Bashundhara
Exercise Book
Write Your Future

ANIKA ISLAM (21101293)

CSE461

Quiz — 15

Attendance — 5

Mid — 20

Final — 35

Lab — 15

Project — 10

LECTURE 1

SUNDAY

DATE: 24/09/23

* Robot — An emboided agent programmed to perform physical tasks.

* Robotics — is a branch of engineering and computer science that deals with the design, construction, operation & use of robots.

* Robots need to be embodied, which AI does not need.

How Robots are used across industries:-

Industrial, Farming and Agriculture, Healthcare, Logistics, Family Robot

LECTURE 2

TUESDAY DATE: 26/09/23

Laws of Robotics:-

A robot must: ① not harm human beings
② always obey human beings
③ protect from harm
④ have a kill switch

* Robots should be used in industries which has many employee and including a robot will make their work easier instead of leading to jobless condition.

* Robots do Dull, Dirty, Difficult, and Dangerous Jobs.

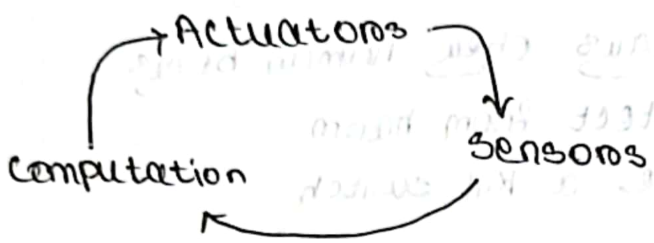
Thumb Rules on the decision of a Robot uses

① consider the four D of Robotics
② not leave human jobless
③ asking whether you can find people who are willing to do the job.
④ make short-term and long-term economic sense

Some special vehicles:-
① Uncrewed vehicle - operates without human presence
② Remote control vehicle (RC) - controlled by remote
③ Unmanned ground vehicle (UGV)
④ Unmanned aerial vehicle (UAV)
⑤ Unmanned surface vehicle (SUV) - operated on water
⑥ Remotely operated unmanned vehicle: (ROUV)

⑦ Autonomous underwater vehicle (AUV)

Anatomy of a robotic system



"Sense — Think — Act"

3 Primitives

AI Primitives within an Agent

* Robot environment sense করে, plan করে to take the necessary action and then movement করে। AI দিয়ে সব learn ও করে।

Hierarchical Paradigm

→ Sense → Plan → Act

Robot Primitive	Input	Output
SENSE	Sensor data	Sensed Info
PLAN	Info (sensed and/or cognitive)	Directives
ACT	directives	Actuator commands

* একটা primitive এর output next primitive এর input.

* optimal but uncertainty handle করতে পারে না

Reactive paradigm

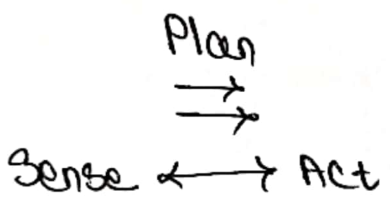
Act ⟷ Sense

Robot Primitive	Input	Output
SENSE	Sensor Data →	Sensed Info
PLAN		
ACT	Sensed info →	Actuator commands

* Robot sense করে environment, optimal path খোঁজে, plan না করে, actuator কে বলে to generate movement

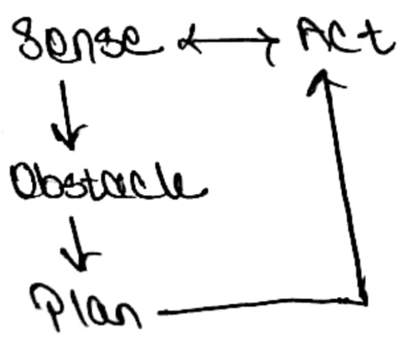
* uncertainty handle করে, but optimal না

Hybrid deliberative/reactive paradigm



* Sense and Act parallely চলে, obstacle না পেলে optical path follow করবে – works a reactive paradigm.

Obstacle পেলে, path change করে by following a new plan – works a hierarchy paradigm



* যখন optimal & uncertainty both handle করা লাগে, we use hybrid.

* Robot needs both local & global model
local ↓ reactive
global ↓ deliberative

* Part of hybrid paradigm can be called as local & global model

Motion subsystem — Act (carry out the decisions)

Recognition subsystem — Sense (sense environment)

Control subsystem — Think (make decisions based on environment)

LECTURE 3

TUESDAY DATE: 03/10/23

* Motion –

* Recognition – Sensor: analogue এ environment থেকে sense করে input নেয়

– ADC: Since computer takes digital input, analogue signal from sensor is converted into digital by ADC.

* Control – Digital controller: sends digital signal to the mechanical parts to operate accordingly in the environment around it.

– DAC: However, robot takes analogue input so the digital signal from digital controller is converted into analogue by DAC.

– Amplifier: This analogue signal is amplified.

Recognition Subsystem

Sensor
- → Active – external power নেওয়া লাগে
- → Passive – external power নেওয়া লাগে না নিজে থেকে কাজ করতে পারে

Ultrasonic Sensor

- converts electrical energy to acoustic wave
- a microcontroller is used to communicated with ultrasonic sensor.

- A trigger is send by microcontroller to the ultrasonic sensor.
- The trigger wave falls, when the ultrasonic sensor starts to generate signal. 8 signals are generated. Timer is ON during this period.
- After 8 signals are generated, the echo wave reaches peak and remains in this high value upto the reflected signal starts to receive by the ultrasonic sensor. At this moment, the echo wave falls back to 0.
- After all the reflected signals is received by the ultrasonic sensor, the timer turns OFF.

$V \rightarrow$ speed of sound

$$S = V \frac{t}{2}$$

$t \rightarrow$ time between sending & receiving signal

$S \rightarrow$ distance

Infrared sensor

- An electronic device that can detect and measure IR radiation

Infrared sensor
- → Active : receiver & emitter, থাকে
- → Passive : receiver থাকে
  sensor এর কাছে আসলে intensity বাড়ে

- Transmitter sends IR radiation to object body and the radiation is reflected back and received by IR Receiver.
- Time is measured.

$$S = \frac{V}{\frac{t}{2}}$$

S → distance

V → speed of light

t → time of sending & receiving radiation

Lidar

- Laser generates an optical pulse. (laser)
- Lidar spins
- Sends and receives pulse
- High speed counter measures time of flight from start and return pulse.
- Measurement is converted into distance.
- পুরো environment এর distance বের করা possible by moving the lidar from place to place + spinning it.
- vertical & horizontal lidars use করলে 3D mapping করতে পারি

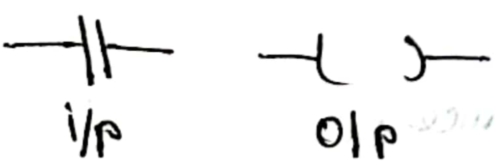
Microcontroller

- microcontroller এর ভিতর microprocessor থাকে + memory, i/p, o/p peripherals integrated থাকে
- e.g arduino

Arduino

Programmable Logic Controller (PLC)

- অনেক i/p and o/p ports থাকে
- centrally control করা যায়
- hundreds of thousands of sensors থাকে
- massive amount of things is handled
- main application: Industry (power plant)
- Ladder diagram use করে PLC তে programming করে



Single Board Computer

- Portable
- Energy efficient
- Processing power & storage কম থাকে in microprocessor so we use Rasberry Pi instead.

LECTURE 4

SUNDAY

DATE: 08/10/23

Motion Subsystem

Manipulator :- অনেক গুলো rigid link থাকে

- Rigid links are connected to power / passive joint

Power joint:
- motion এর জন্য use করি
- motor থাকে

Passive joint:
- motor থাকে না
- movement হয় না
- angle কে ধরার জন্য use করতে পারি

- Robot এর shoulder থেকে wrist পর্যন্ত manipulator

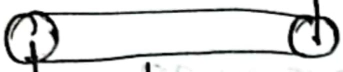
End effector :- যে part actual physical task করে

- Robot এর palm → end-effector
- End effector change করলে functionality change হয়ে যায়

Anthropomorphic type — Human এর মতো যে end-effector বানায়

Transmission

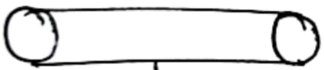
Belt



belt এর জন্য এই part move করে

- belt এর motion transmission
  - belt পিছলিয়ে যায়
- low cost
- efficiency is less
- Heavy speed application এ use করতে পারি না

Chain



- High precision লাগলে chain use করতে পারি
- Rigid points এ লাগাতে হয়
- Expensive

Gear

- ছোট gear বড় gear এর চেয়ে বেশি speed এ rotate করে to cover the distance covered by বড় gear
- Torque কম বেশি করা possible

Rack & pinion

- Rotation এর motion থেকে linear motion এ আনি

প্র। Pneumatic & Hydraulic এর (main) difference কি?

Actuators

Pneumatic actuator — compressed air use করে task করে
— বেশি fast কাজ করতে পারে

Hydraulic actuator — liquid use করে task করে
— অনেক বেশি load carry করতে পারে
— Torque বেশি
— বেশি fast কাজ করতে পারে না
— reservoir use করে

Electric actuator — motor
~~Motor~~

motor → Rotor, Stator

Rotor → rotate করে

DC motor — Electric ~~motor~~, field, magnetic field, force
↳ Faraday's law
— magnetic & electric field এর angle এর উপর, depend করে force
— current বা electric field বাড়লে speed বাড়ে
— cheaply available
→ parallel — no force
→ perpendicular — max force
— high starting torque
— speed control করা যায়

Stepper motor – rotor + pairs of electromagnetic stator
- rotor has 4 poles. Stator has 6 poles.
- কোন pair of poles energised করে, তার উপর rotor movement depend করে
- Rotor এর pole এমন থাকে so that any one of the pole pair of rotor is close to the stator.
- বেশি pole থাকলে more precisely move করে
- precise movement লাগলে stepper motor use করি
- eg: 3D printing
- precision লাগলে → stepper
- accuracy লাগলে → DC motor
- high output torque
- control speed under varying load

Servo motor – DC motor + potentiometer + gear + controller

rotation limit করি (0° to 180°)

motor কতটুকু rotate করে এই info দেয়

angle নিয়ে, interpret করে কতটুকু rotate করবে

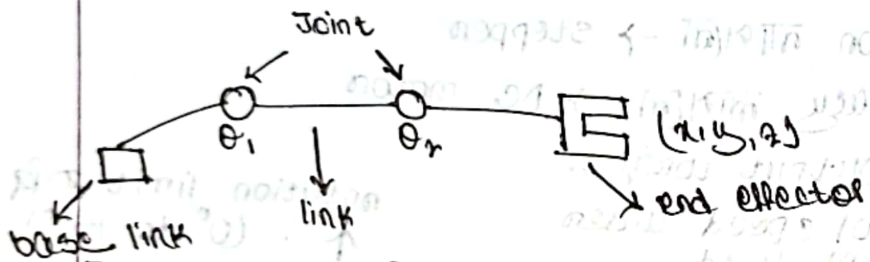
- 0° to 180° rotation করাতে পারি
- Angle set করি
- Continuous rotation হয় না
- Angle set করা লাগলে servo use করি
- control speed under varying load.

LECTURE 6

TUESDAY DATE: 10/10/23

CHAPTER 2: KINEMATICS

* Robot's arm deals with motion
* Kinematics deals with the changes caused by the movement of the arm.
* (x,y) of the end effector controls in which position the robot moves.
* Joint angle needs to be find out.

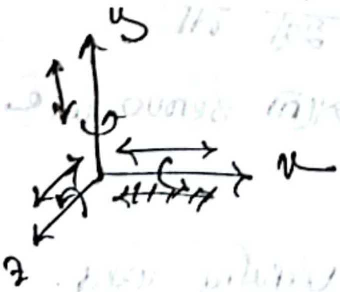


$[\theta_1, \theta_2, \ldots \theta_n] \rightarrow (x,y,z)$

(i) Direct Kinematics → input: θ
output: x,y,z

(ii) Inverse Kinematics → input: x,y,z
output: θ

* Degree of freedom: Robot এর possible motions



6 basic movements

General purpose of robots:-
① Robot has 6 basic movements:
— 3 translational movement
— 3 rotational movement

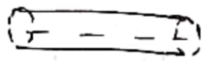
* Action Joint: যে Joint move করতে পারে। যাদের dof আছে
* Passive Joint: যে Joint move করতে পারে না

Redundant robot — more than 6 dof
— no. of motors = no. of dof
— same point এ অনেক ভাবে reach করতে পারে

Deficient robot — less than 6 dof
— অনেক point এ reach করতে পারে না

Types of (Active) Joints : — Prismatic Joint: (slide করতে পারে)
— একটা joint আরেকটা joint এর respect এ slide করে

— Revolute Joint: (always rotate করতে পারে)
— একটা joint আরেকটা joint এর respect এ both slide or rotate করে
— independent.



cylindrical joint motion

— Helical Joint:
— Rotation হলে translation হবে & vice versa.

— Spherical Joint:
— যে কোন direction এ rotate করে

Geometric method: For different robots, we have use different process to find out the final sol^n. That's why it's inefficient.

* Coordinate is assigned for each joint by considering z-axis as the centre.

* প্রত্যেক joint এর জন্য 4 টা parameter দিবো

* Joint angle : Joint এর ভিতরের angle (θ)

* Joint distance : • O_{i-1} থেকে শুরু
  • X_i এর axis কে extend করে Z_{i-1} এ মেলাব
  • Intersection betⁿ Z_{i-1} axis and extension of X_i and O_{i-1} is the joint distance

* একটা coordinate system থেকে আরেকটা coordinate system কে transform করি

* Link twist angle : দুইটা Z-axis এর মাঝের angle (α)

$\alpha_i \rightarrow$ rotation angle from Z_{i-1} to Z_i about X_i
(Z-axis এ rotation)

$a_i \rightarrow$ distance from intersection of Z_{i-1} & X_i to origin of i coordinate along X_i

$d_i \rightarrow$ distance from origin of $(i-1)$ coordinate to intersection of Z_{i-1} & X_i along Z_{i-1} axis

$\theta_i \rightarrow$ rotation angle from X_{i-1} to X_i about Z_{i-1}
(Z-axis কে centre করে rotation)

* For θ_i, যদি কোনো revolute angle mark না করা থাকে, then $\theta_i = \theta_1, \theta_2, \ldots, \theta_n$ এভাবে ধরে নিব

FORWARD KINEMATICS

z - axis: $\begin{bmatrix} P_x' \\ P_y' \\ P_z' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix}$

* z কে centre ধরে rotate করি, so x and y change হয় and z same থাকে

y - axis :

x - axis

Homogenous

$T = \begin{bmatrix} R_{3\times3} & | & P_{3\times1} \\ -- & | & -- \\ f_{1\times3} & | & 1\times1 \end{bmatrix}$

$R_{3\times3}$ → Rotational matrix

$P_{3\times1}$ → Position ~~Perspective~~ matrix

$f_{1\times3} = 0$

$1\times1 = 1$

Denavit – Hartenberg (D-H) Homogenous Transformation matrices

$$A_i = R_{z,\theta_i} \cdot Trans_{z,d_i} \cdot Trans_{x,a_i} \cdot R_{x,\alpha_i}$$

→ Rotation হয়, so no changes in any matrix. তাই identity matrix use করি। z-axis এ just d_i দিয়ে দিব

→ x-axis কে centre করে rotate করে, so x এর জন্য কোন rotate হবে না।
$\begin{bmatrix}1\\0\\0\end{bmatrix}$ → for x

* Joint 1: fixed থাকবে ও তার base fixed.
* Joint fixed থাকে, just robot move করে।
* পুরো robot এর জন্য overall transformation জানা লাগবে
$T_0^3 = (T_0^1)(T_1^2)(T_2^3)$

Initial position vector : 1st Joint/ যেখানে base থেকে calculation করছি তার coordinate.

$^3q = [0,0,0,1]^T$ ---> if কিছু না mention করা থাকে

* Position vector calculate করি to get the actual initial vector.

LECTURE 7

FRIDAY DATE: 27/10/23

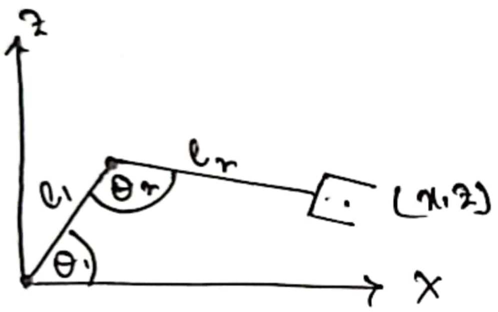
INVERSE KINEMATICS

Forward:

$(\theta_1, \theta_2, \ldots\ldots, \theta_n) \longrightarrow (x, y, z)$

Inverse:

$(x, y, z) \longrightarrow (\theta_1, \theta_2, \ldots\ldots, \theta_n)$



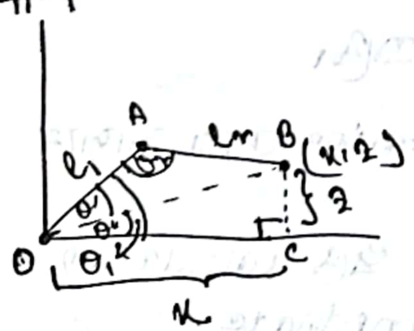
Cosine Rule

$BC^2 = AB^2 + AC^2 - 2 \cdot AB \cdot AC \cos \angle BAC$

$\angle BAC = \dfrac{AB^2 + AC^2 - BC^2}{2 \cdot AB \cdot BC}$

Known: (x, z)
l_1, l_2

#1



$OC = x, \; BC = z$

$OB = \sqrt{x^2 + z^2}$

$\angle OAB = \theta_2 = \cos^{-1}\left(\dfrac{l_1^2 + l_2^2 - (x^2 + z^2)}{2 \cdot l_1 \cdot l_2}\right)$

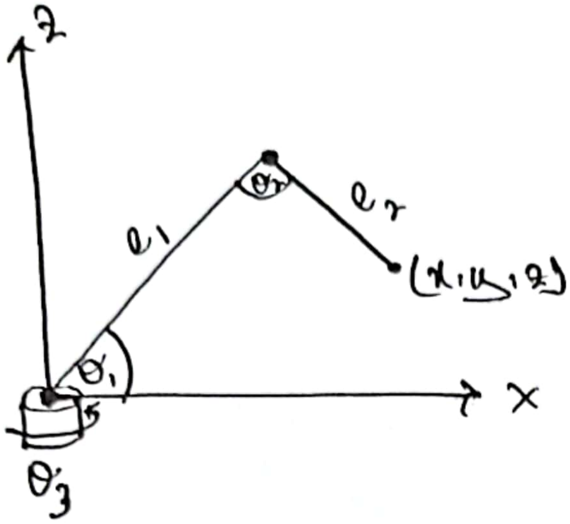
$\theta_1 = \theta' + \theta''$

$\tan \theta'' = \dfrac{z}{x} \Rightarrow \theta'' = \tan^{-1}\left(\dfrac{z}{x}\right)$

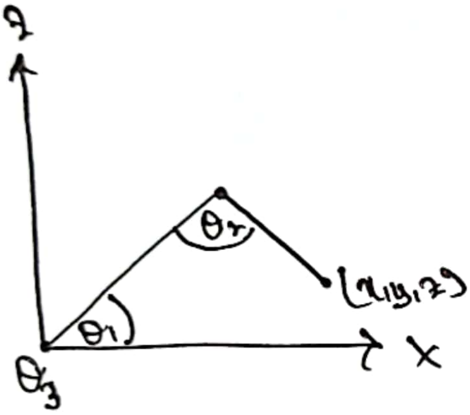
$\theta' = \cos^{-1}\left[\dfrac{l_1^2 + (x^2 + z^2) - l_2^2}{2 \cdot l_1 \cdot \sqrt{x^2 + z^2}}\right]$

*Different orientation এর robot কে অন্য পজিশনে ও [illegible] কিভাবে orientation এ এনে solve করতে হয়

২

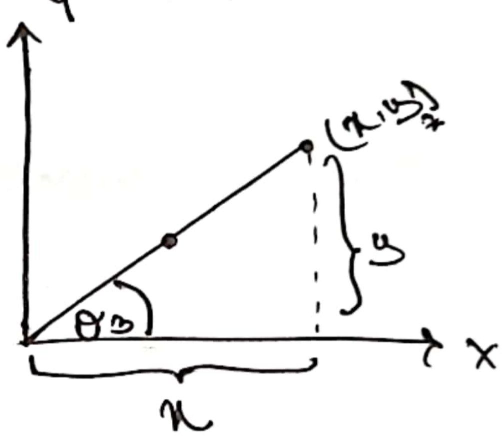


* এখানে ৩ টা axis এ আছে
* Side-view তে (x,z)
* Top-view তে (x,y)



Side-view

Top-view



$$\theta_3 = \tan^{-1}\left(\frac{y}{x}\right)$$

* #1 এর মতো solve করতে হবে

LAB 1

UART (Universal Asynchronous Receiver Transmitter)
I2C (Inter integrated Circuit)
SPI (Serial Peripheral Interface)
→ For Data communication

} sub categories of serial transmission protocol

* LED does not follow any communication protocol

~~LED:~~
GPIO4 এর সাথে resistor connect করব। resistor এর এক end GPIO4 and অপরটা end এর +ve end of [লম্বা] resistor এ connect হবে। LED এর -ve end (short) GRND এ যাবে।

LED কে resistor এর মধ্যে দিয়ে route করছি।

GPIO4 → High → LED এর terminal এ voltage drop হবে → LED ON

GPIO4 → Low → LED এর terminal এ voltage drop হবে না → LED OFF

220Ω → red, red, brown, gold

FINAL

CamScanner

LECTURE 8

THURSDAY

DATE: 17/11/23

ROBOT NAVIGATION

Things to remember for best navigation:

① Best path

② Avoid obstacle

Navigation: Ability to determine your location and plan a path to some goal.

Robot কে endpoint detect করতে হবে, goal এ যাওয়ার জন্য path determine করতে হবে।

Environment সম্পর্কে জানতে হবে, তাই environment model create করতে হবে।

নিজের location জানতে হবে।

Then desired location জানতে হবে।

Environment
- static
- dynamic
- Known
- unknown
- structured
- Unstructured

Environment এর model এ obstacles গুলো include করে নিতে হবে।

কোন environment এ কি ভাবে robot টা operate করবে সে টা set করে দিবে।

ROBOT NAVIGATING

Path Planning: starting থেকে goal এ কেমন করে যেতে পারি

Localization: environment কোথায় position আছি জানা লাগবে

Mapping: environment এর map।

Exploration: কোন position এ ~~robo~~ আমরা জানি না. So robot different position দিয়ে map করবে।

PATH PLANNING

Multiple path দিয়ে goal reach করতে পারি।

Best path [depends on our requirements] → shortest path, quickest path, easiest path

Optimal path বলতে কি বুঝি যে টা robot কে feed করায় দিব।

VISUAL HOMING (Purely Reactive Navigation)

→ goal দেখতে পাবে বা পাবে না

if goal is visible → move straight to goal

if goal is invisible → move randomly [expecting to reach goal at a point]

→ Not optimal as maximum time ই robot কে randomly move করা লাগবে

BUG-BASED PATH PLANNING

→ goal কোন direction এ আছে and goal থেকে কত দূরে আছি জানতে পারি

→ local environment and global goal are known.

if there's no obstacle → move straight towards goal

if there's obstacle → obstacle এর এমন একটা point choose করে move করবে যার সাহায্যে quickest path follow করে goal এ যাওয়া যাবে

→ goal থেকে robot এর distance কমবে যে path follow করলে এমন path choose করবে

→ Not optimal as not always shortest/quickest path will be achieved.

⇒ Better than visual homing.

* For ~~environment~~ visual homing & bug-based path planning, we don't have a map.

☐ Scenario based question on which algo to use

Case 2: We have the environment map

We can represent the map using graph.
Free paths = Edge with small valued weights
Obstacles = No edge / edge with very large weight (∞)
(distance / time)

Paths = Node

METRIC / GLOBAL PATH PLANNING

→ Environment এর map আমাদের use করতে পারি
→ Correct map representation + Path finding Algo

LOCALIZATION

→ আমার current location
→ GPS দিয়ে location জানতে পারি
→ Problems :-

①
If no map is given :-
→ Environment এ reference object আছে কি না

Yes → reference follow করে map create করব

No → problem

প্র Scenario based question on which localization technique to use

DEAD - RECKONING (MOTION)

→ robot এর previous speed, position use করে current position estimate করে,

→ Reference Object নাই

→ Robot এর motor কত move করতে পারি

→ angle change করতে পারি and position track করতে ও করতে কোন location এ যেতে পারি

Problems: sensor এ সমস্যা, motor এ সমস্যা থাকতে পারে, so not accurate

LANDMARK (SENSING) BASED

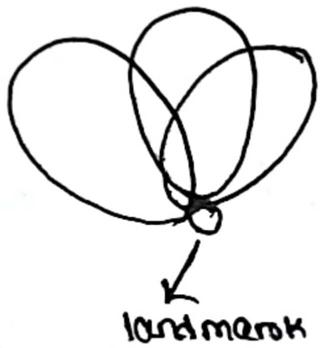
→ Environment এর known landmark use করে location determine করে

→ Landmark এর represent এ object থেকে কত দূরে আছি জানতে পারি

→ Position determine করতে পারি

→ GPS

- Satellite থেকে আমার distance কত দূরে সেই distance বের করে
- প্রতিটা satellite থেকে radio wave send করা যায়, distance measure করা যায়
- তিনটা satellites use করা যায়, distance measure করা যায়
- Intersection point এ object location পাবো



Problems: precision এ error পাবো (not 100% accurate)

STATE ESTIMATION

→ By itself localization algo না

→ অন্যান্য localization technique এর সাথে combine করে technique গুলোর করে accuracy বাড়াই, error কমাই,

→ model রাখি, model দিয়ে predict করি কোথায় object টা থাকতে পারে।

• motion model — আমরা যে action নিলাম, তার জন্য কোন position এ যেতে পারি predict করি।

— prediction + actual system এর সাথে compare করে error কমাই।

LECTURE 9

SUNDAY DATE: 19/11/23

ROBOT NAVIGATION

MAPPING & EXPLORATION

→ যে environment এ আছি সেটা কেমন দেখতে, তার shape কেমন, তার info memory তে store করি → mapping

→ কোন new environment এ operate করলে, সেই environment ব্যাপারে আমরা জানি না, তাই mapping করে জানা লাগবে. — exploration

→ Environment দেখতে কেমন, idea জানা লাগবে, জেনে data store করি

OCCUPACY GRID

→ Unknown
→ Empty
→ Occupied
} state

*Occupied হয় নিয়ে, empty হবে আগে এটা জানা লাগবে না.

→ mapping এর কাজ করলে কত গুলো grid unknown/empty/Occupied

→ Area এর তুলনায় বেশি grid use করলে memory বেশি লাগবে।

→ Area এর তুলনায়, ছোট grid use করলে precise mapping হবে না।

→ Environment accurately represent + Robot motion correct হওয়া লাগবে।

→ Area এর জন্য optimal output যে grid size দিবে, সেই grid use করব।

→ Environment কে grid দিয়ে represent করি।

→ Robot এর জন্য Rasberry Pie বা Jetson Nano use করি so. বেশি memory store করতে পারে না।

→ Occupancy grid 3D হলে height এর include হবে।

Sensor Model

→ Occupancy grid বানানোর জন্য sensor model use করি।

→ Sensor use করি distance measure করতে।

depth sensor model

→ Obstacle আসা আগে পর্যন্ত robot যত টুকু area observe করতে পারে "empty" mark করবে।

→ obstacle আসলে observable part কে "occupied" mark করে দিবে।

→ আমরা randomly grid choose করছি, অনেক বেশি steps লাগে

→ sensor model construct করবে।

→ Environment কে sense করে info নিবে।

→ loop চালাবে upto all the grids are marked as occupied, empty

Exploration:

Frontier

→ এমন unknown cells যার adjacent empty cell থাকবে

→ unknown cell এর adjacent known cell আছে

→ frontier select করে mapping করি

→ আমরা systemitically move করছি, steps কম লাগবে। memory use কম use করতে হবে।

→ choosing best frontier:

- কম দূরে যে frontier কোনটা → steps ↓
- unknown cells কোন frontier এর বেশি
- কোন area তে বেশি map হয়নি
- empty cells highest কোনটায়

* adjacent empty cell আছে so that আমরা ঐ cell এ যেতে পারি এবং mapping continue হয়।

Simple OG Algo (for mapping)

→ Robot 'n' distance move করে with a fixed angle.

→ observable area টা map করে।

→ grid fill করে as empty if an obstacle is not there.

→ grid fill করে as occupied if obstacle থাকে

→ এটা loop এ চলে upto all the grids are known.

→ next grid এ যায় randomly which is not optimised.

→ optimise করার জন্য frontier rule use করি

→ এই Algo তে অনেক steps লাগে, so memory অনেক use হয়।

* Position বুঝে, sense করে mapping করলো, এটা আমরা বলি Simultaneous Localization And mapping (SLAM)
* Map আছে মেনে জানি, so path planning করতে পারি.
→ Navigation.

Use case of mapping and exploration:
→ Robot runs on charge. So, if they move randomly in an unknown place for too long, their battery will run out fast. Frequent charging is required.
→ Where the charger of the robot is? — can be found out by knowing the environment.

Summary
① নিজের position জানা লাগবে
② environment sense করা লাগে
③ map করা লাগবে after knowing the environment
④ Optimal path plan করা লাগে

ECTURE 10

TUESDAY DATE: 22/11/23

INTRODUCTION TO CONTROL SYSTEM THEORY

→ manual বা automatic control চায়

→ Robot কে set করে দিয়ে, Navigation use করে path planning করবে।

→ Robot motor use for movement then straight line এ যাওয়ার জন্য সব motors এর same speed থাকতে লাগবে,

– rotational movement হলে সব motors এর different speed থাকা লাগবে, কত ডিগ্রি rotate করবে সেটা speed এর depend করবে।

→ Goal এ গিয়ে যে robot কে stop করা লাগবে তার জন্য robot কে goal এর কাছাকাছি আসলে speed কমানো বা robot এর braking time control করা লাগবে,

→ actuator কে কি signal provide করতে হয় by controllers.

CYBERNETICS

→ Cybernetics feedback produce করে and আমাকে জানায় আমরা কাজটা ঠিকঠাক করছি কি না।

→ at one point, 5V electrical supply আছে কি না, motor ঠিক করে ঘুরছে কি না — এইসবের feedback দেয়।

→ এই feedback দিয়ে control system কে adapt করে।

→ Cybernetics is the base of robotics.

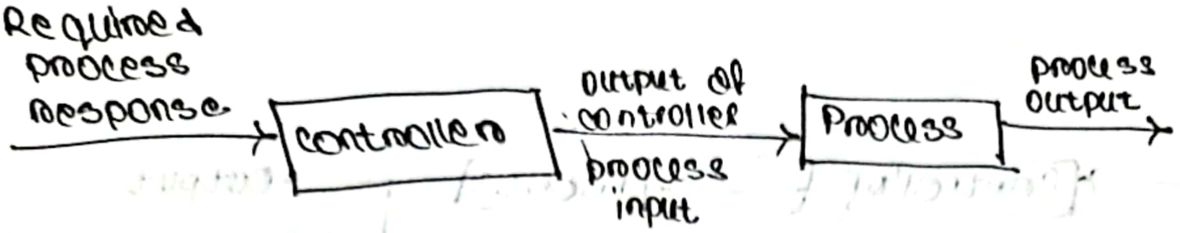
12V reach করতে অনেক time লাগে বা overshoot হয়, then bulb damage হতে পারে।

If we want to change the switch design, we have to analyse the control system such as decreasing transient response and outburst. এগুলোর জন্য control theory দিয়ে analyze করতে পারি যেমন algo use করতে পারি

OPEN LOOP MODEL

Output < Input because:
- controller can be faulty,
- controller and motor এর মধ্যে অনেক wiring থাকে, so কোনো part faulty থাকতে পারে,
- motor can be over-used, so signal is lost due to friction.

* different motors এ voltage pass করালে এর output দিয়ে বারবার manually check করা time-consuming and is not feasible.

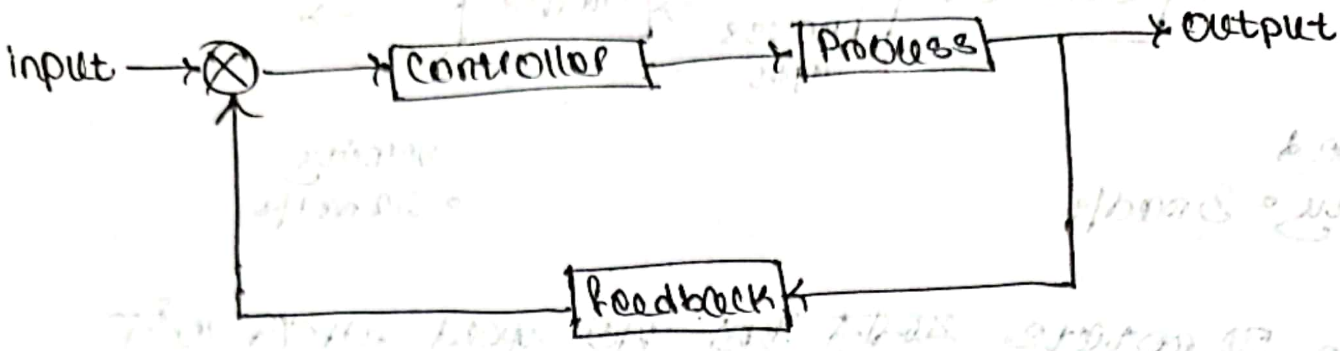


Required velocity = 3 rad/s

velocity = 2.2 rad/s

- motor কে rotate করার জন্য যে speed লাগবে সেটা controller এ pass করি as required process response.
- controller motor কে rotate করানোর জন্য signal পাঠায় as output.
- controller output টা process এর input. এই input দিয়ে motor rotate করে output generate করে।
- এখানে output velocity টা required velocity থেকে কম হয়েছে। controller এটা monitor করতে পারে না। That's a problem.

CLOSED LOOP CONTROL



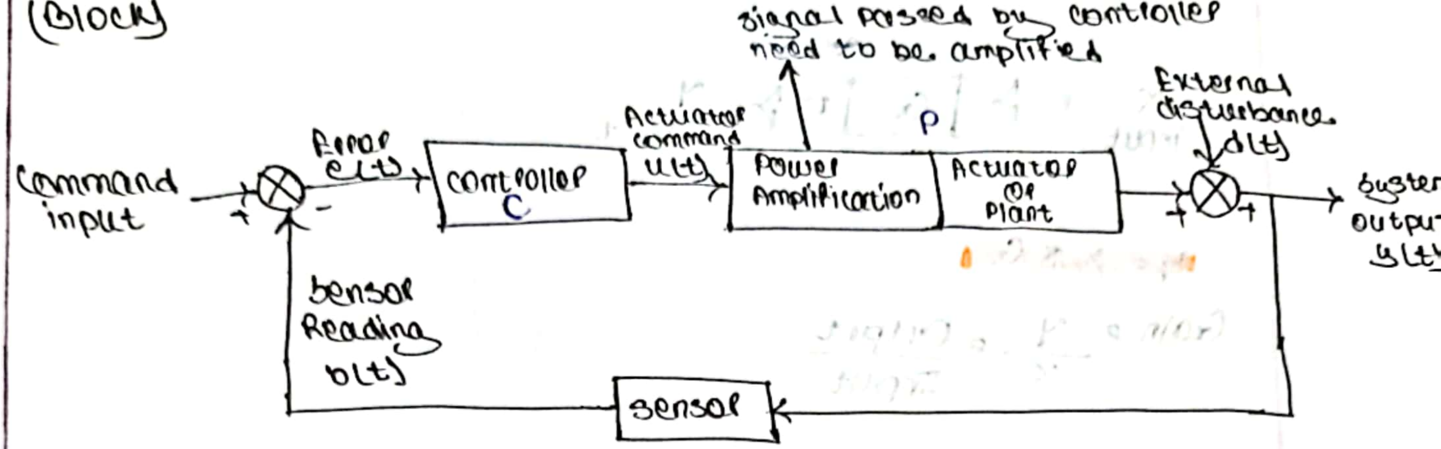
→ Closed loop because by connecting output to feedback and feedback to input, we are ultimately creating a loop.

→ feedback checks the error and sends the error feedback to the controller

→ controller now knows what's the error in velocity and can update/change ~~the~~ signal transmitted to the process to compensate error.

→ So, the process can receive the required signal to produce the required output.

FEEDBACK DIAGRAM
(Block)



→ component গুলো box দিয়ে denote করি।

→ signal কে arrow দিয়ে denote করি।

→ ⊗ — summation | difference
  - signal এ +/- করালে ~~⊗~~ এটা use করব

→ Input controller এ দিলাম

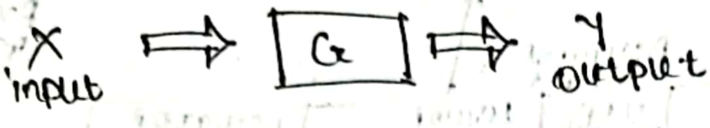
→ controller যে signal দেয় খুব low হয় I বা V দেয়। so directly signal use করলে process করা যায় না ও করলেও পারে। so we use power amplification to amplify signal before sending it to the actuator.

→ motor rotate করে। motor কে ঘুরে বাধা দেয়। so এগুলো external disturbance.

→ External disturbance এর কারণে speed কমে। power কমে।

→ এ change এর কারণে যে output পাই। সেটাই sensor এর মাধ্যমে error count করে controller কে দেয়।

Transfer function



$Y = X * G$

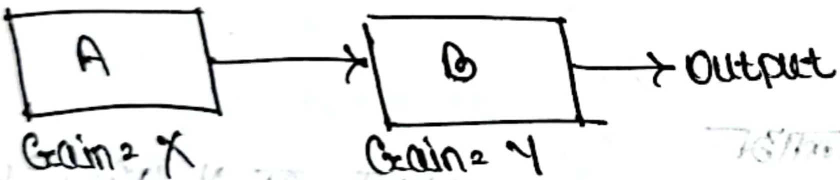
$\text{Gain} = \dfrac{Y}{X} = \dfrac{\text{Output}}{\text{Input}}$

For the diagram in the previous page:

$u(t) = c * e(t)$

$= c * [\,\cdot(t) - b(t)]$

① Cascaded elements: convolution



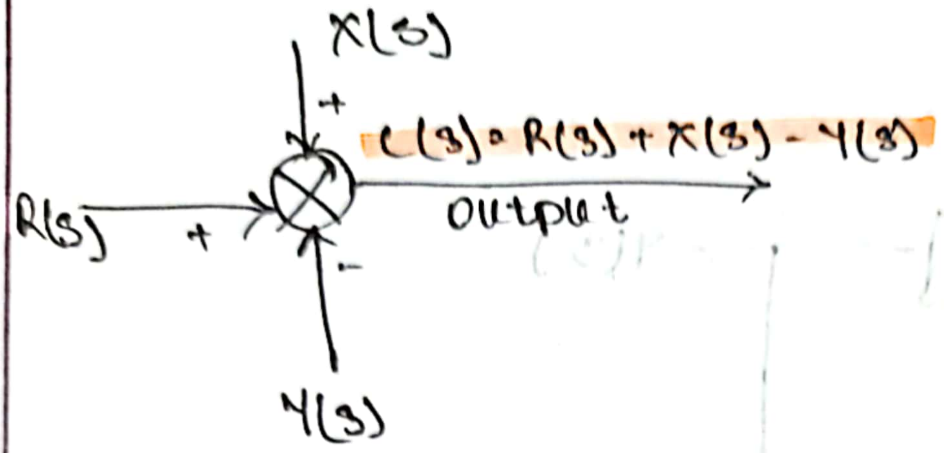
দুটো block এর মাঝে যদি arrow থাকে, তাহলে

Output = Gain of block A (X) * Gain of block B (Y)

হয়ে থাকে।

যদি arrow দিয়ে connect করা block গুলো কে combine করে একটা block এর মধ্যে gain কে multiply করে output পাওয়া যাবে।

⑦ Summation and difference elements



⊗ এর input value গুলো sum/subtract করে output দেয়

$u(t) = c * e(t)$

$= c * x(t) - b(t)$

$= c * x(t) - s * y(t)$

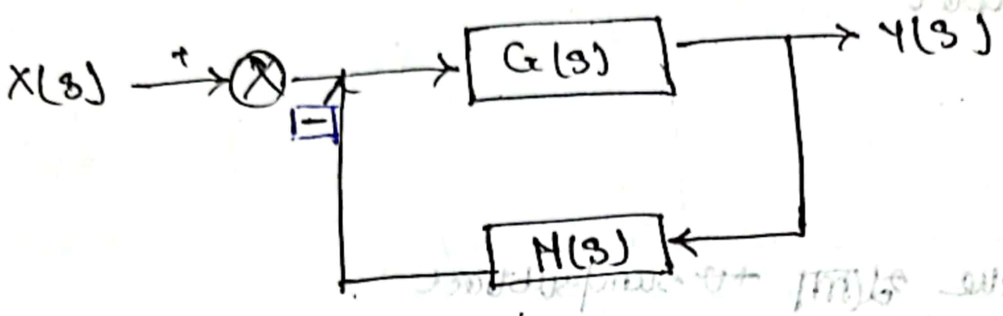
$y(t) = P * u(t)$

$= P * c * x(t) - s * y(t)$

$y = P[c(x - sy)]$

$$y = \frac{(G) * x}{\frac{y}{x}}$$

Feedback connection



$$Y(s) = G\left[X(s) - H(s) * Y(s)\right] \Rightarrow Y(s) + G(s)H(s) * Y(s) = G(s)X(s)$$

$$\Rightarrow Y(s)\left(1 + G(s)H(s)\right) = G(s)X(s)$$

$$Y(s) = \frac{G(s)}{1 \boxed{\mp} G(s)H(s)} * X(s)$$

↳ feedback এর output টা +ve হতে পারে বা -ve হতে পারে, So $\mp$ দিয়েছি।

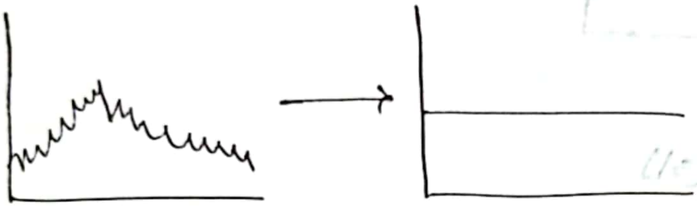
LECTURE 11

SUNDAY DATE: 25/11/23

INTRODUCTION TO CONTROL SYSTEM THEORY (PART-2)

→ frequency domain is better than time domain.

→ Time domain এ noise control করা difficult.
frequency domain এ noise control করা possible. (noise reduction is possible)



time domain → frequency domain

→ We use laplace transformation to convert time domain to frequency domain.

→ Transfer function = $\dfrac{\text{Laplace function of output}}{\text{laplace function of input}}$

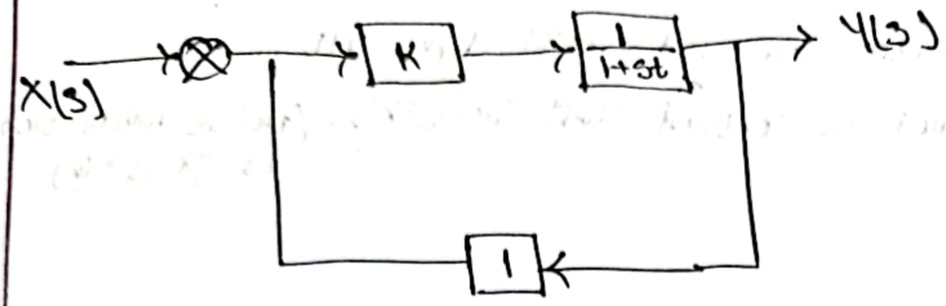
Laplace function of output → output of frequency domain

laplace function of input → input of frequency domain

Transfer function → Gain of frequency domain

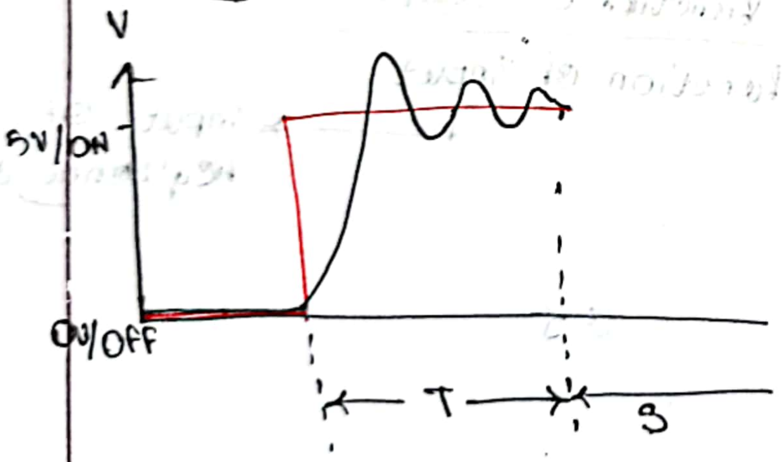
→ feedforward : $\dfrac{Y(s)}{X(s)} = C(s)A(s)$

→ feedback : $\dfrac{Y(s)}{X(s)} = \dfrac{C(s)A(s)}{1+\cancel{(s)}H(s)C(s)A(s)}$



$$Y(s) = \left(\frac{K}{1+st}\right)(X(s) + Y(s))$$

Steady State Vs Transient

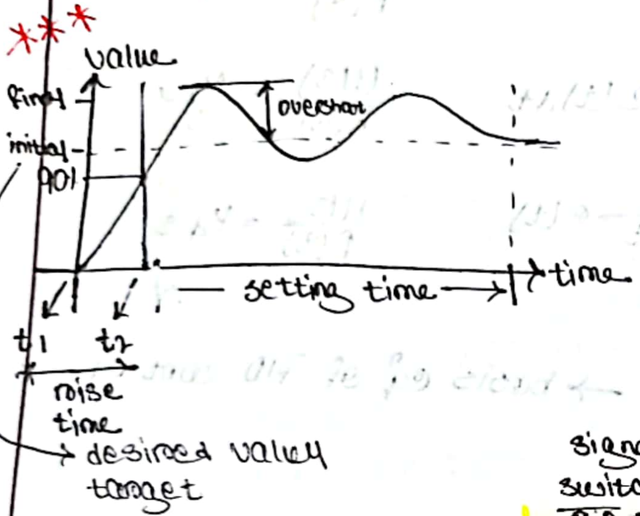


System Response Graph

→ Input change হয়, তাই output change হয়

→ In reality, switch ON করলে component কে operate করার জন্য required voltage যে লাগে তার চেয়ে বেশি voltage যায়

→ আমাদের এই pulse amplitude কমানো লাগবে + transient time কমানো লাগবে।



• overshoot = $\frac{\text{target value} - \text{initial value}}{\text{initial value}} \times 100\%$

(for first oscillation in transient state)

[৫% এর বেশি হতে না]

• rise time = $t_2 - t_1$

signal switch করতে time নেয় ← t_2 → 90% of target value যখন হয়; t_1 → 10% of target value যখন হয়

• setting value = target value এ আসতে এখন এত time লাগে

→ ~~এর~~ এই range এ যায় first peak/trough এর time

eg. 5V ± 0.25 → 5% of 5V

4.75 ~ 5.25

↳ এই range এর মধ্যে যে oscillation এর value থাকে তাকে নিই

↳ highest oscillation এ range এর মধ্যে থাকে, তার time হচ্ছে settling time.

PID controller

Proportional control : $u(t) = K_p e(t)$ $\qquad \frac{U(s)}{E(s)} = K_p$

Integral control : $u(t) = K_i \int_0^s e(t)\,dt$ $\qquad \frac{U(s)}{E(s)} = \frac{K_i}{s}$

Differential control : $u(t) = K_d \frac{d}{dt} e(t)$ $\qquad \frac{U(s)}{E(s)} = K_d s$

*** $u(t) \propto e(t) + \int e(t) + \frac{d}{dt} e(t)$. → basis eqⁿ of PID control

$u(t) = K_p e(t) + K_i \int e(t) + K_D \frac{d}{dt} e(t)$

Laplace transform in both side

$U(s) = K_p E(s) + \frac{K_I}{s} E(s) + K_D s E(s)$

$U(s) = E(s)\left(K_p + \frac{K_I}{s} + K_D s\right)$

$$\frac{U(s)}{E(s)} = K_p + \frac{K_I}{s} + K_D s = \frac{K_p s + K_I + K_D s^2}{s}$$

Effect of controller functions

→ Proportional Action : Error is proportional to

system টা stabilized না,

System কে error দিয়ে চালায়, so, error থাকলে output থাকে, error = 0 হলে, input নাই, so output নাই, আবার বাইরে থেকে input দিতে হবে,

→ Integral Action : এরা error summation করে,

error = 0 হলেও summation of error থাকার কারণে output = 0 হবে না, একটা output থাকে,

system কে stabilize করে,

oscillation reduce করাতে help করে।

→ Differential Action : speed of system বাড়াতে help করে,

desired value and current value যত দূরে থাকে,

desired value যত দূরে, তত fast signal reach করায়,

Proportional action and integral action এ slow processing হয়, differentiation action speed বাড়িয়ে processing fast করে ফেলায়,

PID system – Oscillation reduce করে,

– signal এর speed বাড়ায়,

Ziegler - Nichols Rules for Tuning PID controller

*** K and P are resulting gain and oscillation period

For P control	For PI control	For PID control
$K_p = 0.5K$	$K_p = 0.45K$	$K_p = 0.6K$
	$K_i = 1.2/P$	$K_i = 2/P$
		$K_d = P/0.8$

simple but powerful

LECTURE 12

SUNDAY DATE: 03/12/23

INTRODUCTION TO MACHINE LEARNING

Tumor Size	Age	Condition
10	50	
2	30	
1	45	
5	40	

* Sum of weighted average বের করি [$\underbrace{w1}_{\text{weight}} * \underbrace{x1}_{\text{feature}} + \underbrace{w2}_{\text{weight}} * \underbrace{x2}_{\text{feature}} = \underbrace{y}_{\text{output}}$]

$(0.9*10)+(0.1*50) = 14$ | $(0.9*2)+(0.1*30)$

↓ tumour size কে 90% দিচ্ছি [বেশি imp দিচ্ছি]

↓ age কে 10% দিচ্ছি

* এই weighted sum use করে number line এ বসায়, threshold ঠিক করি

* weights are very important. Prediction এ যারা বেশি important লাগে, তারা বেশি weight নিয়ে থাকে।

* বড় data size থাকলে it is difficult to choose weights manually.

* computer কে manually program করব না, সে নিজে থেকে শিখবে

* Computer data থেকে optimal weight গুলো count করে

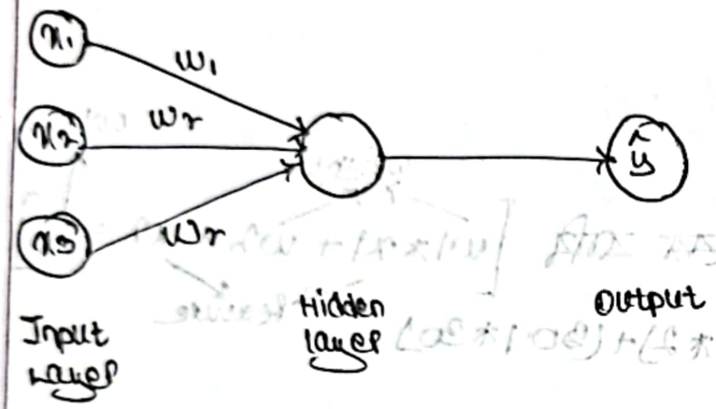
* যেসব ক্ষেত্রে আমরা value predict করি → regression

* specific value predict করতে চাই না, value এর উপর basis করে ~~কে~~ class এর ভিতরে ফেলি → classification

* Mother law of ML → যত গুলো feature আছে তাদের কে weight দিয়ে multiply করি and output বের করি

* Activation function → output 1 এর মধ্যে দেয়

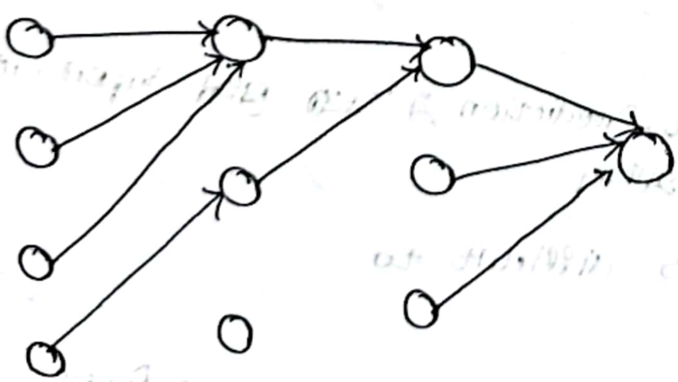
mother law of ML



~~Net~~

Neutral Network

→ basis unit: neuron



- regression হলে activation function এর মধ্যে দিয়ে pass করবে না
- classification এ activation function use করি
- probability মাপে একটা, তারপর output দিবে,

* input এর value weight দিয়ে multiply করে, activation function দিয়ে pass করার পর output বের করি, এই output টা hidden layer এর input হয়ে যায় and same process follow করে,

ML	DL
① 1 input 1 hidden 1 output	① 1 input many hidden 1 output
② manually feature extract করে input হিসেবে দিই + automatic weight	② নিজেই sensor থেকে feature extract করে কাজ করে + automatic weight

* Layer to layer যাওয়ার time এ, neural network আরো ভালো ভালো feature learn করে।

* Neural network is inspired by human brain

LECTURE 13 | TUESDAY | DATE: 12/12/23

CNN and Object Detection

Two types

(1) features are given → manually data extract করত

(2) Raw data → raw data দিয়ে দিত, model features extract করত নিজেই

- sepal width এর দিকে bias হচ্ছে
- ~~weight~~ feature এর importance এর depend করে weight দিত, value হবে। বেশি কোনো importance determine করবে না
- more important feature কে বেশি weight and less importance feature কে কম করে weight দিত।

ML এ handcrafted data use করত।

~~Now~~

DL এ data directly feed করত। machine এ নিজে কাজ করত, কোন feature বেশি important, কত weight বেশি দেওয়া লাগবে। data তে কি changes লাগবে — এর কাজ কি machine করে ...

Object Detection

→ Image – 2D array of pixels
- pixel এর intensity এর উপর depend করে (grey scale এ)

——x——x——

- 3D array of pixels
- pixel এর intensity থাকে (RGB)

Image কে flatten করতে পারি।

3x3 image $\xrightarrow{\text{flatten}}$ 9x1

↓

features

Problems:-

(1) বেশি pixel এর image দিলে, অনেক বেশি features থাকবে, Hidden layers increase করবে, computation cost বাড়বে।

(2) Image এর structure info থাকে না image flatten করার সময়। Image এর surrounding এ কি আছে সেটা হারিয়ে ফেলি, useful info extract is not possible from image.

Convulation

* Kernel আছে → parameters গুলো DL দিয়ে learn করি
→ যে weight এর করতে চাই

- Image এর first 3x3 matrix এর উপরে kernel বসাবো
- summation করে নিব
- kernel কে shift করব
- process repeat করব

*process টা কে convolution বলে

- kernel filter এর value learn করি
- kernel and image matrix use করে feature বের করতে পারি

Image	×	Filter/Kernel	=	feature
5x5		3x3		3x3
n×n		k×k		

$$f = n - k + 1$$
$$= 5 - 3 + 1$$
$$= 3$$

→ output feature এর dimension.

2	4	9	1	4
2	1	4	4	6
1	1	2	9	2
7	3	5	1	3
2	3	4	8	5

image

×

1	2	3
-4	7	4
2	-5	1

filter

=

51		

feature

for 51 in feature:

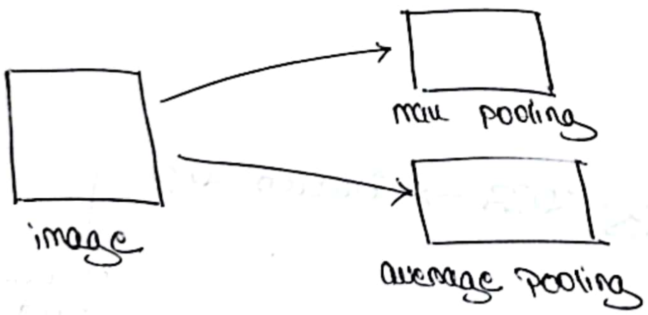
$$(2*1) + (4*2) + (9*3) + (2*-4) + (1*7) + (4*4) + (1*2) + (1*-5) + (2*1)$$

$$= 51$$

from image — from filter

Pooling Layer

- বেশি pixel থাকলে, আমরা smooth ও image পাই
- পাশাপাশি pixel বেশি vary না করলে।
  পাশাপাশি feature almost same
  feature carry করে (for high resolution).
- তাই অনেক redundant feature থাকে,
  computation cost বাড়ে।



- 2x2 matrix নিই।
- max and average
  add করি max
  pooling & average
  pooling matrix এ।
- repeat করি for
  next 2x2 matrix

- Dimension half করে দিলাম, but information
  loss হয় না

image × Filter Kernel = Feature

image 5×5 (25 pixel) × Filter Kernel 3×3 (9 parameters) = Feature (9)

- 3×3 filter দিয়ে 9 feature পাই from 25 pixel input.

image 10×10 (100 pixel) × Filter 3×3 = Feature $[f = 10 - 3 + 1 = 8]$ (8×8) = 64

- 3×3 filter দিয়ে 64 feature পাই from 100 pixel input.

* Image এর pixel যতই বড় হোক না কেন, আমরা same kernel use করতে পারি।

Structure of CNN

→ convolution করব → pooling করব → flatten করব → features পাব → fully-connected neural network → output

* আমরা যখন convolution layer দিয়ে কাজ করি।

৫। convulation math আছে — image & kernel দেওয়া থাকলে output বের করবে।

৬। Dimensionality এর problems + pooling কেন use করি + multiple layer কেন use করি।

— যত বেশি layer use করবে, convulation layer আরো

better + high level feature দিবে

- CNN কি হচ্ছে
- pixel use করলে problem কি
- pooling কেন use করি
- neural network এর structure

Chapter 1
- sensor
- actuator
- controller

Convulation

$$\begin{bmatrix} 2 & 4 & 9 & 1 & 4 \\ 2 & 1 & 4 & 4 & 6 \\ 1 & 1 & 2 & 9 & 2 \\ 7 & 3 & 5 & 1 & 3 \\ 2 & 3 & 4 & 8 & 5 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 3 \\ -4 & 7 & 4 \\ 2 & -5 & 1 \end{bmatrix} = \begin{bmatrix} 51 & 66 & 20 \\ 42 & 49 & 94 \\ 15 & 44 & -2 \end{bmatrix}$$

$(2*1) + (4*2) + (9*3) +$
$(2*-4) + (1*7) + (4*4) +$
$(1*2) + (1*-5) + (2*1)$
$= 51$

Pooling

* convulation করে যে matrix পাবো তাকে pooling করব

$$\begin{bmatrix} 51 & 66 & 20 \\ 42 & 49 & 94 \\ 15 & 44 & -2 \end{bmatrix} \rightarrow \begin{bmatrix} 66 & 94 \\ 49 & 94 \end{bmatrix}$$

max pooling

$$\begin{bmatrix} 51 & 66 & 20 \\ 42 & 49 & 94 \\ 15 & 44 & -2 \end{bmatrix} \rightarrow \begin{bmatrix} 52 & 5\cancel{3}.25 \\ 37.5 & 46.25 \end{bmatrix}$$

average pooling

* যখন দরকার হবে তখন overlap করে filter এর matrix তৈরি।

$$\begin{bmatrix} A & B & C & K & N \\ D & E & F & J & O \\ G & H & I & M & P \end{bmatrix}$$

overlap আসছে

* enough data না থাকলে overlap use করব।

Math
- Convulation + PID pooling
- PID tuning
- response diagram

Derivation
- Block diagram থেকে transfer function derive করা লাগবে
- PID derivation

PID tuning

Gain = 5 = K

Oscillation = 10 = P

P I

$K_p = 0.45K = 0.45 * 5$

$K_i = 1.2/P = \frac{1.2}{10}$

PID

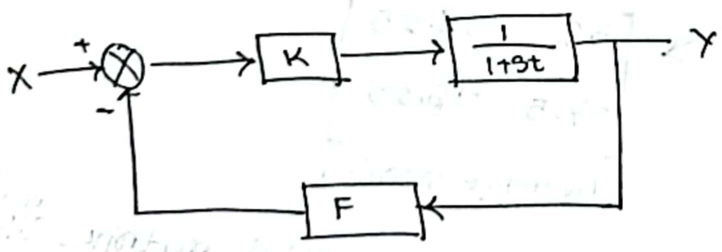
$K_p = 0.6K$

$K_i = 2.0/P$

$K_D = P/8$

P

$K_p = 0.5K$



$$Y = \frac{1}{1+St} * K(X - FY)$$

$$\frac{Y}{X} = \frac{\left(\frac{K}{1+ST}\right)}{\left(1 + \frac{KF}{1+St}\right)}$$

• Chapter 1

(same question as mid)

processing unit আছে controller

↓ microcontroller
→ simple robot use করি।
যে শুধু forward, backward move করে
→ অনেক robot বানালে use করি

↓ Rasberry pie
→ complex robot এর জন্য use করি।

↓ Jetson Nano
→ GPU আছে so fast কাজ করে
→ complex robot এর জন্য use করি।

• Robot Navigation

Path planning –

localization –

mapping & exploration –

(scenario based question আসবে)

• Control System

(overshoot এর math টা, PID basis, PID tuning → zebra nichols
↓
formula use করে math দেওয়া থাকলে করতে হবে

Block diagram – feedback)

• ML

(Regression + classification → last column এ continuous value থাকলে — regression
specific classification থাকলে — classification

↓

Regression আর classification আলাদা হলো?

(Neural network আবার লাগবে)

↓

Input

+

Hidden Layer

+

Output

↙ ↘

1 output (regression)

output = no. of classification (classification)

(CNN কি, pooling, kernel দিয়ে কিভাবে convulation হচ্ছে দেখবো)

LAB
3

SATURDAY

DATE: 25/11/23

ROS
(Robot operating system)

→ বিভিন্ন language এ program করতে পারবো

→ Ubuntu তে work করবে

GUI → operations perform করতে পারবেন

roscore → master কে On করে

python --version → python version check করে দিবে

python3 --version

roscore দিবে → error না আসলে ROS on হলে যাবে [ROS on করলে আর কোন code করা যাবে না]

rosrun turtlesim turtlesim_

folder create — mkdir folder_name

file create — touch file_name

cd — folder এর মধ্যে চলে যায়

LAB 4

SATURDAY DATE: 2/12/23

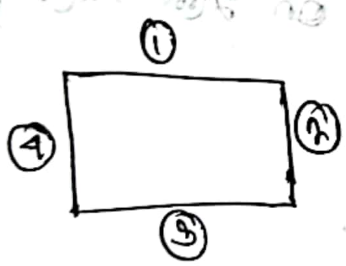
TASK 1

* Rectangle আঁকা লাগবে
* parallely 2 টা horizontal line draw করা লাগবে. as the width.
* line দুইটার end points থেকে vertical line ও draw করবে as the length.
* user input নেওয়া লাগবে

Task 2 যা যা লাগবে as input

Optional

speed
distance — 2 টা লাগবে
move forward



Task 2

* ~~ea~~ spiral draw করা লাগবে
* angular distance cover করবে
* angle দিবে
* Hand code করে দিতে পারবা