream for calibration.
e. Calibrate the camera.

Zhang method for calibrations include:

1. Print calibration pattern using laser printer.
2. Measure the printed pattern because it could be scaled by the printer.
3. Ensure the pattern to be straight by gluing it on a rigid flat surface.
4. Place the pattern in front of camera and move it so that we acquire images that comprise the said pattern in all sides of camera view (up, down, left and right).

This invention is superior to Zhang method by:

- Steps 1,2 and 3 in Zhang are tedious to be done. Because in order to maximize the calibration process, we should be accurate in creating the calibration pattern and measuring its size.
- Step 4 is time consuming. The reason for that is the number of rejected images in calibration process. Because of that rejection we may need to acquire images many times.

Invention fields of application are Augmented reality, where we need to place virtual objects in the real scene in the appropriate position. This include good estimation of distance between surfaces and camera. Also, whenever an accurate camera calibration is required, this invention could be used.

Zhang 2000[10] represents a new flexible method for camera calibration. It only requires the camera to observe a chessboard in different orientations (at least two). The main point is to solve the closed-form mathematical model proposed by the author.

This research shows an impressive improvement of performance over the two used -in the past- techniques for camera calibration which are: Three-Dimensional reference camera calibration and Self-calibration.

Three-Dimensional reference camera calibration requires a 3D object with precisely known geometric configuration, such an object in expensive and requires elaborate setup.

Self-calibration involves moving the camera in a static scene. Later some feature points are extracted from multiple images which contain the same objects. Those feature points are enough to estimate the internal and external parameters of the camera. However, using this method we cannot always obtain reliable results, because we need to estimate so many parameters.

Proposed calibration procedure is achieved according to the following steps:

1. Print a chessboard pattern using good quality printer and attach it to a planar surface.
2. Acquire multiple frames for the pattern with different orientations, either by moving the camera or the pattern in 3D apace.
3. Detect feature points of the pattern in the acquired images.
4. Estimate intrinsic and extrinsic parameters by solving the closed-form mathematical model.

2 System Description

Since it is quite difficult to have a real object in the laboratory that moves with moving patternЫ combined of multiple periodic signals like the ones shown in Figure 2.1, we decided to create that object in Simulink. The object will move on an LCD screen. The vision sensor is a web camera.

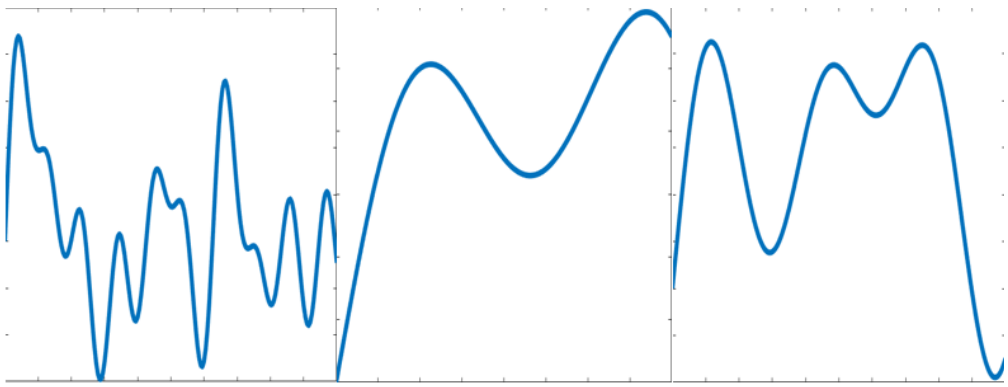

Figure 2.1 – examples of periodic trajectory patterns

2.1 Laboratory equipment

As mentioned earlier, the accuracy of the algorithm will be tested by forcing the end effector of a manipulator to be in as small as possible distance from the target as if it going to grip the object at each moment of the trajectory. The used manipulator is “KUKA youBot manipulator”.

The camera is attached to the last link of the manipulator as in Figure 2.2. During the observation process, the manipulator takes a configuration so that camera sees the whole screen (Figure 2.3). Thus, our system, as hardware, consists of three parts; camera, screen and manipulator. Below we provide a brief description for each one of them:

- **Camera:** it is “Logitech c170”. This is a web camera with maximum resolution of 640x480 pixels. The resolution could be set to smaller one. Since our target is moving on 2D screen, one camera is enough for this research. The distance from the screen could be obtained by camera calibration. However, in real life, this camera is not suitable for tracking systems. We will be in need for camera with special features and better specifications.

Figure 2.2 - Logitech c170 attached to youBot End-effector

- **KUKA youBot manipulator:** it is an educational manipulator from the Germany company KUKA. It is a small manipulator with maximum height 655mm. The KUKA youBot comes with fully open interfaces and allows the developers to access the system on nearly all levels of hardware control. It further comes with an application programming interface (KUKA youBot API), with interfaces and wrappers for recent robotic frameworks such as ROS or OROCOS, with an open source simulation in Gazebo and with some example code that demonstrates how to program the KUKA youBot. The platform and the available software shall enable the user to rapidly develop his/her own mobile manipulation applications.
- **LCD screen:** it is a totally flat screen from ViewSonic. This screen is located at some distance from the manipulator. The distance should not be large, because the manipulator should be able to reach each point from the screen. The distance between

youBot and screen in unknown. The orientation of the screen is also unknown. Those parameters will be estimated using camera calibration.

Figure 2.3 – System Hardware

2.2 How does the system function?

This section is dedicated to clarify step by step how the entire system functions. There are several steps to accomplish the tracking process. Each step is explained in detail. However, there are some steps that are explained briefly here; we just refer to the input and the output of those steps. The reason is those steps are explained in previous/coming separate sections. Figure 2.4 represents flowchart of the system.

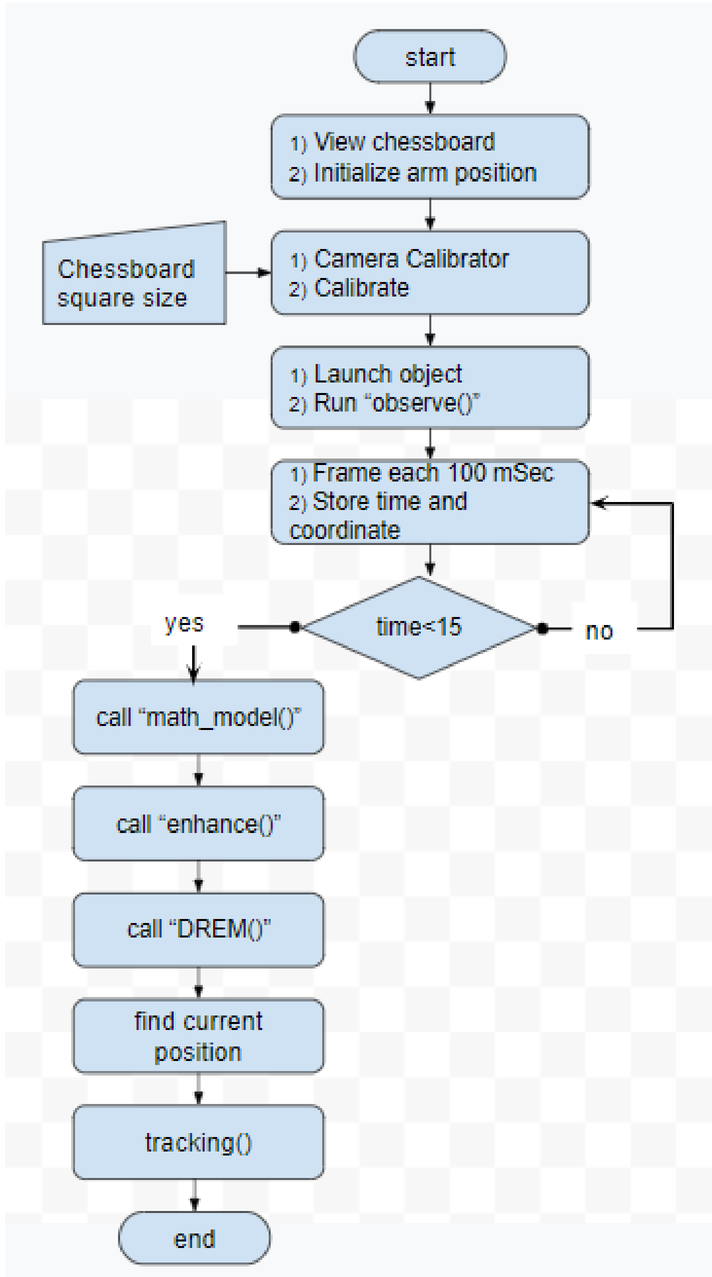

Figure 2.4 – system flowchart

First, we place the manipulator and the LCD screen so they face each other. Remember that the distance between them **should be in a range, for which the manipulator can access each point of the screen**. No need to measure that distance. Also, no need to know the orientation of the screen. After that we start performing the following steps:

A. Choose a chessboard pattern with even number of squares in column and odd number of squares in row. Display that pattern on the LCD screen. Then send commands to the manipulator, so that it takes a specific configuration for which the camera sees the whole screen.

B. Open MATLAB calibrator. Write a program that command youBot to take different configurations (at least two, to optimize and obtain good results take 15 frames) with time interval between those configurations. At each time interval, acquire a frame for the screen. We need at least two different frames for different orientations of the screen. Once one acquires enough frames, he needs to press calibrate. the calibrator asks to enter the real size of a chessboard square in millimeters. Then wait a little bit, save the calibration results in MATLAB workspace. The calibrator offers a representation of camera and screen locations with respect to each other's. Figure 2.5 is an example for that.

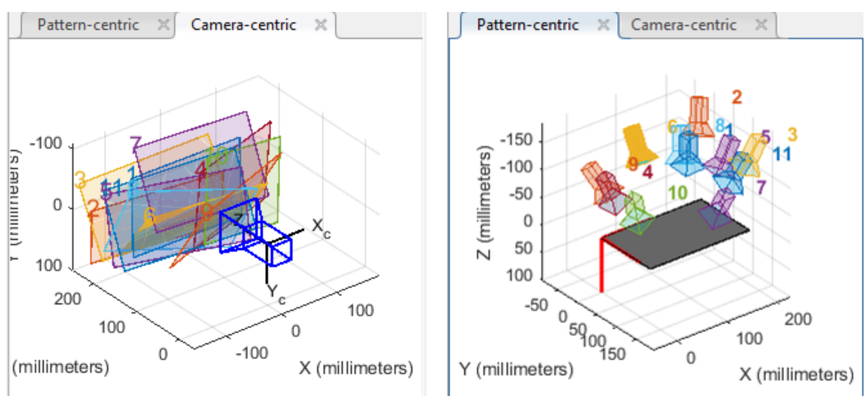

Figure 2.5 – camera and screen locations representation with respect to each other's

C. Open the Target model in Simulink (in later section we describe how to build that model). Choose the movement function of it. And press "run simulation". Go execute the function Main which send a command to run Target model. Also, this function calls many other functions. Each one performs a specific task. All the coming functions are called by Main. But here we continue to say "call function *FunctionName*" which should be interpreted as "the function Main calls the function *FunctionName*".

D. Call function *"observe()"*. This function detects the moving pattern of the Target. It takes frame each 100 mSecond. For each from the Target is detected. When Target is detected, his coordinates and frame time are stored as a pair <time, coordinates>. When the program fails to detect Target in some frame, the value NaN is stored as coordinates. This is useful to check how the data is good before estimating the frequencies. If the pattern contains big number of NaNs, a message will pop up says that data is bad to perform tracking. this process in repeated 15 seconds. After it the camera will be turned off.

E. Call the function *"math_model()"*. This function returns the homogeneous transformation between the youBot base and screen coordinate system (see Figure 2.6). It takes data from camera calibration (rotation matrices and translation vectors), data about youBot configurations during calibration, data about the mathematical transformation between camera and the End-effector. Each one of those data is HT. By multiplying those three HTs together we get one HT represents the transformation between youBot base and screen coordinate system.

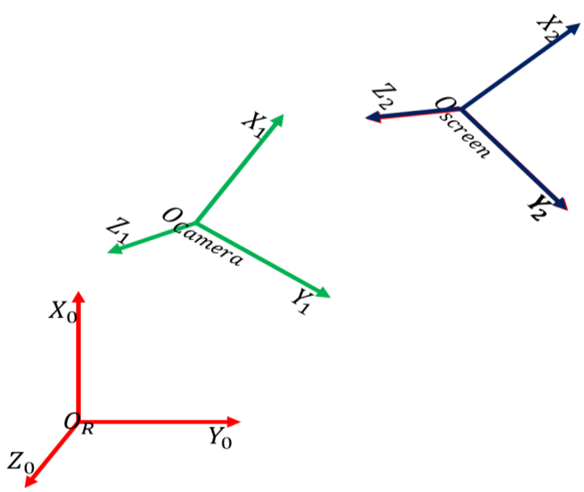

Figure 2.6 – system coordinate frames

F. Call the function *"enhance()"*. This function prepares the detected pattern to be a good input for DREM (Figure 2.7 is an example). Because DREM is sensitive to the amplitude of input signal. *"enhance()"* involves three operations:

   a. Performing interpolation to remove NAN values.
   b. Modifying the signal amplitude.

c. Performing another interpolation to add extra mid-points to the pattern.

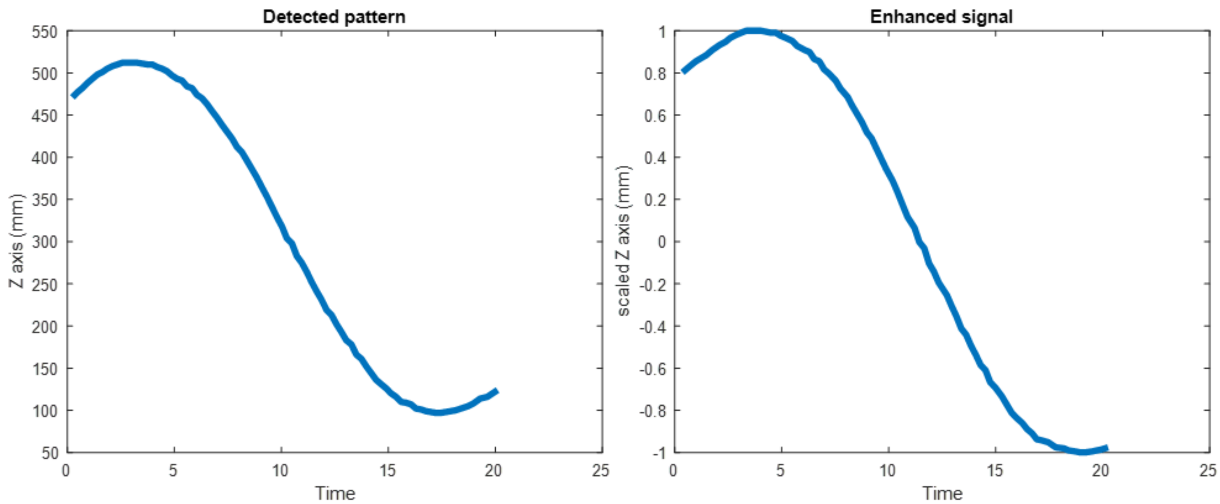

Figure 2.7 – an example clarifies *enhance()* functionality

G. Call the function *"DREM()"*. This function takes, as an input, the enhanced pattern, and return the estimated frequencies existed in that pattern. This process is performed according to DREM. A complete description of DREM is mentioned in a separate section later. Also, the mathematical prove of it is explained in [11][13].

H. Call the function *"find_current_position()"*. This function is the last step before beginning of tracking. we already estimated the frequencies. But to predict the future trajectory we need to know at which point of the signal we stopped observing the Target.

I. Call the function *"tracking()"*. This function calculates the Target coordinates on the screen with respect to youBot base coordinate system, then start following it by trying to keep the end-effector as near as possible from the object.

3 Implementing system software with MATLAB, Calibrator and Simulink

This chapter explains five items:

1. Preparing MATLAB environment to connect with image acquiring device (web camera Logitech c170).
2. Generating the Target in Simulink.
3. How to use “Vredit terminal” in MATLAB.
4. Pinhole camera model.
5. Using MATLAB camera calibrator to perform camera calibration.

Each item is described in details in the following subsections.

3.1 Preparing MATLAB Environment

Before starting work on programming the vision system, we must establish the connection between MATLAB and the image acquisition device (web camera). In MATLAB, if we want to check the existed acquisition devices, we use the following function:

Imaqhwinfo();

Which returns a structure that contains information about the image acquisition adaptors available on the system. An adaptor is the interface between MATLAB and the image acquisition devices connected to the system. The adaptor's main purpose is to pass information between MATLAB and an image acquisition device via its driver.

If we didn’t install Hardware Support Packages, we will get a result like this:

```
ans =

  struct with fields:

    InstalledAdaptors: {}
        MATLABVersion: '9.5 (R2018b)'
          ToolboxName: 'Image Acquisition Toolbox'
       ToolboxVersion: '5.5 (R2018b)'
```

Which means no image acquisition adaptors were found.

To install the support package, we go to Add-Ons explorer (Figure 3.1). We type in search box "Image Acquisition Toolbox Support Package for OS Generic Video Interface" and install it.

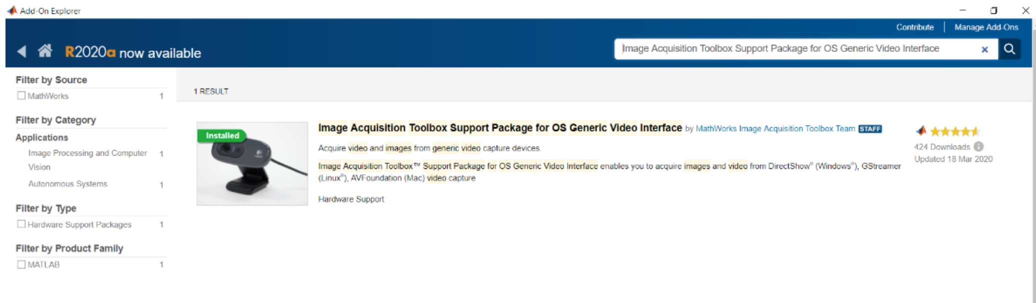

Figure 3.1 – MATLAB Add-Ons explorer

Again we call the function `Imaqhwinfo();` the result will be:

```
ans =

  struct with fields:

    InstalledAdaptors: {'winvideo'}
         MATLABVersion: '9.5 (R2018b)'
           ToolboxName: 'Image Acquisition Toolbox'
        ToolboxVersion: '5.5 (R2018b)'
```

So, our interface to the image acquisition device is 'winvideo'.

The next step is to construct a video input object and connect with the detected web camera. This object will be used in the image acquisition process. We can do that using the following function:

```
vid = videoinput('winvideo');
```

To view web came properties we use the function:

```
Vid_props = getselectedsource(vid);
```

In our case we have the following properties;

```
Device Specific Properties:
  BacklightCompensation = on
  Brightness = 0
  Contrast = 50
  FrameRate = 30.0000
  Gamma = 300
  Hue = 0
  Saturation = 64
  Sharpness = 50
  WhiteBalance = 4600
  WhiteBalanceMode = auto
```

To preview a stream of image frames we use:

```
preview(vid);
```

Now easily we can Acquire a single image frame:

```
frame = getsnapshot(vid);
```

It is important to switch the image frames stream on first before acquiring images, because it saves a lot of time. To clarify this point we did the following experiment:

```
>> tic; img = getsnapshot(obj); toc;
Elapsed time is 1.025977 seconds.
>> preview(obj);
>> tic; img = getsnapshot(obj); toc;
Elapsed time is 0.004704 seconds.
>> 1.025977/0.004704

ans =

  218.1074
```

As we can see acquiring images after the switching the stream on is 218 times faster than acquiring images without switching the video stream in advance.

3.2 Simulink ball model

The main block in this model is “VR Sink”. This block takes .wrl file as source file. According to how the .wrl file was built, VR sink takes more inputs. Ball model shown in Figure 3.2.

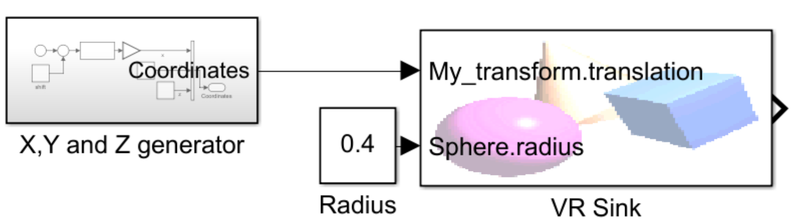

Figure 3.2 – Simulink scheme of the Target

'X, Y and Z generator' is a subsystem provides input signal to VR Sink scheme of which is shown in Figure 3.3.

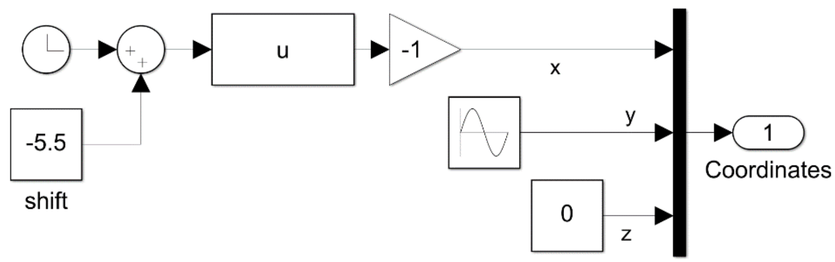

Figure 3.3 – X, Y and Z generator

Open "VR Sink" block, you get a new window shown in Figure 3.4

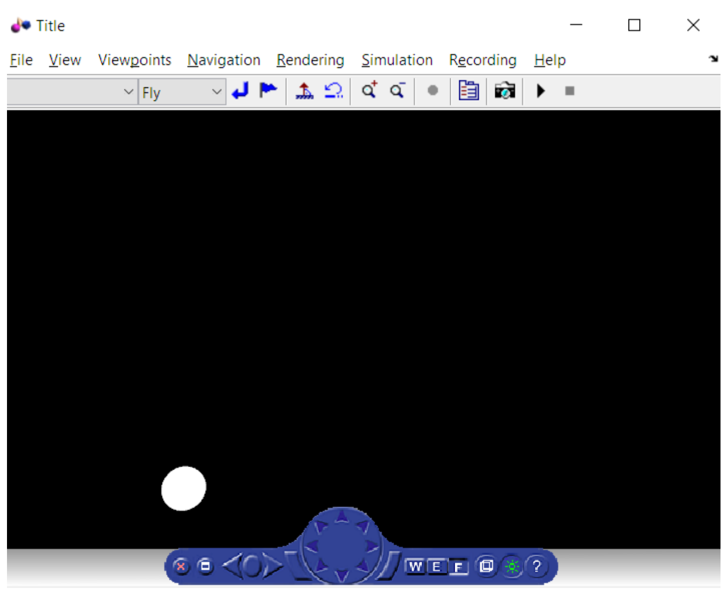

Figure 3.4 – VR Sink video display

In Figure 3.4, choose simulation→Block Parameters then you get new window shown in Figure 3.5.

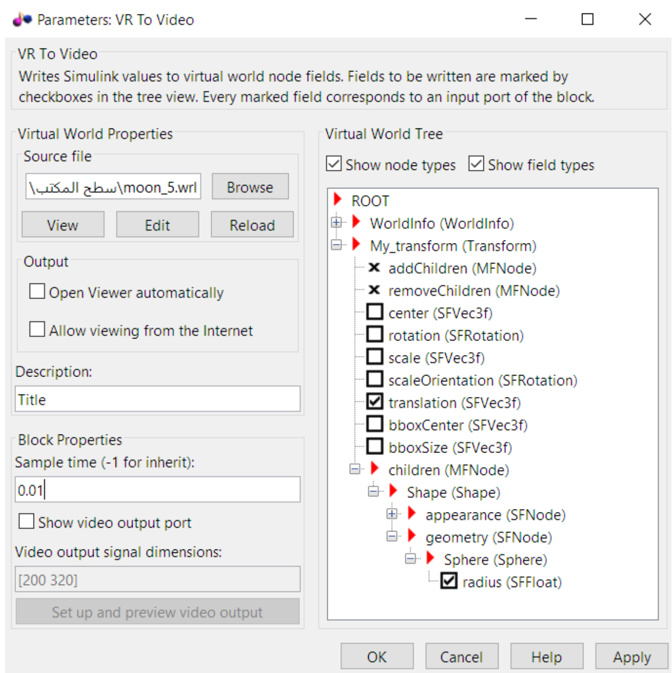

Figure 3.5 – VR to video parameters

In source file press browse and choose the .wrl file you created (it is explained in next subsection). In "Virtual World Tree" check both 'translation' and 'radius' checkboxes. Sample time is chosen so that it is suitable to the camera used. Lower sample rate means slower object movements and vice versa.

3.3 MATLAB Virtual World Editor

Ball model was created with the Virtual World Editor which can be accessed by typing:

```
>> vredit
```

In command window in MATLAB. The default window is shown in Figure 3.6.

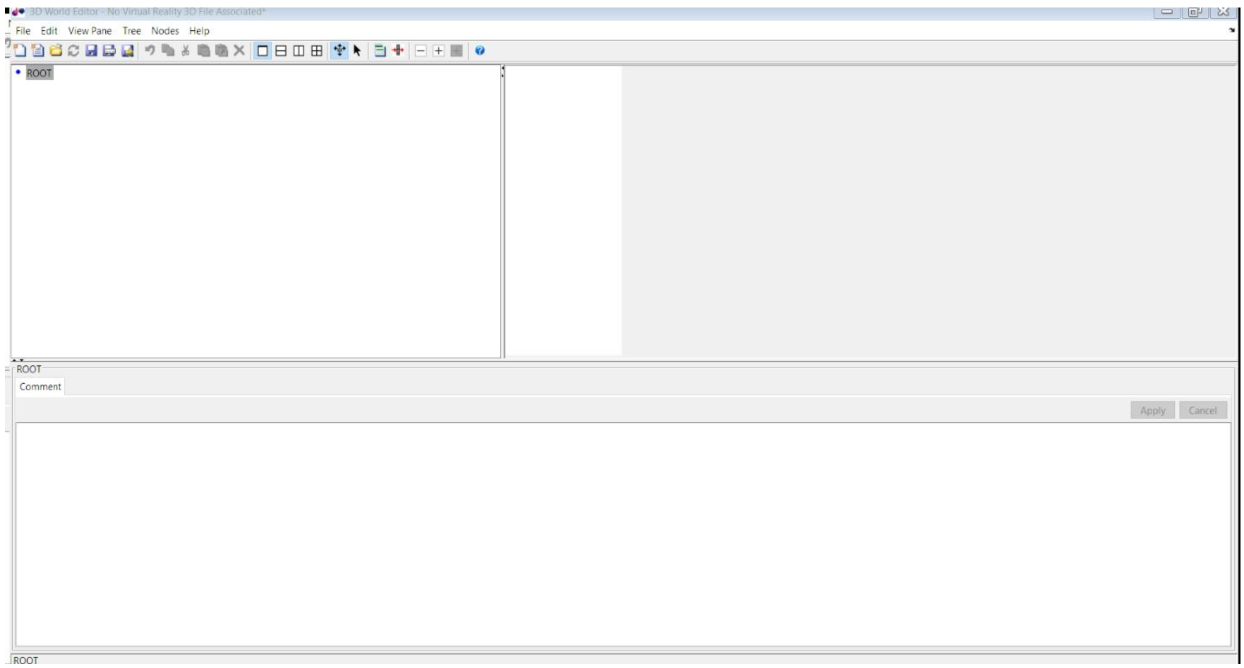

Figure 3.6 – start window in vredit

We need this model to take x, y and z coordinates as input, also we need it to take ball radius as another input. This is necessary to simulate our system when distance between screen and arm is changed from one experiment to another.

Here are steps to create the ball model:

1. Under ‘Root’ node add “WorldInfo”.
2. Under ‘Root’ node add “group→transform”.
3. Under ‘Children’ node add “common→shape”.
4. Under ‘Geometry’ node add ‘geometry→sphere”.
5. It is necessary to change the name of each element you add.

Object tree is similar to what is shown in Figure 3.7.

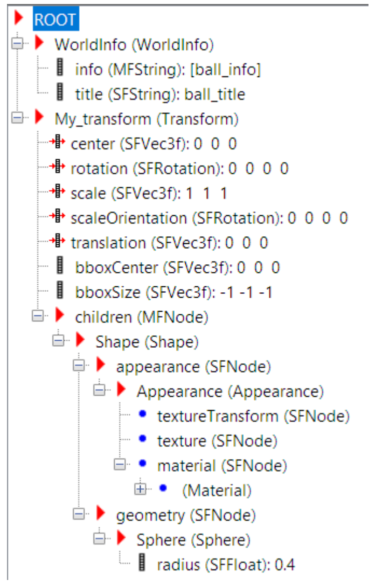

Figure 3.7 – Target tree in vredit

The whole design is show in Figure 3.8.

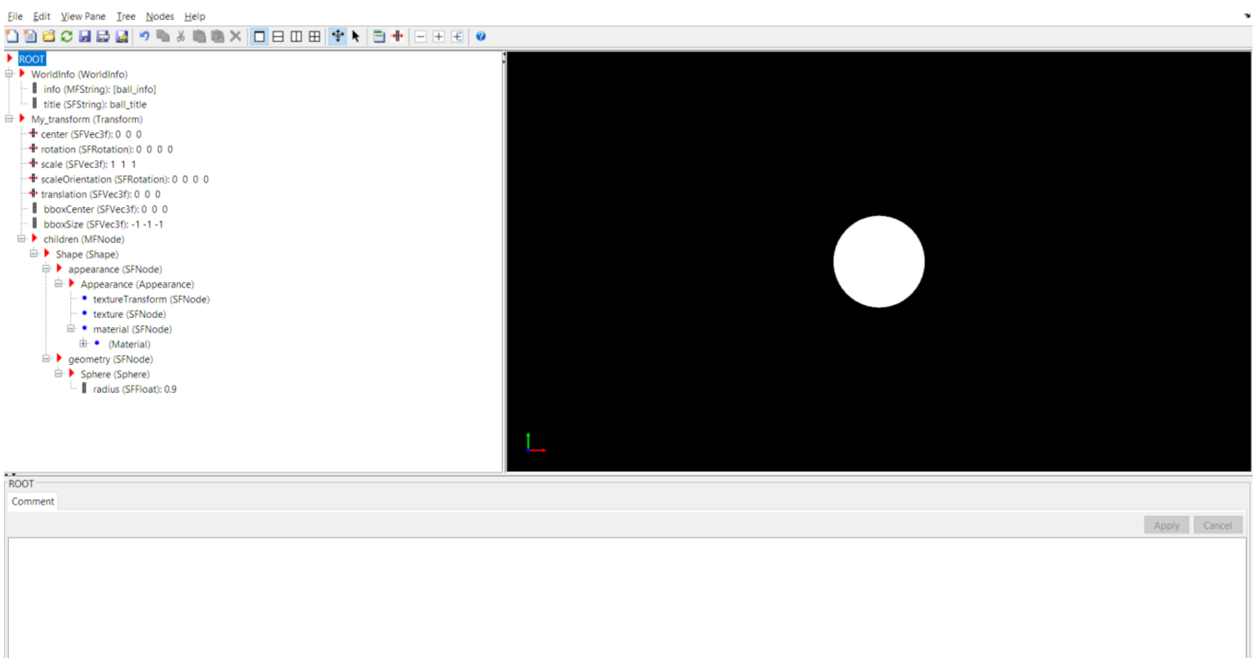

Figure 3.8 – complete design of Target in vredit

3.4 Camera Calibration

Before starting with camera calibration, we need to explain the pinhole camera model.

3.4.1 Pinhole camera model:

Camera is a mapping device from 3D world to a 2D image. This mapping could be formulated as:

$$\left.\begin{array}{l} x = P.X \\ \begin{bmatrix} x \\ y \\ z \end{bmatrix} = P.\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \end{array}\right\} \quad (3.1)$$

So, the dimensions of the matrix P are 3x4. Consider the geometric representation in Figure 3.9:

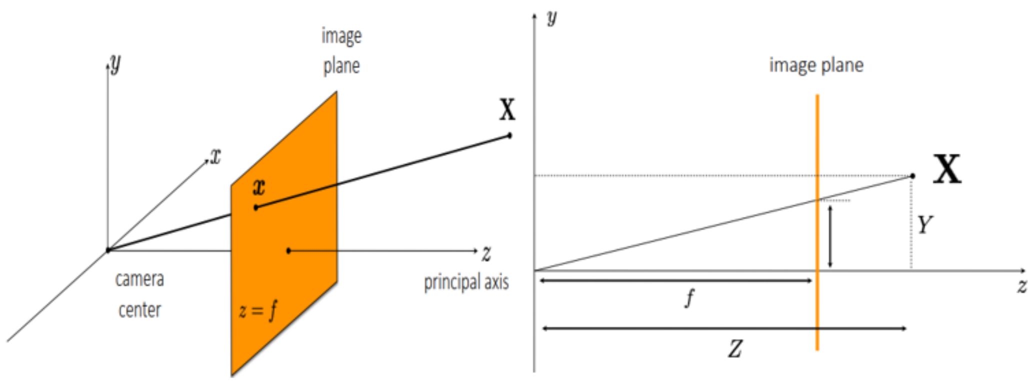

Figure 3.9 – elementary representation of 3D point in camera frame

The point X in the 3D world is mapped to the 2D plan as x. x is represented in image plane as:

$$[X \quad Y \quad Z] \rightarrow [fX/Z \quad fY/Z] \quad (3.2)$$

So now we can write:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} . \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (3.3)$$

In fact, the camera and image have different coordinate systems as shown in Figure 3.10.

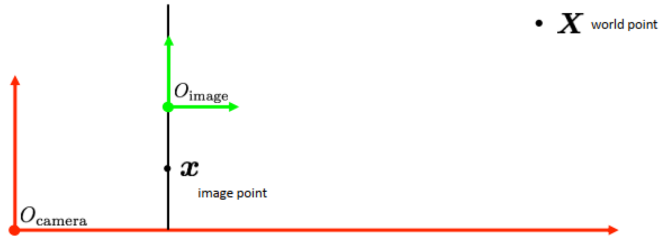

Figure 3.10 – image CS0 alignment from camera CS

Which means that matrix P should change to:

$$P = \begin{bmatrix} f & 0 & p_x & 0 \\ 0 & f & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{3.4}$$

Matrix P can be decomposed to:

$$x = P.X \tag{3.5.1}$$

$$P = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} . \left[\begin{array}{ccc|c} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{array}\right] \tag{3.5.2}$$

let's state the result now: we got a mapping between 3D point represented in camera coordinate system and 2D point allocated in image plane and represented in the same coordinate system of 3D point.

In general, there are three not just two coordinate systems as show in Figure 3.11.

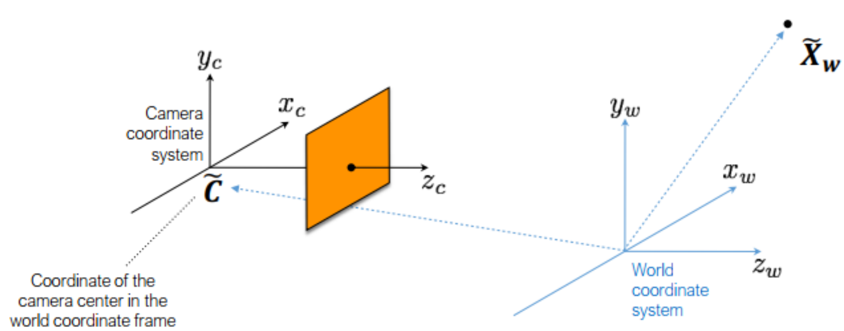

Figure 3.11 – general representation of World, Camera and image CSs

So, a point in world coordinate system need a conversion to be represented in camera coordinate system. This conversion is written as:

$$R.\left(\tilde{X}_W - \tilde{C}\right) \tag{3.6}$$

Substituting 3.6 in 3.5 yields:

$$\left.\begin{aligned}
&x = P.X \\
&P = \begin{bmatrix} f & 0 & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} . \left[\begin{array}{c|c} R & -R.C \\ 0 & 1 \end{array}\right] \\
&P = K.R.\left[I \middle| -C\right] \\
&P = K.\left[R \middle| t\right] \qquad\qquad : t = -R.\mathrm{C}
\end{aligned}\right\} \tag{3.7}$$

A well-known notation for those parameters is:

P : camera matrix.

K : intrinsic parameters.

$\left[R \middle| t\right]$: extrinsic parameters.

In case camera sensor is skewed, which is the general case, another parameter s is added to camera matrix:

$$\left.\begin{aligned}
&x = P.X \\
&P = \begin{bmatrix} f & s & p_x \\ 0 & f & p_y \\ 0 & 0 & 1 \end{bmatrix} . \left[\begin{array}{c|c} R & -R.C \\ 0 & 1 \end{array}\right]
\end{aligned}\right\} \tag{3.8}$$

3.4.2 Camera calibration:

Camera calibration performed according to the approach represented by Zhang [10] and improved in [9]. More specifically this was done using the “Camera Calibrator” from MATLAB. Figure 3.12 represents the main window in MATLAB camera calibrator.

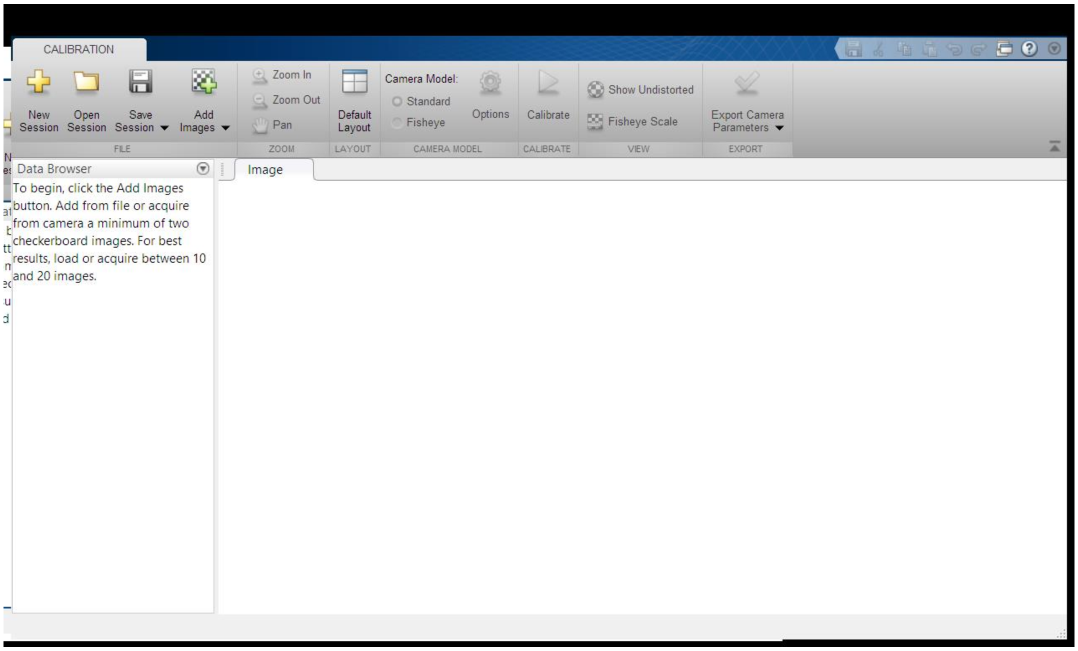

Figure 3.12 – main window in calibrator

This calibrator has a user-friendly interface. User task is just to choose photos and then press calibrate. Once calibration is done, we need to save camera parameters by pressing on “Export Camera Parameters”. Calibrator offers a useful representation of camera and screen plane locations (see Figure 2.5 as an example of this representation).

4 KUKA youBot arm manipulator

A short description from the KUKA youBot manual is quoted:

“ The KUKA youBot comes with fully open interfaces and allows the developers to access the system on nearly all levels of hardware control. It further comes with an application programming interface (KUKA youBot API), with interfaces and wrappers for recent robotic frameworks such as ROS or OROCOS, with an open source simulation in Gazebo and with some example code that demonstrates how to program the KUKA youBot. The platform and the available software shall enable the user to rapidly develop his/her own mobile manipulation applications.

Giving full access to nearly all levels of control of the robot comes at a price, however. It allows the customer to access and control each of its actuators and sensors directly as he or she thinks is best. Customers can develop applications whose functionalities are only limited by the KUKA youBot’s kinematic structure, its drive electronics, and motors.”

It is essentially for us to solve the forward and inverse kinematic tasks of the arm. youBot arm geometric configuration is shown in Figure 4.1.

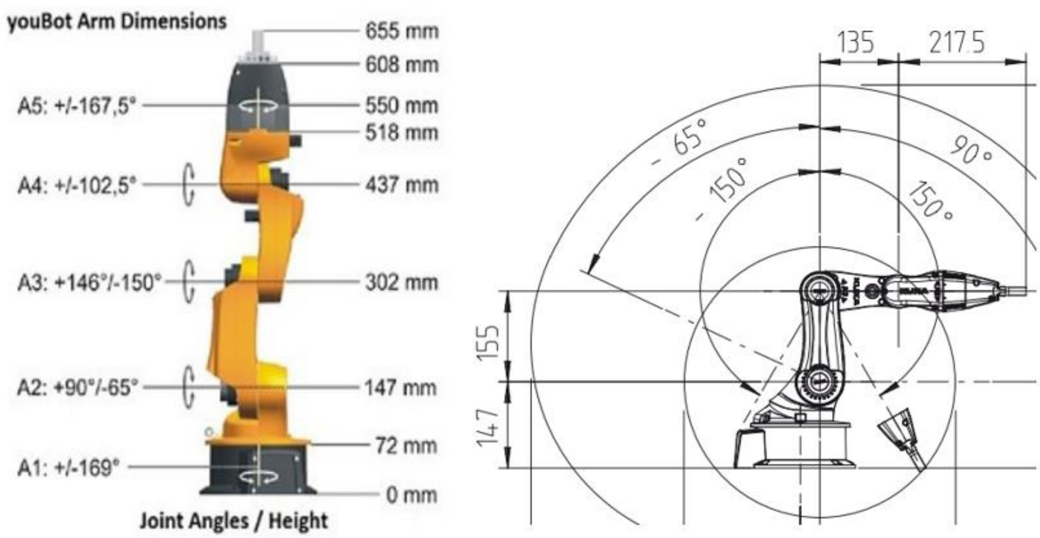

Figure 4.1 – geometrical dimensions of youBot manipulator

4.1 Forward Kinematic task

To build D-H table we need to assign coordinate frames. We can carry out the assignment of those frames according to the following rules:

1. Place the Z_i axis in the direction of joint axis.
2. Place X_i in the direction of the common normal of Z_i, Z_{i-1}.
3. Y_i is then deduced using the right-hand rule.

The result is shown in Figure 4.2.

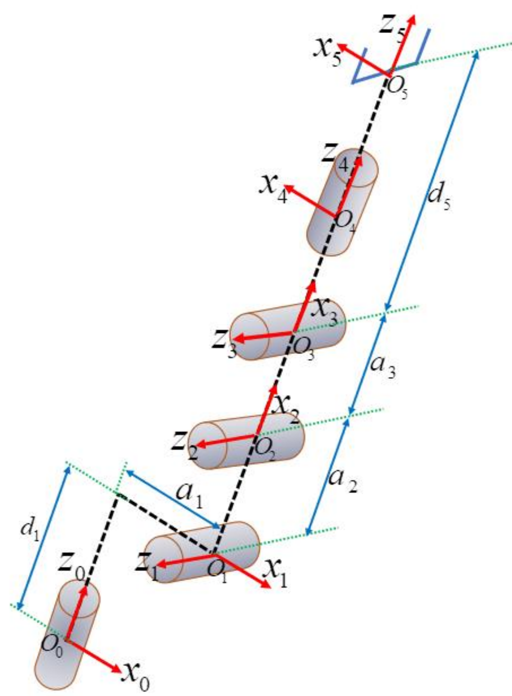

Figure 4.2 – joints coordinate frames in youBot

Where:

$$a_1 = 0.033, a_2 = 0.155, a_3 = 0.135, d_1 = 0.147, d_5 = 0.218 \quad (4.1)$$

measurement unit is meter. To fill D-H table, The following rules must be followed:

1. a_i: is the distance along the axis x_i from o_i to the intersection of the x_i and z_{i-1} axes.

2. d_i: is the distance along the axis z_{i-1} from o_{i-1} to the intersection of the x_i and z_{i-1} axes. d_i is variable if joint i is prismatic.
3. α_i: is the rotation angle around x_i that z_{i-1} needs to become in the direction of z_i .
4. α_i: is the rotation angle around z_{i-1} that x_{i-1} needs to become in the direction of x_i .

Table 4.1 – youBot DH parameters

`	a_i	α_i	d_i	θ_i
1	a_1	$-\pi/2$	d_1	θ_1
2	a_2	0	0	θ_2
3	a_3	0	0	θ_3
4	0	$\pi/2$	0	θ_4
5	0	0	d_5	θ_5

According to DH, any homogeneous transformation between two consequences frames is represented as a production of four transformation:

$$_{i-1}T^i = R_{z,\theta_i}.Trans_{z,d_i}.Trans_{x,a_i}.R_{x,\alpha_i} \tag{4.2}$$

In words: any homogeneous transformation between two consequences frames could be achieved by rotation around z, then linear transition on z, then linear transition on x and rotation around x.

Where:

$$\left.\begin{array}{l} R_{z,\theta_i} = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i} & 0 & 0 \\ s_{\theta_i} & c_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, R_{x,\alpha_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c_{\alpha_i} & -s_{\alpha_i} & 0 \\ 0 & s_{\alpha_i} & c_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ Trans_{z,d_i} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, Trans_{x,a_i} = \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{array}\right\} \tag{4.3}$$

The result of multiplication gives us HT as:

$$_{i-1}T^i = \begin{bmatrix} c_{\theta_i} & -s_{\theta_i}c_{\alpha_i} & s_{\theta_i}s_{\alpha_i} & a_i c_{\theta_i} \\ s_{\theta_i} & c_{\theta_i}c_{\alpha_i} & -c_{\theta_i}s_{\alpha_i} & a_i s_{\theta_i} \\ 0 & s_{\alpha_i} & c_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.4}$$

4.2 Inverse Kinematic task

As we said earlier, arm robot's mission is to verify How accurate the tracking algorithm is. This would be tested by forcing the end effector to follow the moving object.

Our information about the object are the 3D coordinates in the youBot coordinate system. We have no information about the orientation of the object. Maybe one says that it is better to achieve the following task while the last link of the arm is always perpendicular to screen surface. But this is not possible with 5DOf robot arm.

So here we introduce a new approach to solve the IK task for our system. It is called the IK trapezoidal method. **With this approach we avoid every possible singularity.**

The main idea of this approach is: “slope of arm’s fourth link at any time is parallel to the line that connects O_5 and O_1” in Figure 4.3. Consider the following scheme:

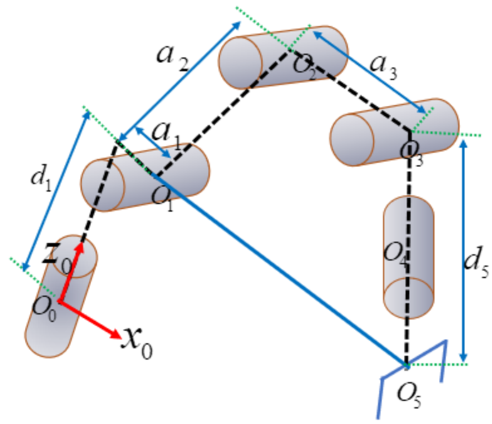

Figure 4.3 – a youBot configuration in trapezoidal method

Which is a random 3D configuration of the arm. 2D representation of Figure 4.3 is shown in Figure 4.4

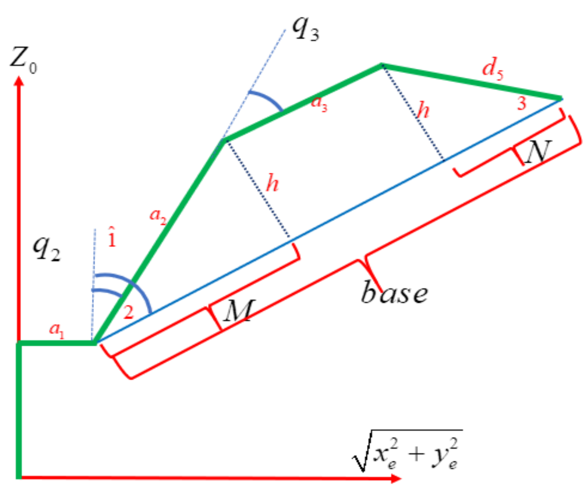

Figure 4.4 – 2D representation of Figure 4.3

The previous 3D representation allows us to obtain the inverse kinematic using triangular laws. This is done using this MATLAB code:

```
q1 = atan(y_obj/x_obj);

% calculate the trapezoid base length:
l_1 = sqrt( (x_obj-a1*cos(q1))^2 + (y_obj-a1*sin(q1))^2 );
base = sqrt(l_1^2+(z_obj-d1)^2);
ang_1 = acos((z_obj-d1)/base);
a = base;
b = a2;
c = a3;
d = d5;
h = sqrt( (a+b-c+d)*(-a+b+c+d)*(a-b-c+d)*(a+b-c-d)/(4*(a-c)^2) );
m = sqrt( b^2-h^2 );
ang_2 = acos( (b*b+m*m-h*h)/(2*m*b) );

% q2
q2 = ang_1 - ang_2;

% q3
q3 = ang_2;

% q4
n = a - m - c;
ang_3 = acos( (n*n+d*d-h*h)/(2*n*d) );
q4 = ang_3;

% we assume that the fifth joint is not moving at all
q5 = pi;

q = [q1 q2 q3 q4 q5];
```

5 Dynamic Regressor Extension and Mixing (DREM)

It is an algorithm for parameters estimation with enhanced performance. For classical linear regression forms, DREM ensures convergence without the irritating condition of persistent excitation.

DREM consists of two main stages:

1. Generate a new regression format via the application of dynamic operator on the original regressor.
2. Generate a convenient mix of these new data to obtain the desired regression format. To the desired regression we will apply standard parameters estimation techniques.

5.1 DREM mathematical model:

In this research we are not going to go through the mathematical proof of DREM. Here we point out the necessary procedures to carry out frequency estimation in an input signal.

We refer to the pattern detected by the camera as $\hat{g}(t)$. Which is the input signal to DREM. Filter the signal with Hurwitz filter of the form:

$$\xi(t) = \frac{\lambda^{2l}}{(p+\lambda)^{2l}} \hat{g}(t) \tag{5.1}$$

Where $(p+\lambda)^{2l}$ is a Hurwitz polynomial, $p = d/dt$ is the differentiation operator, l is the number of frequencies existed in the input signal.

Next, construct the following linear regression model:

$$\xi^{(2l)}(t) = \upsilon^T(t)\vartheta + \varepsilon(t) \tag{5.2}$$

Where $\varepsilon(t)$ is an exponentially decaying term. The regressor is composed of consecutive derivatives of the filter:

$$\upsilon^T(t) = \begin{bmatrix} \xi^{(2l-2)}(t) & \cdots & \xi^{(2)}(t) & \xi(t) \end{bmatrix} \tag{5.3}$$

And the estimated vector is written as:

$$\vartheta^T = \begin{bmatrix} \bar{\theta}_1 & \cdots & \bar{\theta}_{l-1} & \bar{\theta}_l \end{bmatrix} \tag{5.4}$$

Elements of ϑ are related with the following system of equations:

$$\begin{cases} \bar{\theta}_1 = \quad \theta_1 + \theta_2 + \cdots + \theta_l \\ \bar{\theta}_2 = -\theta_1\theta_2 - \theta_2\theta_3 - \cdots - \theta_{l-1}\theta_l \\ \vdots \\ \bar{\theta}_l = \quad (-1)^{l+1}\theta_1\theta_2\cdots\theta_l \end{cases} \tag{5.5}$$

Where $\theta_i = -\omega_i^2$ and ω_i is the i_{th} frequency in the input signal.

At this point we can estimate frequencies using Gradient-based frequency estimator by the following law:

$$\dot{\hat{\vartheta}}(t) = K_\vartheta . \upsilon(t)\left(\xi^{(2l)}(t) - \upsilon^T(t)\hat{\vartheta}(t)\right) \tag{5.6}$$

Where $K_\vartheta \in \mathbb{R}^{1\times l}, K_\vartheta > 0$. However, Gradient-based estimator is in its best cases twice slower than DREM.

Choose $l-1$ distinct delays $\left(d_i, i \in \{1,2,...,l-1\}\right)$, then set $l-1$ filtered signals as follows:

$$\upsilon_{f_i}(t) = \upsilon(t - d_i) \tag{5.7}$$

$$\xi_{f_i}(t) = \xi(t - d_i) \tag{5.8}$$

Using the previous new elements generate two new main vectors as follows:

$$\Upsilon_e(t) := \begin{bmatrix} \upsilon^T(t) \\ \upsilon_{f_i}^T(t) \\ \vdots \\ \upsilon_{f_{l-1}}^T(t) \end{bmatrix} \tag{5.9}$$

$$\Xi_e(t) := \begin{bmatrix} \xi(t) \\ \xi_{f_i}(t) \\ \vdots \\ \xi_{f_i}(t) \end{bmatrix} \tag{5.10}$$

Where $\Upsilon_e(t) \in \mathbb{R}^{l\times l}, \Xi_e(t) \in \mathbb{R}^{l\times 1}$.

Define two new variables as follows:

$$\psi_\phi(t) := \det\{\Upsilon_e(t)\} \tag{5.11}$$

$$\Xi(t) := adj\{\Upsilon_e(t)\}.\Xi_e(t) \tag{5.12}$$

The previous equations give us a set of l equations:

$$\Xi_i(t) = \psi_\phi(t).\vartheta_i \qquad :i \in \{1,2,3,\ldots,l\} \tag{5.13}$$

Thus, estimation law is written as follows:

$$\dot{\hat{\vartheta}}_i(t) = \gamma_i.\psi_\phi(t)\left(\Xi_i(t) - \psi_\phi(t)\hat{\vartheta}(t)\right) \tag{5.14}$$

Where $\gamma_i > 0$.

Below, just for instance, we carry out an application of DREM and Gradient-based frequency estimator on continuous input signal of the form:

$$g(t) = \sin(3.54t) + \sin(9.45t) \tag{5.15}$$

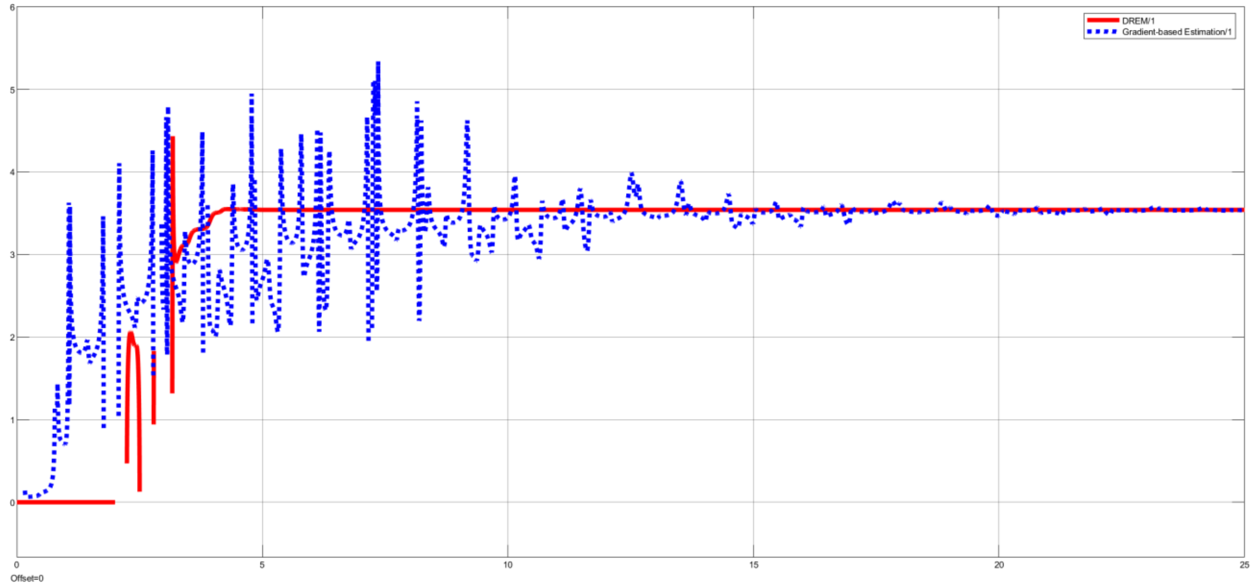

Figure 5.1 – w1 estimation with DREM (continuous line) and Gradient-based (dotted line)

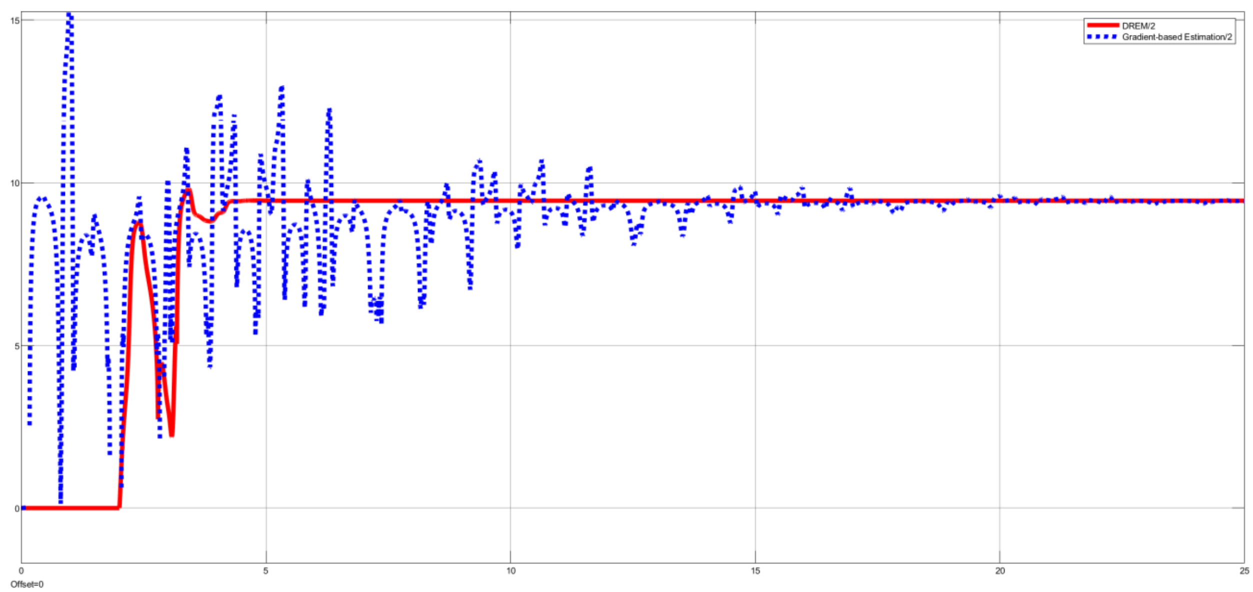

Figure 5.2 – w2 estimation with DREM (continuous line) and Gradient-based (dotted line)

As you can see in Figure 5.1 and Figure 5.2, DREM (the continuous lines) much faster than Gradient-based frequency estimator (the dotted lines).

5.2 DREM implementation in Simulink

DREM simulation in Simulink in not that easy, therefore we clarify (as an example) how to simulate it for a signal of the form:

$$g(t) = \sin(t) + \sin(2t) + \sin(3t)$$

The general scheme shown in Figure 5.3.

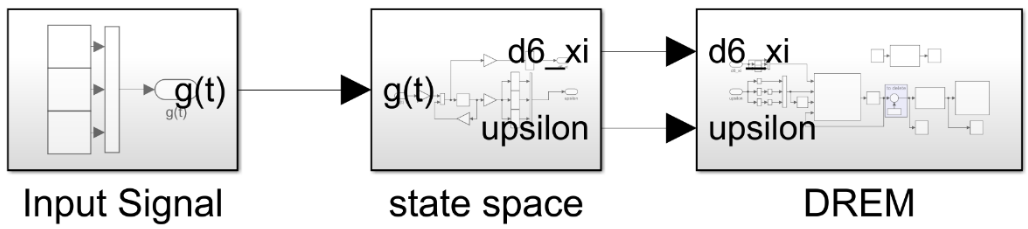

Figure 5.3 – DREM scheme in Simulink

From equation 5.1 we understand that input signal should be passed over a filter of the 6th degree. But as shown in equation 3, we need the consecutive derivatives of $\xi(t)$. To obtain that we convert the filter to a state space. To do that we perform the following steps:

1. Obtain the filter characteristic equation:

```
syms lmda s
degree = 6;
denominator = (s+lmda)^6;
char_equ = expand(denominator)
```

$$TF = \frac{\lambda^6}{s^6 + 6\lambda s^5 + 15\lambda^2 s^4 + 20\lambda^3 s^3 + 15\lambda^4 s^2 + 6\lambda^5 s + \lambda^6} \tag{5.16}$$

2. Find filter transfer function:

```
numerator = [lmda^6];
denominator = [1 6*lmda 15*lmda^2 20*lmda^3 15*lmda^4
6*lmda^5 lmda^6];
f = tf(numerator, denominator)
```

3. Convert the filter to state space:

$$\left.\begin{array}{l}\dot{\xi}=A.\xi+B.g \\ A=\begin{bmatrix} -6\lambda & -15\lambda^2 & -20\lambda^3 & -15\lambda^4 & -6\lambda^5 & -\lambda^6 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \\ \mathrm{B}=\begin{bmatrix}1 & 0 & 0 & 0 & 0 & 0\end{bmatrix}^T \end{array}\right\} \tag{5.17}$$

Equation 15.17 could be written in another form as:

$$\left.\begin{array}{l}\dot{\xi}=A.\xi+B.g \\ \dot{\xi}_1=-6\lambda\xi_1-15\lambda^2\xi_2-20\lambda^3\xi_3-15\lambda^4\xi_4-6\lambda^5\xi_5-\lambda^6\xi_6+g \\ \dot{\xi}_2=\xi_1 \\ \dot{\xi}_3=\xi_2 \\ \dot{\xi}_4=\xi_3 \\ \dot{\xi}_5=\xi_4 \\ \dot{\xi}_6=\xi_5 \end{array}\right\} \tag{5.18}$$

This is enough to find the vector $\upsilon(t)$:

$$\upsilon^T(t)=\begin{bmatrix}\xi_2(t) & \xi_4(t) & \xi_6(t)\end{bmatrix} \tag{5.19}$$

And:

$$\xi^6(t)=\upsilon^T(t)\vartheta+\varepsilon(t)=\dot{\xi}_1(t) \tag{5.20}$$

Figure 5.4 represents Simulink scheme of equations 5.17-5.20 implementation, which is the subsystem “state space” in Figure 5.3.

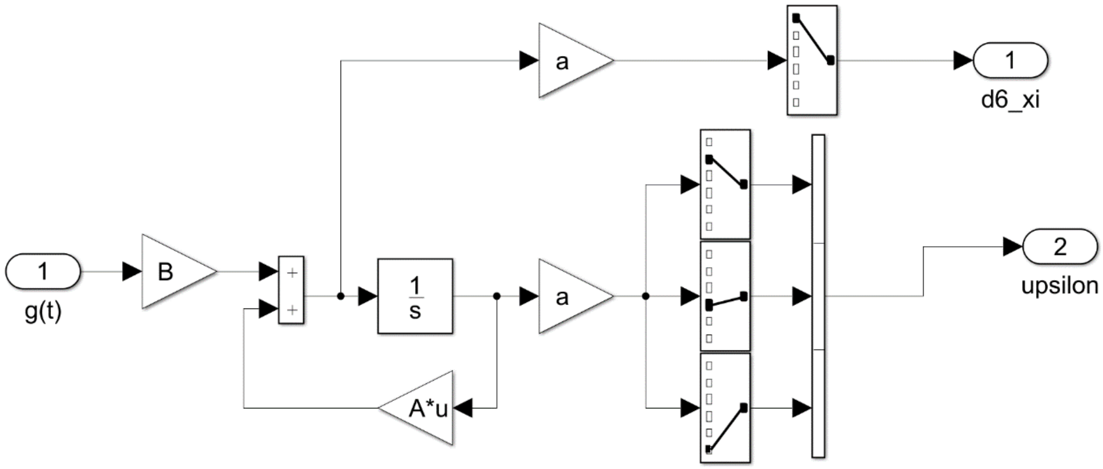

Figure 5.4 – Simulink scheme to convert transfer function to state space

4. Implement equations 5.9 and 5.10 by simply adding two delay blocks with different delay values, then concatenate the resulting terms as shown in Figure 5.5.

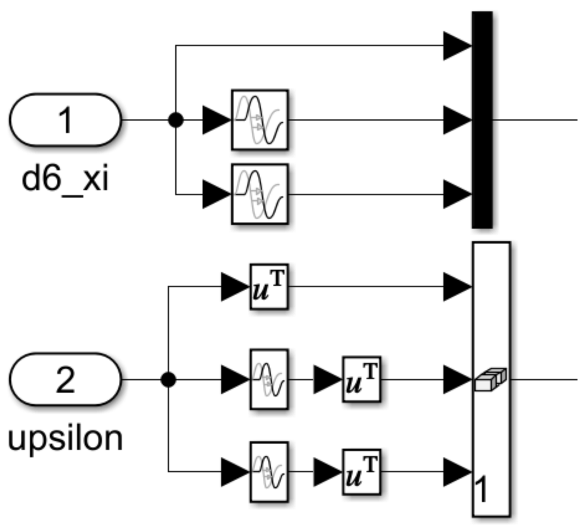

Figure 5.5 - equations 5.9 and 5.10 implementation

5. Now estimate $\hat{\vartheta}(t)$ using the following code:

```
    function d_vartheta_hat =
fcn(Xi_e,Upsilon,adj_Upsilon,vartheta_hat)
    d_vartheta_hat = zeros(3,1);
    psi = det(Upsilon);
    Xi = adj_Upsilon*Xi_e;
    gama =11;
    d_vartheta_hat = gama*psi*(Xi - psi*vartheta_hat);
```

6. To find out frequencies from $\hat{\vartheta}(t)$ we need to solve the system of equations represented in equation 5.5.

```
    function w = fcn(vartheta_hat)

    [a, b, c] = vartheta_hat(1:3);
    r2 = [0+0*i; 0+0*i];
    p2 = [1 -a -b];
       d = sqrt(complex(p2(2)^2 - 4*p2(3)));
       r2(1) = ( -p2(2) + d ) /2;
       r2(2) = ( -p2(2) - d ) /2;
    r2 = real(r2);
    if (r2(1)<=0)
        tta2 = r2(1)
    else
        tta2 = r2(2);
    end
    r3 = [0+0*i; 0+0*i];
    p3 = [1 -a+tta2 c/tta2];
       d = sqrt(complex(p3(2)^2 - 4*p3(3)));
       r3(1) = ( -p3(2) + d ) /2;
       r3(2) = ( -p3(2) - d ) /2;
    r3 = real(r3);
    if (r3(2)<=0)
        tta3 = r3(2)
    else
        tta3 = r3(1);
    end
```

```
    tta1 = a - tta2 - tta3;
    w = [sqrt(complex(-tta1)); sqrt(complex(-tta2));
sqrt(complex(-tta3))]
    end
```

7. Steps 4 through 6 are implemented in subsystem called “DREM” in Figure 5.3. The scheme of this subsystem is shown in Figure 5.6.

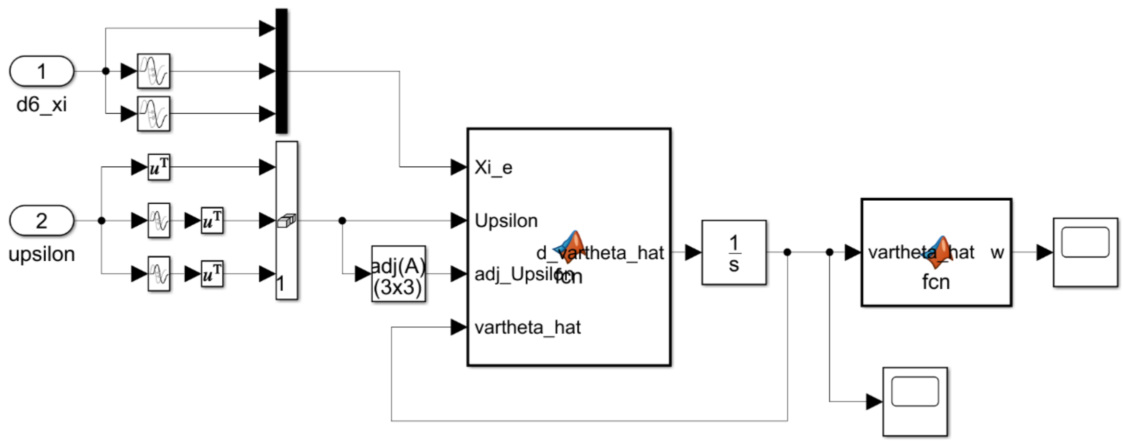

Figure 5.6 – linear delays implementation

8. Simulation results are represented in Figure 5.7.

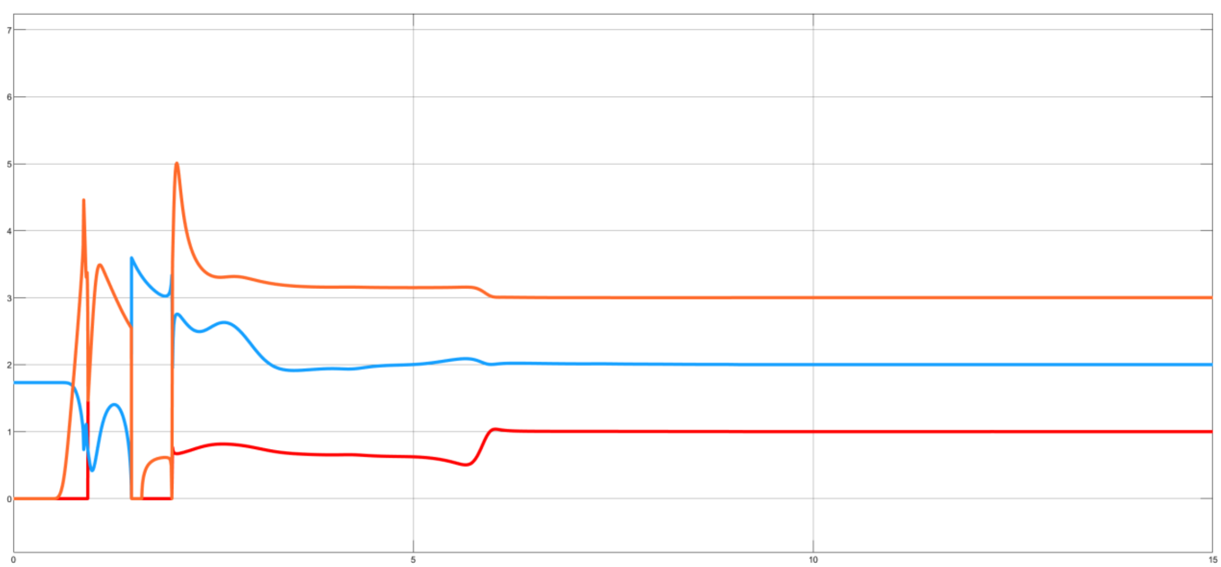

Figure 5.7 – frequencies estimation

5.3 Discrete DREM

In real world, since we take frame each 100mSec, it means we are dealing with discrete signals. So, we can't use the Simulink model of continuous DREM to estimate frequencies. We should implement the discrete counterpart of DREM.

As we clarified in previous sections, to filter our signal and still being able to access the consecutive derivatives we converted the filter to state space. Here we need to convert the continuous state space to a discrete one.

$$\dot{\xi} = A.\xi(t) + B.g(t) \qquad (continous\ model)(5.21)$$

$$\xi[k+1] = A_d\xi[k] + B_d g[k] \qquad (discrete\ model)(5.22)$$

Matrices A_d, B_d are calculated according to those formulas:

$$A_d = e^{A\Delta t} \qquad (5.23)$$

$$B_d = A^{-1}(A_d - I_n)B \qquad (5.24)$$

Where A, B are the same as in equation 5.17, I_n is the unity matrix of size n (n is the size of A), Δt is time interval between samples and:

$$e^{A\Delta t} \approx I_n + A\Delta t + \frac{(A\Delta t)^2}{2} + ... + \frac{(A\Delta t)^k}{k!} \qquad (5.25)$$

Below we explain how to implement discrete DREM for an input signal with two frequencies. For signals with more frequencies, we need just to change matrices sizes.

```
%% Create Discret filter
gamma =3;
a = [gamma^4];
b= [1 4*gamma 6*gamma^2 4*gamma^3 gamma^4];
A=[-b(2) -b(3) -b(4) -b(5);
    1       0      0      0;
    0       1      0      0;
    0       0      1      0];
B = [1;0;0;0];

Ad = eye(4);
for j=1:15
    Ad = Ad + (A*dt)^j/(factorial(j));
end
Bd = A\(Ad-eye(4))*B;
```

Figure 5.8 – discrete DREM, code part_1

Lines 13 through 20 in Figure 5.8 are responsible to convert the filter of degree 4 to continuous state space model. Lines 22 through 26 convert continuous state space to discrete one.

```
29    x(1:4,1) = [0;0;0;0];
30    mu = zeros(2,Ft/dt);
31    d_mu = zeros(2,Ft/dt);
32    d4_ksi = zeros(Ft/dt);
33    Xi_e = zeros(2,Ft/dt);
34    Upsilon = zeros(2,2,Ft/dt);
35
```

Figure 5.9 – discrete DREM, code part_2

Lines 29 through 34 in Figure 5.9 initialize system variables, where dt is time interval between samples, Ft is observation time. In other words, Ft is the time interval, through which we acquire frames from camera. Variables from line 30 through 34 are $\hat{\vartheta}, \dot{\hat{\vartheta}}, \xi^{(4)}(t), \Xi_e(t), \Upsilon_e(t)$ respectively (see equations 5.4, 5.14, 5.20, 5.10 and 5.9).

```
83
84    for k=2:1:Ft/dt
85        x(1:4,k) = Ad*x(1:4,k-1) + Bd(1:4)*u(k-1);
86        slope = (x(1,k)-x(1,k-1))/dt;
87        d4_ksi(k) = slope*a;
88        Upsilon(1,1:2,k) = ([x(2,k) x(4,k)])*a;
89        if (k-5)<2
90            Upsilon(2,1:2,k)=0;
91        else
92            Upsilon(2,1:2,k) = Upsilon(1,1:2,k-5);
93        end
94        Xi_e(1,k)=d4_ksi(k);
95        if (k-5)<2
96            Xi_e(2,k)=0;
97        else
98            Xi_e(2,k)=d4_ksi(k-5);
99        end
100       gama=3;
101       Xi = [Upsilon(2,2,k) -Upsilon(1,2,k); -Upsilon(2,1,k) Upsilon(1,1,k)]*Xi_e(:,k);
102       psi = det(Upsilon(:,:,k));
103       d_mu(:,k) = gama*psi*(Xi-psi*mu(:,k-1));
104       mu(:,k) = mu(:,k-1) + (d_mu(:,k)+d_mu(:,k-1))/2*dt;
```

Figure 5.10 – discrete DREM, code part_3

Lines 84 through 104 in Figure 5.10 estimate $\hat{\vartheta}$. Then we move to solving system of equations (see equation 5) to find frequencies. In our case we have two frequencies, so solving this system is quite easy. It is clarified in Figure 5.11.

```
    temp_mu(:,k) = -1*mu(:,k);
    b = -temp_mu(1,k);
    c = temp_mu(2,k);

    delta = b^2 -4*c;
    tta1(k) = -b/2 - sqrt(abs(delta))/2;
    tta2(k) = -b/2 + sqrt(abs(delta))/2;

    tta1(k) = sqrt(real(tta1(k)));
    tta2(k) = sqrt(real(tta2(k)));
end
```

Figure 5.11 – discrete DREM, code part_4

6 Real Experiments in Laboratory

In this chapter, we present results of three real experiments were carried out. The first experiment was carried out in the laboratory of ITMO university. The last two were carried out in a normal room using mobile phone camera[1]. Each experiment involves different from the other experiments moving pattern. For each experiment we proceed according to the following steps:

1. Show the detected pattern.
2. Estimate frequencies in the detected pattern.
3. Generate future trajectory.
4. Find tracking error.

Experiment 1 was performed with a Target radius of 0.4, which means a quite big Target. This leads to a noisy detected pattern. This in turn leads to not stable (oscillating) DREM estimation. Experiments 2 and 3 were performed with a Target radius of 0.1, which means a quite small Target. This leads to a detected pattern with small amplitude noise. Which lead to more stable DREM estimation.

Because of the situation described in (footnote [1]) we draw the predicted future trajectory on the same frame in which we detected the movement pattern, so the tracking error is measured in pixels. This is a good and dynamic representation of the accuracy of our system, so you can know the real error tracking depending on how far the object surface is from the camera.

As mentioned earlier, Target moves on the said LCD screen. Figure 6.1 is one of many frames captured with the camera.

[1] Research team started performing real experiments at the university in the same period when a decision was taken to close universities because of coronavirus. So, two of those experiments were carried out in the dormitory of ITMO. And because of the fact that research team no longer has access to youBot, system accuracy is tested by drawing the estimated future trajectory on the same frames acquired while the object was moving. And tracking error was so measured in pixels.

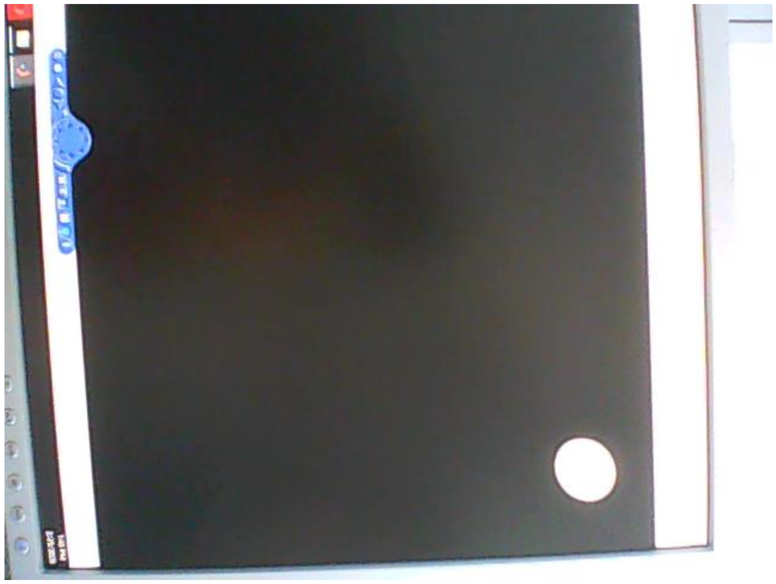

Figure 6.1 – example of acquired frame for the target

6.1 Experiment_1: moving pattern of one frequency

The ball object was assigned with the following moving pattern:

$$y(t) = 3.3\sin(2\pi.f.t) = 3.3\sin(2\pi.0.318t)$$

This object was observed for what about 20 seconds. As we early said, we take frame each 100 mSec, for each frame we detect the object and store frame time and object coordinate. For this experiment the detected moving pattern is shown in Figure 6.2.

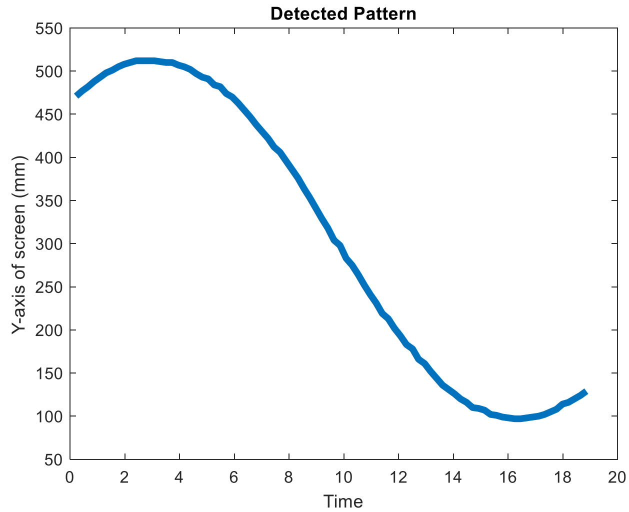

Figure 6.2 – detected moving pattern

As we said earlier, we do enhancement on the detected pattern, so that it becomes a suitable input for DREM algorithm. The enhanced pattern is shown in Figure 6.3.

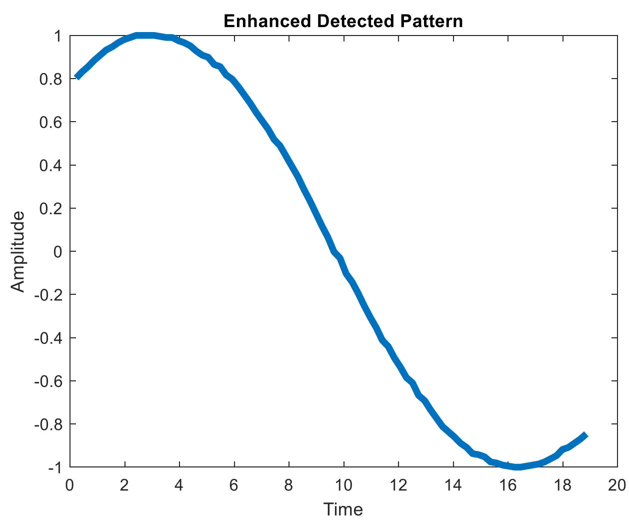

Figure 6.3 – Detected pattern after enhancement

Now we pass the enhanced pattern as input to DREM. DREM estimation is shown in Figure 6.4.

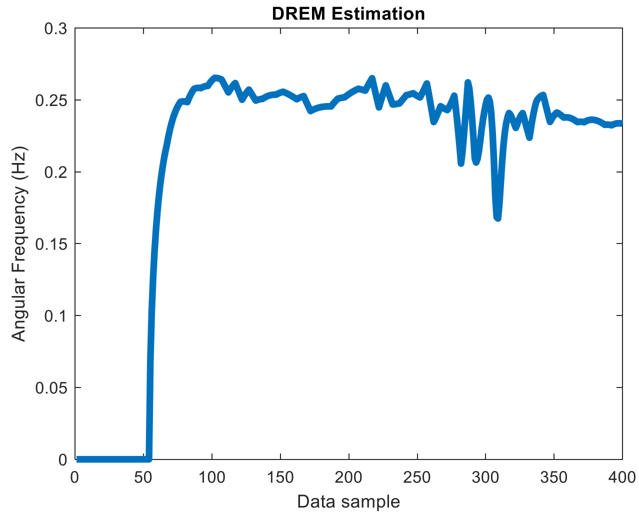

Figure 6.4 – frequency estimation with DREM

As you can see in Figure 6.4, the frequency estimation doesn’t converge to some fixed value. The is because the input signal is noisy. Noise occurs from errors in vision system and camera distortion. However, if we take the average value of estimation over the last two thirds, we get good estimation for tracking task.

For this example, we obtained:

$$\omega = 0.2440 = 2\pi f$$
$$\Rightarrow f = 0.0388$$

It is to be noticed that frequency in input signal and the estimated one have big difference in values. Why is that?

As we said, our object was generated with Simulink. The block “VR sink” takes the 3D model of the object and displays it as a video stream with some specified sample rate. This process has big computation cost which means that we can’t show the object in real time as specified in sinusoidal signal. The object will move exactly as specified with input signal but slower than it. And it gets slower and slower as we increase sample rate. This is not a problem because software of our system records the real time of each frame and it calculates the physical counterpart of time as a shift in millimeters on x axis.

To find tracking error we observe the object for extra 17 seconds, and for the same extra time we predict the future position of the object using our software. Thus, tracking error will be the subtracting of those two figures; as shown in Figure 6.7.

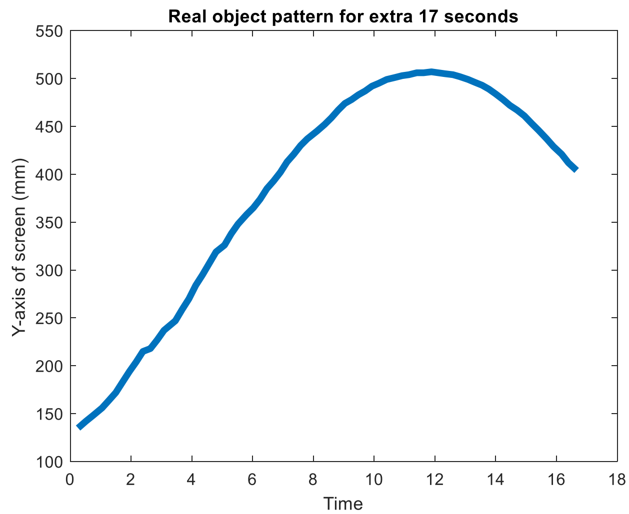

Figure 6.5 – real future trajectory

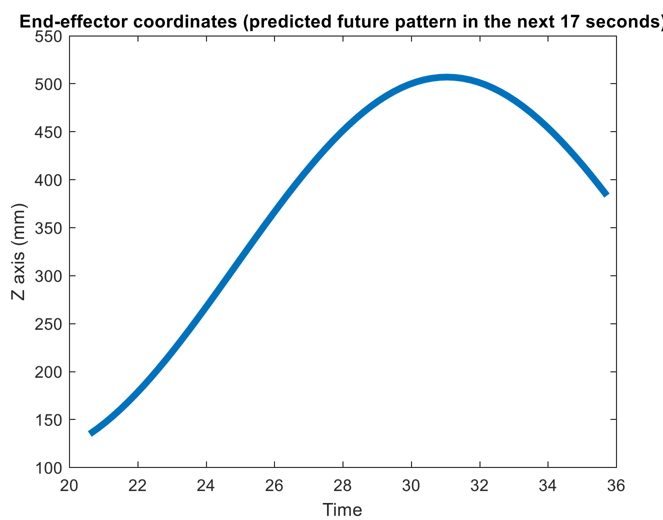

Figure 6.6 – predcited future trajectory

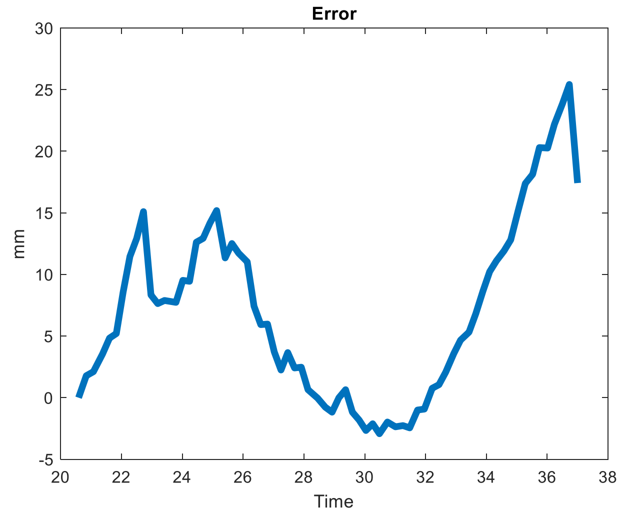

Figure 6.7 – tracking error

As you can see in Figure 6.7, maximum error value is 25 pixels. If the screen is on 400 mm from the camera, then 25pixels = 4.95mm. it is to be noticed that error value gets bigger with time because lack of information obout object position.

6.2 Experiment_2: moving pattern of two frequencies

The ball object was assigned with the following moving pattern:

$$y(t) = 2.66\left(\sin(0.19t) + \sin(1.69t)\right)$$

For this experiment, all figures and results are shown in Figure 6.8.

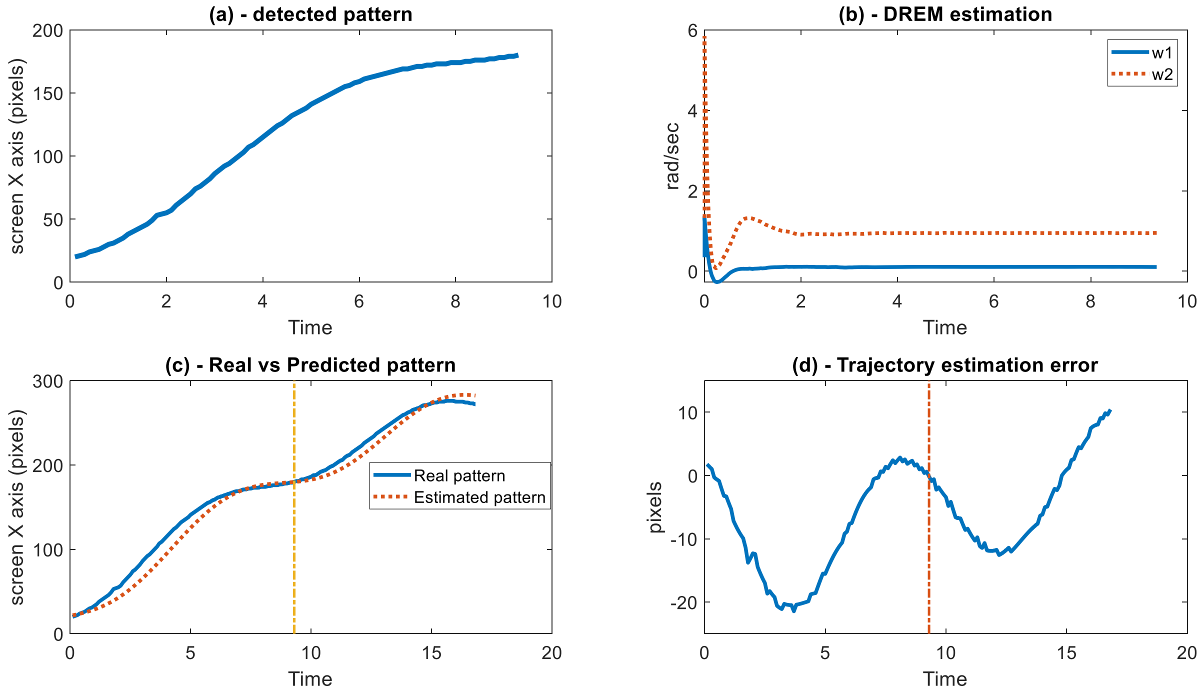

Figure 6.8 – Results of experiment 2. Figure (a) represents the detected pattern. Figure (b) represents frequencies estimation with DREM. Figure (c) represents the estimated trajectory vs the reals trajectory. the dashed vertical line divides the figure into two areas, left during observation, right after turning off the camera. Figure (d) represent the trajectory estimation in pixels. the dashed vertical line divides the figure into two areas, left during observation, right after turning off the camera.

As you can see in Figure 6.8, maximum error value is less than 25 pixels. For a screen about 500mm from the camera, this error value = 4.9mm. It is to be noticed that error value gets bigger with time because lack of information obout object position.

6.3 Experiment_3: moving pattern of four frequencies

The ball object was assigned with the following moving pattern:

$$y(t) = 0.66\left(\sin(0.19t) + \sin(0.09t) + \sin(0.39t) + 0.2\sin(1.69t)\right)$$

For this experiment, all figures and results are shown in Figure 6.9.

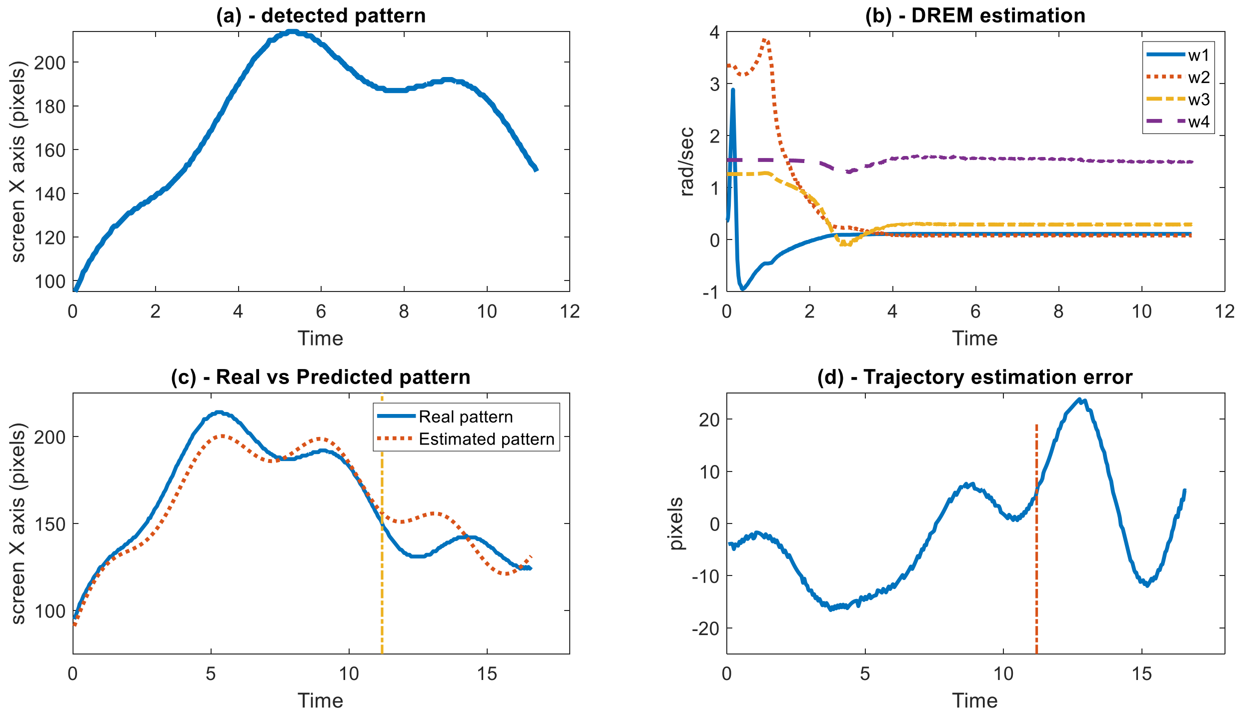

Figure 6.9 – Results of experiment 3. Figure (a) represents the detected pattern. Figure (b) represents frequencies estimation with DREM. Figure (c) represents the estimated trajectory vs the reals trajectory. the dashed vertical line divides the figure into two areas, left during observation, right after turning off the camera. Figure (d) represent the trajectory estimation in pixels. the dashed vertical line divides the figure into two areas, left during observation, right after turning off the camera.

As you can see in Figure 6.9, maximum error value is less than 25 pixels. For a screen about 500mm from the camera, this error value = 4.9mm. It is to be noticed that error value gets bigger with time because lack of information obout object position.

To make the proposed method more realistic, we applied the second experiment on a real scence ehich is a wall of a building (see Figure 6.10). We suppose that a object moved on a small area of that wall 12*5 meters, with a pattern as in experiment 2.

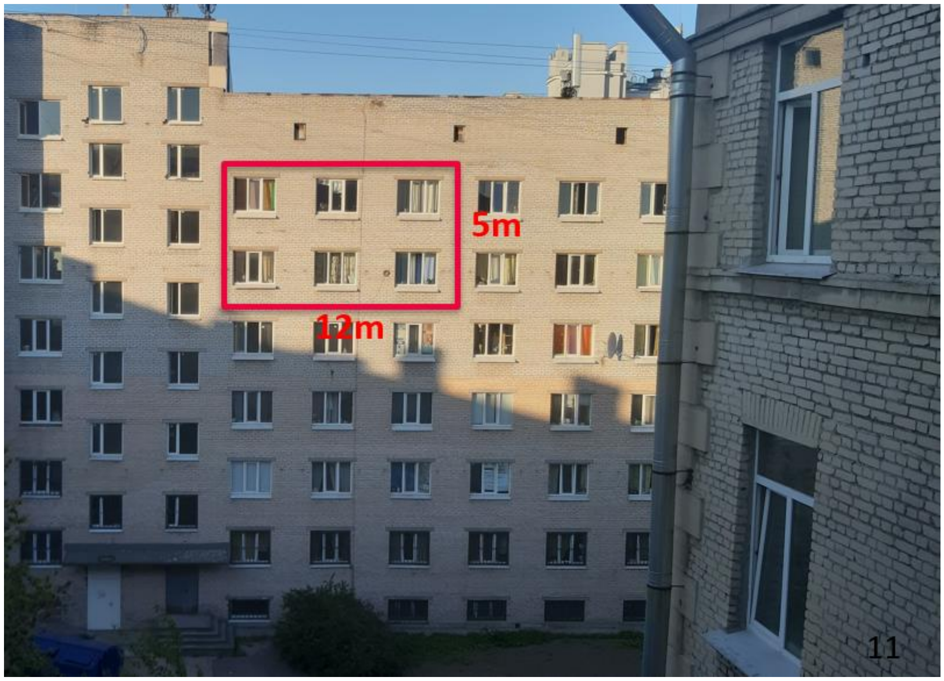

Figure 6.10 – building façade captured by a camera 25 meters far from it.

By enlarging the small considered area, we get a more detailed view see Figure 6.11.

Figure 6.11 – enlarging the area in Figure 6.10.

So by projection of the detedted and estimated patterns on that area we get an estimation error of 30cm (see).

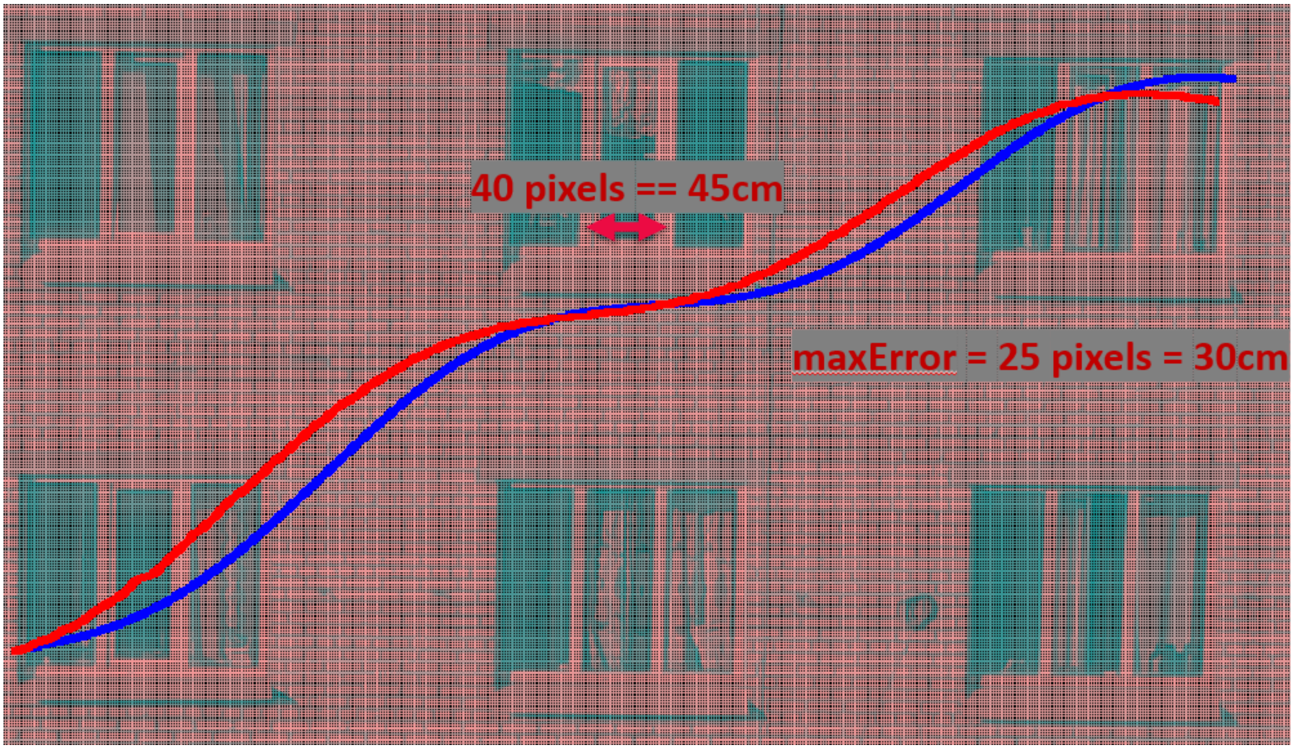

Figure 6.12 – projection of the detedted and estimated patterns on the considered area

Conclusion

In this research, a new approach to track objects and predict future trajectories has been presented and discussed. The efficiency of this method was tested in the laboratory on a target that moves with periodic moving patterns. The results and tracking errors (Figure 6.7, **Error! Reference source not found.** and **Error! Reference source not found.**) show that the tracking error is relatively small when using this system in a laboratory or big hall. Also, results promise that this research could be enhanced by future work (especially in the field of discrete signal processing) and used for tracking objects in unmanned aircraft.

Pros and Cons of the proposed approach:

Pros

1. the implemented system is useful for tracking objects in the conditions of a room, laboratory or big hall especially for those objects that move with a predefined trajectory and don't change their trajectory depending on the environment.
2. with additional work on filtering the detected pattern, more work on discrete signal processing this system could be improved to track objects in environments with wider space, like tracking cars on road or drones in the sky.

Cons

1. This system faces a difficulty when we implement the whole system to work totally automatically. The reason for that is we need in advance to know how many frequencies involved in the detected pattern. This issue could be solved with Fourier Transform. But additional signal processing needed to be done before using Fourier Transform.
2. Noisy detected pattern leads to bad Estimation with DREM. In each experiment in chapter 6, we generate the pattern for the object center. So, we can’t guarantee that the pixel in the object center is detected as the center in each frame. This leads to a noisy signal. And the larger the size of object with respect to frame size, the larger the noise become. For example, detected pattern in

Figure 6.2 is for a Target with radius three times bigger than Target in **Error! Reference source not found.**. It is clear that the later pattern is smoother (less noisy) than the first one.

Future work

It is very good step to test this system on a dataset contains trajectory of some drones and or aircraft. To do this we need in advance to study more about discrete signal processing. For example, to get an accurate estimation with DREM, we need to add some conditions on d_i in equations 5.7 and 5.8.

Bibliography

[1] Erokhin, Denis & Feldman, Alexander & Korepanov, S. (2017). DETECTION AND TRACKING OF MOVING OBJECTS WITH REAL-TIME ONBOARD VISION SYSTEM. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. XLII-2/W4. 67-71. 10.5194/isprs-archives-XLII-2-W4-67-2017.

[2] Sang, Ik & Nam, Sang-Hyun & Yu, Hyun & Roberts, Rodney & Moon, Seungbin. (2006). Conveyor visual tracking using robot vision. FCRAR.

[3] Tsarouchi, Panagiota & Michalos, George & Makris, S. & Chryssolouris, George. (2013). Vision System for Robotic Handling of Randomly Placed Objects. Procedia CIRP. 9. 61-66. 10.1016/j.procir.2013.06.169.

[4] F. Husain, A. Colomé, B. Dellen, G. Alenyà and C. Torras, "Realtime tracking and grasping of a moving object from range video," 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, 2014, pp. 2617-2622, doi: 10.1109/ICRA.2014.6907234.

[5] Y. Shen, "Location Prediction for Tracking Moving Objects," 2009 WRI Global Congress on Intelligent Systems, Xiamen, 2009, pp. 362-366, doi: 10.1109/GCIS.2009.192.

[6] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei and S. Savarese, "Social LSTM: Human Trajectory Prediction in Crowded Spaces," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 961-971, doi: 10.1109/CVPR.2016.110.

[7] A. Wiest, M. Höffken, U. Kreßel and K. Dietmayer, "Probabilistic trajectory prediction with Gaussian mixture models," 2012 IEEE Intelligent Vehicles Symposium, Alcala de Henares, 2012, pp. 141-146, doi: 10.1109/IVS.2012.6232277.

[8] Borisov, Oleg & Gromov, Vladislav & Vedyakov, Alexey & Pyrkin, Anton & Bobtsov, Alexey & Aranovskiy, Stanislav. (2017). Adaptive Tracking of a Multi-Sinusoidal Signal with DREM-Based Parameters Estimation. IFAC-PapersOnLine. 50. 4282-4287. 10.1016/j.ifacol.2017.08.835.

[9] Doganis, Fivos (Yerres, FR). Dynamical camera calibration. 20200005491A1, Dassault Systemes (Velizy Villacoublay, FR), January 2020. http://www.freepatentsonline.com/y2020/0005491.html.

[10] Z. Zhang, "A flexible new technique for camera calibration," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 11, pp. 1330-1334, Nov. 2000, doi: 10.1109/34.888718.

[11] Bobtsov, Alexey & Pyrkin, Anton. (2012). Cancelation of unknown multiharmonic disturbance for nonlinear plant with input delay. International Journal of Adaptive Control and Signal Processing. 26. 10.1002/acs.1283.

[12] Locomotec GmbH. KUKA youBot User Manual. Locomotec. December 6, 2012.

[13] S. Aranovskiy, A. Bobtsov, R. Ortega and A. Pyrkin, "Performance Enhancement of Parameter Estimators via Dynamic Regressor Extension and Mixing*," in IEEE Transactions on Automatic Control, vol. 62, no. 7, pp. 3546-3550, July 2017, doi: 10.1109/TAC.2016.2614889.

[14] Spong, M.W. & Vidyasagar, M. (1989). Robot Dynamics and Control.

[15] Колюбин С.А., Динамика робототехнических систем. Учебное пособие. — СПб.: Университет ИТМО, 2017. — 117 с.

[16] OpenCV.The OpenCV Reference Manual, 2.4.13.7 edition, April 2014. https://docs.opencv.org/2.4/modules/refman.html

[17] E. Marchand, F. Spindler and F. Chaumette, "ViSP for visual servoing: a generic software platform with a wide class of robot control skills," in IEEE Robotics & Automation Magazine, vol. 12, no. 4, pp. 40-52, Dec. 2005, doi: 10.1109/MRA.2005.1577023.

[18] Gabriel Nagy. Ordinary Differential Equations, Mathematics Department, Michigan State University, East Lansing, MI, 48824. September 22, 2019.

[19] Solem, Jan Erick: Programming computer vision with Python. Beijing; Cambridge; Sebastopol [etc.] : O'Reilly, 2012. - ISBN 9781449316549 1449316549.

[20] Snyder, W., & Qi, H. (2017). Fundamentals of Computer Vision. Cambridge: Cambridge University Press. doi:10.1017/9781316882641.

[21] Dickmanns, Ernst. (2007). Dynamic Vision for Perception and Control of Motion. 10.1007/978-1-84628-638-4.

[22] Iliukhin, V.N. & Mitkovskii, K.B. & Bizyanova, D.A. & Akopyan, A.A.. (2017). The Modeling of Inverse Kinematics for 5 DOF Manipulator. Procedia Engineering. 176. 498-505. 10.1016/j.proeng.2017.02.349.

[23] Herman Bruyninckx. Robot Kinematics and Dynamics. Katholieke Universiteit Leuven, Department of Mechanical Engineering, Leuven, Belgium. August 21, 2010.

[24] Hibbeler, R C, and Kai B. Yap. Engineering Mechanics. , 2017. Print.

[25] R. Marino and P. Tomei, "Frequency estimation of periodic signals," 2014 European Control Conference (ECC), Strasbourg, 2014, pp. 7-12, doi: 10.1109/ECC.2014.6862212.

[26] M. Mojiri and A. R. Bakhshai, "An adaptive notch filter for frequency estimation of a periodic signal," in IEEE Transactions on Automatic Control, vol. 49, no. 2, pp. 314-318, Feb. 2004, doi: 10.1109/TAC.2003.821414.

[27] S. Peleg and B. Porat, "Linear FM signal parameter estimation from discrete-time observations," in IEEE Transactions on Aerospace and Electronic Systems, vol. 27, no. 4, pp. 607-616, July 1991, doi: 10.1109/7.85033.

[28] Xu, L., Ding, F. Iterative Parameter Estimation for Signal Models Based on Measured Data. Circuits Syst Signal Process 37, 3046–3069 (2018).