REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE YAHIA FARS DE MEDEA



Faculté de Technologie

Mémoire de fin d'études de Master

Filière : Génie électrique

Spécialité : instrumentation

BALL ON PLATE BALANCING SYSTEM

Proposé et Présenté par:

BENTEBIB NOUR EDDINE

NABI IMAD EDDINE

Dirigé par:

Mr. KHARKHAR ABDELKARIM

Promotion 2020

AKNOWLEDJEMENT

Thanks be to God the almighty and merciful to us endow health, will, patience, during all these long years.

We also express our sincere thanks to our promoter Dr. kherkhar Abdelkarim who guided this job; his enthusiasm, his many advices, his patience and his effective help for the realization of this thesis.

Our sincere thanks go to all teachers and M.EULDJI riadh who helped us to finish this work.

We would also like to express our thanks to jury members For having done us the honor and the pleasure to accept and judge our work.

Finally, we thank all the people who have contributed from near or far, to the smooth running of our work and our studies

DEDICACE

I thank ALLAH the Almighty for giving me the strength and courage to accomplish this memory

I have the honor to dedicate this modest work to:

To my dear parents who supported and encouraged me throughout the period of my studies and to whom I wish them a long and happy life, may God protect them and give me the strength so that I can give them back a little bit of their blessing and their goodness.

To my dear brothers, my sister and her little family.

To all the NABI family and the ZERKANI family

To my dear partner Noureddine

To my dear friends

To all my Instrumentation promotion

And finally, to all those who i love and who loves me

IMAD

DEDICACE

I thank ALLAH the Almighty for giving me the strength and courage to accomplish this memory

I have the honor to dedicate this modest work to:

To my dear parents who supported and encouraged me throughout the period of my studies and to whom I wish them a long and happy life, may God protect them and give me the strength so that I can give them back a little bit of their blessing and their goodness.

To my dear brothers, my friends SALAH, ABDESSALEM AND ABDERAHIM

To all the BENTEBIB family

To my dear partner IMAD

To RIADH EULDJI

To all my Instrumentation promotion

And finally, to all those who i love and who loves me

NOUREDDINE

Abstract:

Ball on plate balancing system is one of the common and simplest control systems as it can be used to apply the control concepts studied in the system modeling course such as the settling time, overshooting, rise time and steady state error. It demonstrates the feedback closed loop system and how input affects the behavior of the output. Moreover, it shows the effect of nonlinearities and how it deviates the system from the theoretical model and how can we deal with this nonlinearities by introducing extra parameters like integration factor which forces the steady state error to be zero.

Résumé:

Le système d'équilibrage boule sur plaque est l'un des systèmes de contrôle les plus courants et les plus simples, car il peut être utilisé pour appliquer les concepts de contrôle étudiés dans le cours de modélisation de système tels que le temps de stabilisation, le dépassement, le temps de montée et l'erreur en régime permanent. Il montre le système de rétroaction en boucle fermée et comment l'entrée affecte le comportement de la sortie. De plus, il montre l'effet des non-linéarités et comment il dévie le système du modèle théorique et comment pouvons-nous gérer ces non-linéarités en introduisant des paramètres supplémentaires comme le facteur d'intégration qui force l'erreur en régime permanent à être nulle.

ملخص

يعد نظام موازنة الكرة على اللوحة أحد أنظمة التحكم الشائعة وأبسطها حيث يمكن استخدامه لتطبيق مفاهيم التحكم التي تمت دراستها في دورة نمذجة النظام مثل وقت الاستقرار والتجاوز ووقت الارتفاع وخطأ الحالة المستقرة. يوضح نظام التغذية المرتدة الحلقة المغلقة وكيف تؤثر المدخلات على سلوك المخرجات. علاوة على ذلك ، فإنه يوضح تأثير اللاخطية وكيف ينحرف النظام عن النموذج النظري وكيف يمكننا التعامل مع هذه اللاخطية من خلال تقديم معلمات إضافية مثل عامل التكامل الذي يفرض خطأ الحالة المستقرة ليكون صفرًا.

Summary

General Introduction: 1
1. Introduction: 4
2. Linear systems: 4
3. Nonlinear systems: 4
4. Controllability and Observability: 5
4.1 Stability*:* 6
4.2 Trajectory and motion tracking: 8
4.3 Path following: 9
4.4 Linearization of nonlinear system 9
5. Control system: 10
5.1 open loop control system: 11
5.1.1 Practical Examples of Open Loop Control System 11
5.1.2 Advantages of open loop control system 11
5.1.3 Desadvantages of open loop control system 12
5.2 Closed loop control system: 12
5.2.1Practical Examples of Closed Loop Control System: 12
5.2.2 Advantages of Closed loop control system: 13
5.2.3 Disadvantages of Closed loop control system: 13
6. Control theory : 14
6.1 Proportional, Integral & Derivative Controller: 14
6.2 Proportional controller : 16
6.2.1 Advantages of proportional controller: 16
6.2.2 Disadvantages of proportional controller: 16
6.3 Proportional and integral controller: 17
6.4 Proportional and derivative controller: 17
6.5 Proportional plus Integral plus Derivative Controller (PID Controller): 18
6.6 Tuning of proportional integral derivative controller: 19
6.7 General Guidelines for Designing a PID Controller: 20
1. Introduction: 22
2. Ball and plate system: 22
3. Arduino: 25
3.1 The Arduino board: 27
3.2 Arduino board components: 28
3.3 The Arduino Family: 30

- **Arduino Uno (R3):** 31
- **RedBoard :** 32
- **Arduino Mega (R3) :** 32
- **Arduino Leonardo** 33

3.4 The use of Arduino: 34

4. Servomotors: 34

4.1 Servo motor control: 36

5. Touch screen: 37

5.1 Components and working of touch screen: 38

5.2 Resistive touch screen: 39

5.3 Advantages of Resistive Touch Screens: 41

6. Conclusion: 41

1.Introduction: 43

2. Mathematical Model: 43

2.1 Approximations and assumptions: 43

2.2 Derivation: 44

2.3 Servo Model: 46

3. Simulink Model: 46

3.1 Ball Model subsystem: 46

3.2 Feedback loop 1 (Servo PID): 46

3.3 Feedback loop 2 (Distance PID): 48

4. Position detection: 49

5. Results: 50

5.1 Simulink results: 50

6. Future research: 52

7. Conclusion: 53

1. Inroduction: 55

2. Virtual Reality Modelling Language : 55

3. Simulink® 3D Animation : 57

3.1 Graphical User Interface (GUI) : 58

4. Simulation models of mechatronics systems in Matlab/Simulink: 58

5. Conclusion: 60

GENERAL CONCLUSION: 61

Futur work : 61

LIST OF FIGURES

Figure I.1: Root Locus Plot of System Indicating Region of Stability……..6

Figure I.2 Open loop control system…….11

Figure I.3 Closed loop control system……..12

Figure I.4 PID Controller system……..15

Figure II.1 System wiring…….....22

Figure II.2 Block diagram of ball and plate system……..24

Figure II.4 The Arduino Board……..26

Figure II.5 Types of Arduino Board …….27

Figure II.6 The Arduino board component……28

Figure I.7 Arduino Uno board……...30

Figure II.13 Types of Servomotors…….....35

Figure II.14 Servomotor control……36

Figure II.15 operation when using the touch screen panel……38

Figure III.1 Block Ball Model……...……...46

Figure III.2 Servo feedback loop model……...…….47

Figure III.3 Distance feedback loop model…….....….48

Figure IV.1 Hierarchical Tree Structure of Nodes ……...……55

Figure IV.3: VRML Rotation Angles …….....56

Figure IV.4 Virtual model of the Ball and Plate in control scheme…….....59

Figure IV.5 Virtual Reality (VR) Model of the Ball and Plate System……...…….59

Figure IV.6 Virtual model of the Ball and Plate and its structure……...…….60

LIST OF TABLES

Table I.1: Effects of increasing a parameter independently……..19

Table III.1 list of parameters……..44

LIST OF ABBREVIATIONS

VRML …… Virtual Reality Modeling Language

ITAE …… Integral of Time Multiplied by Absolute Error

ISE …… Integral of Square Error

3-D …… Three Dimension (al)

B&P …… Ball and Plate

DOF …… Degree of Freedom

SISO …… Single Input Single Output

AAIT …… Addis Ababa Institute of Technology

FSM(C) …… Fuzzy Sliding Mode (Control)

VSC(C) …… Variable Structure (Control)

SMC …… Sliding Mode Control

VR …… Virtual Reality

AD …… Analog to Digital

CCD …… Charge Coupled Device

ISO …… International Organization for Standardization

IEC …… International Electrotechnical Commission

RMS …… Root Mean Square

ave …… Average

GA …… Genetic Algorithm

PID …… proportional–integral–derivative

PD …… proportional–derivative

ADRC …… Active Disturbance Rejection Control

ESO …… Extended State Observer

General Introduction:

Though the extent and range of applications might vary, various educational models are either directly developed or acquired in several universities across the world in an effort to complement the theoretical teaching and learning process. In particular, educational models are available to the research community at large both at undergraduate and post graduate study levels to facilitate the testing of control algorithms in the laboratories in technological institutions. In a similar manner these models are being exploited to a great extent in the development of engineering devices for industrial or household applications as preliminary benchmarking tools before the commencement of major mass productions. The increasing tendency in the preferential usage of educational models in both institutional and commercial research areas has its main purpose which springs from the ability to comfortably test real systems on simplified but fairly close replica of real systems in the laboratory platforms. In many circumstances, the direct experimentation of control algorithms on real systems in the laboratory is either impossible or costly or perhaps might even prove fatal. For instance, in order to design and test a control strategy for a state of the art humanoid robot for industrial applications, it will be unwise and uneconomical to experiment with the robot itself; however, an inverted pendulum, which mimics a walking human being in a less complex manner, will be the right choice for preliminary research as the costs and risks associated with the experimentation on the inverted pendulum are acceptable. Likewise, the Ball and Plate (B&P) system- which is the controlled object in this thesis, is gaining popularity and is arousing the attention of several researchers in a range of disciplines. The B&P systems could be described in simple terms as a system constituting of rectangular (circular) flat plate fixed movably at its center by a pivot so that the plate could execute free inclinations in two orthogonal directions while the (steel) ball keeps rolling freely on the plate. Hence, the objective would be to command the ball to be placed at a given position on the plate or to instruct the Ball to follow a given trajectory without the ball tipping off the edges of the plate. From simple intuition, it is impossible to keep the ball at a fixed position(s) on the plate without the ball showing the tendency to roll off the edges even in the absence of disturbances like wind gust, meaning- the problem is open loop unstable. Unlike other educational models, the B&P system has several

interesting features for the study of unstable and nonlinear systems. Firstly, models like Magnetic Levitation, Inverted Pendulum and Ball and Beam are all one-dimensional with degree of freedoms (DOF): one (1), two (2), and two (2) respectively. The B&P system has four (4) DOF i.e. with two independent motions of the ball about the plane of the plate and another two motions of the combined ball and plate inclinations (rotations) about the fixed pivot at the center of the plate. Secondly, the B&P system offers an opportunity to study trajectory tracking problems in addition to the stability problems that is the only subject that could be studied in the remaining educational models mentioned above. Thirdly, research in the control laboratory at our institution is limited to SISO systems; the multivariable property of the B&P system could be utilized to extend the scope of the research. So the advantage is clearly evident. The B&P system was firstly established in Rockwell laboratory of Czechoslovakia University in the mid-1990s [1]. Important improvements in the mechanical design by some universities in America and Asia were introduced in the years to follow. Nowadays commercial educational model manufactures like HUMUSOFT, GUNT Hamburg, Googol Technology and TecQuipment develop an integrated system with all the associated sensors and drives.

CHAPTER I:

THEORETICAL BACKGROUND AND LITERATURE REVIEW

1. Introduction:

Ball on plate balancing system is one of the common and simplest control systems as it can be used to apply the control concepts studied in the system modeling course such as the settling time, overshooting, rise time and steady state error. It demonstrates the feedback closed loop system and how input affects the behavior of the output. Moreover, it shows the effect of nonlinearities and how it deviates the system from the theoretical model and how can we deal with this nonlinearities by introducing extra parameters like integration factor which forces the steady state error to be zero.

In this chapter we present literature review which comprises an overview of fundamental concept and the review of similar works. In the review of fundamental concepts, most of the pertinent works and the fundamental theories that have been used for the success of this research are reviewed, after which similar works are reviewed.

2. Linear systems:

A system is called linear if the principle of superposition applies. The principle of superposition states that the response produced by the simultaneous application of two different forcing functions is the sum of the two individual responses. Hence, for the linear system, the response to several inputs can be calculated by treating one input at a time and adding the results. It is this principle that allows one to build up complicated solutions to the linear differential equation from simple solutions. In an experimental investigation of a dynamic system if cause and effect are proportional, thus implying that the principle of superposition holds, then the system can be considered linear.

3. Nonlinear systems:

Most physical systems are inherently nonlinear in nature. A common engineering practice in analyzing nonlinear system is to linearize it about a nominal operating point and analyses the resulting linear model. A nonlinear system would be defined in

the context of a linear system. Linear systems must satisfy two principles: superposition and homogeneity.

The principle of superposition states that for two different inputs, x and , in the domain of the function f.

$$f(x+y) = f(x) + f(y) \quad \text{(I.1)}$$

The principle of homogeneity states that for a given input, , in the domain of the function f, and for any real number h .

$$f(kx) = kf(x) \quad \text{(I.2)}$$

Any function that does not satisfy superposition and homogeneity principle is nonlinear.It is worth noting that there is no unifying characteristic of nonlinear systems, except for not satisfying the two above-mentioned principles.

In order to ease the analysis of the ball and plate system, which is a nonlinear system, it is expected that the system should be linearized about its operating point.

The operating point is that point at which all state and input variables are initialized to zero.

4. Controllability and Observability:

A system is said to be controllable if a control vector $u(t)$ exist that will transfer the system from any initial state $x(t_0)$ to some final state $x(t)$ in a finite time interval.

A system is said to be completely state controllable if the Kalman controllability matrix:

$$M_c = [B : AB : \cdots : A^{n-1}B] \quad \text{(I.3)}$$

Is of full rank, that is it contains n linearly independent column or row vectors.

A system is said to be observable if at time t_0 , the system state $x(t_0)$ can be exactly Determined from observation of the output $y(t)$over a finite time interval .

A system is said to be completely state observable if the kalman observability matrix :

$$M_0 = \left[C^T : A^T C^T : (A^T)^{n-'} C^T\right] \quad \text{(I.4)}$$

Is of full rank, that is, it contains n linearly independent rows or columns vectors.

The concept of controllability and observability from the control system perspective plays an important role in verifying whether a control solution exists or not from the Beginning. This is important as most physical systems are controllable (stabilizable) and Observable (detectable), whereas the corresponding mathematical models may not possess the property of controllability and observability. However, if the model is found not to be controllable and observable, it implies the mathematical model is not the true representation of the physical system, and as such may not yield an optimal control solution.

4.1 Stability*:*

The condition for stability is that all the poles must lie in the leit hand of the s-plane (LHP), as shown in Figure I.1.

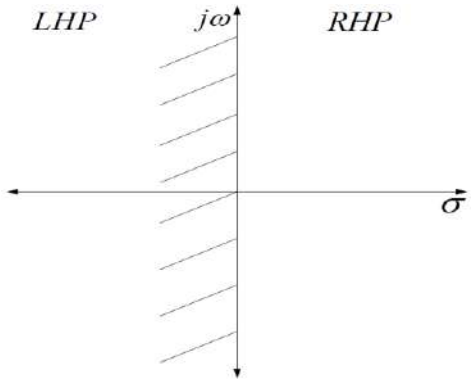


Figure I.1: Root Locus Plot of System Indicating Region of Stability

If a system is unstable (that is its poles exists on the right hand of the s-plane (RHP)) but controllable, it means the poles can be transferred to the left hand plane, but if the system is also observable, the designer has the added flexibility of placing the poles at any location in the left hand plane .

The requirements for the dynamics of a closed loop linear time varying system is stability, because the feedback control of the closed loop system tends to overcorrect

errors that was caused by the oscillations of constant or changing amplitude. Stability can be defined in two categories :

- Ability of a system to return to equilibrium after an arbitrary displacement away from the equilibrium
- The ability ofthe system to produce a bounded output from a bounded input.

The basic stability criterion is directly determined by location of the system poles or eigenvalues i.e. the roots ofthe characteristic equation of the system.

The ability of a system to return to equilibrium to a forced system is related as:

$$\dot{x} = Ax + Bu \tag{I.5}$$

The solution of the first order $\dot{x}(t)$ is given as (Friedland, 2012):

$$\dot{x}(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}\,Bu(\tau)d\tau \tag{I.6}$$

Over an initial state condition $x(0)$, Where:

- e^{At} is the state-transition matrix.
- $x(t)$ is the state solution which captures all the state variables.

The properties of state transition matrix as related to stability are as follows:

- If the real parts of all the characteristics roots are strictly negative (i.e. not zero or positive), then e^{At} tends asymptotically to zero. Hence, no matter how large the initial state $x(0)$ is, $x(t) \to 0$ as $t \to \infty$. The system is said to be asymptotically stable.
- If any characteristic root has a strictly positive real part, the state transition matrix will have at least one term which will tend to infinity as $t \to \infty$. In this case, it is possible to find some initial state which will cause $x(t)$ to become infinite. This system is said to be unstable.
- If all the characteristic roots have non-positive real part, and has any characteristic roots that has a zero real part which is repetitive, then, there is at least one or more term that will tend to infinity as $t \to \infty$. This system is said to be unstable.

If the model under study has been determined to be controllable and observable, then, there is a need to explore the various types of controllers that will be used in effecting control and stabilization of the system in order to achieve a specified performance specification.

4.2 Trajectory and motion tracking:

Stabilization is difficult and challenging, but tracking is practically important. Tracking Of nonlinear systems is usually specified by a control objective to track a trajectory with the exponential convergence rate in order to guarantee performance and robustness.

Trajectory tracking of nonlinear system is achieved based on two kinds of models. The first model considers the velocities of the system as control inputs and uses the system kinematic model. It addresses tracking at a kinematic control level and ignores the system dynamics. The second model uses the system dynamics, where control forces and torques as well as velocities can be control inputs. For a nonlinear system, a dynamic control model is usually integrated with a kinematic control model. A control system developed in such a way has two-level architecture. The lower control level pirates within a kinematic model of the nonlinear system to stabilize its motion to a desired trajectory. The upper control level uses a dynamic model and stabilizes feedback obtained on the lower control level. This allows steering commands, which are velocities at the kinematic level, to be converted to forces and torques. The control system developed within this two-level architecture assumes a perfect knowledge of the system dynamics in most cases.

In classical nonlinear control, tracking usually means trajectory tracking, and it is defined as follows :

Given a nonlinear system whose dynamics is described by :

$$\dot{q}(t) = f(q(t), u(t), t) \quad \text{(I.7)}$$

Where f is a nonlinear vector function, q is a state variable vector, and u is the input vector. With:

$$q(t_0) = q_0 \quad \text{(I.8)}$$

$$z(t) = g(q(t), u(t), t) \quad \text{(I.9)}$$

Where $z(t)$ is the system output.

And a desired output $z_d(t)$, find a control law u such that, starting from anywhere in a bounded region.

The tracking errors:

$$e = z(t) - z_d(t) \quad \text{(I.10)}$$

goes to zero, and the whole system remains bounded .

4.3 Path following:

In the tracking problem, a system is to track a desired motion, typically a trajectory, which means that it is expected to reach a sequence of desired positions in specified time instants. It means that the reference is time parameterized. In path following, a system is required to reach and follow some path without any temporal specification. The basic property of this control task is that a controller acts on a system orientation to drive it to the specified path.

The path following control problem can be formulated then as the stabilization to zero of a scalar path error function, which is a distance to the path. Path following can be treated either as a sub-problem of trajectory tracking or stabilization. In trajectory tracking, the separation of the geometric and timing information along the trajectory is needed, while in the stabilization, the controlled output, which is a distance to the path, is to be made zero[2].

4.4 Linearization of nonlinear system

In control engineering a normal operation of the system may be around an equilibrium point, and the signals may be considered small signals around the equilibrium. (It should be pointed out that there are many exceptions to such a case.) However, if the system operates around an equilibrium point and if the signals involved are small signals, then it is possible to approximate the nonlinear system by a linear system. Such a linear system is equivalent to the nonlinear system considered within a limited operating range. Such a linearized model (linear, time-invariant model) is very important in control engineering. The linearization procedure to be presented in the following is based on the expansion of nonlinear function into a Taylor series about the operating point and the retention of only the linear term. Because we neglect higher order terms of the Taylor series expansion, these neglected

terms must be small enough; that is, the variables deviate only slightly from the operating condition. (Otherwise, the result will be inaccurate)[3].

5. Control system:

A control system is a system of devices or set of devices, that manages, commands, directs or regulates the behavior of other devices or systems to achieve desired results. In other words, the definition of a control system can be simplified as a system, which controls other systems. As the human civilization is being modernized day by day the demand for automation is increasing accordingly. Automation highly requires control of devices.

In recent years, control systems have played a central role in the development and advancement of modern technology and civilization. Practically every aspect of our day-to-day life is affected less or more by some control system. A bathroom toilet tank, a refrigerator, an air conditioner, a geezer, an automatic iron, an automobile all are control system. These systems are also used in industrial processes for more output. We find control systems in the quality control of products, weapons system, transportation systems, power system, space technology, robotics and many more. The principles of control theory are applicable to both engineering and non-engineering field.

The main feature of a control system is that there should be a clear mathematical relationship between input and output of the system. When the relation between input and output of the system can be represented by a linear proportionality, the system is called a linear control system. Again when the relationship between input and output cannot be represented by single linear proportionality, rather the input and output are related by some non-linear relation, the system is referred to as a non-linear control system.

Hence, there are two main types of control system. They are as follow :

- Open loop control system
- Closed loop control system

5.1 open loop control system:

A control system in which the control action is totally independent of output of the system then it is called open loop control system. A manual control system is also an open loop control system. The figure below shows a control system block diagram of an open loop control system in which process output is totally independent of the controller action.

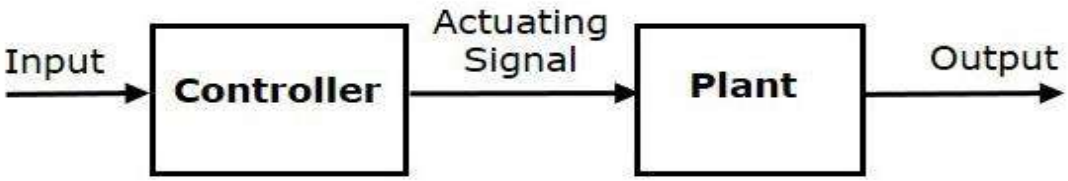


Figure I.2 Open loop control system

5.1.1 Practical Examples of Open Loop Control System

- Electric Hand Drier – Hot air (output) comes out as long as you keep your hand under the machine, irrespective of how much your hand is dried.
- Automatic Washing Machine – This machine runs according to the pre-set time irrespective of washing is completed or not.
- Bread Toaster – This machine runs as per adjusted time irrespective of toasting is completed or not.
- Automatic Tea/Coffee Maker – These machines also function for pre adjusted time only.
- Timer Based Clothes Drier – This machine dries wet clothes for pre-adjusted time, it does not matter how much the clothes are dried.
- Light Switch – Lamps glow whenever light switch is on irrespective of light is required or not.
- Volume on Stereo System – Volume is adjusted manually irrespective of output volume level.

5.1.2 Advantages of open loop control system

- Simple in construction and design.
- Economical.
- Easy to maintain.
- Generally stable.
- Convenient to use as output is difficult to measure.

5.1.3 Desadvantages of open loop control system

- They are inaccurate.
- They are unreliable.
- Any change in output cannot be corrected automatically.

5.2 Closed loop control system:

Control system in which the output has an effect on the input quantity in such a manner that the input quantity will adjust itself based on the output generated is called closed loop control system**.** Open loop control system can be converted in to closed loop control system by providing a feedback. This feedback automatically makes the suitable changes in the output due to external disturbance. In this way closed loop control system is called automatic control system. Figure below shows the block diagram of closed loop control system in which feedback is taken from output and fed in to input**.**

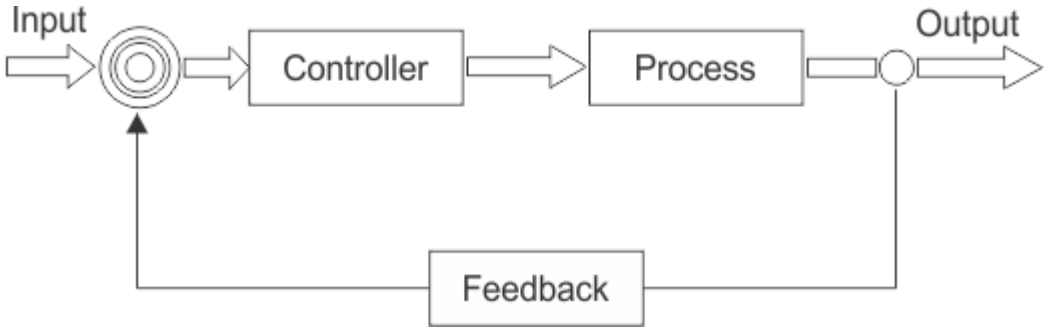


Figure I.3 Closed loop control system

5.2.1Practical Examples of Closed Loop Control System:

- Automatic Electric Iron – Heating elements are controlled by output temperature of the iron.

- Servo Voltage Stabilizer – Voltage controller operates depending upon output voltage of the system.
- Water Level Controller – Input water is controlled by water level of the reservoir.
- Missile Launched and Auto Tracked by Radar – The direction of missile is controlled by comparing the target and position of the missile.
- An Air Conditioner – An air conditioner functions depending upon the temperature of the room.
- Cooling System in Car – It operates depending upon the temperature which it controls.

5.2.2 Advantages of Closed loop control system:

- Closed loop control systems are more accurate even in the presence of non-linearity.
- Highly accurate as any error arising is corrected due to presence of feedback signal.
- Bandwidth range is large.
- Facilitates automation.
- The sensitivity of system may be made small to make system more stable.
- This system is less affected by noise.

5.2.3 Disadvantages of Closed loop control system:

- They are costlier.
- They are complicated to design.
- Required more maintenance.
- Feedback leads to oscillatory response.
- Overall gain is reduced due to presence of feedback.
- Stability is the major problem and more care is needed to design a stable closed loop system [4].

6. Control theory :

A controller is a mechanism that seeks to minimize the difference between the actual value of a system (i.e. the process variable) and the desired value of the system (i.e. the setpoint).

Controllers are a fundamental part of control engineering and used in all complex control systems.

Before we talk about various controllers in detail, it is very essential to know the uses of controllers in the theory of control systems. The important uses of the controllers include:

- Controllers improve the steady-state accuracy by decreasing the steady state error.
- As the steady-state accuracy improves, the stability also improves.
- Controllers also help in reducing the unwanted offsets produced by the system.
- Controllers can control the maximum overshoot of the system.
- Controllers can help in reducing the noise signals produced by the system.
- Controllers can help to speed up the slow response of an over damped system.[5]

As aforementioned, a PID controller will be utilized to control the system. In this section its deferent parts will be explained in detail.

6.1 Proportional, Integral & Derivative Controller:

A proportional-integral-derivative controller (PID controller) is a control loop feedback mechanism (controller) widely used in industrial control systems. A PID controller calculates an error value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error by adjusting the process through use of a manipulated variable. The PID controller algorithm involves three separate constant parameters, and is accordingly sometimes called three-term control: the proportional, the integral and derivative values, denoted P, I, and D. Simply put, these values can be interpreted in terms of time: P depends on the present error, I on the accumulation of past errors, and D is a prediction of future

errors, based on current rate of change. The weighted sum of these three actions is used to adjust the process via a control element such as the position of a control valve, a damper, or the power supplied to a heating element. In the absence of knowledge of the underlying process, a PID controller has historically been considered to be the most useful controller. By tuning the three parameters in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint, and the degree of system oscillation. Note that the use of the PID algorithm for control does not guarantee optimal control of the system or system stability. Some applications may require using only one or two actions to provide the appropriate system control. This is achieved by setting the other parameters to zero. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions. PI controllers are fairly common, since derivative action is sensitive to measurement noise, whereas the absence of an integral term may prevent the system from reaching its target value due to the control action.

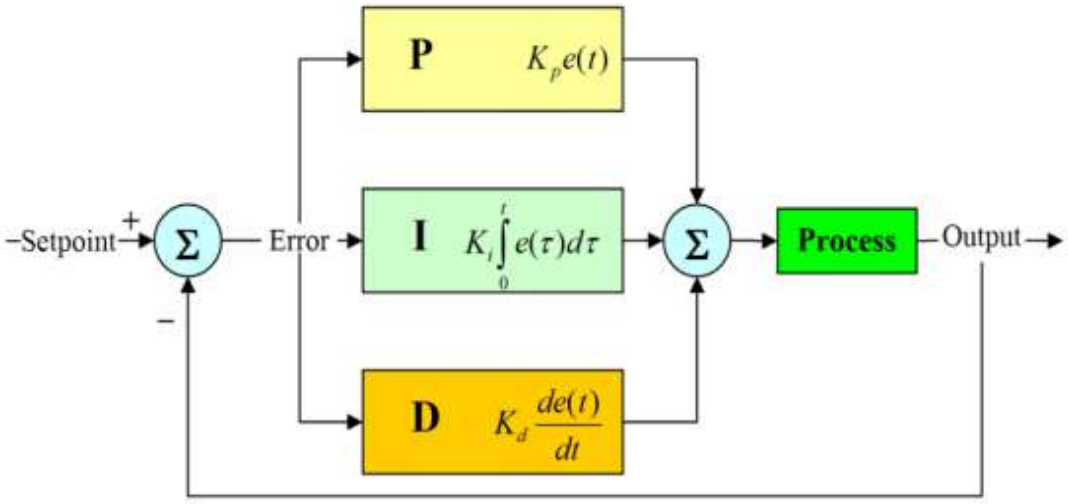


Figure I.4 PID Controller system

6.2 Proportional controller :

All controllers have a specific use case to which they are best suited. We cannot just insert any type of controller at any system and expect a good result – there are certain conditions that must be fulfilled. For a proportional controller**,** there are two conditions and these are written below:

- The deviation should not be large; i.e. there should not be a large deviation between the input and output.
- The deviation should not be sudden.

Now we are in a condition to discuss proportional controllers, as the name suggests in a proportional controller the output (also called the actuating signal) is is directly proportional to the error signal**.** Now let us analyze the proportional controller mathematically. As we know in proportional controller output is directly proportional to the error signal, writing this mathematically we have,

$$P=Kp^{*}(e(t)) \qquad (I.11)$$

Where K_p is proportional constant also known as controller gain.

It is recommended that K_p should be kept greater than unity. If the value of K_p is greater than unity (>1), then it will amplify the error signal and thus the amplified error signal can be detected easily.

6.2.1 Advantages of proportional controller:

- The proportional controller helps in reducing the steady-state error, thus makes the system more stable.
- The slow response of the over damped system can be made faster with the help of these controllers.

6.2.2 Disadvantages of proportional controller:

Now there are some serious disadvantages of these controllers and these are written as follows:

- Due to the presence of these controllers, we get some offsets in the system.
- Proportional controllers also increase the maximum overshoot of the system.

6.3 Proportional and integral controller:

As the name suggests it is a combination of proportional and an integral controller the output (also called the actuating signal) is equal to the summation of proportional and integral of the error signal. Now let us analyze proportional and integral controller mathematically. As we know in a proportional and integral controller output is directly proportional to the summation of proportional of error and integration of the error signal, writing this mathematically we have,

$$PI = Kp(e(t)) + Ki \int e(t)\, dt \qquad (I.2)$$

Where, K_i and k_p proportional constant and integral constant respectively.
Advantages and disadvantages are combinations of the advantages and disadvantages of proportional and integral controllers.
Through the PI controller, we are adding one pole at origin and one zero somewhere away from the origin (in the left-hand side of complex plane). As the pole is at the origin, its effect will be more, hence PI controller may reduce the stability; but its main advantage is that it reduces steady-state error drastically, due for this reason it is one of the most widely used controllers.

6.4 Proportional and derivative controller:

As the name suggests it is a combination of proportional and a derivative controller the output (also called the actuating signal) is equals to the summation of proportional and derivative of the error signal. Now let us analyze proportional and derivative controller mathematically. As we know in a proportional and derivative controller output is directly proportional to the summation of proportional of error and differentiation of the error signal, writing this mathematically we have,

$$PD = Kp\,(e(t)) + Kd\,(d(e(t))/dt) \qquad (I.3)$$

Where, K_d and K_p proportional constant and derivative constant respectively.
Advantages and disadvantages are combinations of advantages and disadvantages of proportional and derivative controllers.

Readers should note that adding ‘zero’ at the proper location in the open-loop transfer function improves stability, while the addition of pole in the open-loop transfer function may reduce the stability. The words “at proper location” in the above sentence are very important & it is called designing of the control system (i.e. both zero & pole should be added at proper points in the complex plane to get the desired result. Inserting the PD controller is like the addition of zero in open-loop transfer function [G(s)H(s)].

The transfer function of the PD controller is K+Tds or Td(s+K/Td), so we have added one zero at -K/Td. By controlling the value of ‘K’ or ‘Td’, the position of the ‘zero’ can be decided. If ‘zero’ is very far away from the imaginary axis, its influence will decrease, if ‘zero’ is on the imaginary axis (or very close to the imaginary axis) it will also be not accepted (root locus generally starts from ‘poles’ & terminates at ‘zero’, Designer’s aim is generally such that root locus should not go towards the imaginary axis, due to this reason ‘zero’ very near to imaginary axis is also not acceptable, hence a moderate position of ‘zero’ should be kept).

Generally, it is said, PD controller improves transient performance and the PI controller improves the steady-state performance of a control system.

6.5 Proportional plus Integral plus Derivative Controller (PID Controller):

A PID controller is generally used in industrial control applications to regulate temperature, flow, pressure, speed, and other process variables.

The transfer function of the PID Controller can be found as:

$$Tds + K + \frac{Ki}{s} \text{ or } \frac{Tds^2+Ks+Ki}{s}$$

It can be observed that one pole at origin is fixed, remaining parameters T_d, K, and Ki decide the position of two zeros. In this case, we can keep two complex zeros or two real zeros as per the requirement, hence PID controller can provide better tuning. In the olden days, the PI controller was one of the best choice of control engineers, because designing (tuning of parameters) of the PID controller was a little difficult,

but nowadays, due to the development of software designing of PID controllers have become an easy task [6].

The effect of increasing each of the controller parameters Kp, Ki and Kd can be summarized as:

Table I.1: Effects of increasing a parameter independently

Parameter	Rise time	Overshoot	Settling time	Steady-state
Kp	Decrease	Increase	Small change	Decrease
Ki	Decrease	Increase	Increase	Eliminate
Kd	No change	Decrease	Decrease	Small effect

6.6 Tuning of proportional integral derivative controller:

All general methods for control design can be applied to PID control. A number of special methods that are tailor-made for PID control have also been developed, these methods are often called tuning methods. Irrespective of the method used it is essential to always consider the key elements of control, load disturbances, sensor noise, process uncertainty and reference signals. The most well-known tuning methods are those developed by Ziegler and Nichols. They have had a major influence on the practice of PID control for more than half a century. The methods are based on characterization of process dynamics by a few parameters and simple equations for the controller parameters. It is surprising that the methods are so widely referenced because they give moderately good tuning only in restricted situations. Plausible explanations may be the simplicity of the methods and the fact that they can be used for simple student exercises in basic control courses.[6]

There are many methods of PID tuning such as:

- Manual tuning.
- Ziegler-Nichols.
- Tyreusluyben.
- Cohen-coon.

- Software tools.

6.7 General Guidelines for Designing a PID Controller:

When you are designing a PID controller for a given system, general guidelines to obtain the desired response are as follows:

- Obtain the transient response of closed-loop transfer function and determine what needs to be improved.
- Insert the proportional controller, Design the value of 'K' through Routh-Hurwitz or suitable software.
- Add an integral part to reduce steady-state error.
- Add the derivative part to increase damping (damping should be between 0.6-0.9). The derivative part will reduce overshoots & transient time.
- Sisotool, available in MATLAB can also be used for proper tuning and to obtain a desired overall response.
- Please note, above steps of tuning of parameters (designing of a control system) are general guidelines. There are no fixed steps for designing controllers.[7]

7. conclusion:

The system can be treated as two different systems operating simultaneously. Hence, similar but independent controllers can be used for controlling each coordinate of the ball motion . Based on the linear model, a preliminary controller is designed with the scheme of a „loop within a loop". The first step involves the design of an inner loop where the encoder feedback is sent to the dc motors to achieve a servo position control. The inner loop is then placed in an outer loop that controls the ball position. The next step in the control system design is to obtain a controller for the outer loop, based on the transfer function between ball position and the corresponding plate angle.

The overall control scheme can be explained as follows. While the controller in the outer loop computes the angle by which the plate should move to balance the ball, the inner loop controller actually moves the plate by that angle. Ideally, the inner loop should do this instantaneously, which is not possible in reality.

CHAPTER II:

BALL AND PLATE SYSTEM

1. Introduction:

This chapter contains details about electrical components that used in the project. And the electrical hardware that used in the project. The system consists of a plate, a touch screen and servo motors. the Touch screen is placed over the plate and the pate is moving by the servo motor which is connected to digital pin, and the servo Motor Y is connected to digtal pin.4.1 show wiring diagram of the project.

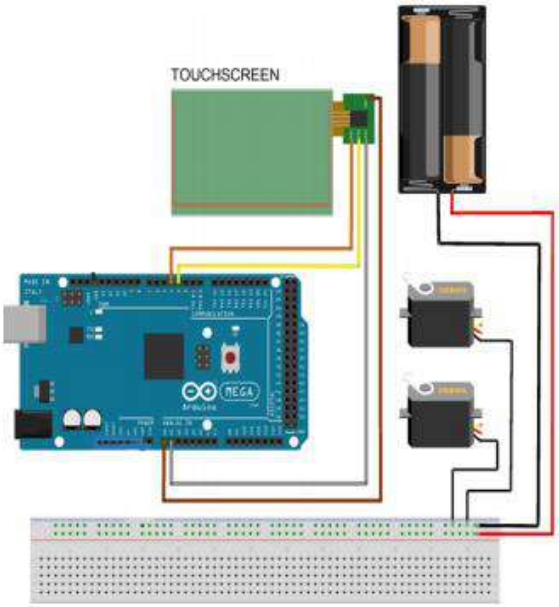


Figure II.1 System wiring

2. Ball and plate system:

The Ball and Plate (B&P) system is unstable, nonlinear, multivariable, and under-actuated. Owing to these properties together with the opportunity to experiment both the stabilization of the ball to static position and trajectory tracking problems simultaneously on the same platform, its popularity is surging in recent publications and among practicing researchers in a range of disciplines. Thus, owing to these features the B&P model is an excellent alternative that could be used in the study of stabilization and trajectory tracking researches in such disciplines as control theory, robotics, optimal control, nonlinear control, multivariable control, etc

The Ball and Plate model is an extension of the classical Ball and Beam system. It consists of a rectangular (circular) flat plate fixed movably at its center by a pivot. The control target is the plate which could rotate about 2 mutually perpendicular axes,

with the ultimate aim of balancing a free rotating ball. The control problem of this system is to control the position of the Ball and Plate system for both static positions and desired path tracking. The inclination of the plate can be manipulated in two perpendicular directions, so that the tilting of the plate will make the ball move on the plate. Static position tracking control of the Ball and Plate system is to hold the ball at a specific position on the plate. Dynamic position tracking control demands the ball to follow a given position reference. Static and dynamic position tracking control could be studied simultaneously on the Ball and Plate system. The block diagram of the Ball and Plate system is shown in Figure (II.14). Briefly the control procedure is described as follows:

- The position sensor obtains the position of the ball on the plate. In most applications, the sensor is either a camera or a touch-screen device.

- Position data is feedback to the control system where the reference angle is calculated either for position command or trajectory tracking problems. If a camera is used as the sensor, the position data is obtained by image processing techniques that identify the target object from its background and determine its position from the analog frame of picture.

- The rotation of the plate along the two mutually orthogonal directions are facilitated by two actuators and thus balancing the ball to the command position or along the desired trajectory. The actuators often used for this application are either DC-motors or stepper motors and in only one instance a pneumatic device is reported.

The block-diagram representation of the control of the B&P system shown in Figure (II.14) is for one of the two orthogonal directions.

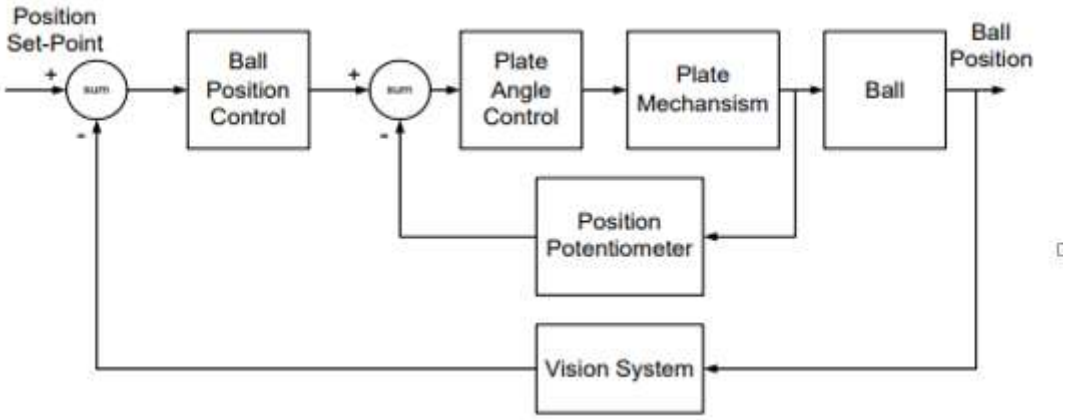


Figure II.2 Block diagram of ball and plate system

The Ball and Plate (B&P) system is unstable, nonlinear, multivariable, and under-actuated. Owing to these properties together with the opportunity to experiment both the stabilization of the ball to static position and trajectory tracking problems simultaneously on the same platform, its popularity is surging in recent publications and among practicing researchers in a range of disciplines. Thus, owing to these features the B&P model is an excellent alternative that could be used in the study of stabilization and trajectory tracking researches in such disciplines as control theory, robotics, optimal control, nonlinear control, multivariable control, etc

The ball-and-plate system is designed with the purpose to have key functionalities in mind. The first is automatic stabilization of the ball's location. The system should also provide feedback on where the ball is located on the sensor, and be able to have variable predestination of the desired ball location. So the ball's desired position can be changed at any time from user input [8].

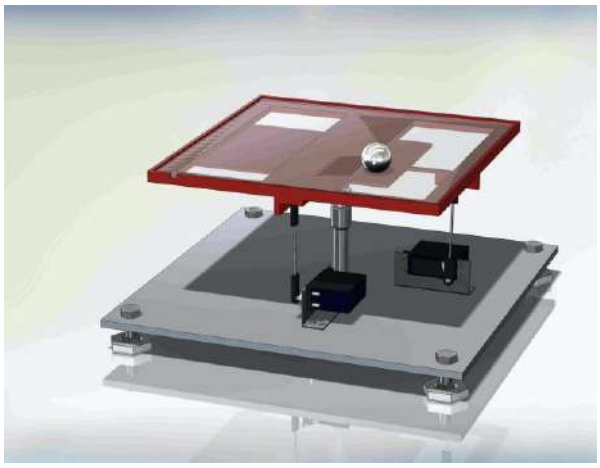


Figure II.3 Ball and Plate model

3. Arduino:

Arduino is a small microcontroller board with a USB plug to connect to your computer and a number of connection sockets that can be wired up to external electronics, such as motors, relays, light sensors, laser diodes, loudspeakers, microphones, etc. They can either be powered through the USB connection from the computer or from a 9V battery. They can be controlled from the computer or programmed by the computer and then disconnected and allowed to work independently. At this point, the Evil Genius might be wondering which top secret government organization they need to break into in order to acquire one. Well, disappointingly, no evil deeds at all are required to obtain one of these devices. The Evil Genius needs to go no further than their favorite online auction site or search engine. Since the Arduino is an open-source hardware design, anyone is free to take the designs and create their own clones of the Arduino and sell them, so the market for the boards is competitive. An official Arduino costs about $30, and a clone often less than $20. The name “Arduino” is reserved by the original makers. However, clone Arduino designs often have the letters “duino” on the end of their name, for example, Freeduino or DFRduino. The software for programming your Arduino is easy to use and also freely available for Windows, Mac, and LINUX computers at no cost.

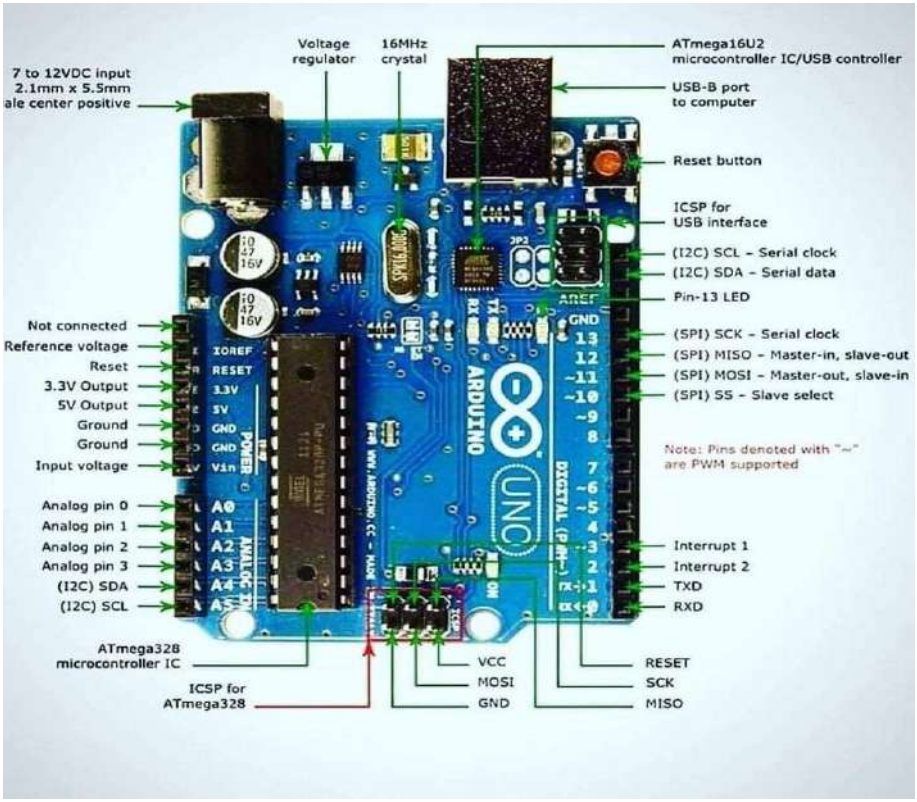


Figure II.4 The Arduino Board

You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing .Over the years Arduino has been the brain of thousands of projects, from everyday objects to complex scientific instruments. A worldwide community of makers - students, hobbyists, artists, programmers, and professionals - has gathered around this open-source platform, their contributions have added up to an incredible amount of accessible knowledge that can be of great help to novices and experts alike. Arduino was born at the Ivrea Interaction Design Institute as an easy tool for fast prototyping, aimed at students without a background in electronics and programming. As soon as it reached a wider community, the Arduino board started changing to adapt to new needs and challenges, differentiating its offer from simple 8-bit boards to products for IOT applications, wearable, 3D printing, and embedded environments. All Arduino boards are completely open-source, empowering users to build them independently and 18 eventually adapt them to their particular needs. The software, too, is opensource, and it is growing through the contributions of user's worldwide [9].

3.1 The Arduino board:

Arduino board is an open-source platform used to make electronics projects. It consists of both a microcontroller and a part of the software or Integrated Development Environment (IDE) that runs on our PC, used to write & upload computer code to the physical board.The platform of an Arduino has become very famous with designers or students just starting out with electronics, and for an excellent cause.

Unlike most earlier programmable circuit boards, the Arduino does not require a separate part of hardware in order to program a new code onto the board you can just use a USB cable. As well, the Arduino IDE uses a basic version of C++, making it simpler to learn the program. At last, Arduino board offers a typical form factor that breaks out the functions of the microcontroller into a more available package

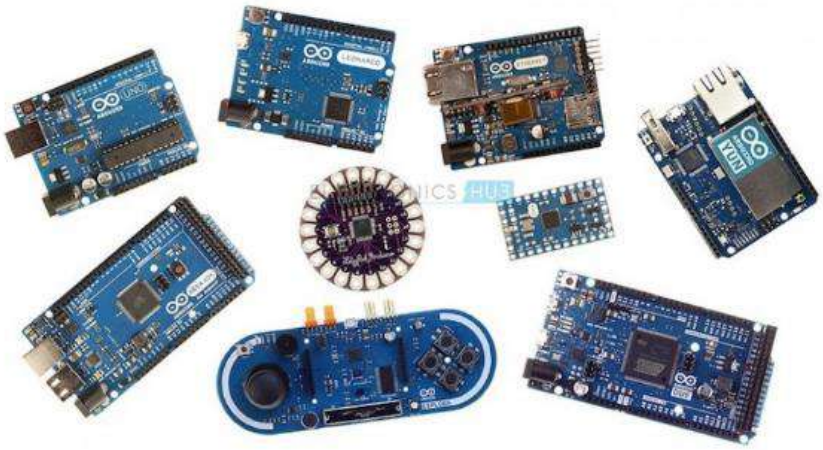


Figure II.5 Types of Arduino Board

Arduino board has been used for making different engineering projects and different applications. The Arduino software is very simple to use for beginners, yet flexible adequate for advanced users. It runs windows, Linux and Mac. Teachers and students in the schools utilize it to design low cost scientific instruments to verify the principles of physics and chemistry. There are numerous other microcontroller platforms obtainable for physical computing. The Netmedia's BX-24, Parallax Basic Stamp, MIT's Handyboard, Phidget and many others present related functionality.

Arduino also makes simpler the working process of microcontroller, but it gives some advantages over other systems for teachers, students and beginners.[9]

- Inexpensive
- Cross-platform
- Simple, clear programming environment
- Open source and extensible software
- Open source and extensible hardware[10]

3.2 Arduino board components:

There are many varieties of Arduino boards that can be used for different purposes. Some boards look a bit different from the one below, but most Arduinos have the majority of these components in common:

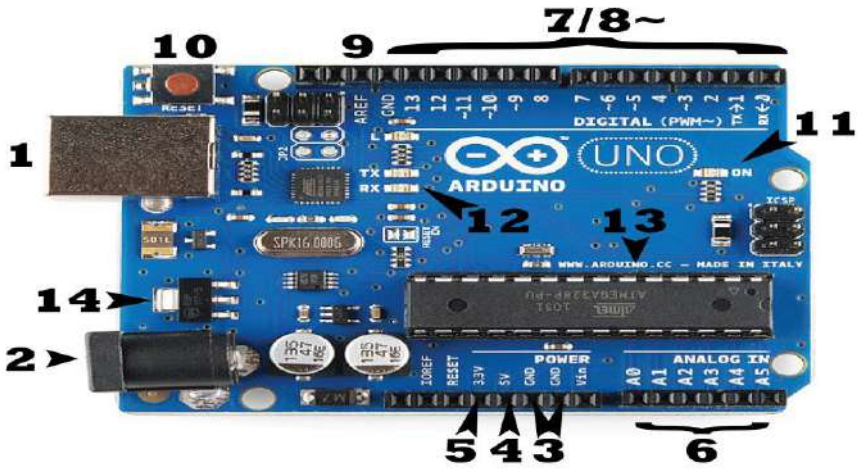


Figure II.6 The Arduino board component

- Power (USB / Barrel Jack) : Every Arduino board needs a way to be connected to a power source. The Arduino UNO can be powered from a USB cable coming from your computer or a wall power supply that is terminated in a barrel jack. In the picture above the USB connection is labeled **(1)** and the barrel jack is labeled **(2)**.

The USB connection is also how you will load code onto your Arduino board. More on how to program with Arduino can be found in our Installing and Programming Arduino tutorial.

- Pins (5V, 3.3V, GND, Analog, Digital, PWM, AREF): The pins on our Arduino are the places where you connect wires to construct a circuit (probably in conjuction with a breadboard and some wire. They usually have black plastic 'headers' that allow you to just plug a wire right into the board. The Arduino has several different kinds of pins, each of which is labeled on the board and used for different functions.

- GND (3)**:** Short for 'Ground'. There are several GND pins on the Arduino, any of which can be used to ground your circuit.
- 5V (4) & 3.3V (5)**:** As you might guess, the 5V pin supplies 5 volts of power, and the 3.3V pin supplies 3.3 volts of power. Most of the simple components used with the Arduino run happily off of 5 or 3.3 volts.
- Analog (6)**:** The area of pins under the 'Analog In' label (A0 through A5 on the UNO) are Analog In pins. These pins can read the signal from an analog sensor (like a temperature sensor) and convert it into a digital value that we can read.
- Digital (7)**:** Across from the analog pins are the digital pins (0 through 13 on the UNO). These pins can be used for both digital input (like telling if a button is pushed) and digital output (like powering an LED).
- PWM (8)**:** You may have noticed the tilde (~) next to some of the digital pins (3, 5, 6, 9, 10, and 11 on the UNO). These pins act as normal digital pins, but can also be used for something called Pulse-Width Modulation (PWM). We have a tutorial on PWM, but for now, think of these pins as being able to simulate analog output (like fading an LED in and out).
- AREF (9)**:** Stands for Analog Reference. Most of the time you can leave this pin alone. It is sometimes used to set an external reference voltage (between 0 and 5 Volts) as the upper limit for the analog input pins

-Reset Button: Just like the original Nintendo, the Arduino has a reset button **(10)**. Pushing it will temporarily connect the reset pin to ground and restart any code that is loaded on the Arduino. This can be very useful if your code doesn't repeat, but you want to test it multiple times. Unlike the original Nintendo however, blowing on the Arduino doesn't usually fix any problems.

- Power Led Indicator: Just beneath and to the right of the word "UNO" on your circuit board, there's a tiny LED next to the word 'ON' **(11)**. This LED should light up whenever you plug your Arduino into a power source. If this light doesn't turn on, there's a good chance something is wrong. Time to re-check our circuit!

- TX RX LEDs: TX is short for transmit, RX is short for receive. These markings appear quite a bit in electronics to indicate the pins responsible for serial communication. In our case, there are two places on the Arduino UNO where TX and RX appear -- once by digital pins 0 and 1, and a second time next to the TX and RX indicator LEDs (12)**.** These LEDs will give us some nice visual indications whenever our Arduino is receiving or transmitting data (like when we're loading a new program onto the board).

- Main IC**:** The black thing with all the metal legs is an IC, or Integrated Circuit (13)**.** Think of it as the brains of our Arduino. The main IC on the Arduino is slightly different from board type to board type, but is usually from the ATmega line of IC's from the ATMEL Company. This can be important, as you may need to know the IC type (along with your board type) before loading up a new program from the Arduino software. This information can usually be found in writing on the top side of the IC. If you want to know more about the difference between various IC's, reading the datasheets is often a good idea.

- Voltage Regulator: The voltage regulator **(14)** is not actually something you can (or should) interact with on the Arduino. But it is potentially useful to know that it is there and what it's for. The voltage regulator does exactly what it says -- it controls the amount of voltage that is let into the Arduino board. Think of it as a kind of gatekeeper; it will turn away an extra voltage that might harm the circuit. Of course, it has its limits, so don't hook up your Arduino to anything greater than 20 volts. [11]

3.3 The Arduino Family:

Arduino makes several different boards, each with different capabilities. In addition, part of being open source hardware means that others can modify and produce derivatives of Arduino boards that provide even more form factors and

functionality. If you're not sure which one is right for your project, check this guide for some helpful hints. Here are a few options that are well-suited to someone new to the world of Arduino:

- **Arduino Uno (R3):**

The Uno is a great choice for your first Arduino. It's got everything you need to get started, and nothing you don't. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a USB connection, a power jack, a reset button and more. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started

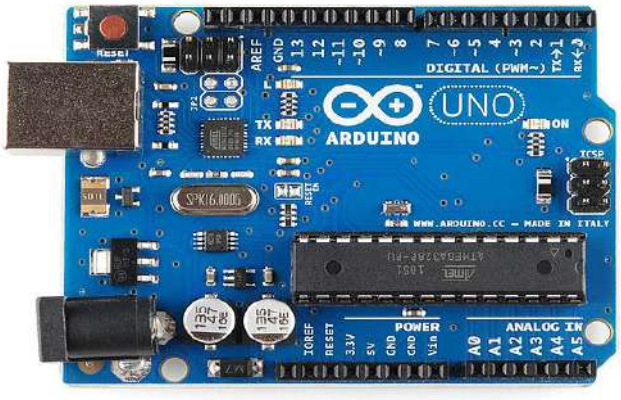


Figure I.7 Arduino Uno board

- **LilyPad Arduino :**

This is LilyPad Arduino main board, LilyPad is a wearable e-textile technology developed by Leah Buechley and cooperatively designed by Leah and SparkFun. Each LilyPad was creatively designed with large connecting pads and a flat back to allow them to be sewn into clothing with conductive thread. The LilyPad also has its own family of input, output, power, and sensor boards that are also built specifically for e-textiles. They're even washable!

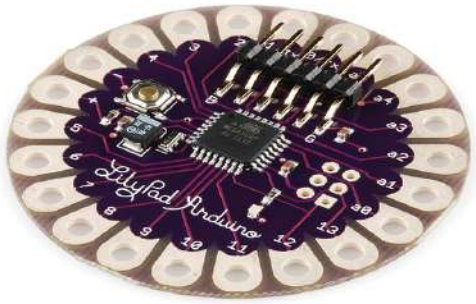


Figure II.8 Lily Arduino board

- **RedBoard :**

At Spark Fun we use many Arduino and we're always looking for the simplest, most stable one. Each board is a bit different and no one board has everything we want -- so we decided to make our own version that combines all our favorite features. The RedBoard can be programmed over a USB Mini-B cable using the Arduino IDE. It'll work on Windows 8 without having to change your security settings (we used signed drivers, unlike the UNO). It's more stable due to the USB/FTDI chip we used, plus it's completely flat on the back, making it easier to embed in your projects. Just plug in the board, select "Arduino UNO" from the board menu and you're ready to upload code. You can power the RedBoard over USB or through the barrel jack. The on-board power regulator can handle anything from 7 to 15VDC.

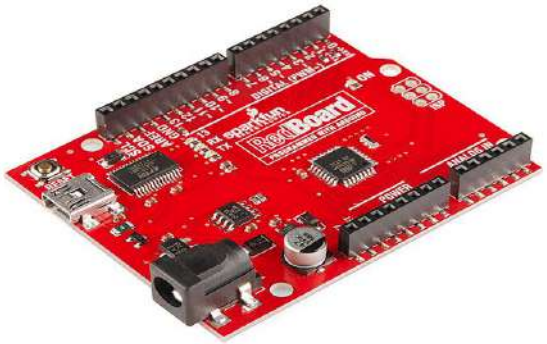


Figure II.9 The Red Board

- **Arduino Mega (R3) :**

The Arduino Mega is like the UNO's big brother. It has lots (54) of digital input/output pins (14 can be used as PWM outputs), 16 analog inputs, a USB

connection, a power jack, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The large number of pins make this board very handy for projects that require a bunch of digital inputs or outputs (like lots of LEDs or buttons).

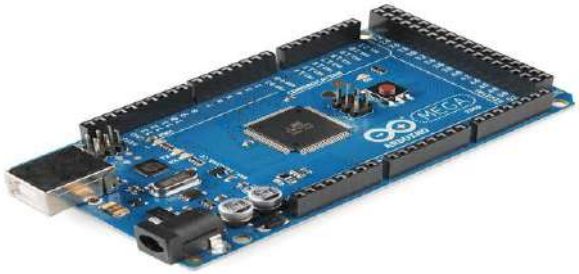


Figure II.10 Arduino Mega Board

- **Arduino Leonardo**

The Leonardo is Arduino's first development board to use one microcontroller with built-in USB. This means that it can be cheaper and simpler. Also, because the board is handling USB directly, code libraries are available which allow the board to emulate a computer keyboard, mouse, and more! [12]



Figure II.11 Lionardo Arduino Board

In addition, Arduino boards also have other types of components called Shields or backpacks. It is a kind of boards that connect to the main board to add an infinity of functions, such as GPS, real-time clocks, radio connectivity, LCD touch screens, development boards, and many more elements. There are even stores with specialized sections on such items.



Figure II.12 Sheilds Board

3.4 The use of Arduino:

Arduino can be used to create standalone elements, connecting to devices, and interacting with both hardware and software. It helps us both to control an element, for example a motor that raises or lowers a blind based on the existing light is a room, thanks to a light sensor connected to the Arduino, or to read the information from a source, such as It can be a keyboard or a web page, and convert the information into an action such as turning on a light and writing what you type on a display.

With Arduino it is possible to automate anything to make autonomous agents (if you want we can call them Robots). To control lights and devices, or anything else you can think of, you can go for an Arduino-based solution, especially in developments of devices connected to the Internet.

Arduino is a technology that has a fast learning curve with basic knowledge of programming and electronics, which allows developing projects in the field of Smart Cities, the Internet of Things, wearable devices, health, leisure, education, robotics, etc …[13].

4. Servomotors:

A servomotor is a linear actuator or rotary actuator that allows for precise control of linear or angular position, acceleration, and velocity. It consists of a motor coupled to a sensor for position feedback. It also requires a relatively sophisticated controller, often a dedicated module designed specifically for use with servomotors.

There are some special types of applications of an electric motor where the rotation of the motor is required for just a certain angle. For these applications, we require some special types of motor with some special arrangement which makes the motor rotate a

certain angle for a given electrical input (signal). For this purpose, servo motor comes into the picture.

The servo motor is usually a simple DC motor controlled for specific angular rotation with the help of additional servomechanism (a typical closed-loop feedback control system). Nowadays, servo systems are used widely in industrial applications.

Servo motor applications are also commonly seen in remote-controlled toy cars for controlling the direction of motion, and it is also very widely used as the motor which moves the tray of a CD or DVD player. Besides these, there are hundreds of servo motor applications we see in our daily life.

The main reason behind using a servo is that it provides angular precision, i.e. it will only rotate as much we want and then stop and wait for the next signal to take further action. The servo motor is unlike a standard electric motor which starts turning as when we apply power to it, and the rotation continues until we switch off the power. We cannot control the rotational progress of electrical motor, but we can only control the speed of rotation and can turn it ON and OFF. Small servo motors are included many beginner Arduino starter kits, as they are easy to operate as part of a small electronics projects.



Figure II.13 Types of Servomotors

If the motor as a controlled device, associated with servomechanism is DC motor, then it is commonly known as a DC Servo Motor. If AC operates the controlled motor, it is known as a AC Servo Motor.[14].

4.1 Servo motor control:

For understanding servo motor control let us consider an example of servomotor that we have given a signal to rotate by an angle of 45o and then stop and wait for further instruction. The shaft of the DC motor is coupled with another shaft called output shaft, with the help of gear assembly. This gear assembly is used to step down the high rpm of the motor's shaft to low rpm at the output shaft of the servo system.

The voltage adjusting knob of a potentiometer is so arranged with the output shaft by means of another gear assembly, that during rotation of the shaft, the knob also rotates and creates an varying electrical potential according to the potentiometer.
This signal i.e. electrical potential is increased with the angular movement of potentiometer knob along with the system shaft from 0o to 45o. This electrical potential or voltage is taken to the error detector feedback amplifier along with the input reference commends i.e. input signal voltage

As the angle of rotation of the shaft increases from 0^o to 45^o the voltage from potentiometer increases. At 45^o this voltage reaches to a value which is equal to the given input command voltage to the system. As at this position of the shaft, there is no difference between the signal voltage coming from the potentiometer and reference input voltage (command signal) to the system, the output voltage of the amplifier becomes zero.

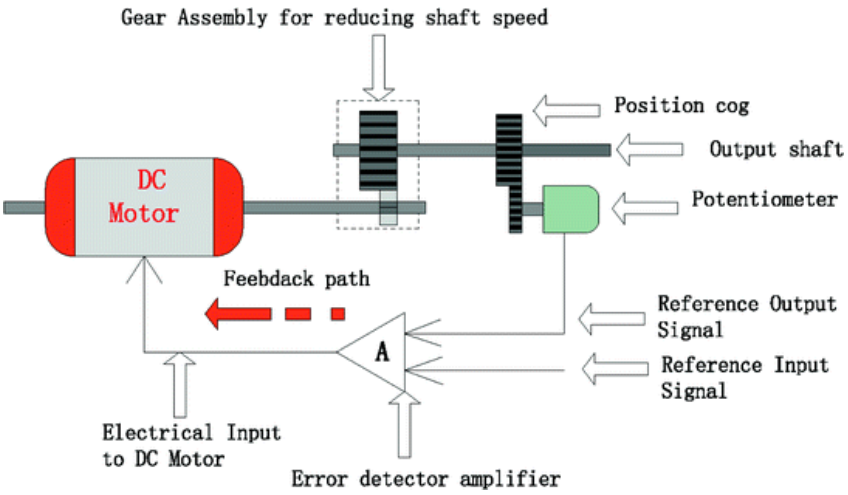


Figure II.14 Servomotor control

As per the picture given above the output electrical voltage signal of the amplifier, acts as input voltage of the DC motor. Hence, the motor will stop rotating after the shaft rotates by 45o. The motor will be at this rest position until another command is given to the system for further movement of the shaft in the desired direction. From this example we can understand the most basic servo motor theory and how servo motor control is achieved.

From this basic working principle of servo motor it can be concluded. The shaft of the servo is connected to a potentiometer. The circuitry inside the servo, to which the potentiometer is connected, knows the position of the servo. The current position will be compared with the desired position continuously with the help of an Error Detection Amplifier. If a mismatch is found, then an error signal is provided at the output of the error amplifier and the shaft will rotate to go the exact location required. Once the desired location is reached, it stops and waits.

Continuous rotation servo motors are actually a modified version of what the servos are actually meant to do, that is, control the shaft position. The 360o rotation servos are actually made by changing certain mechanical connections inside the servo. However, a certain manufacturer like parallax sells these servos as well. With the continuous rotation servo you can only control the direction and speed of the servo, but not the position. [15]

5. Touch screen:

Touch screen technology is the direct manipulation type of gesture based technology. Direct manipulation is the ability to manipulate the digital world inside a screen. A Touch screen is an electronic visual display capable of detecting and locating a touch over its display area. This is generally referred to as touching the display of the device with a finger or hand. This technology most widely used in computers, user interactive machines, smartphones, tablets, etc to replace most functions of the mouse and keyboard.

Touch screen technology has been around for a number of years but advanced touch screen technology has come on in leaps and bounds recently. Companies are including this technology into more of their products. The three most common touch screen technologies include resistive, capacitive, and SAW (surface acoustic wave). Most low-

end touch screen devices contain on a standard printed circuit plug-in board and are used on SPI protocol. The system has two parts, namely; hardware and software. The hardware architecture consists of a stand-alone embedded system using an 8-bit microcontroller, several types of interface, and driver circuits. The system software driver is developed using an interactive C programming language.[16].

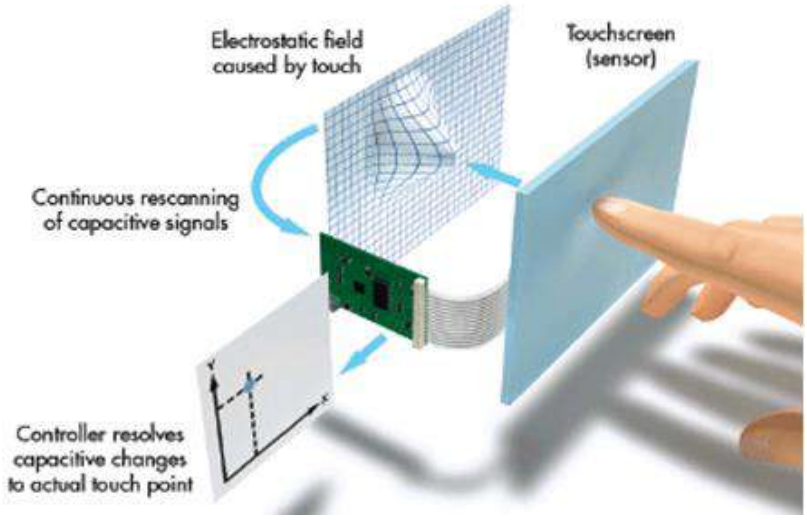


Figure II.15 operation when using the touch screen panel

5.1 Components and working of touch screen:

A basic touch screen is having a touch sensor, a controller, and a software driver as three main components. The touch screen is needed to be combined with a display and a PC to make a touch screen system.

- **Touch sensor:** The sensor generally has an electrical current or signal going through it and touching the screen causes a change in the signal. This change is used to determine the location of the touch of the screen.
- **Controller:** A controller will be connected between the touch sensor and PC. It takes information from the sensor and translates it for the understanding of PC. The controller determines what type of connection is needed.
- **Software driver:** It allows computers and touch screens to work together. It tells OS how to interact with the touch event information that is sent from the controller.[17]

5.2 Resistive touch screen:

A resistive touch screen panel is coated with a thin metallic electrically conductive and resistive layer that causes a change in the electrical current which is registered as a touch event and sent to the controller for processing. Resistive 4- and 5-wire touch sensors are the most popular and most common touch-screen technologies with about 75% market share, mainly due to their low costs and simple interface electronics. The high volume of these screens requires a low-cost reliable interface, often with a low-power element. This can be provided through a range of analog features combined with low-power modes for portable, battery-powered applications. Usually a resistive touch-screen consists of at least three layers Figure 2.16. A flexible membrane made from poly ethylene terephthalate (PET) film is 26 suspended over a rigid substrate made from glass or acrylic, and both surfaces are coated with a transparent conductive film, usually Indium tin oxide (ITO). The conductive ITO layers are kept apart by an insulting spacer along the edges, and by spacer dots on the inner surface of the two ITO layers. In this way, there will be no electrical connection unless pressure is applied to the top sheet (PET film).



Figure II.16 The resistive touch panel

4-wire touch-screens use a single pair of electrodes called bus bars on each ITO layer. The four wires reference X+ (left), X- (right), Y+ (top) and Y- (bottom) locations, which is key to interfacing the sensors. In 5-wire touch-screens the four wires connect to the electrodes, these are referred to as UL (Upper Left), UR (Upper Right), LL

(Lower Left), and LR (Lower Right). The fifth wire is used for sensing the electrode voltage is referred to as the “sense” wire, and is embedded in the top sheet. 5-wire touch-screens is less critical than for 4-wire touch-screens. The pressure points are measured by applying a voltage to the electrode pair in the resistive surface so that a uniform voltage gradient appears across the surface. A second ITO layer is necessary to do a high-resistance voltage measurement.

The most popular resistive architectures use 4-wire or 5-wire configurations of the circuit. The drawing below represents the concept schematic of resistive touch:

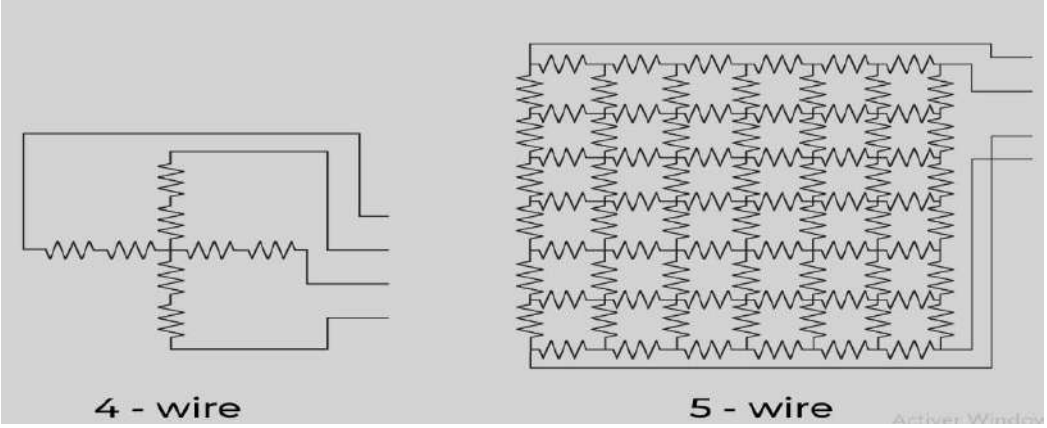


Figure II.17: 4-wire and 5-wire resistive touchscreen

The circuits determine the location in two coordinate-pair dimensions. As the majority of application are based on 4-wire technology, we will focus on this solution to describe the idea.

On the touch screen, the voltage is applied in the Y direction. When touching, two layers presses together creating the contact – in this exact position the voltage can be read from one of the X electrodes. At that point the voltage divider is created, so the Y coordinate can be determined.

The procedure then reruns with the X direction being driven – in this case, the voltage value is read from one of the Y electrodes.

A resistive touch sensor does not depend on the electrical property of capacitance (in contrast to the technology of capacitive touch panels) sensors sense the pressure on the screen surface. Hence, noise immunity is also higher for resistive touch panels.

Resistive touch technology provides accommodation with non–conducting materials. Thanks to that, a stylus and gloves can be used for modules with resistive touch screens. Resistive touch panels are widely used for the applications planned for harsh

environmental conditions. It is also worth to mention, that resistive technology, unlike as capacitive, supports only “single – touch”.

5.3 Advantages of Resistive Touch Screens:

- Most cost-effective touch screen technology
- Easy integration into value-added, turnkey membrane switch assemblies
- Can be operated with gloves and pen/stylus as well as human fingers
- High resistance to dust, moisture and humidity
- Works with handwriting recognition systems.

6. Conclusion:

Throughout This chapter we were discussing about the hardware and the electrical components of ball and plate system.. The objective is to balance a rolling ball in a specific position with the least possible error and smallest settling time achieved for the real system. Linear and nonlinear mathematical model of the system will be derived for simulation purpose. MATLAB will used to evaluate the closed loop system response and to determine the PID parameters. An arduino will be used as the controller in which the PID control algorithm is implemented.

CHAPTER III:

SYSTEM MODELING AND SIMULATION

1.Introduction:

In this paper we are going to study and derive a mathematical model for the ball and plate system, then this model is going to be implemented in Simulink with an animation that illustrates the model. A detailed description is then given about manufacturing a ball and plate device. Finally the results of the Simulink model and the animation will be reported accompanied with recommendations and future research.

2. Mathematical Model:

2.1 Approximations and assumptions:

In this project, The model derived and used is a decoupled model, it assumes that the motion of the ball is uncoupled in the two perpendicular axes of the plate, which means that the motion of the ball due to the rotation of plate around one of its axes is independent on the motion due to the rotation of the plate around the other axis. Accordingly, a 2D-model will be derived for each degree of freedom of the plate, then the principle of superposition will be applied to obtain the exact position of the ball.

The assumptions used for this model are:

- No slipping conditions for the ball.
- The ball is completely symmetric and homogeneous.
- Rolling friction is neglected.
- The ball and plate are in contact at all times.
- The plate surface is perfectly smooth and rigid.

The following table contains the parameters used for deriving the model and their significance :

Table III.1 list of paramerters

Parameter	Description
l	Length of the plate
γ	Arm length of the servo
α	Plate angle in x-direction
β	Plate angle in y-direction
θ	Servo angle
m	Mass of the ball
R	Radius of the ball
I	Moment of inertia of the ball
g	Gravitational acceleration
V	Motor Voltage

2.2 Derivation:

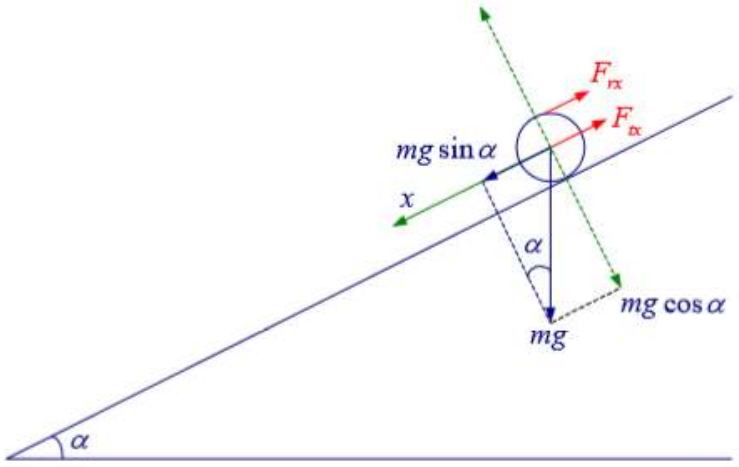


Fig. 1 free body diagram

Considering the free body diagram in fig. 1 where the direction of increasing x is upward along the inclined plane, According to Newton's second law the total force responsible for the translational motion is governed by the following equation

$$F_{tx} = m\frac{d^2x}{dt^2} = m\ddot{x} \qquad (1)$$

and the rotational motion (Torque) is given by

$$T_\gamma = I\frac{\chi}{R}$$

thus, by dividing the last equation by R, and substituting $I = \frac{2}{s}mR^2$, we have

$$F_r = \frac{\tau_r}{R} = I\frac{\ddot{\chi}}{R^2} = \frac{2}{s}m\ddot{x} \qquad (2)$$

We can now say that according to fig.1, $Ftx{=}mgsin\alpha{-}Fr$. Thus by substituting (1) and (2) into the last equation:

$$m\ddot{x} + \frac{2}{5}m\ddot{x} = mgsin\alpha$$

by dividing by m and rearranging the equation, we obtain:

$$\ddot{x} = \frac{5}{7}gsin\alpha \qquad (3)$$

The same equation can be obtained for the y-direction:

$$\ddot{y} = \frac{5}{7}gsin\beta \qquad (4)$$

Equations (3) and (4) can be easily integrated to obtain the position of the ball at any instant. We now have the position of the ball in terms of the angles of the plate, but in order for this model to be integrated into the system, the position of the ball will have to be related to the voltage given to the servo motor in control of the rotation of the plate. Therefore we use the following model.

2.3 Servo Model:

In an attempt to simplify calculations, A DC-Motor was used as an approximation for the servo. The standard transfer function for the DC-motor was used:

$$\frac{\theta(s)}{V(s)} = \frac{C_1}{s(s + C_2)}$$

Where C_1 and C_2 are both functions of the parameters of the motor, and since θ and α or β are related by the geometry of the setup, in our case $\alpha = \frac{r}{l}\theta$, therefore $\ddot{x}$ can now be related to the voltage of the motor.

3. Simulink Model:

The Simulink model for this system is formed of several blocks and subsystems:

3.1 Ball Model subsystem:

As shown in fig.2, this subsystem models equation (3). It has one input, which is the angle α in case of x-direction and β in case of y-direction, and two outputs, x and $\dot{x}$.

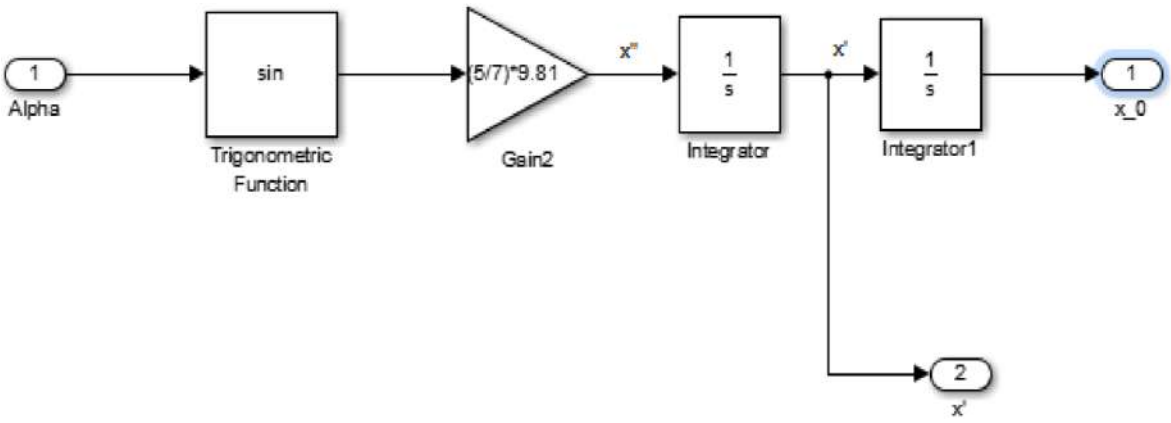


Figure III.1 Block Ball Model

3.2 Feedback loop 1 (Servo PID):

Since the motor used in this model is only an "approximated" servo motor, The voltage corresponding to a specific θ may not be able to achieve that desired θ, which will affect the entire structure. To solve this, the feedback loop (PID controller) shown in

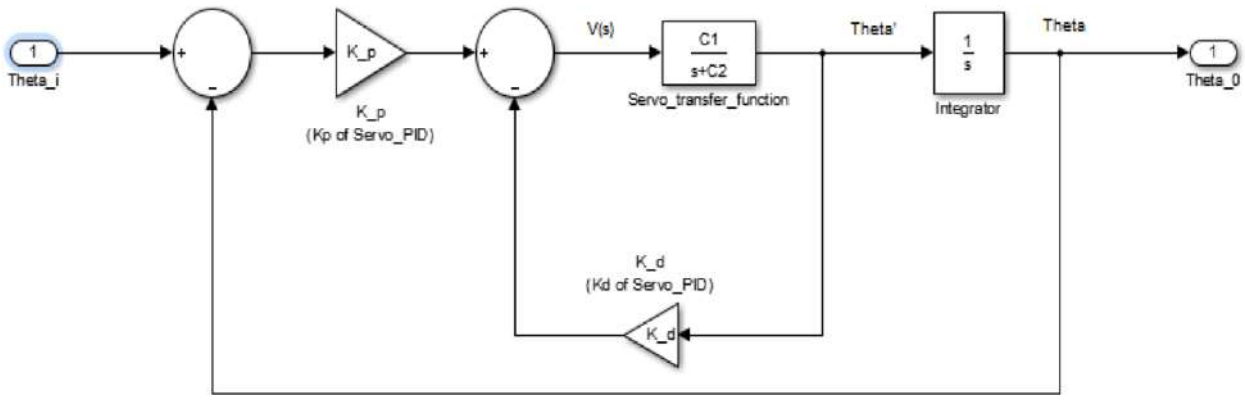


Figure III.2 Servo feedback loop

fig.3 was introduced to ensure θ reaches the desired set-point (Note that this problem only arises in the simulation because of the motor model used, thus this feedback loop will not be necessary when implementing the model on hardware with "real" servo). The input of the subsystem is the required theta and the output is the instantaneous theta. Since the original transfer function was

$$\frac{\theta(s)}{V(s)} = \frac{C_1}{s(s + C_2)}$$

By reducing the block diagram in fig.3, the transfer function for the subsystem with PID feedback loop becomes

$$\frac{\theta_0(s)}{\theta_i(s)} = \frac{K_p C_1}{s^2 + (K_D C_1 + C_2)s + K_p C_1}$$

by comparing the above equation to the general second order transfer function

$$\frac{\theta_0(s)}{\theta_i(s)} = \frac{\omega_n^2}{s^2 + 2\omega_n \zeta s + \omega_n^2}$$

The constants K_p and K_d can be obtained, where

$$K_d = \frac{2\omega_n \zeta - C_2}{c_1} \qquad K_p = \frac{\omega_n^2}{c_1}$$

Therefore, by choosing the parameters ωn and ζ appropriate to the system, K_p and K_d can be determined. The parameters chosen for this system are attached in Appendix B and in the m-file attached to this report.

3.3 Feedback loop 2 (Distance PID):

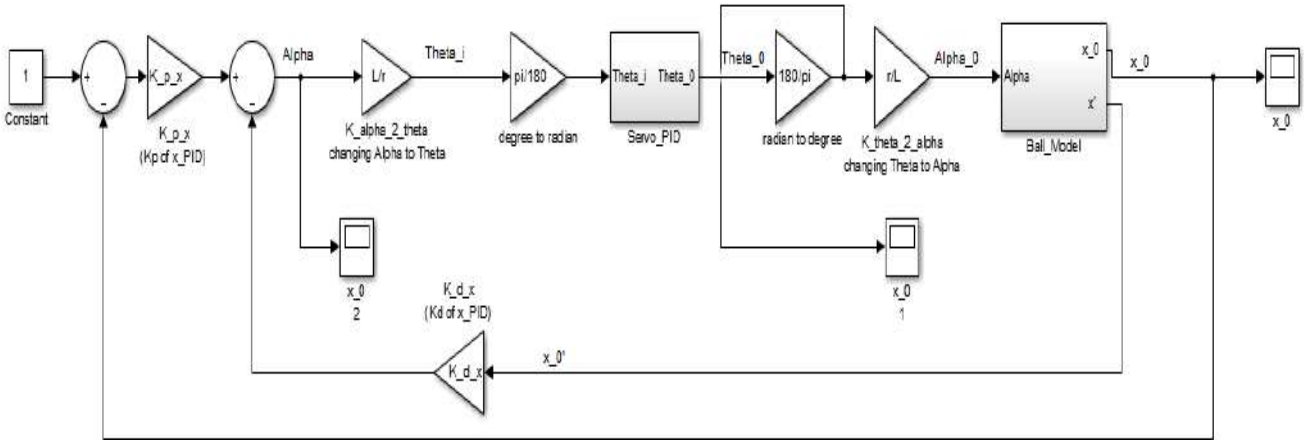


Figure III.3 Distance feedback loop

In fig.III.3, the main feedback loop is shown with the rest of the model, where a similar PID feedback control was used to ensure that the target position is reached. The input here (the constant) is the target x which is x_i and the output is the instantaneous x_0 This is done by applying the PID equation

$$\alpha = K_{px}(x_i - x_0) - K_{dx}\dot{x}$$

by transforming the previous equation into the frequency domain, it becomes

$$\alpha(s) = K_{px}\big(x_i(s) - x_0(s)\big) - K_{dx}sx(s)$$

and since for small α, equation (3) becomes $\ddot{x} = \frac{5}{7}g\alpha$, thus by transferring it into the frequency domain, the equation becomes

$$\alpha(s) = \frac{7s^2}{5g}x_0(s) = \frac{s^2}{7}x_0(s)$$

by substituting into the PID equation, it becomes

$$\frac{s^2}{7}x_0(s) = K_{px}\big(x_i(s) - x_0(s)\big) - K_{dx}sx_0(s)$$

by further simplifying, the transfer function for the feedback loop is obtained

$$\frac{x_0(s)}{x_i(s)} = \frac{7K_{px}}{s^2 + 7K_{dx}s + 7K_p}$$

Finally, by comparing with $G(s) = \frac{\omega_n^2}{s^2+2\omega_n\zeta s+\omega_n^2}$ we get

$$K_d = \frac{2\omega_n\zeta}{7} \qquad K_{px} = \frac{\omega_n^2}{7}$$

4. Position detection:

There are many ways to specify the location of the ball on the plate for the microcontroller such as the touch pad, ultrasonic sensor and image processing. Since the touch pad is a little expensive and not that available and the ultrasonic sensor may experience noise that make the mission harder, we went with the remaining solution, image detection. A cam is suspended perpendicularly above the plate that captures images of size 640×480. We chose the 480 to be the dimension of the Y-axis, hence the 640 is the X-axis. The plate is 35×35 cm square, the camera is adjusted so that the length of the plate is filling the height of the image, that's 35 cm is seen as 480 pixels (Height of the captured image). The camera is also adjusted to make the plate centered with respect to the X-axis.

Images are captured using the suspended cam, Visual Studio code converts it from RGB to HSV because the ball detection is easier that way, then the image is smoothed using Gaussian filter and some parameters are chosen to make the ball white and the plate black in the image to make it easier to detect. The code can now detect any circles in the image, a threshold is implemented to remove the small noise or errors. The next step is easy, is to determine the location of the center of the ball, which we consider the reference point on the ball, and the

radius of the ball. This location is in Pixels not cm, so we must map the values of the pixels to cm.

Since we adjusted the plate to fill the height of the image we know that

35 cm=480 $pixels$

$$or \qquad 1\ cm = \frac{480}{35}\ pixels \qquad \rightarrow y_{cm} = y_{pixels} * \frac{35}{480}$$

For the X position, the plate is centered and we conclude that the plate is $\frac{640-480}{2}$ pixels away from the zero on the X-axis

$$x_{cm} = \left(x_{pixels} - 80\right) * \frac{35}{480}$$

By subtracting 80 from the x position, the reference is shifted 80 pixels to the right (The edge of the plate).

We have succeeded in knowing the instantaneous position of the ball on the plate (The Visual Studio Code is in Appendix C).

5. Results:

Here we discuss the results of the Simulink simulation and the animation. Also the results of the ball detection will be reported. But due to the shortage of time the hardware couldn't be tested, and thus the Arduino code might need some debugging.

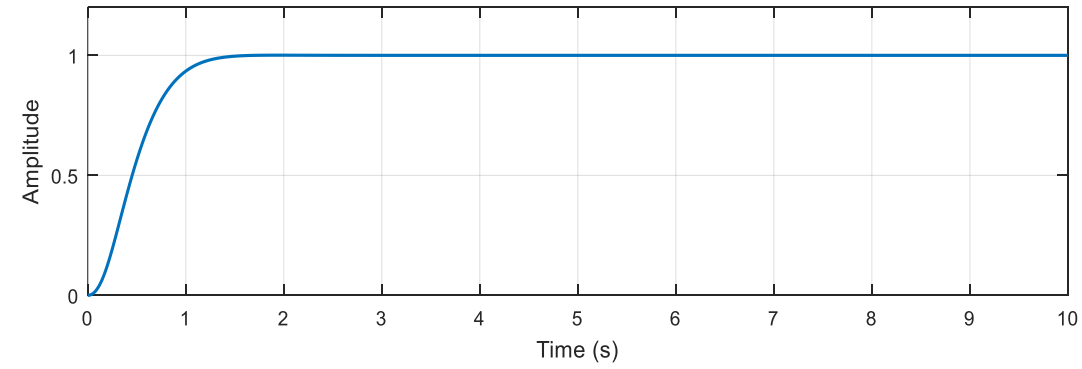
5.1 Simulink results:

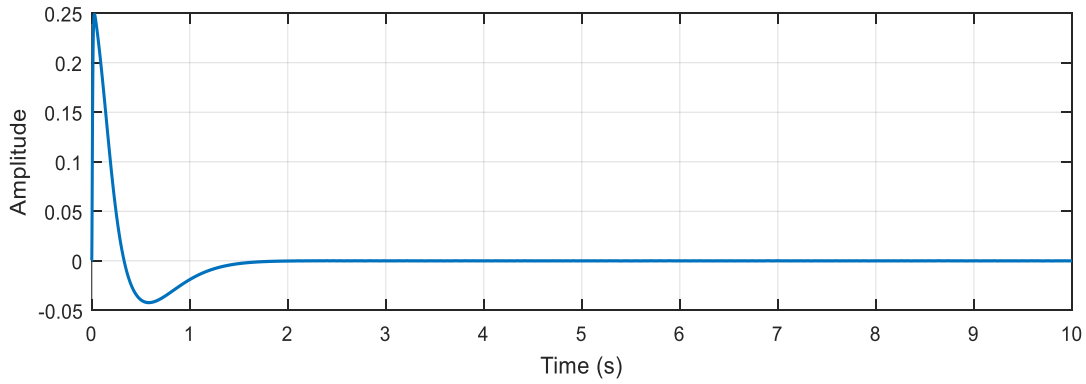
At first, the problem of obtaining a suitable response was approached by trying to achieve a critically damped response for the fastest non oscillatory time response. The response in figure 11 was the result of this trial. Unfortunately, the θ readings required to reach that response were not physically possible as shown in figure 12. As a result, another trial was executed, where the response was allowed to overshoot by a small percentage of about 10% to obtain reasonable practical θ values, figure 13 was the result of that trial, and the figure 14 shows the theta values.

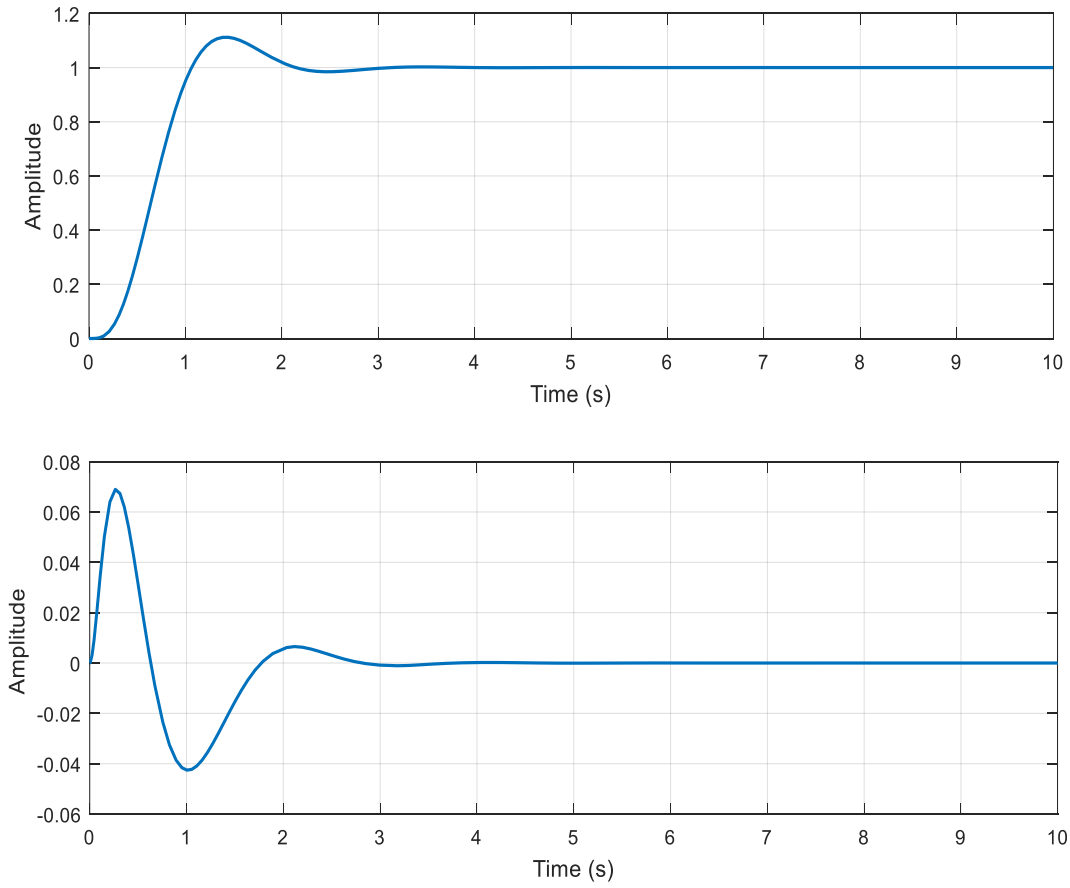
The gains used in the first trial were:K_{px} =2.29, K_{dx}=1.02857, K_p=746.98 and K_p=6.078. The gains for the second trial are those shown in the appendix and in the attached Parameters.m file

The animation results are found in the video attached with the report.

The ball detection code is working and it detects the ball.







6. Future research:

The hardware device must be tested in order to verify the Arduino code. Further improvements can then be done to the Simulink model in order to make it as realistic as possible, one of the improvements is to perform a parameter estimation on the gain of the angle by taking real data from the hardware device and performing parameter estimation in Matlab. The Simulink model can be further improved by adding nonlinear terms such as sin ($\boldsymbol{\theta 1}$) where $\boldsymbol{\theta 1}$ is the corresponding servo angle, also the coupling between the two degrees of freedom due to Coriolis Effect can be added by putting in each degree of freedom equation another term equals to $\boldsymbol{\theta 2}$ where $\boldsymbol{\theta 2}$ the angle of the opposite servo. All these added terms are multiplied by unknown coefficients which can be acquired using the great parameter estimation tool in Matlab.

After acquiring an almost complete model of the hardware in Simulink, optimal control techniques can be applied in order to get the PID constants which will provide the best response according to the cost function which will mainly depend on the setting time and the steady state error.

7. Conclusion:

In this chapter we saw how the ball and plate system control problem is solved going from the non-linear equations to the linear model and simulation using Matlab and Simulink toolbox. We divide the system in two parts x and y because it was the easiest way to have a full control.in the fowling section we will present a matlab model and the response matlab.

A mathematical model of the ball and plate system was developed by using physical and electrical laws. A simplified mathematical model was derived by system parameters. The controller parameters values (Kp, Ki and Kd) were obtained by using manual tuning method from practical model so as to perform best system response

In non-linear simulation the pid control can’t make the system to be stable but in real system the pid control success to make the system stable

CHAPTER IV:

ANIMATION VRML OF BALL ON PLATE SYSTEM

1. Inroduction:

The purpose of this chapter is to give Simulink ® users a better understanding on how to animate a physical system that has more than one component in a virtual reality environment. This chapter will also help the reader to implement a simple PID controller to control a ball on a plate and visualize the performance of the controller.

2. Virtual Reality Modelling Language :

Virtual Reality Modelling Language (VRML) is a file format for describing interactive 3-D object and worlds. It is designed to be used on the Internet, intranets and local client systems. It is also intended to be a universal interchange format for the integrated 3-D graphics and multimedia. It is used in many application areas such as engineering and scientific visualization, entertainment and educational titles, web pages, multimedia presentations and shared virtual worlds .

In the VRML model, a 3-D scene is described by a hierarchical tree structure of objects (nodes). Each node in the tree represents some functionality of the scene. There are 54 different types of nodes. Some of the nodes are shapes nodes, representing real 3-D objects, and some of them are grouping nodes, used for holding child nodes. Figure III.1 shows the hierarchical structure of the nodes.

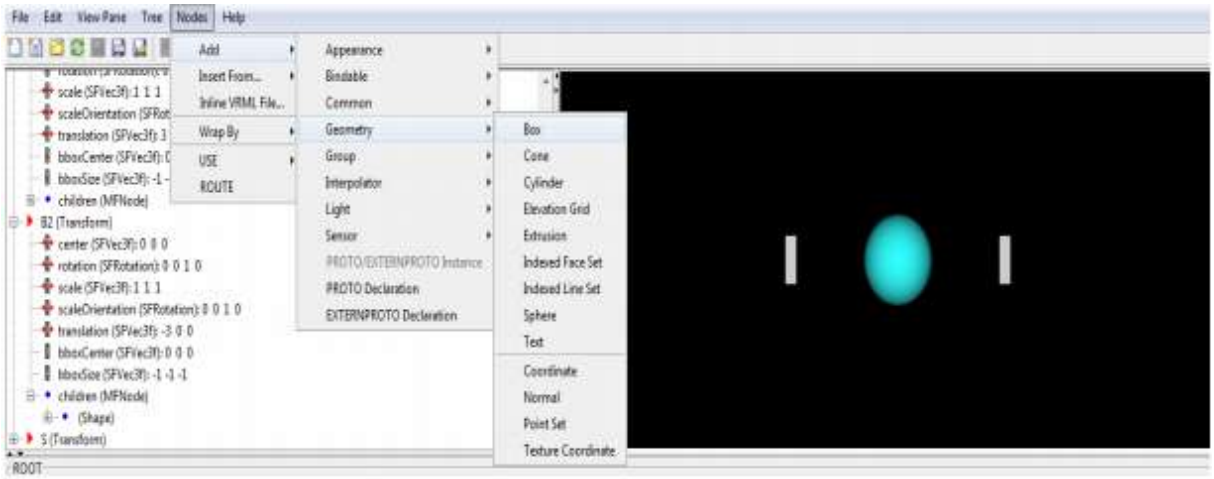


Figure IV.1 Hierarchical Tree Structure of Nodes

Examples of nodes includes:

- Box node: Represents a box in a scene.
- Transformation node: Defines the position, the scale, the scale orientation, rotation translation and the children of its subtree (grouping nodes).

- Material node: Corresponds to material in a scene.
- Directional Light node: Represents lighting in a scene.

Examples of nodes includes :

- Box node: Represents a box in a scene.
- Transformation node: Defines the position, the scale, the scale orientation, rotation translation and the children of its subtree (grouping nodes).
- Material node: Corresponds to material in a scene.
- Directional Light node: Represents lighting in a scene.

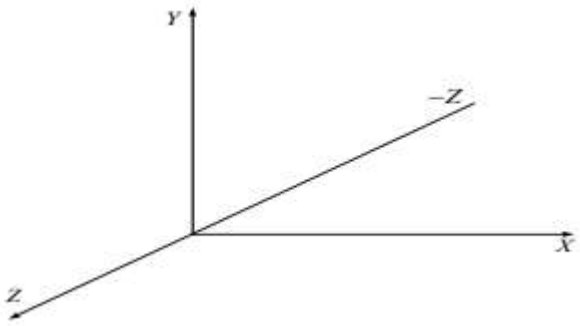


Figure IV.2 VRML Coordinate System (MathWorks, 2015)

The rotation angles are defined using the right-hand rule. The counterclockwise direction is the positive rotation angle of an object moving around that axis. This is shown in Figure 2.14 (MathWorks, 2015).

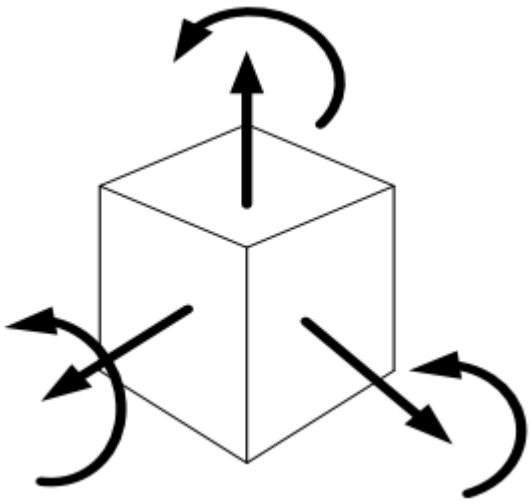


Figure IV.3: VRML Rotation Angles (MathWorks, 2015)

3. Simulink® 3D Animation :

Simulink®3D Animation™ product uses VRML97 technology to deliver a 3-D visualization solution for MATLAB users. The Simulink®3D Animation™ product provides an interface to connect an existing virtual world, defined with VRML, to the Simulink® and MATLAB® environments. It is a useful contribution to a wide use of VRML97 in the field of technical and scientific computation and interactive 3-D animation. The Simulink®3D Animation™ product is a solution for interacting with virtual reality models of dynamic systems over time. It extends the capabilities of the MATLAB and Simulink software into the world of virtual reality graphics. The necessary steps to use the Simulink®3D Animation™ product are:

Step 1: Create virtual worlds or three-dimensional scenes using standard Virtual Reality Modelling Language (VRML).

Step 2: Create and define dynamic systems with the MATLAB and Simulink products.

Step 3:View moving three-dimensional scenes driven by signals from the Simulink environment.

Simulink®3D Animation™ library provides blocks to directly connect Simulink signals with virtual worlds. This connection lets us to visualize the Simulink model as a threedimensional animation. Most of the software features can be implemented with Simulink blocks. Once these blocks are included in the Simulink diagram, one can select a virtual world and connect Simulink signals to the virtual world. It automatically scans a virtual world for available VRML nodes that the Simulink can drive. The VRML node properties are listed in a hierarchical tree-style viewer. One select the degrees of freedom to control from within the Simulink interface. After connecting these inputs to appropriate Simulink signals, the simulation is then viewed with a VRML viewer. It contains a viewer that is the default viewing method for the virtual worlds. To create virtual worlds that are desired to be connected to the Simulink block diagrams, the VRML editor is used. It also includes one of the classic VRML authoring tools, VRealm Builder by Ligos. With the addition of this VRML authoring tools, VRML products provides a complete authoring development, and working environment for carrying out 3-D visual simulations .

3.1 Graphical User Interface (GUI) :

Graphical User Interface (GUI), refers to the universal idea of icons, buttons, etc., that are visually presented to a user as a "front-end" of a software application .However, GUI consist of a figure window containing menus, buttons, text, graphics etc. that a user can manipulate interactively with the mouse and keyboard. There are two main steps in creating a GUI):

- One is designing its layout, and
- The other is writing callback functions that perform the desired operations when the user selects different features.

The strength of GUIs is that they provide a means through which individuals can communicate with the computer without programming commands. This components have become quite standardized and developed into a user friendly and intuitive set of tools. These tools can be used to increase the productivity of a user or to provide a window to the sophistication and power of a MATLAB application for those with little or no MATLAB programming experience The graphics objects components of GUI come in two classes :

- User interface controls (unicontrols)
- User interface menus (unimenus)

The uicontrols and uimenus can be combined with other graphic objects to create informative, intuitive and aesthetically pleasing GUIs. Access to them is done using two approaches, the first approach is a low-level, bottom-up approach where the use of skills to handle graphics to write M-files that implement the GUI. The second approach is the use of MATLAB"s Graphical User interface Development Environment (GUIDE), which is high-level, yet powerful and extremely easy to use; an excellent tool for quickly developing GUIs that takes care of much of the "bookkeeping" usually associated with GUI development [18].

4. Simulation models of mechatronics systems in Matlab/Simulink:

We linked our model to animation using VR sink. The first step we made was to make the design file of the ball and plate. We entered the data which came from the PID x and y coordinates on a multiplexer and entered them on a VR expander to produce x, y, z coordinates of the ball as shown in figure 5. we faced a problem in rotation that when we enter the angle of rotation to the plate and enable it to rotate on x and z axes, it rotates only about the diagonal of the plate as shown in figure 6.we need to have to independent rotation

axes. So we made a transform into a transform and defined two axis of rotation in each transform to have two independent axes of rotation. The transform is the container of geometry. For instance, if we want to make a sphere. We first make a transform and define its coordinates in space and then create inside it the shape we want. The transform is the parent node and the shape is the child node as illustrated in VRML method to represent a 3-D scene

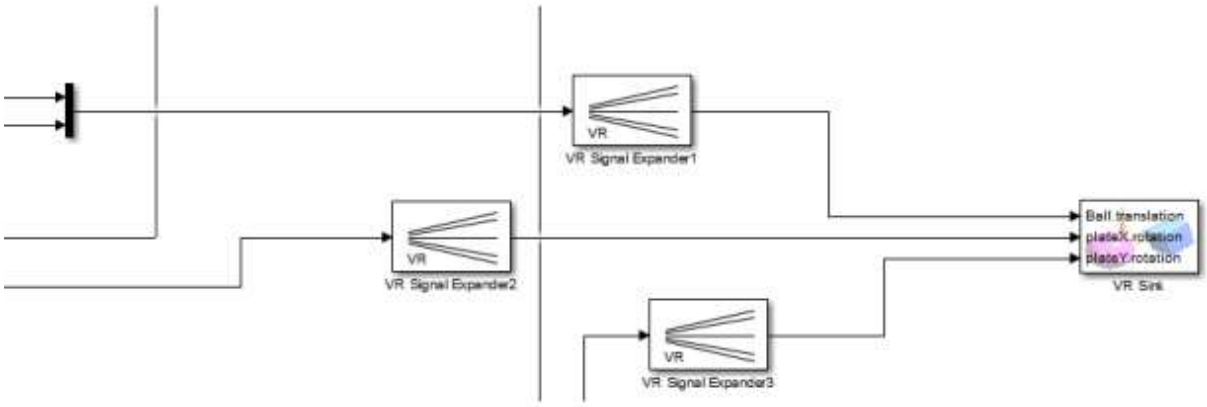


Figure IV.4 Virtual model of the Ball and Plate in control scheme

Connection of the simulation and the virtual model is realized in the tool Simulink by the block VR Sink. This block contains a root folder of the virtual model and enables to choose parts of the model, which should be animated.

Created virtual models are used for testing and comparison of the designed control algorithms, View of the virtual model simulation with the time response of the monitored physical variables is for user more valuable, like the simulation of the simple, none virtual model.

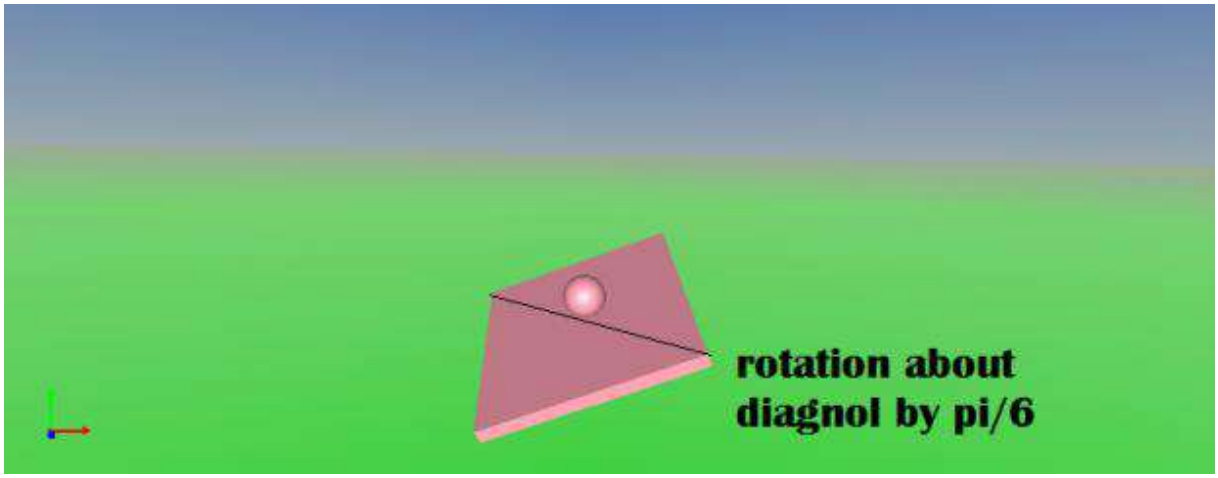


Figure IV.5 Virtual Reality (VR) Model of the Ball and Plate System

Figure (prochain) shows that in transform (plate X) we have defined rotation vector 0 0 1 0 which puts 0 in the x and y to disable the rotation in those directions and puts 1 in the z axis to enable the rotation in z axis. The last number in the rotation vector represents the value of the angle by which the plate will rotate. Inside transform X we made the rotation vector 1 0 0 0 to enable the x axis rotation. The last value in each vector is entered from our model to make the plate rotate about the specified axis with the specified value.

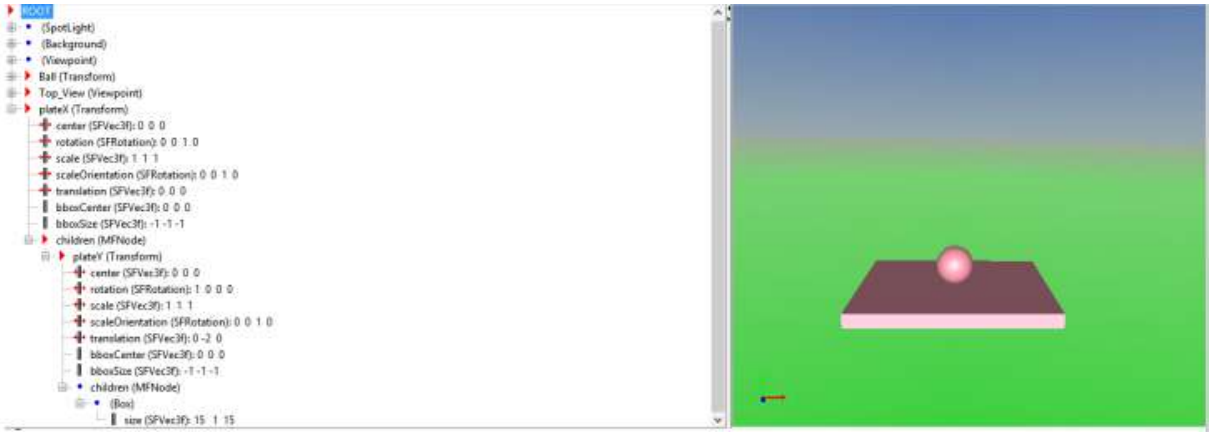


Figure IV.6 Virtual model of the Ball and Plate and its structure

5. Conclusion:

The purpose of this chapter is to give us as Simulink ® users a better understanding on how to animate a physical system that has more than one component in a virtual reality environment. In This chapter we've seen how to implement a simple PID controller to control a ball on a plate and visualize the performance of the controller and construct the Simulink ® model that will animate the virtual scene then running the simulation. The chapter concludes by a control problem for a ball on a platform. The position of the ball is controlled on the platform by means of two independent actuators. The problem will include developing the equations of motion of the ball, developing the two PID controllers for the platform, and animating the ball and the platform. The electronic version of the Simulink ® models, VRML models.

The virtual model B&P provide more illustrative view of the simulation with control algorithms in chosen control scheme. Designed virtual models are used for the presentation of control algorithms in case, that the real model is not available

GENERAL CONCLUSION:

We present a complete Ball and Plate design and simulation procedure. The plant is easy to use and to replicate at a low cost, so ideally anyone can use this guide to obtain the same results that are presented in this chapter. It also provides us the experience of making physical systems, and exposes us to the engineering process of designing, building and testing a real system. The plant can be used by undergraduate us without risk of injury or damaging expensive components so they will be motivated to explore the plant behavior. Building the plant provides an understanding of physical considerations that must be taken into account during the control design process. This provides us with a unique opportunity to learn through experience about control system's theory. The designed plant can be controlled with Simulink in real time, which allows to design and implement almost all kind of controllers on it. Additionally the models that were obtained can be used to test controllers before physical implementation, which provides the means to safely explore and test. The comparisons presented in the previous chapter show a very important underlying quality of the system. The math model, the Sims-cape simulation, the identified transfer function and the real system can all be stabilized by the same controller and all of them have a similar behavior. This fact shows the validity of all the representations and will let us understand why modelling is important and useful. In the same regard, when we compare the models to the real system they can better understand the effects of approximations and assumptions. Future work for the project has two ramifications. The first is to design experiments using the plant to help students learn about specific topics. The second, is to use this project as a guide for us to develop other types of controllable plants, like an inverted pendulum or something more focused on industry processes like mixing and controlling concentration.

Futur work :

The future work and necessary improvements that would be relevant are several, with regards to both controller design and construction. Foremost the implementation of powerful and accurately reading and responding servo motors of the same brand and model to avoid dierent calibrations of the PD controller gains. Secondly, using motors that are analog does impose diculties with regards to reading PWM. However, this issue could be resolved by using digital servo motors having better PWM reading properties. Another improvement is a

improved touch panel, or other ball tracing equipment, with higher reading accuracy considering data collection to achieve better input. These components could likely improve the construction significantly. However, with increased strength in the motors and new reading equipment the construction model would require modifications. From a construction stand point, the play in the joints and links and the tolerances should be improved since this is a large causing factor to shattering of the touch panel or platform. The play is as aforementioned, caused by the servo motors but further enhanced by the play and insucient tolerances in the joints. Completely accurate results and data readings could not be attained due to slow serial communication with the Arduino through USB connection. When attempting to extract data, the consequence was a slower response in the servo motors and thus, inability to maintain the ball on the platform. To avoid this, another system to extrapolate data should be further investigated. Subsequently allowing for a better analysis and discussion. Due to time constraints this was not possible in the project. Another additional improvement could be to introduce a more advanced filter, rather than the low pass filter e.g. Kalman filter. This would give better readings from the touch panel or the altered reading equipment. Lastly, increasing the degrees of freedom of the platform would, presumably, improve the handling of the ball by a smoother reaction to ball displacement. The introduction of additional degrees of freedom would however require a more advanced controller and system due to additional motors and axes involved in the process. The principles of the ball on platform does however remain the same regardless of degrees of freedom.

REFERENCES

[1] Hongwei Liu and Yanyang Liang, Trajectory tracking sliding mode control of ball and plate system, School of Information Engineering Southwest University of Science and Technology, 2010

[2] Abubakar UMAR, Development of tracing position and control in the ball-and-board system using

[3] K. Ogata, Modern control engineering. Englewood Cliffs, N.J.: PrenticeHall, 1970.

a double feedback loop structure, Department of Electrical and Computer Engineering Faculty of Engineering Ahmadu Bello University Zaria, Nigeria, 2017

[4] https://www.electrical4u.com/control-system-closed-loop-open-loop-control-system/

[5] Vipin Jain, Types of Controllers | Proportional Integral and Derivative Controllers, 2020, https://www.electrical4u.com/types-of-controllers-proportional-integral-derivative-controllers/

[6] A.Taifour Ali, Design and implementation of ball and plate system using PID controller, Sudan University of Science and Technology, 2017.

[7] Vipin Jain, Types of Controllers | Proportional Integral and Derivative Controllers, 2020, https://www.electrical4u.com/types-of-controllers-proportional-integral-derivative-controllers/

[8] De-hu Yuan. Pneumatic servo ball & plate system based on touch screen and oscillating cylinder. SJTU-SMC Technology Center. Shanghai Jiao Tong University, china.

[9] S. Monk, 30 Arduino projects for the evil genius. New York: McGrawHill, 2010.

[10] https://www.elprocus.com/different-types-of-arduino-boards/

[11] https://learn.sparkfun.com/tutorials/what-is-an-arduino/all.

[12] https://learn.sparkfun.com/tutorials/what-is-an-arduino/all

[13] https://www.circuitschools.com/what-is-arduino-how-it-works-and-what-you-can-do-with-arduino/

[14] https://www.electrical4u.com/what-is-servo-motor/

[15] https://celebratelife24x7.wordpress.com/2014/11/18/electronics-101-servo-motor/

[16] Wilson, A. TouchLight: An Imaging Touch Screen and Display for Gesture-Based Interaction, International Conference on Multimodal Interfaces, 2004.

[17] https://www.elprocus.com/touch-screen-technology-definition-working-types-applications/

[18] Abubakar UMAR, Development of tracing position and control in the ball-and-board system using a double feedback loop structure, Department of Electrical and Computer Engineering Faculty of Engineering Ahmadu Bello University Zaria, Nigeria, 2017

APPENDIX

Appendix A: Parameters and Arduino code

 System parameters: % motor Parameters

```
R_a=2.6;

K_m=0.00767;

K_t=K_m;

J_m=3.87*(10)^-7;

J_t=0.7*(10)^-7;

K_g=14*5;

B_eq=4*(10)^-3;

eff=0.7395;

J_eq=0.0023;

C1=(eff*K_m*K_g)/(R_a*J_eq);

C2=(B_eq/J_eq)+((eff*K_m*K_m*K_g*K_g)/(R_a*J_eq));

% x_PID

t_p_x=1.8;

damping_ratio_x= 0.65;

w_n_x=3.14/(t_p_x*(1-damping_ratio_x^2)^(1/2));

K_p_x=w_n_x^2/7;

K_d_x=(2*damping_ratio_x*w_n_x)/7;

% Servo_PID

t_p=0.5;

damping_ratio= 0.85;

w_n=3.14/(t_p*sqrt(1-damping_ratio^2));

K_p=w_n^2/C1;
```

K_d=((2*damping_ratio*w_n)-C2)/C1;

% Plate

L=0.35;

r=0.0254;

 Arduino code:

```
#include <Servo.h>
Servo servo1; // creating a servo object
Servo servo2; // creating a servo object
/// Variables ///
float Kp=2.1405;
float Kd=-0.25785;
float Kp_x=0.75277;
float Kd_x=0.42631;
float Setpoint_x; // this is set manually
float Setpoint_y; // this is set manually
float L; // half the length of the plate
float r; // the arm length of the servo
int pos_x=0;
```

```
int pos_y=0;
float error_x, error_d_x, error_x_old, Alpha_x, Theta_setpoint_x;
float error_y, error_d_y, error_y_old, Alpha_y, Theta_setpoint_y;
float error_theta_x, error_theta_old_x, Theta_x, error_d_theta_x, V_x;
float error_theta_y, error_theta_old_y, Theta_y, error_d_theta_y, V_y;
void setup() {
servo1.attach(9); // attaches the servo object to pin 9 on the arduino board
servo2.attach(10); // attaches the servo object to pin 9 on the arduino board
}
void loop() {
////////////////// PID_x //////////////////////////
// put here instead of this line the function that reads the position and assume it gives pos_x
and pos_y relative to the a fixed origin (center of the plate) ///
error_x=pos_x-Setpoint_x;
error_d_x=error_x-error_x_old;
error_x_old=error_x;
Alpha_x=Kp_x*error_x+Kd_x*error_d_x; // alpha is the angle of plate relative to the zero
position
Theta_setpoint_x=Alpha_x*(L/r)*(3.14/180); // Theta is the angle of the servo
////////////////// PID_y //////////////////////////
error_y=pos_y-Setpoint_y;
error_d_y=error_y-error_y_old;
error_y_old=error_y;
Alpha_y=Kp_x*error_y+Kd_x*error_d_y; // alpha is the angle of plate relative to the zero
position
Theta_setpoint_y=Alpha_y*(L/r)*(3.14/180); // Theta is the angle of the servo
```

```
//////////////// Servo_PID_x //////////////////////
Theta_x=servo1.read();
error_theta_x= Theta_x-Theta_setpoint_x;
error_d_theta_x=error_theta_x-error_theta_old_x;
error_theta_old_x=error_theta_x;
V_x=Kp*error_theta_x+Kd*error_d_theta_x;
//////////////// Servo_PID_y //////////////////////
Theta_y=servo2.read();
error_theta_y= Theta_y-Theta_setpoint_y;
error_d_theta_y=error_theta_y-error_theta_old_y;
error_theta_old_y=error_theta_y;
V_y=Kp*error_theta_y+Kd*error_d_theta_y;
/// an equation depending on the parameters of the servo will change the voltage into a
corresponding angle for the servo to be able to use servo.write() ///
```

```
// Theta_x=some_equation(V)*(180/3.14)
servo1.write(Theta_x);
servo2.write(Theta_y);
```

Appendix B: ball detection ‘visual studio’ code

```
#include <iostream>
#include<opencv/cvaux.h>
#include<opencv/highgui.h>
#include<opencv/cxcore.h>
#include <sstream>
#include <string>
#include <opencv\cv.h>
#include<stdio.h>
#include<stdlib.h>
// Need to include this for serial port communication
#include <Windows.h>
int H_MIN = 0;
int H_MAX = 256;
int S_MIN = 0;
int S_MAX = 256;
int V_MIN = 0;
int V_MAX = 256;
using namespace cv;
const string trackbarWindowName = "Trackbars";
void on_trackbar( int, void* )
void createTrackbars(){
//create window for trackbars
namedWindow(trackbarWindowName,0);
//create memory to store trackbar name on window
```

```
char TrackbarName[50];
sprintf( TrackbarName, "H_MIN", H_MIN);
sprintf( TrackbarName, "H_MAX", H_MAX);
sprintf( TrackbarName, "S_MIN", S_MIN);
sprintf( TrackbarName, "S_MAX", S_MAX);
```

```
sprintf( TrackbarName, "V_MIN", V_MIN);
sprintf( TrackbarName, "V_MAX", V_MAX);
createTrackbar( "H_MIN", trackbarWindowName, &H_MIN, H_MAX, on_trackbar );
createTrackbar( "H_MAX", trackbarWindowName, &H_MAX, H_MAX, on_trackbar );
createTrackbar( "S_MIN", trackbarWindowName, &S_MIN, S_MAX, on_trackbar );
createTrackbar( "S_MAX", trackbarWindowName, &S_MAX, S_MAX, on_trackbar );
createTrackbar( "V_MIN", trackbarWindowName, &V_MIN, V_MAX, on_trackbar );
createTrackbar( "V_MAX", trackbarWindowName, &V_MAX, V_MAX, on_trackbar );
}
int main(int argc, char* argv[])
{
FILE *fp;
fp=fopen("file.txt","w");
HANDLE hSerial = CreateFile(L"COM7", GENERIC_READ | GENERIC_WRITE, 0, 0,
OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0);
if (hSerial !=INVALID_HANDLE_VALUE)
{printf("Port opened! \n");
DCB dcbSerialParams;
GetCommState(hSerial,&dcbSerialParams);
dcbSerialParams.BaudRate = CBR_9600;
dcbSerialParams.ByteSize = 8;
dcbSerialParams.Parity = NOPARITY;
dcbSerialParams.StopBits = ONESTOPBIT;
SetCommState(hSerial, &dcbSerialParams);
}
```

```
CV_RGB(0,255,0),
CV_FILLED);
cvCircle(p_imgOriginal,
cvPoint(cvRound(p_fltXYRadius[0]), cvRound(p_fltXYRadius[1])),
cvRound(p_fltXYRadius[2]),
CV_RGB(255,0,0),
3);
}
cvShowImage("Original", p_imgOriginal);
cvShowImage("Processed", p_imgProcessed);
cvReleaseMemStorage(&p_strStorage);
charCheckForEscKey = cvWaitKey(10);
if(charCheckForEscKey == 27) break;
}
cvReleaseCapture(&p_capWebcam);
cvDestroyWindow("Original");
cvDestroyWindow("Processed");
// This closes the Serial Port
CloseHandle(hSerial);
return(0);
}


else
```

```
{
if (GetLastError() == ERROR_FILE_NOT_FOUND)
{printf("Serial port doesn't exist! \n");}
printf("Error while setting up serial port! \n");
}
char outputChars[] = "c"; //
DWORD btsIO;
CvSize size640x480 = cvSize(640, 480);
CvCapture* p_capWebcam; / . .
IplImage* p_imgOriginal;
IplImage* p_imgProcessed;
IplImage* p_imgHSV;
CvMemStorage* p_strStorage;
CvSeq* p_seqCircles;
float* p_fltXYRadius;
int i;
char charCheckForEscKey;
p_capWebcam = cvCaptureFromCAM(1);
if(p_capWebcam == NULL) {
printf("error: capture is NULL \n");
getchar();
return(-1);
}
cvNamedWindow("Original", CV_WINDOW_AUTOSIZE);
cvNamedWindow("Processed", CV_WINDOW_AUTOSIZE);
```

```
createTrackbars();
p_imgProcessed = cvCreateImage(size640x480,
IPL_DEPTH_8U,
1);
p_imgHSV = cvCreateImage(size640x480, IPL_DEPTH_8U, 3);
int x_pos;
int y_pos;
int servoPosition = 90;
int servoOrientation = 0;
int servoPosition1=90;
int servoOrientation1=0;
char si[2];
int g=0;
while(1) {
g++;
p_imgOriginal = cvQueryFrame(p_capWebcam);
if(p_imgOriginal == NULL) {
getchar();
break;
}
cvCvtColor(p_imgOriginal, p_imgHSV, CV_BGR2HSV);
cvInRangeS(p_imgHSV,
cvScalar(H_MIN, S_MIN, V_MIN),
cvScalar(H_MAX, S_MAX, V_MAX),
p_imgProcessed);
```

p_strStorage = cvCreateMemStorage(0);

p_imgProcessed,

```
CV_GAUSSIAN,
9,
9);
p_seqCircles = cvHoughCircles(p_imgProcessed,
p_strStorage,
CV_HOUGH_GRADIENT,
2,
p_imgProcessed->height / 4,
100,
50,
10, //10
400);
if (p_seqCircles->total == 1)
{
p_fltXYRadius = (float*)cvGetSeqElem(p_seqCircles, 1);
x_pos=(p_fltXYRadius[0]-80)*40/(480);
y_pos=p_fltXYRadius[1]*40/480;
si[0]=x_pos+40;
si[1]=y_pos;
WriteFile(hSerial, si, strlen(si), &btsIO, NULL);
printf("%d ball position x = %d, y = %d, r = %f \n",g,x_pos,
y_pos,
p_fltXYRadius[2]);
fprintf(fp,"%d %d %d\n",g,x_pos,y_pos);
servoOrientation = 0;
```

```
cvCircle(p_imgOriginal,
cvPoint(cvRound(p_fltXYRadius[0]), cvRound(p_fltXYRadius[1])),
3,
CV_RGB(0,255,0),
CV_FILLED);
cvCircle(p_imgOriginal,
cvPoint(cvRound(p_fltXYRadius[0]), cvRound(p_fltXYRadius[1])),
cvRound(p_fltXYRadius[2]),
CV_RGB(255,0,0),
3);
}
cvShowImage("Original", p_imgOriginal);
cvShowImage("Processed", p_imgProcessed);
cvReleaseMemStorage(&p_strStorage);
charCheckForEscKey = cvWaitKey(10);
if(charCheckForEscKey == 27) break;
}
cvReleaseCapture(&p_capWebcam);
cvDestroyWindow("Original");
cvDestroyWindow("Processed");
```

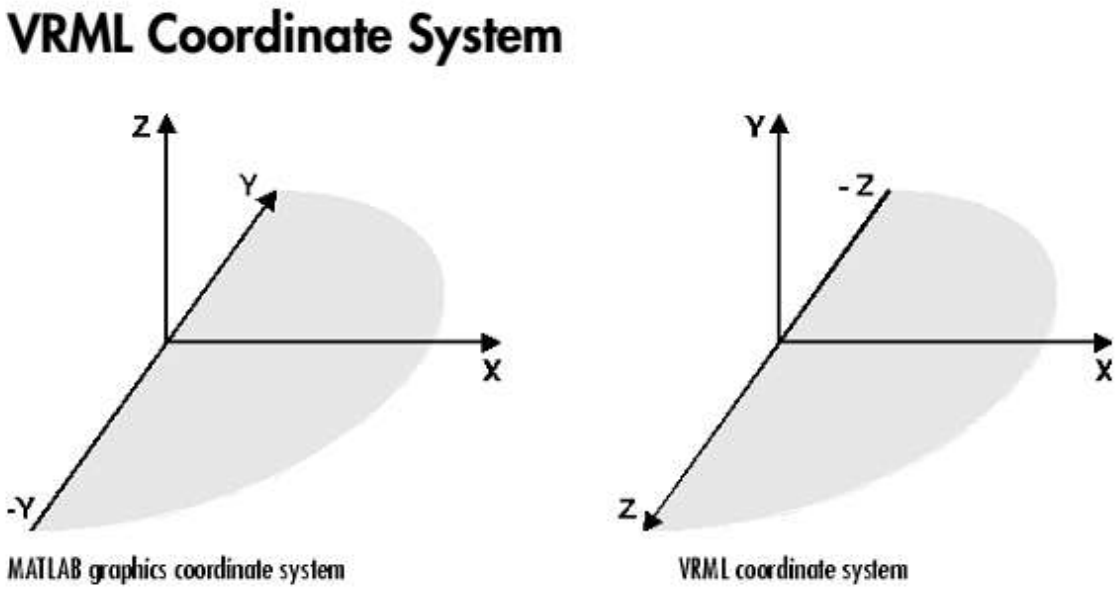
```
// This closes the Serial Port

CloseHandle(hSerial);

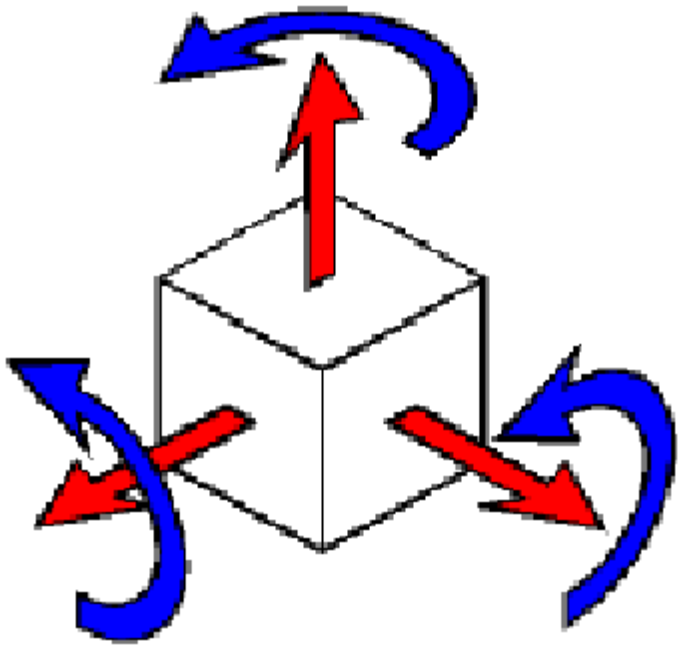
return(0);
```

Appendix c:

Brief description of VRML (Virtual Reality Modeling Language



The VRML coordinate system is different from the MATLAB and Aerospace Blockset™ coordinate systems. VRML uses the *world coordinate system*: the *y*-axis points upward and the *z*-axis places objects nearer or farther from the front of the screen. Understanding the coordinate system is important when you interact with different coordinate systems. SimMechanics uses the same coordinate system as VRML. Rotation angles — In VRML, rotation angles are defined using the *right-hand rule*. Imagine your right hand holding an axis while your thumb points in the direction of the axis toward its positive end. Your four remaining fingers point in a counterclockwise direction. This counterclockwise direction is the positive rotation angle of an object moving around that axis.



VRML method to represent a 3-D scene

VRML uses a hierarchical tree structure of objects (nodes) to describe a 3-D scene. Every node in the tree represents some functionality of the scene. There are many different types of nodes. Some of them are *shape nodes* (representing real 3-D objects), and some of them are *grouping nodes* used for holding child nodes. Here are some example nodes:

• Box — Represents a box in a scene.

• Transform — Defines position, scale, scale orientation, rotation, translation, and children of its subtree (grouping node).

• Material — Corresponds to material in a scene.

• DirectionalLight— Represents lighting in a scene.

• Fog — Allows you to modify the environment optical properties.

• ProximitySensor — Brings interactivity to VRML97. This node generates events when you enter, exit, and move within the defined region in space.

Child objects — In the hierarchical structure of a VRML file, specify the position and orientation of child objects relative to the parent object. The parent object has its local coordinate space defined by its own position and orientation. Moving the parent object also moves the child objects relative to the parent object.

Measurement units — All lengths and distances are measured in *meters*, and all angles are measured in *radians*.