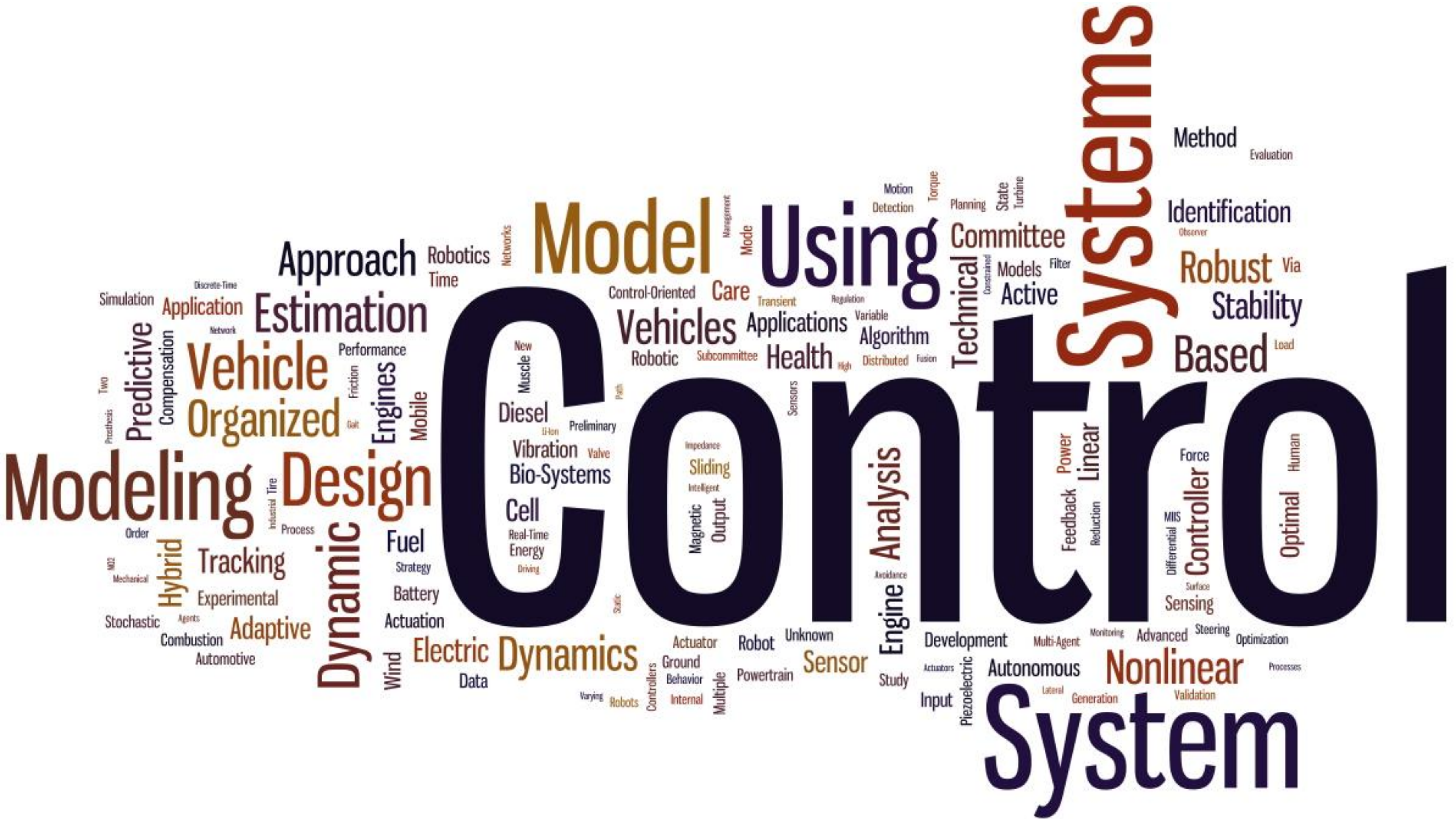

Systems
Method
Evaluation
Motion
Torque
Detection
Planning
State
Turbine
Management
Model
Using
Identification
Observer
Mode
Committee
Networks
Approach
Robotics
Time
Robust
Via
Filter
Models
Constrained
Discrete-Time
Technical
Active
Simulation
Application
Estimation
Control-Oriented
Care
Transient
Regulation
Stability
Network
Variable
Vehicles
Applications
Algorithm
Predictive
Compensation
Vehicle
Performance
New
Subcommittee
Health
Based
Load
Robotic
Distributed
Fusion
High
Two
Friction
Muscle
Organized
Engines
Mobile
Path
Sensors
Prosthesis
Gait
Diesel
Preliminary
Li-Ion
Control
Impedance
Vibration
Valve
Modeling
Design
Bio-Systems
Sliding
Power
Linear
Force
Human
Tire
Intelligent
Analysis
Industrial
Cell
Controller
Feedback
Order
Process
Magnetic
Output
Reduction
MIIS
Optimal
Dynamic
Real-Time
Fuel
Energy
Differential
NO2
Hybrid
Tracking
Strategy
Driving
Avoidance
Mechanical
Surface
Battery
Experimental
Static
Engine
Sensing
Stochastic
Agents
Actuation
Adaptive
Monitoring
Combustion
Unknown
Actuator
Development
Multi-Agent
Advanced
Steering
Optimization
Robot
Electric
Dynamics
Automotive
Ground
Sensor
Nonlinear
Wind
Actuators
Processes
Behavior
Powertrain
Study
Autonomous
Data
Controllers
Piezoelectric
Lateral
Validation
Varying
Robots
Internal
Multiple
Input
Generation
System

Designing an LQR controller for balancing a two-wheeled bike

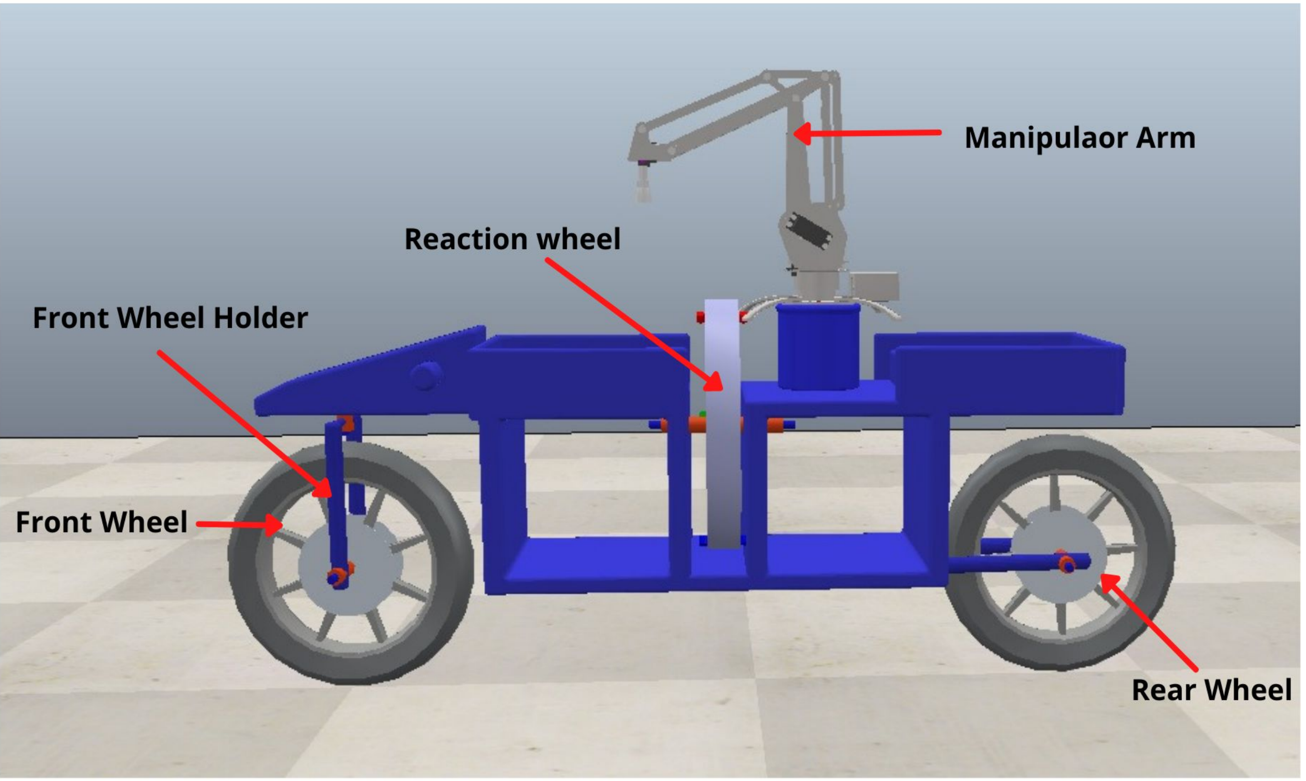
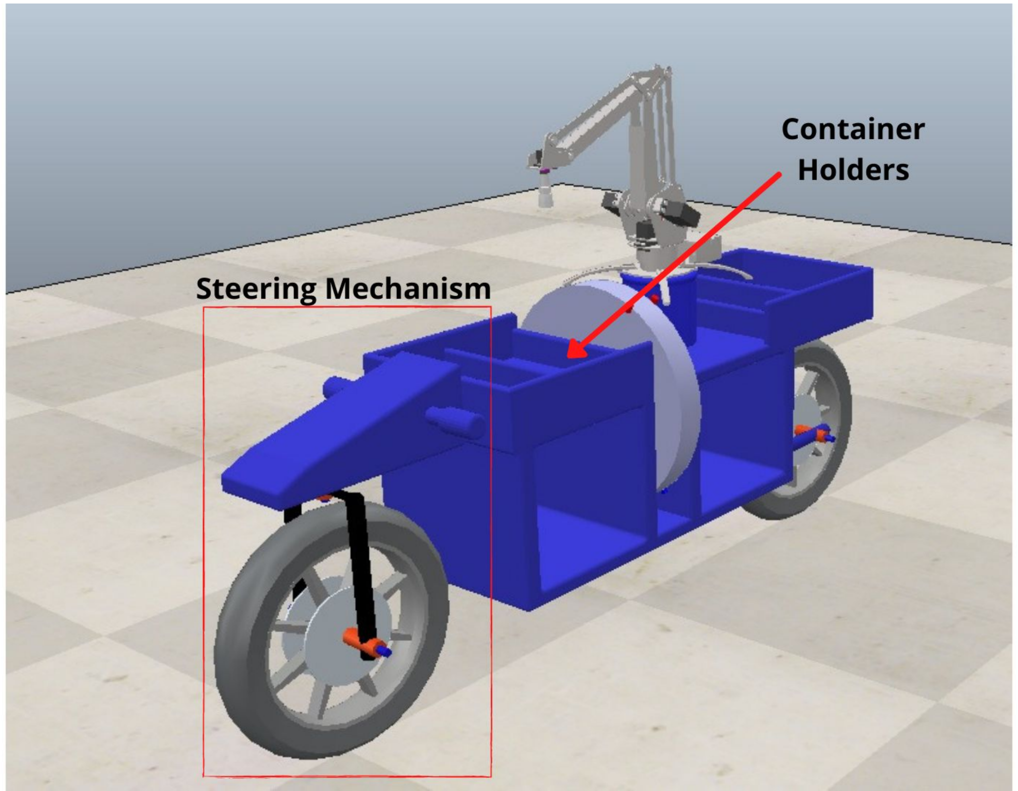
Authors

Authors:

- Jash Shah
- Aryaman Shardul
- Ayush Kaura

Objective

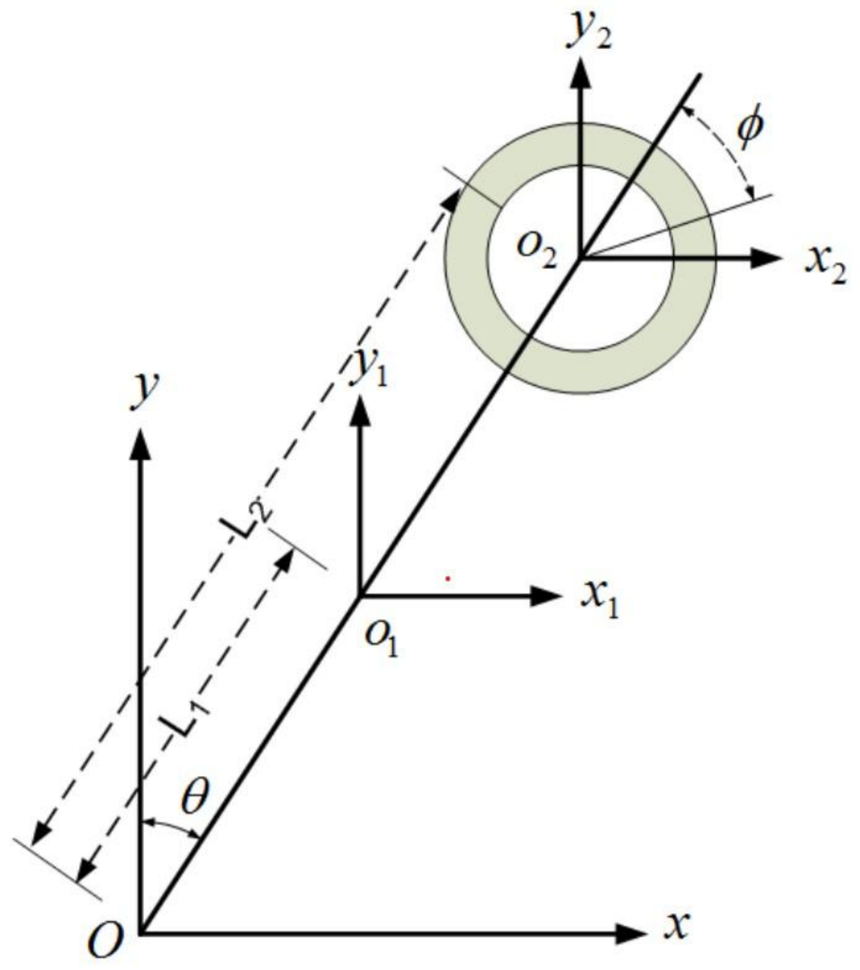
- The goal of this project is to design a two inline-wheel bike simulation capable of balancing itself using a reaction wheel.
- The simulation for this presentation is of an inverted pendulum which uses a control system to keep itself upright.
- The system is able to take into account external variations and using its feedback can change its state accordingly, i.e. in order to maintain the balance, the robot reads sensor input to detect tilt angle and correctly reacts to maintain gyroscope for steady vertical position.


Manipulaor Arm
Reaction wheel
Front Wheel Holder
Front Wheel
Rear Wheel

Container Holders
Steering Mechanism

Overview

- The first step in this project is to analyse the system and figure out the factors that affect the system. Once we identify the variables, we can create a mathematical model for the system.
- The next step is to linearly transform the system to eigenvalue coordinates by using diagonalization.
- By studying the effect of the eigenvalues on the stability of the system, we can design a controller for the same.

The Inverted Pendulum Reaction Wheel System



1 State Space Analysis

1. **Select States of System**

$$X = \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} \text{Angle between pendulum and vertical} \\ \text{Angular velocity of pendulum} \\ \text{Angle made by reaction wheel w.r.t pendulum} \\ \text{Angular velocity of reaction wheel} \end{bmatrix}$$

2. **State Space Model**

$$\dot{X} = \begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \end{bmatrix} \cdot \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix}$$

$$\therefore \dot{X} = A \cdot X$$

$$\therefore \frac{dX}{dt} = A \cdot X$$

$$\therefore X(t) = e^{At} \cdot X(0)$$

$$e^{At} = I + At + \frac{A^2t^2}{2!} + \frac{A^3t^3}{3!} + ... \tag{1}$$

But this expansion will be tough to calculate and hence the equations become unnecessarily complex.
So instead we can make use of diagonalization to linearly transform our system to eigenvalue co-ordinates.

$$\text{i.e } X = P \cdot Z \text{ (P = Modal Matrix)}$$

$$\therefore \dot{X} = P \cdot \dot{Z} = A \cdot X$$

$$\therefore P \cdot \dot{Z} = A \cdot P \cdot Z$$

$$\therefore \dot{Z} = P^{-1} \cdot A \cdot P \cdot Z$$

Considering P is orthogonal Matrix

$$\dot{Z} = D \cdot Z$$

$$\text{D = Diagonal Matrix}$$

$$\therefore \dot{Z} = \begin{bmatrix} \lambda 1 & & & \\ & . & & \\ & & . & \\ & & & \lambda n \end{bmatrix} \cdot Z$$

Equations become decoupled and thus have simple to calculate solutions.

$$i.e\ \dot{z}1 = \lambda 1.z1$$

$$\dot{z}2 = \lambda 2.z2$$

$$.$$

$$.$$

$$.$$

Also,

$$Z(t) = e^{Dt} \cdot Z(0)$$

$$\therefore Z(t) = \begin{bmatrix} e^{\lambda 1} & & & \\ & . & & \\ & & . & \\ & & & e^{\lambda n} \end{bmatrix} \cdot Z(0)$$

Now,
Converting back to original state since it is important to preserve our required system state variables.
Thus, Using Power of Matrix Theorem in equation (1),

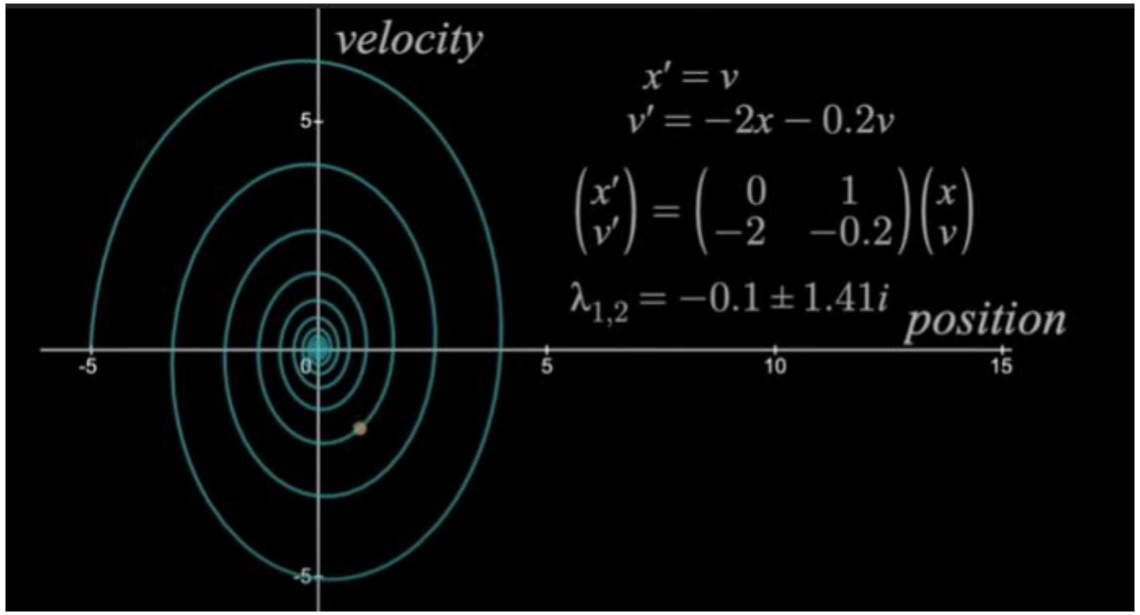
$$e^{At} = P^{-1} \cdot P + P^{-1} \cdot D \cdot Pt + \frac{P^{-1} \cdot D^2 \cdot Pt^2}{2!} + ...$$

$$\therefore e^{At} = P\left[I + D + \frac{D^2t^2}{2!} + \frac{D^3t^3}{3!} + ...\right]P^{-1}$$

$$\therefore e^{At} = P^{-1} \cdot e^{Dt} \cdot P$$

$$\therefore \boxed{X(t) = P^{-1} \cdot e^{Dt} \cdot P \cdot X(0)} \tag{2}$$

Effect of Eigenvalues



Negative Real Part corresponding to damping of system which results in Stability

Studying the Effect of Eigenvalues on Stability of System

From equation (2), we can infer that X(t) will be some linear combination of $e^{\lambda t}$ terms.

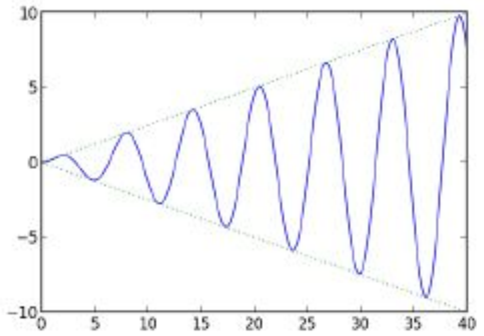
Now,

$$\lambda \in \Im$$

$$\therefore \lambda = a + ib$$

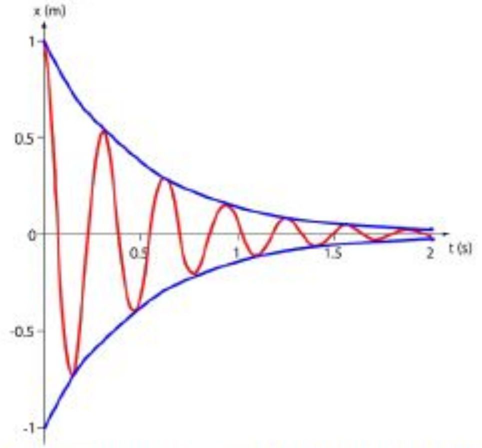
$$\therefore e^{\lambda t} = e^{at} \underbrace{(cos(bt) + i.sin(bt))}_{\text{always 1}}$$

if $a > 0$



System oscillations grow overtime so unstable.

if $a < 0$



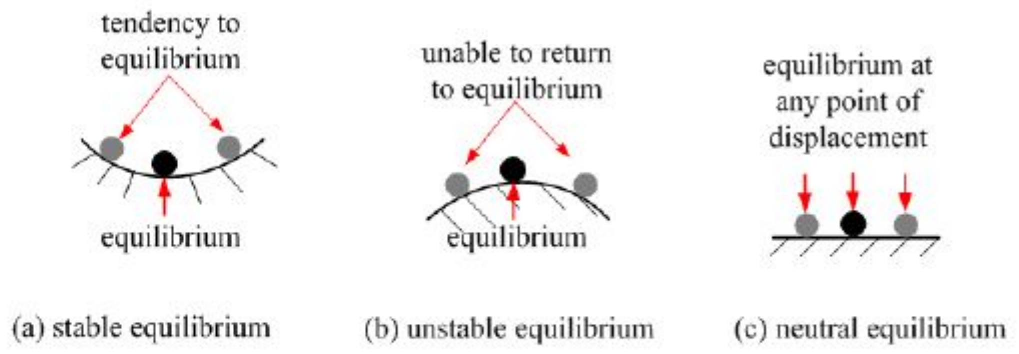
System oscillations increase overtime so reaches stability.

However, this modelling only works if the system is linear. But in most practical applications, the systems are non-linear. So, in order to linearize them we make use of Jacobians.

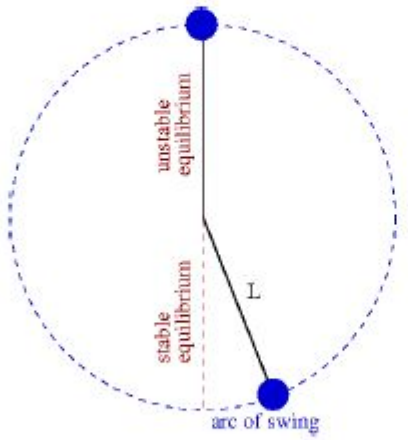
Linearizing Systems using Jacobian

Linearize around a fixed point.

(a) Find the stable equilibrium points for the system.



For our inverted pendulum system, the equilibrium are at:
$\theta = \pi$ *and* $\theta = 0$



Linearizing Systems using Jacobian

(b) Linearize around these equilibrium points:

i.e Non-Linear system acts linear when we look rally close to equilibrium points.

$$\text{Let } \bar{X} \Rightarrow \text{Equilibrium point}$$

For Non-Linear System,

$$\dot{X} = f(x)$$

where f(x) is a non-linear combination of state variables.
Around Equilibrium point,

$$\therefore \dot{X} = f(\bar{X}) + \left.\frac{Df}{Dx}\right|_{X} (X - \bar{X}) + \left.\frac{D^2 f}{DX^2}\right|_{X} (X - \bar{X})^2 + \ldots .$$

This is the Taylor-Series expansion around $\bar{X}$
where, $\left.\frac{Df}{DX}\right|_{X}$ is the Jacobian of $\dot{X} = f(X) at \bar{X}$

$$\left.\frac{Df}{DX}\right|_{\bar{X}} = \begin{bmatrix} \frac{\delta f1}{\delta x1} & \frac{\delta f1}{\delta x2} & . & . \\ . & . & . & . \\ . & . & . & . \\ \frac{\delta fn}{\delta x1} & . & . & \frac{\delta fn}{\delta xn} \end{bmatrix} at \bar{X}$$

Taking only the linear component of this expansion,

$$\dot{X} = \left.\frac{Df}{DX}\right|_{X} (X - \bar{X})$$

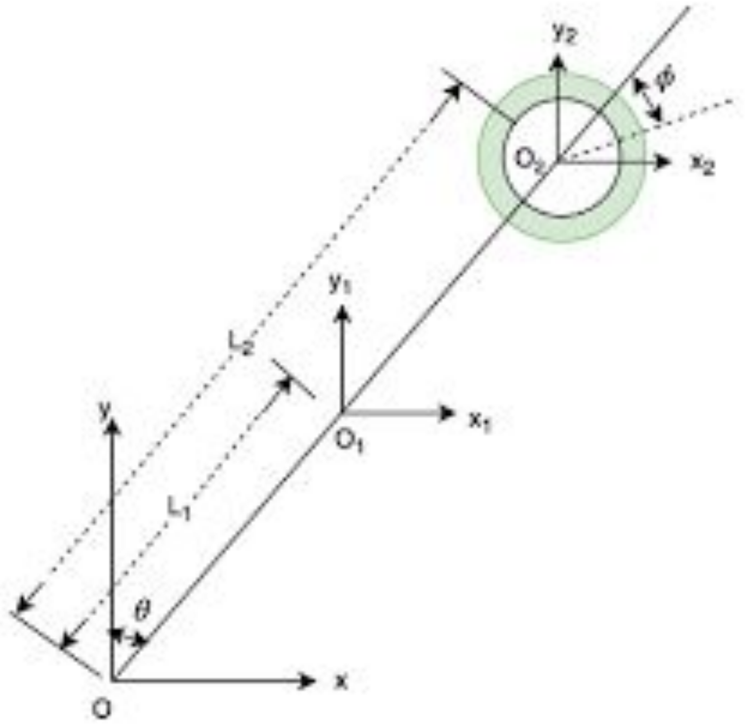
$$\therefore \dot{X} = A \cdot X$$

where,

$$A = \begin{bmatrix} \frac{\delta f1}{\delta x1} & \frac{\delta f1}{\delta x2} & . & . \\ . & . & . & . \\ . & . & . & . \\ \frac{\delta fn}{\delta x1} & . & . & \frac{\delta fn}{\delta xn} \end{bmatrix} at \bar{X}$$

Thus, we come to realize that for a Non-Linear system, the matrix which defines our state space model linearly is basically just the Jacobian of the system taken at the equilibrium point.

Modelling a Physical System



$L(q,\dot{q}) = KE(q,\dot{q}) - PE(q,\dot{q})$ …….. (Lagrangian Operator)

$$L \;=\; \frac{1}{2}(m_1L_1^2 + m_2L_2^2 + I_1 + I_2)\dot{\theta}^2 + I_2\dot{\theta}\dot{\phi} + \frac{1}{2}I_2\dot{\theta}^2 \\ -(m_1L_1 + m_2L_2)\, g\; cos\theta$$

$\frac{\mathrm{d}}{\mathrm{d}t}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = \tau_i$ …….. (Lagrangian Equation for Dynamic Model)

$$(m_1L_1^2 + m_2L_2^2 + I_1 + I_2)\ddot{\theta} + I_2\ddot{\phi} - (m_1L_1 + m_2L_2)g\sin\theta = 0$$

$$I_2(\ddot{\theta} + \ddot{\phi}) = T_r$$

State Space Equation

The equation is a nonlinear mathematical model expression of the inverted pendulum system with a reaction wheel. In order to simplify analysis we linearize the equation about the unstable equilibrium when is ϴ very small, according to Jacobian linearization, we obtain:

$$\begin{bmatrix} \dot{\theta} \\ \ddot{\theta} \\ \dot{\phi} \\ \ddot{\phi} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ b/a & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -b/a & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \theta \\ \dot{\theta} \\ \phi \\ \dot{\phi} \end{bmatrix} + \begin{bmatrix} 0 \\ -1/a \\ 0 \\ (a+I_2)/(aI_2) \end{bmatrix} T_r,$$

where $a = m_1 L_1^2 + m_2 L_2^2 + I_1$

$b = (m_1 L_1 + m_2 L_2) g$

The above equation is of the form

$$\dot{x} = A\,x(t) + B\,u(t)$$

where, A – State Matrix B – Input Matrix

Controllability

Now that we have out System Modelled, we can work towards making a controller for the same.

$$\boxed{\dot{X} = A \cdot X + B \cdot u} \tag{3}$$

where,

$$X \in \Re^{n};$$

$$A \in \Re^{nxn};$$

$$B \in \Re^{nxq};$$

$$u \in \Re^{q};$$

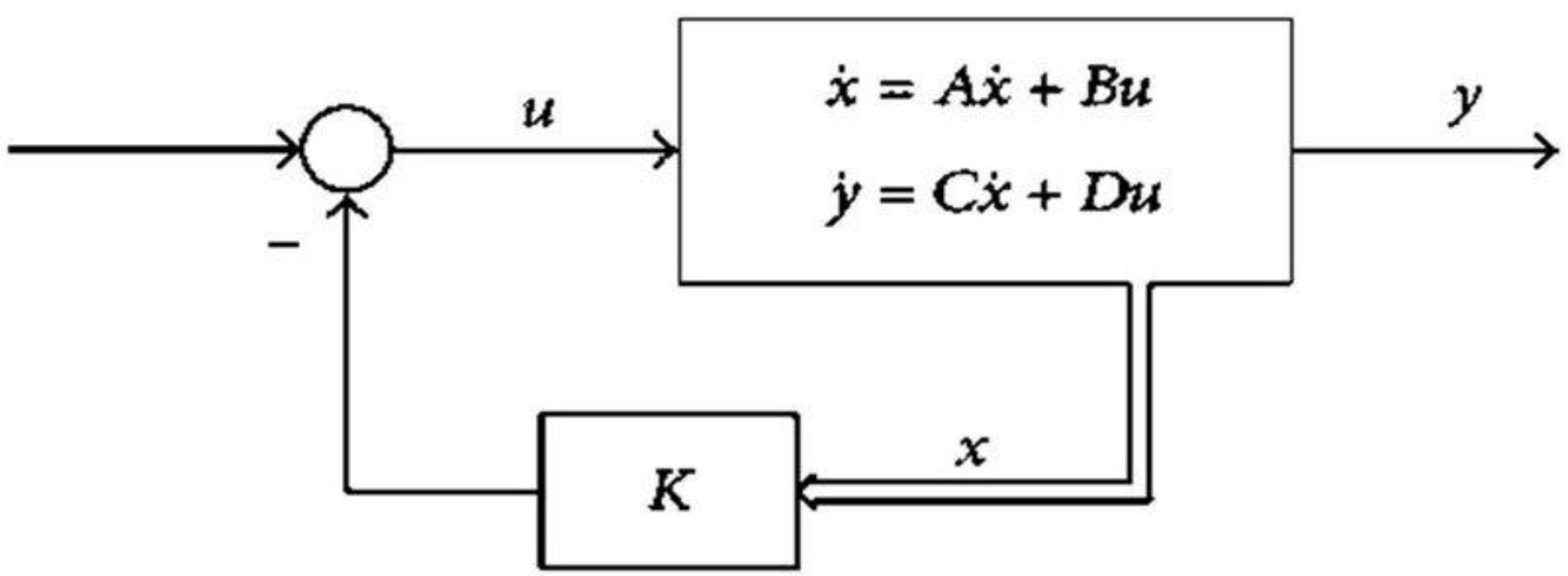
Controllability Matrix C

$$C = \begin{bmatrix} B & A.B & A^2.B & . & . & . & A^{n-1}.B \end{bmatrix}$$

For system to be controllable,

$$\boxed{rank(C) = n}$$

LQR Controller



i.e. U = -K . X

$$\dot{X} = (A - B.K).X$$

Pole Placement

- One method to ensure that the system is stable is to select the gain matrix K in such a way so that the eigenvalues of the (A-BK) matrix are purely real and negative.
- We can select the desired eigenvalues for the system and calculate the K matrix such that (A-BK) has our desired eigenvalues.

$$A-BK = \begin{bmatrix} 0 & 1 & 0 & 0 \\ b/a & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -b/a & 0 & 0 & 0 \end{bmatrix} - \begin{bmatrix} 0 \\ -1/a \\ 0 \\ (a+I2)/(a*I2) \end{bmatrix} * \begin{bmatrix} K1 & K2 & K3 & K4 \end{bmatrix}$$

$$A-BK = \begin{bmatrix} 0 & 1 & 0 & 0 \\ (b+K1)/a & K2/a & K3/a & K4/a \\ 0 & 0 & 0 & 1 \\ -b/a-((a+I2)*K1)/(a*I2) & -((a+I2)*K2)/(a*I2) & -((a+I2)*K3)/(a*I2) & -((a+I2)*K4)/(a*I2) \end{bmatrix}$$

- Now,

$$A\text{-}BK\text{-}\lambda I = \begin{bmatrix} -\lambda & 1 & 0 & 0 \\ (b+K1)/a & K2/a-\lambda & K3/a & K4/a \\ 0 & 0 & -\lambda & 1 \\ -b/a-((a+I2)*K1)/(a*I2) & -((a+I2)*K2)/(a*I2) & -((a+I2)*K3)/(a*I2) & -((a+I2)*K4)/(a*I2)-\lambda \end{bmatrix}$$

- Here values of ƛ are the eigen values.
- Now, we put det(A-BK-ƛI) = 0 and form a equation.
- Select arbitrary values of ƛ as roots of the equation.
- Substitute the values of ƛ in the equation obtained and find the values of K1, K2, K3, K4 to form the K matrix.
- Note that while selecting values of ƛ, it's real part must be negative for it to be stable and positive for it to be unstable.
- The K matrix for our system turns out be:

$$K = \begin{bmatrix} -1.225643 & -0.051416 & 0.850230 & 1.099185 \end{bmatrix}$$

LQR

- Linear Quadratic Regulator(LQR) helps us optimize the K matrix according to our desired response.
- Here we use a cost function,

$$J = \int_0^\infty (x^T Q x + u^T R u)$$

- Where, Q and R are positive semi-definite diagonal matrices and x and u are the state vector and input vector respectively.
- The controller is of the form $u = -Kx$ which is a **Linear** controller and the underlying cost function is **Quadratic** in nature and hence the name **Linear Quadratic Regulator**.

LQR

- Each Qi are the weights for the respective states xi.
- The trick is to choose weights Qi for each state xi so that the desired performance criteria is achieved. Greater the state objective is, greater will be the value of Q corresponding to the said state variable.
- LQR minimizes this cost function J based on the chosen matrices Q and R.

The End

Thank You!