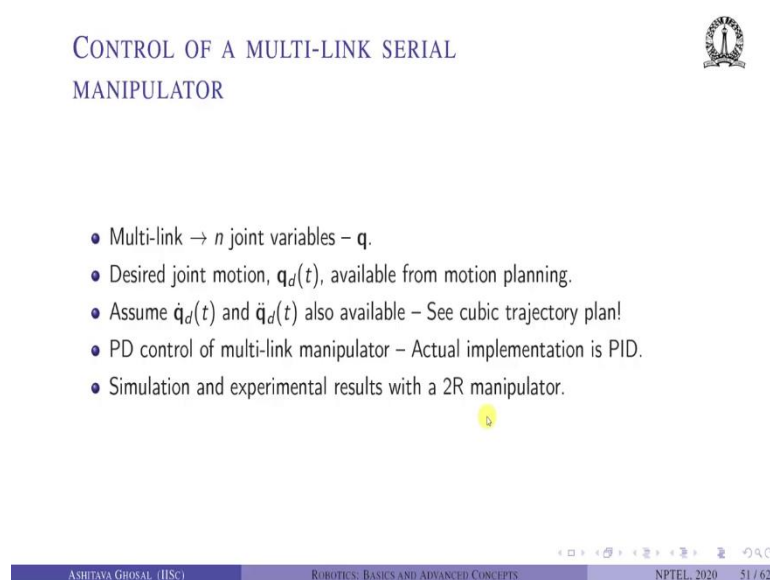


Robotics: Basics and Selected Advanced Concepts
Prof. Ashitava Ghosal
Department of Mechanical Engineering
Indian Institute of Science, Bengaluru

Lecture - 33
Control of a multi-link serial manipulator


Welcome to this NPTEL lectures on Robotics - Basic and Advanced Concepts ok. In the last lecture, we had looked at Control of a single link using linear control schemes. In this lecture, we will look at Control of a multi-link serial manipulator again using linear control schemes ok.

(Refer Slide Time: 00:40)



CONTROL OF A MULTI-LINK SERIAL
MANIPULATOR

- Multi-link $\rightarrow n$ joint variables – \mathbf{q} .
- Desired joint motion, $\mathbf{q}_d(t)$, available from motion planning.
- Assume $\dot{\mathbf{q}}_d(t)$ and $\ddot{\mathbf{q}}_d(t)$ also available – See cubic trajectory plan!
- PD control of multi-link manipulator – Actual implementation is PID.
- Simulation and experimental results with a 2R manipulator.



◀ ◁ ▷ ▶

ASHITAVA GHOSAL (IISc) ROBOTICS: BASICS AND ADVANCED CONCEPTS NPTEL 2020 51 / 62

So, in the case of a multi-link serial manipulator, we have n joint variables which will be denoted by \mathbf{q} . We have the desired joint motion \mathbf{q}_d available from motion planning. We also assume that $\dot{\mathbf{q}}_d$ and $\ddot{\mathbf{q}}_d$ are also available ok. So, when we did this motion planning using cubic trajectory, we could take the derivative of the cubic polynomial and obtained $\dot{\mathbf{q}}_d$. We can take again derivative and obtained $\ddot{\mathbf{q}}_d$. So, these are available.

So, in the lecture, we will look at PD control of a multi-link manipulator; the actual implementation is PID. So, as discussed, we also sometimes add or we not sometimes we always add a I part which will get rid of the steady state errors. But as far as analysis is

concerned, we stick to PD because it is simpler to do ok; I increases the order of the system. So, we will look at some simulation and experimental results with a 2R manipulator.

(Refer Slide Time: 01:55)

PD CONTROL OF MULTI-LINK SERIAL MANIPULATOR

- Extend continuous time control of single link manipulator.
- Feed-forward plus PD instead of PID control algorithm for analysis

$$V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t)$$


Implemented control will also have a integral term!

- Use torque τ acting at the joint instead of voltage V_a in analysis³.
- Control law used in analysis

$$\tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t), \quad e(t) = q_d(t) - q(t)$$

- Linear control law applied to a non-linear system!

³Joint torque is related to the applied voltage at the motor terminals since $T_m = K_t i_a = (K_t/R_a)(V_a - K_g \dot{\theta}_m)$ and $\tau = T_m/n$. One can also find V_a from motor characteristics curves.



ASHITAVA GHOSAL (IISc)
ROBOTICS: BASICS AND ADVANCED CONCEPTS
NPTEL, 2020 52 / 62

So, PD control of multi-link serial manipulator, we extend this whole notion of continuous time control of a single manipulator ok; single link manipulator. So, what do we have? We have the voltage which is applied is $V_a(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t)$.

So, $e(t) = q_d(t) - q(t)$. This is implemented actually with an integral term; but as I mentioned, we will not consider the integral term in this analysis. Other important thing is we actually use torque acting at the joint instead of the voltage in the analysis ok. So, it is this very serious no right because the torque is related to the current ok. $T_m = K_t i_a$.

Then, this current is related to the voltage applied, there is a back emf. Nevertheless, we can think of that voltage and torque are related ok. We can also obtain this voltage from the motor characteristics ok. We get the torque speed curve and that is for every voltage, there is a certain torque at a different speed ok.

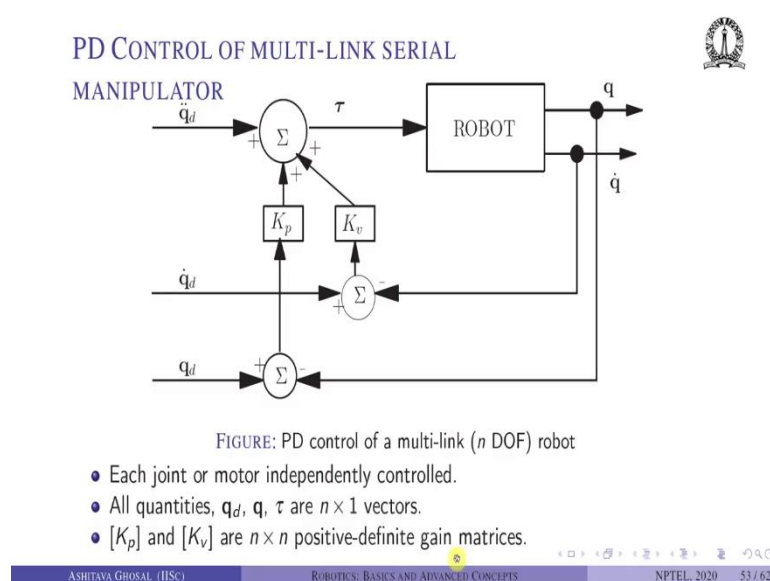
So, as far as analysis is concerned, for the sake of simplicity, we will assume that we are inputting torque not voltage. So, what is the control law which we will use? That it is $\tau(t) = \ddot{q}_d(t) + K_v \dot{e}(t) + K_p e(t)$.

Actually, strictly speaking it should of been voltage which is related to motor torque in some manner; but we are going to assume that it is torque that you are applying and again,

$e(t)$ is the error between what is desired and what is measured and we will see what a Linear control law does when you are applying it to a non-linear system, why?

Because a multi-link robot is known to be a non-linear system. Remember the equations of motion of a multilink serial manipulator contain all kinds of non-linearities; you know $\dot{\theta}_1, \dot{\theta}_2$ in the case of 2R, then sine and cosine of the angles quite a few non-linear terms were there.


(Refer Slide Time: 04:24)



So, how does it look like? The PD control of a multi-link serial manipulator using a linear control scheme looks like this. So, I have this robot, the input is torque, the output we are going to measure \mathbf{q} which is the rotation at the joints and $\dot{\mathbf{q}}$ which is the velocity of the at the joints. Then, we do $\mathbf{q}_d - \mathbf{q}$. So, this is the error multiplied by K_p and then, $\dot{\mathbf{q}}_d - \dot{\mathbf{q}}$ measured multiplied by K_v and to that, we add this $\ddot{\mathbf{q}}$.


So, each joint or motor is independently controlled ok and all the quantities \mathbf{q}_d , \mathbf{q} and $\boldsymbol{\tau}$ are $n \times 1$ vectors. So, if you have a n -degree of freedom serial robot, these are $n \times 1$ vectors and K_p and K_v are $n \times n$ positive definite gain matrices ok. Most, it could be diagonal which is easier to analyze. If it is non diagonal, then there is some coupling; nevertheless, they are positive definite gain matrices. The gains must be positive.

(Refer Slide Time: 05:46)



PD CONTROL OF MULTI-LINK SERIAL MANIPULATOR

- Multi-link manipulator, non-linear system → No *uniform* damping and settling time *everywhere* in workspace.
- PD controls works due to slow speed and large gear ratio at joints!
- Linear control law using one or more microprocessors
- Two main kinds of architecture commonly used in the past.
 - *Joint parallel* – Each joint (PID) controlled by a micro-processor & additional master or ‘coordinating’ processor for GUI, data logging etc.
 - *Functional parallel* – Each/group of function(s)/task(s) handled by a processor.
- Original PUMA robot – 6503 microprocessor at joints and DEC LSI-11 for master, θ_d available every 28 msec, T_s for joint processor was 0.875 msec.
- Modern solution – controller based on industrial PC's to control several joints.



ASHITAVA GHOSAL (IISc) ROBOTICS: BASICS AND ADVANCED CONCEPTS NPTEL, 2020 54 / 62

So, the multi-link manipulator is a non-linear system. We have looked at the equations of motion and they are non-linear and as a result, what will happen is you will not get uniform damping and settling time everywhere in the work space ok. So, the performance of the robot in terms of damping settling time and so on will be different at different places. Because basically the manipulator system is different at different places ok. PD controls still works because basically, we run these robots at slow speed most of the time ok.

And also, because we have large gear ratios. Remember in one of the examples earlier and we have modeling the single link, we at a large gear ratio and due to the large gear ratio, the disturbance torque T_d which is basically the torque acting from further away links onto the link which we are interested is reduced by the gear ratio ok.

So, that is the reason this PD control often works and there are still many manipulators which are serial manipulators, which are manufactured, which use PD control laws, linear control laws. The linear control is most of the time implemented using one or more microprocessors ok.

And earlier in the original days of robotics, they were two main kinds of architecture which were used; one is what is called as a joint parallel architecture. So, each joint is controlled by a microprocessor, an additional master or a coordinating processor is used for graphic user interface, data logging and various other things.

There was for a short while something called as a functional parallel architecture, where each group of functions and tasks were handled by one processor. So, for example, the inverse kinematics was done by one processor; motion planning was done by one processor and so on.

Most of the time, this joint parallel or independent control of each joint is the common way of controlling a serial multilink serial robot. The original PUMA robot for example, had something called as a 6503 microprocessor at each joint. Most of you would not have heard of what is a 6503 microprocessor.

But 50 years back, this was well-known. The master computer or the “coordinating” micro processor was a DEC LSI-11, this is also does not exist anymore ok. θ_d was available every 28 milliseconds ok. So, we compute or do this motion planning and every 28 milliseconds and new θ_d was available.

And the sampling time for the joint processor was 0.875 microseconds ok. Remember we have to calculate the error; we have to sample the input and sample the output that T_s is 0.875 milliseconds. The modern solution is basically a controller based on industrial PC’s to control several joints ok. So, we might have a card which controls all the joints, we might have another card which does may be all the GUIs ok or controls all the other peripheral devices.

(Refer Slide Time: 09:26)

PD CONTROL – 2R MANIPULATOR



Equations of motion are two *nonlinear ODEs* – In *standard form*

$$\begin{pmatrix} \tau_1 \\ \tau_2 \end{pmatrix} = \begin{bmatrix} I_1 + I_2 + m_2 l_1^2 + m_1 r_1^2 + m_2 r_2^2 + 2m_2 l_1 r_2 c_2 & I_2 + m_2 r_2^2 + m_2 l_1 r_2 c_2 \\ I_2 + m_2 r_2^2 + m_2 l_1 r_2 c_2 & I_2 + m_2 r_2^2 \end{bmatrix} \begin{pmatrix} \ddot{\theta}_1 \\ \ddot{\theta}_2 \end{pmatrix} + \begin{pmatrix} -m_2 l_1 r_2 s_2 (2\dot{\theta}_1 + \dot{\theta}_2) \dot{\theta}_2 \\ m_2 l_1 r_2 s_2 \dot{\theta}_1^2 \end{pmatrix} + \begin{pmatrix} m_2 g (l_1 c_1 + r_2 c_{12}) + m_1 g r_1 c_1 \\ m_2 r_2 g c_{12} \end{pmatrix}$$

- Desired joint trajectories $\theta_{i,d}(t)$, $i = 1, 2$ specified.
- Linear control equations used

$$\tau_i(t) = \ddot{\theta}_{i,d}(t) + K_v \dot{e}_i(t) + K_p e_i(t), \quad e_i(t) = \theta_{i,d}(t) - \theta_i(t), \quad i = 1, 2$$

So, let us continue with the PD control of a 2R manipulator. The equations of motion of a 2R manipulator were derived earlier ok, but I am repeating them again. So, τ_1 and τ_2 is given by some mass matrix times $(\ddot{\theta}_1, \ddot{\theta}_2)^T$ some centripetal Coriolis term and a gravity term ok.

So, this mass matrix is 2×2 , it is positive definite symmetric and so on. The desired joint trajectories are specified, nice cubic smooth trajectories and what we are going to use is a linear control which is $\tau_i(t) = \ddot{\theta}_{id}(t) + K_v \dot{e}_i(t) + K_p e_i(t)$; where $e_i(t) = \theta_{id}(t) - \theta_i(t)$. So, on the left hand side, this τ_1 and τ_2 will be substituted by this.

(Refer Slide Time: 10:29)

2R MANIPULATOR: SIMULATION

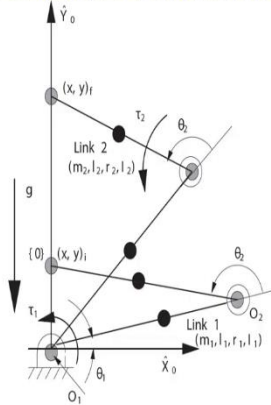


FIGURE: A planar 2R robot

- Planar 2R robot shown in 2 configurations.
- Link 1 parameters – $l_1 = 1m$, $r_1 = 0.773m$, $m_1 = 12.456kg$ and $I_1 = 1.042 \text{ kg} - m^2$.
- Link 2 parameters – $l_1 = 1m$, $r_1 = 0.583m$, $m_1 = 12.456kg$ and $I_1 = 1.042 \text{ kg} - m^2$.
- Payload at the end 2.5 kg.

ASHITAVA GHOSAL (IISc)
ROBOTICS: BASICS AND ADVANCED CONCEPTS
NPTEL 2020 56 / 62

So, let us take an example. So, we have a 2R manipulator. So, basically there are two joints, 1 and 2; so, at O_1 and O_2 . First joint rotates at θ_1 , second joint rotates at θ_2 ; relative rotation, τ_1 is the torque which is acting at the first joint; τ_2 is the torque which is acting at the second joint.

So, we have mass length of the first link location of the CG of the first link r_1 , I_1 which is the z component of the inertia of the first link. Likewise, m_2 , l_2 , r_2 and I_2 which is of the second link ok. We have to choose some parameters. And remember in the case of dynamics, we are chosen l_1 and l_2 as 1 meter. We are sticking to the same length links locations of the CG and the masses and inertia ok. So, m_1 was 12.456 kg; I_1 was 1.042 kg meter square and so on ok.

We will also assume that there is a payload at the end, at this (x, y) and the trajectory is that we want to go from (x, y_i) to (x, y_f) and then, come back and the gravities are acting this way down along the Y axis.

(Refer Slide Time: 11:58)

DESIRED TRAJECTORY



- Tip moves up from $(0, 0.55m)$ to $(0, 1.45m)$ and back to $(0, 0.55m)$.
- Two cases: (a) *fast*: total time is 2 sec, (b) *slow*: total time is 2 min.

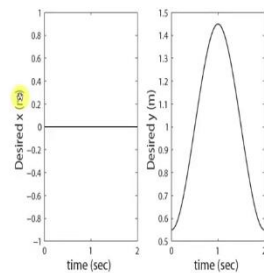


FIGURE: Desired Cartesian trajectory

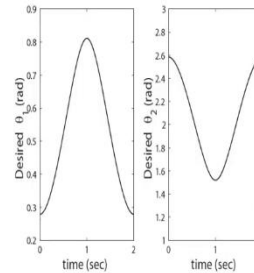


FIGURE: Desired $\theta_1(t)$ and $\theta_2(t)$

So, the tip moves from $(0, 0.55)$ meters to $(0, 1.45)$ meters. It lies completely in the work space and back to $(0, 0.55)$ meters and we will consider two cases in the simulation; one which is fast. So, the total time is 1 second. So, it goes up in 1 second, comes down in 1 second ok. It is pretty fast. So, it is like of the order of 1 meter per second speed and another trajectory which is very slow which is 1 meter per minute of the order of 1 meter per minute ok.

So, what is the desired trajectory? The x would be 0 and we will fit a nice smooth cubic trajectory in the y direction. So, it goes from 0.55 to 1.45 in a cubic profile in 1 seconds and again, it comes down back to 0.55 as in a nice smooth cubic profile in between 1 and 2 seconds ok. Now, for these trajectories, we can find the desired θ_1 and θ_2 by doing inverse kinematics ok.

(Refer Slide Time: 13:12)

2R MANIPULATOR: PD CONTROL



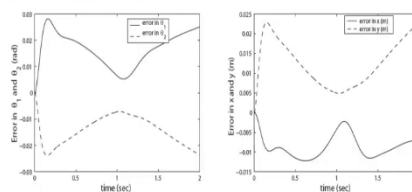
- Desired $\theta_{id}(t)$, $i = 1, 2$ and derivatives obtained using *inverse kinematics*
- Simulation results presented for PD control scheme
- Gain values K_{pi} , K_{vi} are chosen such that $\omega_1 = 85.0$, $\omega_2 = 75.0$, and ξ_i are 2.0
→ System over-damped.



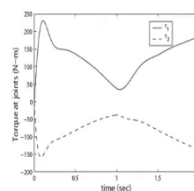
So, we obtained the desired $\theta_{id}(t)$ using inverse kinematics and then, we perform simulations for PD control scheme ok. The gain values K_{pi} and K_{vi} are chosen such that ω_1 and ω_2 are 85 and 75 arbitrarily ok. These are the two natural frequencies. Remember for a second order system, we can describe everything by natural frequency and damping and the ξ_i is chosen 2.0. We basically want over damped ok, we do not want oscillations as it goes to the top.

(Refer Slide Time: 13:51)

PD CONTROL – FAST MOTION



(a) Error in θ_1, θ_2 for fast motion (b) Error in x, y for fast motion



(c) Torques for fast motion

So, we can perform these simulations. We go back to the equations of motion, solve the equation of motions, integrate the equations of motion because we can see everything is known now. So, in this $\theta_{id}(t)$ is known, this is $\theta_{id}(t) - \theta_i(t)$; so, $\theta_{id}(t)$ is known, $\dot{\theta}_{id}(t)$ is known and then, we can rearrange this equation in state space form.

The output of the integration will be $\theta(t)$ which you can subtract and then basically, we can do everything and we can find θ as a function of time ok for the chosen K_p, K_v , for the chosen I_1, I_2, m_2, l_1 or all the parameters ok. The gain values were chosen arbitrarily like this. We could have chosen some other gain values also. So, let us plot θ_1 and θ_2 for the fast motion.

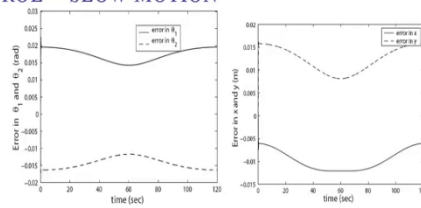
So, as you can see the error in θ_1 goes up ok, as you are going up. So, up till 1 second, it is going up and from 1 to 2 second, it is coming down. So, you can see that the error is of the slightly less than 0.03 radians ok; the Y axis is in radians. So, that is quite a bit 1 radians is 57 degrees approximately. So, 0.03 is like almost how much? 1.5 degrees ok. The error in x and y can also be plotted. What do we want?

The desired x should be always 0, it should just go vertically up and down. However, because of this coupling and because of this this is actually a control, you will never get the exactly desired trajectory. So, we can see that there is an error in x and there is also an error in y and this is of the order of 2 centimeters between 2 and two-and-half centimeters in x and within 1 and one-and-half centimeters in y .

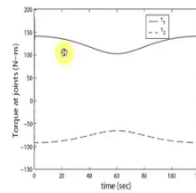
And what is the plot of torque? Because we can once we have solved, we can plot the torque. The torque is of the order of 200 and maximum torque is of the order of 225 Newton meters in τ_1 and the second torque will be less because it is expected. We expect that the second motor to provide less torque, it is seeing less inertia. It is of the order of 150 Newton meter ok. So, the important thing are these nature of these plots.

(Refer Slide Time: 16:37)

PD CONTROL – SLOW MOTION



(d) Error in θ_1, θ_2 for slow motion (e) Error in x, y for slow motion




(f) Torque for slow motion

Whereas, if you look at the slow motion which was you go up in 1 minute and come down in another 1 minute as opposed to 1 second and 1 second, we can see that the error is smaller ok. Previously, the error was between 0.02 and 0.03 radians. The error in x was between around 0.025 ok. So, now the error is less than 0.02 ok; the error in x is less than 0.15; so, 1.5 centimeters ok.

The torques along the trajectory is also smaller. So, instead of previously, we were reaching about 225 or 230 Newton meters, now we can do with less than 150 Newton meters ok.

(Refer Slide Time: 17:37)

PD CONTROL (CONTD.) 

- Maximum error in joint variables larger in case of *fast* motion.
 - Approximately 0.03 rad in fast versus 0.02 rad in slow motion.
 - Approximately 0.023 m in fast versus 0.016 m in slow motion.
- Fast motion → Non-linear inertia, centripetal/Coriolis terms larger
- Linear PD control less effective as *expected!*
- Maximum torque at the joints is larger – Approximately 225 N-m versus 145 N-m
- Torque larger in fast motion due to non-linear terms in equations of motion!
- Curves much smoother in slow motion.

ASHITAVA GHOSAL (IISc) ROBOTICS: BASICS AND ADVANCED CONCEPTS NPTEL, 2020 61 / 62

So, what are we seeing that the maximum error in the joint variables are larger in case of fast motion, they are approximately 0.03 radians in fast versus 0.02 in slow motion ok. Approximately, 0.023 meters in the x whereas, is 0.016 in the slow motion and why is this? Because in the fast motion, we have non-linear inertia, centripetal, Coriolis terms which are much larger ok.

So, the non-linear terms are significantly larger, when it is moving faster. We are using a linear controller ok. So, we are trying to control a non-linear system using a linear controller ok. So, if the non-linearity is more, the performance will be worse that is expected and in a fast motion, the Coriolis term, the centripetal term and the inertia term at much larger than if it is moving slowly.

So, hence, we expect in the fast motion the performance to be worse than when it is moving slowly. So, the linear PD control is less effective for fast motion as expected. Moreover, the maximum torque at the joint is larger ok. This is also because we are seeing larger Coriolis and centripetal and inertia terms ok.

So, its approximately 225 Newton meter versus 145 Newton meters. So, the larger torque in fast motion is due to non-linear terms in the equations of motion as expected ok. Also, the curves are much smoother in slow motion.

(Refer Slide Time: 19:30)

Proportional (P) and derivative(D) control of 2 R planar robot



A description of the 2R planar robot.

- Link length
 - $L_1=150$ mm
 - $L_2=150$ mm
- Actuator details (Motor₁, Motor₂)
 - Maximum torque: 7.6 [Nm]
 - Dimensions: 51 x 32 x 39.5 mm
 - Weight: 105 [g]
 - Standby current: 68 [mA]
 - Stall current: 5.4 [A]
 - Reduction ratio: 362.88: 1
- Programming language: C
- IDE : Code Composer Studio
- Microcontroller: Texas Instruments Tiva C (TM4C123GH6PM)
- Control scheme used: PD control of motor velocity

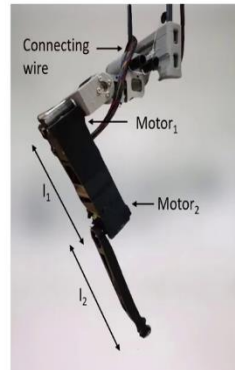


Fig: 2R Planar manipulator.

Pramod & Aditya @ RBCCPS

Till now, we have looked at simulation results of multi-link robots. In the next few slides, I will show you experimental results on a planar 2R robot and I will show you what are the experimental results for a PD control scheme. Remember we have looked at PD control scheme in which there is a proportional gain and a derivative gain, a little bit of description about the experimental set up. So, this is a 2 degree of freedom planar robot. It consists of two rotary joints. The link lengths are 150 mm each. The actuator details; there are two motors, motor 1 and motor 2 as I will show you later, these are 2 DEM motors made by Kondo ok.

The maximum torque which these motors can apply is 7.6 Newton meter. The dimension is quite small, it is 51 mm × 32 mm × 39.5 mm. The weight is about 105 grams. There is a standby current of 68 milli Amperes. There is a stall current of 5.4 Amperes and there is a reduction ratio.

These motors are controlled by a Texas Instrument Microcontroller which is TM4C123GH6PM. We will use programming language C to control the motion of these motors and the control scheme that we will use is PD control of motor velocity ok. This work was done by Pramod and Aditya at the Robert Bosch Centre for Cyber Physical Systems.

(Refer Slide Time: 21:18)

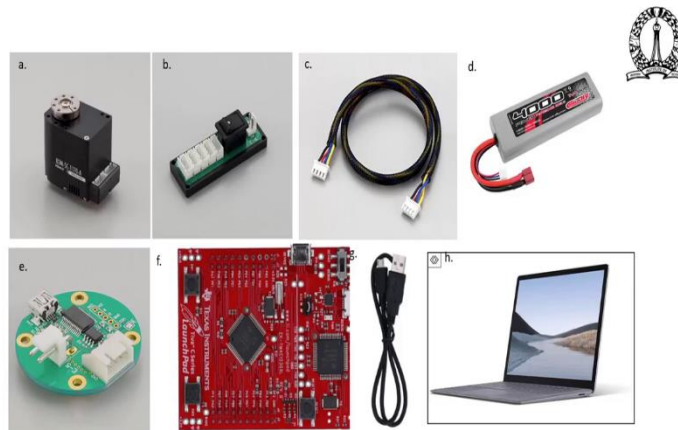


Fig: Component details: a) Servomotor(Kondo: B3M-SC-1170-A) , b) Hub type A for XH connector, c) XH connection cable, d) 11 v battery , e) RS485 USB serial conversion adapter , f) Microcontroller: Texas Instruments Tiva C (TM4C123GH6PM) , g) USB 2.0 cable, h) Laptop

Pramod & Aditya @ NBCCPS

So, here are some of the components. So, the servomotor is Kondo B3M-SC-1170-A. Anybody is interested later to purchase this or get hold of these motors ok; these are reasonably good servomotors, little bit expensive. But nevertheless, these are good servomotors ok. These there are some connectors which are called Hub type A for XH connectors, then there are some cables for this XH connections, we need a battery.

So, this is an 11 Volt lithium-ion battery. Then, we need some kind of a USB to serial conversion adapter, then we have this actual microcontroller which is made by Texas Instrument Tiva C ok, then you need all these cables and laptop. So, with these components and of course the links ok, we can do these experiments.

(Refer Slide Time: 22:18)

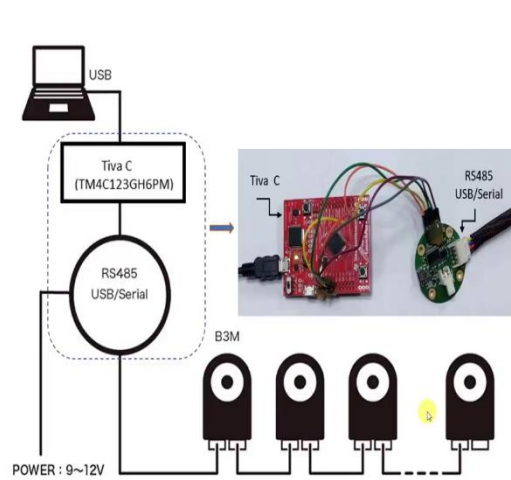
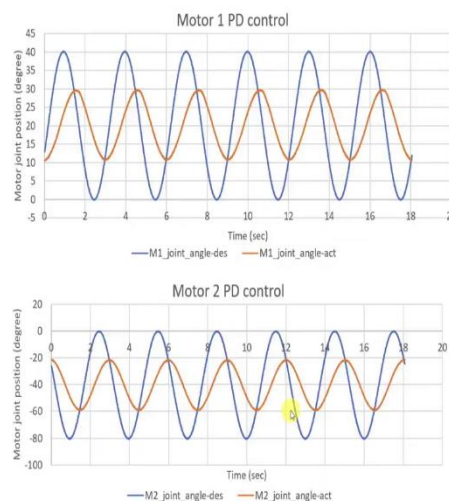


Fig: A schematic of the controller components and connections

Pramod & Aditya @ RBCCPS

So, the way it is connected is from the laptop, you connect using a USB cable to this micro controller, then using this RS485 USB to serial which also requires a power connection, which is this micro controller to this RS485 USB serial. This is the electronics part and you can connect to this B3M motors. So, we are going to connect to two such motors ok.

(Refer Slide Time: 22:49)



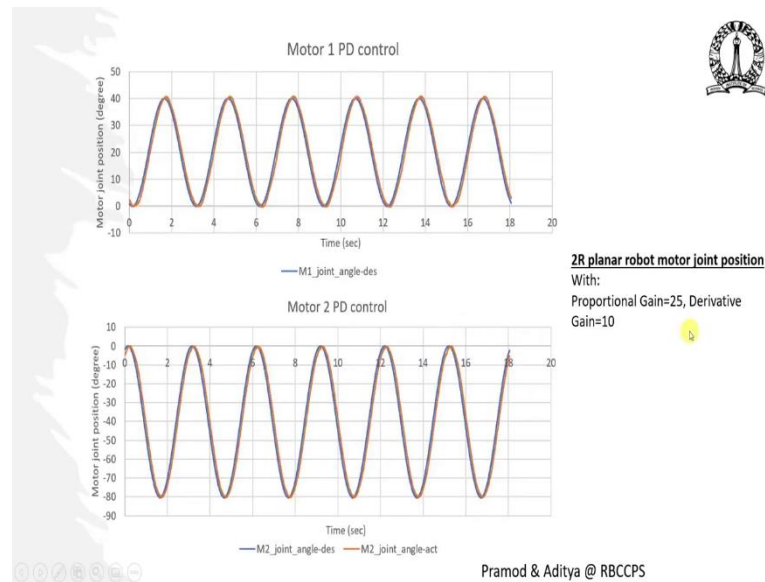
2R planar robot motor joint position
With:
Proportional Gain=2, Derivative
Gain=1

Pramod & Aditya @ RBCCPS

So, for PD control, we need to give a desired motors joint position ok. So, the desired motor joint position for motor 1 and motor 2 are this blue sinusoidal lines ok. So, we want

the first joint which is the first motor to follow this blue trajectory as a function of time. We see that if you use a proportional gain of 2 and a derivative gain of 1 for these two motors, the achieved joint trajectory is this orange line ok. So, clearly, it is not doing a good job with these controller gains ok K_p as 2 and K_d as 1.

(Refer Slide Time: 23:44)



So, let us increase the controller gain. So, we have done several trial and errors and eventually, we obtained a proportional gain of 25 with a derivative gain of 10 and in that case, you can see the blue which was a desired trajectory and orange which is the actual achieved trajectory by the motor are more or less matching ok.

So, what have we done? We have taken this motor and planar 2R robot, we have given a desired joint trajectory for both joint 1 and join 2 and then, we have played around with the proportional and derivative gains till we achieve some reasonable matching between the desired and the measure trajectory. This measure trajectory is from measurement devise on the motor itself ok.

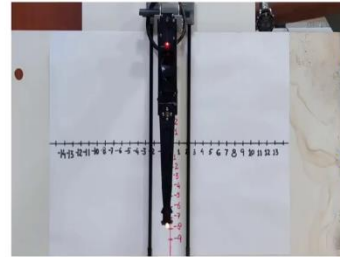
(Refer Slide Time: 24:31)



2-R planar manipulator -- Observation with different proportion and derivative gain setting



Video: Proportion gain (kp)=2, Derivative gain(kd)=1



Video: Proportion gain (kp)=25, Derivative gain (kd) =10

Pramod & Aditya @ RBCCPS

So, the left hand side shows the set up and also the right hand side, this is the 2R robot and we start with a proportional gain of 2 and a derivative gain of 1 ok. This simulation here shows the performance of this planar robot 2R robot, when the proportional gain is 25 and the derivative gain is 10 ok.

So, you can see that there is a scale which is drawn which gives you an idea of how much is the error. So, as you can see when the gains are small or when the gains are low, the motion is very very uneven ok. It is not really tracking this desired trajectory along the Y axis ok; whereas, if you have higher gains, you can see that the motion is much more smoother and it is tracking the trajectory reasonably well ok.

So, we can do similar experiments with other systems and the moral of the story is that we need to play around with the controller gains till we achieve a desired trajectory ok.

Thank you.