Callum Owen-Bridge, 15002504

Electronic Engineering Coursework Reports

March 2018

Appendix

Laboratory Report: Digital Signal Processing

Callum Owen-Bridge 15002504

Digital Signal Processing

Introduction

The purpose of this lab is to understand digital processing signals, this will be achieved through laboratory exercises and research. In this lab different types of digital signals will be displayed and manipulated using MATLAB software. A signal filter will be explored, and its function explained.

Equipment and materials

The equipment used for this lab was software called MATLAB at version: 2017b.

Procedure

Using MATLAB, a discrete-time step waveform was adapted to produce a plot representing a discrete-time ramp waveform.

A sample size of 60 was used starting from sample position -10 up to 50. The step position started at sample number 0.

To adapt this graph to produce a discrete-time ramp, the signal would have to be multiplied by the unit step, to set samples at steps less than zero to zero. A dot operator was used, in MATLAB to multiply element-wise, as shown in figure 1; the dot operator is used on line 9. This dot operator was used to make negative sample numbers zero. Otherwise, the graph produced will contain a ramp in both the negative and positive sides of the y-axis.

```
1  sampleStart=-10;
2  sampleEnd=50;
3  stepPosition=0;
4
5  n = sampleStart:sampleEnd;
6
7  unit_step = n>=stepPosition;
8
9  ramp = unit_step.*n;
10
11 stem(n,ramp);
12 axis([sampleStart,sampleEnd,-2,60]);
13 xlabel('Sample Number');
14 ylabel('Amplitude');
```

Figure 1 a MATLAB script for a ramp signal

Delayed discrete-time graph

A MATLAB script was then created to produce a delayed discrete-time signal, as shown in figure 2. To delay the signal the samples used was copied into another array called y. The delay was set to samples 20. The delay had to be incremented by one, due to MATLAB arrays being indexed from one and not zero. The original signal and delayed signal were then plotted together for comparison.

```
1  sampleStart=-10;
2  sampleEnd=50;
3  stepPosition=0;
4  delay = 20;
5  n = sampleStart:sampleEnd;
6  unit_step = n>=stepPosition;
7  ramp = unit_step.*n;
8
9  subplot(2,1,1);
10 stem(n,ramp);
11 title('Original signal');
12 axis([sampleStart,sampleEnd,-2,60]);
13 xlabel('Sample Number');
14 ylabel('Amplitude');
15
16 y = zeros(1, length(n));
17 for z = delay+1:length(n)
18     y(z) = ramp(z-delay);
19 end
20
21 subplot(2,1,2);
22 stem(n,y);
23 title('delayed signal');
24 axis([sampleStart,sampleEnd,-2,60]);
25 xlabel('Sample Number');
26 ylabel('Amplitude');
```

Figure 2 a MATLAB script for delayed signal

Signal Filter

A filter was then implemented in MATLAB and investigated. It was excited by an impulse input and impulse response, so that the output was the impulse response. The program was adapted to implement a filter of the form shown in figure 3. The y value in the figure 3 equation represents the output, x represents the input and n is the number of samples. A delay of 100 and 300 samples was added and the sound produced was investigated. The sound was heard using the function in figure 4.

$$y(n) = 0.5\,x(n) + 0.5\,y(n-1)$$

Figure 3 filter equation

```
soundsc(y,8000);
```

Figure 4 sound function

Results

Figure 5 shows the result of adapting the discrete-time step into a discrete-time ramp. From sample 10 onwards the value for each sample increases producing a ramp. The x-axis is the number of samples and the y-axis is the amplitude of each sample.

Figure 6 shows the result of delaying a discrete-time signal. In the delayed signal graph, the signal was delayed by 20 samples, creating a shorter signal. The x-axis is the number of samples and the y-axis is the amplitude of each sample.

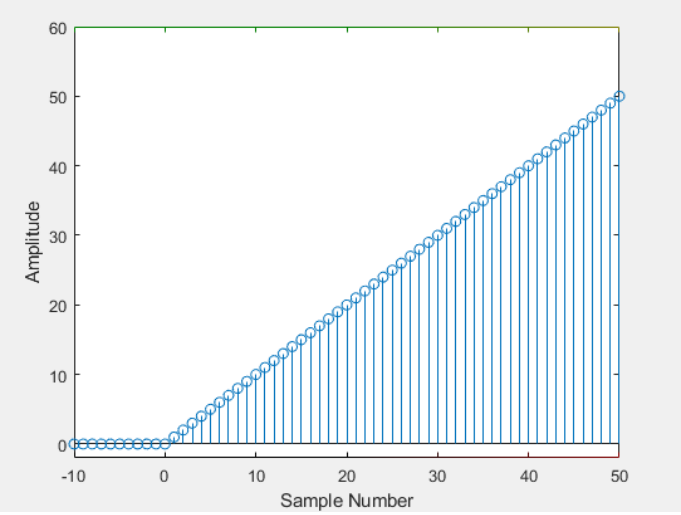


Figure 5 unit ramp signal

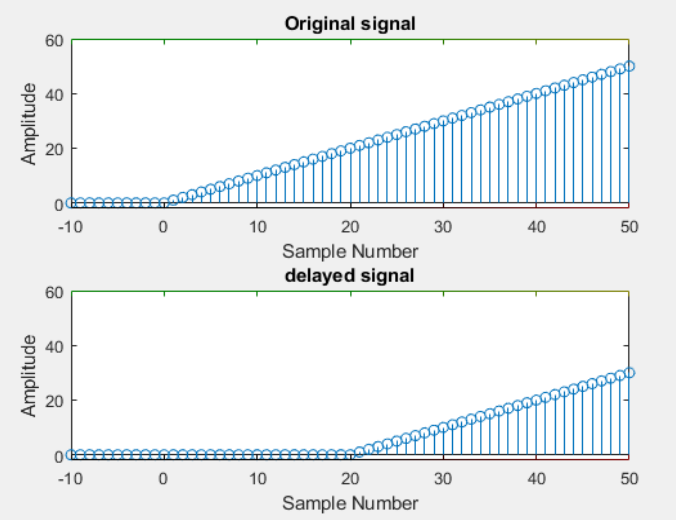


Figure 6 comparison of two unit-ramp signal graphs

```
y(samplesCounter)=0.5*x(samplesCounter)+0.5*x(samplesCounter-1);
```

Figure 7 impulse response equation

Using the impulse response equation shown in figure 7, produced a single short duration “ping” sound, shown in figure 9. Using the equation in figure 8, produced a sound similar to the first equation except the ping lasted longer. Increasing the delay to 300 samples, created a longer duration “ping” sound, as shown in figure 10.

```
y(samplesCounter)=0.5*x(samplesCounter)+0.5*y(samplesCounter-100);
```

Figure 8 impulse response equation with increased sample

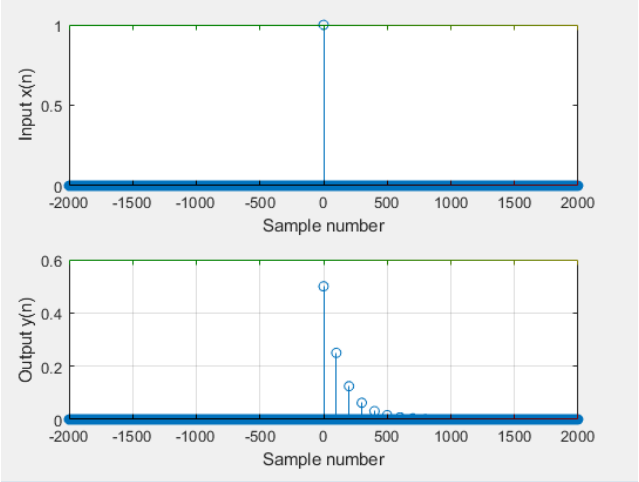


Figure 9 a sound filter with a 100 sample delay

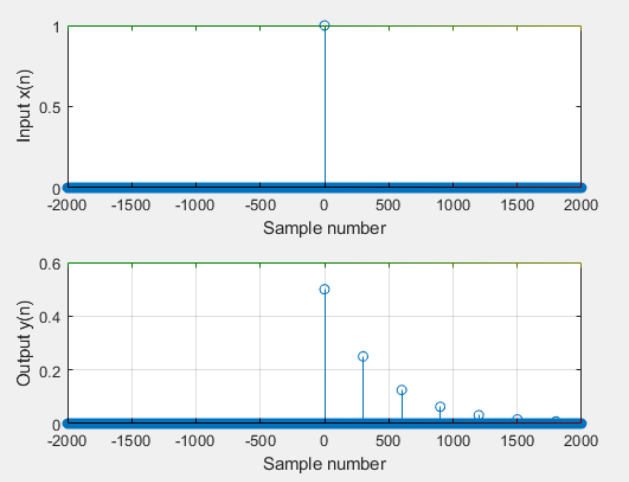


Figure 10 a sound filter with a 300 sample

Research

Digital Signal Processing is used in many applications such as audio signal processing, audio compression, biomedicine and telecommunication. Digital signal processing, are signals in a digital representation and using digital processors to modify and analyse signals. The purpose of processing digital signals could be to remove interferences from a signal or to transform a signal into a required form. (Ifeachor, E. and Jervis, P., 2018).

Digital Signal Processing uses a combination mathematics and analogue signals to achieve its results. the combination of modern technology and digital signal processing allows computers to become more advanced as they can understand human speech and are capable of translations. (D. Stearns, S. and R. Hush, D., 2016).

Conclusion

To conclude, an understanding of digital signal processing has been gained. The results produced a discrete-time ramp signal in graph form, by adapting a discrete-time ramp signal. Adapting discrete-time signals to produce a different signal type is called transforming, this is done when a different form of the discrete-time signal is required.

A delayed signal was produced by creating an array of delayed samples, the comparison of the two plots shows a delay of 20 samples was made. By delaying the signal in the impulse response equation, the sound produced echoes and increasing the delay increased the delay time of the echo. The impulse response produced by this equation is the output created by and impulse signal.

The research made shown that digital signal processing is a widely used and required technology, to achieve advancements in different applications.

References

D. Stearns, S. and R. Hush, D. (2016). *Digital Signal Processing with Examples in MATLAB®, Second Edition*. [online] Google Books. Available at: https://books.google.co.uk/books?hl=en&lr=&id=AWXRBQAAQBAJ&oi=fnd&pg=PP1&dq=digital+signal+processing+impulse+response&ots=8KAPOfbe33&sig=GEGzubIqJeIBcIugFNVGE4FLvkU#v=onepage&q=digital%20signal%20processing%20impulse%20response&f=false [Accessed 10 Mar. 2018].

Ifeachor, E. and Jervis, P. (2001). *Digital Signal Processing*. [online] Google Books. Available at: https://books.google.co.uk/books?hl=en&lr=&id=slHJIO9xmcEC&oi=fnd&pg=PR15&dq=where+is+digital+signal+processing+used&ots=A_eFB6elu6&sig=ov8EsWmbXFdot6O0hMXAIluVnKw#v=onepage&q=where%20is%20digital%20signal%20processing%20used&f=false [Accessed 10 Mar. 2018].

Laboratory Report: Control Theory

Callum Owen-Bridge 15002504

Control Theory

Introduction

The purpose of this lab is to gain an understanding of control theory, state equations, and steady state and transient points of a graph. Control theory is a system which controls the operation of another system to gain a desired output. A control system is made up of either a single device or multiple devices to manage or regulate the behaviour of another system. A control system can represent an open loop or closed loop system. The behaviour of a control system can be represented as a graph, containing a transient and steady state. These states can be tuned to reduce errors present in the states.

Equipment and materials

The equipment used for this lab was software called MATLAB at version: 2017b.

Procedure

State Space representation

This is a mathematical model of a physical system, consisting of a group of inputs, an output and state variables in first order differential equations. To represent a linear system in state space, the state-space model equations are required, as shown in equation 1. In figure 1 X is known as the state vector, Y is known as the output vector, U is the control vector, A is the state matrix, B is the input matrix, C is the output matrix and D is the feedthrough matric.

$$\dot{x} = Ax + Bu$$
$$y = Cx + Du$$

Equation 1 shows the state-space model

Newton's law will be used as an example of how to represent a system in state space. The state variables are mass position (x_1) and velocity (x_2). State variables are variables which can be altered to produce a different output and represent a system in its entire state. Acceleration is represented as $\ddot{x}$.

$x_1 = x$ and $x_2 = \dot{x} = \dot{x}_1$
Therefore, $\dot{x}_2 = \ddot{x} = F/M$

Newton's second law can be represented as: $F = M\ddot{x}$

Figure 2 shows Newton's second law in matrix form, this is used by MATLAB. Figure 3 shows a conversion of a state-space model to state-space form equations.

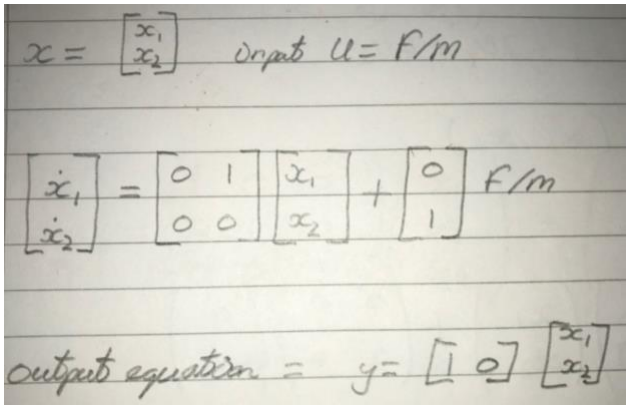


Figure 2 state-space model

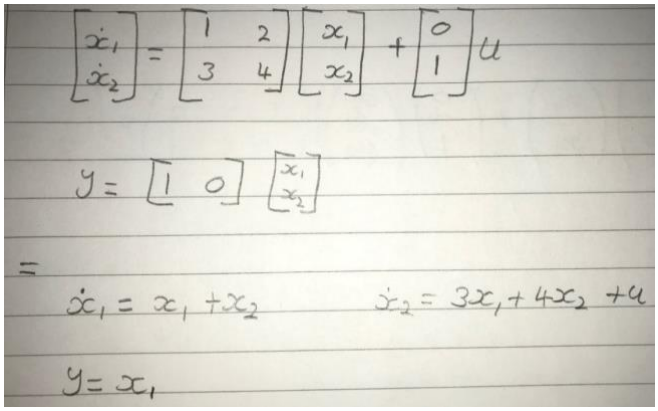
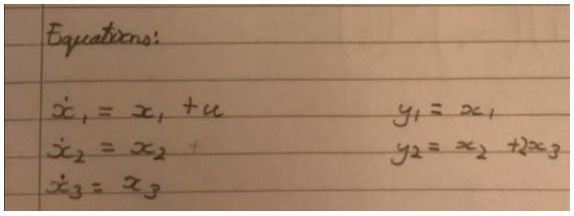
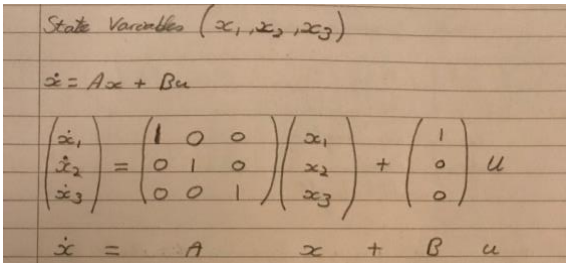


Figure 3 state-space model to state-space form conversion

State-Space model

To convert equations from a state-space form to a state-space model the following steps are made:





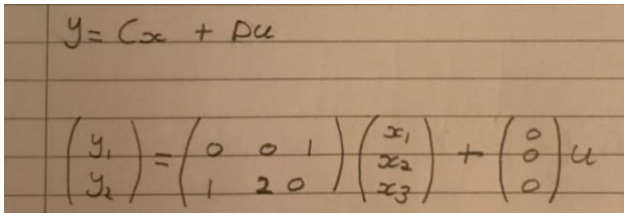


Figure 4 conversion from state-space form equations to a state-space model

Step1:
Compare to the state-space model equations to the equations, shown in figure 1, for both input and output equations. This is shown in figure 4.

Step2:
To find the matrix dimensions, the width of the first matrix is one and three in height as there are three equations. The second matrix is three in width, this is due to the three in height of the third matrix, and three in height. The other matrices have three rows and one column as shown in figure 4.

Step3:
The values within the matrices are dependent on what is within the equation. For example, for the first equation only x_1 and U is present so the top row for the second matrix will be 1 0 0, as these values are then multiplied by each row value in the third matrix.

State-Space representation example

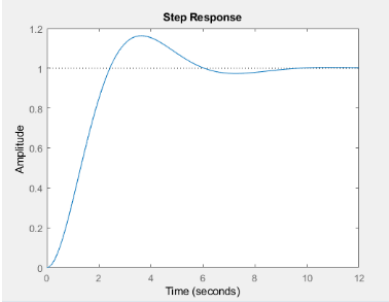
The following state space model for a spring damper system was then defined in MATLAB using a step function, the result of this is shown in figure 5.

$$\dot{x}_1 = x_2$$

$$\dot{x}_2 = -\frac{k}{m}x_1 - \frac{c}{m}x_2 + \frac{1}{m}u$$

$$\begin{bmatrix}\dot{x}_1 \\ \dot{x}_2\end{bmatrix} = \begin{bmatrix}0 & 1 \\ -\frac{k}{m} & -\frac{c}{m}\end{bmatrix}\begin{bmatrix}x_1 \\ x_2\end{bmatrix} + \begin{bmatrix}0 \\ \frac{1}{m}\end{bmatrix}u$$

$$y = [1 \quad 0]\begin{bmatrix}x_1 \\ x_2\end{bmatrix}$$




```
c = 1;
k = 1;
m = 1;

A = [0, 1; -(k/m), -(c/m)];
B = [0; 1/m];
C = [1, 0];
D = [0];

ss_model = ss(A,B,C,D);

step(ss_model);
```

Figure 5 a simulated state space model

Figures 6 to 11 shows the effect on the curve when values m, c and k are altered. As the mass (m) value increases, the transient section of the curve is less stable as shown in figure 6. So, forming a higher peak compared to the one shown in figure 5, so increasing the rise and settling time. When the mass value is decreased the transient section becomes more stable and forms less peaks, as shown in figure 7, reducing rise and settling times significantly. When the damping constant (c) value is increased, it causes the transient section of the curve to smooth and removes errors in the steady state section shown in figure 8. When the c value is decreased, both the steady state and transient sections of the graph become unstable and errors are increased, as shown in figure 9. When the spring constant (k) value is increased, both the steady state and transient sections of the graph become unstable and errors are increased however the values are below one. When the value of k is reduced, the transient and steady space section becomes more stable and forms less peaks however the steady state value plateaus at two. In figure 12 the damping constant has value five, mass has value one and spring constant has value one. This produced a curve with no error in steady state a steady rise time, small settling time and no overshoot in transient state.

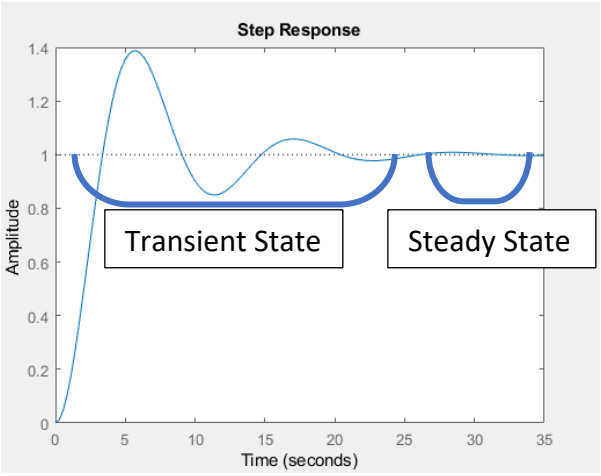


Figure 6 a simulated state space model with m value increased to 3

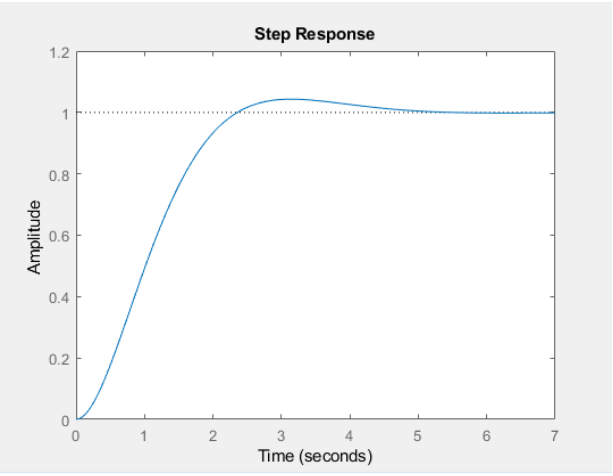


Figure 7 a simulated state space model with m value decreased to 0.5

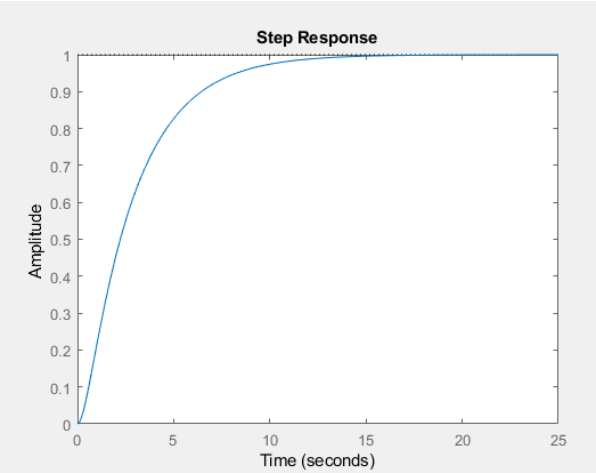


Figure 8 a simulated state space model with c value increased to 3

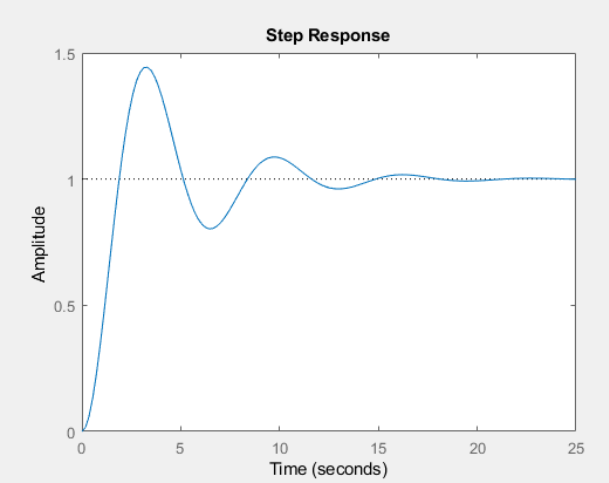


Figure 9 a simulated state space model with c value decreased to 0.5

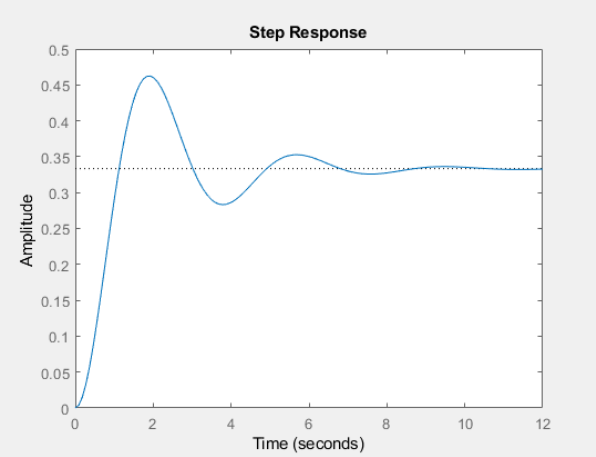


Figure 10 a simulated state space model with k value increased to 3

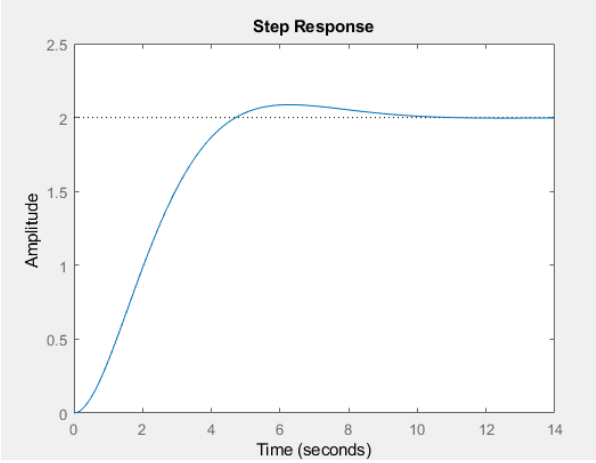


Figure 11 a simulated state space model with k value decreased to 0.5

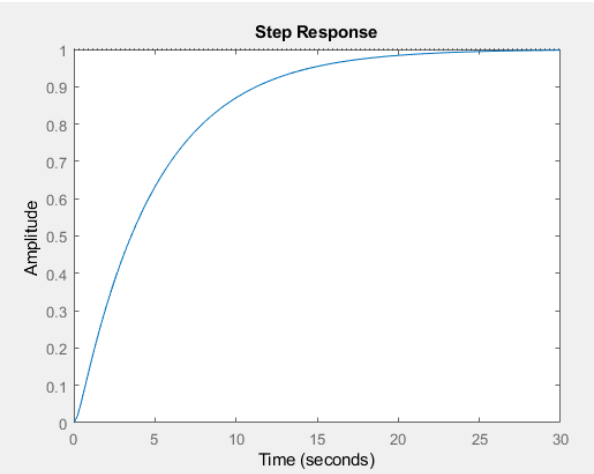


Figure 12 a simulated state space model with c (5), k (1) and m (1) values altered

Conclusion

To conclude, an understanding of control theory has been gained. Newton's second law was used to show how a control system can alter the output of a system and how it is converted in state-space form and state-space model, as shown in figures 2,3 and 4. A state space model for a spring damper system was then simulated in the MATLAB programming language (The MATLAB Inc, MATLAB 2017b, to show how control theory can change a system to gain a desired output. The values for damping constant, mass and spring constant where altered to adjust the transient section of the graph, so that the first spike was reduced. It was found that a spring constant of one, mass of one and damping constant of five produced a stable graph, with no errors in steady state and a steady curve in transient state, shown in figure 12. As a desired system would have a steady rise in transient state, without over shoot, with the rise coming to a plateau producing an error free steady state.

A PID controller could be used to reduce the error in both transient and steady states. PID is a controller which alters a system through use of proportional, integral and derivative mathematical functions. PID controllers allow for a more specific output control, an increase in proportional control decreases rise time, increases overshoot in transient state, creates a small change in settling time and decreases steady state error. An increase in integral control causes a decrease in rise time, increase in transient state overshoot, an increase in settling time and steady state error is removed. An increase in the derivative controller causes a small change in rise time, decreases transient state over shoot, reduced settling time and a small change in steady state error. Integral is used to remove system offset, derivative gain is used to increase system processing speed and proportional gain is used as the main tuning parameter as it is directly proportional to system error.

For future improvements, using Simulink tool in MATLAB will improve the experiment as it will allow for simulations to view system performance by using a loop feedback system.

Laboratory Report: Mechatronics - Basics

Callum Owen-Bridge 15002504

Mechatronics - Basics

Introduction

The purpose of this lab is to gain an understanding of robotics and the basic components to a robot. This will be done through research. Classification of robots and their systems will be analysed and discussed. The types of actuators robots use will be analysed.

Literature Review

Robot Characteristics for classification

There are six characteristics to classify robots:

1. **Mobility:** the movement of a robot and its type of movement, e.g. using tracks or wheels.
2. **Architecture:** the structure of a robot and how it is constructed. The materials used to make the robot. The architecture can influence what the robot can do. The number of degrees of freedom and movement and the robot's workspace is concerned in the architecture.
3. **Typology:** the type of robot; hybrid, serial, modular or parallel.
4. **Performance and velocity:** speed and efficiency that a robot can complete a task in. the precision of the robot. Varied velocities and precisions control what tasks a robot can do.
5. **Trajectory:** the robot workspace. Where the robot can move and how it can get to that position. Stop to stop, point to point, controlled or continuous. This includes the robot's reachable workspace which is where its end effector can arrive in at least one orientation and its dexterous workspace where its end effector can arrive in in any orientation. (Secco, 2018). Figure 1 shows an exmaple of a ARABA robot work space. The shape and volume of a robot workspace are the most important aspects,
6. **Control:** how the robot is controlled. Open- loop, closed-loop or adaptive control.

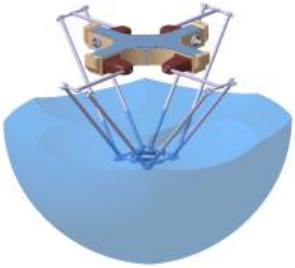


Figure 1 robotic work space

Robot System

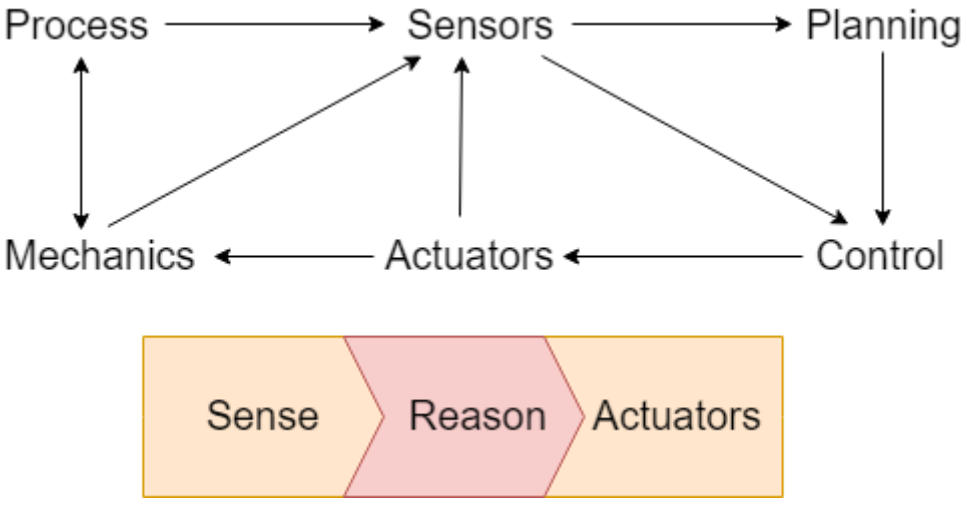


Figure 2 robotic system

Figure 2 shows the main elements of a robotic system. These elements are sense, reason and actuators. Sense, represents the robot's sensors and the actuators represents the robots motors which allow it to move. Reason represents the programming of the robot, this takes in data from the sensors and manipulates the actuators to perform an action.

Sensor System

Figure 3 shows a diagram of how a robotic sensor works from measuring physical entities, processing the signal and forming a perception. The transductor is a term used for both sensors and actuators. The transductor senses a physical entity such as pressure. The transductor transfers a signal which is processed by a computer. The computer then forms a perception from this signal and performs an action. A sensor system uses sensors from both internal and external positions of the robot. (Dutta Professor, D., 2018).

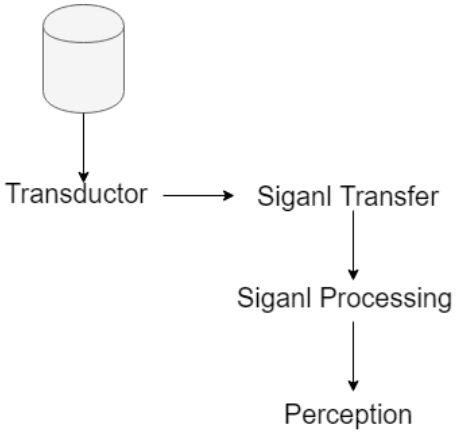


Figure 3 robotic sensory system

Figure 4 two armed robot, F. Zacharias, C. Borst and G. Hirzinger, 2007.

Actuators

Rigid Actuator: stiff, heavy and noisy actuators. They are more precise and stable than soft actuators. Rigid actuators are expensive due to the use of electric motors that run at high velocities therefore requiring high amounts of electric power. There are four types: electrical (such as stepper motors), pneumatic (air pressure), hydraulic (fluid pressure) and advanced actuators (ultrasonic motors and artificial muscles), (Dutta Professor, D., 2018). These types of actuators provide robots with a rigid movement. Figure 4 is an example of a robot using rigid actuators to be able to grab and move objects.

Soft Actuator: soft actuators are made up of flexible materials, which allow it to move freely. These actuators are much lighter, less noisy and produce little friction and are bioinspired. Soft actuators use either air or liquid to perform movements rather than solid components. Soft actuators contain sensors which allows it to become a closed loop system or pressure sensors to allow it to become an open loop system. Figure 5 shows an example of soft actuators being used in robots. Robot B inn figure 5 is inspired by the tentacle of an octopus. These robots are usually made up of soft materials such as elastomers, silicone rubber and biodegradable materials. These materials allow the robots to

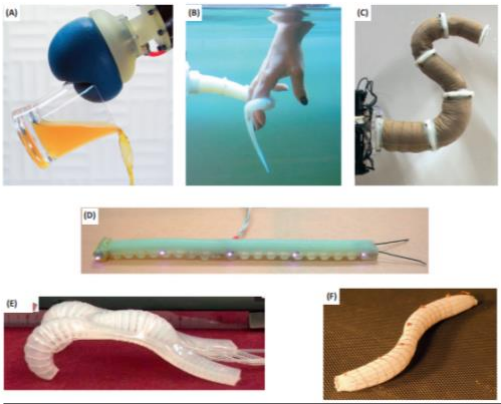


Figure 5 soft actuators (Kim, S., Laschi, C. and Trimmer, B., 2013)

be flexible and have fluid motion. These robots are able to move in multiple directions and angles depending on its structure and materials used, (Kim, S., Laschi, C. and Trimmer, B., 2013).

Soft actuators have different motions such as bending, linear extension and contraction. However, soft actuators are complicated to manufacture, programme their motion and has possibilities of tearing, compared to rigid actuators. (Chen et al., 2018)

Conclusion

There are six characteristics of robotics which help classify a robot, the robots workspace and the way it is controlled determines the task a robot is able to do. The robot system is a simple concept on how a robot performs a task and reacts to its environment. The use of the sensory system for robots helps the robot to react to its environment accordingly. There are two types of actuators a robot can use that provides the robot with different functions and allows it to complete certain tasks, such as moving delicate or heavy objects. Robot design is being inspired by biology to help them to perform unusual tasks.

References

Chen, Y., Le, S., Tan, Q., Lau, O., Wan, F. and Song, C. (2018). *A reconfigurable hybrid actuator with rigid and soft components*.

Dutta Professor, D. (2018). *Sensors and Actuators* [PowerPoint presentation] Iitk.ac.in. Available at: https://www.iitk.ac.in/tkic/workshop/robotics/ppt/day2/Sensors%20and%20Actuators.pdf [Accessed 26 Feb. 2018].

Kim, S., Laschi, C. and Trimmer, B. (2013). Soft robotics: a bioinspired evolution in robotics. *Trends in Biotechnology*, [online] 31(5), pp.287-294. Available at: https://www.sciencedirect.com/science/article/pii/S0167779913000632 [Accessed 26 Feb. 2018].

Martínez, J. (2013). *On the basis of workspaces of robotic manipulators (Part 1)*. [online] Engineer JaU. Available at: https://engineerjau.wordpress.com/2013/07/07/on-the-basis-of-workspaces-of-robotic-manipulators-part-1/ [Accessed 26 Feb. 2018].

Secco, E. (2018). *Robotics* [PowerPoint presentation] [Accessed: 27 Feb. 2018].

Zacharias, F., Borst, C. and Hirzinger, G. (2007). *Capturing robot workspace structure: representing robot capabilities - IEEE Conference Publication*. [online] Ieeexplore.ieee.org. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4399105 [Accessed 26 Feb. 2018].

Laboratory Report: Mechatronics – Mechanical Reasoning

Callum Owen-Bridge 15002504

Mechatronics – Mechanical Reasoning

Introduction

The purpose of this lab is to gain an understanding of robotics and mechanical reasoning. This will be done through research. Degrees of freedom and degrees of mobility will be analysed and discussed with examples. Types of robotic joints will be reviewed with examples of their purpose. Classification for robotics by degrees of freedom and degrees of mobility will be discussed with examples of each type. Then, types of robots will be analysed and discussed with examples to show the purpose of each.

Literature Review

Degrees of Mobility and Degrees of Freedom

Degrees of freedom: number of independent variables related to joint positions. The degrees of freedom are the number of actuators used.

Degrees of mobility: the number of planes an object can move in.
The degrees of mobility for a rigid body in plane is three (x, y, z) and six in space (x, y, z, ρ, θ, φ).

Operative and Gruebler Criterian

Mechanism (M): chain made up of joints and links.

Operative criterion is where M is the number of joint links.

The Gruebler Criterian is the number of links (N) in a robot

M = 6 * N – (number of constraints)

Gruebler says

M = 3 * (n-1) – 2 * (number of joints with one degree of mobility) – (number of joints with two degrees of mobility).

Robotic Joints

Rotary Joints: are joints which allow the link to rotate.

Prismatic joints: is a joint which is telescopic, therefore moves in and out of the joint. These joints are orthogonal to cylindrical joints. (Palmieri, Palpacelli and Carbonari, 2018). Figure 1 is an example of a robot using a telescopic joint.

Spherical joints: is a ball and socket joint which allows the link to move freely (Palmieri, Palpacelli and Carbonari, 2018). Figure 1 is an example of a robot using spherical joints

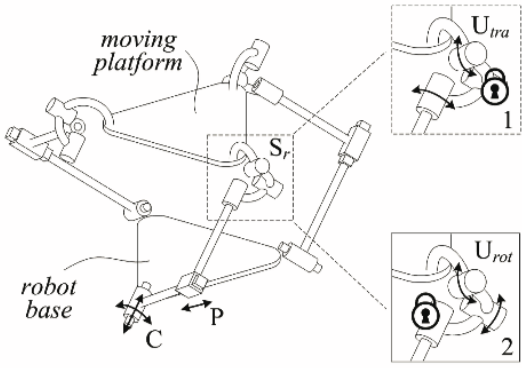


Figure 1 robotic joint diagram

Holonomic, Non-Holonomic and Redundant

Holonomic: the robot degrees of mobility are equal to the number of degrees of freedom the robot can control.

Non-Holonomic: the number of degrees of freedom the robot can control is less than the number of degrees of mobility.

Redundant Robot: the number of degrees of mobility is less than the number of degrees of freedom which the robot can control.

Robotic Types

There are many types of robots which can. Be defined by their joint movements below are examples of some robots defined by their joint movement type.

Translation Translation Translation (X Y Z):

- A basic arm architecture.
- Linear transformation between Cartesian and joint space.
- Movements are decoupled.
- Example is a cargo crane.
- Cubic workspace
- Figure 2 is an example of a Cartesian robot, it is a crane structure.

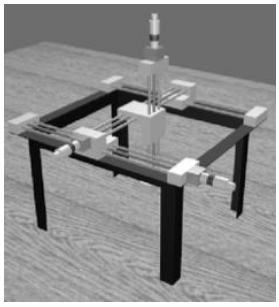


Figure 2 Cartesian Robot (Mohammad Khan, Arshad and Ahmad Choudhry, 2018)

Rotation Translation Translation:

- Has one rotational joint with two fixed prismatic joints.
- Transformation is linear in only Z between Cartesian and joint space.
- This type is less expensive
- Cylindrical workspace

Rotation Rotation Translation

- Contains two rotational joints and a fixed prismatic joint
- Example is a welding robotic arm.
- Polar workspace
- No linear transformation between Cartesian and joint space.

Rotation Rotation Rotation:

- Contains three rotational joints
- Example is a human arm
- This robot is less expensive due to the rotational joints
- Articulated workspace, spherical.
- No linear transformation between Cartesian and joint space.

SCARA (Selective Compliant Assembly Robot Arm):

- Contains two rotational joints and a fixed prismatic joint
- Works from a top to bottom perspective
- Transformation is linear in only Z between Cartesian and joint space.
- All the motors are within the base
- Figure 3 shows an exmaple of a SCARA robot, it id able to perform tasks with speed and precison. (Urrea, Cortés and Pascal, 2018)

Figure 3 SCARA Robot (Urrea, Cortés and Pascal, 2018)

Kinematics

Figure 4 shows a diagram of direct and inverse kinematics. Kinematics is dependent on homogeneous transformation.

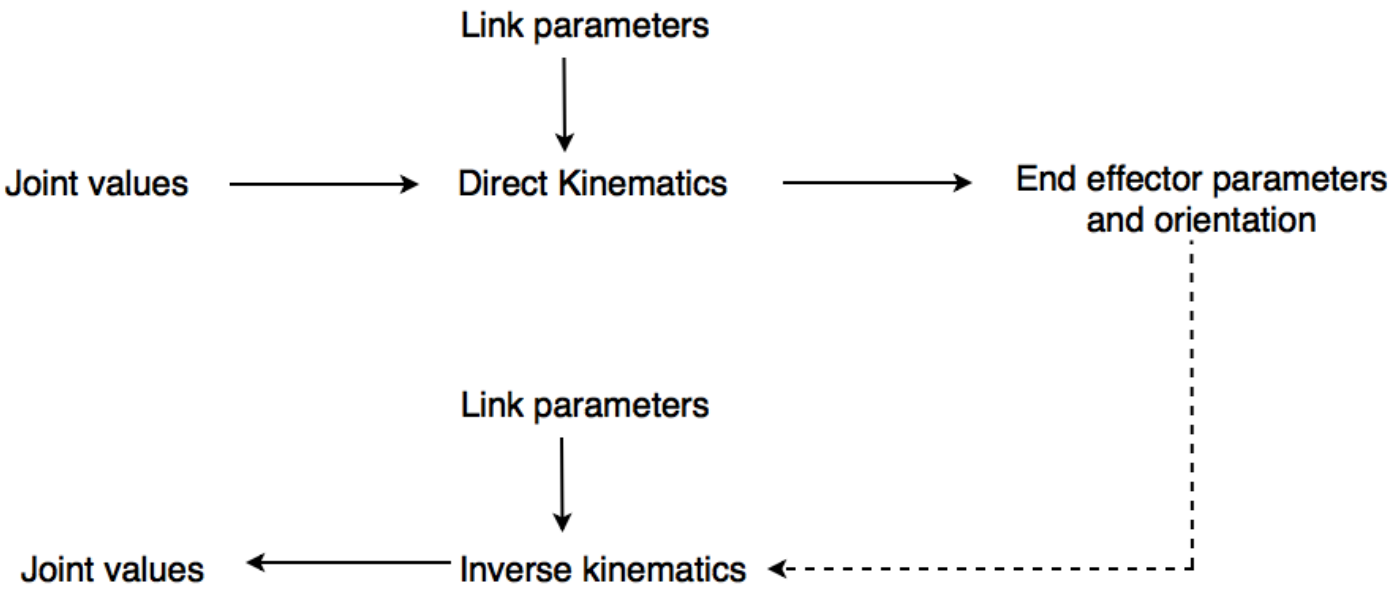


Figure 4 kinematic system

Inverse kinematics: This Is often used to control the movement of a rigid bodied robot (R. Buss, 2018). Inverse kinematics produces joint values by using the end effector position and orientation with link parameters as shown in figure 4.

Direct Kinematics: Produces end effector positions and orientation by using joint values and link parameters as shown in figure 4. This is to solve the Cartesian end effector position and orientation using joint displacements (Spong, Hutchinson and Vidyasagar, 2005).

Conclusion

There are two characteristics to classify a robot by its joints and movement, known as degrees of freedom and degrees of movement. There are many types of robotic joints that allow the robot to complete different tasks, a combination of the joints can allow a robot to become more inspired by nature and perform more natural, precise and smooth movements. There are many types of robotic arms which all perform different functions due to their specific workspaces, this controls what a robot can be used for, as a combination can allow a robot to perform more precise tasks. The kinematics of a robot helps to solve the positions of the robot's end effector as well as the position of the robot links and joints.

References

Mohammad Khan, T., Arshad, M. and Ahmad Choudhry, M. (2018). *Modeling and Control of Cartesian Robot Manipulator - IEEE Conference Publication*. [online] Ieeexplore.ieee.org. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4133519 [Accessed 27 Feb. 2018].

Palmieri, G., Palpacelli, M. and Carbonari, L. (2018). *A lockable spherical joint for robotic applications - IEEE Conference Publication*. [online] Ieeexplore.ieee.org. Available at: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6935581 [Accessed 27 Feb. 2018].

R. Buss, S. (2018). *Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods*. [online] Math.ucsd.edu. Available at: http://math.ucsd.edu/~sbuss/ResearchWeb/ikmethods/iksurvey.pdf [Accessed 27 Feb. 2018].

Spong, M., Hutchinson, S. and Vidyasagar, M. (2005). *Robot modeling and control*. Hoboken, NJ: Wiley.

Urrea, C., Cortés, J. and Pascal, J. (2018). *Design, construction and control of a SCARA manipulator with 6 degrees of freedom*. [online] science direct. Available at: https://www.sciencedirect.com/science/article/pii/S1665642316300931 [Accessed 27 Feb. 2018].