FORCE CONTROL

Andreja Rojko

OUTLINE

- Introduction
- Stiffness control
  - o Stiffness control for 1-DOF
  - o Robot dynamics with contact forces
  - o Stiffness control for n-DOF
- Hybrid position/force control
  - o Example: Hybrid control of Cartesian 2DOF robot
- Final notes
- Exercise: Stiffness control for 2-DOF

Introduction

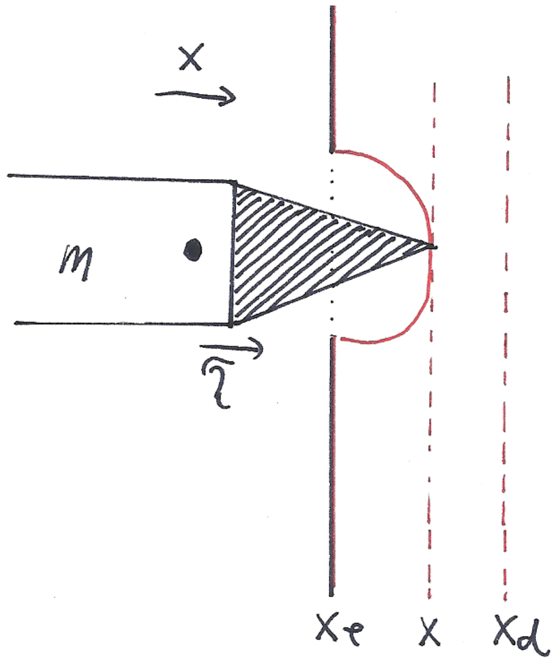
- Controllers derived until now are suitable only for tasks that require the robot to follow desired trajectory and don’t include the contact of robot with environment (spray painting, moving payload)= unconstrained motion.
- Other tasks, as for example polishing, assembling and simmilar require the contact of robot with the environment. This results in the contact forces.
- If stiffness of the environment is low it may be possible to control interaction by controlling the robot position. But if stiffness is high we cannot do this anymore and **force control** is needed.

Introduction

- **Force control**:
  - o Pure force control
  - o Hybrid control: two types
    - o Classic hybrid control: control force along some axis of robot and position along others,
    - o Impedance control, control of force and position for the same axis (won‘t be covered here).
- For complete force control six force components need to be measured; three translational components and three torques. Force/torque sensor is usually mounted at the robot wrist.

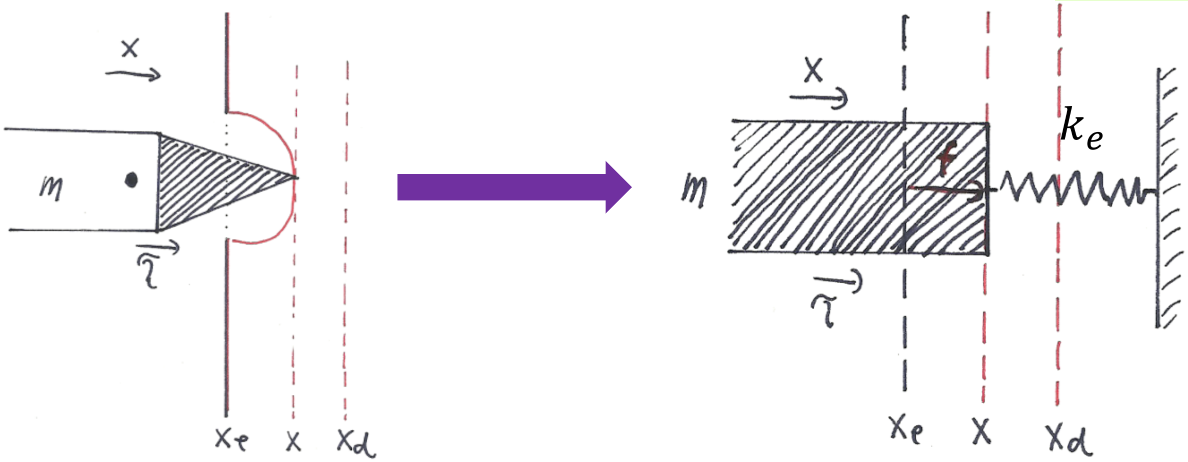
Stiffness control for 1 DOF

- In **stiffness (compliance) control** only the static relationship between the end-effector position and orientation error and the contact force/moment is considered.
- **Problem:** calculation of input force τ, so that 1 DOF robot moves from actual position x to desired position x_d. Desired position is inside of 'wall' with known stiffness.



Stiffness control for 1 DOF

- Environmental (wall) stiffness can be modelled as a linear spring with stiffness $k_e > 0$. Then force is $f = k_e(x - x_e)$.



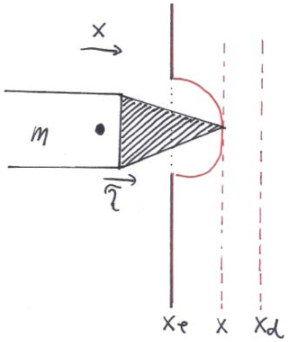
- If friction and gravitation are negligible then complete dynamics is linear: $f = m\ddot{x} + k_e(x - x_e)$
- PD controller seems like suitable choice for stabilizing x to x_d. $f = -k_v\dot{x} + k_p(x_d - x)$. k_v, k_p are positive scalar control gains.
- Closed loop dynamics: $m\ddot{x} + k_v\dot{x} + (k_p + k_e)x = k_p x_d + k_e x_e$

Stiffness control for 1 DOF

Calculated from $m\ddot{x} + k_v\dot{x} + \left(k_p + k_e\right)x = k_p x_d + k_e x_e$ is:

$$\mathrm{H}(s) = \frac{1}{\left(ms^2 + k_v s + \left(k_p + k_e\right)\right)}; \mathrm{X}(s) = \frac{k_p x_d + k_e x_e}{s\left(ms^2 + k_v s + \left(k_p + k_e\right)\right)}$$

For positive k_v, k_p, m, k_e are poles in left half of s-plane



Calculation of steady state position $\bar{x}$:

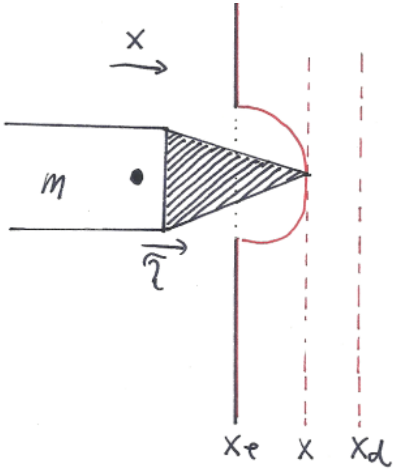
$$\bar{x} = \lim_{s \to 0} sX(s) = \frac{k_p x_d + k_e x_e}{k_p + k_e}$$

Steady state force can be calculated by substituting $\bar{x}$ to model of environment $f = k_e(x - x_e)$: $\bar{f} = \frac{k_p k_e (x_d - x_e)}{k_p + k_e}$

For high environmental stiffness ($k_e \gg k_p$): $\bar{f} = k_p(x_d - x_e)$

Stiffness control for 1 DOF

- **Conclusions:**
  - Model of environment is given by $f = k_e(x - x_e)$.
  - To eliminate position error the robot exerts a steady state force on the environment $\bar{f} = k_p(x_d - x_e)$. Position gain k_p can be seen as desired 'stiffness' of the robot (we can look at the robot as spring with spring constant k_p). Term stiffness control is therefore often associated with PD controller.



- **Issues:**
  - x_e is not known.
  - We can measure x_e by force sensor. But then we can also use explicit force control.

Robot dynamics with contact forces

Robot dynamics in joint space with interaction forces:

$$\tau(t) = M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_e$$

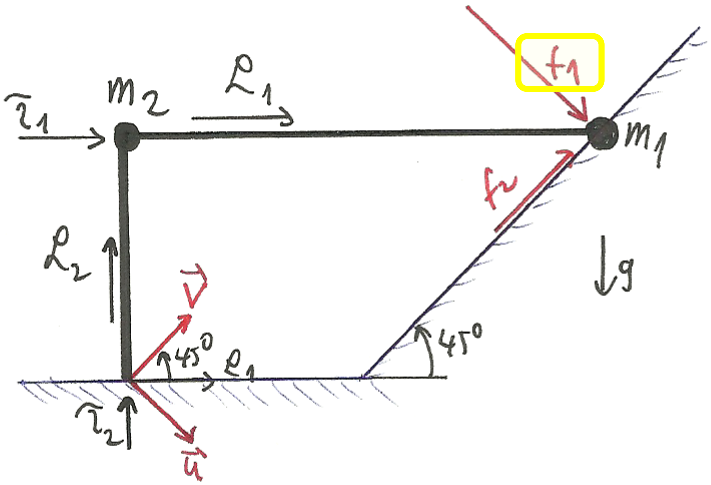
$$\tau_e = J^T(q)f$$

- τ_e , $n \times 1$ force exerted on the enviroment in joint space coord.
- f , $n \times 1$ vector of contact forces and torques in task space.
- Jacobian matrix: $\dot{x} = J(q)\dot{q}$, $J(q) = \begin{bmatrix} I & 0 \\ 0 & T \end{bmatrix} \frac{\partial h}{\partial q}$; T is transf. matrix for converting joint velocities to derivative of roll, pitch, yaw.

- **Note:** Jacobian matrix is defined in terms of a task space coordinate system, which is used in the addressed robot application. Relationship $\tau_e = J^T(q)f$ can be proven by conservation of energy concept.

Robot dynamics with contact forces

Example: Derive dynamics of simple Cartesian robot in contact with slanted surface. The robot should move along surface in direction given by v and at the same time apply a normal force f_1 to a surface in direction u.



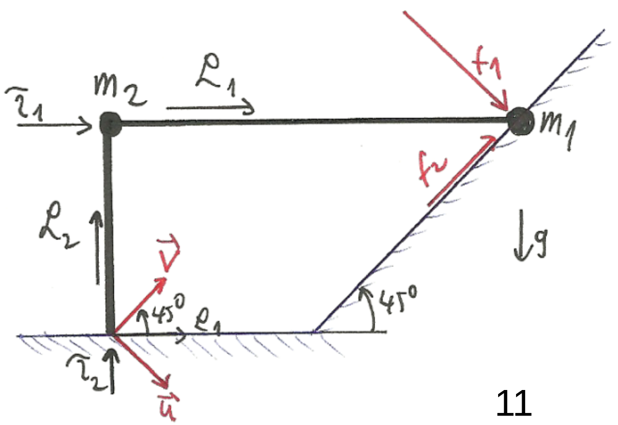
- Robot dynamics: $\tau = \begin{bmatrix} m_1 & 0 \\ 0 & m_1 + m_2 \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ (m_1 + m_2)g \end{bmatrix} + \begin{bmatrix} F_1(\dot{q}_1) \\ F_2(\dot{q}_2) \end{bmatrix}$

Robot dynamics with contact forces

- Task coordinate system is set according to task (the robot should move along surface in direction given by v and at the same time apply a normal force f_1 to a surface in direction u).
- Accordingly the task space vector is: $x = [u, v]^T$.
- Geometry transformation is $x = h(q)$.
- Derivative of x is: $\dot{x} = J(q)\dot{q}$ and $J(q)$ is Jacobian matrix :

$$J(q) = \begin{bmatrix} I & 0 \\ 0 & T \end{bmatrix} \frac{\partial h}{\partial q}$$

- J is assumed to be nonsingular.



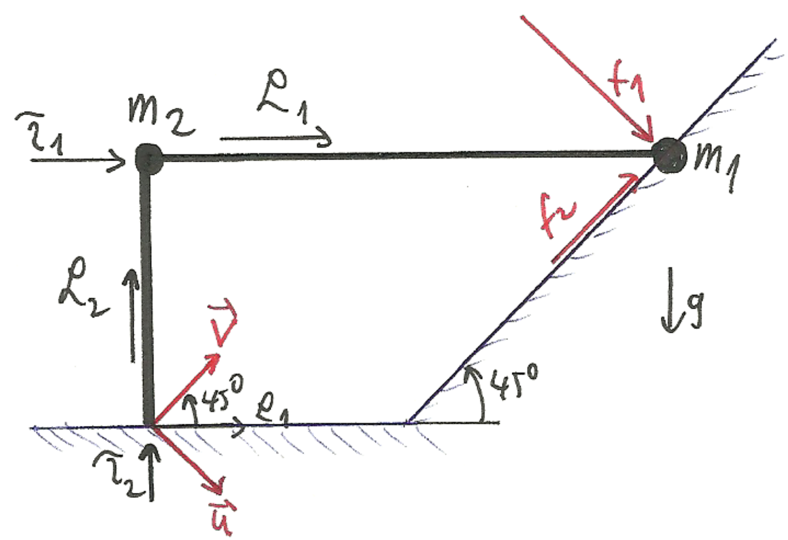
Robot dynamics with contact forces

- Robot dynamics general:

$$\tau(t) = M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + \tau_e$$

- **2DOF case**: $\tau(t) = M(q)\ddot{q} + G(q) + F(\dot{q}) + \tau_e$;

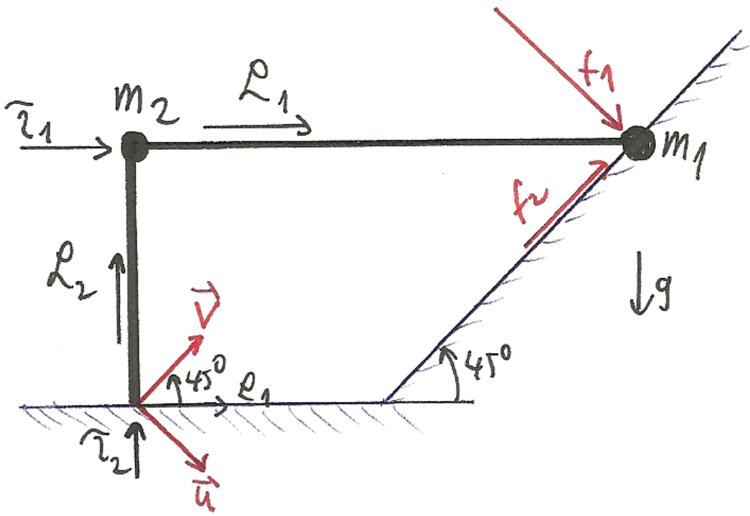
$$f = \begin{bmatrix} f_1 \\ f_2 \end{bmatrix}, \ \tau_e = J^T(q)f$$



Robot dynamics with contact forces

- From geometry it can be calculated:

$$x = \begin{bmatrix} u \\ v \end{bmatrix} = h(q) = \frac{1}{\sqrt{2}} \begin{bmatrix} q_1 - q_2 \\ q_1 + q_2 \end{bmatrix}$$



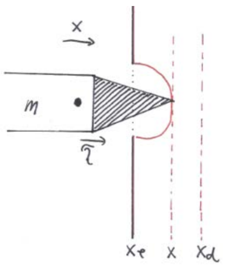
- Jacobian matrix is calculated as:

$$J = \frac{\partial h(q)}{\partial q} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & -1 \\ 1 & 1 \end{bmatrix}$$

- Robot dynamics

$$\tau(t) = M(q)\ddot{q} + G(q) + F(\dot{q}) + J^T(q)f$$

Stiffness control for n-DOF



- **Force exerted on environment**: $f = K_e(x - x_e)$
  - K_e *nxn* positive semi-definite constant matrix of environment stiffness,
  - x_e *nx1* vector in task space which denotes static location of environment.
- Multi DOF **stiffness controller** of PD type:

$$\tau(t) = J^T(q)\left(-K_v\dot{x} + K_p(x_d - x)\right) + G(q) + F(\dot{q}) =$$

$$= J^T(q)\left(-K_v\dot{x} + K_p\tilde{x}\right) + G(q) + F(\dot{q})$$

  - $K_{p,v}$ *nxn* positive-definite constant diagonal matrix
  - $\tilde{x} = x_d - x$ tracking error.
- **Closed loop dynamics** is:

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} = J^T(q)\left(-K_v\dot{x} + K_p\tilde{x} - K_e(x - x_e)\right)$$

Stiffness control for n-DOF

- Stability can be analyzed by Lyapunov analysis.
- It can be also proven that **steady state position** of end effector is:

$$\lim_{t\to\infty} x_i = \frac{K_{pi}x_{di} + K_{ei}x_{ei}}{K_{pi} + K_{ei}}$$

Our case study:

$$\bar{x} = \frac{k_p x_d + k_e x_e}{k_p + k_e}$$

- **Steady state force** exerted at environment is:

$$\lim_{t\to\infty} f_i = \frac{K_{ei}K_{Pi}(x_{di} - x_{ei})}{K_{pi} + K_{ei}}$$

Our case study:

$$\bar{f} = \frac{k_p k_e (x_d - x_e)}{k_p + k_e}$$

- For **high environmental stiffness** $(K_e \gg K_p)$

$$\lim_{t\to\infty} f_i = K_{Pi}(x_{di} - x_{ei})$$

Our case study:

$$\bar{f} = k_p(x_d - x_e)$$

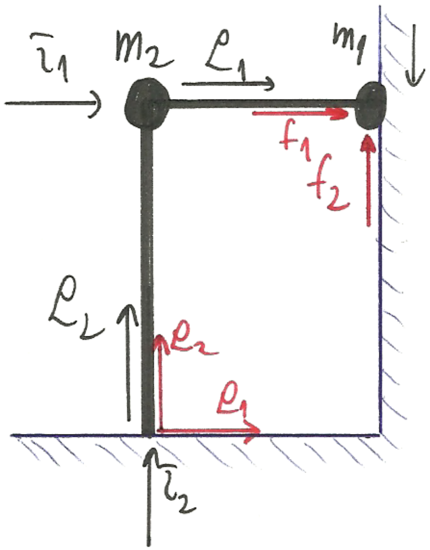
OUTLINE

- Introduction
- Stiffness control
  - Stiffness control for 1-DOF
  - Robot dynamics with contact forces
  - Stiffness control for n-DOF
- Hybrid position/force control
  - Example: Hybrid control of Cartesian 2DOF robot
- Final notes
- Exercise: Stiffness control for 2-DOF

Hybrid position/force control

- Controllers derived until now are suitable only for set point control.
- Such controller won’t perform adequately for example at polishing or grinding when a prescribed trajectory must be followed and at the same time desired force exerted →**hybrid position/force controller** is needed.
- In such controller position and force control problems are decoupled into subtasks via a task space formulation.
- Classic hybrid force control will be implemented. Here force is controlled along some axis of robot and position along others.

Hybrid position/force control for 2 DOF



1. Geometry

$$x = \begin{bmatrix} u \\ v \end{bmatrix} = h(q) = \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \quad , \quad \dot{x} = \begin{bmatrix} \dot{u} \\ \dot{v} \end{bmatrix} = \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \end{bmatrix}$$

$$J = \frac{\partial h(q)}{\partial q} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Task & joint space are here equivalent. Therefore we can refer to joint variables as task space variables for this problem.

2. Robot dynamics

$$\tau = Mq + G + f$$ or by joints

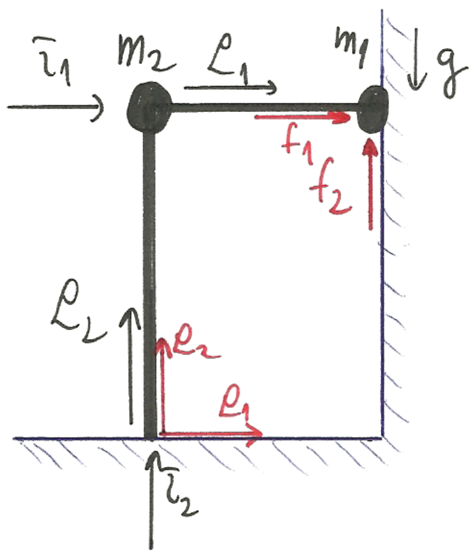
$$\tau_1 = m_1 \ddot{q}_1 + f_1$$

$$\tau_2 = (m_1 + m_2)\ddot{q}_2 + (m_1 + m_2) g + f_2$$

Position controlled should be q_2.
Force controlled should be q_1.

Hybrid position/force control of 2 DOF

Design of position controller for q_2



Tracking error $\tilde{x} = \ell_{2d} - \ell_2$

For position controller we use PD CT controller

$$\tau_2 = (m_1 + m_2)\ddot{\ell}_c + (m_1 + m_2)g + f_2$$

$$\ddot{\ell}_c = \ddot{\ell}_{dc} + k_{Tv}\dot{\tilde{x}} + k_{Tp}\tilde{x}$$

Position error dynamics for this is:

$$\ddot{\tilde{x}} + k_{Tv}\dot{\tilde{x}} + k_{Tp}\tilde{x} = 0$$

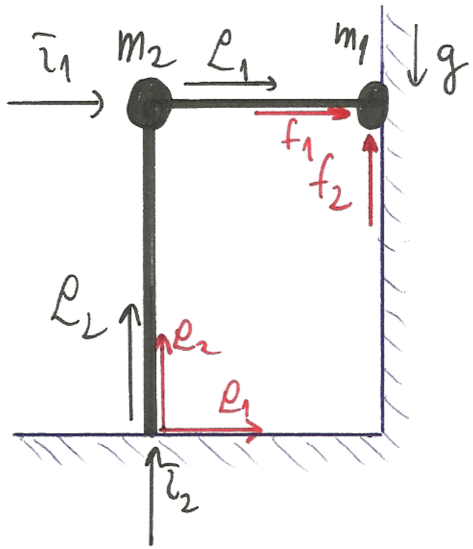
so for $k_{Tv}, k_{Tp} > 0$ standard linear control (here implemented as PD outer loop) gives:

$$\lim_{t \to \infty} \tilde{x} = 0$$

Asymptotic positional tracking is guaranteed.

Note: f_2 needs to be measured!

Design of force controller for q_1



It will be assumed that in this direction the environment can be modeled as spring. Normal force f_1 exerted on environment is given by:

$$\boxed{f_1 = k_e(\ell_1 - \ell_e)}$$

k_e is environment stiffness

Second derivative:

$$\ddot{\ell}_1 = \frac{1}{k_e}\ddot{f}_1 \rightarrow$$

Dynamic eq.

$$\tau_1 = m_1\ddot{\ell}_1 + f_1$$

following is obtained:

$$\tau_1 = \frac{m_1}{k_e}\ddot{f}_1 + f_1$$

This eq. can be now used to design force controller.

Force tracking error:

$$\tilde{f} = f_{d1} - f_1$$

And we use for force controller also CT contr.

$$\boxed{\begin{aligned} \tau_1 &= \frac{m_1}{k_e}a_n + f_1 \\ a_n &= \ddot{f}_{d_1} + k_{Nv}\dot{\tilde{f}} + k_{Np}\tilde{f} \end{aligned}}$$

k_{Nv}, k_{Np} positive control gains

Force tracking error system is now:

$$\ddot{\tilde{f}} + k_{Nv}\dot{\tilde{f}} + k_{Np}\tilde{f} = 0 \Rightarrow \lim_{t\to\infty}\tilde{f} = 0$$

ASYPTOTIC FORCE TRACKING

Final notes

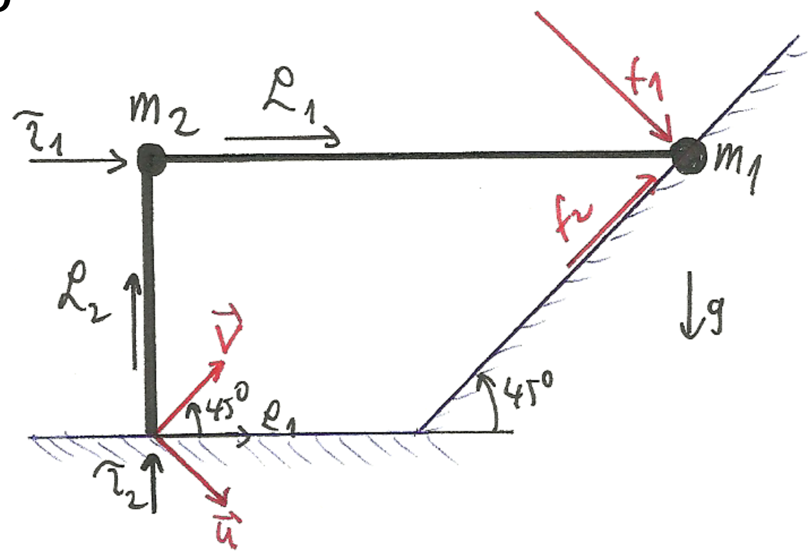
- Hybrid position/force control concept presented on 2 DOF robot can be extended on n-DOF robot by using task space formulation (in n-DOF case task space is not the same as joint space).
- Other option is to design a CT feedback linearizing control that globally linearizes and decouples dynamic equations to $\ddot{x} = [\ddot{u}, \ddot{v}]^T$. Then independent motion or force controller can be designed for each joint separately.
- Issues: force sensor, transformation from measured end-effector forces to task space.
- Advanced: impedance control.

REPORT REQUIRED

Exercise: Stiffness control for 2-DOF

Design and simulate stiffness controller for 2 DOF robot.

- Control goal is to move end effector to final position $v_d = 3m$ and exert final desired normal force $f_{d1} = 2N$.
- Surface friction (f_2) and joint friction can be neglected.
- It is assumed that normal force satisfies: $f_1 = k_e(u - u_e)$;
- $u_e = 3/\sqrt{2}\,m, k_e = 1000\,N/m$.
- Initial position of end-effector is $v_0 = 5m, u_0 = 3/\sqrt{2}\,m$.
- $m_1 = m_2 = 1\,kg,\, k_{pi} = k_{vi} = 10$



Hint:
Use $\lim_{t\to\infty} f_i = f_{di} = K_{Pi}(x_{di} - x_{ei})$ to calculate u_d (u desired)

REPORT REQUIRED

Exercise: Stiffness control for 2-DOF

Example of results